Data science project

Inverse random undersampling (fraudulent dataset)

k18 0205
K18 0318
K18 1113

Introduction

- The class Imbalance problem typically takes place when one class outnumbers the other classes.
- In this case standard classifiers tend to incline towards large classes and ignore the smaller ones.
- Therefore performance of classifiers drops insignificantly.
- Various strategies have been proposed to deal with class imbalance problems
- In this study we will make use of the IRUS algorithm to balance the number of classes in the dataset.

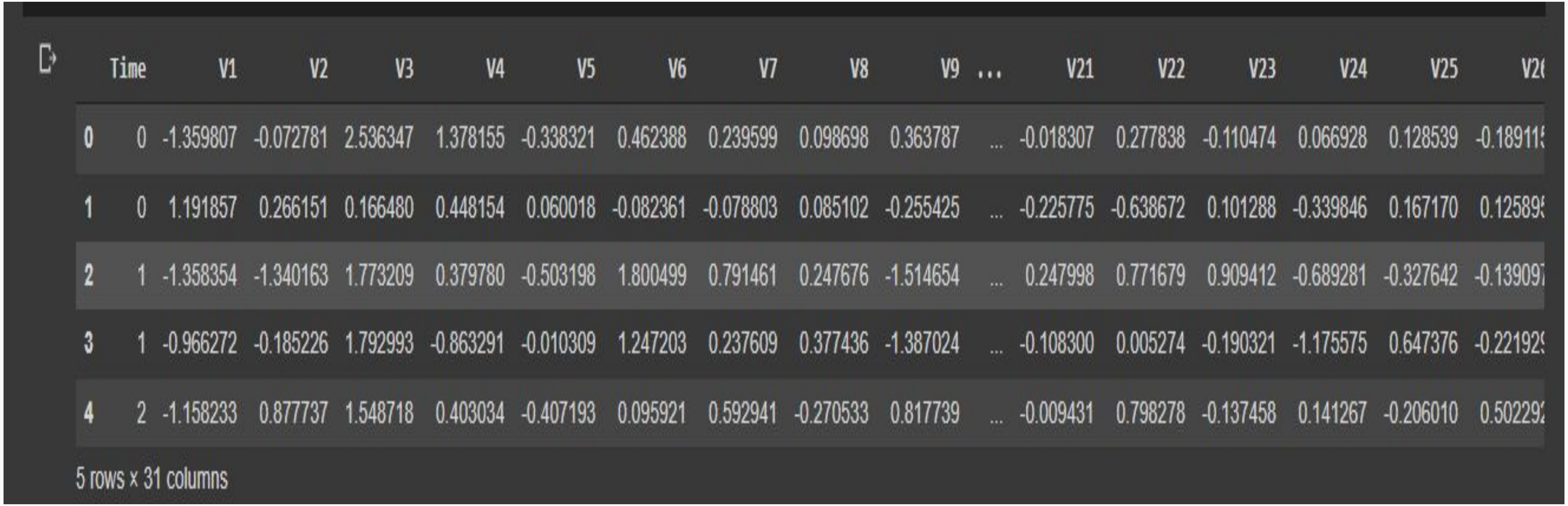
Research Goal

The goal of our paper is to demonstrate the effects of class imbalance on classification models. Moreover, we study the impact of varying class imbalance ratios on classifier accuracy. Furthermore, we assess the improvement in accuracy after implementing the IRUS algorithm on the said dataset.

