Installing transfortmer

```
pip install transformers
```

```
Collecting transformers
  Downloading transformers-4.15.0-py3-none-any.whl (3.4 MB)
     |████████████████████████████████| 3.4 MB 8.8 MB/s
Collecting huggingface-hub<1.0,>=0.1.0
  Downloading huggingface_hub-0.4.0-py3-none-any.whl (67 kB)
     |████████████████████████████████| 67 kB 6.4 MB/s
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.7/dist-packages (from transformers) (4.62.3)
Requirement already satisfied: filelock in /usr/local/lib/python3.7/dist-packages (from transformers) (3.4.2)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.7/dist-packages (from transformers) (2019.12.20)
Collecting tokenizers<0.11,>=0.10.1
  Downloading tokenizers-0.10.3-cp37-cp37m-manylinux_2_5_x86_64.manylinux1_x86_64.manylinux_2_12_x86_64.manylinux2010_x86_64.whl (3.3 MB
     |████████████████████████████████| 3.3 MB 76.1 MB/s
Collecting pyyaml>=5.1
  Downloading PyYAML-6.0-cp37-cp37m-manylinux_2_5_x86_64.manylinux1_x86_64.manylinux_2_12_x86_64.manylinux2010_x86_64.whl (596 kB)
     |████████████████████████████████| 596 kB 75.2 MB/s
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (from transformers) (2.23.0)
Collecting sacremoses
  Downloading sacremoses-0.0.47-py2.py3-none-any.whl (895 kB)
     |████████████████████████████████| 895 kB 51.3 MB/s
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.7/dist-packages (from transformers) (21.3)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.7/dist-packages (from transformers) (1.19.5)
Requirement already satisfied: importlib-metadata in /usr/local/lib/python3.7/dist-packages (from transformers) (4.10.0)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.7/dist-packages (from huggingface-hub<1.0,>=0.1.0->t
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.7/dist-packages (from packaging>=20.0->transformers) (
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from importlib-metadata->transformers) (3.7.0)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python3.7/dist-packages (from requests->transfo
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from requests->transformers) (2021.10.8)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests->transformers) (2.10)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from requests->transformers) (3.0.4)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from sacremoses->transformers) (1.15.0)
Requirement already satisfied: click in /usr/local/lib/python3.7/dist-packages (from sacremoses->transformers) (7.1.2)
Requirement already satisfied: joblib in /usr/local/lib/python3.7/dist-packages (from sacremoses->transformers) (1.1.0)
Installing collected packages: pyyaml, tokenizers, sacremoses, huggingface-hub, transformers
  Attempting uninstall: pyyaml
    Found existing installation: PyYAML 3.13
    Uninstalling PyYAML-3.13:
      Successfully uninstalled PyYAML-3.13
Successfully installed huggingface-hub-0.4.0 pyyaml-6.0 sacremoses-0.0.47 tokenizers-0.10.3 transformers-4.15.0
```

Importing Bert tokenizer, classifier, input feature extractor

```
from transformers import BertTokenizer, TFBertForSequenceClassification
from transformers import InputExample, InputFeatures

model = TFBertForSequenceClassification.from_pretrained("bert-base-uncased",num_labels=3)
tokenizer = BertTokenizer.from_pretrained("bert-base-uncased")
model.summary()
```

```
Downloading: 100%                                   570/570 [00:00<00:00, 16.0kB/s]

Downloading: 100%                                   511M/511M [00:09<00:00, 59.9MB/s]

All model checkpoint layers were used when initializing TFBertForSequenceClassification.
```

Importing Amozon Product Review dataset (Personal Care Applicances) from tensorflow dataset

```
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
```

```
import tensorflow as tf
import tensorflow_datasets as tfds
data = tfds.load('huggingface:amazon_us_reviews/Books_v1_01')
df = tfds.as_dataframe(data)
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
/usr/local/lib/python3.10/dist-packages/tensorflow_datasets/core/lazy_imports_lib.py in _try_import(module_name)
     29   try:
---> 30     mod = importlib.import_module(module_name)
     31     return mod

                                 16 frames
ModuleNotFoundError: No module named 'datasets'

The above exception was the direct cause of the following exception:

ModuleNotFoundError                       Traceback (most recent call last)
/usr/local/lib/python3.10/dist-packages/tensorflow_datasets/core/utils/py_utils.py in reraise(e, prefix, suffix)
    382     else:
    383       exception = RuntimeError(f'{type(e).__name__}: {msg}')
--> 384     raise exception from e
    385   # Otherwise, modify the exception in-place
    386   elif len(e.args) <= 1:

ModuleNotFoundError: No module named 'datasets'
Failed importing datasets. This likely means that the dataset requires additional dependencies that have to be manually installed (usual

---------------------------------------------------------------------------
NOTE: If your import is failing due to a missing package, you can
manually install dependencies using either !pip or !apt.

To view examples of installing some common dependencies, click the
"Open Examples" button below.
---------------------------------------------------------------------------
```

