Credit Card Fraud Detection

Using Supervised ML Techniques

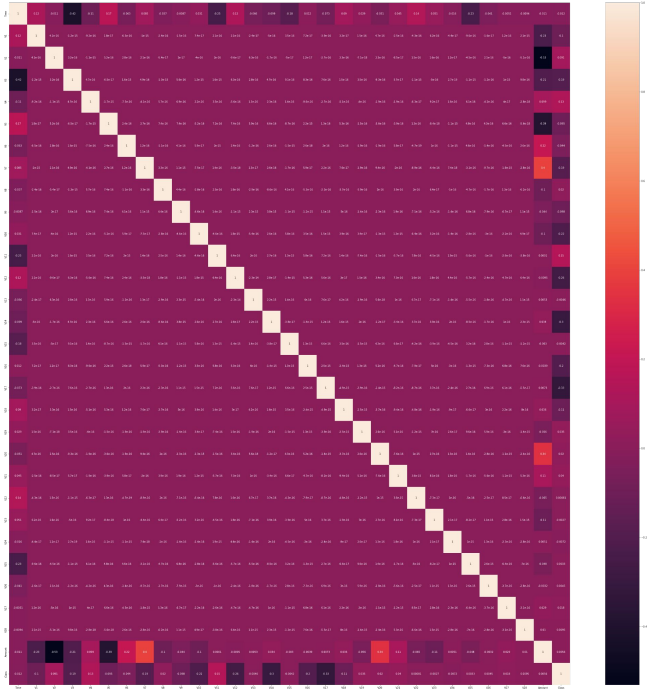
Dataset

- September 2013
- European Cardholders
- 284,807 Transactions
- PCA already performed
  - Protect anonymity of cardholders with Variable names

- 400,000 cases in 2020 alone

Cleaning and Visualization

- PCA already performed
- No missing values
  - Also checked outside default pandas NA values
- 31 variables
  - 30 predictor, 1 response
- Look at correlation Matrix
- Only 492 fraudulent transactions
  - Heavily Skewed Data



Splitting Data

- Undersampling
  - Change proportion of positive cases to negative cases
  - Identify all the legitimate and fraudulent cases
  - Select all fraudulent and three times that many legitimate cases
  - Check for duplicates
- Split use train_test_split
- Check train data set

```
X = df.iloc[:, df.columns != 'Class']
y = df.iloc[:, df.columns == 'Class']
len(y[y.Class ==1])
```

```
492
```

```
number_fraud = len(df[df['Class']==1]) * 3
fraud_index = np.array(df[df['Class']==1].index)
legit_index = np.array(df[df['Class']==0].index)

random_legit_index = np.random.choice(legit_index, number_fraud, replace = False )
under_sample_index = np.concatenate([fraud_index, random_legit_index])
```

```
#Make sure they don't overlap
np.intersect1d(fraud_index , legit_index)
```

```
array([], dtype=int64)
```

```
under_sample = df.iloc[under_sample_index,:]
x_undersample = under_sample.iloc[:, under_sample.columns != 'Class'];
y_undersample = under_sample.iloc[:, under_sample.columns == 'Class'];
```

```
from sklearn.model_selection import train_test_split
X_train_under, X_test_under, y_train_under, y_test_under = train_test_split (x_undersample,y_undersample, test_size = 0.3, random_state = 42)
```

```
y_train_under['Class'].value_counts()
```

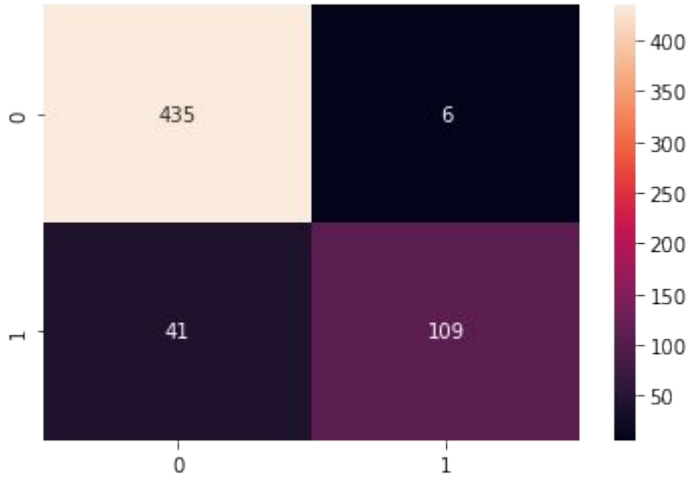
```
0    1035
1     342
Name: Class, dtype: int64
```

