Credit Card Fraud Detection

Bharadiya Pavan (MT2018023), Devarakonda Srinivas Deepak (MT2018031)

1. Problem Statement

Our goal is to apply different machine learning algorithms in order to classify credit card fraud from a dataset which contains transactions made by credit cards in September 2013 by european cardholders.

2. Challenges

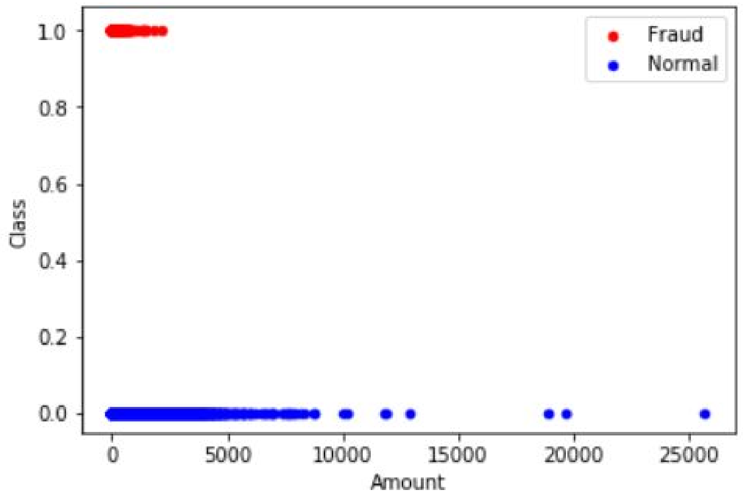
Some challenges we observed from the start is huge imbalance is the dataset,we have 492 frauds out of 284,807 transactions.The positive class (frauds) account for 0.172% of all transactions.In this case, it is much worse to have false negatives than false positives in our predictions because false negatives mean that someone gets away with credit card fraud. False positives, on the other hand, merely cause a complication and possible hassle when a cardholder must verify that they did, in fact, complete said transaction (and not a thief).

3. Dataset

We have taken data from kaggle credit card fraud competition.Link for dataset is https://www.kaggle.com/mlg-ulb/creditcardfraud.

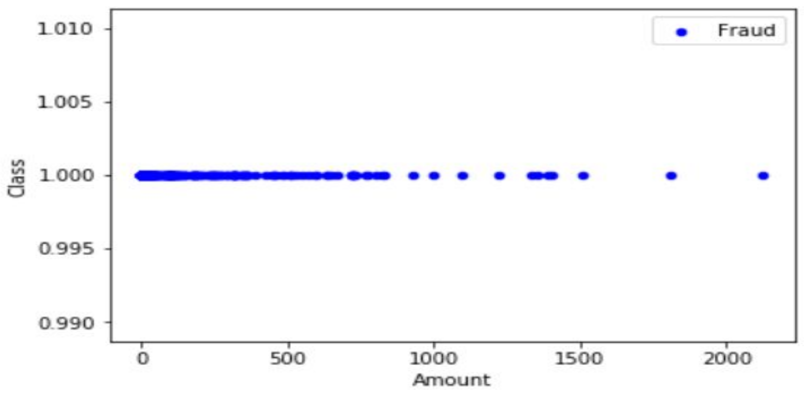
4. Exploratory Data Analysis(EDA)

Plotting fraud and non-fraud transactions with respect to amount.



