**Supply Chain Fraud Detection System**

**Introduction**

Supply chain management is a domain that could greatly benefit from machine learning techniques. As the volume of goods shipped and transported increases, so does the need to accurately predict order fulfillments and detect fraudulent orders. The dataset we are using is called “DataCo SMART SUPPLY CHAIN FOR BIG DATA ANALYSIS” and was collected by the company DataCo, provided by Fabian Constante of Universidad Central del Ecuador, and hosted on Mendeley Data. The dataset’s large size and numerous features allow us to explore the factors that go into supply chain management and see if we can mine meaningful data from shipment data.

The dataset has over 180,000 rows with 53 columns. The target feature is the Order Status column. Orders can be Complete, Pending, Closed, Pending Payment, Processing, Canceled, and Suspected Fraud. In our analysis we want to group the statuses as Not Suspected of Fraud and Suspected of Fraud. There are columns that might help predict if an order was suspected as fraudulent such as the Payment Type and Product Category, however, there are also columns that may have little bearing on the matter such as the Customer’s First Name. As such we do not want to perform PCA which reduces the number dimensions of the dataset while keeping the original data intact would not apply to our situation. Reducing the dimensions without removing insignificant data would mean that we retain noise within our model. We need to perform feature selection to drop columns that are not significant to our target feature and keep the columns that are significant.

**Technology & Literature Survey**

Porouhan et al. (2021) set out to analyze the supply chain dataset using data mining methods. The dataset was obtained from the company DataCo Global. However, on this dataset, machine learning models were applied in addition to R programming. The author explains the usage of a process mining method called Fluxicon Disco, which helped in examining consumers' activities while they shopped from more than 164 different nations. The dataset contains features such as order status, shipping mode, products, category types, payment type, delivery status etc. More importantly, it has 118 ways to manage user orders in addition to the 180,519 items that were ordered online through a website. Due to its volume, this dataset can be used in data analytics research. The author concludes by stating that the data obtained from this study was distinctive because it revealed various demographic categories of customers around the world as well as their preferred purchases, payment options, delivery status, and other information.

In order to reduce future risk Constante et al. (2020) uses a transaction to demonstrate how machine learning approaches might be useful in anticipating fraud in the smart supply chain. This paper's major objective is to demonstrate several semi-supervised machine learning methods that can be used to anticipate various fraud transactions and to statistically compare them. The author tested four various models (C5,Rpart, Random Forest, SVM). The AUROC curve and the f1 score confusion matrix were used to determine the model's accuracy. Additionally, each model has been evaluated using 100000 distinct transactions to estimate the false positive rate, capture processing time, and determine the best model, which is Random Forest, which has an accuracy of 81.55%, the highest of the four models used.

Kumar (2021) used an ensemble technique of supervised machine learning algorithms in supply chain management to construct a prediction model on fraud orders detection. With regard to supply chain management and logistics management operations, fraud orders are a big research problem in the business world since they produce false statistics and disrupt the entire business process. The researcher highlighted some of the crucial challenges with supply chain management studies on fraud orders identification.

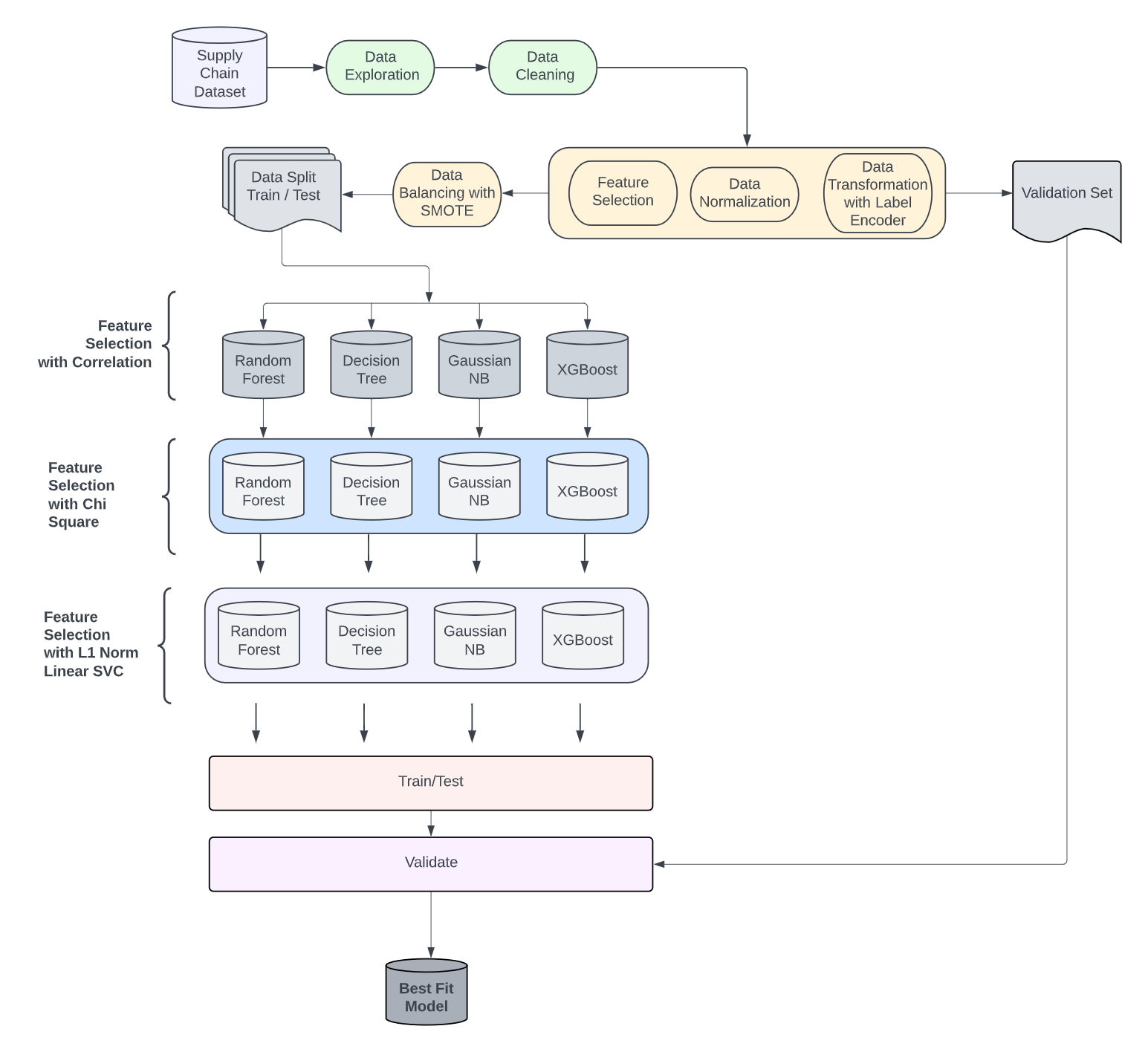
The researcher applied ensemble methods to a prediction model built using various supervised machine learning techniques. The goal of this study was to compare the accuracy levels of various supervised machine learning algorithms, including Logistic Regression (0.69), Random Forest (0.89), K-Neighbor (0.74), Gaussian-NB (0.67), and Decision Tree (0.88). This predictive model can handle unbalanced training datasets and forecast whether sales and orders fall under the category of fraud or not. It has been validated to have an accuracy level of 89%The researcher came to the conclusion that the ensemble approach of the predictive model on the detection of fraud orders using supervised machine learning algorithms in supply chain management is based on Logistic Regression, Random Forest Classifier, K-Neighbors Classifier, Gaussian-NB, and Decision Tree Classifier. This prediction model is classifying whether or not the orders fall under the category of fraud with an accuracy level of 89%. The researcher guarantees that the predictive model will help supply chain management and logistics management businesses to determine if sales and orders are fraudulent or not and will improve corporate operations and processes.

