THE UNIVERSITY of EDINBURGH

Edinburgh Research Explorer

Application of machine learning techniques in railway demand forecasting

Citation for published version:
Alamdari, NE, Anjos, MF & Savard, G 2021, 'Application of machine learning techniques in railway demand forecasting', *International Journal of Revenue Management*, vol. 12, no. 1-2.
https://doi.org/10.1504/IJRM.2021.114970

Digital Object Identifier (DOI):
10.1504/IJRM.2021.114970

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
International Journal of Revenue Management

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 17. May. 2022

Application of Machine Learning Techniques in Railway Demand Forecasting

November 13, 2019

Abstract

Demand forecasting lies at the heart of any revenue management system. It aims to estimate the quantity of a product or service that will be purchased in the future. In this paper, we perform railway demand forecasting for a major European railroad company by taking various contributing parameters into account. Using state-of-the-art machine learning methods and various heuristic feature construction techniques, remarkable results with high forecast accuracy and reasonable computational complexity are achieved. To have multipurpose results, the current problem is explored in two different aggregation levels. Although this paper is focused on demand forecasting in railway industry, the studied methodologies can easily be extended to other transportation or hospitality businesses.

Index Terms— Revenue Management, Demand Forecasting, Feature Engineering, Machine Learning

1 Introduction

Revenue Management (RM) is the application of various analytical tactics and mathematical approaches with the aim of predicting customer behavior at the micro-market level while optimizing price and availability of products [1]. RM tasks are shaped by various components including customer segmentation, demand forecasting, pricing techniques and inventory control management.

Demand forecasting plays a vital role in any traditional revenue management system. All the models aiming to answer the question "How to determine the most efficient capacity allocation and pricing decisions?"rely on the predicted demand as the main building block of an RM system. As emphasized by McGill et al. [2], all RM decisions are made based on different forecasts, particularly, customer demand which provides input data for the capacity and pricing optimization module. Forecasting future demand is a complicated task due to uncertainties caused by the firm's decisions and external factors [3].

Companies can improve the quality of their pricing and capacity control systems by increasing the accuracy of their predicted demand. Over the years, a fundamental collection of forecasting methods has been developed and new improvements have continued to

evolve. Some of these forecasting methods are based on solid mathematical and statistical foundations while some others are largely heuristic in nature. In terms of forecasting methods, since a large number of forecasts have to be made during a limited time period; thus, fast, accurate and simple methods are preferred in RM [4].

One of the early works on statistical demand forecasting in airline industry using time series data was done by Sen [6]. Since then, there have been numerous "time series analysis "-based studies with the aim of improving the forecast accuracy and achieving more stable and generalizable models [7], [8] and [9], [10]. A well-known and extensively explored time-series analysis method is AutoRegressive Integrated Moving Average (ARIMA) [5], which is a generalization of AutoRegressive Moving Average (ARMA) to non-stationary data.

Over the years, various booking models have been explored, models such as pickup, advanced pickup and booking profile which are based on registered bookings over time, and can be of the additive or multiplicative type [11]. More detailed information on these models could be found in the literature [12], [13], [14]. Simple and weighted averages are also among the popular demand forecasting methods which were outperformed by pickup models [15]. Cleophas et al. [34] summarized recent developments in demand forecasting for airline revenue management.

One of the recent categories of models addressing demand prediction in RM is Machine Learning (ML). ML methods [37] are mathematical tools with the core objective of learning to generalize from experience. They mainly rely on the underlying patterns and characteristics of historical data in order to minimize prediction errors of unseen data. In general, ML algorithms are classified into two main categories: supervised learning and unsupervised learning. The goal of supervised learning is to infer a functional mapping according to a set of input-output training examples. Unsupervised learning, on the other hand, discovers patterns and structures hidden in data without having access to labeled output.

Classical statistics-based methods, such as time series, may struggle to cope with high-dimensional data sets and sometimes fail to respond accurately to sudden changes. Machine learning methods, however, are more flexible when dealing with sudden changes in the format of data, missing information, and high-dimensional data sets [11].

Demand forecasting, as a regression prediction problem, has been also studied extensively with the help of various ML techniques. For instance, Ziekow et al. [16] used ML methods to evaluate the use of disaggregated smart home sensor data for household-level demand forecasting. ML methods are also used for urban water demand forecasting in situations with limited data availability. These methods were tested using three years of daily water demand and meteorological data for the city of Calgary in Alberta, Canada [17]. A thorough review paper on the application of machine learning models to commercial building electricity load forecasting was published by Yildiz et al. [18].

Booking demand forecast is also one of the crucial decision-making challenges in service industries which is extensively studied through ML techniques. In an interesting study, Sanz-Garcia et al. [19] developed a hybrid method to estimate hotel room reservations that explores the effects of last-minute reservations. A very recent study on hotel reservation management has been published by da Conceicao Antonio [20] as his PhD thesis, which emphasizes on using ML to predict booking cancellations. With a focus on

service companies, Shadi Sharif et al. [21] analyzed and categorized various statistical and ML techniques used for demand forecasting in revenue management.

In this paper, we forecast the future number of bookings for a major railroad company by taking various contributing factors into account. In order to do so, we use different ML approaches along with heuristic feature engineering techniques.

Forecasting is a complex task, however, it can be broken down into simpler steps. We perform our forecasting task in two aggregation levels. These levels are created based on the zonal data and used to demonstrate the overall performance of the prediction models. At each level, the forecast is considered good if it is accurate, plausible, simple, quick and flexible.

Overall, this research intends to contribute to the application of ML techniques in RM. More specifically, it addresses the problem of demand forecasting in revenue management by proposing:

- new heuristic feature engineering techniques including shallow and deep features,
- exploring the importance of accurate clustering and its integration into data, and
- implementing state-of-the-art machine learning methods in order to discover complex hidden patterns of data and improve the accuracy of predications.