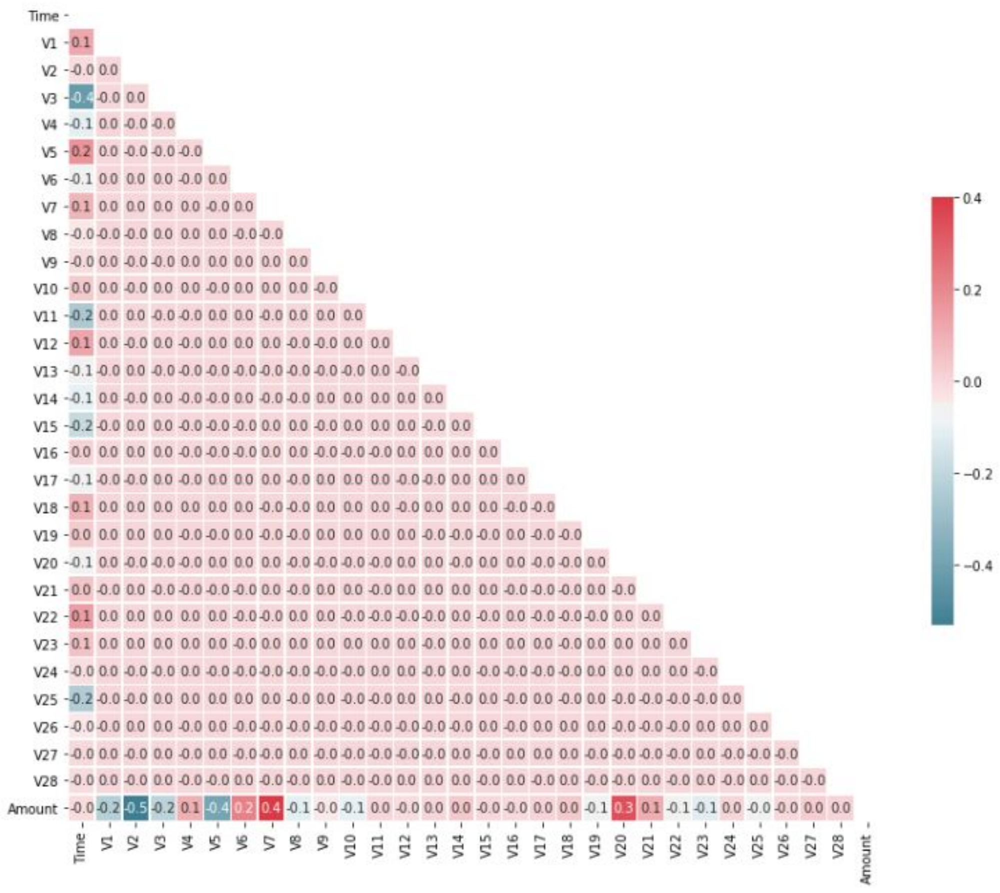
By the above figure we can infer that the data is very imbalanced, class label 1 has only 492 entries and class 0 has 284315.

Checking the amount for which fraud transaction occurred.



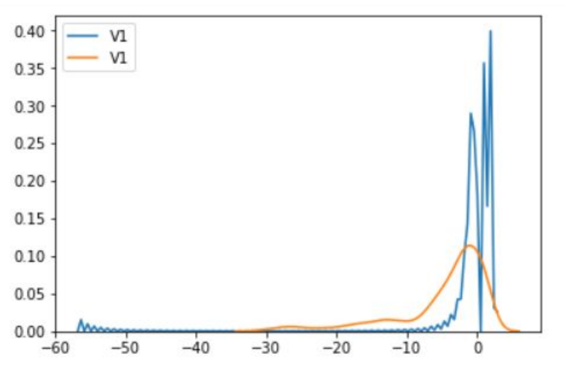
For fraudulent transactions majority of the those transactions occur upto amount 900

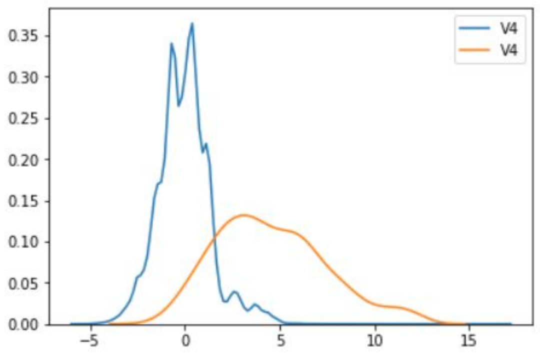
Plotting correlation between all the features

