Project 1: Credit Card Fraud Detection Using Machine Learning Algorithms

Pritam Shrestha
Spring Term 2021
Pritamportfolio.com

Abstract

Credit card fraud detection is a set of activities that are taken to prevent money via credit card transactions. Credit card fraud loses consumers and financial companies billions of dollars annually. In most companies, fraud is identified after it occurs, In the event that they are unable to prevent it in a timely fashion. Thus, fraud detection systems have become essential for banks and financial institutions to avoid losses.

This project aims to build a machine learning model that has the ability to predict with high accuracy, precision, and recall. Initially, it will begin with exploratory data analysis and ends finding the best and accurate model among models. The data being used in this project is downloaded from Kaggle.com. It consists of a combination of fraudulent and non-fraudulent transactions. There are nearly 285000 rows and 31 columns as features. Some of these features are non-descriptive due to security reasons, but a few features are descriptive, and I will use them for data visualization and extract the patterns and insights regarding fraudulent and non-fraudulent transactions. Based on the datasets and structure of the datasets, I must use classification algorithms because target variables or transections are falling into two categories or classes, either normal or fraud. The motive of this project is to identify which new transaction will fall into it. In this online era, fraudulent cases are rising and expanding to many companies day by day. So, this implemented machine learning model could save billions of dollars detecting fraudulent activities.

Used resources:

Resources are something that is used to complete the project. Here I have used the following resources to complete this project.

1) Data source:
2) Other technical resources: Anaconda, Jupyter Notebook, python libraries, etc.

Data Source:

Data is distinct pieces of information, usually formatted in a special way. It can be used to analyze and visualize implementing different machine learning algorithms. Data is the most valuable thing in data science because everything relies on it. Bad data could lead to bad output and project failure. So data finding is a major milestone of the project. Finally, I found data and downloaded data as a CSV file format from the following source.

source**:** https://www.kaggle.com/mlg-ulb/creditcardfraud

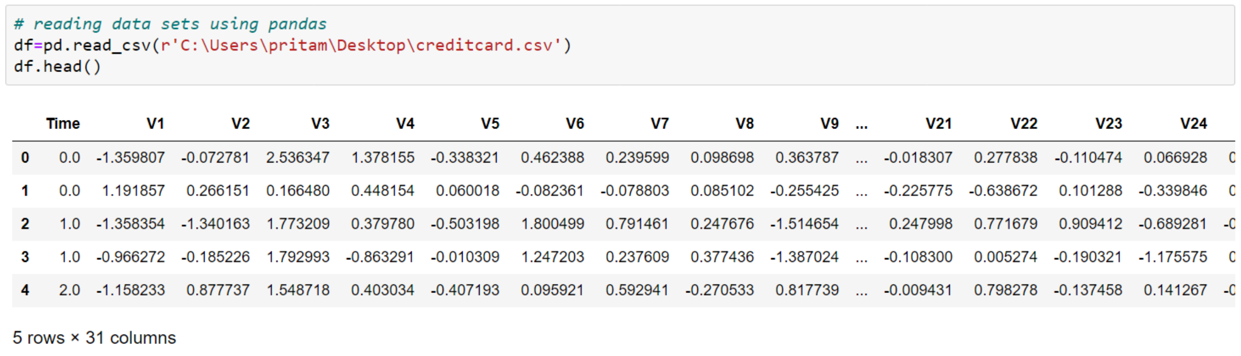
Data understanding:

Understanding the problem statement is the best way of finding a solution. To understand the problem, we need to find the problem first. Problem finding is not easy because it could be related to anything. So first of all, we need to categories the type of problem, whether it is solvable or not. If it is solvable, then they need to pay attention to it. Likewise, I figured out the problem statement and started to find possible factors being responsible for prediction.

The structure of the data is provided as an image. Basically, I have downloaded the data from Kaggle and saved it in a CSV file. Later I read that data using the pandas library of python for further analysis.

