D213 Task 2

Vincent Taylor

Part I

(A1) Research Question: The research question that I am looking to answer is; can the sentiment of reviews be predicted using a nerual network? I believe that this research question is important for the organization due to the fact we would be able to take customer feedback and implement it in order to improve and retain longer term customers.

(A2) Objectives: The goal of this analysis is to be able to build a model that can accurately classify reviews as either positive or negative in order to better help us understand customer sentiment.

(A3) Perscribed Network Recurrent Neural Networks or RNN is designed to work with data that is in sequential order which would be beneficial for our analysis as we have customer reviews where the order matters.

```
!pip install tensorflow
!pip install emoji
!pip install tensorflow scikit-learn
!pip install --upgrade tensorflow keras
!pip install --upgrade adapt
!pip install scikeras
!pip install keras-tuner
```

```
Requirement already satisfied: tensorflow in c:\users\vince\anaconda3\
lib\site-packages (2.15.0)
Requirement already satisfied: tensorflow-intel==2.15.0 in c:\users\
vince\anaconda3\lib\site-packages (from tensorflow) (2.15.0)
Requirement already satisfied: absl-py>=1.0.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (2.1.0)
Requirement already satisfied: astunparse>=1.6.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=23.5.26 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (23.5.26)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in
c:\users\vince\anaconda3\lib\site-packages (from tensorflow-
intel==2.15.0->tensorflow) (0.5.4)
Requirement already satisfied: google-pasta>=0.1.1 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (0.2.0)
Requirement already satisfied: h5py>=2.9.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (3.7.0)
Requirement already satisfied: libclang>=13.0.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (16.0.6)
Requirement already satisfied: ml-dtypes~=0.2.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (0.2.0)
Requirement already satisfied: numpy<2.0.0,>=1.23.5 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (1.24.3)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (3.3.0)
Requirement already satisfied: packaging in c:\users\vince\anaconda3\
lib\site-packages (from tensorflow-intel==2.15.0->tensorflow) (23.0)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!
=4.21.3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (4.25.2)
Requirement already satisfied: setuptools in c:\users\vince\anaconda3\
lib\site-packages (from tensorflow-intel==2.15.0->tensorflow) (68.0.0)
Requirement already satisfied: six>=1.12.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (2.4.0)
Requirement already satisfied: typing-extensions>=3.6.6 in c:\users\
vince\anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (4.7.1)
Requirement already satisfied: wrapt<1.15,>=1.11.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (1.14.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in
c:\users\vince\anaconda3\lib\site-packages (from tensorflow-
intel==2.15.0->tensorflow) (0.31.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (1.60.1)
Requirement already satisfied: tensorboard<2.16,>=2.15 in c:\users\
vince\anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (2.15.2)
Requirement already satisfied: tensorflow-estimator<2.16,>=2.15.0 in
c:\users\vince\anaconda3\lib\site-packages (from tensorflow-
intel==2.15.0->tensorflow) (2.15.0)
Requirement already satisfied: keras<2.16,>=2.15.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (2.15.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in c:\users\vince\
anaconda3\lib\site-packages (from astunparse>=1.6.0->tensorflow-
intel==2.15.0->tensorflow) (0.38.4)
Requirement already satisfied: google-auth<3,>=1.6.3 in c:\users\
vince\anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15-
>tensorflow-intel==2.15.0->tensorflow) (2.27.0)
Requirement already satisfied: google-auth-oauthlib<2,>=0.5 in c:\
users\vince\anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15-
>tensorflow-intel==2.15.0->tensorflow) (1.2.0)
Requirement already satisfied: markdown>=2.6.8 in c:\users\vince\
anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15->tensorflow-
intel==2.15.0->tensorflow) (3.4.1)
Requirement already satisfied: requests<3,>=2.21.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15->tensorflow-
intel==2.15.0->tensorflow) (2.31.0)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0
in c:\users\vince\anaconda3\lib\site-packages (from
tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow) (0.7.2)
Requirement already satisfied: werkzeug>=1.0.1 in c:\users\vince\
anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15->tensorflow-
intel==2.15.0->tensorflow) (2.2.3)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in c:\users\
vince\anaconda3\lib\site-packages (from google-auth<3,>=1.6.3-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(5.3.2)
Requirement already satisfied: pyasn1-modules>=0.2.1 in c:\users\
vince\anaconda3\lib\site-packages (from google-auth<3,>=1.6.3-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(0.2.8)
Requirement already satisfied: rsa<5,>=3.1.4 in c:\users\vince\
anaconda3\lib\site-packages (from google-auth<3,>=1.6.3-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow) (4.9)
Requirement already satisfied: requests-oauthlib>=0.7.0 in c:\users\
vince\anaconda3\lib\site-packages (from google-auth-oauthlib<2,>=0.5-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(1.3.1)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\
vince\anaconda3\lib\site-packages (from requests<3,>=2.21.0-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\vince\
anaconda3\lib\site-packages (from requests<3,>=2.21.0-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\vince\
anaconda3\lib\site-packages (from requests<3,>=2.21.0-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\vince\
anaconda3\lib\site-packages (from requests<3,>=2.21.0-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
```

```
(2023.7.22)
Requirement already satisfied: MarkupSafe>=2.1.1 in c:\users\vince\
anaconda3\lib\site-packages (from werkzeug>=1.0.1-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(2.1.1)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in c:\users\vince\
anaconda3\lib\site-packages (from pyasn1-modules>=0.2.1->google-
auth<3,>=1.6.3->tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0-
>tensorflow) (0.4.8)
Requirement already satisfied: oauthlib>=3.0.0 in c:\users\vince\
anaconda3\lib\site-packages (from requests-oauthlib>=0.7.0->google-
auth-oauthlib<2,>=0.5->tensorboard<2.16,>=2.15->tensorflow-
intel==2.15.0->tensorflow) (3.2.2)
Requirement already satisfied: emoji in c:\users\vince\anaconda3\lib\
site-packages (2.10.1)
Requirement already satisfied: tensorflow in c:\users\vince\anaconda3\
lib\site-packages (2.15.0)
Requirement already satisfied: scikit-learn in c:\users\vince\
anaconda3\lib\site-packages (1.3.0)
Requirement already satisfied: tensorflow-intel==2.15.0 in c:\users\
vince\anaconda3\lib\site-packages (from tensorflow) (2.15.0)
Requirement already satisfied: absl-py>=1.0.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (2.1.0)
Requirement already satisfied: astunparse>=1.6.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=23.5.26 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (23.5.26)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in
c:\users\vince\anaconda3\lib\site-packages (from tensorflow-
intel==2.15.0->tensorflow) (0.5.4)
Requirement already satisfied: google-pasta>=0.1.1 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (0.2.0)
Requirement already satisfied: h5py>=2.9.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (3.7.0)
Requirement already satisfied: libclang>=13.0.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (16.0.6)
Requirement already satisfied: ml-dtypes~=0.2.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (0.2.0)
Requirement already satisfied: numpy<2.0.0,>=1.23.5 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (1.24.3)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (3.3.0)
Requirement already satisfied: packaging in c:\users\vince\anaconda3\
lib\site-packages (from tensorflow-intel==2.15.0->tensorflow) (23.0)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!
=4.21.3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (4.25.2)
Requirement already satisfied: setuptools in c:\users\vince\anaconda3\
lib\site-packages (from tensorflow-intel==2.15.0->tensorflow) (68.0.0)
Requirement already satisfied: six>=1.12.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (2.4.0)
Requirement already satisfied: typing-extensions>=3.6.6 in c:\users\
vince\anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (4.7.1)
Requirement already satisfied: wrapt<1.15,>=1.11.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (1.14.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in
c:\users\vince\anaconda3\lib\site-packages (from tensorflow-
intel==2.15.0->tensorflow) (0.31.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (1.60.1)
Requirement already satisfied: tensorboard<2.16,>=2.15 in c:\users\
vince\anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (2.15.2)
Requirement already satisfied: tensorflow-estimator<2.16,>=2.15.0 in
c:\users\vince\anaconda3\lib\site-packages (from tensorflow-
intel==2.15.0->tensorflow) (2.15.0)
Requirement already satisfied: keras<2.16,>=2.15.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (2.15.0)
Requirement already satisfied: scipy>=1.5.0 in c:\users\vince\
anaconda3\lib\site-packages (from scikit-learn) (1.10.1)
Requirement already satisfied: joblib>=1.1.1 in c:\users\vince\
anaconda3\lib\site-packages (from scikit-learn) (1.2.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\vince\
anaconda3\lib\site-packages (from scikit-learn) (2.2.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in c:\users\vince\
anaconda3\lib\site-packages (from astunparse>=1.6.0->tensorflow-
intel==2.15.0->tensorflow) (0.38.4)
Requirement already satisfied: google-auth<3,>=1.6.3 in c:\users\
vince\anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15-
>tensorflow-intel==2.15.0->tensorflow) (2.27.0)
```

```
Requirement already satisfied: google-auth-oauthlib<2,>=0.5 in c:\
users\vince\anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15-
>tensorflow-intel==2.15.0->tensorflow) (1.2.0)
Requirement already satisfied: markdown>=2.6.8 in c:\users\vince\
anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15->tensorflow-
intel==2.15.0->tensorflow) (3.4.1)
Requirement already satisfied: requests<3,>=2.21.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15->tensorflow-
intel==2.15.0->tensorflow) (2.31.0)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0
in c:\users\vince\anaconda3\lib\site-packages (from
tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow) (0.7.2)
Requirement already satisfied: werkzeug>=1.0.1 in c:\users\vince\
anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15->tensorflow-
intel==2.15.0->tensorflow) (2.2.3)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in c:\users\
vince\anaconda3\lib\site-packages (from google-auth<3,>=1.6.3-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(5.3.2)
Requirement already satisfied: pyasn1-modules>=0.2.1 in c:\users\
vince\anaconda3\lib\site-packages (from google-auth<3,>=1.6.3-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(0.2.8)
Requirement already satisfied: rsa<5,>=3.1.4 in c:\users\vince\
anaconda3\lib\site-packages (from google-auth<3,>=1.6.3-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow) (4.9)
Requirement already satisfied: requests-oauthlib>=0.7.0 in c:\users\
vince\anaconda3\lib\site-packages (from google-auth-oauthlib<2,>=0.5-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(1.3.1)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\
vince\anaconda3\lib\site-packages (from requests<3,>=2.21.0-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\vince\
anaconda3\lib\site-packages (from requests<3,>=2.21.0-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\vince\
anaconda3\lib\site-packages (from requests<3,>=2.21.0-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\vince\
anaconda3\lib\site-packages (from requests<3,>=2.21.0-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(2023.7.22)
Requirement already satisfied: MarkupSafe>=2.1.1 in c:\users\vince\
anaconda3\lib\site-packages (from werkzeug>=1.0.1-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(2.1.1)
```

```
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in c:\users\vince\
anaconda3\lib\site-packages (from pyasn1-modules>=0.2.1->google-
auth<3,>=1.6.3->tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0-
>tensorflow) (0.4.8)
Requirement already satisfied: oauthlib>=3.0.0 in c:\users\vince\
anaconda3\lib\site-packages (from requests-oauthlib>=0.7.0->google-
auth-oauthlib<2,>=0.5->tensorboard<2.16,>=2.15->tensorflow-
intel==2.15.0->tensorflow) (3.2.2)

Requirement already satisfied: tensorflow in c:\users\vince\anaconda3\
lib\site-packages (2.15.0)
Requirement already satisfied: keras in c:\users\vince\anaconda3\lib\
site-packages (2.15.0)
Collecting keras
  Obtaining dependency information for keras from
https://files.pythonhosted.org/packages/b0/b2/104733bb67fde86f3d10010f
0b5c93cfa1d5bf552f904584cf9e5b3ba719/keras-3.0.5-py3-none-
any.whl.metadata
  Using cached keras-3.0.5-py3-none-any.whl.metadata (4.8 kB)
Requirement already satisfied: tensorflow-intel==2.15.0 in c:\users\
vince\anaconda3\lib\site-packages (from tensorflow) (2.15.0)
Requirement already satisfied: absl-py>=1.0.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (2.1.0)
Requirement already satisfied: astunparse>=1.6.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=23.5.26 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (23.5.26)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in
c:\users\vince\anaconda3\lib\site-packages (from tensorflow-
intel==2.15.0->tensorflow) (0.5.4)
Requirement already satisfied: google-pasta>=0.1.1 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (0.2.0)
Requirement already satisfied: h5py>=2.9.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (3.7.0)
Requirement already satisfied: libclang>=13.0.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (16.0.6)
Requirement already satisfied: ml-dtypes~=0.2.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (0.2.0)
Requirement already satisfied: numpy<2.0.0,>=1.23.5 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (1.24.3)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (3.3.0)
Requirement already satisfied: packaging in c:\users\vince\anaconda3\
lib\site-packages (from tensorflow-intel==2.15.0->tensorflow) (23.0)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!
=4.21.3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (4.25.2)
Requirement already satisfied: setuptools in c:\users\vince\anaconda3\
lib\site-packages (from tensorflow-intel==2.15.0->tensorflow) (68.0.0)
Requirement already satisfied: six>=1.12.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (2.4.0)
Requirement already satisfied: typing-extensions>=3.6.6 in c:\users\
