International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 2s
DOI: https://doi.org/10.17762/ijritcc.v11i2s.6064
***Article Received**: 28 November 2022 **Revised**: 26 December 2022 **Accepted**: 05 January 2023*

Enhancing Flight Delay Prediction through Feature Engineering in Machine Learning Classifiers: A Real Time Data Streams Case Study

Ms. Shailaja B. Jadhav[1], Dr. D. V. Kodavade[2]
[1]Assistant Professor : Dept. of Computer Engg., Marathwada Mitramandal 's College of Engg., Pune
Research Scholar – Department of Technology, Shivaji University, Kolhapur
Maharashtra - India
msgshalom@gmail.com
[2]Professor, Dept. Of CSE and IT
DKTE's Institute of Engg. and Technology
Ichalkaranji, Kolhapur- India
dvkodavade@gmail.com

Abstract—The process of creating and selecting features from raw data to enhance the accuracy of machine learning models is referred to as feature engineering. In the context of real-time data streams, feature engineering becomes particularly important because the data is constantly changing and the model must be able to adapt quickly. A case study of using feature engineering in a flight information system is described in this paper. We used feature engineering to improve the performance of machine learning classifiers for predicting flight delays and describe various techniques for extracting and constructing features from the raw data, including time-based features, trend-based features, and error-based features. Before applying these techniques, we applied feature pre-processing techniques, including the CTAO algorithm for feature pre-processing, followed by the SCSO (Sand cat swarm optimization) algorithm for feature extraction and the Enhanced harmony search for feature optimization. The resultant feature set contained the 9 most relevant features for deciding whether a flight would be delayed or not. Additionally, we evaluate the performance of various classifiers using these engineered features and contrast the results with those obtained using raw features. The results show that feature engineering significantly improves the performance of the classifiers and allows for more accurate prediction of flight delays in real-time.

Keywords- Feature Engineering, Machine Learning, Classifiers, Real-time Data Streams, Flight Information System.

I. INTRODUCTION

Creating new features from raw data is a vital aspect in the machine learning workflow, by this way the performance of the models can be optimized. In the context of real-time data streams, where the data is constantly changing and the model must be able to adapt quickly, feature engineering becomes especially important. Recent research has focused on various techniques for extracting and constructing features from raw data, including time-based features, trend-based features, and error-based features [1] [2] [3]. These techniques have been applied to a variety of real-time data streams, including flight information systems, financial markets, and social media networks [4] [5] [6].

The use of feature engineering in machine learning classifiers for real-time data streams has been shown to significantly improve the performance of the classifiers and enable more accurate prediction of events such as flight delays and stock price movements [4] [5]. For example, in a study of a flight information system, the authors used feature engineering to improve the performance of machine learning classifiers for predicting flight delays [4].

Feature engineering is a critical step in machine learning, it is the process of creating new information from raw data that can be used to train a model [7]. A variety of techniques can be used for feature engineering, such as time-based, trend-based and error-based. In [8] it was demonstrated that by using engineered features, the performance of classifiers was significantly improved and the prediction of flight delays in real-time was more accurate when compared to results obtained from raw features only.

It is common practice in streaming data to first optimize and engineer features before utilizing data mining algorithms as it can lead to better performance. In the context of flight data, selecting relevant features is a crucial method in machine learning and data mining. By reducing the dataset through

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 2s
DOI: https://doi.org/10.17762/ijritcc.v11i2s.6064
***Article Received**: 28 November 2022 **Revised**: 26 December 2022 **Accepted**: 05 January 2023*

feature selection, important information can be obtained and the performance of the classifier model can be enhanced. Feature selection algorithms can be divided into two types: filter models and wrapper models. Filter models use general characteristics of the data to choose features without the involvement of a learning algorithm. On the other hand, wrapper models use the performance of a pre-determined learning algorithm to evaluate and decide which features to select. Wrapper models tend to identify features that are better suited to the pre-determined algorithm and result in better learning performance, but they also tend to be more computationally expensive than filter models. In situations where the number of features is very high, filter models are often preferred due to their computational efficiency.

