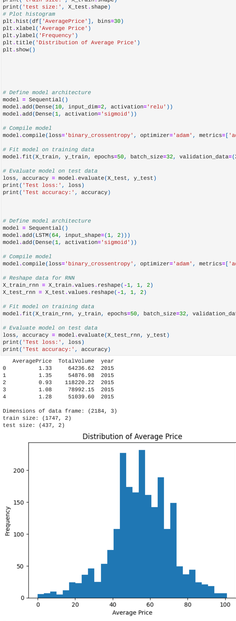
Afaf Guesmia

axg190061

Text Classification 2

In [57]:
```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.neural_network import MLPClassifier
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
from nltk.corpus import stopwords
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
from keras.models import Sequential
from keras.layers import Dense, LSTM, Conv1D, MaxPooling1D, Flatten
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
import numpy as np

import keras
from keras.models import Sequential
from keras.layers import Dense

# Load dataset
df = pd.read_csv('C:/Users/many3/OneDrive/Desktop/Avocado.csv', usecols=['AveragePrice', 'TotalVolume', 'year'],encoding='ISO-8859-1')
print(df.head())
print('\nDimensions of data frame:', df.shape)
df.AveragePrice = df.AveragePrice.astype('category').cat.codes
df.year = df.year.astype('category').cat.codes
df.head()
df.isnull().sum()
X = df.loc[:, ['TotalVolume', 'year']]
y = df.AveragePrice
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
print('train size:', X_train.shape)
print('test size:', X_test.shape)
# Plot histogram
plt.hist(df['AveragePrice'], bins=30)
plt.xlabel('Average Price')
plt.ylabel('Frequency')
plt.title('Distribution of Average Price')
plt.show()




# Define model architecture
model = Sequential()
model.add(Dense(10, input_dim=2, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

# Compile model
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

# Fit model on training data
model.fit(X_train, y_train, epochs=50, batch_size=32, validation_data=(X_test, y_test))

# Evaluate model on test data
loss, accuracy = model.evaluate(X_test, y_test)
print('Test loss:', loss)
print('Test accuracy:', accuracy)



# Define model architecture
model = Sequential()
model.add(LSTM(64, input_shape=(1, 2)))
model.add(Dense(1, activation='sigmoid'))

# Compile model
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

# Reshape data for RNN
X_train_rnn = X_train.values.reshape(-1, 1, 2)
X_test_rnn = X_test.values.reshape(-1, 1, 2)

# Fit model on training data
model.fit(X_train_rnn, y_train, epochs=50, batch_size=32, validation_data=(X_test_rnn, y_test))

# Evaluate model on test data
loss, accuracy = model.evaluate(X_test_rnn, y_test)
print('Test loss:', loss)
print('Test accuracy:', accuracy)
```

```
   AveragePrice  TotalVolume  year
0          1.33     64236.62  2015
1          1.35     54876.98  2015
2          0.93    118220.22  2015
3          1.08     78992.15  2015
4          1.28     51039.60  2015

Dimensions of data frame: (2184, 3)
train size: (1747, 2)
test size: (437, 2)
```




```
Epoch 1/50
55/55 [==============================] - 2s 14ms/step - loss: 5502623.5000 - accuracy: 5.7241e-04 - val_loss: 1302840.8750 - val_accuracy: 0.0000e+00
Epoch 2/50
55/55 [==============================] - 0s 6ms/step - loss: -2047825.6250 - accuracy: 5.7241e-04 - val_loss: -5017301.0000 - val_accuracy: 0.0000e+00
Epoch 3/50
55/55 [==============================] - 0s 7ms/step - loss: -9036272.0000 - accuracy: 5.7241e-04 - val_loss: -10981641.0000 - val_accuracy: 0.0000e+00
Epoch 4/50
55/55 [==============================] - 0s 7ms/step - loss: -15749320.0000 - accuracy: 5.7241e-04 - val_loss: -16650873.0000 - val_accuracy: 0.0000e+00
Epoch 5/50
