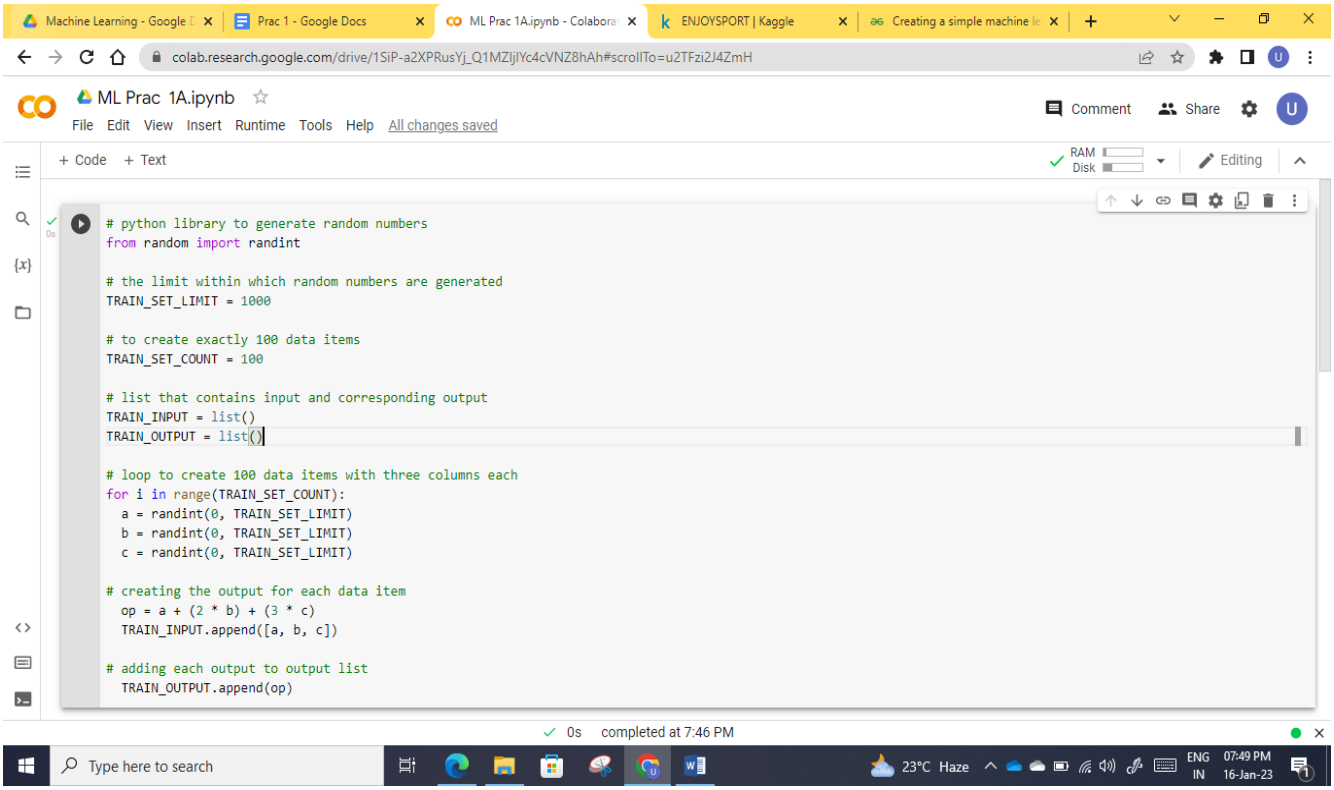
Design a simple machine learning model to train the training instances and test the same

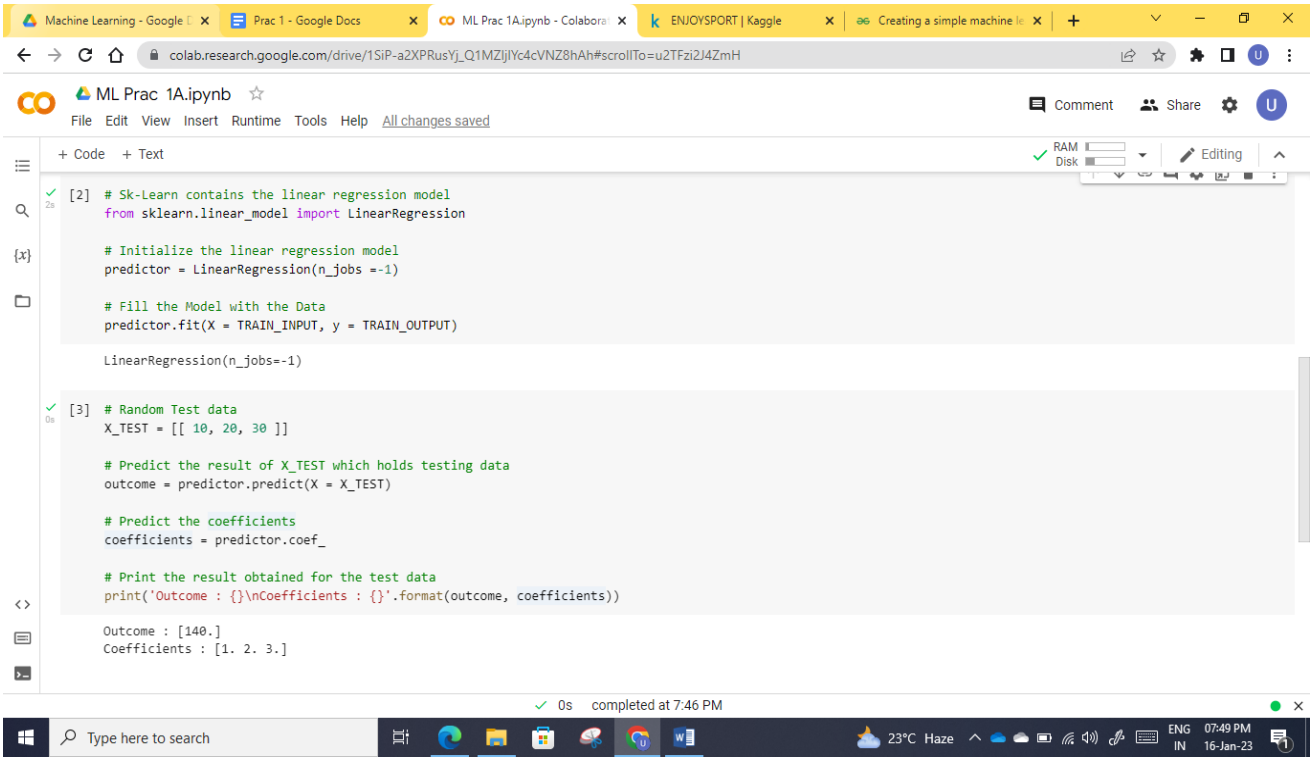
Code :

```
# python library to generate random numbers
from random import randint
# the limit within which random numbers are generated
TRAIN_SET_LIMIT = 1000
# to create exactly 100 data items
TRAIN_SET_COUNT = 100

# list that contains input and corresponding output
TRAIN_INPUT = list()
TRAIN_OUTPUT = list()

# loop to create 100 data items with three columns each
for i in range(TRAIN_SET_COUNT):
        a = randint(0, TRAIN_SET_LIMIT)
        b = randint(0, TRAIN_SET_LIMIT)
        c = randint(0, TRAIN_SET_LIMIT)

        # creating the output for each data item
        op = a + (2 * b) + (3 * c)
        TRAIN_INPUT.append([a, b, c])
        # adding each output to output list
        TRAIN_OUTPUT.append(op)

# Sk-Learn contains the linear regression model
from sklearn.linear_model import LinearRegression
# Initialize the linear regression model
predictor = LinearRegression(n_jobs =-1)
# Fill the Model with the Data
```

```
predictor.fit(X = TRAIN_INPUT, y = TRAIN_OUTPUT)

# Random Test data
X_TEST = [[ 10, 20, 30 ]]
# Predict the result of X_TEST which holds testing data
outcome = predictor.predict(X = X_TEST)
# Predict the coefficients
coefficients = predictor.coef_
# Print the result obtained for the test data
print('Outcome : {}\nCoefficients : {}'.format(outcome, coefficients))
```

Output :




Machine Learning - Google
Prac 1 - Google Docs
ML Prac 1A.ipynb - Colabora
ENJOYSPORT | Kaggle
Creating a simple machine le
colab.research.google.com/drive/1SiP-a2XPRusYj_Q1MZIjIYc4cVNZ8hAh#scrollTo=u2TFzi2J4ZmH
ML Prac 1A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[2] # Sk-Learn contains the linear regression model
from sklearn.linear_model import LinearRegression
# Initialize the linear regression model
predictor = LinearRegression(n_jobs =-1)
# Fill the Model with the Data
predictor.fit(X = TRAIN_INPUT, y = TRAIN_OUTPUT)
LinearRegression(n_jobs=-1)
[3] # Random Test data
X_TEST = [[10, 20, 30]]
# Predict the result of X_TEST which holds testing data
outcome = predictor.predict(X = X_TEST)
# Predict the coefficients
coefficients = predictor.coef_
# Print the result obtained for the test data
print('Outcome : {}\nCoefficients : {}'.format(outcome, coefficients))
Outcome : [140.]
Coefficients : [1. 2. 3.]
0s completed at 7:46 PM
Type here to search
23°C Haze
ENG
IN
07:49 PM
16-Jan-23

Implement and demonstrate the FIND-S algorithm for finding the most specific hypothesis based on a given set of training data samples. Read the training data from a .CSV file

Code :

```
import pandas as pd
import numpy as np
d = pd.read_csv("/content/ENJOYSPORT.csv")
print(d)

a = np.array(d)[:,:-1]
print(" The attributes are: ",a)

t = np.array(d)[:,-1]
print("The target is: ",t)

def fun(c,t):
    for i, val in enumerate(t):
    if val == "Yes":
        specific_hypothesis = c[i].copy()
        break

    for i, val in enumerate(c):
        if t[i] == "Yes":
            for x in range(len(specific_hypothesis)):
                if val[x] != specific_hypothesis[x]:
                    specific_hypothesis[x] = '?'
                else:
                    pass
    return specific_hypothesis

print(" The final hypothesis is:", fun(a,t))
```

Output :

```
In [1]: import pandas as pd
        import numpy as np

In [2]: data = pd.read_csv("ENJOYSPORT.csv")

In [3]: data
```

Out[3]:

	Sky	AirTemp	Humidity	Wind	Water	Forecast	EnjoySport
0	Sunny	Warm	Normal	Strong	Warm	Same	1
1	Sunny	Warm	High	Strong	Warm	Same	1
2	Rainy	Cold	High	Strong	Warm	Change	0
3	Sunny	Warm	High	Strong	Cool	Change	1

```
In [4]: data = pd.read_csv("ENJOYSPORT.csv")

In [5]: d
```

```
Out[5]: array([['Sunny', 'Warm', 'Normal', 'Strong', 'Warm', 'Same'],
               ['Sunny', 'Warm', 'High', 'Strong', 'Warm', 'Same'],
               ['Rainy', 'Cold', 'High', 'Strong', 'Warm', 'Change'],
               ['Sunny', 'Warm', 'High', 'Strong', 'Cool', 'Change']],
              dtype=object)
```

```
In [6]: target = np.array(data)[:,-1]
```

```
[6]: target = np.array(data)[:,-1]

[7]: target
```

```
[7]: array([1, 1, 0, 1], dtype=object)
```

```
[2]: def train(c,t):
         for i , val in enumerate(t):
             if val == 1:
                 specific_hypothesis = c[i].copy()
                 break
         for i , val in enumerate(c):
             if t[i] == 1:
                 for x in range(len(specific_hypothesis)):
                     if val[x] != specific_hypothesis[x]:
                         specific_hypothesis[x] = '?'
                     else:
                         pass
         return specific_hypothesis
```

```
[3]: print("The final hypothesis is : ",train(d,target))
```

```
The final hypothesis is :  ['Sunny' 'Warm' '?' 'Strong' '?' '?']
```

Perform Data Loading, Feature selection (Principal Component analysis) and Feature Scoring and Ranking

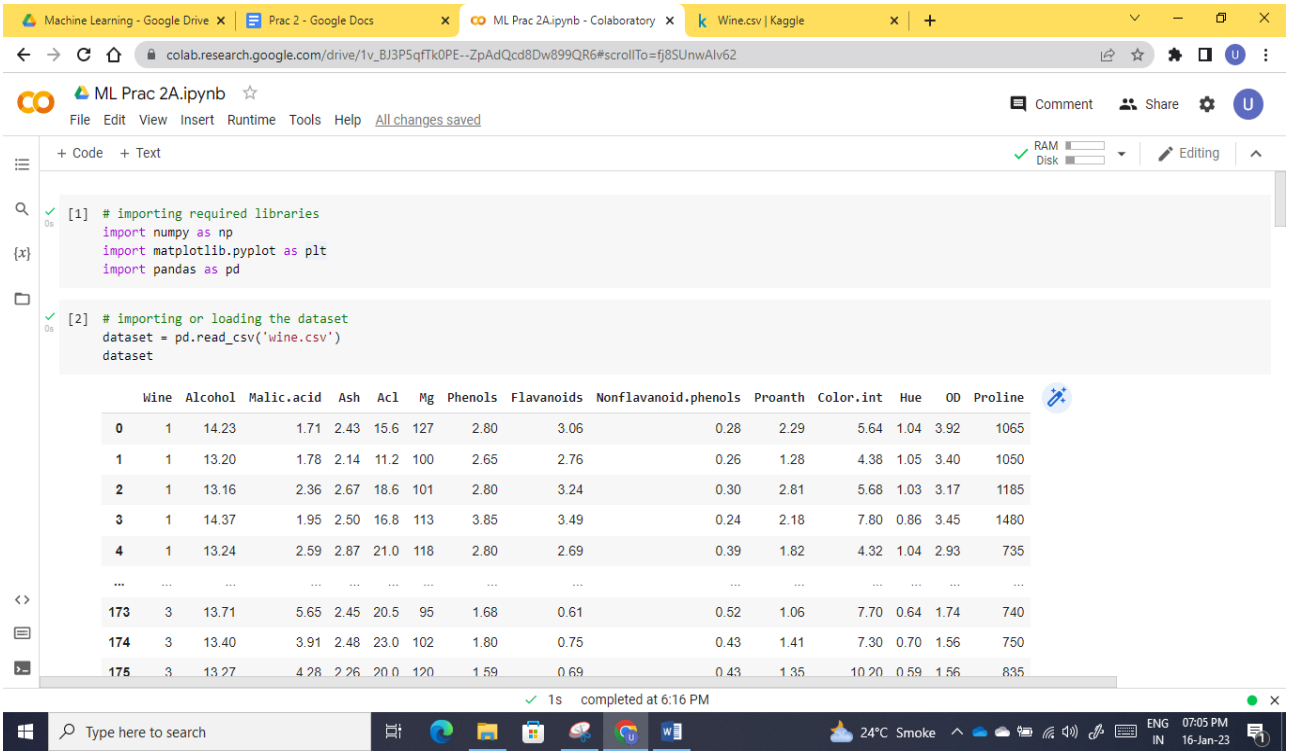
Code :

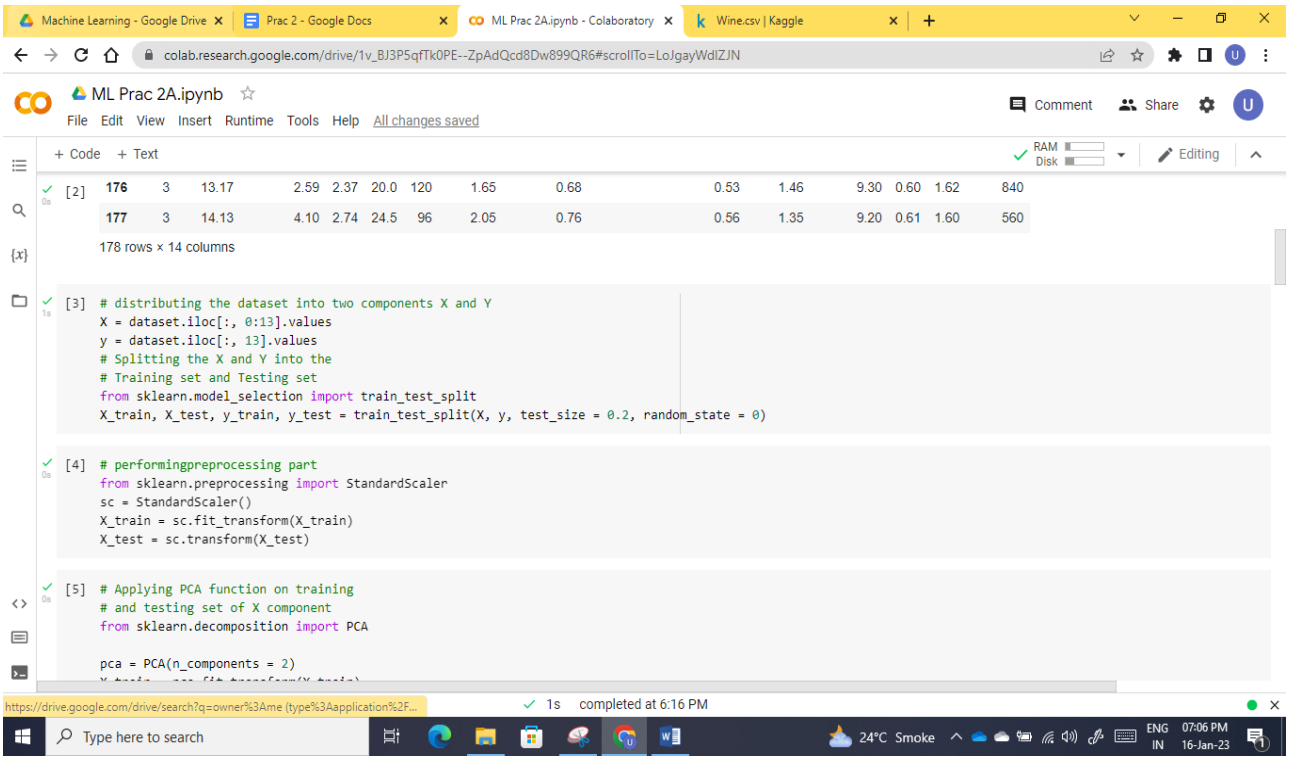
```
# importing required libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
# importing or loading the dataset
dataset = pd.read_csv('wine.csv')
dataset

# distributing the dataset into two components X and Y
X = dataset.iloc[:, 0:13].values
y = dataset.iloc[:, 13].values
# Splitting the X and Y into the
# Training set and Testing set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)

# performing preprocessing part
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Applying PCA function on training
# and testing set of X component
from sklearn.decomposition import PCA
pca = PCA(n_components = 2)
X_train = pca.fit_transform(X_train)
X_test = pca.transform(X_test)
```

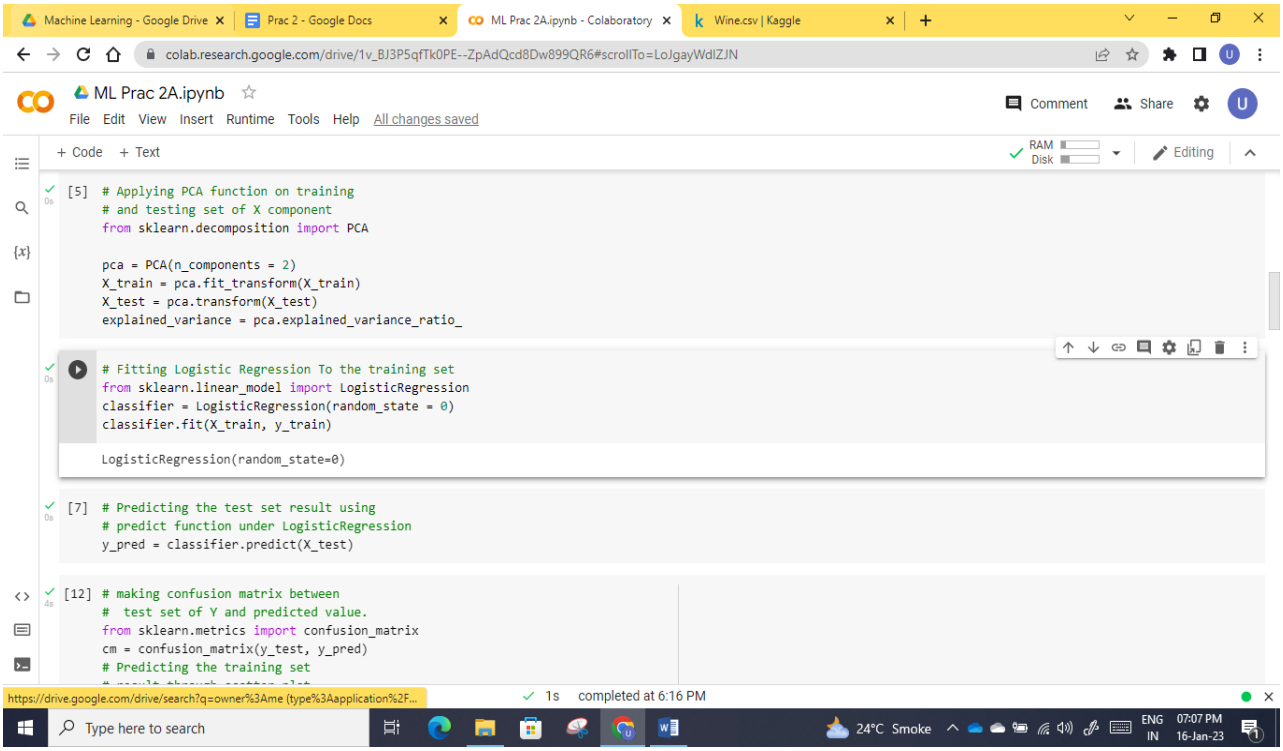
```
explained_variance = pca.explained_variance_ratio_

# Fitting Logistic Regression To the training set
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 0)
classifier.fit(X_train, y_train)
# Predicting the test set result using
# predict function under LogisticRegression
y_pred = classifier.predict(X_test)

# making confusion matrix between
#  test set of Y and predicted value.
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)

# Predicting the training set
# result through scatter plot
from matplotlib.colors import ListedColormap
X_set, y_set = X_train, y_train
X1, X2=np.meshgrid(np.arange(start=X_set[:,0].min()-1,stop=X_set[:, 0].max() + 1,step = 0.01),
                   np.arange(start=X_set[:,1].min()-1,stop=X_set[:, 1].max() + 1, step = 0.01))

plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(),X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.75,cmap = ListedColormap(('yellow', 'white', 'aquamarine')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],c = ListedColormap(('red', 'green',
                'blue'))(i), label = j)
plt.title('Logistic Regression (Training set)')
plt.xlabel('PC1') # for Xlabel
plt.ylabel('PC2') # for Ylabel
plt.legend() # to show legend
# show scatter plot
```

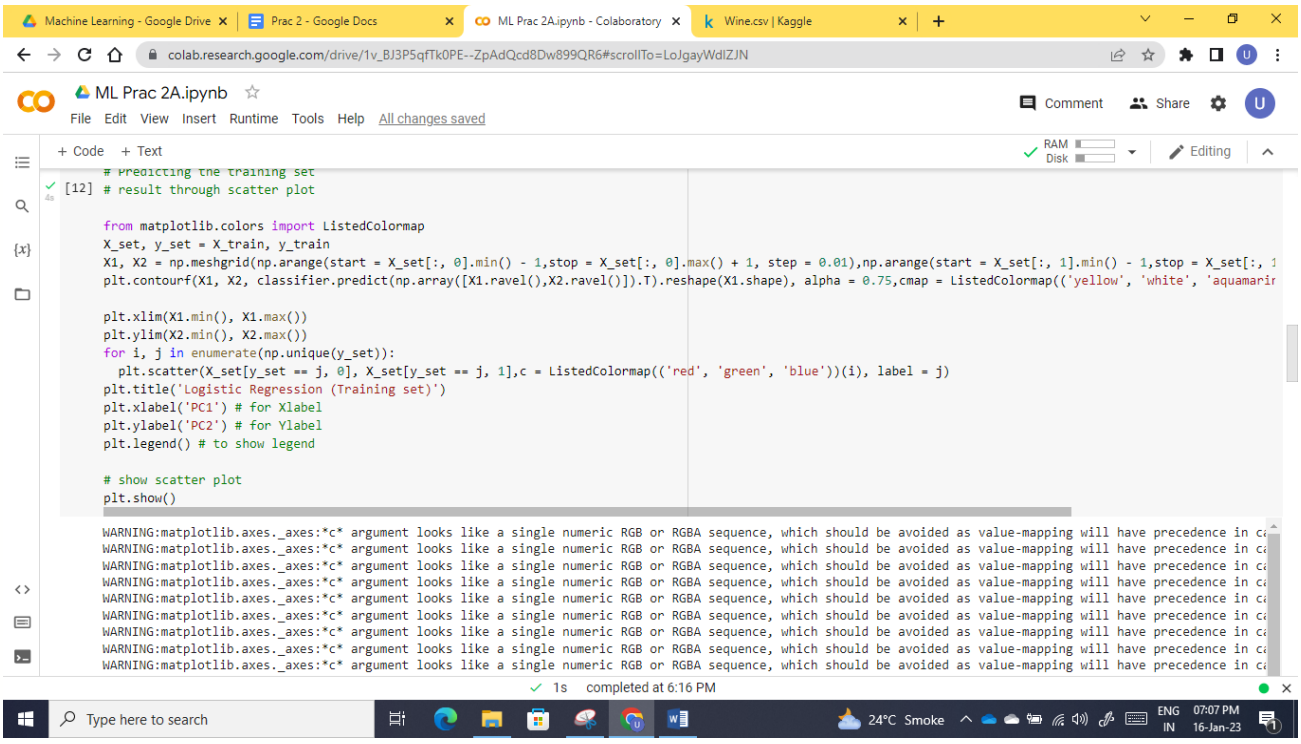
```
plt.show()

# Visualising the Test set results through scatter plot
from matplotlib.colors import ListedColormap
X_set, y_set = X_test, y_test
X1,X2=np.meshgrid(np.arange(start=X_set[:,0].min()-1,stop=X_set[:,0].max()+1,step=0.01),
                  np.arange(start = X_set[:, 1].min() - 1,stop = X_set[:, 1].max() + 1, step = 0.01))

plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(),X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.75,cmap = ListedColormap(('yellow', 'white', 'aquamarine')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())

for i, j in enumerate(np.unique(y_set)):
        plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],c = ListedColormap(('red', 'green',
                    'blue'))(i), label = j)
# title for scatter plot
plt.title('Logistic Regression (Test set)')
plt.xlabel('PC1') # for Xlabel
plt.ylabel('PC2') # for Ylabel
plt.legend()
# show scatter plot
plt.show()
```

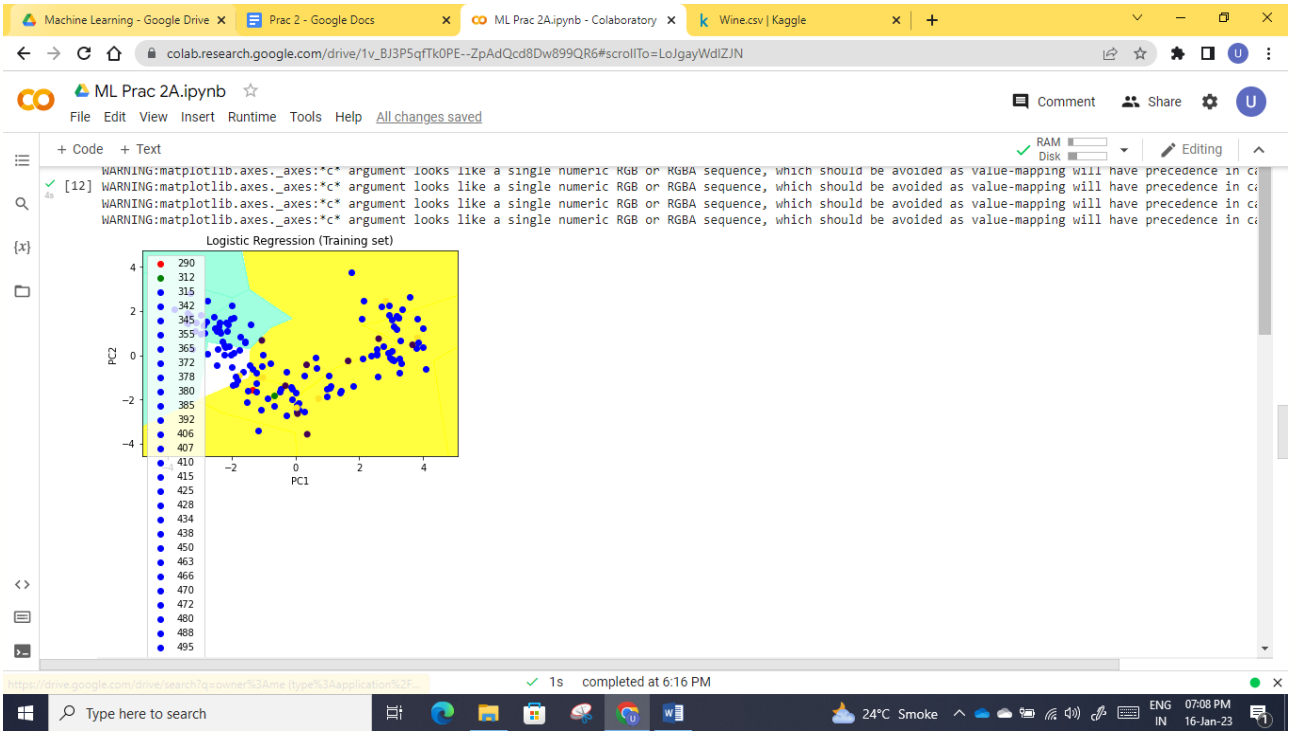
Output :

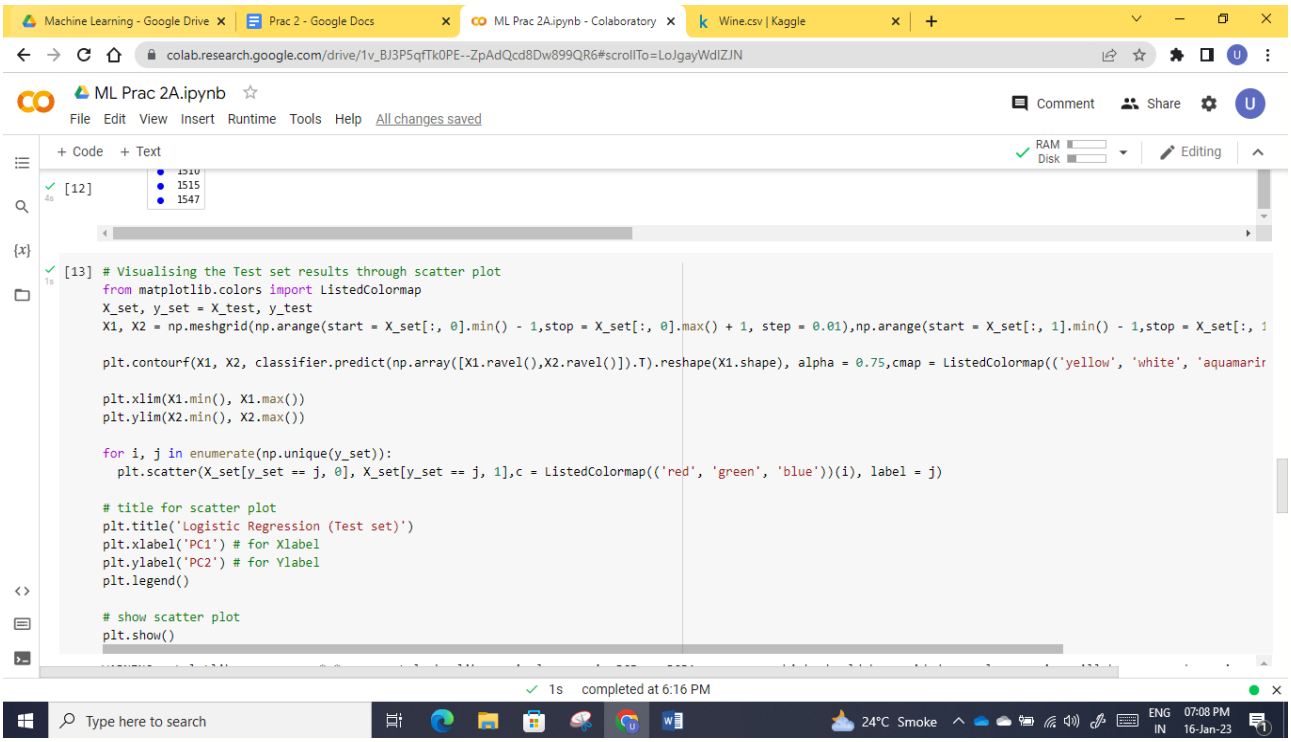

Machine Learning - Google Drive
Prac 2 - Google Docs
ML Prac 2A.ipynb - Colaboratory
Wine.csv | Kaggle
colab.research.google.com/drive/1v_BJ3P5qfTk0PE--ZpAdQcd8Dw899QR6#scrollTo=fj8SUnwAlv62
ML Prac 2A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[1] # importing required libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
[2] # importing or loading the dataset
dataset = pd.read_csv('wine.csv')
dataset
Wine Alcohol Malic.acid Ash Acl Mg Phenols Flavanoids Nonflavanoid.phenols Proanth Color.int Hue OD Proline
0 1 14.23 1.71 2.43 15.6 127 2.80 3.06 0.28 2.29 5.64 1.04 3.92 1065
1 1 13.20 1.78 2.14 11.2 100 2.65 2.76 0.26 1.28 4.38 1.05 3.40 1050
2 1 13.16 2.36 2.67 18.6 101 2.80 3.24 0.30 2.81 5.68 1.03 3.17 1185
3 1 14.37 1.95 2.50 16.8 113 3.85 3.49 0.24 2.18 7.80 0.86 3.45 1480
4 1 13.24 2.59 2.87 21.0 118 2.80 2.69 0.39 1.82 4.32 1.04 2.93 735
173 3 13.71 5.65 2.45 20.5 95 1.68 0.61 0.52 1.06 7.70 0.64 1.74 740
174 3 13.40 3.91 2.48 23.0 102 1.80 0.75 0.43 1.41 7.30 0.70 1.56 750
175 3 13.27 4.28 2.26 20.0 120 1.59 0.69 0.43 1.35 10.20 0.59 1.56 835
1s completed at 6:16 PM
Type here to search
24°C Smoke
ENG IN 07:05 PM 16-Jan-23

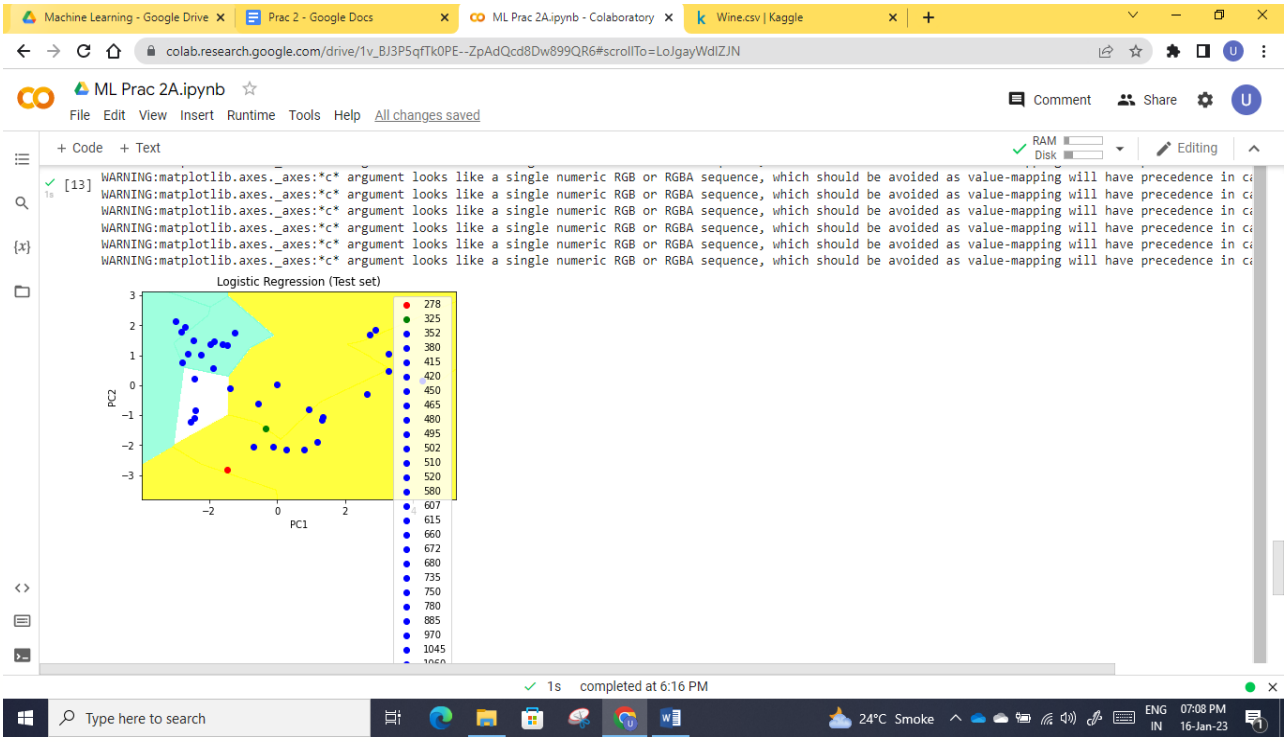

Machine Learning - Google Drive
Prac 2 - Google Docs
ML Prac 2A.ipynb - Colaboratory
Wine.csv | Kaggle
colab.research.google.com/drive/1v_BJ3P5qfTk0PE--ZpAdQcd8Dw899QR6#scrollTo=LoJgayWdIZJN
ML Prac 2A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[2] 176 3 13.17 2.59 2.37 20.0 120 1.65 0.68 0.53 1.46 9.30 0.60 1.62 840
177 3 14.13 4.10 2.74 24.5 96 2.05 0.76 0.56 1.35 9.20 0.61 1.60 560
178 rows × 14 columns
[3] # distributing the dataset into two components X and Y
X = dataset.iloc[:, 0:13].values
y = dataset.iloc[:, 13].values
# Splitting the X and Y into the
# Training set and Testing set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)
[4] # performingpreprocessing part
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
[5] # Applying PCA function on training
# and testing set of X component
from sklearn.decomposition import PCA
pca = PCA(n_components = 2)
1s completed at 6:16 PM
Type here to search
24°C Smoke
ENG IN 07:06 PM 16-Jan-23


Machine Learning - Google Drive
Prac 2 - Google Docs
ML Prac 2A.ipynb - Colaboratory
Wine.csv | Kaggle
colab.research.google.com/drive/1v_BJ3P5qfTk0PE--ZpAdQcd8Dw899QR6#scrollTo=LoJgayWdIZJN
ML Prac 2A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[5] # Applying PCA function on training
# and testing set of X component
from sklearn.decomposition import PCA
pca = PCA(n_components = 2)
X_train = pca.fit_transform(X_train)
X_test = pca.transform(X_test)
explained_variance = pca.explained_variance_ratio_
# Fitting Logistic Regression To the training set
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 0)
classifier.fit(X_train, y_train)
LogisticRegression(random_state=0)
[7] # Predicting the test set result using
# predict function under LogisticRegression
y_pred = classifier.predict(X_test)
[12] # making confusion matrix between
# test set of Y and predicted value.
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
# Predicting the training set
1s completed at 6:16 PM
Type here to search
24°C Smoke
ENG IN
07:07 PM
16-Jan-23


Machine Learning - Google Drive
Prac 2 - Google Docs
ML Prac 2A.ipynb - Colaboratory
Wine.csv | Kaggle
colab.research.google.com/drive/1v_BJ3P5qfTk0PE--ZpAdQcd8Dw899QR6#scrollTo=LoJgayWdIZJN
ML Prac 2A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[12] # result through scatter plot
from matplotlib.colors import ListedColormap
X_set, y_set = X_train, y_train
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1,stop = X_set[:, 0].max() + 1, step = 0.01),np.arange(start = X_set[:, 1].min() - 1,stop = X_set[:, 1
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(),X2.ravel()]).T).reshape(X1.shape), alpha = 0.75,cmap = ListedColormap(('yellow', 'white', 'aquamarir
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],c = ListedColormap(('red', 'green', 'blue'))(i), label = j)
plt.title('Logistic Regression (Training set)')
plt.xlabel('PC1') # for Xlabel
plt.ylabel('PC2') # for Ylabel
plt.legend() # to show legend
# show scatter plot
plt.show()
WARNING:matplotlib.axes._axes:*c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-mapping will have precedence in ca
1s completed at 6:16 PM
Type here to search
24°C Smoke
ENG IN
07:07 PM
16-Jan-23


Machine Learning - Google Drive
Prac 2 - Google Docs
ML Prac 2A.ipynb - Colaboratory
Wine.csv | Kaggle
colab.research.google.com/drive/1v_BJ3P5qfTk0PE--ZpAdQcd8Dw899QR6#scrollTo=LoJgayWdIZJN
ML Prac 2A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[12] WARNING:matplotlib.axes._axes:*c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-mapping will have precedence in ca
WARNING:matplotlib.axes._axes:*c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-mapping will have precedence in ca
WARNING:matplotlib.axes._axes:*c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-mapping will have precedence in ca
WARNING:matplotlib.axes._axes:*c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-mapping will have precedence in ca
Logistic Regression (Training set)
PC1
PC2
290
312
315
342
345
355
365
372
378
380
385
392
406
407
410
415
425
428
434
438
450
463
466
470
472
480
488
495
1s completed at 6:16 PM
Type here to search
24°C Smoke
ENG IN
07:08 PM 16-Jan-23


Machine Learning - Google Drive
Prac 2 - Google Docs
ML Prac 2A.ipynb - Colaboratory
Wine.csv | Kaggle
colab.research.google.com/drive/1v_BJ3P5qfTk0PE--ZpAdQcd8Dw899QR6#scrollTo=LoJgayWdIZJN
ML Prac 2A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[12]
1515
1547
[13] # Visualising the Test set results through scatter plot
from matplotlib.colors import ListedColormap
X_set, y_set = X_test, y_test
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1,stop = X_set[:, 0].max() + 1, step = 0.01),np.arange(start = X_set[:, 1].min() - 1,stop = X_set[:, 1
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(),X2.ravel()]).T).reshape(X1.shape), alpha = 0.75,cmap = ListedColormap(('yellow', 'white', 'aquamarir
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],c = ListedColormap(('red', 'green', 'blue'))(i), label = j)
# title for scatter plot
plt.title('Logistic Regression (Test set)')
plt.xlabel('PC1') # for Xlabel
plt.ylabel('PC2') # for Ylabel
plt.legend()
# show scatter plot
plt.show()
1s completed at 6:16 PM
Type here to search
24°C Smoke
ENG IN
07:08 PM 16-Jan-23


Machine Learning - Google Drive
Prac 2 - Google Docs
ML Prac 2A.ipynb - Colaboratory
Wine.csv | Kaggle
colab.research.google.com/drive/1v_BJ3P5qfTk0PE--ZpAdQcd8Dw899QR6#scrollTo=LoJgayWdIZJN
ML Prac 2A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[13]
WARNING:matplotlib.axes._axes:*c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-mapping will have precedence in ca
Logistic Regression (Test set)
PC2
PC1
278
325
352
380
415
420
450
465
480
495
502
510
520
580
607
615
660
672
680
735
750
780
885
970
1045
1s completed at 6:16 PM
Type here to search
24°C Smoke
ENG
IN
07:08 PM
16-Jan-23

