Name: Nithin D’Souza

Batch ID: 05062021-9AM (weekend)

Topic: Deep learning practical issues

Q. Build an ANN model to predict Delivery Status for a logistic firm. (Refer FedEx dataset)

ann2.ipynb and **ann2.py** files are same but in 2 different extension.

Importing required library

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
import seaborn as sns
import keras
from keras.models import Sequential
from keras.layers import Dense
from keras.wrappers.scikit_learn import KerasClassifier
from sklearn.model_selection import cross_val_score
```

load the dataset (fedex dataset)

```
path = "/content/drive/MyDrive/Colab Notebooks/fedex.csv"
fedex_data = pd.read_csv(path)
```

Copy the path or mount file to Google drive in order to load in Google collab

Check for presence of Nan values

```
fedex_data.isnull().sum().sum()
```

dropping the rows having Nan values and check

```
fedex_data = fedex_data.dropna()
fedex_data.isnull().sum().sum()
```

Viewing top 5 FedEx data

```
fedex_data.head(5)
```

Checking Column Names

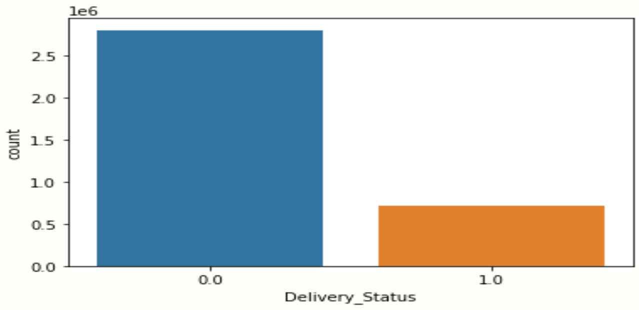
```
fedex_data.columns
```

Viewing target

```
fedex_data['Delivery_Status'].head(5)
```

Summary of Delivery_Status

```
sns.countplot(fedex_data['Delivery_Status'],label="Count")
```



Combining source and destination column into new Path column since source and destination gives path

```
fedex_data['Path'] = fedex_data[['Source', 'Destination']].apply(lambda x: '-'.join(x), axis = 1)
fedex_data['Path'].head(5)
```

Droping 'Source' and 'Destination' along with 'Carrier_Name' since 'Carrier_Num' is already present, 'Carrier_Num' is ID for 'Carrier_Name' so keeping only 'Carrier_Num'

```
fedex_data.drop(['Source', 'Destination', 'Carrier_Name'], axis = 1, inplace= True)
```

Encoding categorical data, Since NN takes only numerical inputs

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
fedex_data['Path'] = le.fit_transform(fedex_data['Path'])
```

Separating Independent and dependent features

```
X  = fedex_data.iloc[:, fedex_data.columns != 'Delivery_Status'].values #independent
y = fedex_data['Delivery_Status'].values   #dependent
```

Viewing Independent and dependent columns

```
print(X[:,:5])
print(y)
```

Splitting the data into train and test with test size as 30%

```
X_train, X_test, y_train, y_test = train_test_split(X, y,test_size = 0.3, random_state = 0)
#Scaling independent variables
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
```

```
X_test = sc.transform(X_test)
```

viewing scaled data before building model

```
X_train[:,:5]
```

Printing Shapes of input along with output values

```
print("X_train",X_train.shape)
print("X_test",X_test.shape)
print("y_train",y_train.shape)
print("y_test",y_test.shape)
```

Defining the model inside a function and
Adding the input and hidden layer followed by output layer also compiling the model

```
def build_model():
  model = Sequential()
  model.add(Dense(10,activation='relu',input_shape=(12,)))
  #adding the second hidden layer
  model.add(Dense(6, activation='relu'))
  #adding the output layer
  model.add(Dense(1, activation='sigmoid'))
  #compiling model
  model.compile(optimizer='adam',loss='binary_crossentropy',metrics=['accuracy'])
  return model
```

Fitting model by using defined model function which as sequential layers.

```
classifier = KerasClassifier(build_fn = build_model , batch_size = 512 , nb_epoch = 20)
```

Evaluating model using cross-validation technique having 10 Folds by using all available concurrent workers

```
accuracy = cross_val_score(estimator = classifier , X=X_train, y=y_train, cv =10 , n_jobs = -1)
```

Printing Mean accuracy of model

```
print("Mean accuracy:",format(100*accuracy.mean(),".2f"),"%")
```