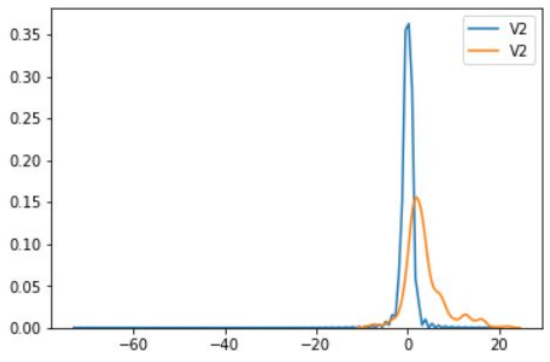


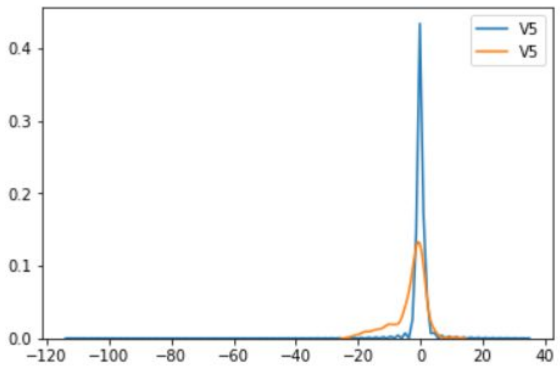
The correlation map does not tell us much. We think it is because that the V-parameters are not directly related with each others and we do not exactly know what is going on in the calculation process of PCA.

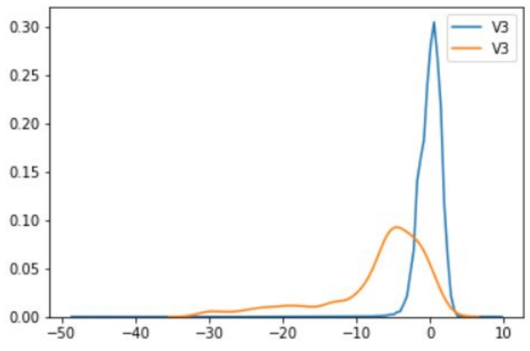
Kernel Density Estimation(KDE) plots of all features (for feature selection)