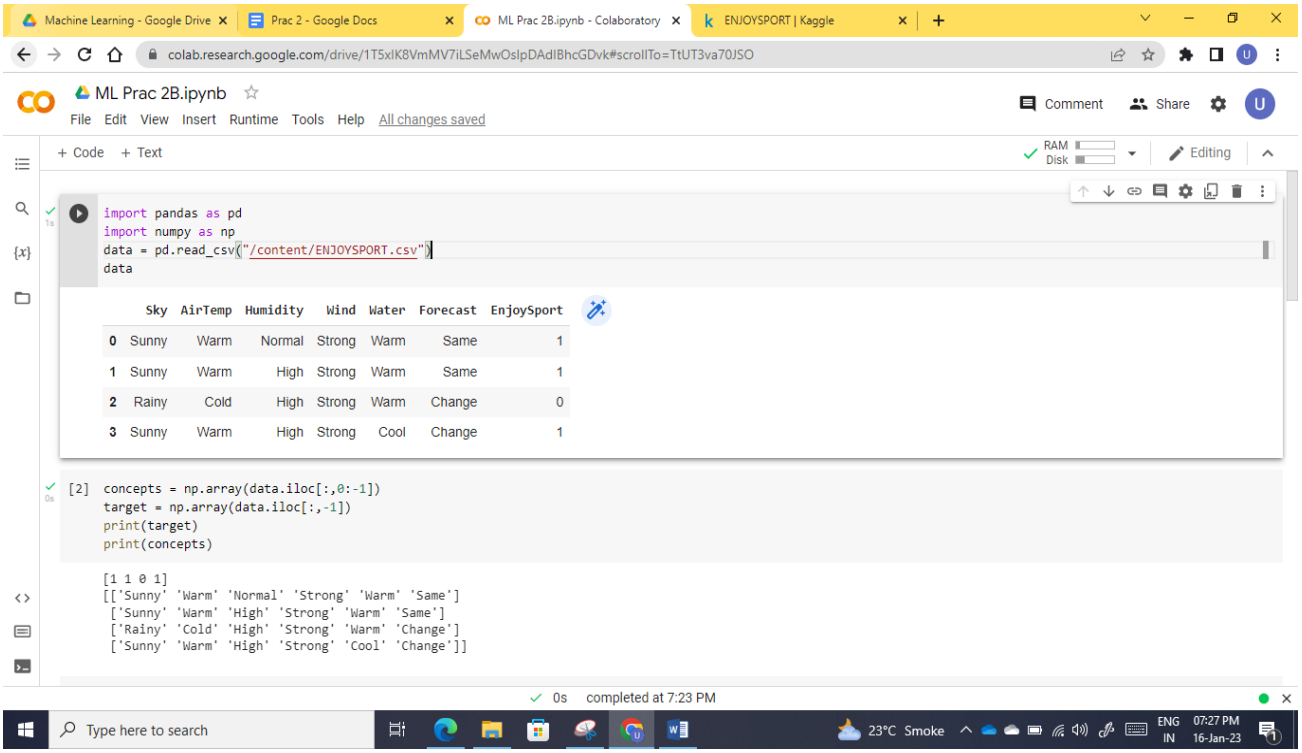
For a given set of training data examples stored in a .CSV file, implement and demonstrate the Candidate-Elimination algorithm to output a description of the set of all hypotheses consistent with the training examples

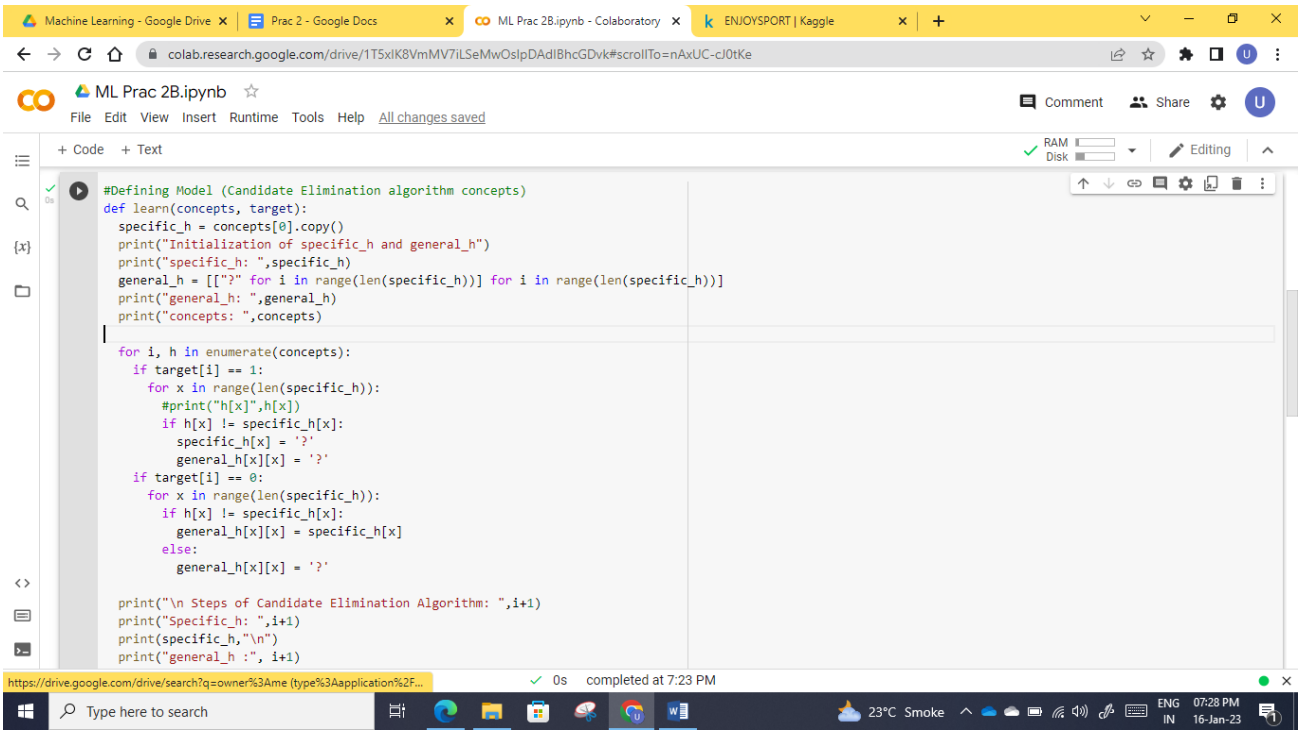
Code :

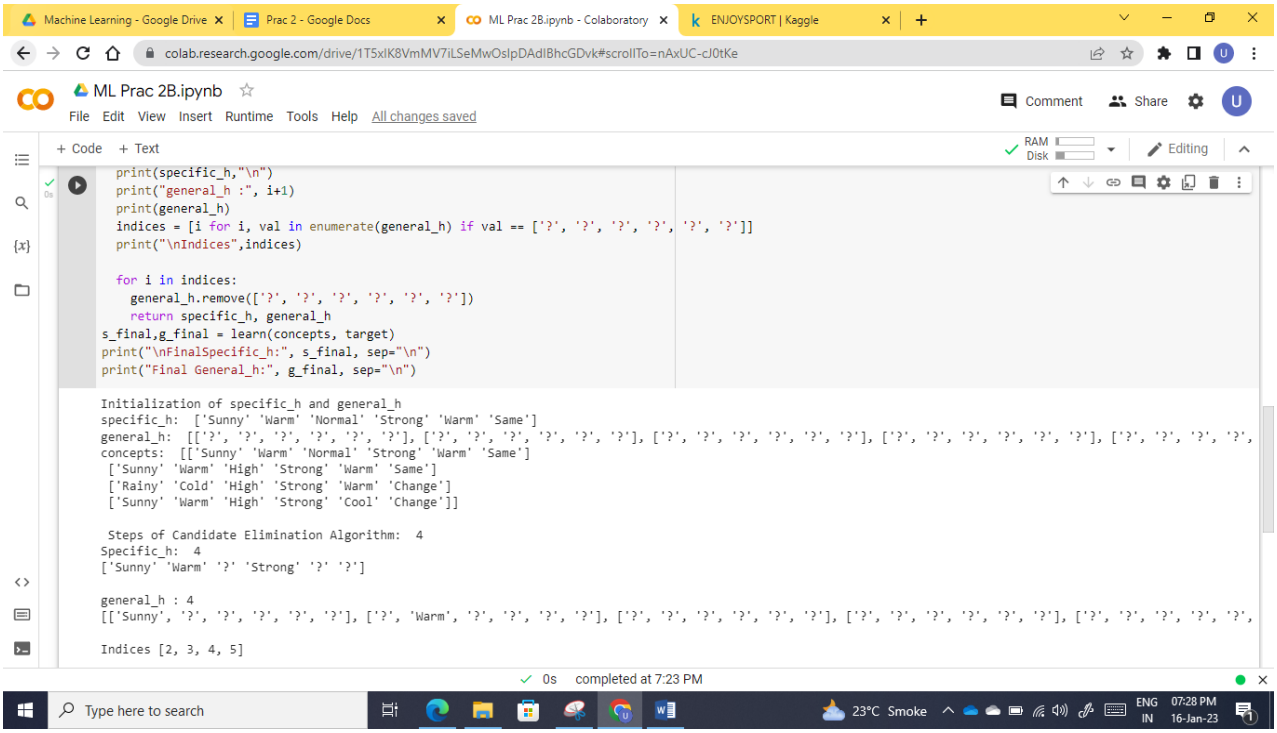
```
import pandas as pd
import numpy as np
data = pd.read_csv("/content/ENJOYSPORT.csv")
data

concepts = np.array(data.iloc[:,0:-1])
target = np.array(data.iloc[:,-1])
print(target)
print(concepts)

#Defining Model (Candidate Elimination algorithm concepts)
def learn(concepts, target):
    specific_h = concepts[0].copy()
    print("Initialization of specific_h and general_h")
    print("specific_h: ",specific_h)
    general_h = [["?" for i in range(len(specific_h))] for i in range(len(specific_h))]
    print("general_h: ",general_h)
    print("concepts: ",concepts)

    for i, h in enumerate(concepts):
        if target[i] == 1:
            for x in range(len(specific_h)):
                if h[x] != specific_h[x]:
                    specific_h[x] = '?'
                    general_h[x][x] = '?'
        if target[i] == 0:
            for x in range(len(specific_h)):
```

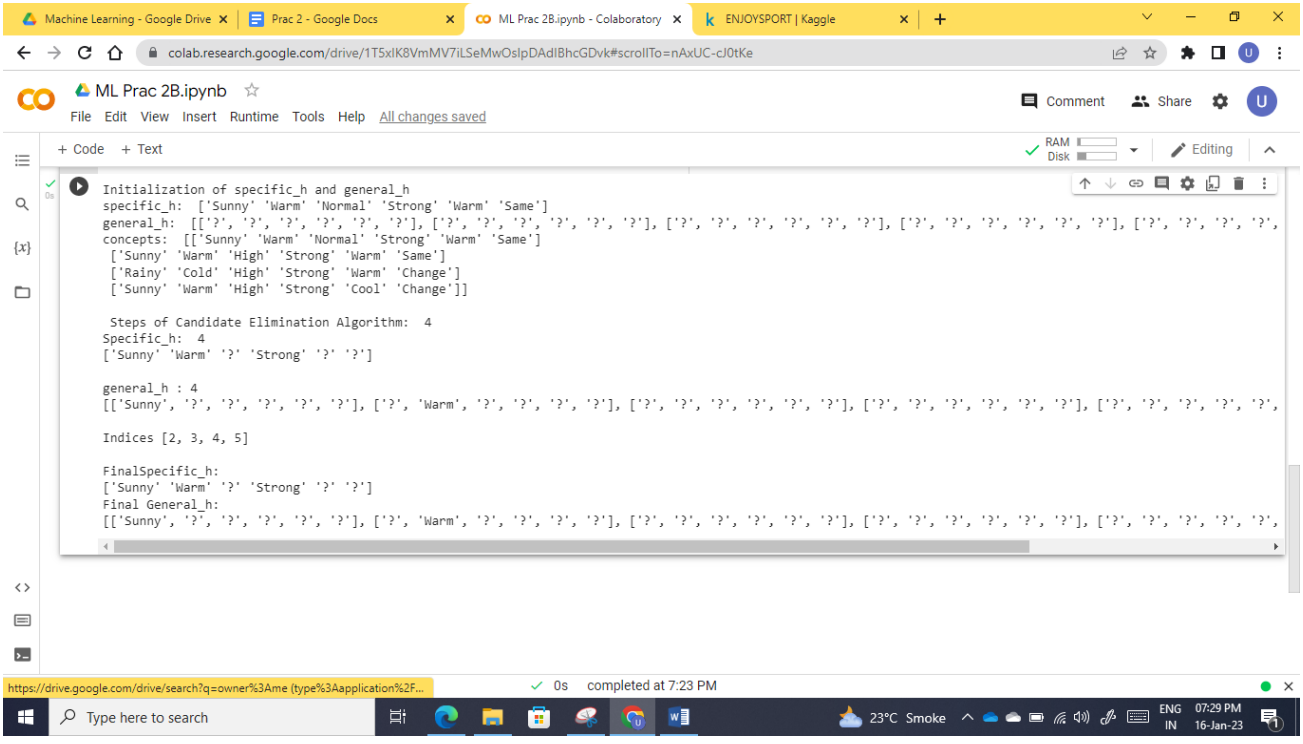
```
                    if h[x] != specific_h[x]:
                        general_h[x][x] = specific_h[x]
                    else:
                        general_h[x][x] = '?'

        print("\n Steps of Candidate Elimination Algorithm: ",i+1)
        print("Specific_h: ",i+1)
        print(specific_h,"\n")
        print("general_h :", i+1)
        print(general_h)
        indices = [i for i, val in enumerate(general_h) if val == ['?', '?', '?', '?', '?', '?']]
        print("\nIndices",indices)

        for i in indices:
            general_h.remove(['?', '?', '?', '?', '?', '?'])
            return specific_h, general_h
s_final,g_final = learn(concepts, target)
print("\nFinalSpecific_h:", s_final, sep="\n")
print("Final General_h:", g_final, sep="\n")
```

Output :


Machine Learning - Google Drive
Prac 2 - Google Docs
ML Prac 2B.ipynb - Colaboratory
ENJOYSPORT | Kaggle
colab.research.google.com/drive/1T5xlK8VmMV7iLSeMwOsIpDAdIBhcGDvk#scrollTo=TtUT3va70JSO
ML Prac 2B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
import pandas as pd
import numpy as np
data = pd.read_csv("/content/ENJOYSPORT.csv")
data
Sky AirTemp Humidity Wind Water Forecast EnjoySport
0 Sunny Warm Normal Strong Warm Same 1
1 Sunny Warm High Strong Warm Same 1
2 Rainy Cold High Strong Warm Change 0
3 Sunny Warm High Strong Cool Change 1
[2] concepts = np.array(data.iloc[:,0:-1])
target = np.array(data.iloc[:,-1])
print(target)
print(concepts)
[1 1 0 1]
[['Sunny' 'Warm' 'Normal' 'Strong' 'Warm' 'Same']
['Sunny' 'Warm' 'High' 'Strong' 'Warm' 'Same']
['Rainy' 'Cold' 'High' 'Strong' 'Warm' 'Change']
['Sunny' 'Warm' 'High' 'Strong' 'Cool' 'Change']]
0s completed at 7:23 PM
Type here to search
23°C Smoke
ENG IN 07:27 PM 16-Jan-23


Machine Learning - Google Drive
Prac 2 - Google Docs
ML Prac 2B.ipynb - Colaboratory
ENJOYSPORT | Kaggle
colab.research.google.com/drive/1T5xlK8VmMV7iLSeMwOsIpDAdIBhcGDvk#scrollTo=nAxUC-cJ0tKe
ML Prac 2B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
#Defining Model (Candidate Elimination algorithm concepts)
def learn(concepts, target):
specific_h = concepts[0].copy()
print("Initialization of specific_h and general_h")
print("specific_h: ",specific_h)
general_h = [["?" for i in range(len(specific_h))] for i in range(len(specific_h))]
print("general_h: ",general_h)
print("concepts: ",concepts)
for i, h in enumerate(concepts):
if target[i] == 1:
for x in range(len(specific_h)):
#print("h[x]",h[x])
if h[x] != specific_h[x]:
specific_h[x] = '?'
general_h[x][x] = '?'
if target[i] == 0:
for x in range(len(specific_h)):
if h[x] != specific_h[x]:
general_h[x][x] = specific_h[x]
else:
general_h[x][x] = '?'
print("\n Steps of Candidate Elimination Algorithm: ",i+1)
print("Specific_h: ",i+1)
print(specific_h,"\n")
print("general_h :", i+1)
0s completed at 7:23 PM
Type here to search
23°C Smoke
ENG IN 07:28 PM 16-Jan-23


Machine Learning - Google Drive
Prac 2 - Google Docs
ML Prac 2B.ipynb - Colaboratory
ENJOYSPORT | Kaggle
colab.research.google.com/drive/1T5xIK8VmMV7iLSeMwOsIpDAdIBhcGDvk#scrollTo=nAxUC-cJ0tKe
ML Prac 2B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
print(specific_h,"\n")
print("general_h :", i+1)
print(general_h)
indices = [i for i, val in enumerate(general_h) if val == ['?', '?', '?', '?', '?', '?']]
print("\nIndices",indices)
for i in indices:
general_h.remove(['?', '?', '?', '?', '?', '?'])
return specific_h, general_h
s_final,g_final = learn(concepts, target)
print("\nFinalSpecific_h:", s_final, sep="\n")
print("Final General_h:", g_final, sep="\n")
Initialization of specific_h and general_h
specific_h: ['Sunny' 'Warm' 'Normal' 'Strong' 'Warm' 'Same']
general_h: [['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?']]
concepts: [['Sunny' 'Warm' 'Normal' 'Strong' 'Warm' 'Same']
['Sunny' 'Warm' 'High' 'Strong' 'Warm' 'Same']
['Rainy' 'Cold' 'High' 'Strong' 'Warm' 'Change']
['Sunny' 'Warm' 'High' 'Strong' 'Cool' 'Change']]
Steps of Candidate Elimination Algorithm: 4
Specific_h: 4
['Sunny' 'Warm' '?' 'Strong' '?' '?']
general_h : 4
[['Sunny', '?', '?', '?', '?', '?'], ['?', 'Warm', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?']]
Indices [2, 3, 4, 5]
0s completed at 7:23 PM
Type here to search
23°C Smoke
ENG IN
07:28 PM
16-Jan-23


Machine Learning - Google Drive
Prac 2 - Google Docs
ML Prac 2B.ipynb - Colaboratory
ENJOYSPORT | Kaggle
colab.research.google.com/drive/1T5xIK8VmMV7iLSeMwOsIpDAdIBhcGDvk#scrollTo=nAxUC-cJ0tKe
ML Prac 2B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
Initialization of specific_h and general_h
specific_h: ['Sunny' 'Warm' 'Normal' 'Strong' 'Warm' 'Same']
general_h: [['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?']]
concepts: [['Sunny' 'Warm' 'Normal' 'Strong' 'Warm' 'Same']
['Sunny' 'Warm' 'High' 'Strong' 'Warm' 'Same']
['Rainy' 'Cold' 'High' 'Strong' 'Warm' 'Change']
['Sunny' 'Warm' 'High' 'Strong' 'Cool' 'Change']]
Steps of Candidate Elimination Algorithm: 4
Specific_h: 4
['Sunny' 'Warm' '?' 'Strong' '?' '?']
general_h : 4
[['Sunny', '?', '?', '?', '?', '?'], ['?', 'Warm', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?']]
Indices [2, 3, 4, 5]
FinalSpecific_h:
['Sunny' 'Warm' '?' 'Strong' '?' '?']
Final General_h:
[['Sunny', '?', '?', '?', '?', '?'], ['?', 'Warm', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?']]
https://drive.google.com/drive/search?q=owner%3Ame (type%3Aapplication%2F...
0s completed at 7:23 PM
Type here to search
23°C Smoke
ENG IN
07:29 PM
16-Jan-23

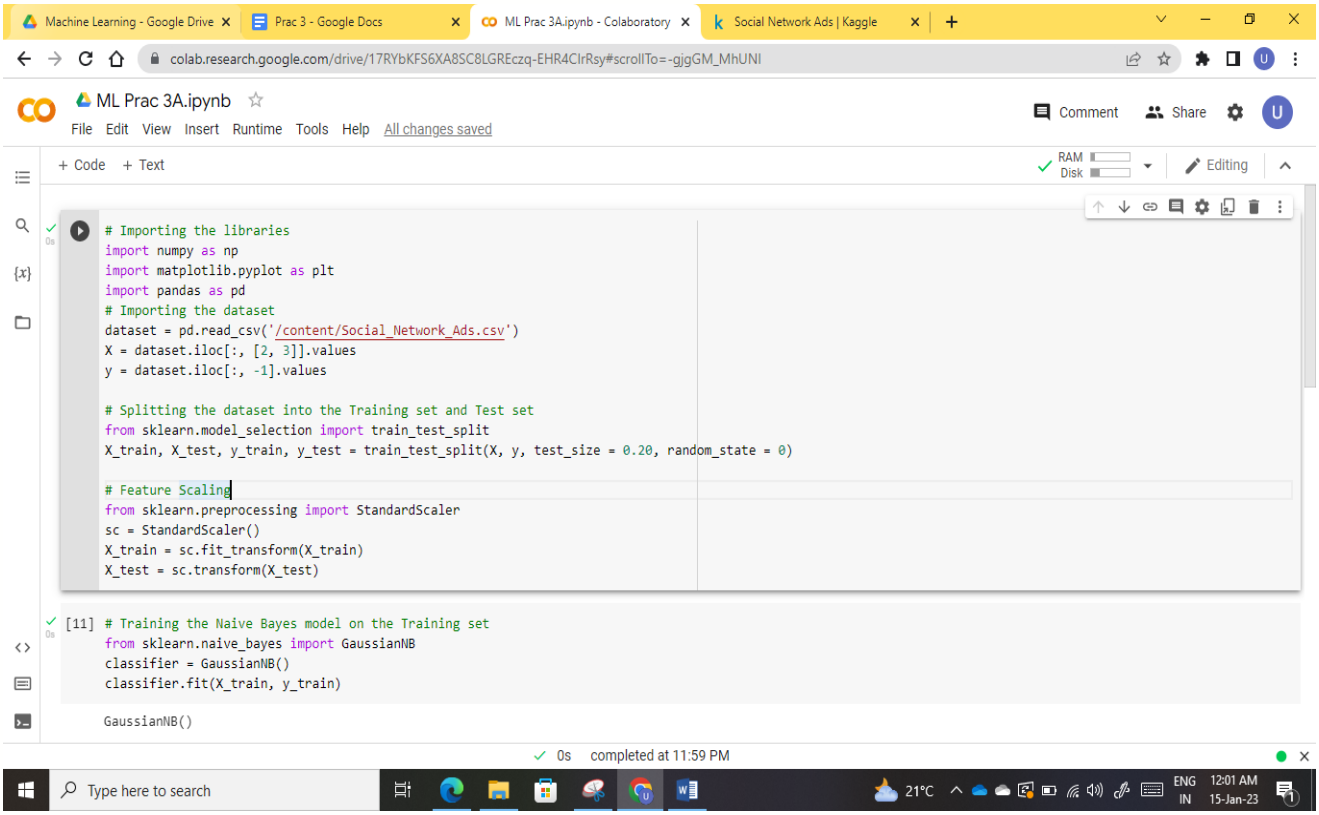
Write a program to implement the naïve Bayesian classifier for a sample training data set stored as a .CSV file. Compute the accuracy of the classifier, considering few test data sets

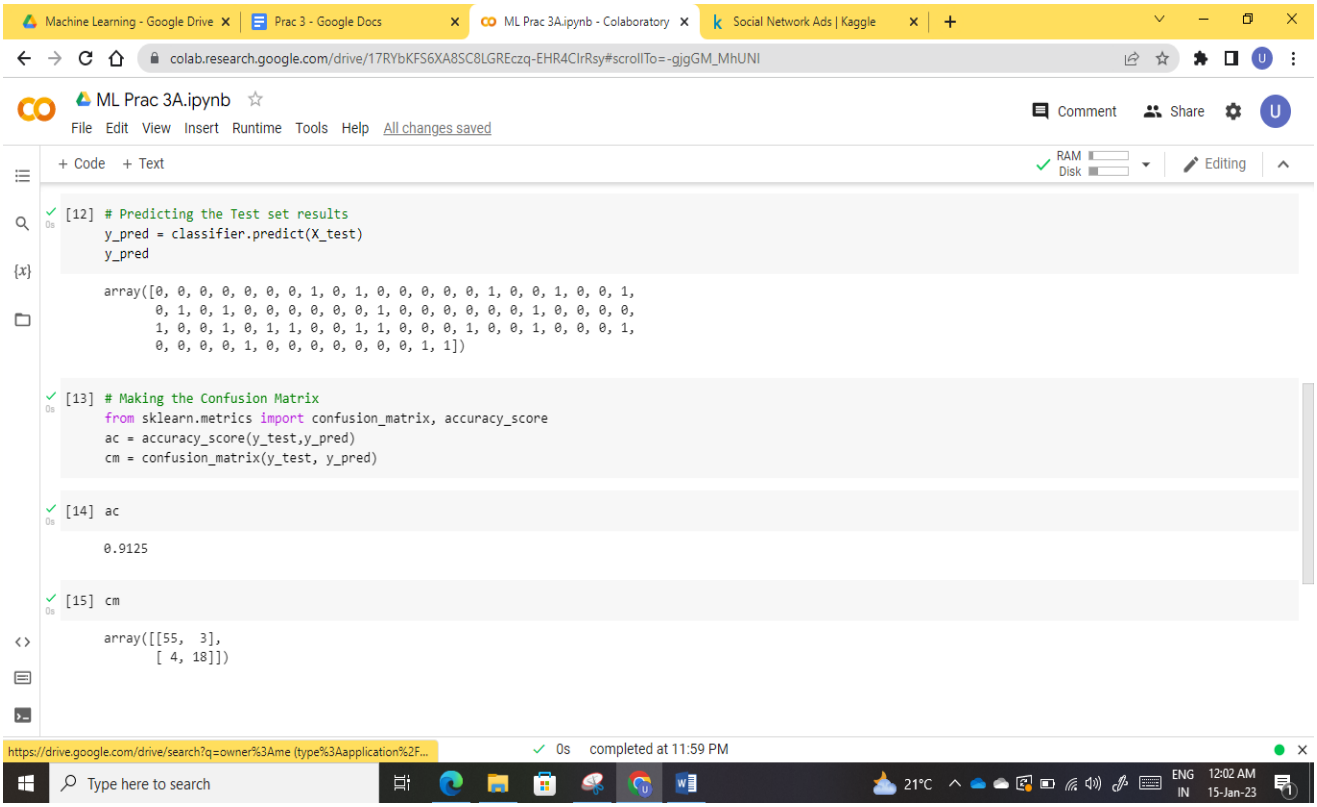
Code :

```
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
# Importing the dataset
dataset = pd.read_csv('/content/Social_Network_Ads.csv')
X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, -1].values

# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state = 0)

# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Training the Naive Bayes model on the Training set
from sklearn.naive_bayes import GaussianNB
classifier = GaussianNB()
classifier.fit(X_train, y_train)
# Predicting the Test set results
y_pred = classifier.predict(X_test)
y_pred
```

```
# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix, accuracy_score
ac = accuracy_score(y_test,y_pred)
cm = confusion_matrix(y_test, y_pred)
ac
cm
```

Output :




Machine Learning - Google Drive
Prac 3 - Google Docs
ML Prac 3A.ipynb - Colaboratory
Social Network Ads | Kaggle
colab.research.google.com/drive/17RYbKFS6XA8SC8LGREczq-EHR4ClrRsy#scrollTo=-gjgGM_MhUNI
ML Prac 3A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[12] # Predicting the Test set results
y_pred = classifier.predict(X_test)
y_pred
array([0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1,
0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1,
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1])
[13] # Making the Confusion Matrix
from sklearn.metrics import confusion_matrix, accuracy_score
ac = accuracy_score(y_test,y_pred)
cm = confusion_matrix(y_test, y_pred)
[14] ac
0.9125
[15] cm
array([[55, 3],
[4, 18]])
0s completed at 11:59 PM
Type here to search
21°C
ENG
IN
12:02 AM
15-Jan-23

Write a program to implement Decision Tree and Random forest with Prediction, Test Score and Confusion Matrix

Decision tree :

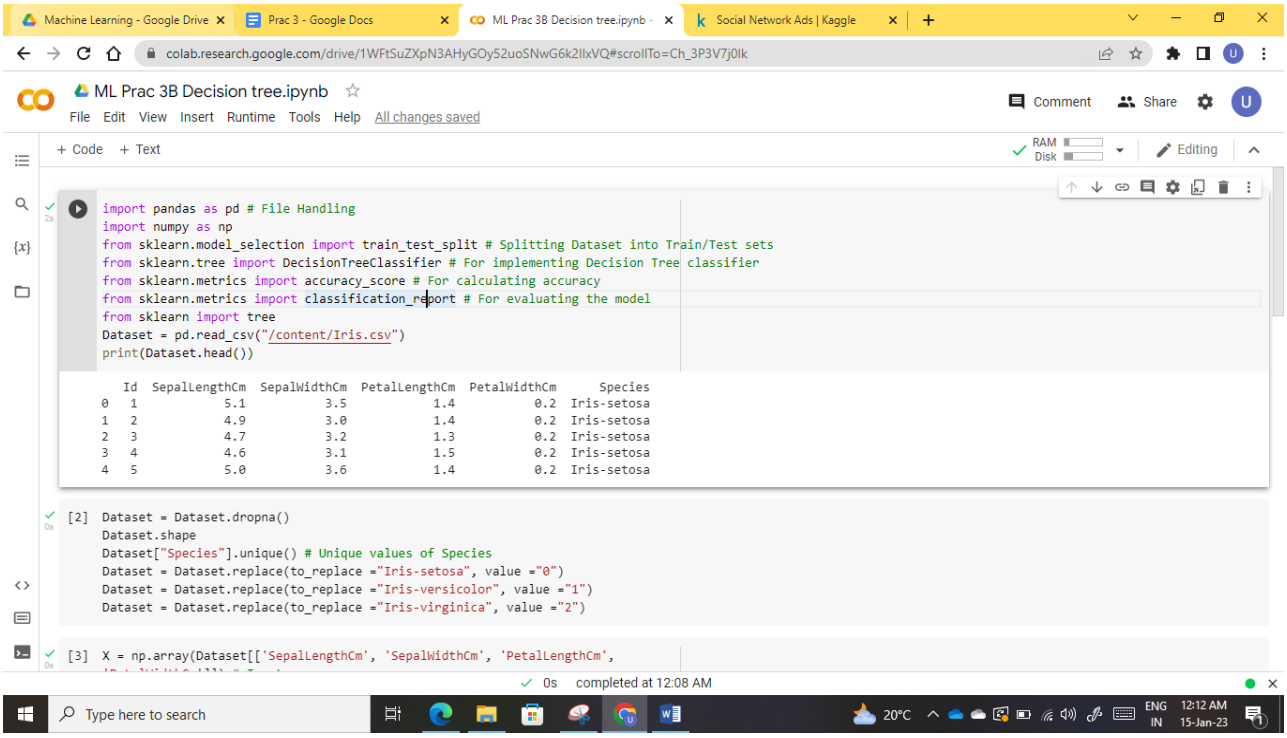
Code :

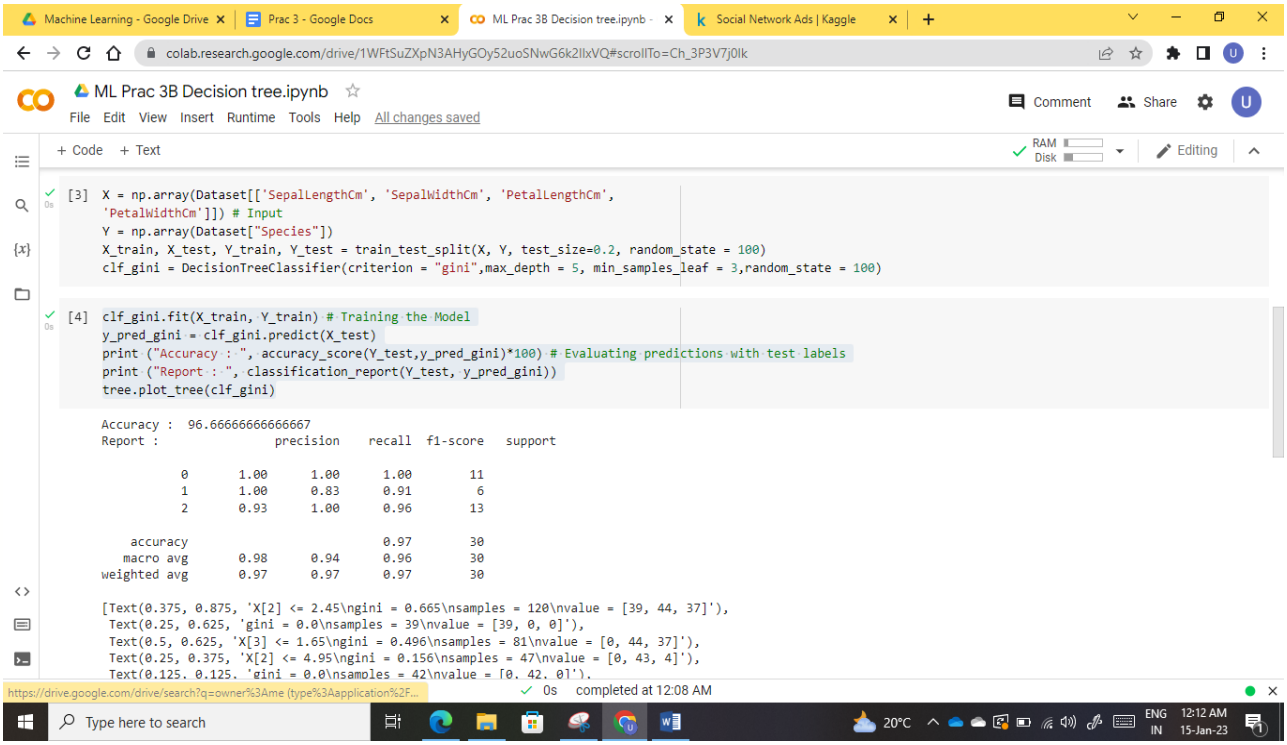
```
import pandas as pd      # File Handling
import numpy as np
from sklearn.model_selection import train_test_split       # Splitting Dataset into Train/Test sets
from sklearn.tree import DecisionTreeClassifier        # For implementing Decision Tree classifier
from sklearn.metrics import accuracy_score          # For calculating accuracy
from sklearn.metrics import classification_report          # For evaluating the model
from sklearn import tree
Dataset = pd.read_csv("/content/Iris.csv")
print(Dataset.head())

Dataset = Dataset.dropna()
Dataset.shape
Dataset["Species"].unique()       # Unique values of Species
Dataset = Dataset.replace(to_replace ="Iris-setosa", value ="0")
Dataset = Dataset.replace(to_replace ="Iris-versicolor", value ="1")
Dataset = Dataset.replace(to_replace ="Iris-virginica", value ="2")

X = np.array(Dataset[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']])
Y = np.array(Dataset["Species"])
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state = 100)
clf_gini = DecisionTreeClassifier(criterion = "gini",max_depth = 5, min_samples_leaf = 3,
                                  random_state = 100)

clf_gini.fit(X_train, Y_train)        # Training the Model
y_pred_gini = clf_gini.predict(X_test)
print ("Accuracy : ", accuracy_score(Y_test,y_pred_gini)*100)
```

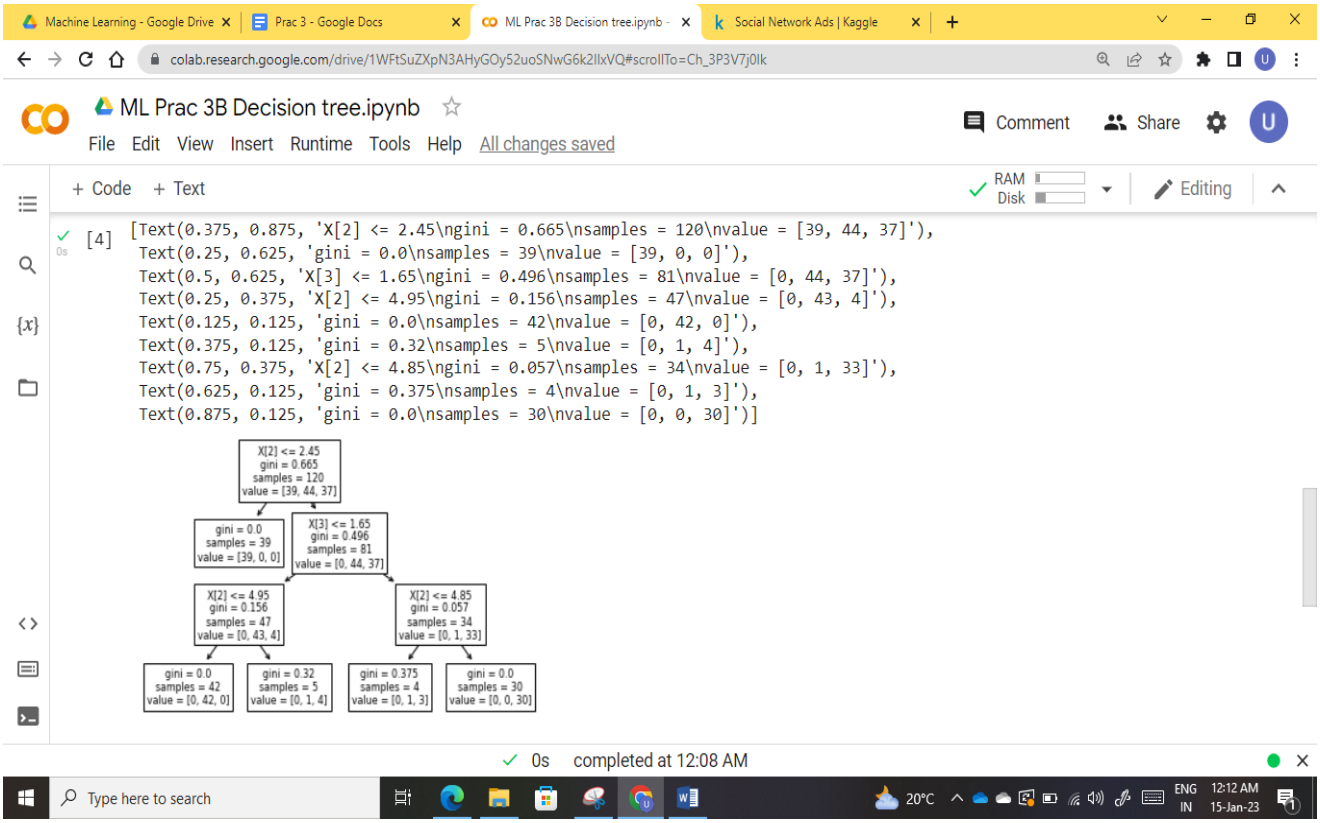
print ("Report : ", classification_report(Y_test, y_pred_gini))

tree.plot_tree(clf_gini)

Output :






Machine Learning - Google Drive
Prac 3 - Google Docs
ML Prac 3B Decision tree.ipynb
Social Network Ads | Kaggle
colab.research.google.com/drive/1WFtSuZXpN3AHyGOy52uoSNwG6k2IlxVQ#scrollTo=Ch_3P3V7j0Ik
ML Prac 3B Decision tree.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[4] [Text(0.375, 0.875, 'X[2] <= 2.45\ngini = 0.665\nsamples = 120\nvalue = [39, 44, 37]'),
Text(0.25, 0.625, 'gini = 0.0\nsamples = 39\nvalue = [39, 0, 0]'),
Text(0.5, 0.625, 'X[3] <= 1.65\ngini = 0.496\nsamples = 81\nvalue = [0, 44, 37]'),
Text(0.25, 0.375, 'X[2] <= 4.95\ngini = 0.156\nsamples = 47\nvalue = [0, 43, 4]'),
Text(0.125, 0.125, 'gini = 0.0\nsamples = 42\nvalue = [0, 42, 0]'),
Text(0.375, 0.125, 'gini = 0.32\nsamples = 5\nvalue = [0, 1, 4]'),
Text(0.75, 0.375, 'X[2] <= 4.85\ngini = 0.057\nsamples = 34\nvalue = [0, 1, 33]'),
Text(0.625, 0.125, 'gini = 0.375\nsamples = 4\nvalue = [0, 1, 3]'),
Text(0.875, 0.125, 'gini = 0.0\nsamples = 30\nvalue = [0, 0, 30]')]
X[2] <= 2.45 gini = 0.665 samples = 120 value = [39, 44, 37]
gini = 0.0 samples = 39 value = [39, 0, 0]
X[3] <= 1.65 gini = 0.496 samples = 81 value = [0, 44, 37]
X[2] <= 4.95 gini = 0.156 samples = 47 value = [0, 43, 4]
X[2] <= 4.85 gini = 0.057 samples = 34 value = [0, 1, 33]
gini = 0.0 samples = 42 value = [0, 42, 0]
gini = 0.32 samples = 5 value = [0, 1, 4]
gini = 0.375 samples = 4 value = [0, 1, 3]
gini = 0.0 samples = 30 value = [0, 0, 30]
0s completed at 12:08 AM
Type here to search
20°C
ENG IN
12:12 AM 15-Jan-23

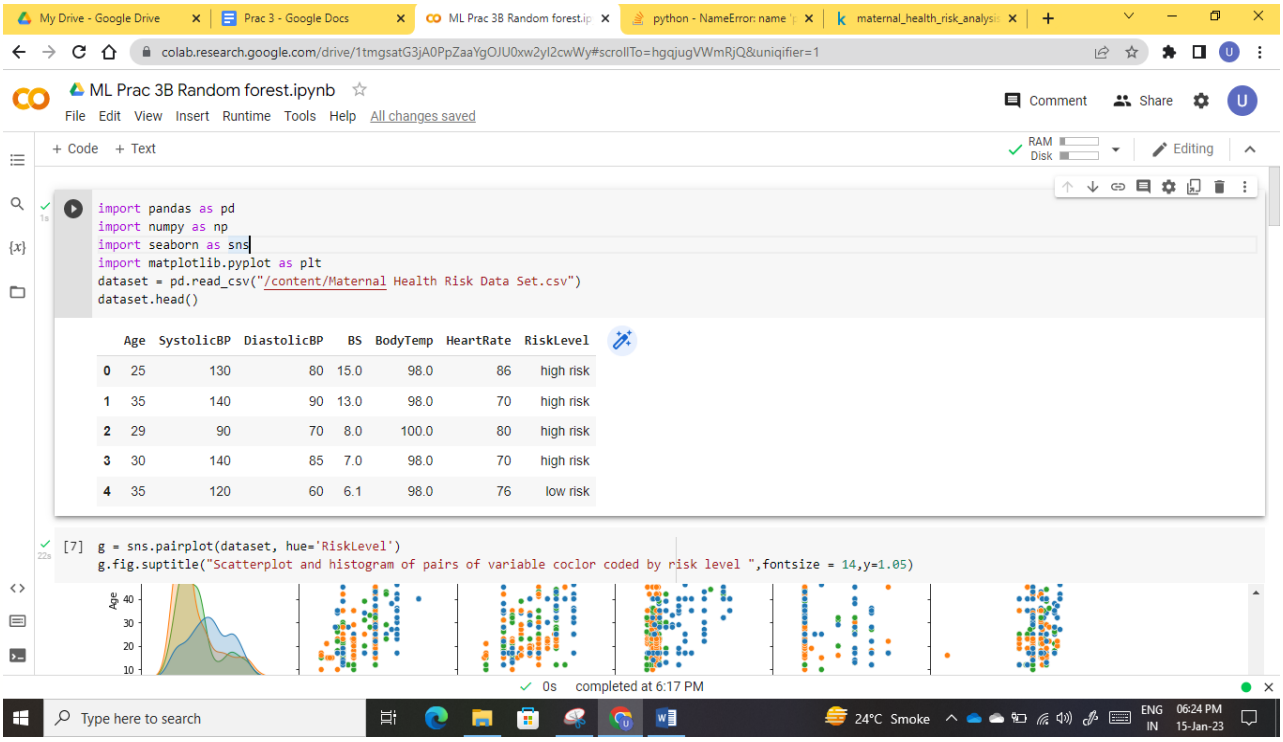
Random forest :

