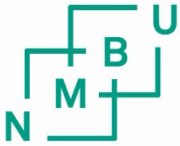


Norwegian University
of Life Sciences

Master's Thesis 2021 30 ECTS
Faculty of Chemistry, Biotechnology and Food Science

Classification of Consumer Goods into 5-digit COICOP 2018 Codes

Daniel Milliam Müller
Industrial Economics

Preface

This thesis was written during the autumn of 2021 and concludes my master's degree in Industrial Economics at the Norwegian University of Life Sciences (NMBU). I am grateful for the opportunity I have had to write this thesis for the Faculty of Chemistry, Biotechnology and Food Science and for the opportunity to collaborate with Statistics Norway.

I would like to thank my excellent supervisors, Kathrine Frey Frøslie and Boriska Toth, for their continued guidance and support throughout this process. It has truly been a wonderful experience working with such enthusiastic, encouraging and inspirational supervisors.

I am also grateful to the *Forbruk* team at Statistics Norway for making this project possible, and for all their help, input and encouragement along the way.

I would also like to take this opportunity to thoroughly thank my family and loved ones that have been of tremendous support and help to me during the writing of this thesis and during my time at NMBU. I suppose it is finally time for me to start talking about other things than the important distinction between food sold at grocery stores or served at restaurants. Thank you for hanging in there.

Daniel Milliam Müller
Ås, 15th December 2021

Abstract

The survey of consumer expenditure is a national survey conducted by Statistics Norway (SSB) with the purpose of collecting detailed data about Norwegian households' annual consumption of different goods and services. The survey has up until its most recent publication in 2012 relied on employees at SSB to manually categorise all registered expenditures into COICOP (Classification of Individual Consumption by Purpose) item codes to produce consumption statistics. This has involved large workloads and high implementation costs, and because of this, SSB wants to modernise and improve the efficiency of the survey for its next planned implementation in 2022.

This study is the result of a 3-month collaboration with SSB to explore the application of supervised machine learning for classification of consumer goods to 5-digit COICOP codes. The purpose of this study has been to explore the potential of using machine learning to automate parts of the survey of consumer expenditure.

This thesis demonstrates how different data sets from separate sources can be combined into a COICOP training data set that can be used to develop and evaluate COICOP classification models. Furthermore, this study explores how these models can be incorporated into a "human-in-the-loop" based classification system to facilitate automatic classification of consumer goods while also maintaining sufficient levels of data quality.

The findings indicate that supervised machine learning is a suited method for classifying consumer goods into 5-digit COICOP codes. Additionally, the results show that the models' prediction probabilities are good indicators of where misclassifications occur. Together, these findings show a promising potential for implementation of a "human-in-the-loop"-based classification system for reliable classification of consumer goods. At the same time, the findings uncover important limitations with the data used in this thesis, as the models were trained on data that the survey of consumer expenditure will not be based on. This thesis has used data sets that were *available*, and these were not necessarily the most *relevant*. Therefore, it is not expected that the developed models will provide immediate value to the objectives of SSB without first being trained on more relevant data.

Sammendrag

Forbruksundersøkelsen er en nasjonal undersøkelse som er utført av Statistisk Sentralbyrå (SSB) med den hensikt å samle inn detaljert forbruksstatistikk om norske husholdninger. Inntil dens foreløpig siste gjennomføring i 2012, har ansatte ved SSB måttet manuelt kode alle registrerte varekjøp inn i COICOP (Classification of Individual Consumption by Purpose) varekoder for å produsere forbruksstatistikk fra undersøkelsen. Dette har medført store arbeidsmengder og høye kostnader, og SSB ønsker derfor nå å modernisere og effektivisere undersøkelsen i forbindelse med dens neste planlagte gjennomføring i 2022.

Denne oppgaven er et resultat av et 3 måneders samarbeid med SSB for å utforske anvendelse av veiledet maskinlæring for å klassifisere forbruksvarer i 5-sifrede COICOP varegrupper. Dette har hatt som hensikt å kartlegge effektiviseringspotensialet ved å bruke maskinlæring til å automatisere deler av forbruksundersøkelsen.

I denne oppgaven demonstreres det hvordan ulike datasett fra ulike kilder kan kombineres til et COICOP treningsdatasett som kan brukes til å utvikle og evalurere COICOP klassifiseringsmodeller. Videre utforsker oppgaven hvordan disse modellene kan brukes i kombinasjon med et "human-in-the-loop"-basert klassifieringssystem for å tilrettelegge for automatisk klassifiering av varer og samtidig ivareta tilstrekkelig datakvalitet.

Funnene antyder at veiledet maskinlæring er en egnet metode for klassifisering av varer til 5-sifrede COICOP varekoder, og i tillegg viser resultatene at modellenes prediksjonssannsynligheter gir en god indikasjon for hvor feil oppstår. Dette gir et godt grunnlag for bruk av et "human-in-the-loop"-basert klassifiseringssystem for pålitelig klassifisering av forbruksvarer. Samtidig avdekker funnene sentrale begrensninger med dataen brukt i denne oppgaven, da modellene ble trent på data som forbruksundersøkelsen ikke vil basere seg på. Bakgrunnen for dette er at oppgaven har brukt de data som var *tilgjengelige*, og disse var ikke nødvendigvis de mest *relevante*. Det kan dermed ikke forventes at de utviklede modellene gir umiddelbar verdi til SSBs formål uten først å bli trent på mer relevante data.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Household Budget Surveys

The household budget surveys (HBS) are surveys that focus on household consumption expenditure. These surveys are conducted in most countries of the world, and they are key components in collecting data to produce household consumption and expenditure statistics (Benedikt et al., 2020). These statistics are of interest to many research institutions, and they contribute to different fields of research. They are typically used in estimations of Gross Domestic Product and Consumer Price Indices, and they are also relevant in research related to food consumption and nutrition (Egge-Hoveid & Brændvang, 2020).

1.1.1 COICOP Classification System

A key part of the household budget survey is to categorise expenditures into corresponding consumption categories. The Classification of Individual Consumption According to Purpose (COICOP) system was developed by the United Nations Statistics department in 1999 with the motivation of providing a standardised framework used to categorise and analyse individual expenditures according to their purpose. This framework would facilitate comparable expenditure statistics across institutions, and it is considered a standard in the production of most expenditure and consumption statistics today (UN, 2018).

Due to the need for a more detailed classification system, a revision to the COICOP system was initiated in 2015. This resulted in the publication of the "COICOP 2018" system in 2018, a classification system that aimed to better fulfil the needs of its users.

The COICOP 2018 system consists of four different levels, where each level represents a different degree of classification detail. These levels are hierarchically ordered with an increasing number of consumption categories and level of detail in the classification. The system uses numeric code to represent the different consumption categories, and depending on the level, this code varies from a 2- to 5-digit scheme. A general overview of the structure in the COICOP 2018 system is illustrated in figure 1.1.

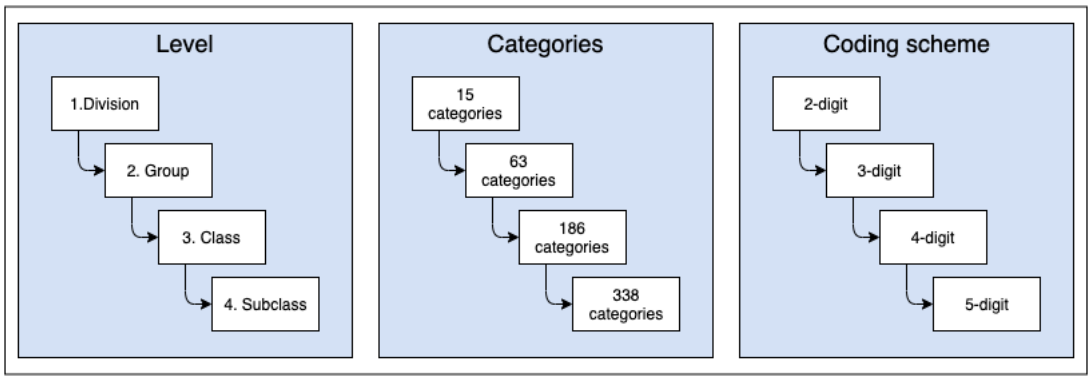


Figure 1.1: Hierarchical structure of the COICOP 2018 classification system

To illustrate how the COICOP 2018 classification system works, consider the example of "Chocolate milk". Figure 1.2 shows how the different categories and COICOP codes are used to categorise "Chocolate milk" depending on the desired level of detail with the COICOP 2018 classification system.

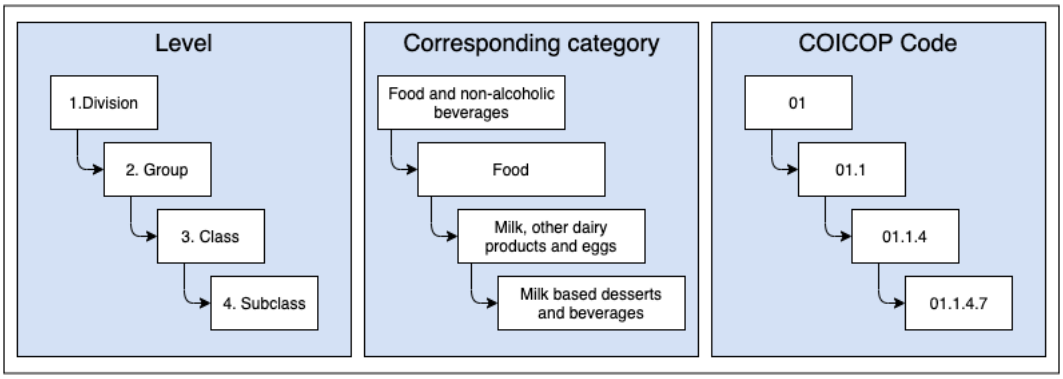


Figure 1.2: COICOP 2018 classification of "chocolate milk"

The levels of the COICOP system provide tools for classification in multiple detail levels. In areas such as health care or fields with similar requirements for privacy and confidentiality, basic levels of categorisation detail might be a suitable way to present statistics, while for research related to nutrition, food consumption or consumption expenditures, more granular levels of detail are often desired. The subclass level was added in the COICOP 2018 revision to accommodate this, and this has consequently become the standard in most modern household budget surveys (Benedikt et al., 2020).

1.2 The Survey of Consumer Expenditure

Statistics Norway (SSB) are the national statistical institute of Norway and the main producer of official statistics in the country. They are responsible for collecting and producing statistics in fields such as economy, population, and society. Statistics Norway are also responsible for coordinating statistics prepared by the Norwegian government and they have the overall responsibility for Norway's participation in international statistics cooperation

SSB are responsible for household statistics in Norway, and they coordinate the collection and production of Norwegian household consumption and expenditure statistics. This is done through their survey of consumer expenditure, a national survey that collects data that specifically relate to Norwegian household spending patterns, where the overall aim of the survey is to provide a detailed picture of Norwegian households' annual consumption of different goods and services (Holmøy & Lillegård, 2014).

SSB have published consumer expenditure statistics since 1958, and the results from the previous survey of consumer expenditure were published in 2012. The expenditure statistics is one of SSB's most in-demand statistics and it acts as the basis for many studies conducted by research institutions such as the Norwegian Labour and Welfare Administration, the Ministry of Finance, and the Ministry of Health. Many research institutions now hold a great interest in obtaining new and updated consumer expenditure statistics from SSB, as the current statistics, which is approaching 10 years of age, is becoming more outdated and consequently less representative of Norwegian household spending patterns (Egge-Hoveid & Brændvang, 2020).

1.2.1 Survey Design 2012

The previous survey of consumer expenditure was conducted by recruiting different Norwegian households and tasking them to keep a diary of their expenditures for two weeks. The survey spanned a full year, which means that there were 26 different 2-week periods in total. The different periods would sum up to a full year, where the different participating households would each be assigned a specific period in which to register their expenditures.

	Dato	Hva slags vare ble kjøpt? Beskriv varen.	Kryss av hvis varen er kjøpt		Mengde	Hva kostet varen?	
			i utlandet	på internett		Kroner	Øre
Eks.1	2/1	Lammestek, fersk			2,3 kg	269	90
Eks.2	2/1	Grovbrød			800 g	26	50
Eks.3	5/1	Jordbær selvplukket			4 kg		
Eks.4	6/1	Laks, røkt			250 g	39	90
Eks.5	9/1	Skummet melk			1 liter	10	90
Eks.6	13/1	Rødvin, kartong (kjøpt i Sverige, svenske kroner)	X		3 liter	149	00

Figure 1.3: Expenditure registration by respondent in diary (Holmøy & Lillegård, 2014, p. 81)

Household Sampling and Data Collection

The households that participated in this survey, were chosen by stratifying Norwegian households on "geographic region" and "type of household". The participants were then randomly selected within each stratum. There was a total of 7100 households selected to participate. Figure 1.4 shows the number of households selected in each stratum.

		Husholdningstype					
	I alt	1	2	3	4	5	6
I alt	7100	1 100	1 200	1 200	1 200	1 200	1 200
Akershus og Oslo	1686	261	285	285	285	285	285
Hedmark og Oppland	562	87	95	95	95	95	95
Sør-Østlandet	1384	214	234	234	234	234	234
Agder og Rogaland	1006	156	170	170	170	170	170
Vestlandet	1184	184	200	200	200	200	200
Trøndelag	615	95	104	104	104	104	104
Nord-Norge	663	103	112	112	112	112	112

Figure 1.4: Selection of households based on region and type of household (Holmøy & Lillegård, 2014, p. 35)

The survey was implemented by splitting it into three parts. First, SSB would conduct an introductory interview with the participating households to gather additional information, such as education levels and occupational status, about the household. Next, each household would record their expenses for their assigned 2-week period by writing down their expenses and saving the receipts from their purchases in a physical diary provided by SSB. Lastly, in the concluding interview, SSB would enquire each household about expenditures that incur irregularly and would likely not be covered in the registered expenses in their assigned period. This applies to expenses related to travel, household appliances, expensive clothing, etc.

After completing the concluding interviews and receiving the household diaries with the associated receipts, SSB would manually classify and register each individual expenditure into their database.

Results and Assessment of Data Quality

The resulting survey ended up with an overall response rate of 48.9%, where the goal had been a response rate of at least 50%. The response rate was measured by looking at the number of households that had completed a full survey, meaning two interviews and a full 2-week period of expenditure registrations. SSB experienced a noticeable increase in drop-off among the respondents as they approached the end of the year. SSB ascribe some of the reason for this to the challenges involved with rescheduling interviews and diary-keeping periods. Rescheduling became increasingly difficult towards the end of the year as there would be fewer available periods to reschedule activities to.

The report from 2012 also states that the implementation of the survey placed a heavy burden on its respondents as it demands a lot of time and effort dedicated to

longer interviews and manual registrations of every single expenditure for a continued period. SSB argue that this is likely a large contributor to the observed drop-offs among its participants (Holmøy and Lillegård, 2014, p. 9).

1.2.2 Survey Design 2022

SSB now plan to conduct a new survey of consumer expenditure to publish updated expenditure statistics in 2022. The report from the survey in 2012 outlines how the survey has suffered from high implementation costs, low response rates and high levels of uncertainties in its results and estimations. This, in addition to the desire to publish expenditure statistics more frequently in the future, has motivated SSB to make significant changes to how the next survey will be conducted and how expenditure statistics will be generated in the future. This has initiated "Forbruk 2022", a project to modernise the processes, routines and methods involved in SSB's survey of consumer expenditure.

The purpose of "Forbruk 2022" is described by Egge-Hoveid and Brændvang as:

> *By modernising the survey of consumer expenditure, we aim to conduct the survey and produce new statistics for Norwegian households in an efficient way with acceptable response rates, and with higher quality and reliability.*

In the new survey of consumer expenditure, SSB plan to expand and improve on data acquisition by combining the survey with financial transaction data provided by the largest grocery stores in Norway. The survey will now consist of two main components: *transaction data* and *survey data*.

Financial Transaction data

SSB plan to utilise financial transaction data as an additional data source to assist in its expenditure statistics production. The data will be provided by the largest grocery stores in Norway: Rema, Coop and NorgesGruppen. This is meant to facilitate the application of big data in the statistics production, which in turn is meant to increase the quality of SSB's expenditure statistics, and how frequent new statistics can be published, while also lowering costs related to data acquisition and statistics production.

Survey data

The survey itself will undergo significant changes to create a better and less demanding user experience for the respondents by allowing the participating households to either automatically register their expenditures by scanning their receipts, or to manually register expenses using SSB's phone app. This is meant to replace the need for a physical diary, which both aims to ease the burden on the participants, as well as the work involved with coding and registering the expenditures into SSB's database by utilising automatic classification of the different expenditures. Figure 1.5 depicts a demo version of SSB's phone app, where the respondent can scan a receipt, manually register a purchase, or manually register a bill.

Figure 1.5: Survey expense registration in phone app (SSB, 2021)

1.2.3 Survey Data from Pilot Study

SSB have conducted a pilot study for the new survey solution. In this study, SSB recruited 600 households to participate. Similar to the survey of consumer expenditure in 2012, the different households were selected based on household types, and each household was tasked with registering their expenses over a 2-week period. The pilot study lasted for 6 weeks in total, starting from 31.05.21 and lasting to 11.07.21. These weeks were split into 5 different 2-week periods in which the different households would scan their receipts or manually register their expenses in SSB's phone app.

Whereas the fundamental structure of the survey used in the pilot study still reuses many components of the previous survey of consumer expenditure, the process of registering and collecting data from expenses was distinctly different from 2012. When a participant scans a receipt, the image of this receipt is processed, and the text contained in different fields of the receipt is extracted. Examples of such fields are shown in figure 1.6. The extracted data is used to construct a data set of different households and their expenses, and as a result, SSB were able to collect a data set containing 14 389 entries of consumer goods and 2 785 unique receipts from the participants of the pilot study.

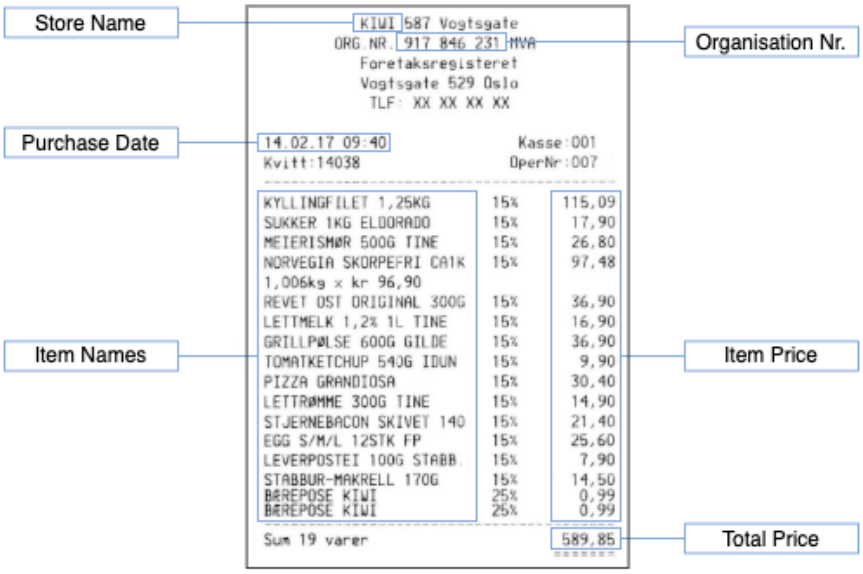


Figure 1.6: Example of text fields extracted from receipts (SSB, 2019)

1.2.4 Streamlined Classification of Items from Receipts

Going forward, SSB aim to explore the potential of applying machine learning to assist in classification of consumer goods into corresponding 5-digit COICOP 2018 subclass codes. Figure 1.7 illustrates a pipeline that exemplifies how scanned receipts can be received as input to automatically output each item in the receipts with its 5-digit COICOP 2018 subclass code.

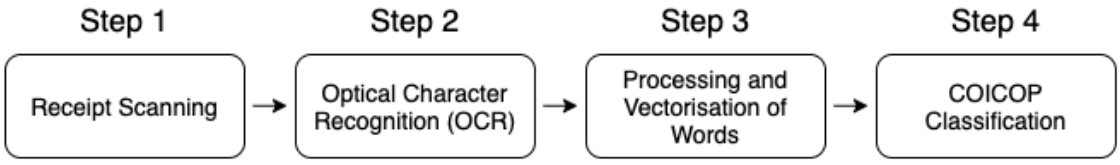


Figure 1.7: Pipeline from scanned receipts to statistics production

1. **Scanning:** Paper receipts are scanned into images. Respondents scan images of receipts using the mobile phone app.
2. **Optical Character Recognition (OCR):** Text is automatically extracted from the images of the receipts. Additionally, meta-data such as total receipt price, date of purchase and store name is retrieved.
3. **Processing and Vectorisation of Words:** Processing of the text output from the OCR step. The output from OCR may contain misspelled words or errors due to characters being wrongly recognised. NLP techniques are used to correct errors and to create a streamlined process to prepare the text data for classification.
4. **COICOP Classification:** Using supervised machine learning with prepared text data from the previous step, the different items are classified into 5-digit COICOP 2018 subclass codes.

As described in subsection 1.2.3, SSB were able to test some of the steps of the pipeline in their pilot study. The first two steps were successful, resulting in the previously mentioned receipts data set. Work on the remaining steps (3 and 4), which involve preparations for and implementation of machine learning to classify each item, has up until this point been limited. This thesis aims to continue this work and to explore the feasibility and potential of implementing the last two steps of the pipeline for SSB to facilitate automatic classification of items from scanned receipts.

1.3 Project Goal and Thesis Description

The goal of this thesis is to explore and implement supervised machine learning for COICOP classification of consumer goods and to evaluate the potential and feasibility of incorporating this into automatic classification of items for the survey of consumer expenditure.

Goal: Implement automatic classification of consumer goods to classify items into 5-digit COICOP 2018 subclass codes based on their item names.

1.3.1 Aims for Thesis

With the stated project goal, this thesis aims to assist Statistics Norway in their work with the survey of consumer expenditure 2022 by proposing methods and designs for incorporating supervised machine learning into automatic classification of consumer goods. Multiple objectives of the project were developed in collaboration with Statistics Norway, and these have been translated into four research questions (RQs) which this thesis aims to address.

RQ1: How can data from auxiliary data sources be combined to assemble a COICOP training data set for training and developing a COICOP classifier model?

The first step involved in developing a COICOP classifier, is to assemble a training set. Statistics Norway possess a wide range of data sets that have been collected through different means, and the first objective of this thesis is to investigate whether some of these data sets can be combined into a data set suited for training machine learning classifiers for COICOP classification.

RQ2: How well do traditional classification models perform on the COICOP training data?

The next objective of this thesis is to evaluate how well traditional supervised machine learning classifiers can learn patterns in the COICOP training data and predict the 5-digit COICOP subclass code based on the item names of consumer goods. The thesis aims to explore whether the classifiers are in fact able to learn some discriminatory information from the item names which typically contain short and concise item descriptions.

RQ3: How well do the performances of the trained COICOP classifiers carry over to unseen samples of scanned receipt data?

The ambition for SSB is to be able to automate parts of the work involved with classifying consumer goods into COICOP categories for the survey of consumer expenditure. Data from scanned receipts is planned to be an important data source for the survey in 2022. This thesis aims to explore the potential of implementing supervised machine learning into automatic classification by assessing how well the performances of the previously trained classifiers (from RQ2) carry over to items from scanned receipts data.

RQ4: What are some of the current limiting factors that prevent Statistics Norway from implementing automated classification of items from scanned receipts?

Based on the results of the preceding research questions, the final objective of this thesis is to outline some prominent limitations that potentially prevent Statistics Norway from currently implementing automated classification of items from scanned receipts.

1.3.2 Research Methods

Several approaches and methodologies have been used in order to answer the research questions and to meet the objectives of this thesis described in the preceding subsection.

To answer the first research question, multiple data sets at SSB have been prepared through high-level filtering operations and conversion of item categorisation coding formats to bring all data sets to the same format. These data sets have then been combined into a single data set consisting of valid entries of *item names* and corresponding *5-digit COICOP 2018 subclass codes*. Each data set used in this thesis and the performed steps with preparing and combining these different data sets are described in chapter 3.

To answer the second research question, a set of supervised machine learning classifiers have been trained on the assembled COICOP training data by employing different count-based feature extraction methods to transform the item names into numeric feature vectors. A portion of the training data was withheld from the classifier models during training, and the models' predictive performances were then evaluated on the withheld data to assess how well they generalise to unseen data. The choice of classifiers, model structure and evaluation protocol are described in chapter 4, while classification results are presented in section 5.2 of chapter 5.

To answer the third research question, the previously explored classifier models are retrained on the full assembled COICOP training data. No partition of the training data is held-out from the model. Instead, the classifier model's predictive performances are evaluated on a different test set of scanned receipt items. This test set is the product of randomly sampling items from the data set produced in the pilot study (see subsection 1.2.3) and manually labelling these items with their 5-digit COICOP subclass code. Subsection 3.2.5 describes the steps involved with acquiring this test set, while the results of the classifier models' performances on this test set are presented in section 5.3 of chapter 5.

To answer the fourth and final research question, the results from the classification performance on the scanned receipt items are explored in detail to investigate where misclassifications typically occur and to attempt to identify the likely reason as to why they occur. Additionally, this thesis explores the reliability of the models' prediction probability scores for each prediction to assess the current potential for implementing these models into an automatic classification system for scanned receipt items for the survey of consumer expenditure. This assessment is done in section 6.1 of chapter 6.

Chapter 2

Background

Chapter 2 covers background theory and information relevant to this thesis. This includes terminology and techniques within the fields of *Machine Learning* (ML) and *Natural Language Processing* (NLP). Section 2.1 introduces some background and key concepts in machine learning. Section 2.2 delves into the theory behind the classifiers relevant to the thesis and introduces several performance metrics. The final section, section 2.3, focuses on theory and methods within the field of Natural Language Processing. All data examples in this chapter are entirely fictional, and these are intended only to demonstrate key parts of relevant theory.

2.1 Basic Concepts in Machine Learning

Machine Learning is considered a sub-field of artificial intelligence (AI), and it specifically focuses on applying self-learning algorithms that learn from data in order to make predictions (Raschka and Mirjalili, 2019, p. 1). Predictions are often tied to classification, regression, or clustering problems. For a classification problem, the objective is to identify which of a set of categories an observation belongs to, such as medical diagnoses or email spam detection. Regression is typically used in problems where the prediction is a continuous value, such as sales forecasts or housing prices. Clustering is the task of dividing observations into groups such that the observations that are more similar to the observations contained in the same group than those in the other groups. Clustering is typically associated with unsupervised learning, where the groups are not defined beforehand.

2.1.1 Key Terms

This subsection provides a brief explanation of terms that are used in the subsequent parts of this chapter.

Samples: Observations, instances, or objects of the data that is collected.

Features: Explanatory variables. The features are usually numeric or categorical, and the machine learning model will typically be based on coefficients of these variables.

Target: Categories, classes, or values to be predicted. The target is discrete or continuous depending on the problem.

Figure 2.1 shows a dataset containing n samples and m features, where the target indicates whether the sample is a food item or not, represented as 1 or 0, respectively.

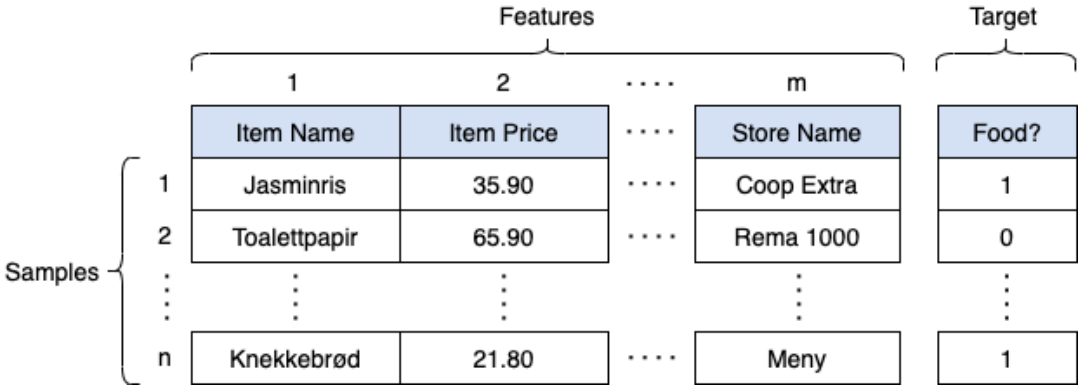


Figure 2.1: Samples, Features and Target

2.1.2 Learning Methods

There are two main branches within the machine learning field, namely *supervised learning*, and *unsupervised learning*. Supervised Learning typically covers machine learning tasks in the context of classification or regression, while unsupervised learning is common in tasks such as clustering.

Supervised Learning

In *supervised learning*, a model is trained using training data set that were the samples have already been labelled with their target value or class. This creates pairs of samples and corresponding target labels, and these pairs are passed to a machine learning algorithm to fit a predictive model that is intended to be able to predict new, unlabelled data observations.