Retrieving the data

- credit card fraud detection
- 31 features out of which Features V1, V2, … V28 are the principal components obtained with PCA,
- The only features which have not been transformed with PCA are 'Time' and 'Amount'. Feature 'Time'

Dataset



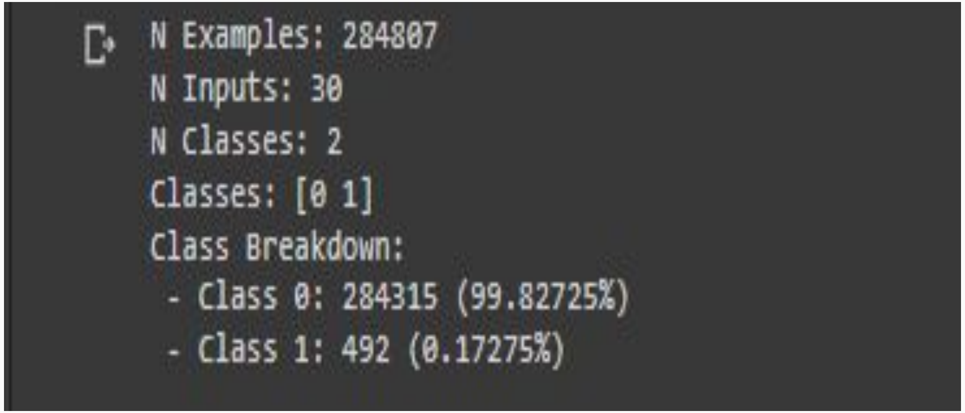
	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V24	V25	V2
0	0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307	0.277838	-0.110474	0.066928	0.128539	-0.18911
1	0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775	-0.638672	0.101288	-0.339846	0.167170	0.12589
2	1	-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	-0.327642	-0.13909
3	1	-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	0.647376	-0.22192
4	2	-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	-0.206010	0.50229

5 rows × 31 columns

Data Prepration

Total number of classes: 2

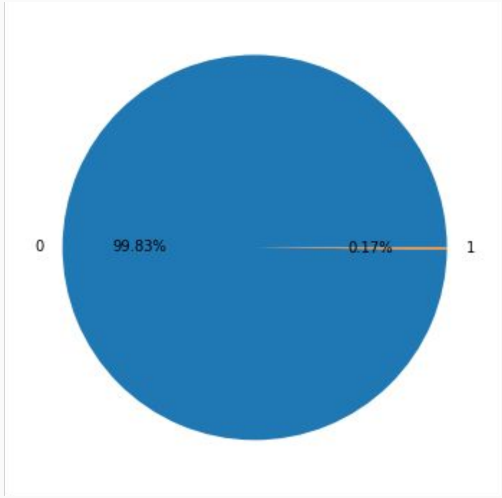
Size of dataset: 284807



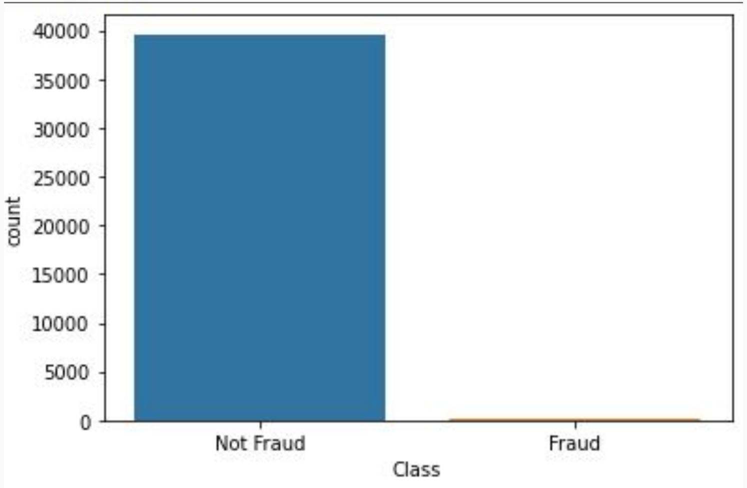
```
N Examples: 284807
N Inputs: 30
N Classes: 2
Classes: [0 1]
Class Breakdown:
 - Class 0: 284315 (99.82725%)
 - Class 1: 492 (0.17275%)
```

