Name : Adepu Shivani

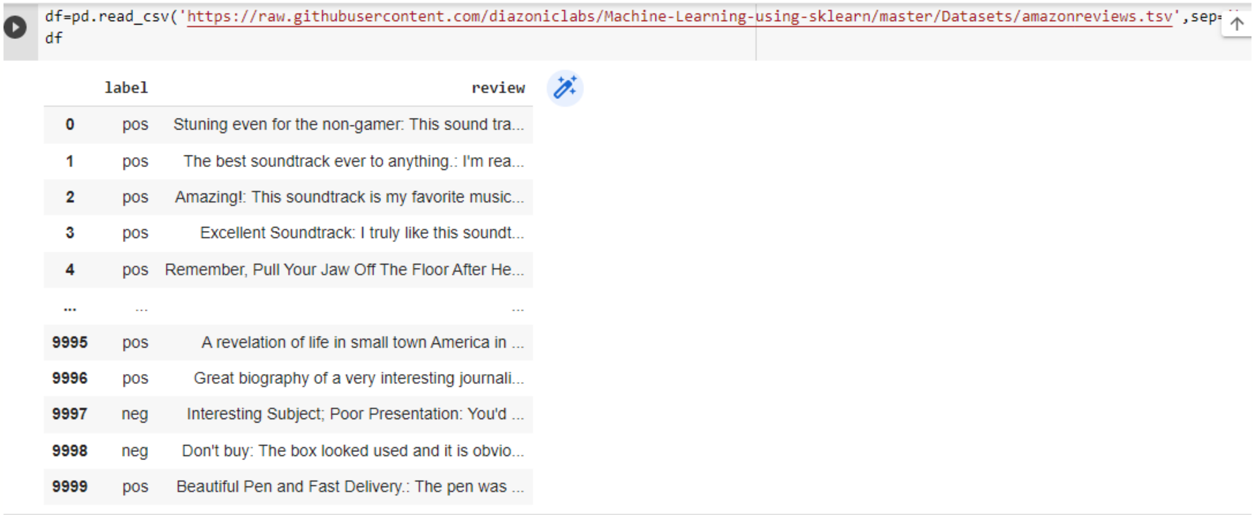
College : JNTUH UNIVERSITY COLLEGE OF ENGINEERING JAGITYAL (JNTUH UCEJ)

Branch : Information Technology

Year : 4th year

Contact : 6300554876

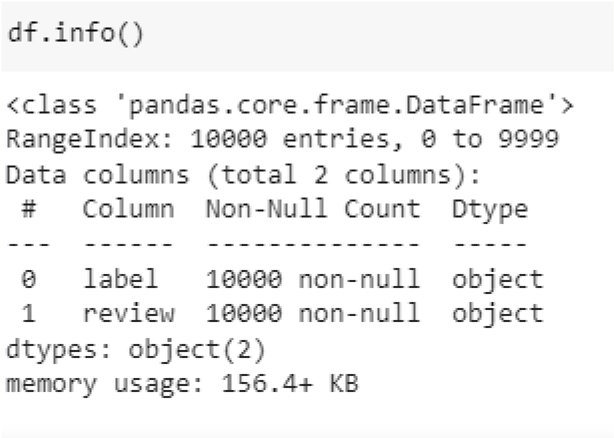
```
#Major Project 1 -->CLASSIFIER/REGRESSION
#Major Project 1 URL → https://colab.research.google.com/drive/1DeVI5PEcCEwHsE1LXtbmcePxOoo65s2D#scrollTo=kTrG6kqAOBLU
#Support vector Classifier (SVC model)
#Dataset - neg pos dataset
#Dataset URL-https://raw.githubusercontent.com/diazoniclabs/Machine-Learning-using-sklearn/master/Datasets/amazonreviews.tsv

import pandas as pd
df=pd.read_csv('https://raw.githubusercontent.com/diazoniclabs/Machine-Learning-using-sklearn/master/Datasets/amazonreviews.tsv',sep='\t')
df
```



```
df=pd.read_csv('https://raw.githubusercontent.com/diazoniclabs/Machine-Learning-using-sklearn/master/Datasets/amazonreviews.tsv',sep=
df
```

	label	review
0	pos	Stuning even for the non-gamer: This sound tra...
1	pos	The best soundtrack ever to anything.: I'm rea...
2	pos	Amazing!: This soundtrack is my favorite music...
3	pos	Excellent Soundtrack: I truly like this soundt...
4	pos	Remember, Pull Your Jaw Off The Floor After He...
...	...	...
9995	pos	A revelation of life in small town America in ...
9996	pos	Great biography of a very interesting journali...
9997	neg	Interesting Subject; Poor Presentation: You'd ...
9998	neg	Don't buy: The box looked used and it is obvio...
9999	pos	Beautiful Pen and Fast Delivery.: The pen was ...

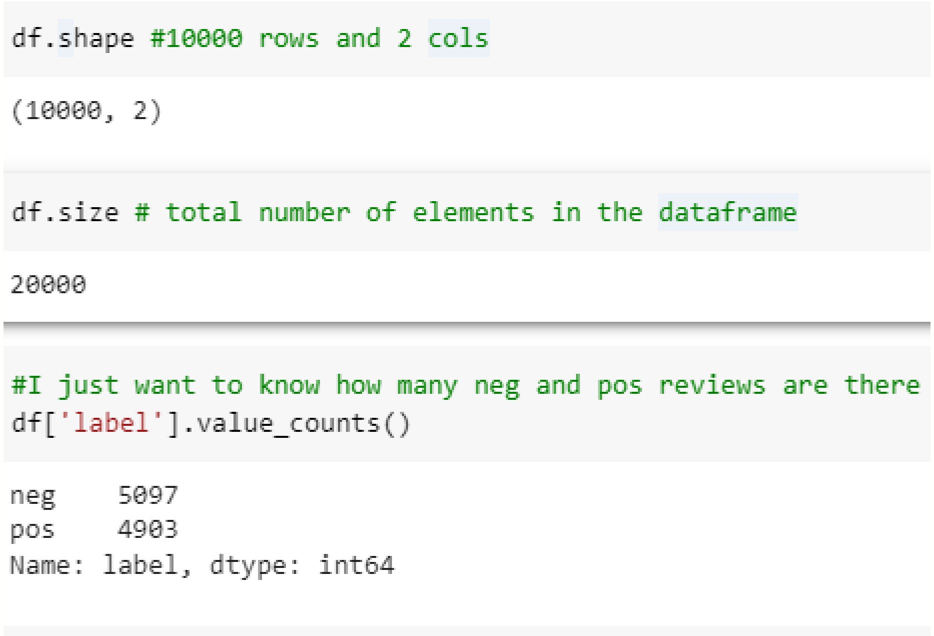
```
df.info()
```



```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   label   10000 non-null  object
 1   review  10000 non-null  object
dtypes: object(2)
memory usage: 156.4+ KB
```

```
df.shape #10000 rows and 2 cols
df.size # total number of elements in the dataframe
#I just want to know how many neg and pos reviews
are there
df['label'].value_counts()
```



```
df.shape #10000 rows and 2 cols
```

```
(10000, 2)
```

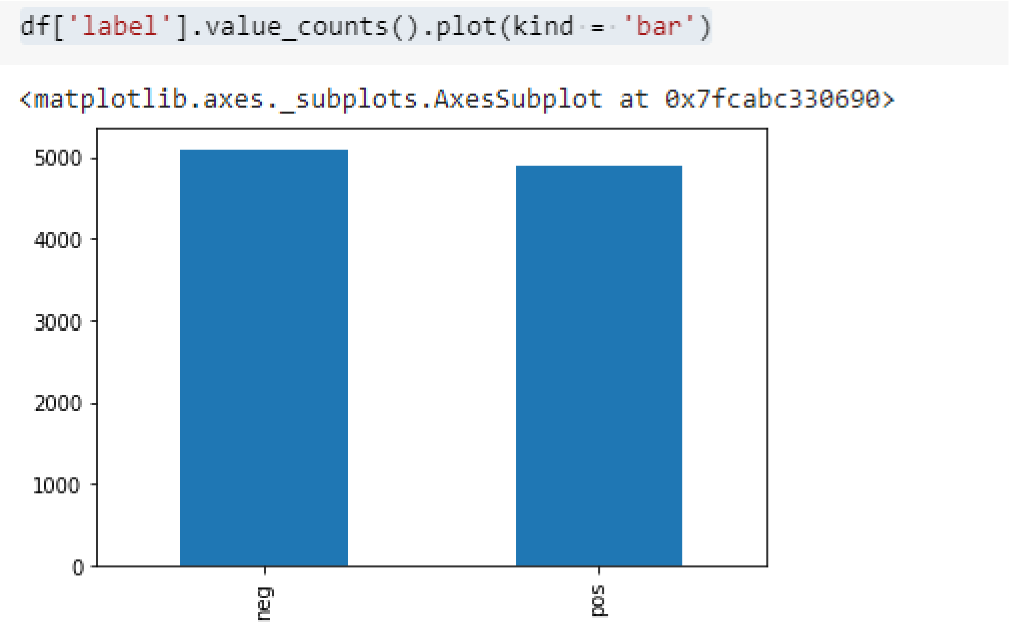
```
df.size # total number of elements in the dataframe
```

```
20000
```

```
#I just want to know how many neg and pos reviews are there
df['label'].value_counts()
```

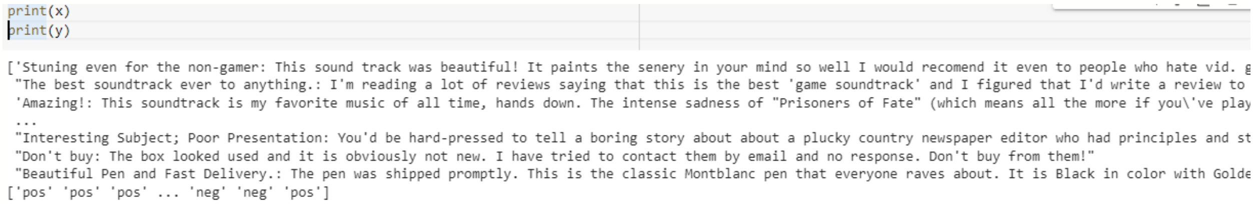
```
neg    5097
pos    4903
Name: label, dtype: int64
```

```
df['label'].value_counts().plot(kind = 'bar')
```



```
df['label'].value_counts().plot(kind = 'bar')
<matplotlib.axes._subplots.AxesSubplot at 0x7fcabc330690>
```

```
#4.divide the data into input and output
x = df.iloc[:,1].values#only when text reviews are involved ,i/p is 1 dimensional
y = df.iloc[:,0].values
print(x)
print(y)
```

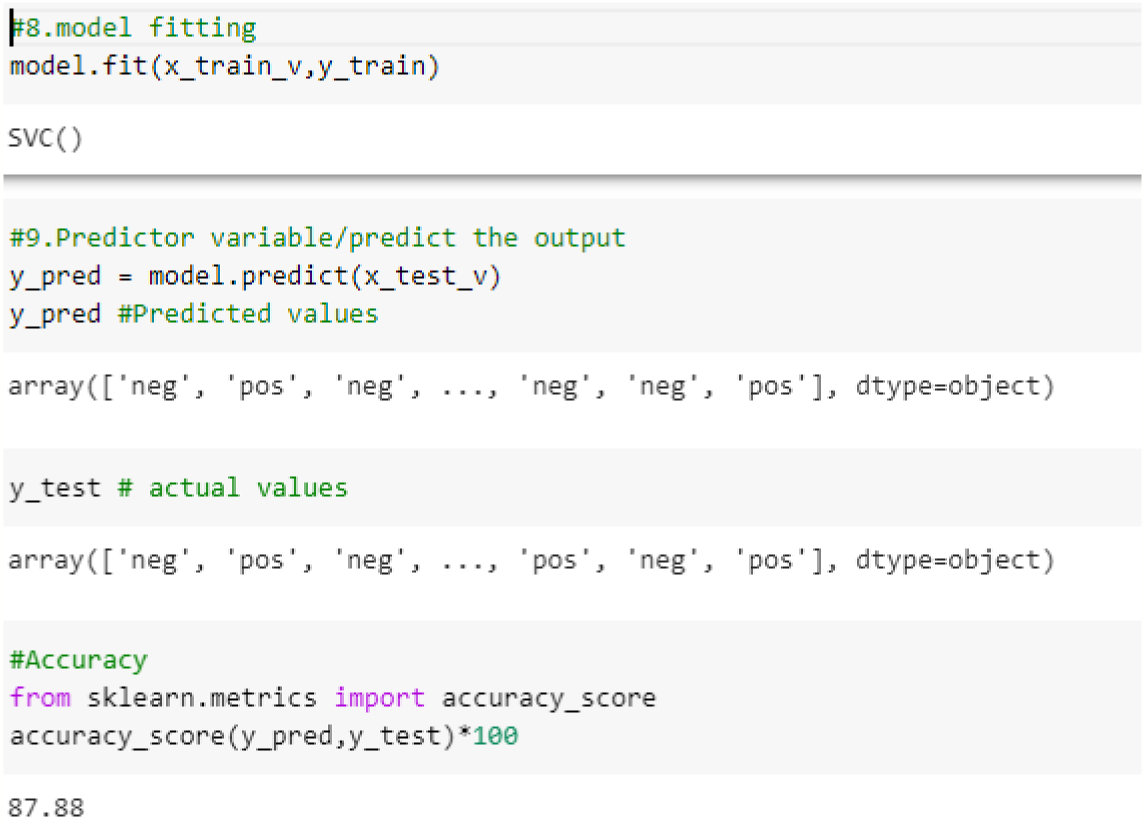


```
print(x)
print(y)
```

```
['Stuning even for the non-gamer: This sound track was beautiful! It paints the senery in your mind so well I would recomend it even to people who hate vid. g
 "The best soundtrack ever to anything.: I'm reading a lot of reviews saying that this is the best 'game soundtrack' and I figured that I'd write a review to
 'Amazing!: This soundtrack is my favorite music of all time, hands down. The intense sadness of "Prisoners of Fate" (which means all the more if you\'ve play
 ...
 "Interesting Subject; Poor Presentation: You'd be hard-pressed to tell a boring story about about a plucky country newspaper editor who had principles and st
 "Don't buy: The box looked used and it is obviously not new. I have tried to contact them by email and no response. Don't buy from them!"
 "Beautiful Pen and Fast Delivery.: The pen was shipped promptly. This is the classic Montblanc pen that everyone raves about. It is Black in color with Golde
['pos' 'pos' 'pos' ... 'neg' 'neg' 'pos']
```

```
#5.train_test_split
```