vince\anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (4.7.1)
Requirement already satisfied: wrapt<1.15,>=1.11.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (1.14.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in
c:\users\vince\anaconda3\lib\site-packages (from tensorflow-
intel==2.15.0->tensorflow) (0.31.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (1.60.1)
Requirement already satisfied: tensorboard<2.16,>=2.15 in c:\users\
vince\anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow) (2.15.2)
Requirement already satisfied: tensorflow-estimator<2.16,>=2.15.0 in
c:\users\vince\anaconda3\lib\site-packages (from tensorflow-
intel==2.15.0->tensorflow) (2.15.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in c:\users\vince\
anaconda3\lib\site-packages (from astunparse>=1.6.0->tensorflow-
intel==2.15.0->tensorflow) (0.38.4)
Requirement already satisfied: google-auth<3,>=1.6.3 in c:\users\
vince\anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15-
>tensorflow-intel==2.15.0->tensorflow) (2.27.0)
Requirement already satisfied: google-auth-oauthlib<2,>=0.5 in c:\
users\vince\anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15-
>tensorflow-intel==2.15.0->tensorflow) (1.2.0)
Requirement already satisfied: markdown>=2.6.8 in c:\users\vince\
anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15->tensorflow-
intel==2.15.0->tensorflow) (3.4.1)
Requirement already satisfied: requests<3,>=2.21.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15->tensorflow-
intel==2.15.0->tensorflow) (2.31.0)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0
```

```
in c:\users\vince\anaconda3\lib\site-packages (from
tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow) (0.7.2)
Requirement already satisfied: werkzeug>=1.0.1 in c:\users\vince\
anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15->tensorflow-
intel==2.15.0->tensorflow) (2.2.3)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in c:\users\
vince\anaconda3\lib\site-packages (from google-auth<3,>=1.6.3-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(5.3.2)
Requirement already satisfied: pyasn1-modules>=0.2.1 in c:\users\
vince\anaconda3\lib\site-packages (from google-auth<3,>=1.6.3-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(0.2.8)
Requirement already satisfied: rsa<5,>=3.1.4 in c:\users\vince\
anaconda3\lib\site-packages (from google-auth<3,>=1.6.3-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow) (4.9)
Requirement already satisfied: requests-oauthlib>=0.7.0 in c:\users\
vince\anaconda3\lib\site-packages (from google-auth-oauthlib<2,>=0.5-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(1.3.1)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\
vince\anaconda3\lib\site-packages (from requests<3,>=2.21.0-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\vince\
anaconda3\lib\site-packages (from requests<3,>=2.21.0-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\vince\
anaconda3\lib\site-packages (from requests<3,>=2.21.0-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\vince\
anaconda3\lib\site-packages (from requests<3,>=2.21.0-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(2023.7.22)
Requirement already satisfied: MarkupSafe>=2.1.1 in c:\users\vince\
anaconda3\lib\site-packages (from werkzeug>=1.0.1-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow)
(2.1.1)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in c:\users\vince\
anaconda3\lib\site-packages (from pyasn1-modules>=0.2.1->google-
auth<3,>=1.6.3->tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0-
>tensorflow) (0.4.8)
Requirement already satisfied: oauthlib>=3.0.0 in c:\users\vince\
anaconda3\lib\site-packages (from requests-oauthlib>=0.7.0->google-
auth-oauthlib<2,>=0.5->tensorboard<2.16,>=2.15->tensorflow-
intel==2.15.0->tensorflow) (3.2.2)
Requirement already satisfied: adapt in c:\users\vince\anaconda3\lib\
site-packages (0.4.4)
```

```
Requirement already satisfied: numpy in c:\users\vince\anaconda3\lib\
site-packages (from adapt) (1.24.3)
Requirement already satisfied: scipy in c:\users\vince\anaconda3\lib\
site-packages (from adapt) (1.10.1)
Requirement already satisfied: tensorflow in c:\users\vince\anaconda3\
lib\site-packages (from adapt) (2.15.0)
Requirement already satisfied: scikit-learn in c:\users\vince\
anaconda3\lib\site-packages (from adapt) (1.3.0)
Requirement already satisfied: cvxopt in c:\users\vince\anaconda3\lib\
site-packages (from adapt) (1.3.2)
Requirement already satisfied: scikeras in c:\users\vince\anaconda3\
lib\site-packages (from adapt) (0.12.0)
Requirement already satisfied: packaging>=0.21 in c:\users\vince\
anaconda3\lib\site-packages (from scikeras->adapt) (23.0)
Requirement already satisfied: tensorflow-io-gcs-
filesystem<0.32,>=0.23.1 in c:\users\vince\anaconda3\lib\site-packages
(from scikeras->adapt) (0.31.0)
Requirement already satisfied: joblib>=1.1.1 in c:\users\vince\
anaconda3\lib\site-packages (from scikit-learn->adapt) (1.2.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\vince\
anaconda3\lib\site-packages (from scikit-learn->adapt) (2.2.0)
Requirement already satisfied: tensorflow-intel==2.15.0 in c:\users\
vince\anaconda3\lib\site-packages (from tensorflow->adapt) (2.15.0)
Requirement already satisfied: absl-py>=1.0.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow->adapt) (2.1.0)
Requirement already satisfied: astunparse>=1.6.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow->adapt) (1.6.3)
Requirement already satisfied: flatbuffers>=23.5.26 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow->adapt) (23.5.26)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in
c:\users\vince\anaconda3\lib\site-packages (from tensorflow-
intel==2.15.0->tensorflow->adapt) (0.5.4)
Requirement already satisfied: google-pasta>=0.1.1 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow->adapt) (0.2.0)
Requirement already satisfied: h5py>=2.9.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow->adapt) (3.7.0)
Requirement already satisfied: libclang>=13.0.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow->adapt) (16.0.6)
Requirement already satisfied: ml-dtypes~=0.2.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow->adapt) (0.2.0)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow->adapt) (3.3.0)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!
=4.21.3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow->adapt) (4.25.2)
Requirement already satisfied: setuptools in c:\users\vince\anaconda3\
lib\site-packages (from tensorflow-intel==2.15.0->tensorflow->adapt)
(68.0.0)
Requirement already satisfied: six>=1.12.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow->adapt) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow->adapt) (2.4.0)
Requirement already satisfied: typing-extensions>=3.6.6 in c:\users\
vince\anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow->adapt) (4.7.1)
Requirement already satisfied: wrapt<1.15,>=1.11.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow->adapt) (1.14.1)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow->adapt) (1.60.1)
Requirement already satisfied: tensorboard<2.16,>=2.15 in c:\users\
vince\anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow->adapt) (2.15.2)
Requirement already satisfied: tensorflow-estimator<2.16,>=2.15.0 in
c:\users\vince\anaconda3\lib\site-packages (from tensorflow-
intel==2.15.0->tensorflow->adapt) (2.15.0)
Requirement already satisfied: keras<2.16,>=2.15.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorflow-intel==2.15.0-
>tensorflow->adapt) (2.15.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in c:\users\vince\
anaconda3\lib\site-packages (from astunparse>=1.6.0->tensorflow-
intel==2.15.0->tensorflow->adapt) (0.38.4)
Requirement already satisfied: google-auth<3,>=1.6.3 in c:\users\
vince\anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15-
>tensorflow-intel==2.15.0->tensorflow->adapt) (2.27.0)
Requirement already satisfied: google-auth-oauthlib<2,>=0.5 in c:\
users\vince\anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15-
>tensorflow-intel==2.15.0->tensorflow->adapt) (1.2.0)
Requirement already satisfied: markdown>=2.6.8 in c:\users\vince\
anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15->tensorflow-
intel==2.15.0->tensorflow->adapt) (3.4.1)
Requirement already satisfied: requests<3,>=2.21.0 in c:\users\vince\
anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15->tensorflow-
intel==2.15.0->tensorflow->adapt) (2.31.0)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0
in c:\users\vince\anaconda3\lib\site-packages (from
tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow->adapt)
(0.7.2)
Requirement already satisfied: werkzeug>=1.0.1 in c:\users\vince\
anaconda3\lib\site-packages (from tensorboard<2.16,>=2.15->tensorflow-
intel==2.15.0->tensorflow->adapt) (2.2.3)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in c:\users\
vince\anaconda3\lib\site-packages (from google-auth<3,>=1.6.3-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow->adapt)
(5.3.2)
Requirement already satisfied: pyasn1-modules>=0.2.1 in c:\users\
vince\anaconda3\lib\site-packages (from google-auth<3,>=1.6.3-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow->adapt)
(0.2.8)
Requirement already satisfied: rsa<5,>=3.1.4 in c:\users\vince\
anaconda3\lib\site-packages (from google-auth<3,>=1.6.3-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow->adapt)
(4.9)
Requirement already satisfied: requests-oauthlib>=0.7.0 in c:\users\
vince\anaconda3\lib\site-packages (from google-auth-oauthlib<2,>=0.5-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow->adapt)
(1.3.1)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\
vince\anaconda3\lib\site-packages (from requests<3,>=2.21.0-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow->adapt)
(2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\vince\
anaconda3\lib\site-packages (from requests<3,>=2.21.0-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow->adapt)
(3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\vince\
anaconda3\lib\site-packages (from requests<3,>=2.21.0-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow->adapt)
(1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\vince\
anaconda3\lib\site-packages (from requests<3,>=2.21.0-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow->adapt)
(2023.7.22)
Requirement already satisfied: MarkupSafe>=2.1.1 in c:\users\vince\
anaconda3\lib\site-packages (from werkzeug>=1.0.1-
>tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0->tensorflow->adapt)
(2.1.1)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in c:\users\vince\
anaconda3\lib\site-packages (from pyasn1-modules>=0.2.1->google-
auth<3,>=1.6.3->tensorboard<2.16,>=2.15->tensorflow-intel==2.15.0-
>tensorflow->adapt) (0.4.8)
Requirement already satisfied: oauthlib>=3.0.0 in c:\users\vince\
anaconda3\lib\site-packages (from requests-oauthlib>=0.7.0->google-
auth-oauthlib<2,>=0.5->tensorboard<2.16,>=2.15->tensorflow-
intel==2.15.0->tensorflow->adapt) (3.2.2)
```

```
Requirement already satisfied: scikeras in c:\users\vince\anaconda3\
lib\site-packages (0.12.0)
Requirement already satisfied: packaging>=0.21 in c:\users\vince\
anaconda3\lib\site-packages (from scikeras) (23.0)
Requirement already satisfied: scikit-learn>=1.0.0 in c:\users\vince\
anaconda3\lib\site-packages (from scikeras) (1.3.0)
Requirement already satisfied: tensorflow-io-gcs-
filesystem<0.32,>=0.23.1 in c:\users\vince\anaconda3\lib\site-packages
(from scikeras) (0.31.0)
Requirement already satisfied: numpy>=1.17.3 in c:\users\vince\
anaconda3\lib\site-packages (from scikit-learn>=1.0.0->scikeras)
(1.24.3)
Requirement already satisfied: scipy>=1.5.0 in c:\users\vince\
anaconda3\lib\site-packages (from scikit-learn>=1.0.0->scikeras)
(1.10.1)
Requirement already satisfied: joblib>=1.1.1 in c:\users\vince\
anaconda3\lib\site-packages (from scikit-learn>=1.0.0->scikeras)
(1.2.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\vince\
anaconda3\lib\site-packages (from scikit-learn>=1.0.0->scikeras)
(2.2.0)
Requirement already satisfied: keras-tuner in c:\users\vince\
anaconda3\lib\site-packages (1.4.6)
Requirement already satisfied: keras in c:\users\vince\anaconda3\lib\
site-packages (from keras-tuner) (2.15.0)
Requirement already satisfied: packaging in c:\users\vince\anaconda3\
lib\site-packages (from keras-tuner) (23.0)
Requirement already satisfied: requests in c:\users\vince\anaconda3\
lib\site-packages (from keras-tuner) (2.31.0)
Requirement already satisfied: kt-legacy in c:\users\vince\anaconda3\
lib\site-packages (from keras-tuner) (1.0.5)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\
vince\anaconda3\lib\site-packages (from requests->keras-tuner) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\vince\
anaconda3\lib\site-packages (from requests->keras-tuner) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\vince\
anaconda3\lib\site-packages (from requests->keras-tuner) (1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\vince\
anaconda3\lib\site-packages (from requests->keras-tuner) (2023.7.22)

# import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import re
import seaborn as sns
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.utils import pad_sequences
from sklearn.linear_model import LinearRegression
```

```python
import warnings
warnings.filterwarnings('ignore')
```

```
WARNING:tensorflow:From C:\Users\vince\anaconda3\Lib\site-packages\
keras\src\losses.py:2976: The name
tf.losses.sparse_softmax_cross_entropy is deprecated. Please use
tf.compat.v1.losses.sparse_softmax_cross_entropy instead.
```

```python
# read in imdb data
df = pd.read_csv('imdb_labelled.txt', sep='	', engine='python',
header=None, names=['review', 'sentiment'])
df.head()
```

```
                                              review  sentiment
0  A very, very, very slow-moving, aimless movie ...          0
1  Not sure who was more lost - the flat characte...          0
2  Attempting artiness with black & white and cle...          0
3       Very little music or anything to speak of.            0
4  The best scene in the movie was when Gerardo i...          1
```

```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 2 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   review     1000 non-null   object
 1   sentiment  1000 non-null   int64 
dtypes: int64(1), object(1)
memory usage: 15.8+ KB
```

```python
df.shape
```

```
(1000, 2)
```

```python
df.groupby(['sentiment'])[['sentiment']].count()
```

```
           sentiment
sentiment           
0                500
1                500
```

```python
# checking for NULL values
print("Null Values:\n", df.isna().sum())

# dropping null values
df = df.dropna()
```

```
# verifying null values were dropped
print("Null Values after dropping:\n", df.isna().sum())
```

```
Null Values:
 review       0
sentiment    0
dtype: int64
Null Values after dropping:
 review       0
sentiment    0
dtype: int64
```

```
# checking for duplicate values
print("Duplicate Values:\n", df.duplicated().sum())

# dropping duplicate values
df = df.drop_duplicates()

# verifying duplicates were dropped
print("Duplicate Values after dropping:\n", df.duplicated().sum())
```