Figure 2.2 shows how the samples and their corresponding target labels are used as training data for the supervised machine learning algorithm. While training, the model attempts to predict the target labels of the samples in the training dataset. The *true* target labels (correct labels) provide direct feedback to these predicted target labels, and the model automatically adjusts itself to be able to make predictions that are better aligned with the true target labels in the next iteration.

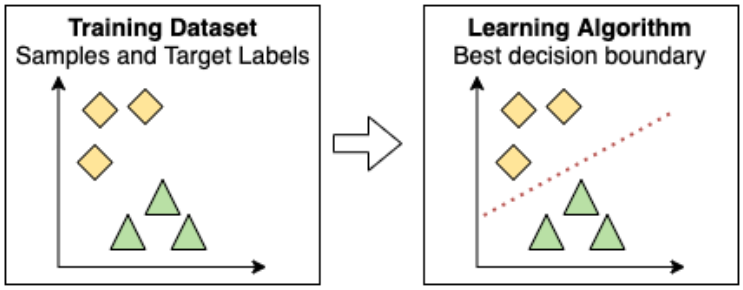


Figure 2.2: Fitting a model with supervised machine learning

Upon reaching a satisfactory performance or a maximum number of iterations, the training stops, and the final model is produced. This is commonly referred to as the "fitted" or "trained" model, and this model will make predictions on new data samples. Figure 2.3 shows how the trained model in figure 2.2 is applied to predict target labels for a collection of unseen samples.

Figure 2.3: Predictions on unseen samples

Unsupervised Learning

The main difference between supervised and unsupervised learning is that for unsupervised learning, the data observations are not labelled before training. This means that the model gets no direct feedback during training as it has no true target labels to adjust itself to. Instead, unsupervised learning methods will typically search for similarities, patterns or other meaningful information in the observations and group similar observations together without the guidance of true target labels (Raschka and Mirjalili, 2019, p. 7).

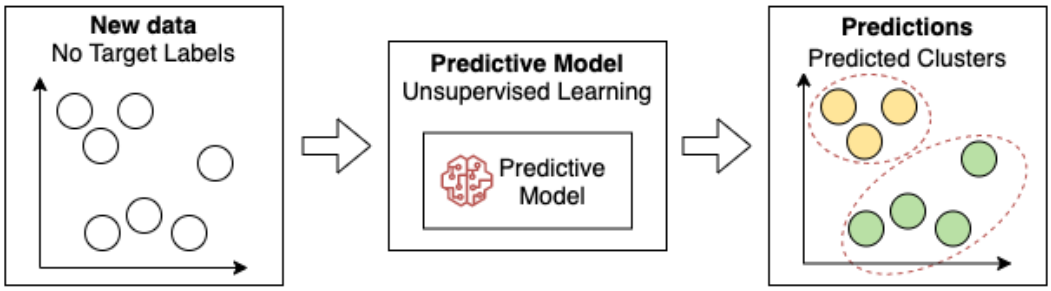


Figure 2.4: Clustering with unsupervised machine learning

Figure 2.4 illustrates how samples are grouped based on their attributes. In unsupervised learning tasks, the ideal number of different groups are not always known beforehand. Additionally, the discriminating features of the data observations are not necessarily obvious, potentially making it difficult to identify *how* the categorisation of the data has been done and what the different categories or groups actually represent.

2.1.3 Bias-Variance Trade-off

In the field of machine learning, the terms *Bias* and *Variance* are often used to describe the performance of a machine learning model. These terms express sources of error that can contribute to a machine learning model not being able to generalise well on data beyond the original training data.

Bias is a measure of the systematic error that is not due to randomness, i.e., it measures how far off a model's predictions are from the correct value in general if the model were to be rebuilt multiple times on different sets of training data. A high level of bias can lead to a model missing important relations between the features and target. A model suffering from high bias is typically referred to as *underfitted*.

Variance is a measure of the error that is due to small fluctuations in the data, meaning that it measures the consistency of the model's classification predictions for a particular sample if the model were retrained multiple times on different subsets of the training data. High variance can lead to a machine learning model adjusting to random noise in the training data, typically resulting in a model performing well on the training data, but it does not generalise well on data it has not seen before. A model suffering from high variance is typically referred to as *overfitted*.

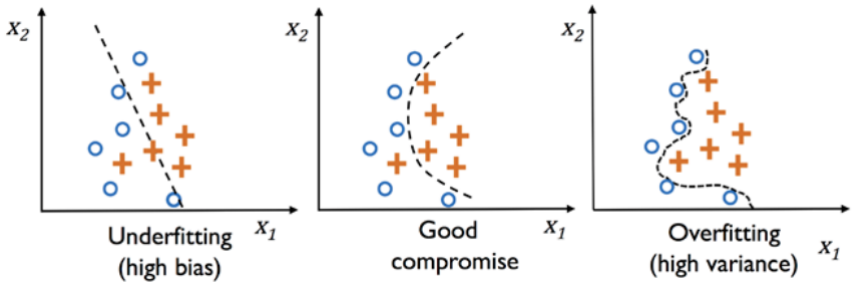


Figure 2.5: Bias-variance trade-off (Raschka and Mirjalili, 2019, p. 76)

Bias-Variance Trade-off is a common compromise in machine learning where one attempts to identify a model that minimises the total error. If the model is too simple, with few parameters, the model is prone to high bias and low variance. However, if the model has too many parameters the model is consequently going to suffer from high variance and low bias. In other words, the model has to be complex enough to avoid underfitting, while at the same time, it should not be too complex to prevent overfitting.

2.2 Classifiers for Text Classification

Text classification is a field within machine learning that aims to assign categories to text documents. Text classification is relevant in tasks such as sentiment analysis or spam detection, and it typically incorporates supervised machine learning methods where the categories (targets) are pre-defined (Minaee et al., 2021).

This section briefly covers the relevant theory behind machine learning classifiers and performance evaluation metrics that are relevant to this thesis.

2.2.1 Logistic Regression

Logistic regression is a popular supervised machine learning model for classification tasks that are based on extracting features and combining them linearly to predict the probability of a sample belonging to a particular class (Raschka and Mirjalili, 2019, p. 61). Given an input feature vector $x = (x_1, ..., x_m)$, the net input, z, is calculated by taking the linear combination of the input values, x, and a corresponding weight vector, $w = (w_1, ..., w_m)$, shown in equation 2.1.

$$z = w_1x_1 + w_2x_2 + ... + w_mx_m \tag{2.1}$$

To calculate the probability that a certain sample belongs to a particular class, logistic regression uses an activation function, $\phi(z)$ to transform the net input values, z. For a binary classification task, e.g., predicting whether a consumer goods item is a food or non-food item, the logistic regression uses the *sigmoid* activation function, shown in equation 2.2, to transform the values.

$$\phi(z) = \frac{1}{1 + e^{-z}} \tag{2.2}$$

Figure 2.6 illustrates that by using the sigmoid activation function, $\phi(z)$, to transform the net input, z, the net input values are transformed into values in the range $[0, 1]$, where larger values of z results in a value for $\phi(z)$ that are closer to 1, while smaller values of z in turn results in $\phi(z)$ being closer to 0.

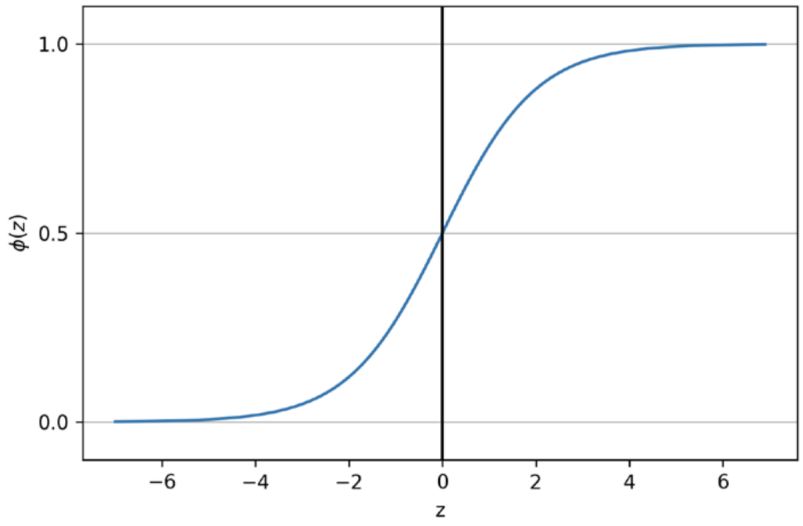


Figure 2.6: Sigmoidal curve (Raschka and Mirjalili, 2019, p. 63)

The output of the sigmoid function, $\phi(z)$, can be interpreted as the probability of a particular sample belonging to the positive class given its feature vector x and its weight coefficients w, which can be expressed as $\phi(z) = P(y = 1|x; w)$ (Raschka and Mirjalili, 2019, p. 64). The predicted class label, $\hat{y}$, of sample i, can therefore be summarised into a threshold function, shown in equation 2.3, where the threshold value is set to 0.5.

$$\hat{y}^{(i)} = \begin{cases} 1, & \text{if } \phi(z^{(i)}) \geq 0.5 \\ 0, & \text{otherwise} \end{cases} \tag{2.3}$$

The logistic regression algorithm is based on supervised learning, and it uses the true labels, y, as direct feedback to its predicted labels, $\hat{y}$, to find optimal values for its weights, w, by adjusting them repeatedly until there is no additional improvement in the algorithms ability to predict the class label or when a pre-defined number of iterations has been reached.

2.2.2 Decision Trees and Random Forests

Decision Trees are a supervised machine learning algorithm that can be used in both classification and regression tasks. Due to their simple structure, Decision Trees offer high levels of control and interpretability, making them a popular choice for many machine learning tasks (Raschka and Mirjalili, 2019, p. 90). The Decision Tree algorithm breaks down data by making decisions based on asking a series of questions. The questions can be seen as individual nodes, and the answers to these questions represent splits at the different nodes. By asking a series of questions, a tree structure is formed, which is finally used to predict the class or value of samples.

As an illustration, consider the example shown in figure 2.7. Here, a simple Decision Tree is employed to decide whether samples of consumer goods are food or non-food items. The samples are first split into two groups depending on whether the price of the consumer good is more than 100. If the item is priced at more than 100, the items are classified as non-food items. Otherwise, the Decision Tree splits the remaining samples into two new groups depending on whether they were purchased from the grocery store "Meny". This results in a total of 50 samples classified as non-food items (40 + 10) and 30 items classified as food items.

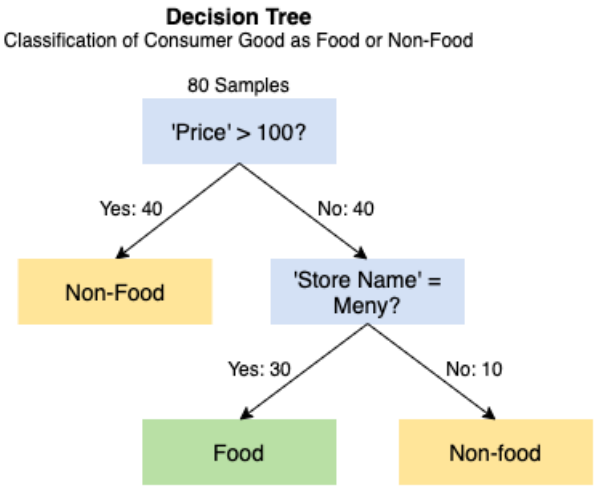


Figure 2.7: Decision Tree example

To decide which questions to ask, i.e., which feature that creates the best split of samples, the Decision Tree classifier adopts a "greedy" *divide-and-conquer* strategy by choosing to test splits on the most important features first. This is typically done by using the *Information Gain* measure as the objective function to maximise. By maximising the Information Gain, shown in equation 2.4, the algorithm can identify which feature that yields the highest information value, enabling the algorithm to make efficient and optimised splits.

$$IG(D_p, f) = I(D_p) - \sum_{j=1}^{m} \frac{N_j}{N_p} I(D_j) \tag{2.4}$$

- IG: Information gain
- I: Impurity measure
- f: Feature that performs the split
- D_p: Data set of the parent node
- D_j: Data set of the j^{th} child node
- N_p: Total number of samples at the parent node
- N_j: Number of samples at the j^{th} child node

In equation 2.4, I refers to the *impurity measure*. This is used to calculate the *impurity* at each node, indicating how many samples that belong to the same class at a particular node. The more samples that belong to the same class within a node, the lower the impurity measure is. For a split that results in a node that contains only samples that belong to a single class, the impurity measure would be at its minimal value. On the other hand, a node that contains more of an even distribution of classes would consequently have a higher impurity measure. *Gini* and *Entropy* are examples of impurity measures that are commonly used. Equation 2.5 shows how impurity is calculated using the *Gini* (I_G) impurity measure.

$$I_G(t) = \sum_{i=1}^{c} p(i|t)(1 - p(i|t)) = 1 - \sum_{i=1}^{c} p(i|t)^2 \tag{2.5}$$

- $p(i|t)$: Proportion of the samples that belong to class c for a particular node t

Random Forest

Random Forest can be considered an extension of the Decision Tree algorithm as the Random Forest algorithm is an ensemble of decision trees that computes its final prediction by aggregating the individual predictions made by each decision tree and assigns the class label by majority voting. Figure 2.8 builds on the previous example in Figure 2.7 and illustrates how an ensemble of k decision trees make predictions for whether samples of consumer goods are food or non-food items. The resulting final prediction is the prediction with the majority of votes.

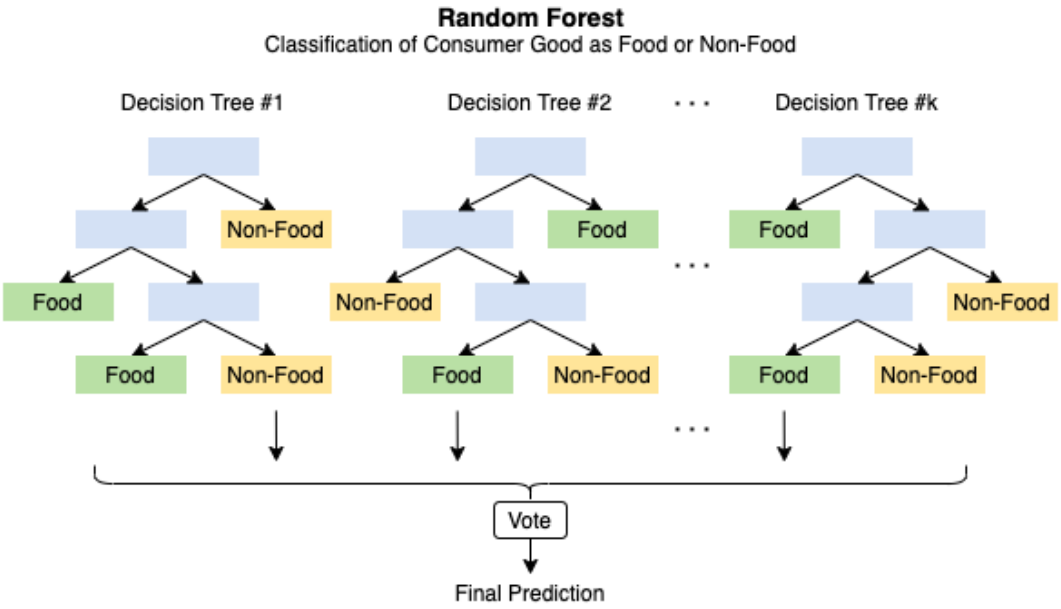


Figure 2.8: Random Forest example

An individual decision tree is prone to overfitting as the tree grows deeper when more splits are made (Raschka and Mirjalili, 2019, p.91). In the random forest algorithm, multiple deep trees are used in the full ensemble, where each decision tree has been fitted to different bootstrap samples (random sampling with replacement) of the full dataset. By averaging predictions over multiple overfitted decision trees, the random forest becomes much more robust to noise and is able to compensate and correct for much of the individual decision trees' tendency to overfit. This comes at the cost of computation cost as a random forest typically fit a couple hundred or thousand individual decision trees to make its ensemble of decision trees on which to base its final predictions (Raschka and Mirjalili, 2019, p. 100).

2.2.3 Evaluation Metrics for Classification

In order to evaluate a classifier, multiple evaluation metrics can be used to assess model performance for specific problems. Accuracy is a popular metric for many classification problems, and for a machine learning classification problem, the accuracy metric measures the ratio between correctly predicted samples and the total number of samples in the full dataset, where a higher accuracy score means that more samples were classified as the correct class.

Other popular metrics are precision, recall and F_1-score. The calculation of these metrics depends on the number of true-positives (TP), true-negatives (TN), false-positives (FP) and false-negatives (FN). True positives and negatives are expressions for the number of correctly classified samples as the positive and negative class, respectively, while false positives and negatives refer to the number of falsely classified samples.

As shown in equation 2.6, the accuracy metric can be calculated as the sum of true positives and true negatives over the sum of false positives, false negatives, true positives, and true negatives, i.e., the sum of correctly classified samples over the total number of samples.

$$Accuracy = \frac{TP + TN}{FP + FN + TP + TN} \tag{2.6}$$

Precision is an evaluation metric that expresses how many of the detected positives are truly positive. It is defined as the number of true positives over the sum of false positives and true positives, shown in equation 2.7. A high precision score indicates that out of the samples classified as positive, a high number of samples were correctly classified.

$$Precision = \frac{TP}{TP + FP} \tag{2.7}$$

Recall is a measure of the true positives over the sum of true positives and false negatives, shown in equation 2.8, meaning that recall expresses the classifier's ability to correctly classify the positive class. By punishing misclassifications of the positive class, while disregarding the negative class, recall can be a useful metric in fraud detection or medical diagnosis (Raschka and Mirjalili, 2019, p. 214).

$$Recall = \frac{TP}{TP + FN} \tag{2.8}$$

Precision and recall are often combined using the F_1-score, shown in equation 2.9. The F_1-score expresses the weighted average of the two metrics, resulting in a measure that attempts to be both correct and not miss any correct predictions. F_1-score is a popular metric for comparing models, and it is especially useful for assessing performance on unbalanced data, where the accuracy metric might be less suitable.

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} = \frac{2TP}{2TP + FP + FN} \tag{2.9}$$

2.2.4 Multi-class Classification

When faced with more than two distinct classes, the machine learning classification problem becomes a multi-class classification problem. In order to extend a binary classifier, such as Logistic Regression, to a multi-class problem, One-Versus-Rest (OvR) is a popular technique. By using OvR in a multi-class classification problem, one classifier is trained per distinct class, where that particular class is treated as the positive class while the rest of the classes are all treated as the negative class. For a multi-class classification of n classes, a total of n classifiers are trained, and each sample is assigned the class label with the overall highest confidence out of the n classes.

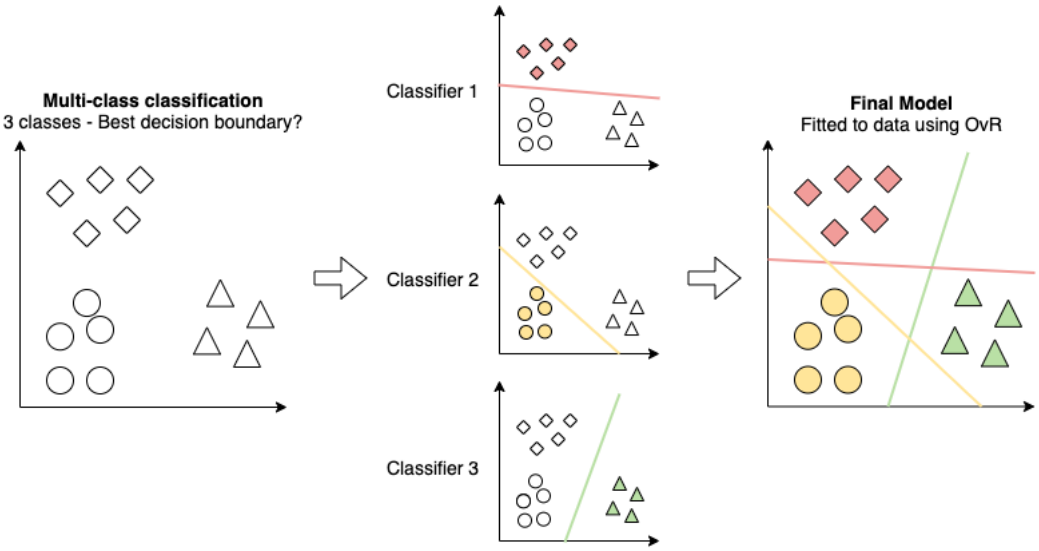


Figure 2.9: OvR in a 3-class classification problem

2.3 Text Representation in Machine Learning

Natural Language Processing (NLP) is a field within linguistics, computer science and artificial intelligence that focuses on the interactions between computers and human language. The expression *natural language* refers to the way humans communicate with each other, and natural language processing can be seen as the automatic manipulation of such language by software to improve the computer's ability to process and derive meaningful information from text samples.

This section introduces relevant methods and concepts that are used in text representation for natural language processing and how this enables the use of classifiers to classify text samples.

2.3.1 Terminology in Text Processing Tasks

This subsection provides a brief explanation of terminology in NLP and text processing tasks that are used in the subsequent parts of this chapter.

Term: A word, symbol or character instance of text.

Document: A text sample or a body of text such as an individual email or a receipt.

Vocabulary: A set of all terms that occur in a document.

Corpus: A collection of documents, such as a collection of emails or receipts.

2.3.2 *N*-grams

N-grams are sequences of n consecutive units in a text, typically sequences of words or characters. When n-grams correspond to a single word in a text ($n = 1$), they are usually referred to as unigrams. Similarly, a bigram ($n = 2$) is used to describe a sequence of length 2, a trigram ($n = 3$) for a sequence of length 3, and so forth.

To illustrate, consider the consumer good item "lambi toalett extra long", a toilet paper variant sold in Norwegian grocery stores. Figure 2.10 shows how different n-grams would partition the item name depending on the chosen value for n.

N-gram type	*N*-gram representation			
Unigram ($n = 1$)	lambi	toalettpapir	extra	long
Bigram ($n = 2$)	lambi toalettpapir	toalettpapir extra	extra long	
Trigram ($n = 3$)	lambi toalettpapir extra	toalettpapir extra long		

Figure 2.10: N-gram representations of "lambi toalettpapir extra long"

N-grams can be used to develop features for machine learning models, where each distinct N-gram serves as an individual feature. This can be applied in NLP-tasks such as spelling correction, text summarisation and speech recognition (Ahmed et al., 2009).

2.3.3 Vectorisation of Words

Vectorisation of words involves turning individual terms from a document or corpus into numeric representation by developing feature vectors from the unique terms. These terms are typically words, characters or n-grams that occur in the documents.

Bag-of-Words

The Bag-of-Words (BoW) is a simple way to represent text where each document is represented as a numerical feature vector. This creates features by first creating a vocabulary of all unique terms across all documents, and then create a feature vector per document and update it by counting the occurrences of each term in the vocabulary. Figure 2.11 shows how a bag-of-words can be applied to documents of arbitrary consumer goods, i_1, i_2, i_3 and i_4, where x_{ij} represents the developed feature vector of consumer good i_j. The list of words with corresponding keys represents the vocabulary constructed from the documents.

Key	Word
0	coop
1	jasminris
2	toalettpapir
3	økonomi
4	lambi
5	extra
6	long
7	husman
8	knekkebrød

Item descriptions
i_1 = coop jasminris
i_2 = toalettpapir økonomi
i_3 = lambi toalettpapir extra long
i_4 = husman knekkebrød økonomi

Feature vectors
X_{i1} = [1, 1, 0, 0, 0, 0, 0, 0, 0]
X_{i2} = [0, 0, 1, 1, 0, 0, 0, 0, 0]
X_{i3} = [0, 0, 1, 0, 1, 1, 1, 0, 0]
X_{i4} = [0, 0, 0, 1, 0, 0, 0, 1, 1]

Figure 2.11: Bag-of-Words from documents of consumer goods

Term Frequency – Inverse Document Frequency

Term Frequency – Inverse Document Frequency (tf-idf) is a technique to derive information about the importance of terms in a document. Tf-idf is calculated by comparing the number of occurrences in a single document to its usage in the entire corpus. The tf-idf transformation results in a weighted numerical representation of the terms instead of using their raw frequencies. Terms that occur frequently in the item descriptions are assigned a lower weight, while the less frequent terms are assigned larger weights to express their distinctiveness. A common way to calculate tf-idf is shown in equations 2.10 and 2.11 (Raschka and Mirjalili, 2019, p. 265).

$$tf\text{-}idf(d,t) = tf(d,t) \times idf(d,t) \tag{2.10}$$

- tf: Term frequency for term t in document d.

$$idf(d,t) = \log(\frac{n_d}{1 + df(d,t)}) \tag{2.11}$$

- n_d: Number of documents in the corpus

- $df(d,t)$: Number of documents d that contain the term t.

Expanding on the previous example, figure 2.12 illustrates how the terms (words) contained in each item description would be transformed using tf-idf. The new feature vectors have been calculated using the default parameters of Scikit-learn's[1] tf-idf transformer, "TfidfTransformer". In this version, "1" is added to each idf-score to prevent zero division (smoothing), resulting in equation 2.12.

$$tf\text{-}idf(d,t) = tf(d,t) \times (idf(d,t) + 1) \tag{2.12}$$

Key	Word
0	coop
1	jasminris
2	toalettpapir
3	økonomi
4	lambi
5	extra
6	long
7	husman
8	knekkebrød

Item descriptions
i_1 = coop jasminris
i_2 = toalettpapir økonomi
i_3 = lambi toalettpapir extra long
i_4 = husman knekkebrød økonomi

Feature vectors
X_{i1} = [0.71, 0.71, 0, 0, 0, 0, 0, 0, 0.]
X_{i2} = [0, 0, 0.71, 0.71, 0, 0, 0, 0, 0.]
X_{i3} = [0, 0, 0.41, 0, 0.53, 0.53, 0.53, 0, 0.]
X_{i4} = [0, 0, 0, 0.49, 0, 0, 0, 0.62, 0.62]

Figure 2.12: Tf-idf transformation of consumer goods

In this example, the entry representing "toalettpapir" in feature vector x_{i3} has been assigned the lowest weight of all entries across all feature vectors as the word "toalettpapir" here occurs in a document that contains 4 words in total, and this particular word occurs in 2 documents (i_2 and i_3) in the corpus.

[1]Scikit-learn is a machine learning software library for the Python programming language (Pedregosa et al., 2011)

Chapter 3

Data

This chapter highlights some key aspects of the data sets used in this thesis. Section 3.1 provides some background information about each data set, while section 3.2 covers the preparation of the data sets and how they were combined into a single training data set for COICOP classification. Lastly, section 3.3 explores some prominent characteristics of the full training data set, as well as some characteristics of the individual data sets.

3.1 Description of Data Sets

All data sets used in this thesis are based on data that have been provided by SSB. The criteria for what would qualify a data set as suitable for this study have been lenient, where all available data sets that contain both consumer goods and an item category code have been included. This has resulted in a selection of data sets that are distinctly different, but they share a common characteristic in that they all contain labelled items in text format.

The following five data sets have been used in this thesis:

Receipts: Data set containing entries of consumer goods that have been extracted from images of receipts or manually registered in SSB's phone app.

Keywords: Data set containing most COICOP subclass codes and a set of common consumer goods that relate to each code.

Transactions: Data set containing entries of consumer goods that have been registered as purchases by Norwegian grocery stores.

CPI: Data set containing entries of non-food items that have been registered by the Consumer Price Index group at SSB.

Imports: Data set containing entries of consumer goods that have been registered as imports by Norwegian customs.

The data sets were originally collected through different means and for different purposes. This section briefly describes how each data set was collected and summarises some key statistics for the respective data sets.

Receipts

The Receipt data set was collected by SSB in their pilot study for the survey of consumer expenditure 2022, previously described in subsection 1.2.3. This data set contains entries of purchased consumer goods, some with their corresponding 5-digit COICOP code. The ones that contain valid COICOP codes are items that have been manually registered into SSB's phone app, while the ones that are missing COICOP codes are items that have been extracted from scanned receipts. The data set contains various features with information about each purchase, such as "store name", "item price" and "purchase date".

Keywords

The Keywords data set is used to facilitate auto-completion of expense registrations in SSB's phone app. As respondents attempt to register their expenses using the phone app, their registrations are matched with similar item names in the Keywords data set. If a match is found, the registration is automatically labelled with the COICOP code that corresponds to the matching item name. This data set contains entries of consumer good descriptions, corresponding 5-digit COICOP codes and item names of each consumer good.

Transactions

The Transactions data set is a product of multiple data sets that have previously been prepared and combined by SSB to assemble a COICOP training data set with 5-digit COICOP 2018 subclass codes. The two main components of this data set are transaction data from Norwegian grocery stores and product catalogues used in previous calculations of Consumer Price Indices (CPI). The transaction data were collected in 2018, while the data set of consumer goods used in the product catalogue are updated for 2021.