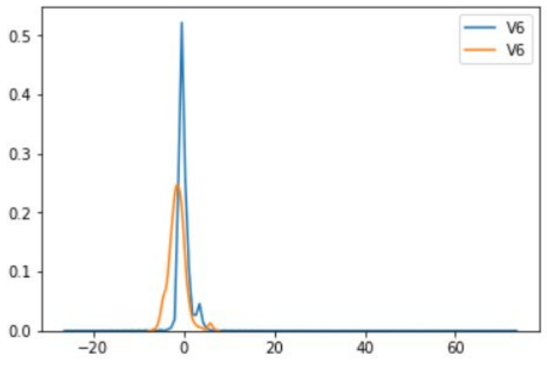


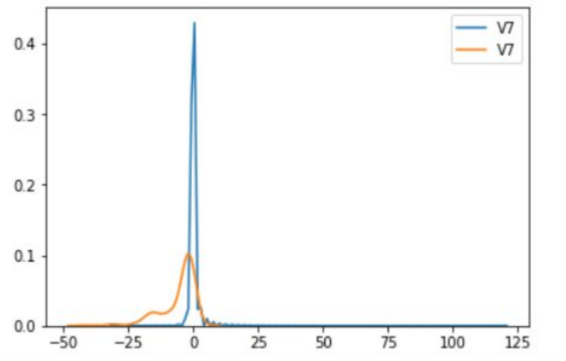
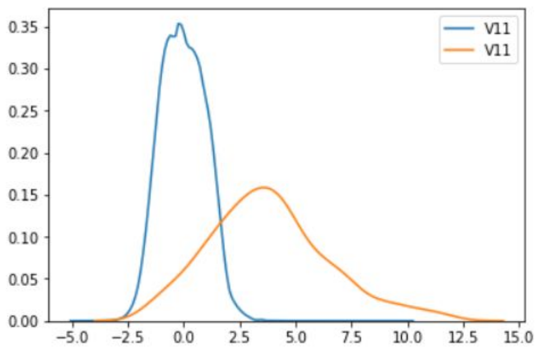
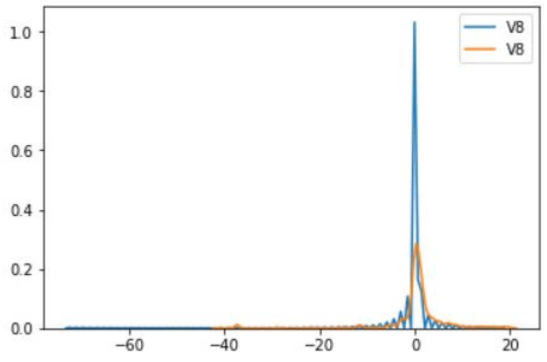
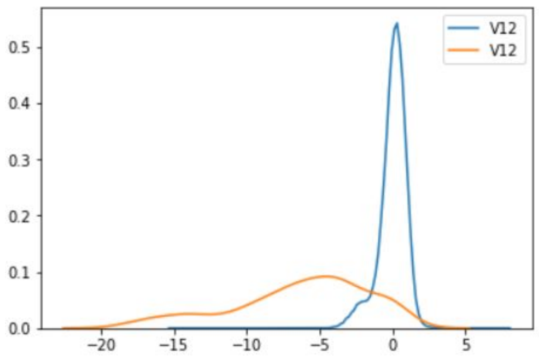
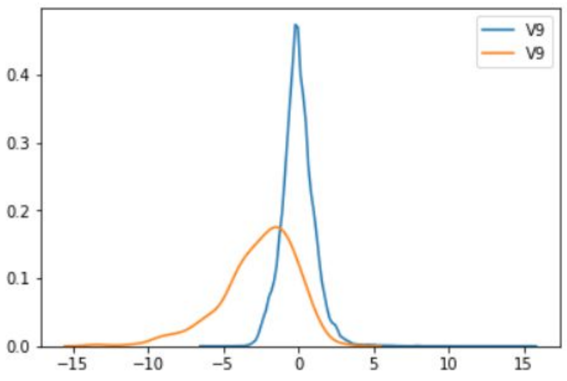
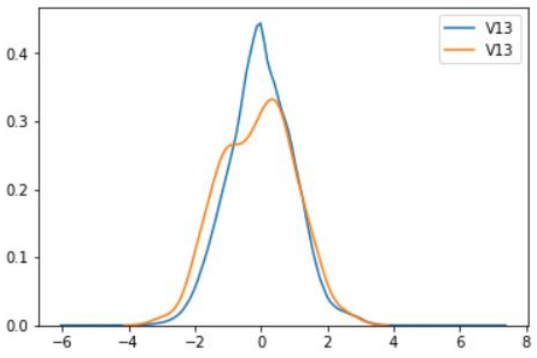
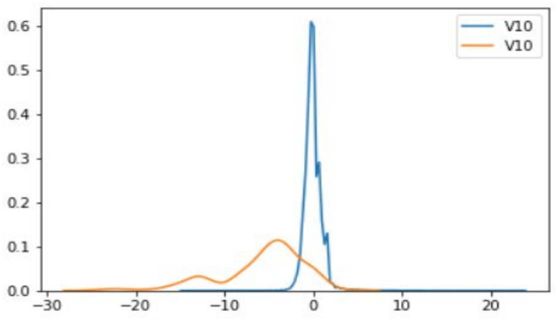
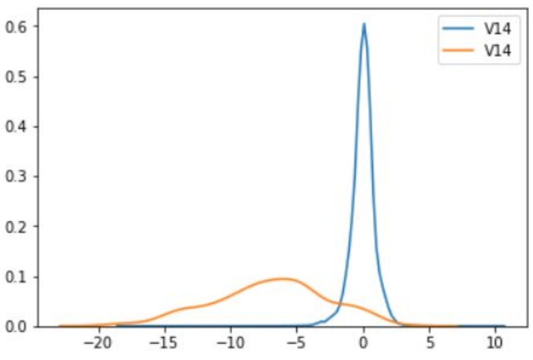