Structure of the data:

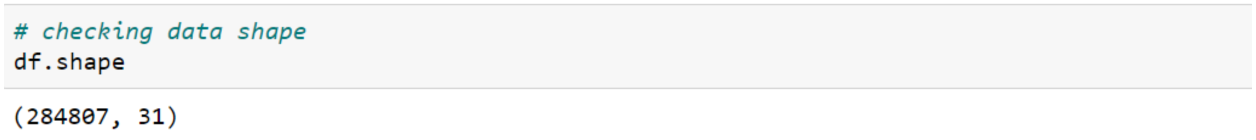
Before implementing the data for modeling, we have to go and analyze data for better understanding and perform exploratory data analysis. Raw data is not always ready for model implementation, so data preparation is another essential part of the data science project. Here is the structure of the downloaded data.



```
# reading data sets using pandas
df=pd.read_csv(r'C:\Users\pritam\Desktop\creditcard.csv')
df.head()
```

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V24
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307	0.277838	-0.110474	0.066928
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775	-0.638672	0.101288	-0.339846
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.247998	0.771679	0.909412	-0.689281
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.108300	0.005274	-0.190321	-1.175575
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	...	-0.009431	0.798278	-0.137458	0.141267

5 rows × 31 columns



```
# checking data shape
df.shape
```

```
(284807, 31)
```

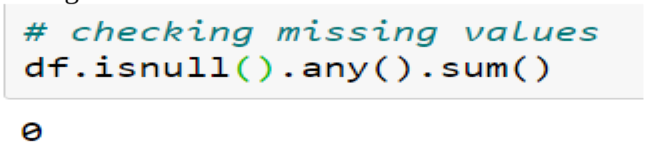
Fig: Structure of the datasets

The datasets consist of 284807 rows and 31 columns. These datasets are huge and very hard to process. I will only take a sample of these datasets for model implementation. Otherwise, my local machine will take a couple of days to process. Before implementing the model, we should go through the Exploratory data analysis steps.

Exploratory data analysis:

Exploratory data analysis is an approach to analyzing the dataset by summarizing its main characteristics by implementing visual methods. The primary purpose of EDA is to see what the data can tell us beyond the formal modeling or hypothesis testing task. Data is not always ready for further analysis, so I did some exploratory data analysis using python.

1) Replaced header: I replaced the header of the downloaded data to make it more readable and understandable.
2) Fixed casing and inconsistent data: When I checked the structure of the data, I did not find any missing data.



```
# checking missing values
df.isnull().any().sum()
```

```
0
```

Fig: checking missing data

3) Checked outliers and bad data: Sometimes, we get outlier. Outlier means extreme values that fall a long way outside of the other observations. Finding outliers and proper handling of the outlier is very important; otherwise, it might lead to the wrong output, or the result might bias to the higher value. But in my dataset, I didn't go through it because most columns are kept unknown due to security reasons.
4) Data cleansing or cleaning: It involves finding and correcting corrupt or inaccurate records from the dataset. I was fortunate, so I have not found any corrupt data.
5) Data transformation: It is a process of converting data from one format to another format that makes data more readable and understandable. Here, I have not used transformation except StandardScaler.
6) Visualization: It is a pictorial representation of data. It helps us to analyze the data, so I plotted a couple of plots and found important information about fraudulent and non-fraudulent transactions.

