FLIGHT DELAY PREDICTION USING MACHINE LEARNING

Electronics and Telecommunication Engineering

Submitted By

KARTIK AWALEY **TRUPTI MAHAJAN**

ASHISH PIPARE

Under the guidance of

Dr. V. K. TAKSANDE

Department of Electronics and Telecommunication Engineering
Priyadarshini College of Engineering,
Nagpur - 440019
2021-22

ABSTRACT

Flight Delay Prediction Using Machine Learning

Precise flight delay prediction is vital for the airline industries and passengers. This thesis focuses on applying several machine learning and auto-ML techniques to predict flight delays. A flight delay is said to occur when an airline lands or takes off later than its scheduled arrival or departure time, respectively. Conventionally, if a flight's departure time or arrival time is greater than 15 minutes than its scheduled departure and arrival times respectively, then it is considered that there is a departure or arrival delay with respect to the corresponding airports. Notable reasons for commercially scheduled flights to be delayed are adverse weather conditions, air traffic congestion, a late reaching aircraft to be used for the flight from a previous flight, maintenance, and security issues. In this research study, a python-based model will be developed for a specific Airline and an Airport from already established models that are available in literature and were implemented in flight delay predictions. Once that is completed, the same model will be used for a different Airline at the same Airport. Later, the model will be implemented for several other Airports to check the adaptability of the models. In this process, there will be an attempt to enhance the existing models by carefully selecting the dataset and features. In the final stage, the results will be compared with the Microsoft Azure Machine Learning Studio, the best model will be deployed using Auto-ML and the existing interpretable machine learning package, LIME will be used to explore local prediction capability of the models. This study has been conducted with the hopes that alongside other increasing numbers of studies in this subject matter, it will contribute to improving on-time performances of flights to benefit airline customers, airline personnel, and airport authorities.

INTRODUCTION

An arrival flight delay is said to occur when an airline lands later than its scheduled arrival time.
Notable reasons for commercially scheduled flights to delay are adverse weather conditions, air traffic congestion, late reaching aircraft to be used for the flight from previous flight, maintenance and security issues.

Flight delays are relatively common in both domestic and international flights. Based on the statistics of the Bureau of Transportation, 18% (average of last 10 years)((*Bureau of Transportation Statistics*, n.d.) of US domestic flights arrive more than 15 minutes late. According to Baik et al., these delays are not only a cause of frustration for the passengers, Airline, and airport authorities but also play an important role in financial loss for all parties (Baik et al., 2010)). Some of the flights are more frequently delayed than others. The continuous technological advancement in data storage enables the storage of a massive amount of data and computational power leads to the development of data analytics. Different government agencies, airport authorities, and Airline companies are collecting significant amounts of data and analyzing these datasets to aid in gaining knowledge about the delays. A robust flight delay prediction model with proper explanation of the delay is not only an interest of travelers but also of Airlines and Airport Authorities.

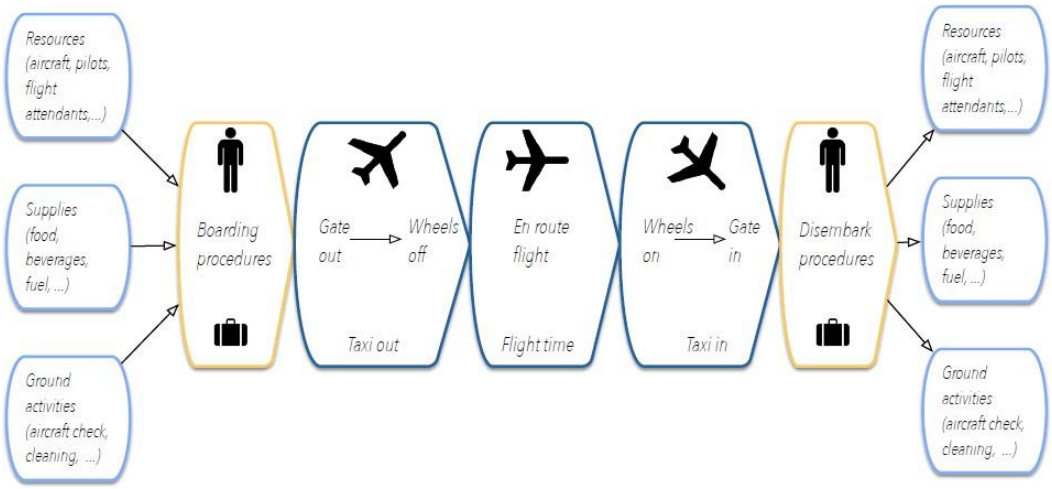


Figure 1: A typical process of Air Transportation System (Sternberg et al., 2017)

PROBLEM STATEMENT & RESEARCH PLAN

Air traffic is a very complex transportation system and the reason for the delay can occur at different stages of the process. Figure 1 is a schematic of this complex process.

In this research study, a python-based model will be developed for a specific Airline and one of their operating or hub Airport from already established models that are available in the literature and were implemented in flight delay predictions. These data-driven methods will only consider historic observations and will be using several years of records (largest public dataset of flight delay) from the Bureau of Transportation Statistics (BTS) of the United States Department of Transportation of US domestic flight delays. The model will not take into account short-term effects, such as current weather or traffic situation. The same model will then be used for a different Airline and it's one of the operating or hub Airports. In this process, there will be an attempt to enhance the existing models by carefully selecting the dataset and features. In the final stage, the results will be compared with Microsoft Azure Machine Learning Studio and also with Azure Auto-ML, and then the interpretation of prediction of delay will be made based on an interpretable machine learning package, Local Interpretable Model-Agnostic Explanations (LIME). This study is being conducted with the hopes that it will contribute to improving on-time performances of flights for the benefit of the airline customers, airline personnel, and airport authorities. Figure 2, which can be found below, represents the Research Plan for Flight Prediction Model and the interpretation of the prediction.