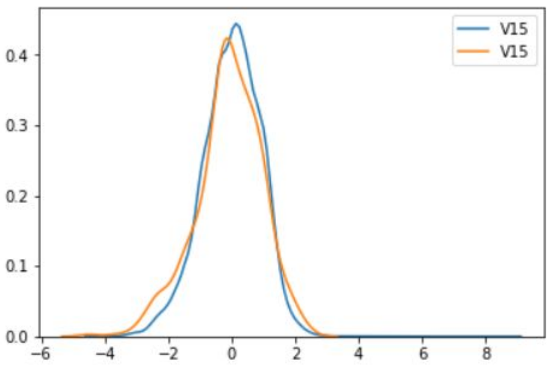
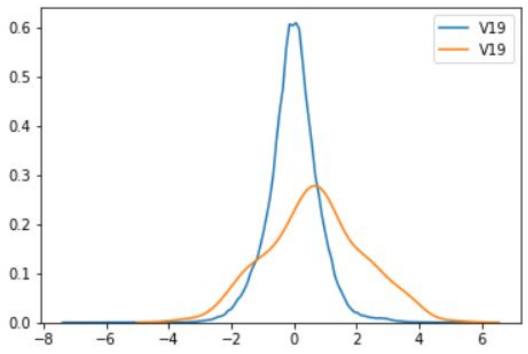
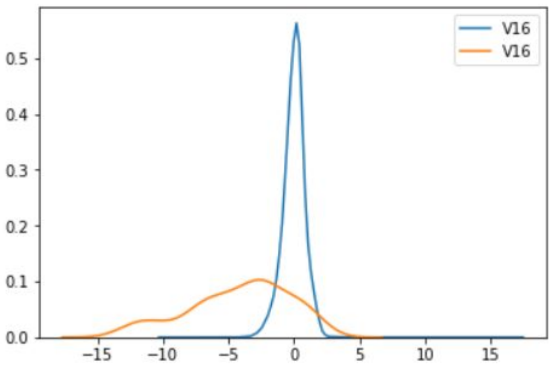
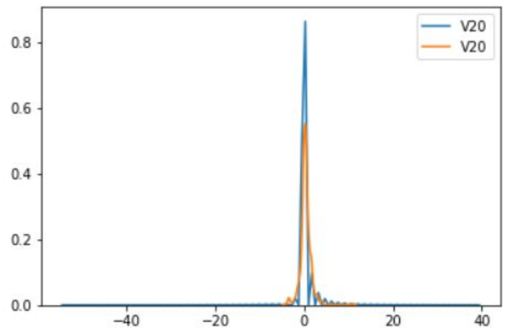
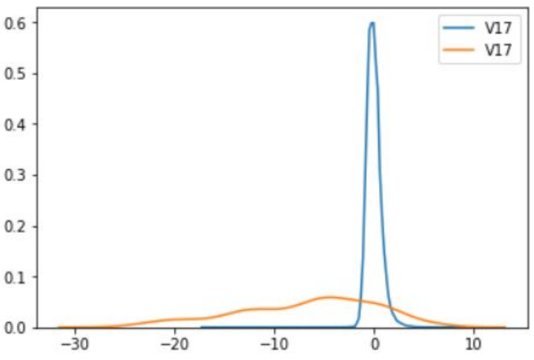
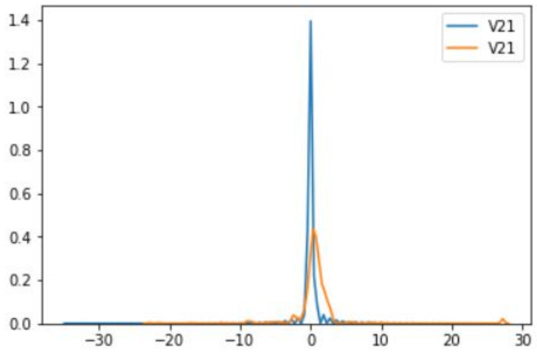
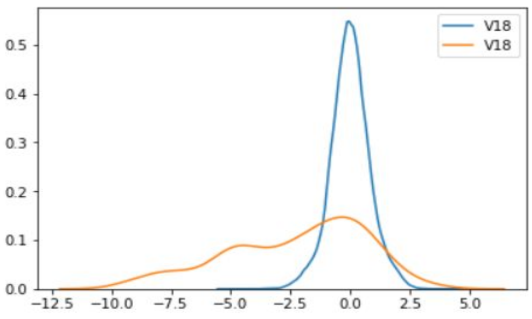
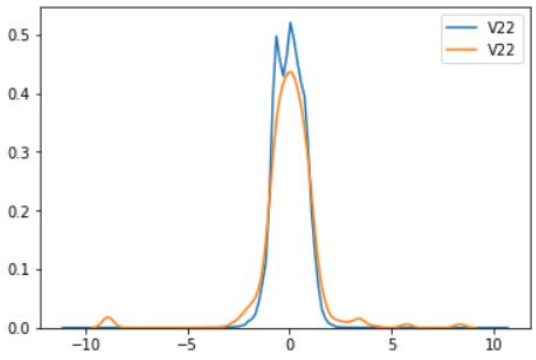


