```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/driv
```

```
import pandas as pd
import tensorflow as tf
import numpy as np
```

```
df = pd.read_csv('/content/tmdb_5000_movies.csv')
df.head(3)
```

	budget	genres	homepage	id	keywords	original_language	or
0	237000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.avatarmovie.com/	19995	[{"id": 1463, "name": "culture clash"}, {"id":...	en	
1	300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "...	http://disney.go.com/disneypictures/pirates/	285	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	en	
2	245000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.sonypictures.com/movies/spectre/	206647	[{"id": 470, "name": "spy"}, {"id": 818, "name...	en	

Next steps: Generate code with df | New interactive sheet

```
df = df['original_title']
```

```
df
```

	original_title
0	Avatar
1	Pirates of the Caribbean: At World's End
2	Spectre
3	The Dark Knight Rises
4	John Carter
...	...
2430	Evil Dead
2431	My Life in Ruins
2432	American Dreamz
2433	Superman IV: The Quest for Peace
2434	How She Move

2435 rows × 1 columns

dtype: object

```
movie_name = df.to_list()
```

```
movie_name
```

```
 'Mad City',
 "Baby's Day Out",
 'The Scarlet Letter',
 'Fair Game',
 'Domino',
 'Jade',
 'Gamer',
 'Beautiful Creatures',
 'Death to Smoochy',
 'Zoolander 2',
 'The Big Bounce',
 'What Planet Are You From?',
 ...]
```

```
tokenizer = tf.keras.preprocessing.text.Tokenizer()

tokenizer.fit_on_texts(movie_name)

seq = tokenizer.texts_to_sequences(movie_name)
```

```
seq[:10]
```

```
[[870],
 [141, 2, 1, 183, 45, 282, 105],
 [871],
 [1, 46, 142, 872],
 [283, 284],
 [106, 8, 15],
 [873],
 [464, 60, 2, 874],
 [74, 90, 5, 1, 465, 61, 62],
 [107, 285, 108, 63, 2, 875]]
```

```
tokenizer.word_index
```

```
 'catching': 984,
 'puss': 985,
 'boots': 986,
 'salt': 987,
 'noah': 988,
 'tintin': 989,
 'azkaban': 990,
 'australia': 991,
 'megamind': 992,
 "philosopher's": 993,
 'r': 994,
 'da': 995,
 'vinci': 996,
 'x2': 997,
 'clash': 998,
 'total': 999,
 'recall': 1000,
 ...}
```

```
x = []
y = []
total_words_dropped = 0

for i in seq:
  if len(i) >1:
    for index in range(1, len(i)):
      x.append(i[:index])
      y.append(i[index])
  else:
    total_words_dropped += 1
print("Total Single Words Dropped are:", total_words_dropped)
```

```
Total Single Words Dropped are: 500
```

```
x[:10]
```

```
[[141],
 [141, 2],
 [141, 2, 1],
 [141, 2, 1, 183],
 [141, 2, 1, 183, 45],
 [141, 2, 1, 183, 45, 282],
 [1],
 [1, 46],
 [1, 46, 142],
 [283]]
```

```
y[:10]
```

```
[2, 1, 183, 45, 282, 105, 46, 142, 872, 284]
```

```
x = tf.keras.preprocessing.sequence.pad_sequences(x)
```

```
x
```

```
array([[  0,   0,   0, ...,   0,   0, 141],
       [  0,   0,   0, ...,   0, 141,   2],
       [  0,   0,   0, ..., 141,   2,   1],
       ...,
       [  0,   0,   0, ...,   1, 423,  11],
       [  0,   0,   0, ...,   0,   0,  53],
       [  0,   0,   0, ...,   0,  53, 443]], dtype=int32)
```

```
x.shape
```

```
(4285, 11)
```

```
y = tf.keras.utils.to_categorical(y)
```

```
y
```

```
array([[0., 0., 1., ..., 0., 0., 0.],
       [0., 1., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 1., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 1.]])
```

```
y.shape
```

```
(4285, 2910)
```

```
vocab_size = len(tokenizer.word_index) + 1
```

```
vocab_size
```

```
2910
```

```
model = tf.keras.Sequential([
    tf.keras.layers.Embedding(vocab_size, 14),
    tf.keras.layers.LSTM(100, return_sequences=True),
    tf.keras.layers.LSTM(100),
    tf.keras.layers.Dense(100, activation='relu'),
    tf.keras.layers.Dense(vocab_size, activation='softmax')
])
```

```
model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
embedding (Embedding)	?	0 (unbuilt)
lstm (LSTM)	?	0 (unbuilt)
lstm_1 (LSTM)	?	0 (unbuilt)
dense (Dense)	?	0 (unbuilt)
dense_1 (Dense)	?	0 (unbuilt)

Total params: 0 (0.00 B)
Trainable params: 0 (0.00 B)
Non-trainable params: 0 (0.00 B)

```
model.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),
    loss = 'categorical_crossentropy',
    metrics=['accuracy']
)
```