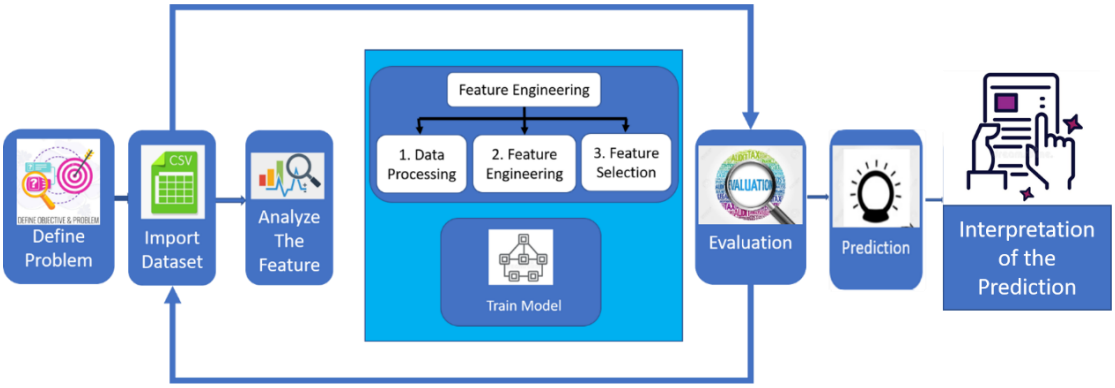


Figure 2: Flow Chart of the Research Plan for Interpretable Flight Delay Prediction

MACHINE LEARNING WORKFLOW

Machine Learning (ML) models are used to learn from data without being explicitly programmed. ML models are code that has been trained to recognize several types of patterns in the data and make a prediction based on that. Machine learning techniques are useful for solving experiments efficiently and effectively. A great amount of data is loaded into a computer program and a model is chosen to fit the data, which allows the program, without any help to make predictions, based on the trained data. Predictive models are only as good as the data from which they are built, thus using valid and relevant data helps with high-performing models. Here, the analogy of garbage-in garbage-out takes into effect which means that if a model is fed garbage, that is exactly what it will return, in other words, the trained model will provide invalid predictions. Based on Figure 3, the workflow of Machine learning includes all the steps required to build the proper machine learning model from scratch.

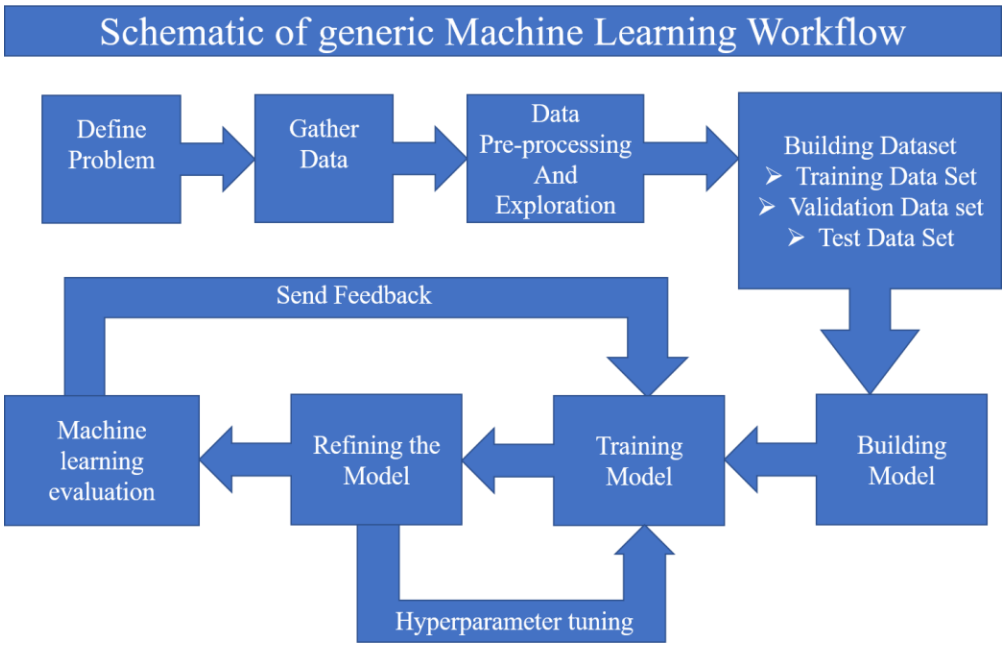


Figure 4:Schematic of generic Machine Learning workflow

Define Problem: This is the first step of machine learning. It is important to identify what exactly one expects as output from a model. This might involve in having some assumptions which mostly come from the domain knowledge.

Gather Data: Based on one problem statement, one must gather an appropriate data set. Quality of data dictates the accuracy of the model, so it is a very important stage of ML workflow. This can be a tedious process because getting the right dataset in the right format can be challenging. It has to correlate with the

outcomes which are being predicted, accumulating data, requires a clear understanding of domain knowledge, and proper engagement in sampling from a large database to capture records to be used in an analysis. Figure (below) is a depiction of different sources of data for machine learning models.