The remainder of this paper is organized as follows: Section 2 provides problem description in course of which general definitions and problem settings are explained in detail. In Section 3, we discuss the details of various types of preprocessing, machine learning and feature engineering techniques which will be used in this paper. In Section 4, the numerical results of applying such methods in both aggregation levels of data are demonstrated and analyzed. Finally, the concluding comments are outlined in Section 5.

2 Problem Definition

We start with introducing some technical terms and general definitions which will be used throughout this paper. Afterwards, we will go through the details of demand forecasting problem in the context of railway industry.

- Market: an origin-destination pair between which the passengers wish to travel
- Itinerary: a specific sequence of legs on which passengers travel from their origin to their ultimate destination
- DBD: number of days before departure
- Booking horizon: time horizon in which bookings are open
- Booking period: booking horizon between each two subsequent DBDs

- Time-range: a predefined time horizon during the departure day which is an aggregation of departure times
- Demand: expected demand of a product in a market which depends on itinerary, time range and period
- Fare Class: different prices for the same itinerary, usually distinguished from one another by the set of restrictions that firms impose
- Product: an itinerary and fare class combination

To have a reliable multipurpose forecast, we treat the data in two different aggregation levels. One of the main reasons for forecasting the potential demand in different levels is to meet the railroad company's specific needs. For instance, for overall planning of all trains, it is sufficient to have a less detailed estimated values. However, it is necessary to perform a more comprehensive forecast for inventory control and pricing purposes.

In this problem setting, DBDs are defined as of 120 days before departure date. The period between DBD119 and DBD-1 (*i.e.,* departure date) is divided into 20 booking periods. Note that booking periods are not necessarily of the same length. In the beginning of the horizon, booking periods consist of several days; however, they become shorter as we get closer to the departure date and get as short as one day within the last few days before the departure day.

Level I

This level provides an overall view of the data. In this top-level, the historical booking information are aggregated by booking periods. We would like to forecast the total number of bookings for all trains departing on a specific departure date and within a certain time range. For example, the illustrated area in Figure 1 presents the total number of bookings that we aim to forecast for a given departure date in the time range of 7:00 am - 9:00 am.

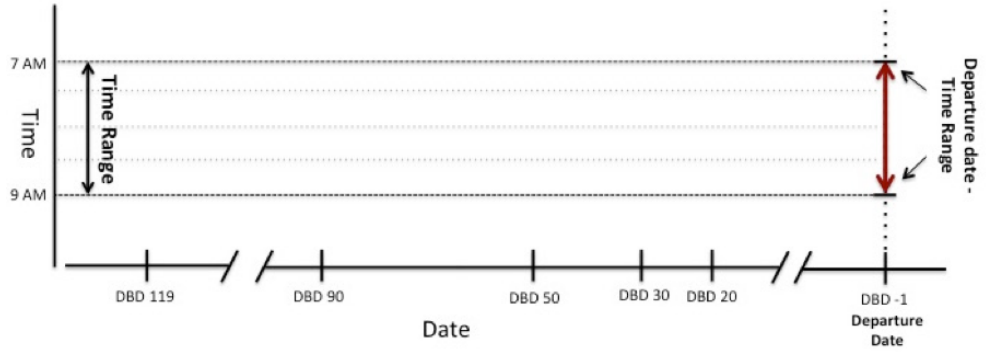

Figure 1: Demand forecasting in Level I

Level II

In level II, we add the dimension of booking period to level I data. Consequently, in this level, the prediction models aim to compute the total number of bookings within each booking period for all trains leaving in a specific time range of a certain departure date. Figure 2 shows the total number of bookings for a given departure date in the time range of 7:00 am - 9:00 am that was particularly booked in the booking period between two consequent DBDs of 90 and 50.

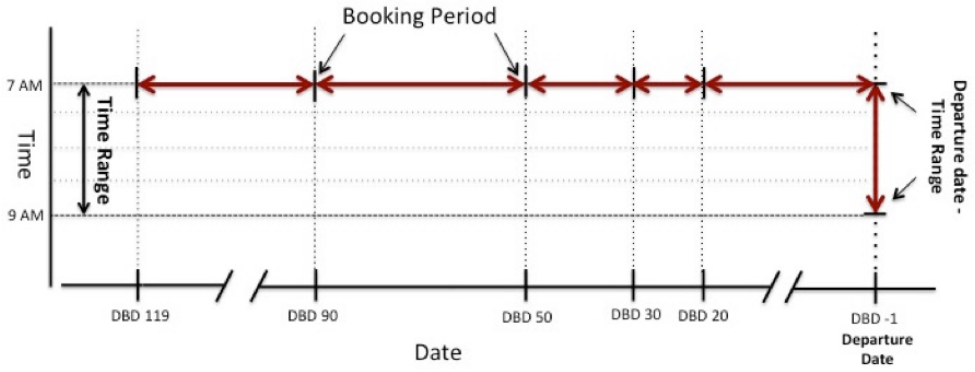

Figure 2: Demand forecasting in Level II

3 Solution Methods

In this section, we explain the details of preprocessing steps, model selection process and feature engineering techniques used for railway demand forecasting. Our dataset consists of two years of historical booking data (*i.e.,* 2013 and 2014) collected from travels between two major European cities, which is provided to us by a major European railroad company. The objective is to predict the potential demand of the future bookings based on the historical data in two different aggregation levels.