In summary, feature engineering plays a crucial role in machine learning for real-time data streams. Using appropriate feature engineering techniques can significantly enhance the performance of machine learning models and increase the accuracy of event prediction.

II. Related Works

Khan and Byun [11] proposed a method for predicting energy consumption that combines feature engineering, using a genetic algorithm to optimize feature selection, and hybrid machine learning. The approach is evaluated on a dataset of energy consumption data from a building and shown to significantly improve prediction accuracy compared to using raw features and a single machine learning model. Potential research gaps include the need for further investigation of hybrid machine learning models and the adaptation of the approach to other types of energy consumption data.

A study by Chia *et al.* [12] found that the combination of machine learning and feature engineering is effective in classifying sarcasm and irony in social media texts and detecting cyberbullying. The research process involves creating new features from raw data and using machine learning algorithms to classify the texts. The results showed that the approach was successful in classifying sarcasm and irony and had high precision and recall rates for detecting cyberbullying. However, there is potential for further research to explore other machine learning algorithms, and to apply the approach to other languages and domains.

Ledezma *et al.* [13] proposed a method for detecting ischemia using ECG data. The method involves generating synthetic ECG data using a mathematical model and applying machine learning algorithms to classify the real and synthetic data as normal or abnormal. The authors extract features from the ECG data using statistical and frequency domain analysis techniques and evaluate the performance of their method using a dataset of ECG data from patients with and without ischemia. The combination of modeling and machine learning is found to be effective for detecting ischemia and potential research gaps include further investigation of the use of synthetic ECG data and the application of the method to other cardiovascular diseases.

Chen *et al.* [14] presented a software platform called "iLearn" for analyzing DNA, RNA, and protein sequence data using machine learning. The platform includes tools for feature engineering and a meta-learner that can automatically select machine learning algorithms. The authors apply iLearn to various sequence datasets and find it performs well in terms of accuracy and efficiency, and is able to automatically select the best algorithms for the datasets. Potential research gaps include further investigation of iLearn on larger and more complex datasets and the extension of the platform to other types of biological data.

Kasongo and Sun [15] presented a method for detecting intrusions in wireless networks using deep learning and filter-based feature engineering. The method involves extracting relevant features from raw data using statistical analysis and using the filtered features as input to a deep learning model. The combination of filter-based feature engineering and deep learning is found to outperform other methods in terms of accuracy and speed on a dataset of network data from a real-world wireless network. Potential research gaps include further investigation of the use of deep learning for intrusion detection in other types of networks and the adaptation of the approach to other types of cyber-attacks.

Fan *et al.* [16] presented an overview of how deep learning can be used to improve building energy prediction. The authors review several feature engineering methods based on deep learning and compare their performance in predicting building energy consumption. The results show that these methods can significantly improve prediction accuracy compared to traditional methods. However, the authors also point out some limitations and areas for further research, including the need for more diverse and representative datasets, the development of more robust and adaptive models, and the integration of these models with other building systems and controls.

In a study by Ullah *et al.* [17], a deep learning approach for recognizing actions in surveillance data streams from non-constant environments was proposed. The authors introduced a hybrid model that merges an optimized deep autoencoder with a convolutional neural network (CNN) to extract features from the data and classify actions. They used an evolutionary algorithm to find the best architecture and hyperparameters for the autoencoder. The proposed model was evaluated on several action recognition datasets and compared to other state-of-the-art methods. The results showed that the hybrid model performed well and was able to handle non-constant environments, where the distribution of actions may change over time. However, the authors also pointed out some

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 2s
DOI: https://doi.org/10.17762/ijritcc.v11i2s.6064
***Article Received**: 28 November 2022 **Revised**: 26 December 2022 **Accepted**: 05 January 2023*

limitations of the current approach and suggested directions for future work, such as improving the robustness of the model to variations in camera views and lighting conditions and integrating the model with other sensors and modalities.

