ASSIGNMENT 7

```
!pip install tensorflow-hub
!pip install tensorflow-datasets
```

```
Looking in indexes: https://pypi.org/simple, https://us-
python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: tensorflow-hub in
/usr/local/lib/python3.7/dist-packages (0.12.0)
Requirement already satisfied: numpy>=1.12.0 in
/usr/local/lib/python3.7/dist-packages (from tensorflow-hub) (1.21.6)
Requirement already satisfied: protobuf>=3.8.0 in
/usr/local/lib/python3.7/dist-packages (from tensorflow-hub) (3.19.6)
Looking in indexes: https://pypi.org/simple, https://us-
python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: tensorflow-datasets in
/usr/local/lib/python3.7/dist-packages (4.6.0)
Requirement already satisfied: tqdm in /usr/local/lib/python3.7/dist-
packages (from tensorflow-datasets) (4.64.1)
Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-
packages (from tensorflow-datasets) (1.21.6)
Requirement already satisfied: requests>=2.19.0 in
/usr/local/lib/python3.7/dist-packages (from tensorflow-datasets)
(2.23.0)
Requirement already satisfied: tensorflow-metadata in
/usr/local/lib/python3.7/dist-packages (from tensorflow-datasets)
(1.10.0)
Requirement already satisfied: typing-extensions in
/usr/local/lib/python3.7/dist-packages (from tensorflow-datasets)
(4.1.1)
Requirement already satisfied: dill in /usr/local/lib/python3.7/dist-
packages (from tensorflow-datasets) (0.3.6)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-
packages (from tensorflow-datasets) (1.15.0)
Requirement already satisfied: termcolor in
/usr/local/lib/python3.7/dist-packages (from tensorflow-datasets)
(2.1.0)
Requirement already satisfied: toml in /usr/local/lib/python3.7/dist-
packages (from tensorflow-datasets) (0.10.2)
Requirement already satisfied: importlib-resources in
/usr/local/lib/python3.7/dist-packages (from tensorflow-datasets)
(5.10.0)
Requirement already satisfied: promise in
/usr/local/lib/python3.7/dist-packages (from tensorflow-datasets)
(2.3)
Requirement already satisfied: absl-py in
/usr/local/lib/python3.7/dist-packages (from tensorflow-datasets)
(1.3.0)
Requirement already satisfied: protobuf>=3.12.2 in
```

```
/usr/local/lib/python3.7/dist-packages (from tensorflow-datasets)
(3.19.6)
Requirement already satisfied: etils[epath] in
/usr/local/lib/python3.7/dist-packages (from tensorflow-datasets)
(0.9.0)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.7/dist-packages (from requests>=2.19.0-
>tensorflow-datasets) (2022.9.24)
Requirement already satisfied: chardet<4,>=3.0.2 in
/usr/local/lib/python3.7/dist-packages (from requests>=2.19.0-
>tensorflow-datasets) (3.0.4)
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>=2.19.0-
>tensorflow-datasets) (1.24.3)
Requirement already satisfied: idna<3,>=2.5 in
/usr/local/lib/python3.7/dist-packages (from requests>=2.19.0-
>tensorflow-datasets) (2.10)
Requirement already satisfied: zipp in /usr/local/lib/python3.7/dist-
packages (from etils[epath]->tensorflow-datasets) (3.10.0)
Requirement already satisfied: googleapis-common-protos<2,>=1.52.0
in /usr/local/lib/python3.7/dist-packages (from tensorflow-metadata-
>tensorflow-datasets) (1.56.4)
```

```python
from tensorflow.keras import layers
from tensorflow import keras
import tensorflow as tf
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

import tensorflow_hub as hub
import tensorflow_datasets as tfds

train_data, validation_data, test_data = tfds.load(
    name="imdb_reviews",
    split=('train[:60%]', 'train[60%:]', 'test'),
    as_supervised=True)
```

```
Downloading and preparing dataset 80.23 MiB (download: 80.23 MiB,
generated: Unknown size, total: 80.23 MiB) to
~/tensorflow_datasets/imdb_reviews/plain_text/1.0.0...
```

{"version_major":2,"version_minor":0,"model_id":"568957ba362444b381f49
6ea0afb5dc1"}

{"version_major":2,"version_minor":0,"model_id":"8b46a0333d1048c08d0df
e4dd315b100"}

{"version_major":2,"version_minor":0,"model_id":"f5d8cbf1c59545ca9ad46
22d0428330a"}