CPI

Newly labelled non-food items emerged from the CPI group at SSB while working on this thesis. These items are additional consumer goods that have not yet been added into the previously described COICOP 2018 training data set by SSB. This data set contains items that are labelled using the ECOICOP coding structure. ECOICOP refers to the European COICOP 2016 coding system, a 6-digit code which the CPI department at SSB use to categorise items.

Imports

The Imports data set is a product of individual customs declarations registered with *TVINN*; the Norwegian customs' electronic system for exchanging customs declarations. This data set contains entries of imported goods from 2018 and their corresponding code in the CN 2008 coding format. The CN (Combined Nomenclature) code is a standardised 8-digit coding framework used for classifying goods for common custom tariffs. An important distinction between CN and COICOP, is that the CN code is also used to categorise items that aren't necessarily intended for consumption.

Summary of Data Sets

Table 3.1 summarises some key statistics of the different data sets in their raw format.

Data Set	Rows	Columns	Coding Format
Receipts	14 389	19	COICOP18
Keywords	2 377	3	COICOP18
Transactions	33 272	37	COICOP18
CPI	82 518	13	ECOICOP
Imports	18 030 591	7	CN2008

Table 3.1: Characteristics of raw data sets

Even though some of the data sets listed in table 3.1 contain a high number of entries, many of these do not provide any value to this thesis. A substantial number of entries lack valid item coding annotation, and the CPI and Imports data sets use different coding formats. Data preparations are therefore required to filter out unwanted data entries and transform all relevant data to the same coding format.

3.2 Preparing and Combining Data Sets

This section describes the high-level processing steps involved with each individual data set. This refers to filtering and mapping operations, such as removal of unwanted columns or transformation of column values. For each data set, the objective is to extract a subset containing only entries of consumer goods and their corresponding 5-digit COICOP subclass code. These subsets will subsequently be combined into a single data set of item names and corresponding 5-digit COICOP subclass codes which can be used as training data for a machine learning classifier.

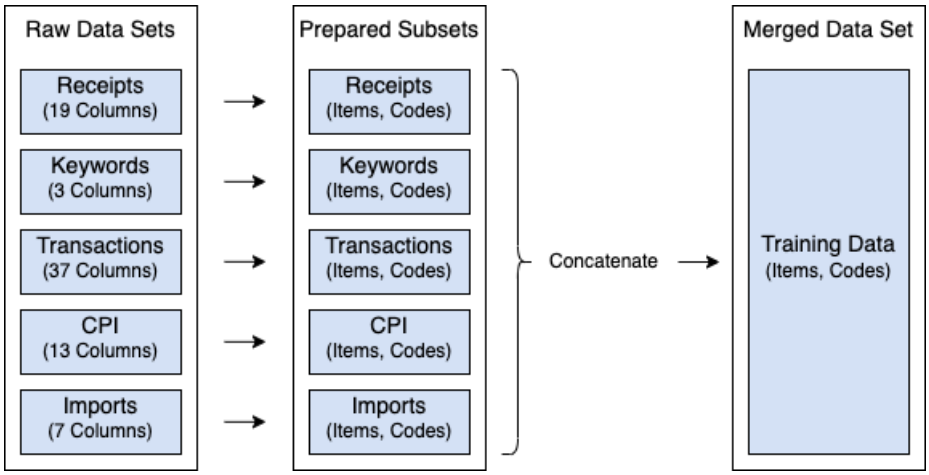


Figure 3.1: Combining data sets into a training data set

Figure 3.1 shows how the different subsets of each data set will be combined into a training data set. Here, "Items" represents item names and "Codes" represents the item coding format in the respective data sets.

3.2.1 Preparation of Receipts and Keywords Data Sets

The Receipts and Keywords data sets were similar in that they both contained entries of item names and corresponding COICOP codes in the 5-digit COICOP 2018 format. In both data sets, the item names and COICOP codes were already in the desired format, and therefore, only a few simple steps were required in the preparation of these data sets. This process is illustrated in figure 3.2.

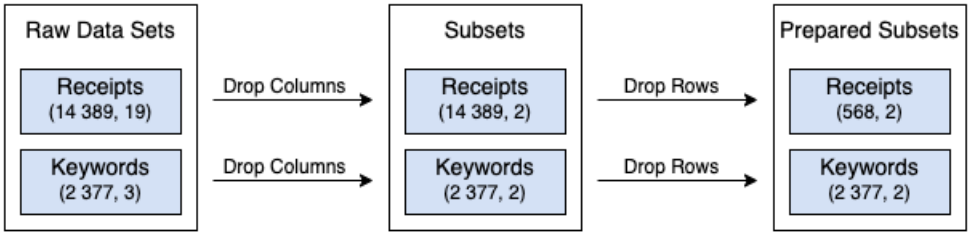


Figure 3.2: Preparation of Receipts and Keywords data sets

For both data sets, a subset was created by extracting the columns for item names and COICOP codes. Next, each row, where the COICOP code was either missing

or invalid, was removed. This resulted in two prepared subsets of 568 and 2 377 entries.

3.2.2 Preparation of Transactions and CPI Data Sets

The Transactions data set, which has previously been used as COICOP training data by SSB, was for the most part already prepared for COICOP classification. The CPI data set, on the other hand, contains many unlabelled entries, and all of these were removed. Next, all columns, except for the ones containing *item names* and COICOP codes, were removed from both data sets. This created two subsets of 29 776 and 23 541 entries, respectively, and these steps are illustrated in figure 3.3.

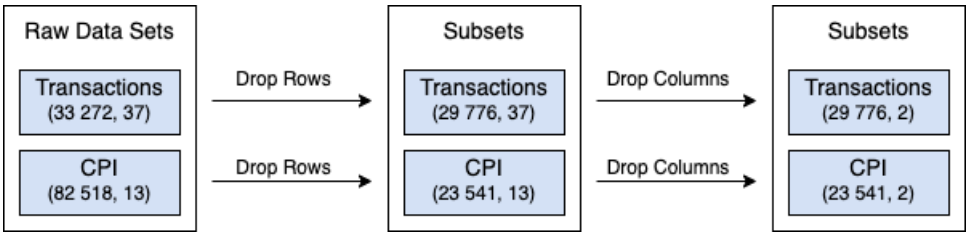


Figure 3.3: Preparations of Transactions and CPI data sets

At this point, each data set contained entries of labelled consumer goods. However, the CPI data set uses the ECOICOP coding format, which differs slightly from the 5-digit COICOP 2018 coding format. SSB have previously constructed their own conversion table between ECOICOP and COICOP 2018 codes, and this table was used to transform the codes into 5-digit COICOP 2018 codes using a many-to-one or one-to-one mapping between the coding formats. This step is shown in figure 3.4.

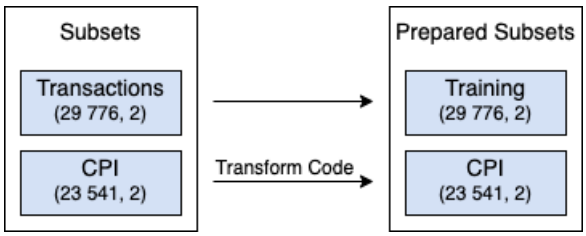


Figure 3.4: Transformation of CPI coding format

3.2.3 Preparation of Imports Data Set

The Imports data set in its original form contained 18 030 591 items entries. However, most of these items were duplicates, and in the first preparation step, all duplicate item name entries were removed. In total, 14 652 184 duplicate rows were removed, and this resulted in a subset of 3 378 407 item entries. This process is illustrated in figure 3.5.

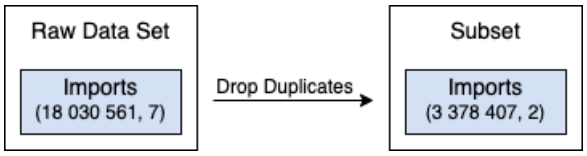


Figure 3.5: Remove duplicate entries in the Imports data set

As mentioned in section 3.1, the Imports data set deviates from the other data sets used in this thesis, mainly due to its CN 2008 coding format. Additionally, the entries in the Imports data set included items that were intended for either consumption or production. COICOP classification is by definition only concerned about items that are intended for consumption, and consequently the production-related items were not relevant to the scope of this thesis. The next step in the preparation of the Imports data set therefore became to transform the coding format of the relevant items into 5-digit COICOP 2018 format.

Code Transformation Pipeline

Eurostat[1] offer publicly available conversion tables between a wide range of item coding formats. These conversion tables made it possible to transform the coding format of the Imports data set into 5-digit COICOP 2018 codes. Additionally, by transforming the coding format into COICOP 2018, all items not intended for consumption were filtered out as there exist no COICOP code conversions for such items. However, Eurostat do not offer conversion tables between all item category codes, and this thesis was unable to identify conversion tables between CN 2008 and COICOP 2018. Therefore, transforming the CN 2008 to COICOP 2018 had to be done in several steps by employing multiple of Eurostat's conversion tables.

Figure 3.6 illustrates the transformation pipeline used in this thesis. This pipeline relies on three conversion tables from Eurostat for the full transformation. First, the CN 2008 codes are transformed to CPA 2008 codes[2], and these are then transformed into CPA 2.1 codes[3]. Finally, the CPA 2.1 codes can be transformed into COICOP 2018 codes.

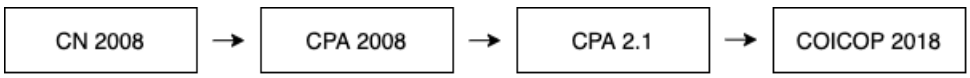


Figure 3.6: Applied code transformation pipeline

[1]Eurostat is the statistical office of the European Union
[2]CPA is a 6-digit coding framework that refers to "Classification of Products by Activity"
[3]CPA 2.1 is an updated coding framework of CPA 2008

Even though the CN 2008 codes are on 8-digit format, which in many cases are more detailed than the 6-digit coding structure of CPA and the 5-digit coding structure of COICOP 2018, the full conversion would in many cases still yield no matches or a set of multiple possible matches. Multiple matches made the transformations ambiguous, and these inconclusive transformations were troublesome in that they require further domain knowledge to resolve. This problem could be avoided by simply using a many-to-one or one-to-one mapping between the coding formats, i.e., a mapping where a code has exactly one match. However, this approach resulted in a large portion of the Imports data set to be lost, where preliminary tests resulted in 96.78% of the 3 378 407 items lost in the transformation pipeline when using many-to-one or one-to-one mappings in all transformation steps.

Custom Search Algorithm

To counteract the large data loss, a *custom search algorithm* for code transformations was developed and implemented into the transformation pipeline. The custom search algorithm would iteratively remove one digit from the right side from both the code intended for transformation and all codes in the same format in the conversion table for a pre-defined range of digits. In this way, possible matches could be detected at lower detail levels, and this was meant to prevent instances of zero matches between transformation steps. Whenever a possible match was found, the algorithm would return the target code of the match(es). The python code for this algorithm is included in Appendix E.2.

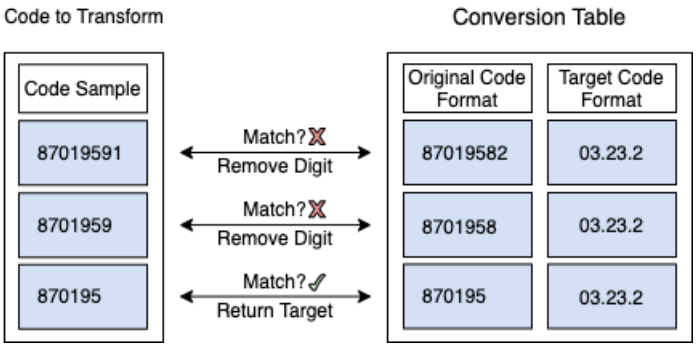


Figure 3.7: Custom Search Algorithm for identifying item code matches

Figure 3.7 illustrates how this is done for an arbitrary 8-digit code. Here, a match is identified whenever the code in the code sample and the original code in the conversion table are identical. In the figure 3.7, a match is not found immediately (at 8-digits), and the right-most digit is removed from the code in the *code sample* and in the *original code* in the conversion table. This is repeated until a match is found at 6 digits, and this resulted in the transformation of code "87019643" to "03.23.2".

Where there are ambiguous mappings, the custom search algorithm would return the set of all possible matches. This is shown in figure 3.8. Here, three distinct CN 2008 codes are transformed to CPA 2008 codes with the custom search algorithm.

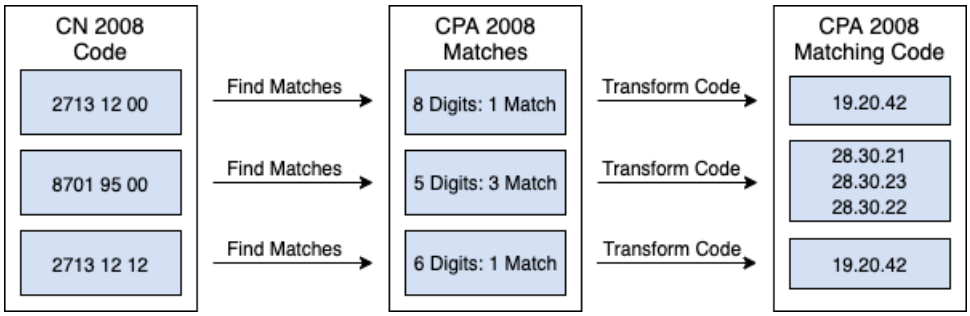


Figure 3.8: Custom Search for matches between CN 2008 and CPA 2008

Performing Code Transformations

By employing the *custom search algorithm* into the *code transformation pipeline*, it became possible to transform the coding format of a significant portion of the Imports data set. The transformation of each item was done by first creating a direct conversion table between the unique CN 2008 codes in the Imports data set and the identified COICOP 2018 code matches. Following the steps of the transformation pipeline, 5 289 out of the 6 183 unique CN 2008 codes in the Imports data set were successfully transformed into COICOP 2018 codes. This is shown in figure 3.9.

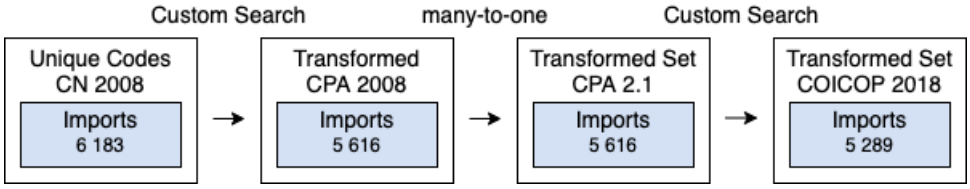


Figure 3.9: Conversion of unique CN 2008 codes to COICOP 2018 codes

The custom conversion table was then used to transform all CN 2008 codes in the full Imports data set. Figure 3.10 illustrates the transformation and data loss involved with this conversion. Here, all entries that have more than one matching COICOP 2018 code with their CN 2008 code were excluded. This resulted in a data set of 1 433 947 items with corresponding 5-digit COICOP 2018 code, meaning that 42.44% of the entries were successfully transformed to a single 5-digit COICOP code.

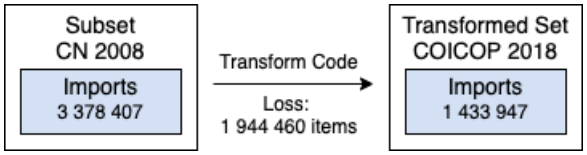


Figure 3.10: Transformation of CN 2008 to COICOP 2018 codes

Finally, in the last preparation steps, all columns except the *item names* and the *5-digit COICOP codes* columns were removed, as well as all rows containing invalid values. This resulted in a prepared Imports subset of 1 433 947 entries with item names and 5-digit COICOP codes.

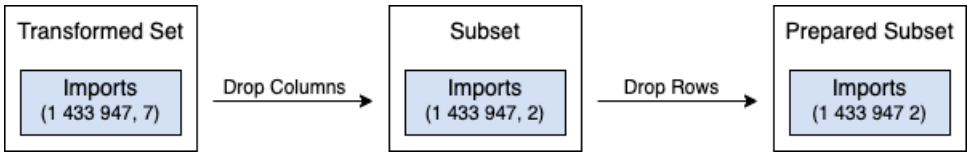


Figure 3.11: Preparation of Imports data set

3.2.4 Combining Data Sets into a Training Data Set

After each data set was prepared, the individual data subsets were combined into a training data set. This was done by vertically concatenating the individual data sets, creating a combined training data set of 1 490 216 rows and 2 columns, as shown in figure 3.12. The resulting training data set was later used to train classifiers, and this, in addition to further text processing and feature extraction from the item names, are described in chapter 4.

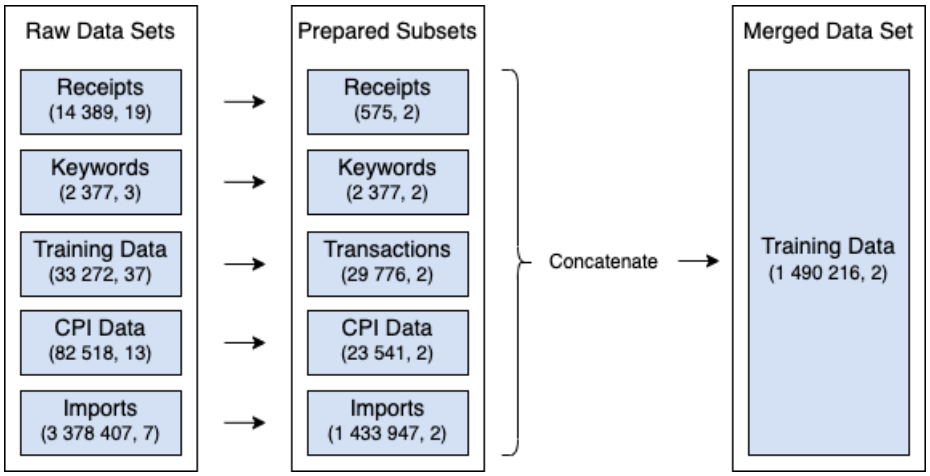


Figure 3.12: Combining data sets into a full training data set

3.2.5 Acquiring a Scanned Receipts Test Set

The previous subsections described the methods employed to prepare and combine the different subsets and the steps involved with acquiring a full COICOP training data set. However, as none of these sets contains labelled items from actual scanned receipts, they may not give good indications of classification performance on such data. In order to explore how well the performance of a classifier model that is trained on data from such diverse sources carry over to items from scanned receipt data, a test set of items from scanned receipts was assembled.

The test set was acquired by drawing a random sample of size $n = 1000$ from the Receipts data set. Due to limited time and available resources, a larger sample size was not feasible. In the sample, only unlabelled items were included, as these represent items that were extracted from scanned receipts. Additionally, all items with a non-positive price value were excluded from the data set before drawing the sample, as these items typically relate to bundle discounts, store-specific offers, or in many cases, ”pant” (bottle deposit).

After drawing the random sample, a coder at SSB manually labelled each item with its 5-digit COICOP 2018 code. The coder was privy to additional information about each sample, where variables such as *price* and *store name* were used to help resolve the ambiguity of inconclusive samples. Despite this, a total of 53 items were still considered too ambiguous to be labelled correctly. These are items with especially succinct item names, containing only descriptions such as "dagligvarer", "diverse", "mat" and so on. Some of these item entries might be the results of erroneous word extraction from the optical character recognition step (see subsection 1.2.4), or simply due to how specific stores would register their own items. Nevertheless, these items were removed from the test data set, resulting in a test set of 947 labelled items from scanned receipts, shown in figure 3.13.

Figure 3.13: Labelling the Scanned Receipts Test Set

3.3 Characteristics of the Data Set

This section presents results from various analyses on the combined COICOP training data set. As described in subsection 3.2.4, this data set consists of two columns: *item names* and *5-digit COICOP codes*. The following subsections outline some observed characteristics and attributes of these variables, first exploring the target class, namely the 5-digit COICOP codes, and then investigating some key characteristics of the words used in the different item names. Additionally, comparisons to the Scanned Receipts Test Set are presented to demonstrate differences between the item types in both data sets.

3.3.1 COICOP Codes

The structure of the COICOP 2018 classification system was previously described in subsection 1.1.1. As outlined in that subsection, there are several hierarchical levels of the COICOP 2018 codes, ranging from 2-digit to 5-digit codes. Whereas the classification of items will be done following the 5-digit COICOP coding structure, the 2-digit COICOP coding format, called "division", can still provide a useful and quick overview of how the items are generally distributed across the different categories. Table 3.2 lists the different COICOP division codes, and these will be used to illustrate the representation of each division category in the training data set in figure 3.14. The complete distribution of subclasses can be found in Appendix A.1.

Code	Description
01	Food and Non-Alcoholic Beverages
02	Alcoholic Beverages, Tobacco and Narcotics
03	Clothing and Footwear
04	Housing, Water, Electricity, Gas and other Fuels
05	Furnishings, Household Equipment and Routine Household Maintenance
06	Health
07	Transport
08	Information and Communication
09	Recreation, Sport and Culture
10	Education Services
11	Restaurants and Accommodation Services
12	Insurance and Financial Services
13	Personal Care, Social Protection and Miscellaneous Goods and Services

Table 3.2: COICOP 2018 division codes (UN, 2018)

Figure 3.14 shows the number of items in each division category in the training data. The largest portion of samples falls under division code 05, where approximately 27% of the total data set is concentrated. Furthermore, division codes 10, 11 and 12 are barely represented.

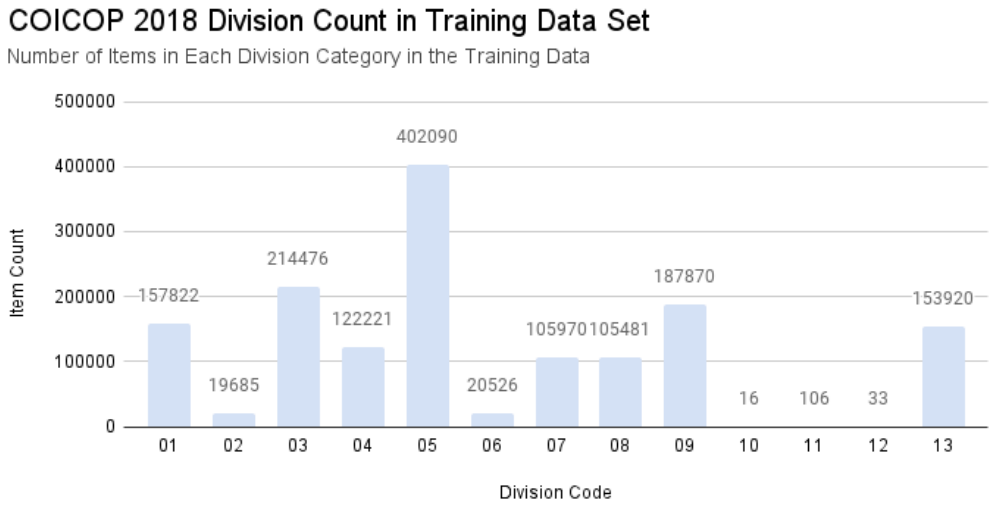


Figure 3.14: Number of items in each COICOP division code in the training set

Figure 3.15 breaks down the training data set into the different individual data sets that were combined and shows the number of samples in each division category for each data set. There is a large difference in the number of samples within each data set, and the plots presented in this figure are therefore on different scales.

Figure 3.15: Number of items in each COICOP division code in each data set

The plots in figure 3.15 show that a large portion of the Receipts and Transactions data are food items (division code 01), while in the Imports data set, most samples are found within division code 05. There are notably few samples representing division codes 10, 11 and 12 in all data sets, where the few existing samples appear exclusively in the Receipt and Keywords data sets.

There is a total of 338 COICOP 2018 subclass codes in the COICOP 2018 coding framework (UN, 2018). However, only 298 of these specifically relate to individual consumption, and these are the ones that are relevant to the scope of this thesis. The motivation behind combining the data sets was to accumulate a COICOP training data set that represents as many as possible of these 298 subclass codes. Table 3.3 summarises the representation of distinct subclasses in the different data sets, as well as in the combined COICOP training data set.

Data Set	Items	Distinct Subclasses	% of 298 Relevant Subclasses
Receipts	575	117	39.26%
Keywords	2 377	274	91.94%
Transactions	29 776	135	46.64%
CPI	23 541	27	9.06%
Imports	1 433 947	117	39.26%
Training Data	1 490 216	283	94.97%

Table 3.3: Number of distinct subclass codes in each data set

As shown in table 3.3, the attempted combination of data sets was unsuccessful in obtaining a training data set where all 298 different subclasses are represented. A total of 15 subclass codes are missing. For a detailed overview of the missing subclass codes, see Appendix A.1. Furthermore, table 3.3 shows that the Keywords data set contains the highest number of distinct subclasses, where only 9 additional distinct subclasses were added to the combined training data set through any of the other four data sets. The CPI data set contains the fewest number of distinct subclass codes out of any of the data sets with only 27 distinct codes represented in total. This means that out of the 23 541 samples in the CPI data set, only 9.06% of all relevant subclasses are found.

Although table 3.3 shows a representation of subclass codes in the combined training data set that is close to the total number of available subclasses (94.97%), it does not fully communicate the class imbalance in the data set. Table 3.4 provides some additional insight to this, showing the total number of distinct subclasses in each data set that are represented by n samples, where $n > 0$.

	Subclasses $n \leq 5$		Subclasses $n \leq 10$		Subclasses $n \leq 15$	
Data Set	Count	Percent	Count	Percent	Count	Percent
Receipts	88	75%	104	89%	109	93%
Keywords	145	53%	204	74%	241	88%
Transactions	22	16%	29	21%	37	27%
CPI	1	4%	1	4%	2	7%
Imports	1	1%	2	2%	2	2%
Training Data	67	24%	90	32%	100	35%

Table 3.4: Number of distinct subclass codes represented by n samples in each data set where $n > 0$. *Percent* refers to the ratio between unique subclass codes represented by n samples and the total number of unique subclass codes in the data set

As shown in table 3.4, a total of 100 subclass codes are represented by 15 or fewer samples in the combined training data set. This means that out of the 283 subclass codes in this data set, 100 (35%) are barely represented by any samples. This suggests that the 1 460 216 samples in the full training data set are unevenly distributed across the represented subclass codes. Figure 3.16 shows the 25 most frequent subclass codes in the training data. Together, the top 25 most frequent subclass codes make up 74% of all samples in the full training data set.

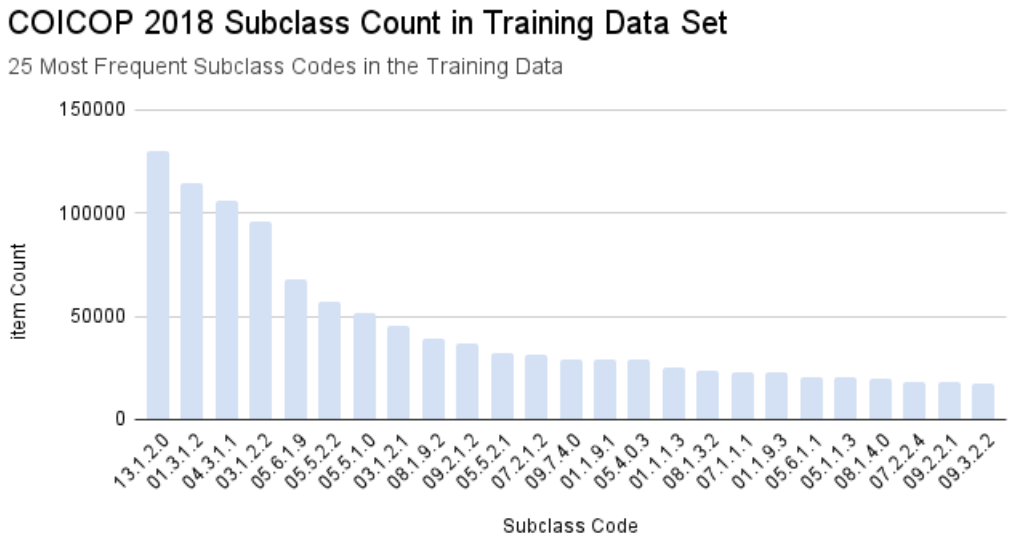


Figure 3.16: Most frequent subclass codes in the training data set

Comparisons with the Scanned Receipts Test Set

There are differences between the COICOP codes represented in the combined training data set and in the Scanned Receipts Test Set. As described in section 3.2, these data sets were sampled differently, and the following paragraphs presents the representation of COICOP codes in the Scanned Receipts Test Set and attempts to summarise some of the prominent differences between the item types in both data sets.

Figure 3.17 shows the number of items within each division category in the Scanned Receipts Test Set. Here, around 68% of the items are concentrated within the food category (division 01). As opposed to the combined training data set, this test set contains close to no samples within division codes 04 and 08. The complete distribution of subclass codes for the Scanned Receipts Test Set is included in Appendix A.2.