**Project Overview**

In this project we collect our data and then perform initial exploratory data analysis to determine the cleaning and preprocessing necessary. After we processed the data we took a hold out set of validation data to be used at the end of the modeling process. We then balanced the training and the test data using SMOTE before training and testing our models. We use the holdout set to validate the performance of the models to determine the best predictor. The figure below shows this process and the following sections cover each step in more detail.

**Figure 1**

*Project Data Flow*

****

*Note.* The above figure shows the data flow diagram for our Fraud detection project.

**Data**

**Data Collection**

The data for this project is taken from a real world manufacturing business company. So the data records in this dataset are also real world transactions which occur between the customers and the company. So this data can be useful for the company to identify supply chain frauds which can affect their business. The precise quantity of our dataset contained 180,519 rows and 53 columns, which have been collected in a timespan of three years by the company. The dataset is licensed by Creative Commons 4.0 and is available online on Mendeley Data repository. The goal of this project is to find a solution by looking at the patterns in the data which can assist the company in preventing frauds in the supply chain. The table in Appendix A displays the variables from the dataset along with their description. We have listed some of the relevant columns along with a short description and also its type.

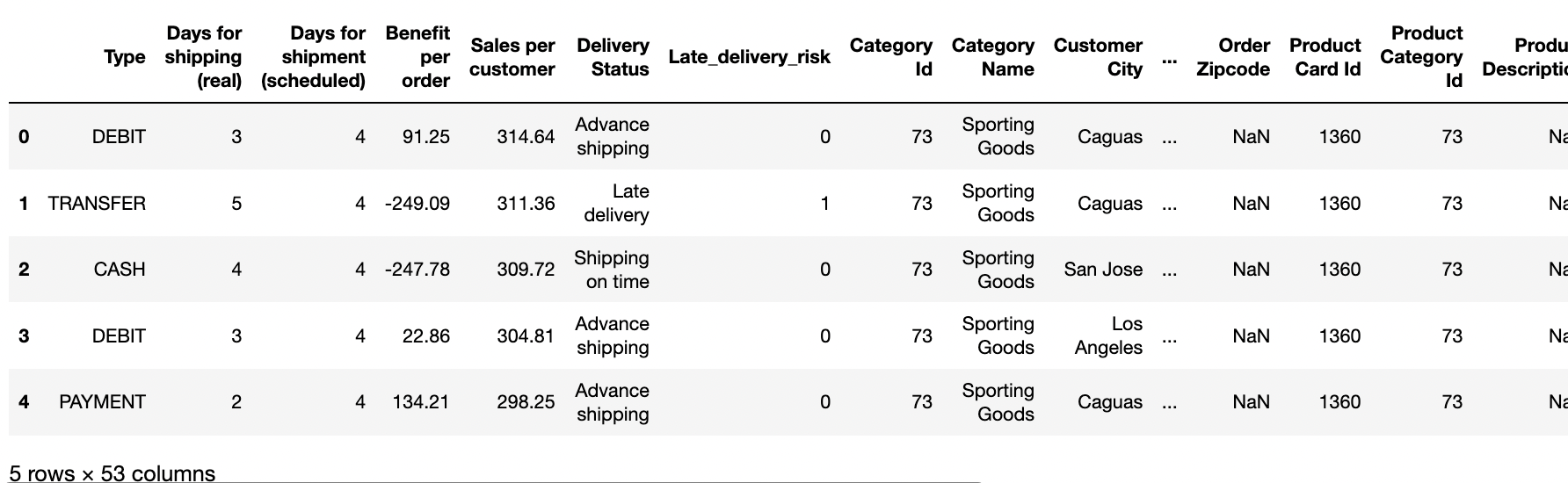
The predictor variable for our project is fraud. The business will always want to predict transactions which are fraud as they are not good for their business. In this project we define fraud as a binary column where the classes are fraud and Non fraud. Initially, there were nine classes such as complete, pending, processing, canceled, pending payment, on hold, closed, payment review and suspected fraud, so we transformed all besides suspected fraud to non fraud. We coded suspected fraud as (0 = yes) and when the transaction was not fraud we coded it as (1 = no)

**Data Engineering**

Gathering necessary information is a vital part of data collection. As we mentioned in our introduction our dataset is called “DataCo SMART SUPPLY CHAIN FOR BIG DATA ANALYSIS” which was collected by the company DataCo. Upon collecting this text data, our team thoroughly checked the data to make sure it contained the right information, this included checking the parameters and quantity of the raw dataset; to make sure the raw dataset was sufficient enough to predict our target feature.

**Figure 2**

*Sample data*

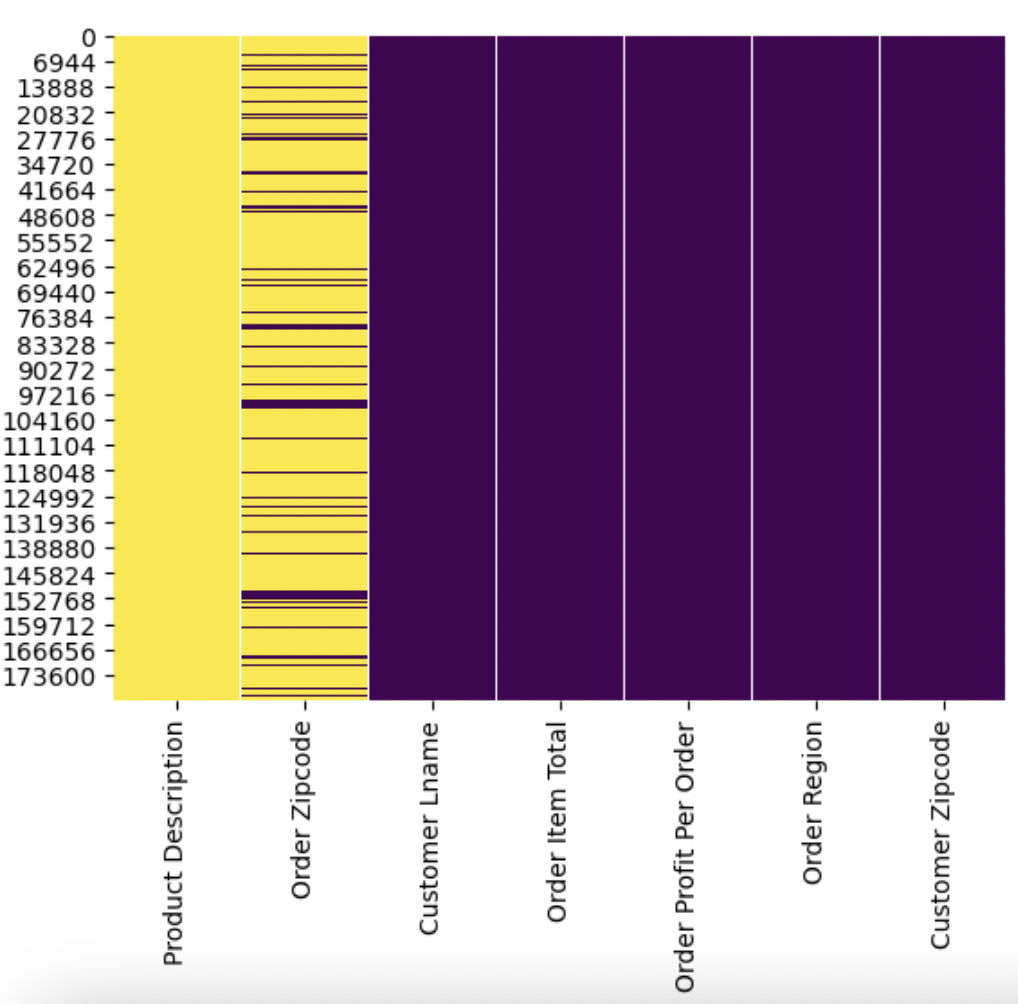


*Note.* The above figure shows a sample data snippet

Our group continued to do Exploratory Data Analysis on our dataset. There were around 24 categorical variables and the remainder were numerical. Only four columns (out of the 53 total) had missing data. Missing values in data can occur for a variety of reasons, including unrecorded observations and data corruption. Many machine learning methods do not accept data with missing values, therefore handling missing data is critical. Having only 4 columns with missing values ensured the efficiency of our data and saved us time for having a low amount of missing values to deal with. The below figure shows the columns that contained the most missing values.

