Machine Learning Nanodegree Capstone Project

Credit Card Fraud Detection using Supervised learning

Proposed by : Ahmed Roshdy

I. Definition

Project overview:

In this project I will build a credit card fraud detection solution for the credit card companies which shall be able to recognize fraudulent credit cards transactions .

I downloaded the dataset from kaggle from here : https://www.kaggle.com/mlg-ulb/creditcardfraud

The datasets 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 Amount, this feature can be used for example-dependant cost-senstive learning. Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.

I will start with the simple model architecture first before training and evaluating it. Then splitting the data , fit the model , and test the model by using predict function

Problem Statement :

The main objective of this project is to detect the fraud credit cards ,Credit card companies shall be able to recognize fraudulent credit card transactions so that customers are not charged for items that they did not purchase

I will visualize the data to see it , and build correlation to check how strong linear relationship.

I will use supervised learning algorithm , because we need to predict a feature which is labeled with 0 or 1 , and I will divide the data into training data and testing data ,and I will use logistic regression model for this problem , and for refinement I will use GridSearchCV model and GradientBoostingRegressor for improving the result and these helped me to improve the results and I used more paramaters for the GridSearchCV

Metrics:

The evaluation metric for this problem is simply precision, recall , and, f1-score

I use F1 score because it is a classifier metric which calculates a mean of precision and recall in a way that emphasizes the lowest value.

F1 is an overall measure of a model's accuracy that combines precision and recall, in that weird way that addition and multiplication just mix two ingredients to make a separate dish altogether. That is, a good F1 score means that you have low false positives and low false negatives, so you're correctly identifying real threats and you are not disturbed by false alarms. An F1 score is considered perfect when it's `1`, while the model is a total failure when it's `0`.

Precision helps when the costs of false positives are high. So let's assume the problem involves the detection of skin cancer. If we have a model that has very low precision, then many patients will be told that they have melanoma, and that will include some misdiagnoses. Lots of extra tests and stress are at stake. When false positives are too high, those who monitor the results will learn to ignore them after being bombarded with false alarms.

Recall helps when the cost of false negatives is high. What if we need to detect incoming nuclear missiles? A false negative has devastating consequences. Get it wrong and we all die. When false negatives are frequent, you get hit by the thing you want to avoid. A false negative is when you decide to ignore the sound of a twig breaking in a dark forest, and you get eaten by a bear. (A false positive is staying up all night sleepless in your tent in a cold sweat listening to every shuffle in the forest, only to realize the next morning that those sounds were made by a chipmunk. Not fun.) If you had a model that let in nuclear missiles by mistake, you would want to throw it out. If you had a model that kept you awake all night because *chipmunks*, you would want to throw it out, too. If, like most people, you prefer to not get eaten by the bear, and also not stay up all night worried about chipmunk alarms, then you need to optimize for an evaluation metric that's a combined measure of precision and recall. Enter the F1 score…

And all of that applied on our case which is going to help us to test the accuracy of our model and help us also to detect the fraud detection credit cards easily , and The math formulas :

$$Recall = \frac{truepositives}{truepositives + falsenegatives} \qquad Precision = \frac{truepositives}{truepositives + falsepositives}$$

$$\textbf{F1 score} = \left(\frac{recall^{-1} + recall^{-1}}{2} \right)^{-1} = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

II. Analysis

Data Exploration:

The dataset used in this project is produced by kaggle website and available to download from https://www.kaggle.com/mlg-ulb/creditcardfraud

the dataset is labeled and consisting of 284807 rows and 31 columns , and our target feature that we should predict is the Class column , and if the value of the class is 0 that means it's transaction without fraud , and if it is equal 1 that means it's transaction with fraud

statistical data of the dataset :

	Time	V1	V2	V3	V4	\
count	284807.000000	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	
mean	94813.859575	3.919560e-15	5.688174e-16	-8.769071e-15	2.782312e-15	
std	47488.145955	1.958696e+00	1.651309e+00	1.516255e+00	1.415869e+00	
min	0.000000	-5.640751e+01	-7.271573e+01	-4.832559e+01	-5.683171e+00	
25%	54201.500000	-9.203734e-01	-5.985499e-01	-8.903648e-01	-8.486401e-01	
50%	84692.000000	1.810880e-02	6.548556e-02	1.798463e-01	-1.984653e-02	
75%	139320.500000	1.315642e+00	8.037239e-01	1.027196e+00	7.433413e-01	
max	172792.000000	2.454930e+00	2.205773e+01	9.382558e+00	1.687534e+01	