3.1 Preprocessing

Data preprocessing is a process to transform raw data into a format that is usable as an informative input to predictive models. Industrial datasets usually require various steps of preprocessing such as data cleaning (*e.g.,* missing values imputation and noisy data smoothing), data transformation (*e.g.,* normalization and aggregation), data reduction, and data discretization.

We start data preprocessing with verifying data type and data representation consistency. Our initial raw dataset consists of a few attributes including departure date and time, booking periods and potential period demand with no missing values.

We extract useful information from the departure date feature and construct new attributes; namely, month, week number, week day and date value. The first three provide us with intuitive and valuable knowledge regarding departure dates. Moreover, the date value is a numerical representation of the departure date that shows the number of seconds since 1970. This method of representation helps us to preserve intervals and keep the order of events.

As the next step, we perform data discretization, a method to reduce the number of values of a continuous feature by dividing it into predefined number of intervals. In this step, we split each departure day into six time ranges based on the popularity of departure times (*e.g.,* 19:00-23:59 is one of the time ranges).

Many machine learning models require numerical values as their input, and this translates into the necessity of transforming categorical features into numerical ones. Depending on the nature of the categorical data, we have various options to do so such as one hot encoding and integer number assignment.

One hot encoding is one of the most well-known encoding schemes used to transform a single categorical variable into its corresponding binary variables. Each binary variable takes "1"when its associated category is present, and "0"otherwise [23].

For example, categorical features such as month and weekday could be integrated into the data using one hot encoding technique. However, it may result in dimension augmentation. This should not be problematic since we have a large enough number of samples to avoid overfitting. The categorical booking periods attribute; however, is different as there is an order in it which allows for assignment of integer numbers.

As an initial clean format of data available to us, we can perform an outliers detection algorithm now. Outliers are extreme perturbations in the data caused by occasional unpredictable events. Outlier detection is considered an extremely important step since outliers can impose remarkable noise on the mean and variance of the entire dataset and distort the real pattern of the data. In this study, we use modified Z-score to detect and then remove extreme outliers.

Z-score or standard score [36] discovers by centering and rescaling of data, and then, detecting the points that are far from the mean. When using mean and standard deviation themselves are directly affected by outliers, this method is not robust enough. In modified z-score, although the intuition is the same, we use the median and Median Absolute Deviation (MAD) to measure central tendency and dispersion, respectively. Thus, modified z-score turns out to be a more robust method in terms of detecting outliers [22].

Upon completion of this step, the data is ready for further analysis.

3.2 Model Selection

We start this section with a brief review of ML models that will be extensively used in this paper. Afterwards, we explain the model selection process for level I and level II data. Finally, we describe the feature engineering process for level II data.

3.2.1 Model Description

Many tasks in machine learning can be expressed as a classification or a regression problem. Regression estimates the conditional expectation of a dependent variable given the independent ones whereas classification predicts categorical class labels.

The simplest regression model is linear regression which is capable of capturing linear relationships between predictors and target, but we mainly deal with more complex and nonlinear tasks such as demand forecasting in the real world.

In general, tree-based models and Neural Networks (NNs) are two main categories of models used for demand forecasting in the literature. Both of them are supervised learning methods used for regression and classification purposes. The intuition behind NNs is to extract linear combinations of inputs as derived features, and then, to model the target as a non-linear function of those features [24].

A Decision Tree (DT), as a building block of any tree-based method, is a decision making tool that uses a tree-like model to estimate the value of a target variable by learning simple decision rules deduced from data attributes. DTs are the foundation of very powerful predictive models such as tree-based bagging and boosting ensemble models [37].

Bagging (bootstrap aggregating) is an ensemble averaging meta-algorithm that improves the prediction accuracy by reducing variance [32]. In decision tree-based bagging, each bootstrapped sample is used as a training set to grow a decision tree, and the result is the average over the predictions of all trees. Random forests, also known as random decision forests, is a practical ensemble method designed based on the bagging idea [26].

Boosting methods are built sequentially over weak regressors in order to reduce the bias [33]. The final meta-algorithm is a linear weighted combination of the base estimators with a reduced generalization error. One of the most well-known boosting models is Gradient Boosting Trees (GBT) which is a generalization of boosting to arbitrary differentiable loss functions using decision trees as base estimators [25].

3.2.2 Model Selection

In this section, we describe the model selection process in the level I and the level II data.

Level I

We start model selection with the level I data. Note that the data used in this level are already preprocessed, the outliers have been removed and the basic features such as time range added to ensure improved performance of any ML method we may choose. Having a top level aggregated data, we expect that application of a proper ML method will result in an acceptable performance.

Once we applied various ensemble tree-based and neural networks methods on level I data, we achieved sufficiently good results with both NNs and GBT. Since the acquired performance was considered to be efficient according to the industry's guidelines, we focused on level II as a more detailed and complex level.

Level II

We start model selection of level II with evaluating various regression methods in order to compare their performances and achieve a benchmark for more advanced techniques. On this level, among the initially tested methods, GBT outperforms others.

For this aggregation level, considering the fact that the data are more complex, the common ML regressors do not improve the results more than a certain limit. Thus, various combinations of different models such as regular and weighted mixture of regressors are explored. Among these models, stacking, also known as stacked generalization, provides the most accurate predictions while keeping the processing time in a reasonable range.

Stacking is a meta-learner that consists of multiple model mixtures. The idea is to learn a function that combines the predictions of the individual regressors and feed them as input into the final meta learner. This method was originally introduced in 1992 by David H. Wolpert [27] for a classification task. Algorithm 1 represents the general approach to a regression task.

Algorithm 1 Stacking algorithm

1: Train n different regressors $R_1, R_2, \ldots, R_n$ (the base regression models).
2: Obtain predictions of each regressor.
3: Form a new dataset using predicted values: the meta data.
4: Train a separate regressor on the meta data: the meta regressor.

