University Dzemal Bijedic Mostar

Faculty of information technologies

**Credit Card Fraud Detection using Logistic regression, Random Forest and XGBoost algorithms**

Seminarski rad iz predmeta Umjetna inteligencija

**Mentor: Student:**

Prof. dr. Migdat Hodžić Zorka Kunovac

Mostar, januar 2022. godine

Contents

[1. Introduction 1](#_Toc93840231)

[2. Business understanding and Data understanding 3](#_Toc93840232)

[3. Data preparation 5](#_Toc93840233)

[4. Modeling 11](#_Toc93840234)

[A. Logistic regression 13](#_Toc93840235)

[B. Model ensemble and Random Forest 15](#_Toc93840236)

[C. XGBoost classifier 17](#_Toc93840237)

[5. Evaluation and deployment 19](#_Toc93840238)

[6. Conclusion 20](#_Toc93840239)

[7. References 21](#_Toc93840240)

Table of Figures

[Figure 1: Evolution of total value of card fraud using cards issued within SEPA.[1][3] 2](#_Toc93840253)

[Figure 2: Learning a model from a set of historical instances [5] 3](#_Toc93840254)

[Figure 3: Using the model to make predictions for new query instances [5] 3](#_Toc93840255)

[Figure 4: Actual dataset part 1 5](#_Toc93840256)

[Figure 5: Actual dataset part 2 6](#_Toc93840257)

[Figure 6: Statistical summary part 1 6](#_Toc93840258)

[Figure 7: Statistical summary part 2 6](#_Toc93840259)

[Figure 8: Statistical summary part 3 7](#_Toc93840260)

[Figure 9: Countplot 7](#_Toc93840261)

[Figure 10: Correlated with Class 8](#_Toc93840262)

[Figure 11: Heatmap of variables 9](#_Toc93840263)

[Figure 12: Splitting the dataset 9](#_Toc93840264)

[Figure 13: Feature scaling 10](#_Toc93840265)

[Figure 14: Confusion matrix [11] 11](#_Toc93840266)

[Figure 15: Accuracy formula 12](#_Toc93840267)

[Figure 16: Precision formula [11] 12](#_Toc93840268)

[Figure 17: Recall formula [11] 12](#_Toc93840269)

[Figure 18: F1-score formula [11] 12](#_Toc93840270)

[Figure 19: Logistic regression [13] 13](#_Toc93840271)

[Figure 20: Logistic regression building model 13](#_Toc93840272)

[Figure 21: Measured performance for Logistic regression 14](#_Toc93840273)

[Figure 22: Cross validation for Logistic regression 14](#_Toc93840274)

[Figure 23: Random Forest [15] 15](#_Toc93840275)

[Figure 24: Measured performance for Random Forest 16](#_Toc93840276)

[Figure 25: Cross validation for Random Forest 16](#_Toc93840277)

[Figure 26: XGBoost classifier [18] 17](#_Toc93840278)

[Figure 27: Measured performance for XGBoost 18](#_Toc93840279)

[Figure 28: Cross validation for XGBoost 18](#_Toc93840280)

[Figure 29: Final results 19](#_Toc93840281)

[Figure 30: Predicting a single observation 19](#_Toc93840282)

Abstract

In the modern world credit cards have become a common thing, life is almost unimaginable without them. The percentage of people using credit cards is growing every day in almost every country. This also has an impact on the increase in the number of credit card frauds.

Credit card frauds are probably most serious fraud type, especially given the amount of time and money banks invest in the security of their systems and yet credit card fraud is one of the most common types of fraud. It is important that credit card companies can recognize fraudulent credit card transactions so that customers are not charged for items that they did not purchase.

The goal of this project is to maximize the percentage of detected fraudulent credit card transactions using the three most accurate machine learning algorithms.

Keywords: credit cards, frauds, transactions, machine learning, algorithms

# Introduction

Worldwide financial losses caused by credit card fraudulent activities are worth tens of billions of dollars. One American over ten has been a victim of credit card fraud. [1][2] According to the European Central Bank (ECB) report [1][3], the total level of card fraud losses amounted to €1.8 billion in 2018 in the Single European Payment Area (SEPA).

There exists a lot of scenarios that may lead a fraudster to successfully perform fraudulent payments with a credit card. As technology evolves, both in terms of fraud prevention and ease of use of payment systems, so do fraudster techniques. Fraudsters adapt by moving from the old targets to seeking the vulnerability of modern technologies, they also benefit from the changes in characteristics of genuine transactions.[1]

It is useful to differentiate two transaction scenarios. The first, called card-present (CP) scenarios, refer to scenarios where a physical card is needed, such as transactions at a store (also referred to as a point-of-sale - POS) or transactions at a cashpoint (for instance at an automated teller machine - ATM). The second, called card-not-present (CNP) scenarios, refers to scenarios where a physical card does not need to be used, which include payments performed on the Internet, by phone, or by mail.[1]

This is important because the techniques used to compromise a card vary, depending on whether a physical copy of the card needs to be produced or not. Fraudsters are more likely to exploit the deficiencies of card-not-present scenarios than card-present ones, probably because CP scenarios have existed for more than two decades now and have become robust to fraud attacks. Another reason is physical barriers can often help to prevent CP fraud. In the 2019 Nilson report is stated that CNP scenarios accounted for 54% of all losses to fraud for the year 2018, while only accounting for less than 15% of all purchase worldwide (CNP+POS+ATM). [1][4] The percentage of CNP fraud in Europe is much higher and was reported to account for 79% of all transactions from cards issued within SEPA in the 2020 report on card fraud of the European Central Bank [1][3] as shown in the Figure 1.