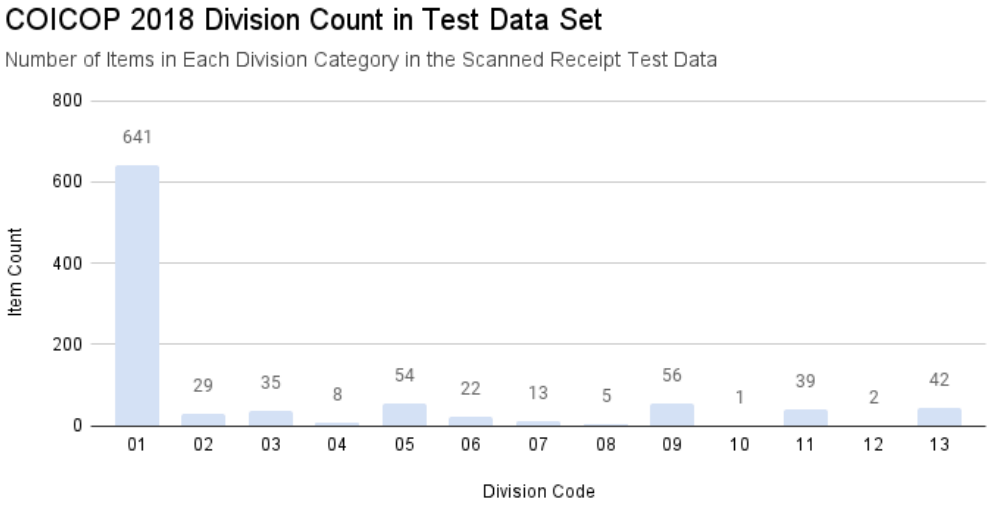


Figure 3.17: Number of items in each COICOP division code in the test data set

Figure 3.18 shows the 25 most frequent subclass codes in the Scanned Receipts Test Set. This data set contains 131 distinct subclass codes, and the top 25 most frequent ones make up 61% of all samples in the full Scanned Receipts Test Set. Among the subclass codes included in this figure, 6 of them are also among the top 25 most frequent subclass codes of the combined training data set (see figure 3.16), and these are highlighted in yellow.

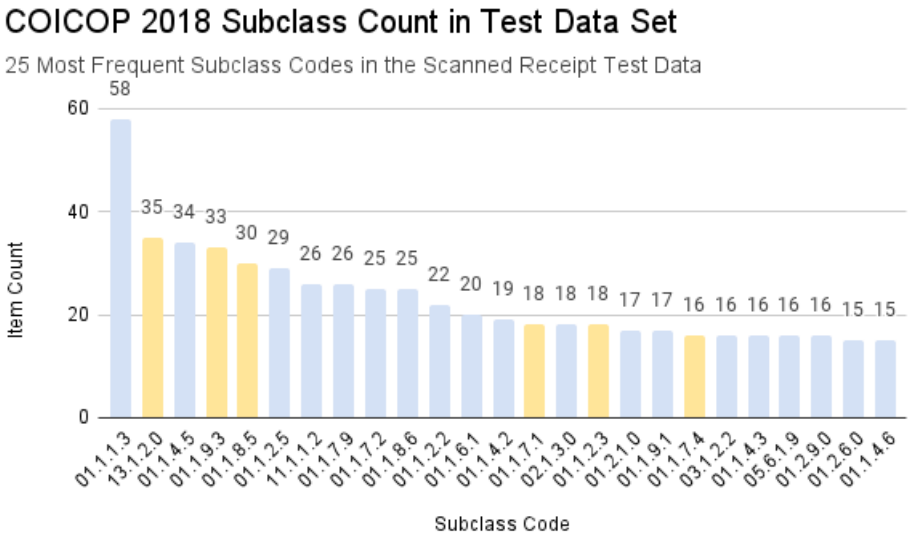


Figure 3.18: Most frequent subclass codes in the test set. Yellow columns indicate which subclass codes that are also among the most frequent in the training data set

As shown in figure 3.18, some of the most frequent subclass codes in the Scanned Receipts Test Set also occur among the most frequent ones in the combined training data set. However, among the remaining most frequent subclass codes in the Scanned Receipts Test Set, some are represented by notably few samples in the combined training data set. Figure 3.19 shows the number of samples in the combined training data set that represent the 25 most frequent subclass codes in the Scanned Receipts Test Set.

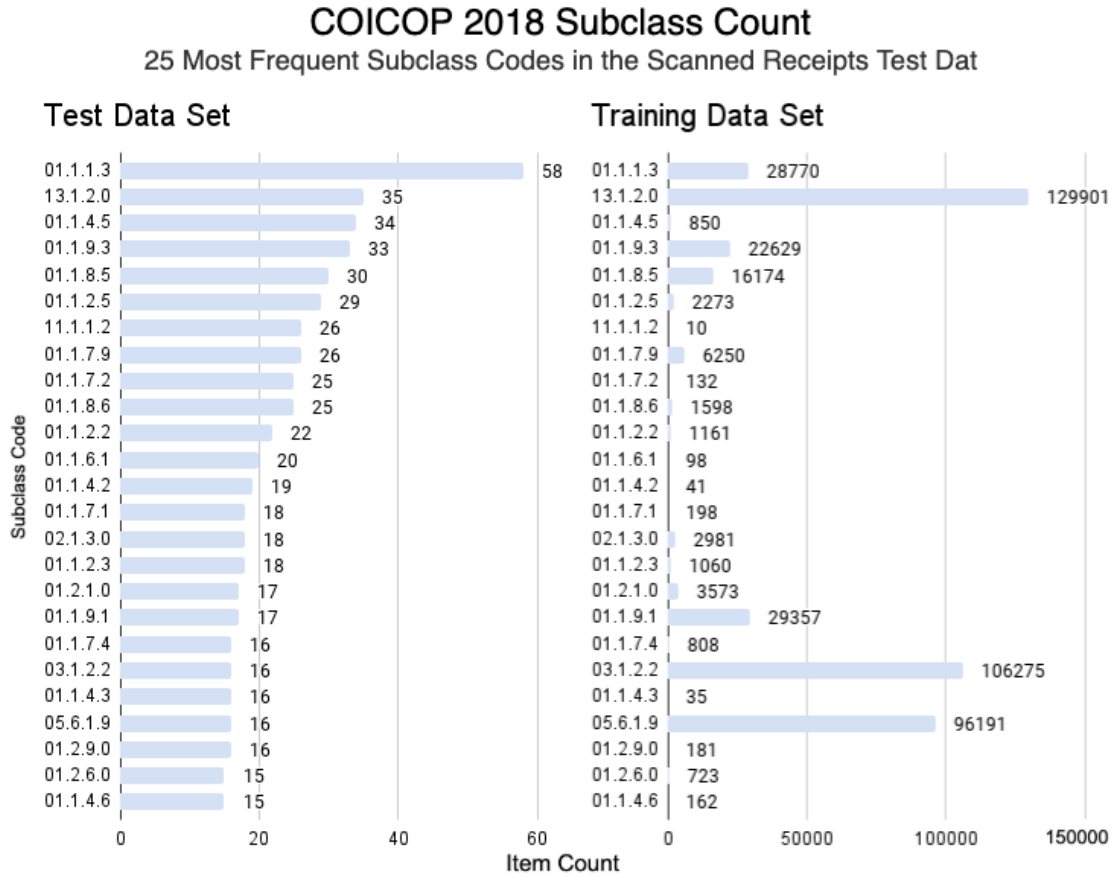


Figure 3.19: Number of samples in the training data set representing the most frequent subclass codes in the test data set

As illustrated in figure 3.19, there is a large variation in the number of samples representing the most frequent subclass codes of the Scanned Receipts Test Set. In the combined training data set, subclass codes 13.1.2.0, 03.1.2.2 and 05.6.1.9 are represented by nearly 100 000 samples each, while subclass codes 11.1.1.2, 01.1.6.1, 01.1.4.2 and 01.1.4.3 are represented by less than 100 samples.

3.3.2 Item Names

This subsection explores attributes of the item names that occur in the combined training data set. When training a classifier model, the item names will be used to predict the 5-digit COICOP 2018 code, and this subsection attempts to highlight some of the salient aspects of the item names.

Figure 3.20 shows that there is a disparity between the number of words in each data set, where the item names in the Imports set are typically longer than in any of the other data sets. However, since the different data sets are from different sources, and originally collected for different purposes, some variation between the item names is to be expected.

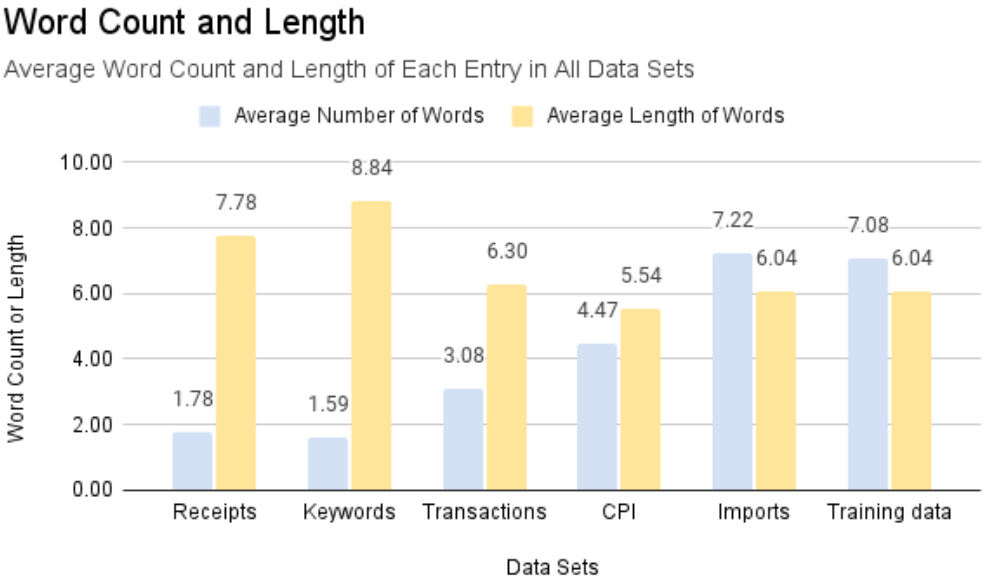


Figure 3.20: Average word count and length in each data set

The length of each word is typically shorter for the Imports and Transaction data, indicating that the item names in these data sets might contain more instances of incomplete words, abbreviations, quantity measures, or other special expressions. The item names in the Receipts and Keywords data set contain very few words on average, while each word is typically longer than in the Transactions and Imports data sets. This could mean that there is less ”noisy” text in these item names, and that each word in these data sets provides more valuable information for COICOP classification.

The number of unique words gives an indication of the variation of words that exists within each data set. Table 3.5 shows the number of distinct words and number of distinct words per item in the different data sets, illustrating that there are a total of 549 434 distinct words in the combined training data set.

Data Set	Items	Distinct Words	Distinct Words per Item
Receipts	575	304	0.53
Keywords	2 377	2 398	1.01
Transactions	29 776	17 717	0.60
CPI	23 541	14 051	0.60
Imports	1 433 947	532 656	0.37
Training Data	1 490 216	549 434	0.37

Table 3.5: Distinct words in each data set

The column *Distinct Words per Item* in table 3.5 describes the ratio between the distinct words and the total number of items in each data set. This can be seen as a measure of the uniqueness of each item name in each data set. For the Keywords data set, this measure is above 1, meaning that there are more distinct words than there are entries in this data set. The Imports data set has the lowest value of 0.37, meaning that there is approximately only one unique word for every third item name in this data set. With the aim of assembling a large vocabulary of item names for the classifiers to learn, the Keywords data set seems to be the most bang for the buck as it introduces many new words over relatively few entries. Still, a majority of the unique words originate from the Imports data set, and these words will contribute to most of the unique item names in the vocabulary.

Table 3.6 shows the percentage of entries in each data set that contain exclusively numerical, alphabetical, or alphanumerical characters. Additionally, it shows the percentage of entries that contain special characters, such as "-", "&", "%", i.e., not only alphanumeric characters.

Data Set	Only Num	Only Alphabetic	Only AlphaNum	Special Ch.
Receipts	0.00 %	77.91 %	78.09 %	21.91 %
Keywords	0.00 %	83.68 %	83.80 %	16.20 %
Transactions	0.02 %	78.05 %	86.77 %	13.23 %
CPI	0.00 %	18.44 %	55.14 %	44.86 %
Imports	0.01 %	19.75 %	79.20 %	20.80 %
Training Data	0.01 %	21.02 %	78.98 %	21.02 %

Table 3.6: Item names characteristics in each data set

Some of these special characters and numeric values in the item names might not be useful to determine which COICOP subclass code an item belongs to, and some of them might only add noise to the data. This is further explored in section 4.1 in the next chapter of this thesis.

Chapter 4

Model Architecture

Subsection 1.2.4 described the proposed pipeline that converts images of scanned receipts to fully classified consumer goods. As outlined in the same subsection, the work involved with this thesis aimed to explore approaches to step 3 and 4; *Processing and Vectorisation of Words* and *COICOP Classification*. Whereas chapter 3 described the data collection procedure and the choices that went into the selection and preparation of data sets, this chapter describes the design of the classifier models and how these models will be evaluated.

This chapter is split into three sections, where the first section, 4.1, describes the chosen approach to step 3 of the pipeline: *Processing and Vectorisation of Words*. This includes the steps involved with preparation of data before training a classifier model. The next section, 4.2, summarises the design of the COICOP classifiers themselves, namely which classifiers to explore and how these were trained. Hence, this section describes the chosen approach to step 4 of the pipeline; *COICOP Classification*. Lastly, section 4.3 outlines a proposed classification system that facilitates automatic classification and describes how to assess the classifier models against that system.

4.1 Text Processing

"The quality of the data and the amount of useful information that it contains are key factors that determine how well a machine learning model can learn" (Raschka & Mirjalili, 2019, p. 109). In the previous chapter, section 3.2 described the higher-level filtering steps that went into the preparation of the data sets. This section presents the document-level data preparation steps, describing the chosen natural language processing (NLP) methods for text pre-processing and the feature extraction of item names.

Table 3.6 showed that 24.52% of the samples in the training data contain special characters, and around 0.01% of the samples contain only digits. These terms typically contain very little discriminatory or valuable information for a classifier to learn (Škrlj et al., 2021). Pre-processing of text samples is often used to filter out such non-alphanumeric characters, along with other specific terms that might only add noise. This can for instance be done by simply removing the "noisy" terms or replacing them with pre-defined terms that might contain more valuable information.

Classifying items from receipts is a peculiar field of text classification in that each item name is often succinct, and the language is typically specific and sparse. There are generally no verbs nor adverbs, but rather nouns, and in some cases, adjectives used to describe the items (Maslova & Da Cruz, 2019). Unlike longer paragraphs of text, these short item names hardly communicate much, if any, contextual information, and this can result in ambiguous item names that even a human would struggle to classify. Combining these ambiguous item names with a classification framework (COICOP 2018) which requires high levels of detail in the classifications, the classification task might even be impossible. Hence, stripping too much of the raw item names might negatively affect the classifiers' performance. This thesis has therefore exercised some restraint in the pre-processing of the item names to avoid removing too much of potentially discriminating semantic information.

4.1.1 Pre-Processing of Item Names

To illustrate the effects of the text pre-processing steps used in this thesis, this subsection presents a handful of item names from the Receipts data set. All of which were extracted from images of scanned receipts as part of the pilot study (see subsection 1.2.3).

Nr	Items Names Extracted from Scanned Receipts
1	solsikkefrø 1 kg first price
2	Nicotinell tyggegum 2mg icemin\n96 ENPAC
3	Yoghurt vanilje 4x150g tine\n\kr\n1 stk - 1 %
4	1x apple cider vinegar sjampo\n385ml
5	100% APP JUICE M/K

Table 4.1: Items names extracted from scanned receipts

The ordering in which the text pre-processing steps are applied plays an important role in obtaining the intended text representation. For instance, functions that substitute specific expressions will become less effective if parts of those expressions were removed in another step prior to running the substitution function. This thesis decided on the following text pre-processing pipeline, and the ordering of these steps, are as follows:

1. **Lowercase**: Change all letters to lowercase
2. **Quantities**: Substitutes quantities with specified expressions
3. **Numbers**: Removes numbers
4. **Special Characters**: Removes a set of special characters, e.g., ("-", "%", "&")
5. **Single Characters**: Removes characters that occur alone
6. **Newline Characters**: Removes newline character ("\n")
7. **White Space**: Removes white space between characters

The text pre-processing pipeline was specifically designed with the output from the OCR-step in mind. As described in subsection 1.2.4, the OCR step in the pipeline outputs text that has been extracted from images of scanned receipts, and the items listed in table 4.1, are examples of such text samples. The following paragraphs summarises the effects of the pre-processing steps above, and table 4.2 shows how the different item names from table 4.1 were transformed after all steps of the text pre-processing pipeline were performed in order.

First, all letter in the item names were changed to lowercase. This was done to simplify processing of words that are different in capitalisation only, and to later avoid creating redundant features from these words. In this way, occurrences of "JUICE" and "juice" will be recognised as the same word.

Since COICOP categories are generally not concerned with the quantity measures of items, these measures could be removed. However, in an attempt to retain some semantic information from these quantitative measures, they were instead substituted to either "FAST" for the solid goods or "FLYTENDE for the liquid goods. The words chosen to replace the quantity measures were deliberately capitalised in order to express their distinctiveness from words that stem from the original data set. This transforms item names such as "solsikkefrø 1 kg first price" to "solsikkefrø FAST first price".

Some additional pre-processing steps were also performed, including removing numbers, removing newline character ("\n"), and removing white space between characters. This was done in an attempt to derive less noisy text samples that are ready to be extracted into feature vectors. The python code with each text processing step is included in Appendix E.1. The result after performing each pre-processing step is shown in table 4.2.

Nr	Pre-Processed Items Names
1	solsikkefrø FAST first price
2	nicotinell tyggegum FAST icemin enpac
3	yoghurt vanilje FAST tine kr stk
4	apple cider vinegar sjampo FLYTENDE
5	app juice

Table 4.2: Pre-processed items names

4.1.2 Feature Extraction

After the initial text pre-processing steps, feature extraction was done to create numerical features of the item names. Two different methods were explored to create numerical feature vectors: vectorisation with *bag-of-words* and *tf-ifd*. These have all been previously described in subsection 2.3.3.

The count-based feature extraction methods, bag-of-words (CountVectorizer) and tf-ifd (TfidfVectorizer), were implemented using Scikit-learn. This thesis explored using different N-gram ranges at both word- and character-level to construct feature

vectors from item names. The combinations of feature extractors and N-gram ranges that were used in this thesis are listed in table 4.3.

Name	Feature Extractor	Analyser	N-gram Range
CV-w	CountVectorizer	word	(1,1)
CV-w22	CountVectorizer	word	(2,2)
CV-w23	CountVectorizer	word	(2,3)
CV-w33	CountVectorizer	word	(3,3)
CV-ch22	CountVectorizer	character	(2,2)
CV-ch23	CountVectorizer	character	(2,3)
CV-ch33	CountVectorizer	character	(3,3)
TFIDF-w	TfidfVectorizer	word	(1,1)
TFIDF-w22	TfidfVectorizer	word	(2,2)
TFIDF-w23	TfidfVectorizer	word	(2,3)
TFIDF-w33	TfidfVectorizer	word	(3,3)
TFIDF-ch22	TfidfVectorizer	character	(2,2)
TFIDF-ch23	TfidfVectorizer	character	(2,3)
TFIDF-ch33	TfidfVectorizer	character	(3,3)

Table 4.3: Chosen feature extraction methods and parameter settings

The different N-gram ranges and analyser methods determine the dimension of the transformed feature matrix. As the count-based feature extractor methods aim to represent the frequencies of individual terms in a document, the constructed feature matrices tend to get high-dimensional (Shahmirzadi et al., 2019). Table 4.4 shows how the number of features created varies depending on the choice of analyser and N-gram range for the full COICOP training set.

Analyser	N-gram Range	Number of Created Features
word	(1,1)	203 798
word	(2,2)	824 764
word	(2,3)	1 692 374
word	(3,3)	867 610
character	(2,2)	939
character	(2,3)	18 891
character	(3,3)	17 952

Table 4.4: Number of features created from the COICOP training data set for different variation of feature extractor analyser and N-gram range

As shown in table 4.4, the number of created features largely depends on the *analyser* and *N-gram range*. For the training data set used in this thesis, the word-based vectorisation methods created more feature than the character-based ones. Whereas more semantic information might be kept in the word-based extraction methods, the character-based methods appear to be more computationally favourable as they appear to require a lower dimensionality feature space to represent the terms. Nevertheless, all combinations of feature extraction methods listed in table 4.3 were explored in this thesis.

4.2 Classifier Model

This section summarises the design of the classifier models that were implemented for classification of items into 5-digit COICOP codes. This includes the choice of classifiers, how the classifier models were trained and evaluated, and how the best performing combination of classifier and feature extractor was identified.

4.2.1 Classifiers

A set of traditional classifiers were chosen to explore the potential of implementing supervised machine learning for classification of items into 5-digit COICOP 2018 subclass codes. These classifiers were largely chosen on the basis of being classifiers that SSB have used in their previous work, and due to their advantages of being easy to implement and typically performing well in text classification tasks without requiring much tweaking or optimisation (Kowsari et al., 2019). The chosen classifiers are listed in 4.5, and each classifier was implemented using Scikit-learn.

Name	Classifier	Scikit-Learn Module
LR	Logistic Regression	sklearn.linear_model.LogisticRegression
RF	Random Forest	sklearn.ensemble.RandomForestClassifier

Table 4.5: Chosen classifiers and their Scikit-learn module

Some classifiers are slower to train on larger sized data sets than others. For instance, Random Forest is trained by fitting several decision trees to the data, and using multiple trees can be computationally expensive as it increases prediction complexity. Logistic Regression, on the other hand, is a simpler model that is typically fast to train. However, at the same time, the Logistic Regression model might be less accurate than Random Forest on large feature data sets (Kowsari et al., 2019).

4.2.2 Performance Metrics

In order to evaluate the performance of the classifiers, the choice of performance metrics must first be defined. Among the performance metrics presented in subsection 2.2.3, the *accuracy* score is perhaps the simplest metric to interpret as it reflects the ratio between correct classifications and total samples in the data set. However, as shown in section 3.3, not all subclass codes are equally represented in neither the assembled COICOP training data nor in the Scanned Receipts Test Set. In these situations, the accuracy metric on its own might fail to accurately communicate the performance of the classifiers as higher accuracy scores can occur if the classifiers simply predict the majority classes for every sample. Therefore, performance metric scores *accuracy*, *precision*, *recall*, and $F1$-*score* are all presented together to give a more detailed picture of each classifiers' overall performance. These metrics, and their calculations, have all been previously described in subsection 2.2.3.

4.2.3 Training and Evaluation Protocols

To meet the objectives of research questions 2 and 3 (see section 1.3), this thesis conducted two primary experiments with training and evaluating COICOP classifiers. These experiments are summarised as follows:

Held-out Data: The classifiers are trained on a training partition of the COICOP training data and evaluated on a held-out test partition.

Scanned Receipts: The classifiers are trained on the full COICOP training data and evaluated on the Scanned Receipts Test Set.

The following paragraphs describe the training and evaluation protocols employed in both experiments.

Held-out Data

In the *Held-out Data* experiment, the classifiers in table 4.5 were trained using a *hold-out* method. This involves splitting the full data set into two partitions, where one partition is used to train the model, while the other is used to evaluate the model's performance on data that it has never seen during training. These partitions are typically referred to as the *training set* and *test set* (Raschka and Mirjalili, 2019, p. 121).

Figure 4.1 illustrates the employed training and evaluation protocol which utilises a *hold-out* method. This thesis used 80% of the data as a training set, while 20% of the data was kept as a test set. This split was made by stratifying on the COICOP subclasses in the data set to ensure that both the training and test partitions contained proportional representations of each COICOP subclass code.

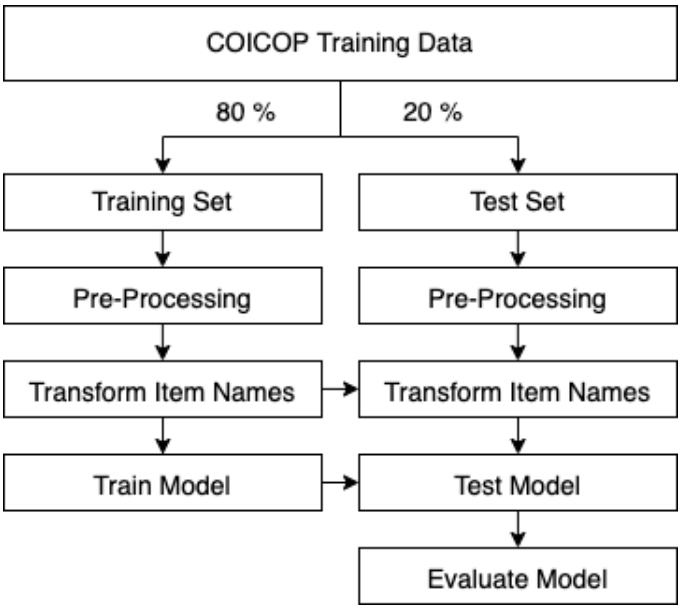


Figure 4.1: Held-out Data: Training and evaluation protocol

After the split into training and test partitions was made, the item names in each partition were pre-processed by applying the pre-processing steps described in subsection 4.1.1. Next, all samples, which at this point contained item names in text format, were transformed into feature vectors using one of the different feature extraction methods described in subsection 4.1.2.

In an attempt to get an unbiased assessment of the model's generalisation performance on unseen data, the test set was kept away from the model at all times during training. This also included the *item names* in the test set, and the feature extractor would therefore only learn to represent the item names that occurred in the training set. The test set is meant to represent unseen data, and some of these item names are expected to be item names that the model has not seen before. As shown in figure 4.2, the feature extractor learns to represent the items names in the training set and is subsequently used to transform the item names in both the training set and test set into numeric feature vectors.

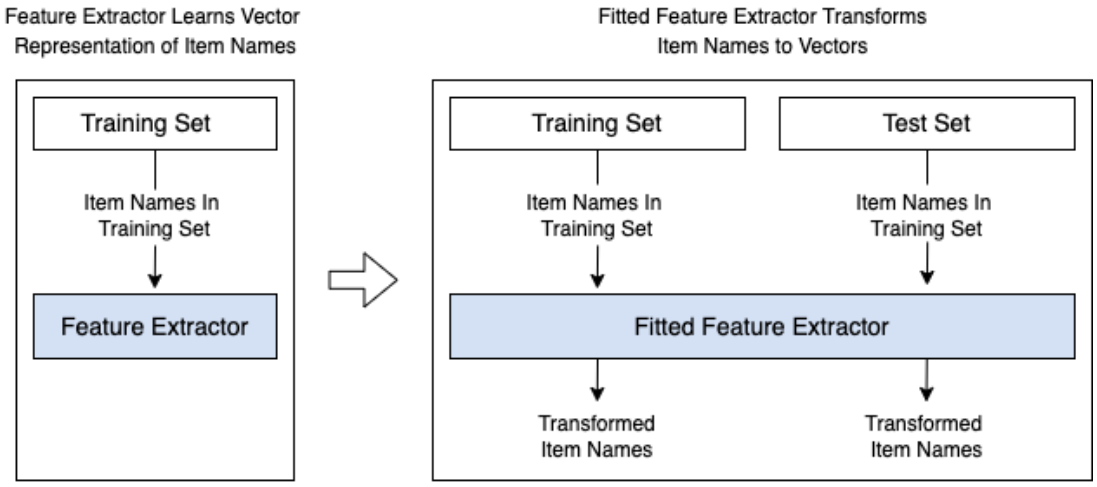


Figure 4.2: Transforming item names with a feature extractor

After transforming the item names into numeric feature vectors, the models were trained on the data in the vectorised training set as shown in figure 4.1. The trained models then attempted to predict the COICOP subclass codes in the vectorised test set. The performance metrics of each model were calculated and used to evaluate their performance on the held-out test set.

However, due to time constraints, the best performing combination of feature extractor and classifier was identified using a subset containing 10% of the full COICOP training data set. Training and evaluation on this subset was done using the *hold-out* method with an 80/20 split, following the same steps shown in figure 4.1. This means that the optimal combination of feature extraction method and classifier might not have been identified, but this approach should still have provided an indication of which combination that is expected to perform well on the full training data set.

In the final evaluation of the classifier models in the *Held-out Data* experiment, the identified best-performing combination of feature extractor and classifier on the subset was retrained and evaluated on the full COICOP training data. This was done

by using a hold-out test set of 20%, following the training and evaluation protocol in figure 4.1. The model performances from this evaluation will be presented using the performance metrics described in subsection 4.2.2.

Scanned Receipts

The *Scanned Receipts* experiment was conducted to assess how well the performances of classifiers trained on the COICOP training data carry over to items from scanned receipts. The best performing models from the *Held-out Data* experiment, i.e., the identified best-performing combination of feature extractors and classifiers, were used in this experiment.