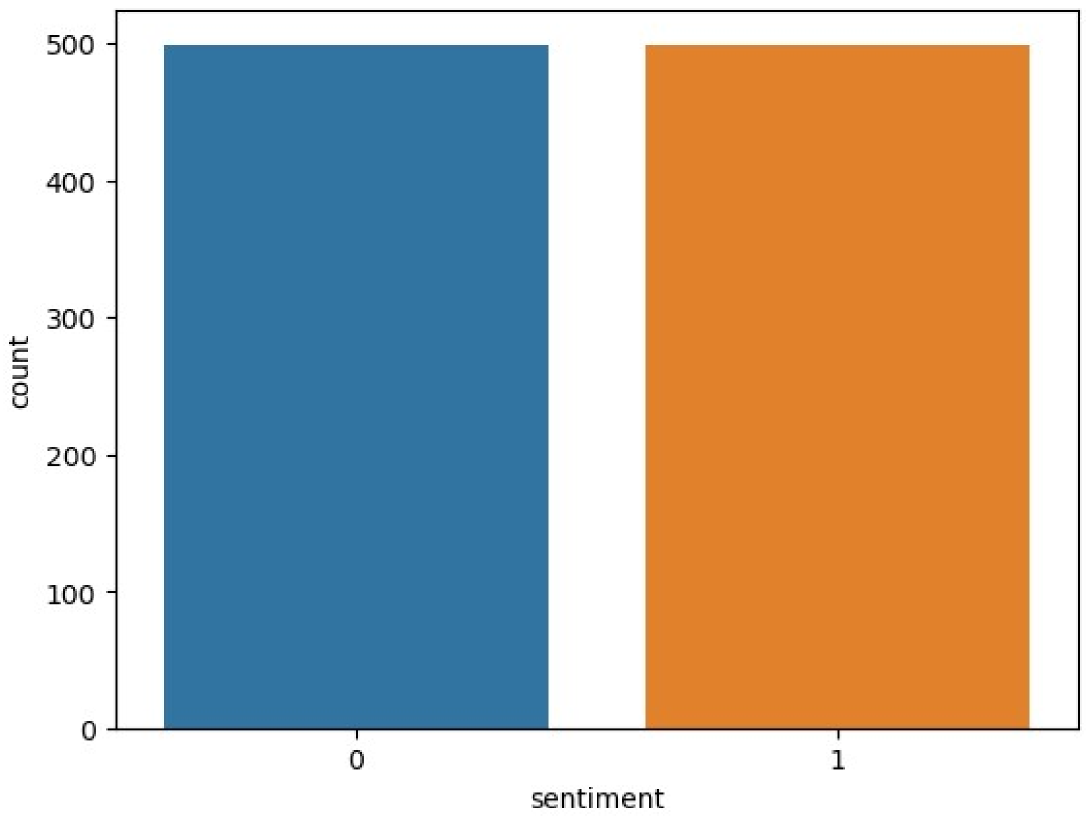
```
Duplicate Values:
 3
Duplicate Values after dropping:
 0
```

```
df.shape
```

```
(997, 2)
```

```
sns.countplot(df, x='sentiment')
```

```
<Axes: xlabel='sentiment', ylabel='count'>
```



```
df.describe()

       sentiment
count  997.000000
mean     0.499498
std      0.500251
min      0.000000
25%      0.000000
50%      0.000000
75%      1.000000
max      1.000000

# adding column for word count
def no_of_words(text):
    words= text.split()
    word_count = len(words)
    return word_count

df['word_count'] = df['review'].apply(no_of_words)

df.head()

                                              review  sentiment
word_count
0  A very, very, very slow-moving, aimless movie ...          0
```

```
13
1  Not sure who was more lost - the flat characte...          0
19
2  Attempting artiness with black & white and cle...          0
31
3         Very little music or anything to speak of.          0
8
4  The best scene in the movie was when Gerardo i...          1
21
```