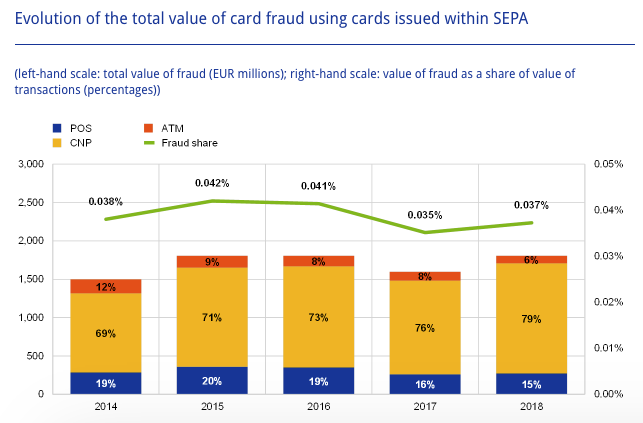


Figure 1: Evolution of total value of card fraud using cards issued within SEPA.[1][3]

# Business understanding and Data understanding

Credit card fraud detection is complex problem. It requires finding fraudulent transaction out of millions of daily transactions. Due to fast growing data today is impossible for human specialist to detect patterns from transaction data. Because of that, it is necessary to use machine learning techniques.

Machine learning is an automated process that extracts patterns from data. To build the models used in predictive data analytics, supervised machine learning was used in this project. Supervised machine learning techniques automatically learn a model of the relationship between a set of descriptive features and a target feature based on a set of historical examples, and then it is possible to use this model to make predictions for new instances. These two separate steps are shown in Figure 2 and Figure 3 [5].

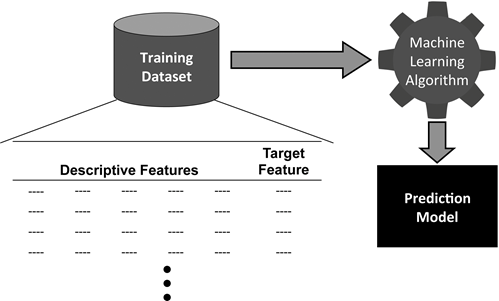


Figure 2: Learning a model from a set of historical instances [5]

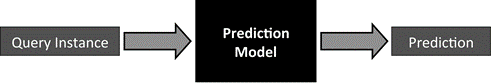


Figure 3: Using the model to make predictions for new query instances [5]

Predictive data analytics projects are not handed fully formed. Rather, analytics projects are initiated in response to a business problem, and it is necessary to decide how to address this business problem using analytics techniques. The first part of this document was attempted to answer questions: “What is the business problem? What are the goals that the business wants to achieve?” to define business problem. Again, business problem is that credit card companies should be able to recognize fraudulent credit card transactions so that customers are not charged for items that they did not purchase. So, for this problem will be built a machine learning classification model to predict whether a transaction is fraudulent or not.

Before starting the analysis, it is necessary to understand more about data that is available for this analysis. The dataset, that will be analyzed contains transactions made by credit cards in September 2013 by European cardholders. This dataset are transactions that occurred in two days, and there are 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions. [19]

It contains only numerical input variables which are the result of a PCA transformation. Principal component analysis (PCA) is a technique for reducing the dimensionality of such datasets, increasing interpretability but at the same time minimizing information loss. It creates new uncorrelated variables that maximize variance. [6] Unfortunately, due to confidentiality issues, the original features and more background information about the data cannot be provided. Features V1 to V28 are components obtained with PCA. The only features which have not been transformed with PCA are 'Time' and 'Amount'. Feature 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset. The feature 'Amount' is the transaction Amount. Feature 'Class' is the response variable, and it takes value 1 in case of fraud and 0 otherwise. [19]

# Data preparation

Project code is written in Google Colab. Colab is a free Jupyter notebook environment that runs entirely in the cloud. Code is separated into four sections: Data preprocessing, building model, final model and predicting a single observation. Two stages can be recognized. The first stage consists of building a prediction model from a set of labeled historical data. This process is called supervised learning since the label of the transactions (genuine or fraudulent) is known. In the second stage, the prediction model obtained from the supervised learning process is used to predict the label of new transactions.[9]

A prediction model is a parametric function with parameters θ, also called a hypothesis, that takes an input x from an input domain X⊂Rn, and outputs a prediction y=h(x,θ) over an output domain Y⊂R [7][8][9]:

h(x,θ):X→Y

The input domain X usually differs from the space of raw transaction data for two reasons. First, for mathematical reasons, most supervised learning algorithms require the input domain to be real-valued, that is, X⊂Rn, which requires to transform transaction features that are not real numbers, such as timestamps or categorical variables. Second, it is usually beneficial to enrich transaction data with other variables that may improve the detection performance of the prediction model. This process is referred to as feature engineering (also known as data preprocessing). [9]

For fraud detection, the output domain Y is usually the predicted class for a given input x, that is Y= {0,1}. Considering that output class is binary, these prediction models are called binary classifiers. [9]

Project coding start within section called data preprocessing and first step was importing the libraries and dataset. Imported libraries are: Numpy is used for working with arrays, pandas to work with data frame and dataset. Matplotlib.pyplot and seaborn are digital visualization libraries.

First things checked through data exploration part of data preprocessing are dataset.head() which is function which returns the first n rows for the object based on position and dataset shape. Dataset shape in this case is, precisely, 284807 observations and 31 columns.

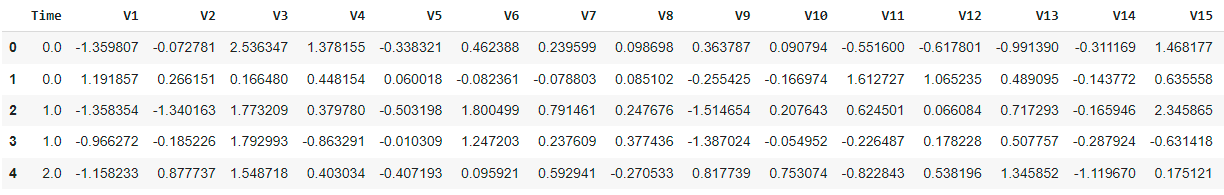


Figure 4: Actual dataset part 1

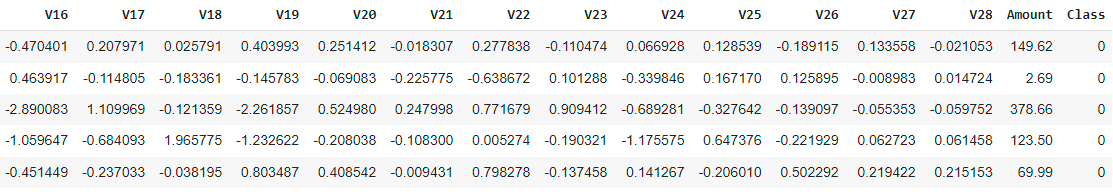


Figure 5: Actual dataset part 2

Actual dataset is shown in the Figures 4 and 5. In pictures is visible that all variables are numerical and that information was also obtained from data insight. Next thing is statistical summary, shown in following figures.