**Figure 3**

*Missing data*

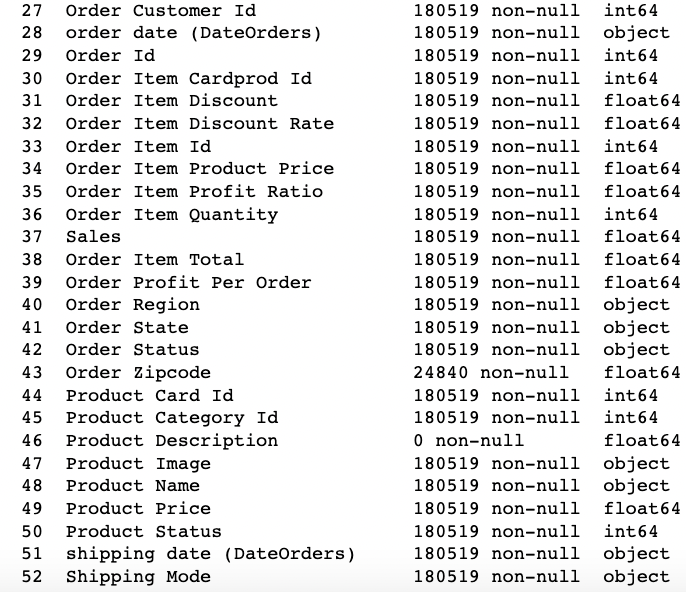
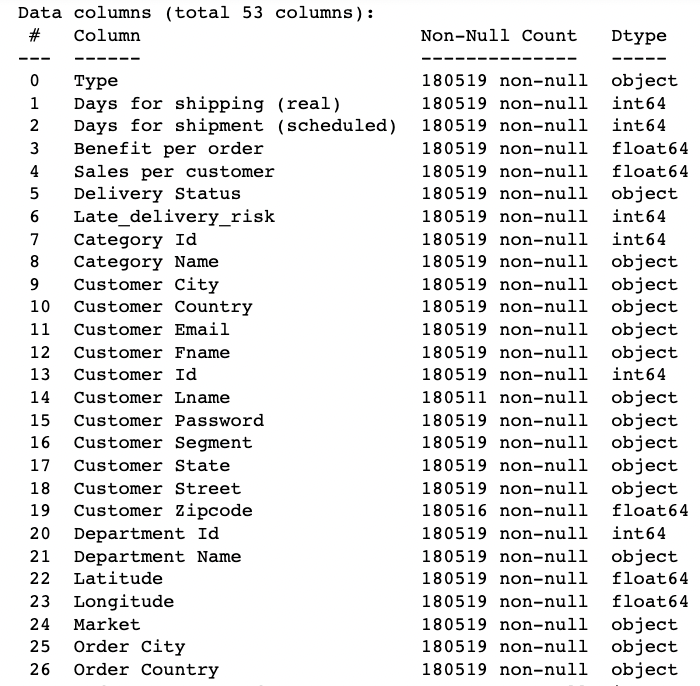


*Note.* The above figure shows the distribution of null values in yellow per input feature in the x-axis and the count of total data points in the y-axis.

Data preparation, also known as data preprocessing, is performed in order to make the data more comprehensible so that machine learning algorithms can function efficiently. The datasets include numerical, textual, and null values. We need data pre-processing when we comprehend the data, which includes data cleaning, which manages missing and noisy data, eliminating outliers, and handling inconsistent data. The last stage is data transformation, which involves data normalization and reformatting in order to make our dataset suitable for machine learning models. The figure below shows the information of our dataset.

**Figure 4**

*Data Description*



*Note.* Figures show the DataCo data description showing the data types and missing values in the dataset

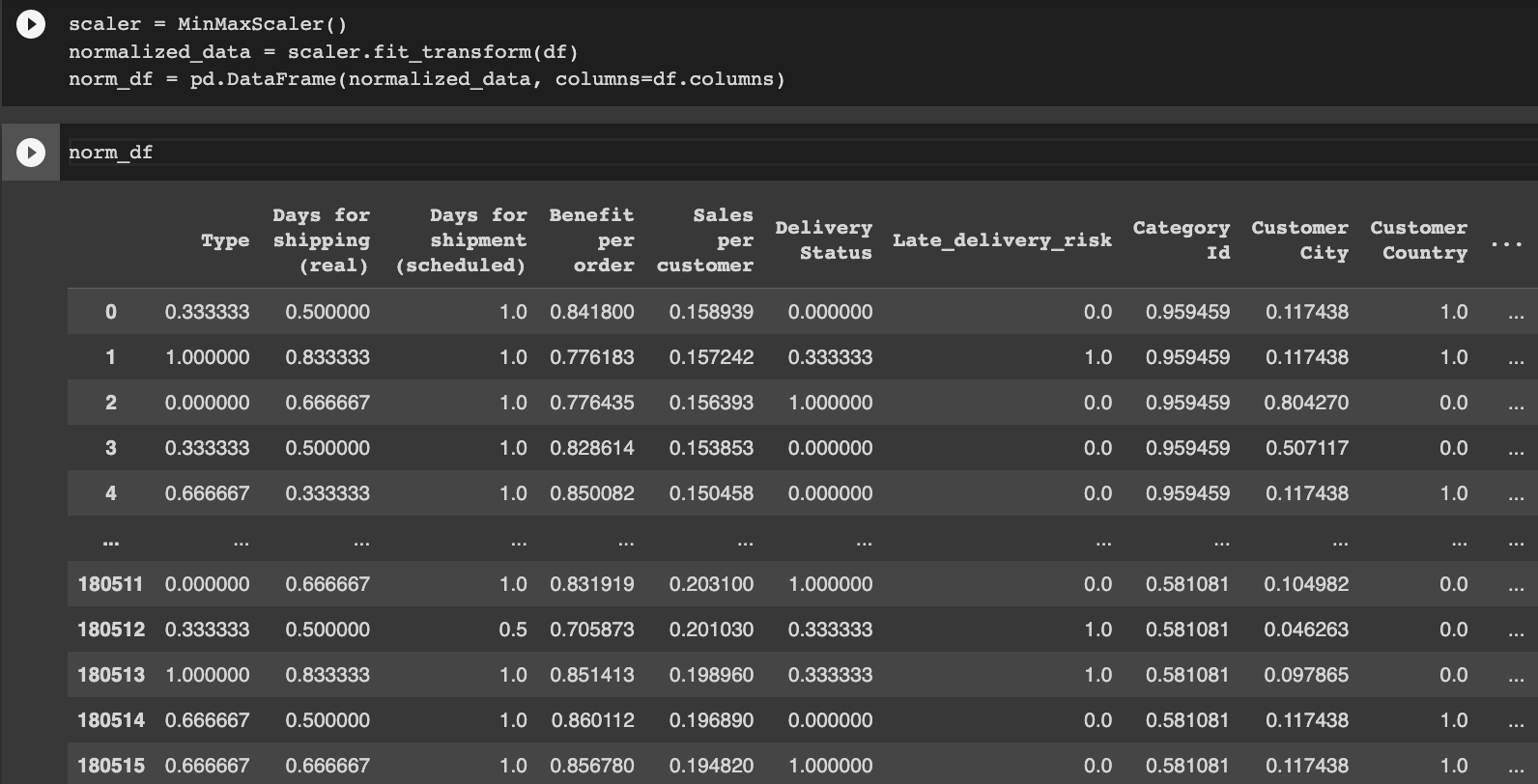
We noticed missing values when we first imported the dataset and did some preliminary data exploration. To handle missing data, we used the dropna() function to remove them. We performed preliminary text pre-processing to detect noise in our text data. There were more than fifty columns, some of which had white spaces that we stripped with the str.strip() technique. We also wanted to find and eliminate outliers in our dataset. To handle these outliers, we applied the InterQuartile Range (IQR). Quartile 1 is the lowest quartile, containing 25% of the data, whereas quartile 3 contains 75% of the data. We created a range to detect outliers, and every data point that goes outside of this range is considered an outlier. The range for these outliers are defined as data points that are less than Q1 - 1.5 IQR or greater than Q3 + 1.5 IQR. As a result, any data point that was less than the Lower Bound or greater than the Upper Bound was removed. Since certain columns also held the date as a string, we converted the columns from a string to datetime, then from datetime to a single day of the week, in order to analyze day of week rather than time and hour.

