VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

Programme	:	**M.Tech Software Engineering**	Semester	:	**Fall2021**
Course	:	**Natural Language Processing**	Code	:	**SWE1017**
Faculty	:	**Dr. Tulasi Prasad Sarkar**	Slot	:	**G1**

NLP FINAL REVIEW DOCUMENT

PROJECT TITLE

TEXT SUMMARIZATION USING TEXT RANKING

TEAM MEMBERS

Setty Ruthvik-18MIS1048

Supriya.T-18MIS1064

PROGRAM CODE

```
import numpy as np

import pandas as pd

import nltk

from keras import backend as K

from matplotlib import pyplot

nltk.download('punkt') # one time execution

import re

df = pd.read_csv("/Users/DELL/Desktop/tennis_articles_v4.csv")

df.head()
```

```
df['article_text'][0]

df['article_text'][1]

df['article_text'][2]

from nltk.tokenize import sent_tokenize

sentences = []

for s in df['article_text']:sentences.append(sent_tokenize(s))

sentences = [y for x in sentences for y in x] # flatten list

sentences[:5]

!wget http://nlp.stanford.edu/data/glove.6B.zip

!unzip glove*.zip

# Extract word vectors

word_embeddings = {}

f = open('glove.6B.100d.txt', encoding='utf-8')

for line in f:values = line.split()

word = values[0]

coefs = np.asarray(values[1:], dtype='float32')

word_embeddings[word] = coefs

f.close()

len(word_embeddings)

# remove punctuations, numbers and special characters

clean_sentences = pd.Series(sentences).str.replace("[^a-zA-Z]", " ")
```

```
# make alphabets lowercase

clean_sentences = [s.lower() for s in clean_sentences]

nltk.download('stopwords')

from nltk.corpus import stopwords

stop_words = stopwords.words('english')

# function to remove stopwords

def remove_stopwords(sen):sen_new = " ".join([i for i in sen if i not in stop_words])

returnsen_new

# removestopwords from the sentences

clean_sentences = [remove_stopwords(r.split()) for r in clean_sentences]

# Extract word vectors

word_embeddings = {}

f = open('glove.6B.100d.txt', encoding='utf-8')

for line in f:values  = line.split()

word = values[0]

coefs = np.asarray(values[1:], dtype='float32')

word_embeddings[word] = coefs

f.close()

sentence_vectors = []

for i in clean_sentences:

    if len(i) != 0:v = sum([word_embeddings.get(w, np.zeros((100,))) for w in i.split
()])/(len(i.split())+0.001)

else:v = np.zeros((100,))
```

```
sentence_vectors.append(v)

# similarity matrix

sim_mat = np.zeros([len(sentences), len(sentences)])

from sklearn.metrics.pairwise import cosine_similarity

for i in range(len(sentences)):

    for j in range(len(sentences)):

        if i != j:sim_mat[i][j] = cosine_similarity(sentence_vectors[i].reshape(1,100),sentence_vectors[j].reshape(1,100))[O,0]

import networkx as nx

nx_graph = nx.from_numpy_array(sim_mat)

scores = nx.pagerank(nx_graph)

ranked_sentences = sorted(((scores[i],s) for i,s in enumerate(sentences)), reverse=True)

# Extract top 10 sentences as the summary

for i in range(10):print(ranked_sentences[i][1])

#model building

K.clear_session()

latent_dim = 500

encoder_inputs = Input(shape=(max_len_text,))

enc_emb = Embedding(x_voc_size, latent_dim,trainable=True)(encoder_inputs)

encoder_lstm1 = LSTM(latent_dim,return_sequences=True,return_state=True)

encoder_output1, state_h1, state_c1 = encoder_lstm1(enc_emb)

encoder_lstm2 = LSTM(latent_dim,return_sequences=True,return_state=True)
```

```
encoder_output2, state_h2, state_c2 = encoder_lstm2(encoder_output1)

decoder_inputs = Input(shape=(None,))

dec_emb_layer = Embedding(y_voc_size, latent_dim,trainable=True)

dec_emb = dec_emb_layer(decoder_inputs)

decoder_lstm = LSTM(latent_dim, return_sequences=True, return_state=True)

decoder_outputs,decoder_fwd_state, decoder_back_state = decoder_lstm(dec_emb,initial_state=[state_h, state_c])

Attention layer attn_layer = AttentionLayer(name='attention_layer')

attn_out, attn_states = attn_layer([encoder_outputs, decoder_outputs])

decoder_concat_input = Concatenate(axis=-1, name='concat_layer')([decoder_outputs, attn_out])

decoder_dense = TimeDistributed(Dense(y_voc_size, activation='softmax'))

decoder_outputs = decoder_dense(decoder_concat_input)

# Define the model

model = Model([encoder_inputs, decoder_inputs], decoder_outputs)

model.summary()

model.compile(optimizer='rmsprop', loss='sparse_categorical_crossentropy')

history=model.fit([x_tr,y_tr[:,:-1]], y_tr.reshape(y_tr.shape[0],y_tr.shape[1], 1)[:,1:] ,epochs=50,callbacks=[es],batch_size=512, validation_data=([x_val,y_val[:,:-1]], y_val.reshape(y_val.shape[0],y_val.shape[1], 1)[:,1:]))

pyplot.plot(history.history['loss'], label='train')

pyplot.plot(history.history['val_loss'], label='test')

pyplot.legend() pyplot.show()

def seq2summary(input_seq):
    newString=''
    for i in input_seq:
      if((i!=0 and i!=target_word_index['start']) and i!=target_word_index['end']):
        newString=newString+reverse_target_word_index[i]+' '
    return newString


def seq2text(input_seq):
    newString=''
    for i in input_seq:
      if(i!=0):
        newString=newString+reverse_source_word_index[i]+' '
    return newString
for i in range(len(x_val)):
  print("Review:",seq2text(x_val[i]))
  print("Original summary:",seq2summary(y_val[i]))
  print("Predicted summary:",decode_sequence(x_val[i].reshape(1,max_len_text)))
  print("\n")
```

SUMMARIZED OUTPUT

When I'm on the courts or when I'm on the court playing, I'm a competitor and I want to beat every single person whether they're in the locker room or across the net.So I'm not the one to strike up a conversation about the

weather and know that in the next few minutes I have to go and try to win a tennis match.

Major players feel that a big event in late November combined with one in January before the Australian Open will mean too much tennis and too little rest.

Speaking at the Swiss Indoors tournament where he will play in Sundays final against Romanian qualifier Marius Copil, the world number three said that given the impossibly short time frame to make a decision, he opted out of any commitment.

"I felt like the best weeks that I had to get to know players when I was playing were the Fed Cup weeks or the Olympic weeks, not necessarily during the tournaments.

Currently in ninth place, Nishikori with a win could move to within 125 points of the cut for the eight-man event in London next month.

He used his first break point to close out the first set before going up 3-0 in the second and wrapping up the win on his first match point.

