Credit Card Fraud Detection

CAPSTONE PROJECT

Vidhi Rajesh Chitalia | Rutgers Business School | 3rd December 2020

Inspiration

With the growth of the e-commerce industry and community and the dire to make the global economy cashless, there have been people choosing digital payments over cash. Fraud is a huge national and worldwide problem; we will see credit card fraud rise at rates faster than ever. Credit card fraud is a form of identity theft involving criminal deception for personal financial gain. Here are some quick stats about credit card frauds:

- There were 650,572 cases of identity **theft** in the **U.S.** in 2019. Of those, 41 percent, or just over 270,000, were **credit card fraud**
- In 2018, $24.26 Billion was lost due to payment card fraud worldwide
- Credit card fraud was ranked #1 type of Identity theft fraud
- The United States leads as the most credit fraud-prone country with 38.6% of reported card fraud losses in 2018

To identify the factors leading to scams and minimize their extent, I decided to build a project that recognizes whether the transaction is a fraud or genuine.

Datasets

To perform analysis and build a model that correctly identifies the transaction, I have used two datasets: **German Credit** data by UCI Machine Learning and **Credit Card Fraud Detection** by Machine Learning Group-ULB from Kaggle.

- The German credit data is a fantastic dataset with variables that help decide whether a person is a good credit risk or a bad credit risk. There are originally **1000** entries and **nine** attributes in the dataset: the first five instances and features shown below.

```
In [4]: german_credit.head(20)
```

Out[4]:

	Age	Sex	Job	Housing	Saving accounts	Checking account	Credit amount	Duration	Purpose
0	67	male	2	own	NaN	little	1169	6	radio/TV
1	22	female	2	own	little	moderate	5951	48	radio/TV
2	49	male	1	own	little	NaN	2096	12	education
3	45	male	2	free	little	little	7882	42	furniture/equipment
4	53	male	2	free	little	little	4870	24	car
5	35	male	1	free	NaN	NaN	9055	36	education

- The second dataset contains transactions made by credit cards in September 2013 in a span of two days. Therefore, I have about **284807** transactions and **31** columns containing only numerical input values resulting from PCA transformation.

```
In [3]: data.head(20)
```

Out[3]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307	
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775	-(
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.247998	
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.108300	
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	...	-0.009431	
5	2.0	-0.425966	0.960523	1.141109	-0.168252	0.420987	-0.029728	0.476201	0.260314	-0.568671	...	-0.208254	-(

Data Dictionary

German credit dataset has nine selected attributes that are important for analysis.

- Age (numeric)
- Sex (text: male, female)
- Job (numeric: 0- unskilled and non-resident, 1- unskilled and resident, 2- skilled, 3- highly skilled)
- Housing (text: own, rent, free)
- Savings account (text: little, moderate, quite rich, rich)
- Checking's account (numeric)
- Credit amount (numeric)
- Duration (numeric in months)
- Purpose (text: car, furniture/equipment, radio/TV, domestic appliances, repairs, education, business, vacation/others)

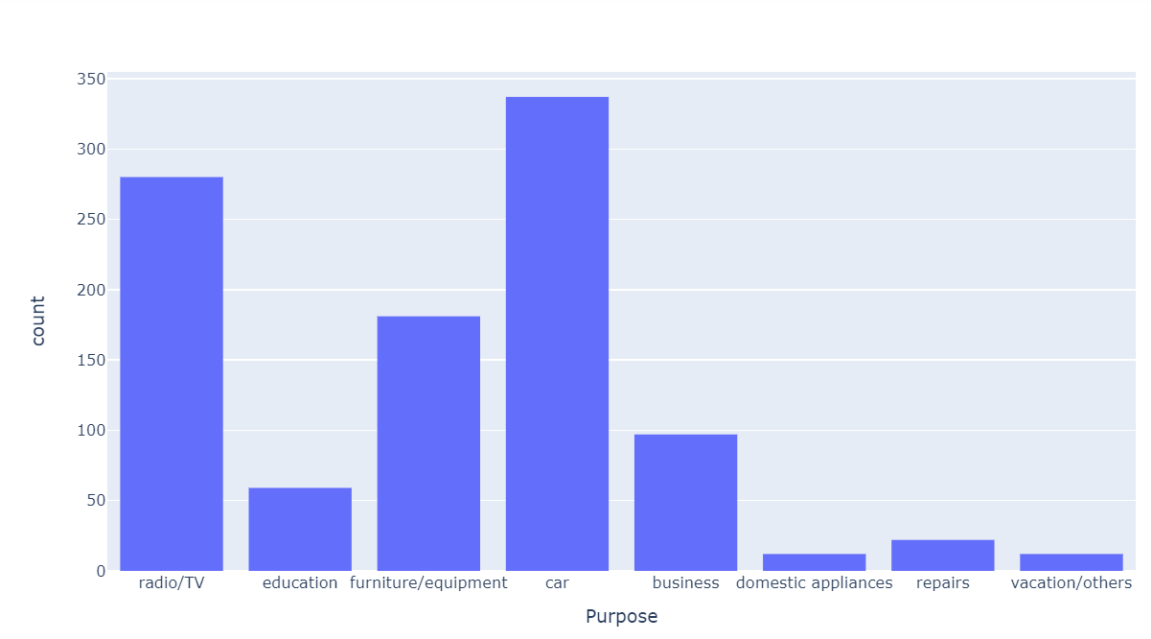
Unfortunately, due to confidentiality issues, credit card fraud detection by ULB cannot provide the original features and more background information of the data. Features V1, V2, … V28 is the principal components obtained with PCA; the only non-transformed parts with PCA are 'Time' and 'Amount.'