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,random_state = 0)

#6.Apply TF-IDF vectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
vect = TfidfVectorizer()
x_train_v = vect.fit_transform(x_train)
x_test_v = vect.transform(x_test)

#7.Apply CLASSIFIER/REGRESSOR/CLUSTERER
from sklearn.svm import SVC
model = SVC()

#8.model fitting
model.fit(x_train_v,y_train)

#9.Predictor variable/predict the output
y_pred = model.predict(x_test_v)
y_pred #Predicted values
y_test # actual values

#Accuracy
from sklearn.metrics import accuracy_score
accuracy_score(y_pred,y_test)*100
```



```
#8.model fitting
model.fit(x_train_v,y_train)
```

```
SVC()
```

```
#9.Predictor variable/predict the output
y_pred = model.predict(x_test_v)
y_pred #Predicted values
```

```
array(['neg', 'pos', 'neg', ..., 'neg', 'neg', 'pos'], dtype=object)
```

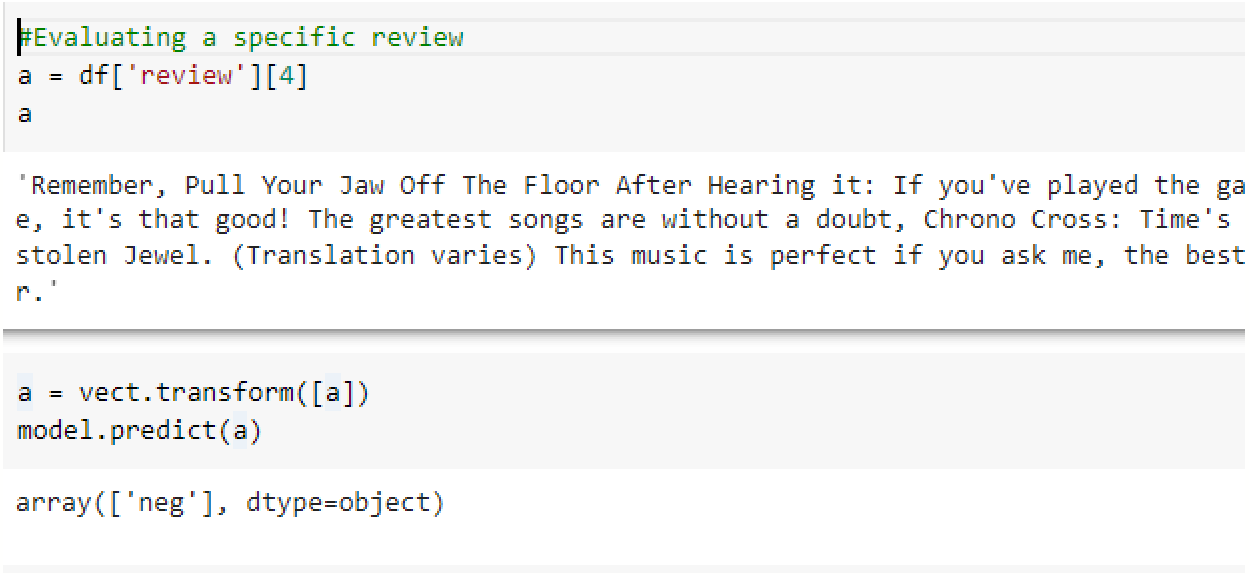
```
y_test # actual values
```

```
array(['neg', 'pos', 'neg', ..., 'pos', 'neg', 'pos'], dtype=object)
```

```
#Accuracy
from sklearn.metrics import accuracy_score
accuracy_score(y_pred,y_test)*100
```

```
87.88
```

```
#Evaluating a specific review

a = df['review'][4]

a

a = vect.transform([a])

model.predict(a)
```



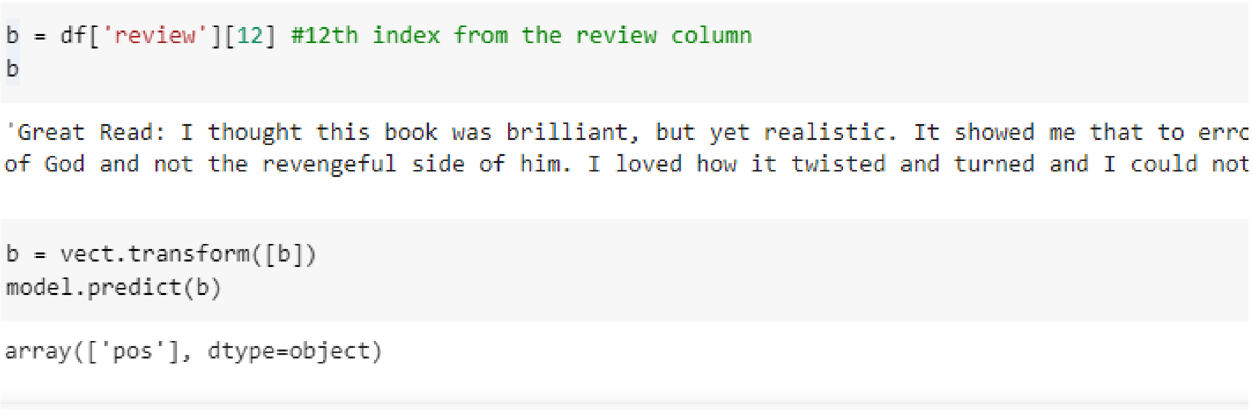
```
#Evaluating a specific review
a = df['review'][4]
a
```

```
'Remember, Pull Your Jaw Off The Floor After Hearing it: If you've played the ga
e, it's that good! The greatest songs are without a doubt, Chrono Cross: Time's
stolen Jewel. (Translation varies) This music is perfect if you ask me, the best
r.'
```

```
a = vect.transform([a])
model.predict(a)
```

```
array(['neg'], dtype=object)
```

```
b = df['review'][12] #12th index from the review column
b
b = vect.transform([b])
model.predict(b)
```



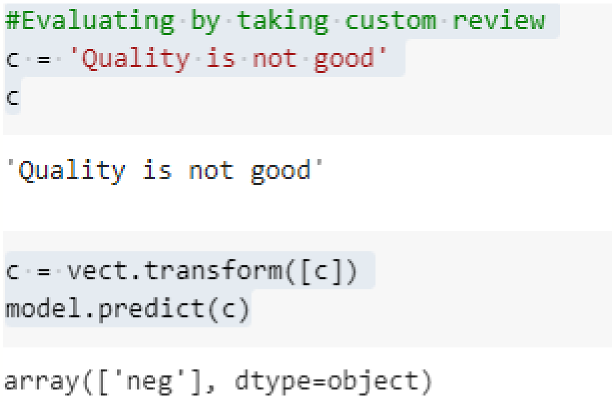
```
b = df['review'][12] #12th index from the review column
b
```

```
'Great Read: I thought this book was brilliant, but yet realistic. It showed me that to errc
of God and not the revengeful side of him. I loved how it twisted and turned and I could not
```

```
b = vect.transform([b])
model.predict(b)
```

```
array(['pos'], dtype=object)
```

```
#Evaluating by taking custom review
c = 'Quality is not good'
c
c = vect.transform([c])
model.predict(c)
```



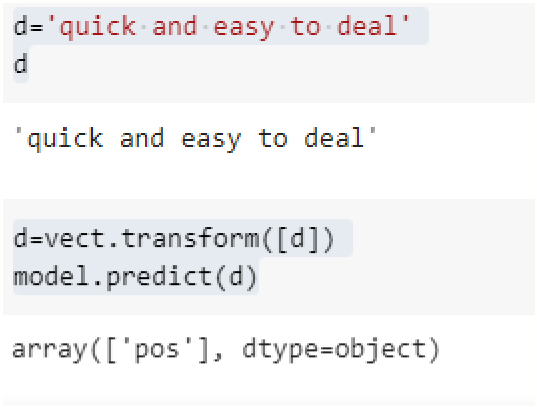
```
#Evaluating by taking custom review
c = 'Quality is not good'
c
```

```
'Quality is not good'
```

```
c = vect.transform([c])
model.predict(c)
```

```
array(['neg'], dtype=object)
```

```
d='quick and easy to deal'
d
d=vect.transform([d])
model.predict(d)
```



```
d='quick and easy to deal'
d
```

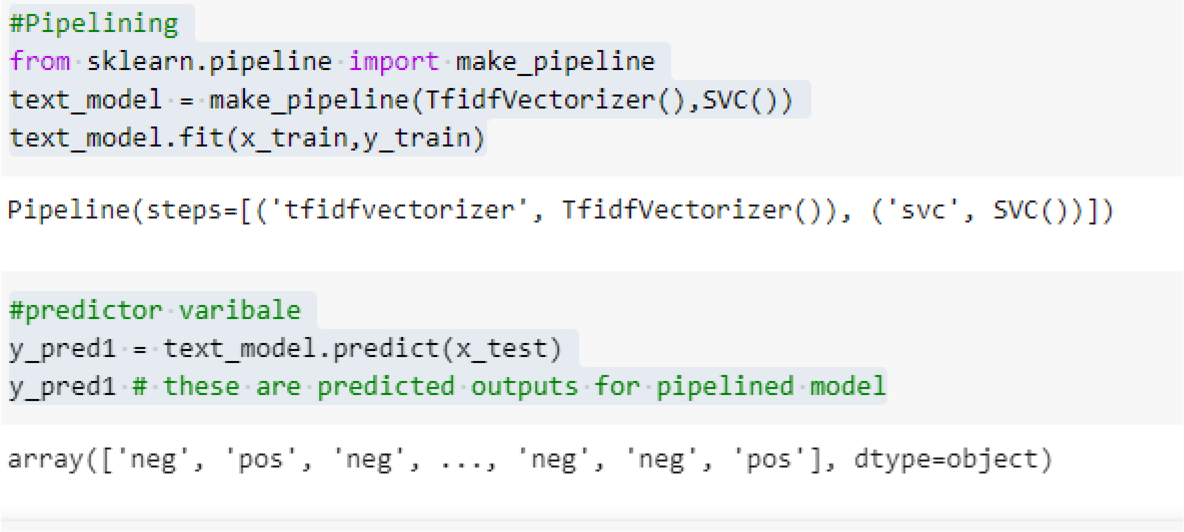
```
'quick and easy to deal'
```

```
d=vect.transform([d])
model.predict(d)
```

```
array(['pos'], dtype=object)
```

```
#1.Gather data and divide into i/p and o/p
#2.Applied train_test_split
#3.Applied TfidfVectorizer
#4.Apply SVC
#5.Predicted the output
#If ever I have to deploy my model,I will have to perform p
ipelining
#Pipelining - Combining of 2 or more modules
#So here we want to combine/pipeline TdidfVectorizer and  S
VC

#Pipelining
from sklearn.pipeline import make_pipeline
text_model = make_pipeline(TfidfVectorizer(),SVC())
```

```
text_model.fit(x_train,y_train)
#predictor varibale
y_pred1 = text_model.predict(x_test)
y_pred1 # these are predicted outputs for pipelined model
```



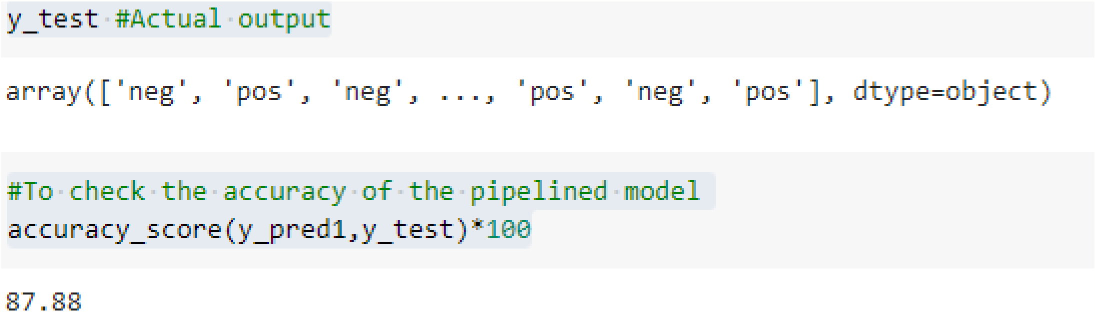
```
#Pipelining
from sklearn.pipeline import make_pipeline
text_model = make_pipeline(TfidfVectorizer(),SVC())
text_model.fit(x_train,y_train)
```

```
Pipeline(steps=[('tfidfvectorizer', TfidfVectorizer()), ('svc', SVC())])
```

```
#predictor varibale
y_pred1 = text_model.predict(x_test)
y_pred1 # these are predicted outputs for pipelined model
```