The Spaniard broke Anderson twice in the second but didn't get another chance on the South African's serve in the final set.

"We also had the impression that at this stage it might be better to play matches than to train.

The competition is set to feature 18 countries in the November 18-24 finals in Madrid next year, and will replace the classic home-and-away ties played four times per year for decades.

Federer said earlier this month in Shanghai in that his chances of playing the Davis Cup were all but non-existent.

The top 10 sentences are selected and displayed as summary of the article

OUTPUT SCREENSHOTS

```
In[1]:
```

```
importnumpyas npimport
pandas as pdimport nltk

nltk.download('punkt') # one time execution
```

```
[nltk_data] Downloading package punkt to
[nltk_data]      C:\Users\dell\AppData\Roaming\nltk_data...
[nltk_data]    Package punktis alreadyup-to-date!
```

```
In[2]:    df= pd.read_csv("tennis_articles_v4.csv")

In[3]:    df.head()

Out[3]:
```

	article_id	article_text	source
0	1	Maria Sharapova has basically no friends aste...	https://www.tennisworldusa.org/tennis/news/Mar...
1	2	BASEL, Switzerland (AP), Roger Federer advance...	http://www.tennis.com/pro-game/2018/10/copil-s...
2	3	Roger Federer has revealed thatorganisersof...	https://scroll.in/field/899938/tennis-roger-fe...
4	5	Federer, 37, first broke through on tour over ...	https://www.express.co.uk/sport/tennis/1036101...
3	4	Kei Nishikoriwill try to end his long losing...	http://www.tennis.com/pro-game/2018/10/nishiko...

```
In [4]:
          df['article_text'][0]
Out[4]:"MariaSharapovahasbasicallynofriendsastennisplayersontheWTATour.TheRussianplayerh
        asnoproblemsinopenlyspeakingaboutitandinarecentinterviewshesaid:'Idon'treallyhi
        deanyfeelingstoomuch.Ithinkeveryoneknowsthisismyjobhere.WhenI'monthecourtsorw
        henI'monthecourtplaying,I'macompetitorandIwanttobeateverysinglepersonwhetherthe
        y'reinthelockerroomoracrossthenet.SoI'mnottheonetostrikeupaconversationabouth
        eweatherandknowthatinthenextfewminutesIhavetogoandtrytowinatennismatch.I'ma pretty
        competitive girl. I say my hellos, but I'm not sending any players flowers as well. Uhm,
        I'mnotreallyfriendlyorclosetomanyplayers.Ihavenotalotoffriendsawayfromthecourt
        s.'Whenshesaidsheisnotreallyclosetoalotofplayers,isthatsomethingstrategicthats
        heisdoing?Isitdifferentonthemen'stourthanthewomen'stour?'No,notatall.Ithinkjustbecauseyou'rei
        nthesamesportdoesn'tmeanthatyouhavetobefriendswitheveryonejustb
        ecauseyou'recategorized,you'reatennisplayer,soyou'regoingtogetalongwithtennisplayers.Ithinkev
        erypersonhasdifferentinterests.Ihavefriendsthathavecompletelydifferentj
        obsandinterests,andI'vemettheminverydifferentpartsofmylife.Ithinkeveryonejustthinksbecausewe'
        retennisplayersweshouldbethegreatestoffriends.Butultimatelytennisis
        justaverysmallpartofwhatwedo.Therearesomanyotherthingsthatwe'reinterestedin,th at wedo.'"

In [5]:
          df['article_text'][1]
Out[5]:"BASEL,Switzerland(AP),RogerFedereradvancedtothe14thSwissIndoorsfinalofhiscareerby
```

```
beatingseventh-seededDaniilMedvedev6-1,6-
4onSaturday.Seekinganinthtitleathishometownevent,anda99thoverall,Federerwillplay93th-
rankedMariusCopilonSunday.Federerdominatedthe20th-rankedMedvedevandhadhisfirstmatch-
pointchancetobreakserveagainat5-1.He
thendroppedhisservetolove,andletanothermatchpointslipinMedvedev'snextservicegame
bynettingabackhand.HeclinchedonhisfourthchancewhenMedvedevnettedfromthebaseline.C
opilupsetexpectationsofaFedererfinalagainstAlexanderZverevina6-3,6-7(6),6-4winoverthefifth-
rankedGermanintheearliersemifinal.TheRomanianaimsforafirsttitleafterarrivingatBaselwithoutaca
reerwinoveratop-10opponent.CopilhastwoafteralsobeatingNo.
6MarinCilicinthesecondround.Copilfired26acespastZverevandneverdroppedserve,clinchingafter21/2
hourswithaforehandvolleywinnertobreakZverevforthesecondtimeinthes
emifinal.HecamethroughtworoundsofqualifyinglastweekendtoreachtheBaselmaindraw,includingbeatin
gZverev'solderbrother,Mischa.FedererhadaneasiertimethaninhisonlypreviousmatchagainstMedvedev,
athree-setteratShanghaitwoweeksago."
```

```
In[6]:
        ##SPLITTING INTO SENTENES

        fromnltk.tokenizeimport sent_tokenize sentences = []

        fors in df['article_text']:

          sentences.append(sent_tokenize(s))
```

```
In[7]:  sentences[:5]
```

```
Out[7]: ['Maria Sharapovahas basically no friends as tennis players on the WTATour.',
         "TheRussianplayerhasnoproblemsinopenlyspeakingaboutitandinarecentinterviewshesaid: 'I don't
        really hide any feelings toomuch.",
         'I think everyone knows this is my jobhere.',
         "WhenI'monthecourtsorwhenI'monthecourtplaying,I'macompetitorandIwanttobeateverysingleperson
        whetherthey'reinthelockerroomoracrossthenet.SoI'mnottheonetostrikeupaconversationabouttheweat
        herandknowthatinthenextfewminutesIhavetogoandtry to win a tennismatch.",
         "I'm a pretty competitive girl."]
```

```
In[8]:
        #FROM GLOVE WORD EMBEDDINGS

        # Extract word vectors

        word_embeddings= {}
        f = open('glove.6B.100d.txt', encoding='utf-8')
        forline in f:

            values= line.split()
```

```
In[9]:  len(word_embeddings)
```

```
Out[9]: 400000
```

```
In[10]:
        #TEXT PROCESSING

        # remove punctuations, numbers and special characters

        clean_sentences= pd.Series(sentences).str.replace("[^a-zA-Z]", " ")
```

```
In[11]: nltk.download('stopwords')

        [nltk_data] Downloading package stopwords to
        [nltk_data]     C:\Users\dell\AppData\Roaming\nltk_data...
        [nltk_data]   Package stopwordsis alreadyup-to-date!
```

```
Out[11]: True
```

```
In[13]:
        fromnltk.corpusimport stopwordsstop_words=
        stopwords.words('english')
```

```
In[14]:
        ## define a function to remove these stopwordsfrom our dataset. # functionto remove stopwords

        defremove_stopwords(sen):
            sen_new= " ".join([i for i in senif i not in stop_words])

            returnsen_new
```

```
In[15]:
        #Vector Representation of Sentences # Extract word
        vectors word_embeddings= {}

        f = open('glove.6B.100d.txt', encoding='utf-8')
        forline in f:

            values= line.split()
            word = values[0]
            coefs= np.asarray(values[1:], dtype='float32')
            word_embeddings[word] = coefs
```