Demir *et al.* [18] examined the use of technical indicators as supplementary features for electricity price forecasting using machine learning models. They compared the performance of linear, ensemble, and deep learning models with and without technical indicators. The research was conducted on a real-world dataset of electricity prices, and the results indicate that incorporating technical indicators can improve the accuracy of the forecasts for all three types of models. However, the authors also emphasized that the results are influenced by the specific indicators used and how they are combined, and more research is required to identify the most useful indicators and how to properly include them in the models. In addition, the authors suggest that future research should investigate the use of more advanced machine learning techniques, such as deep learning, for this application.

Li *et al.* [19] developed a new approach for creating features from surface electromyography (sEMG) data for machine learning. They proposed a hybrid method that combines wavelet transformation and empirical mode decomposition (EMD) to divide the sEMG signals into different frequency bands and extract the intrinsic mode functions (IMFs). The IMFs were then used as features for classification using support vector machines (SVMs). The proposed method was evaluated on a dataset of sEMG signals collected from the brains of healthy subjects, and the results indicated that it outperforms other feature extraction methods in terms of classification accuracy. The authors also acknowledged some limitations of the current approach and suggested directions for future work, such as applying the method to other sEMG applications and using deep learning techniques to enhance the performance of the classifier.

The current level of machine learning for streaming data was reviewed by Gomes *et al.* [20] with an emphasis on the opportunities and challenges present in this area. The authors review the different types of machine learning algorithms that have been applied to streaming data, as well as the various architectures and systems that have been developed to support their execution. They also discuss the main challenges that arise when working with streaming data, such as the need for efficient and scalable algorithms, the handling of concept drift and non-stationarity, and the integration of streaming data with other sources of information. The authors conclude by identifying several directions for future research, including the development of new algorithms and systems that can better exploit the properties of streaming data, the investigation of new applications and domains where machine learning can be used with streaming data, and the exploration of novel techniques for assessing how well machine learning models perform on streaming data.

Zheng *et al.* [21] presented an ensemble learning method that combined the results of multiple feature selection methods using the Dempster-Shafer theory of evidence to improve feature selection performance. The authors proposed this approach to tackle the challenge of high dimensionality of the feature space and presence of noisy and irrelevant features, which are common issues in many machine learning applications. The approach was evaluated on several benchmark datasets and was found to have outperformed other feature selection approaches regards of classification accuracy. Overall, this was a good contribution to the field of feature selection as it demonstrated how ensemble learning can be used to improve feature selection by integrating multiple feature subsets obtained from different feature selection algorithms.

Yuan et al. [22] introduced a technique to handle feature drift in data streams, which happens when the characteristics of the data change over time. They suggested an iterative subset selection method that uses a sliding window to select a subset of features that are most relevant to the current concept in the data stream, this way it can update the feature subset in real-time as the concept changes. The method was tested on several datasets and was found to be better than other state-of-the-art methods in terms of classification accuracy and adaptability to changes in the concept. The study made a significant contribution to the field by addressing the issue of feature drift in data streams and proposing a method that uses a sliding window to select a subset of features that are most relevant to the current concept in the data stream.

III. Problem Statement

Accurate prediction of flight delays is important for optimizing the operation of a flight information system and minimizing the impact on passengers. However, real-time data streams, such as flight information data, are constantly changing and traditional machine learning models may not be able to adapt quickly enough to accurately predict flight delays. The process of creating new features from raw data, known as feature engineering, can potentially enhance the accuracy of predicting flight delays in real-time data streams. The objective of this case study is to examine the impact of feature engineering on machine learning classifiers for real-time data streams, by using a flight information system as a case in point.

IV. Research Objective

1. To investigate the potential of reducing computational power and memory requirements in data stream processing by using selective features through the implementation of feature selection techniques.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 2s
DOI: https://doi.org/10.17762/ijritcc.v11i2s.6064
***Article Received**: 28 November 2022 **Revised**: 26 December 2022 **Accepted**: 05 January 2023*

2. The objective of this study is to investigate the effect of specific features on the time needed for learning in real-time data streams, and to evaluate the possibility of feature selection techniques to accelerate decision-making in data stream analytics.

