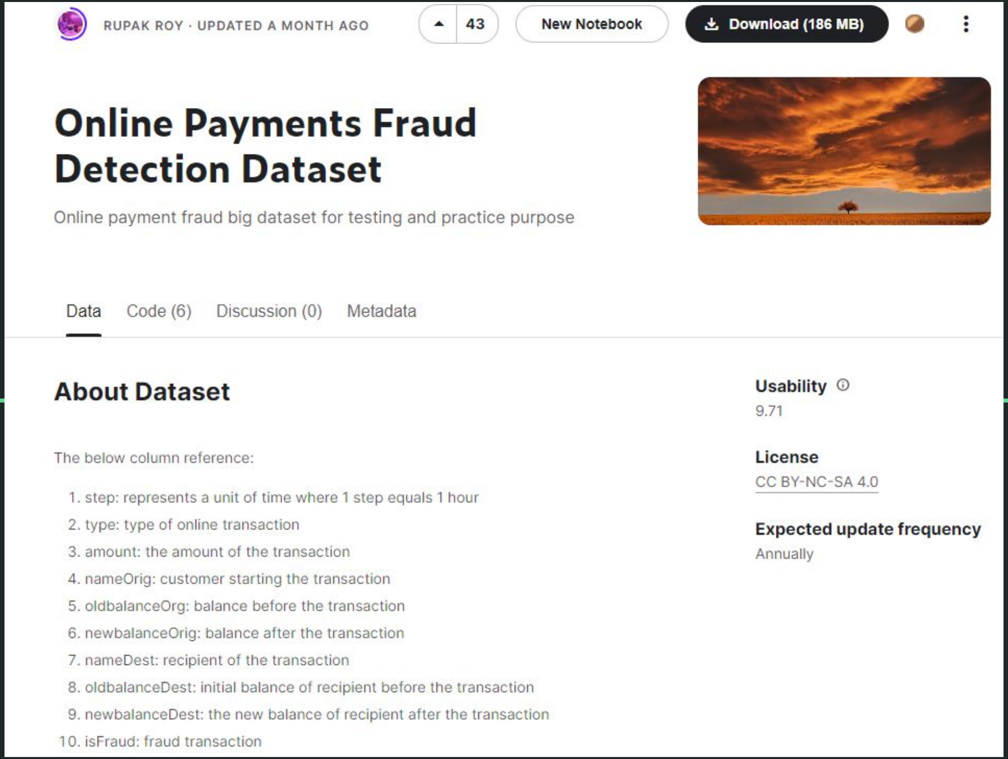
Machine Learning Project - Online Payments Fraud Detection

By: James Cheung

Online Payments Fraud Detection Dataset

- From Kaggle
- Columns:
  - step (1 step equals 1 hour)
  - type of online transaction
  - amount of transaction
  - initiating customer name
  - balance before the transaction
  - balance after the transaction
  - recipient name
  - initial balance of recipient
  - new balance of recipient
  - whether is fraud transaction



Online Payments Fraud Detection Dataset

Online payment fraud big dataset for testing and practice purpose

About Dataset

The below column reference:

1. step: represents a unit of time where 1 step equals 1 hour
2. type: type of online transaction
3. amount: the amount of the transaction
4. nameOrig: customer starting the transaction
5. oldbalanceOrg: balance before the transaction
6. newbalanceOrig: balance after the transaction
7. nameDest: recipient of the transaction
8. oldbalanceDest: initial balance of recipient before the transaction
9. newbalanceDest: the new balance of recipient after the transaction
10. isFraud: fraud transaction