In[16]:

```
sentence_vectors= []
fori in clean_sentences:

  iflen(i) != 0:
    v = sum([word_embeddings.get(w, np.zeros((100,))) for w ini.split()])/(len(i.split())+0.001)
  else:
```

In[17]:

```
# similarity matrix

#We will use Cosine Similarity to compute the similarity between a pair of sentences.
```

In[18]:

```
# initialize the matrix with cosine similarity scores.

fori in range(len(sentences)):

  forj in range(len(sentences)):
```

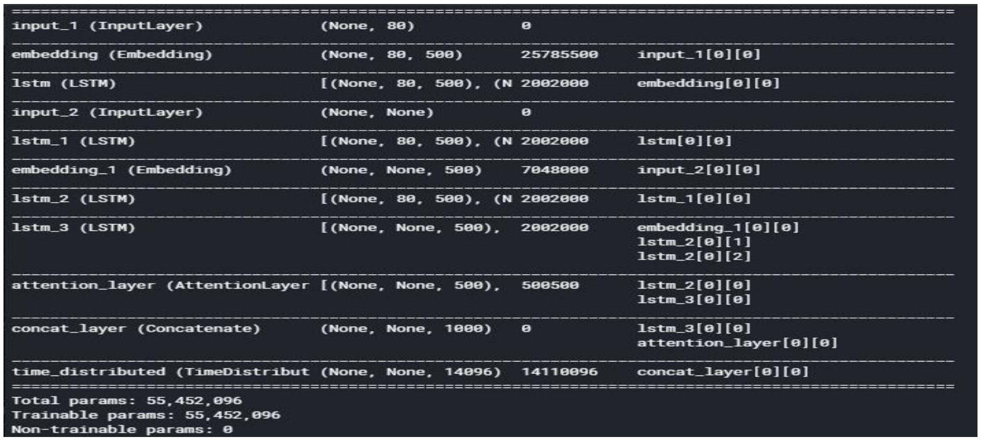
In[19]:

```
#Applying PageRank Algorithm

importnetworkxas nx
```

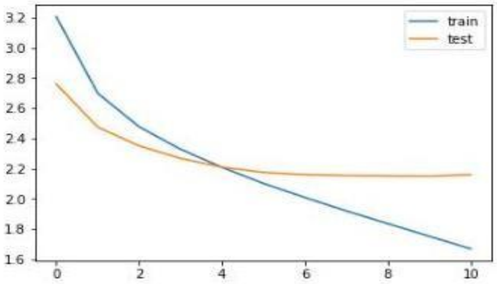
In[21]:

```
#Summary Extraction

ranked_sentences= sorted(((scores[i],s) for i,sin enumerate(sentences)),reverse=True)
# Extract top 10 sentences as the summary
```

```
WhenI'monthecourtsorwhenI'monthecourtplaying,I'macompetitorandIwanttobeatever
ysinglepersonwhetherthey'reinthelockerroomoracrossthenet.SoI'mnottheonetostrike
upaconversationabouttheweatherandknowthatinthenextfewminutesIhavetogoandtryto        win      a
tennismatch.
MajorplayersfeelthatabigeventinlateNovembercombinedwithoneinJanuarybeforetheAustralianOpenwil
lmeantoomuchtennisandtoolittlerest.
SpeakingattheSwissIndoorstournamentwherehewillplayinSundaysfinalagainstRomanianqua
lifierMariusCopil,theworldnumberthreesaidthatgiventheimpossiblyshorttimeframetoma        ke     a
decision, he opted out ofany commitment.
"IfeltlikethebestweeksthatIhadtogettoknowplayerswhenIwasplayingweretheFedCup weeks or the
Olympic weeks, not necessarily during thetournaments.
Currentlyinninthplace,Nishikoriwithawincouldmovetowithin125pointsofthecutforthe        eight-man
event in London nextmonth.
```



```
==================================================================================================
input_1 (InputLayer)            (None, 80)           0
__________________________________________________________________________________________________
embedding (Embedding)           (None, 80, 500)      25785500    input_1[0][0]
__________________________________________________________________________________________________
lstm (LSTM)                     [(None, 80, 500), (N 2002000     embedding[0][0]
__________________________________________________________________________________________________
input_2 (InputLayer)            (None, None)         0
__________________________________________________________________________________________________
lstm_1 (LSTM)                   [(None, 80, 500), (N 2002000     lstm[0][0]
__________________________________________________________________________________________________
embedding_1 (Embedding)         (None, None, 500)    7048000     input_2[0][0]
__________________________________________________________________________________________________
lstm_2 (LSTM)                   [(None, 80, 500), (N 2002000     lstm_1[0][0]
__________________________________________________________________________________________________
lstm_3 (LSTM)                   [(None, None, 500),  2002000     embedding_1[0][0]
                                                                 lstm_2[0][1]
                                                                 lstm_2[0][2]
__________________________________________________________________________________________________
attention_layer (AttentionLayer [(None, None, 500),  500500      lstm_2[0][0]
                                                                 lstm_3[0][0]
__________________________________________________________________________________________________
concat_layer (Concatenate)      (None, None, 1000)   0           lstm_3[0][0]
                                                                 attention_layer[0][0]
__________________________________________________________________________________________________
time_distributed (TimeDistribut (None, None, 14096)  14110096    concat_layer[0][0]
==================================================================================================
Total params: 55,452,096
Trainable params: 55,452,096
Non-trainable params: 0
```



```
Predicted output:['Maria Sharapovahas basically no friends as tennis players on the
WTATour.',"TheRussianplayerhasnoproblemsinopenlyspeakingaboutitandinarecentinterviewsh
esaid: 'I don't really hide any feelings toomuch.",'I think everyone knows this is my
jobhere.',"WhenI'monthecourtsorwhenI'monthecourtplaying,I'macompetitorandIwanttobeate
verysinglepersonwhetherthey'reinthelockerroomoracrossthenet.SoI'mnottheonetostrikeupa
conversationabouttheweatherandknowthatinthenextfewminutesIhavetogoandtry to win a
tennismatch.","I'm a pretty competitive girl."]
```