In summary, each base regressors prediction is used as a new feature in the meta dataset (*e.g.*, data fed into the meta regressor), then, a meta regressor is applied to the meta data. Figure 3 illustrates an schematic view of stacking algorithm, where, $R1, R2, \ldots, R5$ refer to base regressors.

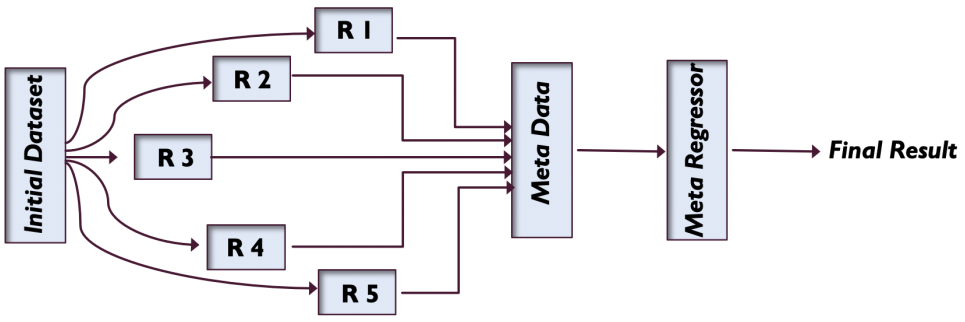

Figure 3: An schematic view of stacking algorithm

Although stacking outperforms all other examined regression methods, the high complexity of level II data necessitates further accuracy of forecasting results to ensure that the obtained predictions are a useful source of reliable information for industrial purposes.

3.3 Feature Engineering

Feature engineering is the process of using domain knowledge to design attributes that improve the performance of machine learning algorithms. We start with shallow features, which consider shallow characteristics hidden in the data and can be easily extracted from the dataset. Deep features, on the other hand, are the ones that require an algorithmic approach to be constructed, and unlike the shallow ones, they aim to capture deeper characteristics of the data.

Note that we perform feature engineering only on level II data since we have already achieved desirable results on level I thanks to selecting proper ML methods.

Shallow Features

At this step, shallow features are explored. Since ML algorithms are designed to capture hidden characteristics of the data, having more informed attributes will increase the possibility of discovering such characteristics.

By using shallow features, we intend to capture the trends of bookings for each departure date starting as of 120 days before the departure date. We define an observation date as the date on which we are observing a snapshot of the bookings made so far for all the departure dates. An observation date can be any day during the year.

Four new features are also constructed in addition to the observation date attribute. We divide the 120-day booking window prior to each departure date into four 30-day periods and dedicate one attribute to every single one of them. Each attribute indicates the total number of bookings made during its associated month. The observations occur every seven days that means our data are updated on a weekly basis.

For example, if a departure date is July 1st, we will have four new features for the bookings made during March, April, May, and June, separately. If our observation date is some time prior to March 1st, the features will have zero values because the booking has not started for this specific departure date yet.

By moving forwards within the booking window, the booking information will be updated every seven days. For instance, having an observation on April 7th means we have full information regarding total bookings made in March for the departures on July 1st. However, at this point, only one week of data is available for the month of April, and the rest of attributes (*i.e.,* one and two months prior to the departure date) are zero.

These features extracted valuable information from our dataset and improved the accuracy of demand estimation.

Another category of shallow features is external features. We gathered some external data that may affect bookings such as weather abnormalities in the departure and the arrival cities (*e.g.,* snow storm, extreme heat warning, flooding, etc.) along with max, min, and mean temperatures on the departure day. Although weather forecast might not be accurate long ahead of the departure date, it is still an easily accessible data and can be updated anytime needed.

Deep Features

We realized that having proper clusterings of data, as an added feature, would reduce the forecast error significantly. To examine this assumption, K-means clustering is applied to the dataset while having access to the target variable, and the cluster labels are added as a new feature to the whole dataset. This results in achieving the lowest bound of error and validates our initial assumption.

K-means [35] is a simple unsupervised learning method that clusters unlabeled given points into K-predefined number of clusters using an expectation-maximization algorithm. Starting with randomly defined K centroids, the data points are assigned to the closest centroids so that each group creates a cluster. At the next step, the model recalculates K new centroids and assigns the closest points to them, resulting in a new set of K clusters. This loop runs until centroids do not move anymore.

In order to find the optimal number of clusters, the K-means algorithm is pipelined to a supervised regression method (Any appropriate regression method is applicable at this step, including random forests.) for evaluating the errors of each value of K. The "elbow method"is used to find the optimal K.

To take advantage of this finding in regular settings, where the target variable is not provided, the following steps are taken. First, using K-means method, the train set is clustered into k predefined clusters and the obtained labels are considered as a new target variable for the train set. Afterwards, test data are classified into the same cluster labels. Finally, the cluster numbers are added to the original test and train sets and we proceed with the regression problem. Note that in step 4, the accuracy of classification task is as important as that of the final regression problem and it can be performed using various ML techniques such as stacking.

Algorithm 2 displays the steps of generating a clustering-based feature.

Algorithm 2 Clustering-based Feature Construction

1: Cluster train data into K predefined clusters using K-Means algorithm.
2: Modify train set by removing the demand feature from the dataset.
3: Consider cluster labels as the new target variable in the train set.
4: As a classification task, classify test data into the same K cluster labels.
5: Use the predicted labels as a new feature in both original train and test sets.
6: Apply a regression model (*e.g.,* stacking) to the dataset with the new added feature.

In the next section, we provide numerical results associated with the selected models and added features.