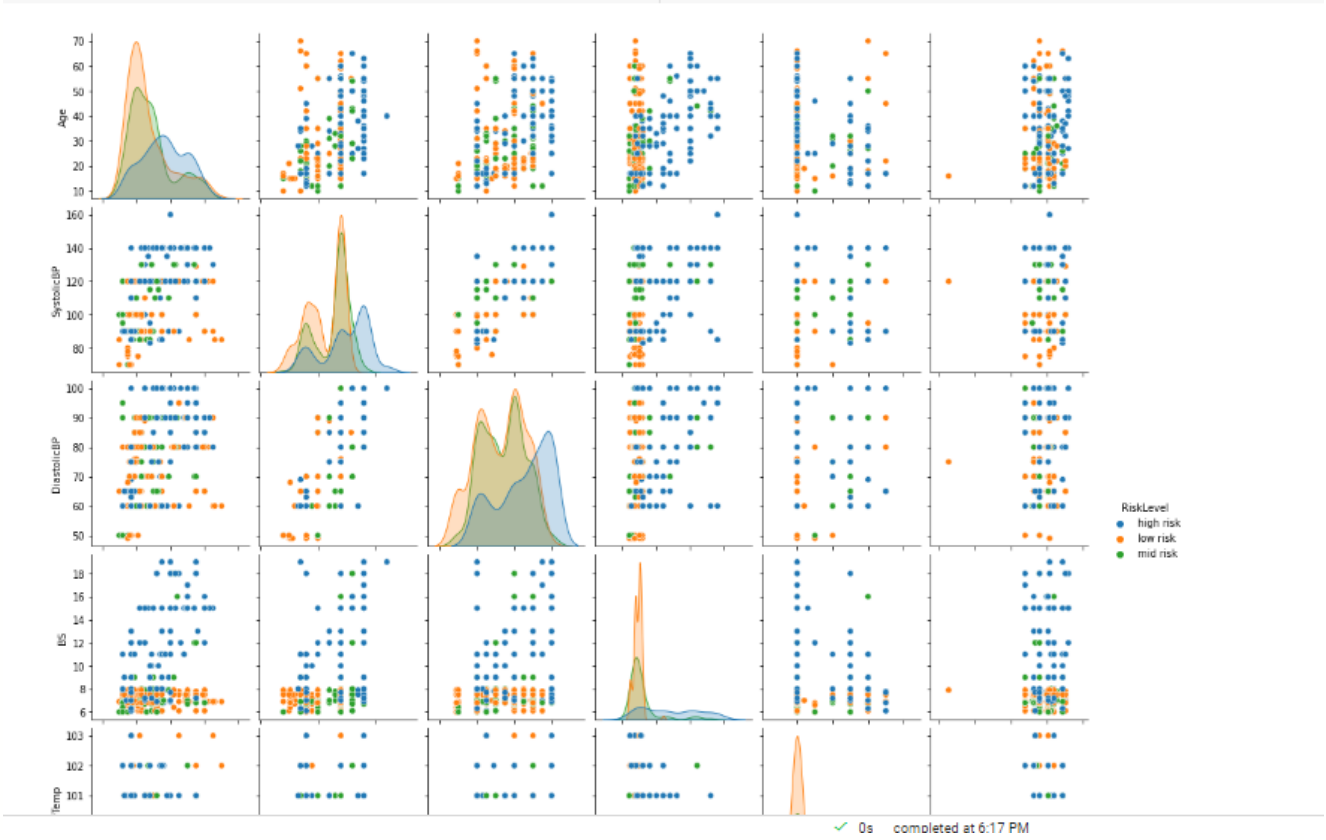
Code :

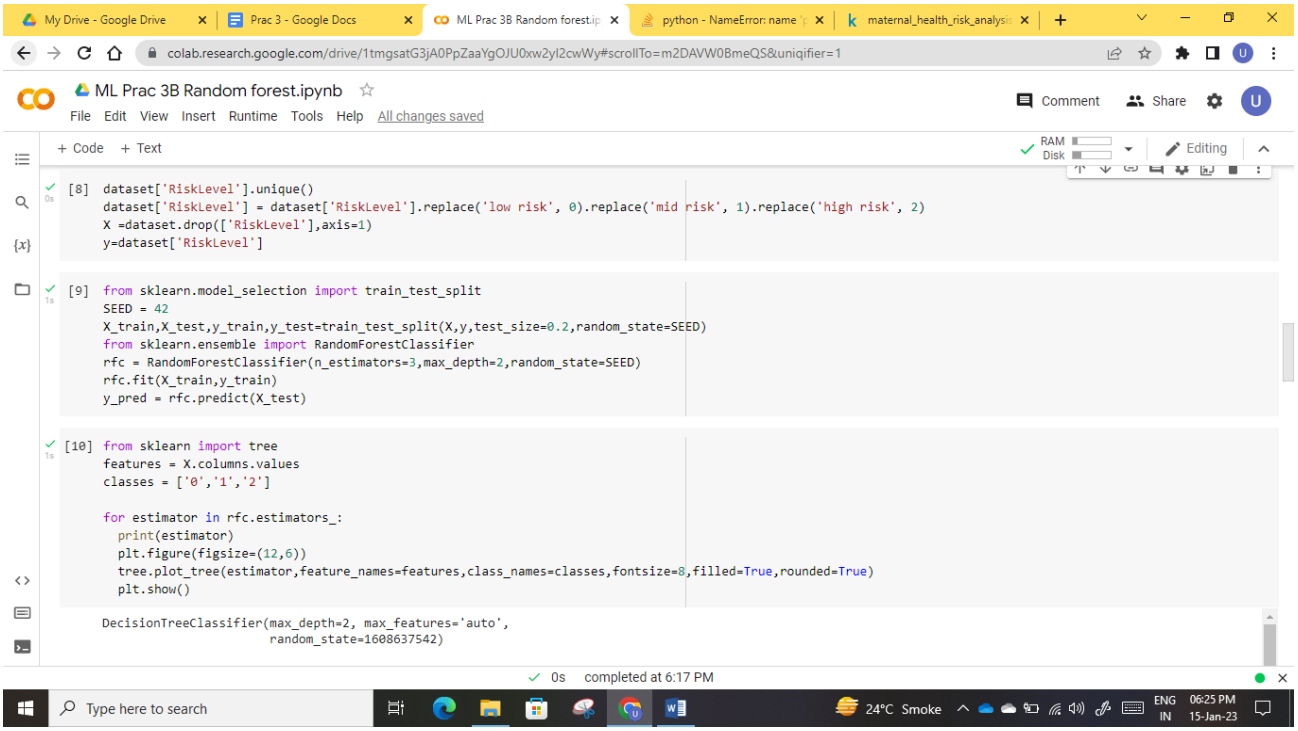
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
dataset = pd.read_csv("/content/Maternal Health Risk Data Set.csv")
dataset.head()

g = sns.pairplot(dataset, hue='RiskLevel')
g.fig.suptitle("Scatterplot and histogram of pairs of variable coclor coded by risk level ",fontsize
                = 14,y=1.05)

dataset['RiskLevel'].unique()
dataset['RiskLevel'] = dataset['RiskLevel'].replace('low risk', 0).replace('mid risk', 1). replace
                        ('high risk', 2)
X =dataset.drop(['RiskLevel'],axis=1)
y=dataset['RiskLevel']

from sklearn.model_selection import train_test_split
SEED = 42
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=SEED)
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(n_estimators=3,max_depth=2,random_state=SEED)
rfc.fit(X_train,y_train)
y_pred = rfc.predict(X_test)

from sklearn import tree
features = X.columns.values
classes = ['0','1','2']
for estimator in rfc.estimators_:
        print(estimator)
```

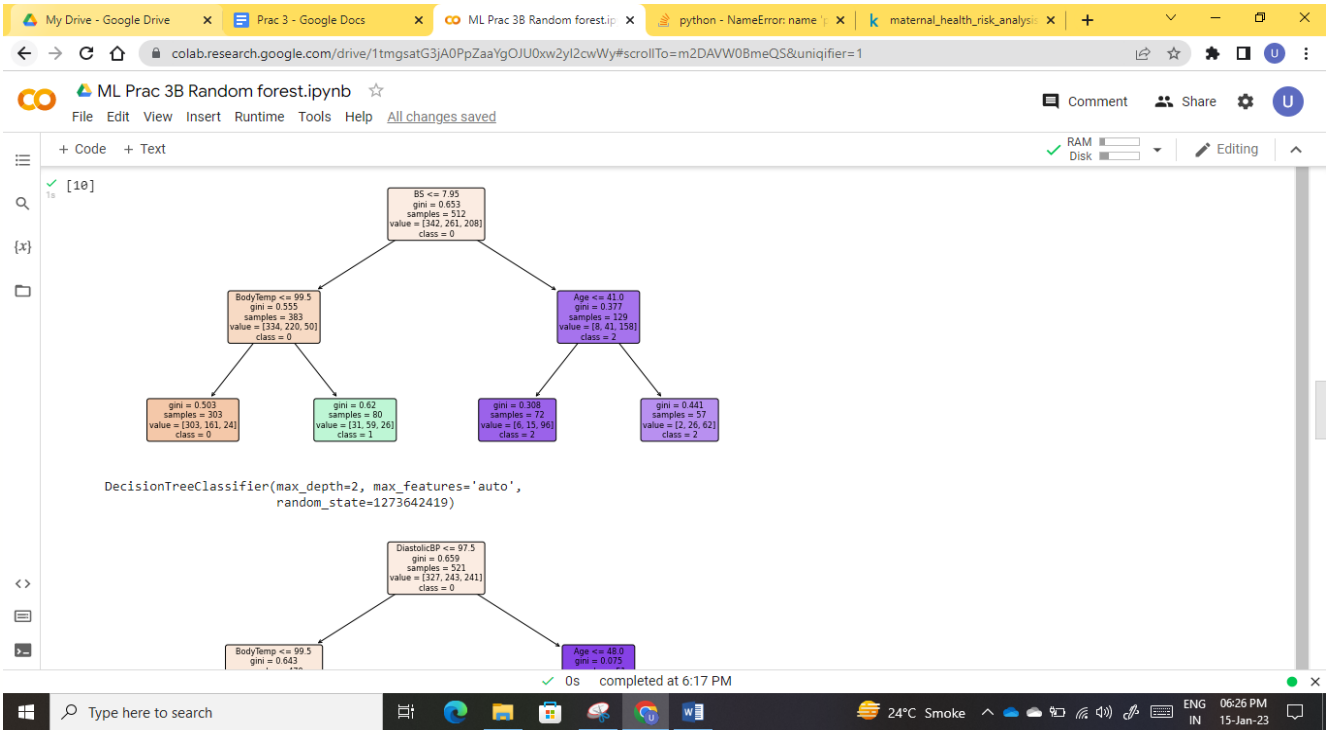
```
        plt.figure(figsize=(12,6))
        tree.plot_tree(estimator,feature_names=features,class_names=classes,fontsize=8,filled=
                True,rounded=True)
        plt.show()

# Organizing feature names and importances in a DataFrame
features_df = pd.DataFrame({'features': rfc.feature_names_in_, 'importances': rfc.feature _
                        importances_ })
# Sorting data from highest to lowest
features_df_sorted = features_df.sort_values(by='importances', ascending=False)
# Barplot of the result without borders and axis lines
g = sns.barplot(data=features_df_sorted, x='importances', y ='features', palette="rocket")
sns.despine(bottom = True, left = True)
g.set_title('Feature importances')
g.set(xlabel=None)
g.set(ylabel=None)
g.set(xticks=[])

rfc_ = RandomForestClassifier(n_estimators=900, max_depth=7,random_state=SEED)
rfc_.fit(X_train, y_train)
y_pred = rfc_.predict(X_test)

from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
cm_= confusion_matrix(y_test, y_pred)
sns.heatmap(cm_, annot=True, fmt='d').set_title('Maternal risks confusion matrix (0 = low risk, 1
                        = medium risk, 2 = high risk) for 900 trees with 8 levels')
print(classification_report(y_test,y_pred))
```

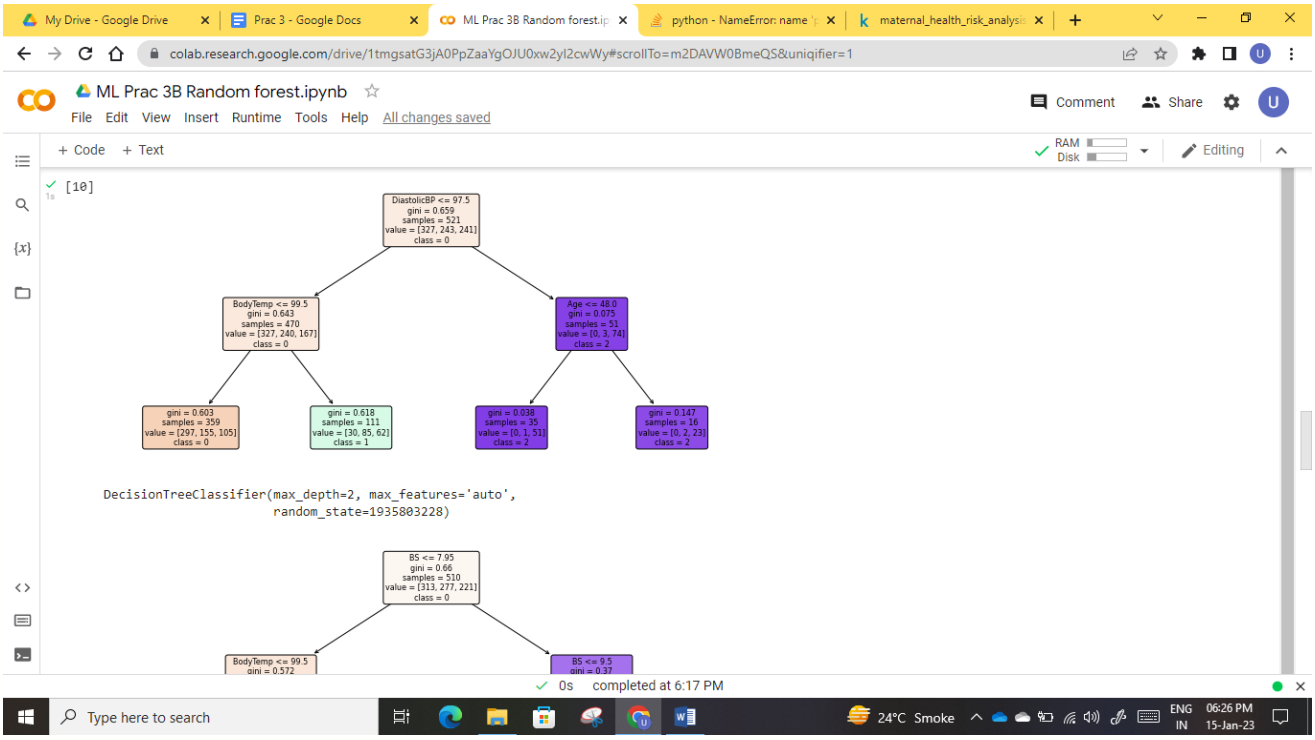
Output :

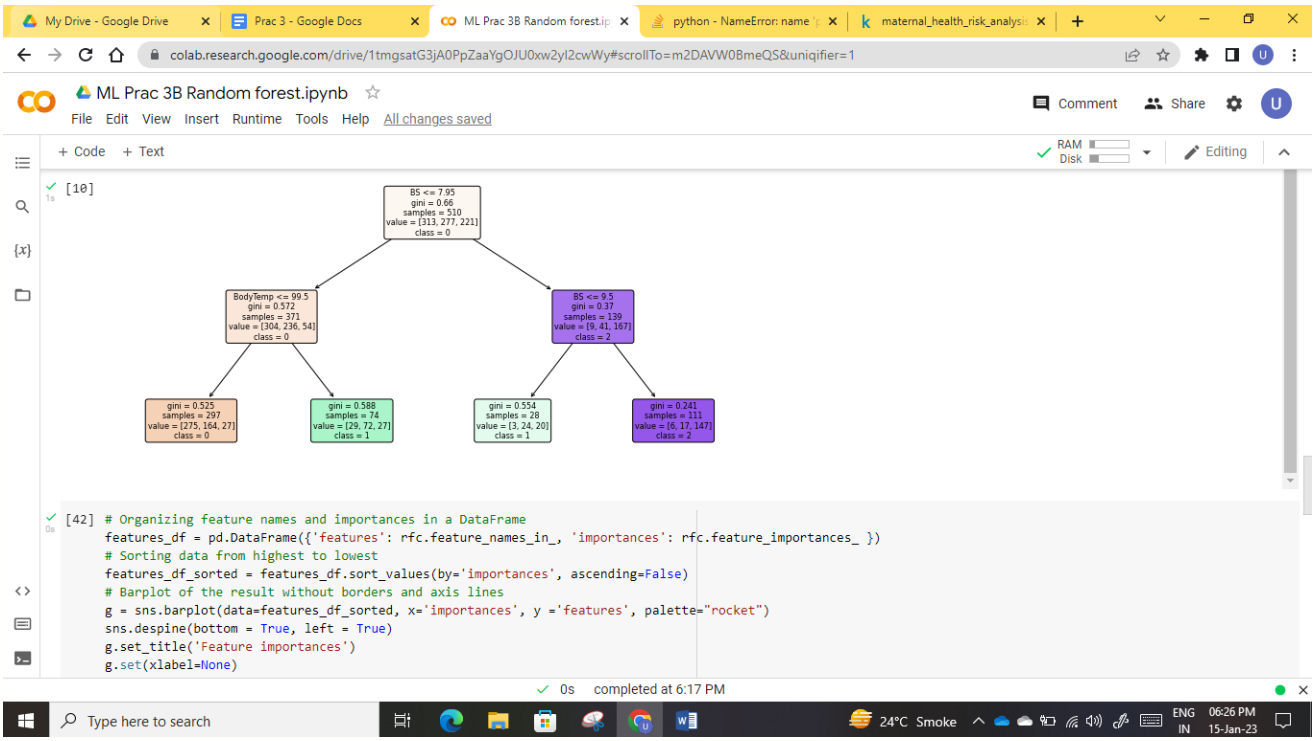

My Drive - Google Drive
Prac 3 - Google Docs
ML Prac 3B Random forest.ipynb
python - NameError: name
maternal_health_risk_analysis
colab.research.google.com/drive/1tmgsatG3jA0PpZaaYgOJU0xw2yl2cwWy#scrollTo=hgqjugVWmRjQ&uniqifier=1
ML Prac 3B Random forest.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
dataset = pd.read_csv("/content/Maternal Health Risk Data Set.csv")
dataset.head()
Age SystolicBP DiastolicBP BS BodyTemp HeartRate RiskLevel
0 25 130 80 15.0 98.0 86 high risk
1 35 140 90 13.0 98.0 70 high risk
2 29 90 70 8.0 100.0 80 high risk
3 30 140 85 7.0 98.0 70 high risk
4 35 120 60 6.1 98.0 76 low risk
[7] g = sns.pairplot(dataset, hue='RiskLevel')
g.fig.suptitle("Scatterplot and histogram of pairs of variable coclor coded by risk level ",fontsize = 14,y=1.05)
Age
0s completed at 6:17 PM
Type here to search
24°C Smoke
ENG IN
06:24 PM 15-Jan-23

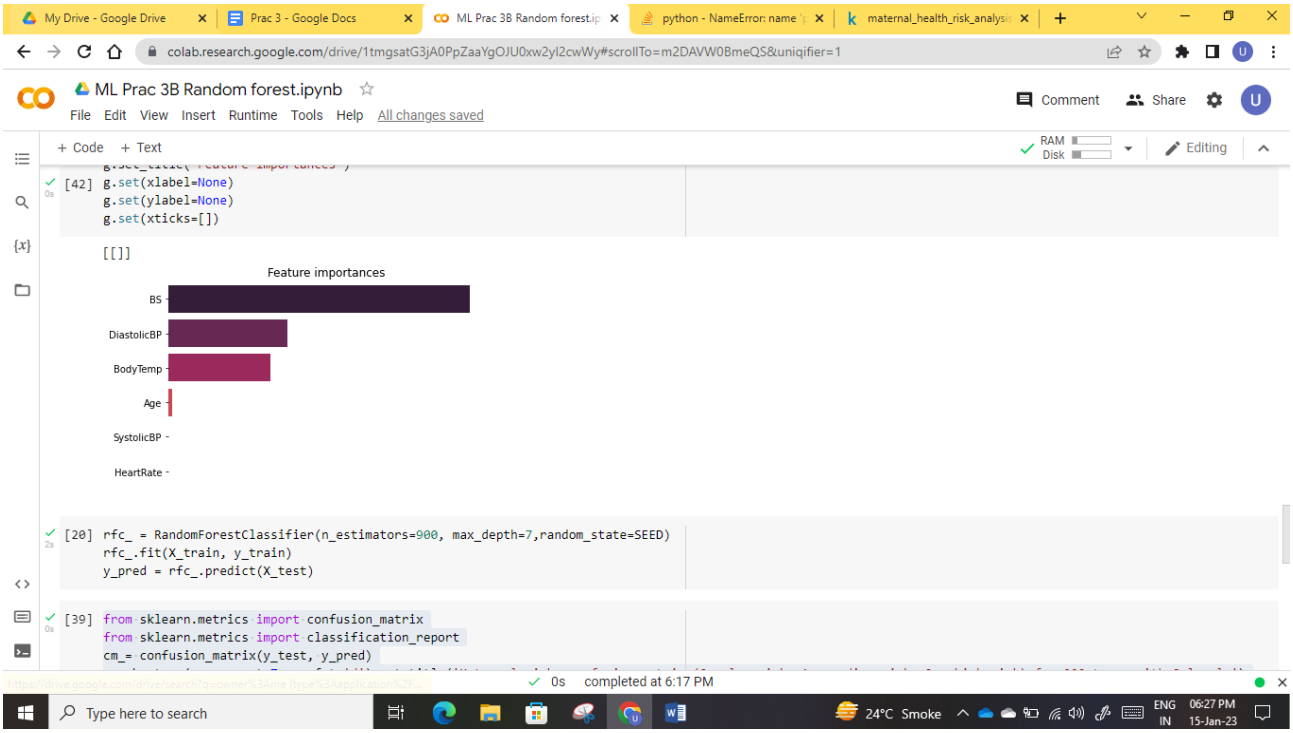

Age
SystolicBP
DiastolicBP
BS
Temp
RiskLevel
high risk
low risk
mid risk
0s completed at 6:17 PM

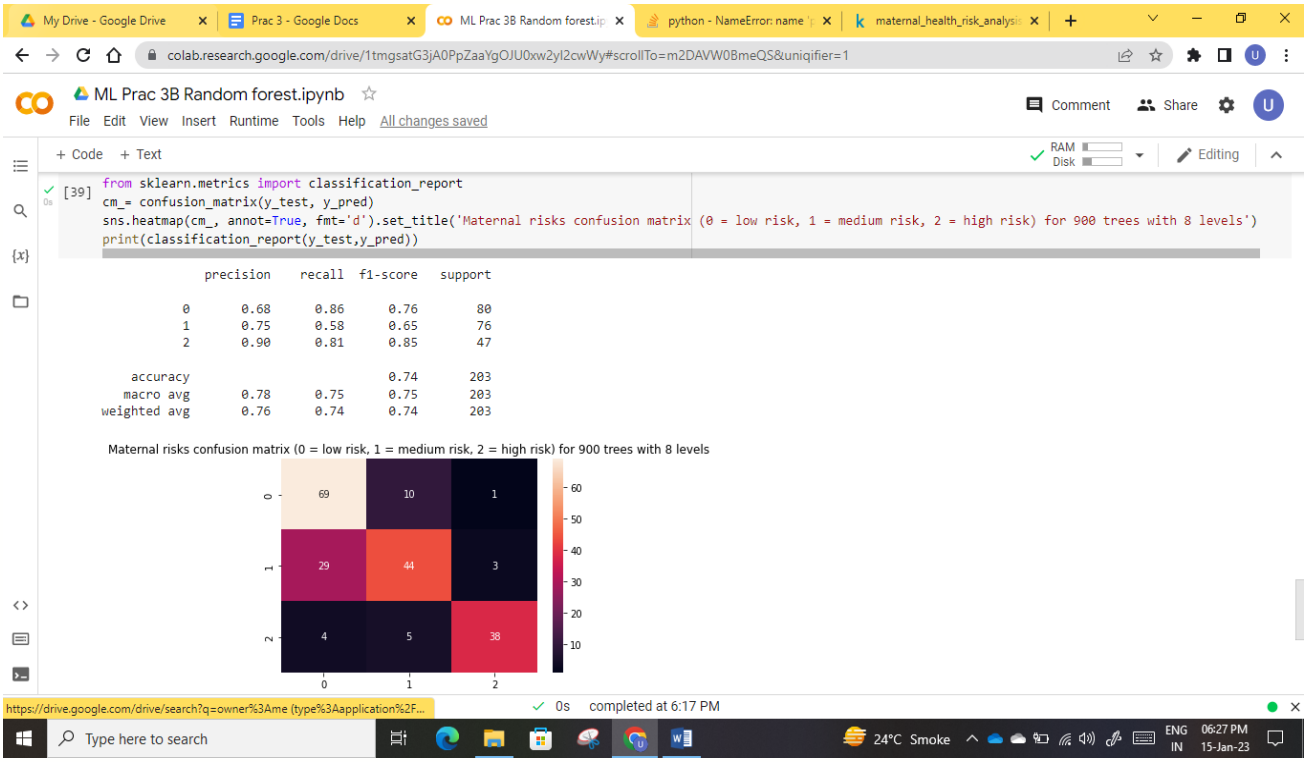

ML Prac 3B Random forest.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
[8] dataset['RiskLevel'].unique()
dataset['RiskLevel'] = dataset['RiskLevel'].replace('low risk', 0).replace('mid risk', 1).replace('high risk', 2)
X =dataset.drop(['RiskLevel'],axis=1)
y=dataset['RiskLevel']
[9] from sklearn.model_selection import train_test_split
SEED = 42
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=SEED)
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(n_estimators=3,max_depth=2,random_state=SEED)
rfc.fit(X_train,y_train)
y_pred = rfc.predict(X_test)
[10] from sklearn import tree
features = X.columns.values
classes = ['0','1','2']
for estimator in rfc.estimators_:
print(estimator)
plt.figure(figsize=(12,6))
tree.plot_tree(estimator,feature_names=features,class_names=classes,fontsize=8,filled=True,rounded=True)
plt.show()
DecisionTreeClassifier(max_depth=2, max_features='auto',
random_state=1608637542)
0s completed at 6:17 PM


ML Prac 3B Random forest.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
[10]
BS <= 7.95
gini = 0.653
samples = 512
value = [342, 261, 208]
class = 0
BodyTemp <= 99.5
gini = 0.555
samples = 383
value = [334, 220, 50]
class = 0
Age <= 41.0
gini = 0.377
samples = 129
value = [8, 41, 158]
class = 2
gini = 0.503
samples = 303
value = [303, 161, 24]
class = 0
gini = 0.62
samples = 80
value = [31, 59, 26]
class = 1
gini = 0.308
samples = 72
value = [6, 15, 96]
class = 2
gini = 0.441
samples = 57
value = [2, 26, 62]
class = 2
DecisionTreeClassifier(max_depth=2, max_features='auto',
random_state=1273642419)
DiastolicBP <= 97.5
gini = 0.659
samples = 521
value = [327, 243, 241]
class = 0
BodyTemp <= 99.5
gini = 0.643
Age <= 48.0
gini = 0.075
0s completed at 6:17 PM


My Drive - Google Drive
Prac 3 - Google Docs
ML Prac 3B Random forest.ip
python - NameError: name
maternal_health_risk_analysis
colab.research.google.com/drive/1tmgsatG3jA0PpZaaYgOJU0xw2yI2cwWy#scrollTo=m2DAVW0BmeQS&uniqifier=1
ML Prac 3B Random forest.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[10]
DiastolicBP <= 97.5
gini = 0.659
samples = 521
value = [327, 243, 241]
class = 0
BodyTemp <= 99.5
gini = 0.643
samples = 470
value = [327, 240, 167]
class = 0
Age <= 48.0
gini = 0.075
samples = 51
value = [0, 3, 74]
class = 2
gini = 0.603
samples = 359
value = [297, 155, 105]
class = 0
gini = 0.618
samples = 111
value = [30, 85, 62]
class = 1
gini = 0.038
samples = 35
value = [0, 1, 51]
class = 2
gini = 0.147
samples = 16
value = [0, 2, 23]
class = 2
DecisionTreeClassifier(max_depth=2, max_features='auto',
random_state=1935803228)
BS <= 7.95
gini = 0.66
samples = 510
value = [313, 277, 221]
class = 0
BodyTemp <= 99.5
gini = 0.572
BS <= 9.5
gini = 0.37
0s completed at 6:17 PM
Type here to search
24°C Smoke
ENG
IN
06:26 PM
15-Jan-23


My Drive - Google Drive
Prac 3 - Google Docs
ML Prac 3B Random forest.ip
python - NameError: name
maternal_health_risk_analysis
colab.research.google.com/drive/1tmgsatG3jA0PpZaaYgOJU0xw2yI2cwWy#scrollTo=m2DAVW0BmeQS&uniqifier=1
ML Prac 3B Random forest.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[10]
BS <= 7.95
gini = 0.66
samples = 510
value = [313, 277, 221]
class = 0
BodyTemp <= 99.5
gini = 0.572
samples = 371
value = [304, 236, 54]
class = 0
BS <= 9.5
gini = 0.37
samples = 139
value = [9, 41, 167]
class = 2
gini = 0.525
samples = 297
value = [275, 164, 27]
class = 0
gini = 0.588
samples = 74
value = [29, 72, 27]
class = 1
gini = 0.554
samples = 28
value = [3, 24, 20]
class = 1
gini = 0.241
samples = 111
value = [6, 17, 147]
class = 2
[42] # Organizing feature names and importances in a DataFrame
features_df = pd.DataFrame({'features': rfc.feature_names_in_, 'importances': rfc.feature_importances_ })
# Sorting data from highest to lowest
features_df_sorted = features_df.sort_values(by='importances', ascending=False)
# Barplot of the result without borders and axis lines
g = sns.barplot(data=features_df_sorted, x='importances', y ='features', palette="rocket")
sns.despine(bottom = True, left = True)
g.set_title('Feature importances')
g.set(xlabel=None)
0s completed at 6:17 PM
Type here to search
24°C Smoke
ENG
IN
06:26 PM
15-Jan-23


My Drive - Google Drive
Prac 3 - Google Docs
ML Prac 3B Random forest.ip
python - NameError: name
maternal_health_risk_analysis
colab.research.google.com/drive/1tmgsatG3jA0PpZaaYgOJU0xw2yl2cwWy#scrollTo=m2DAVW0BmeQS&uniqifier=1
ML Prac 3B Random forest.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[42] g.set(xlabel=None)
g.set(ylabel=None)
g.set(xticks=[])
[[]]
Feature importances
BS
DiastolicBP
BodyTemp
Age
SystolicBP
HeartRate
[20] rfc_ = RandomForestClassifier(n_estimators=900, max_depth=7,random_state=SEED)
rfc_.fit(X_train, y_train)
y_pred = rfc_.predict(X_test)
[39] from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
cm_= confusion_matrix(y_test, y_pred)
0s completed at 6:17 PM
Type here to search
24°C Smoke
ENG IN
06:27 PM
15-Jan-23


My Drive - Google Drive
Prac 3 - Google Docs
ML Prac 3B Random forest.ip
python - NameError: name
maternal_health_risk_analysis
colab.research.google.com/drive/1tmgsatG3jA0PpZaaYgOJU0xw2yl2cwWy#scrollTo=m2DAVW0BmeQS&uniqifier=1
ML Prac 3B Random forest.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[39] from sklearn.metrics import classification_report
cm_= confusion_matrix(y_test, y_pred)
sns.heatmap(cm_, annot=True, fmt='d').set_title('Maternal risks confusion matrix (0 = low risk, 1 = medium risk, 2 = high risk) for 900 trees with 8 levels')
print(classification_report(y_test,y_pred))
precision recall f1-score support
0 0.68 0.86 0.76 80
1 0.75 0.58 0.65 76
2 0.90 0.81 0.85 47
accuracy 0.74 203
macro avg 0.78 0.75 0.75 203
weighted avg 0.76 0.74 0.74 203
Maternal risks confusion matrix (0 = low risk, 1 = medium risk, 2 = high risk) for 900 trees with 8 levels
69
10
1
29
44
3
4
5
38
0s completed at 6:17 PM
Type here to search
24°C Smoke
ENG IN
06:27 PM
15-Jan-23

For a given set of training data examples stored in a .CSV file implement Least Square Regression algorithm

Code :

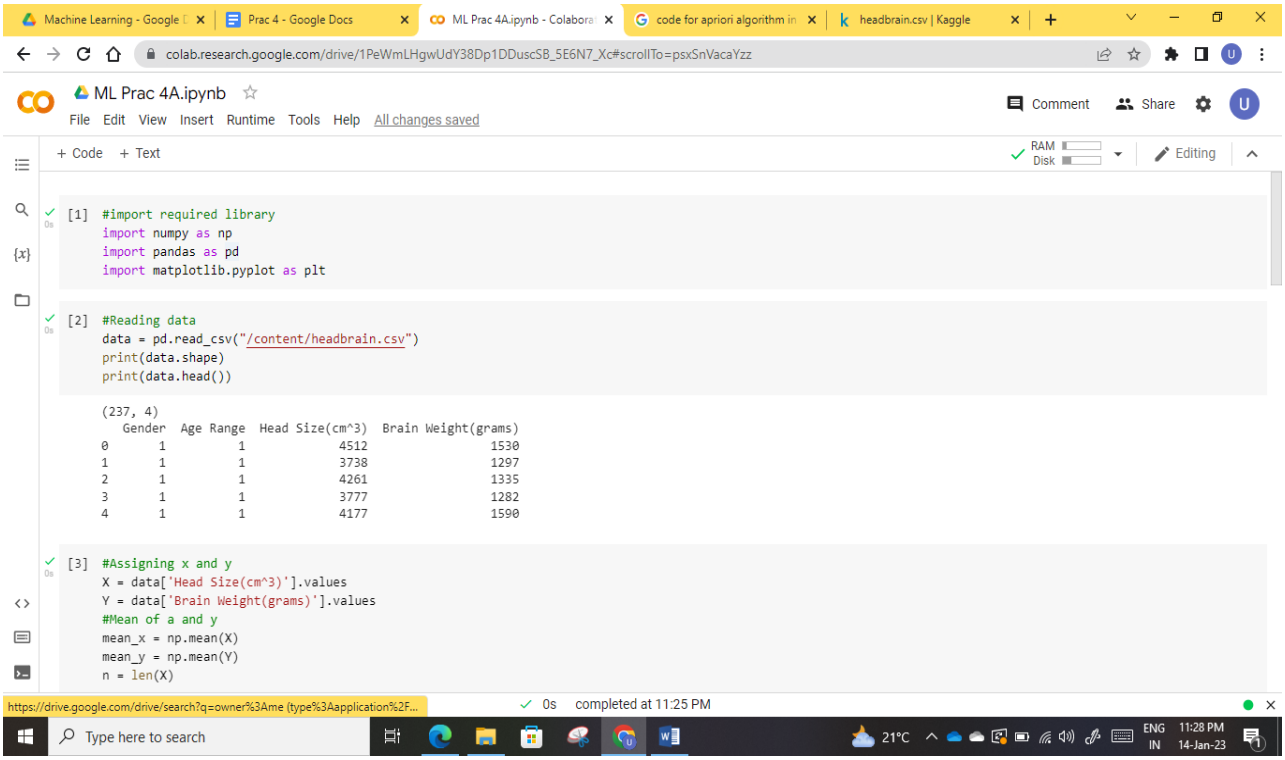
```
#import required library
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

#Reading data
data = pd.read_csv("/content/headbrain.csv")
print(data.shape)
print(data.head())

#Assigning x and y
X = data['Head Size(cm^3)'].values
Y = data['Brain Weight(grams)'].values
#Mean of a and y
mean_x = np.mean(X)
mean_y = np.mean(Y)
n = len(X)

# Using the formula to calculate 'm' and 'c'
numer = 0
denom = 0
for i in range(n):
        numer += (X[i] - mean_x) * (Y[i] - mean_y)
        denom += (X[i] - mean_x) ** 2
        m = numer / denom
        c = mean_y - (m * mean_x)

# Printing coefficients
```

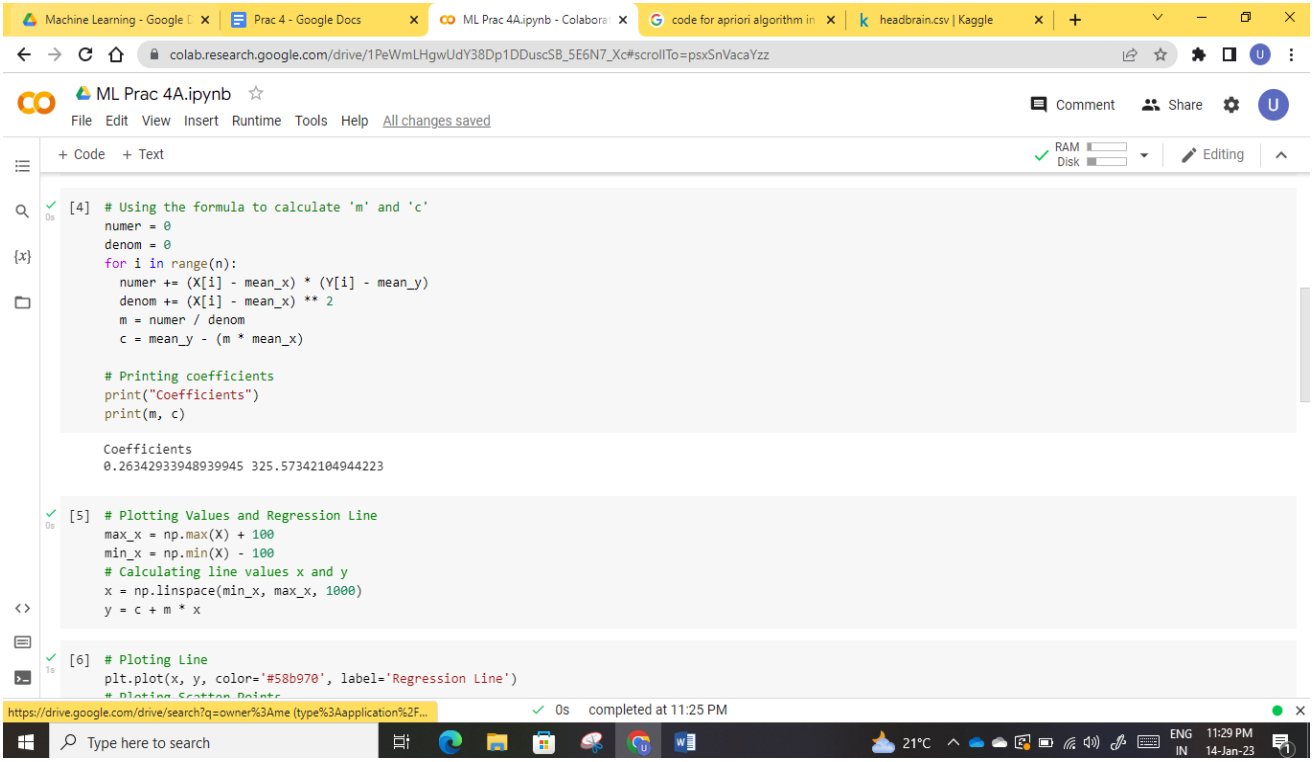
```
print("Coefficients")
print(m, c)
# Plotting Values and Regression Line
max_x = np.max(X) + 100
min_x = np.min(X) - 100
# Calculating line values x and y
x = np.linspace(min_x, max_x, 1000)
y = c + m * x

# Ploting Line
plt.plot(x, y, color='#58b970', label='Regression Line')
# Ploting Scatter Points
plt.scatter(X, Y, c='#ef5423', label='Scatter Plot')
plt.xlabel('Head Size in cm3')
plt.ylabel('Brain Weight in grams')
plt.legend()
plt.show()