```
array(['neg', 'pos', 'neg', ..., 'neg', 'neg', 'pos'], dtype=object)
```

```
y_test #Actual output
#To check the accuracy of the pipelined model
accuracy_score(y_pred1,y_test)*100
```



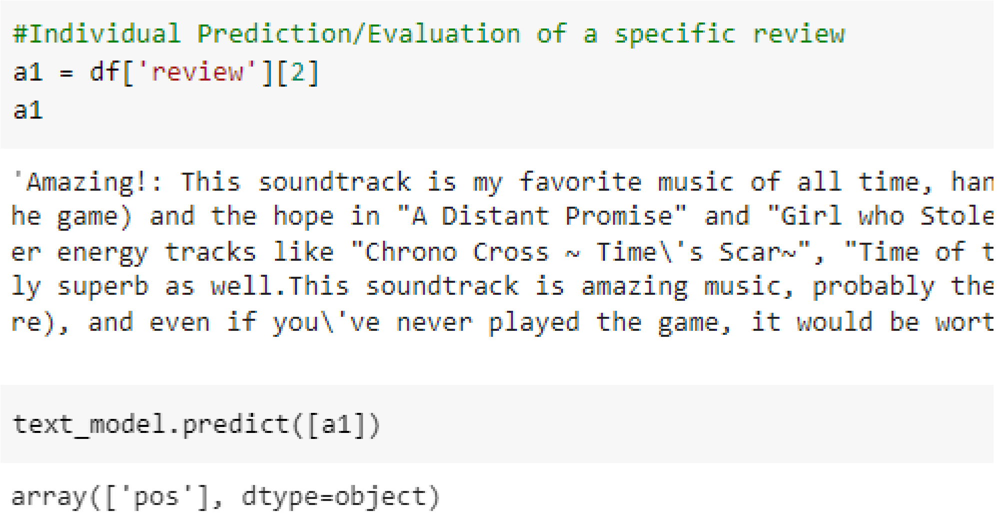
```
y_test #Actual output
```

```
array(['neg', 'pos', 'neg', ..., 'pos', 'neg', 'pos'], dtype=object)
```

```
#To check the accuracy of the pipelined model
accuracy_score(y_pred1,y_test)*100
```

```
87.88
```

```
#Individual Prediction/Evaluation of a specific review
a1 = df['review'][2]
a1
text_model.predict([a1])
```



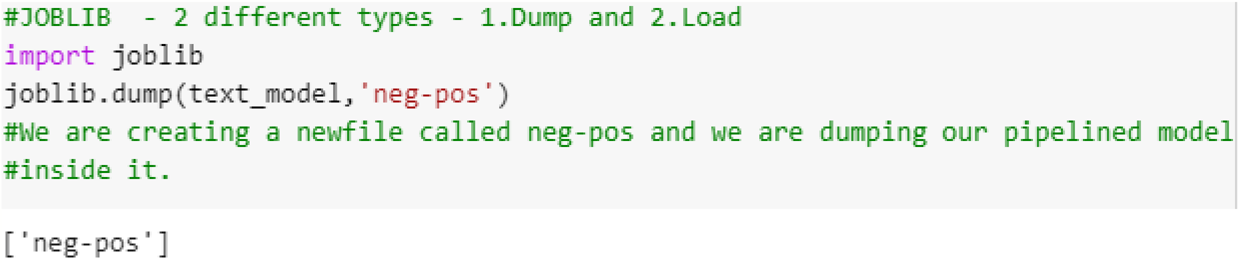
```
#Individual Prediction/Evaluation of a specific review
a1 = df['review'][2]
a1
```

```
'Amazing!: This soundtrack is my favorite music of all time, han
he game) and the hope in "A Distant Promise" and "Girl who Stole
er energy tracks like "Chrono Cross ~ Time\'s Scar~", "Time of t
ly superb as well.This soundtrack is amazing music, probably the
re), and even if you\'ve never played the game, it would be wort
```

```
text_model.predict([a1])
```

```
array(['pos'], dtype=object)
```

```
#JOBLIB  - 2 different types - 1.Dump and 2.Load
import joblib
joblib.dump(text_model,'neg-pos')
#We are creating a newfile called neg-
pos and we are dumping our pipelined model
#inside it.
```

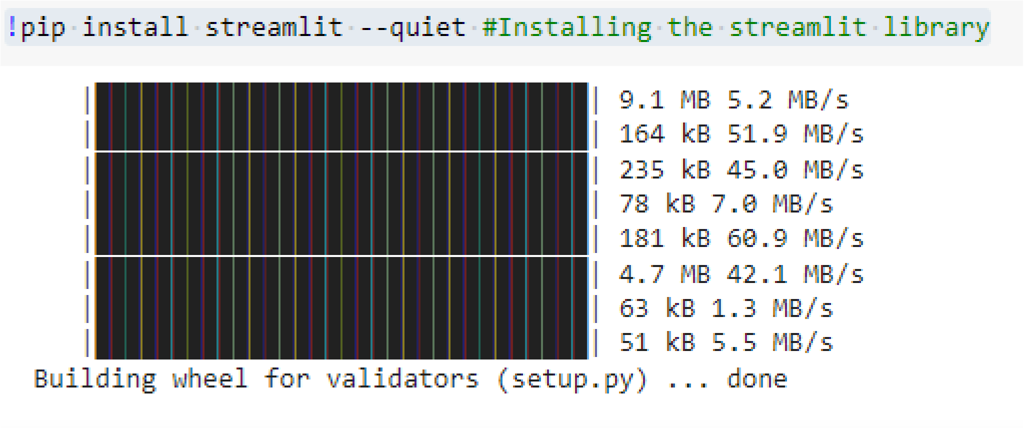


```
#JOBLIB  - 2 different types - 1.Dump and 2.Load
import joblib
joblib.dump(text_model,'neg-pos')
#We are creating a newfile called neg-pos and we are dumping our pipelined model
#inside it.
```

```
['neg-pos']
```

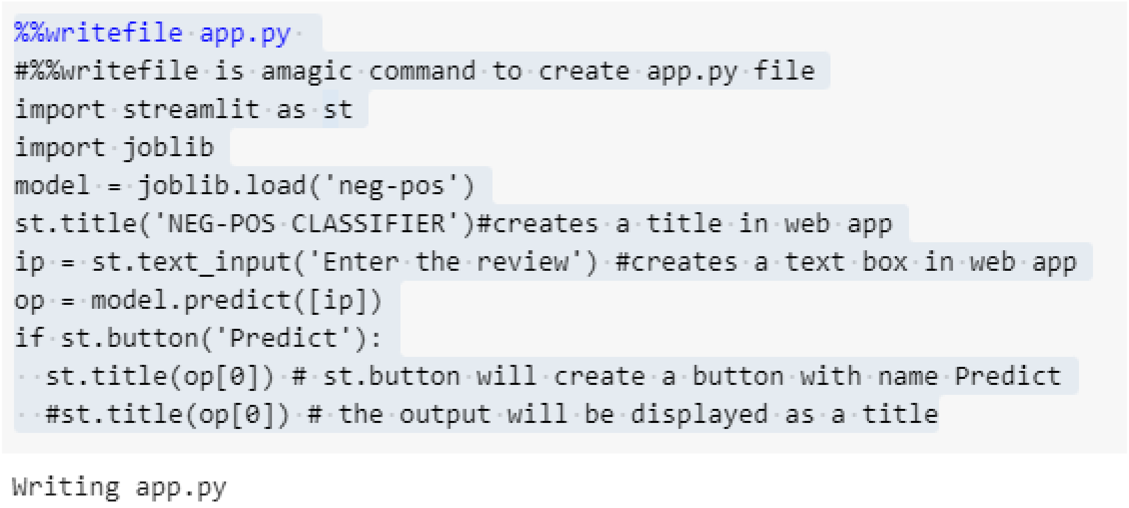
```
#We are creating a STREAMLIT WEB APPLICATION
#Deployment are of 2 types
#1.Temporary deployment
#2.Permanent deployment
```

```
#Temporary deployment - Local host
!pip install streamlit --quiet
#Installing the streamlit library
```



```
!pip install streamlit --quiet #Installing the streamlit library

     |████████████████████████████████| 9.1 MB 5.2 MB/s
     |████████████████████████████████| 164 kB 51.9 MB/s
     |████████████████████████████████| 235 kB 45.0 MB/s
     |████████████████████████████████| 78 kB 7.0 MB/s
     |████████████████████████████████| 181 kB 60.9 MB/s
     |████████████████████████████████| 4.7 MB 42.1 MB/s
     |████████████████████████████████| 63 kB 1.3 MB/s
     |████████████████████████████████| 51 kB 5.5 MB/s
  Building wheel for validators (setup.py) ... done
```

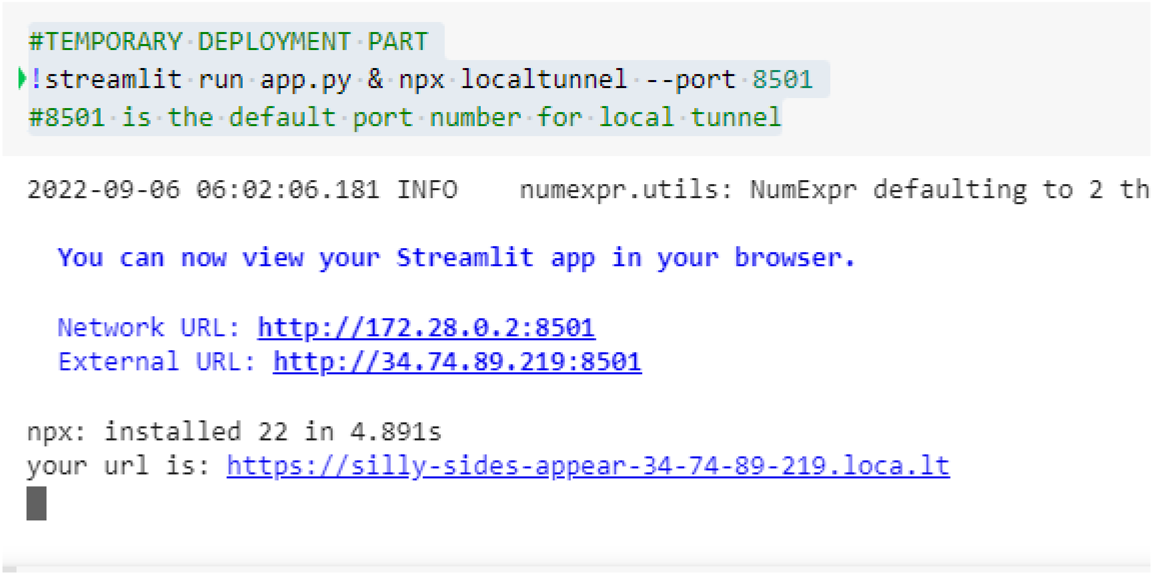
```
%%writefile app.py
#%%writefile is amagic command to create app.py file
import streamlit as st
import joblib
model = joblib.load('neg-pos')
st.title('NEG-
POS CLASSIFIER')#creates a title in web app
ip = st.text_input('Enter the review') #creates a text
box in web app
op = model.predict([ip])
if st.button('Predict'):
  st.title(op[0]) # st.button will create a button with
 name Predict
  #st.title(op[0]) # the output will be displayed as a
title
```



```
%%writefile app.py
#%%writefile is amagic command to create app.py file
import streamlit as st
import joblib
model = joblib.load('neg-pos')
st.title('NEG-POS CLASSIFIER')#creates a title in web app
ip = st.text_input('Enter the review') #creates a text box in web app
op = model.predict([ip])
if st.button('Predict'):
  st.title(op[0]) # st.button will create a button with name Predict
  #st.title(op[0]) # the output will be displayed as a title
```

```
Writing app.py
```

```
#TEMPORARY DEPLOYMENT PART
!streamlit run app.py & npx localtunnel --port 8501
#8501 is the default port number for local tunnel
```

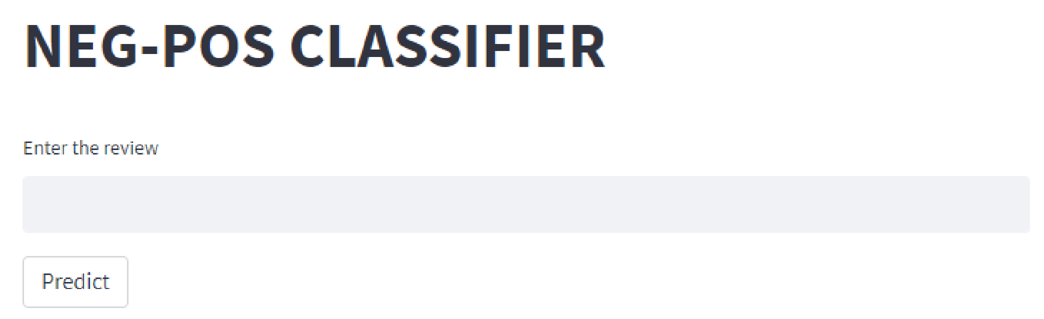


```
#TEMPORARY DEPLOYMENT PART
!streamlit run app.py & npx localtunnel --port 8501
#8501 is the default port number for local tunnel
```

```
2022-09-06 06:02:06.181 INFO     numexpr.utils: NumExpr defaulting to 2 th

  You can now view your Streamlit app in your browser.