Statistical Summarization

Iris DataSet :

sepal_length	sepal_width	petal_length	petal_width	Species
5.1	3.5	1.4	0.2	Setosa
4.9	3	1.4	0.2	Setosa
4.7	3.2	1.3	0.2	Setosa
4.6	3.1	1.5	0.2	Setosa
5	3.6	1.4	0.2	Setosa
5.4	3.9	1.7	0.4	Setosa
4.6	3.4	1.4	0.3	Setosa
5	3.4	1.5	0.2	Setosa
4.4	2.9	1.4	0.2	Setosa
4.9	3.1	1.5	0.1	Setosa
5.4	3.7	1.5	0.2	Setosa
4.8	3.4	1.6	0.2	Setosa
4.8	3	1.4	0.1	Setosa
4.3	3	1.1	0.1	Setosa
5.8	4	1.2	0.2	Setosa
5.7	4.4	1.5	0.4	Setosa
5.4	3.9	1.3	0.4	Setosa
5.1	3.5	1.4	0.3	Setosa
5.7	3.8	1.7	0.3	Setosa
5.1	3.8	1.5	0.3	Setosa
5.4	3.4	1.7	0.2	Setosa
5.1	3.7	1.5	0.4	Setosa
4.6	3.6	1	0.2	Setosa
5.1	3.3	1.7	0.5	Setosa
4.8	3.4	1.9	0.2	Setosa
5	3	1.6	0.2	Setosa
5	3.4	1.6	0.4	Setosa
5.2	3.5	1.5	0.2	Setosa
5.2	3.4	1.4	0.2	Setosa
4.7	3.2	1.6	0.2	Setosa
4.8	3.1	1.6	0.2	Setosa
5.4	3.4	1.5	0.4	Setosa
5.2	4.1	1.5	0.1	Setosa
5.5	4.2	1.4	0.2	Setosa
4.9	3.1	1.5	0.2	Setosa
5	3.2	1.2	0.2	Setosa
5.5	3.5	1.3	0.2	Setosa
4.9	3.6	1.4	0.1	Setosa
4.4	3	1.3	0.2	Setosa
5.1	3.4	1.5	0.2	Setosa
5	3.5	1.3	0.3	Setosa
4.5	2.3	1.3	0.3	Setosa

4.4	3.2	1.3	0.2	Setosa
5	3.5	1.6	0.6	Setosa
5.1	3.8	1.9	0.4	Setosa
4.8	3	1.4	0.3	Setosa
5.1	3.8	1.6	0.2	Setosa
4.6	3.2	1.4	0.2	Setosa
5.3	3.7	1.5	0.2	Setosa
5	3.3	1.4	0.2	Setosa
7	3.2	4.7	1.4	Versicolor
6.4	3.2	4.5	1.5	Versicolor
6.9	3.1	4.9	1.5	Versicolor
5.5	2.3	4	1.3	Versicolor
6.5	2.8	4.6	1.5	Versicolor
5.7	2.8	4.5	1.3	Versicolor
6.3	3.3	4.7	1.6	Versicolor
4.9	2.4	3.3	1	Versicolor
6.6	2.9	4.6	1.3	Versicolor
5.2	2.7	3.9	1.4	Versicolor
5	2	3.5	1	Versicolor
5.9	3	4.2	1.5	Versicolor
6	2.2	4	1	Versicolor
6.1	2.9	4.7	1.4	Versicolor
5.6	2.9	3.6	1.3	Versicolor
6.7	3.1	4.4	1.4	Versicolor
5.6	3	4.5	1.5	Versicolor
5.8	2.7	4.1	1	Versicolor
6.2	2.2	4.5	1.5	Versicolor
5.6	2.5	3.9	1.1	Versicolor
5.9	3.2	4.8	1.8	Versicolor
6.1	2.8	4	1.3	Versicolor
6.3	2.5	4.9	1.5	Versicolor
6.1	2.8	4.7	1.2	Versicolor
6.4	2.9	4.3	1.3	Versicolor
6.6	3	4.4	1.4	Versicolor
6.8	2.8	4.8	1.4	Versicolor
6.7	3	5	1.7	Versicolor
6	2.9	4.5	1.5	Versicolor
5.7	2.6	3.5	1	Versicolor
5.5	2.4	3.8	1.1	Versicolor
5.5	2.4	3.7	1	Versicolor
5.8	2.7	3.9	1.2	Versicolor
6	2.7	5.1	1.6	Versicolor
5.4	3	4.5	1.5	Versicolor

6	3.4	4.5	1.6	Versicolor
6.7	3.1	4.7	1.5	Versicolor
6.3	2.3	4.4	1.3	Versicolor
5.6	3	4.1	1.3	Versicolor
5.5	2.5	4	1.3	Versicolor
5.5	2.6	4.4	1.2	Versicolor
6.1	3	4.6	1.4	Versicolor
5.8	2.6	4	1.2	Versicolor
5	2.3	3.3	1	Versicolor
5.6	2.7	4.2	1.3	Versicolor
5.7	3	4.2	1.2	Versicolor
5.7	2.9	4.2	1.3	Versicolor
6.2	2.9	4.3	1.3	Versicolor
5.1	2.5	3	1.1	Versicolor
5.7	2.8	4.1	1.3	Versicolor
6.3	3.3	6	2.5	Virginica
5.8	2.7	5.1	1.9	Virginica
7.1	3	5.9	2.1	Virginica
6.3	2.9	5.6	1.8	Virginica
6.5	3	5.8	2.2	Virginica
7.6	3	6.6	2.1	Virginica
4.9	2.5	4.5	1.7	Virginica
7.3	2.9	6.3	1.8	Virginica
6.7	2.5	5.8	1.8	Virginica
7.2	3.6	6.1	2.5	Virginica
6.5	3.2	5.1	2	Virginica
6.4	2.7	5.3	1.9	Virginica
6.8	3	5.5	2.1	Virginica
5.7	2.5	5	2	Virginica
5.8	2.8	5.1	2.4	Virginica
6.4	3.2	5.3	2.3	Virginica
6.5	3	5.5	1.8	Virginica
7.7	3.8	6.7	2.2	Virginica
7.7	2.6	6.9	2.3	Virginica
6	2.2	5	1.5	Virginica
6.9	3.2	5.7	2.3	Virginica
5.6	2.8	4.9	2	Virginica
7.7	2.8	6.7	2	Virginica
6.3	2.7	4.9	1.8	Virginica
6.7	3.3	5.7	2.1	Virginica
7.2	3.2	6	1.8	Virginica
6.2	2.8	4.8	1.8	Virginica
6.1	3	4.9	1.8	Virginica

6.4	2.8	5.6	2.1	Virginica
7.2	3	5.8	1.6	Virginica
7.4	2.8	6.1	1.9	Virginica
7.9	3.8	6.4	2	Virginica
6.4	2.8	5.6	2.2	Virginica
6.3	2.8	5.1	1.5	Virginica
6.1	2.6	5.6	1.4	Virginica
7.7	3	6.1	2.3	Virginica
6.3	3.4	5.6	2.4	Virginica
6.4	3.1	5.5	1.8	Virginica
6	3	4.8	1.8	Virginica
6.9	3.1	5.4	2.1	Virginica
6.7	3.1	5.6	2.4	Virginica
6.9	3.1	5.1	2.3	Virginica
5.8	2.7	5.1	1.9	Virginica
6.8	3.2	5.9	2.3	Virginica
6.7	3.3	5.7	2.5	Virginica
6.7	3	5.2	2.3	Virginica
6.3	2.5	5	1.9	Virginica
6.5	3	5.2	2	Virginica
6.2	3.4	5.4	2.3	Virginica
5.9	3	5.1	1.8	Virginica

