**Credit Card Fraud Detection** 

Credit card fraud is increasing over the years. In 2020, there were 393,207 reported cases of credit card fraud, which is an increase of 44.7% from 2019. People can imagine the amount of money that people have lost because of these frauds. Machine learning algorithms are used to detect fraud, which we will be using for this project. It is important that credit card companies are able to recognize fraudulent credit card transactions so that customers are not charged for items that they did not purchase. In this project we want to answer the question of "Does there exist an opportunity for credit card companies to detect fraud within transactions by using machine learning algorithms."

## The description of the dataset is explained well on the page where the data was collected from.

1. Data

"The dataset contains transactions made by credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class

(frauds) account for 0.172% of all transactions. It contains only numerical input variables which are the result of a PCA transformation. Unfortunately, due to confidentiality issues, we

cannot provide the original features and more background information about the data. 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' contains the seconds elapsed between each transaction and the first transaction in the dataset. The feature 'Amount' is the transaction

1 in case of fraud and 0 otherwise." • Credit Card Fraud Dection

## 75000

'Class'. We plot the features on a histogram to see the distribution of the different features.

3. EDA

As mentioned above V1, V2, ..., V28 are principal components obtained with PCA. There are other variables such as 'Time', 'Amount',

Amount, this feature can be used for example-dependant cost-sensitive learning. Feature 'Class' is the response variable and it takes value 2. Data Wrangling The data set was very clean with zero null values with no duplicates.

150

100

50

Number of Transactions

10<sup>4</sup>

 $10^{3}$ 

 $10^{2}$ 

10<sup>1</sup>

75000 200000 150000

150000 10000 75000 5000 6000 150000 150000 Class

0.12 0.011 0.42 0.11 0.17 0.063 0.085 0.037 0.0087 0.031 0.25 0.12 0.066 0.099 0.18 0.012 0.073 0.09 0.029 0.051 0.045 0.14 0.051 0.016 0.23 0.041 0.0051 0.0094 0.011 0.012 1 41e-16 -1.2e-15-9/2e-16 18e-17 -6.5e-16 -1e-15 -2.4e-16-1.5e-16 74e-17 21e-16 21e-16 -2.4e-17 -5e-16 3.5e-16 7/2e-17 -3.9e-16 3/2e-17 15e-16 4/7e-16 -2.5e-16 4/3e-16 6/2e-16 4/4e-17 9/6e-16 1.6e-17 12e-16 21e-15 -0.23 -0.1

0.011 41e-16 1 32e-16-1.1e-15-52e-16-28e-16-21e-16-5-4e-17 2e-17 4e-16-2e-16-9-6e-17-63e-16-1.7e-16-5e-17-12e-17-2.7e-16-33e-16-7.1e-18-25e-16-8.5e-17-15e-16-16e-16-12e-17-45e-16-21e-16-5e-16-5e-16-5e-16-5e-16-5e-16-5e-17

0.17 18-17 52-16 6.5-17-1.7-15 1 24-16 27-16 74-16 74-16 5.2-16 74-16 5.2-16 74-16 5.9-16 66-16 5.9-16 66-16 8.7-16 62-15 13-16 53-16 1.5-16 3.6-16 3.9-16 13-16 8.4-18 1.1-15 4.8-16 4.3-16 66-16 5.6-18 0.39 0.095

0.25 21e-16 2e-16 16e-15 35e-16 72e-16 2e-15 14e-16 25e-16 14e-16 4 6e-16 1 64e-16 2e-16 2e-16 2e-16 13e-15 58e-16 7.2e-16 14e-16 7.4e-16-1.5e-16-5.7e-16 7.8e-16 4.5e-16 19e-15 5.6e-16 -1e-16 -2.6e-16 3.8e-16 0.0001 015 0.12 21e-16 9 6e-17 63e-16 -5.6e-16 7.4e-16 2.4e-16 -3.5e-18 1.8e-16 -1.1e-15 1.8e-15 6.4e-16 1 -2.3e-14 2.8e-17 -1.4e-15 5.3e-16 5.6e-16 3.e-17 1.5e-16 3.4e-16 7.3e-16 1.6e-16 1.8e-16 4.4e-16 5.7e-16 2.4e-16 4.7e-16 6.4e-16 0.0095 0.26 .0066 .2.4e-17 63e-16 28e-16 13e-16 59e-16 .1.2e-16 13e-17 .2.9e-16 23e-15 .5.4e-16 2e-16 .2.3e-14 1 22e-15 14e-16 6e-16 76e-17 42e-16 .1.9e-16 56e-18 1e-16 67e-17 7.1e-16 .1.4e-16 .5.5e-16 1.8e-16 4.7e-16 11e-15 0.0053 0.004