I chose to add a word_count column in order to see if reviews that had negative sentiment had more or less words.

```
pos_reviews =  df[df.sentiment == 1]
neg_reviews =  df[df.sentiment == 0]
```

```
from collections import Counter
count = Counter()
for text in pos_reviews['review'].values:
    for word in text.split():
        count[word] +=1
count.most_common(15)
```

```
[('the', 357),
 ('and', 250),
 ('a', 236),
 ('of', 203),
 ('is', 165),
 ('I', 138),
 ('to', 129),
 ('in', 106),
 ('this', 101),
 ('The', 82),
 ('was', 80),
 ('film', 68),
 ('that', 66),
 ('movie', 64),
 ('it', 59)]
```

```
pos_words = pd.DataFrame(count.most_common(15))
pos_words.columns = ['word', 'count']
pos_words.head()
```

```
  word  count
0  the    357
1  and    250
2    a    236
3   of    203
4   is    165
```

```python
import plotly.express as px
px.bar(pos_words, x='count', y='word', title='Common words in positive
reviews', color = 'word')
```

```
{"config":{"plotlyServerURL":"https://plot.ly"},"data":
[{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"the","marker":
{"color":"#636efa","pattern":
{"shape":""}},"name":"the","offsetgroup":"the","orientation":"h","show
legend":true,"textposition":"auto","type":"bar","x":
[357],"xaxis":"x","y":["the"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"and","marker":
{"color":"#EF553B","pattern":
{"shape":""}},"name":"and","offsetgroup":"and","orientation":"h","show
legend":true,"textposition":"auto","type":"bar","x":
[250],"xaxis":"x","y":["and"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"a","marker":
{"color":"#00cc96","pattern":
{"shape":""}},"name":"a","offsetgroup":"a","orientation":"h","showlege
nd":true,"textposition":"auto","type":"bar","x":[236],"xaxis":"x","y":
["a"],"yaxis":"y"},{"alignmentgroup":"True","hovertemplate":"word=%
{y}<br>count=%{x}<extra></extra>","legendgroup":"of","marker":
{"color":"#ab63fa","pattern":
{"shape":""}},"name":"of","offsetgroup":"of","orientation":"h","showle
gend":true,"textposition":"auto","type":"bar","x":
[203],"xaxis":"x","y":["of"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"is","marker":
{"color":"#FFA15A","pattern":
{"shape":""}},"name":"is","offsetgroup":"is","orientation":"h","showle
gend":true,"textposition":"auto","type":"bar","x":
[165],"xaxis":"x","y":["is"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"I","marker":
{"color":"#19d3f3","pattern":
{"shape":""}},"name":"I","offsetgroup":"I","orientation":"h","showlege
nd":true,"textposition":"auto","type":"bar","x":[138],"xaxis":"x","y":
["I"],"yaxis":"y"},{"alignmentgroup":"True","hovertemplate":"word=%
{y}<br>count=%{x}<extra></extra>","legendgroup":"to","marker":
{"color":"#FF6692","pattern":
{"shape":""}},"name":"to","offsetgroup":"to","orientation":"h","showle
gend":true,"textposition":"auto","type":"bar","x":
[129],"xaxis":"x","y":["to"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"in","marker":
{"color":"#B6E880","pattern":
{"shape":""}},"name":"in","offsetgroup":"in","orientation":"h","showle
gend":true,"textposition":"auto","type":"bar","x":
```

```
[106],"xaxis":"x","y":["in"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"this","marker":
{"color":"#FF97FF","pattern":
{"shape":""}},"name":"this","offsetgroup":"this","orientation":"h","sh
owlegend":true,"textposition":"auto","type":"bar","x":
[101],"xaxis":"x","y":["this"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"The","marker":
{"color":"#FECB52","pattern":
{"shape":""}},"name":"The","offsetgroup":"The","orientation":"h","show
legend":true,"textposition":"auto","type":"bar","x":
[82],"xaxis":"x","y":["The"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"was","marker":
{"color":"#636efa","pattern":
{"shape":""}},"name":"was","offsetgroup":"was","orientation":"h","show
legend":true,"textposition":"auto","type":"bar","x":
[80],"xaxis":"x","y":["was"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"film","marker":
{"color":"#EF553B","pattern":
{"shape":""}},"name":"film","offsetgroup":"film","orientation":"h","sh
owlegend":true,"textposition":"auto","type":"bar","x":
[68],"xaxis":"x","y":["film"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"that","marker":
{"color":"#00cc96","pattern":
{"shape":""}},"name":"that","offsetgroup":"that","orientation":"h","sh
owlegend":true,"textposition":"auto","type":"bar","x":
[66],"xaxis":"x","y":["that"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"movie","marker":
{"color":"#ab63fa","pattern":
{"shape":""}},"name":"movie","offsetgroup":"movie","orientation":"h","
showlegend":true,"textposition":"auto","type":"bar","x":
[64],"xaxis":"x","y":["movie"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"it","marker":
{"color":"#FFA15A","pattern":
{"shape":""}},"name":"it","offsetgroup":"it","orientation":"h","showle
gend":true,"textposition":"auto","type":"bar","x":
[59],"xaxis":"x","y":["it"],"yaxis":"y"}],"layout":
{"barmode":"relative","legend":{"title":
{"text":"word"},"tracegroupgap":0},"template":{"data":{"bar":
[{"error_x":{"color":"#2a3f5f"},"error_y":
{"color":"#2a3f5f"},"marker":{"line":
{"color":"#E5ECF6","width":0.5},"pattern":
{"fillmode":"overlay","size":10,"solidity":0.2}},"type":"bar"}],"barpo
```

```
lar":[{"marker":{"line":{"color":"#E5ECF6","width":0.5},"pattern":
{"fillmode":"overlay","size":10,"solidity":0.2}},"type":"barpolar"}],"
carpet":[{"aaxis":
{"endlinecolor":"#2a3f5f","gridcolor":"white","linecolor":"white","min
orgridcolor":"white","startlinecolor":"#2a3f5f"},"baxis":
{"endlinecolor":"#2a3f5f","gridcolor":"white","linecolor":"white","min
orgridcolor":"white","startlinecolor":"#2a3f5f"},"type":"carpet"}],"ch
oropleth":[{"colorbar":
{"outlinewidth":0,"ticks":""},"type":"choropleth"}],"contour":
[{"colorbar":{"outlinewidth":0,"ticks":""},"colorscale":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"contour"}],"contourcarpet":[{"colorbar":
{"outlinewidth":0,"ticks":""},"type":"contourcarpet"}],"heatmap":
[{"colorbar":{"outlinewidth":0,"ticks":""},"colorscale":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"heatmap"}],"heatmapgl":[{"colorbar":
{"outlinewidth":0,"ticks":""},"colorscale":[[0,"#0d0887"],
[0.1111111111111111,"#46039f"],[0.2222222222222222,"#7201a8"],
[0.3333333333333333,"#9c179e"],[0.4444444444444444,"#bd3786"],
[0.5555555555555556,"#d8576b"],[0.6666666666666666,"#ed7953"],
[0.7777777777777778,"#fb9f3a"],[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"heatmapgl"}],"histogram":[{"marker":{"pattern":
{"fillmode":"overlay","size":10,"solidity":0.2}},"type":"histogram"}],
"histogram2d":[{"colorbar":{"outlinewidth":0,"ticks":""},"colorscale":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"histogram2d"}],"histogram2dcontour":
[{"colorbar":{"outlinewidth":0,"ticks":""},"colorscale":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"histogram2dcontour"}],"mesh3d":[{"colorbar":
{"outlinewidth":0,"ticks":""},"type":"mesh3d"}],"parcoords":[{"line":
{"colorbar":{"outlinewidth":0,"ticks":""}},"type":"parcoords"}],"pie":
[{"automargin":true,"type":"pie"}],"scatter":[{"fillpattern":
{"fillmode":"overlay","size":10,"solidity":0.2},"type":"scatter"}],"sc
```

```
atter3d":[{"line":{"colorbar":{"outlinewidth":0,"ticks":""}},"marker":
{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scatter3d"}],"scattercarpet":
[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scattercarpet"}],"scattergeo":
[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scattergeo"}],"scattergl":
[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scattergl"}],"scattermapbox":
[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scattermapbox"}],"scatterpolar"
:[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scatterpolar"}],"scatterpolargl
":[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scatterpolargl"}],"scatterterna
ry":[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scatterternary"}],"surface":
[{"colorbar":{"outlinewidth":0,"ticks":""},"colorscale":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"surface"}],"table":[{"cells":{"fill":
{"color":"#EBF0F8"},"line":{"color":"white"}},"header":{"fill":
{"color":"#C8D4E3"},"line":
{"color":"white"}},"type":"table"}]},"layout":{"annotationdefaults":
{"arrowcolor":"#2a3f5f","arrowhead":0,"arrowwidth":1},"autotypenumbers
":"strict","coloraxis":{"colorbar":
{"outlinewidth":0,"ticks":""}},"colorscale":{"diverging":
[[0,"#8e0152"],[0.1,"#c51b7d"],[0.2,"#de77ae"],[0.3,"#f1b6da"],
[0.4,"#fde0ef"],[0.5,"#f7f7f7"],[0.6,"#e6f5d0"],[0.7,"#b8e186"],
[0.8,"#7fbc41"],[0.9,"#4d9221"],[1,"#276419"]],"sequential":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],[1,"#f0f921"]],"sequentialminus":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],[1,"#f0f921"]]},"colorway":
["#636efa","#EF553B","#00cc96","#ab63fa","#FFA15A","#19d3f3","#FF6692"
,"#B6E880","#FF97FF","#FECB52"],"font":{"color":"#2a3f5f"},"geo":
{"bgcolor":"white","lakecolor":"white","landcolor":"#E5ECF6","showlake
s":true,"showland":true,"subunitcolor":"white"},"hoverlabel":
{"align":"left"},"hovermode":"closest","mapbox":
{"style":"light"},"paper_bgcolor":"white","plot_bgcolor":"#E5ECF6","po
```

```
lar":{"angularaxis":
{"gridcolor":"white","linecolor":"white","ticks":""},"bgcolor":"#E5ECF
6","radialaxis":
{"gridcolor":"white","linecolor":"white","ticks":""}},"scene":
{"xaxis":
{"backgroundcolor":"#E5ECF6","gridcolor":"white","gridwidth":2,"lineco
lor":"white","showbackground":true,"ticks":"","zerolinecolor":"white"}
,"yaxis":
{"backgroundcolor":"#E5ECF6","gridcolor":"white","gridwidth":2,"lineco
lor":"white","showbackground":true,"ticks":"","zerolinecolor":"white"}
,"zaxis":
{"backgroundcolor":"#E5ECF6","gridcolor":"white","gridwidth":2,"lineco
lor":"white","showbackground":true,"ticks":"","zerolinecolor":"white"}
},"shapedefaults":{"line":{"color":"#2a3f5f"}},"ternary":{"aaxis":
{"gridcolor":"white","linecolor":"white","ticks":""},"baxis":
{"gridcolor":"white","linecolor":"white","ticks":""},"bgcolor":"#E5ECF
6","caxis":
{"gridcolor":"white","linecolor":"white","ticks":""}},"title":
{"x":5.0e-2},"xaxis":
{"automargin":true,"gridcolor":"white","linecolor":"white","ticks":"",
"title":
{"standoff":15},"zerolinecolor":"white","zerolinewidth":2},"yaxis":
{"automargin":true,"gridcolor":"white","linecolor":"white","ticks":"",
"title":
{"standoff":15},"zerolinecolor":"white","zerolinewidth":2}}},"title":
{"text":"Common words in positive reviews"},"xaxis":
{"anchor":"y","domain":[0,1],"title":{"text":"count"}},"yaxis":
{"anchor":"x","categoryarray":
["it","movie","that","film","was","The","this","in","to","I","is","of"
,"a","and","the"],"categoryorder":"array","domain":[0,1],"title":
{"text":"word"}}}}
```

```
for text in neg_reviews['review'].values:
    for word in text.split():
        count[word] +=1
count.most_common(15)
```

```
[('the', 655),
 ('a', 411),
 ('and', 410),
 ('of', 368),
 ('is', 323),
 ('I', 254),
 ('to', 244),
 ('this', 204),
 ('in', 191),
 ('The', 185),
 ('was', 179),
 ('that', 145),
 ('it', 138),
 ('movie', 131),
 ('film', 107)]
```

```
neg_words = pd.DataFrame(count.most_common(15))
neg_words.columns = ['word', 'count']
neg_words.head()
```

```
  word  count
0  the    655
1    a    411
2  and    410
3   of    368
4   is    323
```

```
import plotly.express as px
px.bar(neg_words, x='count', y='word', title='Common words in negative
reviews', color = 'word')
```