4 Numerical Results

We start this section with a general analysis of data. Afterwards, we explain the choice of error evaluation metrics. As the next step, we report the results of various demand

forecasting techniques used to predict the number of bookings in different aggregation levels of data: level I and level II.

The computational operations have been carried out on a 2.9 GHz 5-core computer with 16 GB of RAM and the codes are written in Python 3.5. Moreover, we use 2013 dataset for training and validation, and 20% of the data from the first quarter of 2014 dataset is considered to be the test set.

In this study, we focus on tree-based methods and neural networks as two common categories of regression methods to tackle the demand forecasting problem. The ensemble tree-based methods we explore are either bagging methods such as Random Forests (RF) [26] and Extremely Randomized Tress (ERT) [29] or boosting methods such as Gradient Boosting Trees (GBT) [25] and AdaBoost [30].

Evaluation Metrics

Calculating demand forecast accuracy is a process of determining the accuracy of predicted demand compared to actual customer demand. In this paper, we use both Weighted Average Percentage Error (WAPE) and Root Mean Square Error (RMSE) metrics to measure the accuracy of our demand forecasting methods.

WAPE is the quotient of the sum of the absolute deviations divided by the total actual demand. The equation is as follows:

$$WAPE = \frac{\sum_{i=1}^{n} |F_i - A_i|}{\sum_{i=1}^{n} A_i} \times 100 \tag{1}$$

where, A and F represent actual and predicted values of demand, respectively. In general, WAPE is easy to understand and interpret because it measures the error in the percentage format. Moreover, since the denominator is a sum over all actual values, WAPE is capable of handling small or zero actual demands. This is an important feature in our case since in some circumstances we have actual demand of zero.

RMSE, on the other hand, is the standard deviation of the residuals.

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (F_i - A_i)^2} \tag{2}$$

Here again, A and F denote actual and predicted values of demand, respectively. As the above equation indicates, since the residuals are squared before they are averaged, the RMSE gives a moderately high weight to large errors. Consequently, when large errors are particularly undesirable (*e.g.,* in demand forecasting tasks), RMSE could be the evaluation metric of the choice.

Data Analysis

As data visualization is a critical tool for data analysis, we provide some useful insights into the structure and patterns of our dataset.

Figure 4 shows the actual cumulative demand for each time range as we start from the initial booking period of DBD119-DBD90 and end on the departure date DBD0-DBD-1. Obviously, bookings increase significantly as we get closer to the departure date.

This figure also illustrates the differences in the customers' booking behaviors in various time ranges. The time range of 16:00 - 19:00 is the most popular one and 09:00 - 12:30 is the second most demanded option. That is, throughout the year, most on-demand travels happen either in the evening or right before noon.

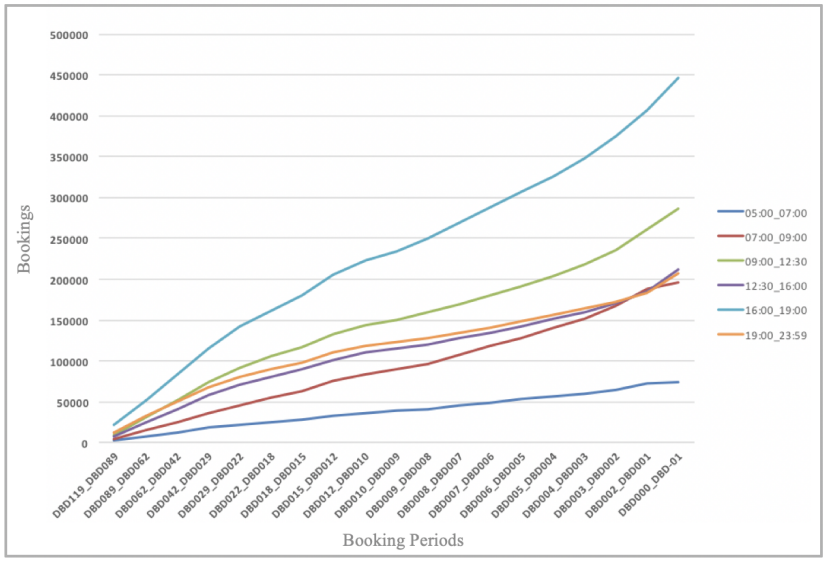

Figure 4: Total actual cumulative bookings by booking period and time-range

The trend in the number of bookings based on booking periods and time ranges is illustrated in Figure 5. Note that only the data of one month is displayed for the sake of clarity and simplified representation. This graph shows a meaningful difference between the number of bookings occurred in early morning compared to the rest of the day.

During the illustrated month, in almost all time ranges, the bookings follow a similar pattern: starting low in booking demand, reaching the first peak within the initial couple of booking periods, breaking the trend and hitting the minimum about a week before the departure date, and finally raising and reaching the highest number of bookings on the exact departure date.

The picks of this pattern can be explained by the price-sensitive and time-sensitive customers' behavior: price-sensitive customers tend to purchase their ticket much earlier in the booking horizon and time-sensitive customers (*i.e.,* business travelers) book closer to the departure date.