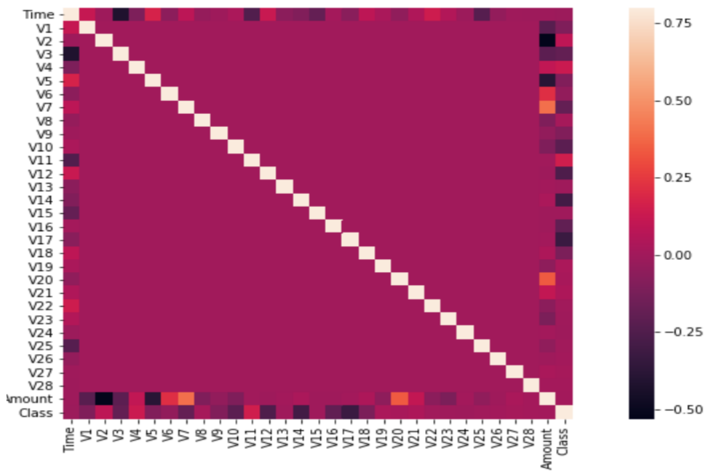


Fig: Correlation metrics

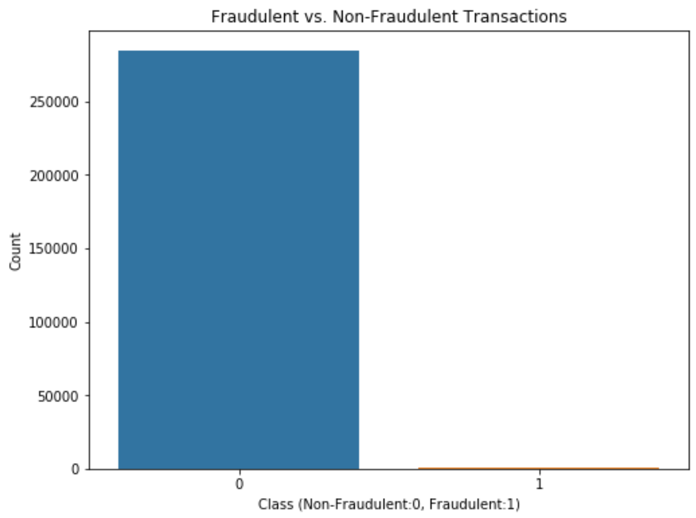


Fig: Fraudulent and non-fraudulent transactions

Based on the above plots, I have found the vast majority of data are non-fraudulent, and only a few transactions are fraudulent.

Feature Selection and Feature scaling:

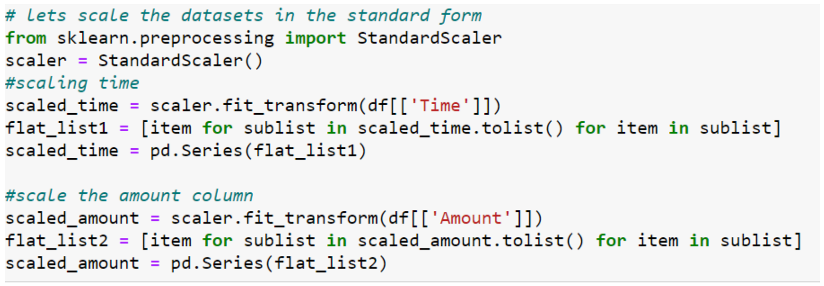
Feature Selection:

Feature selection is the process of selecting a subset of relevant features or variables for modeling or use in the model construction. It is also called variable selection. In my case, I have 31 variables or features; most of them are correlated to the target variable, so I did include all of them as my input for modeling.

1) Input features: Time, Amount, V1 to V28
2) Target variable: Class

Feature scaling:

Feature scaling is the method to standardize the independent features present in the data. This is also called the normalization method. If it is not done, then machine learning algorithms tend to weight greater values higher and consider smaller values as the lower values, regardless of the unit of the values. So, to avoid this problem, we can use either normalization or standardization, but I have performed a standardization method using StandardScaler to make my input features normal and standard format.

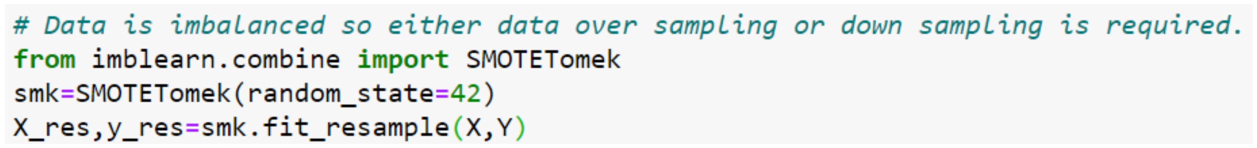


```python
# lets scale the datasets in the standard form
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
#scaling time
scaled_time = scaler.fit_transform(df[['Time']])
flat_list1 = [item for sublist in scaled_time.tolist() for item in sublist]
scaled_time = pd.Series(flat_list1)

#scale the amount column
scaled_amount = scaler.fit_transform(df[['Amount']])
flat_list2 = [item for sublist in scaled_amount.tolist() for item in sublist]
scaled_amount = pd.Series(flat_list2)
```

Fig: data standardization using StandardScaler

Handling imbalanced datasets:

Dataset is the main asset of any data science project. An output directly depends on the nature of the datasets. Highly imbalanced data could lead to the wrong output because the model might be biased towards the majority of of datasets. So, handling the imbalanced dataset is the best practice of data preparation in data science. It can be controlled using different methods such as oversampling and downsampling. Here, I have implemented oversampling technique using imblearn.

