In[]:

```
## Name : Tanmay Gujar
##Rollno:I4159
##Subject:LP-IV(DL)
```

In[1]:

```
import pandas as
pdimport numpy as
npimporttensorflowastf
importmatplotlib.pyplotasplt
importseabornassns
fromsklearn.model_selectionimporttrain_test_split

fromsklearn.preprocessingimportStandardScaler
fromsklearn.metricsimportconfusion_matrix,recall_score,accuracy_score,precision_score

RANDOM_SEED=2021
TEST_PCT=0.3
LABELS=["Normal","Fraud"]
```

In[2]:

```
dataset=pd.read_csv("creditcard.csv")
```

In[3]:

```
#checkforanynullvalues
print("Any nulls in the
dataset",dataset.isnull().values.any())print('
------------ ')
print("No. of unique labels",len( dataset[' Class'].
unique()))print("Labelvalues",dataset.Class.unique())

#0isfornormalcreditcardtranscation
#1isforfraudulentcreditcardtranscation
print('----- ')
print("BreakdownofNormalandFraudTranscations")print(pd.value_
counts(dataset['Class'],sort=True))
```

```
AnynullsinthedatasetFalseNo.o
............
funiquelabels2
Labelvalues[01]
............
BreakdownofNormalandFraudTranscations0
    284315
1       492
Name:Class,dtype:int64
```

In[4]:

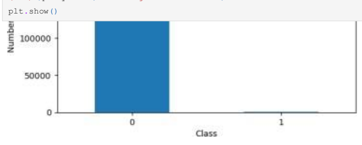
```
#visualizing theimbalanceddataset
count_classes =
pd.value_counts(dataset['Class'],sort=True)count_classes.plot(kind='bar
',rot=0)plt.xticks(range(len(dataset['Class'].unique())),dataset.Class.unique()
)plt.title("Frequencybyobservationnumber")
plt.xlabel("Class")plt.ylabel("Numb
erofObservations")
```

Out[4]:Text(0,0.5,'NumberofObservations')

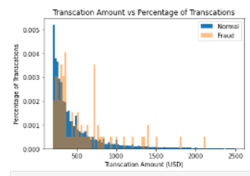
```
#Savethenormal andfradulenttranscations inseperate dataframe
normal_dataset = dataset[dataset.Class ==
0]fraud_dataset=dataset[dataset.Class==1]

#Visualize transcationamountsfor normal andfradulenttranscations
bins    =      np.linspace(200,2500,100)plt.hist(normal_dataset.Amount,bins=
bins,alpha=1,density=True ,label='
Normal')plt.hist(fraud_dataset.Amount,bins=bins,alpha =0.5,density=True,label=
'Fraud')plt.legend(loc='upperright')
plt.title("Transcation Amount vs Percentage of
Transcations")plt.xlabel("Transcation    Amount
(USD)")plt.ylabel("PercentageofTranscations")

plt.show()
```



In[4]:



In[5]:

```
dataset
```

Out[5]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V24	V25	V26	V27	V28	Amount	Class
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307	0.277838	-0.110474	0.066928	0.128539	-0.189115	0.133558	-0.021053	149.62	0
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775	-0.638672	0.101288	-0.339846	0.167170	0.125895	-0.008983	0.014724	2.69	0
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.247998	0.771679	0.909412	-0.689281	-0.327642	-0.139097	-0.055353	-0.059752	378.66	0
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.108300	0.005274	-0.190321	-1.175575	0.647376	-0.221929	0.062723	0.061458	123.50	0
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	...	-0.009431	0.798278	-0.137458	0.141267	-0.206010	0.502292	0.219422	0.215153	69.99	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
284802	172786.0	-11.881118	10.071785	-9.834783	-2.066656	-5.364473	-2.606837	-4.918215	7.305334	1.914428	...	0.213454	0.111864	1.014480	-0.509348	1.436807	0.250034	0.943651	0.823731	0.77	0
284803	172787.0	-0.732789	-0.055080	2.035030	-0.738589	0.868229	1.058415	0.024330	0.294869	0.584800	...	0.214205	0.924384	0.012463	-1.016226	-0.606624	-0.395255	0.068472	-0.053527	24.79	0
284804	172788.0	1.919565	-0.301254	-3.249640	-0.557828	2.630515	3.031260	-0.296827	0.708417	0.432454	...	0.232045	0.578229	-0.037501	0.640134	0.265745	-0.087371	0.004455	-0.026561	67.88	0
284805	172788.0	-0.240440	0.530483	0.702510	0.689799	-0.377961	0.623708	-0.686180	0.679145	0.392087	...	0.265245	0.800049	-0.163298	0.123205	-0.569159	0.546668	0.108821	0.104533	10.00	0
284806	172792.0	-0.533413	-0.189733	0.703337	-0.506271	-0.012546	-0.649617	1.577006	-0.414650	0.486180	...	0.261057	0.643078	0.376777	0.008797	-0.473649	-0.818267	-0.002415	0.013649	217.00	0