{"version_major":2,"version_minor":0,"model_id":"da02488602674a888cc4c
90a39ee94f6"}

{"version_major":2,"version_minor":0,"model_id":"3456624227774a3aa3179
3427823289b"}

{"version_major":2,"version_minor":0,"model_id":"b332a8407d9541f2b7321
3672891a6db"}

{"version_major":2,"version_minor":0,"model_id":"b142f577f4324497bba21
9c5212c259f"}

{"version_major":2,"version_minor":0,"model_id":"bf82d6fedd554676859a3
7e0c6974682"}

{"version_major":2,"version_minor":0,"model_id":"23038e8e707f4308a3a8e
cea7f4cc229"}

```
Dataset imdb_reviews downloaded and prepared to
~/tensorflow_datasets/imdb_reviews/plain_text/1.0.0. Subsequent calls
will reuse this data.
```

```
train_examples_batch, train_labels_batch =
next(iter(train_data.batch(10)))
```

```
train_examples_batch
```

```
<tf.Tensor: shape=(10,), dtype=string, numpy=
array([b"This was an absolutely terrible movie. Don't be lured in by
Christopher Walken or Michael Ironside. Both are great actors, but
this must simply be their worst role in history. Even their great
acting could not redeem this movie's ridiculous storyline. This movie
is an early nineties US propaganda piece. The most pathetic scenes
were those when the Columbian rebels were making their cases for
revolutions. Maria Conchita Alonso appeared phony, and her pseudo-love
affair with Walken was nothing but a pathetic emotional plug in a
movie that was devoid of any real meaning. I am disappointed that
there are movies like this, ruining actor's like Christopher Walken's
good name. I could barely sit through it.",
       b'I have been known to fall asleep during films, but this is
usually due to a combination of things including, really tired, being
warm and comfortable on the sette and having just eaten a lot. However
on this occasion I fell asleep because the film was rubbish. The plot
development was constant. Constantly slow and boring. Things seemed to
happen, but with no explanation of what was causing them or why. I
admit, I may have missed part of the film, but i watched the majority
of it and everything just seemed to happen of its own accord without
any real concern for anything else. I cant recommend this film at
all.',
        b'Mann photographs the Alberta Rocky Mountains in a superb
fashion, and Jimmy Stewart and Walter Brennan give enjoyable
performances as they always seem to do. <br /><br />But come on
Hollywood - a Mountie telling the people of Dawson City, Yukon to
elect themselves a marshal (yes a marshal!) and to enforce the law
themselves, then gunfighters battling it out on the streets for
control of the town? <br /><br />Nothing even remotely resembling that
happened on the Canadian side of the border during the Klondike gold
rush. Mr. Mann and company appear to have mistaken Dawson City for
Deadwood, the Canadian North for the American Wild West.<br /><br
/>Canadian viewers be prepared for a Reefer Madness type of enjoyable
howl with this ludicrous plot, or, to shake your head in disgust.',
        b'This is the kind of film for a snowy Sunday afternoon when
the rest of the world can go ahead with its own business as you
descend into a big arm-chair and mellow for a couple of hours.
Wonderful performances from Cher and Nicolas Cage (as always) gently
row the plot along. There are no rapids to cross, no dangerous waters,
just a warm and witty paddle through New York life at its best. A
family film in every sense and one that deserves the praise it
received.',
        b'As others have mentioned, all the women that go nude in this
film are mostly absolutely gorgeous. The plot very ably shows the
hypocrisy of the female libido. When men are around they want to be
pursued, but when no "men" are around, they become the pursuers of a
14 year old boy. And the boy becomes a man really fast (we should all
be so lucky at this age!). He then gets up the courage to pursue his
true love.',
        b"This is a film which should be seen by anybody interested in,
effected by, or suffering from an eating disorder. It is an amazingly
accurate and sensitive portrayal of bulimia in a teenage girl, its
causes and its symptoms. The girl is played by one of the most
brilliant young actresses working in cinema today, Alison Lohman, who
was later so spectacular in 'Where the Truth Lies'. I would recommend
that this film be shown in all schools, as you will never see a better
on this subject. Alison Lohman is absolutely outstanding, and one
marvels at her ability to convey the anguish of a girl suffering from
this compulsive disorder. If barometers tell us the air pressure,
Alison Lohman tells us the emotional pressure with the same degree of
accuracy. Her emotional range is so precise, each scene could be
measured microscopically for its gradations of trauma, on a scale of
rising hysteria and desperation which reaches unbearable intensity.
Mare Winningham is the perfect choice to play her mother, and does so
with immense sympathy and a range of emotions just as finely tuned as
Lohman's. Together, they make a pair of sensitive emotional
oscillators vibrating in resonance with one another. This film is
really an astonishing achievement, and director Katt Shea should be
proud of it. The only reason for not seeing it is if you are not
interested in people. But even if you like nature films best, this is
after all animal behaviour at the sharp edge. Bulimia is an extreme
version of how a tormented soul can destroy her own body in a frenzy
```