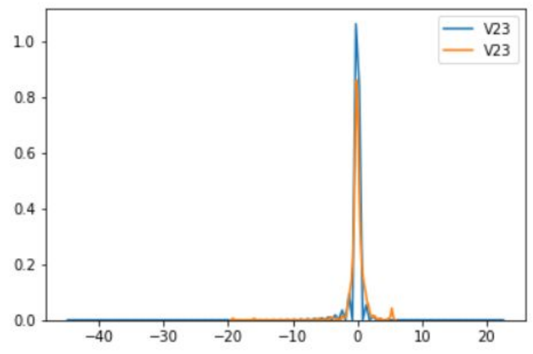


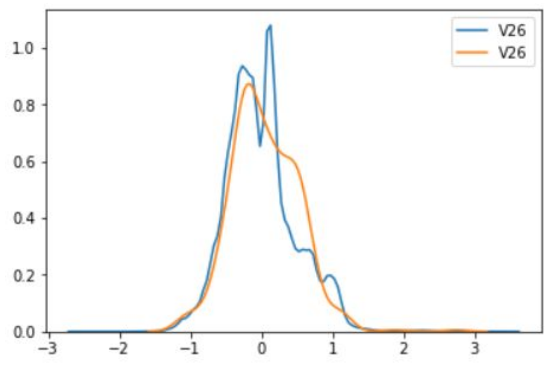


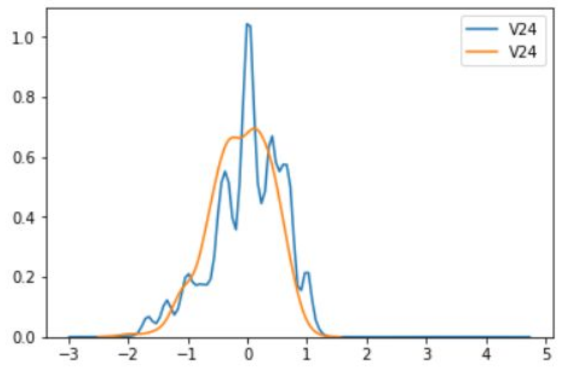


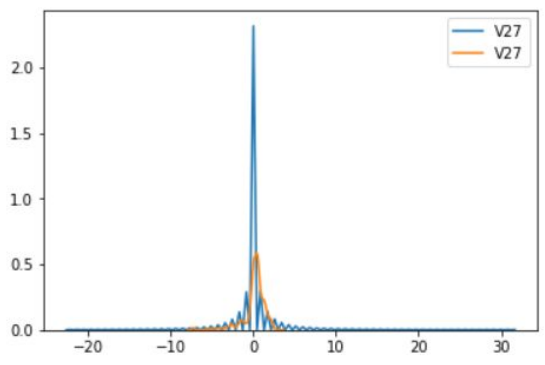

V7
V7

V11
V11

V8
V8

V12
V12

V9
V9

V13
V13

V10
V10

V14
V14

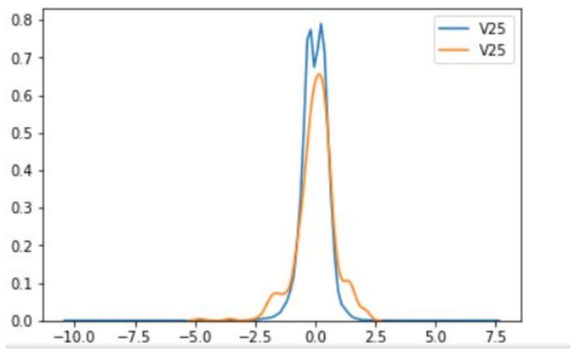

V15
V15

V19
V19

V16
V16

V20
V20

V17
V17

V21
V21

V18
V18

V22
V22

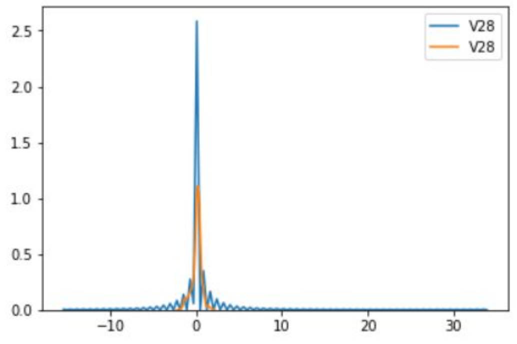












From the above plots we use only those features that look distinct for anomaly class and normal class.

5. Data Preprocessing

One challenging aspect of the dataset was that there were 30 features, but in order to protect confidentiality, all but 28 of them had been PCA transformed and had unknown labels. The known, non-transformed features were ’Time’, which measures the seconds between the transaction and the first transaction in the 2-day time period, and ’Amount’, which is the cost of the transaction. In order to offset the imbalance in the dataset, we oversampled the fraud (class = 1) portion of the data, adding Gaussian noise to each row using Synthetic Minority Oversampling Technique(SMOTE).

6. Data Modeling

The 4 models we used were logistic regression,decision tree and random forest, gradient boosting machine, and SVM.

a. The first model we used was logistic regression, as it was a good candidate for binary classification. On each logistic regression model we trained, we made the constant C to be 100000. We trained three different configurations: a basic logistic regression with no preprocessing whatsoever, a logistic regression with oversampling on the scarce fraudulent data points and data scaling, and a logistic regression with balanced weights for each class (which greatly helped the data unbalance).