OPEN EXAMPLES SEARCH STACK OVERFLOW

```
a=!curl -X GET  "https://datasets-server.huggingface.co/first-rows?dataset=amazon_us_reviews&config=Books_v1_00&split=train"
```

Creating dataset using only review and rating

```
df.columns
col=df[['data/review_body','data/star_rating']]
dataset=col.copy()
```

```
dataset = dataset.rename(columns={'data/review_body': 'review', 'data/star_rating': 'rating'})
dataset.tail()
```

	review	rating
85976	b"This is the real deal. Don't bother with the...	5
85977	b'I like the Bryton Picks very much. Have orde...	5
85978	b"I have had a Remington before but needed a n...	3
85979	b"I was surprised that it really didn't do muc...	2
85980	b'The blades were an excellent fit for my T-li...	5

Converting to three classes sentiment analysis (Negative, Nuetral, and Positive review)

```
def change_rating(a):
  if a<2:
    return 0
  elif a==3:
    return 1
  else:
    return 2

dataset['review']=dataset['review'].str.decode("utf-8")
dataset['rating']=dataset['rating'].apply(change_rating)


reviews = dataset['review'].values.tolist()
```

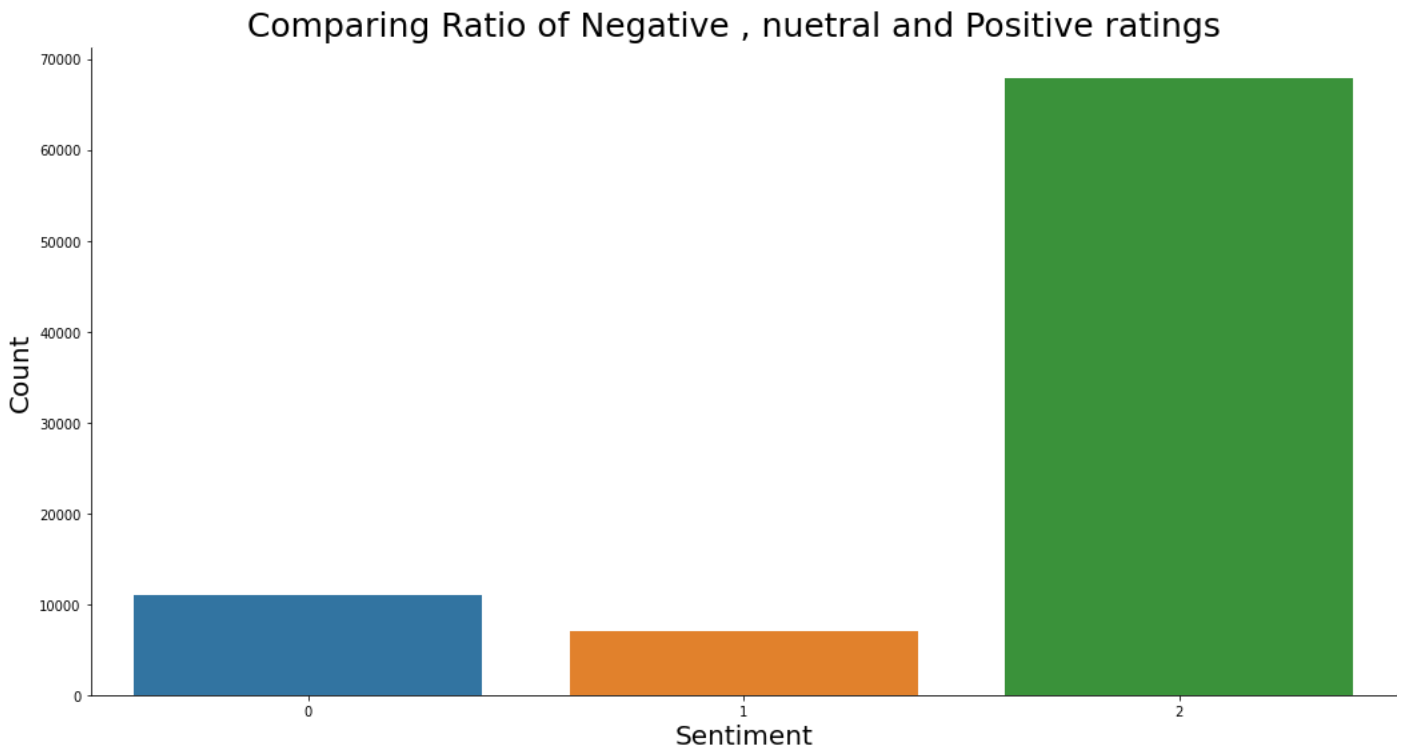
Displaying one review

```
reviews[0]
```

```
'These glasses are an excellent value.  The fit is good and they are very comfortable.  Because of my legal blindness, there aren't a lot
because they are reasonably priced I can have more than one pair available.'
```