0.18 3.5e-16 -5e-17 9.1e-16 1.4e-16 -8.7e-16 -1.5e-15 -1.7e-16 4.1e-16 -1.1e-15 5.8e-16 1.3e-15 -1.4e-15 1.4e-16 -3.8e-17 1 1.3e-15 6.6e-16 3.5e-16 -1.5e-15 4.3e-16 6.6e-17 4.2e-16 -3.9e-16 4.5e-16 3.2e-16 2.8e-16 -1.1e-15 -1.2e-15 0.003 0.004

0.073 3.9e-16-2.7e-16 7.6e-16-2.7e-16 1.3e-16 2e-16 2.2e-16 2.3e-16 1.1e-15 1.5e-15 7.2e-16 5.6e-16 7.6e-17 1.2e-15 6.6e-16 2.5e-15 4.9e-15-3.9e-16-1.4e-15-8.2e-16-8.7e-16-3.7e-16-2.4e-16-2.7e-16-6.9e-16-6.1e-16-5.5e-17 0.0073 0.33 009 32e-17 33e-16 15e-16 5.1e-16 5.3e-16 12e-16 7.6e-17 -3.7e-16 5e-16 39e-16 14e-16 3e-17 42e-16 16e-15 3.5e-16 2.4e-15 4.9e-15 1 2.5e-15 -3.7e-16 9.4e-16 4.8e-16 1.9e-16 9e-17 6.6e-17 3e-16 22e-16 8e-16 0036 0.11

0.051 47e-16 2.5e-16 9.3e-16-1.9e-16-3.6e-16-1.9e-16 9.4e-16 2e-16 2.5e-16 2.5e-16-1.5e-16-3.4e-16 5.6e-18-1.2e-17 43e-16 5.2e-16-1.4e-15-3.7e-16 2.6e-16 1.9e-16 1.1e-15 2.7e-16 1.3e-16 1.4e-16 2.8e-16-1.1e-15-2.4e-16 0.34 0.02 0.045 - 2.5-16-8.5-17-1-9-16-3.9-16-3.9-16-5.8-17 - 2-16 3.9-16 3.9-16 1.2-16 1.2-15-5.7-16-7.3-16 1.2-15-7.9-16 7.3-16 1.2-16-5.6-17-16-8.2-16-9.4-16-5.16-16-7.6-16 7.6-16 7.6-16 1.2-15-5.16-16-1.2-16-5.5-16-5.5-

28e-16 16e-15 -7.5e-16 24e-16 1 12e-16 -1.1e-16 41e-16 5.9e-17 2e-15 24e-16 -1.2e-16 2.6e-16 -1.5e-15 2.6e-18 2e-16 1.2e-16 -1.9e-16 -1.9e-16 5.8e-17 4.7e-19 1e-16 -1.1e-15 4.6e-16 -1.4e-16 4.5e-16 2.6e-16 022 0.04e

4e-16 12e-15 22e-16 5.2e-16 5.9e-17 7.5e-17 2.8e-16 4.6e-16 1 46e-16 18e-15 5.4e-16 26e-16 5.8e-16 3.5e-16 15e-15 39e-16 34e-17 1.3e-15 12e-15 6.4e-16 32e-16 1.4e-16 2.8e-16 3.e-16 2.2e-16 4.9e-17 2.2e-16 4.9e-17 2.2e-16 3.e-16 3.e-16

-1.7e-16 4 7e-16 23e-16 66e-16 26e-16 26e-16 36e-16 38e-15 26e-16 27e-16 28e-17 22e-15 1 38e-17 -1.4e-15 12e-15 16e-15 2e-16 -1.2e-17 -3.4e-16 37e-16 39e-16 2e-16 8.5e-16 -1.7e-16 23e-15 0.034 0.3