- Time (numeric: seconds elapsed between each transaction and the first transaction in the dataset)
- Amount (numeric: transaction amount)
- Class (**Target variable**, numeric values: 1- fraud, 0- non-fraud)

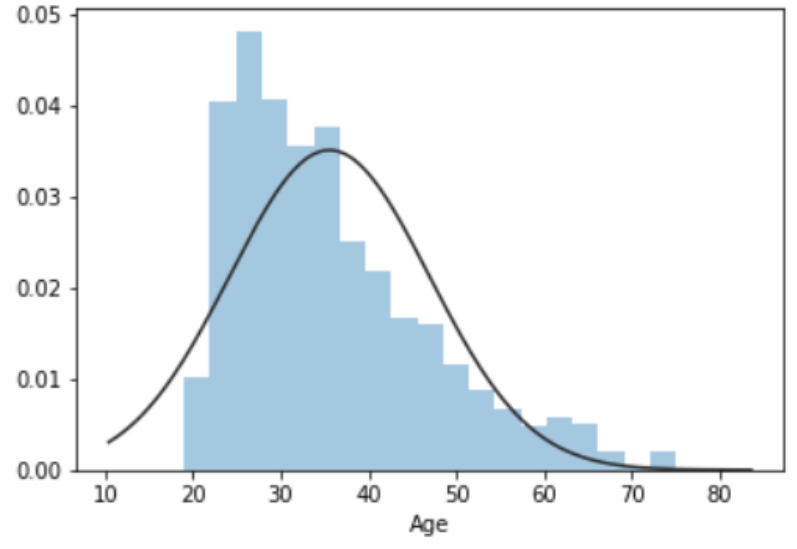
Analysis and Model Building

The first and foremost step in any data science project is to cleanse the data, handle any missing values, and find meaningful insights. The next step is to apply machine learning algorithms by building models to predict accuracy, precision score, and recall. Below are some helpful steps that I have taken to draw insights from the two data.

- For the first dataset (**German credit**), I performed data cleaning and handled missing values. Performed exploratory analysis to find that Germans take the most credit for buying a car, followed by purchasing radio/TV and furniture.

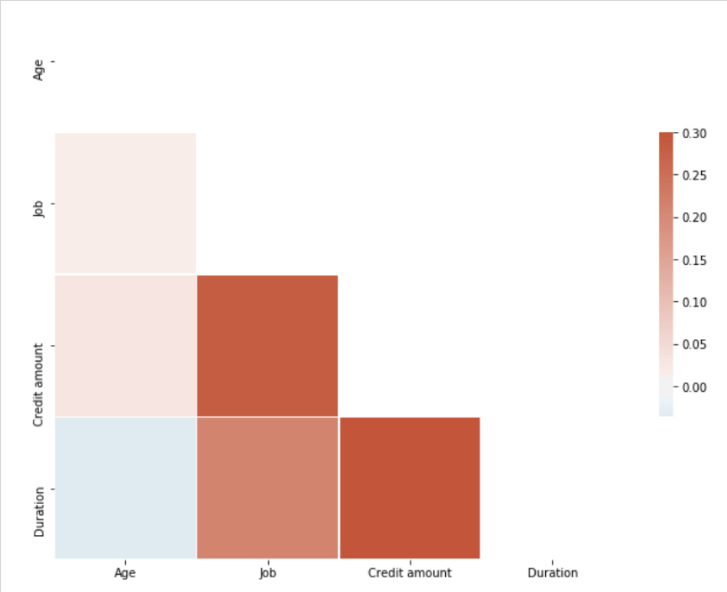


Drawing a normal distribution for the age resulted in skewness, and thus the mean age for people taking credit is 35 years.