55/55 [==============================] - 0s 6ms/step - loss: -21903242.0000 - accuracy: 5.7241e-04 - val_loss: -21825400.0000 - val_accuracy: 0.0000e+00
Epoch 6/50
55/55 [==============================] - 0s 6ms/step - loss: -27732736.0000 - accuracy: 5.7241e-04 - val_loss: -26868856.0000 - val_accuracy: 0.0000e+00
Epoch 7/50
55/55 [==============================] - 0s 6ms/step - loss: -33583560.0000 - accuracy: 5.7241e-04 - val_loss: -32178176.0000 - val_accuracy: 0.0000e+00
Epoch 8/50
55/55 [==============================] - 0s 6ms/step - loss: -40093544.0000 - accuracy: 5.7241e-04 - val_loss: -38308452.0000 - val_accuracy: 0.0000e+00
Epoch 9/50
55/55 [==============================] - 0s 6ms/step - loss: -47567328.0000 - accuracy: 5.7241e-04 - val_loss: -45315932.0000 - val_accuracy: 0.0000e+00
Epoch 10/50
55/55 [==============================] - 0s 6ms/step - loss: -56127560.0000 - accuracy: 5.7241e-04 - val_loss: -53403036.0000 - val_accuracy: 0.0000e+00
Epoch 11/50
55/55 [==============================] - 0s 6ms/step - loss: -65976868.0000 - accuracy: 5.7241e-04 - val_loss: -62516952.0000 - val_accuracy: 0.0000e+00
Epoch 12/50
55/55 [==============================] - 0s 7ms/step - loss: -77147912.0000 - accuracy: 5.7241e-04 - val_loss: -72921968.0000 - val_accuracy: 0.0000e+00
Epoch 13/50
55/55 [==============================] - 0s 6ms/step - loss: -89673288.0000 - accuracy: 5.7241e-04 - val_loss: -84543912.0000 - val_accuracy: 0.0000e+00
Epoch 14/50
55/55 [==============================] - 0s 6ms/step - loss: -103540288.0000 - accuracy: 5.7241e-04 - val_loss: -97173104.0000 - val_accuracy: 0.0000e+00
Epoch 15/50
55/55 [==============================] - 0s 6ms/step - loss: -118644664.0000 - accuracy: 5.7241e-04 - val_loss: -111005296.0000 - val_accuracy: 0.0000e+00
Epoch 16/50
55/55 [==============================] - 0s 6ms/step - loss: -135017184.0000 - accuracy: 5.7241e-04 - val_loss: -125852080.0000 - val_accuracy: 0.0000e+00
Epoch 17/50
55/55 [==============================] - 0s 6ms/step - loss: -152590640.0000 - accuracy: 5.7241e-04 - val_loss: -141683680.0000 - val_accuracy: 0.0000e+00
Epoch 18/50
55/55 [==============================] - 0s 6ms/step - loss: -171305408.0000 - accuracy: 5.7241e-04 - val_loss: -158739344.0000 - val_accuracy: 0.0000e+00
Epoch 19/50
55/55 [==============================] - 0s 6ms/step - loss: -191256672.0000 - accuracy: 5.7241e-04 - val_loss: -176439936.0000 - val_accuracy: 0.0000e+00
Epoch 20/50
55/55 [==============================] - 0s 6ms/step - loss: -212159440.0000 - accuracy: 5.7241e-04 - val_loss: -195388640.0000 - val_accuracy: 0.0000e+00
Epoch 21/50
55/55 [==============================] - 0s 6ms/step - loss: -234225728.0000 - accuracy: 5.7241e-04 - val_loss: -215155792.0000 - val_accuracy: 0.0000e+00
Epoch 22/50
55/55 [==============================] - 0s 6ms/step - loss: -257263376.0000 - accuracy: 5.7241e-04 - val_loss: -236928336.0000 - val_accuracy: 0.0000e+00
Epoch 23/50
55/55 [==============================] - 0s 6ms/step - loss: -281373632.0000 - accuracy: 5.7241e-04 - val_loss: -257198560.0000 - val_accuracy: 0.0000e+00
Epoch 24/50
55/55 [==============================] - 0s 6ms/step - loss: -306375488.0000 - accuracy: 5.7241e-04 - val_loss: -279896992.0000 - val_accuracy: 0.0000e+00
Epoch 25/50
55/55 [==============================] - 0s 6ms/step - loss: -332463200.0000 - accuracy: 5.7241e-04 - val_loss: -302089184.0000 - val_accuracy: 0.0000e+00