The table in a statistical summary that describes continuous features include a row containing the mean, minimum, 1st quartile, 2st quartile, 3rd quartile, maximum, and standard deviation statistics for that feature as well as the total number of instances in the ABT.

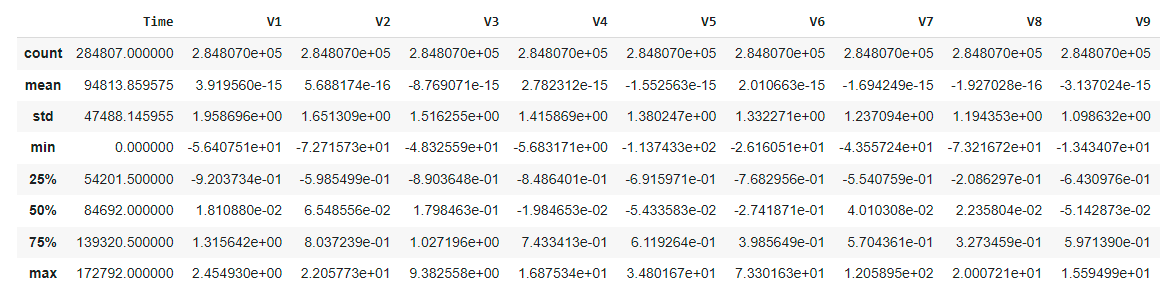


Figure 6: Statistical summary part 1

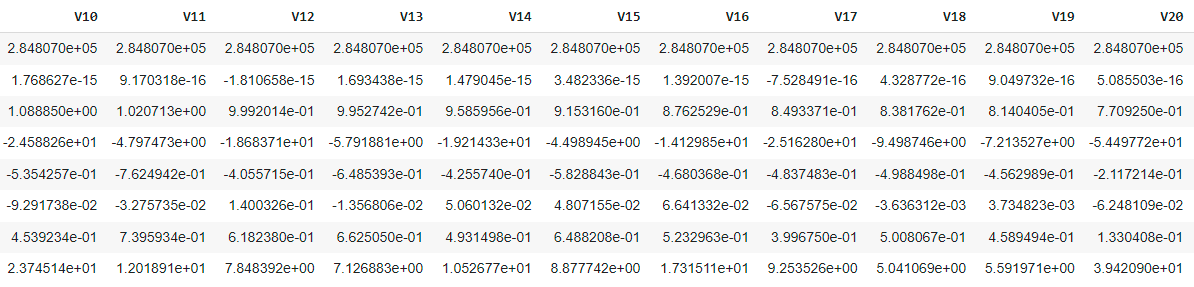


Figure 7: Statistical summary part 2

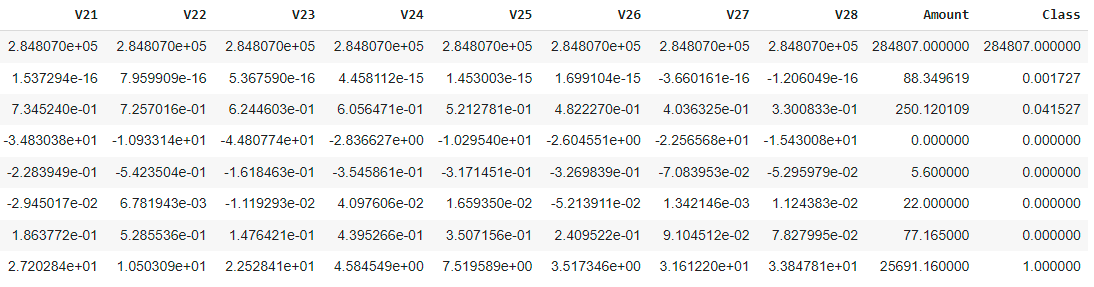


Figure 8: Statistical summary part 3

After statistical summary, next step is dealing with missing values. The simplest approach to handling missing values is to simply drop from an ABT any features that have them. This, however, can result in massive, and frequently needless, loss of data. For example, if in an ABT containing 1,000 instances, one value is missing for a particular feature, it would be extreme to remove that whole feature. As a rule, only features that are missing more than 60% of their values should be considered for complete removal, otherwise more subtle techniques should be used. [5]

Fortunately, this dataset has no missing values so next step is encoding categorical data. This implies checking if there is any categorical data and if there is then next step within encoding categorical data is one hot encoding. It would convert the categorical values into the numbers. It was previously mentioned that all data are numerical so there is no need for one hot encoding, but let imagine that there are categorical values, for example that column Class contains two unique values: F for fraudulent transaction and G for genuine. This means that a new column should be created, for example Class\_F that would ask if transaction is fraudulent or genuine and instead of letters F and G there would be binary numbers 1 for fraudulent and 0 for genuine.

Next part is data visualization. There would be shown different diagrams for easier data understanding. First is countplot of column Class which is target variable.