of despair. And if we don't sympathise with people suffering from the depths of despair, then we are dead inside.",
b'Okay, you have:<br /><br />Penelope Keith as Miss Herringbone-Tweed, B.B.E. (Backbone of England.) She\'s killed off in the first scene - that\'s right, folks; this show has no backbone!<br /><br />Peter O\'Toole as Ol\' Colonel Cricket from The First War and now the emblazered Lord of the Manor.<br /><br />Joanna Lumley as the ensweatered Lady of the Manor, 20 years younger than the colonel and 20 years past her own prime but still glamourous (Brit spelling, not mine) enough to have a toy-boy on the side. It\'s alright, they have Col. Cricket\'s full knowledge and consent (they guy even comes \'round for Christmas!) Still, she\'s considerate of the colonel enough to have said toy-boy her own age (what a gal!)<br /><br />David McCallum as said toy-boy, equally as pointlessly glamourous as his squeeze. Pilcher couldn\'t come up with any cover for him within the story, so she gave him a hush-hush job at the Circus.<br /><br />and finally:<br /><br />Susan Hampshire as Miss Polonia Teacups, Venerable Headmistress of the Venerable Girls\' Boarding-School, serving tea in her office with a dash of deep, poignant advice for life in the outside world just before graduation. Her best bit of advice: "I\'ve only been to Nancherrow (the local Stately Home of England) once. I thought it was very beautiful but, somehow, not part of the real world." Well, we can\'t say they didn\'t warn us.<br /><br />Ah, Susan - time was, your character would have been running the whole show. They don\'t write \'em like that any more. Our loss, not yours.<br /><br />So - with a cast and setting like this, you have the re-makings of "Brideshead Revisited," right?<br /><br />Wrong! They took these 1-dimensional supporting roles because they paid so well. After all, acting is one of the oldest temp-jobs there is (YOU name another!)<br /><br />First warning sign: lots and lots of backlighting. They get around it by shooting outdoors - "hey, it\'s just the sunlight!"<br /><br />Second warning sign: Leading Lady cries a lot. When not crying, her eyes are moist. That\'s the law of romance novels: Leading Lady is "dewy-eyed."<br /><br />Henceforth, Leading Lady shall be known as L.L.<br /><br />Third warning sign: L.L. actually has stars in her eyes when she\'s in love. Still, I\'ll give Emily Mortimer an award just for having to act with that spotlight in her eyes (I wonder . did they use contacts?)<br /><br />And lastly, fourth warning sign: no on-screen female character is "Mrs." She\'s either "Miss" or "Lady."<br /><br />When all was said and done, I still couldn\'t tell you who was pursuing whom and why. I couldn\'t even tell you what was said and done.<br /><br />To sum up: they all live through World War II without anything happening to them at all.<br /><br />OK, at the end, L.L. finds she\'s lost her parents to the Japanese prison camps and baby sis comes home catatonic. Meanwhile (there\'s always a "meanwhile,") some young guy L.L. had a crush on (when, I don\'t know) comes home from some wartime tough spot and is found living on the street by Lady of the Manor (must be some street if SHE\'s going to find him there.) Both war casualties are whisked away to recover at Nancherrow (SOMEBODY has to be "whisked away"

```
SOMEWHERE in these romance stories!)<br /><br />Great drama.',
       b'The film is based on a genuine 1950s novel.<br /><br
/>Journalist Colin McInnes wrote a set of three "London novels":
"Absolute Beginners", "City of Spades" and "Mr Love and Justice". I
have read all three. The first two are excellent. The last, perhaps an
experiment that did not come off. But McInnes\'s work is highly
acclaimed; and rightly so. This musical is the novelist\'s ultimate
nightmare - to see the fruits of one\'s mind being turned into a
glitzy, badly-acted, soporific one-dimensional apology of a film that
says it captures the spirit of 1950s London, and does nothing of the