Figure 4.3 illustrates the employed training and evaluation protocol where the models are trained on the full COICOP training data set and evaluated on the Scanned Receipts Test Set.

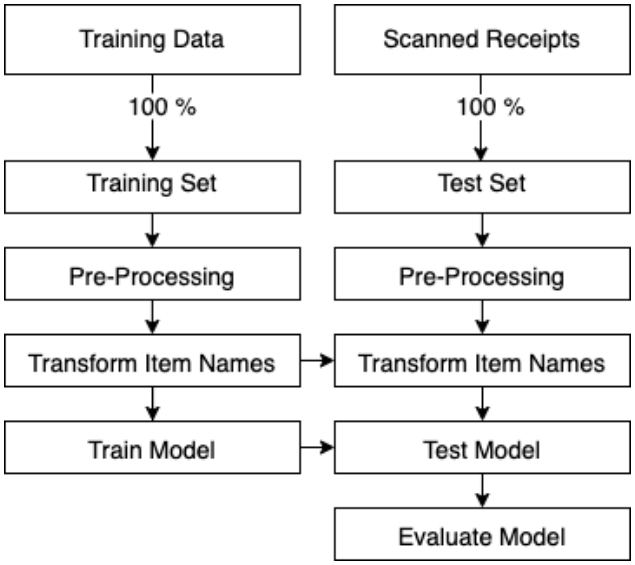


Figure 4.3: Scanned Receipts: Training and evaluation protocol

The item names in both the training and set were pre-processed by applying the steps described in subsection 4.1.1. Similar to how the feature extractor would only learn to represent item names in the training partition in the *Held-out Data* experiment, the feature extractor in this experiment learned to represent the items in the full COICOP training data. The fitted feature extractor was subsequently used to transform both the item names in the full COICOP training data set and the items in the Scanned Receipts Test Set to numeric feature vectors.

After transforming the item names, the models were trained on the data in the vectorised COICOP training data set and evaluated on the vectorised Scanned Receipts Test Set. The model performances from this evaluation will be presented using the performance metrics described in subsection 4.2.2.

4.3 Automatic Classification System

To evaluate the potential for implementing an automatic classification system into the statistics production for Statistics Norway, a classification system suited for automated COICOP classification must first be identified. By relying on machine learning to automate the classification process of items in the survey, mistakes are bound to happen as there will always be items that the classifier has never seen before or items that the classifier is conflicted on. However, manually reviewing each classification to make sure that no misclassifications occur, is hardly different from manually labelling each item. Instead, employing a system that is able to signal when mistakes potentially occur, so that only these predictions are manually reviewed, may be a pertinent compromise.

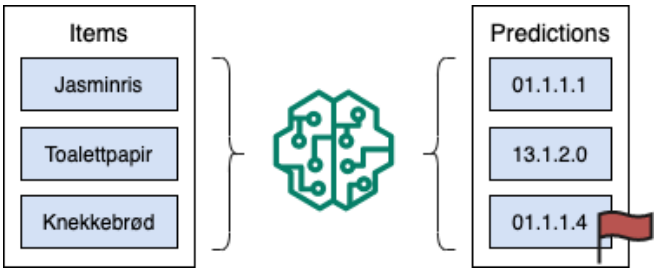


Figure 4.4: Predictions for items where uncertain classifications are flagged

Figure 4.4 illustrates the envisioned concepts of this system. Here, predictions are made for each item, but the prediction for "Knekkebrød" is flagged as the classifier believes that a mistake may have occurred. The two non-flagged predictions are automatically handled, while the flagged prediction is the only one that requires human attention. In order to identify a suitable approach to such a system, this thesis consults related work from different national statistics offices (NSOs).

4.3.1 Related Work

The amount of previous work related to machine learning for automatic 5-digit COICOP classification of items appears to be limited. Among the identified studies, most studies typically relate to the application of machine learning for production of the Consumer Price Index, while there is also one that directly relates to classification of items from household budget surveys. Among these, two studies appear to be particularly relevant for the classification system envisioned in this thesis, and the findings that were relevant to this study are summarised in the following paragraphs.

Machine Learning in the Consumer Price Index

Myklatun (2019) explored the potential and benefits of utilising machine learning for the classification of items from transaction data for the production of the Consumer Price Index (CPI) for Statistics Norway. The motivation behind this was to increase the coverage and accuracy of the CPI estimations and to decrease the labour burden involved with manually labelling new items every month. Myklatun states that in

the market of food and non-alcoholic beverages, there is an estimated 400-1200 new items introduced every month, and whereas previously seen items could sometimes be automatically classified using rule-based classifications, all these new items would still have to be manually labelled.

Myklatun explored the use of machine learning for the previously unseen items and proposed the use of *prediction probability scores* for each item to determine which items should be flagged for manual labelling. Using this approach, Myklatun found that for predictions with prediction probabilities above 20%, the overall accuracy of the predictions was 95%. With this approach, Statistics Norway were able to reduce the amount of manual labour required in the CPI productions significantly at the cost of an estimated 5 % incorrect classifications.

Classification of Shopping Receipts

In a collaboration between the national statistics offices in UK and Netherlands, Benedikt et al. (2020) explored the use of machine learning to modernise parts of the household budget survey by automatically classifying items from receipts. Similar to Statistics Norway's survey of consumer expenditure, this household budget survey utilised 5-digit COICOP subclass codes in the classification of items. Because of this detailed coding framework, Benedikt et al. found that fully automating the classifications poses many challenges. Many receipts contain short item names such as "fresh milk". When using the detailed 5-digit COICOP 2018 structure to classify milk, one needs information about the type of milk to make necessary distinctions between milk types such as "whole milk" or "skimmed milk". Benedikt et al. argued that because of this, the classifier will ultimately make mistakes.

In order to reduce the number of mistakes made, Benedikt et al. proposed the use of a "human-in-the-loop" classification system. This system also relies on *prediction probability scores* to determine when human intervention is required to manually review predictions. Furthermore, through manually reviewing items, more labelled data is generated which can be used to enrich the training data to improve model performance in the future. This process is referred to as *Active Learning*. Figure 4.5 illustrates the proposed classification system. Here, "Confident" refers to the prediction probability of a prediction.

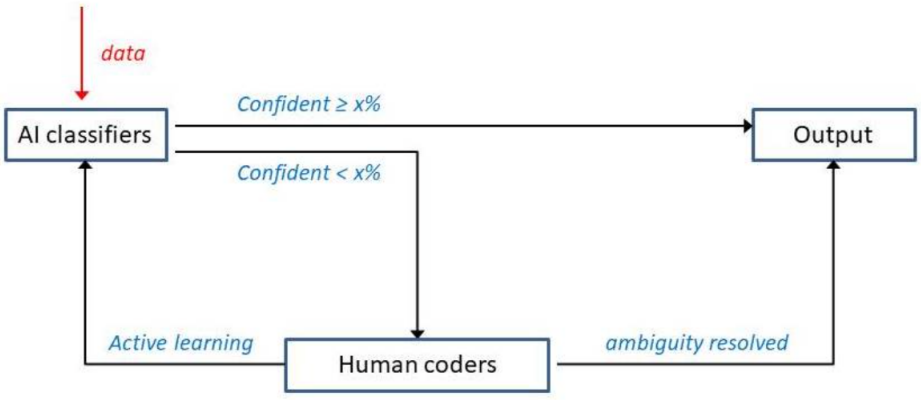


Figure 4.5: "Human-in-the-Loop" classification of items (Benedikt et al., 2020)

Although Benedikt et al. did not report results from implementing this system into real applications of statistics production, the preliminary test showed promising results, where some of the best classifiers showed the potential for automatically classifying 62% of the items at the cost of 3% incorrect classifications.

4.3.2 Proposed System for Automatic Classification

Based on the outlined classification systems in the related works, prediction probabilities appear to be a suited metric for signalling when misclassifications are expected to occur. Furthermore, with the addition of Active Learning in a "human-in-the-loop"-based classification system as Benedikt et al. explored, there is a potential for continuous model improvements by giving direct feedback to items that the model has never seen before or are conflicted on.

Because of the auspicious results from the studies of Myklatun and Benedikt et al., the models explored in this thesis will also facilitate a similar classification system in the future. In this way, Statistics Norway can decide on the appropriate threshold value that determines which portion of the data that needs manual review and which that should be automatically classified. Figure 4.6 illustrates this concept with an arbitrary distribution of prediction probabilities for items and separates these items into manual review or automatic classification based on which side of the threshold value, $T = 80$ %, they fall.

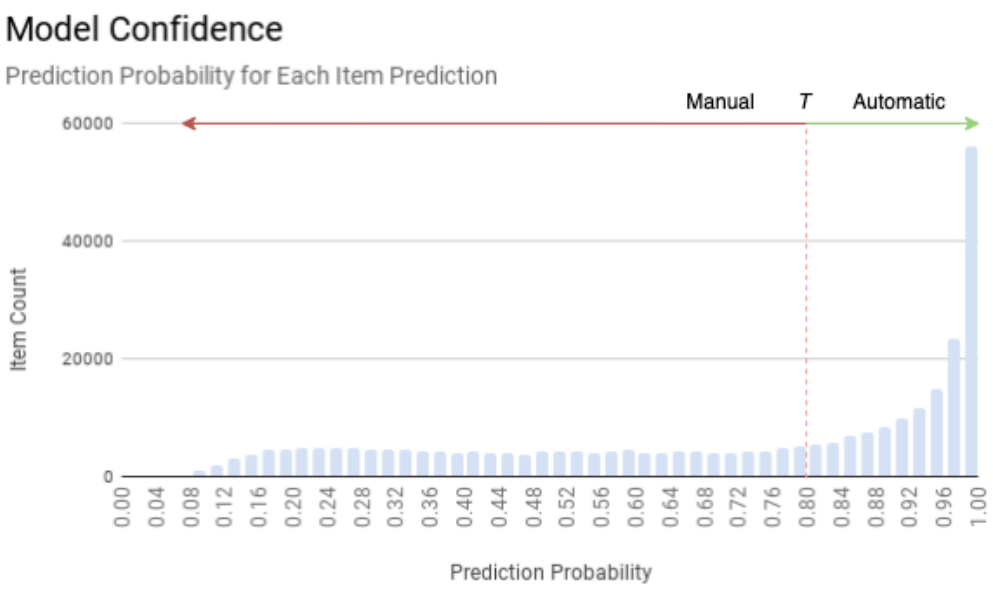


Figure 4.6: Prediction probabilities partitioned by threshold value T

Figure 4.7 expands on the example from figure 4.4 and uses both prediction probabilities and threshold value, $T = 80\%$, to determine if an item should be manually reviewed or be automatically classified. Here, "Knekkebrød" falls below the predefined threshold value and is flagged for manual review. After assigning the correct COICOP code to this item, the classifications are finalised, and the labelled item is added to the training data.

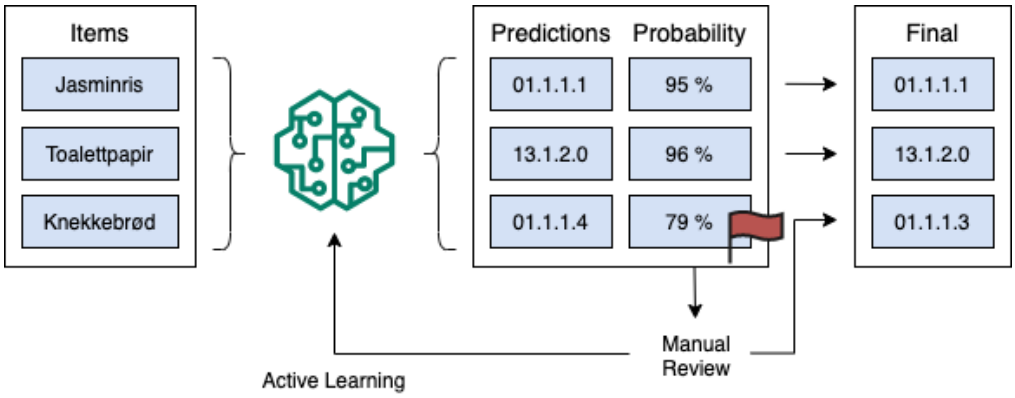


Figure 4.7: "Human-in-the-loop"-based classification system for items from scanned receipts

In the evaluation of the current potential and limitations for implementing an automated classification system for items from scanned receipts, this thesis has based its assessments on whether the explored and evaluated models are fit-for-purpose for a similar "human-in-the-loop"-based classification system.

4.3.3 Evaluation Potential for Automatic Classification

To explore the potential for automatic classification based on the described classification system in subsection 4.3.2, the frequencies of the prediction probabilities for predictions both on the held-out test set and the scanned receipt data were explored in detail. The number of misclassifications that occurred above specified threshold values were calculated to illustrate how many errors that would happen if all predictions above the specified prediction probability threshold were automatically classified instead of manually reviewed by a human. Figure 4.8 illustrates how the threshold value segments the data and how the error rates for predictions above the threshold are calculated.

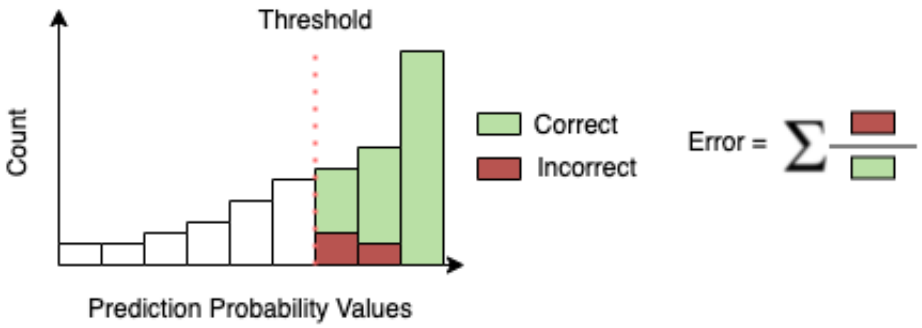


Figure 4.8: Calculating rate of error above a specified threshold value

Chapter 5

Model Results

This chapter presents results from the training and evaluation of the COICOP classifiers on the assembled training data, previously described in subsection 3.2.4. Section 5.1 presents the best performing feature extractor method for each classifier that will be used in evaluations in the subsequent sections of this chapter. Section 5.2 presents the performance results of classifiers on the training data where each classifier was evaluated on a hold-out test set from the training data. Section 5.3 presents results that show how well the previously trained classifiers carry over to the Scanned Receipts Test Set, previously described in subsection 3.2.5. In this chapter, results related to research questions 2 and 3 are presented, while the next chapter further interprets these results.

5.1 Model Selection

In the model selection process, all classifiers were tested with the different combinations of *feature extractors*, *analyser* and *N-gram range*, as listed in table 4.3. The best performing feature extraction method for each classifier was identified by evaluating performance on a subset of 10% of the full training data. Both classifiers, listed in table 4.5, were implemented using their default values with the Scikit-learn library. Table 5.1 shows the best performing models on the subset of the training data, while performance scores for every combination of feature extractor methods and classifiers are included in Appendix B.1.

Classifier	Feature Extractor	Analyser	N-gram Range
Logistic Regression	CountVectorizer	Character	(3,3)
Random Forest	CountVectorizer	Character	(2,3)

Table 5.1: Classifier Models: Chosen feature extractor, analyser and N-gram range for both classifiers

The classifier models of table 5.1 are both used in the evaluation of model performances on held-out test data from the full training data set in section 5.2 and in evaluations on items from scanned receipt data in section 5.3.

5.2 Model Performances on Held-out Data

5.2.1 Model Predictions

Following the described steps of the *Held-out Data* experiment in subsection 4.2.3, the Logistic Regression and Random Forest classifier models from table 5.1 were evaluated on the full training data set using a 20% hold-out set as a test set. Table 5.2 shows the performance of the Logistic Regression and Random Forest models on the training and test partition of the full training data set. The highlighted cells show the highest performance metric scores on the hold-out test set. Here, the model performances on the training set are also presented to illustrate how well the classifiers were able to fit to the training data. The differences between performance scores on the training set and the test set can indicate whether the models were overfitting to the training set.

Classifier Model		Performance Metrics			
Classifier	Extractor	Precision	Recall	F1-Score	Accuracy
Training Set (80%)					
Logistic Regression	CV-ch33	0.845	0.844	0.844	0.844
Random Forest	CV-ch23	0.984	0.984	0.984	0.984
Hold-out Set (20%)					
Logistic Regression	CV-ch33	0.820	0.820	0.819	0.819
Random Forest	CV-ch23	**0.870**	**0.869**	**0.868**	**0.869**

Table 5.2: Performance of classifier models on training and test partitions of the training data set

The results from table 5.2 indicate that both models were able to learn patterns in the training data, where the Random Forest model was able to achieve high performance scores on the training partition of the training set. However, when assessing its performance on the hold-out set, more than a 10 percentage points drop in performance is observed for all performance metrics. The Logistic Regression model did not fit as well to the training data as the Random Forest model, achieving an accuracy score of 84.44%, which is 14 percentage points lower. Still, the drop-off in performance on the hold-out set was much lower for the Logistic Regression model, indicating that this model might have been able to reach a better comprise between bias and variance than the Random Forest model. Regardless, Random Forest still proved to be the overall best performing model of the two on the hold-out test set.

Table 5.3 breaks down the overall accuracy into average accuracy by division category, showing that the Random Forest model also achieved the highest average accuracy for all division categories in the hold-out set. The highlighted cells show the highest average accuracy scores for samples within each division category. For a full description what the different division codes represent, see section 3.3.

Hold-out Set		Classifier Models	
Code	Items	Logistic Regression	Random Forest
01	31 553	0.786	**0.842**
02	3 934	0.728	**0.764**
03	42 888	0.855	**0.893**
04	24 429	0.829	**0.906**
05	80 338	0.800	**0.863**
06	4 102	0.814	**0.839**
07	21 166	0.820	**0.865**
08	21 092	0.866	**0.899**
09	37 560	0.752	**0.798**
10	1	**1.000**	**1.000**
11	20	0.850	**0.859**
12	2	0.000	0.000
13	30 774	0.907	**0.939**

Table 5.3: Average model accuracy on samples within each COICOP division code in the held-out test set

Based on the results in table 5.2 and table 5.3, the Random Forest model appears to be the overall best performing model on the COICOP training data. Still, several misclassifications occurred in the hold-out set for both models. Figure 5.1 shows the 25 most frequently misclassified subclass codes for the Random Forest model. In total, this model made 38 949 misclassifications, and the 25 most frequently misclassified subclass codes represent 69% of all misclassifications made.

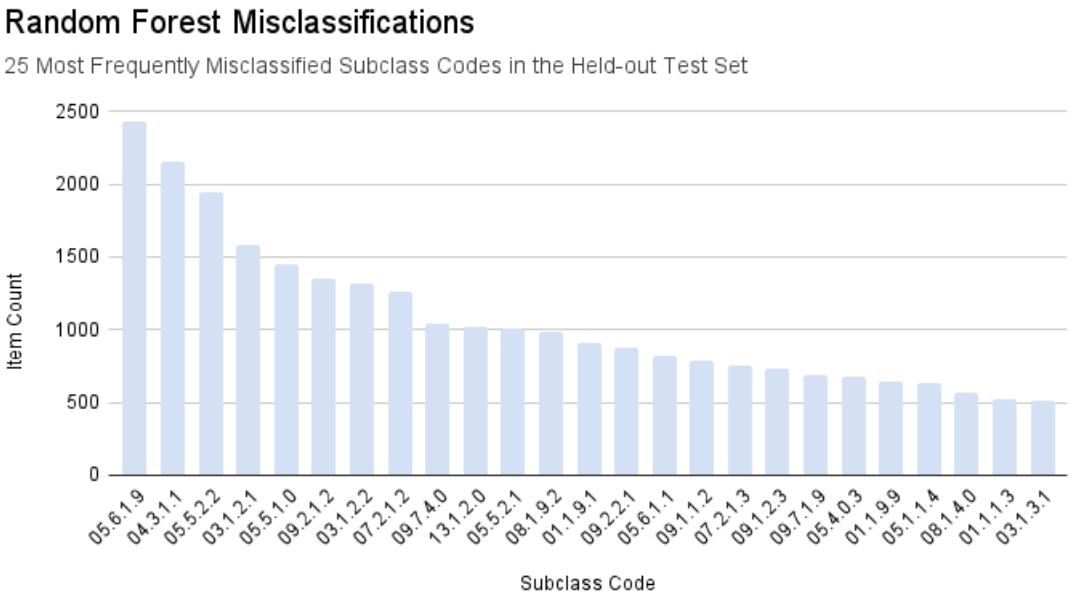


Figure 5.1: Most frequently misclassified subclass codes in the held-out test set by the Random Forest model

Similarly, figure 5.2 shows the 25 most frequently misclassified subclass codes for the Logistic Regression model. This model made 53 840 misclassifications in total, and the 25 most frequently misclassified subclass codes represent 73% of all misclassifications made.

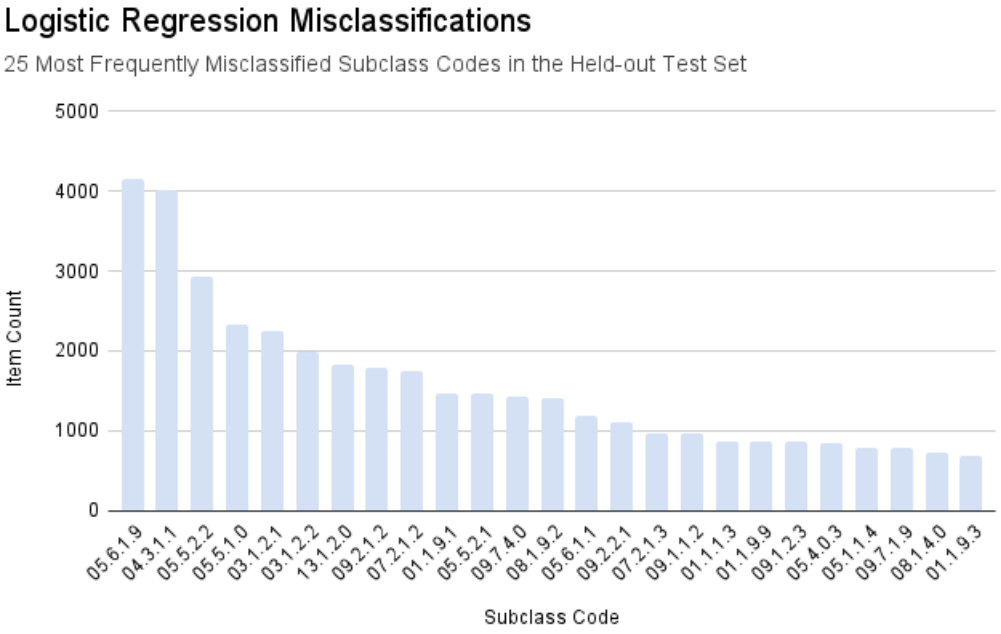


Figure 5.2: Most frequently misclassified subclass codes in the held-out test set by the Logistic Regression model

Figures 5.1 and 5.2 indicate that both models typically misclassify similar subclass codes. However, notably fewer misclassifications were made by the Random Forest model.

5.2.2 Distribution of Prediction Probabilities

Misclassifications on the unseen testing data are to be expected as there will be item names that the models have never seen before. Ideally, this should also be reflected in the model's prediction probabilities, where one can get an indication of which item names that are unknown or inconclusive to the models and where misclassifications are likely to occur. The following paragraphs show the distribution of prediction probabilities for the Logistic Regression model and the Random Forest model, as well as the error rates at different threshold values.

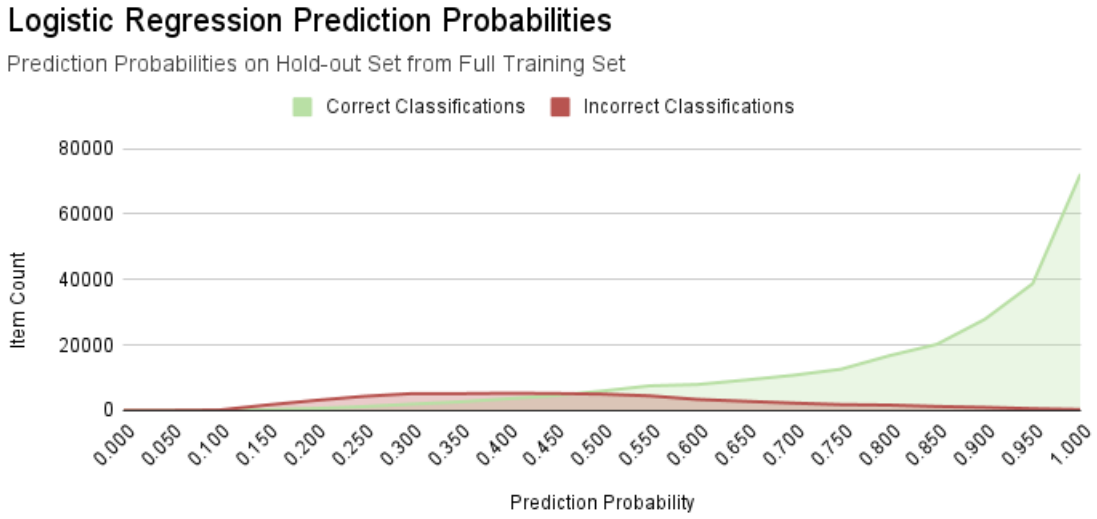


Figure 5.3: Distribution of prediction probabilities for Logistic Regression predictions on the held-out test set

Figure 5.3 illustrates the distribution of prediction probabilities by correct and incorrect classification of samples for the Logistic Regression model. In this visualisation, the prediction probabilities have been bucketed with a bucket size of 0.05, meaning that the graph might be less detailed than the actual distribution. The left-skewed distribution in the figure indicates that the model was confident in most of its predictions. Furthermore, among the predictions that have a high prediction probability value, most were in fact correct classifications. Still, figure 5.3 shows that incorrect predictions also occurred for higher prediction probability values, but they were less frequent than correct predictions when the prediction probability is above 50%. Table 5.4 shows the number of items classified with a prediction probability above specified threshold values, T, and the error rates, ER, for these predictions.

Count	$T \geq 95\%$	$T \geq 90\%$	$T \geq 85\%$	$T \geq 80\%$
Number of Items	72 469	111 686	140 433	161 830
Of Total Data	24.33%	37.50%	47.15%	54.33%
Error Rate (ER)	0.00	0.01	0.01	0.02

Table 5.4: Number of items above threshold value (T) for predictions made by the Logistic Regression model on the held-out test set

As shown in table 5.4, 54.33% of the total data was classified with a prediction probability above 80% ($T \geq 80\%$). Among these predictions, 2% were misclassifications. Alternatively, table 5.5 shows how much of the data that can be used with specified error rates. Table 5.5 shows that 73.70% of the data could be automatically handled by the classifier at the cost of 5% misclassifications - if the threshold value were to define which partition of the data that would be automatically classified without the need for manual review.

Count	$ER = 0.00$	$ER = 0.01$	$ER = 0.03$	$ER = 0.05$
Number of Items	22 448	135 271	191 879	219 515
Of Total Data	7.54%	45.41%	64.42%	73.70%
Threshold (T)	$T \geq 99\%$	$T \geq 86\%$	$T \geq 71\%$	$T \geq 60\%$

Table 5.5: Number of items and threshold value (T) for a specified error rate (ER) for predictions made by the Logistic Regression model on the held-out test set

Similarly, the distribution of prediction probabilities for the Random Forest model predictions on the hold-out set is shown in figure 5.4. This graph indicates that the Random Forest model is generally confident in its predictions, as most predictions are concentrated above 95% prediction probability.

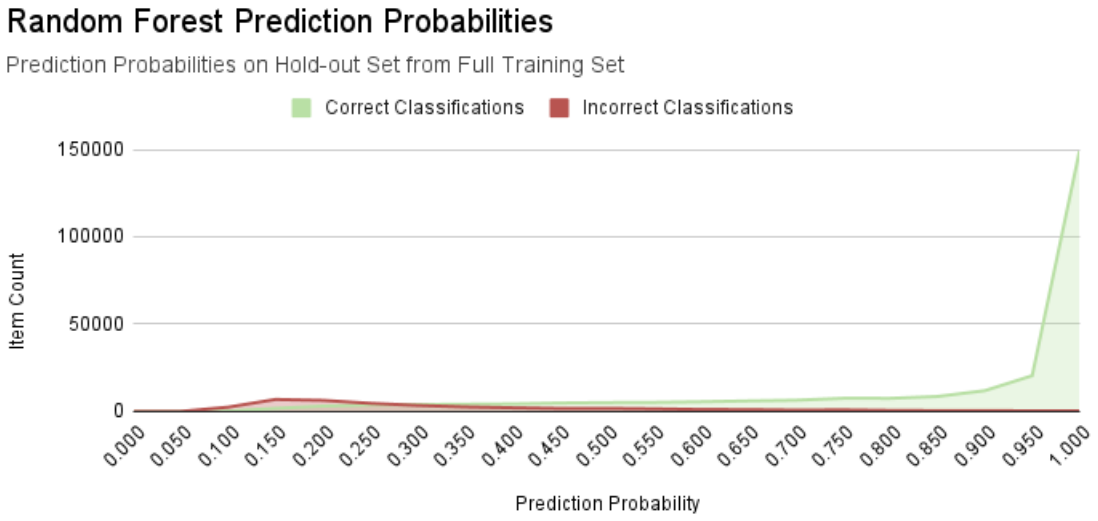


Figure 5.4: Distribution of prediction probabilities for Random Forest predictions on the held-out test set

The confidence of the Random Forest model is also reflected in table 5.6, which shows that the prediction probabilities are $\geq 95\%$ for 57.10% of the data predicted.