284807rows×31columns

In[6]:

```
sc=StandardScaler()
dataset['Time'] = sc.fit_transform(dataset['Time'].values.reshape(-
1,1))dataset['Amount']=sc.fit_transform(dataset['Amount'].values.reshape(-1,1))
```

In[7]:

```
raw_data=dataset.values
#Thelastelementcontainsifthetranscationisnormalwhichisrepresentedby0andiffraudthen1
labels=raw_data[:,-1]

#Theotherdatapointsaretheelectrocadriogramdata
data=raw_data[:,0:-1]

train_data,test_data,train_labels,test_labels=train_test_split(data,labels,test_size=0.2,random_state=2021)
```

In[8]:

```
min_val=tf.reduce_min(train_data)max_v
al=tf.reduce_max(train_data)

train_data=(train_data-min_val)/(max_val-
min_val)test_data=(test_data-min_val)/(max_val-min_val)

train_data = tf.cast(train_data,tf .float32
)test_data=tf.cast(test_data,tf.float32)
```

In[9]:

```
train_labels=train_labels.astype(bool)test_lab
els=test_labels.astype(bool)

#Creating normal and fraud
datasetsnormal_train_data=train_data[~train_la
bels]normal_test_data=test_data[~test_label
s]

fraud_train_data                          =
train_data[train_labels]fraud_test_data=tes
t_data[test_labels]
print("No.      of      records      in      Fraud      Train
Data=",len(fraud_train_data))print("No. of records in Normal Train
Data=",len(normal_train_data))print("No.ofrecordsinFraudTestData=",
```

```
No. of records in Fraud Train Data=
389No.ofrecordsinNormalTrainData=227456No
.ofrecordsinFraudTestData=103
No.ofrecordsinNormalTestData=56859
```

In[10]:

```
nb_epoch=50
batch_size=64
input_dim=normal_train_data.shape[1]
#numofcolumns,30
encoding_dim=14
hidden_dim1=int(encoding_dim/2)hid
den_dim2=4
learning_rate=1e-7
```

In[11]:

```
#inputlayer
input_layer=tf.keras.layers.Input(shape=(input_dim,))

#Encoder
encoder=tf.keras.layers.Dense(encoding_dim,activation= "tanh",activity_regularizer=tf.keras.regularizers.
l2(learning_rate))(input_layer)encoder=tf.keras.layers.Dropout(0.2)(encoder)
encoder=tf.keras.layers.Dense(hidden_dim1,activation='relu')(encoder)
encoder=tf.keras.layers.Dense(hidden_dim2,activation=tf.nn.leaky_relu)(encoder)

#Decoder
decoder=tf.keras .layers.Dense(hidden_dim1
,activation='relu')(encoder)decoder=tf.keras.layers.Dropout(0.2)(decoder)
decoder=tf.keras.layers.Dense(encoding_dim,
activation='relu')(decoder)decoder=tf.keras.layers.Dense(input_dim,activatio
n='tanh')(decoder)

#Autoencoder
autoencoder=tf.keras.Model(inputs=input_layer,outputs=decoder)autoencoder.
summary()
```

```
Model:"model"
Layer(type)                 OutputShape              Param#
=================================================================
 input_1(InputLayer)         [(None,30)]              0
 dense(Dense)                (None,14)                434
 dropout(Dropout)            (None,14)                0
 dense_1(Dense)              (None,7)                 105
 dense_2(Dense)              (None,4)                 32
 dense_3(Dense)              (None,7)                 35
 dropout_1(Dropout)          (None,7)                 0
 dense_4(Dense)              (None,14)                112
 dense_5(Dense)              (None,30)                450

=================================================================
Totalparams:1,168
Trainableparams:1,168
Non-trainableparams:0
```