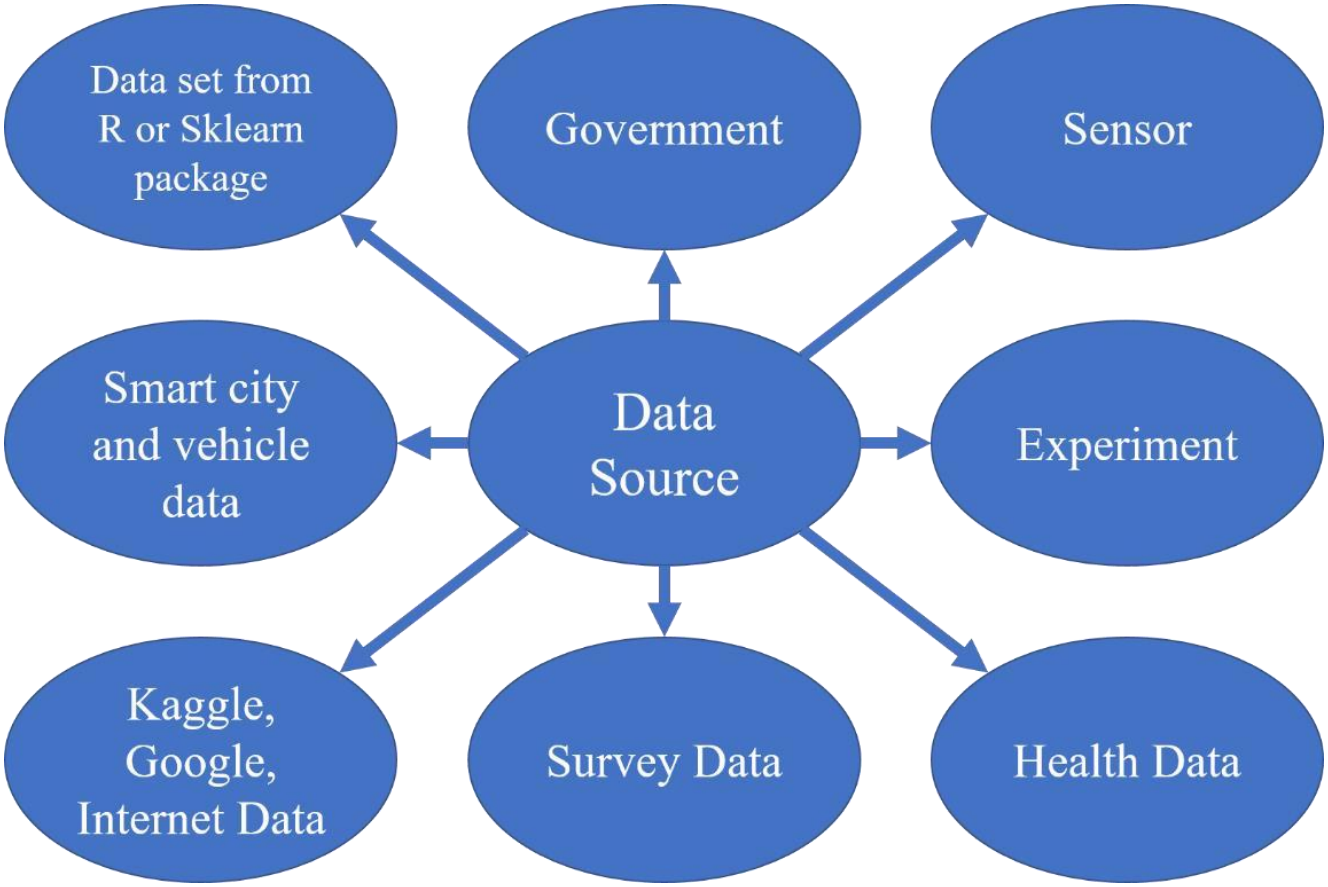


Figure 5: Different sources of data for machine learning models

Data Pre-processing: Data preprocessing is the process of cleaning data and preparing it to be used to train the model. Most scientists believe data cleaning and formatting can be considered the most challenging part of any project (Shmueli et al., 2017). In the real world, data is incomplete, inconsistent, and inaccurate which means that there are errors and outliers present in the data which causes there to be a lack in patterns and trends. According to (Shmueli et al., 2017) data preprocessing enhances the quality of data to stimulate the extraction of meaningful insights. Some of the key steps of data preprocessing are

a) gathering the dataset: In this case, data was gathered from the Bureau of Transportation Statistics for 2016-2019, using only American Airlines (AA) and Southwest (WN), limited to the US domestic flights and a couple of airports.

b) Importing all the required libraries: importing libraries and dependencies, into the Python environment will make tasks easier, as it has built-in functions and models that can be used instead of doing that ourselves. For example, some of the libraries that will be used are pandas, which is used for data cleaning and analysis, NumPy which is a library that is mostly used for, multi- dimensional arrays and matrices, along with mathematical functions to operate on these arrays, matplotlib which is used for analyzing and visualizing charts or graphs, as well as several other libraries which will be used throughout the project.

c) Importing the dataset: When running python programs, datasets are required for data analysis. Python has several modules for importing external data. For example, the method used in this project is by importing the CSV to enable us to read each row in the file using a comma as a delimiter, which is best described in (*Importing Data in Python*, n.d.)

d) The next step in ML is feature engineering, it is the process of using domain knowledge to extract features from raw data. These features can be used to improve the performance of machine learning algorithms. The following list of techniques exist in feature engineering: Imputation, Handling Outliers, Binning, Log Transform, One-Hot Encoding, Grouping Operations, Scaling, some of which will be used in this thesis.

i. Data cleaning or identifying and handling the missing values: according to (Shmueli et al., 2017) from a statistics point of view, it is important to understand different types of missing data. The type of missing data will help with filling in the data or discarding them. In python, *isnull* and *notnull* can also be used to summarize missing values, this will be discussed in greater detail further in this paper.

ii. Encoding the categorical data: According to (Shmueli et al., 2017) data scientists spend 80% of their time cleaning and preparing data, in this process converting categorical data is an inevitable activity. It helps improve the model quality and provides better feature engineering. For example, in some of the feature binary variables containing either 0 or 1, where 0 represents the absence, and 1 represents the presence.

iii. Feature Scaling is also sometimes involved in this step of ML workflow. It helps to normalize the data through standardization, which involves rescaling the properties of a standard normal distribution with a mean of zero and a standard deviation of one. Normalization is a scaling technique in which values are rescaled, hence varying between zero and one, it is also known as Min-Max scaling. **Error! Reference source not found.** is the formula for normalization where Xmax and Xmin signify the maximum and the minimum values of the feature, respectively.

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Another type of scaling often used is Standardization where the values are focused around the mean with a component standard deviation. **Error! Reference source not found.** isthe formula for standardization, where μ represents the mean of the feature values and σ represents the standard deviation of the feature values.

$$X' = \frac{X - \mu}{\sigma}$$

Once preprocessing has been complete, data exploration is the first step in data analysis where the use of data visualization and statistical techniques describes the nature of the dataset. Data exploration helps us visually explore and categorize relationships between different features, structures, and the presence of outliers. It also helps us understand the patterns and trends present in the raw data.

Building Data set: Based on the data, and the purpose of the problem, one has to determine the Machine Learning task to be performed amongst the following tasks: classification, prediction, clustering, and partitioning the data accordingly. For example, if the problem is a supervised (prediction), the dataset is divided into three parts: training, validation, and test datasets. Many Machine Learning techniques such as regression, neural nets, decision trees, etc, can be used in this iterative process. Below figure is an example of the process of splitting the data set.