sort.<br /><br />Thank goodness Colin McInnes wasn\'t alive to witness
it.',
       b'I really love the sexy action and sci-fi films of the sixties
and its because of the actress\'s that appeared in them. They found
the sexiest women to be in these films and it didn\'t matter if they
could act (Remember "Candy"?). The reason I was disappointed by this
film was because it wasn\'t nostalgic enough. The story here has a
European sci-fi film called "Dragonfly" being made and the director is
fired. So the producers decide to let a young aspiring filmmaker
(Jeremy Davies) to complete the picture. They\'re is one real
beautiful woman in the film who plays Dragonfly but she\'s barely in
it. Film is written and directed by Roman Coppola who uses some of his
fathers exploits from his early days and puts it into the script. I
wish the film could have been an homage to those early films. They
could have lots of cameos by actors who appeared in them. There is one
actor in this film who was popular from the sixties and its John
Phillip Law (Barbarella). Gerard Depardieu, Giancarlo Giannini and
Dean Stockwell appear as well. I guess I\'m going to have to continue
waiting for a director to make a good homage to the films of the
sixties. If any are reading this, "Make it as sexy as you can"! I\'ll
be waiting!',
       b'Sure, this one isn\'t really a blockbuster, nor does it
target such a position. "Dieter" is the first name of a quite popular
German musician, who is either loved or hated for his kind of acting
and thats exactly what this movie is about. It is based on the
autobiography "Dieter Bohlen" wrote a few years ago but isn\'t meant
to be accurate on that. The movie is filled with some sexual offensive
content (at least for American standard) which is either amusing (not
for the other "actors" of course) or dumb - it depends on your
individual kind of humor or on you being a "Bohlen"-Fan or not.
Technically speaking there isn\'t much to criticize. Speaking of me I
find this movie to be an OK-movie.'],
      dtype=object)>
```

```
train_labels_batch
```

```
<tf.Tensor: shape=(10,), dtype=int64, numpy=array([0, 0, 0, 1, 1, 1,
0, 0, 0, 0])>
```

```
embedding = "https://tfhub.dev/google/nnlm-en-dim50/2"
hub_layer = hub.KerasLayer(embedding, input_shape=[],
                           dtype=tf.string, trainable=True)
hub_layer(train_examples_batch[:3])

<tf.Tensor: shape=(3, 50), dtype=float32, numpy=
array([[ 0.5423195 , -0.0119017 ,  0.06337538,  0.06862972, -
0.16776837,
        -0.10581174,  0.16865303, -0.04998824, -0.31148055,
0.07910346,
         0.15442263,  0.01488662,  0.03930153,  0.19772711, -
0.12215476,
        -0.04120981, -0.2704109 , -0.21922152,  0.26517662, -
0.80739075,
         0.25833532, -0.3100421 ,  0.28683215,  0.1943387 , -
0.29036492,
         0.03862849, -0.7844411 , -0.0479324 ,  0.4110299 , -
0.36388892,
        -0.58034706,  0.30269456,  0.3630897 , -0.15227164, -
0.44391504,
         0.19462997,  0.19528408,  0.05666234,  0.2890704 , -
0.28468323,
        -0.00531206,  0.0571938 , -0.3201318 , -0.04418665, -
0.08550783,
        -0.55847436, -0.23336391, -0.20782952, -0.03543064, -
0.17533456],
       [ 0.56338924, -0.12339553, -0.10862679,  0.7753425 , -
0.07667089,
        -0.15752277,  0.01872335, -0.08169781, -0.3521876 ,  0.4637341
,
        -0.08492756,  0.07166859, -0.00670817,  0.12686075, -
0.19326553,
        -0.52626437, -0.3295823 ,  0.14394785,  0.09043556, -0.5417555
,
         0.02468163, -0.15456742,  0.68333143,  0.09068331, -
0.45327246,
         0.23180096, -0.8615696 ,  0.34480393,  0.12838456, -
0.58759046,
        -0.4071231 ,  0.23061076,  0.48426893, -0.27128142, -0.5380916
,
         0.47016326,  0.22572741, -0.00830663,  0.2846242 , -0.304985
,
         0.04400365,  0.25025874,  0.14867121,  0.40717036, -
0.15422426,
        -0.06878027, -0.40825695, -0.3149215 ,  0.09283665, -
0.20183425],
       [ 0.7456154 ,  0.21256861,  0.14400336,  0.5233862 ,
0.11032254,
         0.00902788, -0.3667802 , -0.08938274, -0.24165542,