Most columns also included duplicate columns, for example, category name had a second column with category code, therefore the information in both columns was the same. In the initial dataset, the customer name and password were banned for privacy concerns, and we didn't need to know their names since we already had the customer id. So we removed the columns that were no longer relevant to us. Label encoding was also employed to translate categorical labels into numeric form, allowing them to be read by machine.

We also wanted to transform our data by normalizing it before training for our machine learning models. Normalization is also a frequently used technique. We normalize our data to improve the performance of our Machine Learning algorithms. Normalization also minimizes the training process's sensitivity to feature size. As a result, the coefficients improve after training. To normalize, we'll utilize the sklearn package's MinMaxScaler() function. The figure below shows our data in normalized format.

**Figure 5**

*Data after Normalization*

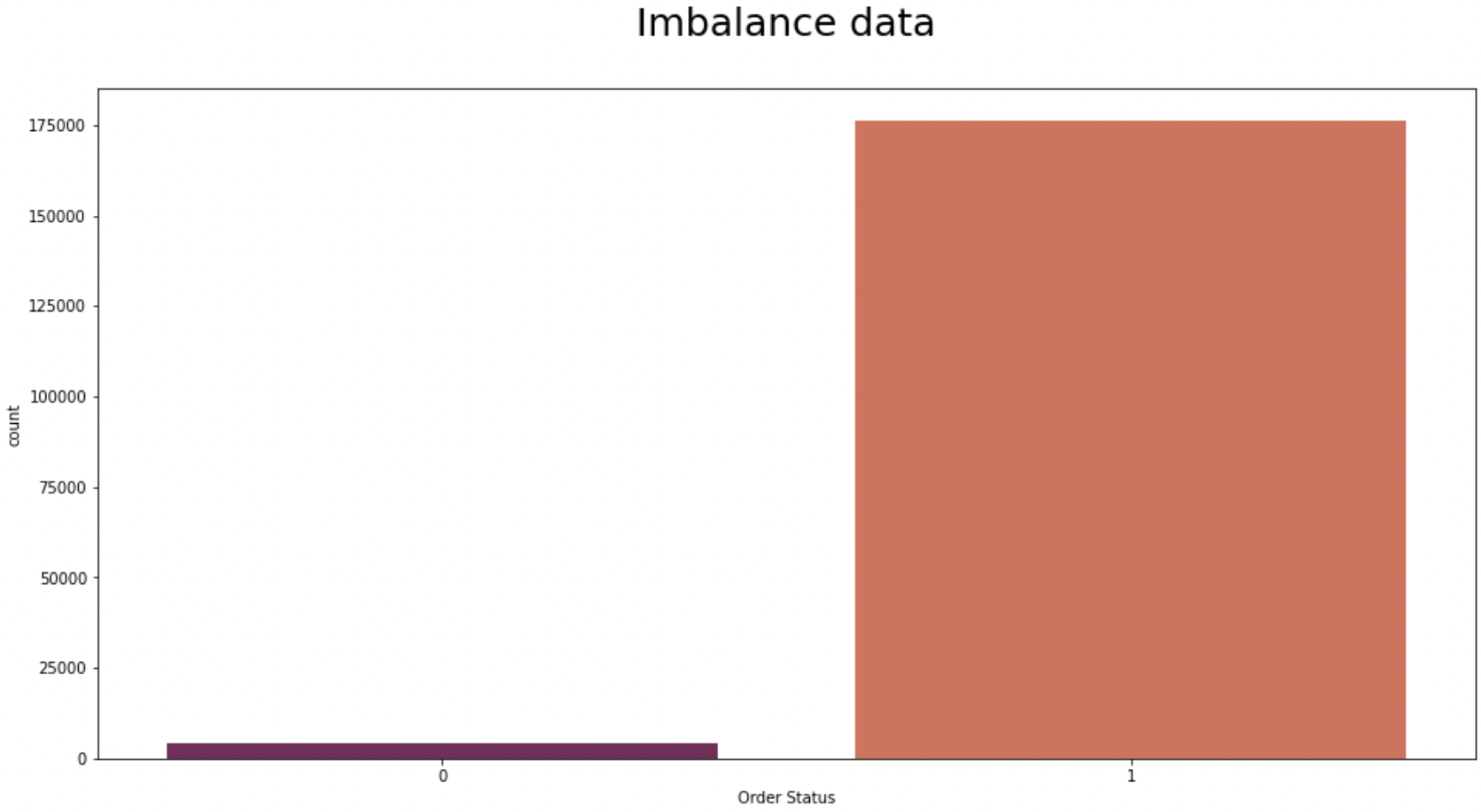


*Note.* The figure shows a sample of data after performing Data Normalization.

To make sure our data wasn't imbalanced we did a count plot on our target feature and realized it was imbalanced. To deal with imbalanced data, we applied a method known as Synthetic Minority Oversampling Technique (SMOTE). Smote is a statistical method used to boost the amount of instances in our dataset. The component produces new instances based on current minority cases that are supplied as input. The figure below shows our Order Status before and after with the use of SMOTE.