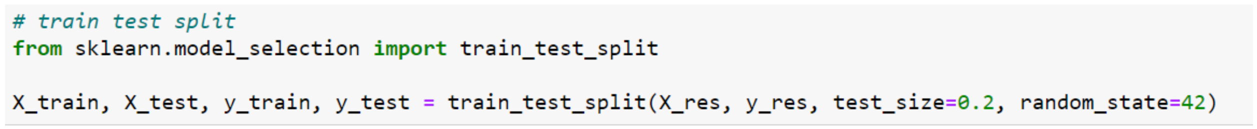


```python
# Data is imbalanced so either data over sampling or down sampling is required.
from imblearn.combine import SMOTETomek
smk=SMOTETomek(random_state=42)
X_res,y_res=smk.fit_resample(X,Y)
```

Fig: imbalance dataset handling technique

Data Modeling:

Data modeling means training machine learning algorithms to predict the target from the features. For the implementation of the model, we have to split out data into two data sets, such as the training dataset and the testing dataset, so I have split my input and target dataset into two datasets.



```python
# train test split
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X_res, y_res, test_size=0.2, random_state=42)
```

Fig: data splitting using sklearn

1) Training dataset: It contains 80% of the data for training purposes.

2) Testing dataset. It contains 20% of the data for testing purposes.

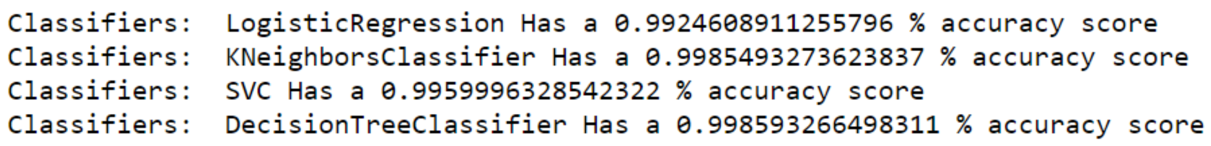
Model selection:

Now input and target data are ready to train the model and test the model, so I have selected the following models for data modeling.

1) LogisticRegression
2) LinearDiscriminantAnalysis
3) KNeighborsClassifier
4) DecisionTreeClassifier
5) SVC (Support Vector Machine)
6) RandomForestClassifier

Result:

Finally, I have implemented a few models using hyperparameter tuning and k-fold cross-validation. As a result, I have looked into some essential matrices such as accuracy, precision, and recall. Accuracy is the way of measuring how much the model is predicting the accurate value of the target variable. The obtained accuracy was as follows.

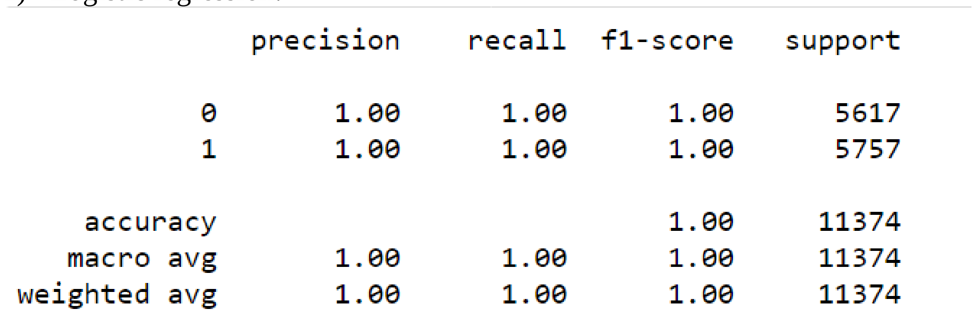


```
Classifiers:  LogisticRegression Has a 0.9924608911255796 % accuracy score
Classifiers:  KNeighborsClassifier Has a 0.9985493273623837 % accuracy score
Classifiers:  SVC Has a 0.9959996328542322 % accuracy score
Classifiers:  DecisionTreeClassifier Has a 0.998593266498311 % accuracy score
```