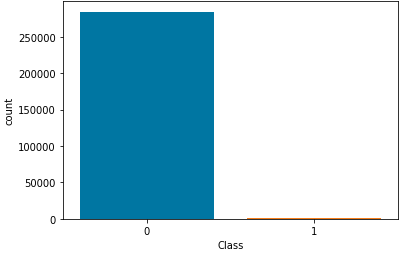


Figure 9: Countplot

0 represent non-fraudulent transactions and 1 represent fraudulent. As mentioned before this is real data and that is reason for such unbalance between fraudulent and non-fraudulent transactions. Sum of non-fraudulent transaction is 284315 and sum of fraud transactions is 492 which is pretty good number considering it is real data. The positive class (frauds) account for 0.172% of all transactions. [19]

Next step is correlation matrix and heatmap.

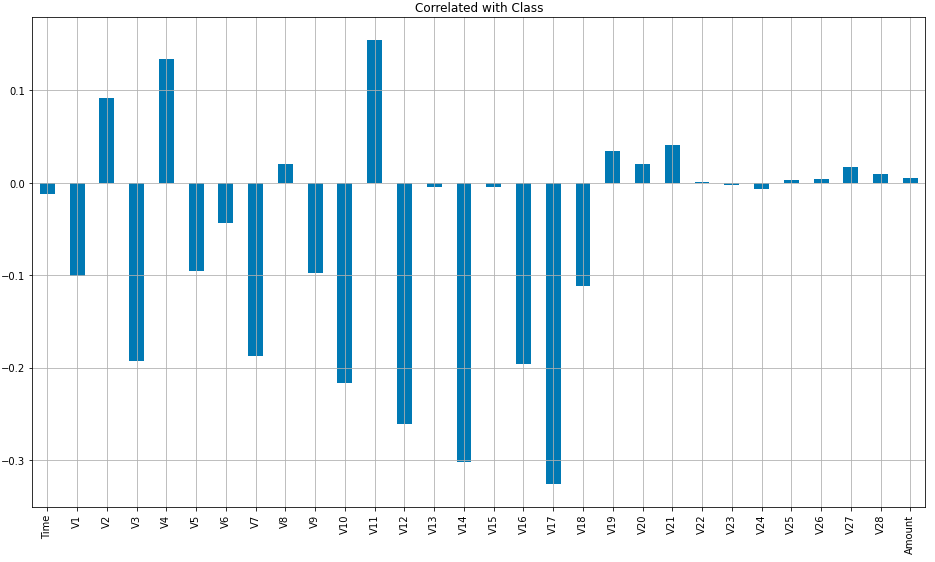


Figure 10: Correlated with Class

In Figure 10 is shown correlation of independent variables with dependent variable Class. Most of variables are somewhat negatively correlated with Class. There are several variables that appear to be positive correlated but it cannot be said to be significant. Strongest correlation with variable Class has variable V17 and it is moderate negative correlation.

In Figure 11 is shown heatmap which show in color how strong is correlation variables with each other. Light color represent strong positive correlation and lightest color is shown diagonally. Variable Amount show positive correlation with variable V7, also with V20. That is because every variable is totally correlated with itself. Darkest color represents strong negative correlation. Variable Amount have negative correlation with variable V2. Purple, which is shown the most represent that there is no correlation between variables.

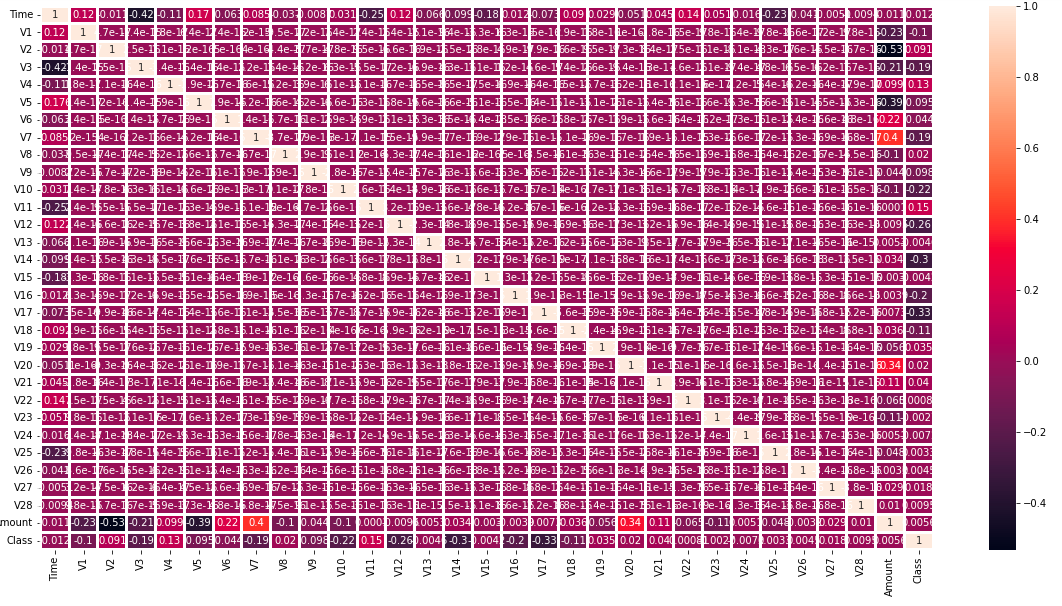


Figure 11: Heatmap of variables

After data visualization next step is splitting the dataset. Dataset is first divided as x and y, where x is matrix of features or independent variables. Those are all variables except Class which is target variable. y is target variable or dependent variable Class. After that train\_test\_split is imported from sklearn.model\_selection and dataset is separated in way that test size is 20% and that left 80% for training. Result of splitting the dataset is shown in figure 12.