**Figure 6**

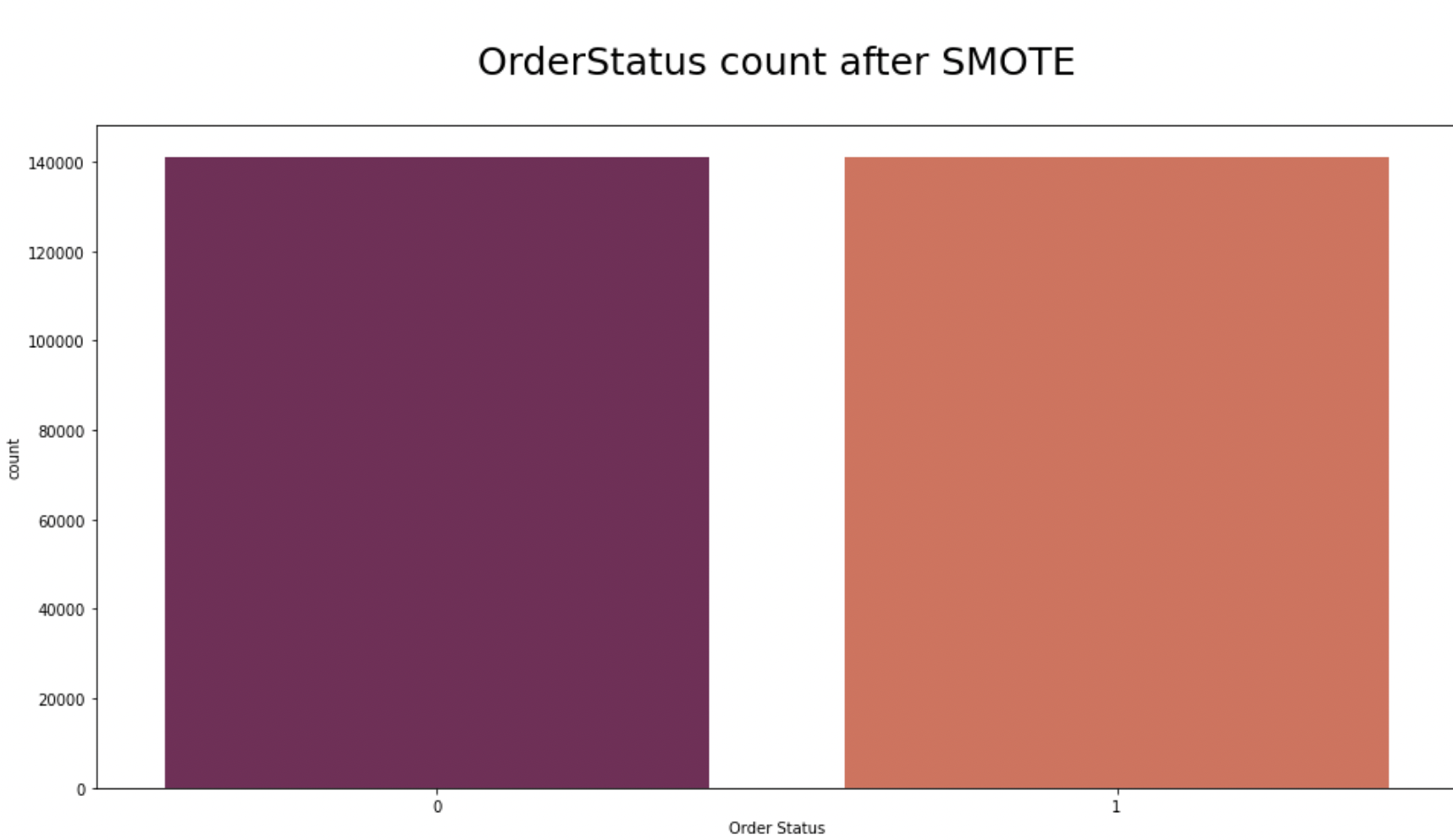
*Target Feature Imbalanced Classes*



*Note.* The graph above shows the target feature, Order Status class imbalance, where ‘0’ is the Fraudulent Transactions and ‘1’ represents the Non Fraud transactions

**Figure 7**

*Target Feature Balanced Classes*



*Note.* The graph shows that the target features classes are now balanced after performing the Data balancing technique, SMOTE.

**Models**

From the data exploration, we understood that our project objective is to flag a transaction fraudulent or not which is a classification problem. From our technology and literature survey, we identified a few classification machine learning models that identify fraudulent transactions. The model takes relevant data columns as input and learns the patterns in which a transaction has been identified as either fraud or Non fraud based on the Order Status feature. We proposed four machine learning classification models and their comparison in this section.

***Decision Tree:***

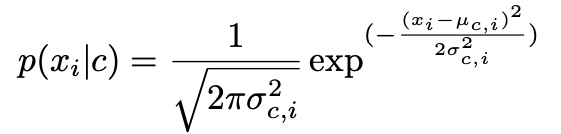
Decision trees is a tree type classification algorithm where each node of the tree presents a feature or attribute while each link in the tree shows decision and the leaf provides an outcome. Decision tree output can be categorical or a continuous value. The decision tree model is known to mimic human thinking by learning from the data and making interpretations. Decision trees are comparatively faster than other techniques, however when the trees grow complex and dense, the model tends to overfit. The trees use various algorithms to recognize the best features, the best split to build the tree and hence produces a subset of the population.

Decision trees are known to map non linear relationships well unlike linear models, hence we chose this model for our classification problem.

***Gauassian Naive Bayes***

One of the top 10 data mining methods is the naive Bayes classifier, which is an effective classifier. In many different applications, including text categorization, document evaluation, and data stream classification, Naive Bayes is a helpful classifier that is frequently employed. The generative model-based classifier Naive Bayes has a quick learning and testing cycle.The Bayesian rule and probability theorems are the foundation of Bayesian classifiers. It has been established that the challenge of developing an ideal Bayesian classifier from training data is NP-hard. Naive Bayes, a condensed form of the Bayesian classifier, makes two assumptions. The first is that attributes are conditionally independent given the class label, and the second is that no latent attribute influences the label prediction process.

Gaussian naïve Bayes technique assumes that attribute values will have a Gaussian distribution given the class label. Assume, for instance, that ith attribute is continuous and that, given the class label c, c,i and c2,i, respectively, reflect its mean and variance. With the class label c, equation given below, commonly known as the normal distribution, calculates the likelihood of seeing the value xi in the ith attribute. We used the Gaussian Naive Bayes model to test if the most important features that remained after feature selection were independent of each other and could result in a strong predictor.

******

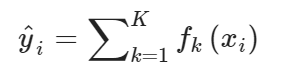
***Random Forest***

A Random forest model can be implemented as a classification model as well as a regression model. For this project we will use random forest as a classification model. Random forest is an ensemble technique, which is a collection of multiple decision trees which work together to predict a common goal. The prediction made from each decision tree is combined and the majority output is considered as the final output. For example, if we build a random forest which contains 100 decision trees, where 80 trees predict output as ‘1’ and the other remaining 20 trees predict ‘0’, then the final output is given as ‘1’, as it is the majority output. The number of decision trees can be any random number, where multiple trees can be overfitted, but the combination of these all trees together is not prone to overfitting. Random forest models are difficult to interpret because of the multiple decision trees present in it.

For this project as the data present is non linear, models like Naive Bayes and Support vector machines without any kernels are not relevant to use. Meanwhile random forest has shown good performance for problems having non linear relationships because of the decision trees present in it. So when the data is having some non linearity random forest is used.

***XGBoost***

An application of gradient boosting algorithm, XGBoost is faster when compared to other implementations of gradient boosting. Boosting in general is an ensemble model where improvements are made based on the performance of existing models. Gradient boosting uses an individual weak learner like a decision tree, iteratively adding new trees to minimize the objective function. These iterations continue until the specified boosting iterations are met and the prediction model obtains the final form. Gradient descent is used to achieve this efficiently and reduces the objective function by finding the steepest descent. XGBoost further implements improvements thus resulting in overall generalization and speed of computation. XGBoost uses second order gradients to obtain a better understanding of the direction of loss functions and uses regularization techniques to reduce model complexity and overfitting.The XGBoost model can be expressed as shown in the equation below.