Fig: Accuracy comparison

Actually, I have got more than 98% accuracy implementing these models. Among them, logistic regression and random forest were the best and had more than 99% accuracy. The only accuracy is not enough to decide whether the model is best or wore. There are many other factors or matrices also responsible for measuring the effectiveness of the model. Usually, true positive, true negative, false positive, and false negative also play a vital role in prediction. So, I have gone through precision and recall using these two best models.

1) Logistic regression:



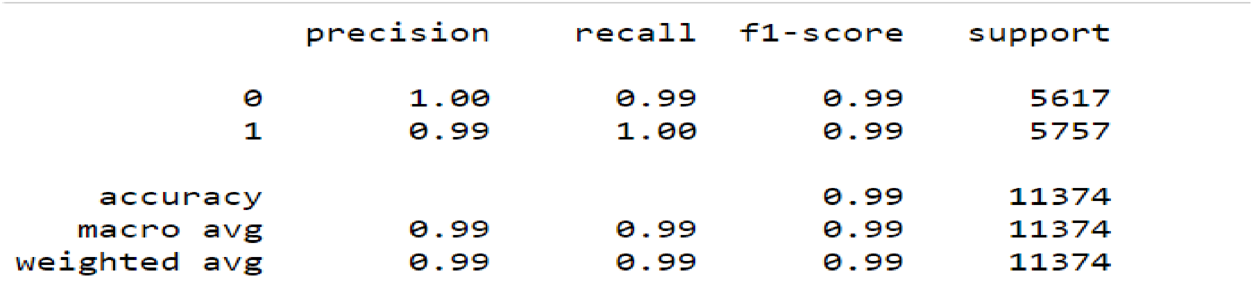
```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      5617
           1       1.00      1.00      1.00      5757

    accuracy                           1.00     11374
   macro avg       1.00      1.00      1.00     11374
weighted avg       1.00      1.00      1.00     11374
```

Fig: precision, recall, and f1-score.

For this project, logistic regression is the best model because it's precision and recall value is 100%.

2) Random forest:

Another best model is random forest because its accuracy is 99%. Its precision and recall are also more than 99%, which indicates that the model does not detect only less than 1% fraudulent cases. Still, we can improve this using more effective training and testing samples.



```
              precision    recall  f1-score   support

           0       1.00      0.99      0.99      5617
           1       0.99      1.00      0.99      5757

    accuracy                           0.99     11374
   macro avg       0.99      0.99      0.99     11374
weighted avg       0.99      0.99      0.99     11374
```

Fig: model metrics

Problem/Issues:

In this project, I have solved the classification problem using a few machine learning algorithms implementing various methodologies. While completing this project, I have encountered many problems. I had a plan to implement pipeline and greadSearchCV, but I could not apply this technique due to the computational difficulty of my local machine. However, I have completed this project using other methods.

Discussion/Conclusion

Finally, we can see that several algorithms showed similar results, but a few outperformed than rest. Namely, the Logistics regression algorithm predicted more accurately than the alternative classifiers for this dataset. In this model, The 0 class (non-fraudulent) transactions are predicted with 100% precision whereas, the 1class (fraudulent) transactions are also predicted with 100% precision. This means that the logistic regression model is predicting more accurately.

On the other hand, the random forest algorithm is predicted non-fraudulent transactions with 100% and fraudulent transactions with 99% precision. Only 1% of fraudulent transactions that are fraudulent remain undetected by the system. This can be further improved by providing more training data.

Hence, it is concluded that we were able to use a logistic regression model to produce an accurate model for predicting fraudulent credit card transactions.

Acknowledgment:

I can't express enough thanks to my professor Fadi Alsaleem for his continuous support and encouragement. The completion of this project could not have been accomplished without the help of my classmates and my beautiful daughter, Prisumsa.