In[12]:

```
cp=tf.keras.callbacks.ModelCheckpoint(filepath="autoencoder_fraud.h5",mode='min',monitor='val_loss',verbose=2,save_best_only=True)
#Defineourearlystopping
early_stop=tf.keras.callbacks.EarlyStopping(
                monitor='val_loss',
                min_delta=0.0001,p
                atience=10,verbose
                =11,mode='min',
                restore_best_weights=True
)
```

In[13]:

```
autoencoder.compile(metrics=['accuracy'],loss='mean_squared_error',optimizer='adam')
```

In[14]:

```
history=autoencoder.fit(normal_train_data,normal_train_data,epochs=nb_epoch,
                        batch_size=batch_size,shuffle=True,vali
                        dation_data =
                        (test_data,test_data),verbose=1,
                        callbacks=[cp,early_stop]).history
```

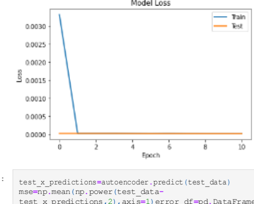
```
Epoch1/50
3543/3554[==========================>.]-ETA:0s-loss:0.0033-accuracy:0.0372
Epoch1:val_lossimprovedfrominfto0.00002,savingmodeltoautoencoder_fraud.h5
3554/3554[==============================] -11s2ms/step-loss:0.0033-accuracy:0.0372-val_loss:2.0179e-05-val_accuracy:0.0343
Epoch2/50
3545/3554[==========================>.]-ETA:0s-loss:1.9583e-05-accuracy:0.0609
Epoch2:val_lossdidnotimprovefrom0.00002
3554/3554[==============================] -6s2ms/step-loss:1.9599e-05-accuracy:0.0609-val_loss:2.0190e-05-val_accuracy:0.0078
Epoch3/50
3517/3554[==========================>.]-ETA:0s-loss:1.9562e-05-accuracy:0.0619
Epoch3:val_lossimprovedfrom0.00002to0.00002,savingmodeltoautoencoder_fraud.h5
3554/3554[==============================] -7s2ms/step-loss:1.9580e-05-accuracy:0.0619-val_loss:2.0025e-05-val_accuracy:0.0420
Epoch4/50
3527/3554[==========================>.]-ETA:0s-loss:1.9545e-05-accuracy:0.0599
Epoch4:val_lossdidnotimprovefrom0.00002
3554/3554[==============================] -6s2ms/step-loss:1.9530e-05-accuracy:0.0601-val_loss:2.0277e-05-val_accuracy:0.2168
Epoch5/50
3549/3554[==========================>.]-ETA:0s-loss:1.8826e-05-accuracy:0.1758
Epoch5:val_lossimprovedfrom0.00002to0.00002,savingmodeltoautoencoder_fraud.h5
3554/3554[==============================] -6s2ms/step-loss:1.8831e-05-accuracy:0.1759-val_loss:1.8344e-05-val_accuracy:0.2184
Epoch6/50
3539/3554[==========================>.]-ETA:0s-loss:1.7526e-05-accuracy:0.2362
Epoch6:val_lossimprovedfrom0.00002to0.00002,savingmodeltoautoencoder_fraud.h5
3554/3554[==============================] -8s2ms/step-loss:1.7518e-05-accuracy:0.2363-val_loss:1.7095e-05-val_accuracy:0.3538
Epoch7/50
3516/3554[==========================>.]-ETA:0s-loss:1.8826e-05-accuracy:0.1125
Epoch7:val_lossdidnotimprovefrom0.00002
3554/3554[==============================] -5s2ms/step-loss:1.8813e-05-accuracy:0.1137-val_loss:1.7990e-05-val_accuracy:0.2041
Epoch8/50
3529/3554[==========================>.]-ETA:0s-loss:1.7206e-05-accuracy:0.2141
Epoch8:val_lossdidnotimprovefrom0.00002
3554/3554[==============================] -5s1ms/step-loss:1.7328e-05-accuracy:0.2143-val_loss:1.7153e-05-val_accuracy:0.2781
Epoch9/50
3518/3554[==========================>.]-ETA:0s-loss:1.6837e-05-accuracy:0.2481
Epoch9:val_lossimprovedfrom0.00002to0.00002,savingmodeltoautoencoder_fraud.h5
3554/3554[==============================] -6s2ms/step-loss:1.6825e-05-accuracy:0.2482-val_loss:1.6811e-05-val_accuracy:0.3518
Epoch10/50
3516/3554[==========================>.]-ETA:0s-loss:1.6692e-05-accuracy:0.2509
Epoch10:val_lossimprovedfrom0.00002to0.00002,savingmodeltoautoencoder_fraud.h5
3554/3554[==============================] -6s2ms/step-loss:1.6682e-05-accuracy:0.2510-val_loss:1.6469e-05-val_accuracy:0.3492
Epoch11/50
3526/3554[==========================>.]-ETA:0s-loss:1.6569e-05-accuracy:0.2484
Epoch11:val_lossimprovedfrom0.00002to0.00002,savingmodeltoautoencoder_fraud.h5
Restoringmodelweightsfromtheendofthebestepoch:1.
3554/3554 [==============================] - 7s 2ms/step - loss: 1.6561e-05 - accuracy: 0.2484 - val_loss: 1.6237e-05 - val_accuracy: 0
.2865Epoch11:earlystopping
```