Program Code :

```
### Statistical

Summarization

import pandas as pd

import numpy as np from

scipy import stats

    import matplotlib.pyplot as plt

    %matplotlib inline


    # read dataset

    df = pd.read_csv("C:\\Users\\DELL\\Desktop\\iris.csv")


    def histo():

        # create histogram

        bin_edges = np.arange(0, df['sepal_length'].max() + 1, 0.5)

        fig = plt.hist(df['sepal_length'], bins=bin_edges)


        # add plot labels

        plt.xlabel('count')

        plt.ylabel('sepal length')


    histo()

    plt.show()

    x = df['sepal_length'].values
    x.dtype # dtype means type to use in computing the SD. for array of integers,the defualt is
    float64.
```

### Sample Mean:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^n = x_i$$

```
sum(i for i in x) / len(x)
```

```
x_mean = np.mean(x)
x_mean
```

```
histo()
plt.axvline(x_mean, color='darkorange')
plt.show()
```

### Sample Variance:

$$Var_x = \frac{1}{n-1}\sum_{i=1}^n (x_i - \bar{x})^2$$

```
sum([(i - x_mean)**2 for i in x]) / (len(x) - 1)
```

```
var = np.var(x, ddof=1) #ddof means delta degree of freedom. by default ddof =0
var
```

```
df['sepal_length'].var()
```

```
histo()
```

```
plt.axvline(x_mean + var, color='darkorange')
plt.axvline(x_mean - var, color='darkorange')
plt.show()
```

### Sample Standard Deviation:

$$Std_x = \sqrt{\frac{1}{n-1}{\sum_{i=1}^n (x_i - \bar{x})^2}}$$

```
(sum([(i - x_mean)**2 for i in x]) / (len(x) - 1))**0.5
```

```
np.sqrt(np.var(x, ddof=1))
```

```
std = np.std(x, ddof=1)
std
```

```
df['sepal_length'].std() # note that Bessel's correction+ is the default
```

```
histo()
plt.axvline(x_mean + std, color='darkorange')
plt.axvline(x_mean - std, color='darkorange')
plt.show()
```

### Min/Max:

```
np.min(x)
np.max(x)
```

### Mode:

```
lst = list(x)
mode = max(set(lst), key=lst.count)
mode

lst.count(mode)

stats.mode(x)
```

### 25th and 75th Percentile:

```
y = np.sort(x)
percentile_25th = y[round(0.25 * y.shape[0]) + 1]
percentile_25th

percentile_75th = y[round(0.75 * y.shape[0]) - 1]
percentile_75th

np.percentile(x, q=[25, 75], interpolation='lower')

df['sepal_length'].quantile(0.25, interpolation='lower')
```

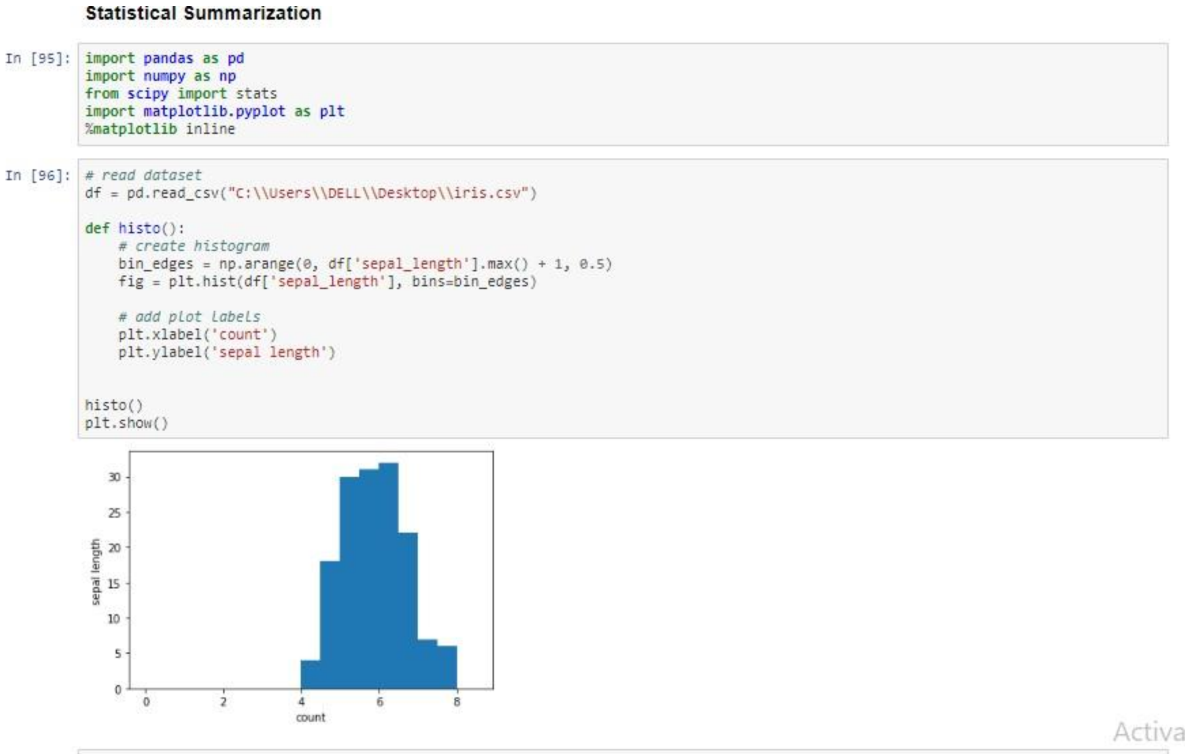
```
df['sepal_length'].quantile(0.75, interpolation='lower')


histo()

plt.axvline(percentile_75th, color='darkorange')

plt.axvline(percentile_25th - var, color='darkorange')

plt.show()
```

### Median (50th Percentile):

```
x = np.sort(x)


tmp = round(0.5 * x.shape[0])


if x.shape[0] % 2:

    median = x[tmp - 1]

else:

    median = x[tmp - 1] + (x[tmp] - x[tmp - 1]) / 2.


median


np.median(x)


histo()

plt.axvline(median, color='darkorange')
```

plt.show()