  Network URL: http://172.28.0.2:8501
  External URL: http://34.74.89.219:8501

npx: installed 22 in 4.891s
your url is: https://silly-sides-appear-34-74-89-219.loca.lt
```

```
#Temporary Deployment url :
```

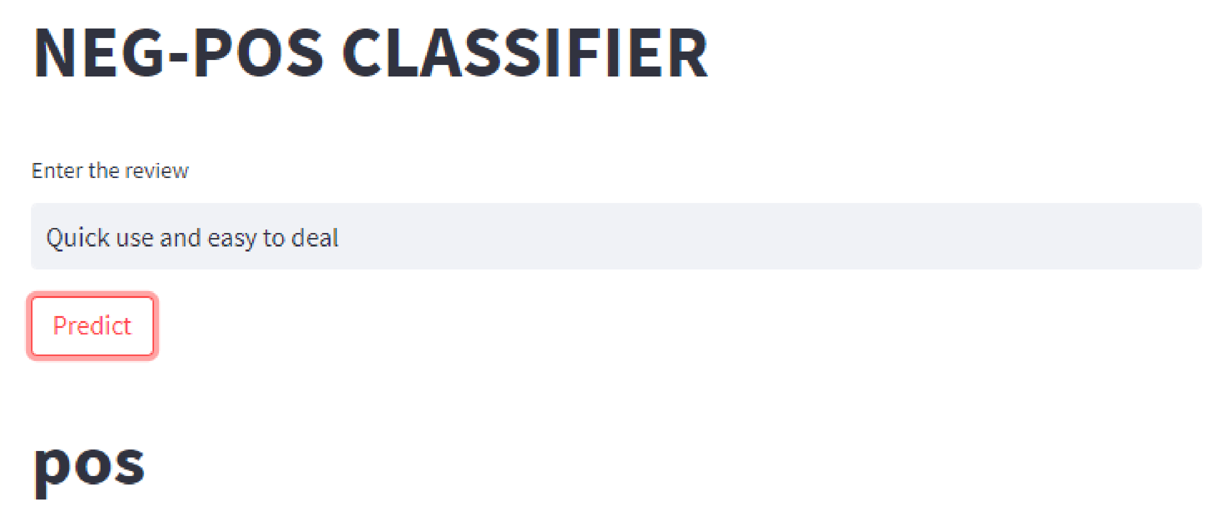
https://silly-sides-appear-34-74-89-219.loca.lt/



NEG-POS CLASSIFIER

Enter the review

```
#Enter the review as: Quick use and easy to deal
#Click on Predict button and check whether the review
is neg or pos classifier
```



NEG-POS CLASSIFIER

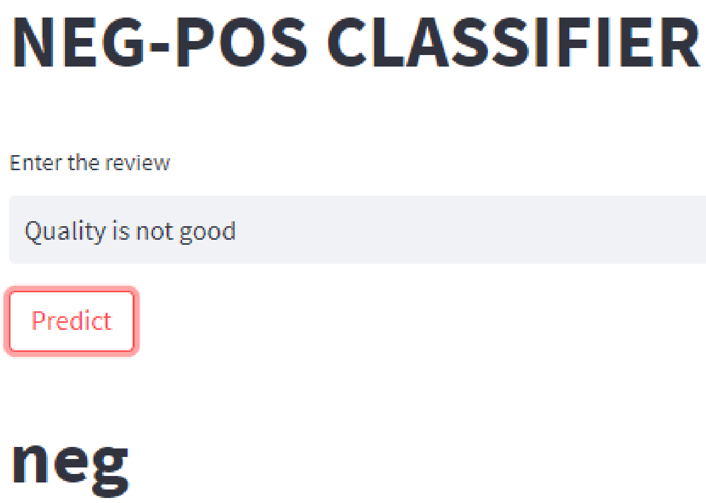
Enter the review

Quick use and easy to deal

pos

```
#Enter the another custom review: Quality is not good
#Click on Predict button and check whether the review
is pos or neg classifier
```

#



NEG-POS CLASSIFIER

Enter the review

Quality is not good

neg

#Steps for creating Heroku:

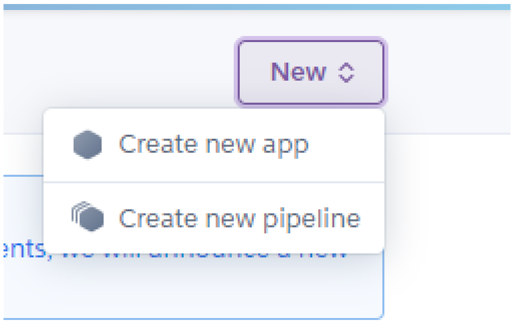
#Till now we made a temporary deployment now we are making a permannet deployment by creating app.

#First create or signin into your github account and create a new Repositories, assign a name to new repositories i.e, Rinex-majorprojects.

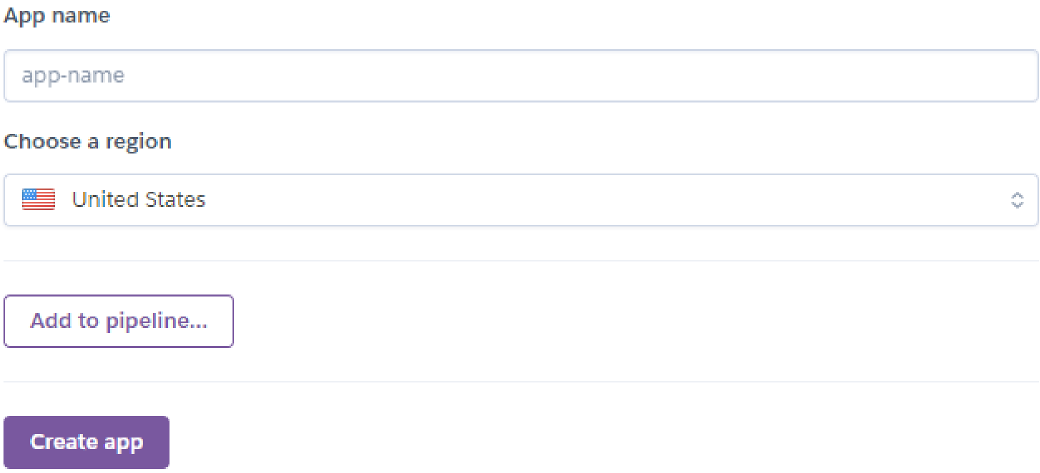
#Now open the created Repositories and create new files in the Rinexmajorprojects repositories by clicking on create new file,i.e. the files are Procfile,app.py, requirements.txt,setup.sh and upload the required file, Here we are uploaded neg-pos file in the respiratory.

#Make sure that the repositories is in public.