{"config":{"plotlyServerURL":"https://plot.ly"},"data":
[{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"the","marker":
{"color":"#636efa","pattern":
{"shape":""}},"name":"the","offsetgroup":"the","orientation":"h","show
legend":true,"textposition":"auto","type":"bar","x":
[655],"xaxis":"x","y":["the"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"a","marker":
{"color":"#EF553B","pattern":
{"shape":""}},"name":"a","offsetgroup":"a","orientation":"h","showlege
nd":true,"textposition":"auto","type":"bar","x":[411],"xaxis":"x","y":
["a"],"yaxis":"y"},{"alignmentgroup":"True","hovertemplate":"word=%
{y}<br>count=%{x}<extra></extra>","legendgroup":"and","marker":
{"color":"#00cc96","pattern":
{"shape":""}},"name":"and","offsetgroup":"and","orientation":"h","show
legend":true,"textposition":"auto","type":"bar","x":
[410],"xaxis":"x","y":["and"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"of","marker":
{"color":"#ab63fa","pattern":
{"shape":""}},"name":"of","offsetgroup":"of","orientation":"h","showle
gend":true,"textposition":"auto","type":"bar","x":
[368],"xaxis":"x","y":["of"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"is","marker":
{"color":"#FFA15A","pattern":
{"shape":""}},"name":"is","offsetgroup":"is","orientation":"h","showle
gend":true,"textposition":"auto","type":"bar","x":
[323],"xaxis":"x","y":["is"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"I","marker":

```
{"color":"#19d3f3","pattern":
{"shape":""}},"name":"I","offsetgroup":"I","orientation":"h","showlege
nd":true,"textposition":"auto","type":"bar","x":[254],"xaxis":"x","y":
["I"],"yaxis":"y"},{"alignmentgroup":"True","hovertemplate":"word=%
{y}<br>count=%{x}<extra></extra>","legendgroup":"to","marker":
{"color":"#FF6692","pattern":
{"shape":""}},"name":"to","offsetgroup":"to","orientation":"h","showle
gend":true,"textposition":"auto","type":"bar","x":
[244],"xaxis":"x","y":["to"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"this","marker":
{"color":"#B6E880","pattern":
{"shape":""}},"name":"this","offsetgroup":"this","orientation":"h","sh
owlegend":true,"textposition":"auto","type":"bar","x":
[204],"xaxis":"x","y":["this"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"in","marker":
{"color":"#FF97FF","pattern":
{"shape":""}},"name":"in","offsetgroup":"in","orientation":"h","showle
gend":true,"textposition":"auto","type":"bar","x":
[191],"xaxis":"x","y":["in"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"The","marker":
{"color":"#FECB52","pattern":
{"shape":""}},"name":"The","offsetgroup":"The","orientation":"h","show
legend":true,"textposition":"auto","type":"bar","x":
[185],"xaxis":"x","y":["The"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"was","marker":
{"color":"#636efa","pattern":
{"shape":""}},"name":"was","offsetgroup":"was","orientation":"h","show
legend":true,"textposition":"auto","type":"bar","x":
[179],"xaxis":"x","y":["was"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"that","marker":
{"color":"#EF553B","pattern":
{"shape":""}},"name":"that","offsetgroup":"that","orientation":"h","sh
owlegend":true,"textposition":"auto","type":"bar","x":
[145],"xaxis":"x","y":["that"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"it","marker":
{"color":"#00cc96","pattern":
{"shape":""}},"name":"it","offsetgroup":"it","orientation":"h","showle
gend":true,"textposition":"auto","type":"bar","x":
[138],"xaxis":"x","y":["it"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"movie","marker":
{"color":"#ab63fa","pattern":
{"shape":""}},"name":"movie","offsetgroup":"movie","orientation":"h","
```

```
showlegend":true,"textposition":"auto","type":"bar","x":
[131],"xaxis":"x","y":["movie"],"yaxis":"y"},
{"alignmentgroup":"True","hovertemplate":"word=%{y}<br>count=%
{x}<extra></extra>","legendgroup":"film","marker":
{"color":"#FFA15A","pattern":
{"shape":""}},"name":"film","offsetgroup":"film","orientation":"h","sh
owlegend":true,"textposition":"auto","type":"bar","x":
[107],"xaxis":"x","y":["film"],"yaxis":"y"}],"layout":
{"barmode":"relative","legend":{"title":
{"text":"word"},"tracegroupgap":0},"template":{"data":{"bar":
[{"error_x":{"color":"#2a3f5f"},"error_y":
{"color":"#2a3f5f"},"marker":{"line":
{"color":"#E5ECF6","width":0.5},"pattern":
{"fillmode":"overlay","size":10,"solidity":0.2}},"type":"bar"}],"barpo
lar":[{"marker":{"line":{"color":"#E5ECF6","width":0.5},"pattern":
{"fillmode":"overlay","size":10,"solidity":0.2}},"type":"barpolar"}],"
carpet":[{"aaxis":
{"endlinecolor":"#2a3f5f","gridcolor":"white","linecolor":"white","min
orgridcolor":"white","startlinecolor":"#2a3f5f"},"baxis":
{"endlinecolor":"#2a3f5f","gridcolor":"white","linecolor":"white","min
orgridcolor":"white","startlinecolor":"#2a3f5f"},"type":"carpet"}],"ch
oropleth":[{"colorbar":
{"outlinewidth":0,"ticks":""},"type":"choropleth"}],"contour":
[{"colorbar":{"outlinewidth":0,"ticks":""},"colorscale":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"contour"}],"contourcarpet":[{"colorbar":
{"outlinewidth":0,"ticks":""},"type":"contourcarpet"}],"heatmap":
[{"colorbar":{"outlinewidth":0,"ticks":""},"colorscale":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"heatmap"}],"heatmapgl":[{"colorbar":
{"outlinewidth":0,"ticks":""},"colorscale":[[0,"#0d0887"],
[0.1111111111111111,"#46039f"],[0.2222222222222222,"#7201a8"],
[0.3333333333333333,"#9c179e"],[0.4444444444444444,"#bd3786"],
[0.5555555555555556,"#d8576b"],[0.6666666666666666,"#ed7953"],
[0.7777777777777778,"#fb9f3a"],[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"heatmapgl"}],"histogram":[{"marker":{"pattern":
{"fillmode":"overlay","size":10,"solidity":0.2}},"type":"histogram"}],
"histogram2d":[{"colorbar":{"outlinewidth":0,"ticks":""},"colorscale":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"histogram2d"}],"histogram2dcontour":
[{"colorbar":{"outlinewidth":0,"ticks":""},"colorscale":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"histogram2dcontour"}],"mesh3d":[{"colorbar":
{"outlinewidth":0,"ticks":""},"type":"mesh3d"}],"parcoords":[{"line":
{"colorbar":{"outlinewidth":0,"ticks":""}},"type":"parcoords"}],"pie":
[{"automargin":true,"type":"pie"}],"scatter":[{"fillpattern":
{"fillmode":"overlay","size":10,"solidity":0.2},"type":"scatter"}],"sc
atter3d":[{"line":{"colorbar":{"outlinewidth":0,"ticks":""}},"marker":
{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scatter3d"}],"scattercarpet":
[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scattercarpet"}],"scattergeo":
[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scattergeo"}],"scattergl":
[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scattergl"}],"scattermapbox":
[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scattermapbox"}],"scatterpolar"
:[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scatterpolar"}],"scatterpolargl
":[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scatterpolargl"}],"scatterterna
ry":[{"marker":{"colorbar":
{"outlinewidth":0,"ticks":""}},"type":"scatterternary"}],"surface":
[{"colorbar":{"outlinewidth":0,"ticks":""},"colorscale":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],
[1,"#f0f921"]],"type":"surface"}],"table":[{"cells":{"fill":
{"color":"#EBF0F8"},"line":{"color":"white"}},"header":{"fill":
{"color":"#C8D4E3"},"line":
{"color":"white"}},"type":"table"}]},"layout":{"annotationdefaults":
{"arrowcolor":"#2a3f5f","arrowhead":0,"arrowwidth":1},"autotypenumbers
":"strict","coloraxis":{"colorbar":
{"outlinewidth":0,"ticks":""}},"colorscale":{"diverging":
[[0,"#8e0152"],[0.1,"#c51b7d"],[0.2,"#de77ae"],[0.3,"#f1b6da"],
[0.4,"#fde0ef"],[0.5,"#f7f7f7"],[0.6,"#e6f5d0"],[0.7,"#b8e186"],
[0.8,"#7fbc41"],[0.9,"#4d9221"],[1,"#276419"]],"sequential":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],[1,"#f0f921"]],"sequentialminus":
[[0,"#0d0887"],[0.1111111111111111,"#46039f"],
[0.2222222222222222,"#7201a8"],[0.3333333333333333,"#9c179e"],
[0.4444444444444444,"#bd3786"],[0.5555555555555556,"#d8576b"],
[0.6666666666666666,"#ed7953"],[0.7777777777777778,"#fb9f3a"],
[0.8888888888888888,"#fdca26"],[1,"#f0f921"]]},"colorway":
["#636efa","#EF553B","#00cc96","#ab63fa","#FFA15A","#19d3f3","#FF6692"
,"#B6E880","#FF97FF","#FECB52"],"font":{"color":"#2a3f5f"},"geo":
{"bgcolor":"white","lakecolor":"white","landcolor":"#E5ECF6","showlake
s":true,"showland":true,"subunitcolor":"white"},"hoverlabel":
{"align":"left"},"hovermode":"closest","mapbox":
{"style":"light"},"paper_bgcolor":"white","plot_bgcolor":"#E5ECF6","po
lar":{"angularaxis":
{"gridcolor":"white","linecolor":"white","ticks":""},"bgcolor":"#E5ECF
6","radialaxis":
{"gridcolor":"white","linecolor":"white","ticks":""}},"scene":
{"xaxis":
{"backgroundcolor":"#E5ECF6","gridcolor":"white","gridwidth":2,"lineco
lor":"white","showbackground":true,"ticks":"","zerolinecolor":"white"}
,"yaxis":
{"backgroundcolor":"#E5ECF6","gridcolor":"white","gridwidth":2,"lineco
lor":"white","showbackground":true,"ticks":"","zerolinecolor":"white"}
,"zaxis":
{"backgroundcolor":"#E5ECF6","gridcolor":"white","gridwidth":2,"lineco
lor":"white","showbackground":true,"ticks":"","zerolinecolor":"white"}
},"shapedefaults":{"line":{"color":"#2a3f5f"}},"ternary":{"aaxis":
{"gridcolor":"white","linecolor":"white","ticks":""},"baxis":
{"gridcolor":"white","linecolor":"white","ticks":""},"bgcolor":"#E5ECF
6","caxis":
{"gridcolor":"white","linecolor":"white","ticks":""}},"title":
{"x":5.0e-2},"xaxis":
{"automargin":true,"gridcolor":"white","linecolor":"white","ticks":"",
"title":
{"standoff":15},"zerolinecolor":"white","zerolinewidth":2},"yaxis":
{"automargin":true,"gridcolor":"white","linecolor":"white","ticks":"",
"title":
{"standoff":15},"zerolinecolor":"white","zerolinewidth":2}}},"title":
{"text":"Common words in negative reviews"},"xaxis":
{"anchor":"y","domain":[0,1],"title":{"text":"count"}},"yaxis":
{"anchor":"x","categoryarray":
["film","movie","it","that","was","The","in","this","to","I","is","of"
,"and","a","the"],"categoryorder":"array","domain":[0,1],"title":
{"text":"word"}}}}
```