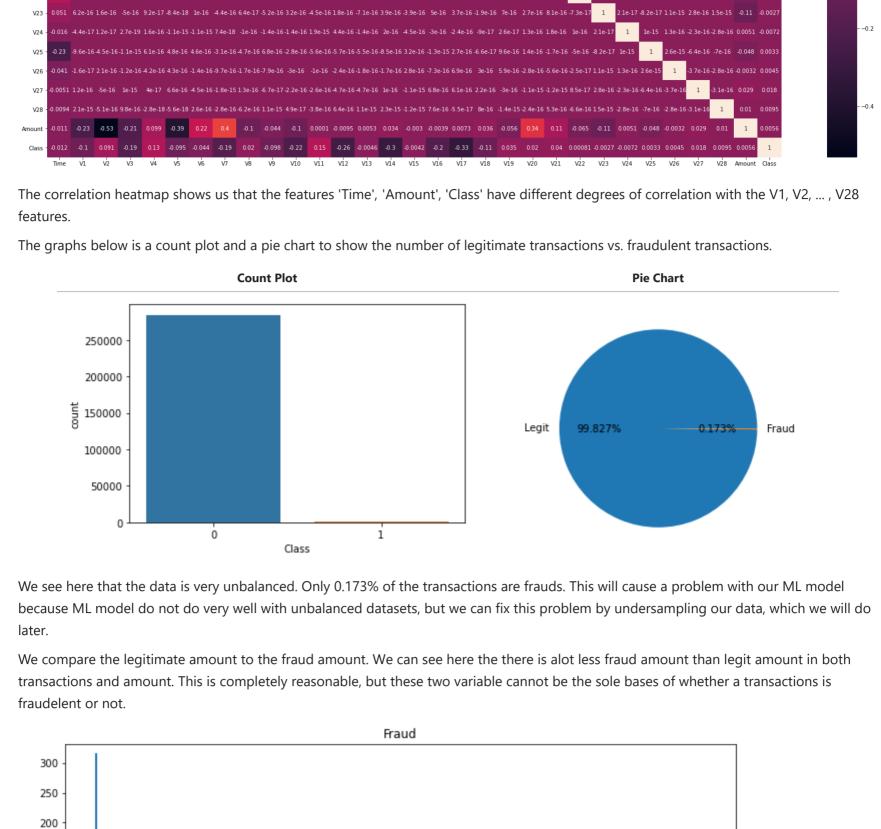
3.3e-16 1.1e-15-7-5e-17 1.4e-16-3.5e-18 1.3e-17 2.6e-16-1.7e-16 5.9e-17 2.2e-16 7.6e-17-1.9e-16-9.4e-16-2e-16-8.9e-16-4.4e-16 7.4e-18-3.1e-16-9.7e-16-1.8e-15-2.8e-16-0.4e-16-2e-16-8.9e-16-4.4e-16-7.4e-18-3.1e-16-9.7e-16-1.8e-15-2.8e-16-0.4e-16-2e-16-8.9e-16-4.4e-16-7.4e-18-3.1e-16-9.7e-16-1.8e-15-2.8e-16-0.4e-16-2e-16-8.9e-16-4.4e-16-7.4e-18-3.1e-16-9.7e-16-1.8e-15-2.8e-16-0.4e-16-2e-16-8.9e-16-4.4e-16-7.4e-18-3.1e-16-9.7e-16-1.8e-15-2.8e-16-0.4e-16-2e-16-8.9e-16-16-7.e-16-7.e-16-9.7e-16-1.8e-15-2.8e-16-0.4e-16-2e-16-0.4e-16-7.e-16-7.e-16-9.7e-16-18-15-2.8e-16-0.4e-16-2e-16-0.4e-16-7.e-16-7.e-16-9.7e-16-18-15-2.8e-16-0.4e-16-2e-16-0.4e-16-2e-16-0.4e-16-2e-16-0.4e-16-2e-16-0.4e-16-2e-16-0.4e-16-7.e-16-0.4e-16-0

6e-16 -1.4e-15 1.3e-15 1 2.5e-15 -2.4e-15 1.3e-15 5.2e-16 4.7e-16-7.9e-17 5e-16 -3e-16 -1.3e-15 -7.3e-16 6.8e-16 7.6e-16 -0.0039 0.2