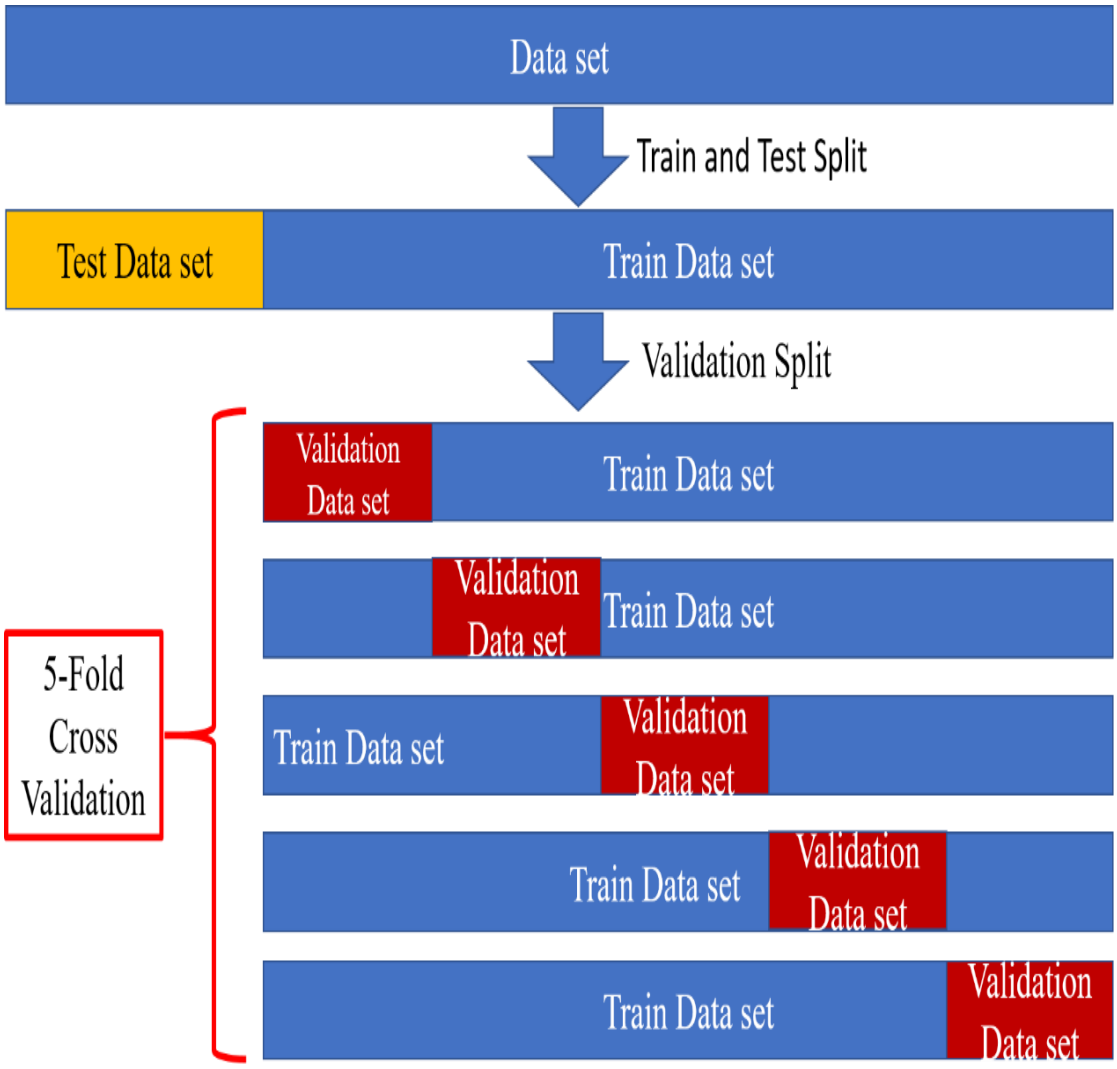


Figure 6:Process of splitting the data set

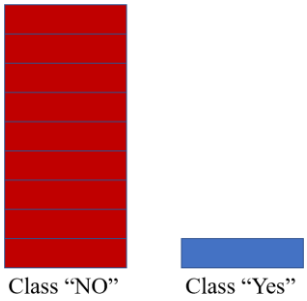
Dealing with Imbalanced data

As most of the machine learning algorithms are designed to maximize accuracy and reduce error, it works best when the number of samples in each class are about equal but in real data set this is very rare.

Class imbalance is a problem that arises in machine learning classification problems. A classification problem in machine learning in which a target must be predicted given some input. There is a great chance that the distribution values may be different and due to this difference in the class, the algorithms may be biased towards the majority values and will not perform well on the minority values, as a result, the difference in class affects the outcome of the model.

For an example, if someone has a two-class problem (e.g., yes or no). If 10% data points are of the class of "Yes", 90% for the class of "No" class. The "No" class is the majority class and "Yes" class is the minority. Here "No" class to "Yes" class ratio is very high. This problem is referred to as a class imbalance.

Figure 7:Example of imbalanced dataset



There are many techniques which are used in dealing with the imbalanced data. Below are some examples of those techniques.

Oversampling: This technique tries to increase the size of minority samples to create a balance.