Count	$T \geq 95\%$	$T \geq 90\%$	$T \geq 85\%$	$T \geq 80\%$
Number of Items	170 081	182 401	191 384	199 423
Of Total Data	57.10%	61.24%	64.25%	66.95%
Error Rate (ER)	0.00	0.01	0.01	0.01

Table 5.6: Number of items above threshold value (T) for predictions made by the Random Forest model on the held-out test set

Table 5.7 shows how much of the data that can be used with specified error rates. The table shows that 85.64% of the data could be automatically classified if an error rate of 5% is tolerated. Additionally, the table also shows that the prediction probabilities for 51.03% of all predictions were $\geq 99\%$.

Count	$ER = 0.00$	$ER = 0.01$	$ER = 0.03$	$ER = 0.05$
Number of Items	152 002	199 423	235 191	255 092
Of Total Data	51.03%	66.95%	74.51%	85.64%
Threshold (T)	$T \geq 99\%$	$T \geq 80\%$	$T \geq 55\%$	$T \geq 39\%$

Table 5.7: Number of items and threshold value (T) for a specified error rate (ER) for predictions made by the Random Forest model on the held-out test set

As the previous paragraphs have shown, both classifiers typically make correct predictions when the prediction probability value is high. Although the Random Forest model indicates a better separation of misclassifications and correct classifications than the Logistic Regression model. However, it should be noted that the two models calculate prediction probabilities differently.

As previously described in subsection 2.2.1, Logistic Regression transforms the net input sum z of each sample with the sigmoid activation function, $\phi(z)$, to obtain a probability value between 0 and 1. This value is used by the Logistic Regression model as its prediction probability. Random Forest, on the other hand, is a collection of Decision Trees. Each Decision Tree returns a decision; 1 for the positive class and 0 for the negative class. The Random Forest counts the number of votes of all the Decision Trees and predicts the majority class. The prediction probability is calculated as the number of votes for this class divided by the number of trees in the model (Pedregosa et al., 2011).

5.3 Model Performances on Scanned Receipts

The preceding section showed that the Logistic Regression and Random Forest models were both able to achieve performance scores above 80% when testing on held-out test data. The results presented in this section summarise how well the performances of these models carry over to predictions on unseen items from Scanned Receipt Data, acquired through manual labelling of items from the pilot study (see subsection 3.2.5).

5.3.1 Model Predictions

Following the described steps of the *Scanned Receipts* experiment in subsection 4.2.3, both models were retrained on every sample in the training data set without partitioning the data into a training and hold-out set. After both models were fitted to the full training data set, each model attempted to predict the subclass code of the items in the Scanned Receipts Test Set. Table 5.8 shows the performance of both models on the Scanned Receipts Test Set, where the highlighted cells show the highest scoring performance metric.

Classifier Model		Performance Metrics			
Classifier	Extractor	Precision	Recall	F1-Score	Accuracy
Logistic Regression	CV-ch33	0.568	0.485	0.489	0.485
Random Forest	CV-ch23	**0.613**	**0.526**	**0.530**	**0.526**

Table 5.8: Performance of classifier models on Scanned Receipts Test Set

Table 5.8 shows that neither model was able to reproduce its performance from the *Held-out Data* experiment on the Scanned Receipts Test Set. The models achieved similar scores for most performance metrics, where only a few percentage points separates the models. Although the scores are similar, the Random Forest model achieved slightly higher scores.

Table 5.9 breaks down the overall accuracy of each model into average accuracy by division category, showing that the Random Forest model achieves the highest average accuracy for samples within most division categories. However, the Logistic Regression model is slightly ahead or on par with the Random Forest model for predictions on samples within four of the division categories. For a full description what the different division codes represent, see section 3.3.

Test Set		Classifier Models	
Code	Items	Logistic Regression	Random Forest
01	641	0.569	**0.621**
02	29	0.517	**0.586**
03	35	0.257	**0.314**
04	8	**0.125**	**0.125**
05	54	0.370	**0.426**
06	22	0.000	**0.136**
07	13	**0.385**	**0.385**
08	5	**0.200**	0.000
09	56	**0.304**	0.214
10	1	0.000	0.000
11	39	0.000	0.000
12	2	0.000	0.000
13	42	0.619	**0.667**

Table 5.9: Average model accuracy on samples within each COICOP division code in the Scanned Receipts Test Set

As shown in table 5.9, there is variation in the average accuracy scores for the different division categories. Both models were generally able to achieve higher accuracy scores for division category 13, and to some extent for categories 01 and 02. Still, the models appear to have struggled with classifications on samples within the other division categories, and for samples within division category 10, 11 and 12, no model was able to make a single correct prediction. Additionally, for samples within division category 08, the Random Forest model was not able to make any correct prediction.

Figure 5.5 illustrates the most frequently misclassified subclass codes in the Scanned Receipts Test Set by the Logistic Regression model. A total of 488 out of 947 samples in the Scanned Receipts Test Set were misclassified. Among these misclassifications, most samples correspond to food items (division code 01), while the overall most frequently misclassified subclass code was 11.1.1.2. The misclassifications included in figure 5.5 show 64.73% of all misclassifications made on the Scanned Receipts Test Set by the Logistic Regression model.

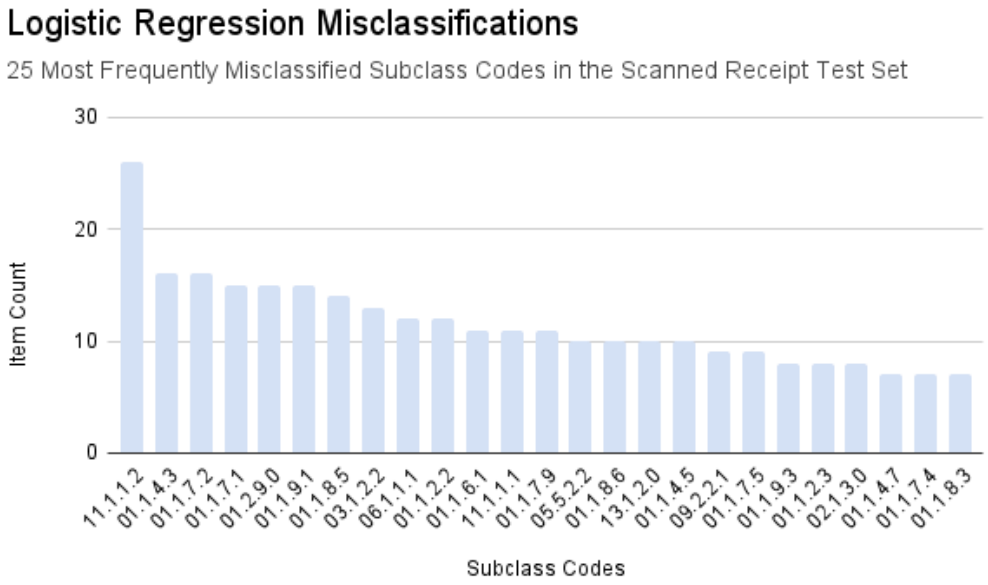


Figure 5.5: Most frequently misclassified subclass codes by frequency in the Scanned Receipts Test Set by the Logistic Regression model

Similarly, figure 5.6 shows the 25 most misclassified subclass codes by the Random Forest model. A total of 449 of the 947 samples in the Scanned Receipts Test Set were misclassified by this model. Like the Logistic Regression model, the Random Forest model was unable to correctly classify any sample belonging to division category 11, and consequently, subclass code 11.1.1.2 was also the overall most frequently misclassified subclass code by the Random Forest model. The misclassifications included in figure 5.5 shows 58.80% of all misclassifications made on the Scanned Receipts Test Set by the Random Forest model.

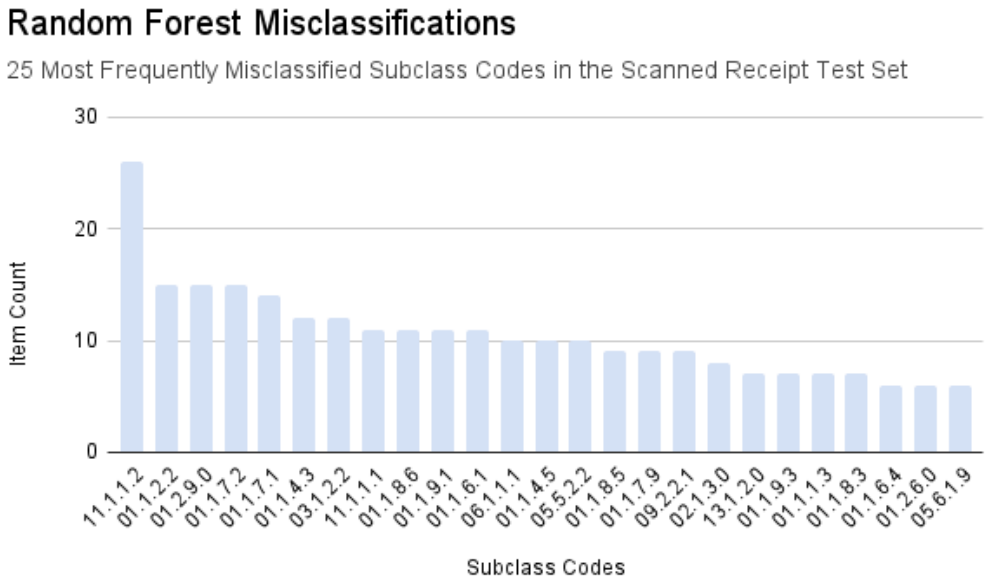


Figure 5.6: Most frequently misclassified subclass codes by frequency in the Scanned Receipts Test Set by the Random Forest model

5.3.2 Distribution of Prediction Probabilities

To get an indication of how well suited the classifier models that were trained on the assembled COICOP training data are for a "human-in-the-loop"-based system, the following paragraphs explore the prediction probabilities of each prediction for both models.

Figure 5.7 shows the distribution of prediction probabilities for the Logistic Regression model. Again, for this visualisation, the prediction probabilities have been bucketed with a bucket size of 0.05, meaning that the graph might be less detailed than the actual distribution.

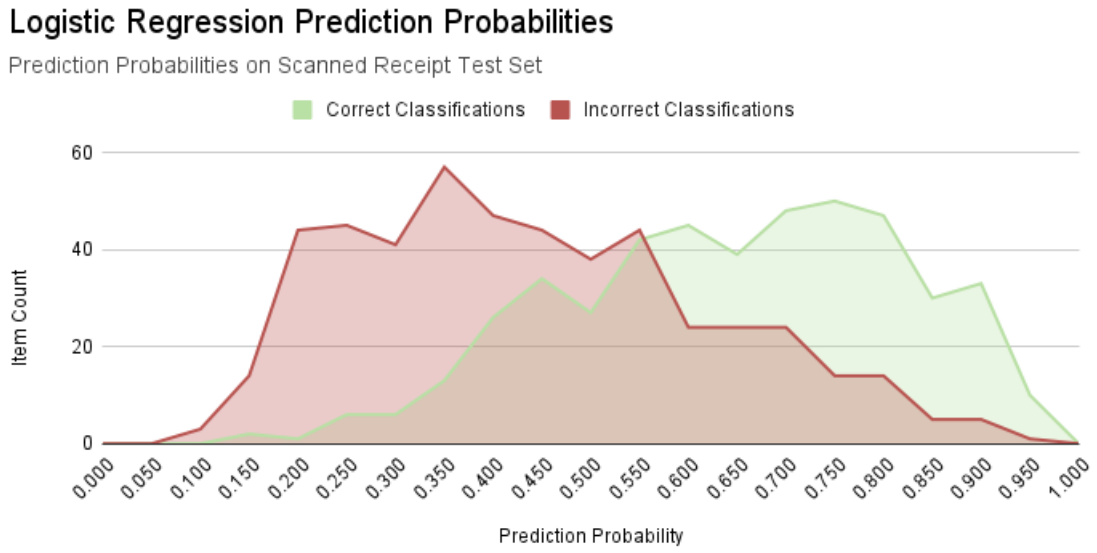


Figure 5.7: Distribution of prediction probabilities for Logistic Regression predictions on the Scanned Receipts Test Set

The graph in figure 5.7 shows a much less skewed distribution than what was previously observed on the held-out test data in figure 5.3. This graph indicates that for predictions with prediction probability values up to 55%, most predictions were in fact misclassifications. In contrast to the distribution of prediction probabilities on the held-out test data, there is a notable overlap between misclassifications and correct classifications, also for the higher prediction probability values. Table 5.10 shows the number of items classified with a prediction probability above specified threshold values, T, and the error rates, ER, for these predictions.

Count	$T \geq 95\%$	$T \geq 90\%$	$T \geq 85\%$	$T \geq 80\%$
Number of Items	11	49	84	145
Of Total Data	1.16%	5.17%	8.87%	15.31%
Error Rate (ER)	0.09	0.12	0.13	0.17

Table 5.10: Number of items above threshold value (T) and the error rate (ER) for predictions made by the Logistic Regression model on the Scanned Receipt Test Set

Table 5.10 shows that the Logistic Regression model was less confident in its predictions, as considerably fewer predictions had high prediction probability scores. 15.31% of the predictions had a prediction probability of 80% or higher. However, at the same time, only 11 predictions in total had a prediction probability of 95% or higher. Additionally, the error rates were higher than they were for similar thresholds on prediction on the held-out test set. Using a threshold for 80% prediction probability, 17% of the predictions were incorrect.

Similarly, figure 5.8 shows the distribution of prediction probabilities for the Random Forest model. In this graph, there is a clearer separation of misclassifications and correct classifications than for the Logistic Regression model in figure 5.7. This graph shows that most misclassifications are concentrated on the lower prediction probability values, while the opposite appears to be the case for most correct classifications.

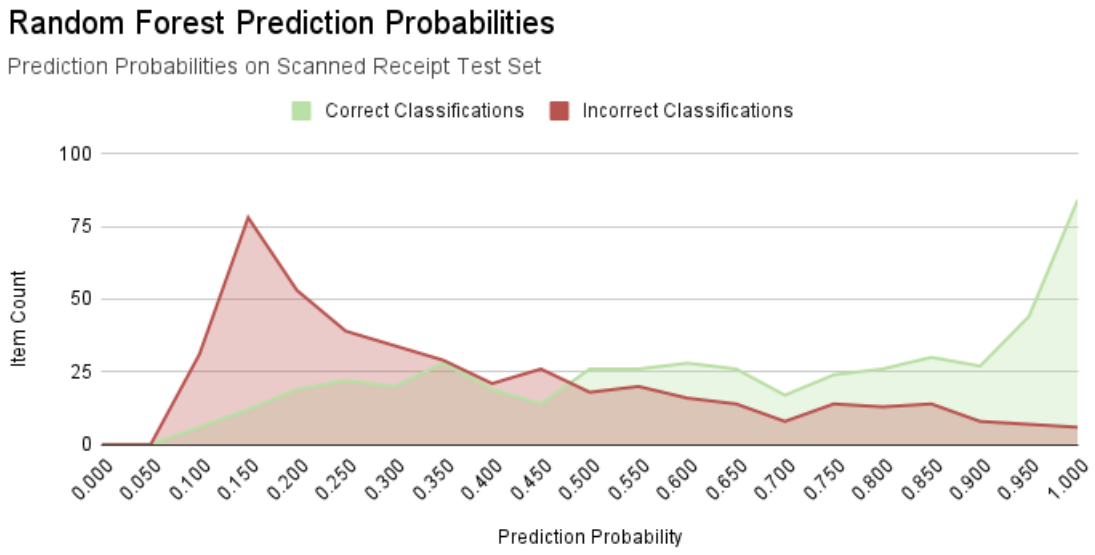


Figure 5.8: Distribution of prediction probabilities for Random Forest predictions on the Scanned Receipts Test Set

Table 5.11 presents the number of predictions and error rates located above specified thresholds. This table shows that 220 of the predictions have a prediction probability of 80% or higher, and this makes up 23.23% of the total Scanned Receipts Test Set. However, a total of 16% of these are misclassifications. The results in table 5.11 show a generally lower error rate for predictions by the Random Forest model with higher prediction probabilities than table 5.10 showed for the Logistic Regression model. Still, figures 5.7 and 5.8 show that there are overlapping misclassifications and correct classifications for predictions at all prediction probability values for both models.

Count	$T \geq 95\%$	$T \geq 90\%$	$T \geq 85\%$	$T \geq 80\%$
Number of Items	90	141	176	220
Of Total Data	9.50%	14.89%	18.59%	23.23%
Error Rate (ER)	0.07	0.09	0.12	0.16

Table 5.11: Number of items above threshold value (T) and the error rate (ER) for predictions made by the Random Forest model on the Scanned Receipt Test Set

Chapter 6

Discussion

In this study, the concept of utilising machine learning to classify consumer goods into 5-digits COICOP 2018 subclass codes has been explored. This includes ways in which to prepare and combine data sets from auxiliary data sources to acquire a training data set of item names and COICOP codes, application of traditional COICOP classifiers for learning patterns in the training data, and evaluation of the trained COICOP classifiers performance on items from scanned receipt data.

This chapter is split into four sections, where section 6.1 evaluates the findings of the preceding chapters and summarises how these findings relate to the different objectives of this thesis. Section 6.2 outlines some of the limitations of this study, as well as some of the limiting factors that have influenced the scope of this thesis. Section 6.3 compares the findings of this thesis to findings in related studies. Lastly, section 6.4 presents some recommendations for future work. This section also describes how further development can be done to improve model performance, as well as some suggestions for how Statistics Norway can proceed in their development of an automatic classification system based on the findings in this study.

6.1 Evaluation of Results and Thesis Objectives

The primary results in this thesis relate to both the acquisition of COICOP training data and to the training and evaluation of COICOP classifiers. Chapter 3 described the involved steps and outcome of the data acquisition, while chapter 5 presented the results related to classifier performances. This section aims to evaluate these findings and to put them in the context of the main objectives of this thesis.

6.1.1 Evaluation of Objectives

The following paragraphs list the different research questions from section 1.3 and summarise the main findings related to each objective.

RQ1: How can data from auxiliary data sources be combined to assemble a COICOP training data set for training and developing a COICOP classifier model?

Chapter 3 of this thesis described the performed steps in acquiring a COICOP training data set. The findings in this chapter showed that data from auxiliary data

sources at Statistics Norway *can* be combined into a COICOP training data set of *Item Names* and *5-digit COICOP subclass codes*. Through high-level filtering operations and transformation of coding formats, a single data set was obtained from 5 distinct data sources.

Out of the different data sets, the Imports data set required the most preparation work in order to be suited as COICOP training data. Through the use of Eurostat's conversion tables combined with a custom-made search algorithm, almost half of the entries in the original Imports data set (excluding duplicates) were successfully converted into item names associated with 5-digit COICOP subclass codes. The Imports data set formed a significant part of the final combined training data set, resulting in 96.22 % of all training samples. In total, the combined training data set contains 283 unique subclass codes with 1 490 216 entries of item names.

However, as the assembled COICOP training data set is the result of five individual data sets, the sources of these data sets have largely determined the subclass codes represented in them. For example, one would not find many instances of subclass codes corresponding to *travel insurance* by considering a data set generated from sales at Rema 1000. Therefore, not all subclass codes are present in the assembled training data set, where 15 codes are missing and 100 (35 %) are represented by 15 or fewer samples.

Despite these shortcomings, the assembled COICOP training data set still provides valuable utility for the objective of this thesis by successfully facilitating training and evaluation of COICOP classifiers. Furthermore, the assembled training data set can also contribute to further work and research related to item classification by providing a prepared COICOP training data set. Additionally, with the description of the steps involved with the preparation of this data set, this study also provides a design for how data from the different data sources can be prepared and combined into a COICOP training data set.

RQ2: How well do traditional classification models perform on the COICOP training data?

Chapter 4 described the choices that went into the design of the classifier models. As the stated purpose of this study has been to evaluate the potential of implementing machine learning for automatic COICOP classification of consumer goods, a set of out-of-the-box classifiers were implemented using the Scikit-learn library to explore whether COICOP classification based on item names is feasible. Despite their simple structure, the results in section 5.2 showed that the employed classifier models were able to learn most patterns in the training data and to recognise prominent features in unseen held-out test data. Additionally, the prediction probabilities of the models appeared to be good indicators of where misclassifications typically occur, and this indicates that there should be a potential for implementing a "human-in-the-loop" integration with the COICOP classification system.

Ultimately, these findings show that COICOP classification can successfully be done using quite basic, but easy to implement, classification models in combination with

simple text representation methods. Based on these observations, it seems reasonable to assume that further tweaking and optimising of the models' design, and perhaps with the implementation of more advanced text representation methods, the performances of the models would only further increase.

RQ3: How well do the performances of the trained COICOP classifiers carry over to unseen samples of scanned receipt data?

A test data set of scanned receipt items from the pilot study (see subsection 1.2.3) was acquired through manual labelling. The aim of this was to quantify how well the previously trained COICOP classifiers would perform if they were immediately implemented for classification of items from the survey of consumer expenditure. The results in this study found that there was a noticeable drop-off in performance when predicting the 5-digit COICOP subclass code for these items, where each model was only able to correctly classify around half of the samples. Additionally, the results showed that the average prediction accuracy for samples within most division categories was typically quite low.

It should be noted that there are differences between the scanned receipt data set that was used in this evaluation and the assembled COICOP training set that was used to train the classifier models. As these data sets do not contain samples from the same population, some misclassifications are expected to occur. A portion of the misclassifications appears to be the result of systematic errors that are likely caused by a lack of representative training data, as the models struggled to predict most of the subclasses they had barely seen before, while they typically performed better on more familiar subclass codes.

However, in contrast to the findings on the held-out test data in RQ2, the prediction probability values of each prediction on the Scanned Receipts Test Set do not appear to be clear indicators of where misclassifications occur. Figure 5.4 showed that despite high prediction probability, misclassifications happened quite often. This subsequently limits the current benefits of implementing a "human-in-the-loop"-based classification system for automatic classifications, as all predictions will have to be manually reviewed by a human when the prediction probability values of the model cannot be trusted. However, by reviewing each classification, model performance may quickly improve through the benefits of active learning, and subsequently, the prediction probabilities will likely become more dependable. This idea is further discussed in section 6.4.

RQ4: What are some of the current limiting factors that prevent Statistics Norway from implementing automated classification of items from scanned receipts?

This thesis' ability to discuss current limitations at Statistics Norway is limited by the materials and the methods that were used. This means that in this assessment, it is assumed that the Scanned Receipts Test Set provides a precise indicator of expected item purchases to occur in future surveys and that they are correctly labelled by the coder that performed the labelling. Furthermore, it is also assumed that Statistics Norway are limited to the data sets that have been used in this thesis.

The following paragraphs explore where some of the more prominent misclassifications occur and attempt to point out some likely reasons as to why they happen.

Perhaps the most obvious source of error is the subclass codes in the Scanned Receipts Test Set that the models have barely seen, if at all, during training. Table 6.1 lists instances of subclass codes that are present in the Scanned Receipts Test Set, and that are represented by only 5 or fewer samples in the training data set. For these samples, the models have likely not been able to learn any patterns that could be applied to the unseen samples. Consequently, no model was able to correctly predict these subclass codes.

Subclass Code	Items in Test Data	Items in Training Data
02.1.2.2	2	5
04.4.1.1	1	0
04.4.3.1	2	0
07.3.2.2	1	5
08.3.4.0	2	5
12.1.3.0	1	1
12.2.2.0	1	4
13.3.0.9	1	2

Table 6.1: COICOP subclass codes with low representation in the Training Data Set

As table 6.1 shows, these subclass codes do not occur often in the Scanned Receipts Test Set either. Thus, they have not made a large impact on the overall performance scores of the models in the evaluation. Nevertheless, these findings still outline an important limitation for future predictions of these particular subclass codes, as the trained COICOP classifiers will most likely never be able to correctly predict any samples with these subclass codes in the future without adding additional training data.

On the other hand, among the most frequent subclass codes in the Scanned Receipts Test Set, the accuracy scores of the predictions are varied. Table 6.2 shows the ten most frequent subclass codes in the Scanned Receipts Test Set, and here, the accuracy scores show that the models are still able to recognise patterns in samples corresponding to some of the subclass codes. This is reflected in the higher model performance scores in table 6.2.

Code	Test Set	Training Set	Model Performance	
Subclass	Count	Count	LR	RF
01.1.1.3	58	28 770	0.828	**0.897**
13.1.2.0	35	129 901	0.800	**0.943**
01.1.4.5	34	850	**0.853**	0.647
01.1.9.3	33	22 629	**0.758**	**0.758**
01.1.8.5	30	16 174	0.467	**0.733**
01.1.2.5	29	2 273	**0.862**	**0.862**
11.1.1.2	26	10	0.000	0.000
01.1.7.9	26	6 250	0.654	**0.769**
01.1.7.2	25	132	0.440	**0.520**
01.1.8.6	25	1 598	0.600	**0.680**

Table 6.2: Average accuracy of each model for the most frequent COICOP subclass codes in the Scanned Receipts Test Set

Despite table 6.2 showing that correct predictions do occur, this in itself does not necessarily bring much value to an automated "human-in-the-loop"-based classification system. As previously noted, the prediction probabilities on the Scanned Receipts Test Set do not currently seem to be relevant indicators of where misclassifications occur, and when the probability predictions cannot be trusted, each prediction on unseen samples will have to be manually reviewed. Whereas table 6.1 outlined the issue with lack of sufficient training data for particular subclass codes, the issue of high prediction probability values for misclassifications may come down to the lack of *representative* training data for the scanned receipt items. This means that the high prediction probability scores may indicate that the models have in fact seen similar item names before, but misclassifications occur because they have not seen that the particular item names can correspond to the same subclass codes as they do in the Scanned Receipts Test Set.

Items	Training Set		Test Set
Item Name	Occurrences	Label	Label
Clausthaler	26	02.3.1.0	01.2.9.0
Kokt Skinke	29	01.1.2.3	01.1.2.2
Isbergsalat	9	01.1.7.4	01.1.7.1
Badeshorts	118	03.2.1.3	03.1.2.1
Blomsterpotte	40	09.3.1.2	09.3.1.1

Table 6.3: Same item names with different labels in the Training Data Set and the Scanned Receipts Test Set

Table 6.3 lists some of the more prominent examples of this, where the exact same item names occur in both data sets, but the labelled subclass code is not the same. Because of this, none of these items were "correctly" classified despite all models expressing high confidence (>90 %) in these predictions. This highlights another source of error, perhaps due to disparity in the labelling procedures of the two data sets, or as a consequence of how the training data was prepared and combined (see section 3.2). However, exploring the validity of the labels and the labelling proce-

dures is beyond the scope of this thesis and has not been explored further.

It is likely that the issue with similar item names not corresponding to the same subclass codes in the data sets can be improved by training on more representative training data, i.e. data from the same population that have been labelled following the same labelling procedures. However, it should also be noted that even training on more representative data might not necessarily be a silver bullet for increased performance and reliability in future predictions. The way in which some of these items are represented in the Scanned Receipts Test Set makes them intrinsically ambiguous, and they are consequently very difficult to classify correctly.

For example, some of the samples within division category 11 in the Scanned Receipts Test Set have item names that can make the classifications difficult. As shown in table 6.4, the item names of some of these samples appear to be item names describing ordinary food items that would typically belong in division category 01. However, as these items were served at restaurants or cafés, they should belong in division category 11.

Item Name	Predicted	Prediction Confidence	True Label
Wienerbrød	01.1.1.3	98.26 %	11.1.1.2
Ostekake	01.1.1.3	97.17 %	11.1.1.2
Burger	01.1.2.5	96.99 %	11.1.1.2
Pizza	01.1.1.1	96.83 %	11.1.1.1
Baguette Ost og Skinke	01.1.9.1	95.88 %	11.1.1.2

Table 6.4: Random Forest misclassified samples with high prediction confidence

Without any additional context, there is no clear way to make a distinction on whether the item was sold at a grocery store or served at a restaurant. When manually labelling these items, the coder at SSB was privy to information about price and store name for each sample, and for those samples, these parameters added enough contextual information to significantly reduce the ambiguity. A model that bases its predictions solely on item names will not be able to make reliable predictions for these items, even with more representative training data added. Still, if a classifier has seen that "wienerbrød" ("danish") can possibly relate to subclass codes within both the food (01) and the restaurant category (11), the classifier would likely express a lower confidence score for this item as it will have experienced that these correlations are ambiguous.