	V5	V6	V7	V8	V9	\
count	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	
mean	-1.552563e-15	2.010663e-15	-1.694249e-15	-1.927028e-16	-3.137024e-15	
std	1.380247e+00	1.332271e+00	1.237094e+00	1.194353e+00	1.098632e+00	
min	-1.137433e+02	-2.616051e+01	-4.355724e+01	-7.321672e+01	-1.343407e+01	
25%	-6.915971e-01	-7.682956e-01	-5.540759e-01	-2.086297e-01	-6.430976e-01	
50%	-5.433583e-02	-2.741871e-01	4.010308e-02	2.235804e-02	-5.142873e-02	
75%	6.119264e-01	3.985649e-01	5.704361e-01	3.273459e-01	5.971390e-01	
max	3.480167e+01	7.330163e+01	1.205895e+02	2.000721e+01	1.559499e+01	

```
       ...           V21           V22           V23           V24  \
count  ...  2.848070e+05  2.848070e+05  2.848070e+05  2.848070e+05
mean   ...  1.537294e-16  7.959909e-16  5.367590e-16  4.458112e-15
std    ...  7.345240e-01  7.257016e-01  6.244603e-01  6.056471e-01
min    ... -3.483038e+01 -1.093314e+01 -4.480774e+01 -2.836627e+00
25%    ... -2.283949e-01 -5.423504e-01 -1.618463e-01 -3.545861e-01
50%    ... -2.945017e-02  6.781943e-03 -1.119293e-02  4.097606e-02
75%    ...  1.863772e-01  5.285536e-01  1.476421e-01  4.395266e-01
max    ...  2.720284e+01  1.050309e+01  2.252841e+01  4.584549e+00

                V25           V26           V27           V28         Amount  \
count  2.848070e+05  2.848070e+05  2.848070e+05  2.848070e+05  284807.000000
mean   1.453003e-15  1.699104e-15 -3.660161e-16 -1.206049e-16      88.349619
std    5.212781e-01  4.822270e-01  4.036325e-01  3.300833e-01     250.120109
min   -1.029540e+01 -2.604551e+00 -2.256568e+01 -1.543008e+01       0.000000
25%   -3.171451e-01 -3.269839e-01 -7.083953e-02 -5.295979e-02       5.600000
50%    1.659350e-02 -5.213911e-02  1.342146e-03  1.124383e-02      22.000000
75%    3.507156e-01  2.409522e-01  9.104512e-02  7.827995e-02      77.165000
max    7.519589e+00  3.517346e+00  3.161220e+01  3.384781e+01   25691.160000

              Class
count  284807.000000
mean        0.001727
std         0.041527
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         1.000000

[8 rows x 31 columns]
```