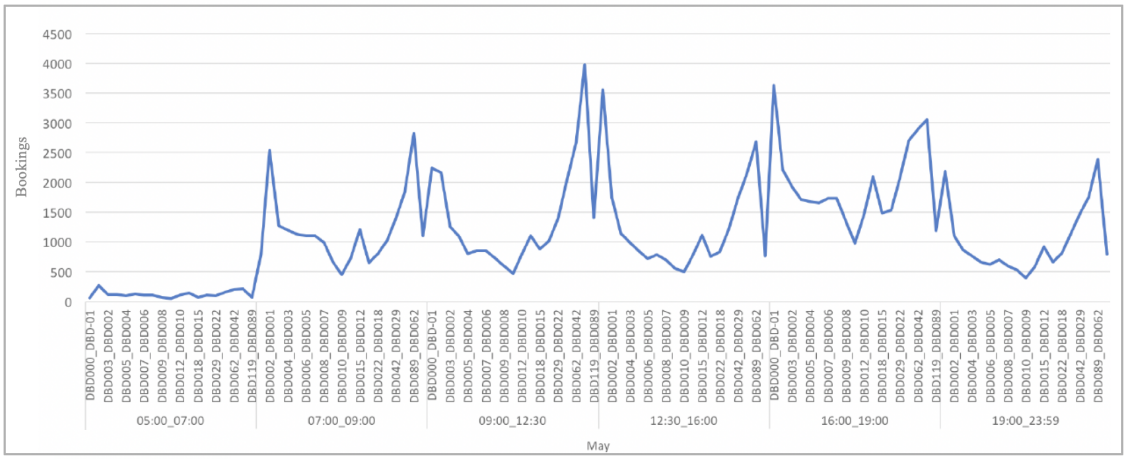

Figure 5: Total actual bookings trend by booking period and time-range (May).

Results of Level I Data

After preprocessing step and data analysis, we explore various ML methods on level I data including various ensemble tree-based methods and NNs. Among them, GBT and NNs outperform others. Although GBT achieved only slightly more accurate results than NNs did, we pick GBT as the best performing model.

Industry-wise, there should be a trade-off between the time and efforts dedicated to tuning a model on the one hand, and robustness and accuracy of the results on the other hand. Having almost the same level of computational complexity and accuracy, the NN requires very precise tuning to provide the same results as GBT does. Overall, GBT is more robust and generalizable for the purpose of this problem.

The experiments show that in our case study, ERT is competitive with NNs but not as accurate as GBT. Table 1 provides the results of applying all learning methods to level I data. As illustrated, with a WAPE test result of 10.21%, GBT outperforms other models. Note that the processing time for all applied methods of level I is around one minute. Thus, processing time is not a contributing factor in model selection at this step.

Considering various factors, including acceptable accuracy, computational complexity and robustness, GBT satisfies the company's requirements and we explore the more complex data of level II in the next section.

Results of Level II Data

On this level, we evaluate the performance of the same ensemble tree-based and NNs methods as in level I data. Here again, we observe that GBT outperforms other regressors. However, the achieved results are much less accurate with a higher percentage of the errors due to finer level of aggregation and increased level of data complexity. We use the results of this step as a benchmark to compare with those of the future steps.

Table 1: Results of learning methods on level I data

Algorithm	WAPE Test %	WAPE Train %	RMSE Test	RMSE Train
NNs	10.78 %	7.14 %	96.78	72.10
AdaBoost	12.18 %	8.30 %	119.26	91.08
ERT	11.65 %	8.13 %	105.76	86.21
RF	13.46 %	9.89 %	141.70	100.31
GBT	10.21 %	7.12 %	91.03	69.85

Table 2 provides results of applying all learning methods to level II data after initial preprocessing and outlier removing. As demonstrated, at this level, the boosting methods (i.e., AdaBoost and GBT) are very competitive in terms of accuracy.

Table 2: Results of learning methods on level II data

Algorithm	WAPE Test %	WAPE Train %	RMSE Test	RMSE Train	Processing Time (min.)
NNs	37.54 %	35.86 %	19.31	18.10	2.40
AdaBoost	36.49 %	36.31 %	18.87	18.61	3.54
ERT	38.88 %	37.04 %	20.52	19.11	2.15
RF	40.11 %	38.76 %	22.37	20.19	2.04
GBT - BENCHMARK	35.90 %	35.08 %	18.83	18.51	3.34

Having the benchmark and best performing regressor at this step, we can explore the effects of outlier removal that we used as the last step of preprocessing section. That is, we apply GBT on fully preprocessed data except for the outliers removal, and then, we compare the results to the ones achieved after detecting and removing the outliers. Table 3 shows performance improvement of GBT as a result of outliers removal.

Table 3: GBT result improvement due to outliers removal

Algorithm	WAPE Test %	WAPE Train %	RMSE Test	RMSE Train
GBT - Before outliers removal	51.03 %	50.49 %	27.65	27.28
GBT - After outliers removal	35.9 %	35.08 %	18.83	18.51

At the next step of the model selection process, we examine the effects of combining various regressors. The motivation behind this idea is that methods like averaging helps

with decreasing the prediction error by reducing variance. To do so, we start with computing predicted demand using selected regressors. At the next step, we average over the predicted values. As an example, Table 4 shows the error reduction using averaging over three regressors: RF, GBT, and ERT.

Table 4: GBT result improvement due to averaging over predictions of regressors

Algorithm	WAPE Test %	WAPE Train %	RMSE Test	RMSE Train	Processing Time (min.)
GBT - BENCHMARK	35.9 %	35.08 %	18.83	18.51	3.34
Mixture of Regressors	29.01 %	28.56 %	15.61	15.32	7.58

Weighted averaging is another technique used to reduce errors; however, the slight accuracy improvement depends on precise weight assignment. The preference is to find a more robust method of a noticeably higher performance.

According to Table 5, stacking results in a remarkably better performance by decreasing the WAPE test result to 23.76%. Although it requires longer processing time, it is still much lower than the industrial computational complexity limits.

Table 5: Performance improvement as a result of applying stacking to level II data

Algorithm	WAPE Test %	WAPE Train %	RMSE Test	RMSE Train	Processing Time (min.)
GBT - BENCHMARK	35.9 %	35.08 %	18.83	18.51	3.34
Mixture of Regressors	29.01 %	28.56 %	15.61	15.32	7.58
Stacking	23.76 %	23.99 %	13.01	12.98	11.31

The main challenge in improving the results of stacking algorithm lies in the choice of base estimators and the meta regressor. This issue is still referred to as "black art"in the literature. Although there have been some attempts to automate this process or to define a criterion for selection process, no satisfying method has been developed so far [28].