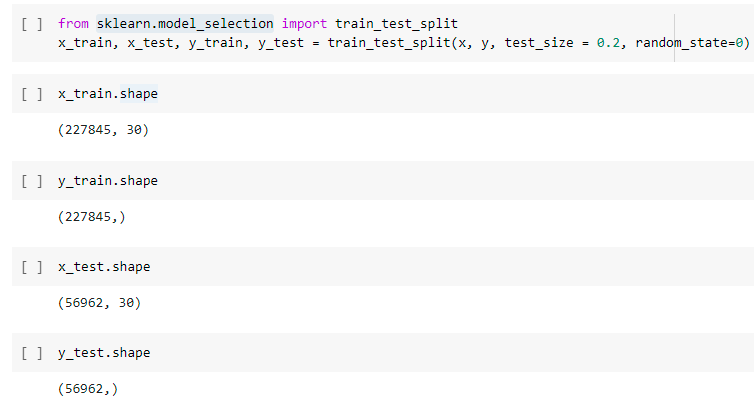


Figure 12: Splitting the dataset

After splitting the dataset, next step is feature scaling. Feature scaling is a method used to normalize the range of independent variables or features of data. In data processing, it is also known as data normalization and is performed during the data preprocessing step.[10]

In the figure 13 is shown process of feature scaling.

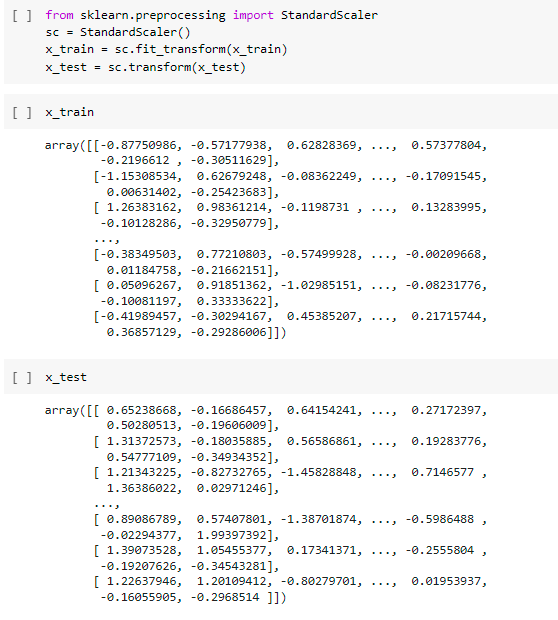


Figure 13: Feature scaling

# Modeling

Data preprocessing part of project is done with feature scaling. In “Building the model” section three algorithms are chosen for modeling and those are: Logistic regression, Random Forest and XGBoost classifier. All three algorithms are classification algorithms as target variable is represented in binary values. Binary classification is the task of classifying the elements of a set into two groups, in this case fraud or not.

The following performance was measured for each of the algorithms: confusion matrix, accuracy score, f1\_score, precision score, recall score and then cross validation.

A Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the number of target classes. The matrix compares the actual target values with those predicted by the machine learning model. This show how well classification model is performing and what kinds of errors it is making.[11]

For a binary classification problem there would be a 2 x 2 matrix as shown below, in figure 14, with 4 values [11]:

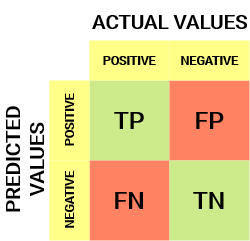


Figure 14: Confusion matrix [11]

The target variable has two values: Positive or Negative, the columns represent the actual values of the target variable, the rows represent the predicted values of the target variable [11]

**True Positive (TP)**

* The predicted value matches the actual value
* The actual value was positive and the model predicted a positive value [11]

**True Negative (TN)**

* The predicted value matches the actual value
* The actual value was negative and the model predicted a negative value [11]

**False Positive (FP) – Type 1 error**

* The predicted value was falsely predicted
* The actual value was negative but the model predicted a positive value [11]

**False Negative (FN) – Type 2 error**

* The predicted value was falsely predicted
* The actual value was positive, but the model predicted a negative value [11]



Figure 15: Accuracy formula

Later in document will be shown that accuracy in Logistic regression is 0.999192. So that seems like fairly good result but dataset in this project is highly imbalanced with mostly non-fraudulent cases so that means model does not predict fraud cases with 99% accuracy but non-fraudulent transactions.

Precision tells how many of the correctly predicted cases actually turned out to be positive [11].



Figure 16: Precision formula [11]

This would determine whether selected model is reliable or not.

Recall tells many of the actual positive cases we were able to predict correctly with selected model [11].

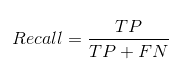


Figure 17: Recall formula [11]

In practice, when precision increase, the recall goes down, and vice-versa. The F1-score captures both in a single value:

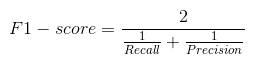


Figure 18: F1-score formula [11]

F1-score is a harmonic mean of Precision and Recall, and it is maximum when Precision is equal to Recall.

## Logistic regression

Logistic regression is used to estimate discrete values, for example binary values like 0/1 based on a given set of independent variables. It is a basic but important classification algorithm in machine learning that uses one or more independent variables to determine an outcome. Logistic regression tries to find a best-fitting relationship between the dependent variable and a set of independent variables. The best-fitting line in this algorithm looks like S-shape as shown in Figure 19. [12]

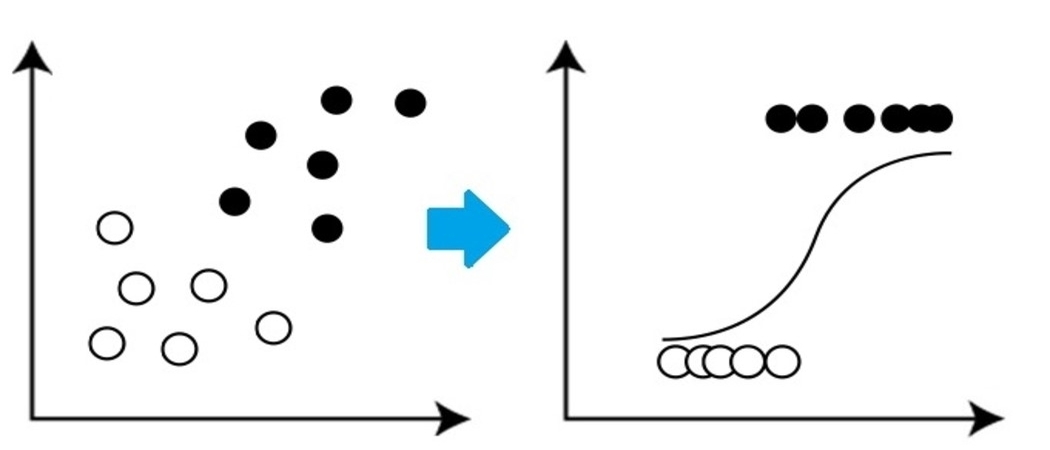


Figure 19: Logistic regression [13]

Process of building model for logistic regression algorithm is shown in the figure 20.