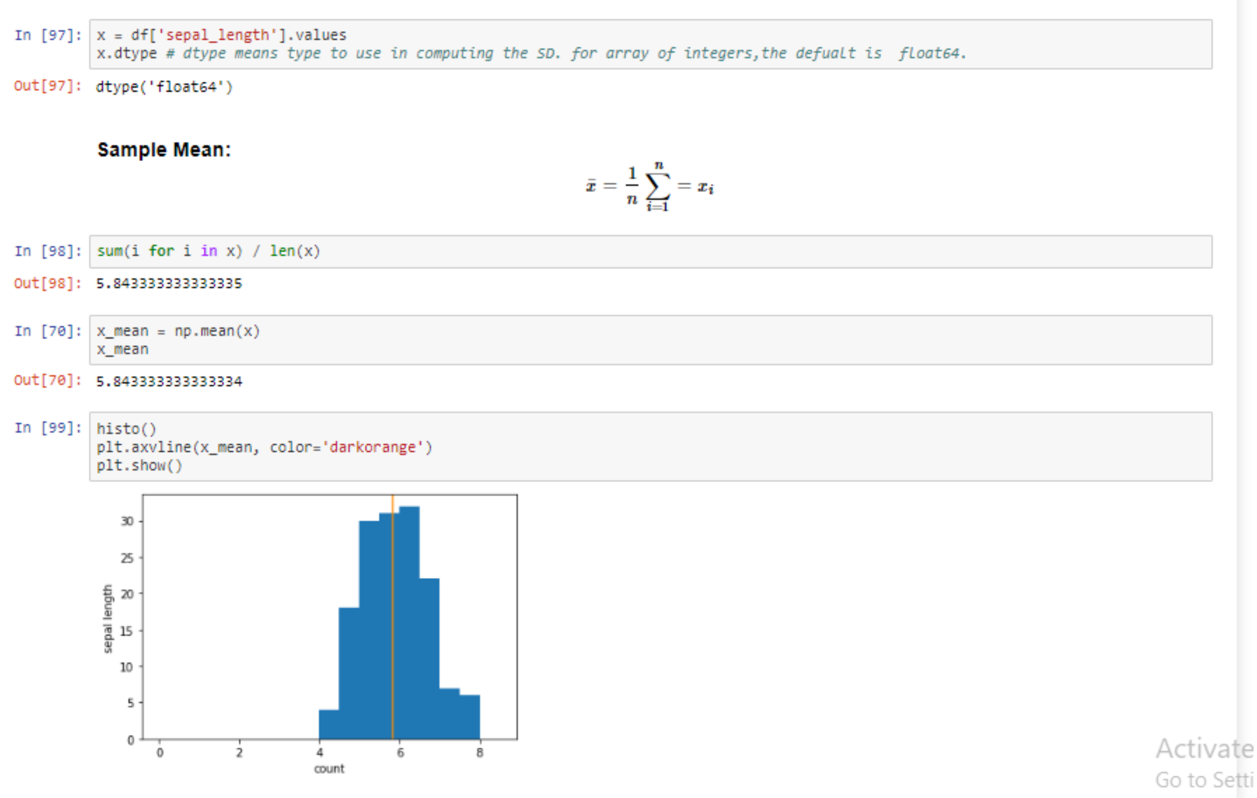
OUTPUT SCREENSHOTS



Statistical Summarization

```
In [95]: import pandas as pd
         import numpy as np
         from scipy import stats
         import matplotlib.pyplot as plt
         %matplotlib inline
```

```
In [96]: # read dataset
         df = pd.read_csv("C:\\Users\\DELL\\Desktop\\iris.csv")

         def histo():
             # create histogram
             bin_edges = np.arange(0, df['sepal_length'].max() + 1, 0.5)
             fig = plt.hist(df['sepal_length'], bins=bin_edges)

             # add plot labels
             plt.xlabel('count')
             plt.ylabel('sepal length')


         histo()
         plt.show()
```

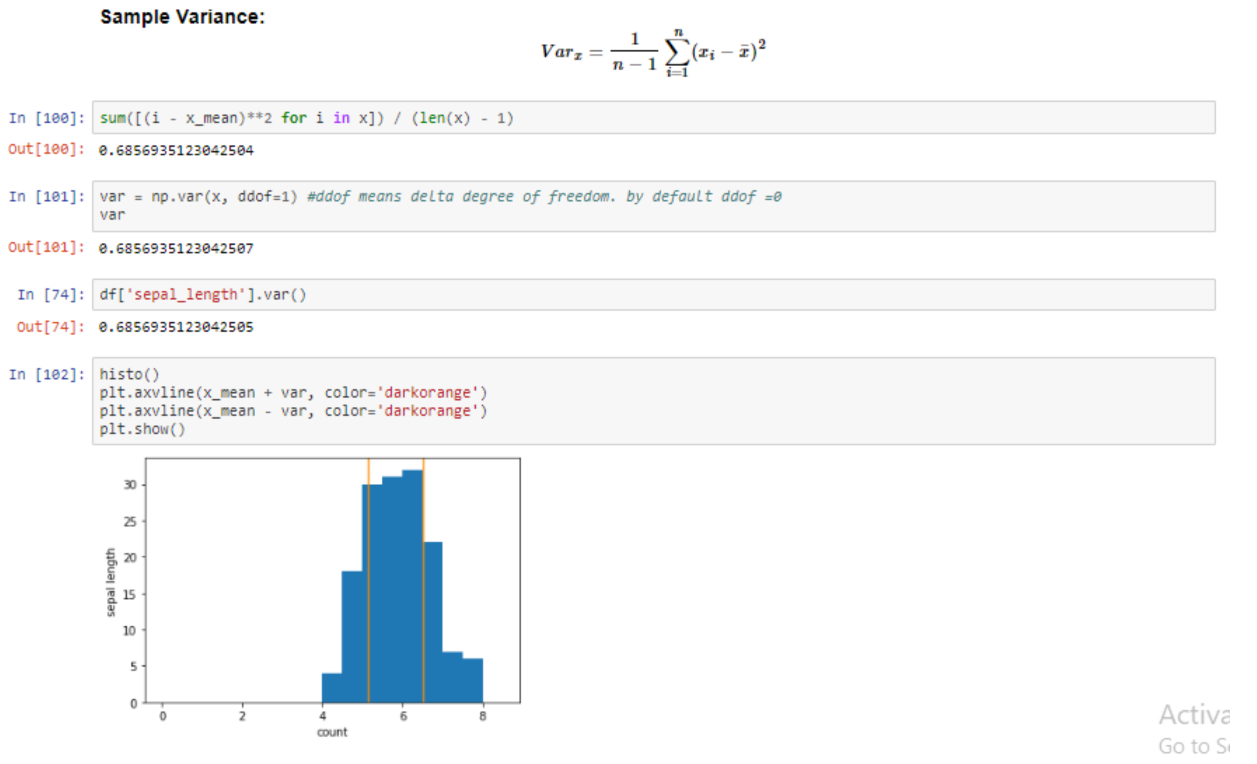


```
In [97]: x = df['sepal_length'].values
         x.dtype # dtype means type to use in computing the SD. for array of integers,the defualt is  float64.

Out[97]: dtype('float64')
```