PART II Data Preparation

(B1) Unusual Characters:

Below is my code for identifying the presence of unusual characters. I frirst began by printing the unique list of characters contained throughout the entire dataset. After this process I then created three unique array categories in order to explain the presence of Alpha, Numeric and non-alphanumeric characters. Once that portion was completed I lowered the casing of all characters in every review, and then removed numeric and non-alphanumeric characters, and lastly removed the stop words from our review column using the nltk library.

```
commentary = df['review']
list_of_char = []
for comment in commentary:
    for char in comment:
        if char not in list_of_char:
            list_of_char.append(char)
print(list_of_char)
```

```
['A', ' ', 'v', 'e', 'r', 'y', ',', 's', 'l', 'o', 'w', '-', 'm', 'i',
'n', 'g', 'a', 'b', 'u', 't', 'd', 'f', '.', 'N', 'h', 'c', 'k', 'p',
'&', 'x', 'V', 'T', 'G', 'I', "'", 'W', 'S', 'L', 'J', 'B', 'F', 'M',
'H', 'C', '"', '\x96', 'z', '?', 'q', 'Y', 'j', 'P', 'U', 'R', 'E',
'1', '3', ';', '/', 'O', '2', '9', '0', ':', '*', 'D', 'Q', 'é', '(',
')', '!', 'K', '$', '7', '5', 'Z', '\x85', '8', '+', '%', '4', 'å',
'6', '\x97', 'X']
```

```
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

reviews = df['review']

alpha_chars = {char for review in reviews for token in
word_tokenize(review.lower()) for char in token if char.isalpha()}
num_chars = {char for review in reviews for token in
word_tokenize(review.lower()) for char in token if char.isdigit()}
nonal_num_chars = {char for review in reviews for token in
word_tokenize(review.lower()) for char in token if not char.isalnum()}

print('Alpha Characters:')
print(alpha_chars)
print('Total of', len(alpha_chars), 'unique English letters in this
dataset\n')

print('Numeric Characters:')
print(num_chars)
print('Total of', len(num_chars), 'unique numerical characters in this
dataset\n')

print('Non-alphanumeric characters:')
print(nonal_num_chars)
print('Total of', len(nonal_num_chars), 'unique special characters in
this dataset')

Alpha Characters:
{'y', 'é', 'f', 'z', 'k', 'o', 'q', 'i', 't', 'a', 'e', 's', 'u', 'l',
'r', 'j', 'c', 'x', 'g', 'b', 'p', 'd', 'v', 'm', 'n', 'h', 'w', 'å'}
Total of 28 unique English letters in this dataset

Numeric Characters:
{'2', '1', '7', '3', '9', '5', '8', '4', '0', '6'}
Total of 10 unique numerical characters in this dataset

Non-alphanumeric characters:
{'*', '!', '&', ';', "'", '`', ')', '-', '\x97', '%', '.', '+', ':',
',', '(', '\x96', '/', '$', '?'}
Total of 19 unique special characters in this dataset

# making characters lowercase
df.review = df.review.str.lower()

# downloading stopwords from nltk library
nltk.download('stopwords')

# printing stopwords
stop_words = set(stopwords.words('english'))
print(stop_words)

# review text Cleaning
def clean_reviews(text):

     # removing brackets
    regex = re.compile('<.*?>') # r'<.*?>'
    text = re.sub(regex, '', text)

    # removing special characters
    pattern = re.compile('[^a-zA-z0-9\s]')
    text = re.sub(pattern,'',text)

    # removing numbers
    pattern = re.compile('\d+')
    text = re.sub(pattern,'',text)

    # converting text to lower case
    text = text.lower()

    # Tokenization of words
    text = word_tokenize(text)

    # Stop words removal
    text = [word for word in text if not word in stop_words]

    return text

# using the clean_reviews function on the dataset
df['review'] = df['review'].apply(clean_reviews)
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\vince\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

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

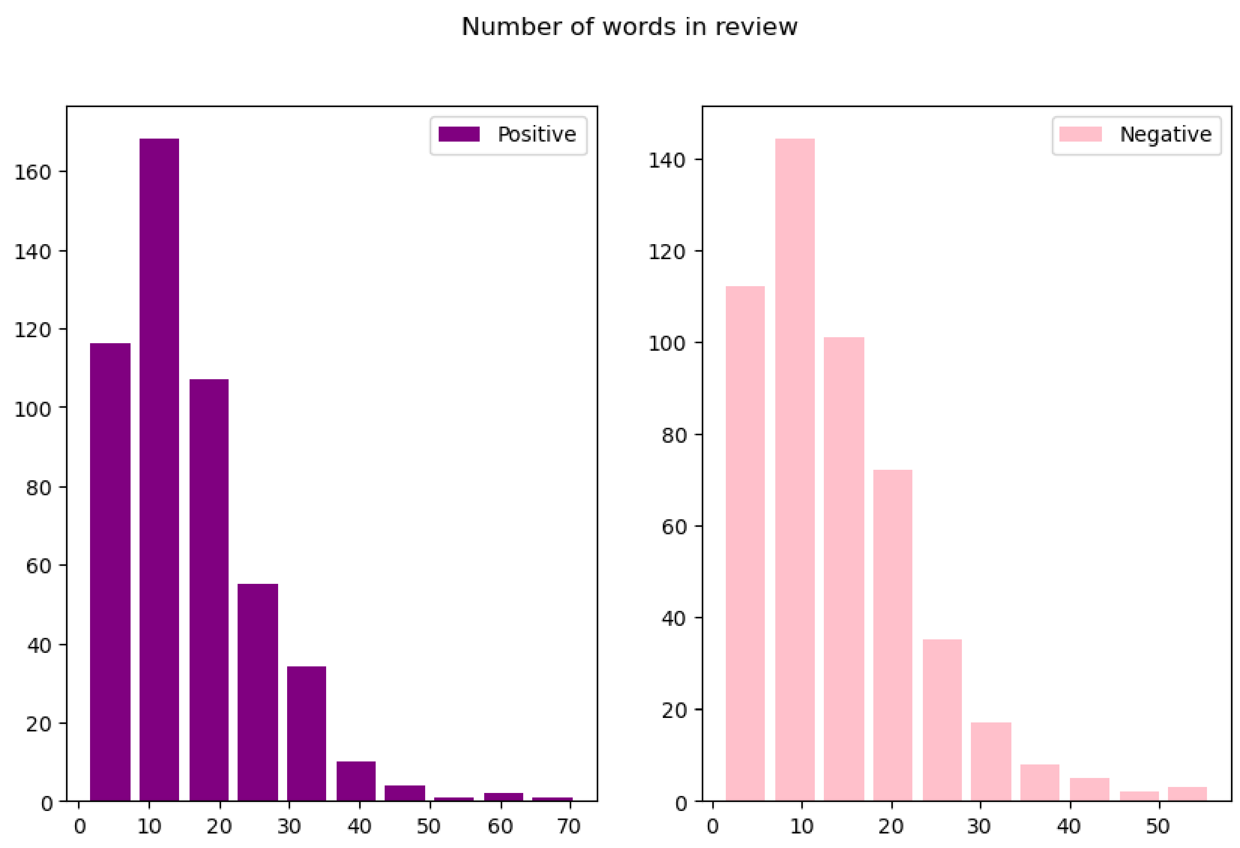
```
# printing dataframe post character cleaning
df
```

```
                                                    review  sentiment
word_count
0    [slowmoving, aimless, movie, distressed, drift...          0
13
1    [sure, lost, flat, characters, audience, nearl...          0
19
2    [attempting, artiness, black, white, clever, c...          0
31
3                        [little, music, anything, speak]          0
8
```

```
4    [best, scene, movie, gerardo, trying, find, so...          1
21
..                                                 ...        ...
...
995  [got, bored, watching, jessice, lange, take, c...          0
11
996  [unfortunately, virtue, films, production, wor...          0
14
997                              [word, embarrassing]          0
6
998                            [exceptionally, bad]          0
2
999   [insult, ones, intelligence, huge, waste, money]          0
15

[997 rows x 3 columns]
```

Vocabulary Size:

```
# vocabulary size
text_data = df['review'].astype(str)

# tokenize the text data into individual words to get unique words
vocabulary = set(word for text in text_data for word in
word_tokenize(text))

# calculate the vocabulary size
vocabulary_size = len(vocabulary)

# print count of unique words contained in review col
print("Vocabulary Size:", vocabulary_size)

Vocabulary Size: 3007

 I used the tokenizer from the keras library to get the vocabulary
size. The vocabulary size is the unique count of words that appears in
our review column in our dataframe.