```
#Now create or login into Heroku account
#click on NEW-->CREATE NEW APP
```



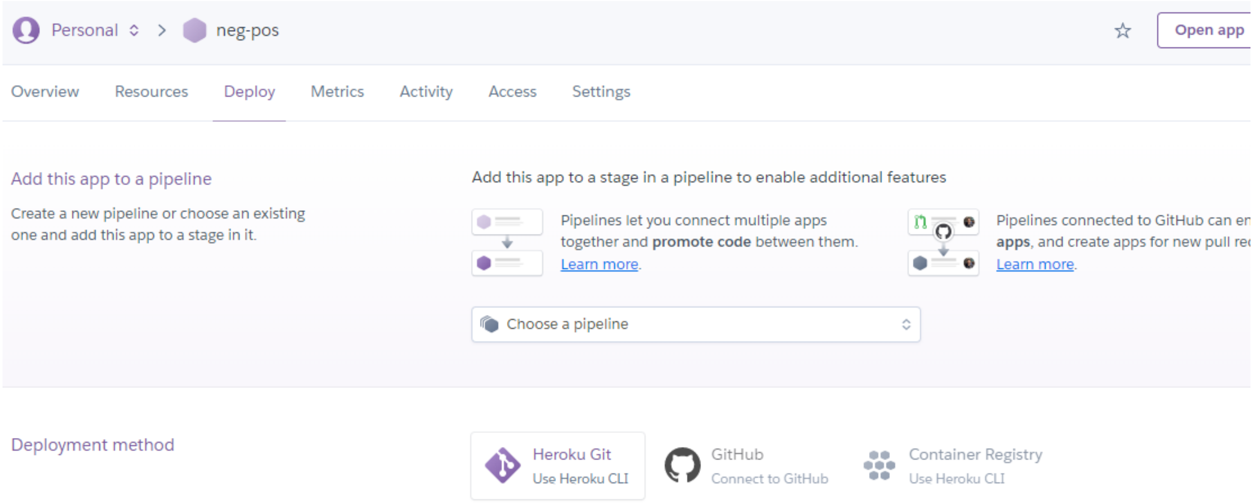
```
#Enter the app name and choose a region and clicl on cr
eate app button
```



App name: app-name

Choose a region: United States

#The screen will be displayed as below,



Add this app to a pipeline

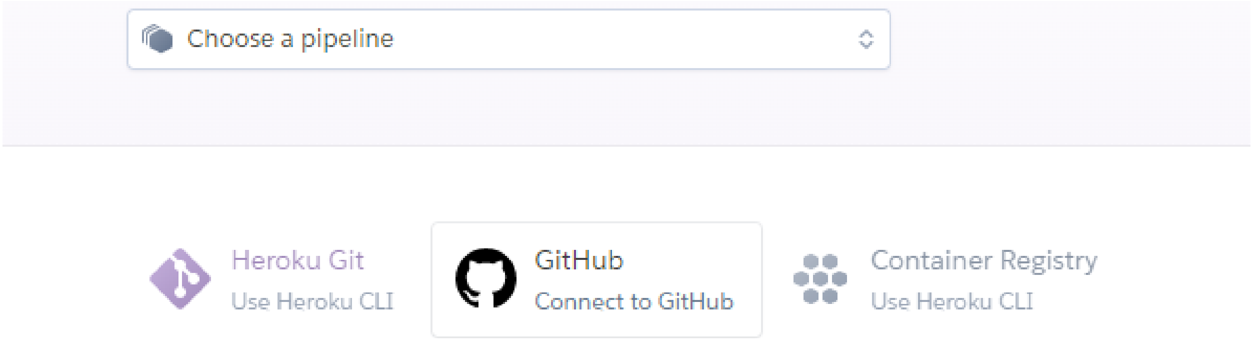
Create a new pipeline or choose an existing one and add this app to a stage in it.

Add this app to a stage in a pipeline to enable additional features

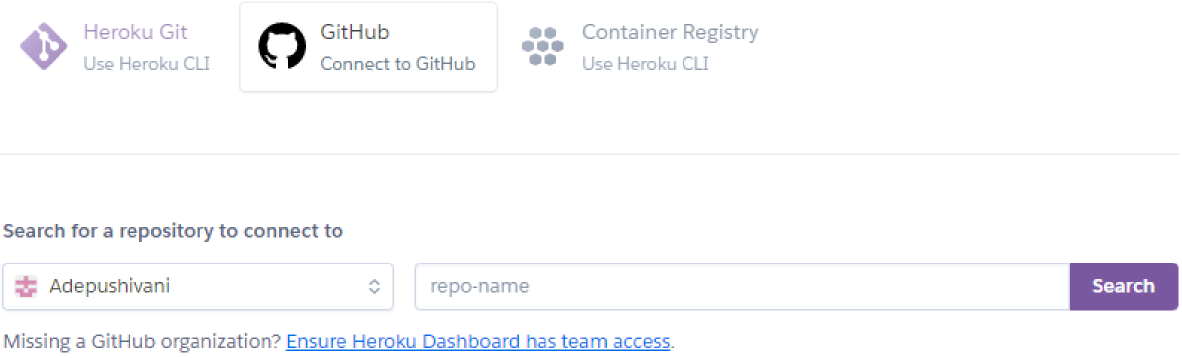
Pipelines let you connect multiple apps together and **promote code** between them. Learn more.

Pipelines connected to GitHub can en **apps**, and create apps for new pull re Learn more.

#Now click on Github to connect your github account to app



#Now search the Repositories which was created in your github account.



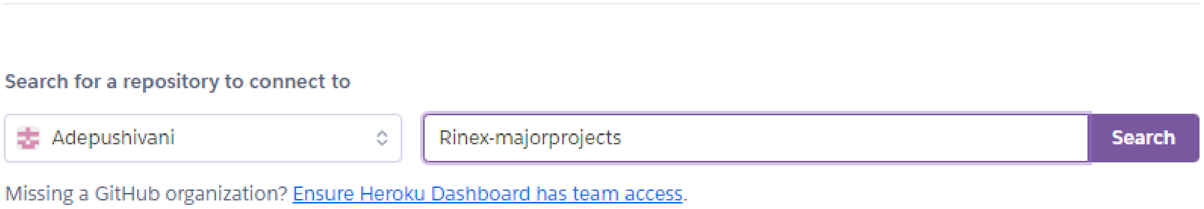
Search for a repository to connect to

Adepushivani

repo-name

Missing a GitHub organization? Ensure Heroku Dashboard has team access.

#I have created Rinex-majiorprojects repositorie in my github account, and searching the same repositorie in the search engine and click on search , you will find your repositorie and click on connect.



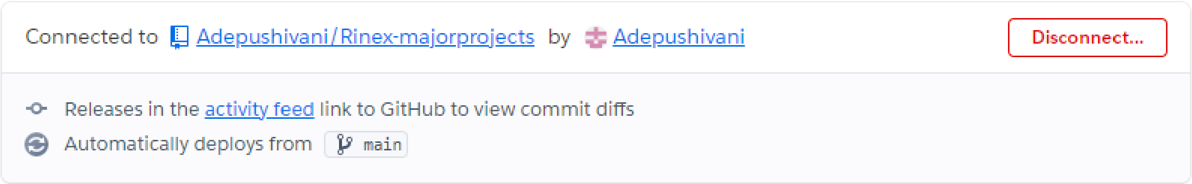
Search for a repository to connect to

Adepushivani

Rinex-majorprojects

Missing a GitHub organization? Ensure Heroku Dashboard has team access.

#Now you are connected to your appropriate Rinex-majorprojects, if you want to disconnect to your repositorie you can click on Disconnect button



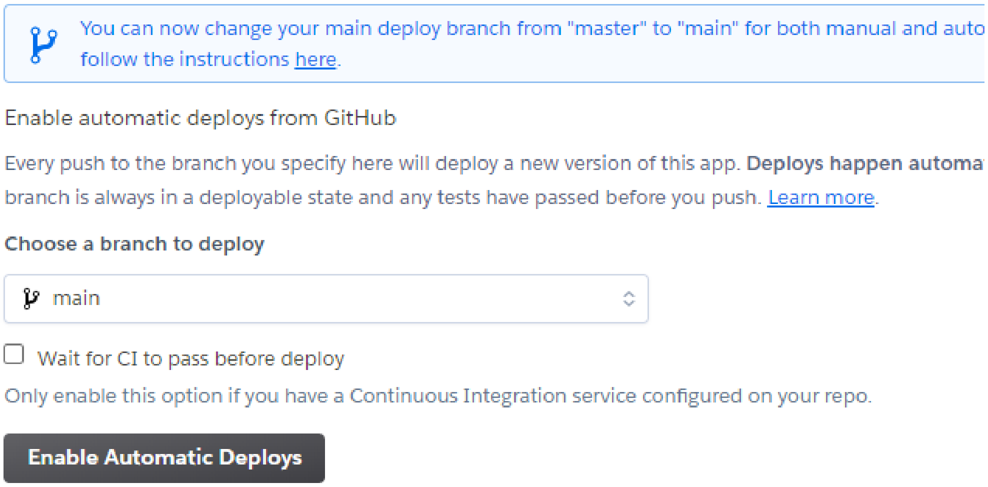
Connected to Adepushivani/Rinex-majorprojects by Adepushivani

Disconnect...

Releases in the activity feed link to GitHub to view commit diffs

Automatically deploys from main

#Now click on Enable Automatic Deploys



You can now change your main deploy branch from "master" to "main" for both manual and auto
follow the instructions here.

Enable automatic deploys from GitHub

Every push to the branch you specify here will deploy a new version of this app. **Deploys happen automa**
branch is always in a deployable state and any tests have passed before you push. Learn more.

Choose a branch to deploy

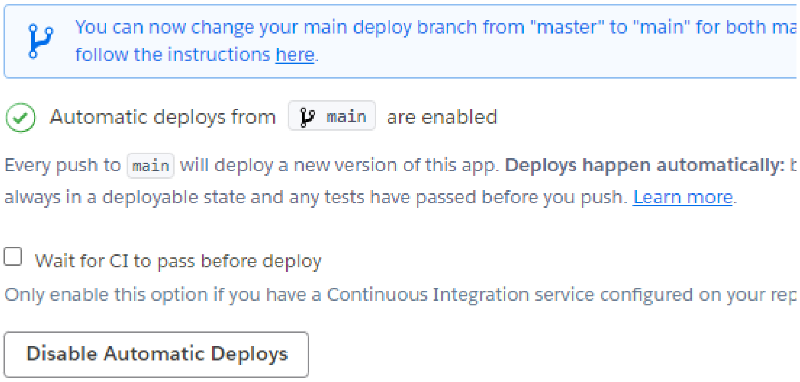
main

□ Wait for CI to pass before deploy

Only enable this option if you have a Continuous Integration service configured on your repo.

Enable Automatic Deploys

#You can also click on Disable Automatic Deploys button
 if you want to disable the deployments.



You can now change your main deploy branch from "master" to "main" for both ma
follow the instructions here.

✓ Automatic deploys from main are enabled

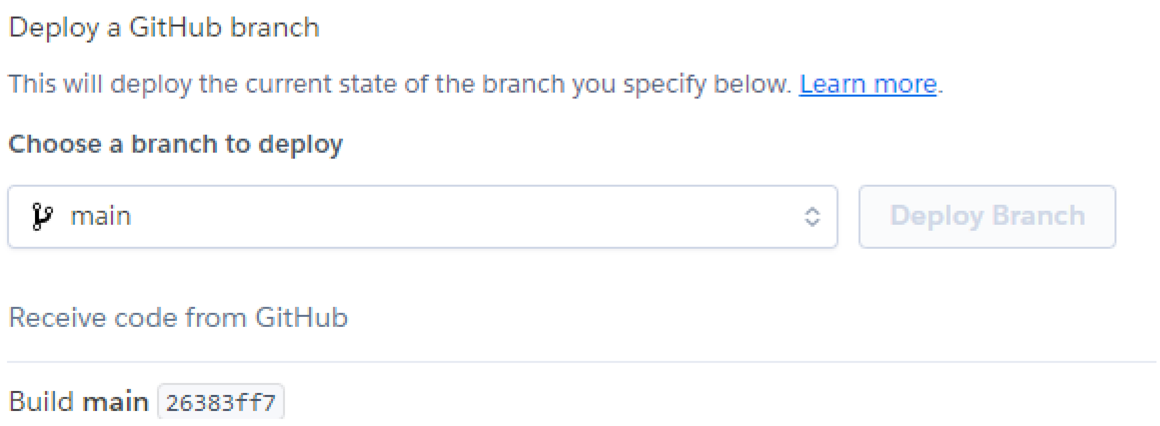
Every push to `main` will deploy a new version of this app. **Deploys happen automatically:** t
always in a deployable state and any tests have passed before you push. Learn more.

□ Wait for CI to pass before deploy

Only enable this option if you have a Continuous Integration service configured on your rep

Disable Automatic Deploys

#Now click on Deploy button for permanent deployment of app



Deploy a GitHub branch

This will deploy the current state of the branch you specify below. Learn more.

Choose a branch to deploy

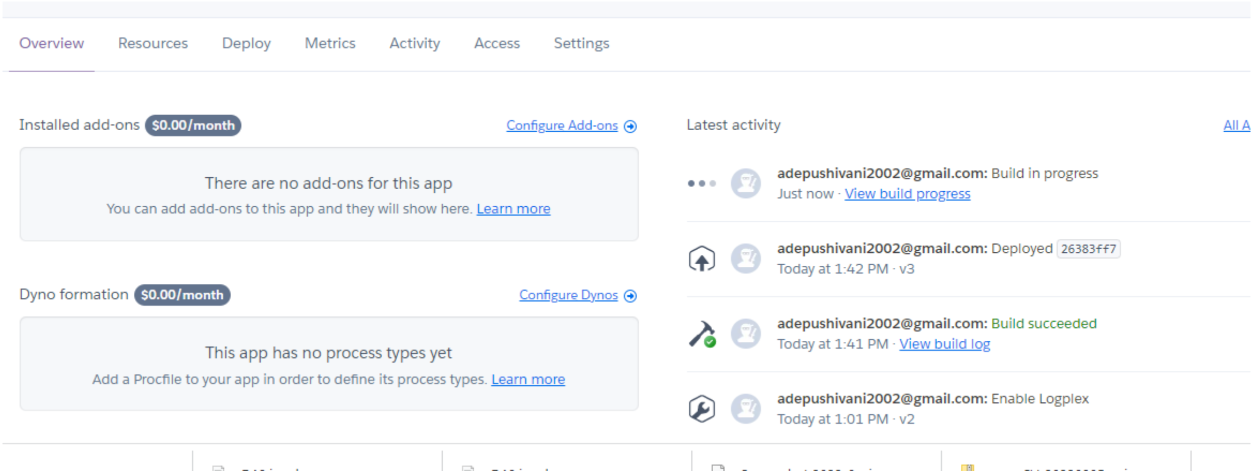
main

Receive code from GitHub

Build main 26383ff7

#After pressing on deploy button it shows you're your app was deployed successfully.

#Now click on overview at top left of screen, the scre en will be displayed as shown below.



Installed add-ons $0.00/month

There are no add-ons for this app
You can add add-ons to this app and they will show here. Learn more

Dyno formation $0.00/month

This app has no process types yet
Add a Procfile to your app in order to define its process types. Learn more

Latest activity

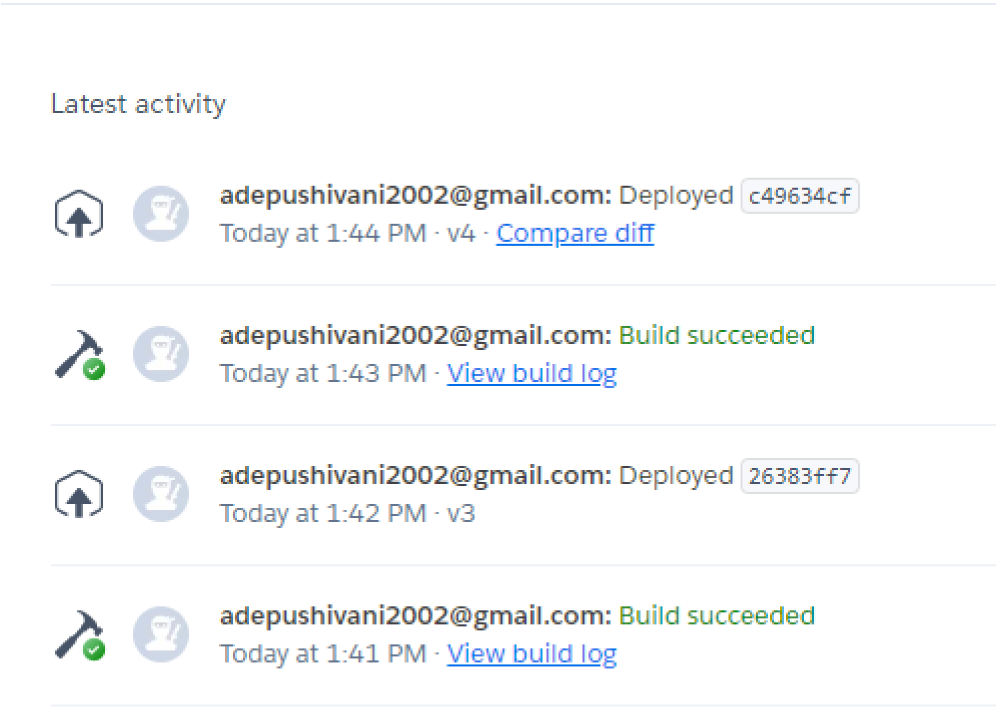
adepushivani2002@gmail.com: Build in progress
Just now · View build progress

adepushivani2002@gmail.com: Deployed 26383ff7
Today at 1:42 PM · v3

adepushivani2002@gmail.com: Build succeeded
Today at 1:41 PM · View build log

adepushivani2002@gmail.com: Enable Logplex
Today at 1:01 PM · v2

#Click on view build log/progress



Latest activity

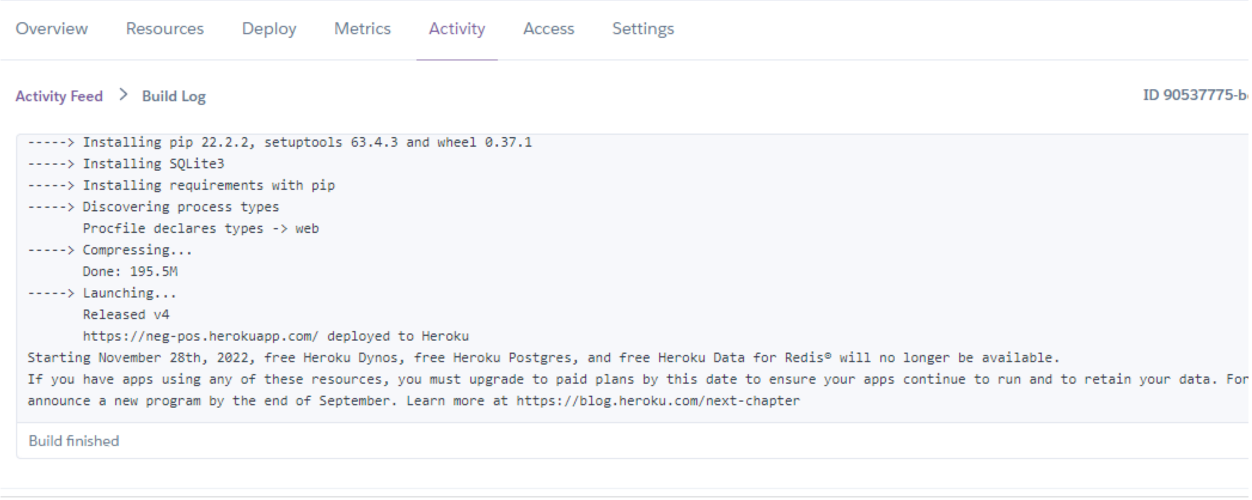
adepushivani2002@gmail.com: Deployed c49634cf
Today at 1:44 PM · v4 · Compare diff

adepushivani2002@gmail.com: Build succeeded
Today at 1:43 PM · View build log

adepushivani2002@gmail.com: Deployed 26383ff7
Today at 1:42 PM · v3

adepushivani2002@gmail.com: Build succeeded
Today at 1:41 PM · View build log

#The build log will be displayed,



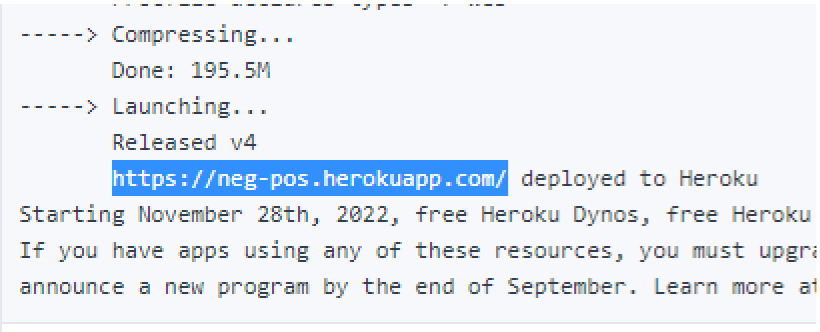
Activity Feed > Build Log

ID 90537775-b

```
-----> Installing pip 22.2.2, setuptools 63.4.3 and wheel 0.37.1
-----> Installing SQLite3
-----> Installing requirements with pip
-----> Discovering process types
       Procfile declares types -> web
-----> Compressing...
       Done: 195.5M
-----> Launching...
       Released v4
       https://neg-pos.herokuapp.com/ deployed to Heroku
Starting November 28th, 2022, free Heroku Dynos, free Heroku Postgres, and free Heroku Data for Redis® will no longer be available.
If you have apps using any of these resources, you must upgrade to paid plans by this date to ensure your apps continue to run and to retain your data. For
announce a new program by the end of September. Learn more at https://blog.heroku.com/next-chapter
```

Build finished

#Now scroll down and copy
the url which is displayed in build log

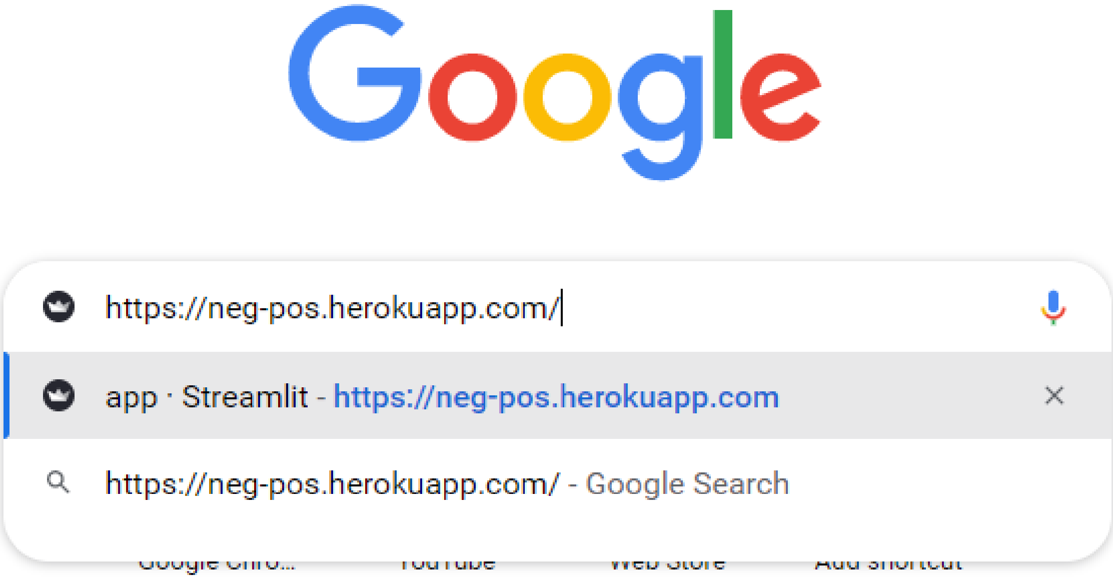