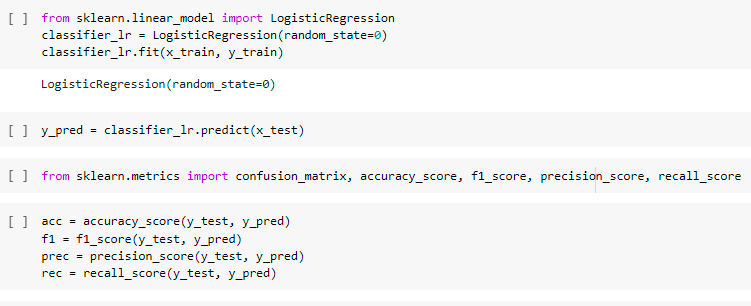


Figure 20: Logistic regression building model

Data frame i.e., result and Confusion matrix are shown in the figure 21.

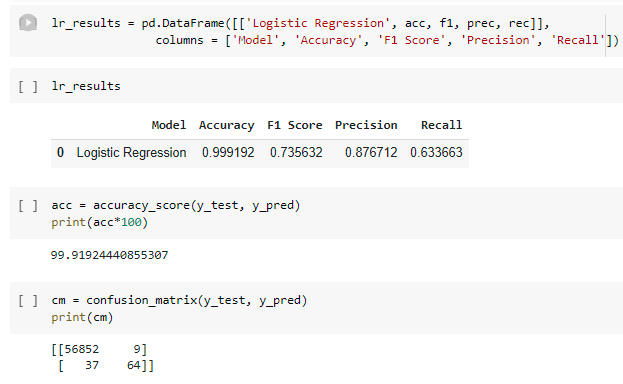


Figure 21: Measured performance for Logistic regression

Cross validation for Logistic regression is shown in the Figure 22.

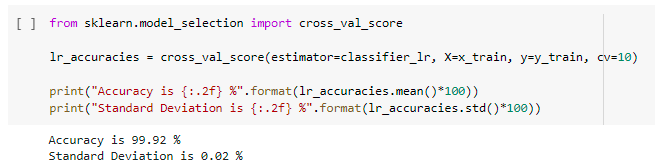


Figure 22: Cross validation for Logistic regression

## Model ensemble and Random Forest

Much of the focus of machine learning is on developing the single most accurate prediction model possible for a given task. Rather than creating a single model, there is generated a set of models and then made predictions by aggregating the outputs of these models. A prediction model that is composed of a set of models is called a model ensemble. The reason for using ensemble methods is the idea that a committee of experts working together on a problem are more likely to solve it successfully than a single expert working alone. However, steps should be taken to guard against group think. In the context of ensemble models, this means that each model should make predictions independently of the other models in the ensemble. [5]

Random Forest is an ensemble of decision trees. The forest chooses the classification having the most votes (over all the trees in the forest).[14] It is a forest of randomly created decision trees. Each node in the decision tree works on a random subset of features to calculate the output. The random forest then combines the output of individual decision trees to generate the final output. [15]

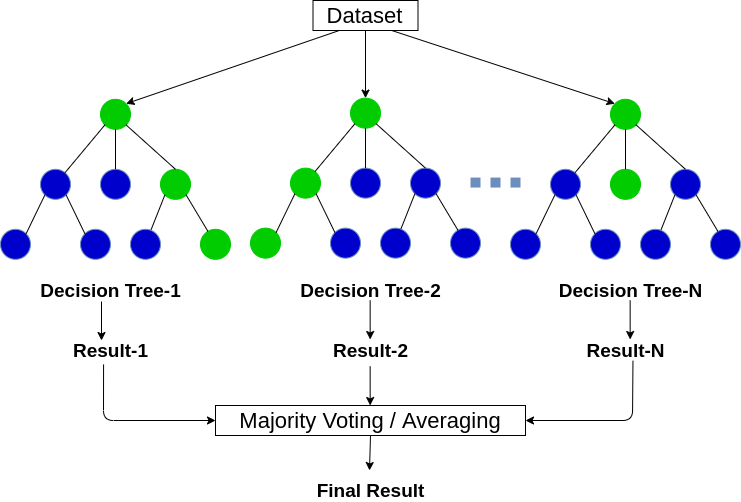


Figure 23: Random Forest [15]

Process of building model for random forest algorithm is same as for logistic regression and results are shown in the figure 24.

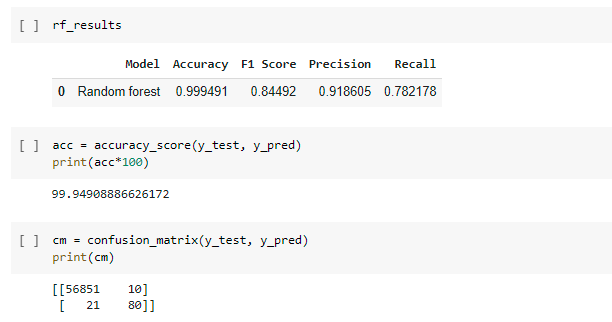


Figure 24: Measured performance for Random Forest

Cross validation has been done for Random Forest algorithm too and it is shown in the Figure 25.