Where the yi^ is the predicted class , xi is a vector of input features in our data, while fk is a tree at the k-th instance. Since XGBoost addresses the non linearity and handles the disadvantages of decision trees and random forest models, hence we chose this model for our project.

**Feature Selection Methods**

This section covers the feature selection methods we applied. The different selection methods and their respective criteria should give us a robust amount of predictors to choose from. They will all be compared to models that were trained with no feature selection applied to the data.

***No Feature Selection (Baseline)***

This group of models are trained on all the cleaned and preprocessed data. This serves as the baseline or benchmark to compare the following models to determine which feature selection method worked best with our data and models. This may favor the random forest and XGBoost models as both are ensemble models that feature bagging and subspace sampling. They handle larger and high dimensional datasets well compared to decision trees and Gaussian Naive Bayes models, which may be more sensitive to the noise in the data.

***Correlation***

We utilized Pearson’s Correlation coefficient to determine the most relevant features. We explored the correlation between the input features and the target feature using Seaborn’s heatmap. We established threshold of |.015| and selected all the features that had a correlation with the target feature that was greater than or equal to the threshold. This reduced the input features to only four: Type, Delivery Status, Late\_delivery\_risk, Order Region.

***Linear SVC***

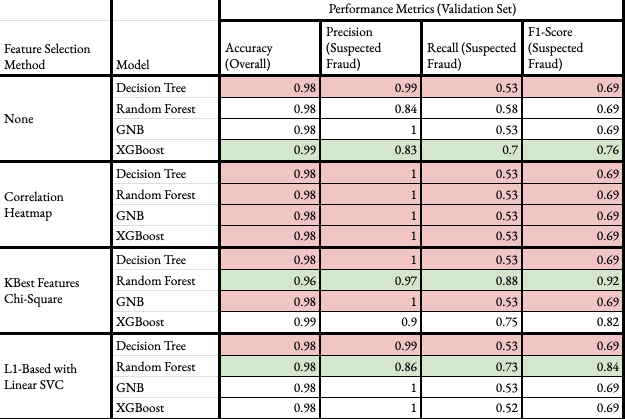
We utilized sklearn’s feature\_selection module to perform feature selection. Specifically, we used SelectFromModel and specified using the LinearSVC model. A Linear SVC model was created which penalized the L1 norm. This results in features with many of the features having estimated coefficients of zero. We then extracted the features with a non-zero coefficient and used them to train our models. This method resulted in the following 15 features: Type, Days for shipping (real), Benefit per order, Delivery Status, Late\_delivery\_risk, Customer Segment, Customer Zipcode, Order City, Order Country, order date (DateOrders), Order Item Id, Order Profit Per Order, Order State, shipping date (DateOrders), Shipping Mode.

***K-Best Features***

This feature selection method also came from sklearn’s feature\_selection. In this method we used the SelectKBest method. This utilizes univariate statistical tests to determine the most influential features for the model. We used the chi-square value to determine the K best features. The chi-square value shows if the input features are related to the output feature. We wanted to include all the related features but needed to make sure we included the best features without including irrelevant features. We tested 5, 10, 15, and 25 features to determine the ideal amount of features to include. We found that for the decision tree, random forest, and Gaussian Naive Bayes models that the ideal number of features was 10 whereas for XGBoost the ideal number was 25. The best performing model in this group had the following 10 features: Type, Delivery Status, Late\_delivery\_risk, Customer Country, Customer Id, Customer Segment, Customer Zipcode, Order Customer Id, order date (DateOrders), Order Region.

**Figure 7**

*Proposed Methods and Models Performance*

****

*Note.*  The above figure shows the comparative results of the proposed feature selection methods and the models to identify the best performing methods and models to implement classification for fraud or non fraud transactions.

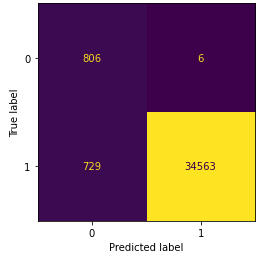
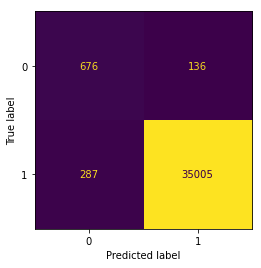
The figure above displays the results of each model grouped by the feature selection method applied. Each model was evaluated using accuracy, precision, recall, and F1-scores with respect to the Fraudulent class. Green rows indicate the highest performing model for that feature selection method while red rows show the worst performing model or models.

***No Feature Selection (Baseline)***

The models with no feature selection performed act as the baseline for this study. The decision tree performed the worst out of the four. The accuracy and precision scores initially suggest strong performance, but the low recall and F1-score indicate high false negatives for the Fraudulent class. The XGBoost model performed the best out of the group with similarly high accuracy and precision, but higher recall and F1-score. The high accuracy shows that the model performed well at differentiating between the two classes over all, however the data imbalance between the two classes caused this figure to bloat as it is strong at predicting the majority class. While the XGBoost model had a lower precision score than the decision tree, having higher scores for recall and F1-score shows that the model is better at finding more of the Fraudulent instances. Both models’ confusion matrices further illustrate the difference in performance. While the decision tree has a higher True Positive amount, it also has a high number of False Negatives, relative to the number of instances of the class. The XGBoost model has higher False Positives and lower True Positive predictions, but it has a much lower amount of False Negatives showing that the model is more nuanced in classifying fraudulent instances. The decision tree overwhelmingly assumes an instance to be fraudulent which drives the number of false positives up, while the XGBoost does better to generalize its predictions. Note that the decision tree is on the left and the XGBoost is on the right in the figure below.

**Figure 8**

*No Feature Selection - Confusion Matrix*

*Note.* The above comparative confusion matrix shows the Decision Tree on the left and XGBoost on the right.

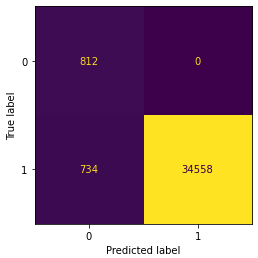
***Correlation***

Using the correlation heatmap for feature selection yielded the worst aggregate results. Each model had identical, poor results. Examining their confusion matrix, shown below, shows that each model had zero false negative predictions but almost an equal number of true positives and false positives. This can be a result of an over-reduction of features. There were only four features that met the arbitrary threshold that was set. This is less than half of the number of features used after performing the KBest Features method, and almost one-fourth of the features used in the L1 method. The over-reduction may have led to a rigid model regardless of the algorithm. Anytime any of the models saw validation data that slightly resembled fraudulent instances from the training set, they automatically predicted fraud with no indication of robustness or generalizability. This is further detailed below in the Observations section.

This poor result is also largely due to taking features that were correlated with the target feature rather than eliminating features that were correlated with each other.

**Figure 9**

*Correlation - Confusion Matrix*

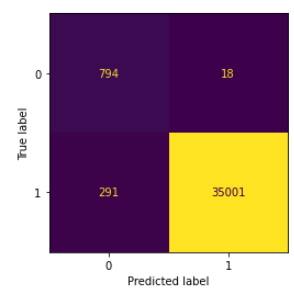
**

***Linear SVC***

The top performing model when L1-Based feature selection was applied was the random forest model. Once again, other models had higher precision scores when predicting instances of Fraud, but the higher recall and F1-scores indicate that it made less errors in its predictions of Fraud.

**Figure 10**

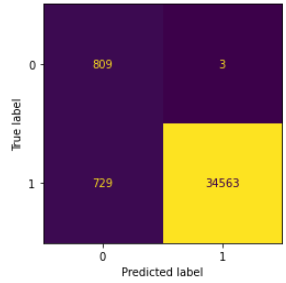
*Random Forest with L1-Based Feature Selection*



The worst performer was the decision tree with no change in performance from the baseline model. The decision tree continued to overfit to the training data and over-predicted instances of fraud.