-1.5e-15 1.3e-15 -3.9e-16-2.5e-15 1 2.6e-16 5.1e-16-1.2e-15 7e-16 2.6e-17 9.6e-16 5.9e-16 3e-16-1.4e-15 0.056 0.035

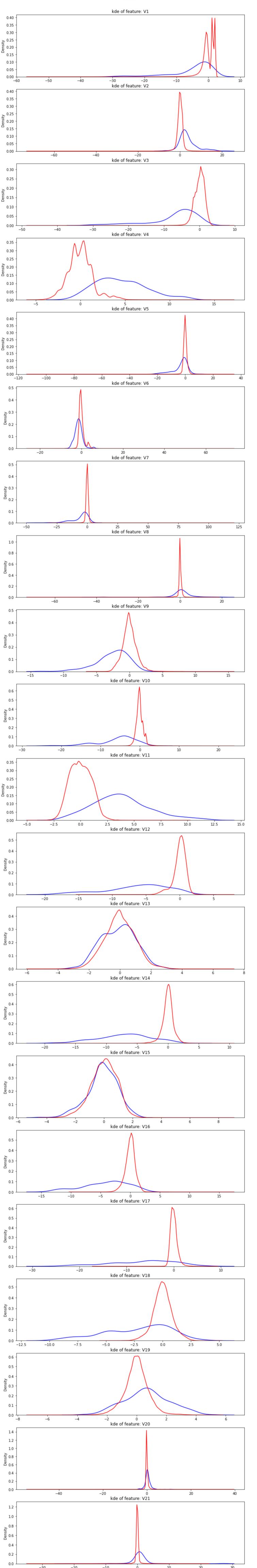
This was expected because it was stated that all of the features were transformed using PCA.

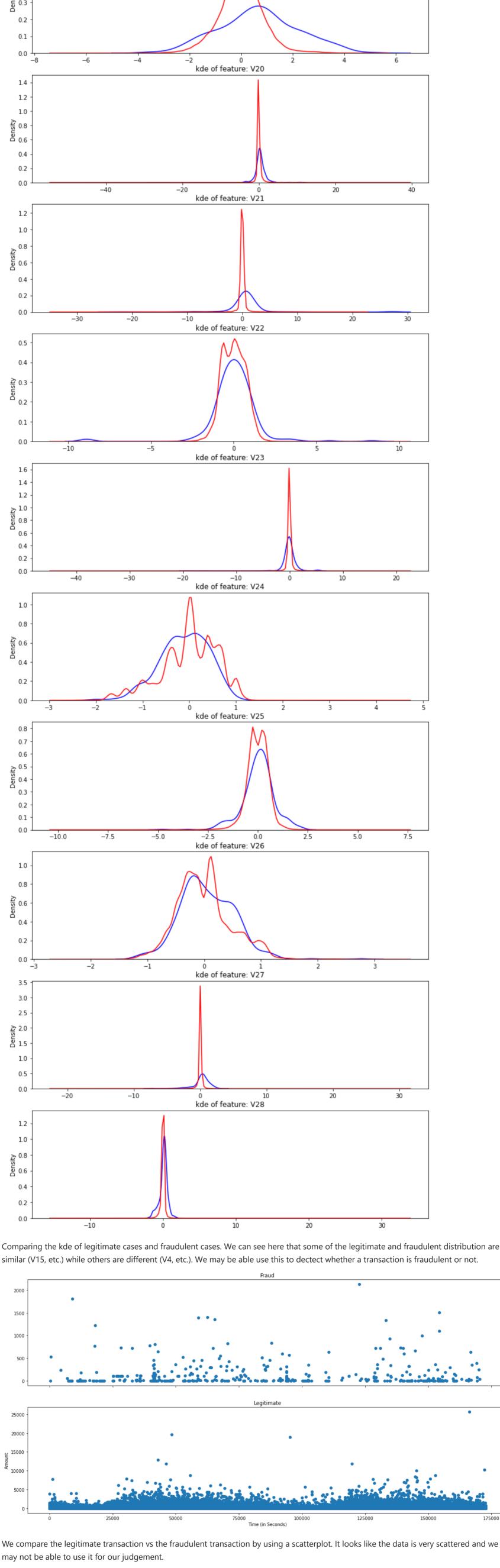
We then plot use a correlation heatmap to see the correlation between all of the features.