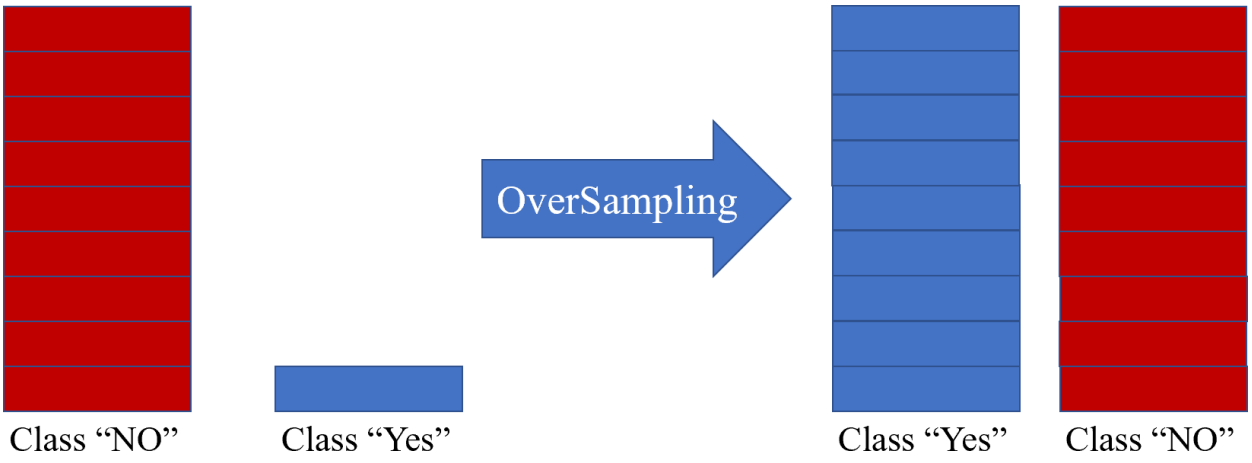


Figure 8: Oversampling example, creating balance

One of the popular oversampling method is Synthetic Minority Over-sampling Technique (SMOTE). By using a distance measure, SMOTE algorithm selects two or more similar instances to create synthetic samples from the minority class. below is a schematic of the algorithm.

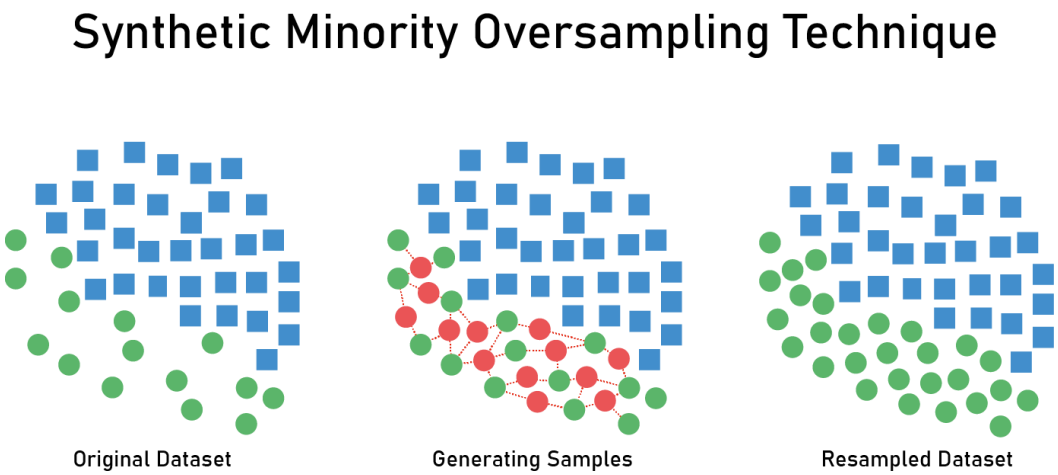


Figure 9:SMOTE (Bank Data: SMOTE. This Will Be a Short Post before We… | by Zaki Jefferson | Analytics Vidhya | Medium, n.d.)

Undersampling: Undersampling works in opposite way of oversampling, it aims to decrease the size of the majority class to balance the dataset.

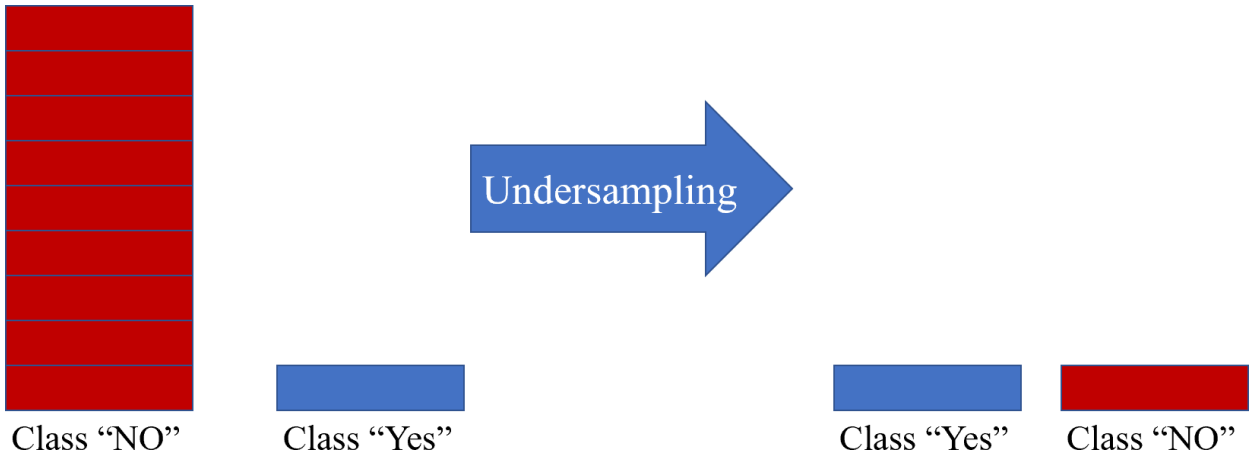


Figure 10: Undersampling technique example, creating balance

Penalize Model: One can penalize the model to pay more attention to the minority class by imposing an additional cost on the model for making classification mistakes on the minority class during training.

Assigning Weights for the Classes: As stated earlier, machine learning algorithms are not very useful with biased class data. When training algorithms, skewed distribution of the classes can be taken into consideration. This can be accomplished by giving different weights to both the majority and minority classes. The formula to calculate weights is **Error! Reference source not found.**:

Where,

$$wj=n\_samples / (n\_classes * n\_samplesj)$$

wj represents the weight for each class n_samples represents
the total number of samples
n_classes represents the total number of unique classes in the target n_samples represents
the total number of rows of the respective class

In order to achieve more accurate and meaningful results, weights were distributed among the classes.

Building Model: Before diving into models, understanding algorithms is important, Machine learning algorithms can be divided into 3 broad categories:

- Supervised learning
- Unsupervised learning, and

- Semi-supervised learning.