0.33384594,
        -0.11194605, -0.01460047, -0.0071645 ,  0.19562712,
0.00685216,
        -0.24886718, -0.42796347,  0.18620004, -0.05241098, -
0.66462487,
         0.13449019, -0.22205497,  0.08633006,  0.43685386,  0.2972681
,
         0.36140734, -0.7196889 ,  0.05291241, -0.14316116, -0.1573394
,
        -0.15056328, -0.05988009, -0.08178931, -0.15569411, -
0.09303783,
        -0.18971172,  0.07620788, -0.02541647, -0.27134508, -0.3392682
,
        -0.10296468, -0.27275252, -0.34078008,  0.20083304, -
0.26644835,
         0.00655449, -0.05141488, -0.04261917, -0.45413622,
0.20023568]],
      dtype=float32)>
```

```
model = tf.keras.Sequential()
model.add(hub_layer)
model.add(tf.keras.layers.Dense(16, activation='relu'))
model.add(tf.keras.layers.Dense(1))

model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 keras_layer (KerasLayer)    (None, 50)                48190600  
                                                                 
 dense (Dense)               (None, 16)                816       
                                                                 
 dense_1 (Dense)             (None, 1)                 17        
                                                                 
=================================================================
Total params: 48,191,433
Trainable params: 48,191,433
Non-trainable params: 0
_________________________________________________________________
```

```
model.compile(optimizer='adam',

loss=tf.keras.losses.BinaryCrossentropy(from_logits=True),
              metrics=['accuracy'])
```

```
history = model.fit(train_data.shuffle(10000).batch(512),
                    epochs=10,
                    validation_data=validation_data.batch(512),
                    verbose=1)
```

```
Epoch 1/10
30/30 [==============================] - 24s 736ms/step - loss: 0.6605
```

```
- accuracy: 0.5386 - val_loss: 0.6124 - val_accuracy: 0.6131
Epoch 2/10
30/30 [==============================] - 23s 753ms/step - loss: 0.5427
- accuracy: 0.7014 - val_loss: 0.5033 - val_accuracy: 0.7454
Epoch 3/10
30/30 [==============================] - 20s 634ms/step - loss: 0.4068
- accuracy: 0.8290 - val_loss: 0.4026 - val_accuracy: 0.8188
Epoch 4/10
30/30 [==============================] - 19s 629ms/step - loss: 0.2923
- accuracy: 0.8887 - val_loss: 0.3480 - val_accuracy: 0.8395
Epoch 5/10
30/30 [==============================] - 19s 631ms/step - loss: 0.2145
- accuracy: 0.9253 - val_loss: 0.3204 - val_accuracy: 0.8543
Epoch 6/10
30/30 [==============================] - 21s 715ms/step - loss: 0.1584
- accuracy: 0.9517 - val_loss: 0.3086 - val_accuracy: 0.8692
Epoch 7/10
30/30 [==============================] - 24s 805ms/step - loss: 0.1155
- accuracy: 0.9692 - val_loss: 0.3073 - val_accuracy: 0.8632
Epoch 8/10
30/30 [==============================] - 20s 654ms/step - loss: 0.0840
- accuracy: 0.9792 - val_loss: 0.3114 - val_accuracy: 0.8730
Epoch 9/10
30/30 [==============================] - 22s 727ms/step - loss: 0.0613
- accuracy: 0.9881 - val_loss: 0.3227 - val_accuracy: 0.8703
Epoch 10/10
30/30 [==============================] - 19s 637ms/step - loss: 0.0450
- accuracy: 0.9937 - val_loss: 0.3309 - val_accuracy: 0.8689
```

```
#accuracy score

results = model.evaluate(test_data.batch(512), verbose=2)

for name, value in zip(model.metrics_names, results):
  print("%s: %.3f" % (name, value))
```

```
49/49 - 5s - loss: 0.3580 - accuracy: 0.8532 - 5s/epoch - 101ms/step
loss: 0.358
accuracy: 0.853
```

```
#LOGIC FOR NEGATIVE REVIEW

pred = model.predict(["This was an absolutely terrible movie"])
if(pred[0][0]<0) :
  print("Negative Review ")
else :
  print("Positive Review ")
```

```
1/1 [==============================] - 0s 182ms/step
Negative Review
```

```
#LOGIC FOR POSITIVE REVIEW
```

```
pred = model.predict(["This was an amazing movie"])
if(pred[0][0]<0) :
  print("Negative Review ")
else :
  print("Positive Review ")
```

```
1/1 [==============================] - 0s 96ms/step
Positive Review
```