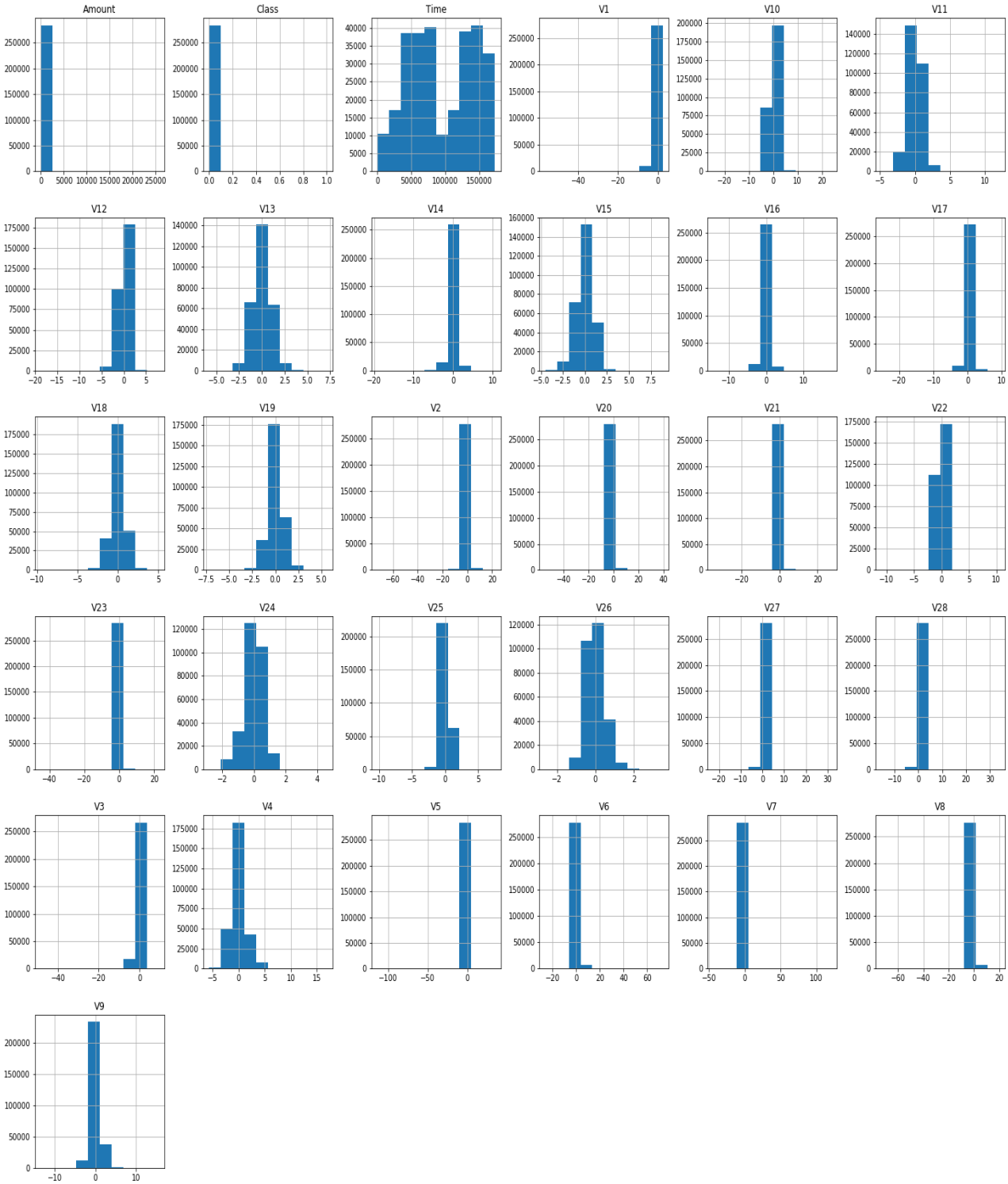
Some data samples :

Time	V1	V2	.....	Amount	Class
0	0.0	-1.359807		149.62	0
1	0.0	1.191857		2.69	0
2	1.0	-1.358354		378.66	0
3	1.0	-0.966272		123.50	0
4	2.0	-1.158233		69.99	0

That’s sample of the column and sample of the rows

Exploratory Visualization:

our target feature that we should predict is the Class column


Amount
Class
Time
V1
V10
V11
V12
V13
V14
V15
V16
V17
V18
V19
V2
V20
V21
V22
V23
V24
V25
V26
V27
V28
V3
V4
V5
V6
V7
V8
V9

Solution statement, algorithms and technique:

The proposed solution to this problem is to apply supervised learning algorithm to detect if the new transaction is fraud or not , and we will use the logistic regression model , after a lot of trying I found that the logistic regression model give the best result.

I'm using logistic regression algorithm because Logistic regression focuses on estimating the probability of an event occurring based on the previous data provided. It is used to cover a binary dependent variable that is where only two values, 0 and 1, represent outcomes.

There are many other algorithms like SVM but I use Logistic regression, SVM Support Vector Machine algorithms are supervised learning models that analyze data used for classification and regression analysis. They essentially filter data into categories, which is achieved by providing a set of training examples, each set marked as belonging to one or the other of the two categories. The algorithm then works to build a model that assigns new values to one category or the other.

Benchmark Model :

For the benchmark model, we will use logistic regression model.

And The 0 class (transactions without fraud) is predicted with 100% precision and recall whereas the 1 class (transactions

which are fraudulent) has 90% precision. This means that 10% of the transactions which are fraudulent remain undetected by the system. This can be further improved by providing more training data.

III. Methodology

Data Preprocessing

First I analyzed the data and visualized it.

Build a Correlation matrix to see if there is any strong Correlation between different variables in our dataset and to see how strong our linear relationships

We divided the data into a training set and validation set with an 80% training and 20% testing , by using train_test_split function

use Logistic Regression model to

train the model and test it

Implementation

First we build a correlation matrix to see if there is any strong correlation between different variables in our dataset and to see how strong our linear relationships.