Ultimately, the observations in the preceding paragraphs indicate that a portion of errors and misclassifications can most likely be ascribed to the lack of training data that is representative of scanned receipt items. This involves a general lack of training data for some subclass codes, while for others, the disparity between the training data and the scanned receipt data appears to be limiting. Therefore, the absence of representative training data is likely the current major limiting factor for implementing an automatic classification system for items from scanned receipts with an acceptable degree of reliability.

6.2 Limitation of the Study

This thesis acts as a preliminary study of the potential of implementing machine learning for automatic classification of items for the survey of consumer expenditure. Whereas the findings of this study are the result of a collaboration with Statistics Norway over a period of three months, the scope of what could be explored was ultimately restricted by the time frame of this project. This proved to be particularly relevant for the acquisition of data, as a majority of the training data used in this study became available late in the project. This caused necessary changes to the original scope of this thesis, and this subsequently determined which methods that would be feasible, and which results that would be obtainable.

6.2.1 Data

The available data sets have played a large role in defining the scope of this project. Whereas some data sets were readily available from the start, several data sets were not available until the latter stages of this project, while some turned out not to be available within the specified time frame at all. Among these, manually labelled training data of items from scanned receipts turned out not to be available while working on this project besides a small data sample used for testing (see subsection 3.2.5). Therefore, as the models evaluated in this thesis were not trained on data from scanned receipts, but were still evaluated on it, the performance assessments do not necessarily paint an accurate picture of how well these models could perform on scanned receipts data from the survey if given more representative data to train on.

The choice of data sets largely determines what a classifier can learn, and consequently the obtainable results. Among the data sets that ended up being available in this study, the only variables that the data sets had in common were *item names* and *5-digit COICOP 2018 subclass codes*. These variables enabled the model to learn correlations between the different item names and corresponding COICOP codes. However, they did not enable the models to predict COICOP codes based on any other variable than the item names. As the item names in receipts are typically quite succinct, this limitation can restrict a classifier's ability to make important distinctions for items, and this was shown in section 6.1, where the "wienerbrød problem" highlights this very disadvantage. As food served at restaurants or cafés are typically more expensive than food sold in grocery stores, the inclusion of *item price* might prove to be a useful predictor variable in these classifications. Additionally, the *store names* of where an item was sold might also prove to be a helpful variable for many items, as the type of store can help indicate the type of products they typically sell.

6.2.2 Models

Whereas the choice of data affects what the classifier can learn, the design of the model contributes to how a classifier learns. This study has employed two traditional classifiers from the Scikit-learn library, in combination with count-based feature extractors, to train and develop COICOP classifiers. Originally, this study aspired

to investigate the performances of several additional classifier models, and among these were different variants of Support Vector Machines (SVM) and an implementation of a majority voting classifier. However, due to limited time and available computing resources, this turned out not to be feasible. Training these models on the full training data set, or even most subsets of the training data required more computational resources than what was available. However, a linear SVM classifier was trained on a small subset of the training data without any samples from the Imports data set. The results from this process indicate that the SVM model achieved performance scores that were on par or slightly better than the models evaluated in this thesis. The results from performance evaluations of the SVM model are included in Appendix C.3. For similar reasons, no tuning of the models was done on the full training data set. However, some limited tuning on a narrow range of hyperparameters was done for the models on a subset of the full training data set. The steps involved in this process and the results are included in Appendix B.2.

6.3 Results in Related Studies

Section 4.3 presented two related works that implemented classification systems based on prediction probabilities of classifier models with the purpose of streamlining statistics production. This section compares the findings in these related works with the findings in this thesis.

In the works of Benedikt et al. (2020), they explored a wide range of classifier models for classification of items extracted from scanned images of receipts. Similar to the findings in this study, Benedikt et al. found the character-based feature extraction methods to work well with simple implementations of Scikit-learn classifiers. In their study, they found that Support Vector Machines (SVM), Logistic Regression and Random Forest performed very similarly, with only 0.2 percentage points separating SVM from the rest. Similarly, in the works of Myklatun (2019), Support Vector Machines and Logistic Regression were among the top performers.

However, out of the two studies, it is only Benedikt et al. that explored classification of items into 5-digit COICOP subclass codes, and therefore, it is perhaps their results that are the most relevant to the findings in this thesis. In the evaluation of model performances, Benedikt et al. used held-out partitions from their training data to evaluate their models, and their results show that most models achieved accuracy scores around 75% to 85%. These results are similar to the accuracy scores obtained by the same models on the held-out test data in this thesis, where both models achieved accuracy scores above 81% (see section 5.2).

In contrast to this thesis, Benedikt et al. also explored the COICOP classification performance of the shallow neural network text classifier *fastText*[1]. Overall, fastText achieved similar performance scores as SVM, obtaining 85% accuracy for predictions on their test data. Additionally, the FastText classifier appeared to be one of the best candidates for a "human-in-the-loop" automatic classification system, where 62% of the items in their test set could automatically be classified at the cost of 3% incorrect classifications. These results were only matched by the Random Forest classifier, which also showed the best potential of the models evaluated in this thesis.

It should be noted that despite employing similar methods, the results from the study of Benedikt et al. are not directly comparable to the results in this thesis as they have used their own data sets in their experiments. Still, the similar results would suggest that the suitability of the models for 5-digit COICOP classification tasks are not simply limited to the use of a specific data set. These observations would also further support the idea that similar performance metrics and results should be obtainable for scanned receipt data from the survey of consumer expenditure, provided that the models are given a sufficiently representative data set to train on.

[1] fastText is a open-source library for efficient learning of word representations and sentence classification (Joulin et al., 2016)

6.4 Future Work

The findings in this study provide insight into the state of Statistics Norway's current potential for implementing automatic classification of consumer goods. These findings indicate that there is a promising potential for automatic classification through a "human-in-the-loop" integrated classification system given representative data to train classifiers on. However, because of the lack thereof, the preliminary performance results on the scanned receipt data in this study should be treated as somewhat inconclusive. Still, they do highlight some existing deficiencies in Statistics Norway's current ability to accomplish automatic classification of receipt items for the survey of consumer expenditure, as the current need to acquire representative training data appears to be crucial.

Whereas this study has been limited both in terms of scope and timespan, these limitations *do* also provide opportunities for future studies and research into ways to further develop and optimise models for COICOP classification. Furthermore, this study forms a foundation on which Statistics Norway can base their assessments on whether to pursue the development of an automatic COICOP classification system through supervised machine learning. The following subsections outline some of the observed potential for further work related to the applied methods, and some proposed next steps on the path to developing an automatic classification system for items from the survey of consumer expenditure.

6.4.1 Methods

This study has not placed a large emphasis on exploring different ways to process and represent words in the item names. Section 4.1 describes the applied pre-processing techniques and feature extractor methods in this study, and whereas these methods have enabled this thesis to reach its stated objectives, more modern and advanced techniques for representing semantic information do exist. The count-based feature extraction methods used in this thesis are able to learn occurrences of words (or characters) in text, but they are not able to learn which words that are more similar than others. Word embeddings layers in neural network models can do this, as they instead represent words as vectors in a vector space where related words have a similar vector representation (Raschka and Mirjalili, 2019, p.590). With this approach, the models might be able to also learn that item names such as *raspeball*, *komle*, and *klubb* (which all refers to the same dish) are in fact related, which might prove to further increase classification performance.

Even though the findings in this study show the need to acquire representative training data for scanned receipt items, there also appears to be a potential for further model performance improvements on the Scanned Receipts Test Set through experimentation with the currently available data sets. The assembled COICOP training data set is a product of 5 distinct data sets, and among them, each data set might vary in terms of their representativeness of items in the Scanned Receipts Test Set. For example, the Imports data set (which also make up the overwhelming majority of the assembled COICOP training data set) might not be as representative of scanned receipt items as the Transactions data set or the manually registered Re-

ceipts data set. Preliminary tests indicate that the exclusion of the Imports data set when training classifiers, yields better classification performance on the Scanned Receipts Test Set. These results are shown in Appendix D.1. Additionally, custom weighting could be assigned to the different data sets of the assembled COICOP training data in order to emphasise which item names that are more important for a classifier to learn than others. Preliminary tests have also been done for this, where assigning custom weights to the different data sets when training classifiers, shows some classification performance improvements on the Scanned Receipts Test Set. These results are included in Appendix D.2.

Furthermore, exploration of additional classifiers and further fine-tuning of models might also yield better performance results. This study primarily explored the use of Logistic Regression and Random Forest, and while these classifiers were able to learn patterns in the training data quite well, there are many other classifiers that might prove to perform better on the same data. As previously shown, a common finding in related studies is that Support Vector Machines (SVM) are able to perform well for similar problems. Although this study has only performed preliminary testing with linear SVM on a subset of the full training data due to hardware restrictions, these results indicate a promising potential for the SVM classifier as it currently is the overall highest scoring model for tests on the Scanned Receipts Test Set. Additionally, exploration of neural network models might also be of interest as these have become state-of-the-art in many fields of text classification (Minaee et al., 2021). However, as these models are typically computationally expensive to train (Kowsari et al., 2019), training complex neural networks might not be feasible with the current hardware resources available at Statistics Norway. However, the fastText classifier, which showed promising results for Benedikt et al. (2020), offers fast training times on large data sets (Joulin et al., 2016), and it could therefore be a good alternative to pursue if the opportunity presents itself in the future.

6.4.2 Practical Applications

The research in this study aims to build a proof of concept, illustrating that automatic COICOP classification can be done using supervised machine learning. The tests on 297 859 items from held-out test data show promising results and they indicate that there is a real potential for automation while preserving data quality through a "human-in-the-loop" integration. Still, there is work remaining to replicate these results on items from scanned receipt data, and subsequently in the development of a classifier system that is fit-for-purpose for automatic classification of registered items from the survey of consumer expenditure. As outlined in this study, the main limiting factor appears to be the lack of representative training data. To address this issue, there are some approaches that may be relevant. However, depending on how Statistics Norway want to allocate their resources, not all may be equally realistic.

One simple approach is to annotate more samples from scanned receipt data. This requires the allocation of human resources to do the labelling, but in this way, one can acquire training data that is representative of the data that the models are intended to classify in the future. However, the findings in this study indicate that

there is a potential for doing this in a partially strategic manner, where underperforming subclass codes could first be targeted in some way. Since the scanned receipt data set contains unlabelled samples, finding perfect instances of these samples in the unstructured data can be difficult. However, for certain codes, there might be variables in the scanned receipt data set that can be used for this. For example, *store name* could be used to quickly identify items sold at restaurants and cafés.

Alternatively, or combined with the above, Statistics Norway could implement a "human-in-the-loop" classification system right away, provided that most predictions are manually reviewed in the beginning. As more labelled data are added through human intervention, prediction performances are expected to improve with the active learning aspect of this system. However, this also requires the allocation of human resources to review the quality of the predictions, especially in the beginning. Still, in this way, Statistics Norway can closely monitor the development of the classification system's performance and determine when a satisfactory trade-off between invested resources and prediction quality is achieved.

Chapter 7

Conclusion

Overall, the findings in this study show that basic machine learning models and natural language processing techniques appear to be sufficiently capable methods for 5-digit COICOP classification of consumer goods. At the same time, the findings also highlight important limitations with the data sets, as the results indicate that they do not enable the models to reliably predict items for the survey of consumer expenditure. It is therefore the conclusion of this study that the explored methods show sufficient potential for further development, where the next step of which would be to acquire training data more relevant to Statistics Norway's survey of consumer expenditure.

Bibliography

Ahmed, F., De Luca, E., & Nürnberger, A. (2009). Revised n-gram based automatic spelling correction tool to improve retrieval effectiveness. *Polibits*, *40*, 39–48. https://doi.org/10.17562/PB-40-6

Benedikt, L., Joshi, C., Nolan, L., de Wolf, N., & Schouten, B. (2020). *Optical character recognition and machine learning classification of shopping receipts*. Eurostat.

Egge-Hoveid, K., & Brændvang, A. K. (2020). *Prosjektbegrunnelse forbruksstatistikk 2022* [Unpublished Internal Document at Statistics Norway].

Holmøy, A., & Lillegård, M. (2014). *Forbruksundersøkelsen 2012. dokumentasjon av datainnsamling, analyse av datakvalitet og beregning av frafallsvekter*. Statistisk Sentralbyrå.

Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2016). Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Kowsari, K., Meimandi, K. J., Heidarysafa, M., Mendu, S., Barnes, L. E., & Brown, D. E. (2019). Text classification algorithms: A survey. *CoRR*, *abs/1904.08067*. http://arxiv.org/abs/1904.08067

Maslova, O., & Da Cruz, A. (2019). Product labels classification from receipt: Word2vec and cnn approach [Accessed: 2021-12-06]. https://blog.aboutgoods-company.com/receipt-labels-classification-word2vec-and-cnn-approach/

Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., & Gao, J. (2021). Deep learning–based text classification: A comprehensive review. *ACM Comput. Surv.*, *54*(3). https://doi.org/10.1145/3439726

Myklatun, K. (2019). *Utilizing machine learning in the consumer price index*. Nordic Statistical Meeting.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Raschka, S., & Mirjalili, V. (2019). *Python machine learning: Machine learning and deep learning with python, scikit-learn, and tensorflow* (3rd). Packt Publishing Ltd.

Shahmirzadi, O., Lugowski, A., & Younge, K. (2019). Text similarity in vector space models: A comparative study. *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, 659–666. https://doi.org/10.1109/ICMLA.2019.00120

Škrlj, B., Martinc, M., Kralj, J., Lavrač, N., & Pollak, S. (2021). Tax2vec: Constructing interpretable features from taxonomies for short text classification.

Computer Speech and Language, *65*, 101104. https://doi.org/10.1016/j.csl.2020.101104

SSB. (2019). *Forbruksstatistikk før og nå* [Unpublished Internal Document at Statistics Norway].

SSB. (2021). *Forbruk 2022: Nytt digitalt design* [Unpublished Internal Document at Statistics Norway].

UN. (2018). *Classification of individual consumption according to purpose (coicop) 2018* [Accessed: 2010-08-22]. https://unstats.un.org/unsd/classifications/unsdclassifications/COICOP_2018_-_pre-edited_white_cover_version_-_2018-12-26.pdf

Appendix A

Data

This appendix chapter includes additional information about the data sets used in this thesis. The following sections are particularly relevant to chapter 3 of this thesis.

A.1 Missing Subclass Codes in Training Data

The assembled COICOP training data set consists of 283 unique COICOP 2018 subclass codes in total. There are 298 subclass codes the COICOP 2018 coding framework that relate to individual consumption, and table A.1 shows all 15 subclass codes that are missing from the assembled COICOP training data set.

Code	Description
01.1.2.1	Live land animals
01.3.0.0	Services for processing primary goods for food and non-alcoholic beverages
04.1.1.0	Actual rentals paid by tenants for main residence
04.2.1.0	Imputed rentals of owner-occupiers for main residence
04.2.2.0	Other imputed rentals
04.4.1.1	Water supply through network systems
04.4.1.2	Water supply through other systems
04.4.3.1	Sewage collection through sewer systems
04.4.3.2	Sewage collection through onsite sanitation systems
04.4.4.1	Maintenance charges in multi-occupied buildings
04.4.4.9	Other services related to dwelling
05.4.0.4	Repair and hire of glassware, tableware and household utensils
10.1.0.1	Early childhood education
11.1.2.9	Other canteens, cafeterias and refectories
13.2.9.2	Repair or hire of other personal effects not elsewhere classified

Table A.1: Missing subclass codes from the Training Data Set

A.2 Subclass Codes in Training Data Set

Figure A.1 shows the distribution of subclass codes in the assembled COICOP training data set.

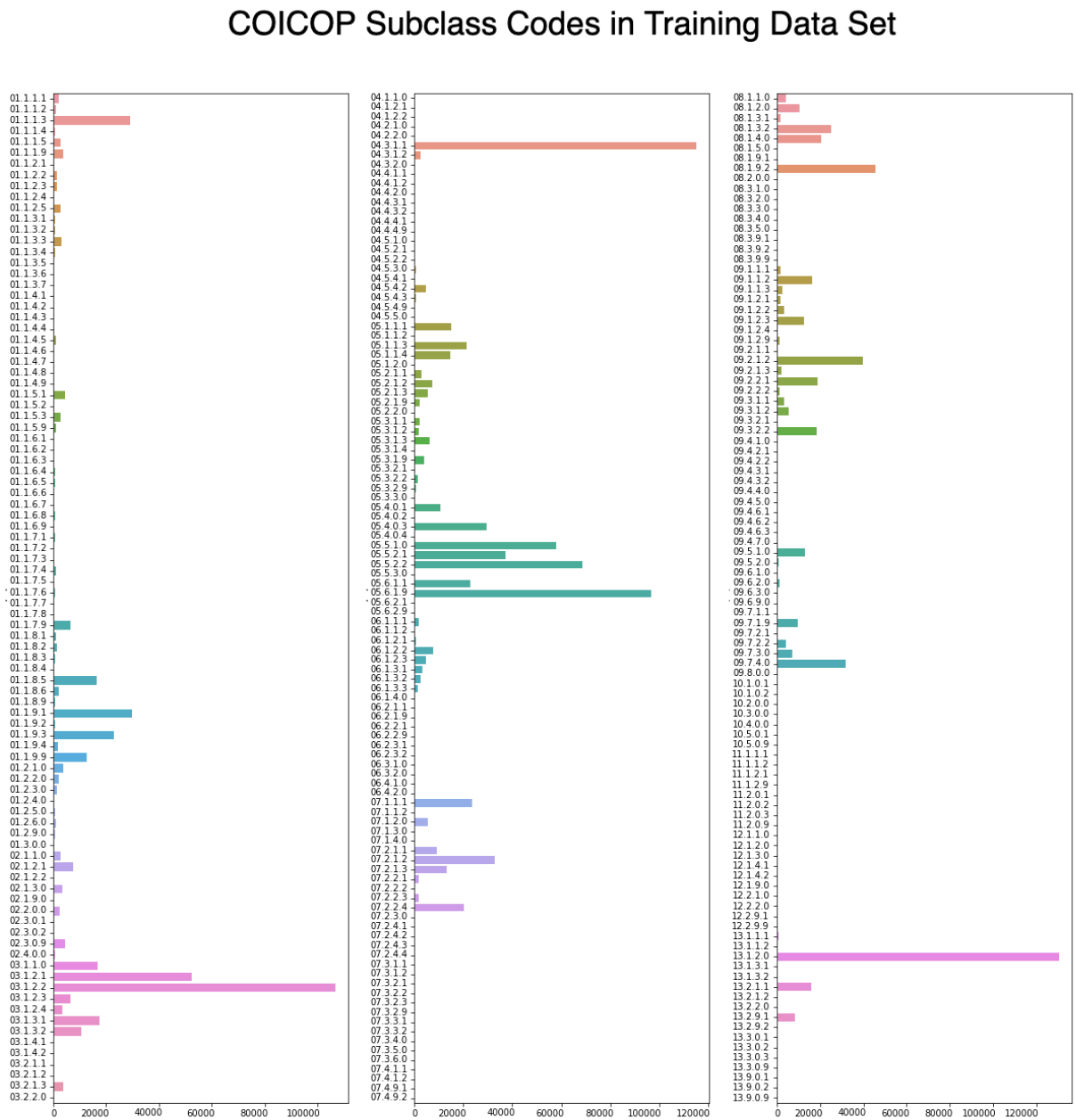


Figure A.1: Count of subclass codes in Training Data Set

A.3 Subclass Codes in Test Data Set

Figure A.2 shows the distribution of subclass codes in the Scanned Receipts Test Set.

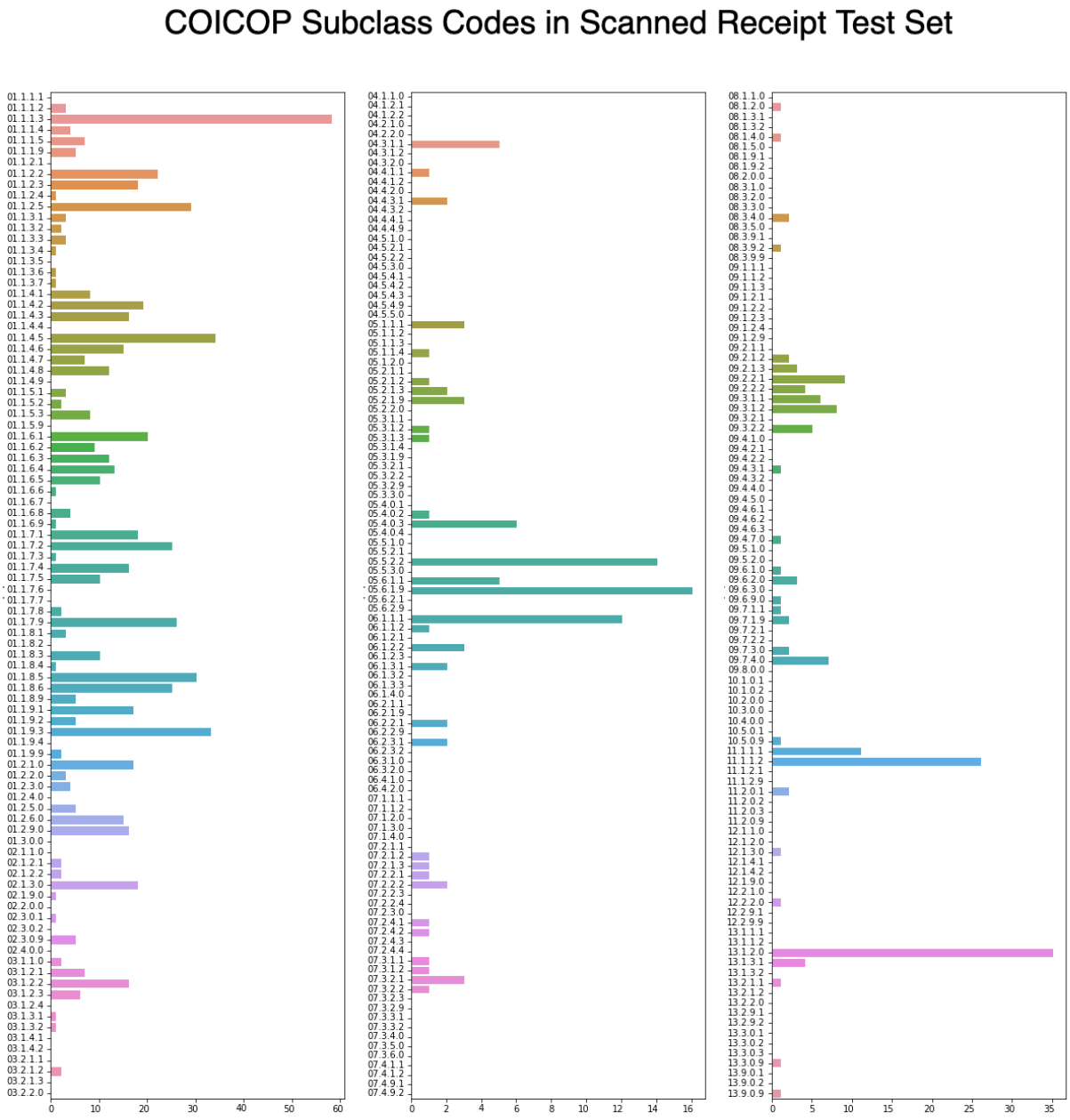


Figure A.2: Count of subclass codes in Scanned Receipts Test Set

Appendix B

Model Results

This appendix chapter includes additional results from model training and evaluation. The following sections are particularly relevant to chapter 5 of this thesis.

B.1 Model Selection

Table B.1 and B.2 show the performance metric scores for the Logistic Regression classifier and Random Forest classifier combined with each feature extractor variant. The highlighted cells indicate the highest performing feature extractor with that particular classifier. In all performance evaluations, a subset of 10 % of the full training data set has been used. The subset was assembled by randomly sampling from the full training data set, where the sampling was stratified on the COICOP subclass codes.

Logistic Regression on Subset of Training Data Set				
Feature Extractor	Precision	Recall	F1-Score	Accuracy
CV-w	0.805	0.767	0.766	0.767
CV-w22	0.809	0.479	0.521	0.479
CV-w23	0.820	0.479	0.524	0.479
CV-w33	0.693	0.277	0.258	0.277
CV-ch22	0.725	0.728	0.721	0.728
CV-ch23	0.835	0.837	0.833	0.837
CV-ch33	**0.837**	**0.840**	**0.835**	**0.840**
TFIDF-w	0.760	0.719	0.713	0.720
TFIDF-w22	0.713	0.401	0.400	0.401
TFIDF-w23	0.713	0.401	0.400	0.401
TFIDF-w33	0.686	0.247	0.221	0.247
TFIDF-ch22	0.647	0.653	0.627	0.653
TFIDF-ch23	0.750	0.760	0.737	0.760
TFIDF-ch33	0.739	0.748	0.725	0.748

Table B.1: Logistic Regression feature extraction test scores on hold-out set from subset of training data set

The results from table B.1 indicate that Logistic Regression with CountVectorizer using N-grams of 3 characters is the best performing model. Note that all models

in this table were trained on a 10 % subset of the full training data. The results might therefore not be accurate of the optimal feature extraction method of the full training data set. Still, this should give an indication of which feature extraction method that is expected to perform well with the Logistic Regression classifier.

Similarly, the results from table B.2 indicate that Random Forest with CountVectorizer using N-grams of 2 and 3 characters is the best performing model.

Random Forest on Subset of Training Data Set				
Feature Extractor	Precision	Recall	F1-Score	Accuracy
CV-w	0.802	0.773	0.774	0.773
CV-w22	0.834	0.592	0.644	0.592
CV-w23	**0.835**	0.588	0.641	0.588
CV-w33	0.833	0.362	0.412	0.362
CV-ch22	0.809	0.812	0.803	0.812
CV-ch23	0.821	**0.825**	**0.818**	**0.825**
CV-ch33	0.820	0.823	0.817	0.823
TFIDF-w	0.764	0.742	0.741	0.742
TFIDF-w22	0.806	0.441	0.491	0.441
TFIDF-w23	0.803	0.441	0.491	0.441
TFIDF-w33	0.781	0.254	0.287	0.254
TFIDF-ch22	0.793	0.793	0.782	0.793
TFIDF-ch23	0.810	0.817	0.809	0.817
TFIDF-ch33	0.815	0.819	0.810	0.819

Table B.2: Random Forest feature extraction test scores on hold-out set from subset of training data set

B.2 Model Tuning

Most machine learning classifiers implemented in Scikit-learn come with a set of hyperparameters (user-specified classifier parameters) that can be adjusted to improve the classifiers' ability to capture patterns in the data. In order to identify a set of suitable hyperparameters, each classifier is typically retrained multiple times while slightly tweaking certain hyperparameters between each iteration. The set of identified optimal hyperparameters is normally chosen as the one that achieves the highest performance for a specified performance metric (Raschka & Mirjalili, 2019, p. 207).

Some experiments with tuning the different classifiers were performed. However, due to computational limitations, the explored hyperparameters and hyperparameter settings were narrowed down substantially. Additionally, the tuning was done on the same 10 % subset of the full training data set as used in section B.1. The subset was assembled by randomly sampling from the full training data set, where the sampling was stratified on the COICOP subclass codes.

The observed best-performing combinations of feature extraction methods and classifiers from tables B.1 and B.2 were tuned using GridSearchCV. The evaluation was done using 5-fold cross-validation and $F1$-score, resulting in a total of 75 trained Logistic Regression models and 60 trained Random Forest models. Figures B.1 and B.2 shows the $F1$-score for the different hyperparameter values. Here, the $F1$-score shows the averaged $F1$-score across all cross-validation (CV) folds for each hyperparameter value.

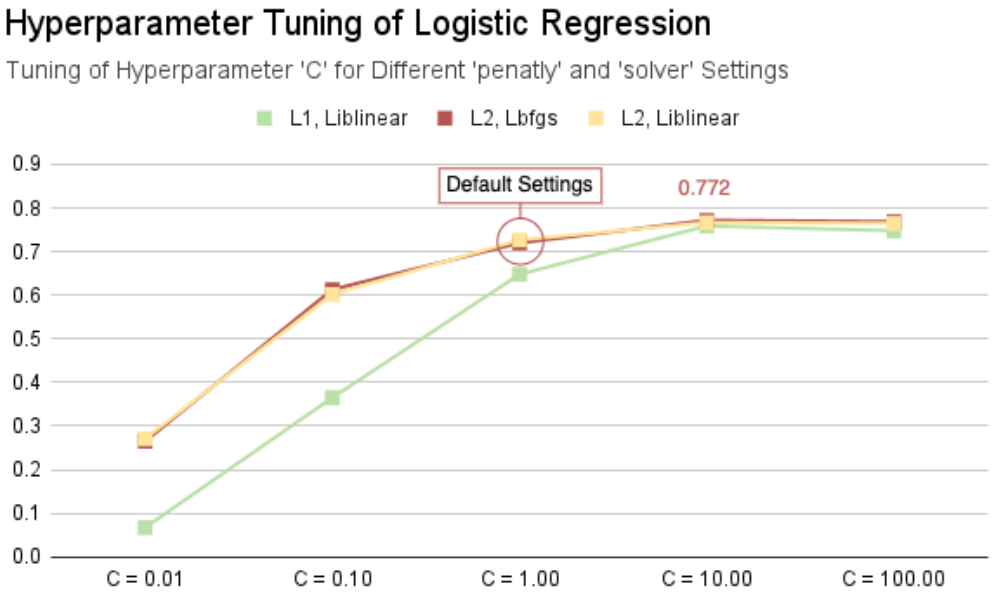


Figure B.1: Logistic Regression hyperparameter tuning

The default Scikit-learn implementation of Logistic Regression uses *L2* penalty, *lbfgs* solver and inverse regularisation strength, $C = 1.00$. Figure B.1 shows that for

higher values of "C", the models were able to achieve very similar $F1$-scores. Logistic Regression with "L1" penalty and "liblinear" solver fell off for lower values of "C", however. The two "L2"-based models performed very similarly for all values of "C", where each model peaked in performance with regularisation parameter value, "C" set to 10. For higher values, performance started to fall off for all models.