In[15]:

```
plt.plot(history['loss'],linewidth = 2,label =
'Train')plt.plot(history['val_loss'],linewidth=2,label='Test')
plt.legend(loc='upperright')
plt.title('ModelLoss')plt.y
label('Loss')plt.xlabel('
Epoch')

#plt.ylim(ymin=0.70,ymax=1)

plt.show()
```



In[16]:

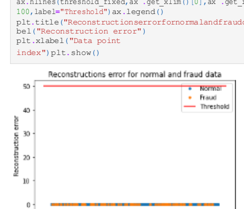
```
test_x_predictions=autoencoder.predict(test_data)
mse=np.mean(np.power(test_data-
test_x_predictions,2),axis=1)error_df=pd.DataFrame({'Reconstruction_erro
r':mse,
                        'True_class':test_labels})
```

```
1781/1781[==============================]-2s865us/step
```

In[17]:

```
threshold_fixed=50
groups=error_df.groupby('True_class')fig,ax=p
lt.subplots()

forname,groupingroups:
        ax.plot(group.index, group.Reconstruction_error,marker
                ='o',ms=3.5,linestyle='',label="Fraud"ifname==1else"Normal")
ax.hlines(threshold_fixed,ax .get_xlim()[0],ax .get_xlim()[1],colors="r",zorder=
100,label="Threshold")ax.legend()
plt.title("Reconstructionserrorfornormalandfrauddata")plt.yla
bel("Reconstruction error")
plt.xlabel("Data point
index")plt.show()
```

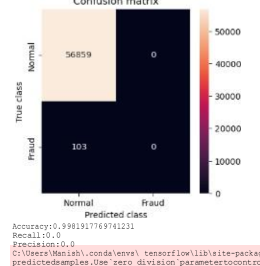


In[19]:

```
threshold_fixed=52
pred_y=[1ife>threshold_fixedelse0
          forein
        error_df.Reconstruction_error.values]e
rror_df['pred']=pred_y
conf_matrix=confusion_matrix(error_df.True_class,pred_y)

plt.figure(figsize=(4,4))
sns.heatmap(conf_matrix,xticklabels=LABELS,yticklabels=LABELS,annot=True,fmt="d")plt.tit
le("Confusionmatrix")
plt.ylabel("Trueclass")
plt.xlabel(" Predicted
class")plt.show()

#PrintAccuracy,PrecisionandRecall
print("Accuracy :", accuracy_score(error_df['
True_class'],error_df['pred']))print("Recall:",recall_score(error_df['True_cla
ss'],error_df['pred']))print("Precision:",precision_score(error_df['True_class'
],error_df['pred']))
```




```
Accuracy:0.9981917769741231
Recall:0.0
Precision:0.0
C:\Users\Manish\.conda\envs\ tensorflow\lib\site-packages\sklearn\metrics\_classification.py:1318: UndefinedMetric Warning: Precision is ill-defined and being set to 0. 0 due to no
predictedsamples.Use`zero_division`parametertocontrolthisbehavior.
  _warn_prf(average,modifier,msg_start,len(result))
```