Supervised learning is the function that maps an input to an output based on example input-output pairs. Unsupervised learning is a type of algorithm that learns patterns from untagged data. According to (Truong et al., 2018), semi-supervised learning is an approach that combines a small amount of labeled data with a large amount of unlabeled data during training. As can be interpreted from the name, semi-supervised learning falls between unsupervised learning and supervised learning. Machine learning algorithms build a model based on sample data, identified as the "training data", to make predictions. There are many models in Machine Learning but the following four models: Random Forest, AdaBoost, XgBoost, Neural network will be looked into.

Random forest: Random forest is one of the most used algorithms, it can be used for both classification and regression tasks, classification will be used in the case of this thesis. To use this algorithm in Python, the RandomForest Classifier and BaggingClassifier libraries are required to be imported. It is the supervised, ensemble learning, usually trained with the “bagging” method which helps with the overall result. The Random Forest algorithm will be used because it is very easy to measure the importance of each feature in the prediction. The hyperparameters are used to increase the predictive power of the model or to make the model quicker. Some common hyperparameters for increasing the predictive power are the n_estimators which is the number of trees the algorithm builds. In general, a higher number of trees increases the performance and makes the predictions more stable. Another important hyperparameter is max_features, which is the maximum number of features random forest considers when splitting a node. The last hyperparameter is min_sample_leaf which helps determine the minimum number of leafs required to split an internal node. Although the Random Forest algorithm is versatile, it is slow in creating predictions once the model is made. The Random Forest algorithm uses the Gini Index which is a measure for classification type problems. For the purpose of this thesis, the function being used takes in the parameters gini which measures the quality of a split and the maximum depth of the tree. If the maximum depth is denoted to none, then the nodes are expanded until all leaves contain less than min_samples_split samples.

$$Gini = 1 - \sum_{i=1}^{C} (p_i)^2$$

The formula **Error! Reference source not found.** uses probability to determine the Gini of each branch on a node; deciding which branch is more likely to occur. Here, P_i represents the relative frequency of the class in the dataset and c represents the number of classes. This can also be done using entropy. The formula for entropy is shown below:

$$Entropy = \sum_{i=1}^{C} -p_i * \log_2(p_i)$$

Entropy manages the probability of a certain outcome for making a decision (the equation above). This algorithm can be extremely useful with different types of data sets. It is easy to use, fast to train and finds an accurate representation.

AdaBoost: The next model which is used in this thesis is AdaBoost, short for "Adaptive Boosting. To use this algorithm in Python, the AdaBoostClassifier library needs to be imported. It focuses on classification problems and uses an ensemble learning method. AdaBoost uses an iterative approach to learn from the mistakes of weak classifiers, and turns them into strong ones as shown below.

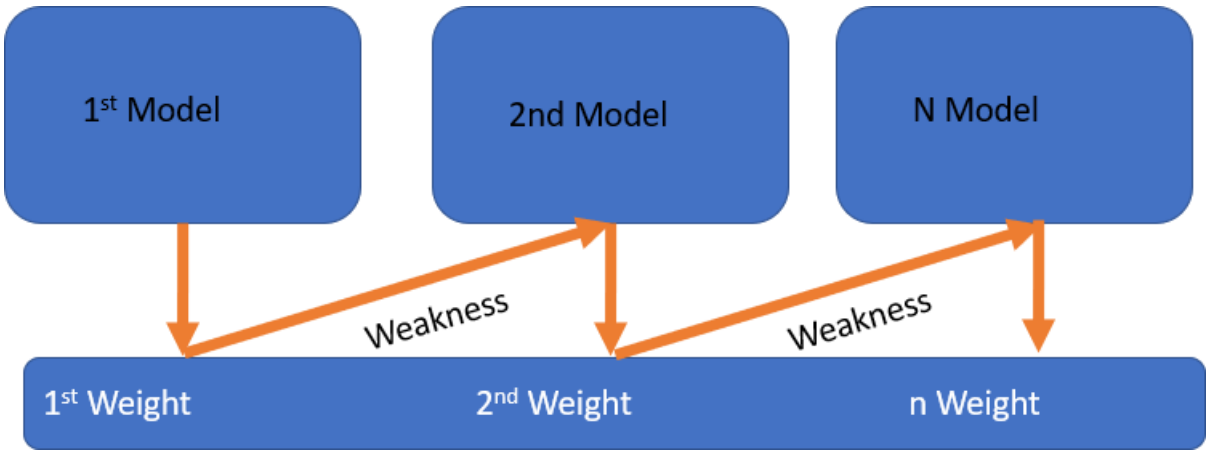


Figure 11:Adaboost Classifier iteration approach

The way the model works is, it makes n number of decision trees during the training, as the first decision tree is made, and the record which is incorrectly classified is given more priority. Two hyperparameters that are mostly used are the number of estimators (n_estimators) and the learning rate. Only those incorrect records are sent as input for the next model and the process will

continue. The final equation for classification can be represented **Error! Reference source not found.**:

$$F(x) = sign\left(\sum_{m=1}^{M} \theta_m f_m(x)\right)$$

where f_m stands for the m'th weak classifier and θ_m is the corresponding weight. It represents the weighted combination of M weak classifiers.

XGBoost: Another algorithm which will be looked into is XGBoost, it is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. It helps in prediction problems involving unstructured data. This algorithm has many advantages such as Regularization which also helps to reduce overfitting. Overfitting occurs when the model tries to contain all the data points in the provided dataset, which reduces the accuracy of the model. The overfitted model results in low bias and high variance. XGBoost has an in-built procedure to handle missing values on each node, which allows users to run cross-validation at each iteration of the boosting process. The final equation for XGBoost can be represented as **Error! Reference source not found.**:

$$\mathcal{L}_{split} = \frac{1}{2}\left[\frac{(\sum_{i\in I_L} g_i)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{(\sum_{i\in I_R} g_i)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{(\sum_{i\in I} g_i)^2}{\sum_{i\in I} h_i + \lambda}\right]$$

Multilayer Perceptron (MLP): The last algorithm/model which will be used in this project is Multilayer Perceptron (MLP). MLPs are suitable for classification prediction problems. It utilizes a supervised learning technique called backpropagation. It is an algorithm that calculates a complicated gradient. In the Multilayer perceptron, there are combinations of neurons. For instance, in a three-layer network, the first layer is the input layer, the middle layer is the hidden layer and the last layer is the output layer, as can be seen Figure 12.