Sample Mean:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} = x_i$$

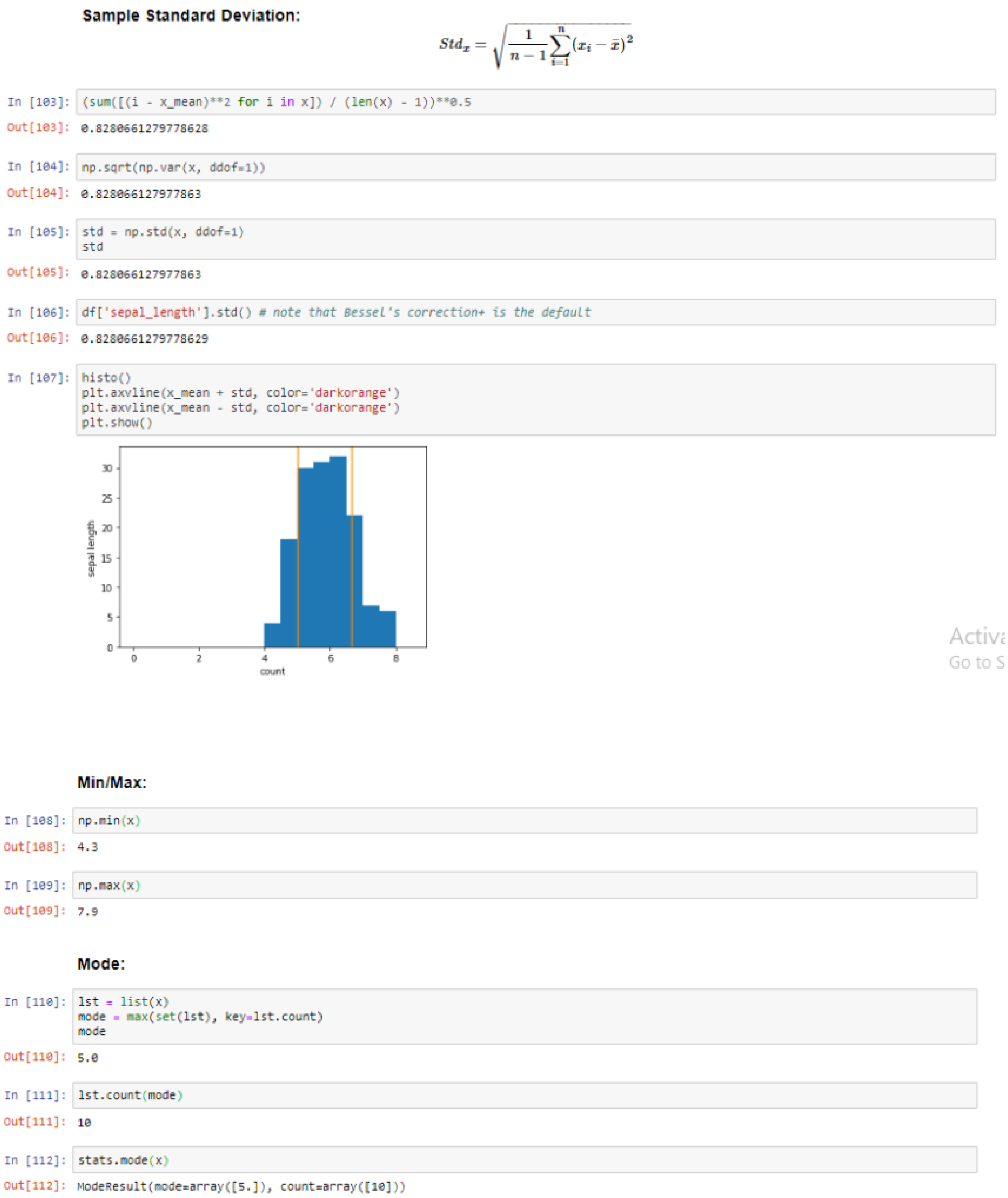
```
In [98]: sum(i for i in x) / len(x)

Out[98]: 5.843333333333335

In [70]: x_mean = np.mean(x)
         x_mean

Out[70]: 5.843333333333334

In [99]: histo()
         plt.axvline(x_mean, color='darkorange')
         plt.show()
```



Sample Variance:

$$Var_x = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

```
In [100]: sum([(i - x_mean)**2 for i in x]) / (len(x) - 1)

Out[100]: 0.6856935123042504

In [101]: var = np.var(x, ddof=1) #ddof means delta degree of freedom. by default ddof =0
          var

Out[101]: 0.6856935123042507

 In [74]: df['sepal_length'].var()

 Out[74]: 0.6856935123042505

In [102]: histo()
          plt.axvline(x_mean + var, color='darkorange')
          plt.axvline(x_mean - var, color='darkorange')
          plt.show()
```



Sample Standard Deviation:

$$Std_x = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

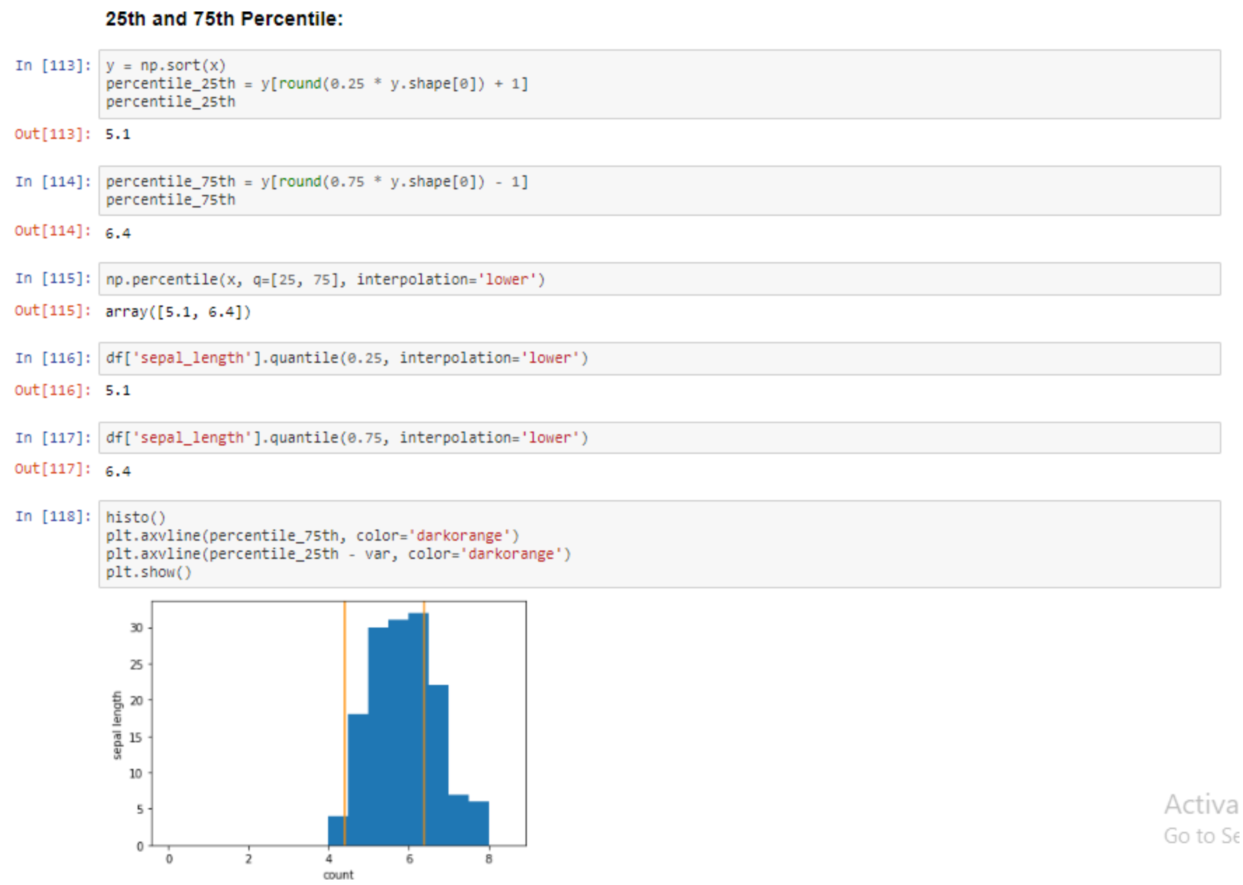
```
In [103]: (sum([(i - x_mean)**2 for i in x]) / (len(x) - 1))**0.5
Out[103]: 0.8280661279778628

In [104]: np.sqrt(np.var(x, ddof=1))
Out[104]: 0.828066127977863

In [105]: std = np.std(x, ddof=1)
          std
Out[105]: 0.828066127977863

In [106]: df['sepal_length'].std() # note that Bessel's correction+ is the default
Out[106]: 0.8280661279778629

In [107]: histo()
          plt.axvline(x_mean + std, color='darkorange')
          plt.axvline(x_mean - std, color='darkorange')
          plt.show()
```

Min/Max:

```
In [108]: np.min(x)
Out[108]: 4.3

In [109]: np.max(x)
Out[109]: 7.9
```

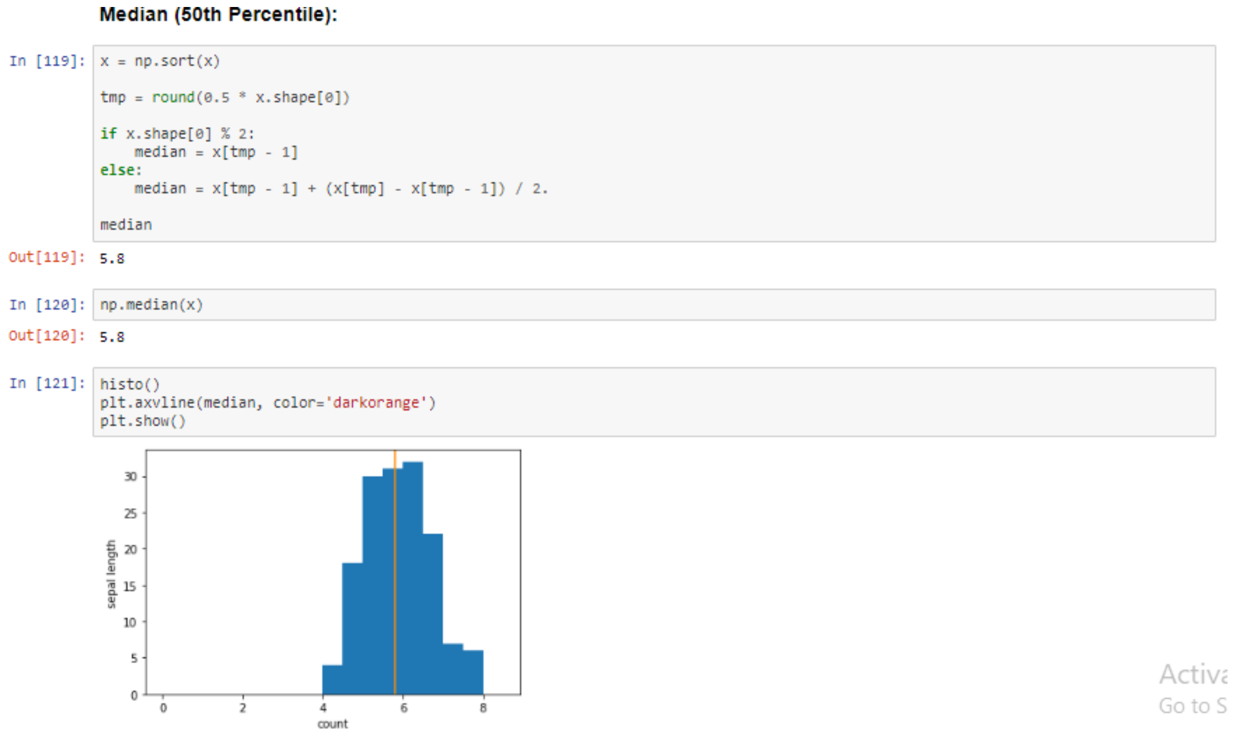
Mode:

```
In [110]: lst = list(x)
          mode = max(set(lst), key=lst.count)
          mode
Out[110]: 5.0

In [111]: lst.count(mode)
Out[111]: 10

In [112]: stats.mode(x)
Out[112]: ModeResult(mode=array([5.]), count=array([10]))
```



25th and 75th Percentile:

```
In [113]: y = np.sort(x)
          percentile_25th = y[round(0.25 * y.shape[0]) + 1]
          percentile_25th
Out[113]: 5.1

In [114]: percentile_75th = y[round(0.75 * y.shape[0]) - 1]
          percentile_75th
Out[114]: 6.4

In [115]: np.percentile(x, q=[25, 75], interpolation='lower')
Out[115]: array([5.1, 6.4])

In [116]: df['sepal_length'].quantile(0.25, interpolation='lower')
Out[116]: 5.1

In [117]: df['sepal_length'].quantile(0.75, interpolation='lower')
Out[117]: 6.4

In [118]: histo()
          plt.axvline(percentile_75th, color='darkorange')
          plt.axvline(percentile_25th - var, color='darkorange')
          plt.show()
```



Median (50th Percentile):

```
In [119]: x = np.sort(x)

          tmp = round(0.5 * x.shape[0])

          if x.shape[0] % 2:
              median = x[tmp - 1]
          else:
              median = x[tmp - 1] + (x[tmp] - x[tmp - 1]) / 2.

          median
Out[119]: 5.8

In [120]: np.median(x)
Out[120]: 5.8

In [121]: histo()
          plt.axvline(median, color='darkorange')
          plt.show()
```