Data visualization

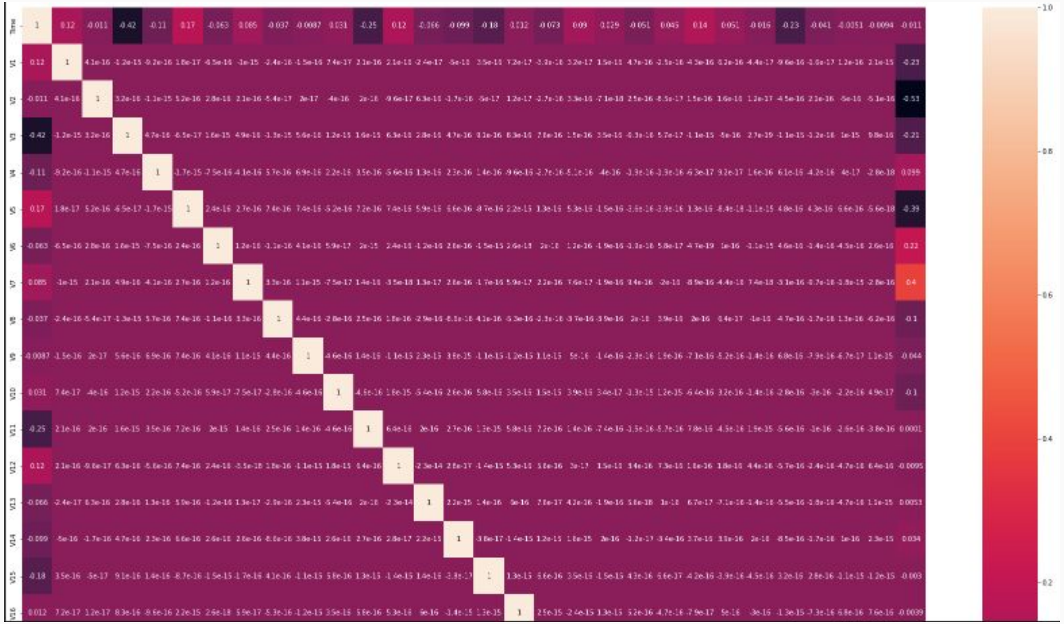
Pie chart



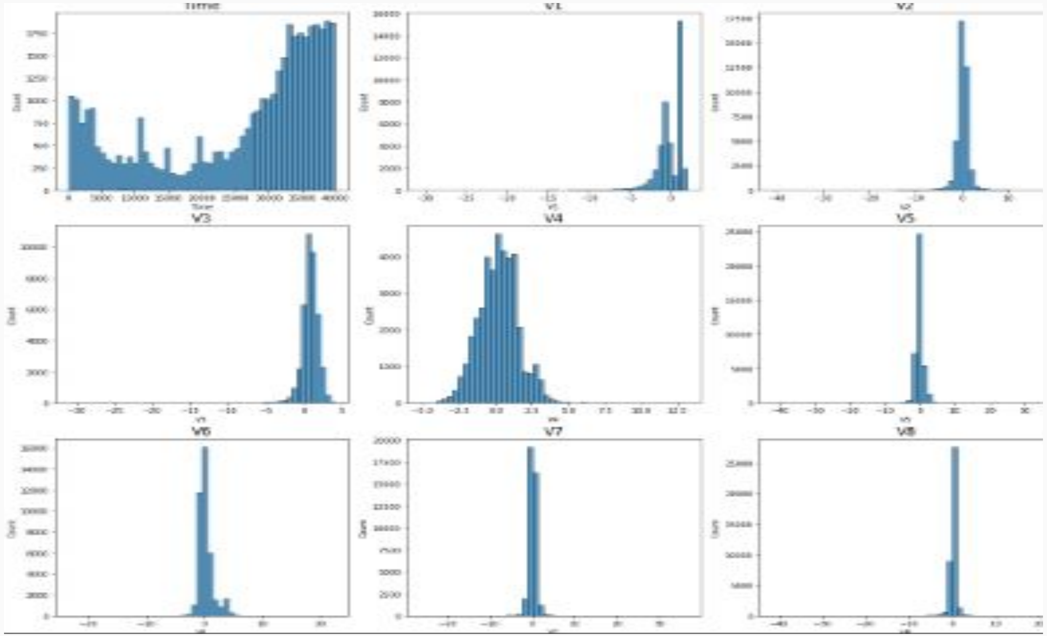
Bar graph



Correlation Matrix



Histogram


Time
Count
0
250
500
750
1000
1250
1500
1750
0
5000
10000
15000
20000
25000
30000
35000
40000
Time
V1
V2
V3
V4
V5
V6
V7
V8

Summary of EDA

- Data looks clean