```
-----> Compressing...
       Done: 195.5M
-----> Launching...
       Released v4
       https://neg-pos.herokuapp.com/ deployed to Heroku
Starting November 28th, 2022, free Heroku Dynos, free Heroku
If you have apps using any of these resources, you must upgra
announce a new program by the end of September. Learn more at
```

#This is the permanent deployment url:

https://neg-pos.herokuapp.com/

#Copy the url and open the new tab and paste the
url in search engine

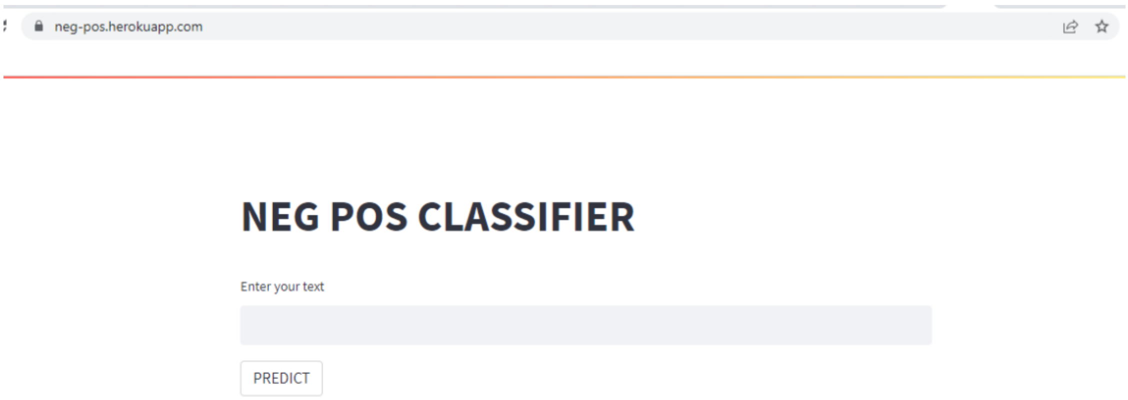


https://neg-pos.herokuapp.com/

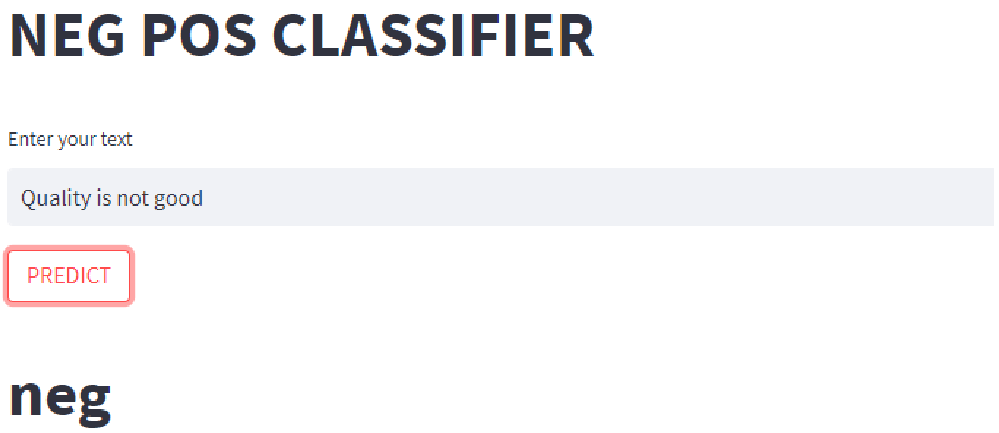
app · Streamlit - https://neg-pos.herokuapp.com

https://neg-pos.herokuapp.com/ - Google Search

```
# click on enter, the app will be displayed as shown
below.
```



```
#Enter the text and click on predict button, the output
 will be predicted.
```

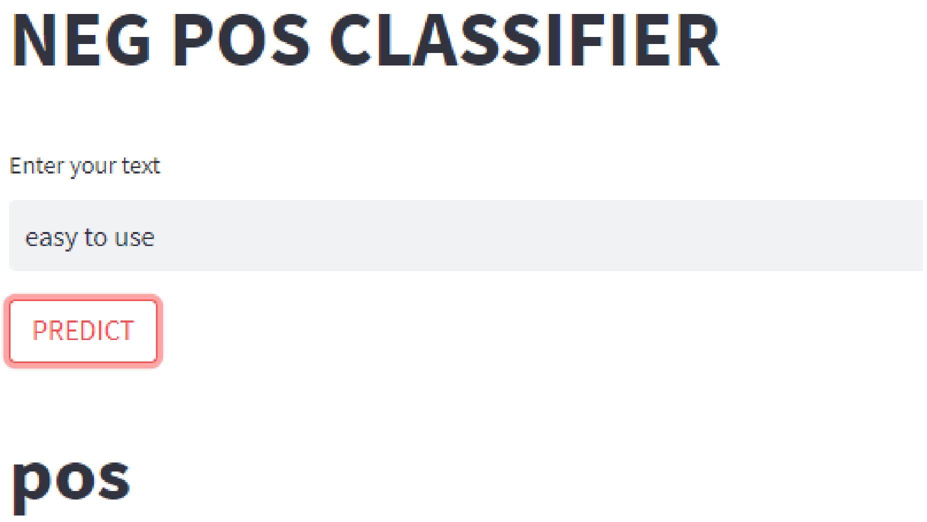


Quality is not good

neg

```
#Enter the another text and precict the output.
```

```
#This is the permanent deployment of Web app using
Heroku.
```



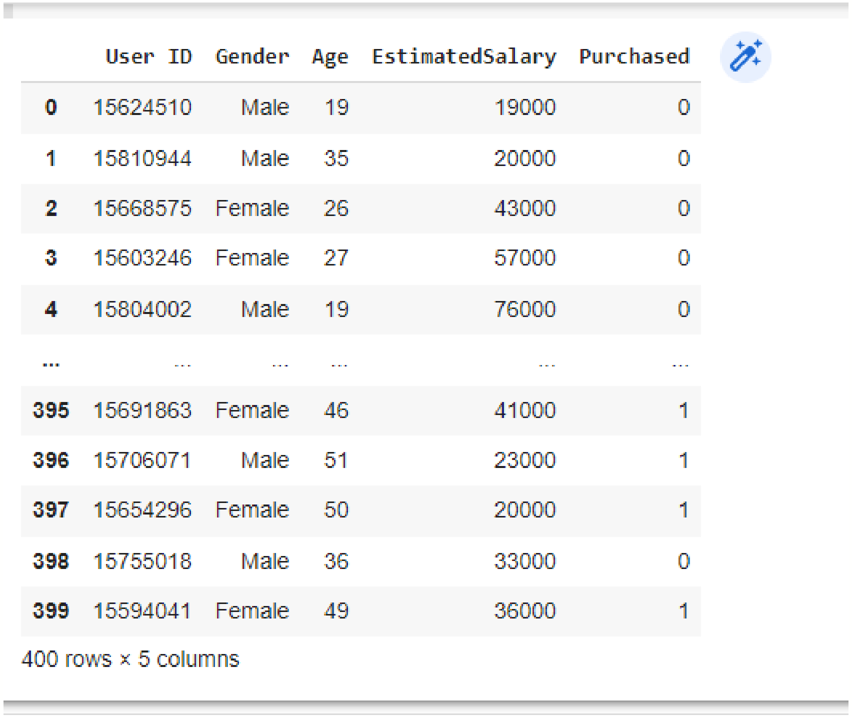
NEG POS CLASSIFIER

Enter your text

easy to use

pos

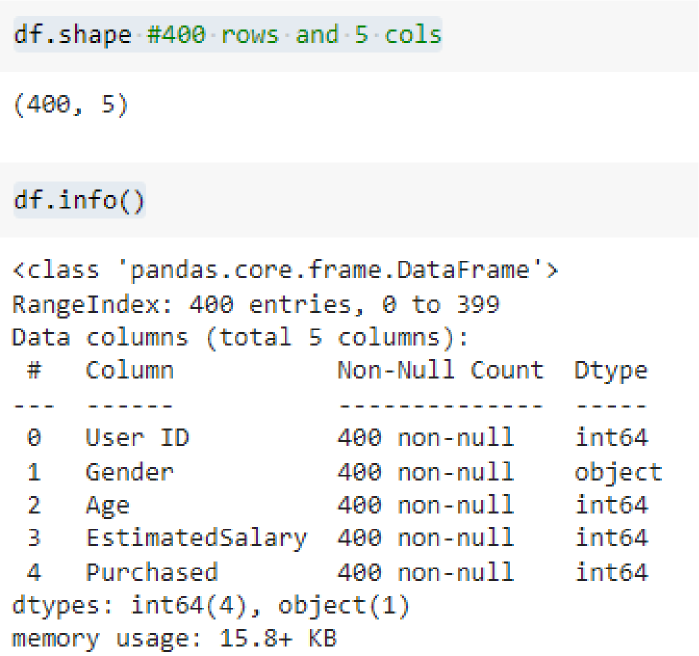
```
#Major Project 2-->K Means Clustering
#Major Project 2 URL→
https://colab.research.google.com/drive/1GDzBDdeSK5osSsowe8BmajZOQLbZ-udv#scrollTo=8ME07NCELirl
#UNSUPERVISED LEARNING -CLUSTERING - K MEANS CLUSTERING
#IN CLUSTERING- THERE IS NO y(OUTPUT),we only consider i/p to train our model.
#DatasetURL:https://raw.githubusercontent.com/ameenmanna8824/DATASETS/main/Social Network Ads.csv
#1.take data and create dataframe
import pandas as pd
df = pd.read_csv('https://raw.githubusercontent.com/ameenmanna8824/DATASETS/main/Social_Network_Ads.csv')
df
```



	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0
...	...	...	...	...	...
395	15691863	Female	46	41000	1
396	15706071	Male	51	23000	1
397	15654296	Female	50	20000	1
398	15755018	Male	36	33000	0
399	15594041	Female	49	36000	1

400 rows × 5 columns

```
df.shape #400 rows and 5 cols
df.info()
```



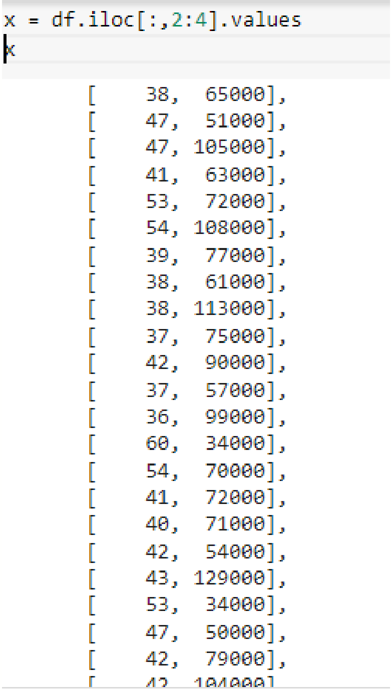
```
df.shape #400 rows and 5 cols
```

```
(400, 5)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 5 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   User ID          400 non-null    int64 
 1   Gender           400 non-null    object
 2   Age              400 non-null    int64 
 3   EstimatedSalary  400 non-null    int64 
 4   Purchased        400 non-null    int64 
dtypes: int64(4), object(1)
memory usage: 15.8+ KB
```

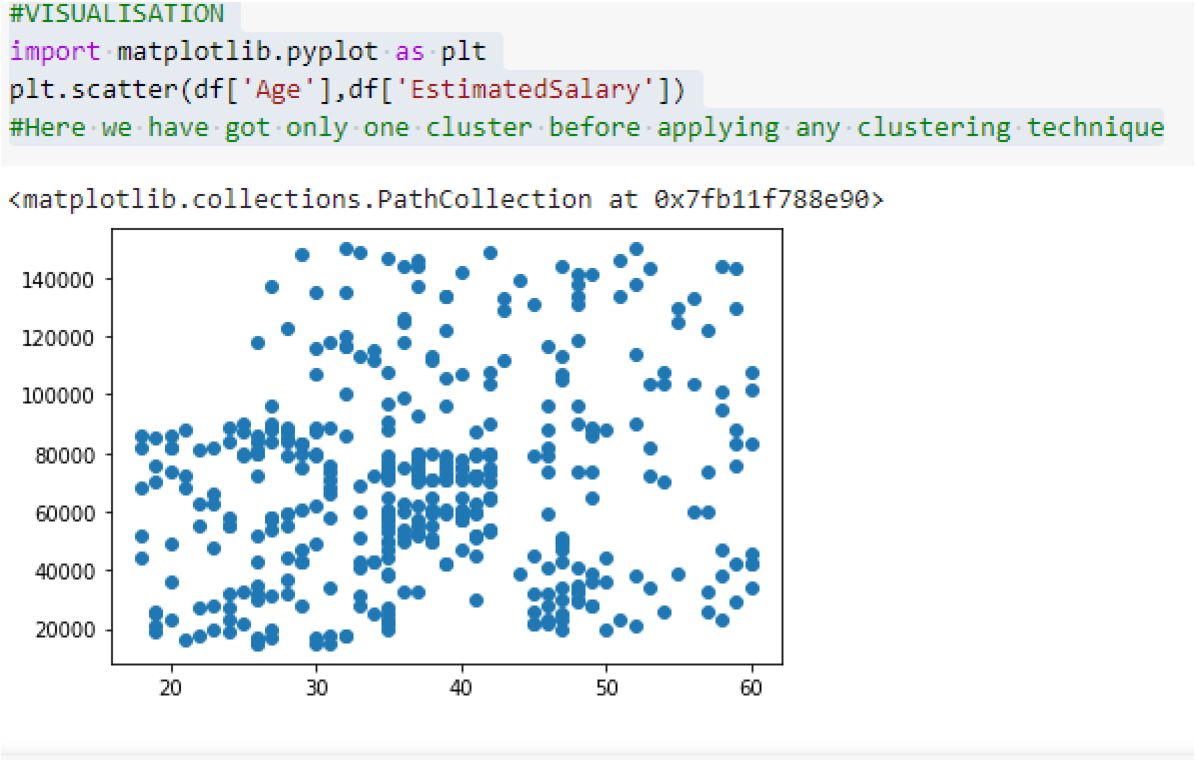
```
#4.divide the into i/p
x = df.iloc[:,2:4].values
x
```