The first-pass solution

I will start with the simple model architecture first before, training and evaluating it. I split the data into training data and testing data by using train_test_split function from sklearn.model_selection

import LogisticRegression from sklearn.linear_model

get an object from LogisticRegression

Fit the model and pass the training data to fit() function

using predict function to test the model and pass the test data to it

import classification_report from sklearn.metrics to show the results using classification_report function to check the precision , recall , and f1-score :

	precision	recall	f1-score	support
0	1.00	1.00	1.00	56859
1	0.90	0.68	0.77	103
accuracy			1.00	56962
macro avg	0.95	0.84	0.89	56962
weighted avg	1.00	1.00	1.00	56962

Refinement :

For refinement I used GridSearchCV model and GradientBoostingRegressor for improving the result and these helped me to improve the results and I used more paramaters for the GridSearchCV to get the best_estimator_ , best_score_ , and best_params_ that can be using to get the best results ,

- And The best score across ALL searched params is: 0.9999618264982186

- The best parameters across ALL searched params: {'learning_rate': 0.03, 'max_depth': 6, 'n_estimators': 1000, 'subsample': 0.2}

IV. Results

Model Evaluation and Validation

I will start with the simple model architecture first, before training and evaluating it. Then splitting the data , fit the model , and test the model by using predict function,

At the end I used the grid search model to test the acuuracy and I got the best paramaters that I can use to get the best result ,

It was very useful for me to make refinment , and to evaluate my model and know if I got the best result or not.

The final model reasonable and aligning with solution expectations

The final parameters of the model appropriate

The final model been tested with various inputs to evaluate whether the model generalizes well to unseen data

Justification

Let me tell you that the final results found stronger than the benchmark result reported earlier and as I analyzed the result and the grid search cv model in last section , it was amazing , and it gave me the best result , and it was the best solution for solving this problem.

And It's a good job !!!
The 0 class (transactions without fraud) is predicted with 100% precision and recall whereas the 1 class (transactions which are fraudulent) has 90% precision. This means that 10% of the transactions which are fraudulent remain undetected by the system. This can be further improved by providing more training data.

And The final results are compared to the benchmark result or threshold with some type of statistical analysis. Justification is made as to whether the final model and solution is significant enough to have adequately solved the problem.

V. Conclusion

Free-Form Visualization

This is the result from the last solution which is :

Results from Grid Search
==========
======

The best estimator across ALL searched params:
GradientBoostingRegressor(alpha=0.9, criterion='friedman_mse',
init=None,
learning_rate=0.03, loss='ls', max_depth=6,
max_features=None, max_leaf_nodes=None,
min_impurity_decrease=0.0,
min_impurity_split=None,
min_samples_leaf=1, min_samples_split=2,
min_weight_fraction_leaf=0.0,
n_estimators=1000,
n_iter_no_change=None, presort='auto',
random_state=None, subsample=0.2, tol=0.0001,
validation_fraction=0.1, verbose=0,
warm_start=False)

The best score across ALL searched params:
0.9999618264982186

The best parameters across ALL searched params:
{'learning_rate': 0.03, 'max_depth': 6, 'n_estimators': 1000,
'subsample': 0.2}

And the result from logistic regression model is :

	precision	recall	f1-score	support
0	1.00	1.00	1.00	56859
1	0.90	0.68	0.77	103
accuracy			1.00	56962
macro avg	0.95	0.84	0.89	56962
weighted avg	1.00	1.00	1.00	56962

and this mean It's a good job !!!
The 0 class (transactions without fraud) is predicted with 100% precision and recall whereas the 1 class (transactions which are fraudulent) has 90% precision. This means that 10% of the transactions which are fraudulent remain undetected by the system. This can be further improved by providing more training data.

A visualization has been provided that emphasizes an important quality about the project with a thorough discussion. Visual cues are clearly defined.

And that’s sample of the dataset :

Time	V1	V2	……	Amount	Class
0	0.0	-1.359807		149.62	0
1	0.0	1.191857		2.69	0
2	1.0	-1.358354		378.66	0
3	1.0	-0.966272		123.50	0
4	2.0	-1.158233		69.99	0

That’s sample of the column and sample of the rows

Reflection

You can check the earlier you will notify that I summarized the entire process that I used for this project

I think there weren’t any interesting aspects of the project

I think also that there weren’t any difficult aspects of the project

The final model and solution fit my expectations for the problem

Really I have learned a lot from this project when I started to look for the dataset, the idea of the project that I need to work on

I searched for the best algorithm that I need to work on to get the best result and I found that the best is Supervised learning , and I tried more models to get a better on to use

It was interesting for me really , but I got tired for doing that job , and the biggest challenge for this is this documentation that I write because I don’t like writing at all , I tried a lot to do everything correct

Improvement

I just think the further improvements that if we just increase the training data in the future, we will get better results.

Of course there are many algorithms or techniques I researched that I did not know how to implement, but I would consider using if you knew how.