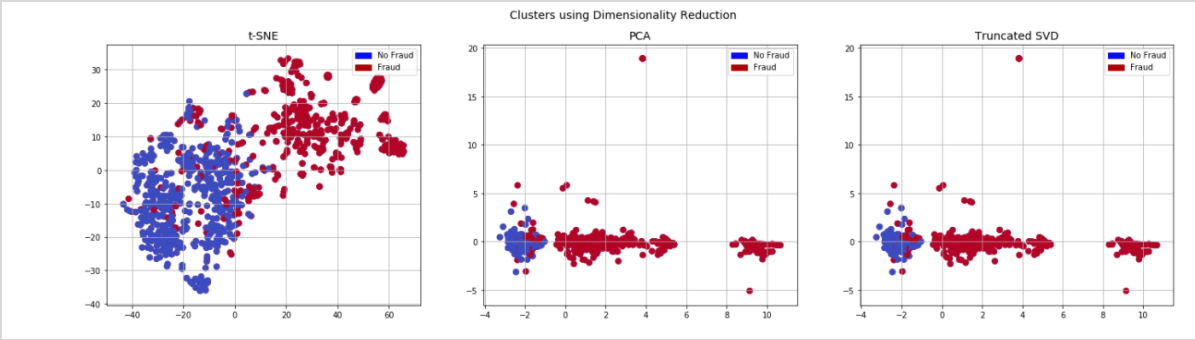


One exciting insight that I came across was that people did not prefer to take domestic appliances on credit. Lastly, I drew a heatmap to find any existing

correlations between the attributes where credit amount, job, and duration have the most substantial relationship.



- Since the second dataset had PCA transformed components, there were not any missing or null values. An impressive library called T-SNE, Truncated SVD in python, has been used to identify groups within the data.



The dataset was highly imbalanced, with **99.83% non-fraud values** and only **0.17% fraud values**.

```
In [5]: print("Non Frauds", round(data["Class"].value_counts()[0]/len(data)*100, 2),"% of the dataset")
        print("Fraud", round(data["Class"].value_counts()[1]/len(data)*100, 2),"% of the dataset")

Non Frauds 99.83 % of the dataset
Fraud 0.17 % of the dataset
```

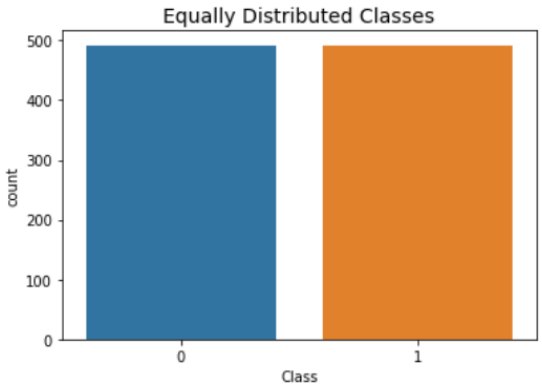
It was mandated to have the dataset balanced, scaled, and transformed for a better performing model, so all these were the next steps to follow.

Visualizing the balanced dataset for the target variable

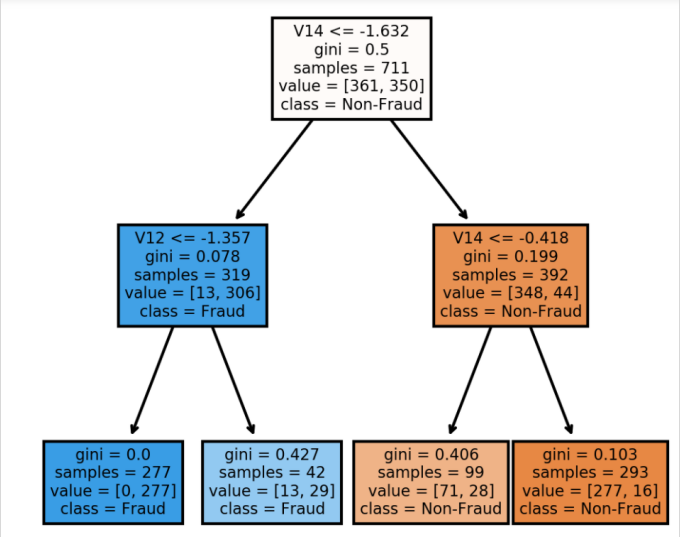
```
In [14]: print('Distribution of the Classes in the subsample dataset')
         print(new_df['Class'].value_counts()/len(new_df))

         sns.countplot('Class', data=new_df)
         plt.title('Equally Distributed Classes', fontsize=14)
         plt.show()
```

```
Distribution of the Classes in the subsample dataset
1    0.5
0    0.5
Name: Class, dtype: float64
```