Thanks to my wife Suchan as well. The countless times you kept the children during our hectic schedule will not be forgotten.

Appendix:

Used libraries

1) Pandas: open-source data analysis and manipulation tool.
2) Numpy: Main library for computation and array construction.
3) Matplotlib: data visualization and presentation.
4) Seaborn: data visualization and presentation.
5) Sklearn: Machine learning implementation.

References:

1) Machine Learning Group. (2018, March 23). Creditcard Fraud Detection https://www.kaggle.com/mlg-ulb/creditcardfraud/home?select=creditcard.csv
    a) This is my data source, and I will be using this data in my credit card fraud detection project.

2) Fabrizio Carcillo, Yann-Aël Le Borgne, Olivier Caelen, Frederic Oblé, Gianluca Bontempi. (2019) Combining Unsupervised and Supervised Learning in Credit Card Fraud Detection Information Sciences. https://www.researchgate.net/publication/333143698_Combining_Unsupervised_and_Supervised_Learning_in_Credit_Card_Fraud_Detection
    a) Supervised learning techniques are widely employed in credit card fraud detection, as they make use of the assumption that fraudulent patterns can be learned from an analysis of past transactions. It gives some insights into fraud detection techniques.
3) Data Flair Team. (2019, October 10). 11 Top Machine Learning Algorithms used by Data Scientists. https://data-flair.training/blogs/machine-learning-algorithms/
    a) It provides brief descriptions of 11 machine learning algorithms which helps us to choose the best machine learning algorithms for this project.
4) Naik K. (2019, June 25). Credit Card Fraud Detection using Machine Leaning from Kaggle. https://www.youtube.com/watch?v=frM_7UMD_-A&ab_channel=KrishNaik
    a) This video is beneficial in understanding the fundamental concepts of normal transactions and fraudulent transactions. It also teaches us how to implement the machine learning algorithm to detect fraud transactions.
5) Eduonix(2018, Feb 22). Build a complete project in Machine learning| Credit card fraud Detection 2019 https://www.youtube.com/watch?v=gCWBFyFTxVU&t=2053s&ab_channel=EduonixLearningSolutions
    a) This video teaches us how to implement machine learning algorithms for fraud detection. For example, isolation forest and local outlier factor.
6) Dal Pozzolo, Andrea. (2018). Adaptive Machine learning for Credit Card Fraud Detection. http://di.ulb.ac.be/map/adalpozz/pdf/Dalpozzolo2015PhD.pdf
    a) This Ph.D. thesis describes a wide range of credit card fraud detection. It also included the impact of fraud, challenges in data-driven fraud detection systems, understanding sampling methods, unbalanced data streams, and formalization of real-world fraud detection systems.
7) Nadim, A. H., Sayem, I. M., Mutsuddy, A., & Chowdhury, M. S. (2020, February 13). Analysis of Machine Learning Techniques for Credi Card Fraud Detection. https://ieeexplore.ieee.org/document/8995753
    a) This article discussed the use of machine learning algorithms to address credit card fraud, investigating the use of various regression algorithms in the process.
8) Eryk Lewinson. (2018, July 2). Outlier Detection with Isolation Forest. https://towardsdatascience.com/outlier-detection-with-isolation-forest-3d190448d45e
    a) This article is beneficial and describes isolation forest machine learning algorithms for outlier detection.
9) Prakash Verma. (2020, October 27). Isolation Forest Algorithms for Anomaly Detection. https://heartbeat.fritz.ai/isolation-forest-algorithm-for-anomaly-detection-2a4abd347a5
    a) This article describes the isolation forest algorithms, which helps us to implement that concept in my project.
10) Scikit Learn Team. (2020). Outlier detection with local outlier factor (LOF). https://scikitlearn.org/stable/auto_examples/neighbors/plot_lof_outlier_detection.html#:~:text=The%20Local%20Outlier%20Factor%20(LOF,lower%20density%20than%20their%20neighbors.
    a) LOF is another method that I am going to implement in my project, so this article provides its brief description and sample code.