In this study, we use GBT, ERT, Linear regression, RF and KNN as base regressors and RF as the meta regressor. Note that KNN stands for K-Nearest Neighbors [31] and is a simple yet powerful supervised learning method for both classification and regression tasks. In a regression problem, given an example to predict, KNN performs based on finding K most similar examples from the training data, called nearest neighbors, and estimates its value as an aggregation of the target values associated with its nearest neighbors.

In stacking, each one of the implemented regressors contributes to capturing underlying characteristics of data using different methodologies. Thus, having various categories of regressors increases the overall performance of stacking method. For example, we include ensemble tree-based (both boosting and bagging), linear regression and K-nearest neighbors methods in the base regressors.

We explored other various innovative methods in order to combine different regressors such as weighted stacking and double stacking. In weighted stacking, the predictions of each base regressor (*i.e.,* features of the meta data) are given normalized weights. To assign weights automatically, we use the feature importance characteristic of RF. When applied to data, RF is capable of assigning importance values to each feature within the dataset. In double stacking, we consider meta data as an initial dataset for another stacking method. In both cases, we decided to ignore the negligible improvements to keep the method simple for industrial purposes and avoid increasing the processing time.

Table 6 demonstrates the results of applying shallow features to our dataset. Although the accuracy increase using shallow features is not as high as when stacking is used, this is a single step of the feature engineering process and we expect that the overall combination of shallow and deep features lead to satisfying final results.

Table 6: Performance improvement as a result of applying shallow features to level II data

Algorithm	WAPE Test %	WAPE Train %	RMSE Test	RMSE Train	Processing Time (min.)
GBT - BENCHMARK	35.9 %	35.08 %	18.83	18.51	3.34
Mixture of Regressors	29.01 %	28.56 %	15.61	15.32	7.58
Stacking	23.76 %	23.99 %	13.01	12.98	11.31
Shallow Features	21.24 %	20.18 %	12.25	11.64	24.30

As explained in Section 3.3, accurate clustering has a significant role in improving the performance of demand forecasting. To validate the clustering effect experimentally, we clustered our dataset into $K = 10$ groups using K-means clustering method while having access to the actual target variable.

Table 7: Accuracy improvement as a result of adding optimal clustering feature

Algorithm	WAPE Test %	WAPE Train %	RMSE Test	RMSE Train
Stacking	35.9 %	35.08 %	18.83	18.51
Stacking with optimal clustering	9.65 %	9.66 %	4.76	4.74

Note that this experiment is performed separately and exceptionally in order to validate the assumption about the importance of clustering-based feature, and it was not integrated into any parts of our actual methods. As a result of this experiment, the forecast test error dramatically drops to below 10% WAPE, which validates our initial assumption. The comparison of stacking results before and after adding optimal clustering feature is demonstrated in Table 7.

We constructed the deep feature based on this finding. The forecast accuracy improvement using deep feature is demonstrated in Table 8. Using clustering-based feature, we

successfully reduced the WAPE test and train results to 18.78% and 16.92%, respectively.

Table 8: Performance improvement as a result of applying deep feature to level II data

Algorithm	WAPE Test %	WAPE Train %	RMSE Test	RMSE Train	Processing Time (min.)
GBT - BENCHMARK	35.9 %	35.08 %	18.83	18.51	3.34
Mixture of Regressors	29.01 %	28.56 %	15.61	15.32	7.58
Stacking	23.76 %	23.99 %	13.01	12.98	11.31
Shallow Features	21.24 %	20.18 %	12.25	11.64	24.30
Deep Feature	18.78 %	16.92 %	11.31	10.62	36.15

In summary, using various preprocessing, machine learning and feature engineering techniques, we reduced WAPE test result from the benchmark of 35.9% to 18.78% which is a remarkable result for this aggregation level. Meanwhile, we succeeded to keep the processing time within the acceptable range according to the transportation company's time constraints and limitations.

5 Conclusion

Demand forecasting is a crucial part of any traditional revenue management system. In this research, we addressed the problem of demand forecasting in the context of railway industry.

To gain multipurpose forecast information, we tackled this problem using two different aggregation levels of data: level I and level II, with the former being the top level that provided an overall view of the data and the latter a more complex level because of having an additional dimension of booking period. Dealing with lower degree of complexity in level I, we achieved our desired results by performing proper preprocessing steps and applying ensemble tree-based methods.

In level II, however, outstanding results were obtained by combining a wide variety of preprocessing, machine learning and feature engineering techniques. We not only used various state-of-the-art machine learning methods, but also developed two different types of heuristic features; namely, shallow features and deep features. The former aims to discover shallow characteristics of data, while the latter is dedicated to extract more complex information.

We realized that having proper and accurate data clusters as features could significantly reduce the forecast error. The deep attribute is mainly constructed based on this discovery. We successfully reduced the forecasting test WAPE result of level II data from the benchmark of 35.9% to 18.78%, while keeping the processing time and overall model performance in a reasonable range.

References

[1] Chase Jr, Charles W. "Revenue management: a review." The Journal of Business Forecasting 18.1 (1999): 2.

[2] McGill, Jeffrey I., and Garrett J. Van Ryzin. "Revenue management: Research overview and prospects." Transportation science 33.2 (1999): 233-256.

[3] Zeni, Richard H. Improved forecast accuracy in airline revenue management by unconstraining demand estimates from censored data. Universal-Publishers, 2001.