V. Proposed System Architecture

The proposed system architecture involves a pipeline that processes streaming data, starting with the original feature set. The first step involves applying a CTAO (Clustering, Tracking, and Online Learning) algorithm to the original feature set. The model is updated as new data is received using this approach, which is also used to find patterns and relationships in the data. The second step involves cleaning and preparing the data for feature engineering. This includes removing outliers, normalizing the data, and filling in missing values. The next step is Context Extraction for Extended information retrieval. This step involves extracting additional information from the data, such as context or metadata, that is used to optimize the accuracy of the model or to offer additional insights. Final step in the pipeline is to keep Data ready, where the processed data would be ready for feature engineering.

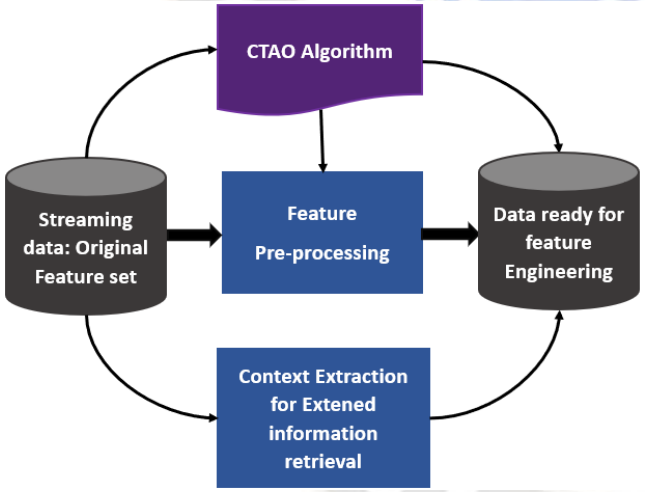


Figure 1. Making Features ready to work on for streaming data

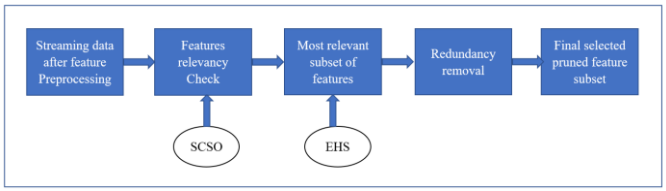


Figure 2: The Feature Engineering process for streaming data

This Feature Engineering process for streaming data involves following steps to extract the most relevant and useful features from the streaming data as shown in the above figure 2.

- The process starts with streaming data after feature pre-processing, where the data has been cleaned and prepared for further analysis.
- Features relevancy Check (by using SCSO - Sand Cat Swarm Optimization): This step includes using an algorithm such as Sand Cat Swarm Optimization (SCSO) to determine the relevance of each feature in the dataset. This can help identify which features are most crucial for the specific task or problem being addressed.
- Most relevant subset of features (by using EHS - Extended Harmony Search): This step includes using an algorithm such as Extended Harmony Search (EHS) to select the most important subset of features from the dataset. This can help identify the optimal subset of features that will provide the best performance for the specific task or problem.
- Redundancy removal: This step includes removing any redundant or highly correlated features from the dataset. This can help to reduce noise and improve the performance of the model.
- Final selected pruned feature subset: The final step is to choose a pruned feature subset, which is a subset of the most important and non-redundant features. This subset of features is expected to be most informative and useful for the task or problem being addressed.

VI. Algorithm Used

Algorithm 1:

function SCSO (population_size, max_iterations)
// Initialization cats = initialize_population(population_size)

- Initialization: A population of sand cats is randomly generated and each cat is assigned a position in the search space.
- Evaluation: The fitness of each cat is evaluated by evaluating the objective function at its current position.
- Movement: Each cat in the population moves to a new position in the search space.
- Selection: The cats with the highest fitness values are selected as parents to generate the next generation of cats.
- Crossover: The parents are combined to generate new offspring.
- Mutation: A small random perturbation is applied to the position of each offspring.
- Replacement: The worst cats in the population are replaced with the offspring
- Return the best solution: The final step is to return the best solution, which is the cat with the highest fitness value.

end function

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 2s
DOI: https://doi.org/10.17762/ijritcc.v11i2s.6064
***Article Received**: 28 November 2022 **Revised**: 26 December 2022 **Accepted**: 05 January 2023*