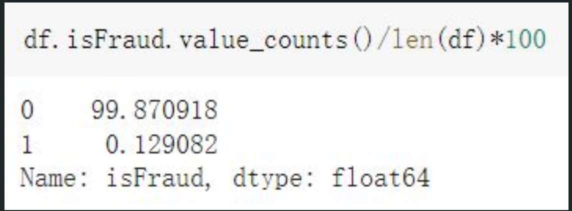
Usability
9.71

License
CC BY-NC-SA 4.0

Expected update frequency
Annually

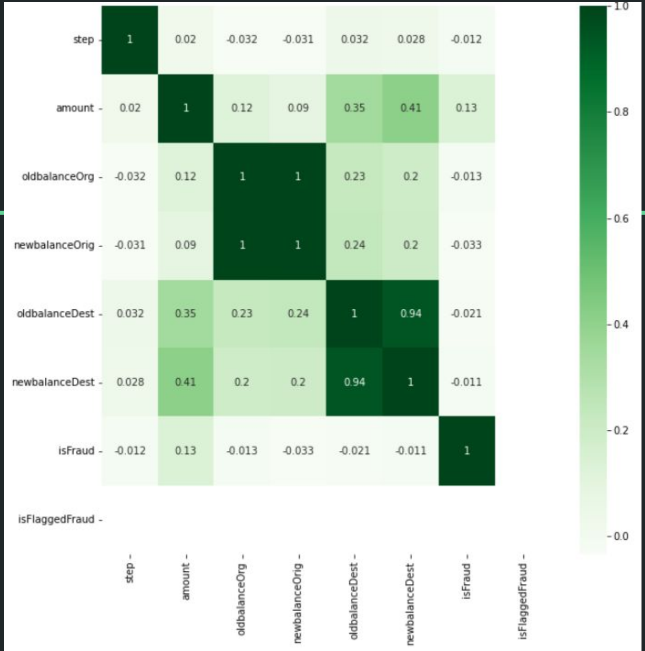
Importing Data to Google Collab

- Total data set: 6.36 million rows x 9 columns
- Not Fraud Records vs Fraud Records
  - 99.87% vs 0.13%
- Dropping Irrelevant Columns
- Selecting only 10000 rows
- OneHotEncoding Transaction Types
  - Cash Out
  - Payment
  - Cash In
  - Transfer Debit
- StandardScaler for Feature Scaling



```
df.isFraud.value_counts()/len(df)*100
```

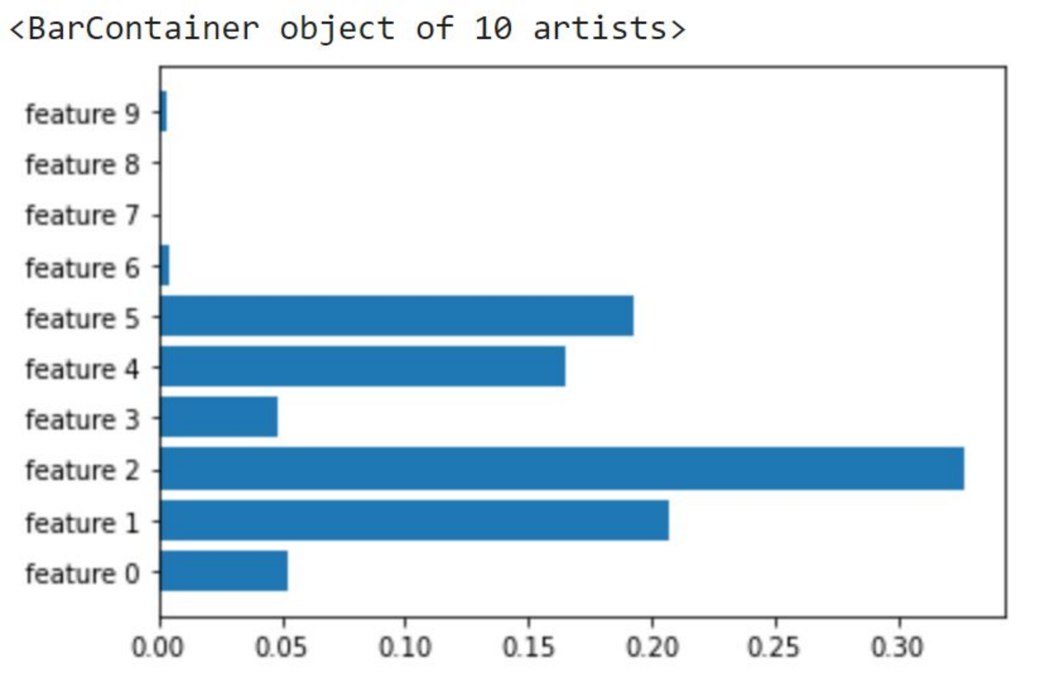
```
0    99.870918
1     0.129082
Name: isFraud, dtype: float64
```



	step	amount	oldbalanceOrg	newbalanceOrig	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud
step	1	0.02	-0.032	-0.031	0.032	0.028	-0.012	
amount	0.02	1	0.12	0.09	0.35	0.41	0.13	
oldbalanceOrg	-0.032	0.12	1	1	0.23	0.2	-0.013	
newbalanceOrig	-0.031	0.09	1	1	0.24	0.2	-0.033	
oldbalanceDest	0.032	0.35	0.23	0.24	1	0.94	-0.021	
newbalanceDest	0.028	0.41	0.2	0.2	0.94	1	-0.011	
isFraud	-0.012	0.13	-0.013	-0.033	-0.021	-0.011	1	
isFlaggedFraud								