Epoch 26/50
55/55 [==============================] - 0s 6ms/step - loss: -359256640.0000 - accuracy: 5.7241e-04 - val_loss: -326722720.0000 - val_accuracy: 0.0000e+00
Epoch 27/50
55/55 [==============================] - 0s 6ms/step - loss: -387189408.0000 - accuracy: 5.7241e-04 - val_loss: -351479424.0000 - val_accuracy: 0.0000e+00
Epoch 28/50
55/55 [==============================] - 0s 6ms/step - loss: -415752256.0000 - accuracy: 5.7241e-04 - val_loss: -377284608.0000 - val_accuracy: 0.0000e+00
Epoch 29/50
55/55 [==============================] - 0s 7ms/step - loss: -445482112.0000 - accuracy: 5.7241e-04 - val_loss: -403229696.0000 - val_accuracy: 0.0000e+00
Epoch 30/50
55/55 [==============================] - 0s 6ms/step - loss: -475850528.0000 - accuracy: 5.7241e-04 - val_loss: -430330592.0000 - val_accuracy: 0.0000e+00
Epoch 31/50
55/55 [==============================] - 0s 6ms/step - loss: -507157920.0000 - accuracy: 5.7241e-04 - val_loss: -458301280.0000 - val_accuracy: 0.0000e+00
Epoch 32/50
55/55 [==============================] - 0s 5ms/step - loss: -539452544.0000 - accuracy: 5.7241e-04 - val_loss: -486593376.0000 - val_accuracy: 0.0000e+00
Epoch 33/50
55/55 [==============================] - 0s 6ms/step - loss: -572453824.0000 - accuracy: 5.7241e-04 - val_loss: -515848192.0000 - val_accuracy: 0.0000e+00
Epoch 34/50
55/55 [==============================] - 0s 6ms/step - loss: -606182080.0000 - accuracy: 5.7241e-04 - val_loss: -546212288.0000 - val_accuracy: 0.0000e+00
Epoch 35/50
55/55 [==============================] - 0s 6ms/step - loss: -640890880.0000 - accuracy: 5.7241e-04 - val_loss: -576518592.0000 - val_accuracy: 0.0000e+00
Epoch 36/50
55/55 [==============================] - 0s 6ms/step - loss: -676153088.0000 - accuracy: 5.7241e-04 - val_loss: -608158272.0000 - val_accuracy: 0.0000e+00
Epoch 37/50
55/55 [==============================] - 0s 6ms/step - loss: -712399232.0000 - accuracy: 5.7241e-04 - val_loss: -639760640.0000 - val_accuracy: 0.0000e+00
Epoch 38/50
55/55 [==============================] - 0s 6ms/step - loss: -749332992.0000 - accuracy: 5.7241e-04 - val_loss: -672336832.0000 - val_accuracy: 0.0000e+00
Epoch 39/50
55/55 [==============================] - 0s 7ms/step - loss: -786948864.0000 - accuracy: 5.7241e-04 - val_loss: -706430272.0000 - val_accuracy: 0.0000e+00
Epoch 40/50
55/55 [==============================] - 0s 6ms/step - loss: -825546432.0000 - accuracy: 5.7241e-04 - val_loss: -740109248.0000 - val_accuracy: 0.0000e+00
Epoch 41/50
55/55 [==============================] - 0s 6ms/step - loss: -864771520.0000 - accuracy: 5.7241e-04 - val_loss: -774582912.0000 - val_accuracy: 0.0000e+00
Epoch 42/50
55/55 [==============================] - 0s 6ms/step - loss: -904650944.0000 - accuracy: 5.7241e-04 - val_loss: -809684864.0000 - val_accuracy: 0.0000e+00
Epoch 43/50
55/55 [==============================] - 0s 6ms/step - loss: -945321728.0000 - accuracy: 5.7241e-04 - val_loss: -845258432.0000 - val_accuracy: 0.0000e+00
Epoch 44/50
55/55 [==============================] - 0s 6ms/step - loss: -986362688.0000 - accuracy: 5.7241e-04 - val_loss: -882427392.0000 - val_accuracy: 0.0000e+00
Epoch 45/50
55/55 [==============================] - 0s 6ms/step - loss: -1028666048.0000 - accuracy: 5.7241e-04 - val_loss: -919172224.0000 - val_accuracy: 0.0000e+00