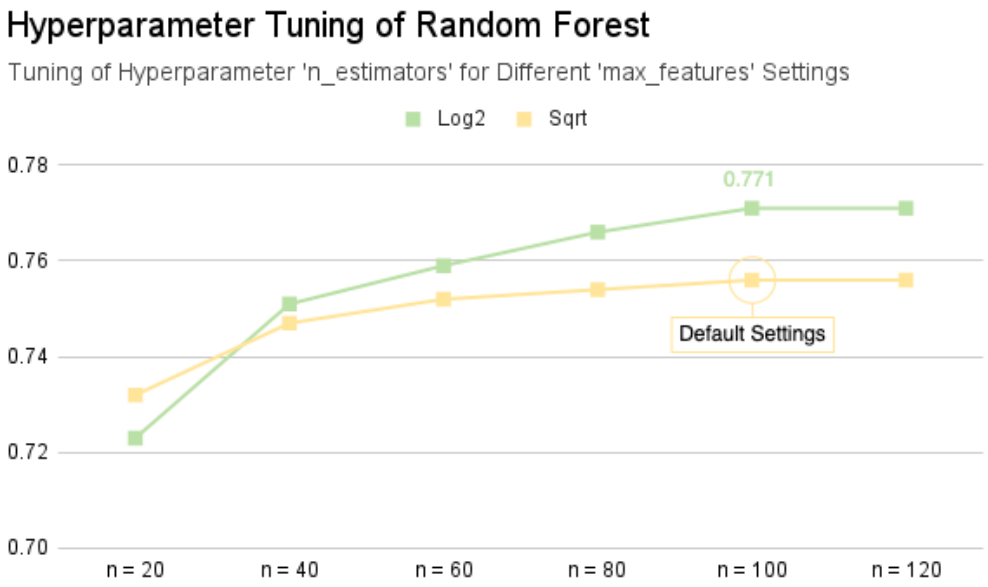


Figure B.2: Random Forest hyperparameter tuning

The default Scikit-learn implementation of Random Forest uses *n_estimators* = 100 and *max_features* set to 'auto'. According to the Scikit-learn documentation, the 'auto' and 'sqrt' settings calculate the number of features in the exact same way (Pedregosa et al., 2011). Therefore, 'sqrt' is treated as the default setting for *max_features* in this evaluation. As figure B.2 shows, all evaluated Random Forest models were able to achieve very similar $F1$-scores. For 20 estimators, "sqrt" was slightly better than "log2", however, for all other evaluated *n_estimators*, "log2" was the best performing Random Forest model. As the number of estimators increased, the $F1$-score increased until it peaked at 100 estimators. When further increasing the number of estimators, the $F1$-score seemed to not increase any further for both variants.

Table B.3 summarises the top performing combination of the evaluated hyperparameter values for both classifiers.

Classifier Model		Tuning Results	
Classifier	Extractor	Hyperparameters	Hyperparameter Value
Logistic Regression	CV-ch33	C	10.0
		penalty	L2
		solver	liblinear
Random Forest	CV-ch23	n_estimators	100
		max_features	log2

Table B.3: Best performing hyperparameters values for each classifier model

The preliminary results of this section indicate that model performance can be further increased with hyperparameter tuning. However, because of the aforementioned limitations with computational resources, this thesis has not explored model tuning any further.

Appendix C

Support Vector Machines

This thesis originally aspired to explore the potential of support vector machines (SVM) for 5-digit COICOP subclass classification. However, due to limited time and available computing resources, this turned out not to be feasible. Instead, this Appendix presents classification performance results of the SVM classifier on a subset of the training data. This subset is simply the exclusion of the Imports data set of the full assembled training data set (for more information about the different data sets in the assembled training data set, see section 3.1). First, some background theory of the SVM classifier is introduced, and then, the results from performance evaluations on held-out test data and Scanned Receipts Test set are presented.

C.1 SVM Theory

Support vector machines (SVM) is a popular supervised machine learning algorithm, where the main idea is to identify an optimal hyperplane in a finite-dimensional feature space that differentiates between samples depending on their class (target label).

For each of the n samples in an $n \times m$-dimensional dataset, each sample, i, belongs to either a positive or a negative class. This can be expressed as (x_i, y_i) for $i = \{1, ..., n\}$, where x is the m-dimensional feature vector and y^i indicates which class sample i belongs to ($y^i = 1$ for the positive class and $y^i = -1$ for the negative class).

Support vector machines aim to identify an $m - 1$ dimensional hyperplane that separates the observations into two classes. In figure C.1, the samples are represented in a two-dimensional feature space, resulting in a one-dimensional hyperplane (line) that acts as the decision boundary between the classes.

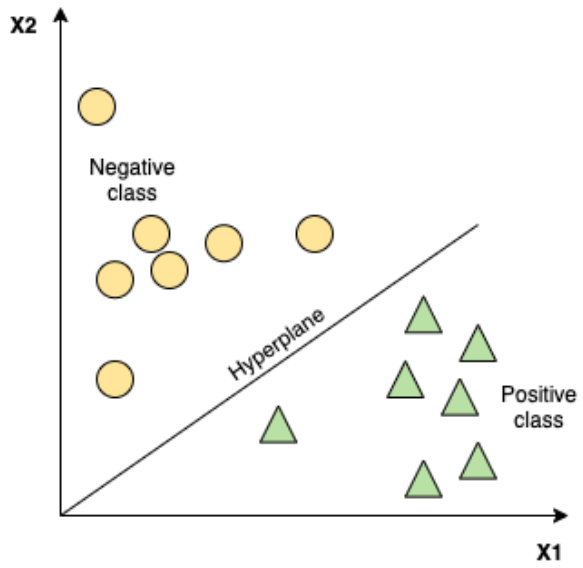


Figure C.1: Hyperplane in a 2-dimensional feature space

For samples that are linearly separable, there is an infinite number of possible hyperplanes that separates the samples into each class. Many of these hyperplanes will not generalise well to new data, however, as smaller margins are more prone to overfitting (Raschka and Mirjalili, 2019, p. 79). Therefore, the hyperplane that maximises the distance between the hyperplane and its nearest sample, is chosen as the optimal hyperplane. The hyperplane can be expressed as all data points that satisfy equation C.1. Here, w represents a weight vector that is orthogonal to the hyperplane, and b represents the bias, which can be seen as the offset from the origin.

$$w^T x + b = 0 \tag{C.1}$$

The distance between the hyperplane and its nearest samples is called the *margin*. These samples are called support vectors, which form the negative and positive hyperplanes that are parallel to the decision boundary, shown in figure C.2. These hyperplanes can be expressed as:

$$w^T x_{pos} + b = 1 \tag{C.2}$$

$$w^T x_{neg} + b = -1 \tag{C.3}$$

By subtracting the two equations C.2 and C.3 from each other and normalising this by the length of the vector w, we arrive at the following equation:

$$\frac{w^T(x_{pos} - x_{neg})}{\|w\|} = \frac{2}{\|w\|} \tag{C.4}$$

The left side of equation C.4 represents the distance between the positive and negative hyperplane, which is the margin that SVM aims to maximise. This means that the objective function of SVM becomes to maximise the expression $\frac{2}{\|w\|}$, under the constraint that each sample is classified correctly, meaning that no samples should

fall into the area between the positive and the negative hyperplane, but every sample should be located on the correct side of the margin.

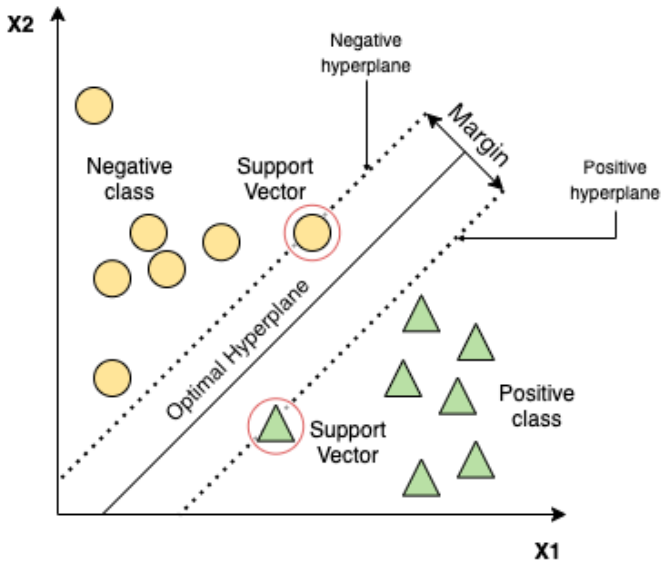


Figure C.2: Hyperplane that maximises the margin

C.2 SVM Performance on Training Set

This section presents results from classifier performance on the training data set. As previously described, the training data set in this chapter refers to a subset of the full assembled COICOP training data set. This subset excludes all samples from the Import set (for more information about the different data sets in the assembled training data set, see section 3.1). This section also presents performance results of the Logistic Regression and Random Forest models on the same subset.

Classifier Model		Performance Metrics			
Classifier	Extractor	Precision	Recall	F1-Score	Accuracy
Training Set (80 %)					
SVM	TFIDF-ch33	0.953	0.953	0.950	0.953
LR	CV-ch33	0.987	0.987	0.987	0.987
RF	CV-ch23	0.991	0.991	0.991	0.991
Hold-out Set (20 %)					
SVM	TFIDF-ch33	0.817	0.806	0.794	0.806
LR	CV-ch33	**0.844**	**0.846**	**0.842**	**0.846**
RF	CV-ch23	0.824	0.827	0.820	0.827

Table C.1: Performance of classifier models on training data without the Imports data set

As shown in both table C.1 and C.2, the Logistic Regression model is the best performing model for all performance metrics. However, all models produced similar performance metric scores overall.

Hold-out Set		Classifier Models		
Code	Items	SVM	LR	RF
01	4 480	0.764	**0.787**	0.777
02	927	0.875	**0.913**	0.907
03	1 378	0.907	0.927	**0.933**
04	54	0.426	**0.556**	0.463
05	1 466	0.789	**0.856**	0.823
06	175	0.503	**0.789**	0.703
07	101	0.673	**0.802**	0.752
08	45	0.422	**0.511**	0.489
09	1 385	0.765	**0.844**	0.815
10	1	0.000	**1.000**	**1.000**
11	20	0.850	0.900	**0.950**
12	2	**0.500**	**0.500**	**0.500**
13	1 166	**0.967**	0.949	0.959

Table C.2: Average model accuracy on samples within each COICOP division code in the held-out test set from training data without the Imports data set

C.3 SVM Performance on Scanned Receipts Test Set

To evaluate model performance on the Scanned Receipts Test Set, the models from C.2 are retrained on the full training data set (without any samples from the Imports data set). The model performances on the Scanned Receipt Test Set are presented in this section.

Classifier Model		Performance Metrics			
Classifier	Extractor	Precision	Recall	F1-Score	Accuracy
SVM	TFIDF-ch33	0.628	**0.566**	0.540	**0.566**
LR	CV-ch33	**0.635**	0.560	**0.553**	0.560
RF	CV-ch23	0.603	0.556	0.540	0.556

Table C.3: Performances on Scanned Receipts Test Set of classifier models trained on training data set without Imports data set

Table C.3 shows that both the Logistic Regression and SVM models achieve the highest performance metric scores on the Scanned Receipts Test Set. Table C.4 shows that the SVM classifier obtains the highest accuracy for samples within division codes 01, 02, 05 and 13.

Test Set		Training Data		
Code	Items	SVM	LR	RF
01	641	**0.677**	0.665	0.658
02	29	**0.655**	0.586	0.621
03	35	0.114	0.114	**0.143**
04	8	0.000	0.000	0.000
05	54	**0.370**	0.352	0.333
06	22	0.227	**0.318**	0.273
07	13	0.385	**0.462**	**0.462**
08	5	0.200	**0.400**	**0.400**
09	56	0.250	**0.321**	0.268
10	1	0.000	0.000	0.000
11	39	0.000	0.000	0.000
12	2	0.000	0.000	0.000
13	42	**0.810**	0.738	0.690

Table C.4: Average model accuracy on samples within each COICOP division code in the Scanned Receipts Test Set

Similar to how the prediction probabilities of Logistic Regression and Random Forest models were evaluated on the Scanned Receipts Test Set in section 5.3.2, this section presents the prediction probabilities and error rates at specified threshold values, T, for the SVM classifier model. Note that this model has been trained on training data without any samples from the Imports data set.

Figure C.3 shows the distribution of prediction probabilities for the SVM classifier model. In this visualisation, the prediction probabilities have been bucketed with a

bucket size of 0.05, meaning that the graph might show a less detailed view than the actual distribution.

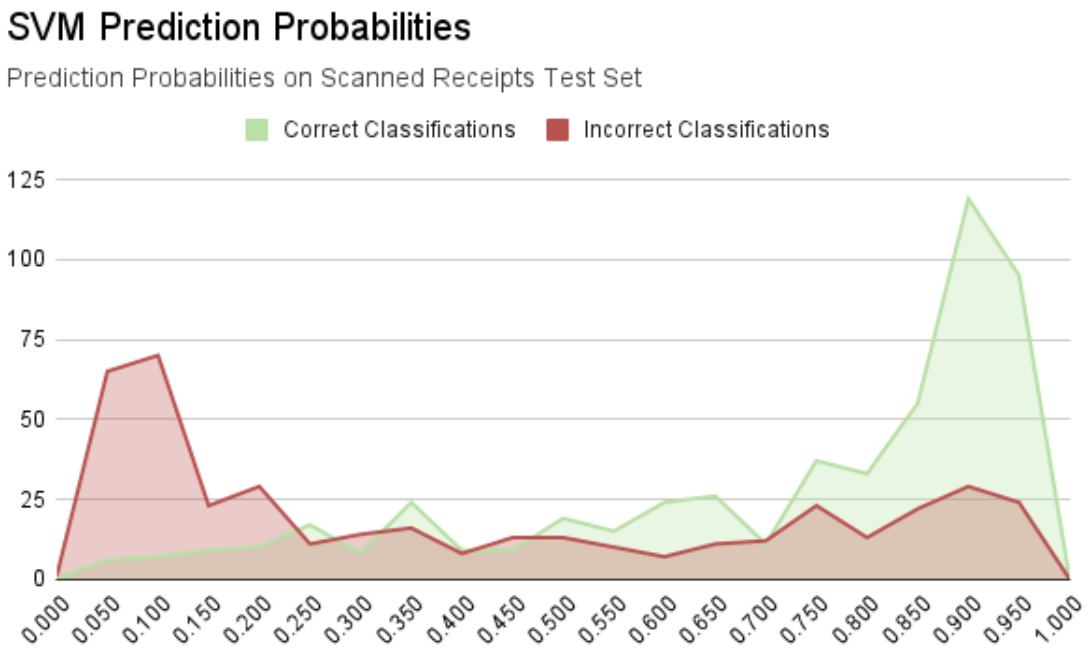


Figure C.3: Distribution of prediction probabilities for SVM predictions on samples in the Scanned Receipts Test Set

As shown in table C.5, the error rate of classifications is above 25 % for all threshold values. As section 5.3.2 shows, these error rates are higher than the error rates at the same threshold values for both the Logistic Regression and Random Forest models.

Count	$T \geq 95$ %	$T \geq 90$ %	$T \geq 85$ %	$T \geq 80$ %
Number of Items	119	267	344	390
Of Total Data	12.57 %	28.19 %	36.33 %	41.18 %
Error Rate (ER)	0.25	0.25	0.28	0.29

Table C.5: SVM: Number of samples in the Scanned Receipt Test Set above threshold value (T) and the classification error rate (ER) of these samples

Appendix D

Modifications of Training Data

D.1 Model Performances without Imports Data

Tables D.1 and D.2 show the performance of the classifier models when trained on full training data and training data without the Imports data. Because of hardware constraints, the SVM model was only trained on the training data set without the Imports data set. As shown in table D.1, the best performing models on the Scanned Receipts Test Set were trained on training data without the Imports data set.

Classifier Model		Performance Metrics			
Classifier	Extractor	Precision	Recall	F1-Score	Accuracy
Without Imports Data					
SVM	TFIDF-ch33	0.628	**0.566**	0.540	**0.566**
Logistic Regression	CV-ch33	**0.635**	0.560	**0.553**	0.560
Random Forest	CV-ch23	0.603	0.556	0.540	0.556
Full Training Data Set					
Logistic Regression	CV-ch33	0.568	0.485	0.489	0.485
Random Forest	CV-ch23	0.613	0.526	0.530	0.526

Table D.1: Performances on Scanned Receipts Test Set of classifier models trained on full training data set and on training data without Imports data

Table D.2 shows that the model trained on training data without Imports data achieves the highest accuracy scores for most division categories. The models trained on the full training data set are only able to achieve the highest average accuracy scores for samples within division code 03, 04 and 05.

Test Set		Without Imports Data			Full Training Set	
Code	Items	SVM	LR	RF	LR	RF
01	641	**0.677**	0.665	0.658	0.569	0.621
02	29	**0.655**	0.586	0.621	0.517	0.586
03	35	0.114	0.114	0.143	0.257	**0.314**
04	8	0.000	0.000	0.000	**0.125**	**0.125**
05	54	0.370	0.352	0.333	0.370	**0.426**
06	22	0.227	**0.318**	0.273	0.000	0.136
07	13	0.385	**0.462**	**0.462**	0.385	0.385
08	5	0.200	**0.400**	**0.400**	0.200	0.000
09	56	0.250	**0.321**	0.268	0.304	0.214
10	1	0.000	0.000	0.000	0.000	0.000
11	39	0.000	0.000	0.000	0.000	0.000
12	2	0.000	0.000	0.000	0.000	0.000
13	42	**0.810**	0.738	0.690	0.619	0.667

Table D.2: Average model accuracy for samples within each COICOP division code in the Scanned Receipts Test Set. Both for models trained on the full training data set and for models trained on data set without the Imports data

D.2 Model Performances with Custom Weighting of Training Data Sets

As mentioned in section 6.4, each data set in the COICOP training data might not be equally representative of items from scanned receipts. This section presents performance results from model evaluations where custom weights were applied to samples within the different data sets and compares the model performance with models trained on the non-weighted COICOP training data set.

The weights assigned to samples within each data set are listed in table D.3. The choice of weights in this experiment was based on subjective assumptions of which data sets that might be more representative of items from scanned receipts. Further experimentation with custom weighting of data sets is encouraged as the results indicate that it is possible to increase model performances on the Scanned Receipts Test Set with this approach.

Data Set	Items	Sample Weights
Receipts	575	1.00
Keywords	2 377	1.00
Transactions	29 776	0.70
CPI	23 541	0.35
Imports	1 433 947	0.10

Table D.3: Assigning custom sample weights to samples within each data set

As shown in table D.4, the Logistic Regression model, that was trained on the weighted training data set, was able to achieve the higher scores for all performance metrics. The accuracy of the Logistic Regression model on the Scanned Receipts Test Set increased by 8 percentage points by applying sample weights to the training data. Similarly, Random Forest performed better when trained on weighted training data, where the accuracy increased by 2.6 percentage points.

Classifier Model		Performance Metrics			
Classifier	Extractor	Precision	Recall	F1-Score	Accuracy
LR	CV-ch33	0.568	0.485	0.489	0.485
LR (Weighted)	CV-ch33	**0.645**	**0.565**	**0.566**	**0.565**
RF	CV-ch23	0.613	0.526	0.530	0.526
RF (Weighted)	CV-ch23	0.620	0.548	0.544	0.548

Table D.4: Performances of weighted and non-weighted models on Scanned Receipts Test Set

Figure D.5 breaks down the overall accuracy into average accuracy by division category. Here, the Logistic Regression model trained on weighted training data was able to achieves the highest accuracy for samples within most division categories.

Hold-Out Set		Classifier Models			
Code	Items	LR	LR (Weighted)	RF	RF (Weighted)
01	641	0.569	**0.655**	0.621	0.646
02	29	0.517	0.655	0.586	**0.690**
03	35	0.257	**0.314**	**0.314**	0.286
04	8	**0.125**	**0.125**	**0.125**	**0.125**
05	54	0.370	0.407	**0.426**	0.407
06	22	0.000	**0.227**	0.136	0.136
07	13	**0.385**	**0.385**	**0.385**	**0.385**
08	5	0.200	**0.400**	0.000	0.000
09	56	**0.304**	**0.304**	0.214	0.232
10	1	0.000	0.000	0.000	0.000
11	39	0.000	0.000	0.000	0.000
12	2	0.000	0.000	0.000	0.000
13	42	0.619	**0.786**	0.667	0.738

Table D.5: Average accuracy of weighted and non-weighted models for samples within each COICOP division code in the Scanned Receipts Test Set

Although the results presented in this section are preliminary, where no comprehensive exploration of different sample weighting has been done, they still highlight a clear potential for model performance improvements on the Scanned Receipts Test Set. This suggests that the models' performance on the Scanned Receipts Test Set can be further increased without the need to acquire additional training data.

Appendix E

Python Code

This Appendix includes a selection of python code that has been previously referenced to in this thesis. As the code developed in this thesis belongs to Statistics Norway, only limited and approved code excerpts are included.

E.1 Pre-processing Code

Figure E.1 illustrates the python code used to implement the pre-processing pipeline. This code was used to pre-process the item names in the data sets.

```
import re


def tokeniser(string, filler_words=[]):
    """Tokenises a string entry of words.

    Args:
        string (string) : String entry to be tokenised
        filler_words (list) : Optional noisy text entries to remove. E.g., grocery store names

    Returns:
        string: Tokenised string
    """

    ## Processing and tokenization ##

    # Convert to string
    string = str(string)

    # Lowercase string
    string = string.lower()

    # Removes -, %, &, keeps 'æøå'
    string = re.sub('[^a-ø0-9]', ' ', string)

    # Replace Quantities with "FAST"
    string = re.sub(r'(\d*,?\.?)\d+\s*(((kilo|mili)?\s*grams?)|gr|(k|m)?g)', " FAST ", string)

    # Replace Quantities with "VÆSKE"
    string = re.sub(r'(\d*,?\.?)\d+\s*((mili|centi)?\s*liter(s)?|((c|m)?l))', " FLYTENDE ", string)

    # Removes Numbers (1 or more)
    string = re.sub(r'\d+', " ", string)

    # Removes Additional Special Characters
    string = re.sub(r"[/.,;`<>\+\-\*\?\'\’]", "", string)

    # Removes single char words
    string = re.sub(r'\b\w{1,1}\b', '', string)

    # Matches Whitespace Characters (also removes '\n')
    string = re.sub(r'\s+', " ", string)


    # Check if any filler words are given
    if filler_words:

        # Remove pre-defined filler words
        filler_word_set = set(filler_words)

    # If no filler words are given
    else:

        # Create empty set
        filler_word_set = set()

    # Create word tokens from string entries
    word_tokens = [word.strip() for word in string.split(" ") if word not in filler_word_set]

    # Return tokenised string sentence
    return " ".join(word_tokens).strip()
```

Figure E.1: Python Code: Pre-processing of Item Names

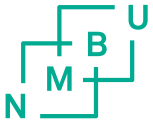
E.2 Custom Search Algorithm Code

Figure E.2 illustrates the python code used to implement the Custom Search Algorithm. This code was used to transform the coding format of the Imports data set.

```
import pandas as pd
import numpy as np


def remove_whitespace_dots(string):
    """Removes white space and dots from string

    Args:
        string (string) : string entry to process
    """

    string = str(string)
    string = string.replace(" ", "")
    string = string.replace(".", "")
    return string


def remove_digit(string, length):
    """Removes a specified length of digits

    Args:
        string (string) : string entry to process
        length (int) : Length of string to keep
    """

    string = str(string)
    string = string[:length]
    return string


def cn_to_cpa(number, dataframe, digits_list = [8, 7, 6, 5, 4], source_col='CN08', target_col='CPA08'):
    """Converts a CN 2008 code to CPA 08 code

    Args:
        number (int or string) : Code number to transform
        dataframe (pandas.DataFrame): (CPA08 : CN08) conversion table to use in transformation
        digits_list (list) : List of digits to use in the search
        source_col (string) : Name of column in conversion table with current code format (CN08)
        target_col (string) : Name of column in conversion table with target code format (CPA08)
    """

    number = str(number)
    df = dataframe.copy()

    match = []
    unique_matches = []

    # Iterate over list of digits
    for digit in digits_list:

        # If no unqiue matches have been found
        if not unique_matches:

            # Check if number (code number) is longer than digit
            if len(number) > digit:

                # If longer, remove digits
                # Remove digits in number so it corresponds with number of digits in the loop
                number = remove_digit(string = number, length = digit)

                # Remove a digit in 'CN08' column
                df[source_col] = df[source_col].map(lambda s:remove_digit(s, length=digit))

                # Get all matches in conversion table
                match = df.loc[df[source_col] == number, target_col]

                # Store unique matches
                list_of_matches = match.tolist()
                unique_matches = set(list_of_matches)

            # If number (code number) is same length as digit
            else:

                # Get all matches in conversion table
                match = df.loc[df[source_col] == number, target_col]


            # Store unique matches
            list_of_matches = match.tolist()
            unique_matches = set(list_of_matches)

    return unique_matches
```

```python
def cpa08_cpa21(df, mapping_df, column_name):
    """Converts a CPA 08 code to CPA 2.1 code

    Args:
        df (pandas.DataFrame) : Dataframe containing CPA08 codes to transform to CPA21 codes
        mapping_df (pandas.DataFrame) : CPA08 : CPA21) conversion table to use in transformation
        column_name (string) : Name of column with CPA08 codes in df
    """

    # Create dataframe copy for manipulations
    dataframe = df.copy()

    # Loop over rows in dataframe
    for row in dataframe[column_name].iteritems():

        # Check whether input is a set
        if isinstance(row[1], set):

            # Create empty set
            set_a = set()

            # Iterate over entries in set
            for i in dataframe[column_name][row[0]]:

                # Find match
                match = mapping_df.loc[mapping_df['CPA08'] == i, 'CPA21']

                list_of_matches = match.tolist()

                # Get unique entries from list of matches
                unique_matches = set(list_of_matches)

                # Iterative over new matches
                for unique_match in unique_matches:
                    set_a.add(unique_match)

                # Add new matches to column entry
                dataframe[column_name][row[0]] = set_a

    return dataframe[column_name]


def cpa_to_coicop(df, mapping_df, column_name, digits_list = [6,5,4,3], source_col='CPA21', target_col='COICOP18'):
    """Converts a CPA 2.1 code to COICOP 2018 code

    Args:
        df (pandas.DataFrame) : Dataframe containing CPA21 codes to transform to COIOCP18 codes
        mapping_df (pandas.DataFrame) : (CPA21 : COICOP18) conversion table to use in transformation
        column_name : Name of column with CPA21 codes in df
        digits_list (list) : List of digits to use in the search
        source_col (string) : Name of column in conversion table with current code format (CPA21)
        target_col (string) : Name of column in conversion table with target code format (COICOP18)
    """

    # Create dataframe copy for manipulations
    dataframe = df.copy()

    # Iterate over rows in dataframe
    for row in dataframe[column_name].iteritems():

        # Check whether input is a set
        if isinstance(row[1], set):

            # Iterate over entries in set
            for i in dataframe[column_name][row[0]]:

                # Iterative Mapping with custom digit range
                matches = cn_to_cpa(str(i),
                                    dataframe=mapping_df,
                                    digits_list=digits_list,
                                    source_col='CPA21',
                                    target_col='COICOP18')


                # Assign matches
                dataframe[column_name][row[0]] = matches

        else:
            # Assign np.nan where CPA21 code is not a set
            dataframe[column_name][row[0]] = np.nan

    return dataframe[column_name]
```

Figure E.2: Python Code: Custom Search Algorithm Code



Norges miljø- og biovitenskapelige universitet
Noregs miljø- og biovitskapelege universitet
Norwegian University of Life Sciences

Postboks 5003
NO-1432 Ås
Norway