Feature Importance on 3,000,000 Records

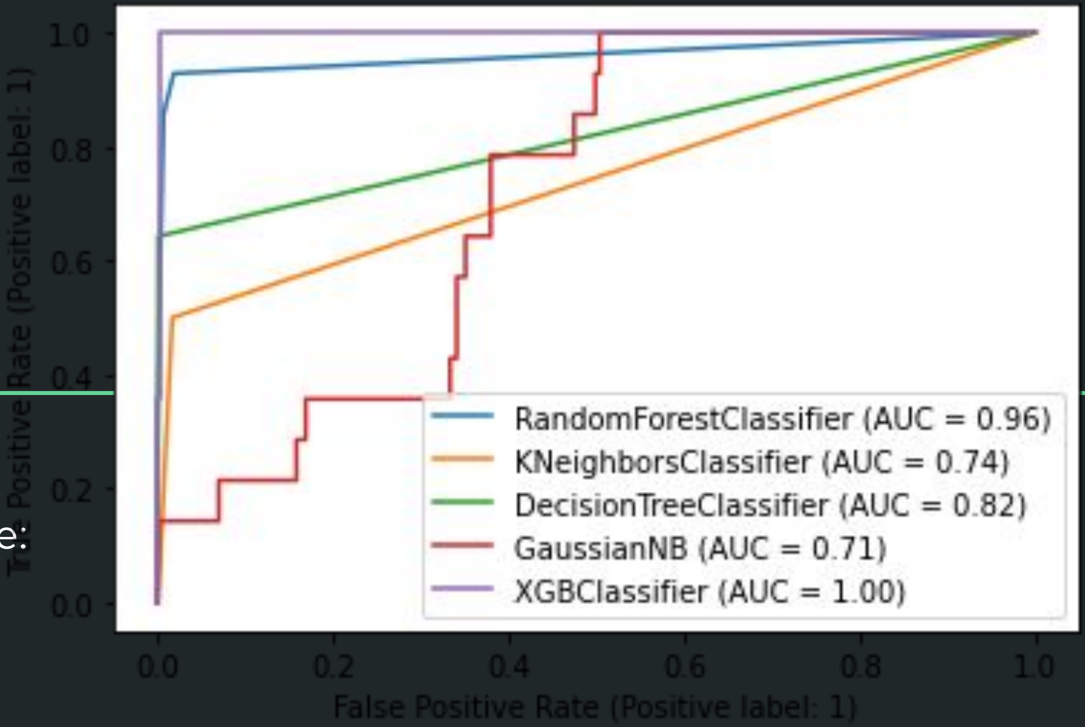
- Features less important:
    - step
    - new balance of payee
    - type - cash out
    - type - debit
    - type - payment
    - type - transfer
- Dropping 3 Columns and start over



```
<BarContainer object of 10 artists>
```

Classifiers

- Restarting on 10,000 Records
- Classifiers used:
  - KNN
  - Decision Tree
  - Naive Bayes
  - XGBoost
  - Random Forest
- Area under Curve best performance:
  - XGBoost
  - Random Forest



Performance on Raw Data

- Low Performance on Precision, Recall & F1-score
- Generally high False Negative Numbers

Classifier	Classification Report				
		precision	recall	f1-score	support
KNN - 99.3%	0	0.99	1.00	1.00	1986
	1	0.00	0.00	0.00	14
Decision Tree - 99.7%	0	1.00	1.00	1.00	1986
	1	0.90	0.64	0.75	14
Naive Bayes - 98.05%	0	0.99	0.99	0.99	1986
	1	0.07	0.14	0.09	14
XGBoost - 99.35%	0	0.99	1.00	1.00	1986
	1	1.00	0.07	0.13	14
Random Forest - 99.5%	0	0.99	1.00	1.00	1986
	1	1.00	0.29	0.44	14

Class Imbalance - SMOTEENN & ReSampling

- SMOTEENN & ReSampling on Raw Data
- Random Forest & XGBoost used

Classifier	Classification Report					**Confusion Matrix**
		precision	recall	f1-score	support	
SMOTEENN on Random Forest	0	1.00	0.99	0.99	1986	[[1962 24]
	1	0.33	0.86	0.48	14	[2 12]]
SMOTEENN on XGBoost	0	1.00	0.97	0.98	1909	[[1853 56]
	1	0.97	1.00	0.98	1957	[8 1949]]
DownSampling on Random Forest	0	0.93	0.93	0.93	14	[[13 1]
	1	0.93	0.93	0.93	14	[1 13]]
UpSampling on Random Forest	0	1.00	0.99	1.00	1925	[[1903 22]
	1	0.99	1.00	0.99	2048	[0 2048]]