# visualizing count of words in postive and negative reviews.
fig, ax = plt.subplots(1,2, figsize=(10,6))
ax[0].hist(df[df['sentiment'] == 1]['word_count'], label='Positive',
color='purple', rwidth=0.8);
ax[1].hist(df[df['sentiment'] == 0]['word_count'], label='Negative',
color='pink', rwidth=0.8);

ax[0].legend(loc='upper right');
ax[1].legend(loc='upper right');

fig.suptitle("Number of words in review")
plt.show()
```



```
I wanted to visualize the count of unique words in each review and
whether the sentiment was positive or negative. to my suprise there
was no correlation shown with a movie having a poor review and the
critic writing more words.
```

Proposed Word Embedding Length:

```
from tensorflow.keras.preprocessing.text import Tokenizer

# Tokenize the text data
tokenizer = Tokenizer()
tokenizer.fit_on_texts(df['review'])
sequences = tokenizer.texts_to_sequences(df['review'])

# Get the maximum sequence length
max_sequence_length = max(len(seq) for seq in sequences)

# Set the proposed word embedding length
proposed_embedding_length = int(np.ceil(max_sequence_length * 0.1))
proposed_embedding_length =
int(round(np.sqrt(np.sqrt(len(vocabulary))), 0))

print("Max Word Length", proposed_embedding_length)
print("Proposed Word Embedding Length:", proposed_embedding_length)
```

```
Max Word Length 7
Proposed Word Embedding Length: 7
```

The above code I used to prepare the text data for use in our NLP model. Because the model cannot read text the data needed to be tokenized using the tokenizer function from the tensorflow library. Essentially tokenization is a process used to convert text characters into a sequence of tokens for further processing. I then needed to fit the tokenizer on the text data using `fit_on_texts`, which also builds the vocab size on the text in our review column. `texts_to_sequences` is then used in order to actually do the convert the texts to sequences based on the vocab size. I then have to calculate the max sequence length in order to ensure I am properly padding uniform sequences later for the neural network. I lastly calculate the proposed word embedding link for the first portion I used 10% as an arbitrary percent in order to get a starting estimate and then later decided to also add the sqrt of the vocab size due to the rubric not being clear on which was needed.

```python
# Calculate the lengths of all sequences
seq_len = [len(seq) for seq in sequences]

# Maximum sequence length
max_len = np.max(seq_len)

# Median sequence length
med_len = np.median(seq_len)

# Minimum sequence length
min_len = np.min(seq_len)

print("Maximum Sequence Length:", max_len)
print("Median Sequence Length:", med_len)
print("Minimum Sequence Length:", min_len)
```

```
Maximum Sequence Length: 41
Median Sequence Length: 6.0
Minimum Sequence Length: 0
```

```python
df.head()
```

```
                                              review  sentiment
word_count
0  [slowmoving, aimless, movie, distressed, drift...          0
13
1  [sure, lost, flat, characters, audience, nearl...          0
19
2  [attempting, artiness, black, white, clever, c...          0
31
3                  [little, music, anything, speak]          0
8
```

```
4  [best, scene, movie, gerardo, trying, find, so...         1
21
```

I then printed statistics of the sequence lengths in the dataset in for the padding process. I will be using the max sequence length.

Tokenization process:

(B2) Describe Goals of Tokenization Process

The goals of the tokenization process for this analysis are as follows. The data will first need to be split into training, validation and test sets using train_test_split from the sklearn library. It will be split using a 70/30 split. I will also need to tokenize the text data in the different sets.

```
from sklearn.model_selection import train_test_split

# Split the data into train, validation, and test sets
x_train, x_test, y_train, y_test = train_test_split(df.review, 
df.sentiment, train_size=0.7, stratify=df.sentiment, random_state=2)
x_val, x_test, y_val, y_test = train_test_split(x_test, y_test, 
test_size=0.5, stratify=y_test, random_state=2)

# Tokenize the text data
tokenizer = Tokenizer(oov_token='OOV')
tokenizer.fit_on_texts(x_train)

# Convert text data to sequences
train_seq = tokenizer.texts_to_sequences(x_train)
test_seq = tokenizer.texts_to_sequences(x_test)
val_seq = tokenizer.texts_to_sequences(x_val)

train_seq[0]

[163, 164, 777, 778, 441, 17, 30, 779, 442, 780, 83, 2, 15, 48, 781]
```

B3

Explain padding sequence As explained above for the NLP model to work the sequence length of each review needs to contain the same count of characters. What I am doing in the padding process is taking the maximum length of characters that were previously identified and using post padding to fill sequences that are shorter than that with 0. I chose to use post padding in order to ensure I could keep the maximum number of reviews compared to pre padding which would remove reviews that are greater than the max_len number that was identified and also probably make the model itself more accurate. Below is a screenshot an example where we can see that becuase the review did not meet the amount of characters required it was filled with 0. As previously mentioned pre padding can make the model more accurate due to the 0s that appear in post padding, to ensure we are giving our model a fair chance I will mask 0 to eliminate that issue.

```
# truncating reviews to max amount of words
pad_train = pad_sequences(train_seq, maxlen=max_len, padding='post')
pad_test = pad_sequences(test_seq, maxlen=max_len, padding='post')
pad_val = pad_sequences(val_seq, maxlen=max_len, padding='post')

pad_train[0]
```

```
array([163, 164, 777, 778, 441,  17,  30, 779, 442, 780,  83,   2,
15,
        48, 781,   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])
```

B4 Categories of Sentiment:

There will be two categories of sentiment used with one representing postive and zero representing negative. I will use the sigmoid as the activation function. I chose simgoid due to its common use for models where probability prediction is expressed as an output. V7 labs goes on to further cement my choice by stating *"Since probability of anything exists only between the range of 0 and 1, sigmoid is the right choice because of its range."* Moreover a downside of the choosing sigmoid is that it is more prone to the gradient vanishing problem. The vanishing gradient problem summarized is esentially as more layers are added to the neura network it becomes harder to train due as the loss function begins to approach zero.

B5

```
# provided copy of cleaned dataframe
df.to_csv('d213_task2.csv')
```

Part III

(C1)

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, GRU, Dense
from tensorflow.keras import optimizers
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint
import kerastuner as kt

def build_model(hp):
    n_neurons = hp.Int("n_neurons", min_value=16, max_value=256)
    learning_rate = hp.Float("learning_rate", min_value=1e-4,
max_value=1e-2, sampling="log")

    model = Sequential(name="GRU_Model")
    model.add(Embedding(vocabulary_size,
                        proposed_embedding_length,
                        input_length=max_len,
                        mask_zero=True))
    model.add(GRU(n_neurons,
                  activation='tanh',
                  return_sequences=False))
    model.add(Dense(1, activation='sigmoid'))

    optimizer = optimizers.Nadam(learning_rate=learning_rate)
    model.compile(loss="binary_crossentropy",
                  optimizer=optimizer,
                  metrics=['accuracy'])

    return model

# Hyperparameter tuning
early_stopping_cb = EarlyStopping(patience=2)
model_checkpoint_cb = ModelCheckpoint("best_model.h5",
save_best_only=True)

random_search_tuner = kt.Hyperband(
    build_model, objective="val_accuracy", overwrite=True,
    directory="hp_search_results", project_name="d213_task2", seed=42,
    max_epochs=10, factor=3, hyperband_iterations=2)

random_search_tuner.search(pad_train, y_train, epochs=10,
                           validation_data=(pad_val, y_val),
                           callbacks=[early_stopping_cb,
model_checkpoint_cb])

Trial 60 Complete [00h 00m 28s]
val_accuracy: 0.7666666507720947

Best val_accuracy So Far: 0.8133333325386047
Total elapsed time: 00h 21m 32s

# provide the output of the model summary
best_model = random_search_tuner.get_best_models(num_models=1)[0]
best_model.summary()

Model: "GRU_Model"

_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 embedding (Embedding)       (None, 41, 7)             21049     
                                                                 
 gru (GRU)                   (None, 241)               180750    
                                                                 
 dense (Dense)               (None, 1)                 242       
                                                                 
=================================================================
Total params: 202041 (789.22 KB)
Trainable params: 202041 (789.22 KB)
```

```
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________

# number of parameters
best_model.summary()

# Calculate total number of parameters
total_parameters = sum([layer.count_params() for layer in
best_model.layers])
print(f"Total number of parameters in the model: {total_parameters}")

Model: "GRU_Model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 41, 7)             21049

 gru (GRU)                   (None, 241)               180750

 dense (Dense)               (None, 1)                 242

=================================================================
Total params: 202041 (789.22 KB)
Trainable params: 202041 (789.22 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
Total number of parameters in the model: 202041

# print best trial
best_trial = random_search_tuner.oracle.get_best_trials(num_trials=1)
[0]
best_trial.summary()

Trial 0048 summary
Hyperparameters:
n_neurons: 241
learning_rate: 0.004511709522511237
tuner/epochs: 4
tuner/initial_epoch: 0
tuner/bracket: 1
tuner/round: 0
Score: 0.8133333325386047
```

(C3) Evaluation Metric

```
# Evaluation
score = best_model.evaluate(pad_test, y_test, verbose=0)
print(f"Test loss: {score[0]} / Test accuracy: {score[1]}")

Test loss: 0.6288843750953674 / Test accuracy: 0.8199999928474426
```

```
# Training the best model
best_model.fit(pad_train, y_train, epochs=5,
               validation_data=(pad_val, y_val),
callbacks=[early_stopping_cb, model_checkpoint_cb])

Epoch 1/5
22/22 [==============================] - 12s 107ms/step - loss: 0.0219
- accuracy: 0.9928 - val_loss: 0.8314 - val_accuracy: 0.7733
Epoch 2/5
22/22 [==============================] - 2s 98ms/step - loss: 0.0092 -
accuracy: 0.9957 - val_loss: 1.1637 - val_accuracy: 0.7533
Epoch 3/5
22/22 [==============================] - 2s 98ms/step - loss: 0.0265 -
accuracy: 0.9928 - val_loss: 0.8595 - val_accuracy: 0.7533

<keras.src.callbacks.History at 0x159c1acf2d0>

train_acc = best_trial.metrics.get_history('accuracy')[0].value[0]
train_loss = best_trial.metrics.get_history('loss')[0].value[0]

val_acc = best_trial.metrics.get_history('val_accuracy')[0].value[0]
val_loss = best_trial.metrics.get_history('val_loss')[0].value[0]

print(f"Train loss: {train_loss} / Train accuracy: {train_acc}")
print()
print(f"Validation loss: {val_loss} / Validation accuracy: {val_acc}")
print()

Train loss: 0.04848959296941757 / Train accuracy: 0.9856528043746948

Validation loss: 0.5766692161560059 / Validation accuracy:
0.8133333325386047
```

(C2)

- **Number of Layers:** The number of layers contained in my model is three. First embedding layer converts each sequence into a vector, this assists the model learn the meaning of words. Next is the Gate Recurrent layer or GRU I selected due to it being a extremly effective way to handle sequenced data, this layer helps the model learn the dependencies between the words in a sequence. Moreover GRU layer is what is actually reading the vectors and remebers the the important information, this is the most important in terms of machine learning. Finally the dense layer takes all of the information recieved from the GRU layer and makes a judgement on what the sentiment is (0 or 1) to represent positive or negative sentiment.