0 Legit 105

10° 5000 10000 20000 25000 15000 Amount kde of feature: V1 0.40 0.35 0.30 0.25 0.20 0.15 0.10 0.05





1 0.94 0.97 0.95 93 0.95 accuracy 0.95 0.96 197 macro avg 0.95 weighted avg 0.95 0.95 0.95 197

Tuned DecisionTree Parameters: {'criterion': 'entropy', 'max\_depth': 3, 'min\_samples\_leaf': 6}

0.91

0.91

0.91

0.91

0.91

0.94

0.95

0.94

0.94

support

99

98

197

197

197

0.95

0.94

0.94

0.94

0.94

support

102

197

197

197

95

0.96

recall f1-score

We are going to undersample because the data is imbalanced. We do this by taking a random sample (n=492) out of the legitimate

1

We split our data 80% training data and 20% testing data. The 80/20 train/test split was chosen because it is often the standard in the

It is important to scale the data when working with the ML models. If the data is not scaled, it can cause a problem with the model and makes it so that the model will not be able to learn properly. For this project, we used the StandardScaler form sklearn.preprocessing to

We are going to test 4 different models, Logistic Regression, Decision Tree Classifier, Support Vector Classifier, K Nearest Neighbors

Tuned Logistic Regression Parameters: {'C': 100, 'penalty': '12'}

0.94

Classifier. We chose these models because we want to predict whether a certain transaction is fraudulent or not. We also use GridSearchCV

datasets for all of the features to make sure that the data has not been changed that much.

Class

**Count Plot** 

transaction and then concatenating it with the fraudulent transaction, so that our data has 50% legitmiate transactions and 50% fraudulent transactions. This is so that our ML model will be able to predict the fradulent transactions better. We also check the mean of both of the

**Pie Chart** 

50.000%

50.000%

Fraud

support

104

Legit

## accuracy macro avg 0.91 weighted avg 0.91 **Support Vector Classifier**

• Support Vector Classifier

1

4. Preprocessing

500

400

300

200

100

0

industry.

scaled the data.

5. Modeling

**Logistic Regression** 

• Logistic Regression

Ó

to pick out the best parameters for each of the model.

Explanation of Logistic Regression can be found in the link below.

Best score is 0.9453438684189308

precision

Explanation of Decision Tree Classifier can be found in the link below.

0.90

0.92

Explanation of Support Vector Classifier can be found in the link below.

0.97

Accuracy is 0.9543147208121827

0

**Decision Tree Classifier** 

Best score is 0.9161251310166895 Accuracy is 0.9086294416243654

• Decision Tree Classifier

Best score is 0.9428122228493108 Accuracy is 0.9441624365482234 precision recall f1-score

0.95

0.94

0.94

0.94

precision recall f1-score

0.92

0.90

0.91

0.91

Tuned SVC Parameter: {'C': 0.7, 'kernel': 'linear'}

**K Nearest Neighbors Classifier** Explanation of K Nearest Neighbors can be found in the link below. • K Nearest Neighbors Classifier

0

1

accuracy

macro avg

weighted avg

Tuned kNN Parameter: {'algorithm': 'auto', 'n\_neighbors': 3} Best score is 0.9072321212609852

Accuracy is 0	93401015228	4264			
	precision	recall	f1-score	support	
0	0.97	0.91	0.94	108	
1	0.90	0.97	0.93	89	
accuracy			0.93	197	
macro avg	0.93	0.94	0.93	197	
weighted avg	0.94	0.93	0.93	197	
6. Conclusion					
The best model here is th	ne Logistic Regression I	Model. The Logis	tic Regression Mode	el has the highest be	st score as well as the highest

changed because of the random sampling. There was difference in results, but Logistic Regression and SVC always had the best results out of all the models. 7. Final Thoughts We preformed undersampling on the data because the data was imbalanced. Undersampling is a technique that people use to balance uneven datasets by keeping all of the data in the minority class and decreasing the size of the majority class. We then starndarized that

accuracy out of all of the model. Support Vector Classifier comes in at a close second. I reran the code many times to see if the results

data used split the data into training set and testing set. We modeled various classifiers and tuned it using GridSearch and found that Logistic Regression with the best parameters is the best model because because it has best scores when you compare the precision, recall, accuracy. This project purpose was to teach myself how to use classification models and I would say that it was a success. I gain more confidence in my data science skills.