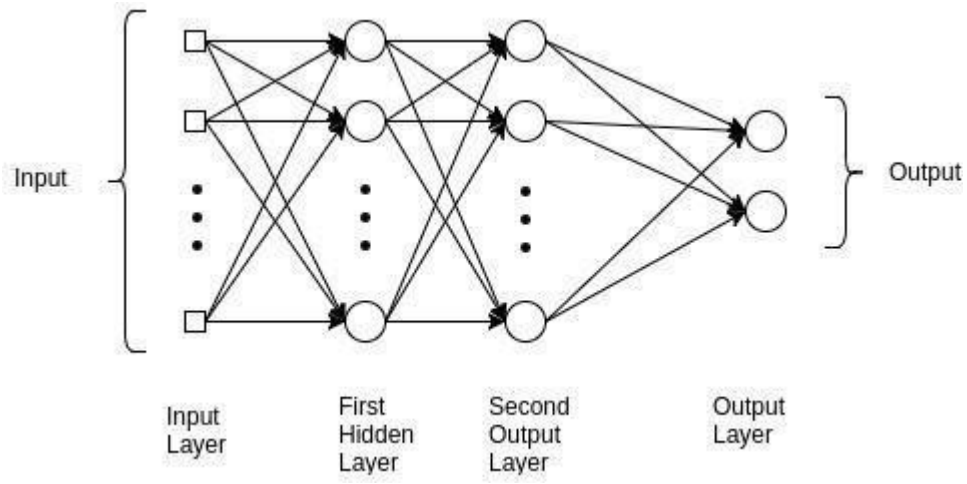


Figure 12: Multilayer perceptron-NN

MLP networks are composed of many functions that are chained together. Each layer is represented as y = f(WxT + b) where f represents the activation function, W represents the set of parameters, or weights, in the layer, and x represents the input vector. To use this algorithm in, Python, the tensorflow and keras library needs to be imported. Keras works as an interface for TensorFlow which focuses on the inference of deep neural networks.

Training and testing the model: According to Wiley et al, (*Data Mining for Business Analytics: Concepts, Techniques, and Applications in R | Wiley*, n.d.), once the model is selected based on the problem definition and domain knowledge, it is time to feed the training set to the algorithm so that it can learn suitable parameters and features used in the data set. Validation data set is mainly used in modifying or discarding variables and includes a process of tuning model-specific hyperparameters until a satisfactory accuracy level is accomplished.

In the testing stage, a test dataset is used to verify that model using accurate features. In this part of the workflow, one should return to training the model based on the feedback to improve accuracy and desired output.

Model Evaluation and Feedback: It can be done using accuracy, precision, recall and F1-score. Generally, finding accuracy can determine whether a model is being trained correctly and how it may function. The problem with using accuracy is that it does not do well when one has a severe class imbalance, that is why precision, recall must also be considered. **Error! Reference source**

not found.Error! Reference source not found. is the formula for how accuracy is mathematically interpreted.

$$Accuracy = \frac{truepositives + truenegatives}{totalexamples}$$

Precision is helpful when false positives are high, precision can help predict positives. Let us assume a model has very low precision, which can lead to the assumption that there are a lot more delayed flights, and this may be a false conclusion. **Error! Reference source not found.** is the mathematical equation for precision.

$$precision = \frac{true\ positives}{true\ positives + false\ positives}$$

The opposite of precision is recall, recall helps when the false negatives are high, as false negatives can also be misleading. **Error! Reference source not found.** is the mathematical equation for recall:

$$Recall = \frac{truepositives}{truepositives + falsenegatives}$$

When modelling, machine learning algorithms assume that the data is evenly distributed within classes. If the imbalanced data is not taken care of, the model may predict high accuracy, but this will not have any value to the objective. According to (James et al., 2000) the F1-score is simply the harmonic mean of precision (PRE) and recall (REC). The balance between precision and recall can be found using the F1-score metric **Error! Reference source not found.**, which is beneficial toward imbalanced datasets.

$$\text{F1 Score} = \frac{2 \text{ x (Precision x Recall)}}{\text{Precision + Recall}}$$

Interpretable Machine Learning:

Application of Machine learning is now widespread but ML models are still considered as a black box which is a barrier to the adoption of machine learning. Why can a ML model make certain

prediction? That is one of the big questions in the implementation of ML. Interpretability of those prediction can help to increase the trust, to select better models and to reduce the bias that exist in the data set. Figure below is a pictorial explanation of the importance of interpretable machine learning. (*Interpretable Machine Learning - Christoph Molnar - Google Books*, n.d.) ((Masís, 2021)

Figure 13: Importance of interpretable ML

Lime stands for Local Interpretable Model-Agnostic Explanations. LIME allows end-users to interpret predictions and take action. It is model-agnostic, implying it can be applied to any machine learning model.

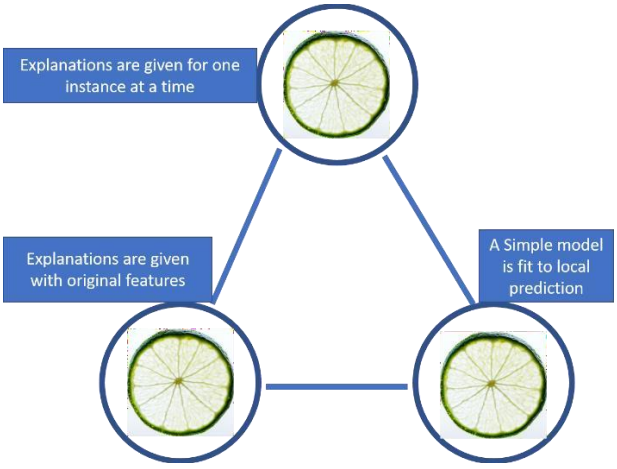


Figure 14: schematic of how LIME works

Above Figure is a schematic of how LIME works. The technique attempts to understand the model by studying the input of data and recognizing how the predictions change. In simple words, LIME assumes a black-box machine learning model and examines the relationship between input and output. It helps understand feature importances on a dataset level. Also, it allows verification of the problem statement, but when using LIME, it is important to accurately interpret the output.