# Calculating Root Mean Squares Error
rmse = 0
for i in range(n):
        y_pred = c + m * X[i]
        rmse += (Y[i] - y_pred) ** 2
rmse = np.sqrt(rmse/n)
print("RMSE")
print(rmse)
# Calculating R2 Score
ss_tot = 0
ss_res = 0
for i in range(n):
        y_pred = c + m * X[i]
        ss_tot += (Y[i] - mean_y) ** 2
        ss_res += (Y[i] - y_pred) ** 2
r2 = 1 - (ss_res/ss_tot)
```

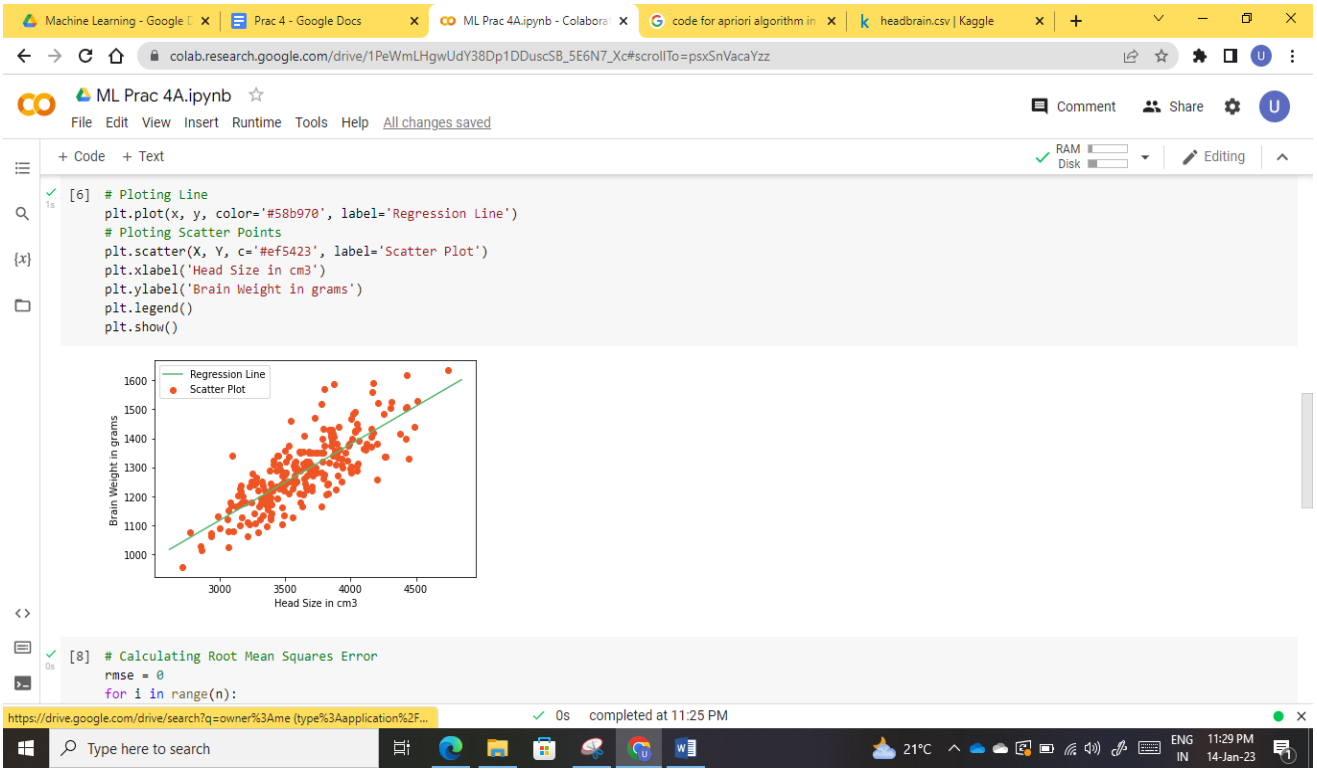
print("R2 Score")

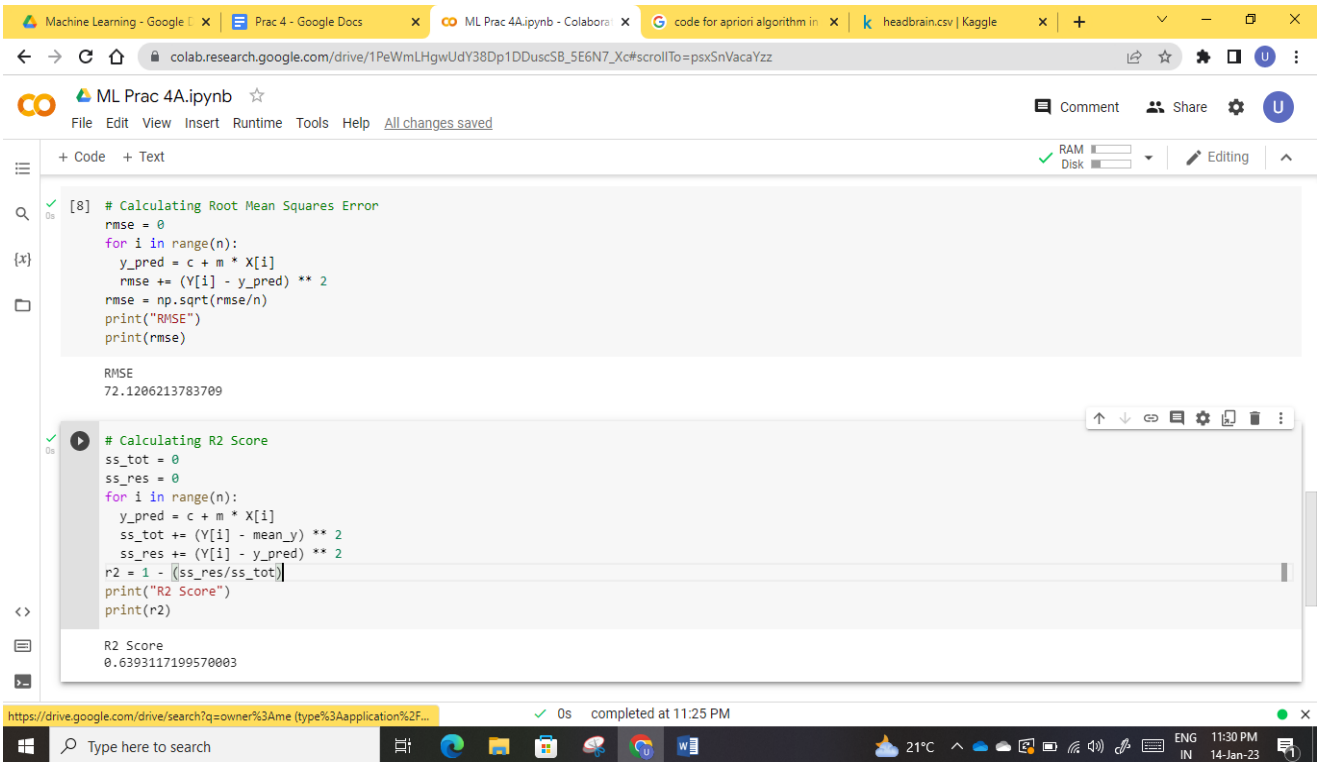
print(r2)

Output :






Machine Learning - Google
Prac 4 - Google Docs
ML Prac 4A.ipynb - Colabora
code for apriori algorithm in
headbrain.csv | Kaggle
colab.research.google.com/drive/1PeWmLHgwUdY38Dp1DDuscSB_5E6N7_Xc#scrollTo=psxSnVacaYzz
ML Prac 4A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[6] # Ploting Line
plt.plot(x, y, color='#58b970', label='Regression Line')
# Ploting Scatter Points
plt.scatter(X, Y, c='#ef5423', label='Scatter Plot')
plt.xlabel('Head Size in cm3')
plt.ylabel('Brain Weight in grams')
plt.legend()
plt.show()
Regression Line
Scatter Plot
Brain Weight in grams
Head Size in cm3
[8] # Calculating Root Mean Squares Error
rmse = 0
for i in range(n):
0s completed at 11:25 PM
Type here to search
21°C
ENG IN
11:29 PM
14-Jan-23


Machine Learning - Google
Prac 4 - Google Docs
ML Prac 4A.ipynb - Colabora
code for apriori algorithm in
headbrain.csv | Kaggle
colab.research.google.com/drive/1PeWmLHgwUdY38Dp1DDuscSB_5E6N7_Xc#scrollTo=psxSnVacaYzz
ML Prac 4A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[8] # Calculating Root Mean Squares Error
rmse = 0
for i in range(n):
y_pred = c + m * X[i]
rmse += (Y[i] - y_pred) ** 2
rmse = np.sqrt(rmse/n)
print("RMSE")
print(rmse)
RMSE
72.1206213783709
# Calculating R2 Score
ss_tot = 0
ss_res = 0
for i in range(n):
y_pred = c + m * X[i]
ss_tot += (Y[i] - mean_y) ** 2
ss_res += (Y[i] - y_pred) ** 2
r2 = 1 - (ss_res/ss_tot)
print("R2 Score")
print(r2)
R2 Score
0.6393117199570003
0s completed at 11:25 PM
Type here to search
21°C
ENG IN
11:30 PM
14-Jan-23

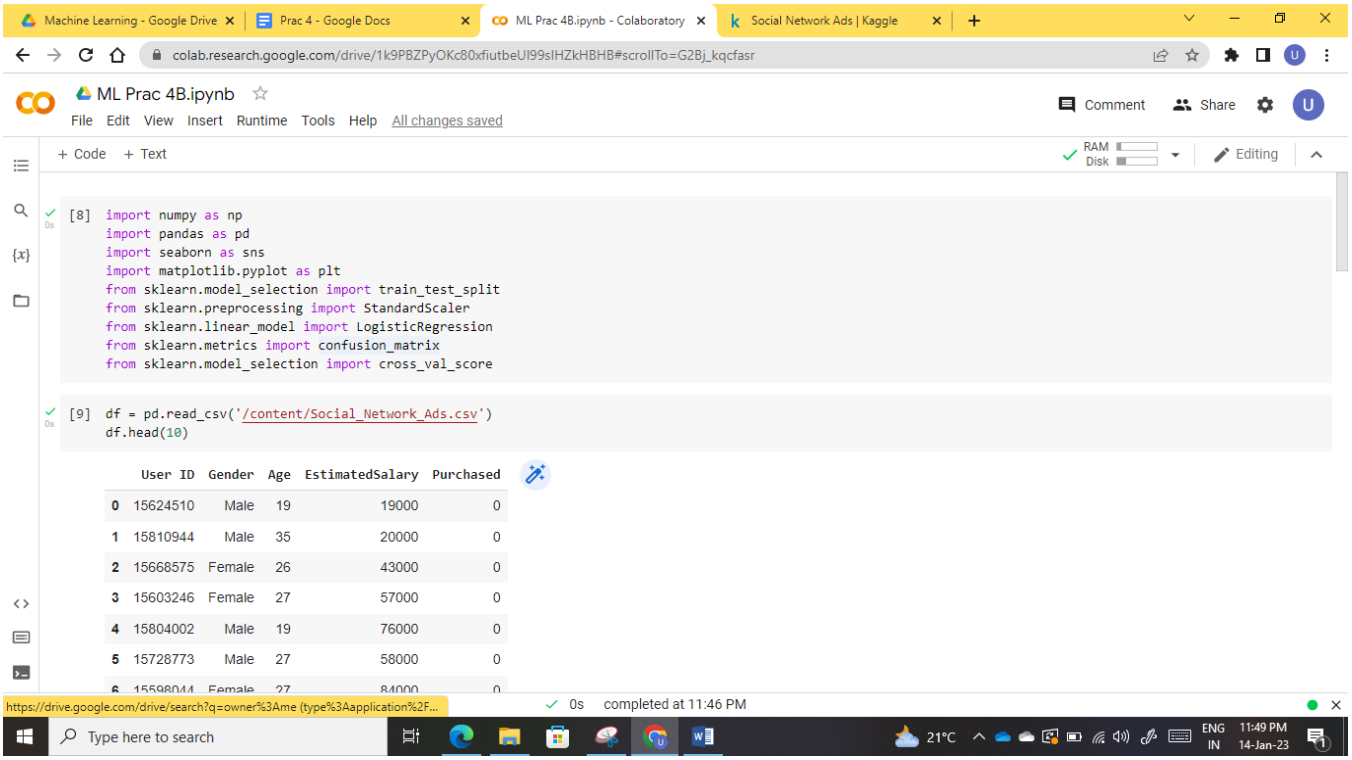
For a given set of training data examples stored in a .CSV file implement Logistic Regression algorithm

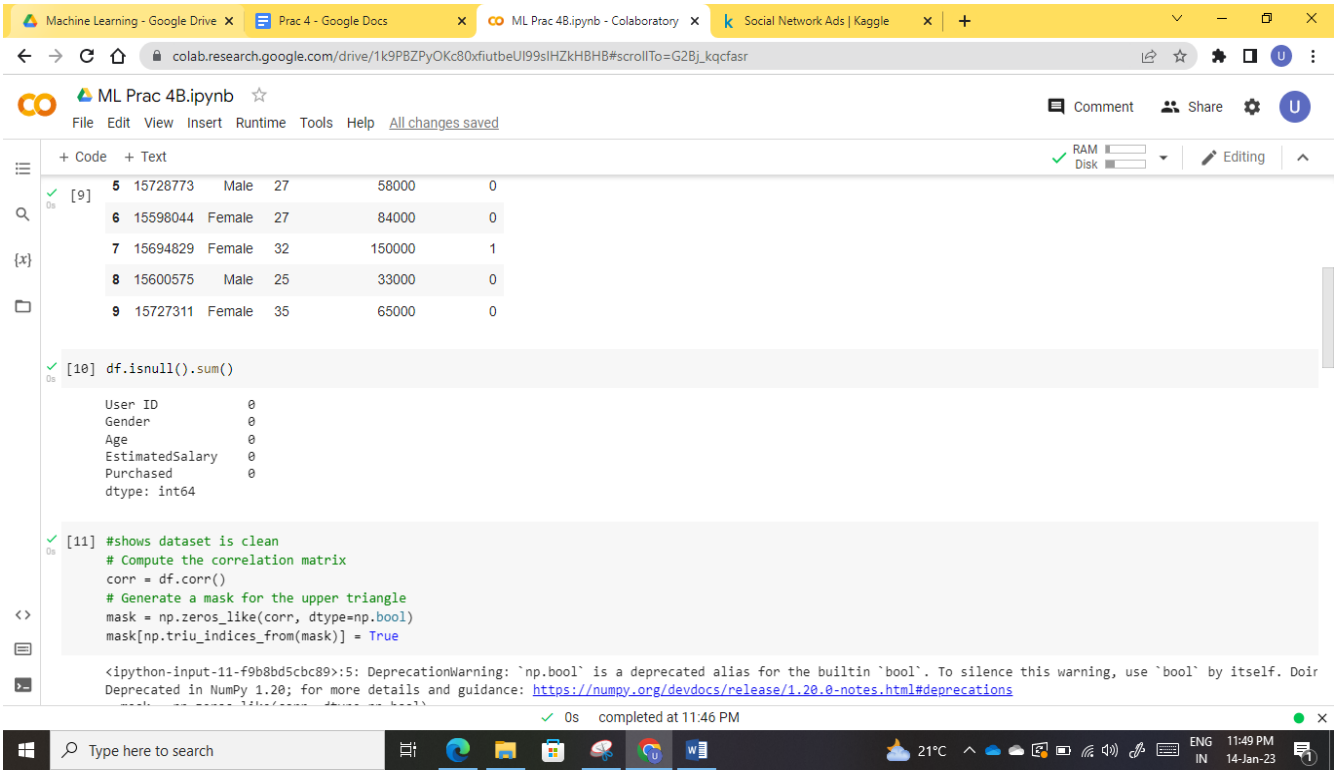
Code :

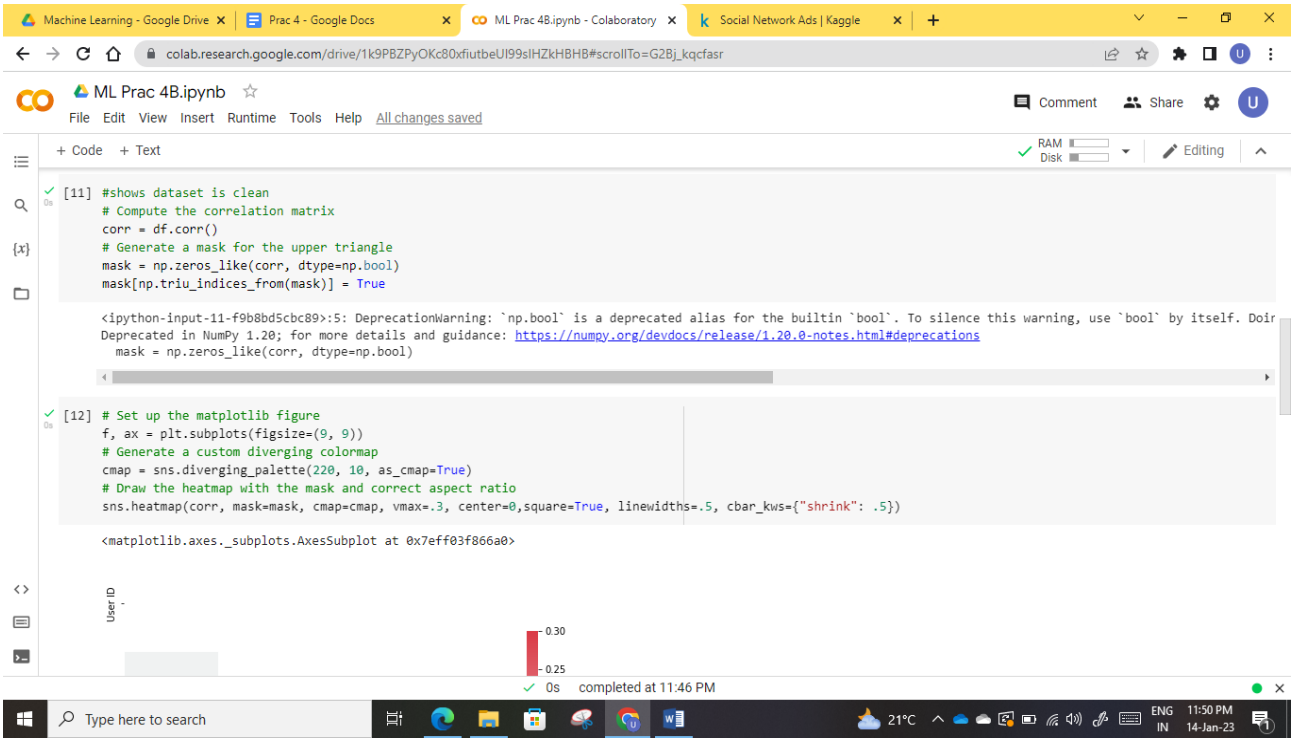
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import cross_val_score
df = pd.read_csv('/content/Social_Network_Ads.csv')
df.head(10)
df.isnull().sum()

#shows dataset is clean
# Compute the correlation matrix
corr = df.corr()
# Generate a mask for the upper triangle
mask = np.zeros_like(corr, dtype=np.bool)
mask[np.triu_indices_from(mask)] = True

# Set up the matplotlib figure
f, ax = plt.subplots(figsize=(9, 9))
# Generate a custom diverging colormap
cmap = sns.diverging_palette(220, 10, as_cmap=True)
# Draw the heatmap with the mask and correct aspect ratio
sns.heatmap(corr, mask=mask, cmap=cmap, vmax=.3, center=0,square=True, linewidths=.5,
            cbar_kws={"shrink": .5})
```

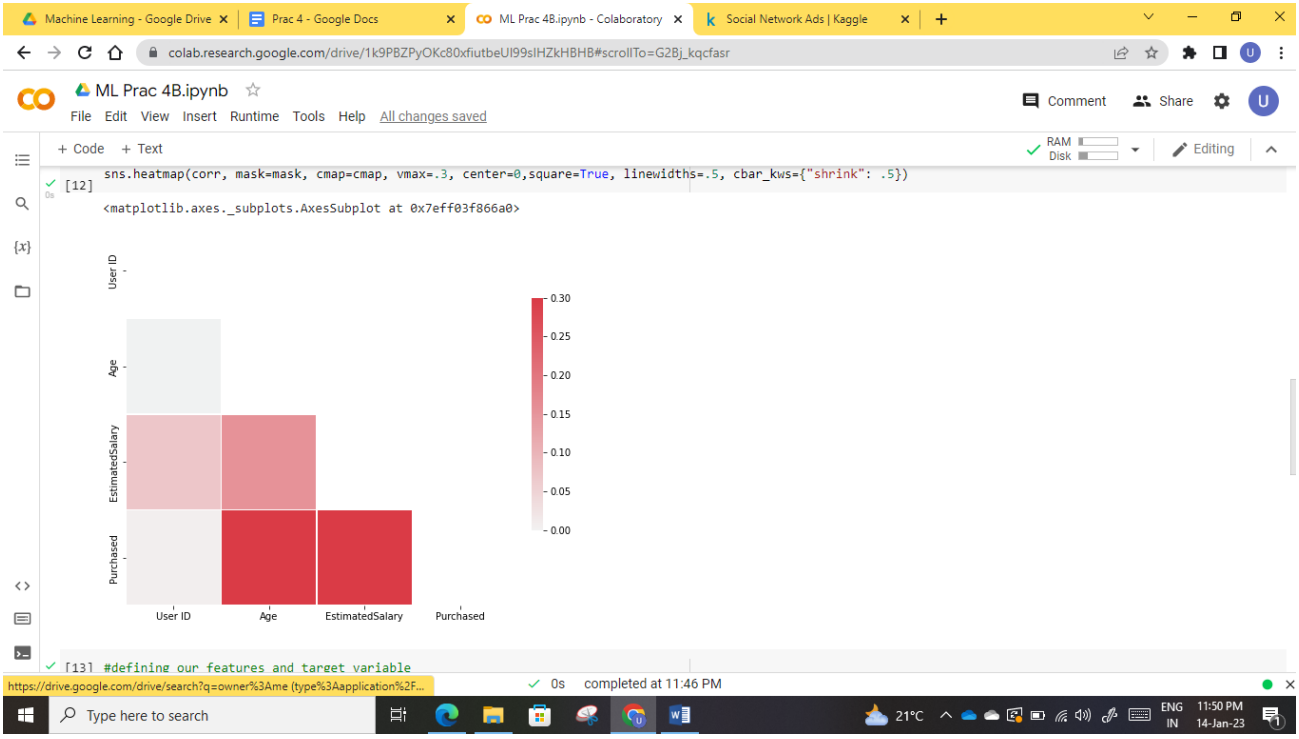
```
#defining our features and target variable
X = df.iloc[:, [2, 3]].values
y = df.iloc[:, 4].values
# Splitting the dataset into the Training set and Test set - 80-20 split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state = 0)

#Feature scaling as range of estimated salary and age is different
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

#Classifying and prediction
classifier = LogisticRegression(random_state = 0)  #Logistic classifier
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)  #predicting test results
y_pred

#Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
#using K-Fold cross validation to get the mean Accuracy
accuracies = cross_val_score(estimator = classifier, X = X_train, y = y_train, cv = 10)
print('Mean Accuracy: {0:.2f}, Std of Accuracy: {1:.2f}'. format(accuracies.mean(),
      accuracies.std()))
```

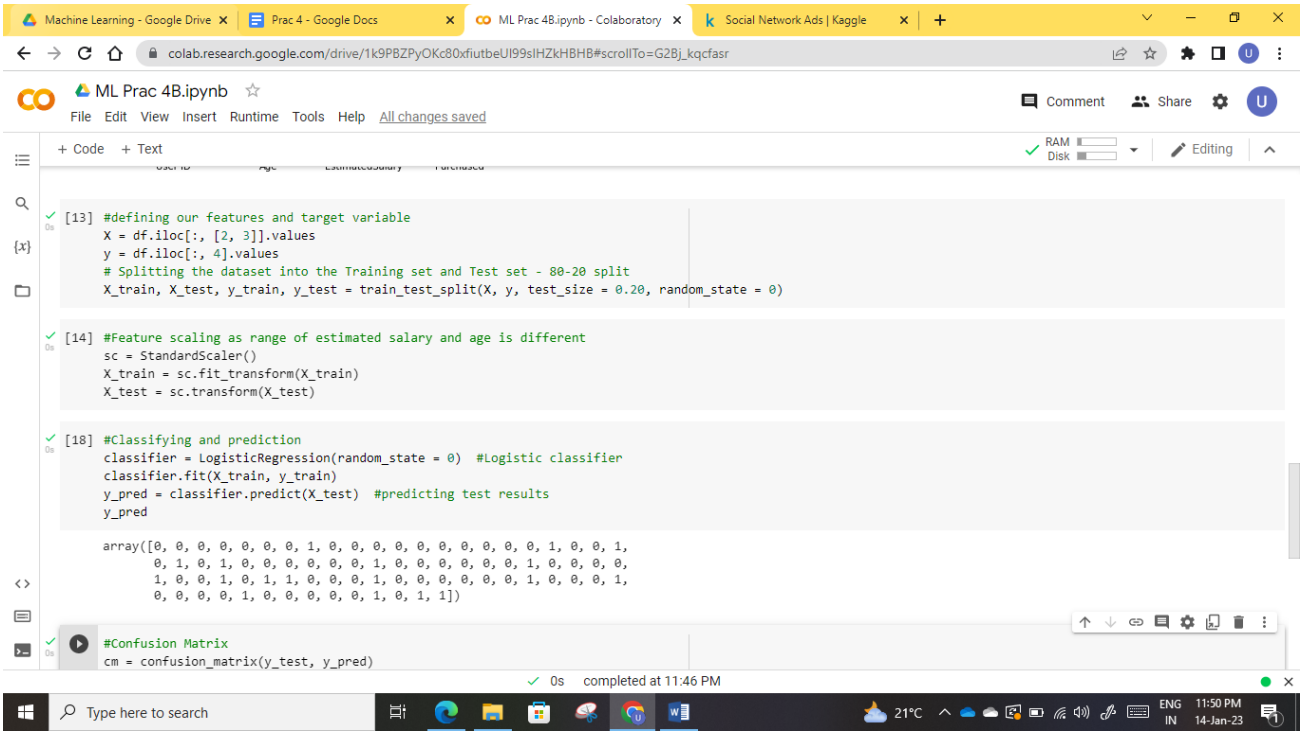
Output :

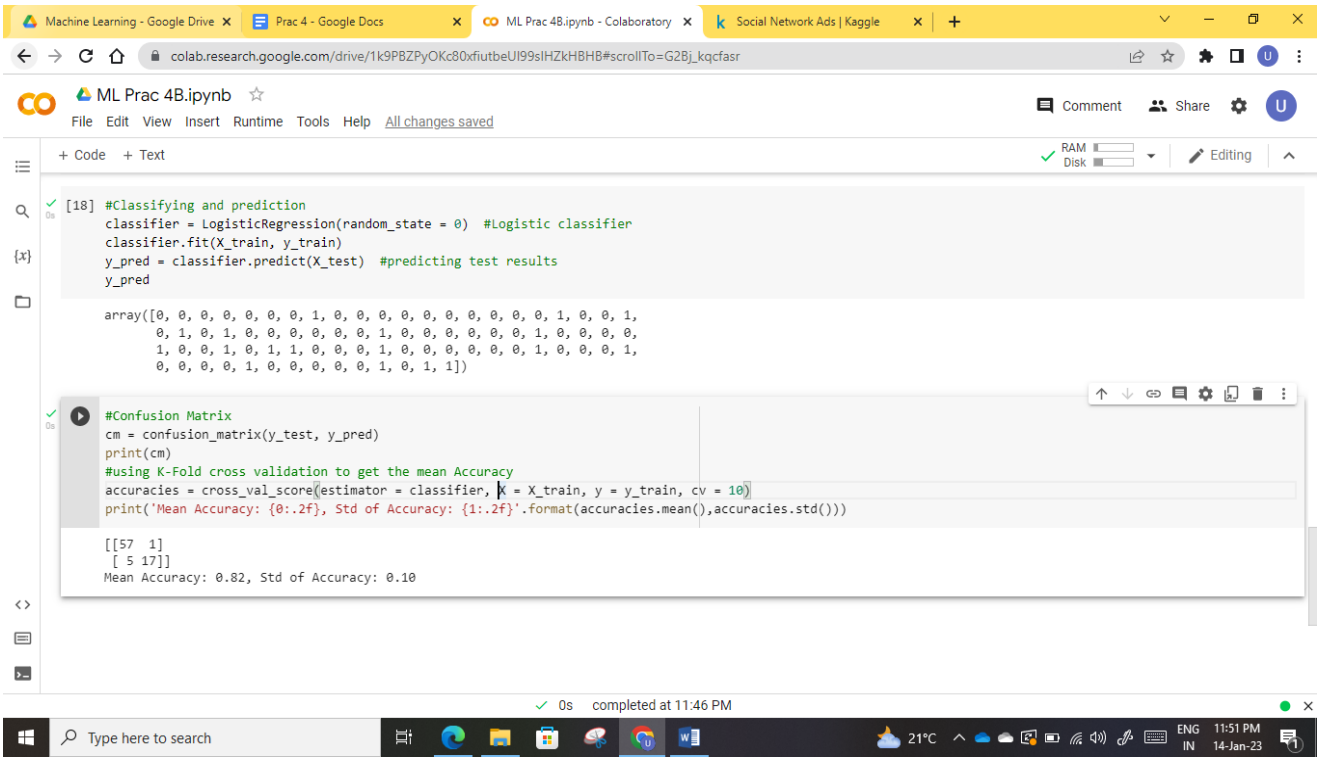

Machine Learning - Google Drive
Prac 4 - Google Docs
ML Prac 4B.ipynb - Colaboratory
Social Network Ads | Kaggle
colab.research.google.com/drive/1k9PBZPyOKc80xfiutbeUI99sIHZkHBHB#scrollTo=G2Bj_kqcfasr
ML Prac 4B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[8] import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import cross_val_score
[9] df = pd.read_csv('/content/Social_Network_Ads.csv')
df.head(10)
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
5 15728773 Male 27 58000 0
6 15598044 Female 27 84000 0
0s completed at 11:46 PM
Type here to search
21°C
ENG IN
11:49 PM 14-Jan-23


Machine Learning - Google Drive
Prac 4 - Google Docs
ML Prac 4B.ipynb - Colaboratory
Social Network Ads | Kaggle
colab.research.google.com/drive/1k9PBZPyOKc80xfiutbeUI99sIHZkHBHB#scrollTo=G2Bj_kqcfasr
ML Prac 4B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[9]
5 15728773 Male 27 58000 0
6 15598044 Female 27 84000 0
7 15694829 Female 32 150000 1
8 15600575 Male 25 33000 0
9 15727311 Female 35 65000 0
[10] df.isnull().sum()
User ID 0
Gender 0
Age 0
EstimatedSalary 0
Purchased 0
dtype: int64
[11] #shows dataset is clean
# Compute the correlation matrix
corr = df.corr()
# Generate a mask for the upper triangle
mask = np.zeros_like(corr, dtype=np.bool)
mask[np.triu_indices_from(mask)] = True
<ipython-input-11-f9b8bd5cbc89>:5: DeprecationWarning: `np.bool` is a deprecated alias for the builtin `bool`. To silence this warning, use `bool` by itself. Doir
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
0s completed at 11:46 PM
Type here to search
21°C
ENG IN
11:49 PM 14-Jan-23


Machine Learning - Google Drive
Prac 4 - Google Docs
ML Prac 4B.ipynb - Colaboratory
Social Network Ads | Kaggle
colab.research.google.com/drive/1k9PBZPyOKc80xfiutbeUI99sIHZkHBHB#scrollTo=G2Bj_kqcfasr
ML Prac 4B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[11] #shows dataset is clean
# Compute the correlation matrix
corr = df.corr()
# Generate a mask for the upper triangle
mask = np.zeros_like(corr, dtype=np.bool)
mask[np.triu_indices_from(mask)] = True
<ipython-input-11-f9b8bd5cbc89>:5: DeprecationWarning: `np.bool` is a deprecated alias for the builtin `bool`. To silence this warning, use `bool` by itself. Doir
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
mask = np.zeros_like(corr, dtype=np.bool)
[12] # Set up the matplotlib figure
f, ax = plt.subplots(figsize=(9, 9))
# Generate a custom diverging colormap
cmap = sns.diverging_palette(220, 10, as_cmap=True)
# Draw the heatmap with the mask and correct aspect ratio
sns.heatmap(corr, mask=mask, cmap=cmap, vmax=.3, center=0,square=True, linewidths=.5, cbar_kws={"shrink": .5})
<matplotlib.axes._subplots.AxesSubplot at 0x7eff03f866a0>
User ID
0.30
0.25
0s completed at 11:46 PM
Type here to search
21°C
ENG IN
11:50 PM
14-Jan-23


Machine Learning - Google Drive
Prac 4 - Google Docs
ML Prac 4B.ipynb - Colaboratory
Social Network Ads | Kaggle
colab.research.google.com/drive/1k9PBZPyOKc80xfiutbeUI99sIHZkHBHB#scrollTo=G2Bj_kqcfasr
ML Prac 4B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[12] sns.heatmap(corr, mask=mask, cmap=cmap, vmax=.3, center=0,square=True, linewidths=.5, cbar_kws={"shrink": .5})
<matplotlib.axes._subplots.AxesSubplot at 0x7eff03f866a0>
User ID
Age
EstimatedSalary
Purchased
0.30
0.25
0.20
0.15
0.10
0.05
0.00
[13] #defining our features and target variable
0s completed at 11:46 PM
Type here to search
21°C
ENG IN
11:50 PM
14-Jan-23


Machine Learning - Google Drive
Prac 4 - Google Docs
ML Prac 4B.ipynb - Colaboratory
Social Network Ads | Kaggle
colab.research.google.com/drive/1k9PBZPyOKc80xfiutbeUI99sIHZkHBHB#scrollTo=G2Bj_kqcfasr
ML Prac 4B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[13] #defining our features and target variable
X = df.iloc[:, [2, 3]].values
y = df.iloc[:, 4].values
# Splitting the dataset into the Training set and Test set - 80-20 split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state = 0)
[14] #Feature scaling as range of estimated salary and age is different
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
[18] #Classifying and prediction
classifier = LogisticRegression(random_state = 0) #Logistic classifier
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test) #predicting test results
y_pred
array([0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,
0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1,
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1])
#Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
0s completed at 11:46 PM
Type here to search
21°C
ENG IN
11:50 PM
14-Jan-23


Machine Learning - Google Drive
Prac 4 - Google Docs
ML Prac 4B.ipynb - Colaboratory
Social Network Ads | Kaggle
colab.research.google.com/drive/1k9PBZPyOKc80xfiutbeUI99sIHZkHBHB#scrollTo=G2Bj_kqcfasr
ML Prac 4B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[18] #Classifying and prediction
classifier = LogisticRegression(random_state = 0) #Logistic classifier
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test) #predicting test results
y_pred
array([0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,
0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1,
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1])
#Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
#using K-Fold cross validation to get the mean Accuracy
accuracies = cross_val_score(estimator = classifier, X = X_train, y = y_train, cv = 10)
print('Mean Accuracy: {0:.2f}, Std of Accuracy: {1:.2f}'.format(accuracies.mean(),accuracies.std()))
[[57 1]
[5 17]]
Mean Accuracy: 0.82, Std of Accuracy: 0.10
0s completed at 11:46 PM
Type here to search
21°C
ENG IN
11:51 PM
14-Jan-23

Write a program to demonstrate the working of the decision tree based ID3 algorithm. Use an appropriate data set for building the decision tree and apply this knowledge to classify a new sample

Code :

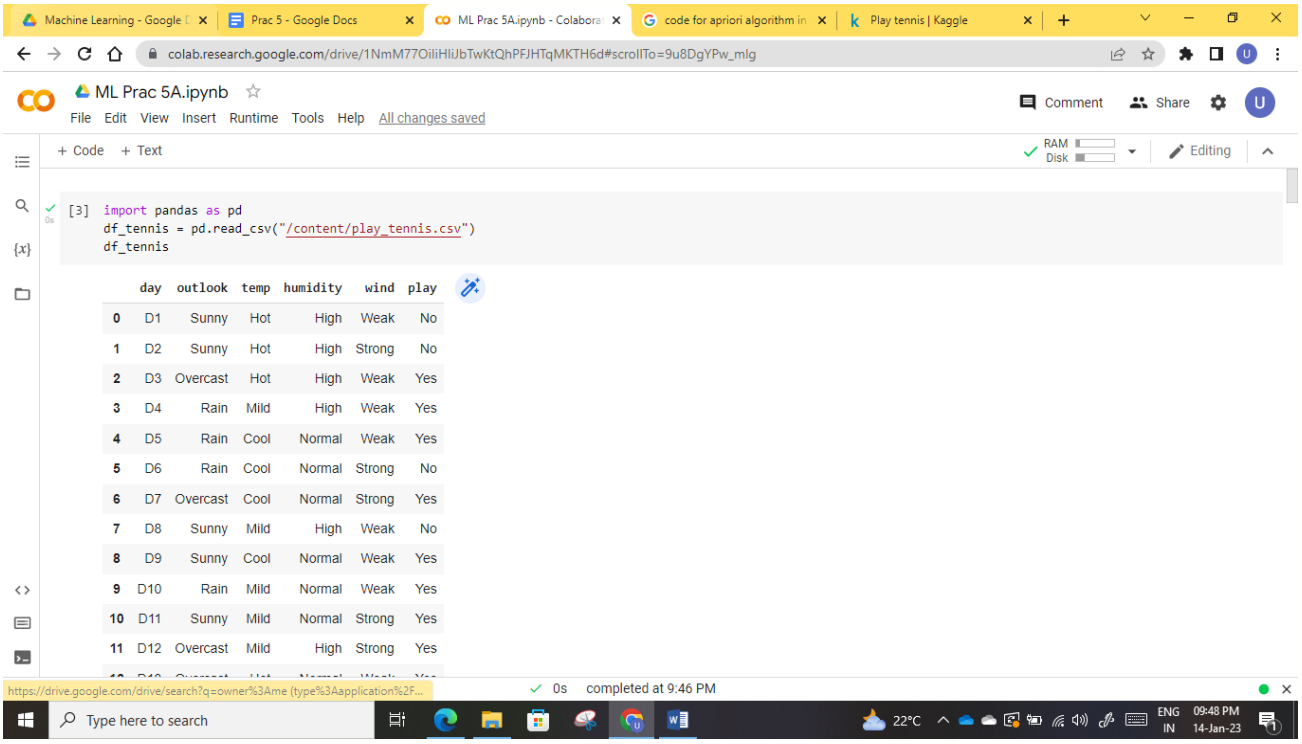
```
import pandas as pd
df_tennis = pd.read_csv("/content/play_tennis.csv")
df_tennis

def entropy(probs):
    import math
    return sum( [-prob*math.log(prob, 2) for prob in probs] )
#Function to calulate the entropy of the given Data Sets/List with respect to target attributes
def entropy_of_list(a_list):
    from collections import Counter
    cnt = Counter(x for x in a_list)   # Counter calculates the propotion of class
    num_instances = len(a_list)*1.0   # = 14
    print("\n Number of Instances of the Current Sub Class is {0}:".format(num_instances ))
    probs = [x / num_instances for x in cnt.values()]  # x means no of YES/NO
    print("\n Classes:",min(cnt),max(cnt))
    print(" \n Probabilities of Class {0} is {1}:".format(min(cnt),min(probs)))
    print(" \n Probabilities of Class {0} is {1}:".format(max(cnt),max(probs)))
    return entropy(probs) # Call Entropy :

# The initial entropy of the YES/NO attribute for our dataset.
print("\n  INPUT DATA SET FOR ENTROPY CALCULATION:\n", df_tennis['play'])
total_entropy = entropy_of_list(df_tennis['play'])
print("\n Total Entropy of play Data Set:",total_entropy)

#Information gain
def information_gain(df, split_attribute_name, target_attribute_name, trace=0):
    print("Information Gain Calculation of ",split_attribute_name)
```

```
        # Split Data by Possible Vals of Attribute:
        df_split = df.groupby(split_attribute_name)
        # Proportion of Obs in Each Data-Split
        nobs = len(df.index) * 1.0
        df_agg_ent = df_split.agg({target_attribute_name : [entropy_of_list, lambda x:
                        len(x)/nobs] })[target_attribute_name]
        df_agg_ent.columns = ['Entropy', 'PropObservations']
        # Calculate Information Gain:
        new_entropy = sum( df_agg_ent['Entropy'] * df_agg_ent['PropObservations'] )
        old_entropy = entropy_of_list(df[target_attribute_name])
        return old_entropy - new_entropy

print('Info-gain for Outlook is :'+str( information_gain(df_tennis, 'outlook','play')),"\n")
print('\n Info-gain for Humidity is: ' + str( information_gain(df_tennis, 'humidity','play')),"\n")
print('\n Info-gain for Wind is:' + str( information_gain(df_tennis, 'wind','play')),"\n")
print('\n Info-gain for Temperature is:' + str( information_gain(df_tennis, 'temp','play')),"\n")

def id3(df, target_attribute_name, attribute_names, default_class=None):
        ## Tally target attribute:
        from collections import Counter
        cnt = Counter(x for x in df[target_attribute_name])# class of YES /NO
        ## First check: Is this split of the dataset homogeneous?
        if len(cnt) == 1:
                return next(iter(cnt))
        elif df.empty or (not attribute_names):
                return default_class  # Return None for Empty Data Set
        ## Otherwise: This dataset is ready to be devied up!
        else:
                # Get Default Value for next recursive call of this function:
                default_class = max(cnt.keys()) #No of YES and NO Class
                # Compute the Information Gain of the attributes:
                gainz = [information_gain(df, attr, target_attribute_name) for attr in
                        attribute_names]
                index_of_max = gainz.index(max(gainz)) # Index of Best Attribute
```

```
            # Choose Best Attribute to split on:
            best_attr = attribute_names[index_of_max]
            # Create an empty tree, to be populated in a moment
            tree = {best_attr:{}} # Iniiate the tree with best attribute as a node
            remaining_attribute_names = [i for i in attribute_names if i != best_attr]
            for attr_val, data_subset in df.groupby(best_attr):
                subtree = id3(data_subset,target_attribute_name ,remaining_ attribute_
                            names,default_class)
                tree[best_attr][attr_val] = subtree
            return tree

# Get Predictor Names (all but 'class')
attribute_names = list(df_tennis.columns)
print("List of Attributes:", attribute_names)
attribute_names.remove('play') #Remove the class attribute
print("Predicting Attributes:", attribute_names)

# Run Algorithm:
from pprint import pprint
tree = id3(df_tennis,'play',attribute_names)
print("\n\nThe Resultant Decision Tree is :\n")
#print(tree)
pprint(tree)
attribute = next(iter(tree))
print("Best Attribute :\n",attribute)
print("Tree Keys:\n",tree[attribute].keys())

#Classification accuracy
def classify(instance, tree, default=None): # Instance of Play Tennis with Predicted
    #print("Instance:",instance)
    attribute = next(iter(tree)) # Outlook/Humidity/Wind
    print("Key:",tree.keys())  # [Outlook,Humidity,Wind ]
    print("Attribute:",attribute) # [Key /Attribute Both are same ]
    if instance[attribute] in tree[attribute].keys(): # Value of the attributs in  set of Tree keys
```

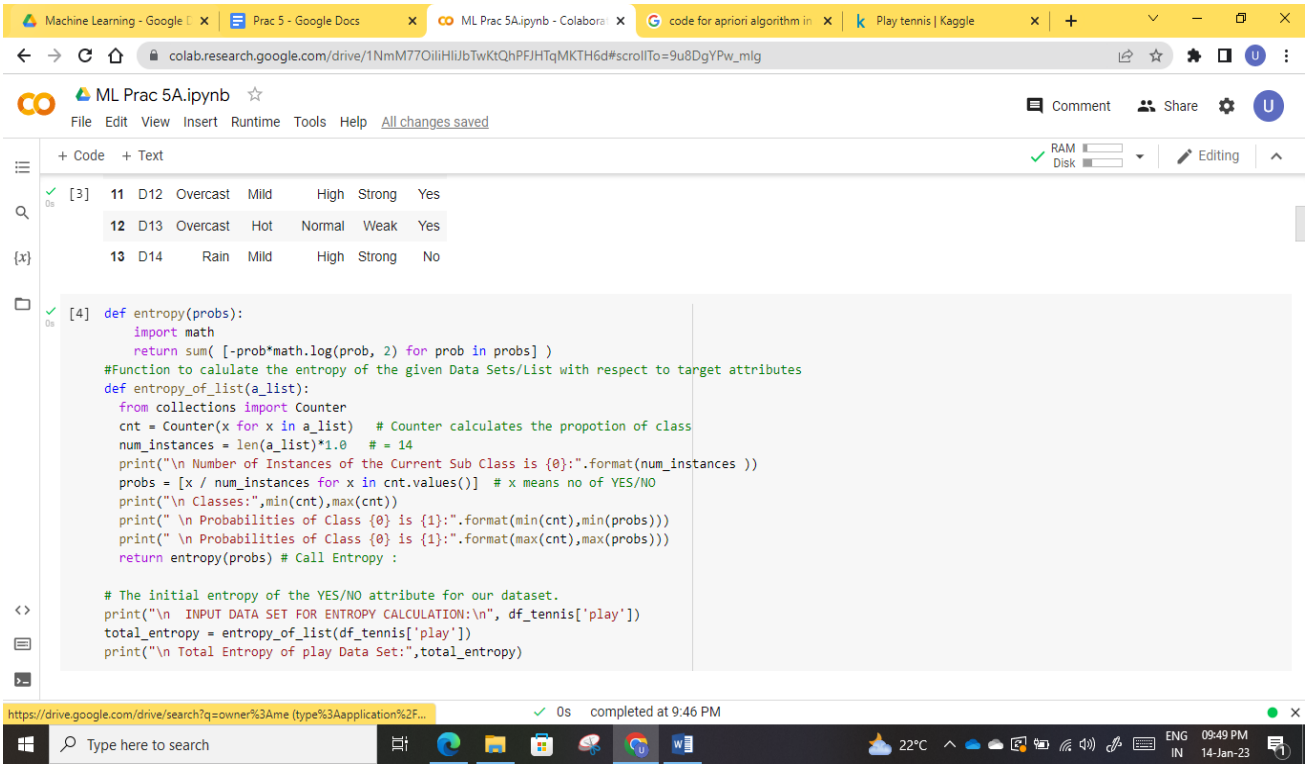
```
                result = tree[attribute][instance[attribute]]
                print("Instance Attribute:",instance[attribute],"TreeKeys :",tree[attribute].keys())
                if isinstance(result, dict): # this is a tree, delve deeper
                        return classify(instance, result)
                else:
                        return result # this is a label
        else:
                return default

df_tennis['predicted'] = df_tennis.apply(classify, axis=1, args=(tree,'No') )
print(df_tennis['predicted'])
print('\n Accuracy is:\n' + str( sum(df_tennis['play']==df_tennis['predicted'] ) / (1.0*len
                                (df_tennis.index)) ))
df_tennis[['play', 'predicted']]
df_tennis['predicted'] = df_tennis.apply(classify, axis=1, args=(tree,'No') )
print(df_tennis['predicted'])
print('\n Accuracy is:\n' + str( sum(df_tennis['play']==df_tennis['predicted'] ) / (1.0*len
                                (df_tennis.index)) ))
df_tennis[['play', 'predicted']]



#Classification accuracy training/testing set
training_data = df_tennis.iloc[1:4]
test_data = df_tennis.iloc[-4:]
train_tree = id3(training_data,'play',attribute_names)
test_data['predicted2'] = test_data.apply(classify,axis=1,args=(train_tree,'Yes') ) # <---- train_data
tree
print ('\n\n Accuracy is : ' + str( sum(test_data['play']==test_data['predicted2'] ) /
(1.0*len(test_data.index))))
```