Modelling Data

- Large number of Variables
  - No KNN or standard Decision Tree
- Cannot assume linearity
  - No Multi-Linear or Logistic Regression
- Two Main Models
  - Naives-Bayes
  - Random Forest
- Look at Recall Score and Execution Time

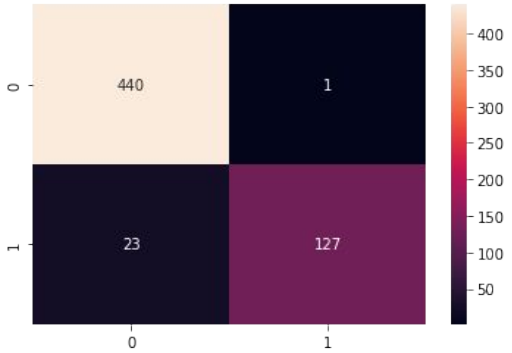
Gaussian Naive-Bayes

- Duration: 0:00:00.005505
- Recall Score: 0.7266666666666667

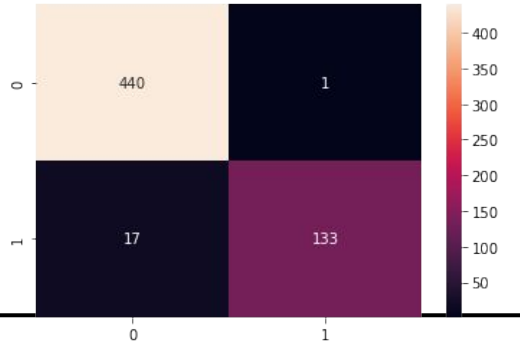


Random Forest

- Max_depth: 2
  - Duration: 0:00:00.233200
  - Recall Score: 0.8466666666666667

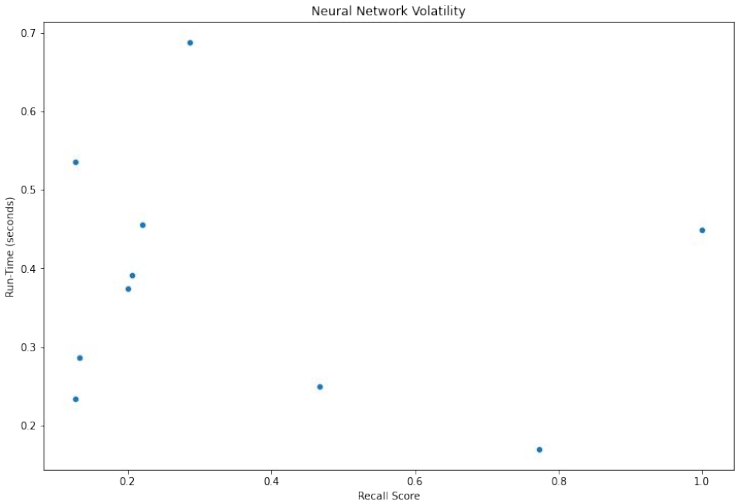


- Max-depth: 10
  - Duration: 0:00:00.448384
  - Recall Score: 0.8866666666666667



Improvement: Neural Network

- Potential for high recall score/low execution time
  - Highly volatile
- Research actively done in this field
- Requires significantly more time and research to implement correctly
- Ran loop to show volatility over 10 iterations



Conclusion

- Use supervised ML techniques to identify credit card fraud
- Initially used NB and Random Forest
  - Saw improvement when increasing max_depth but limited
- Potential with Neural Networks
  - Currently volatile
  - Need more research and time