Displaying the counts for each group

```
import matplotlib.pyplot as plt
import seaborn as sns
fig, axes = plt.subplots(1, figsize=(15,8))
fig.suptitle("Comparing Ratio of Negative , nuetral and Positive ratings", fontsize = 25)
plt.tight_layout(pad = 3.5)
sns.countplot(x = "rating", data = dataset)
axes.set_xlabel("Sentiment", fontsize = 20)
axes.set_ylabel("Count", fontsize = 20)
sns.despine()
```



Removing stopwords and applying porter stemmer

```
from nltk.stem import PorterStemmer
import numpy as np


def remove_stopwords(words):
```

```
  stopwords=['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've",
           "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself',
           'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself', 'they', 'them',
           'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "that'll",
           'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had',
           'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as',
           'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during',
           'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under',
           'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both',
           'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so',
           'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now', 'd',
           'll', 'm', 'o', 're', 've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'doesn',
           "doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'mightn', "mightn't",
           'mustn', "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't", 'won', "wo
  st=[word for word in words.split() if word not in stopwords]
  st=' '.join(st)
  return st

def porter_stemmer(words):
  stemmer = PorterStemmer()
  st=[stemmer.stem(word) for word in words.split()]
  st=' '.join(st)
  return st



dataset['review'] = dataset['review'].apply(remove_stopwords)
dataset['review'] = dataset['review'].apply(porter_stemmer)


train=dataset[:30000]
test=dataset[30000:38000]
```

Functions for converting the words to input words and input features for preparing the training and validation dataset for bert classifier model

```
def convert_data_to_examples(train, test, DATA_COLUMN, LABEL_COLUMN):
train_InputExamples, validation_InputExamples = convert_data_to_examples(train, test, DATA_COLUMN, LABEL_COLUMN)

train_data = convert_examples_to_tf_dataset(list(train_InputExamples), tokenizer)
train_data = train_data.shuffle(100).batch(32).repeat(2)

validation_data = convert_examples_to_tf_dataset(list(validation_InputExamples), tokenizer)
validation_data = validation_data.batch(32)
```

```
/usr/local/lib/python3.7/dist-packages/transformers/tokenization_utils_base.py:2232: FutureWarning: The `pad_to_max_length` argument is
  FutureWarning,
```

Model compilation and fitting

```
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=3e-5, epsilon=1e-08, clipnorm=1.0),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=[tf.keras.metrics.SparseCategoricalAccuracy('accuracy')])

model.fit(train_data, epochs=3, validation_data=validation_data)
```

```
Epoch 1/3
1876/1876 [==============================] - 1590s 838ms/step - loss: 0.3280 - accuracy: 0.8661 - val_loss: 0.4153 - val_accuracy: 0.847
Epoch 2/3
1876/1876 [==============================] - 1572s 838ms/step - loss: 0.1729 - accuracy: 0.9304 - val_loss: 0.5087 - val_accuracy: 0.834
Epoch 3/3
1876/1876 [==============================] - 1572s 838ms/step - loss: 0.0908 - accuracy: 0.9678 - val_loss: 0.6345 - val_accuracy: 0.844
<keras.callbacks.History at 0x7f9385724f10>
```

Convert the dataset to countvectorizer

```
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
countVect = CountVectorizer()
X_train_countVect = countVect.fit_transform(train['review'])
```

Applying Multonomial Naive Bais for sentiment analysis for comparing results

```
from sklearn.naive_bayes import  MultinomialNB
from sklearn.metrics import accuracy_score
mnb = MultinomialNB()
mnb.fit(X_train_countVect, train['rating'])
```

```
MultinomialNB()
```

```
predictions = mnb.predict(countVect.transform(test['review']))
print ("\nAccuracy on validation set: {:.4f}".format(accuracy_score(test['rating'], predictions)))
```

```
Accuracy on validation set: 0.8317
```