```
x = df.iloc[:,2:4].values
x
```

```
       [    38,  65000],
       [    47,  51000],
       [    47, 105000],
       [    41,  63000],
       [    53,  72000],
       [    54, 108000],
       [    39,  77000],
       [    38,  61000],
       [    38, 113000],
       [    37,  75000],
       [    42,  90000],
       [    37,  57000],
       [    36,  99000],
       [    60,  34000],
       [    54,  70000],
       [    41,  72000],
       [    40,  71000],
       [    42,  54000],
       [    43, 129000],
       [    53,  34000],
       [    47,  50000],
       [    42,  79000],
       [    42, 104000]
```

```
#VISUALISATION-Before applying cluster
import matplotlib.pyplot as plt
plt.scatter(df['Age'],df['EstimatedSalary'])
#Here we have got only one cluster before applying any
clustering technique
```

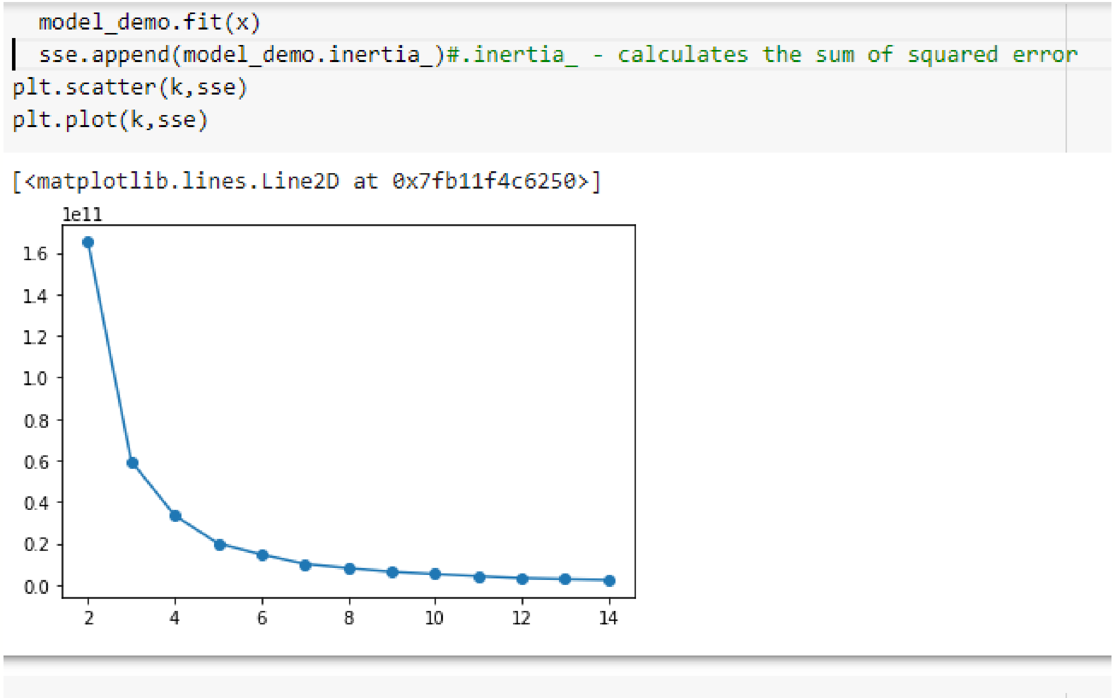


```
#VISUALISATION
import matplotlib.pyplot as plt
plt.scatter(df['Age'],df['EstimatedSalary'])
#Here we have got only one cluster before applying any clustering technique
```

```
<matplotlib.collections.PathCollection at 0x7fb11f788e90>
```

```
#Here our main task is to find out the number of
clusters(k)
import numpy as np
np.sqrt(400) # 400 is the total no of points
#No of cluster - k
#k value should not exceed the square root of the total
 no of points
#output - 20.0
#We need to find out the number of clusters(k)
#1.ELBOW METHOD - Slightly Confusing
```

```
#2.SILHOUETTE SCORE METHOD - Very accurate
#1.ELBOW METHOD
from sklearn.cluster import KMeans
k = range(2,15)# my range is in between 2 and 14
sse = [] #blank list
#for i in range(2,15):
for i in k :
  model_demo = KMeans(n_clusters = i,random_state = 0)
  model_demo.fit(x)
  sse.append(model_demo.inertia_)#.inertia_ - calculate the sum of squared error
plt.scatter(k,sse)
plt.plot(k,sse)
```

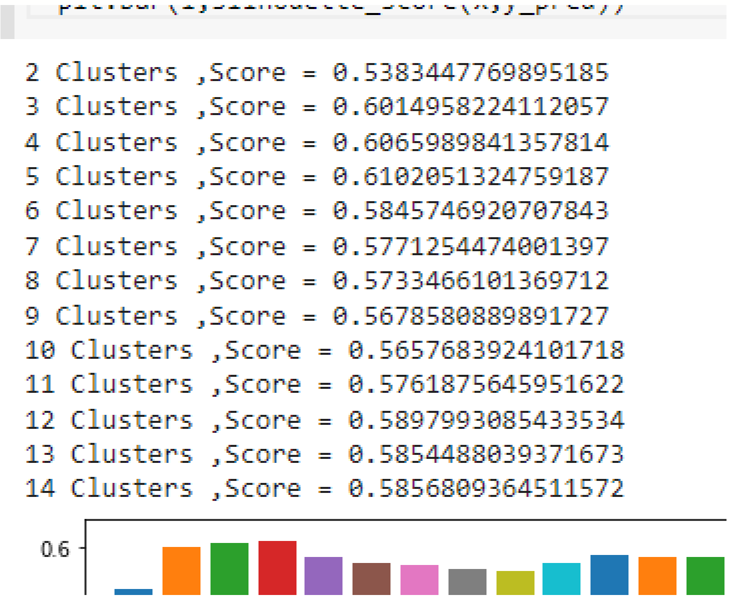


```
  model_demo.fit(x)
  sse.append(model_demo.inertia_)#.inertia_ - calculates the sum of squared error
plt.scatter(k,sse)
plt.plot(k,sse)
```

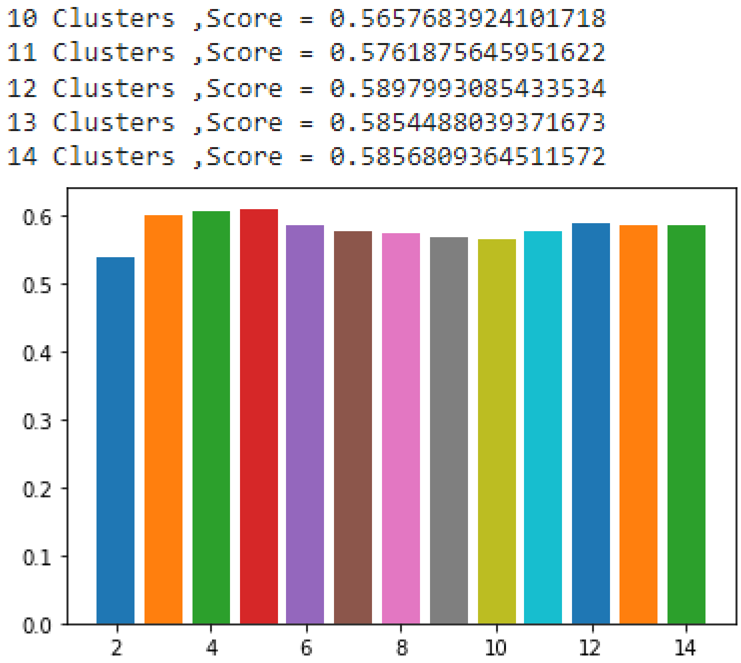
```
[<matplotlib.lines.Line2D at 0x7fb11f4c6250>]
```

```
#We will now consider the point at which the eblow is more prominent(projecting from something)
```

```
# We will consider k as 5 for now , but we are not sure
#2.SILHOUETTE SCORE METHOD
from sklearn.metrics import silhouette_score
k = range(2,15)
for i in k:
  model_demo = KMeans(n_clusters = i,random_state = 0)
  model_demo.fit(x)
  y_pred = model_demo.predict(x)
  print(f"{i} Clusters ,Score = {silhouette_score(x,y_p
red)}")
  plt.bar(i,silhouette_score(x,y_pred))
```

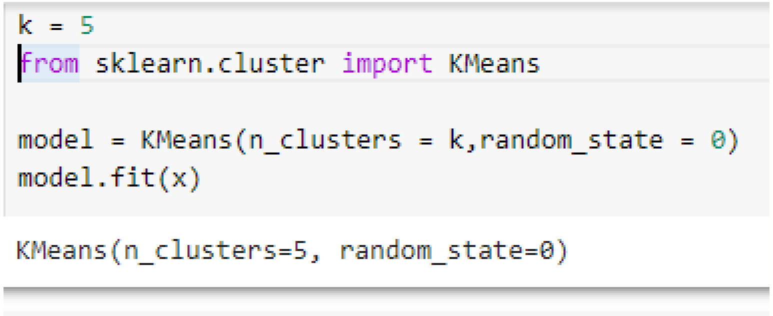


```
  plt.bar(i,silhouette_score(x,y_pred))

2 Clusters ,Score = 0.5383447769895185
3 Clusters ,Score = 0.6014958224112057
4 Clusters ,Score = 0.6065989841357814
5 Clusters ,Score = 0.6102051324759187
6 Clusters ,Score = 0.5845746920707843
7 Clusters ,Score = 0.5771254474001397
8 Clusters ,Score = 0.5733466101369712
9 Clusters ,Score = 0.5678580889891727
10 Clusters ,Score = 0.5657683924101718
11 Clusters ,Score = 0.5761875645951622
12 Clusters ,Score = 0.5897993085433534
13 Clusters ,Score = 0.5854488039371673
14 Clusters ,Score = 0.5856809364511572
```



```
10 Clusters ,Score = 0.5657683924101718
11 Clusters ,Score = 0.5761875645951622
12 Clusters ,Score = 0.5897993085433534
13 Clusters ,Score = 0.5854488039371673
14 Clusters ,Score = 0.5856809364511572
```

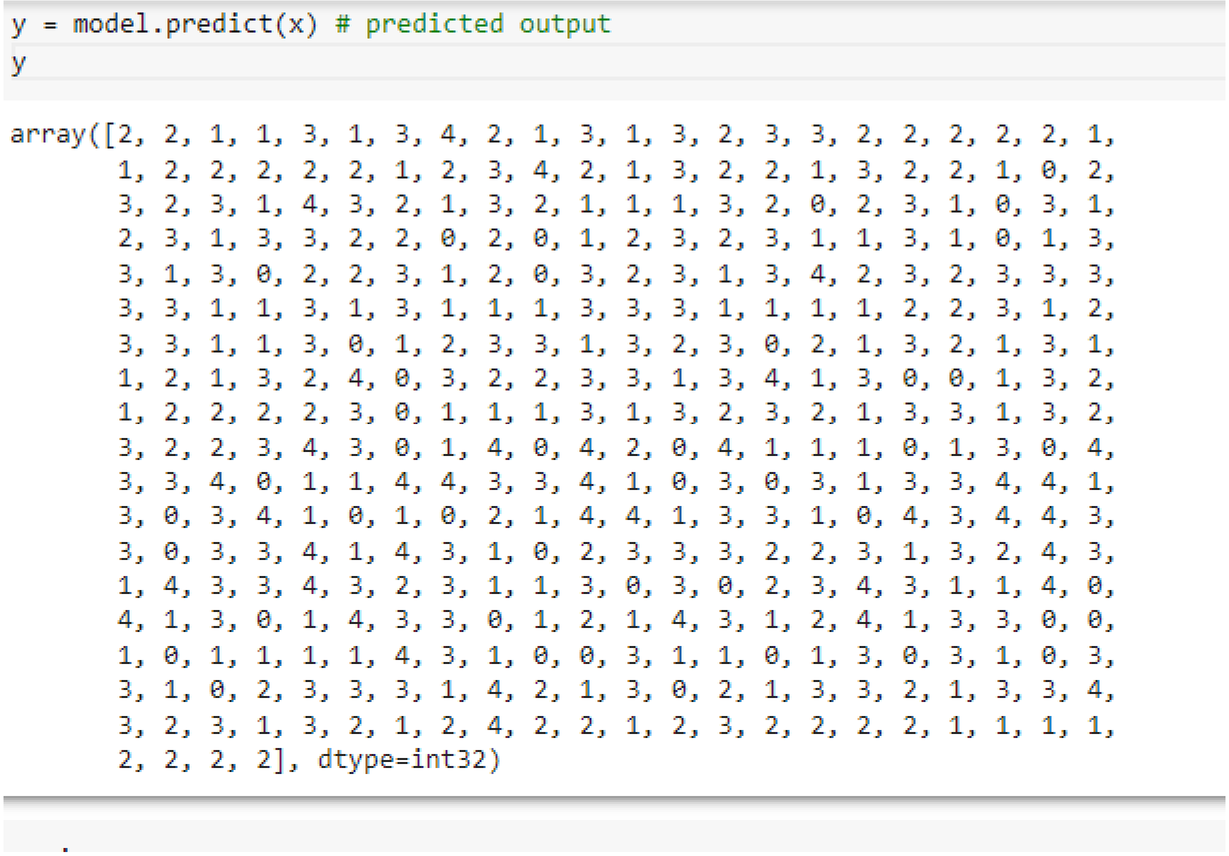
```
#CONFIRMATION : THE No OF CLUSTERS TO BE CONSIDERED IS
5(silhouette_score is maximum)