**Figure 11**

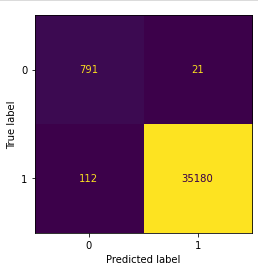
*Decision Tree Confusion Matrix with L1-Based Feature Selection*



***KBest Features***

Applying the KBest Features method yielded the best results overall. The random forest model performed best overall with the highest precision, recall, and F1-score despite not having the highest accuracy overall. Random Forest’s algorithm utilizes bagging with replacement and subspace sampling. This coupled with selecting the 15 best features possibly resulted in many diverse trees trained on the most impactful features. It was able to generalize well when faced with the validation set. The confusion matrix below shows a high amount of true fraudulent predictions and a balanced amount of false positives and false negatives.

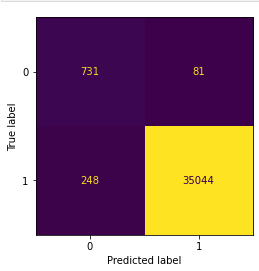
**Figure 12**



XGBoost did not perform as well as the random forest model, but better than the decision tree and Gaussian Naive Bayes models. It is worth noting that the best performing XGBoost model used 25 features, or roughly half the baseline model. This caused a noticeable increase in recall and F1-score. This could be due to the data being large enough to test trees trained on many different combinations of features which allowed for more robust learners to get boosted. The confusion matrix below shows that the XGBoost model performed relatively well, but not to the level of the random forest.

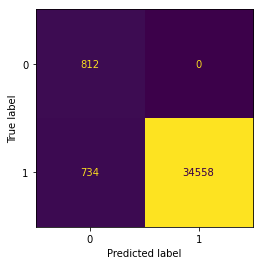
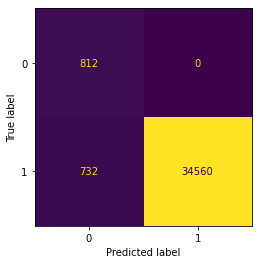
**Figure 13**

*Confusion matrix for XGBoost with KBest Features*



The decision tree and Gaussian Naive Bayes models both performed poorly in identical fashion. Their confusion matrices are nearly identical with an immaterial difference in false positive predictions of Fraud.

**Figure 14**

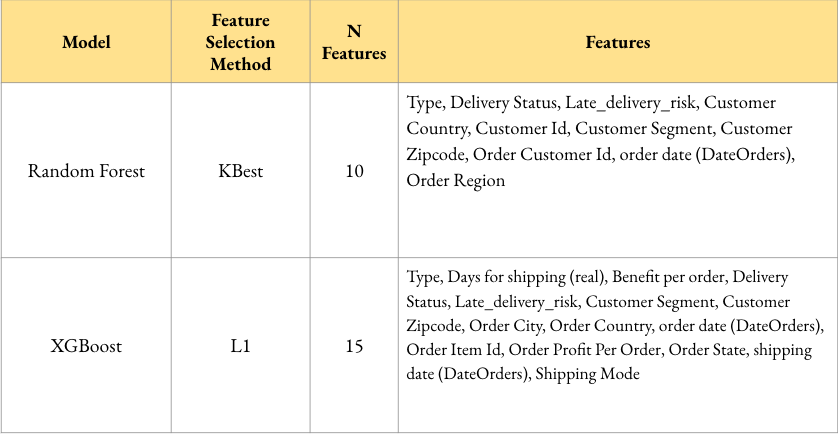
 

*Note:* Left confusion matrix is for the decision tree. Right is for XGBoost

Knowing the best performing model allows for a more focused analysis on the included features. The table below displays a summary of the best predicting random forest model and the features that were included in its modeling. The next section details the new knowledge that was uncovered by investigating these important features.

**Figure 15**

*Best performing Method and Model*

****

**Observations**

***Overfitting Models***

Overfitted models were seen with each level of feature selection. There are a few potential explanations. One being the use of oversampling and specifically using SMOTE. SMOTE works by creating synthetic instances of the minority class in the attempt to create a balanced amount of instances between the classes. These synthetic instances are created by copying using a cluster of instances of the minority class and replicating them close to that original cluster. In our case, this means that SMOTE created instances with values for each feature that are similar to instances of Fraud that it saw. This can lead to overfitting by over-representing a subgroup of Fraud instances making it so that any model that sees new data that resembles the SMOTE data will likely assume it is Fraud. This also decreases any model’s ability to differentiate between Non Fraud instances that may share resemblance to Fraud instances that SMOTE used to generate the synthetic instances.

This effect could be amplified by the reduction in features. The extreme case is seen with the Correlation group. This group only used four input features, which oversimplified the models. Then, being trained with SMOTE data and validated with real data showed that each model greatly overfitted on the training data as seen with the .53 recall score. The validation set likely had instances of Non Fraud that resemble the SMOTE instances of Fraud, causing the high amounts of false positives.

***Regional Effects***

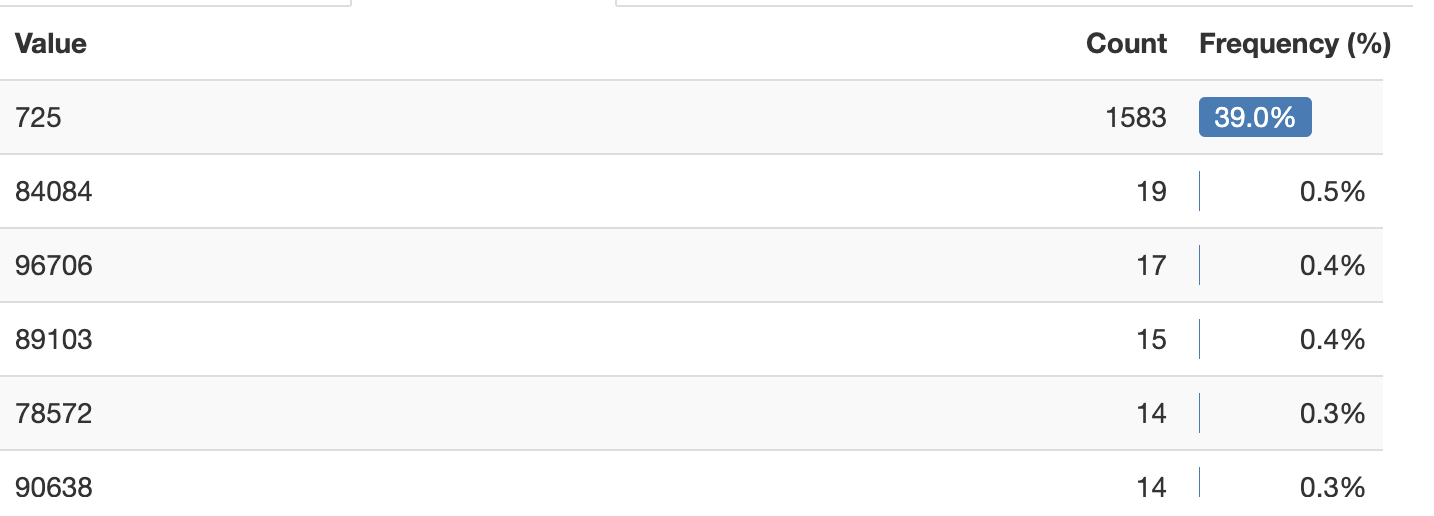
We found that regional features such as the Order Region and Customer Zipcode were parts of the most successful models.