The way LIME gives explanation of the model is by approximating the black box model for each prediction to explain, permute the observation n times, then let the complex model to predict the outcome of all permuted observations. It calculates the distance from all permutations to the original observation. Followed by converting the distance to a similarity score. Select m features best describing the complex model outcome from the permuted data. Then it fits a simple model to the permuted data, explaining the complex model outcome with the m features from the permuted data weighted by its similarity to the original observation. Finally, extract the feature weights from the simple model and use these as explanations for the complex models local behavior. Figure below explain steps that LIME uses for the interpretation (Ribeiro et al., 2016).

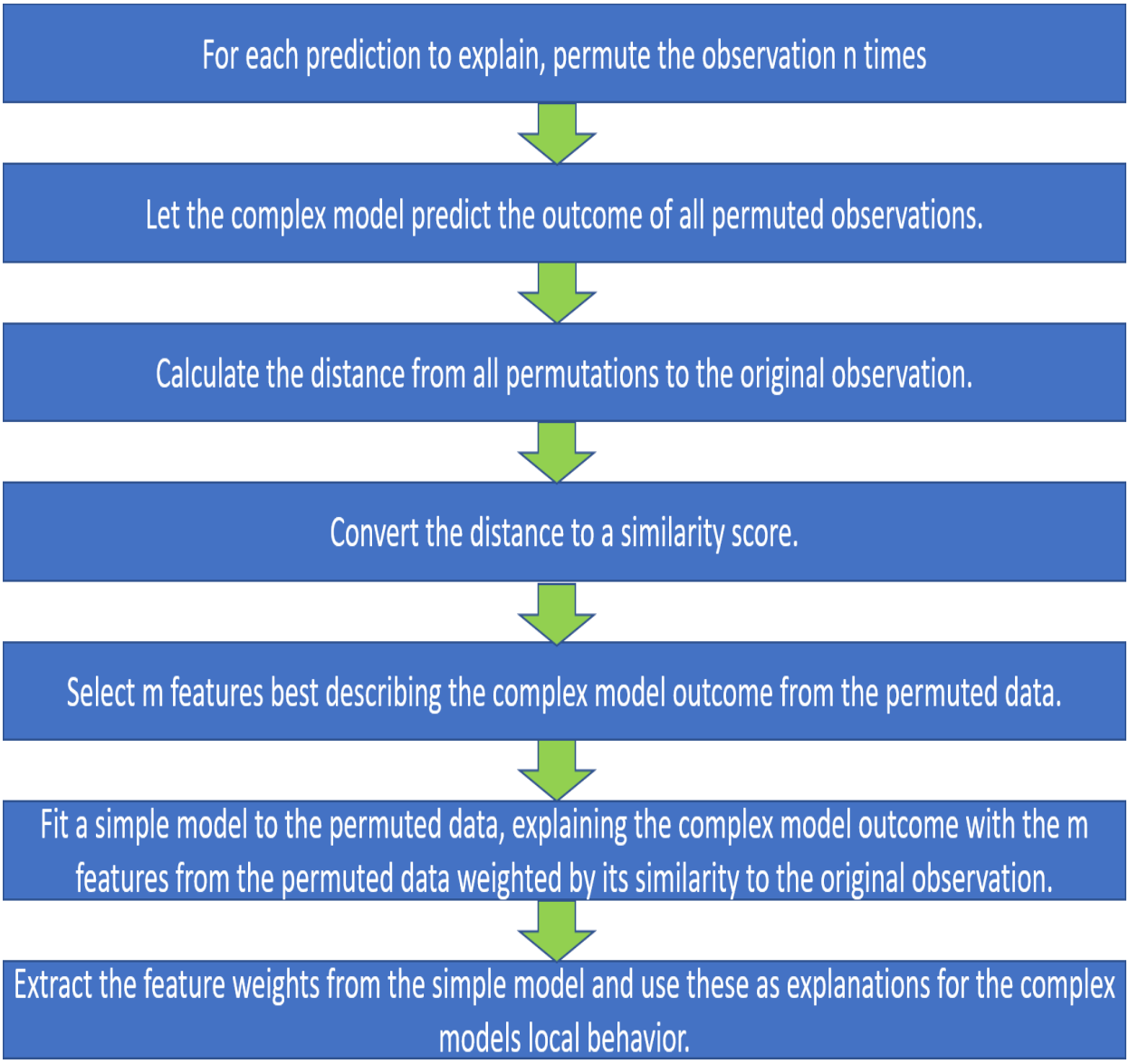


Figure 15: Steps that LIME uses for the interpretation

CONCLUSION

During this master's thesis, different prediction models and various evaluation methods are explored using different applications and services. By using historic data, interesting results were observed on the predictability of delays. The best delay prediction method emerged to be the most specific one, which takes into account all the combinations of categorical features.

The performances of the models were interesting to evaluate, due to the numerous features used. From the exploratory data analysis (EDA), we found that AA flights departing from ORD and arriving at DFW are the most delayed flights from the exploratory data analysis (EDA), we found that AA flights departing from ORD and arriving at DFW are the most delayed flights. For southwest airlines departing from LAX had the most total number of delays. There is a relationship between arrival and departure delay. Also, found June and July are the worse month when it comes to total number of delays.

For AA, model accuracy using Python for Random Forest obtained 85% accuracy, Adaboost obtained 83% accuracy, XGBoost obtained 90% accuracy, Gradient Boost obtained 88% accuracy and MLP obtained 86% accuracy. For WN, model accuracy using Python for Random Forest obtained 81% accuracy, Adaboost obtained 83% accuracy, XGBoost obtained 86% accuracy, Gradient Boost 84% accuracy and MLP obtained 86% accuracy. The accuracy for AA of the random forest model using Azure Studio for is 96%, and Azure-AutoML gives 82% for XGBoost and 92% for Random Forest. The differences in accuracy between Python and Azure AutoML may be the result of the data pre-processing, as Azure AutoML automates the machine learning workflow (that includes data preprocess) on the other hand for Python model data preprocess requires domain knowledge.

In the business world, we can save time and money by improving the understanding of our machine learning model prediction. Using LIME, it is possible to find out the cause of delay from the prediction model and take action to mitigate those reasons.