Output :



```
[3] import pandas as pd
    df_tennis = pd.read_csv("/content/play_tennis.csv")
    df_tennis
```

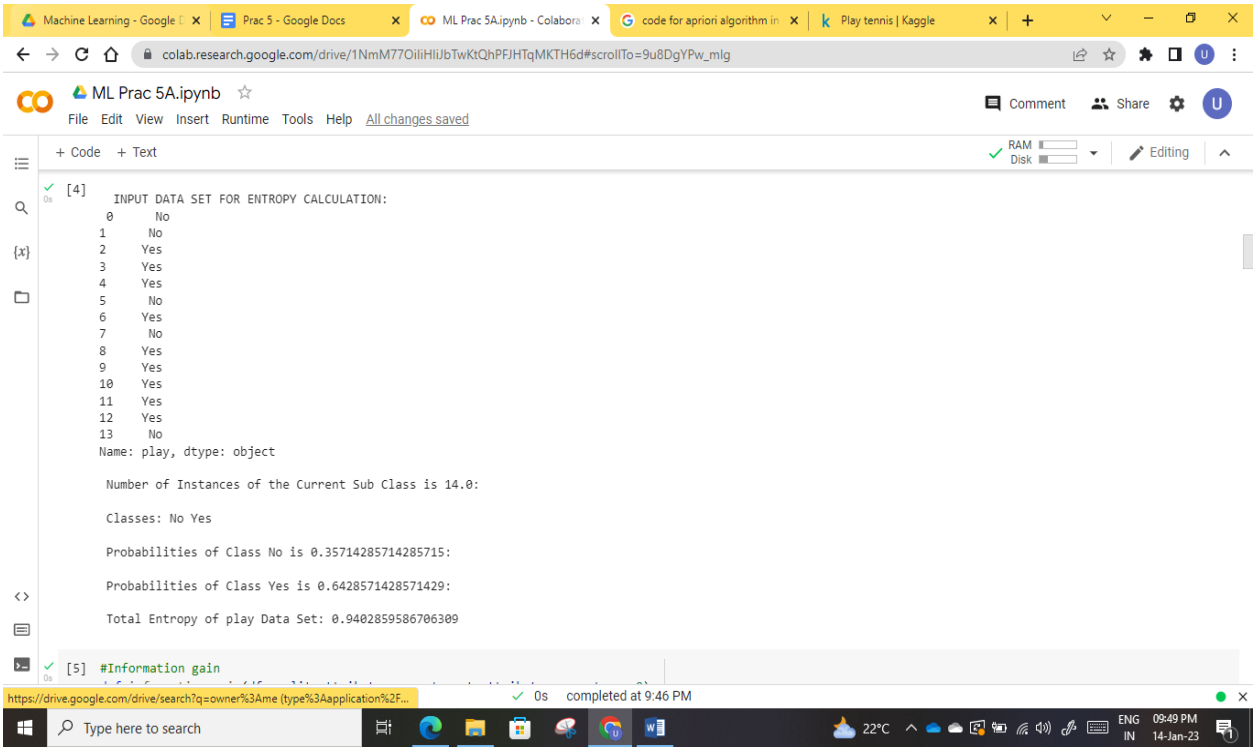
	day	outlook	temp	humidity	wind	play
0	D1	Sunny	Hot	High	Weak	No
1	D2	Sunny	Hot	High	Strong	No
2	D3	Overcast	Hot	High	Weak	Yes
3	D4	Rain	Mild	High	Weak	Yes
4	D5	Rain	Cool	Normal	Weak	Yes
5	D6	Rain	Cool	Normal	Strong	No
6	D7	Overcast	Cool	Normal	Strong	Yes
7	D8	Sunny	Mild	High	Weak	No
8	D9	Sunny	Cool	Normal	Weak	Yes
9	D10	Rain	Mild	Normal	Weak	Yes
10	D11	Sunny	Mild	Normal	Strong	Yes
11	D12	Overcast	Mild	High	Strong	Yes

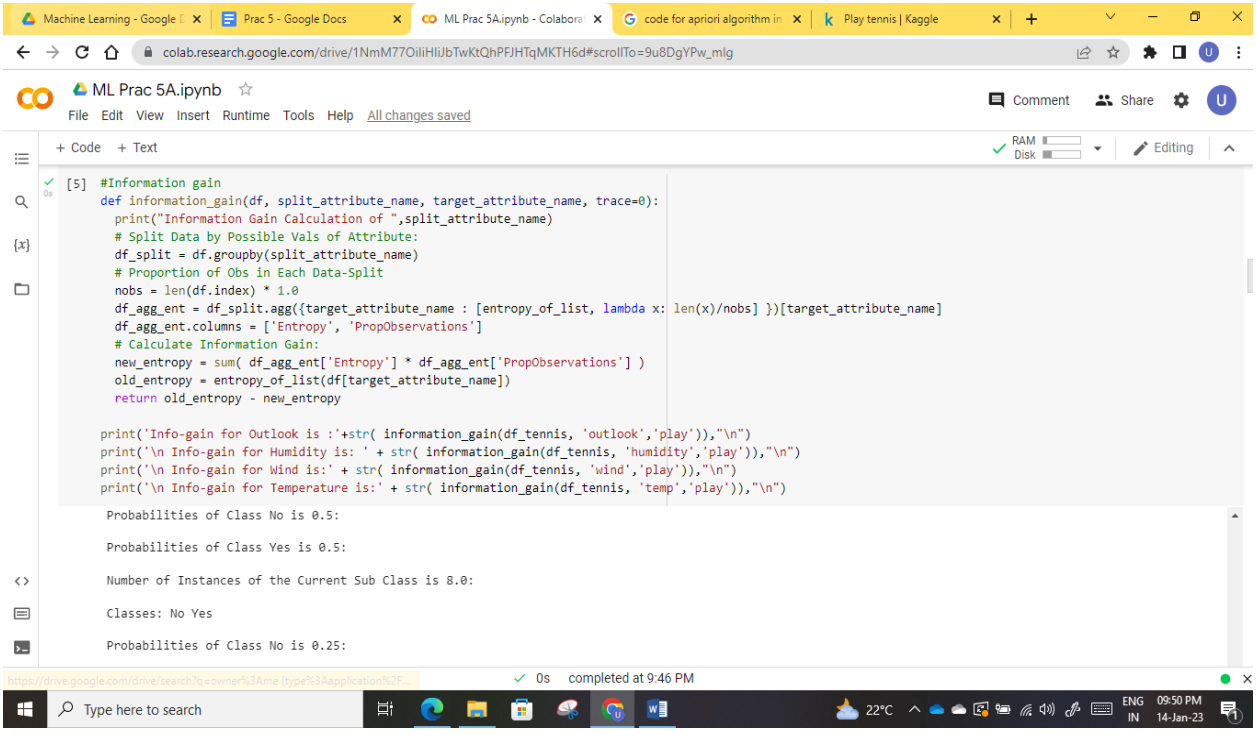


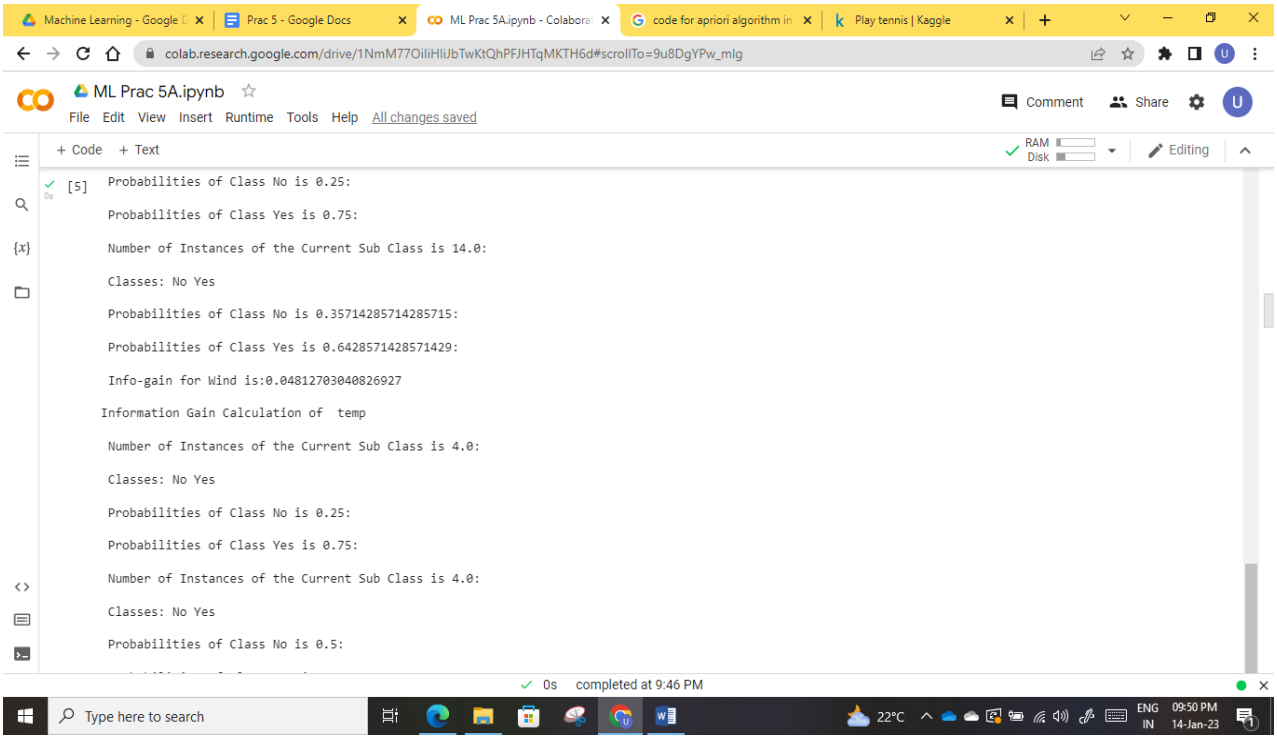
```
[3]
```

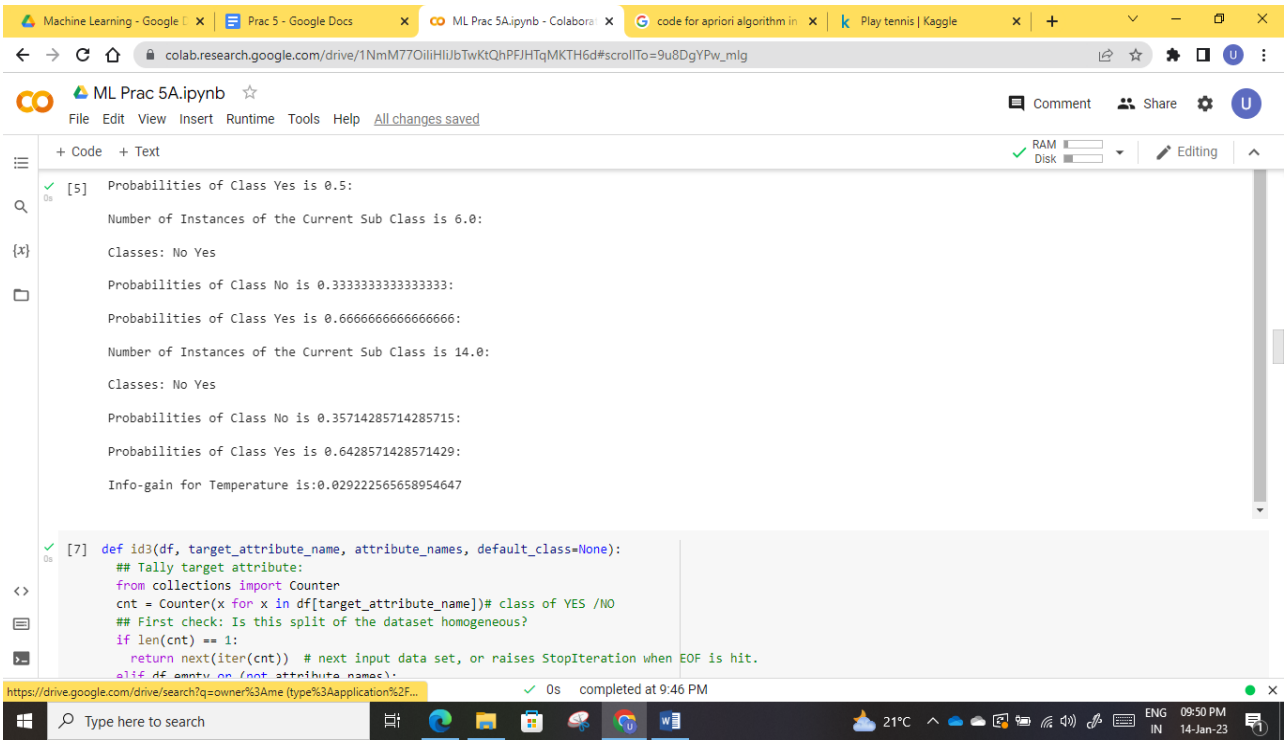
11	D12	Overcast	Mild	High	Strong	Yes
12	D13	Overcast	Hot	Normal	Weak	Yes
13	D14	Rain	Mild	High	Strong	No

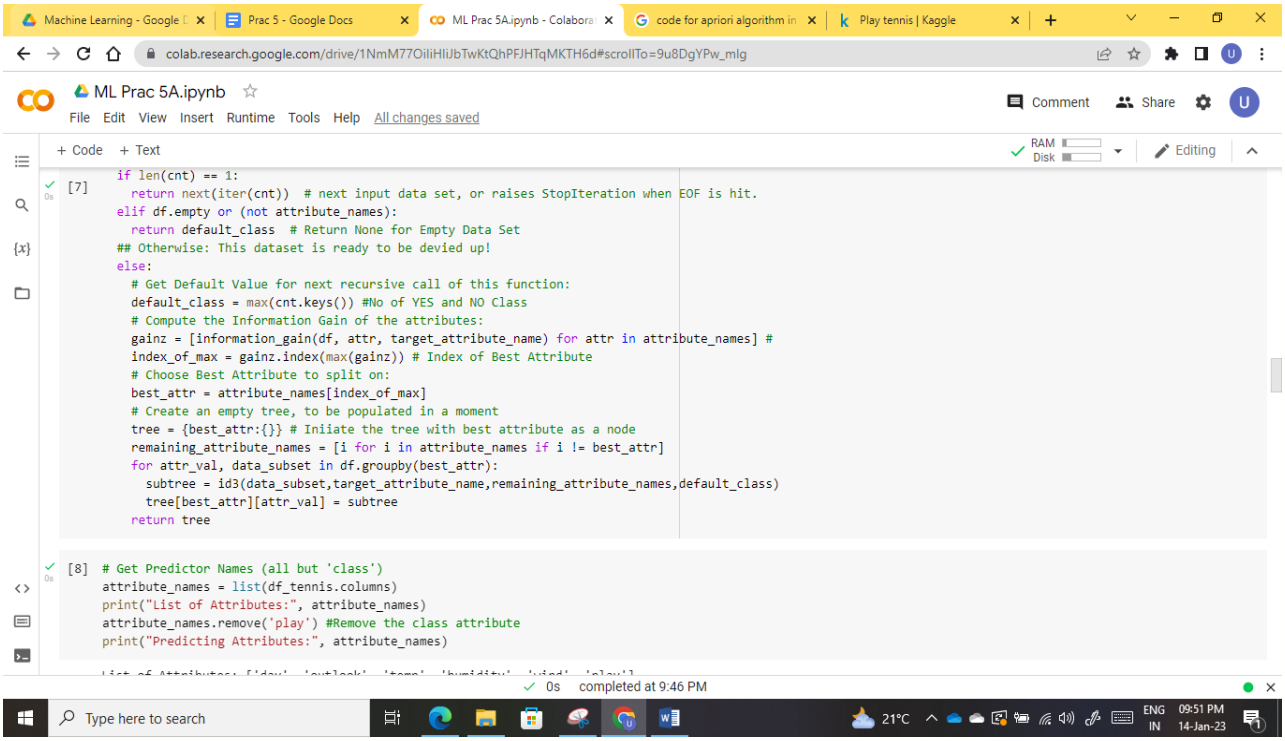
```
[4] def entropy(probs):
        import math
        return sum( [-prob*math.log(prob, 2) for prob in probs] )
    #Function to calulate the entropy of the given Data Sets/List with respect to target attributes
    def entropy_of_list(a_list):
      from collections import Counter
      cnt = Counter(x for x in a_list)   # Counter calculates the propotion of class
      num_instances = len(a_list)*1.0   # = 14
      print("\n Number of Instances of the Current Sub Class is {0}:".format(num_instances ))
      probs = [x / num_instances for x in cnt.values()]  # x means no of YES/NO
      print("\n Classes:",min(cnt),max(cnt))
      print(" \n Probabilities of Class {0} is {1}:".format(min(cnt),min(probs)))
      print(" \n Probabilities of Class {0} is {1}:".format(max(cnt),max(probs)))
      return entropy(probs) # Call Entropy :

    # The initial entropy of the YES/NO attribute for our dataset.
    print("\n  INPUT DATA SET FOR ENTROPY CALCULATION:\n", df_tennis['play'])
    total_entropy = entropy_of_list(df_tennis['play'])
    print("\n Total Entropy of play Data Set:",total_entropy)
```

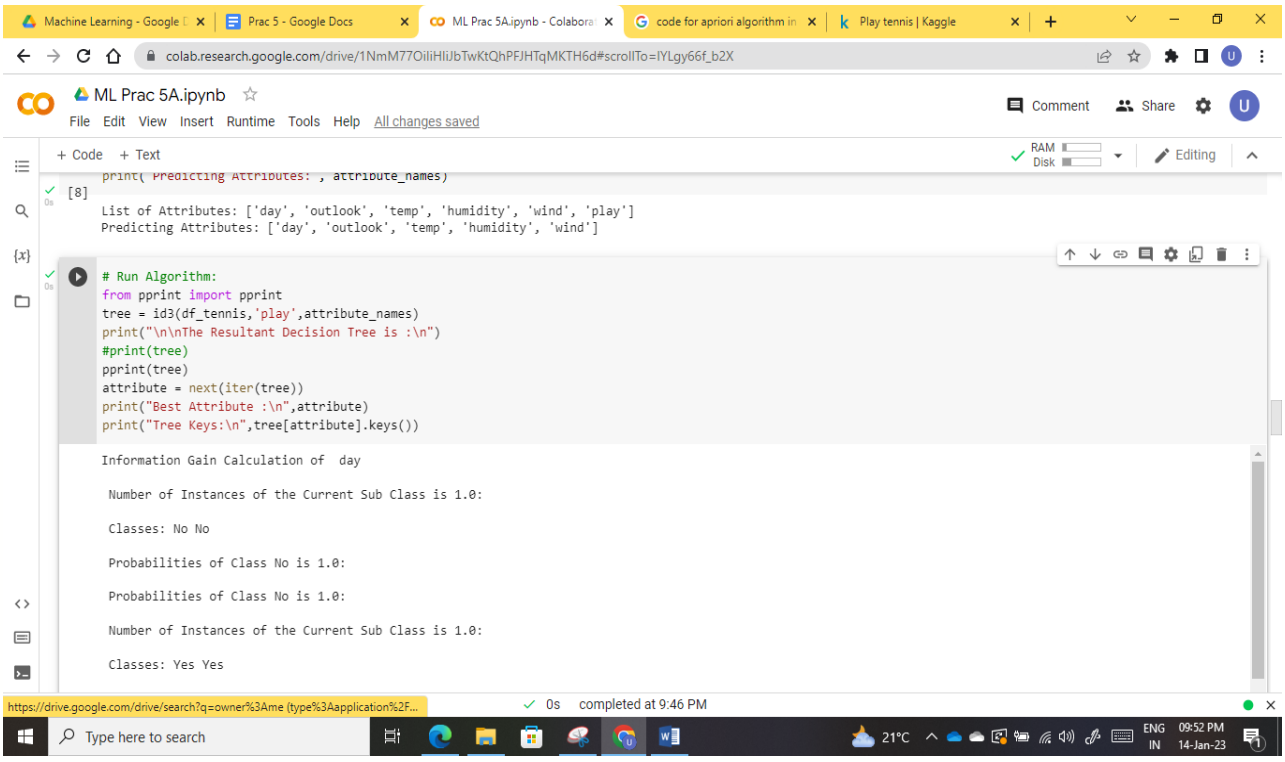

Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHliJbTwKtQhPFJHTqMKTH6d#scrollTo=9u8DgYPw_mlg
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[4]
INPUT DATA SET FOR ENTROPY CALCULATION:
0 No
1 No
2 Yes
3 Yes
4 Yes
5 No
6 Yes
7 No
8 Yes
9 Yes
10 Yes
11 Yes
12 Yes
13 No
Name: play, dtype: object
Number of Instances of the Current Sub Class is 14.0:
Classes: No Yes
Probabilities of Class No is 0.35714285714285715:
Probabilities of Class Yes is 0.6428571428571429:
Total Entropy of play Data Set: 0.9402859586706309
[5] #Information gain
0s completed at 9:46 PM
Type here to search
22°C
ENG
IN
09:49 PM
14-Jan-23

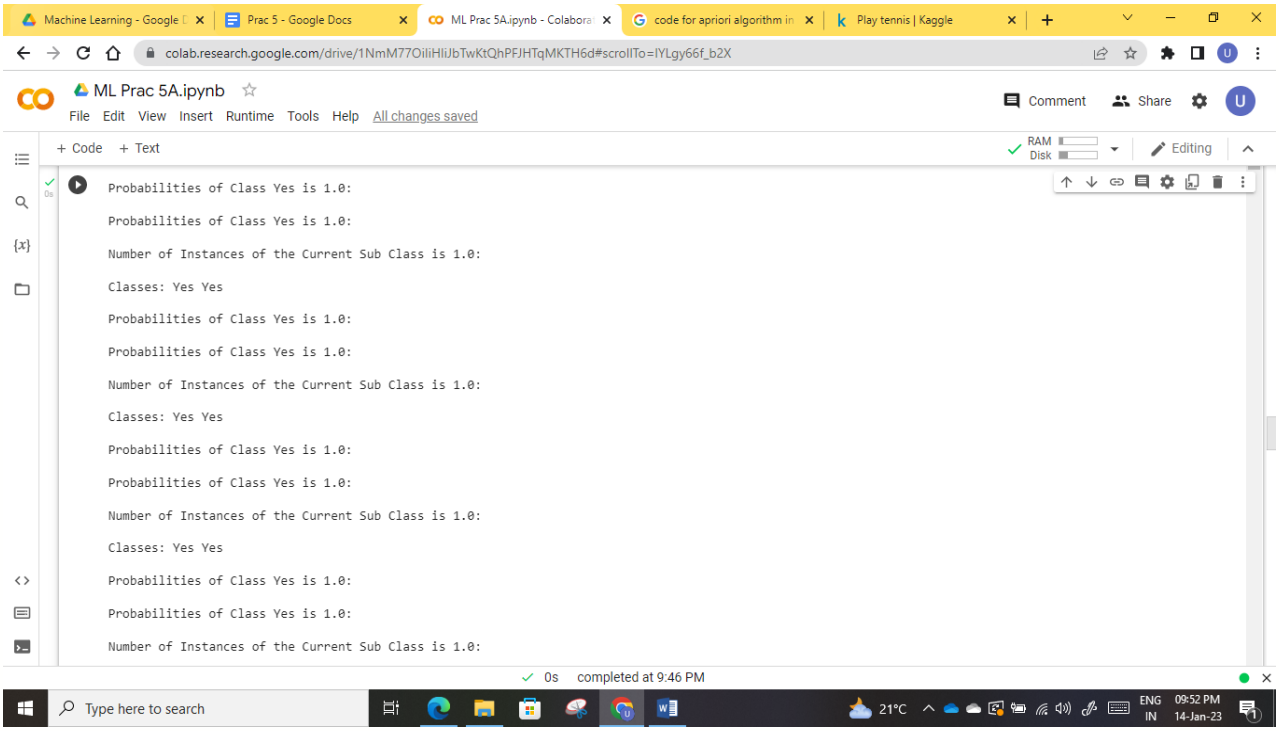

Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHliJbTwKtQhPFJHTqMKTH6d#scrollTo=9u8DgYPw_mlg
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[5] #Information gain
def information_gain(df, split_attribute_name, target_attribute_name, trace=0):
print("Information Gain Calculation of ",split_attribute_name)
# Split Data by Possible Vals of Attribute:
df_split = df.groupby(split_attribute_name)
# Proportion of Obs in Each Data-Split
nobs = len(df.index) * 1.0
df_agg_ent = df_split.agg({target_attribute_name : [entropy_of_list, lambda x: len(x)/nobs] })[target_attribute_name]
df_agg_ent.columns = ['Entropy', 'PropObservations']
# Calculate Information Gain:
new_entropy = sum(df_agg_ent['Entropy'] * df_agg_ent['PropObservations'])
old_entropy = entropy_of_list(df[target_attribute_name])
return old_entropy - new_entropy
print('Info-gain for Outlook is :'+str(information_gain(df_tennis, 'outlook','play')),"\n")
print('\n Info-gain for Humidity is: ' + str(information_gain(df_tennis, 'humidity','play')),"\n")
print('\n Info-gain for Wind is:' + str(information_gain(df_tennis, 'wind','play')),"\n")
print('\n Info-gain for Temperature is:' + str(information_gain(df_tennis, 'temp','play')),"\n")
Probabilities of Class No is 0.5:
Probabilities of Class Yes is 0.5:
Number of Instances of the Current Sub Class is 8.0:
Classes: No Yes
Probabilities of Class No is 0.25:
0s completed at 9:46 PM
Type here to search
22°C
ENG
IN
09:50 PM
14-Jan-23

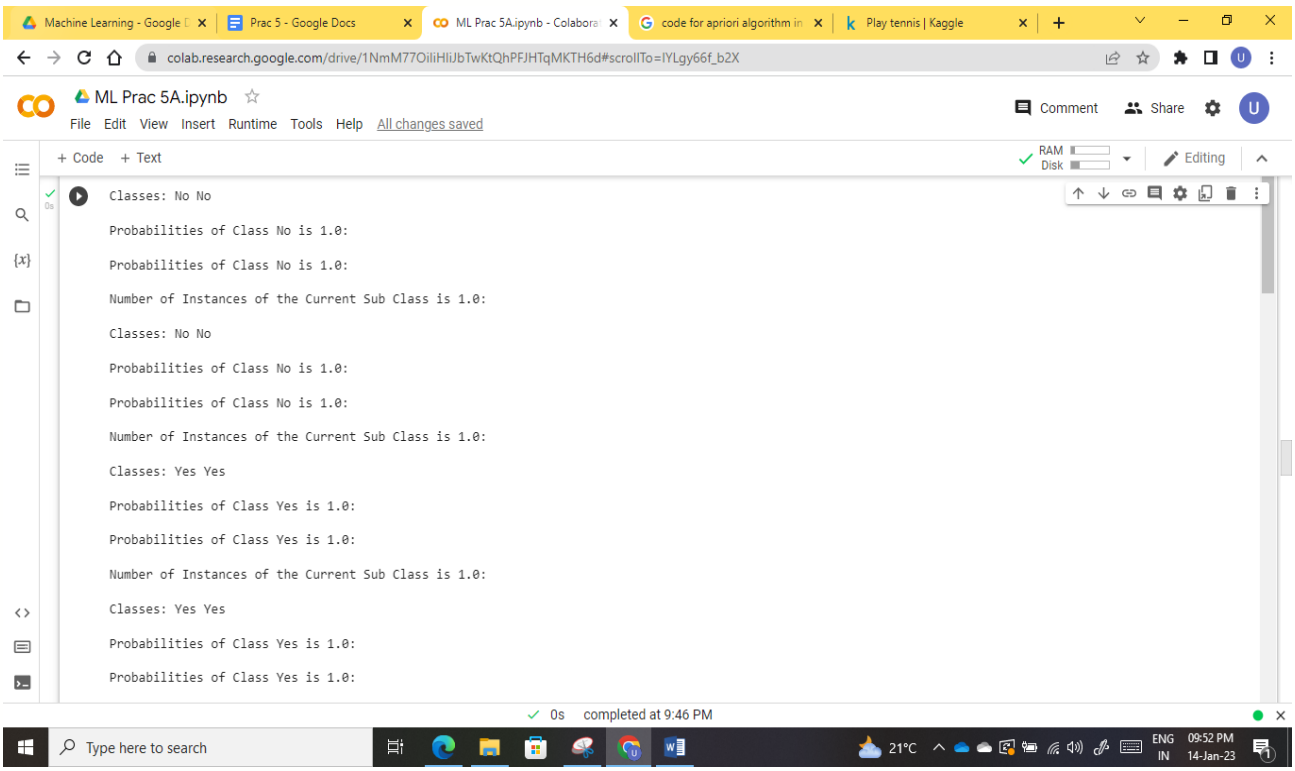

Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHIiJbTwKtQhPFJHTqMKTH6d#scrollTo=9u8DgYPw_mlg
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[5] Probabilities of Class No is 0.25:
Probabilities of Class Yes is 0.75:
Number of Instances of the Current Sub Class is 14.0:
Classes: No Yes
Probabilities of Class No is 0.35714285714285715:
Probabilities of Class Yes is 0.6428571428571429:
Info-gain for Wind is:0.0481270304082692
Information Gain Calculation of temp
Number of Instances of the Current Sub Class is 4.0:
Classes: No Yes
Probabilities of Class No is 0.25:
Probabilities of Class Yes is 0.75:
Number of Instances of the Current Sub Class is 4.0:
Classes: No Yes
Probabilities of Class No is 0.5:
0s completed at 9:46 PM
Type here to search
22°C
ENG IN
09:50 PM
14-Jan-23

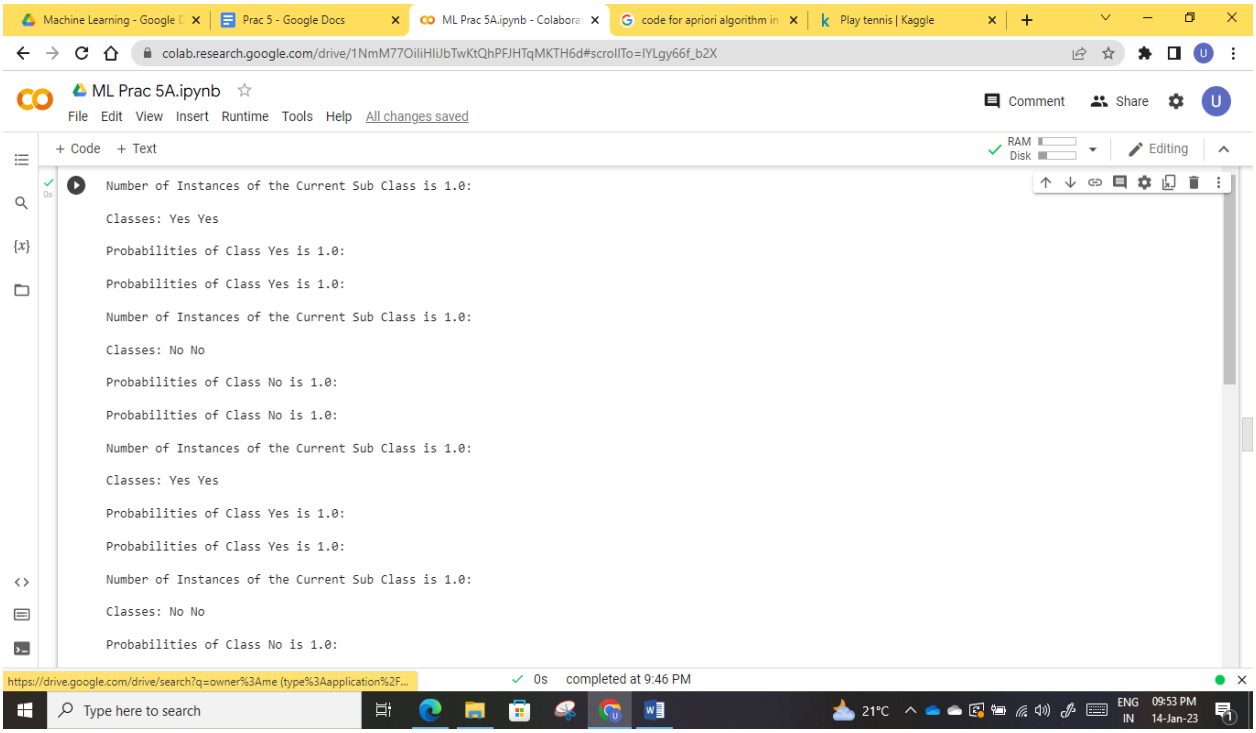

Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHIiJbTwKtQhPFJHTqMKTH6d#scrollTo=9u8DgYPw_mlg
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[5] Probabilities of Class Yes is 0.5:
Number of Instances of the Current Sub Class is 6.0:
Classes: No Yes
Probabilities of Class No is 0.3333333333333333:
Probabilities of Class Yes is 0.6666666666666666:
Number of Instances of the Current Sub Class is 14.0:
Classes: No Yes
Probabilities of Class No is 0.35714285714285715:
Probabilities of Class Yes is 0.6428571428571429:
Info-gain for Temperature is:0.029222565658954647
[7] def id3(df, target_attribute_name, attribute_names, default_class=None):
Tally target attribute:
from collections import Counter
cnt = Counter(x for x in df[target_attribute_name])# class of YES /NO
First check: Is this split of the dataset homogeneous?
if len(cnt) == 1:
return next(iter(cnt)) # next input data set, or raises StopIteration when EOF is hit.
elif df empty or (not attribute names):
0s completed at 9:46 PM
Type here to search
21°C
ENG IN
09:50 PM
14-Jan-23

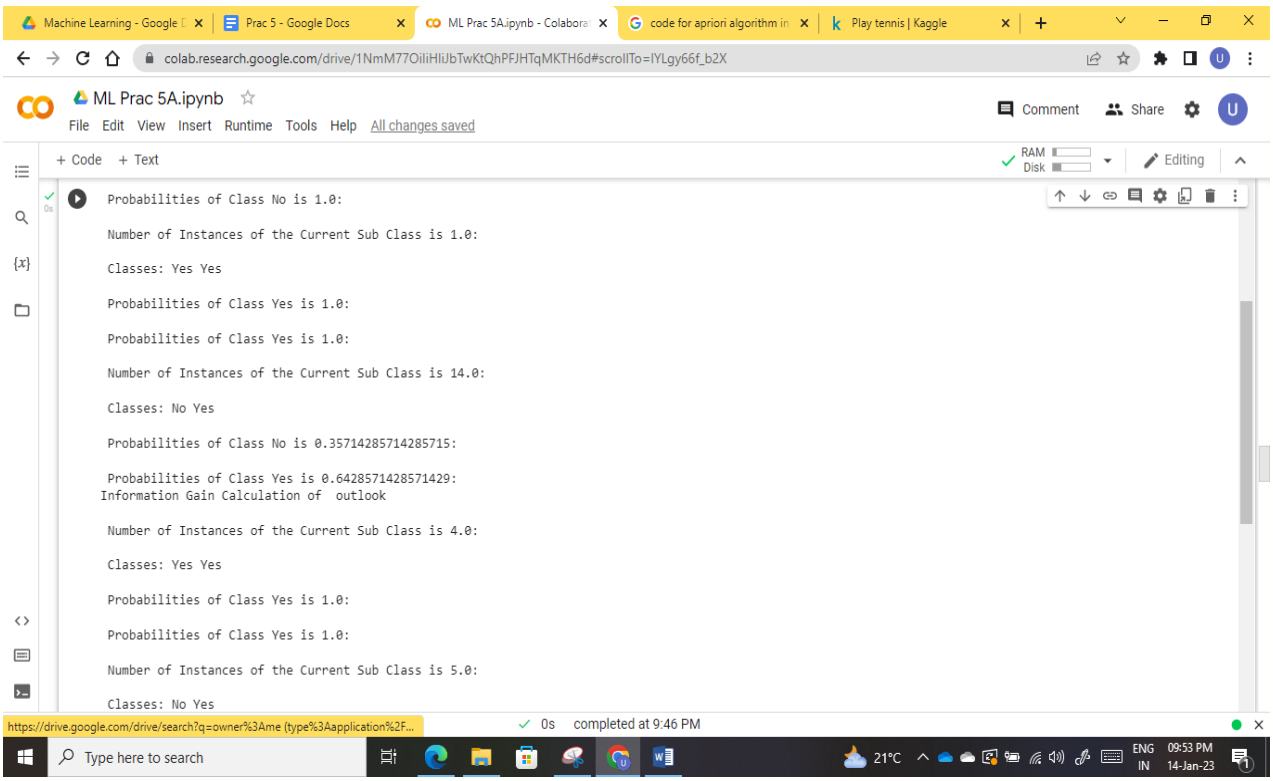

Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHIiJbTwKtQhPFJHTqMKTH6d#scrollTo=9u8DgYPw_mlg
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[7]
if len(cnt) == 1:
return next(iter(cnt)) # next input data set, or raises StopIteration when EOF is hit.
elif df.empty or (not attribute_names):
return default_class # Return None for Empty Data Set
Otherwise: This dataset is ready to be devied up!
else:
# Get Default Value for next recursive call of this function:
default_class = max(cnt.keys()) #No of YES and NO Class
# Compute the Information Gain of the attributes:
gainz = [information_gain(df, attr, target_attribute_name) for attr in attribute_names] #
index_of_max = gainz.index(max(gainz)) # Index of Best Attribute
# Choose Best Attribute to split on:
best_attr = attribute_names[index_of_max]
# Create an empty tree, to be populated in a moment
tree = {best_attr:{}} # Initiate the tree with best attribute as a node
remaining_attribute_names = [i for i in attribute_names if i != best_attr]
for attr_val, data_subset in df.groupby(best_attr):
subtree = id3(data_subset,target_attribute_name,remaining_attribute_names,default_class)
tree[best_attr][attr_val] = subtree
return tree
[8]
# Get Predictor Names (all but 'class')
attribute_names = list(df_tennis.columns)
print("List of Attributes:", attribute_names)
attribute_names.remove('play') #Remove the class attribute
print("Predicting Attributes:", attribute_names)
0s completed at 9:46 PM
Type here to search
21°C
ENG IN
09:51 PM
14-Jan-23

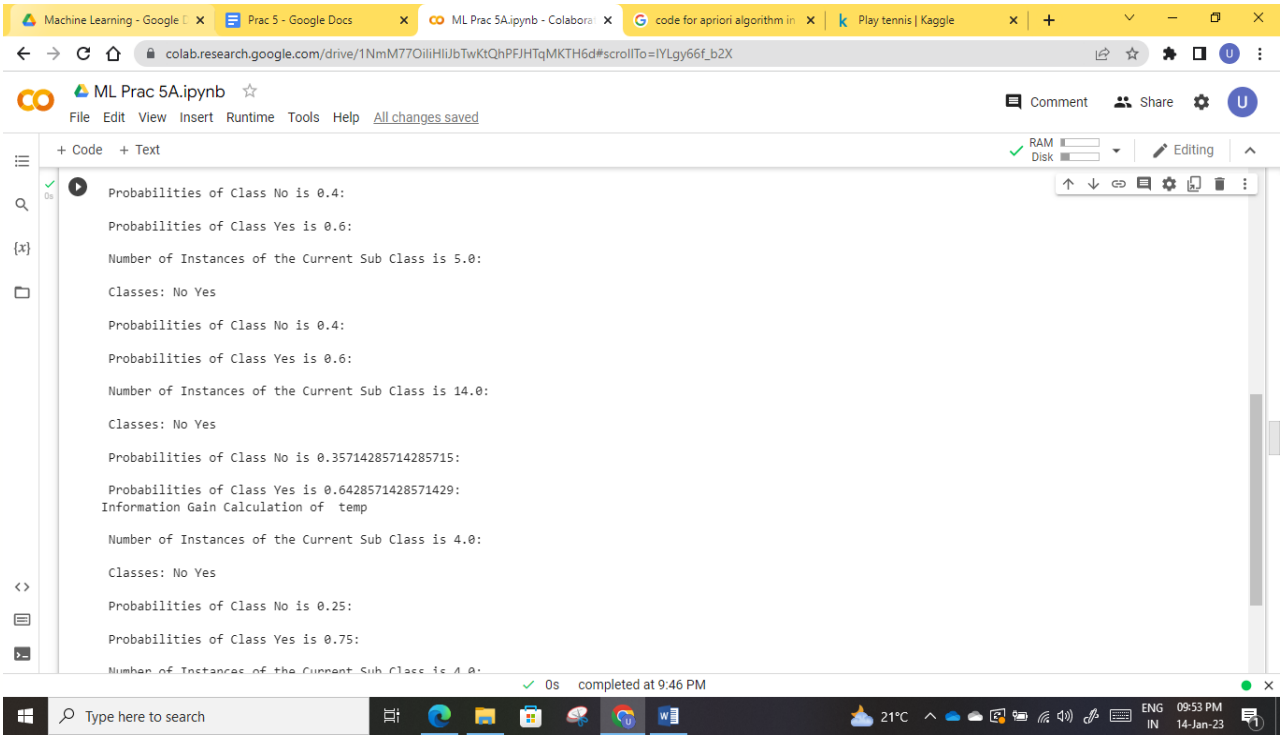

Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHIiJbTwKtQhPFJHTqMKTH6d#scrollTo=IYLgy66f_b2X
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[8]
List of Attributes: ['day', 'outlook', 'temp', 'humidity', 'wind', 'play']
Predicting Attributes: ['day', 'outlook', 'temp', 'humidity', 'wind']
# Run Algorithm:
from pprint import pprint
tree = id3(df_tennis,'play',attribute_names)
print("\n\nThe Resultant Decision Tree is :\n")
#print(tree)
pprint(tree)
attribute = next(iter(tree))
print("Best Attribute :\n",attribute)
print("Tree Keys:\n",tree[attribute].keys())
Information Gain Calculation of day
Number of Instances of the Current Sub Class is 1.0:
Classes: No No
Probabilities of Class No is 1.0:
Probabilities of Class No is 1.0:
Number of Instances of the Current Sub Class is 1.0:
Classes: Yes Yes
https://drive.google.com/drive/search?q=owner%3Ame (type%3Aapplication%2F...
0s completed at 9:46 PM
Type here to search
21°C
ENG IN
09:52 PM
14-Jan-23