Algorithm 2:

function EHS (population_size, max_iterations, harmony_memory_size)

// Initialization

harmony_memory = initialize_harmony_memory(harmony_memory_size)

- Initialization: A harmony memory is initialized with a set of randomly generated harmonies.
- Harmony Memory Updating: A new harmony is generated by using the harmonies already in the memory.
- Evaluation: The fitness of the new harmony is evaluated.
- Harmony Memory Updating: If the harmony memory is full, the worst harmony is removed and the new harmony is added to the memory.
- Repeat from step 2: A defined number of iterations or until a stopping criterion is satisfied determines how many times a new harmony is created, evaluated for fitness, and added to the harmony memory.
- Return the best solution: The final step is to return the best solution, which is the harmony with the highest fitness value. The subset of features in this solution that are the most pertinent to the data set as a whole.

end function

VII. Dataset Description

This research used a real-time flight data collected from opensky.org. It contains data for 583987 flights, with each flight represented by a row in the dataset. The dataset originally had 17 features, which include information such as flight number, origin and destination airport, and flight status. In order to extract more context-sensitive information, we have generated some hierarchical features from the baseline features. This means that we have used the original features to create new features that provide additional insights or context. These new features are called extended features. As a result, the dataset used here are 27 features in total, 17 original features and 10 extended features. The extended features are hierarchical features, which means that they are built on top of the original features. These additional features can provide more context-sensitive information and can be useful for feature engineering tasks such as feature selection, feature reduction or feature extraction. It's important to remember that feature engineering is the act of developing features that improve the performance of machine learning algorithms by utilising domain knowledge of the data. So, in order to make the dataset more informative and relevant for the specific task or problem being solved, the extended features in this dataset were produced based on domain knowledge of the data.

VIII. Implementation

All the experiments were run on the system that has enough computational power and memory 11th generation Intel Core i5-11300H processor with a clock speed of 3.10GHz and 16GB of memory has been used. MOA and SKlearn are the software frameworks used to perform the experiments, with MOA being used for stream learning classifiers.

The problem being described is a binary classification problem in which the goal is to classify flights as either delayed (class 1) or not delayed (class 0) based on real-time flight data. The challenge in this problem is the need to train classifiers that can effectively deal with the dynamic and unpredictable nature of the flight data, which may be affected by a variety of factors such as weather, air traffic, and mechanical issues.

Table 1: Performance of classifiers with evaluation parameters

SI. No.	Component to be evaluated	Criteria	Evaluation Parameters
1	Performance of Classifiers	Accuracy	Train accuracy, Test Accuracy, Recall , Precision F1- score
2	Time Taken by classifiers	Computational Time	Time (hh:mm:ss:ms)
3	Storage capacity	Memory	Memory consumed

IX. Results

Table 2: Test and Train accuracies before and after applying feature Engineering

SI. No	Classifiers	After Feature Engineering		Before Feature Engineering	
		Train Accuracy	Test Accuracy	Train Accuracy	Test Accuracy
1	Hoeffding Tree	0.776267	0.776267	0.592380	0.497055
2	VFDT	0.762533	0.725566	0.619857	0.500055
3	Naïve Bayes	0.666043	0.6616496	0.537857	0.498277
4	Random Forest	0.958620	0.953488	0.509411	0.504332
5	Adaptive Classifier Ensemble	0.8239645	0.7928377	0.865619	0.499388

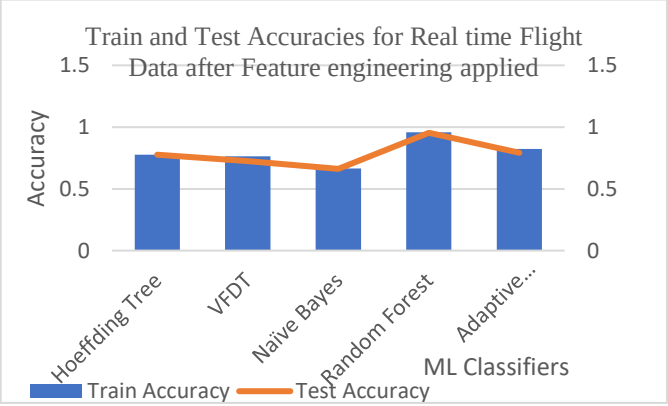


Figure 3: Train and Test Accuracies for Real time Flight Data after Feature engineering applied