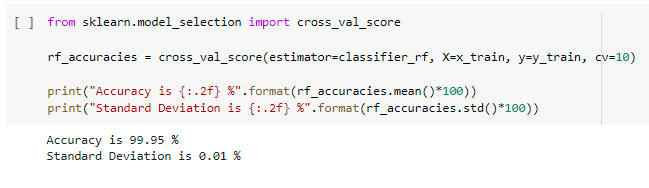


Figure 25: Cross validation for Random Forest

## XGBoost classifier

XGBoost (Extreme Gradient Boosting) is gradient boosting algorithm that has an immensely high predictive power which makes it the best choice for accuracy in events as it possesses both linear model and the tree learning algorithm, making the algorithm almost 10x faster than existing gradient booster techniques.[14] XGBoost is used for supervised learning problems, where the training data (with multiple features) xi is used to predict a target variable yi. [16]

Boosting is an ensemble of learning algorithms which combines the prediction of several base estimators to improve robustness over a single estimator. It combines multiple weak or average predictors to a build strong predictor [14] i.e., in boosting, the individual models are not built on completely random subsets of data and features but sequentially by putting more weight on instances with wrong predictions and high errors. The idea behind this is that instances, which are hard to predict correctly will be focused on during learning, so that the model learns from past mistakes.

XGBoost is also called a regularized boosting technique. This helps to reduce overfit modelling and has a massive support for a range of languages such as Scala, Java, R, Python, Julia, and C++.[14]

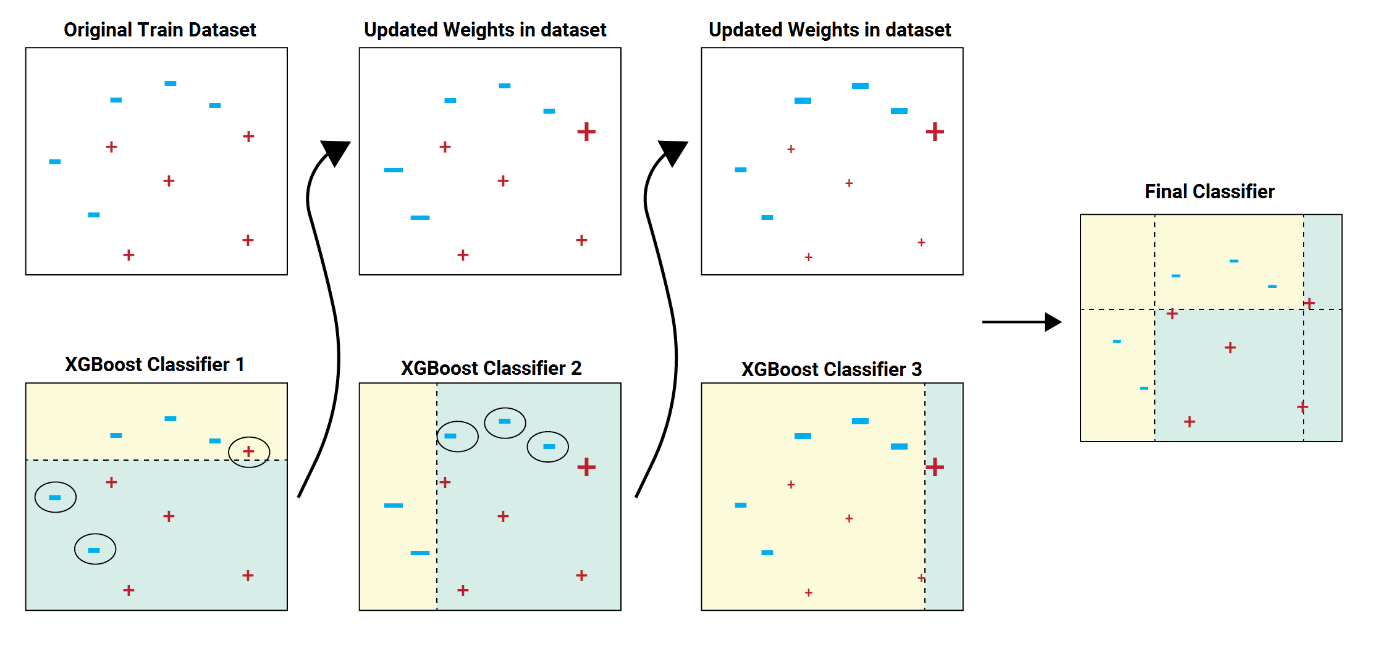


Figure 26: XGBoost classifier [18]

In the above image example, the train dataset is passed to the classifier 1. The yellow background indicates that the classifier predicted hyphen and blue background indicates that it predicted plus. The classifier 1 model incorrectly predicts two hyphens and one plus. These are highlighted with a circle. The weights of these incorrectly predicted data points are increased and sent to the next classifier. That is to classifier 2. The classifier 2 correctly predicts the two hyphen which classifier 1 was not able to. But classifier 2 also makes some other errors. This process continues and we have a combined final classifier which predicts all the data points correctly. [18]

The classifier models can be added until all the items in the training dataset is predicted correctly or a maximum number of classifier models are added. [18]

Process of building model for XGBoost classifier is same as for other two algorithms and results are shown in the figure 27.

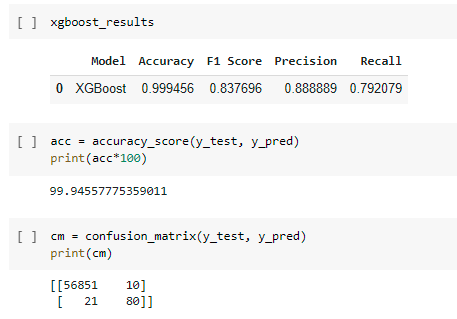


Figure 27: Measured performance for XGBoost

Cross validation for XGBoost algorithm is shown in the Figure 28.