Epoch 46/50
55/55 [==============================] - 0s 6ms/step - loss: -1071472640.0000 - accuracy: 5.7241e-04 - val_loss: -956408832.0000 - val_accuracy: 0.0000e+00
Epoch 47/50
55/55 [==============================] - 0s 6ms/step - loss: -1114818688.0000 - accuracy: 5.7241e-04 - val_loss: -994906880.0000 - val_accuracy: 0.0000e+00
Epoch 48/50
55/55 [==============================] - 0s 6ms/step - loss: -1159046528.0000 - accuracy: 5.7241e-04 - val_loss: -1034489344.0000 - val_accuracy: 0.0000e+00
Epoch 49/50
55/55 [==============================] - 0s 6ms/step - loss: -1204069888.0000 - accuracy: 5.7241e-04 - val_loss: -1073923800.0000 - val_accuracy: 0.0000e+00
Epoch 50/50
55/55 [==============================] - 0s 6ms/step - loss: -1249393792.0000 - accuracy: 5.7241e-04 - val_loss: -1114075520.0000 - val_accuracy: 0.0000e+00
14/14 [==============================] - 0s 4ms/step - loss: -1114075520.0000 - accuracy: 0.0000e+00
Test loss: -1114075520.0
Test accuracy: 0.0
Epoch 1/50
55/55 [==============================] - 7s 31ms/step - loss: 19.0395 - accuracy: 5.7241e-04 - val_loss: -2.4367 - val_accuracy: 0.0000e+00
Epoch 2/50
55/55 [==============================] - 0s 9ms/step - loss: -22.8465 - accuracy: 5.7241e-04 - val_loss: -43.5603 - val_accuracy: 0.0000e+00
Epoch 3/50
55/55 [==============================] - 0s 9ms/step - loss: -64.5046 - accuracy: 5.7241e-04 - val_loss: -84.3678 - val_accuracy: 0.0000e+00
Epoch 4/50
55/55 [==============================] - 1s 9ms/step - loss: -105.9951 - accuracy: 5.7241e-04 - val_loss: -125.3012 - val_accuracy: 0.0000e+00
Epoch 5/50
55/55 [==============================] - 0s 9ms/step - loss: -147.4477 - accuracy: 5.7241e-04 - val_loss: -166.1102 - val_accuracy: 0.0000e+00
Epoch 6/50
55/55 [==============================] - 0s 9ms/step - loss: -188.8746 - accuracy: 5.7241e-04 - val_loss: -206.7889 - val_accuracy: 0.0000e+00
Epoch 7/50
55/55 [==============================] - 0s 9ms/step - loss: -230.2607 - accuracy: 5.7241e-04 - val_loss: -247.6222 - val_accuracy: 0.0000e+00
Epoch 8/50
55/55 [==============================] - 0s 9ms/step - loss: -271.6537 - accuracy: 5.7241e-04 - val_loss: -288.4814 - val_accuracy: 0.0000e+00
Epoch 9/50
55/55 [==============================] - 1s 9ms/step - loss: -313.0376 - accuracy: 5.7241e-04 - val_loss: -329.1891 - val_accuracy: 0.0000e+00
Epoch 10/50
55/55 [==============================] - 1s 10ms/step - loss: -354.4960 - accuracy: 5.7241e-04 - val_loss: -369.9765 - val_accuracy: 0.0000e+00
Epoch 11/50
55/55 [==============================] - 0s 9ms/step - loss: -395.9312 - accuracy: 5.7241e-04 - val_loss: -410.7656 - val_accuracy: 0.0000e+00
Epoch 12/50
55/55 [==============================] - 1s 9ms/step - loss: -437.3672 - accuracy: 5.7241e-04 - val_loss: -451.6279 - val_accuracy: 0.0000e+00
Epoch 13/50
55/55 [==============================] - 1s 9ms/step - loss: -478.8189 - accuracy: 5.7241e-04 - val_loss: -492.4077 - val_accuracy: 0.0000e+00
Epoch 14/50
55/55 [==============================] - 0s 9ms/step - loss: -520.3107 - accuracy: 5.7241e-04 - val_loss: -533.2303 - val_accuracy: 0.0000e+00
Epoch 15/50
55/55 [==============================] - 0s 9ms/step - loss: -561.7698 - accuracy: 5.7241e-04 - val_loss: -574.2296 - val_accuracy: 0.0000e+00