- No null values
- Majority of features are well distributed around mean
- There are some correlated features. but not strong enough to drop
- Dataset is highly imbalanced.

Data models

The following Three models were used:

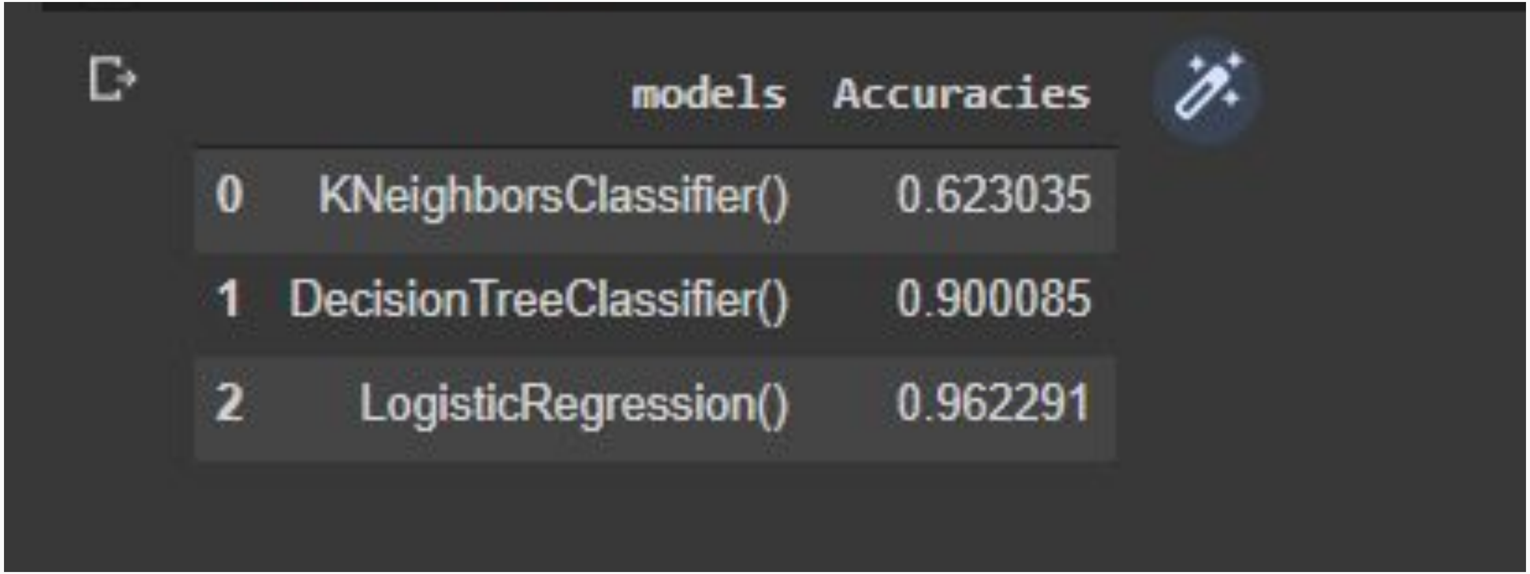
- KNN
- Decision Tree
- Logistic Regression