[4] Talluri, Kalyan T., and Garrett J. Van Ryzin. The theory and practice of revenue management. Vol. 68. Springer Science and Business Media, 2006.

[5] Box, George EP, et al. Time series analysis: forecasting and control. John Wiley and Sons, 2015.

[6] Sen, Ashish. "Examining air travel demand using time series data." Journal of Transportation Engineering 111.2 (1985): 155-161.

[7] Devoto, Roberto, C. Farci, and F. Lilliu. "Analysis and forecast of air transport demand in Sardinia's airports as a function of tourism variables." WIT Transactions on The Built Environment 60 (2002).

[8] Lim, Christine, and Michael McAleer. "Time series forecasts of international travel demand for Australia." Tourism Management 23.4 (2002): 389-396.

[9] Taylor, James W. "Short-term electricity demand forecasting using double seasonal exponential smoothing." Journal of the Operational Research Society 54.8 (2003): 799-805.

[10] Garcia-Ascanio, Carolina, and Carlos Mate. "Electric power demand forecasting using interval time series: A comparison between VAR and iMLP." Energy Policy 38.2 (2010): 715-725.

[11] Sharif Azadeh, Shadi. Demand Forecasting in Revenue Management Systems. Diss. Ecole Polytechnique de Montreal, 2013.

[12] Zakhary, Athanasius, Neamat El Gayar, and Amir F. Atiya. "A comparative study of the pickup method and its variations using a simulated hotel reservation data." ICGST international journal on artificial intelligence and machine learning 8 (2008): 15-21.

[13] Mishra, Shankar, and V. Viswanathan. "Revenue management with restriction-free pricing." Proceedings of the AGIFORS Revenue Management and Distribution Study Group Meeting. 2003.

[14] Gorin, Thomas Olivier. Airline revenue management: Sell-up and forecasting algorithms. Diss. Massachusetts Institute of Technology, 2000.

[15] Ja, S., B. Rao, and S. Chandler. "Passenger recapture estimation in airline RM." Presentation at 41st AGIFORS Symposium. Sydney, Australia. 2001.

[16] Ziekow, Holger, et al. "The potential of smart home sensors in forecasting household electricity demand." 2013 IEEE International Conference on Smart Grid Communications (SmartGridComm). IEEE, 2013.

[17] Tiwari, Mukesh K., and Jan F. Adamowski. "An ensemble wavelet bootstrap machine learning approach to water demand forecasting: A case study in the city of Calgary, Canada." Urban Water Journal 14.2 (2017): 185-201.

[18] Yildiz, Baran, Jose I. Bilbao, and Alistair B. Sproul. "A review and analysis of regression and machine learning models on commercial building electricity load forecasting." Renewable and Sustainable Energy Reviews 73 (2017): 1104-1122.

[19] Sanz-Garcia, Andres, et al. "Application of genetic algorithms to optimize a truncated mean k-nearest neighbours regressor for hotel reservation forecasting." International Conference on Hybrid Artificial Intelligence Systems. Springer, Berlin, Heidelberg, 2012.

[20] da Conceicao Antonio, Nuno Miguel. Hotel Revenue Management: Using Data Science to Predict Booking Cancellations. Diss. ISCTE-IUL, 2019.

[21] Azadeh, Shadi Sharif, Patrice Marcotte, and Gilles Savard. "A taxonomy of demand uncensoring methods in revenue management." Journal of Revenue and Pricing Management 13.6 (2014): 440-456.

[22] Seo, Songwon. A review and comparison of methods for detecting outliers in univariate data sets. Diss. University of Pittsburgh, 2006.

[23] Lantz, Brett. Machine learning with R. Packt Publishing Ltd, 2013.

[24] Schalkoff, Robert J. Artificial neural networks. McGraw-Hill Higher Education, 1997.

[25] Elith, Jane, John R. Leathwick, and Trevor Hastie. "A working guide to boosted regression trees." Journal of Animal Ecology 77.4 (2008): 802-813.

[26] Breiman, Leo. "Random forests." Machine learning 45.1 (2001): 5-32.

[27] Wolpert, David H. "Stacked generalization." Neural networks 5.2 (1992): 241-259.

[28] Sesmero, M. Paz, Agapito I. Ledezma, and Araceli Sanchis. "Generating ensembles of heterogeneous classifiers using stacked generalization." Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 5.1 (2015): 21-34.

[29] Geurts, Pierre, Damien Ernst, and Louis Wehenkel. "Extremely randomized trees." Machine learning 63.1 (2006): 3-42.

[30] Freund, Yoav, and Robert E. Schapire. "A decision-theoretic generalization of on-line learning and an application to boosting." Journal of computer and system sciences 55.1 (1997): 119-139.

[31] Martinez, Francisco, et al. "Time Series Forecasting with KNN in R: the tsfknn Package." The R Journal (2019).

[32] Breiman, Leo. "Bagging predictors." Machine learning 24.2 (1996): 123-140.

[33] Schapire, Robert E. "The boosting approach to machine learning: An overview." Nonlinear estimation and classification. Springer, New York, NY, 2003. 149-171.

[34] Cleophas, Catherine, Michael Frank, and Natalia Kliewer. "Recent developments in demand forecasting for airline revenue management." International Journal of Revenue Management 3.3 (2009): 252-269.

[35] Likas, Aristidis, Nikos Vlassis, and Jakob J. Verbeek. "The global k-means clustering algorithm." Pattern recognition 36.2 (2003): 451-461.

[36] Shiffler, Ronald E. "Maximum Z scores and outliers." The American Statistician 42.1 (1988): 79-80.

[37] James, Gareth, et al. An introduction to statistical learning. Vol. 112. New York: springer, 2013.