Epoch 16/50
55/55 [==============================] - 1s 9ms/step - loss: -603.2582 - accuracy: 5.7241e-04 - val_loss: -614.9781 - val_accuracy: 0.0000e+00
Epoch 17/50
55/55 [==============================] - 1s 10ms/step - loss: -644.7047 - accuracy: 5.7241e-04 - val_loss: -655.8083 - val_accuracy: 0.0000e+00
Epoch 18/50
55/55 [==============================] - 0s 8ms/step - loss: -686.2042 - accuracy: 5.7241e-04 - val_loss: -696.4968 - val_accuracy: 0.0000e+00
Epoch 19/50
55/55 [==============================] - 0s 7ms/step - loss: -727.6508 - accuracy: 5.7241e-04 - val_loss: -737.3849 - val_accuracy: 0.0000e+00
Epoch 20/50
55/55 [==============================] - 0s 6ms/step - loss: -769.1319 - accuracy: 5.7241e-04 - val_loss: -778.3550 - val_accuracy: 0.0000e+00
Epoch 21/50
55/55 [==============================] - 0s 8ms/step - loss: -810.6725 - accuracy: 5.7241e-04 - val_loss: -819.0785 - val_accuracy: 0.0000e+00
Epoch 22/50
55/55 [==============================] - 0s 8ms/step - loss: -852.1271 - accuracy: 5.7241e-04 - val_loss: -860.0739 - val_accuracy: 0.0000e+00
Epoch 23/50
55/55 [==============================] - 0s 7ms/step - loss: -893.6501 - accuracy: 5.7241e-04 - val_loss: -900.9567 - val_accuracy: 0.0000e+00
Epoch 24/50
55/55 [==============================] - 0s 7ms/step - loss: -935.1481 - accuracy: 5.7241e-04 - val_loss: -941.8395 - val_accuracy: 0.0000e+00
Epoch 25/50
55/55 [==============================] - 0s 7ms/step - loss: -976.6291 - accuracy: 5.7241e-04 - val_loss: -982.7310 - val_accuracy: 0.0000e+00
Epoch 26/50
55/55 [==============================] - 0s 6ms/step - loss: -1018.1395 - accuracy: 5.7241e-04 - val_loss: -1023.5629 - val_accuracy: 0.0000e+00
Epoch 27/50
55/55 [==============================] - 0s 6ms/step - loss: -1059.6605 - accuracy: 5.7241e-04 - val_loss: -1064.3490 - val_accuracy: 0.0000e+00
Epoch 28/50
55/55 [==============================] - 0s 6ms/step - loss: -1101.1367 - accuracy: 5.7241e-04 - val_loss: -1105.2632 - val_accuracy: 0.0000e+00
Epoch 29/50
55/55 [==============================] - 0s 6ms/step - loss: -1142.6426 - accuracy: 5.7241e-04 - val_loss: -1146.0499 - val_accuracy: 0.0000e+00
Epoch 30/50
55/55 [==============================] - 0s 8ms/step - loss: -1184.1335 - accuracy: 5.7241e-04 - val_loss: -1186.9875 - val_accuracy: 0.0000e+00
Epoch 31/50
55/55 [==============================] - 0s 6ms/step - loss: -1225.6340 - accuracy: 5.7241e-04 - val_loss: -1227.8490 - val_accuracy: 0.0000e+00
Epoch 32/50
55/55 [==============================] - 0s 6ms/step - loss: -1267.1718 - accuracy: 5.7241e-04 - val_loss: -1268.7000 - val_accuracy: 0.0000e+00
Epoch 33/50
55/55 [==============================] - 0s 9ms/step - loss: -1308.7137 - accuracy: 5.7241e-04 - val_loss: -1309.5510 - val_accuracy: 0.0000e+00
Epoch 34/50
55/55 [==============================] - 0s 9ms/step - loss: -1350.2095 - accuracy: 5.7241e-04 - val_loss: -1350.5431 - val_accuracy: 0.0000e+00
Epoch 35/50
55/55 [==============================] - 0s 9ms/step - loss: -1391.7747 - accuracy: 5.7241e-04 - val_loss: -1391.3669 - val_accuracy: 0.0000e+00
Epoch 36/50
55/55 [==============================] - 0s 8ms/step - loss: -1433.2729 - accuracy: 5.7241e-04 - val_loss: -1432.3436 - val_accuracy: 0.0000e+00
Epoch 37/50