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 2s
DOI: https://doi.org/10.17762/ijritcc.v11i2s.6064
***Article Received**: 28 November 2022 **Revised**: 26 December 2022 **Accepted**: 05 January 2023*

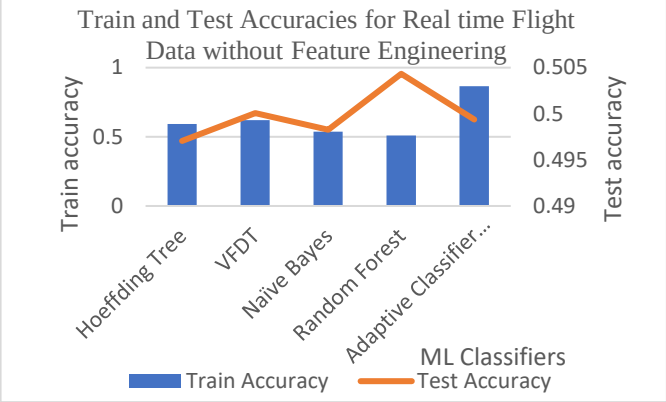
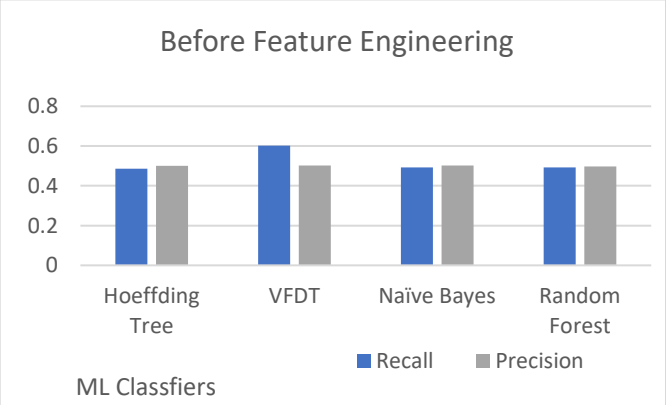


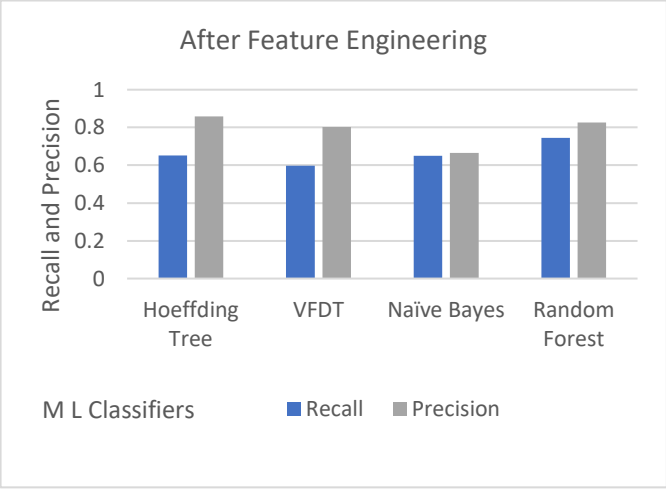
Figure 4: Train and Test Accuracies for Real time Flight Data without Feature Engineering

Table 3: Recall and Precision before and after applying feature Engineering

SI.No.	Classifier	After Feature Engineering		Before Feature Engineering	
		Recall	Precision	Recall	Precision
1	Hoeffding Tree	0.651298	0.859006	0.486438	0.500908
2	VFDT	0.59762	0.802779	0.601984	0.502392
3	Naïve Bayes	0.650605	0.664442	0.493274	0.502132
4	Random Forest	0.744093	0.825469	0.492833	0.497440



(a)



(b)

