1. **Train a classifier to predict if a transaction is a fraudulent transaction or not?**

Here, I would like to give the brief description of approaches I followed to train the classifier; I will submit the code with respective approaches.

* + 1. After performing EDA (Exploratory Data Analysis) to credit card fraudulent data, it is found that data is imbalanced, and there is not much correlation (>0.4) among the features. There are different ways to handle the imbalance, such as Oversampling, Under sampling.
    2. After performing Oversampling on train data, I found that, there is a good balance between Precision, Recall and F1-score as in binary classification the metric which is important than accuracy is Precision, recall and F1-score. However, under sampling have much lesser precision and F1-score when compared to Oversampling method.

The below classification report describes the precision, recall and f1-score for both oversampling and under sampling.

Table

Description automatically generated

**Classification report of Under sampling method**

**Table

Description automatically generated**

**Classification report of Oversampling method**

* + 1. The other approaches used are PCA (Principal Component Analysis) to reduce the dimensions of the dataset as the current dataset is having 30 features, and Hyper parameter tuning using GridSearchCV and retraining the model based upon feature importance’s threshold using SelectFromModel library, but none of the methods is performing good enough i.e., precision, recall scores are as low as 0.75.

So far, from the approaches, the best approach in which the model is performing better was Oversampling i.e., SMOTE with Xgboost Classifier.

1. **How would you deploy this model in production? (Explain the approach) and How to call the Model?**

Here for deploying to an API and to a cloud platform i.e., in Production I have chosen the following,

* + 1. Flask
    2. Google Cloud Platform (GCP)

With flask, there are two different ways to receive the features for prediction:

* + Features sent through JSON object – either Postman or requests module in Python.
  + Features sent through web form.

For the current problem definition, Flask API is created to receive the features in JSON format and used Postman to visualize the prediction of model using .html page.

API response is handled with try-exception blocks

Created flask API endpoint for model prediction is deployed to GCP and the model can be called from Postman by sending JSON object consisting of features (in Body of Postman) with the following link.

<https://shaped-glazing-321018.nn.r.appspot.com/predict_api/>

**Screenshots:**

After deploying into GCP, and calling from the above link on Postman for the features:

{"Time": 45014.000000, "V1":-2.004881, "V2":-1.127553, "V3":0.563607,"V4": 0.143416,"V5": 1.358909,"V6": -0.983896,"V7": -0.755049,"V8": 0.658284,"V9": -0.624140,"V10": -0.538880,"V11": 0.797106,"V12": 0.736617,"V13": -0.365795,"V14": 0.828498,"V15": -0.058298,"V16": -0.568928,"V17": 0.471337,"V18": -0.205489,"V19": 1.223548,"V20": 0.471835,"V21": 0.286698,"V22": 0.149270,"V23": -0.209613,"V24": -0.187195,"V25": -0.250038,"V26": 1.172414,"V27": -0.044708,"V28": -0.272800,"Amount": 2.370000}

Graphical user interface, text, application, email

Description automatically generated

Test from localhost:

Graphical user interface, text, application, email

Description automatically generated