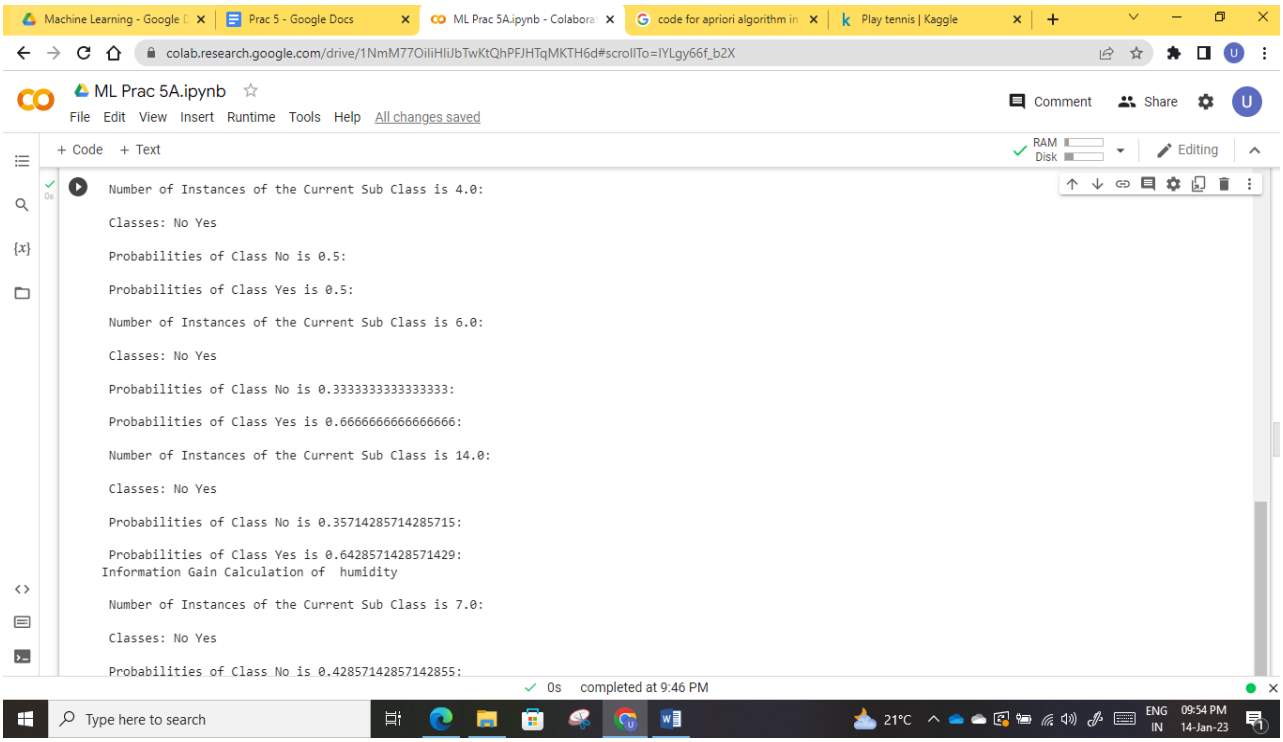

Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHIiJbTwKtQhPFJHTqMKTH6d#scrollTo=IYLgy66f_b2X
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
Probabilities of Class Yes is 1.0:
Probabilities of Class Yes is 1.0:
Number of Instances of the Current Sub Class is 1.0:
Classes: Yes Yes
Probabilities of Class Yes is 1.0:
Probabilities of Class Yes is 1.0:
Number of Instances of the Current Sub Class is 1.0:
Classes: Yes Yes
Probabilities of Class Yes is 1.0:
Probabilities of Class Yes is 1.0:
Number of Instances of the Current Sub Class is 1.0:
Classes: Yes Yes
Probabilities of Class Yes is 1.0:
Probabilities of Class Yes is 1.0:
Number of Instances of the Current Sub Class is 1.0:
0s completed at 9:46 PM
Type here to search
21°C
ENG
IN
09:52 PM
14-Jan-23

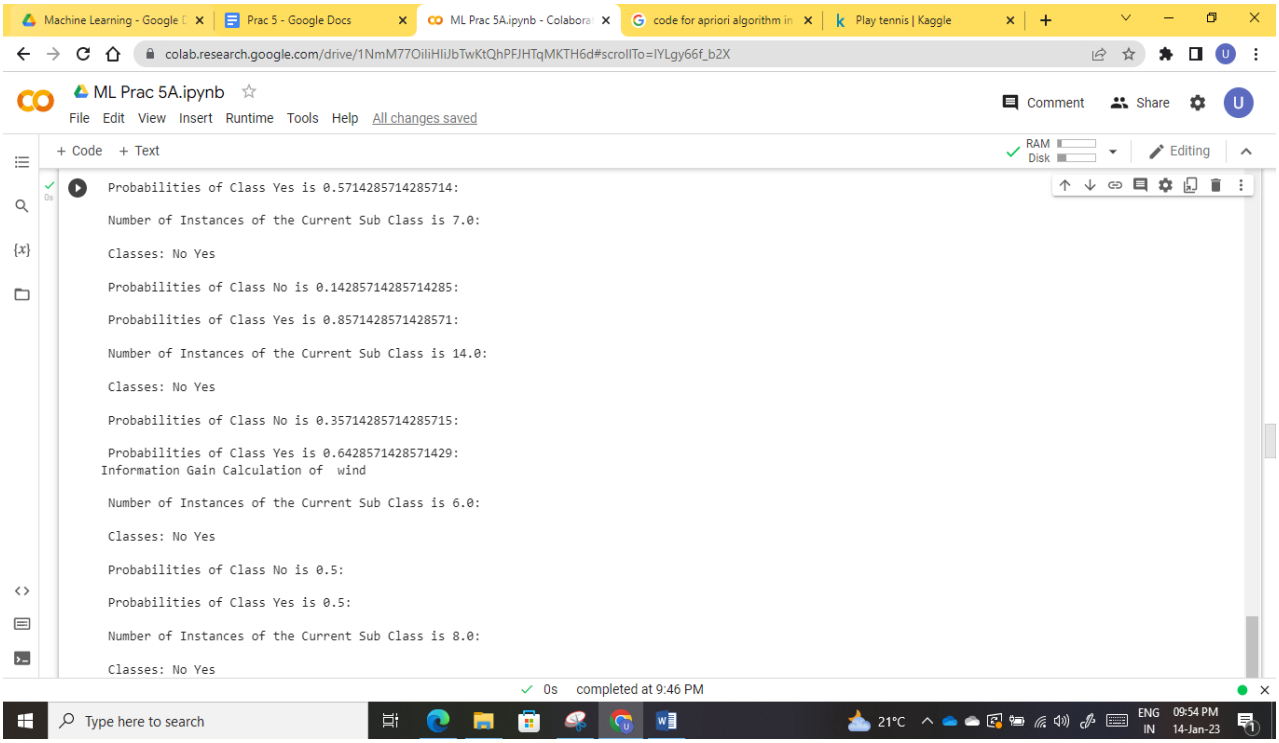

Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHIiJbTwKtQhPFJHTqMKTH6d#scrollTo=IYLgy66f_b2X
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
Classes: No No
Probabilities of Class No is 1.0:
Probabilities of Class No is 1.0:
Number of Instances of the Current Sub Class is 1.0:
Classes: No No
Probabilities of Class No is 1.0:
Probabilities of Class No is 1.0:
Number of Instances of the Current Sub Class is 1.0:
Classes: Yes Yes
Probabilities of Class Yes is 1.0:
Probabilities of Class Yes is 1.0:
Number of Instances of the Current Sub Class is 1.0:
Classes: Yes Yes
Probabilities of Class Yes is 1.0:
Probabilities of Class Yes is 1.0:
0s completed at 9:46 PM
Type here to search
21°C
ENG
IN
09:52 PM
14-Jan-23

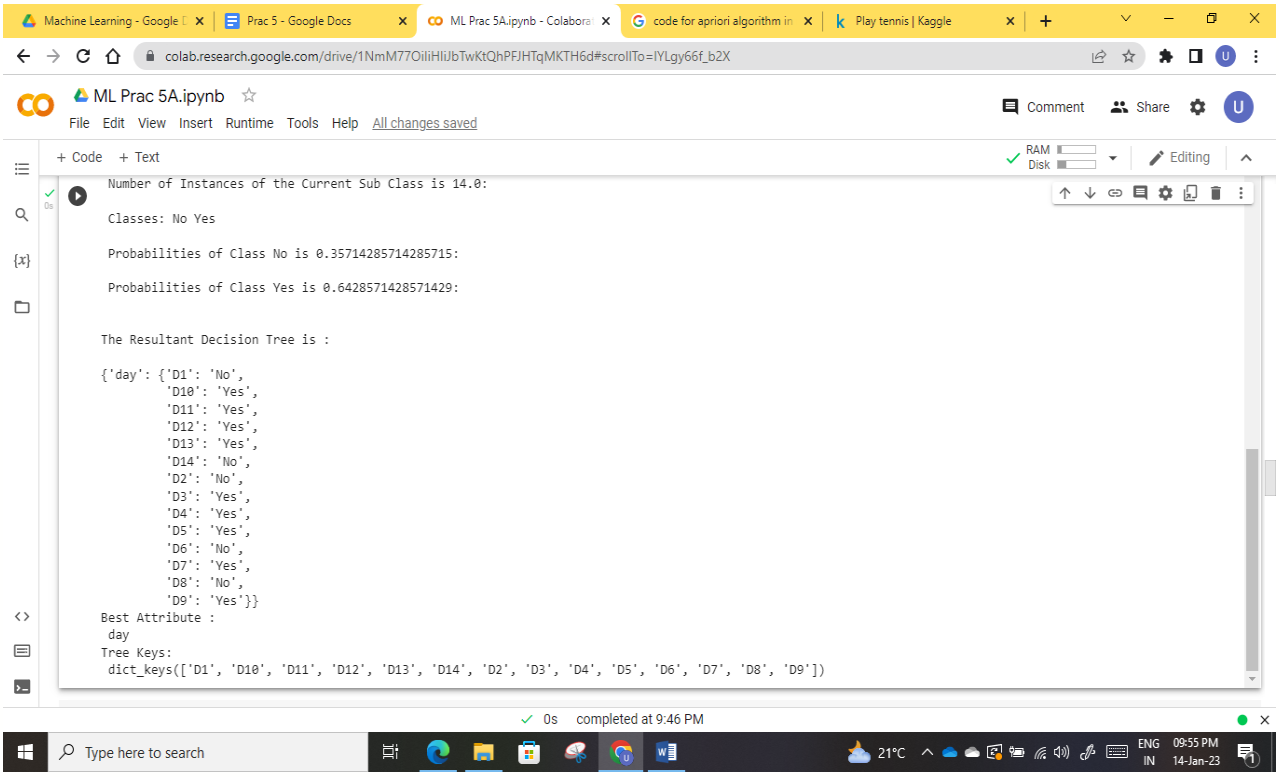

Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHIiJbTwKtQhPFJHTqMKTH6d#scrollTo=IYLgy66f_b2X
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
Number of Instances of the Current Sub Class is 1.0:
Classes: Yes Yes
Probabilities of Class Yes is 1.0:
Probabilities of Class Yes is 1.0:
Number of Instances of the Current Sub Class is 1.0:
Classes: No No
Probabilities of Class No is 1.0:
Probabilities of Class No is 1.0:
Number of Instances of the Current Sub Class is 1.0:
Classes: Yes Yes
Probabilities of Class Yes is 1.0:
Probabilities of Class Yes is 1.0:
Number of Instances of the Current Sub Class is 1.0:
Classes: No No
Probabilities of Class No is 1.0:
0s completed at 9:46 PM
Type here to search
21°C
ENG IN
09:53 PM
14-Jan-23

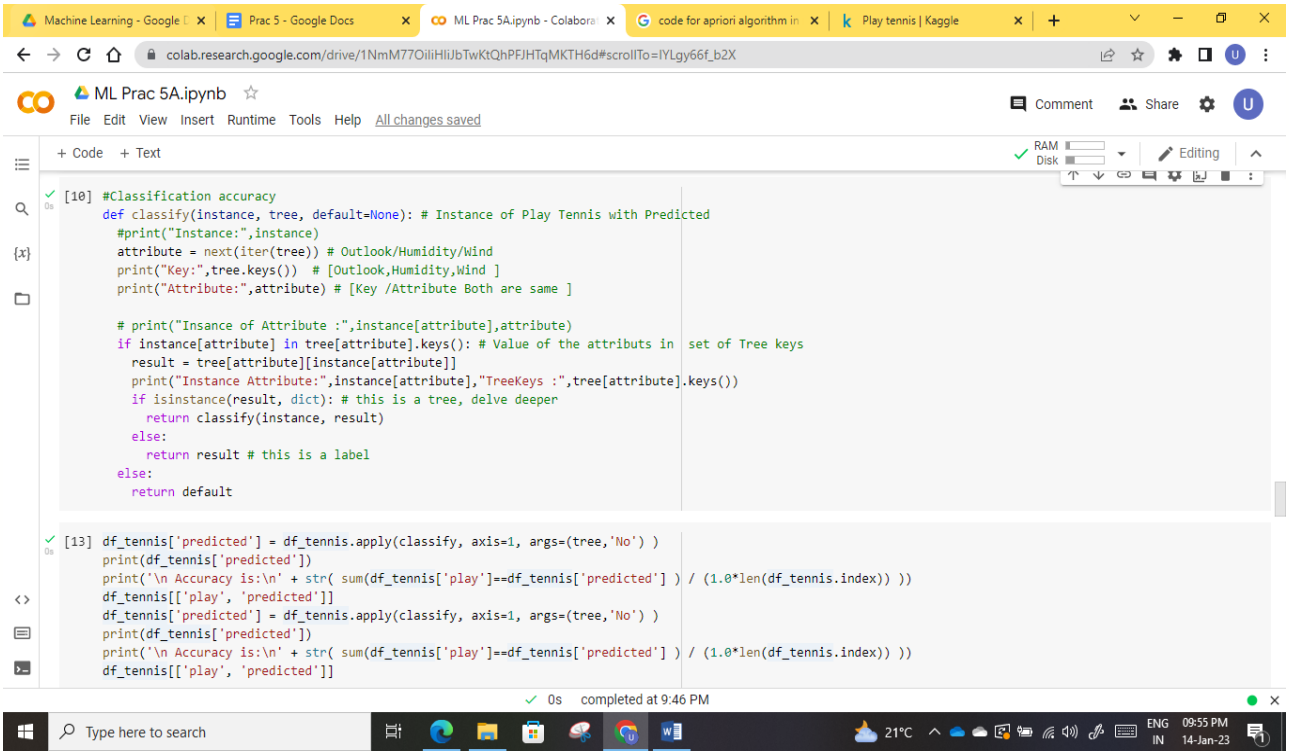

Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHIiJbTwKtQhPFJHTqMKTH6d#scrollTo=IYLgy66f_b2X
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
Probabilities of Class No is 1.0:
Number of Instances of the Current Sub Class is 1.0:
Classes: Yes Yes
Probabilities of Class Yes is 1.0:
Probabilities of Class Yes is 1.0:
Number of Instances of the Current Sub Class is 14.0:
Classes: No Yes
Probabilities of Class No is 0.35714285714285715:
Probabilities of Class Yes is 0.6428571428571429:
Information Gain Calculation of outlook
Number of Instances of the Current Sub Class is 4.0:
Classes: Yes Yes
Probabilities of Class Yes is 1.0:
Probabilities of Class Yes is 1.0:
Number of Instances of the Current Sub Class is 5.0:
Classes: No Yes
0s completed at 9:46 PM
Type here to search
21°C
ENG IN
09:53 PM
14-Jan-23

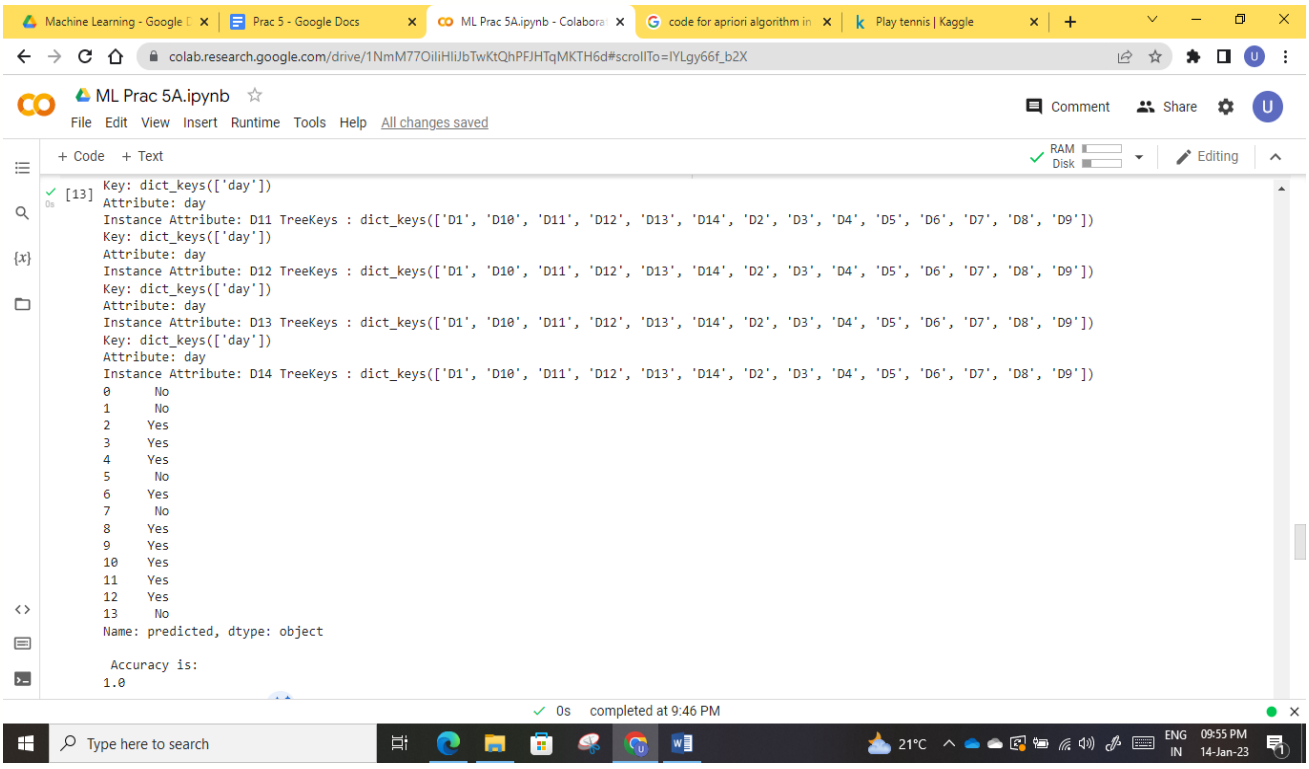

Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHIiJbTwKtQhPFJHTqMKTH6d#scrollTo=IYLgy66f_b2X
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
Probabilities of Class No is 0.4:
Probabilities of Class Yes is 0.6:
Number of Instances of the Current Sub Class is 5.0:
Classes: No Yes
Probabilities of Class No is 0.4:
Probabilities of Class Yes is 0.6:
Number of Instances of the Current Sub Class is 14.0:
Classes: No Yes
Probabilities of Class No is 0.35714285714285715:
Probabilities of Class Yes is 0.6428571428571429:
Information Gain Calculation of temp
Number of Instances of the Current Sub Class is 4.0:
Classes: No Yes
Probabilities of Class No is 0.25:
Probabilities of Class Yes is 0.75:
0s completed at 9:46 PM
Type here to search
21°C
ENG IN
09:53 PM 14-Jan-23

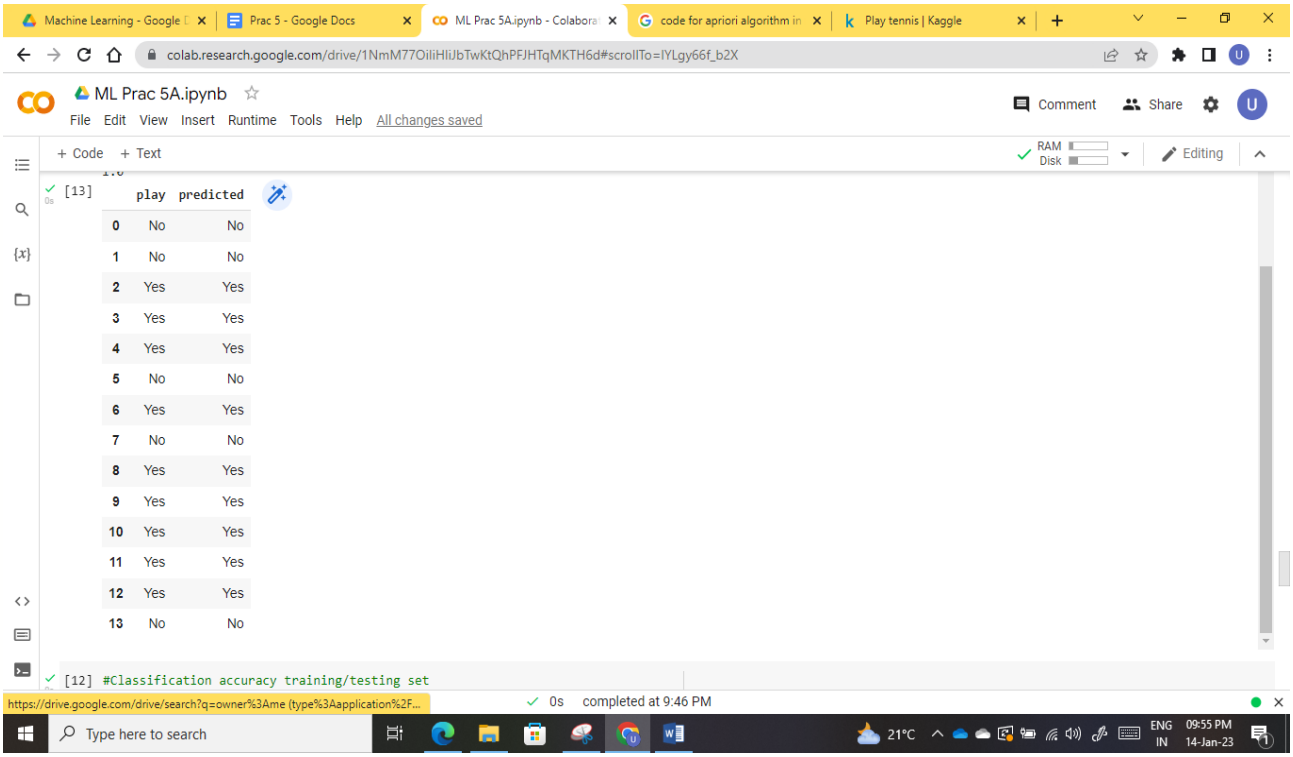

Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHIiJbTwKtQhPFJHTqMKTH6d#scrollTo=IYLgy66f_b2X
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
Number of Instances of the Current Sub Class is 4.0:
Classes: No Yes
Probabilities of Class No is 0.5:
Probabilities of Class Yes is 0.5:
Number of Instances of the Current Sub Class is 6.0:
Classes: No Yes
Probabilities of Class No is 0.3333333333333333:
Probabilities of Class Yes is 0.6666666666666666:
Number of Instances of the Current Sub Class is 14.0:
Classes: No Yes
Probabilities of Class No is 0.35714285714285715:
Probabilities of Class Yes is 0.6428571428571429:
Information Gain Calculation of humidity
Number of Instances of the Current Sub Class is 7.0:
Classes: No Yes
Probabilities of Class No is 0.42857142857142855:
0s completed at 9:46 PM
Type here to search
21°C
ENG IN
09:54 PM 14-Jan-23

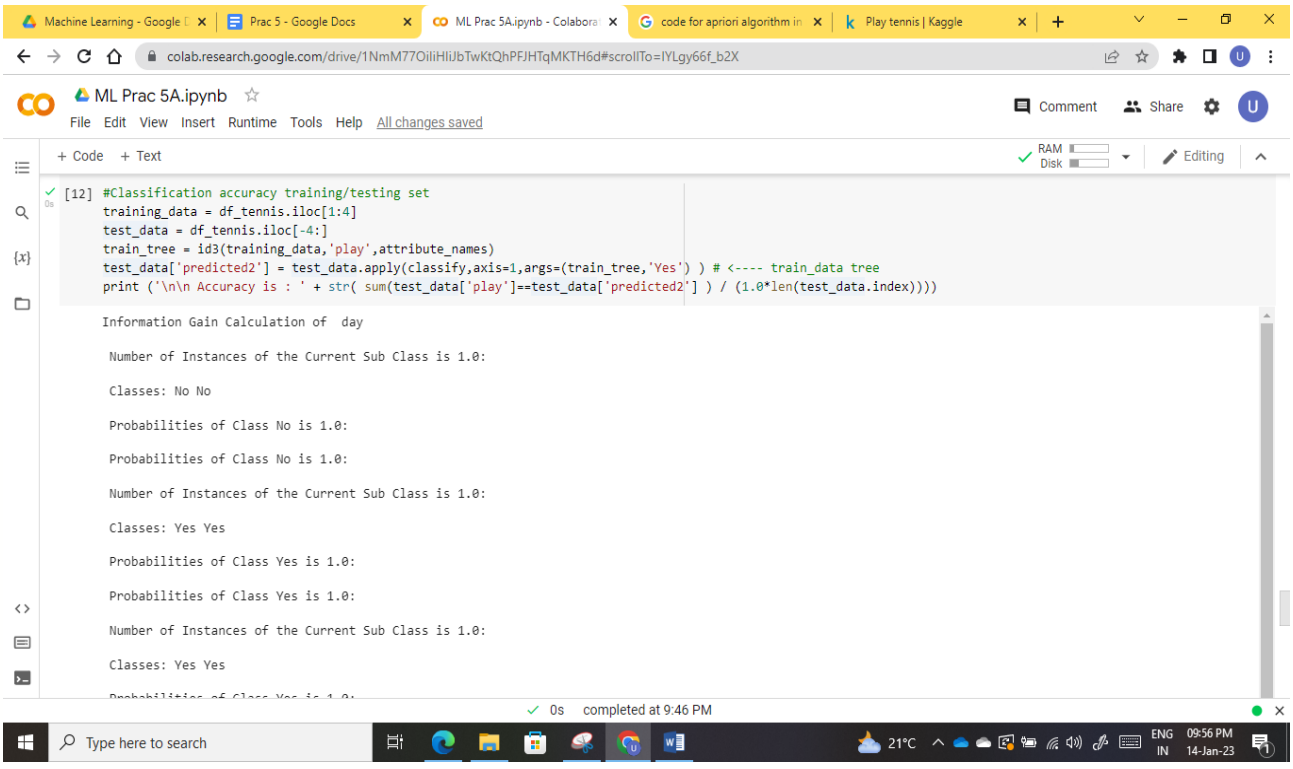

Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHIiJbTwKtQhPFJHTqMKTH6d#scrollTo=IYLgy66f_b2X
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
Probabilities of Class Yes is 0.5714285714285714:
Number of Instances of the Current Sub Class is 7.0:
Classes: No Yes
Probabilities of Class No is 0.14285714285714285:
Probabilities of Class Yes is 0.8571428571428571:
Number of Instances of the Current Sub Class is 14.0:
Classes: No Yes
Probabilities of Class No is 0.35714285714285715:
Probabilities of Class Yes is 0.6428571428571429:
Information Gain Calculation of wind
Number of Instances of the Current Sub Class is 6.0:
Classes: No Yes
Probabilities of Class No is 0.5:
Probabilities of Class Yes is 0.5:
Number of Instances of the Current Sub Class is 8.0:
Classes: No Yes
0s completed at 9:46 PM
Type here to search
21°C
ENG IN
09:54 PM
14-Jan-23

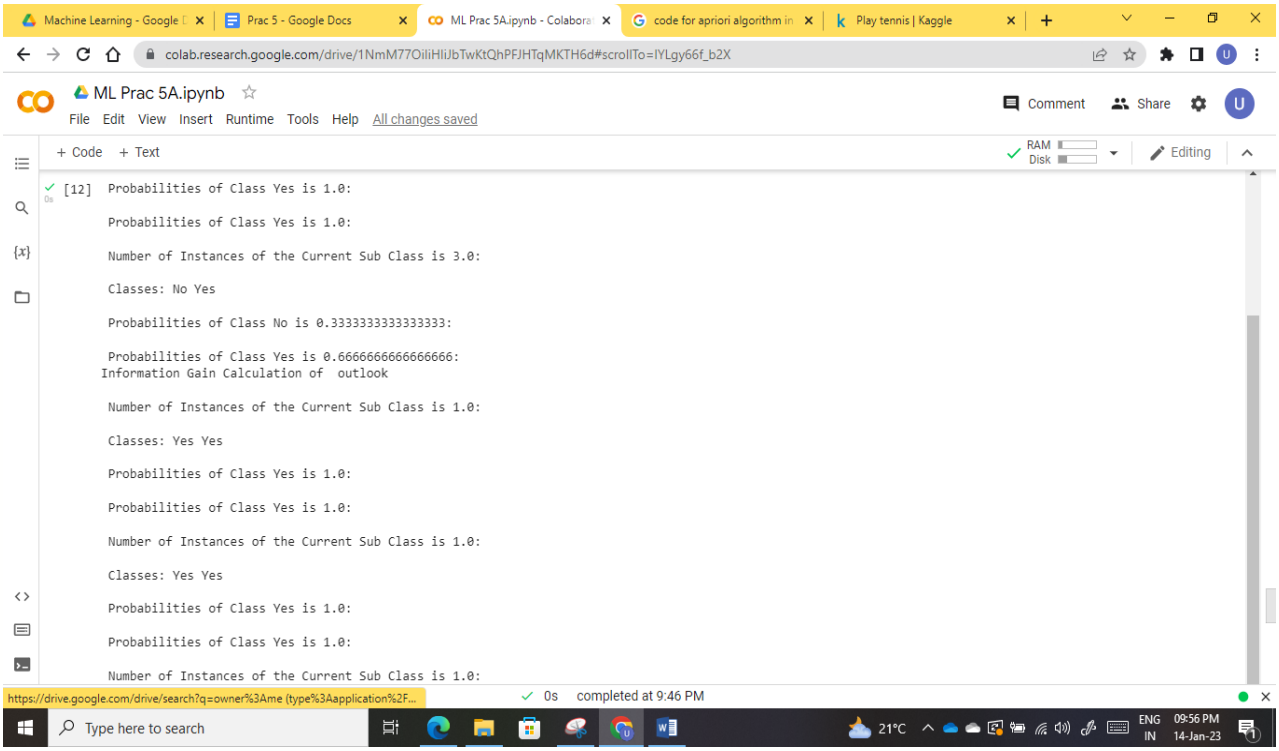

Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHIiJbTwKtQhPFJHTqMKTH6d#scrollTo=IYLgy66f_b2X
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
Number of Instances of the Current Sub Class is 14.0:
Classes: No Yes
Probabilities of Class No is 0.35714285714285715:
Probabilities of Class Yes is 0.6428571428571429:
The Resultant Decision Tree is :
{'day': {'D1': 'No',
'D10': 'Yes',
'D11': 'Yes',
'D12': 'Yes',
'D13': 'Yes',
'D14': 'No',
'D2': 'No',
'D3': 'Yes',
'D4': 'Yes',
'D5': 'Yes',
'D6': 'No',
'D7': 'Yes',
'D8': 'No',
'D9': 'Yes'}}
Best Attribute :
day
Tree Keys:
dict_keys(['D1', 'D10', 'D11', 'D12', 'D13', 'D14', 'D2', 'D3', 'D4', 'D5', 'D6', 'D7', 'D8', 'D9'])
0s completed at 9:46 PM
Type here to search
21°C
ENG IN
09:55 PM
14-Jan-23

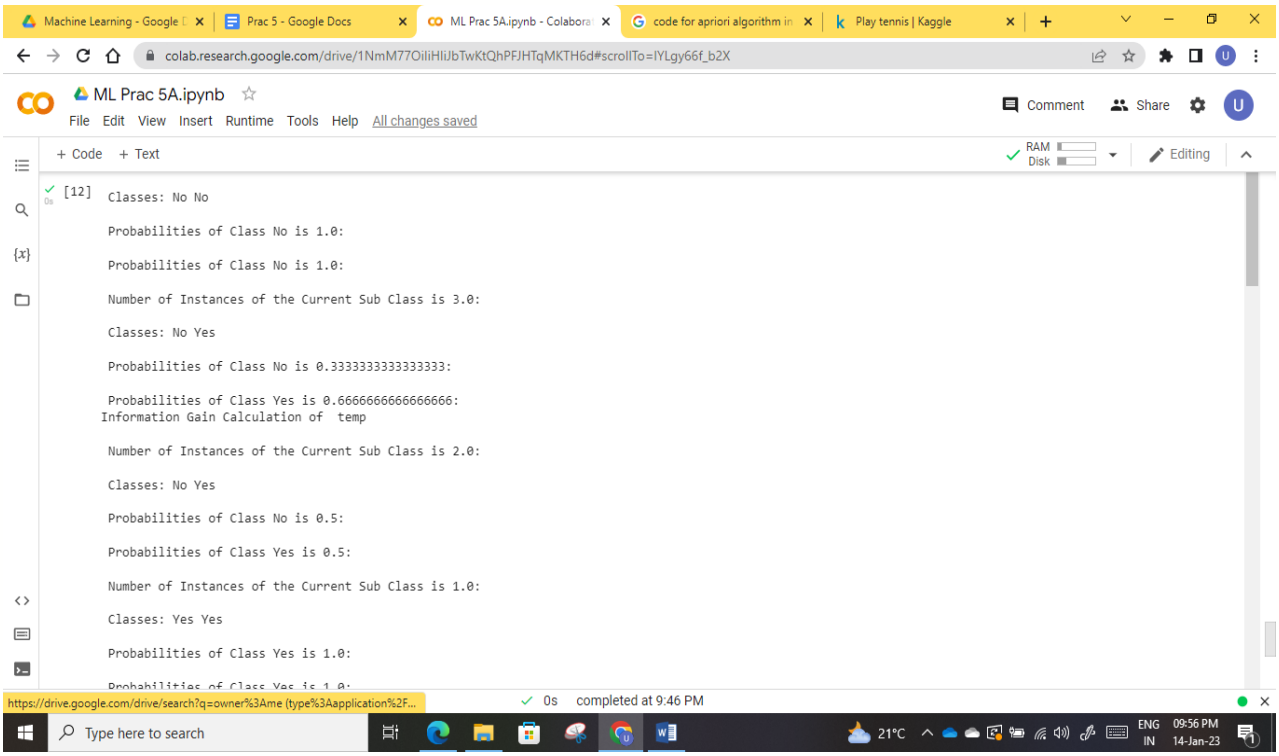

Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHIiJbTwKtQhPFJHTqMKTH6d#scrollTo=IYLgy66f_b2X
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[10] #Classification accuracy
def classify(instance, tree, default=None): # Instance of Play Tennis with Predicted
#print("Instance:",instance)
attribute = next(iter(tree)) # Outlook/Humidity/Wind
print("Key:",tree.keys()) # [Outlook,Humidity,Wind]
print("Attribute:",attribute) # [Key /Attribute Both are same]
print("Insance of Attribute :",instance[attribute],attribute)
if instance[attribute] in tree[attribute].keys(): # Value of the attributs in set of Tree keys
result = tree[attribute][instance[attribute]]
print("Instance Attribute:",instance[attribute],"TreeKeys :",tree[attribute].keys())
if isinstance(result, dict): # this is a tree, delve deeper
return classify(instance, result)
else:
return result # this is a label
else:
return default
[13] df_tennis['predicted'] = df_tennis.apply(classify, axis=1, args=(tree,'No'))
print(df_tennis['predicted'])
print('\n Accuracy is:\n' + str(sum(df_tennis['play']==df_tennis['predicted']) / (1.0*len(df_tennis.index))))
df_tennis[['play', 'predicted']]
df_tennis['predicted'] = df_tennis.apply(classify, axis=1, args=(tree,'No'))
print(df_tennis['predicted'])
print('\n Accuracy is:\n' + str(sum(df_tennis['play']==df_tennis['predicted']) / (1.0*len(df_tennis.index))))
df_tennis[['play', 'predicted']]
0s completed at 9:46 PM
Type here to search
21°C
ENG IN
09:55 PM 14-Jan-23

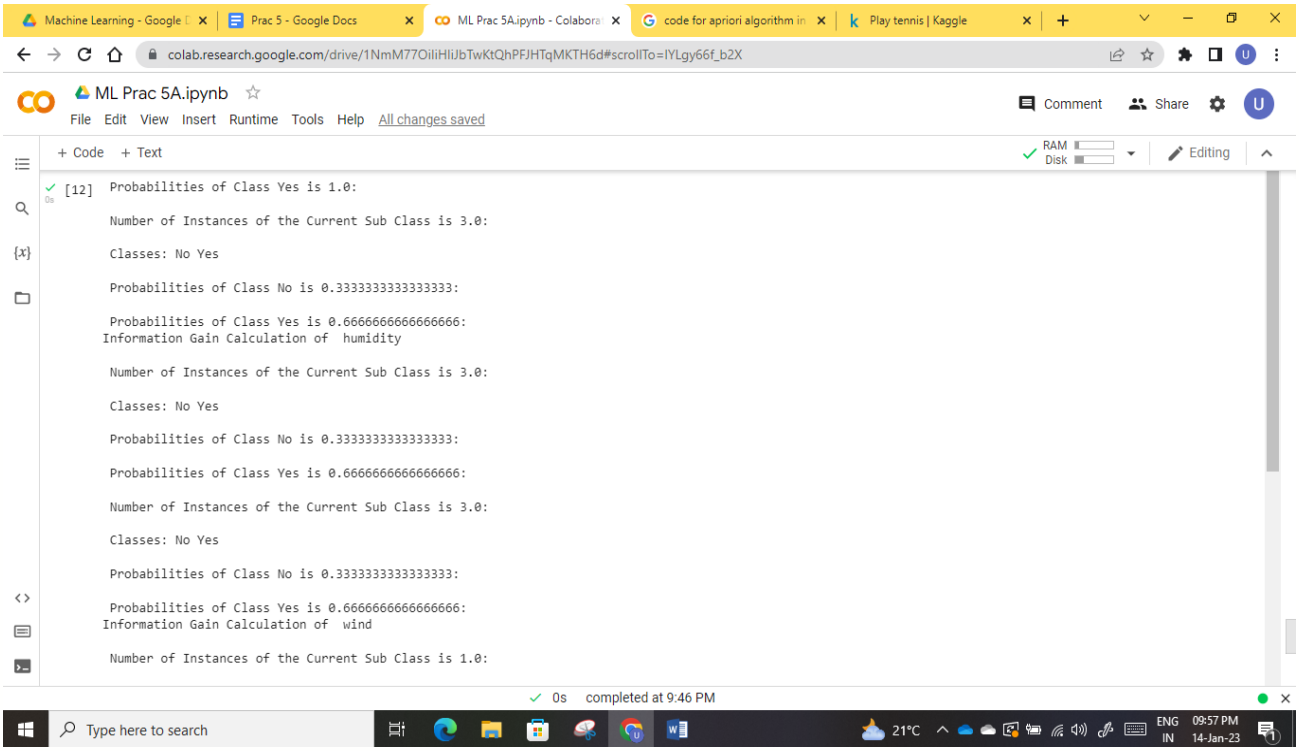

Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHIiJbTwKtQhPFJHTqMKTH6d#scrollTo=IYLgy66f_b2X
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[13] Key: dict_keys(['day'])
Attribute: day
Instance Attribute: D11 TreeKeys : dict_keys(['D1', 'D10', 'D11', 'D12', 'D13', 'D14', 'D2', 'D3', 'D4', 'D5', 'D6', 'D7', 'D8', 'D9'])
Key: dict_keys(['day'])
Attribute: day
Instance Attribute: D12 TreeKeys : dict_keys(['D1', 'D10', 'D11', 'D12', 'D13', 'D14', 'D2', 'D3', 'D4', 'D5', 'D6', 'D7', 'D8', 'D9'])
Key: dict_keys(['day'])
Attribute: day
Instance Attribute: D13 TreeKeys : dict_keys(['D1', 'D10', 'D11', 'D12', 'D13', 'D14', 'D2', 'D3', 'D4', 'D5', 'D6', 'D7', 'D8', 'D9'])
Key: dict_keys(['day'])
Attribute: day
Instance Attribute: D14 TreeKeys : dict_keys(['D1', 'D10', 'D11', 'D12', 'D13', 'D14', 'D2', 'D3', 'D4', 'D5', 'D6', 'D7', 'D8', 'D9'])
0 No
1 No
2 Yes
3 Yes
4 Yes
5 No
6 Yes
7 No
8 Yes
9 Yes
10 Yes
11 Yes
12 Yes
13 No
Name: predicted, dtype: object
Accuracy is:
1.0
0s completed at 9:46 PM
Type here to search
21°C
ENG IN
09:55 PM 14-Jan-23

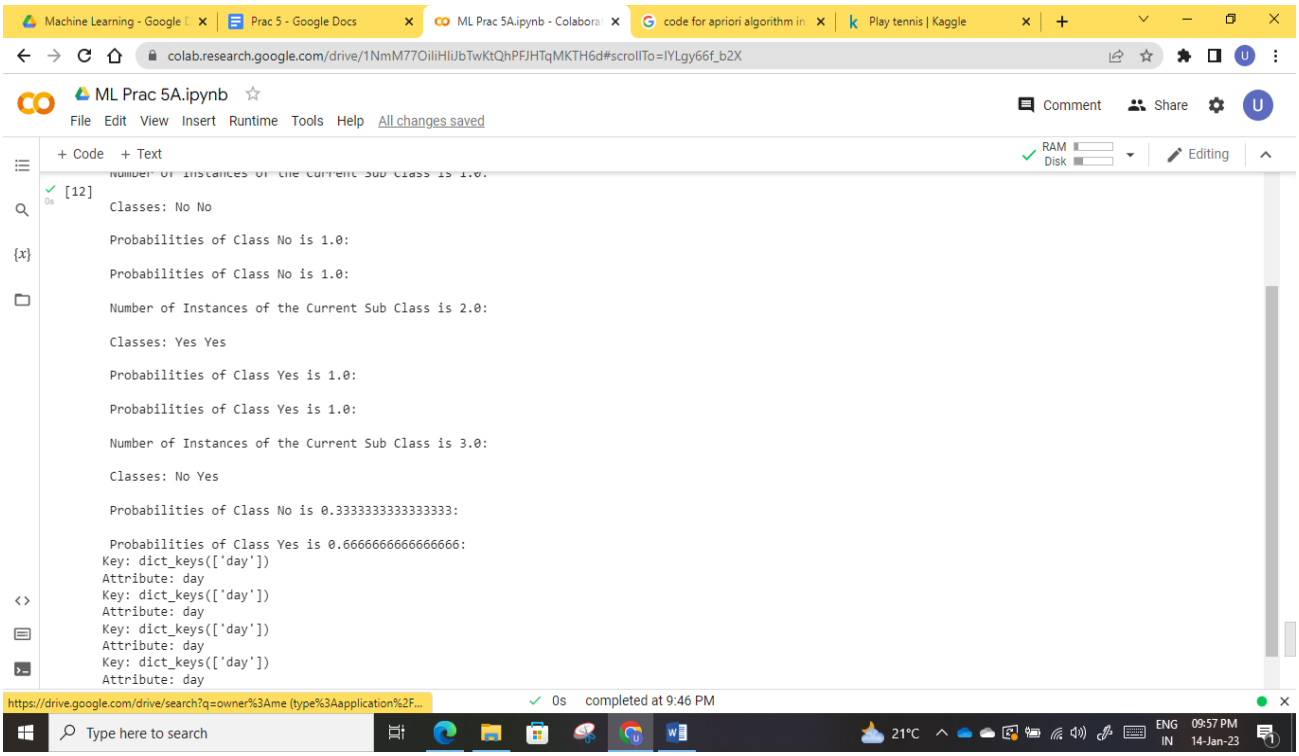

Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHIiJbTwKtQhPFJHTqMKTH6d#scrollTo=IYLgy66f_b2X
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[13]
play predicted
0 No No
1 No No
2 Yes Yes
3 Yes Yes
4 Yes Yes
5 No No
6 Yes Yes
7 No No
8 Yes Yes
9 Yes Yes
10 Yes Yes
11 Yes Yes
12 Yes Yes
13 No No
[12] #Classification accuracy training/testing set
0s completed at 9:46 PM
Type here to search
21°C
ENG IN
09:55 PM
14-Jan-23