I also performed some data visualizations using Plotly, Matplotlib, and Seaborn libraries in the jupyter notebook. I performed clusterings such as K-Means and Spectral Clustering to find the non-fraud and fraud groups in the data. The last step is to build a model. For consistency, I split the entire data into **75% training set** and **25% test set**. I trained the model with five models: Logistic Regression, Random Forest, Support Vector, Decision Tree, and Logit. It is essential to understand the split for the decision tree. Therefore, I have visualized the decision tree.



To test the testing set model and evaluate its performance, I used metrics like calculating Accuracy Score and plotting ROC-AUC. I have built a confusion matrix for a more in-depth understanding and calculated the F1_scores, precision, and recall.

Models	Confusion Matrix
Decision Tree	[[129 1] [13 94]]
Support Vector	[[128 2] [13 94]]
Logistic Regression	[[128 2] [10 97]]
Random Forest	[[129 1] [18 89]]

$$\text{Precision} = \frac{TP}{TP+FP}$$

Models	Precision Values
Decision Tree	0.91
Support Vector	0.91
Logistic Regression	0.93
Random Forest	0.88

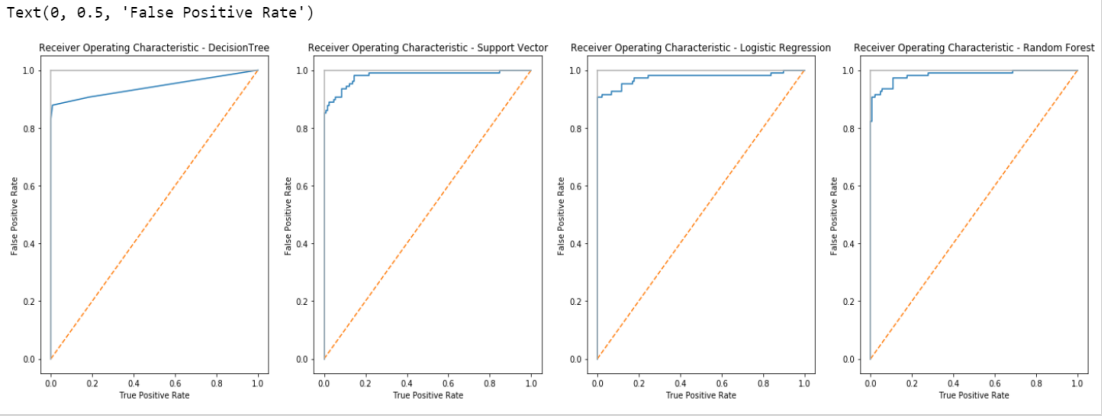
$$\text{Recall} = \frac{TP}{TP+FN}$$

Models	Recall Values
Decision Tree	0.99
Support Vector	0.98
Logistic Regression	0.98
Random Forest	0.99

$$\text{F1\_score} = 2\frac{Precision*Recall}{Precision+Recall}$$

Models	F1_score
Decision Tree	0.94
Support Vector	0.93
Logistic Regression	0.95
Random Forest	0.92

Models	Accuracy Score
Decision Tree	92%
Support Vector	93%
Logistic Regression	94%
Random Forest	92%



Models	ROC-AUC Values
Decision Tree	0.94
Support Vector	0.98
Logistic Regression	0.97
Random Forest	0.99

Conclusion

There are many criteria's like F1_score, precision, recall, accuracy score, ROC-AUC values for choosing and evaluating the best model. I have compared the confusion matrix, F1_score, precision, recall, accuracy score, and AUC values. My problem is to detect whether a transaction is a scam or a genuine one; I cannot afford a fraud classified as a non-fraud. Choosing a model with the highest recall (specificity) is of utmost importance. The values calculated in the above table show **Decision Tree** and **Random Forest** models, giving a **recall** score of **0.99**.

Since there is a tradeoff between the two models, the next metric to consider is the ROC-AUC curve and the accuracy. The difference between the Accuracy and ROC-AUC curve values is that the former is calculated based on predicted classes while the latter on predicted scores. The ROC-AUC value for Random Forest is the **maximum (0.99),** i.e., most of my fraud transactions classify frauds and vice versa. Thus, in conclusion, **Random Forest Classification** serves as the best model for my implementation.