#7.APPLY CLUSTERER
k = 5
from sklearn.cluster import KMeans
model = KMeans(n_clusters = k,random_state = 0)
model.fit(x)
```



```
k = 5
from sklearn.cluster import KMeans

model = KMeans(n_clusters = k,random_state = 0)
model.fit(x)
```

```
KMeans(n_clusters=5, random_state=0)
```

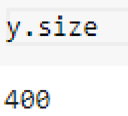
```
y = model.predict(x) # predicted output
y
```



```
y = model.predict(x) # predicted output
y
```

```
array([2, 2, 1, 1, 3, 1, 3, 4, 2, 1, 3, 1, 3, 2, 3, 3, 2, 2, 2, 2, 2, 1,
       1, 2, 2, 2, 2, 2, 1, 2, 3, 4, 2, 1, 3, 2, 2, 1, 3, 2, 2, 1, 0, 2,
       3, 2, 3, 1, 4, 3, 2, 1, 3, 2, 1, 1, 1, 3, 2, 0, 2, 3, 1, 0, 3, 1,
       2, 3, 1, 3, 3, 2, 2, 0, 2, 0, 1, 2, 3, 2, 3, 1, 1, 3, 1, 0, 1, 3,
       3, 1, 3, 0, 2, 2, 3, 1, 2, 0, 3, 2, 3, 1, 3, 4, 2, 3, 2, 3, 3, 3,
       3, 3, 1, 1, 3, 1, 3, 1, 1, 1, 3, 3, 3, 1, 1, 1, 1, 2, 2, 3, 1, 2,
       3, 3, 1, 1, 3, 0, 1, 2, 3, 3, 1, 3, 2, 3, 0, 2, 1, 3, 2, 1, 3, 1,
       1, 2, 1, 3, 2, 4, 0, 3, 2, 2, 3, 3, 1, 3, 4, 1, 3, 0, 0, 1, 3, 2,
       1, 2, 2, 2, 2, 3, 0, 1, 1, 1, 3, 1, 3, 2, 3, 2, 1, 3, 3, 1, 3, 2,
       3, 2, 2, 3, 4, 3, 0, 1, 4, 0, 4, 2, 0, 4, 1, 1, 1, 0, 1, 3, 0, 4,
       3, 3, 4, 0, 1, 1, 4, 4, 3, 3, 4, 1, 0, 3, 0, 3, 1, 3, 3, 4, 4, 1,
       3, 0, 3, 4, 1, 0, 1, 0, 2, 1, 4, 4, 1, 3, 3, 1, 0, 4, 3, 4, 4, 3,
       3, 0, 3, 3, 4, 1, 4, 3, 1, 0, 2, 3, 3, 3, 2, 2, 3, 1, 3, 2, 4, 3,
       1, 4, 3, 3, 4, 3, 2, 3, 1, 1, 3, 0, 3, 0, 2, 3, 4, 3, 1, 1, 4, 0,
       4, 1, 3, 0, 1, 4, 3, 3, 0, 1, 2, 1, 4, 3, 1, 2, 4, 1, 3, 3, 0, 0,
       1, 0, 1, 1, 1, 1, 4, 3, 1, 0, 0, 3, 1, 1, 0, 1, 3, 0, 3, 1, 0, 3,
       3, 1, 0, 2, 3, 3, 3, 1, 4, 2, 1, 3, 0, 2, 1, 3, 3, 2, 1, 3, 3, 4,
       3, 2, 3, 1, 3, 2, 1, 2, 4, 2, 2, 1, 2, 3, 2, 2, 2, 2, 1, 1, 1, 1,
       2, 2, 2, 2], dtype=int32)
```

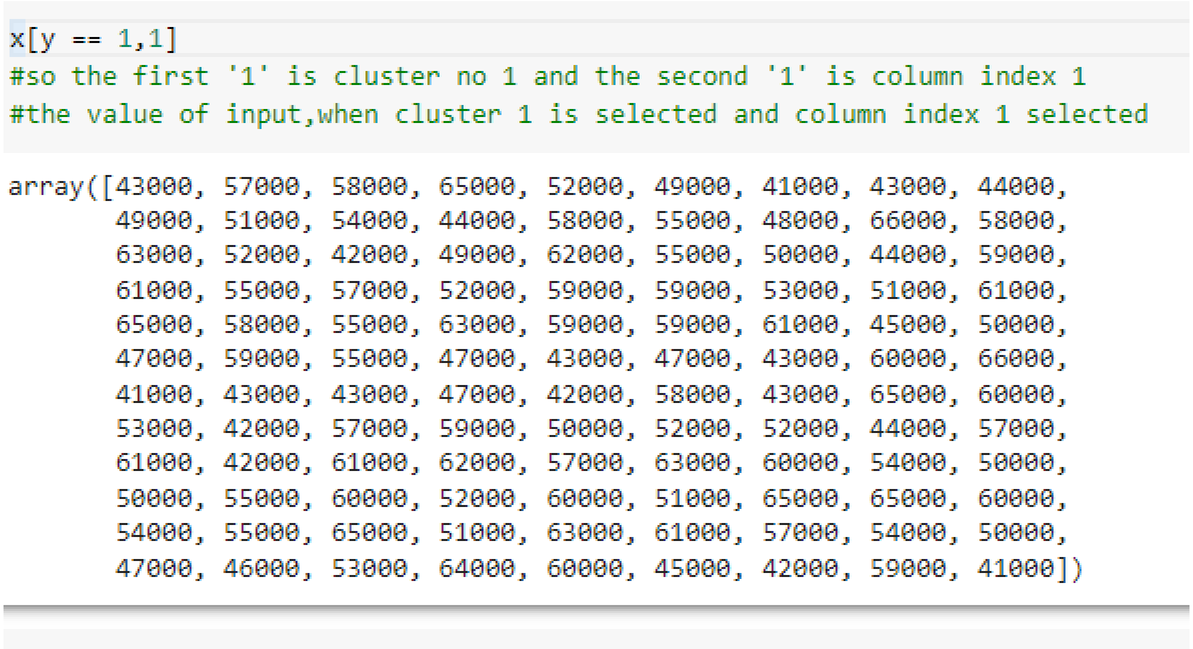
```
y.size
```



```
y.size
```

```
400
```

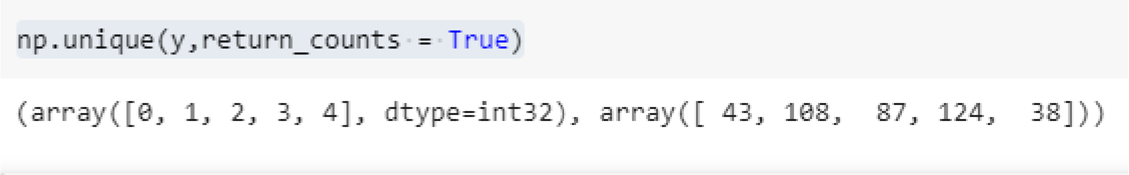
```
x[y == 1,1]
#so the first '1' is cluster no 1 and the second '1' is
 column index 1
#the value of input,when cluster 1 is selected and colu
mn index 1 selected
```



```
x[y == 1,1]
#so the first '1' is cluster no 1 and the second '1' is column index 1
#the value of input,when cluster 1 is selected and column index 1 selected
```

```
array([43000, 57000, 58000, 65000, 52000, 49000, 41000, 43000, 44000,
       49000, 51000, 54000, 44000, 58000, 55000, 48000, 66000, 58000,
       63000, 52000, 42000, 49000, 62000, 55000, 50000, 44000, 59000,
       61000, 55000, 57000, 52000, 59000, 59000, 53000, 51000, 61000,
       65000, 58000, 55000, 63000, 59000, 59000, 61000, 45000, 50000,
       47000, 59000, 55000, 47000, 43000, 47000, 43000, 60000, 66000,
       41000, 43000, 43000, 47000, 42000, 58000, 43000, 65000, 60000,
       53000, 42000, 57000, 59000, 50000, 52000, 52000, 44000, 57000,
       61000, 42000, 61000, 62000, 57000, 63000, 60000, 54000, 50000,
       50000, 55000, 60000, 52000, 60000, 51000, 65000, 65000, 60000,
       54000, 55000, 65000, 51000, 63000, 61000, 57000, 54000, 50000,
       47000, 46000, 53000, 64000, 60000, 45000, 42000, 59000, 41000])
```

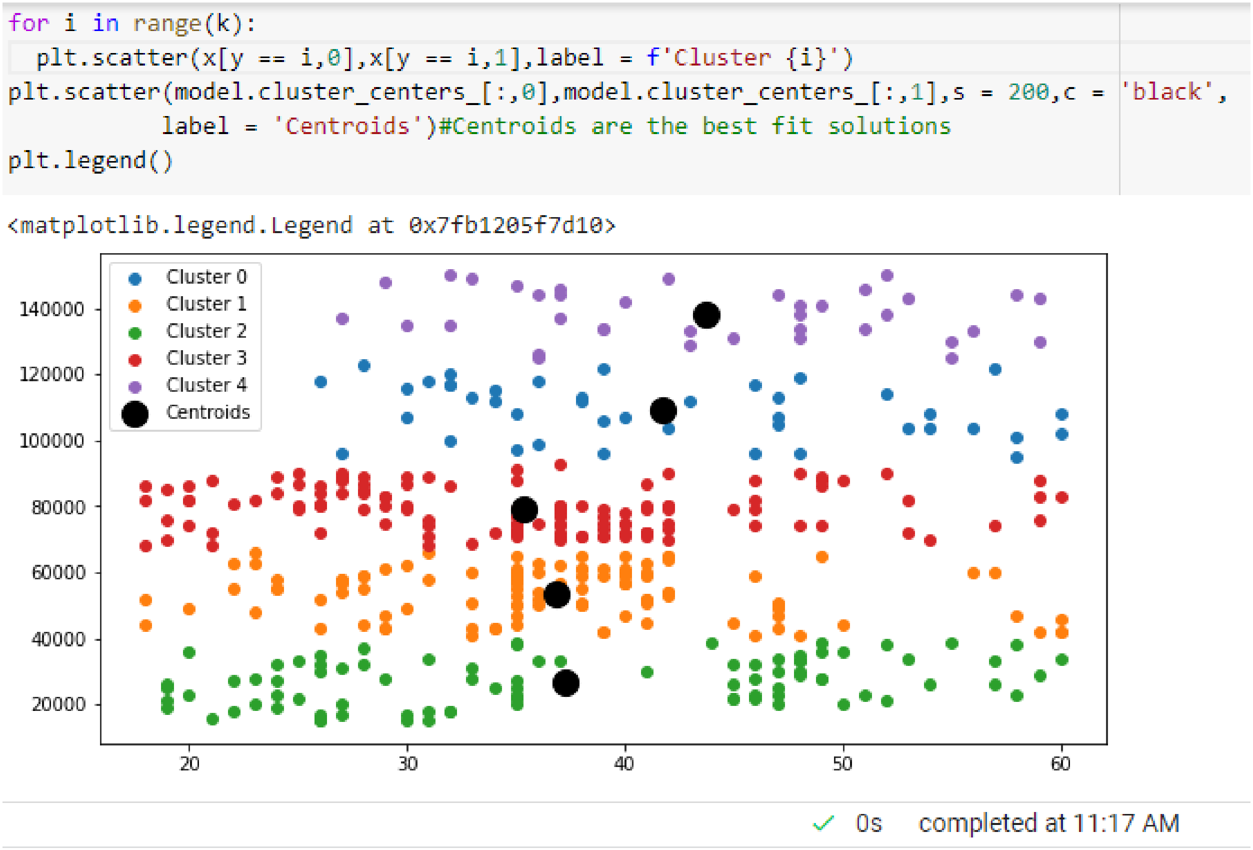
```
np.unique(y,return_counts = True)
```



```
np.unique(y,return_counts = True)
```

```
(array([0, 1, 2, 3, 4], dtype=int32), array([ 43, 108,  87, 124,  38]))
```

```
#FINAL VISUALISATION
plt.figure(figsize = (10,5))
for i in range(k):
  plt.scatter(x[y == i,0],x[y == i,1],label = f'Cluster
 {i}')
plt.scatter(model.cluster_centers_[:,0],model.cluster_c
enters_[:,1],s = 200,c = 'black',
           label = 'Centroids')#Centroids are the best
fit solutions
plt.legend()
```



```
for i in range(k):
  plt.scatter(x[y == i,0],x[y == i,1],label = f'Cluster {i}')
plt.scatter(model.cluster_centers_[:,0],model.cluster_centers_[:,1],s = 200,c = 'black',
           label = 'Centroids')#Centroids are the best fit solutions
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7fb1205f7d10>
```

✓ 0s completed at 11:17 AM

#My Github account URL: https://github.com/Adepushivani

(Or)

https://github.com/Adepushivani/Rinex-majorprojects