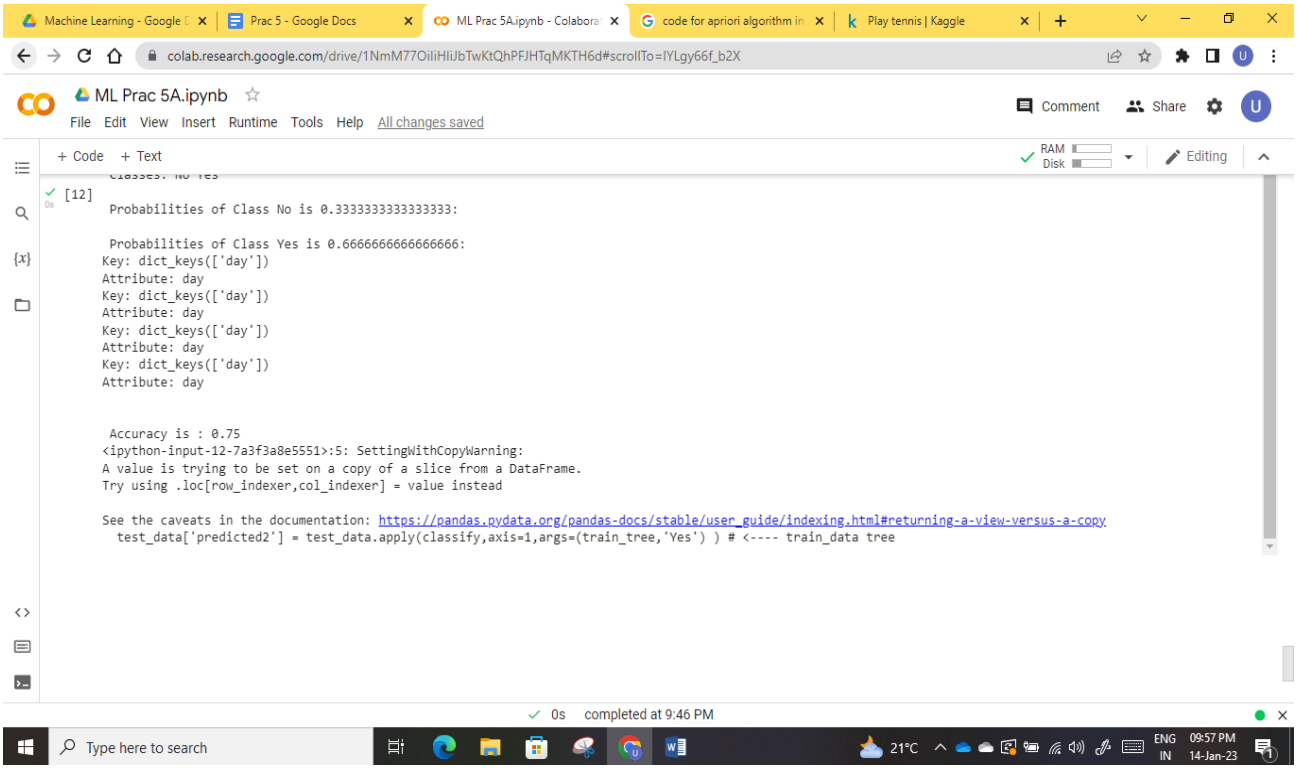

Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHIiJbTwKtQhPFJHTqMKTH6d#scrollTo=IYLgy66f_b2X
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[12] #Classification accuracy training/testing set
training_data = df_tennis.iloc[1:4]
test_data = df_tennis.iloc[-4:]
train_tree = id3(training_data,'play',attribute_names)
test_data['predicted2'] = test_data.apply(classify,axis=1,args=(train_tree,'Yes')) # <---- train_data tree
print ('\n\n Accuracy is : ' + str(sum(test_data['play']==test_data['predicted2']) / (1.0*len(test_data.index))))
Information Gain Calculation of day
Number of Instances of the Current Sub Class is 1.0:
Classes: No No
Probabilities of Class No is 1.0:
Probabilities of Class No is 1.0:
Number of Instances of the Current Sub Class is 1.0:
Classes: Yes Yes
Probabilities of Class Yes is 1.0:
Probabilities of Class Yes is 1.0:
Number of Instances of the Current Sub Class is 1.0:
Classes: Yes Yes
0s completed at 9:46 PM
Type here to search
21°C
ENG IN
09:56 PM
14-Jan-23


Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHIiJbTwKtQhPFJHTqMKTH6d#scrollTo=IYLgy66f_b2X
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[12] Probabilities of Class Yes is 1.0:
Probabilities of Class Yes is 1.0:
Number of Instances of the Current Sub Class is 3.0:
Classes: No Yes
Probabilities of Class No is 0.3333333333333333:
Probabilities of Class Yes is 0.6666666666666666:
Information Gain Calculation of outlook
Number of Instances of the Current Sub Class is 1.0:
Classes: Yes Yes
Probabilities of Class Yes is 1.0:
Probabilities of Class Yes is 1.0:
Number of Instances of the Current Sub Class is 1.0:
Classes: Yes Yes
Probabilities of Class Yes is 1.0:
Probabilities of Class Yes is 1.0:
Number of Instances of the Current Sub Class is 1.0:
https://drive.google.com/drive/search?q=owner%3Ame (type%3Aapplication%2F...
0s completed at 9:46 PM
Type here to search
21°C
ENG IN
09:56 PM 14-Jan-23


Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHIiJbTwKtQhPFJHTqMKTH6d#scrollTo=IYLgy66f_b2X
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[12] Classes: No No
Probabilities of Class No is 1.0:
Probabilities of Class No is 1.0:
Number of Instances of the Current Sub Class is 3.0:
Classes: No Yes
Probabilities of Class No is 0.3333333333333333:
Probabilities of Class Yes is 0.6666666666666666:
Information Gain Calculation of temp
Number of Instances of the Current Sub Class is 2.0:
Classes: No Yes
Probabilities of Class No is 0.5:
Probabilities of Class Yes is 0.5:
Number of Instances of the Current Sub Class is 1.0:
Classes: Yes Yes
Probabilities of Class Yes is 1.0:
https://drive.google.com/drive/search?q=owner%3Ame (type%3Aapplication%2F...
0s completed at 9:46 PM
Type here to search
21°C
ENG IN
09:56 PM 14-Jan-23


Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHliJbTwKtQhPFJHTqMKTH6d#scrollTo=IYLgy66f_b2X
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[12] Probabilities of Class Yes is 1.0:
Number of Instances of the Current Sub Class is 3.0:
Classes: No Yes
Probabilities of Class No is 0.3333333333333333:
Probabilities of Class Yes is 0.6666666666666666:
Information Gain Calculation of humidity
Number of Instances of the Current Sub Class is 3.0:
Classes: No Yes
Probabilities of Class No is 0.3333333333333333:
Probabilities of Class Yes is 0.6666666666666666:
Number of Instances of the Current Sub Class is 3.0:
Classes: No Yes
Probabilities of Class No is 0.3333333333333333:
Probabilities of Class Yes is 0.6666666666666666:
Information Gain Calculation of wind
Number of Instances of the Current Sub Class is 1.0:
0s completed at 9:46 PM
Type here to search
21°C
ENG IN
09:57 PM 14-Jan-23


Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHliJbTwKtQhPFJHTqMKTH6d#scrollTo=IYLgy66f_b2X
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[12]
Classes: No No
Probabilities of Class No is 1.0:
Probabilities of Class No is 1.0:
Number of Instances of the Current Sub Class is 2.0:
Classes: Yes Yes
Probabilities of Class Yes is 1.0:
Probabilities of Class Yes is 1.0:
Number of Instances of the Current Sub Class is 3.0:
Classes: No Yes
Probabilities of Class No is 0.3333333333333333:
Probabilities of Class Yes is 0.6666666666666666:
Key: dict_keys(['day'])
Attribute: day
Key: dict_keys(['day'])
Attribute: day
Key: dict_keys(['day'])
Attribute: day
Key: dict_keys(['day'])
Attribute: day
https://drive.google.com/drive/search?q=owner%3Ame (type%3Aapplication%2F...
0s completed at 9:46 PM
Type here to search
21°C
ENG IN
09:57 PM 14-Jan-23


Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5A.ipynb - Colabora
code for apriori algorithm in
Play tennis | Kaggle
colab.research.google.com/drive/1NmM77OiliHIiJbTwKtQhPFJHTqMKTH6d#scrollTo=IYLgy66f_b2X
ML Prac 5A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[12]
Probabilities of Class No is 0.3333333333333333:
Probabilities of Class Yes is 0.6666666666666666:
Key: dict_keys(['day'])
Attribute: day
Key: dict_keys(['day'])
Attribute: day
Key: dict_keys(['day'])
Attribute: day
Key: dict_keys(['day'])
Attribute: day
Accuracy is : 0.75
<ipython-input-12-7a3f3a8e5551>:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
test_data['predicted2'] = test_data.apply(classify,axis=1,args=(train_tree,'Yes')) # <---- train_data tree
0s completed at 9:46 PM
Type here to search
21°C
ENG
IN
09:57 PM
14-Jan-23

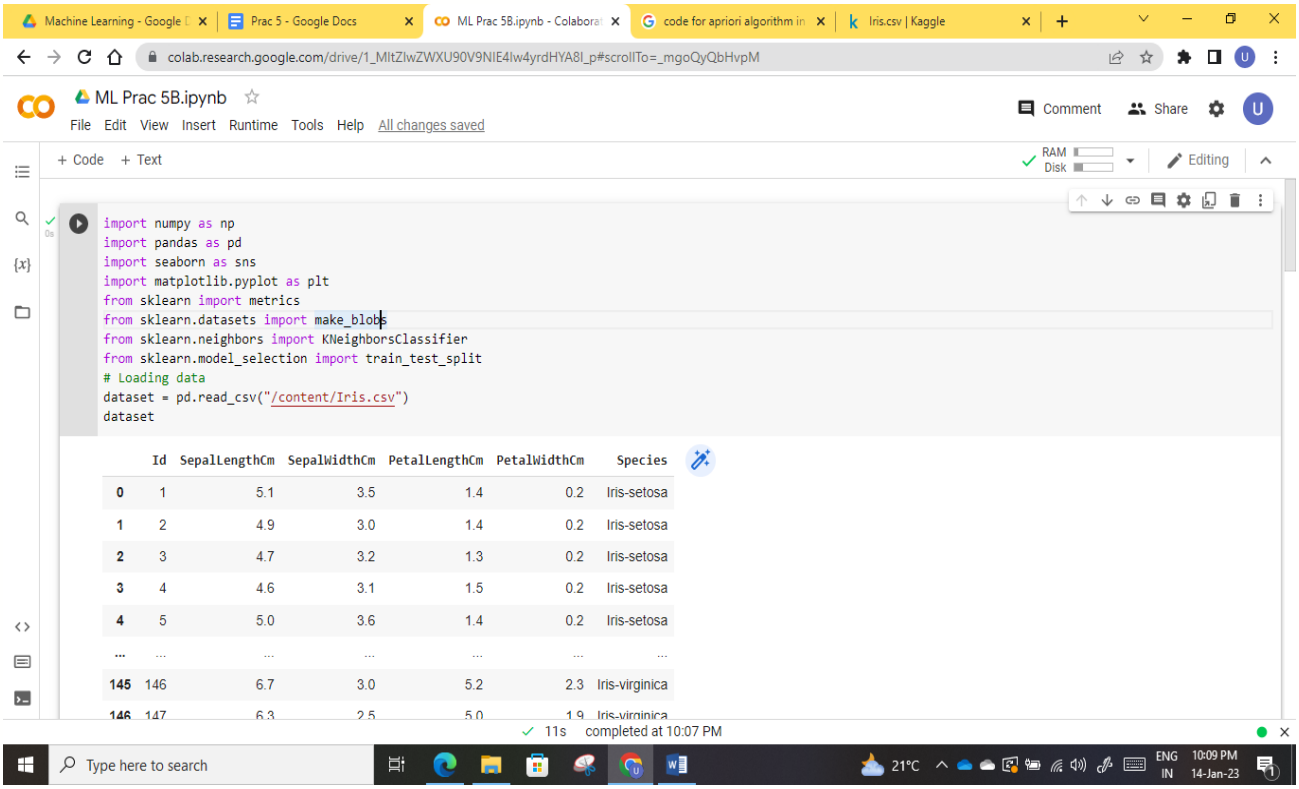
Write a program to implement k-Nearest Neighbour algorithm to classify the iris data set

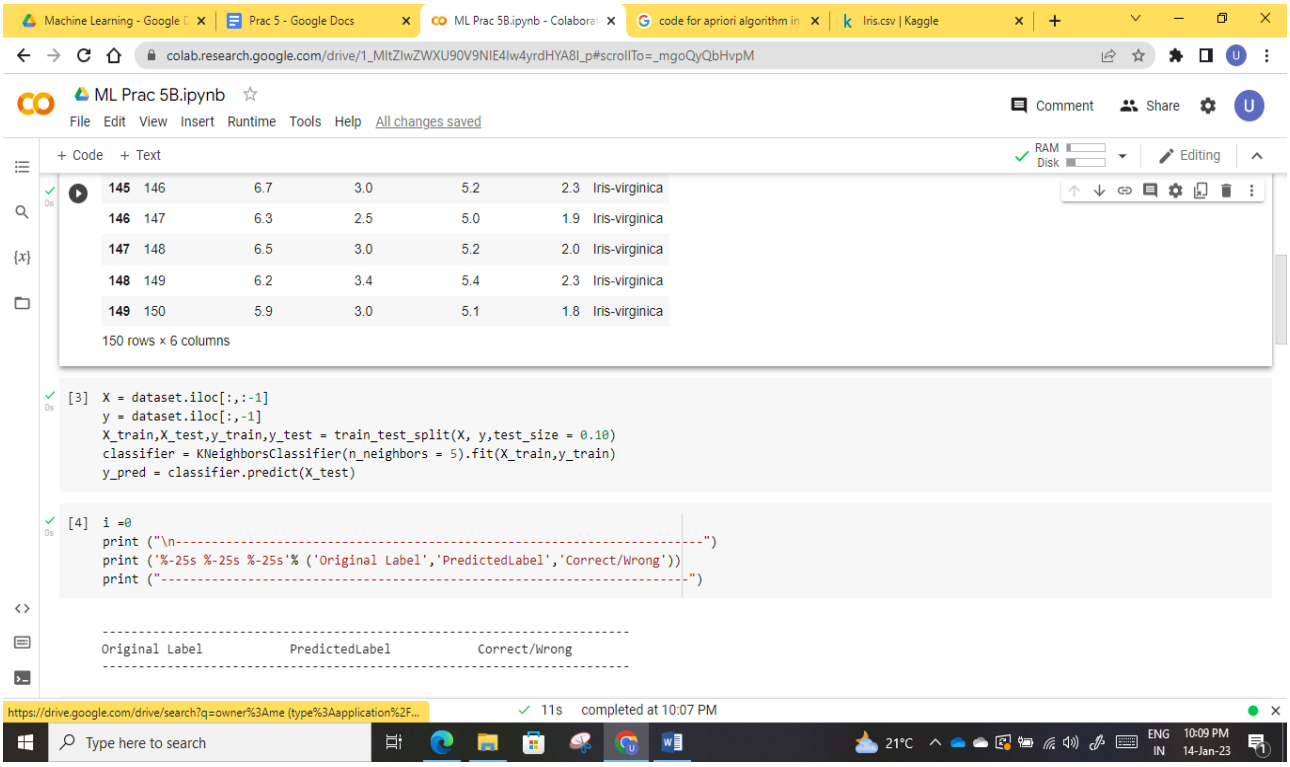
Code :

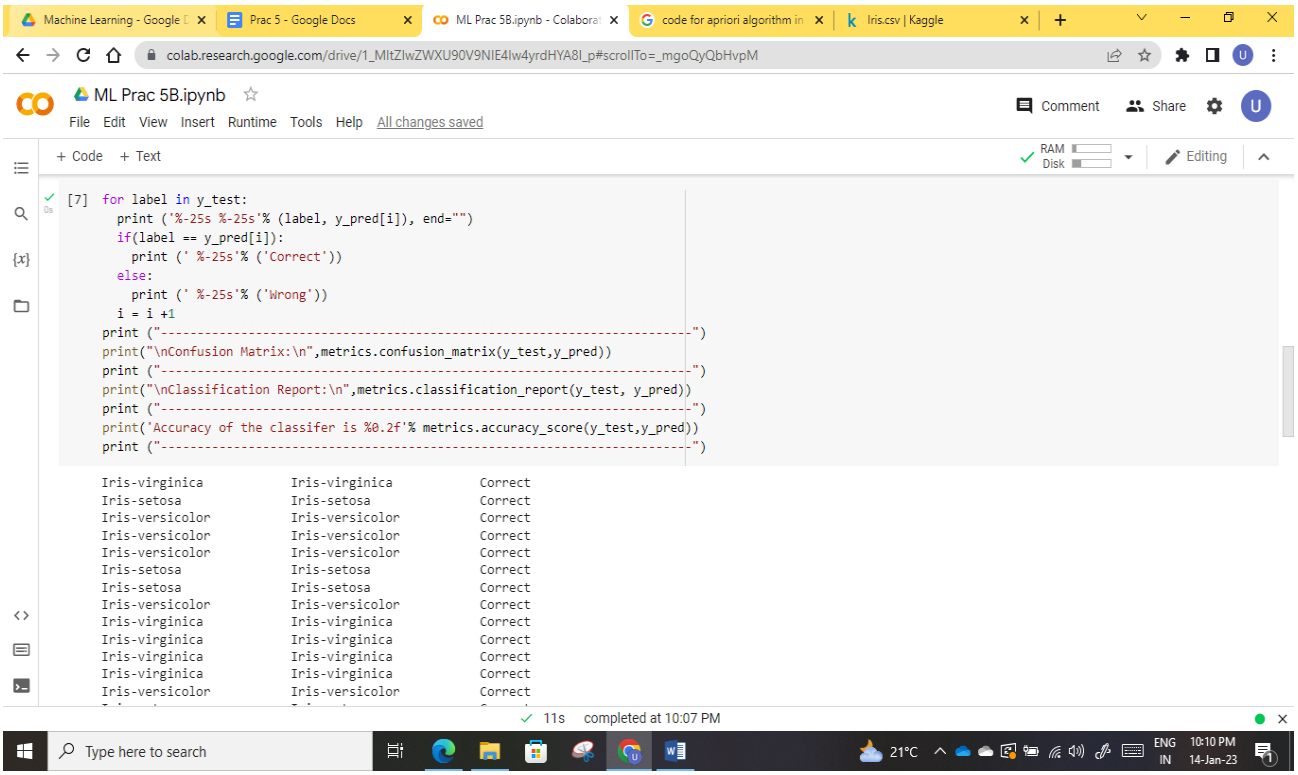
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import metrics
from sklearn.datasets import make_blobs
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
# Loading data
dataset = pd.read_csv("/content/Iris.csv")
dataset

X = dataset.iloc[:,:-1]
y = dataset.iloc[:,-1]
X_train,X_test,y_train,y_test = train_test_split(X, y,test_size = 0.10)
classifier = KNeighborsClassifier(n_neighbors = 5).fit(X_train,y_train)
y_pred = classifier.predict(X_test)

i =0
print ("\n------------------------------------------------------------------------")
print ('%-25s %-25s %-25s'% ('Original Label','PredictedLabel','Correct/Wrong'))
print ("------------------------------------------------------------------------")

for label in y_test:
        print ('%-25s %-25s'% (label, y_pred[i]), end="")
        if(label == y_pred[i]):
                print (' %-25s'% ('Correct'))
        else:
```

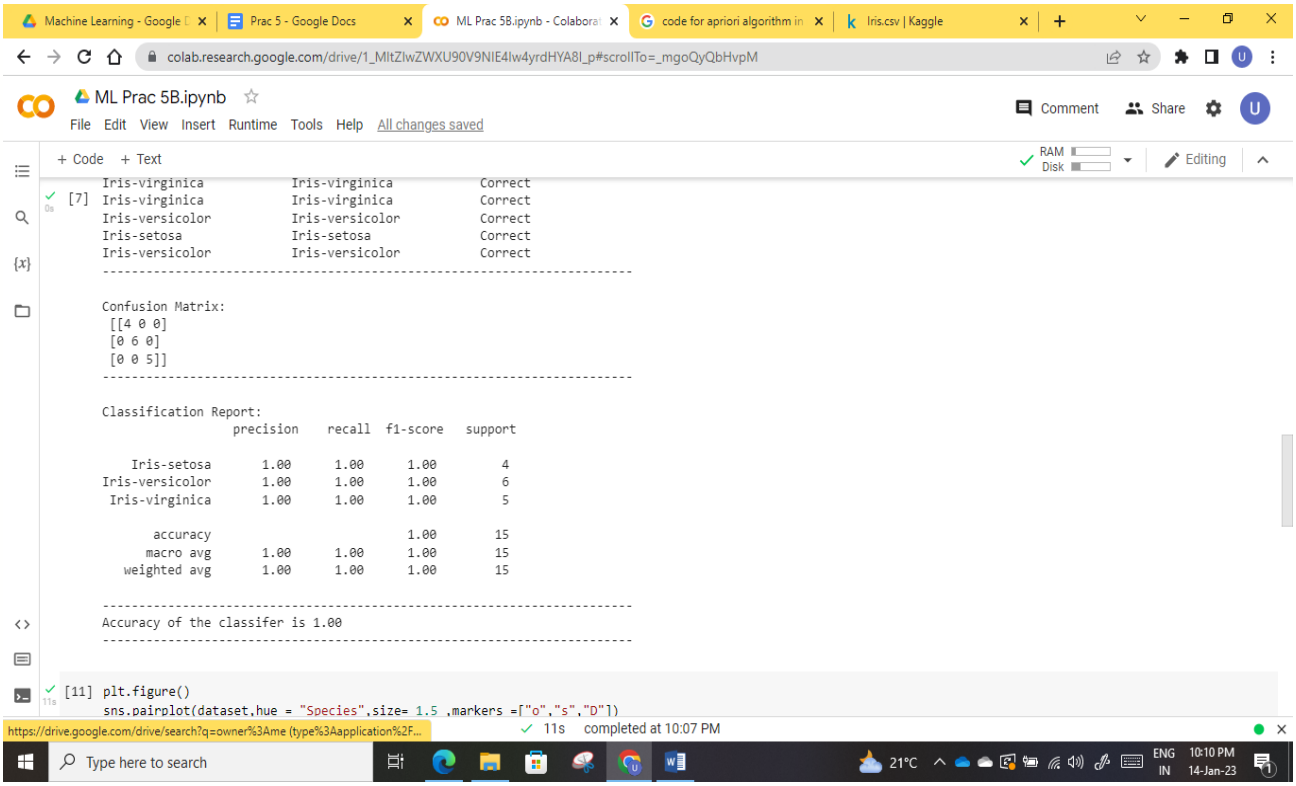
```
                print (' %-25s'% ('Wrong'))
        i = i +1
print ("----------------------------------------------------------------------")
print("\nConfusion Matrix:\n",metrics.confusion_matrix(y_test,y_pred))
print ("----------------------------------------------------------------------")
print("\nClassification Report:\n",metrics.classification_report(y_test, y_pred))
print ("----------------------------------------------------------------------")
print('Accuracy of the classifer is %0.2f'% metrics.accuracy_score(y_test,y_pred))
print ("----------------------------------------------------------------------")

plt.figure()
sns.pairplot(dataset,hue = "Species",size= 1.5 ,markers =["o","s","D"])
plt.show()
```

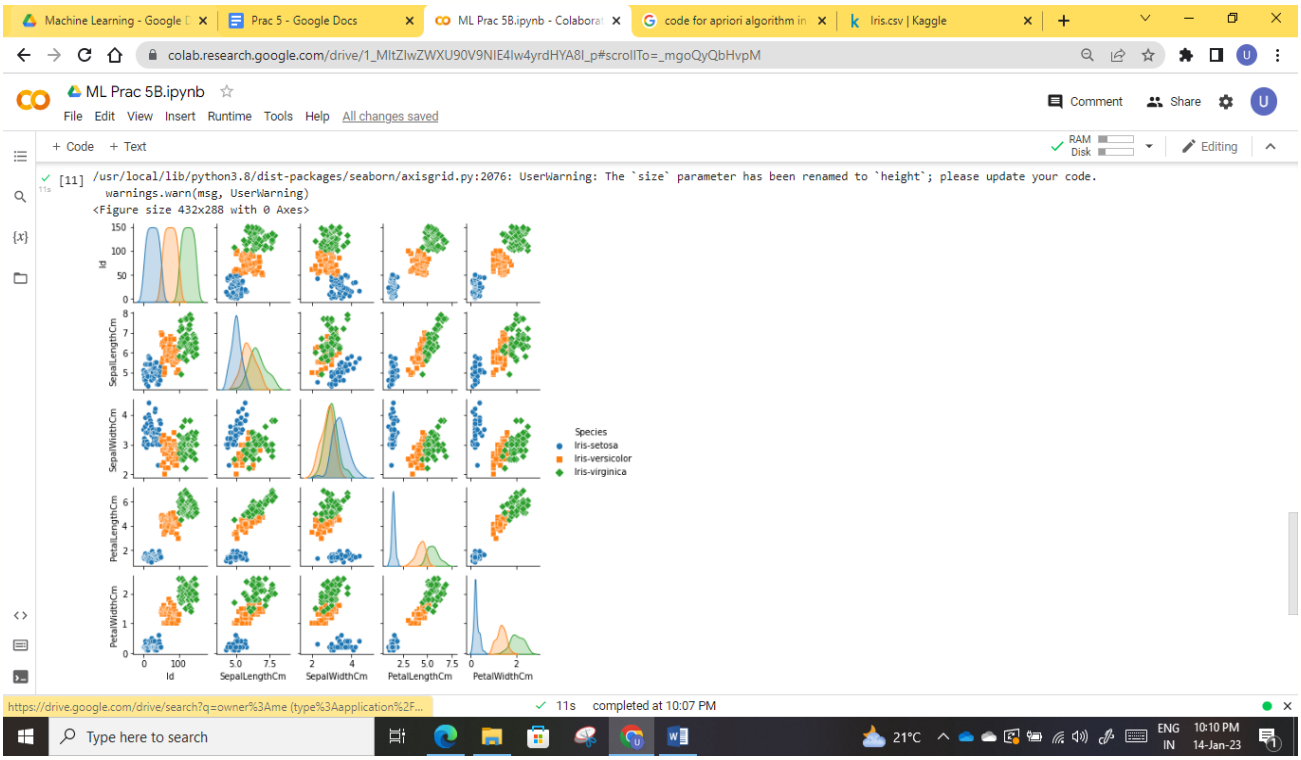
Output :




Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5B.ipynb - Colabora
code for apriori algorithm in
Iris.csv | Kaggle
colab.research.google.com/drive/1_MItZIwZWXU90V9NIE4Iw4yrdHYA8I_p#scrollTo=_mgoQyQbHvpM
ML Prac 5B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
145 146 6.7 3.0 5.2 2.3 Iris-virginica
146 147 6.3 2.5 5.0 1.9 Iris-virginica
147 148 6.5 3.0 5.2 2.0 Iris-virginica
148 149 6.2 3.4 5.4 2.3 Iris-virginica
149 150 5.9 3.0 5.1 1.8 Iris-virginica
150 rows × 6 columns
[3] X = dataset.iloc[:,:-1]
y = dataset.iloc[:,-1]
X_train,X_test,y_train,y_test = train_test_split(X, y,test_size = 0.10)
classifier = KNeighborsClassifier(n_neighbors = 5).fit(X_train,y_train)
y_pred = classifier.predict(X_test)
[4] i =0
print ("\n----------")
print ('%-25s %-25s %-25s'% ('Original Label','PredictedLabel','Correct/Wrong'))
print ("----------")
Original Label PredictedLabel Correct/Wrong
11s completed at 10:07 PM
Type here to search
21°C
ENG IN
10:09 PM
14-Jan-23


Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5B.ipynb - Colabora
code for apriori algorithm in
Iris.csv | Kaggle
colab.research.google.com/drive/1_MItZIwZWXU90V9NIE4Iw4yrdHYA8I_p#scrollTo=_mgoQyQbHvpM
ML Prac 5B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[7] for label in y_test:
print ('%-25s %-25s'% (label, y_pred[i]), end="")
if(label == y_pred[i]):
print (' %-25s'% ('Correct'))
else:
print (' %-25s'% ('Wrong'))
i = i +1
print ("----------")
print("\nConfusion Matrix:\n",metrics.confusion_matrix(y_test,y_pred))
print ("----------")
print("\nClassification Report:\n",metrics.classification_report(y_test, y_pred))
print ("----------")
print('Accuracy of the classifer is %0.2f'% metrics.accuracy_score(y_test,y_pred))
print ("----------")
Iris-virginica Iris-virginica Correct
Iris-setosa Iris-setosa Correct
Iris-versicolor Iris-versicolor Correct
Iris-versicolor Iris-versicolor Correct
Iris-versicolor Iris-versicolor Correct
Iris-setosa Iris-setosa Correct
Iris-setosa Iris-setosa Correct
Iris-versicolor Iris-versicolor Correct
Iris-virginica Iris-virginica Correct
Iris-virginica Iris-virginica Correct
Iris-virginica Iris-virginica Correct
Iris-virginica Iris-virginica Correct
Iris-versicolor Iris-versicolor Correct
11s completed at 10:07 PM
Type here to search
21°C
ENG IN
10:10 PM
14-Jan-23


Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5B.ipynb - Colaborat
code for apriori algorithm in
Iris.csv | Kaggle
colab.research.google.com/drive/1_MItZIwZWXU90V9NIE4Iw4yrdHYA8I_p#scrollTo=_mgoQyQbHvpM
ML Prac 5B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
Iris-virginica Iris-virginica Correct
[7] Iris-virginica Iris-virginica Correct
Iris-versicolor Iris-versicolor Correct
Iris-setosa Iris-setosa Correct
Iris-versicolor Iris-versicolor Correct
Confusion Matrix:
[[4 0 0]
[0 6 0]
[0 0 5]]
Classification Report:
precision recall f1-score support
Iris-setosa 1.00 1.00 1.00 4
Iris-versicolor 1.00 1.00 1.00 6
Iris-virginica 1.00 1.00 1.00 5
accuracy 1.00 15
macro avg 1.00 1.00 1.00 15
weighted avg 1.00 1.00 1.00 15
Accuracy of the classifer is 1.00
[11] plt.figure()
sns.pairplot(dataset,hue = "Species",size= 1.5 ,markers =["o","s","D"])
11s completed at 10:07 PM
Type here to search
21°C
ENG IN
10:10 PM
14-Jan-23

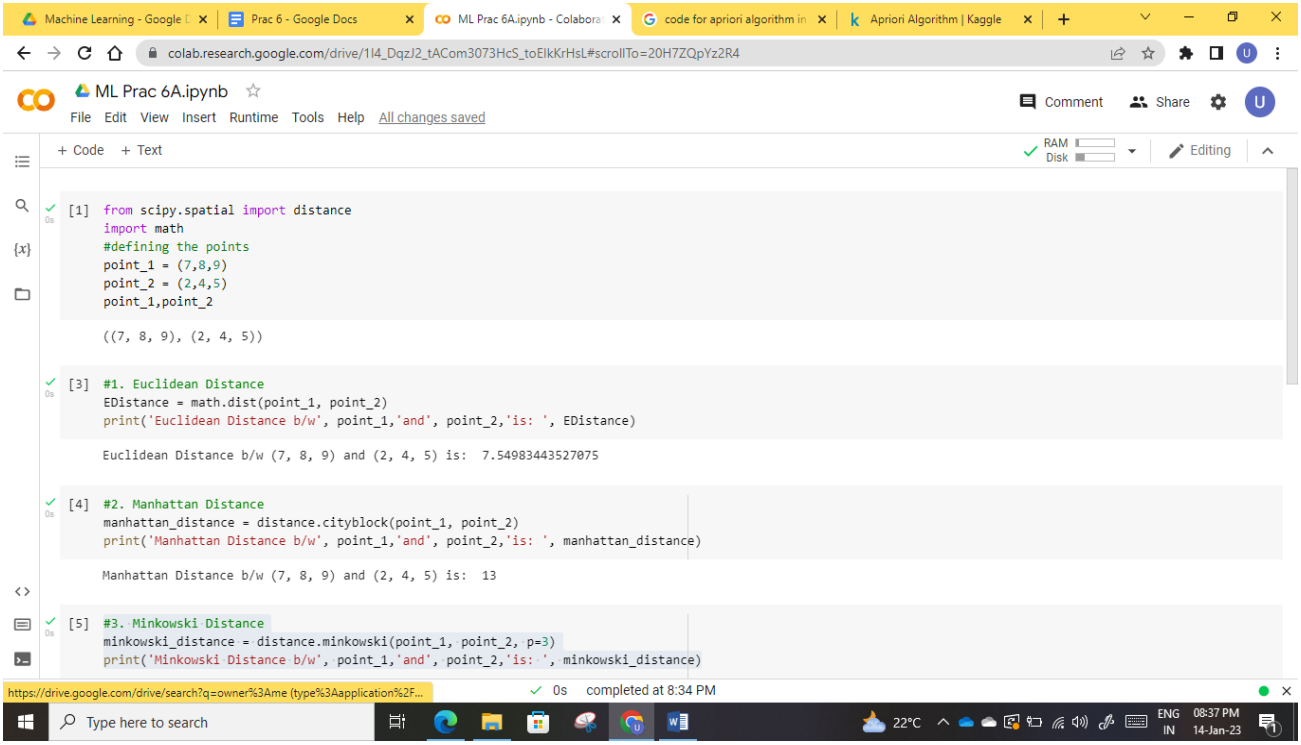

Machine Learning - Google
Prac 5 - Google Docs
ML Prac 5B.ipynb - Colaborat
code for apriori algorithm in
Iris.csv | Kaggle
colab.research.google.com/drive/1_MItZIwZWXU90V9NIE4Iw4yrdHYA8I_p#scrollTo=_mgoQyQbHvpM
ML Prac 5B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[11] /usr/local/lib/python3.8/dist-packages/seaborn/axisgrid.py:2076: UserWarning: The `size` parameter has been renamed to `height`; please update your code.
warnings.warn(msg, UserWarning)
<Figure size 432x288 with 0 Axes>
Id
SepalLengthCm
SepalWidthCm
PetalLengthCm
PetalWidthCm
Species
Iris-setosa
Iris-versicolor
Iris-virginica
11s completed at 10:07 PM
Type here to search
21°C
ENG IN
10:10 PM
14-Jan-23

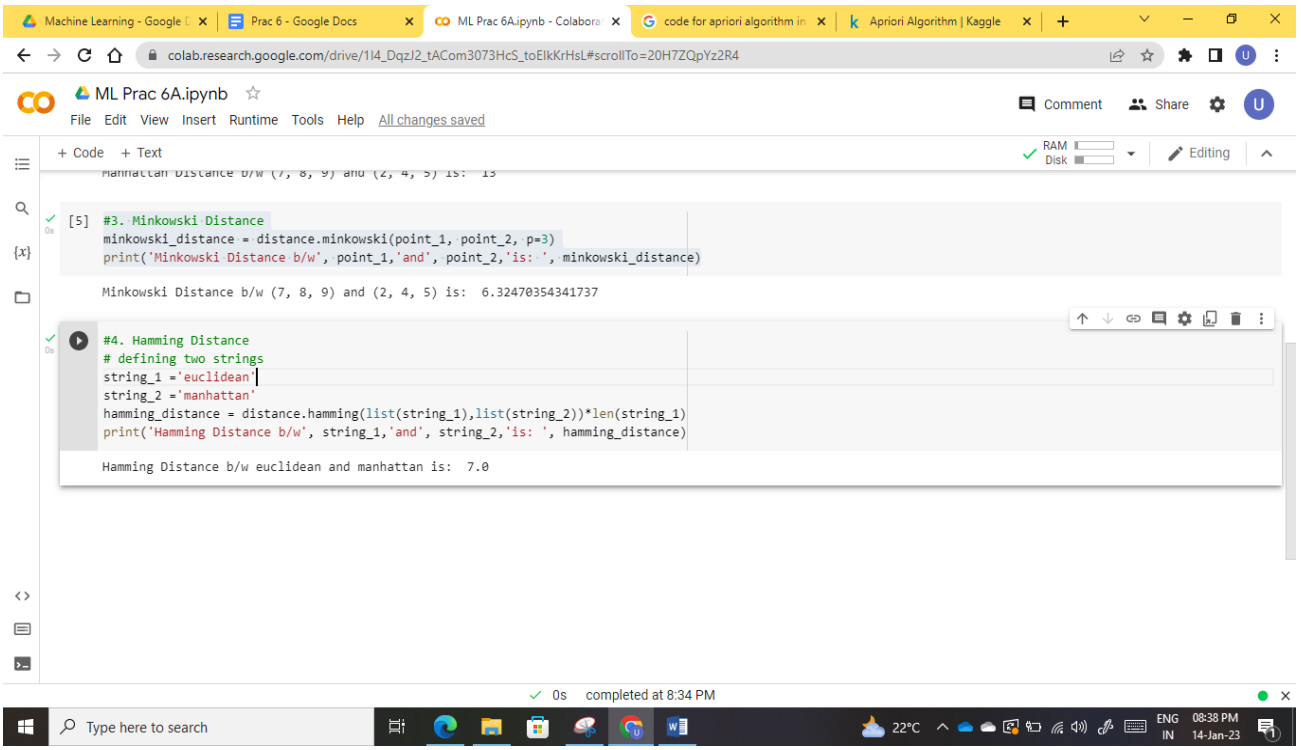
Implement the different Distance methods (Euclidean) with Prediction, Test Score and Confusion Matrix

Code :

```
from scipy.spatial import distance
import math
#defining the points
point_1 = (7,8,9)
point_2 = (2,4,5)
point_1,point_2

#1. Euclidean Distance
EDistance = math.dist(point_1, point_2)
print('Euclidean Distance b/w', point_1,'and', point_2,'is: ', EDistance)

#2. Manhattan Distance
manhattan_distance = distance.cityblock(point_1, point_2)
print('Manhattan Distance b/w', point_1,'and', point_2,'is: ', manhattan_distance)

#3. Minkowski Distance
minkowski_distance = distance.minkowski(point_1, point_2, p=3)
print('Minkowski Distance b/w', point_1,'and', point_2,'is: ', minkowski_distance)