b. The Second model we used was DecisionTreeClassifier. We used model with no preprocessing whatsoever, and the trained model with oversampling same as logistic regression which improved the model but it was not better than logistic regression.
c. The third model we tried was random which was again tested before and after oversampling. The model gave better results than decision tree but failed to beat the logistic regression.
d. The last model we tried was GradientBoostingClassifier which was again tested before and after oversampling. The model gave better results than decision tree and random forest but failed to beat the logistic regression.

7. Results

Logistic Regression :

Without preprocessing : Accuracy = 0.9980, FNR(% of number of misclassification) =0.3,ROC area = 0.83
After Over sampling : Accuracy = 0.9822, FNR =0.0914, ROC area = 0.94
After Balanced Class Weights : Accuracy = 0.9757, FNR =0.09375, ROC area = 0.94

Decision Tree:

Without preprocessing : Accuracy = 0.9991, FNR = 0.2192,ROC area = 0.89
After Over sampling : Accuracy = 0.9975, FNR =0.1871, ROC area = 0.91

Random Forest Classifier:

Without preprocessing : Accuracy = 0.9995, FNR = 0.1925,ROC area = 0.90
After Over sampling : Accuracy = 0.9994, FNR =0.1604, ROC area = 0.91

GradientBoostingClassifier:

Without preprocessing : Accuracy = 0.9991, FNR =0.3368,ROC area = 0.83
After Over sampling : Accuracy = 0.9942, FNR =0.18716, ROC area = 0.94

9. Conclusion

Logistic Regression was the best among all the models with good balance of Accuracy and FNR. We got least FNR using this model.