55/55 [==============================] - 1s 9ms/step - loss: -1474.8049 - accuracy: 5.7241e-04 - val_loss: -1473.1807 - val_accuracy: 0.0000e+00
Epoch 38/50
55/55 [==============================] - 0s 9ms/step - loss: -1516.3108 - accuracy: 5.7241e-04 - val_loss: -1514.0415 - val_accuracy: 0.0000e+00
Epoch 39/50
55/55 [==============================] - 0s 9ms/step - loss: -1557.8083 - accuracy: 5.7241e-04 - val_loss: -1554.9731 - val_accuracy: 0.0000e+00
Epoch 40/50
55/55 [==============================] - 0s 9ms/step - loss: -1599.3489 - accuracy: 5.7241e-04 - val_loss: -1595.7476 - val_accuracy: 0.0000e+00
Epoch 41/50
55/55 [==============================] - 0s 9ms/step - loss: -1640.8610 - accuracy: 5.7241e-04 - val_loss: -1636.6317 - val_accuracy: 0.0000e+00
Epoch 42/50
55/55 [==============================] - 0s 9ms/step - loss: -1682.3579 - accuracy: 5.7241e-04 - val_loss: -1677.5583 - val_accuracy: 0.0000e+00
Epoch 43/50
55/55 [==============================] - 0s 9ms/step - loss: -1723.8544 - accuracy: 5.7241e-04 - val_loss: -1718.4192 - val_accuracy: 0.0000e+00
Epoch 44/50
55/55 [==============================] - 0s 8ms/step - loss: -1765.3503 - accuracy: 5.7241e-04 - val_loss: -1759.2623 - val_accuracy: 0.0000e+00
Epoch 45/50
55/55 [==============================] - 0s 9ms/step - loss: -1806.8374 - accuracy: 5.7241e-04 - val_loss: -1800.1458 - val_accuracy: 0.0000e+00
Epoch 46/50
55/55 [==============================] - 0s 9ms/step - loss: -1848.3278 - accuracy: 5.7241e-04 - val_loss: -1840.9270 - val_accuracy: 0.0000e+00
Epoch 47/50
55/55 [==============================] - 0s 9ms/step - loss: -1889.8330 - accuracy: 5.7241e-04 - val_loss: -1881.8030 - val_accuracy: 0.0000e+00
Epoch 48/50
55/55 [==============================] - 0s 9ms/step - loss: -1931.3304 - accuracy: 5.7241e-04 - val_loss: -1922.6808 - val_accuracy: 0.0000e+00
Epoch 49/50
55/55 [==============================] - 0s 9ms/step - loss: -1972.8458 - accuracy: 5.7241e-04 - val_loss: -1963.6410 - val_accuracy: 0.0000e+00
Epoch 50/50
55/55 [==============================] - 0s 9ms/step - loss: -2014.3466 - accuracy: 5.7241e-04 - val_loss: -2004.5703 - val_accuracy: 0.0000e+00
14/14 [==============================] - 0s 4ms/step - loss: -2004.5703 - accuracy: 0.0000e+00
Test loss: -2004.5703125
Test accuracy: 0.0
```

Analysis of the performance of various approaches

In this assignment, I explored the performance of various approaches in neural networks for different tasks. Specifically, we discussed the effectiveness of Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs) in tasks such as language modeling, image classification, and speech recognition.

While both RNNs and CNNs can be effective in certain scenarios, the choice of which model to use ultimately depends on the nature of the data and the specific task at hand. In class, we discussed guidelines on which model might work best in different circumstances. Armed with this knowledge, I was able to apply these models to my own project using an Excel file that I also used for text classification.

It is worth noting that while CNNs are more efficient when it comes to processing images and videos, RNNs are able to handle inputs of variable length and can capture the temporal dependencies in the data. Ultimately, the choice of which model to use depends on the specific characteristics of the data and the task at hand.

In [ ]: