National College of Ireland

Configuration Manual

MSc Research Project
Data Analytics

Kartik Sharma
Student ID: 21125813

School of Computing
National College of Ireland

Supervisor: Prof.Christian Horn

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Kartik Sharma
Student ID:	21125813
Programme:	Data Analytics
Year:	2022
Module:	MSc Research Project
Supervisor:	Prof.Christian Horn
Submission Due Date:	15/12/2022
Project Title:	Configuration Manual
Word Count:	355
Page Count:	9

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Kartik Sharma
Date:	15th December 2022

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Kartik Sharma
21125813

1 System Configuration

The project is done on 64-Bit Windows 11 operating system with 16GB ram with AMD Ryzen 9 5900HX processor having a base clock speed of 3.30 GHz.

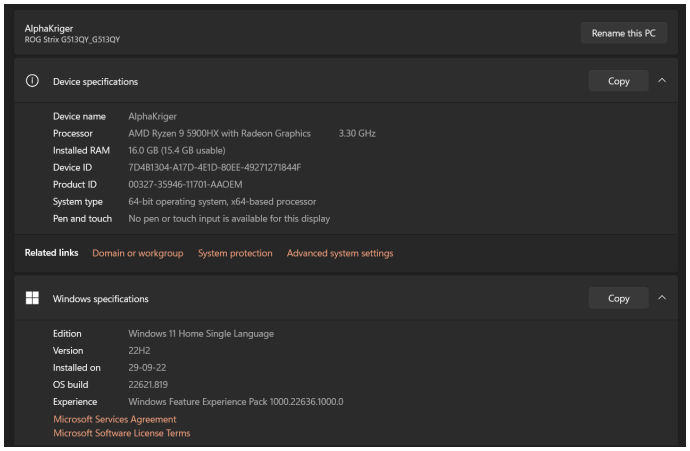


Figure 1: System Configuration

2 Software Requirement

For the project we have used following software :

1. Anaconda 2.3.3
2. Python 3.9.7
3. Jupyter Notebook

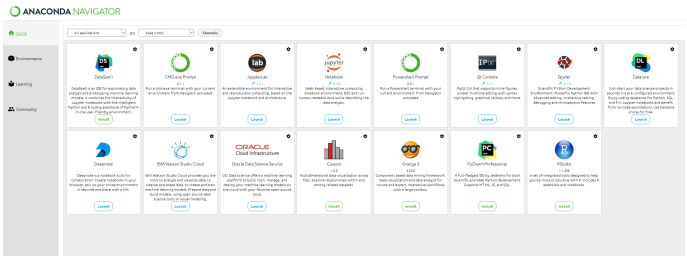


Figure 2: Software Requirement

3 Python Libraries

The project uses following python libraries :

1. TensorFlow
2. Keras
3. numpy
4. matplotlib
5. pandas
6. sklearn
7. itertools
8. nltk
9. transformers
10. scipy

4 Dataset

- The Data is originally created by Edmunds(USA-based car selling business), and for the study, the data is taken from the Kaggle.com where it was publicly published. Original Dataset link: https://tinyurl.com/bdfjau8y
- The Data consists of reviews from car owners in the USA from 2002 to 2018 in separate (comma separated value(.csv)) files for each car brand.

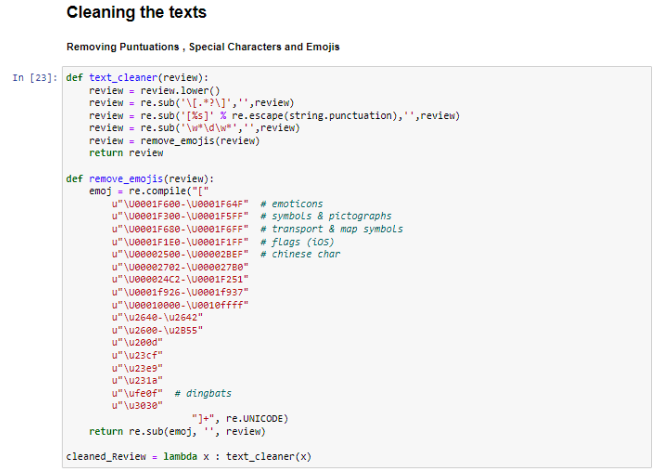
5 Data Preprocessing

- All the data is combined into one common dataset to begin the project.
- Data is cleaned and not required columns were dropped from the data frame ,refer to figure 3.

```
In [4]: tempData =[['vechileTempName','ReviewTempTitle','Review','Rating']]

In [5]: reviewDataSet  = pd.DataFrame()

In [6]: reviewDataSet

Out[6]: —

In [ ]:

In [7]: for i in range (0,len(item_lst)):
            tempDataSet = pd.read_csv(path+item_lst[i],engine='python', error_bad_lines=False)
            #print(path+item_lst[i])
            tempDataSet =  tempDataSet.drop(['Unnamed: 0','Author_Name'],axis=1)
            reviewDataSet = reviewDataSet.append(tempDataSet,ignore_index=True)

In [8]: reviewDataSet = reviewDataSet.dropna()
        reviewDataSet = reviewDataSet.reset_index()
        reviewDataSet = reviewDataSet.drop(['index'],axis=1)

In [9]: reviewDataSet.shape

Out[9]: (214008, 5)

In [10]: brands = []

         for i in item_lst:
             i = i.replace('Scraped_Car_Review_', '')
             i  = i.replace('Scrapped_Car_Review_', '')
             brnd_name = i[:-4]
             brands.append(brnd_name)
```

Figure 3: Data Processing step 1

- The text prepossessing and cleaning is done before training the models.



Cleaning the texts

Removing Puntuations , Special Characters and Emojis

```
In [23]: def text_cleaner(review):
             review = review.lower()
             review = re.sub('\[.*?\]','',review)
             review = re.sub('[%s]' % re.escape(string.punctuation),'',review)
             review = re.sub('\w*\d\w*','',review)
             review = remove_emojis(review)
             return review

         def remove_emojis(review):
             emoj = re.compile("["
                 u"\U0001F600-\U0001F64F"  # emoticons
                 u"\U0001F300-\U0001F5FF"  # symbols & pictographs
                 u"\U0001F680-\U0001F6FF"  # transport & map symbols
                 u"\U0001F1E0-\U0001F1FF"  # flags (iOS)
                 u"\U00002500-\U00002BEF"  # chinese char
                 u"\U00002702-\U000027B0"
                 u"\U000024C2-\U0001F251"
                 u"\U0001f926-\U0001f937"
                 u"\U00010000-\U0010ffff"
                 u"\u2640-\u2642"
                 u"\u2600-\u2B55"
                 u"\u200d"
                 u"\u23cf"
                 u"\u23e9"
                 u"\u231a"
                 u"\ufe0f"  # dingbats
                 u"\u3030"
                                   "]+", re.UNICODE)
             return re.sub(emoj, '', review)

         cleaned_Review = lambda x : text_cleaner(x)
```

Figure 4: Data Processing step 2

- Figure 5 is the final processed and cleaned dataset that was finally used to train the Neural Networks

Out[59]:

	cleaned_Reviews	Rob_Emotion
0	its been a great delivery vehicle for my cafe...	positive
1	bought this car as a commuter vehicle for a v...	negative
2	this van rocks its the best lots of \rroom i ...	positive
3	great work vehicle drives nice has lots of ro...	positive
4	good solid frame and suspension well equippe...	negative
...	...	...
212847	i have owned the tiguan for a year and month...	positive
212848	now had months with suv nice suv handles we...	negative
212849	smaller dimensions and my driving experience ...	positive
212850	we have had our tiguan for a month prior to ...	positive
212851	the tiguan s can be had for a reasonable and...	positive

212852 rows × 2 columns

Figure 5: Processed Dataset for Neural Network

6 Data Analysis

The Sales were compared to the Sentiments of consumers per brand to find the pattern between sales and sentiments.

Finding the Pattern for Ford

In [30]:
```
tempData = []
brandDataSet = roberta_Senti_Dataset[roberta_Senti_Dataset['Brand'] == 'ford']
brandDataSet = brandDataSet.reset_index()
brandDataSet.head()
```

Out[30]:

	index	Rob_Emotion	Review_Date	Brand	Rob_Emo_Quantified
0	8356	positive	2018-06-06	ford	3
1	8357	positive	2017-08-12	ford	3
2	8358	positive	2017-06-15	ford	3
3	8359	positive	2017-05-18	ford	3
4	8360	positive	2016-01-03	ford	3

In [31]:
```
from tqdm import tqdm
import numpy

# Getting Average Emotion per year
tempData = []
brandEmotionDataSet = pd.DataFrame()

for i in range (0,len(brandDataSet.index)):
        tempData.append('temp')
brandEmotionDataSet['AverageEmotion'] = tempData
brandEmotionDataSet['Year'] = tempData

year=[]
for i in range (0,len(brandDataSet.index)):
    year.append(brandDataSet['Review_Date'][i].year)
year = list(set(year))
year = numpy.array(year)

for i in range(len(year)):
    emoValue = 0
    count = 0
    averageEmotion = 0
    for j in range (0,len(brandDataSet.index)):
        if year[i] == brandDataSet['Review_Date'][j].year:
            emoValue = emoValue + brandDataSet['Rob_Emo_Quantified'][j]
            count = count+1
    averageEmotion = emoValue/count
    brandEmotionDataSet['Year'][i] = year[i]
    brandEmotionDataSet['AverageEmotion'][i] = averageEmotion

brandEmotionDataSet.drop(brandEmotionDataSet[brandEmotionDataSet.AverageEmotion == 'temp'].index, inplace=True)
brandEmotionDataSet
```

Figure 6: Processing data to generate sentiment graph

Comparing with Ford Sales

In [35]:
```
SalesData = pd.read_csv('FordSales.csv')
SalesData = SalesData[['Year','Sales']]

for i in range (0,len(SalesData.index)):
    SalesData['Sales'][i] = re.sub(',','',SalesData['Sales'][i])

SalesData
```

Out[35]:

	Year	Sales
0	2002	2990472
1	2003	2886575
2	2004	2766025
3	2005	2634041
4	2006	2415059
5	2007	2087048
6	2008	1680321
7	2009	1440653
8	2010	1752511
9	2011	2057210
10	2012	2160859
11	2013	2403542
12	2014	2376841
13	2015	2501855
14	2016	2487487
15	2017	2484041
16	2018	2381635
17	2019	2293984

Figure 7: Processing sales data to generate graph

Ford Sales Vs Sentiment

In [36]:
```
SalesData['Year']= pd.to_numeric(SalesData['Year'])
SalesData['Sales']= pd.to_numeric(SalesData['Sales'])
SalesData = SalesData.set_index('Year')
SalesData.plot(title="Ford Sales Analysis",grid=True)
brandEmotionDataSet.plot(title="Ford Sentiment Analysis",grid=True)
```

Out[36]: <AxesSubplot:title={'center':'Ford Sentiment Analysis'}, xlabel='Year'>

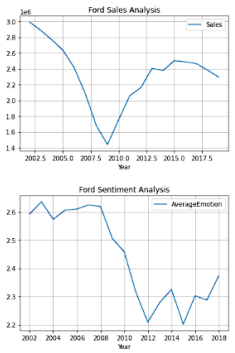
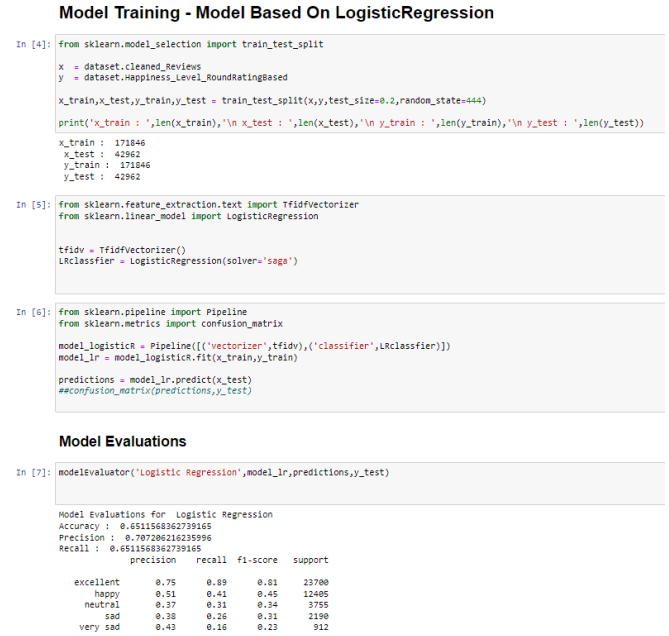


Figure 8: Analysing the Sales and Sentiment Graphs

7 Model Training and Testing

Three approaches were taken to extract the sentiments from the texts, and after evaluation the best approach was combined with neural network to create final model.

1. Logistic Regression



Model Training - Model Based On LogisticRegression

In [4]:
```
from sklearn.model_selection import train_test_split

x  = dataset.cleaned_Reviews
y  = dataset.Happiness_Level_RoundRatingBased

x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=444)

print('x_train : ',len(x_train),'\n x_test : ',len(x_test),'\n y_train : ',len(y_train),'\n y_test : ',len(y_test))
```

```
x_train :  171846
 x_test :  42962
 y_train :  171846
 y_test :  42962
```

In [5]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression


tfidv = TfidfVectorizer()
LRclassfier = LogisticRegression(solver='saga')
```

In [6]:
```
from sklearn.pipeline import Pipeline
from sklearn.metrics import confusion_matrix

model_logisticR = Pipeline([('vectorizer',tfidv),('classifier',LRclassfier)])
model_lr = model_logisticR.fit(x_train,y_train)

predictions = model_lr.predict(x_test)
##confusion_matrix(predictions,y_test)
```

Model Evaluations

In [7]:
```
modelEvaluator('Logistic Regression',model_lr,predictions,y_test)
```

```
Model Evaluations for  Logistic Regression
Accuracy :  0.6511568362739165
Precision :  0.707206216235996
Recall :  0.6511568362739165
              precision    recall  f1-score   support

   excellent       0.75      0.89      0.81     23700
       happy       0.51      0.41      0.45     12405
     neutral       0.37      0.31      0.34      3755
         sad       0.38      0.26      0.31      2190
    very sad       0.43      0.16      0.23       912
```

Figure 9: Logistic Regression Model and Evaluation

2. Vader Sentiment Scoring

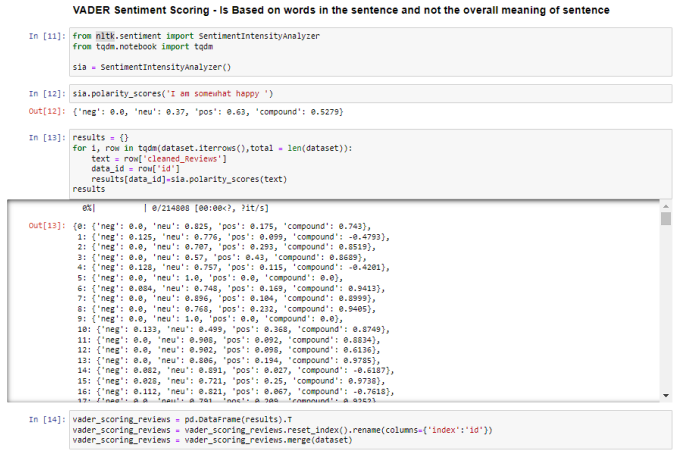


Figure 10: Vader Sentiment Scoring

3. RoBERTa Pretrained model

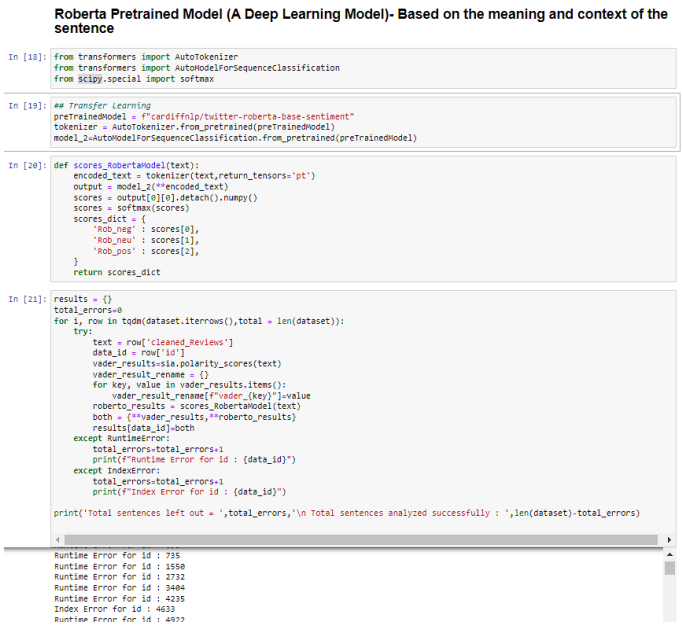


Figure 11: RoBERTa pretrained model

Model Comparison

```
In [25]: sns.pairplot(data=result_DF,vars=['neg', 'neu', 'pos', 'Rob_neg', 'Rob_neu', 'Rob_pos'],hue='Rounded_Rating',palette='tab10')
         plt.show()
```

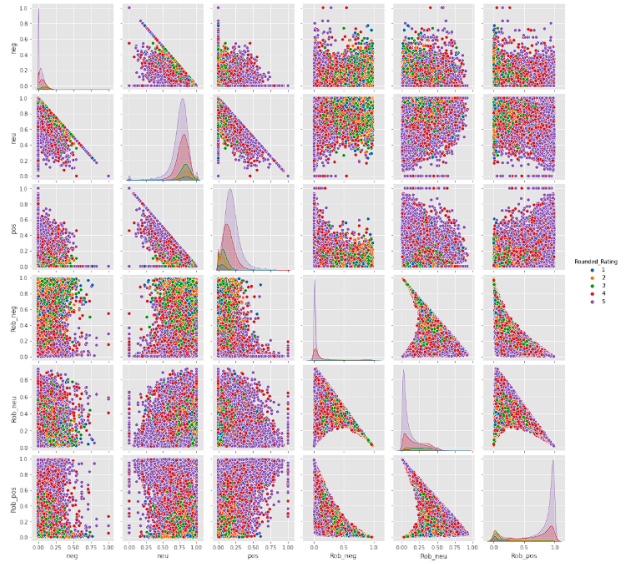


Figure 12: RoBERTa pretrained model Vs Vader Sentiment Scoring

4. Neural Networks

Emotion Detection Using Neural Network and TensorFlow - Model Creation

```
In [58]: import tensorflow as tf
         import keras

         from nltk.stem.porter import PorterStemmer
         from nltk.corpus import stopwords
```

```
In [59]: dataset_NeuralNet = result_DF[['cleaned_Reviews','Rob_Emotion']]
         dataset_NeuralNet
```

Out[59]:

	cleaned_Reviews	Rob_Emotion
0	its been a great delivery vehicle for my cafe...	positive
1	bought this car as a commuter vehicle for a v...	negative
2	this van rocks its the best lots of \rroom i ...	positive
3	great work vehicle drives nice has lots of ro...	positive
4	good solid frame and suspension well equippe...	negative
...	...	...
212847	i have owned the tiguan for a year and month...	positive
212848	now had months with suv nice suv handles we...	negative
212849	smaller dimensions and my driving experience ...	positive
212850	we have had our tiguan for a month prior to ...	positive
212851	the tiguan s can be had for a reasonable and...	positive

212852 rows × 2 columns

```
In [60]: dataset_NeuralNet.isna().any(axis=1).sum()
Out[60]: 0
```

```
In [61]: ps = PorterStemmer()

         def dataPreprocessing(text):
             text = text.split()

             ##stemming
             text = [ps.stem(word) for word in text if not word in stopwords.words('english')]
             return " ".join(text)
```

Figure 13: Data Preprocessing for Neural network

```
In [62]: dataset_NeuralNet['cleaned_Reviews'] = dataset_NeuralNet['cleaned_Reviews'].apply(lambda x: dataPreprocessing(x))

In [63]: from sklearn.preprocessing import LabelEncoder

         lable_encoder = LabelEncoder()
         dataset_NeuralNet['N_label'] = lable_encoder.fit_transform(dataset_NeuralNet['Rob_Emotion'])

         dataset_NeuralNet
```

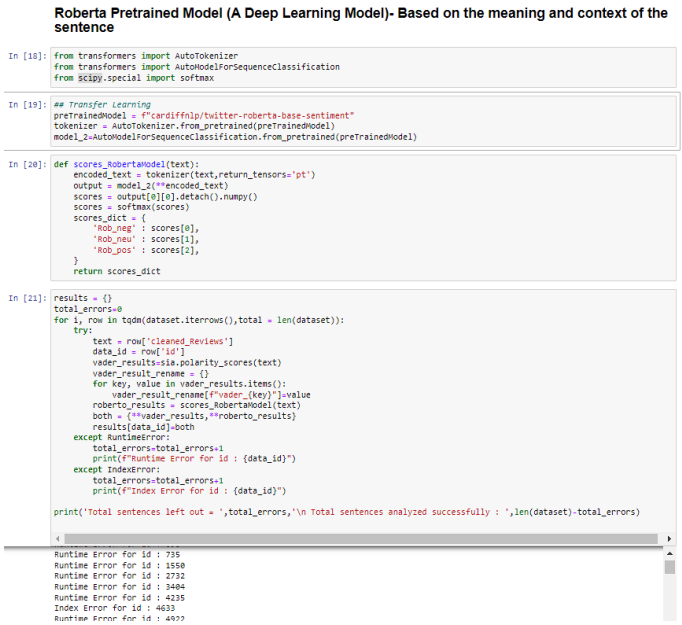
Out[63]:

	cleaned_Reviews	Rob_Emotion	N_label
0	great deliveri vehicl cafe busi good power eco...	positive	2
1	bought car commut vehicl van pool disappoint v...	negative	0
2	van rock best lot room cant lot cargo mine to...	positive	2
3	great work vehicl drive nice lot room easi han...	positive	2
4	good solid frame suspens well equip full power...	negative	0
...	...	...	...
212847	own tiguan year month bought tiguan want small...	positive	2
212848	month suv nice suv handl well acceler well ama...	negative	0
212849	smaller dimens drive experi put tiguan class e...	positive	2
212850	tiguan month prior troubl free passat overal r...	positive	2
212851	tiguan reason get pay averag around superior d...	positive	2

212852 rows × 3 columns

```
In [64]: from sklearn.feature_extraction.text import CountVectorizer

         cv= CountVectorizer(max_features=5000,ngram_range=(1,3))

         dataset_NeuralNet_CV = cv.fit_transform(dataset_NeuralNet['cleaned_Reviews']).toarray()

In [65]: dataset_NeuralNet_CV

Out[65]: array([[0, 0, 0, ..., 0, 0, 0],
                [0, 1, 0, ..., 0, 0, 0],
                [0, 0, 0, ..., 0, 0, 0],
                ...,
                [0, 0, 0, ..., 0, 0, 0],
                [0, 0, 0, ..., 0, 0, 0],
                [0, 0, 0, ..., 0, 0, 0]], dtype=int64)

In [66]: x_train,x_test,y_train,y_test = train_test_split(dataset_NeuralNet_CV,dataset_NeuralNet['N_label'],test_size=0.30,random_state=4

In [67]: from tensorflow.keras.layers import Dense
         from tensorflow.keras.models import Sequential

In [68]: model = Sequential()
         model.add(Dense(12,input_shape=(x_train.shape[1],),activation='relu'))
         model.add(Dense(8,activation='relu'))
         model.add(Dense(8,activation='softmax'))

         model.compile(loss='sparse_categorical_crossentropy',optimizer='adam',metrics=['accuracy'])

         model.fit(x_train,y_train,epochs=10,batch_size=10)

         Epoch 1/10
         14900/14900 [==============================] - 18s 1ms/step - loss: 0.1761 - accuracy: 0.8629
         Epoch 2/10
         14900/14900 [==============================] - 12s 816us/step - loss: 0.1071 - accuracy: 0.8836
```

Figure 14: Layers Creation for Neural network



Roberta Pretrained Model (A Deep Learning Model)- Based on the meaning and context of the sentence

```
In [18]: from transformers import AutoTokenizer
         from transformers import AutoModelForSequenceClassification
         from scipy.special import softmax

In [19]: ## Transfer Learning
         preTrainedModel = f"cardiffnlp/twitter-roberta-base-sentiment"
         tokenizer = AutoTokenizer.from_pretrained(preTrainedModel)
         model_2=AutoModelForSequenceClassification.from_pretrained(preTrainedModel)

In [20]: def scores_RobertaModel(text):
             encoded_text = tokenizer(text,return_tensors='pt')
             output = model_2(**encoded_text)
             scores = output[0][0].detach().numpy()
             scores = softmax(scores)
             scores_dict = {
                 'Rob_neg' : scores[0],
                 'Rob_neu' : scores[1],
                 'Rob_pos' : scores[2],
             }
             return scores_dict

In [21]: results = {}
         total_errors=0
         for i, row in tqdm(dataset.iterrows(),total = len(dataset)):
             try:
                 text = row['cleaned_Reviews']
                 data_id = row['id']
                 vader_results=sia.polarity_scores(text)
                 vader_result_rename = {}
                 for key, value in vader_results.items():
                     vader_result_rename[f"vader_{key}"]=value
                 roberto_results = scores_RobertaModel(text)
                 both = {**vader_results,**roberto_results}
                 results[data_id]=both
             except RuntimeError:
                 total_errors=total_errors+1
                 print(f"Runtime Error for id : {data_id}")
             except IndexError:
                 total_errors=total_errors+1
                 print(f"Index Error for id : {data_id}")

         print('Total sentences left out = ',total_errors,'\n Total sentences analyzed successfully : ',len(dataset)-total_errors)

         Runtime Error for id : 735
         Runtime Error for id : 1550
         Runtime Error for id : 2732
         Runtime Error for id : 3404
         Runtime Error for id : 4235
         Index Error for id : 4633
         Runtime Error for id : 4922
```

Figure 15: Evaluating the Neural network