Team Name: Deep Thinkers

Team Member: Gaurav Kumar Daharia

Theme of Hackathon: Real time Payment

Problem Statement:

It is a good that now day's people are supporting online payments and at the same time many third party are ensuring there transaction is safe and secure. But problem arises when these parties don't make any autonomous changes which can detect the anomaly in the transaction because as per theme real time payment give immediate access to beneficiary, so from here we can say that beneficiary might be an unauthorized user. So in order to prevent such fraud type of problem I have developed something which can be integrated with real time scenario and it might be possible that this problem can be solved.

Description of the desired application:

I have developed a deep learning model which can classify easily about any transaction whether a particular transaction is a genuine or fraud transaction. I know It's pretty simple to say it's a common type of problem solution but I would like say one thing i.e. most of the fraud detectors are using clustering algorithm for detecting anomaly in transaction.

But my provided solution in completely based on classification and I have used deep neural network which is much more efficient and robust than any clustering algorithm. Neural network are likely to perform well in real life scenario and they are adaptable too, so these are the only reasons for using **deep neural network** for classification purpose.

My provided solution may have some competitor like clustering algorithm trained model are good in unlabelled dataset but my intuition behind using neural network is because of it robustness and they less prone to noise because clustering algorithm may show wrong result due to noise in dataset but a neural network not because it can adapt itself from any changes.

Solutioning and Methodology:

1. **How does it help to solve the problem?**
   As per theme I have created a deep learning model which classifies whether a particular transaction is a fraud transaction or not. Now question here is how it can help to solve this particular problem, so let us suppose that a person is in party or somewhere else but not performing any transaction and that person is not knowing that there is a background transaction is going behind him/her, so by taking this kind of situation in mind I have prepared a model which will look forward for such anonymous transaction by having look over the person location and time stamp of person as well as time stamp of transaction, so by knowing these certain values my model will generate an alert and will send message to bank as well as that person to block his/her card.

2. **What are the impact metrics that one can use to analyse the effect of the solution?**
   There are various metric to analyse the effect of my provided solution:
   - Adaptive nature of neural network.
   - Less prone to external noise.
   - Robust to classification.

   The only reason of choosing this particular technology because it is easy to integrate with any technology, it does not require high specific hardware, just only GPU is needed for its better functioning.

3. **How easily can your solution be implemented and how effective will it be?**
   My provided solution can be integrated in backend with any technology like web or any cloud platform. This particular model in platform independent and it only require data to learn and predict the output.
   The use case this particular solution is very vast it can used with any technology in backend and it will not only provide benefit to end user but also to any big industry whose major focus is in handling or monitoring transaction.

4. **Who all can leverage your solution?**
   This provided solution will be useful to everyone i.e. from a normal user to any organisation. This particular deep learning model so robust that it can be used by anyone in this whole world. Just require a GPU support in backend for better results. It easy to integrate it with real time data and rest my model will do its job of its own.

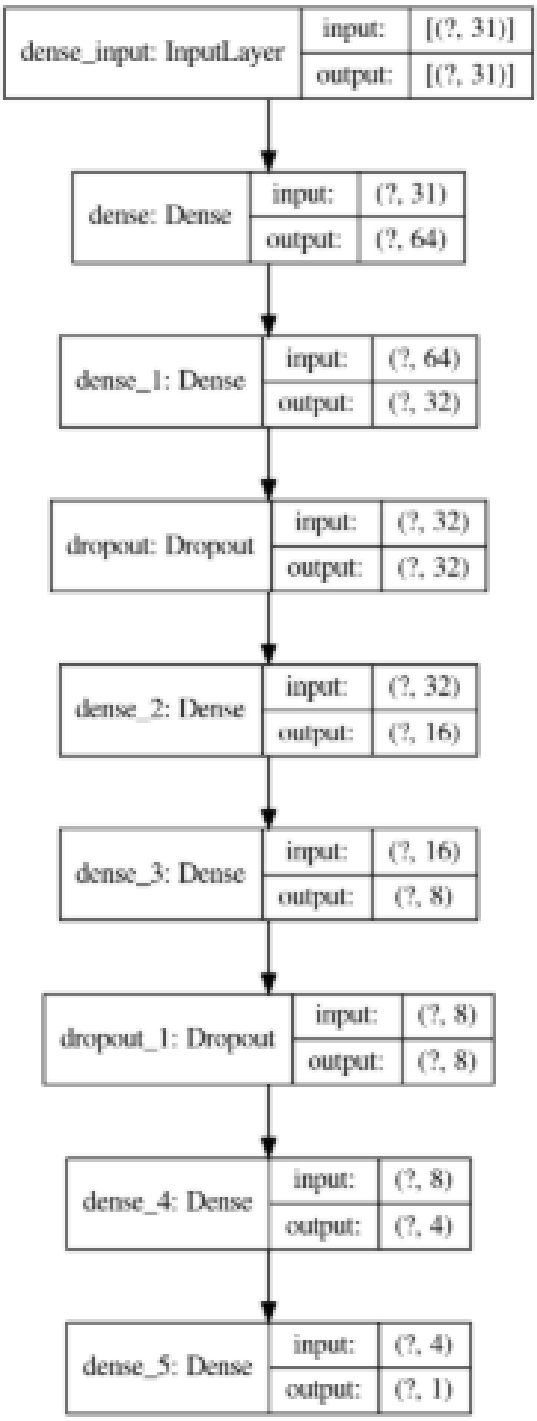
5. **Tech Stack Used:**
   - **Keras Framework: -** This is a deep learning framework which I have used to develop a deep neural network.
   - **Matplotlib: -** I have used Matplotlib for visualization purpose.
   - **Pandas: -** It is well known python library for creating data frames.
   - **SMOTE: -** Used for balancing the unbalanced classes of dataset.

   I have used these particular tools for developing my deep learning model for classification purpose.

   - Matplotlib had helped to visualize the unbalanced class present in dataset.
   - Pandas had helped me for creating data frames for model.
   - Smote library had helped me to balance my unbalanced class of dataset.
   - Keras is well known framework for deep learning and using this particular framework I had created a deep neural network for classification purpose.

Visualisations

1. Neural Network Architecture

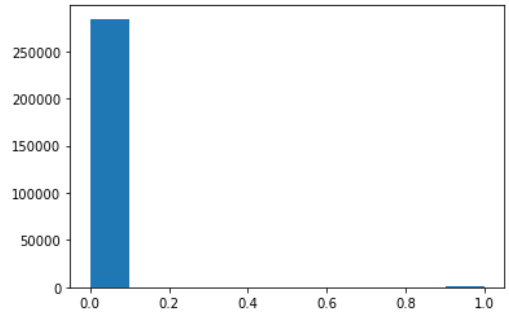


I have used five layer dense neural network for developing my classification model. I have also used Dropout layer in between these network which controls and regulates my model to train more effectively. This dropout layer help my model to learn different ways to extract values from given data points and classify it to its belonging class.

2. **Unbalance Class plot:**

Unbalance class plot

```
plt.hist(Y)
plt.show()
```



This is the plot of unbalance class distribution here:

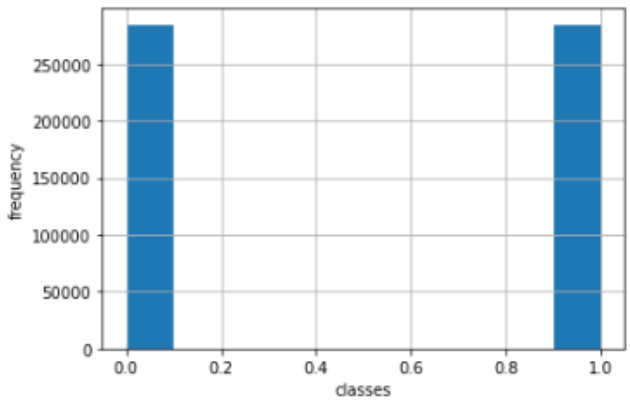
- Zero: - genuine transaction class
- One: - Fraud transaction class

This was the major problem and in order to overcome with this problem I have resampled the dataset.

3. **Resampled Class plot:**

After Resampling the data we have effectively balanced our target classes

```
plt.hist(resampled_class)
plt.grid()
plt.xlabel('classes')
plt.ylabel('frequency')
plt.show()
```



Now after resampling my class distribution looks like this.

4. Classification Report

This report shows everything about performance of model and it had used some tem like precision and recall, below I have mentioned there definition.

- **Precision:** precision (also called positive predictive value) is the fraction of relevant instances among the retrieved instances.
- **Recall:** while recall (also known as sensitivity) is the fraction of the total amount of relevant instances that were actually retrieved.

```
print(classification_report(ytest,prediction))
```

```
              precision    recall  f1-score   support

           0       0.90      0.99      0.95     56724
           1       0.99      0.89      0.94     57002

    accuracy                           0.94    113726
   macro avg       0.95      0.94      0.94    113726
weighted avg       0.95      0.94      0.94    113726
```

This particular report shows that model is 94% accurate in overall classification and it is 90 % sure that a particular data point belongs of class zero and 99% sure that a particular data point belongs to class one.