- **Number of Parameters:** The number of parameters for my model is first the embedding layer which contains the vocabulary_size of 3007 characters and a embedded length of 7 . The GRU layers parameters are being determined by the hp tuner from the keras library, the dense layer is also using the optimal number of n_neurons plus one.

(C3)

- **Justification of Hyperparameters:** Activation Functions : I am using 'tanh' activation in the GRU layer, due to its ability to help the model capture patterns in sequential data.
- **Number of Nodes:** The range I selected for n_neurons was between 16 and 256. This allows the model to explore different levels of complexity. I chose the range arbitrarily as it was listed on the keras website, but I believe this to be a sufficient range, I was worried about the risk of underfitting. The hyperband (hp) tuner found that 187 resulted in the best accuracy score.
- **Loss Function:** For loss function I chose Binary cross-entropy due to its ability to handle binary classification tasks. Moreover analytics vidhya goes on to state that *"It quantifies the dissimilarity between probability distributions, aiding model training by penalizing inaccurate predictions."*
- **Optimizer:** The optimizer I selected for my model is Nadam which is an combines both Adam and Nesterov which improves momentum. Nadam works by computing the gradient of the loss function for a batch of data.
- **Stopping Criteria:** The stopping criteria used for my analysis is early stopping. I am using a patience of two which means that if validation loss does not improve for two consecutive epochs the model stops and this prevents overfitting. The model had a validation accuracy score of 83%, meaning the model performs well on unseen data. The training accuracy score of 1 which means that the model is able to perfectly classify training data which leads me to believe that there may be some overfitting. Lastly the test accuracy score of 77% means that the model performs pretty well on the testing data.
- **Evaluation Metric:** Lastly the validation metric that I am using for the model is validation accuracy which ensures that the model performs well on unseen data.

Part IV Model Evaluation:

(D1)

- **Impact of Stopping Critieria:** As previously discuseed the stopping criteria is required in order to prevent overfitting by the neural network model. It also helps increase the tuning process by not using as many epochs. Below is a screenshot of the final training epoch.

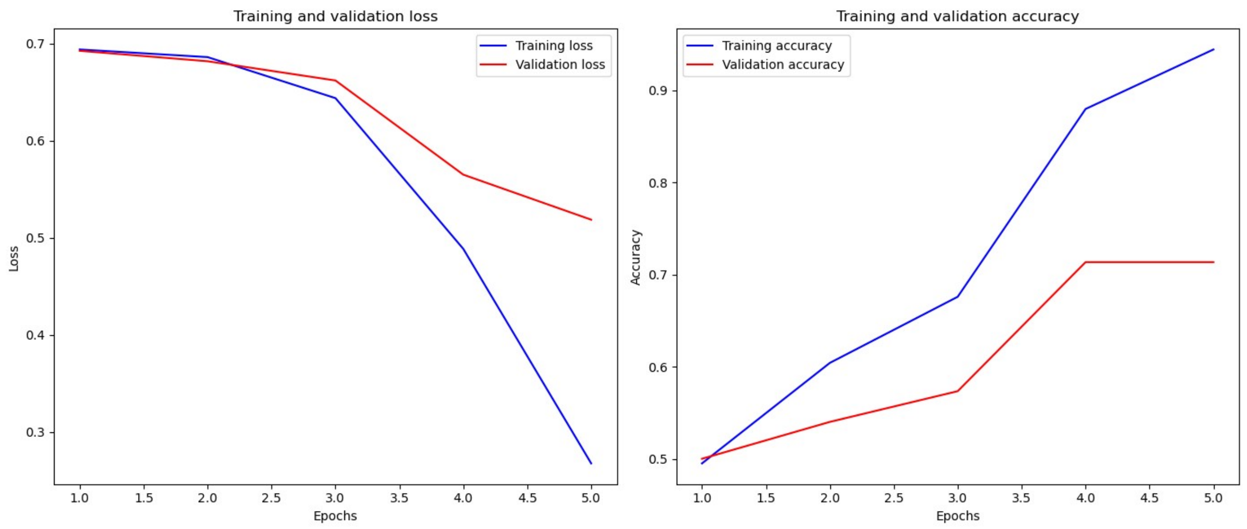
(D2)

- **Fitness of model:** As previously discussed the model is most likely overfit with due to the perfect training score of 1. I used the early stopping criteria in order to try to prevent overfitting as it is commonly used but that did not seem to work. I assesed the fitness by evaluating the model on both testing and validation sets.

```
new_model = Sequential.from_config(best_model.get_config())
new_model.compile(loss="binary_crossentropy", optimizer="adam",
metrics=["accuracy"])

history = new_model.fit(pad_train, y_train, epochs=5,
                        validation_data=(pad_val, y_val),
callbacks=[early_stopping_cb])

Epoch 1/5
22/22 [==============================] - 14s 219ms/step - loss: 0.6941
- accuracy: 0.4950 - val_loss: 0.6927 - val_accuracy: 0.5000
Epoch 2/5
22/22 [==============================] - 2s 102ms/step - loss: 0.6862
- accuracy: 0.6040 - val_loss: 0.6819 - val_accuracy: 0.5400
Epoch 3/5
22/22 [==============================] - 3s 118ms/step - loss: 0.6439
- accuracy: 0.6758 - val_loss: 0.6621 - val_accuracy: 0.5733
Epoch 4/5
22/22 [==============================] - 2s 100ms/step - loss: 0.4886
- accuracy: 0.8795 - val_loss: 0.5651 - val_accuracy: 0.7133
Epoch 5/5
22/22 [==============================] - 2s 96ms/step - loss: 0.2676 -
accuracy: 0.9440 - val_loss: 0.5187 - val_accuracy: 0.7133
```

(D3)

```
history_dict = history.history
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']
acc_values = history_dict['accuracy']
val_acc_values = history_dict['val_accuracy']

epochs = range(1, len(loss_values) + 1)

plt.figure(figsize=(14, 6))

plt.subplot(1, 2, 1)
plt.plot(epochs, loss_values, 'b', label='Training loss')
plt.plot(epochs, val_loss_values, 'r', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()

plt.subplot(1, 2, 2)
plt.plot(epochs, acc_values, 'b', label='Training accuracy')
plt.plot(epochs, val_acc_values, 'r', label='Validation accuracy')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.tight_layout()
plt.show()
```



```
# (D4) Predictive Accuracy
print(f"Train loss: {train_loss} / Train accuracy: {train_acc}")
print()
print(f"Validation loss: {val_loss} / Validation accuracy: {val_acc}")
print()

Train loss: 0.04848959296941757 / Train accuracy: 0.9856528043746948

Validation loss: 0.5766692161560059 / Validation accuracy:
0.8133333325386047
```

(D4)

- **Training Loss and Accuracy:** The training loss is low at 3.5%, which proves that the model was able to fit the training data well. The training accuracy is high 99%, which means that the model has learned the patterns in the training data with high accuracy, but more likely the model is showing overfitting.
- **Validation Loss and Accuracy:** The validation loss is higher 60% compared to the training loss, which leads me to believe that the model definitely shows signs of overfitting on the training data. Moreover, the validation accuracy of 82% is good and shows that the model is able to perform well on unseen data.

Part V

```
# Saving the best model (E)
best_model.save("best_model.h5")
```

(F)

- **Functionality of Neural Network:** The code for my model is designed for binary classification tasks using a GRU for sequence processing. Moreover this model specifically analyzes sentiment of reviews from the imdb database but could be used interchangeably for other reviews were we have binary outcomes. The network itself begins with an embedding layer to convert integer coded words into vectors. The second layer is the GRU layer which is similar to a LTSM model but the training is faster due to having less parameters. The final layer is the dense layer with one neuron and has a sigmoid activation function. The optimizer being used for my network is the Nadam which as previously discussed uses Nesterov momentum in order to improve convergence during training.

(G)

- **Recommendation:** My recommendation for this analysis would be to try to first fix the overfitting issue, but then implementing this into a production enviornment and use for predicting sentiment on our own telecom company reviews to see if we can start training the model on more in house data to improve its accuracy.

Sources

Géron, A. (2019). Hands-on machine learning with Scikit-Learn and TensorFlow concepts, tools, and techniques to build intelligent systems (2nd ed.). O'Reilly Media, Inc.

Sentiment Analysis with an Recurrent Neural Networks (RNN). (2022, October 7). GeeksforGeeks. https://www.geeksforgeeks.org/sentiment-analysis-with-an-recurrent-neural-networks-rnn/

Activation Functions in Neural Networks [12 Types & Use Cases]. (n.d.). Www.v7labs.com. https://www.v7labs.com/blog/neural-networks-activation-functions#:~:text=Sigmoid%20%2F%20Logistic%20Activation%20Function

Li, S. (2018, June 2). A Beginner's Guide on Sentiment Analysis with RNN. Medium; Towards Data Science. https://towardsdatascience.com/a-beginners-guide-on-sentiment-analysis-with-rnn-9e100627c02e

Caner. (2020, April 3). Padding for NLP. Medium. https://medium.com/@canerkilinc/padding-for-nlp-7dd8598c916a

Team, K. (n.d.). Keras documentation: Getting started with KerasTuner. Keras.io. https://keras.io/guides/keras_tuner/getting_started/

Basic text classification | TensorFlow Core. (n.d.). TensorFlow. https://www.tensorflow.org/tutorials/keras/text_classification

English, M. L. in P. (2023, June 3). Deep Learning Course — Lesson 7.5: Nadam (Nesterov-accelerated Adaptive Moment Estimation). Medium. https://medium.com/@nerdjock/deep-learning-course-lesson-7-5-nadam-nesterov-accelerated-adaptive-moment-estimation-efe9050d5b9b#:~:text=Here%20is%20how%20it%20works

Deepanshi. (2021, August 14). Easy Hyperparameter Tuning in Neural Networks using Keras Tuner. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/08/easy-hyperparameter-tuning-in-neural-networks-using-keras-tuner/

Lecture 19 - RNN Implementation. (n.d.). Www.youtube.com. Retrieved February 20, 2024, from https://www.youtube.com/watch?v=q12VPh-bK7k&t=73s

Natural Language Processing - Tokenization (NLP Zero to Hero - Part 1). (n.d.). Www.youtube.com. https://www.youtube.com/watch?v=fNxaJsNG3-s&list=PLQY2H8rRoyvzDbLUZkbudP-MFQZwNmU4S