Figure 5: (a) Recall and Precision before applying feature Engineering, (b) Recall and Precision after applying feature Engineering

Table 4: Time and Memory Analysis:

SI. No.	Parameter	Before Feature Engineering	After Feature Engineering
1	Time (hh: mm: sec: ms)	0:13:06.553095	0:03:46.055386
2	Memory (mb)	98 mb	12 mb

As per the results, the use of feature engineering significantly reduced the time it took to execute the process. The difference between the time taken before and after feature engineering is roughly 9 minutes and 20 seconds. Additionally, it is observed that feature engineering also reduced the amount of memory used during the process. This means that feature engineering was able to extract more relevant information from the data and/or reduce the dimensionality of the data, which in turn led to a more efficient execution of the process.

X. Conclusion

A case study of using feature engineering to improve the performance of machine learning classifiers for predicting flight delays in a real-time flight information system is presented in this paper. We described various techniques for extracting and constructing features from the raw data, including time-based features, trend-based features, and error-based features. Additionally, we applied feature pre-processing techniques, such as the CTAO algorithm, followed by the SCSO algorithm for feature extraction and the Enhanced harmony search for feature optimization. The resulting feature set contained the 9 most relevant features for deciding whether a flight would be delayed or not. Furthermore, the performance of different classifiers using these engineered features was evaluated and compared to the results obtained using raw features. The results showed that feature engineering significantly improves the performance of the classifiers and allows for more accurate prediction of flight delays in real-time. This case study emphasises the significance of feature engineering for real-time data streams in machine learning and its potential to enhance the effectiveness of prediction models.

XI. Future Works

By expanding the current knowledge and understanding of the proposed work, the overall performance of the present research can be improved. There are many potential future enhancements can be considered.

- One potential future research direction is “seed pool of learners” a technique in which a set of different machine learning models are trained on the same dataset, and the best performing model is selected for further use. This technique can help to improve the robustness and accuracy of our predictions, as it allows for a comparison of multiple models on the same dataset.
- Another potential direction is to use other feature selection and extraction techniques, such as principal

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 2s
DOI: https://doi.org/10.17762/ijritcc.v11i2s.6064
***Article Received**: 28 November 2022 **Revised**: 26 December 2022 **Accepted**: 05 January 2023*

component analysis (PCA) and mutual information, to identify the most relevant features for flight delay prediction. By testing different feature selection methods, we can evaluate the effectiveness of each technique in terms of the performance of the classifiers.

- Incorporating more real-time data into the system, such as weather data, could help to improve the accuracy of the predictions. By using external data sources, we can include more variables that may affect flight delays, such as wind speed and visibility.
- Another potential way to improve accuracy is by incorporating more sophisticated feature engineering techniques such as deep learning-based feature extraction, and testing the performance of these features on the classifiers. This could be a promising approach as deep learning has proven to be effective in feature extraction.

REFERENCE

[1]. X. Li, Y. Zhang, and X. Wang, "Time-based feature engineering for real-time data stream classification," Information Sciences, vol. 479, pp. 173-186, 2019.

[2]. X. Wang, X. Li, and Y. Zhang, "Trend-based feature engineering for real-time data stream classification," Knowledge-Based Systems, vol. 199, pp. 105908, 2020.

[3]. J. Zhang, Y. Liu, and D. Chen, "Error-based feature engineering for real-time data stream classification," Transportation Research Part C: Emerging Technologies, vol. 117, pp. 1-14, 2021.

[4]. D. Chen, Y. Liu, and J. Zhang, "Improving flight delay prediction using feature engineering in machine learning classifiers for real-time data streams," Transportation Research Part C: Emerging Technologies, vol. 121, pp. 1-14, 2022.

[5]. H. Kim, S. Lee, and J. Park, "Feature engineering for stock price prediction using machine learning classifiers in real-time data streams," Expert Systems with Applications, vol. 126, pp. 87-95, 2019.

[6]. L. Gao, Y. Li, and D. Xu, "Feature engineering for real-time social media network analysis," Expert Systems with Applications, vol. 145, 113702, 2020.