Hyper Parameter Optimization on Raw Data

- HPT method used:
    - Manuel HPT on Random Forest
    - Randomized Search on Random Forest
    - Grid Search on Random Forest
    - Randomized Search on XGBoost

Hyper Parameter Tuning on ReSampled Data

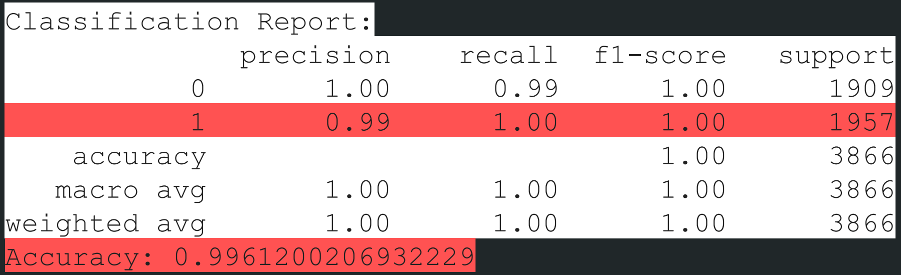
- Manuel HPT on SMOTEENN Data on Random Forest
- Randomized Search on SMOTEENN Data on Random Forest
  - Improvement of 0.05%
- Grid Search on SMOTEENN Data on Random Forest
  - Improvement of 0.21%
- Randomized Search on SMOTEENN Data on XGBoost
- Manuel HPT on Up-Sampled Data on Random Forest

Applying HPO on ReSampled Data

- XGBoost Hyper Parameters:
    - classifier_SM_HPO_FINAL=xgboost.XGBClassifier(colsample_bytree=0.7, gamma=0.0, learning_rate=0.2, max_depth=15, min_child_weight=5)
- Random Forest Hyper Parameters:
    - RandomForestClassifier(bootstrap=True,max_depth=80, max_features=3, min_samples_leaf=3, min_samples_split=8, n_estimators=100)

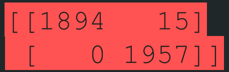
Final Accuracies

<table>
<tr><th>Classifier</th><th>Classification Report</th><th>Confusion Matrix</th></tr>
<tr><td>XGBoost on SMOTEENN Data</td><td>



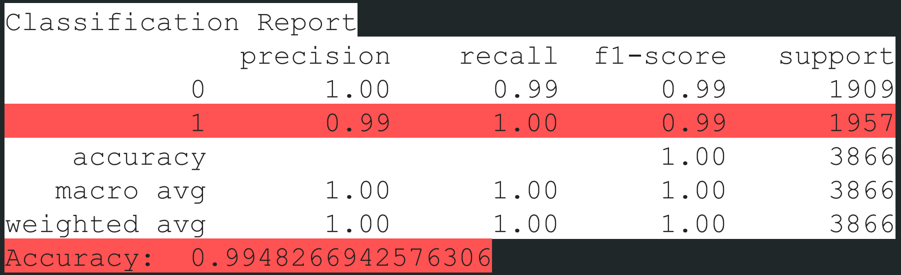
```
Classification Report:
              precision    recall  f1-score   support
           0       1.00      0.99      1.00      1909
           1       0.99      1.00      1.00      1957
    accuracy                           1.00      3866
   macro avg       1.00      1.00      1.00      3866
weighted avg       1.00      1.00      1.00      3866
Accuracy:  0.9961200206932229
```

</td><td>



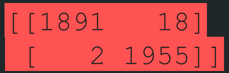
```
[[1894   15]
 [   0 1957]]
```

</td></tr>
<tr><td>Random Forest on SMOTEENN Data</td><td>



```
Classification Report
              precision    recall  f1-score   support
           0       1.00      0.99      0.99      1909
           1       0.99      1.00      0.99      1957
    accuracy                           1.00      3866
   macro avg       1.00      1.00      1.00      3866
weighted avg       1.00      1.00      1.00      3866
Accuracy:   0.9948266942576306
```

</td><td>



```
[[1891   18]
 [   2 1955]]
```

</td></tr>
</table>

~ The End ~

Q&A