```
model.fit(x, y, epochs=250)
```

```
134/134 ━━━━━━━━━━━━━━━━━━━━ 2s 10ms/step - accuracy: 0.6446 - loss: 1.4438
Epoch 230/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 2s 13ms/step - accuracy: 0.6425 - loss: 1.4889
Epoch 231/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 2s 15ms/step - accuracy: 0.6603 - loss: 1.3913
Epoch 232/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 2s 12ms/step - accuracy: 0.6635 - loss: 1.4036
Epoch 233/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 1s 11ms/step - accuracy: 0.6621 - loss: 1.3776
Epoch 234/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 1s 10ms/step - accuracy: 0.6581 - loss: 1.4113
Epoch 235/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 1s 10ms/step - accuracy: 0.6594 - loss: 1.4169
Epoch 236/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 1s 10ms/step - accuracy: 0.6514 - loss: 1.4378
Epoch 237/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 1s 10ms/step - accuracy: 0.6594 - loss: 1.3839
Epoch 238/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 1s 10ms/step - accuracy: 0.6658 - loss: 1.3836
Epoch 239/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 2s 14ms/step - accuracy: 0.6674 - loss: 1.3763
Epoch 240/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 2s 15ms/step - accuracy: 0.6633 - loss: 1.3800
Epoch 241/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 2s 12ms/step - accuracy: 0.6593 - loss: 1.4176
Epoch 242/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 1s 10ms/step - accuracy: 0.6653 - loss: 1.4156
Epoch 243/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 1s 10ms/step - accuracy: 0.6679 - loss: 1.3487
Epoch 244/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 1s 11ms/step - accuracy: 0.6762 - loss: 1.3356
Epoch 245/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 1s 10ms/step - accuracy: 0.6504 - loss: 1.4116
Epoch 246/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 1s 10ms/step - accuracy: 0.6504 - loss: 1.4367
Epoch 247/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 1s 11ms/step - accuracy: 0.6560 - loss: 1.3700
Epoch 248/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 2s 15ms/step - accuracy: 0.6699 - loss: 1.4094
Epoch 249/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 2s 16ms/step - accuracy: 0.6686 - loss: 1.3593
Epoch 250/250
134/134 ━━━━━━━━━━━━━━━━━━━━ 2s 11ms/step - accuracy: 0.6824 - loss: 1.3273
<keras.src.callbacks.history.History at 0x7886a610c710>
```

```
model.save('nwp.h5')
```

```
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(mo
```

```
import os
print("current working directory:", os.getcwd())
```

```
current working directory: /content
```

```
vocab_array = np.array(list(tokenizer.word_index.keys()))
```

```
vocab_array
```

```
array(['the', 'of', '2', ..., 'dreamz', 'peace', 'move'], dtype='<U14')
```

```
def make_prediction(text, n_words):
  for i in range(n_words):
    text_tokenize = tokenizer.texts_to_sequences([text])
    text_padded = tf.keras.preprocessing.sequence.pad_sequences(text_tokenize, maxlen=14)
    prediction = np.squeeze(np.argmax(model.predict(text_padded), axis = -1))
    prediction = str(vocab_array[prediction - 1])
    print(vocab_array[np.argsort(model.predict(text_padded)) - 1].ravel()[:-3])
    text +=" " + prediction
  return text
```

```
make_prediction("cloudy",10)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 263ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step
['godfather' 'treader' 'chocolate' ... 'a' 'to' 'just']
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step
['abiding' 'ides' 'smell' ... 'fire' '40' 'you']
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 32ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step
['gate' 'bosses' 'monte' ... 'attic' 'grave' 'revenge']
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step
['justice' 'invisible' 'fox' ... '2' 'soldiers' 'to']
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 32ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step
['dusk' 'but' 'arrow' ... '3' 'i' '2']
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 32ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step
['instinct' 'ransom' 'w' ... 'you' '3' 'dark']
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step
['naked' 'funeral' 'bigger' ... 'from' 'men' 'vs']
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 32ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step
['chocolate' 'report' 'insult' ... 'city' 'legend' '2']
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 51ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 33ms/step
['hoax' 'andreas' 'olympiques' ... 'with' 'to' 'in']
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 35ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step
['pi' 'jeux' 'reborn' ... 'dawn' 'the' '3']
'cloudy with the air of cholera water 3 jenkins 1 2'
```

```
make_prediction("mars", 10)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 68ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 29ms/step
['mess' 'gambler' 'diaries' ... 'ops' 'twist' 'beverly']
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 32ms/step
['university' 'mommas' 'libertine' ... 'game' 'eyes' 'after']
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 32ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 32ms/step
['possession' 'ugly' 'hell' ... 'or' 'war' '3d']
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 29ms/step
['union' 'risk' 'ass' ... 'forward' 'county' 'know']
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 32ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 33ms/step
['duck' '19' 'saint' ... 'me' 'all' '2']
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 38ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 32ms/step
['pink' 'opus' "holland's" ... 'at' 'to' 'los']
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 37ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step
['very' 'fall' 'naked' ... 'day' 'dinosaurs' 'the']
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 33ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 45ms/step
['bfg' 'proposal' 'hustle' ... 'last' 'ii' '3']
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 49ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 49ms/step
['ever' 'reaction' 'scientists' ... 'island' 'money' 'things']
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 51ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 49ms/step
['simple' 'if' 'thirty' ... 'of' 'man' 'dead']
'mars needs moms her groove back by shadows the west story'
```