Inverse random under sampling

The following steps were performed in the algorithm:

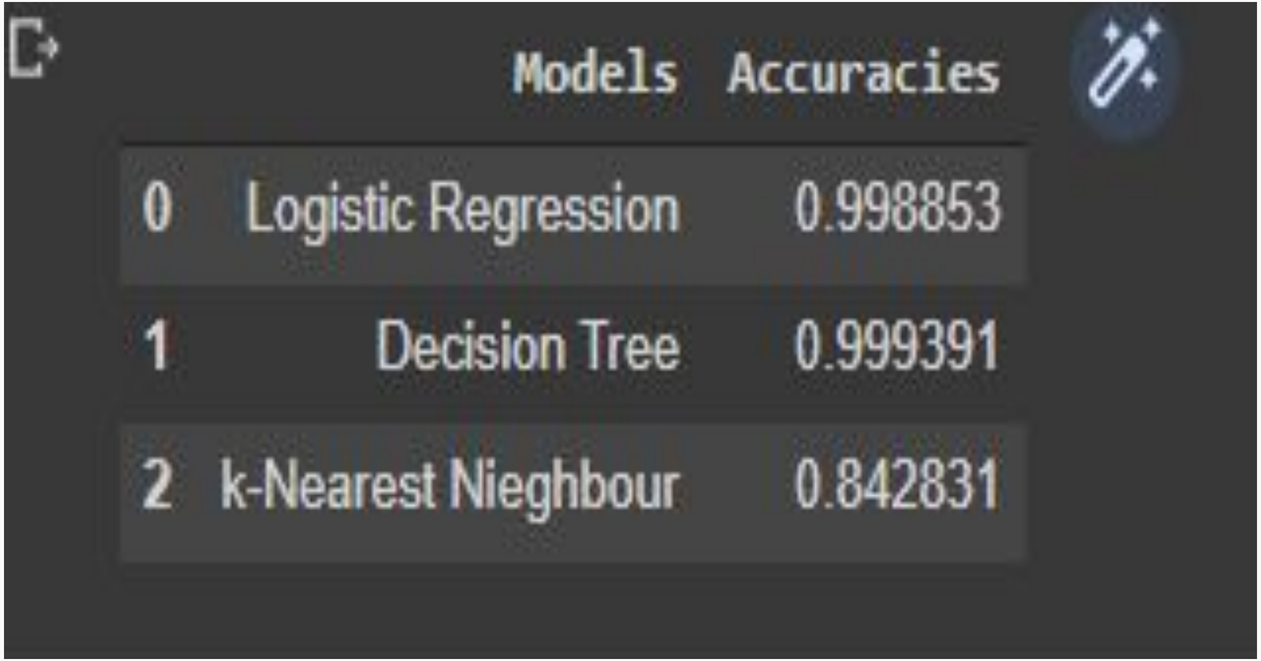
1. Dataset was divided on the basis of classes. XNmaj consisted of majority class while XMin consisted of minority class
2. The length of both classes was figured out. XNmaj measured 19900 rows while Xmin measured 360 rows
3. The value of S was set given that S<XNmin
4. Values of sets was calculated using formula; int(1.5*cield(Nmaj/s))
5. The models were trained using dataset that was XNmaj U XNmin
6. The norm and confidence was calculated

Results with IRUS algorithm



	models	Accuracies
0	KNeighborsClassifier()	0.623035
1	DecisionTreeClassifier()	0.900085
2	LogisticRegression()	0.962291

Results without IRUS alogorithm



	Models	Accuracies
0	Logistic Regression	0.998853
1	Decision Tree	0.999391
2	k-Nearest Nieghbour	0.842831

Conclusion

We can see the models gave high accuracy in both cases (with or without IRUS implementation) However models that performed without IRUS implementation were more biased towards the majority class. Using the IRUS algorithm we were able to balance out the occurrences of both classes in the dataset so that the model’s prediction was accurate