The International Consumer Protection and Enforcement Network (2022) published a dashboard summarizing the findings from over 65 consumer protection agencies from various countries. They found that online shopping fraud was the most common form of fraud and that France, Spain, and the United Kingdom were in the top ten countries that reported the most instances of fraud. This aligns with our dataset as the Western Europe Region had the most instances of fraud orders.

Analyzing the Order Zipcode we found the 725 zip code represented the majority of fraudulent transactions in this specific dataset. Most of the zip codes with Fraudulent transactions only had 20 or fewer instances compared to 725 with 1583 of the 4026 total Fraudulent transactions.

**Figure 16**

*Snippet of Zip Codes with the Most Instances of Fraud*



The 725 zip code is home to Las Vegas County in Nevada, USA. The Las Vegas Review -Journal (2020) shared that according to the Federal Trade Commision, in 2019, the year our dataset was published, Nevada reported the highest incidences of fraud per capita in the United States. Furthermore, the Las Vegas, Henderson Paradise, Nevada metropolitan area, zip code 725, also ranks among the highest fraud per capita among other metropolitan areas.

Going forward, this model could be updated to place more weight on orders originating from Western European countries and being shipped to the Las Vegas Nevada area.

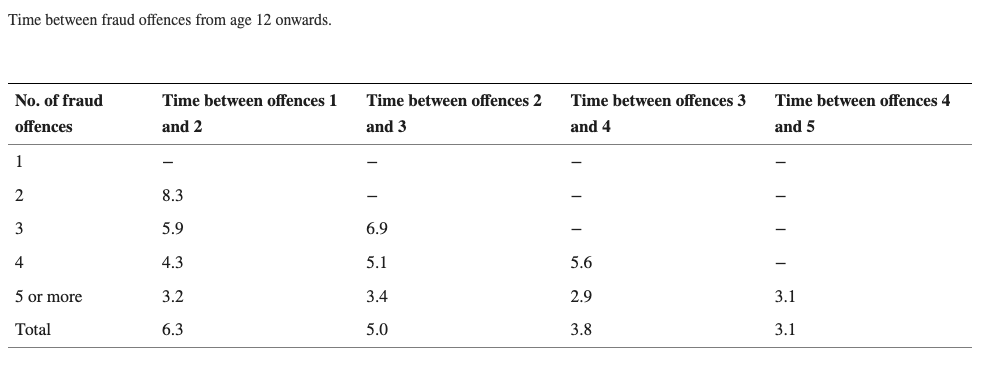
***Repeat Offenders***

Examining the Customer Id feature, we found that there were 1,429 unique customer ID’s while there were 4026 instances of Fraud. Further, there were 1,008 customer ID’s with more than one instance of Fraud. This means that over two-thirds of all the fraudulent orders were connected to repeat offenders and shows that even a single instance of fraud by a new customer may be a precursor for more instances in the future.

This point is supported by Geest et al.’s study (2016) where they tracked the development of fraudsters in the Netherlands. Their findings showed that the younger a person is convicted of their first fraud, the more likely they are to become repeat offenders or career fraudsters. They found that the repeat offenders are convicted in shorter time increments. It can be inferred that once a fraudster is active, they remain active and are frequently committing fraud. Their table below summarizes this point.

**Figure 17**

*Table from Geest et al’s Study Showing Time Between Fraud Offences*



**Conclusion**

The focus of this study was to develop a fraud-detection system built with the publicly available dataset. Our exploratory data analysis led us to choose the feature selection methods of correlation, KBest, and L1-based Linear SVC. We used the decision tree, random forest, Gaussian Naive Bayes, and XGBoost models as our class predictors. We found that the random forest paired with the KBest Features selection method was the best performer as it did not suffer from overfitting like many of the other models had.

Analyzing a couple of the important features exposed the need to scrutinize where orders are shipping from, and where the orders are shipping to. Also, to closely examine any instance of fraud as it may be a precursor for more fraud.

There are some limitations to our study. One being that the data is only of suspected fraud. This means our models include instances where a non-fraudulent order was labeled as fraud within the dataset. Next, there is a shallow understanding of the customers. Characteristics of the customers such as age and sex are not included while they may be important features to determine potential fraud.

**References**

Ashok Kumar. (2021). Ensemble Technique on Predictive Analysis and Fraud Orders Detection using Supervised Machine Learning Algorithms in Supply Chain Management. *Turkish Online Journal of Qualitative Inquiry*, *12*(7), 12161–12175.

Constante-Nicolalde, FV., Guerra-Terán, P., Pérez-Medina, JL. (2020). Fraud Prediction in Smart Supply Chains Using Machine Learning Techniques. In: Botto-Tobar, M., Zambrano Vizuete, M., Torres-Carrión, P., Montes León, S., Pizarro Vásquez, G., Durakovic, B. (eds) Applied Technologies. ICAT 2019. Communications in Computer and Information Science, vol 1194. Springer, Cham. https://doi.org/10.1007/978-3-030-42520-3\_12

Dentzer, B. (2020, June 13). Nevada tops list of 2019 fraud reports. *Las Vegas Review-Journal*. https://www.reviewjournal.com/news/politics-and-government/nevada/nevada-tops-list-of-2019-fraud-reports-1975471/

International Consumer Protection and Enforcement Network. (2020). *econsumer.gov International Fraud Report*. Tableau Public. Retrieved December 13, 2022, from <https://public.tableau.com/app/profile/federal.trade.commission/viz/eConsumer/Infographic>

Markovic, S., Bryan, J.L., Rezaee, R. *et al.* Application of XGBoost model for in-situ water saturation determination in Canadian oil-sands by LF-NMR and density data. *Sci Rep* 12, 13984 (2022). https://doi.org/10.1038/s41598-022-17886-6

P. Porouhan and W. Premchaiswadi, "Big Data Analytics of Supply Chains with Process Mining," 2021 19th International Conference on ICT and Knowledge Engineering (ICT&KE), 2021, pp. 1-5, doi: 10.1109/ICTKE52386.2021.9665705.

van der Geest, V. R., Weisburd, D., & Blokland, A. A. J. (2016). Developmental trajectories of offenders convicted of fraud: A follow-up to age 50 in a Dutch conviction cohort. *European Journal of Criminology*, *14*(5), 543–565. https://doi.org/10.1177/1477370816677620

**Appendix A**

| **Variable** | **Description** | **Type** |
| --- | --- | --- |
| Type | Type of payment | Categorical |
| Days for shipping(real) | Actual time taken for shipping | Numerical |
| Days for shipment | Estimated time for shipping | Numerical |
| Delivery Status | Status update on delivery | Categorical |
| Late\_delivery\_risk | Delivery risk indicator | Numerical(binary) |
| Category Name | Name of item category | Categorical |
| Customer City | City name of customer | Categorical |
| Customer Country | Country name of customer | Categorical |
| Customer Segment | Type of the customer | Categorical |
| Customer State | State name of customer | Categorical |
| Department Name | Name of the department for the product | Categorical |
| Market | Country’s region | Categorical |
| Order City | City name where product is ordered | Categorical |
| Order Country | Country name where product is ordered | Categorical |
| Order Item Discount Rate | Discount rate for product | Numerical |
| Order Profit Per Order | Profit for the order | Numerical |
| Order Region | Region of the ordered product | Categorical |
| Order State | State of the ordered product | Categorical |
| Product Name | Name for the product | Categorical |
| Product Price | Price for the product | Numerical |
| Order date (DateOrders) | Date on which the order is made | Categorical |