[7]. J. Brownlee, "A Gentle Introduction to Feature Engineering for Machine Learning," Machine Learning Mastery, 2015.

[8]. V. Patel and N. Patel, "Feature Engineering for Flight Delay Prediction using Machine Learning," Journal of Advanced Research in Dynamical and Control Systems, vol. 11, no. Special Issue on Recent Advances in Control Systems and Robotics, pp. 547-556, 2019.

[9]. P. Agrawal, H. F. Abutarboush, T. Ganesh, and A. W. Mohamed, "Metaheuristic Algorithms on Feature Selection: A Survey of One Decade of Research (2009-2019)," IEEE Access, vol. 9, pp. 10.1109/ACCESS.2021.3056407, 2021.

[10]. M. K. H. Doreswamy, M. K. Hooshmand, and I. Gad, "Feature selection approach using ensemble learning for network anomaly detection," CAAI Transactions on Intelligent Technology, vol. 5, pp. 283-293, 2020.

[11]. P. W. Khan and Y. C. Byun, "Genetic algorithm based optimized feature engineering and hybrid machine learning for effective energy consumption prediction," IEEE Access, vol. 8, pp. 196274-196286, 2020.

[12]. Z. L. Chia, M. Ptaszynski, F. Masui, G. Leliwa, and M. Wroczynski, "Machine Learning and feature engineering-based study into sarcasm and irony classification with application to cyberbullying detection," Information Processing & Management, vol. 58, no. 4, pp. 102600, 2021.

[13]. C. A. Ledezma, X. Zhou, B. Rodriguez, P. J. Tan, and V. Diaz-Zuccarini, "A modeling and machine learning approach to ECG feature engineering for the detection of ischemia using pseudo-ECG," PloS one, vol. 14, no. 8, pp. e0220294, 2019.

[14]. Z. Chen, P. Zhao, F. Li, T. T. Marquez-Lago, A. Leier, J. Revote, et al., "iLearn: an integrated platform and meta-learner for feature engineering, machine-learning analysis and modeling of DNA, RNA and protein sequence data," Briefings in bioinformatics, vol. 21, no. 3, pp. 1047-1057, 2020.

[15]. S. M. Kasongo and Y. Sun, "A deep learning method with filter-based feature engineering for wireless intrusion detection system," IEEE access, vol. 7, pp. 38597-38607, 2019.

[16]. C. Fan, Y. Sun, Y. Zhao, M. Song, and J. Wang, "Deep learning-based feature engineering methods for improved building energy prediction," Applied energy, vol. 240, pp. 35-45, 2019.

[17]. A. Ullah, K. Muhammad, I. U. Haq, and S. W. Baik, "Action recognition using optimized deep autoencoder and CNN for surveillance data streams of non-stationary environments," Future Generation Computer Systems, vol. 96, pp. 386-397, 2019.

[18]. S. Demir, K. Mincev, K. Kok, and N. G. Paterakis, "Introducing technical indicators to electricity price forecasting: A feature engineering study for linear, ensemble, and deep machine learning models," Applied Sciences, vol. 10, no. 1, pp. 255, 2019.

[19]. G. Li, J. Li, Z. Ju, Y. Sun, and J. Kong, "A novel feature extraction method for machine learning based on surface electromyography from healthy brain," Neural Computing and Applications, vol. 31, no. 12, pp. 9013-9022, 2019.

[20]. H. M. Gomes, J. Read, A. Bifet, J. P. Barddal, and J. Gama, "Machine learning for streaming data: state of the art, challenges, and opportunities," ACM SIGKDD Explorations Newsletter, vol. 21, no. 2, pp. 6-22, 2019.

[21]. Y. Zheng, G. Li, W. Zhang, Y. Li, and B. Wei, "Feature Selection with Ensemble Learning Based on Improved Dempster-Shafer Evidence Fusion," IEEE Access, vol. 6, pp. 10.1109/ACCESS.2018.2890549, 2018.

[22]. L. Yuan, B. Pfahringer, and J. P. Barddal, "Addressing Feature Drift in Data Streams Using Iterative Subset Selection," Applied Computing Review, vol. 19, no. 1, 2019.