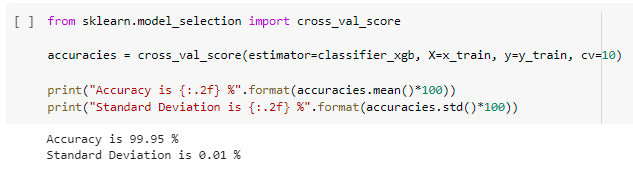


Figure 28: Cross validation for XGBoost

# Evaluation and deployment

In final result, the performance of all algorithms is compared to decide which algorithm proved to be the most accurate. Final results that show performance of all three algorithms are shown in the figure 29.

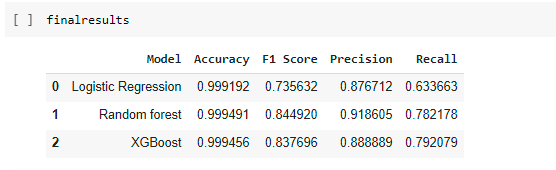


Figure 29: Final results

In the figure is visible that Random Forest algorithm have little better performance than other two algorithms so it is chosen as final model and for predicting a single observation. Also, Random Forest’s confusion matrix also shows better values, with lesser mistakes.

Now, as Random Forest is chosen as final model, it is time to use it in practice.

Let say this will be a single observation for which should be predicted if it is fraudulent or genuine:

single\_obs = [[0.0, -1.359807,  -0.072781,  2.536347, 1.378155, -0.338321,  0.462388, 0.239599, 0.098698, 0.363787, 0.090794, -0.551600,  -0.617801,  -0.991390,  -0.311169,  1.468177, -0.470401,  0.207971, 0.025791, 0.403993, 0.251412, -0.018307,  0.277838, -0.110474,  0.066928, 0.128539, -0.189115,  0.133558, -0.021053,  149.62

]]

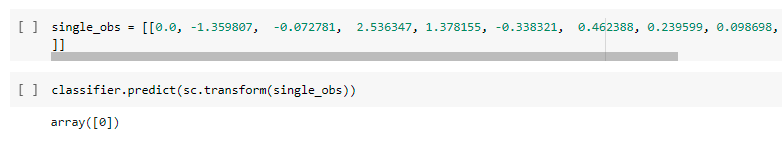


Figure 30: Predicting a single observation

The final model proved to be correct, the value for a given observation should indeed be 0 or that transaction is non-fraudulent.

# Conclusion

Random Forest algorithm may have proved to be the best or most accurate, but other algorithms also showed extremely good performance and quality. It would not be wrong to use other algorithms considering their quality, speed and accuracy.

The XGBoost algorithm also proved to be particularly good, with very little difference between it and the Random Forest algorithm and should definitely have all the recommendations for using it. It is extremely popular among competitors on most popular data analytic sites, but in this case, credit card fraud detection Random Forest, by statistic, is more popular than any other algorithm and mostly used in solving these cases. It avoids the overfitting problem and gives better prediction when compared with a single model, but computation time is high and reduces model interpretability due to increase of complexity.

# 7. References

[1] Jupyter Book, Reproducible Machine Learning for Credit Card Fraud detection - Practical handbook, <https://fraud-detection-handbook.github.io/fraud-detection-handbook/Chapter_2_Background/CreditCardFraud.html>

[2] Statistic Brain Research Institute. Credit card fraud statistics. April 2018. [Online; Last consulted 30-March-2021]. URL: <https://www.statisticbrain.com/credit-card-fraud-statistics/> .

[3] European Central Bank. 6th report on card fraud. August 2020. URL: <https://www.ecb.europa.eu/pub/cardfraud/html/ecb.cardfraudreport202008~521edb602b.en.html#toc2> .

[4] Nilson report. Nilson report issue 1164. November 2019. URL: <https://nilsonreport.com/upload/content_promo/The_Nilson_Report_Issue_1164.pdf> .

[5] John D. Kelleher & Brian Mac Namee & Aoife D’Arcy, Fundamentals of Machine Learning for Predictive Data Analytics , <https://dokumen.pub/fundamentals-of-machine-learning-for-predictive-data-analytics-algorithms-worked-examples-and-case-studies-1-edition-july-24-2015-978-0262029445.html> .

[6] Ian T. Jolliffe and Jorge Cadima, Principal component analysis: a review and recent developments, <https://royalsocietypublishing.org/doi/10.1098/rsta.2015.0202>

[7] Fabrizio Carcillo. Beyond Supervised Learning in Credit Card Fraud Detection: A Dive into Semi-supervised and Distributed Learning. Université libre de Bruxelles, 2018.

[8] Andrea Dal Pozzolo. Adaptive machine learning for credit card fraud detection. Université libre de Bruxelles, 2015.

[9] Machine learning for credit card fraud detection <https://fraud-detection-handbook.github.io/fraud-detection-handbook/Chapter_2_Background/MachineLearningForFraudDetection.html>

[10] Raghav Vashisht, When to perform a Feature Scaling? <https://www.atoti.io/when-to-perform-a-feature-scaling/>

[11] Aniruddha Bhandari, Everything you Should Know about Confusion Matrix for Machine Learning, <https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/>

[12] Gaurav Sharma, 5 Classification Algorithms you should know – introductory guide!, <https://www.analyticsvidhya.com/blog/2021/05/5-classification-algorithms-you-should-know-introductory-guide/>

[13] <https://www.equiskill.com>

[14] Sunil Ray, Commonly used Machine Learning Algorithms (with Python and R Codes), <https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/>

[15] Abhishek Sharma, Decision Tree vs. Random Forest – Which Algorithm Should you Use?, <https://www.analyticsvidhya.com/blog/2020/05/decision-tree-vs-random-forest-algorithm/>

[16] Official documentation, XGBoost Tutorials - Introduction to Boosted Trees <https://xgboost.readthedocs.io/en/stable/tutorials/model.html>

[17] Machine Learning Basics - Gradient Boosting & XGBoost <https://shirinsplayground.netlify.app/2018/11/ml_basics_gbm/>

[18] Ishan Shah, Introduction to XGBoost in Python, <https://blog.quantinsti.com/xgboost-python/>

[19] <https://www.kaggle.com/mlg-ulb/creditcardfraud>