#4. Hamming Distance
# defining two strings
string_1 ='euclidean'
string_2 ='manhattan'
hamming_distance = distance.hamming(list(string_1),list(string_2))*len(string_1)
print('Hamming Distance b/w', string_1,'and', string_2,'is: ', hamming_distance)
```

Output :


ML Prac 6A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
[1] from scipy.spatial import distance
import math
#defining the points
point_1 = (7,8,9)
point_2 = (2,4,5)
point_1,point_2
((7, 8, 9), (2, 4, 5))
[3] #1. Euclidean Distance
EDistance = math.dist(point_1, point_2)
print('Euclidean Distance b/w', point_1,'and', point_2,'is: ', EDistance)
Euclidean Distance b/w (7, 8, 9) and (2, 4, 5) is: 7.549834435270 75
[4] #2. Manhattan Distance
manhattan_distance = distance.cityblock(point_1, point_2)
print('Manhattan Distance b/w', point_1,'and', point_2,'is: ', manhattan_distance)
Manhattan Distance b/w (7, 8, 9) and (2, 4, 5) is: 13
[5] #3. Minkowski Distance
minkowski_distance = distance.minkowski(point_1, point_2, p=3)
print('Minkowski Distance b/w', point_1,'and', point_2,'is: ', minkowski_distance)
0s completed at 8:34 PM


ML Prac 6A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
[5] #3. Minkowski Distance
minkowski_distance = distance.minkowski(point_1, point_2, p=3)
print('Minkowski Distance b/w', point_1,'and', point_2,'is: ', minkowski_distance)
Minkowski Distance b/w (7, 8, 9) and (2, 4, 5) is: 6.32470354341737
#4. Hamming Distance
# defining two strings
string_1 ='euclidean'
string_2 ='manhattan'
hamming_distance = distance.hamming(list(string_1),list(string_2))*len(string_1)
print('Hamming Distance b/w', string_1,'and', string_2,'is: ', hamming_distance)
Hamming Distance b/w euclidean and manhattan is: 7.0
0s completed at 8:34 PM

Implement the classification model using clustering for the following techniques with K means clustering with Prediction, Test Score and Confusion Matrix

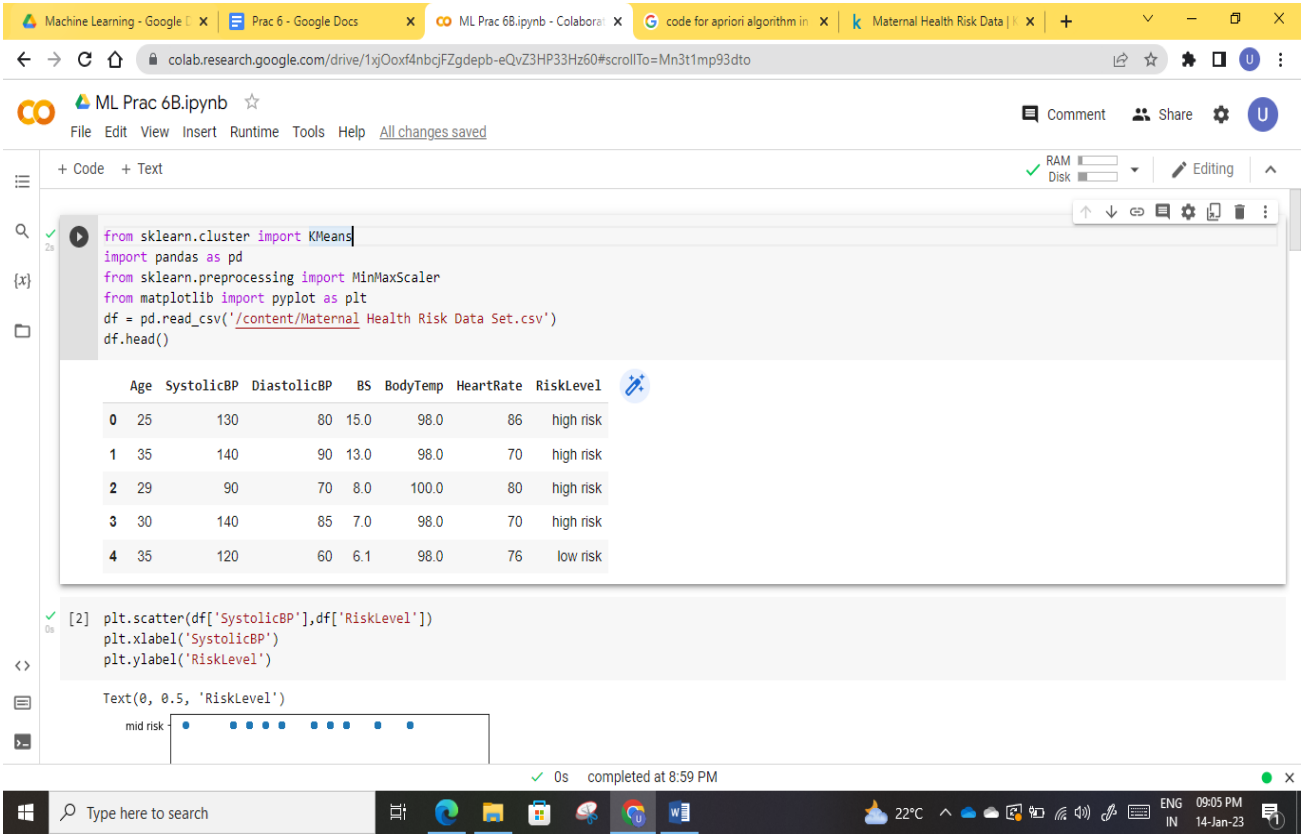
Code :

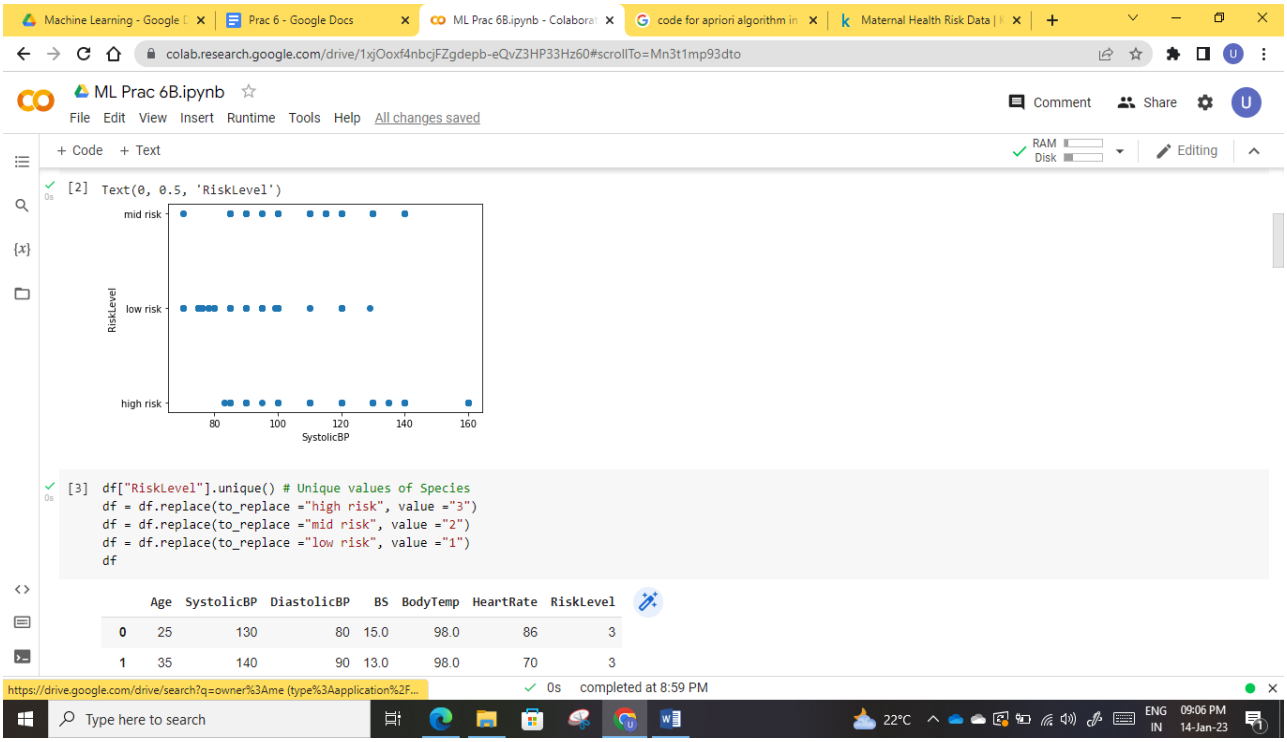
```
from sklearn.cluster import KMeans
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from matplotlib import pyplot as plt
df = pd.read_csv('/content/Maternal Health Risk Data Set.csv')
df.head()

plt.scatter(df['SystolicBP'],df['RiskLevel'])
plt.xlabel('SystolicBP')
plt.ylabel('RiskLevel')

df["RiskLevel"].unique() # Unique values of Species
df = df.replace(to_replace ="high risk", value ="3")
df = df.replace(to_replace ="mid risk", value ="2")
df = df.replace(to_replace ="low risk", value ="1")
df

km = KMeans(n_clusters=3)
predicted = km.fit_predict(df[['SystolicBP','RiskLevel']])
df['cluster']=predicted
#df.drop(['Cluster'],axis=1,inplace=True)
df.head()

df1=df[df.cluster==0]
df2=df[df.cluster==1]
df3=df[df.cluster==2]
plt.scatter(km.cluster_centers_[:,0],km.cluster_centers_[:,1],color='purple',marker='*',label=
            'Centroid')
```

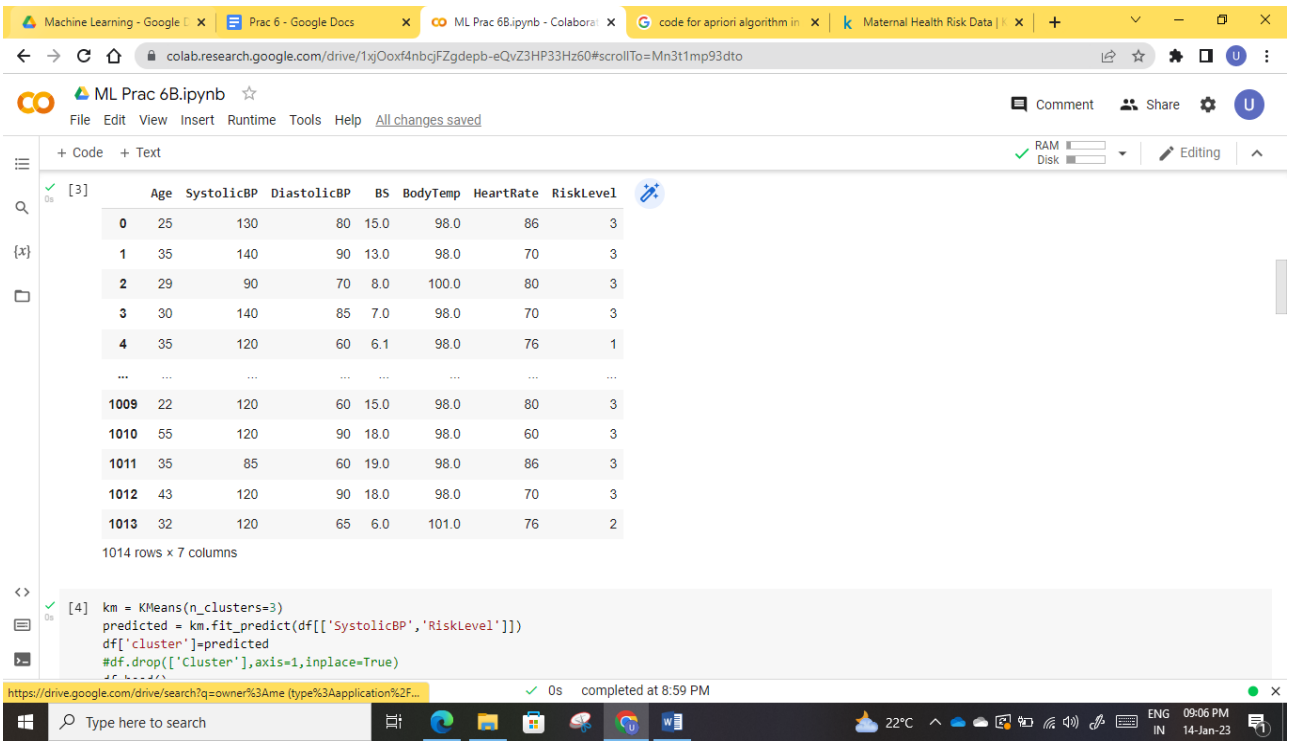
```
plt.xlabel('SystolicBP')
plt.ylabel('RiskLevel')
plt.legend()

scaler =MinMaxScaler()
scaler.fit(df[['RiskLevel']])
df['RiskLevel']=scaler.transform(df[['RiskLevel']])
scaler.fit(df[['SystolicBP']])
df['SystolicBP']=scaler.transform(df[['SystolicBP']])
df.head()

plt.scatter(df['SystolicBP'],df['RiskLevel'])
plt.xlabel('SystolicBP')
plt.ylabel('RiskLevel')
km=KMeans(n_clusters=3)
predicted=km.fit_predict(df[['SystolicBP','RiskLevel']])
df['cluster']=predicted
df.head()

df1=df[df.cluster==0]
df2=df[df.cluster==1]
df2=df[df.cluster==2]
plt.scatter(df1['SystolicBP'],df1['RiskLevel'],color='green')
plt.scatter(df2['SystolicBP'],df2['RiskLevel'],color='red')
plt.scatter(df3['SystolicBP'],df3['RiskLevel'],color='black')
plt.scatter(km.cluster_centers_[:,0],km.cluster_centers_[:,1],color='purple',marker='*',label='Centroi
d')
plt.xlabel('SystolicBP')
plt.ylabel('RiskLevel')
plt.legend()

#Elbow plot
sse = []
k_range =range(1,10)
```

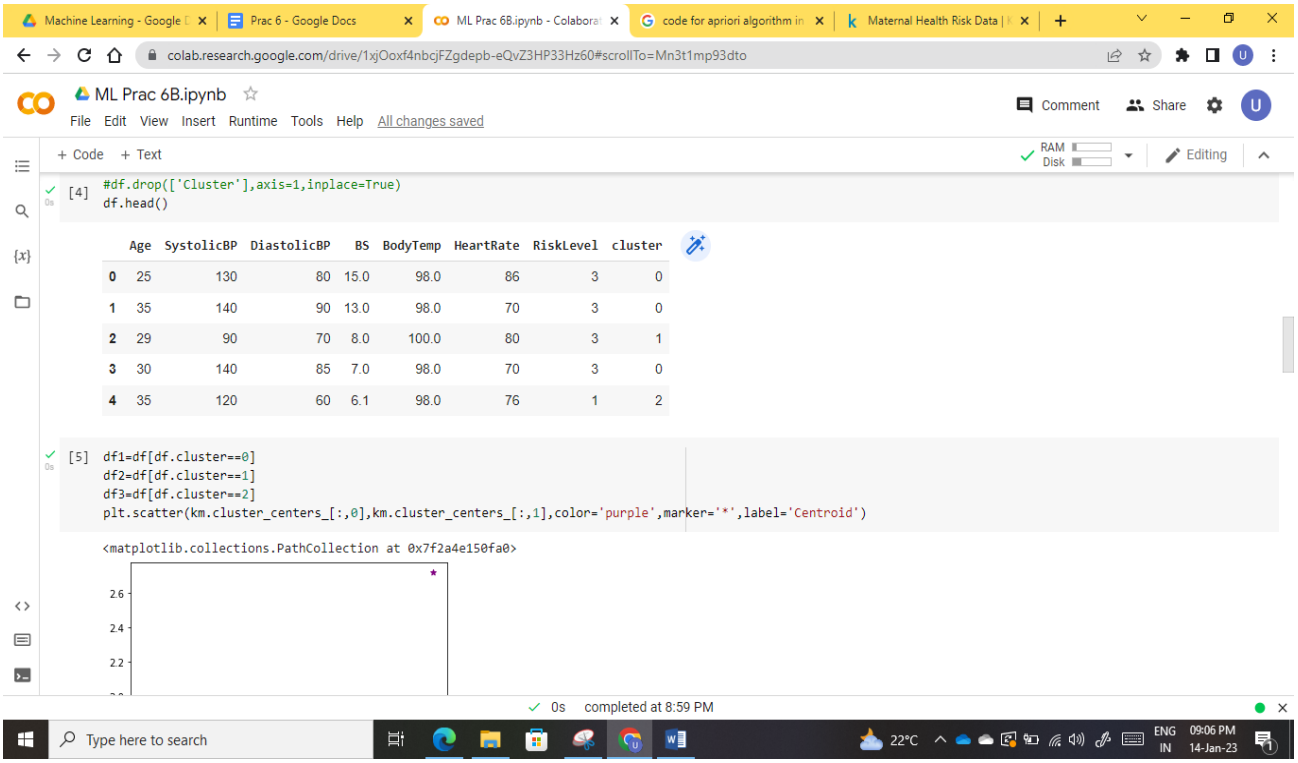
```
for k in k_range:
        km = KMeans(n_clusters=k)
        km.fit(df[['SystolicBP','RiskLevel']])
        sse.append(km.inertia_)
plt.xlabel('K')
plt.ylabel('Sum of squared error')
plt.plot(k_range,sse)
```

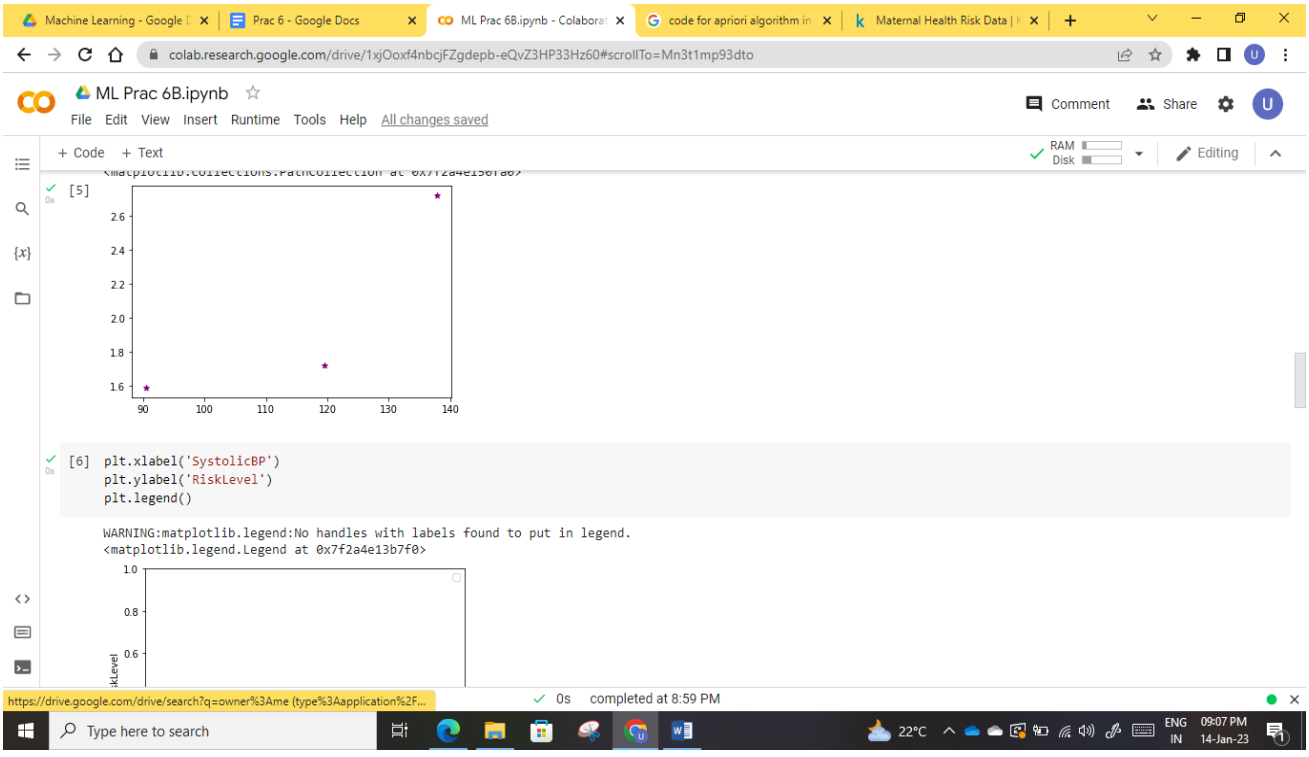
Output :

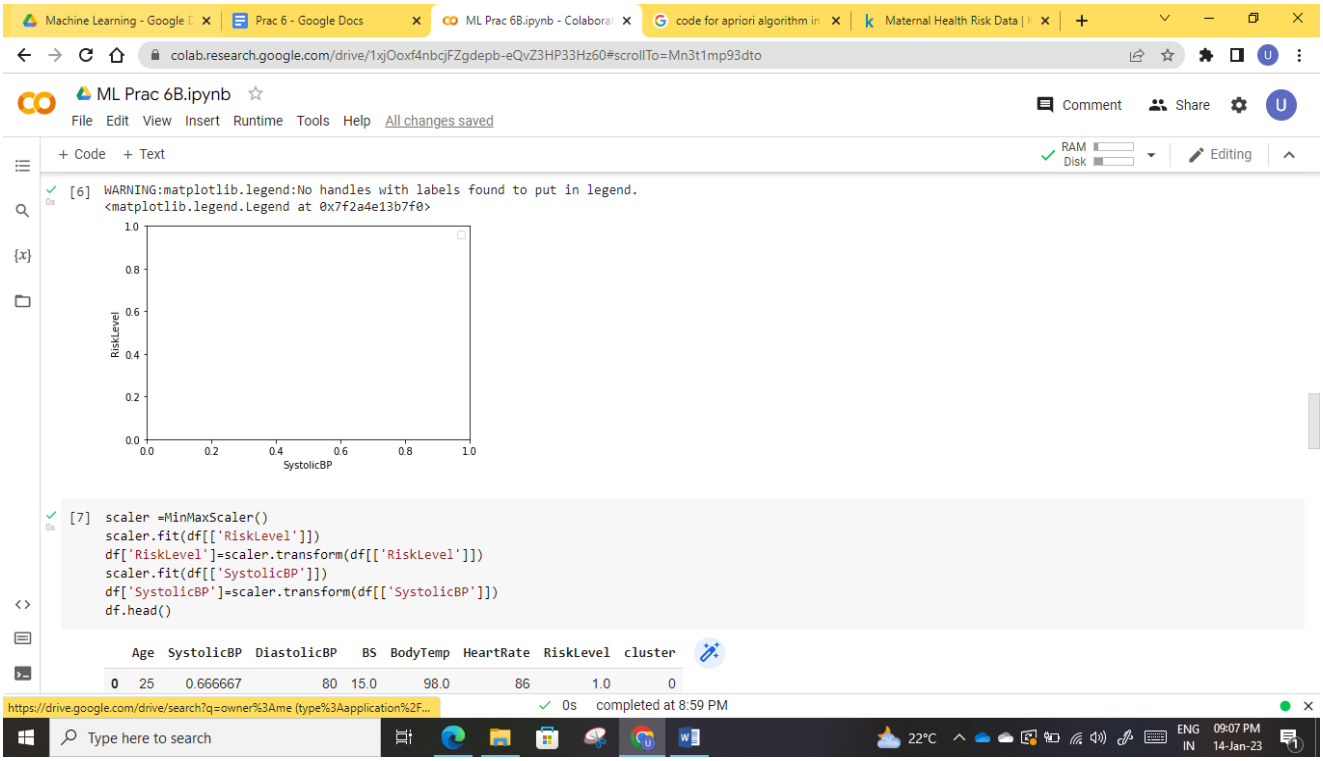


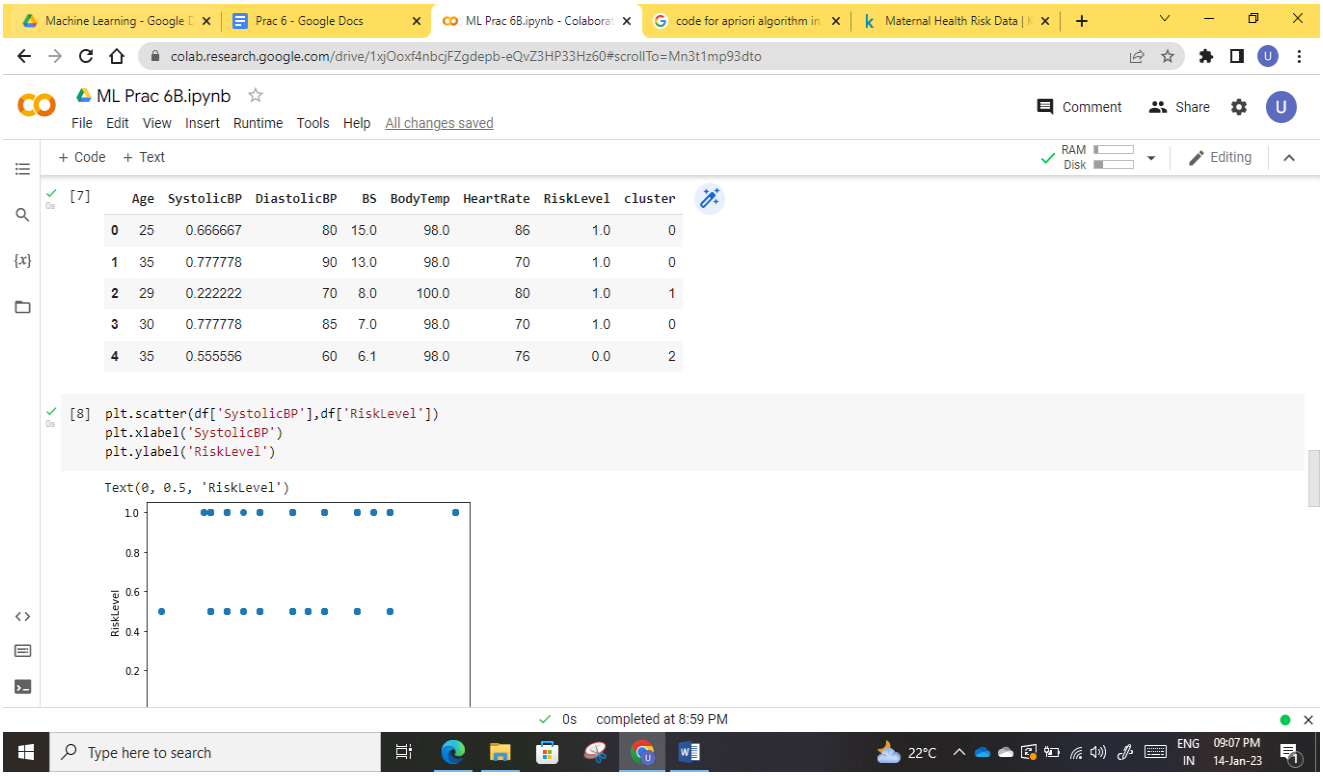

Machine Learning - Google
Prac 6 - Google Docs
ML Prac 6B.ipynb - Colabora
code for apriori algorithm in
Maternal Health Risk Data |
colab.research.google.com/drive/1xjOoxf4nbcjFZgdepb-eQvZ3HP33Hz60#scrollTo=Mn3t1mp93dto
ML Prac 6B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[2] Text(0, 0.5, 'RiskLevel')
mid risk
low risk
high risk
RiskLevel
80
100
120
140
160
SystolicBP
[3] df["RiskLevel"].unique() # Unique values of Species
df = df.replace(to_replace ="high risk", value ="3")
df = df.replace(to_replace ="mid risk", value ="2")
df = df.replace(to_replace ="low risk", value ="1")
df
Age SystolicBP DiastolicBP BS BodyTemp HeartRate RiskLevel
0 25 130 80 15.0 98.0 86 3
1 35 140 90 13.0 98.0 70 3
0s completed at 8:59 PM
Type here to search
22°C
ENG IN
09:06 PM 14-Jan-23

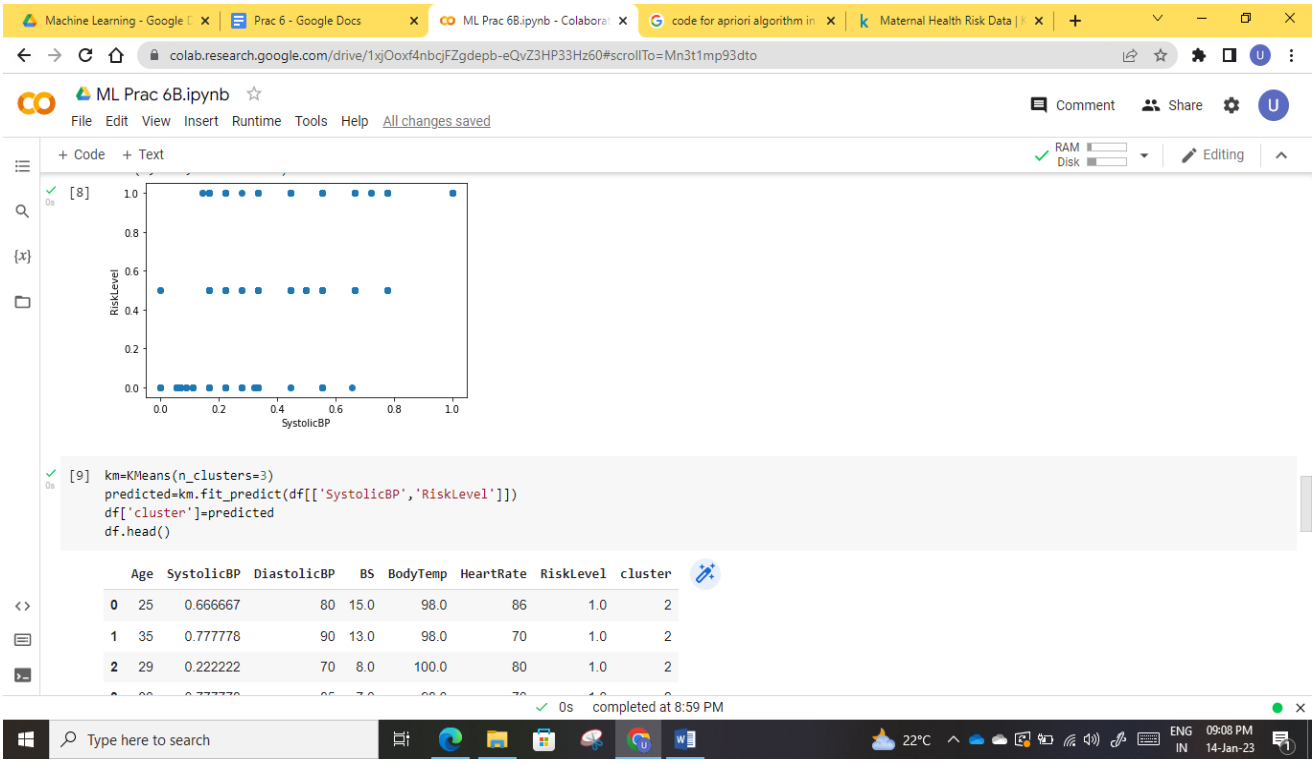

Machine Learning - Google
Prac 6 - Google Docs
ML Prac 6B.ipynb - Colabora
code for apriori algorithm in
Maternal Health Risk Data |
colab.research.google.com/drive/1xjOoxf4nbcjFZgdepb-eQvZ3HP33Hz60#scrollTo=Mn3t1mp93dto
ML Prac 6B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[3]
Age SystolicBP DiastolicBP BS BodyTemp HeartRate RiskLevel
0 25 130 80 15.0 98.0 86 3
1 35 140 90 13.0 98.0 70 3
2 29 90 70 8.0 100.0 80 3
3 30 140 85 7.0 98.0 70 3
4 35 120 60 6.1 98.0 76 1
1009 22 120 60 15.0 98.0 80 3
1010 55 120 90 18.0 98.0 60 3
1011 35 85 60 19.0 98.0 86 3
1012 43 120 90 18.0 98.0 70 3
1013 32 120 65 6.0 101.0 76 2
1014 rows × 7 columns
[4] km = KMeans(n_clusters=3)
predicted = km.fit_predict(df[['SystolicBP','RiskLevel']])
df['cluster']=predicted
#df.drop(['Cluster'],axis=1,inplace=True)
0s completed at 8:59 PM
Type here to search
22°C
ENG IN
09:06 PM 14-Jan-23

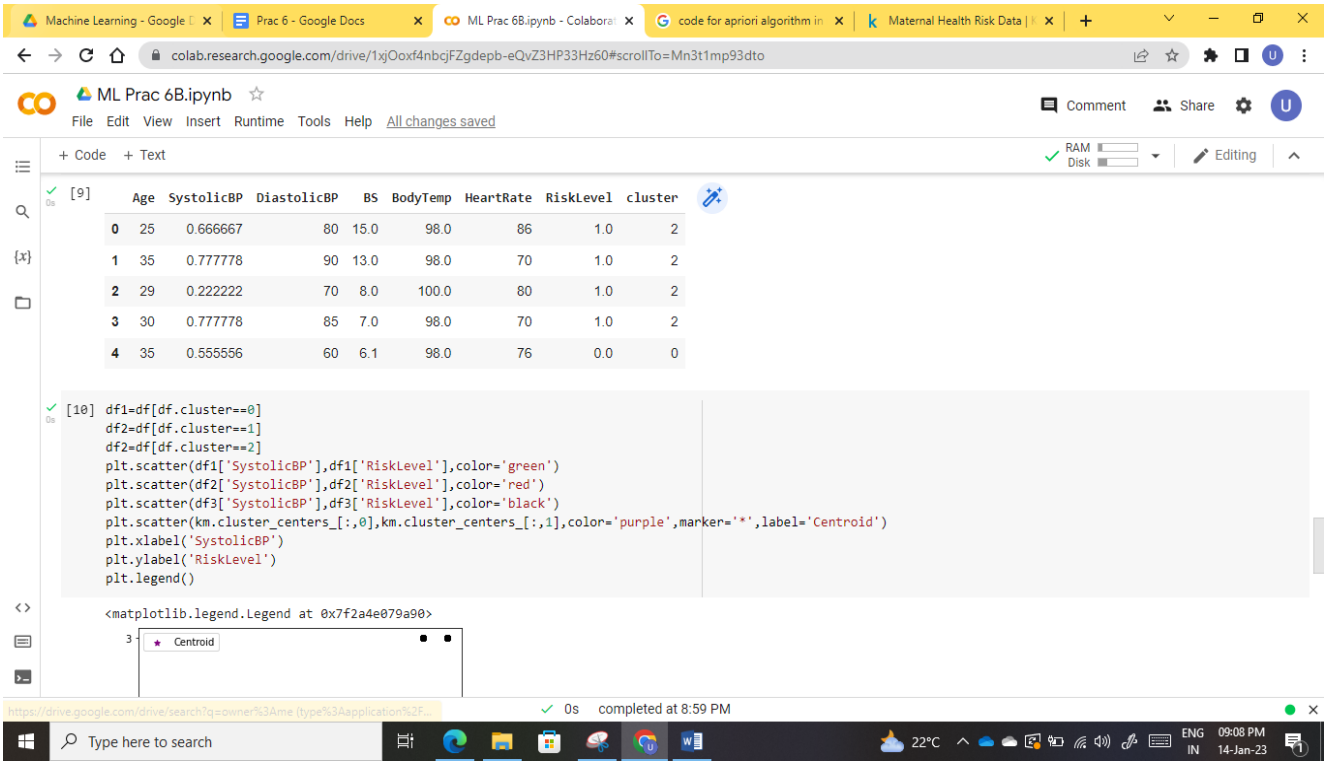

Machine Learning - Google
Prac 6 - Google Docs
ML Prac 6B.ipynb - Colabora
code for apriori algorithm in
Maternal Health Risk Data |
colab.research.google.com/drive/1xjOoxf4nbcjFZgdepb-eQvZ3HP33Hz60#scrollTo=Mn3t1mp93dto
ML Prac 6B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[4] #df.drop(['Cluster'],axis=1,inplace=True)
df.head()
Age SystolicBP DiastolicBP BS BodyTemp HeartRate RiskLevel cluster
0 25 130 80 15.0 98.0 86 3 0
1 35 140 90 13.0 98.0 70 3 0
2 29 90 70 8.0 100.0 80 3 1
3 30 140 85 7.0 98.0 70 3 0
4 35 120 60 6.1 98.0 76 1 2
[5] df1=df[df.cluster==0]
df2=df[df.cluster==1]
df3=df[df.cluster==2]
plt.scatter(km.cluster_centers_[:,0],km.cluster_centers_[:,1],color='purple',marker='*',label='Centroid')
<matplotlib.collections.PathCollection at 0x7f2a4e150fa0>
2.6
2.4
2.2
0s completed at 8:59 PM
Type here to search
22°C
ENG IN
09:06 PM
14-Jan-23

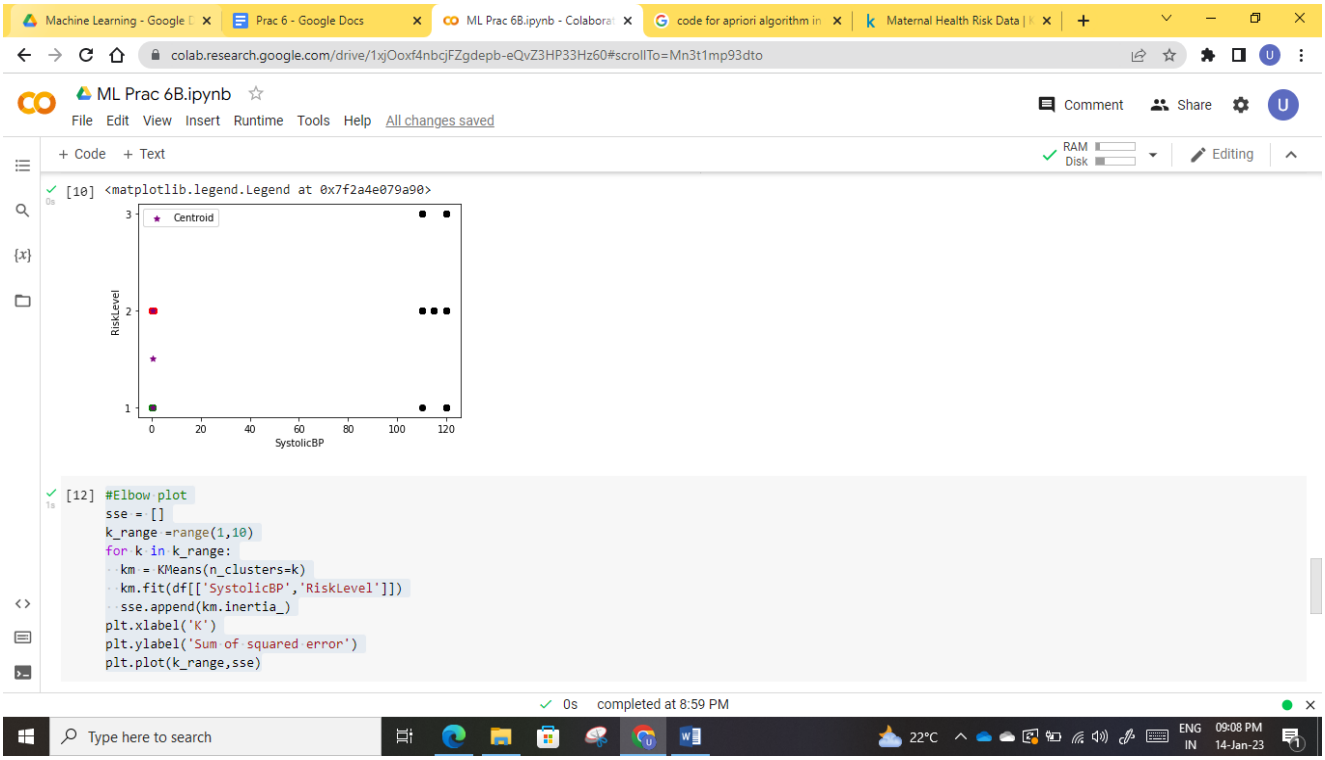

Machine Learning - Google
Prac 6 - Google Docs
ML Prac 6B.ipynb - Colabora
code for apriori algorithm in
Maternal Health Risk Data |
colab.research.google.com/drive/1xjOoxf4nbcjFZgdepb-eQvZ3HP33Hz60#scrollTo=Mn3t1mp93dto
ML Prac 6B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[5]
2.6
2.4
2.2
2.0
1.8
1.6
90
100
110
120
130
140
[6] plt.xlabel('SystolicBP')
plt.ylabel('RiskLevel')
plt.legend()
WARNING:matplotlib.legend:No handles with labels found to put in legend.
<matplotlib.legend.Legend at 0x7f2a4e13b7f0>
1.0
0.8
0.6
0s completed at 8:59 PM
Type here to search
22°C
ENG IN
09:07 PM
14-Jan-23

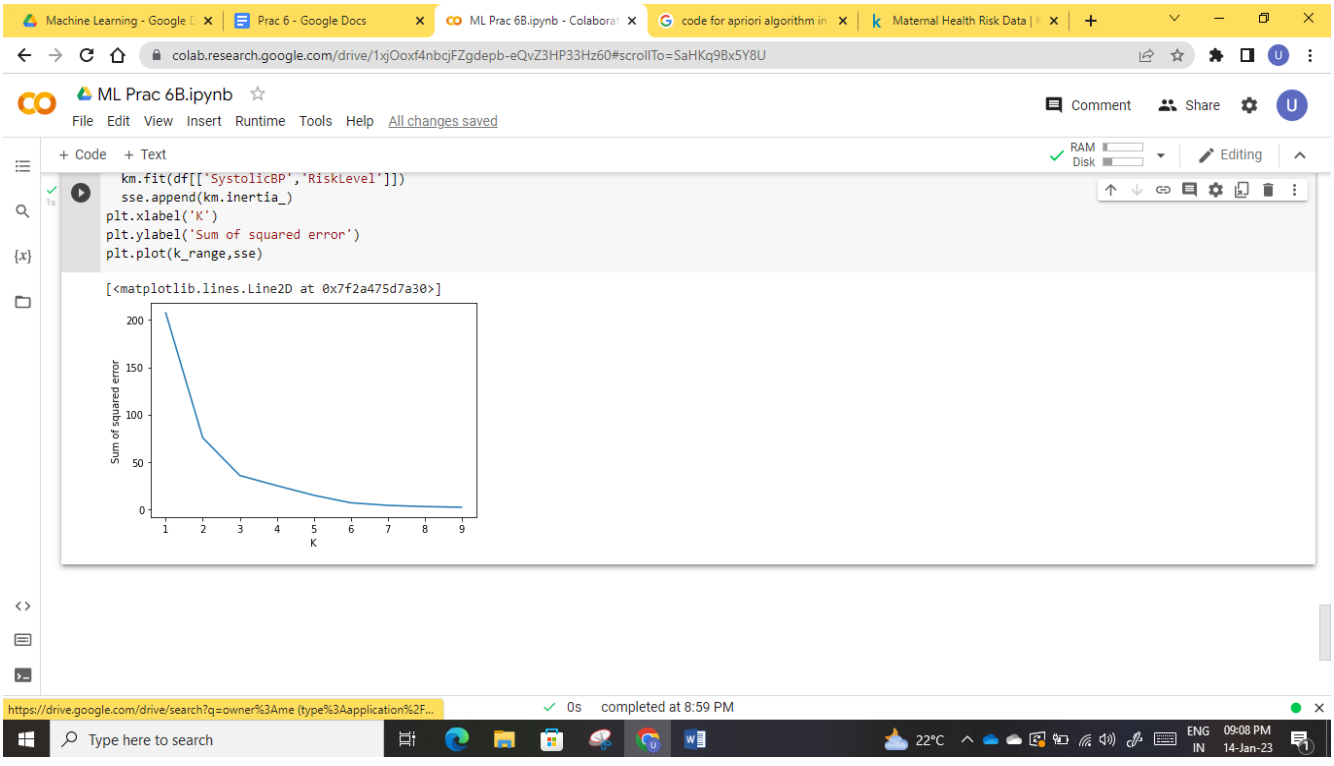

Machine Learning - Google
Prac 6 - Google Docs
ML Prac 6B.ipynb - Colabora
code for apriori algorithm in
Maternal Health Risk Data
colab.research.google.com/drive/1xjOoxf4nbcjFZgdepb-eQvZ3HP33Hz60#scrollTo=Mn3t1mp93dto
ML Prac 6B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[6] WARNING:matplotlib.legend:No handles with labels found to put in legend.
<matplotlib.legend.Legend at 0x7f2a4e13b7f0>
RiskLevel
SystolicBP
[7] scaler =MinMaxScaler()
scaler.fit(df[['RiskLevel']])
df['RiskLevel']=scaler.transform(df[['RiskLevel']])
scaler.fit(df[['SystolicBP']])
df['SystolicBP']=scaler.transform(df[['SystolicBP']])
df.head()
Age SystolicBP DiastolicBP BS BodyTemp HeartRate RiskLevel cluster
0 25 0.666667 80 15.0 98.0 86 1.0 0
0s completed at 8:59 PM
Type here to search
22°C
ENG IN
09:07 PM 14-Jan-23


Machine Learning - Google
Prac 6 - Google Docs
ML Prac 6B.ipynb - Colabora
code for apriori algorithm in
Maternal Health Risk Data
colab.research.google.com/drive/1xjOoxf4nbcjFZgdepb-eQvZ3HP33Hz60#scrollTo=Mn3t1mp93dto
ML Prac 6B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[7]
Age SystolicBP DiastolicBP BS BodyTemp HeartRate RiskLevel cluster
0 25 0.666667 80 15.0 98.0 86 1.0 0
1 35 0.777778 90 13.0 98.0 70 1.0 0
2 29 0.222222 70 8.0 100.0 80 1.0 1
3 30 0.777778 85 7.0 98.0 70 1.0 0
4 35 0.555556 60 6.1 98.0 76 0.0 2
[8] plt.scatter(df['SystolicBP'],df['RiskLevel'])
plt.xlabel('SystolicBP')
plt.ylabel('RiskLevel')
Text(0, 0.5, 'RiskLevel')
RiskLevel
0s completed at 8:59 PM
Type here to search
22°C
ENG IN
09:07 PM 14-Jan-23


Machine Learning - Google
Prac 6 - Google Docs
ML Prac 6B.ipynb - Colabora
code for apriori algorithm in
Maternal Health Risk Data |
colab.research.google.com/drive/1xjOoxf4nbcjFZgdepb-eQvZ3HP33Hz60#scrollTo=Mn3t1mp93dto
ML Prac 6B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[8]
RiskLevel
SystolicBP
[9] km=KMeans(n_clusters=3)
predicted=km.fit_predict(df[['SystolicBP','RiskLevel']])
df['cluster']=predicted
df.head()
Age SystolicBP DiastolicBP BS BodyTemp HeartRate RiskLevel cluster
0 25 0.666667 80 15.0 98.0 86 1.0 2
1 35 0.777778 90 13.0 98.0 70 1.0 2
2 29 0.222222 70 8.0 100.0 80 1.0 2
0s completed at 8:59 PM
Type here to search
22°C
ENG IN
09:08 PM 14-Jan-23


Machine Learning - Google
Prac 6 - Google Docs
ML Prac 6B.ipynb - Colabora
code for apriori algorithm in
Maternal Health Risk Data |
colab.research.google.com/drive/1xjOoxf4nbcjFZgdepb-eQvZ3HP33Hz60#scrollTo=Mn3t1mp93dto
ML Prac 6B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[9]
Age SystolicBP DiastolicBP BS BodyTemp HeartRate RiskLevel cluster
0 25 0.666667 80 15.0 98.0 86 1.0 2
1 35 0.777778 90 13.0 98.0 70 1.0 2
2 29 0.222222 70 8.0 100.0 80 1.0 2
3 30 0.777778 85 7.0 98.0 70 1.0 2
4 35 0.555556 60 6.1 98.0 76 0.0 0
[10] df1=df[df.cluster==0]
df2=df[df.cluster==1]
df2=df[df.cluster==2]
plt.scatter(df1['SystolicBP'],df1['RiskLevel'],color='green')
plt.scatter(df2['SystolicBP'],df2['RiskLevel'],color='red')
plt.scatter(df3['SystolicBP'],df3['RiskLevel'],color='black')
plt.scatter(km.cluster_centers_[:,0],km.cluster_centers_[:,1],color='purple',marker='*',label='Centroid')
plt.xlabel('SystolicBP')
plt.ylabel('RiskLevel')
plt.legend()
<matplotlib.legend.Legend at 0x7f2a4e079a90>
Centroid
0s completed at 8:59 PM
Type here to search
22°C
ENG IN
09:08 PM 14-Jan-23


Machine Learning - Google
Prac 6 - Google Docs
ML Prac 6B.ipynb - Colabora
code for apriori algorithm in
Maternal Health Risk Data |
colab.research.google.com/drive/1xjOoxf4nbcjFZgdepb-eQvZ3HP33Hz60#scrollTo=Mn3t1mp93dto
ML Prac 6B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[10] <matplotlib.legend.Legend at 0x7f2a4e079a90>
Centroid
RiskLevel
SystolicBP
[12] #Elbow plot
sse = []
k_range =range(1,10)
for k in k_range:
km = KMeans(n_clusters=k)
km.fit(df[['SystolicBP','RiskLevel']])
sse.append(km.inertia_)
plt.xlabel('K')
plt.ylabel('Sum of squared error')
plt.plot(k_range,sse)
0s completed at 8:59 PM
Type here to search
22°C
ENG
IN
09:08 PM
14-Jan-23


Machine Learning - Google
Prac 6 - Google Docs
ML Prac 6B.ipynb - Colabora
code for apriori algorithm in
Maternal Health Risk Data |
colab.research.google.com/drive/1xjOoxf4nbcjFZgdepb-eQvZ3HP33Hz60#scrollTo=SaHKq9Bx5Y8U
ML Prac 6B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
km.fit(df[['SystolicBP','RiskLevel']])
sse.append(km.inertia_)
plt.xlabel('K')
plt.ylabel('Sum of squared error')
plt.plot(k_range,sse)
[<matplotlib.lines.Line2D at 0x7f2a475d7a30>]
Sum of squared error
K
https://drive.google.com/drive/search?q=owner%3Ame (type%3Aapplication%2F...
0s completed at 8:59 PM
Type here to search
22°C
ENG
IN
09:08 PM
14-Jan-23

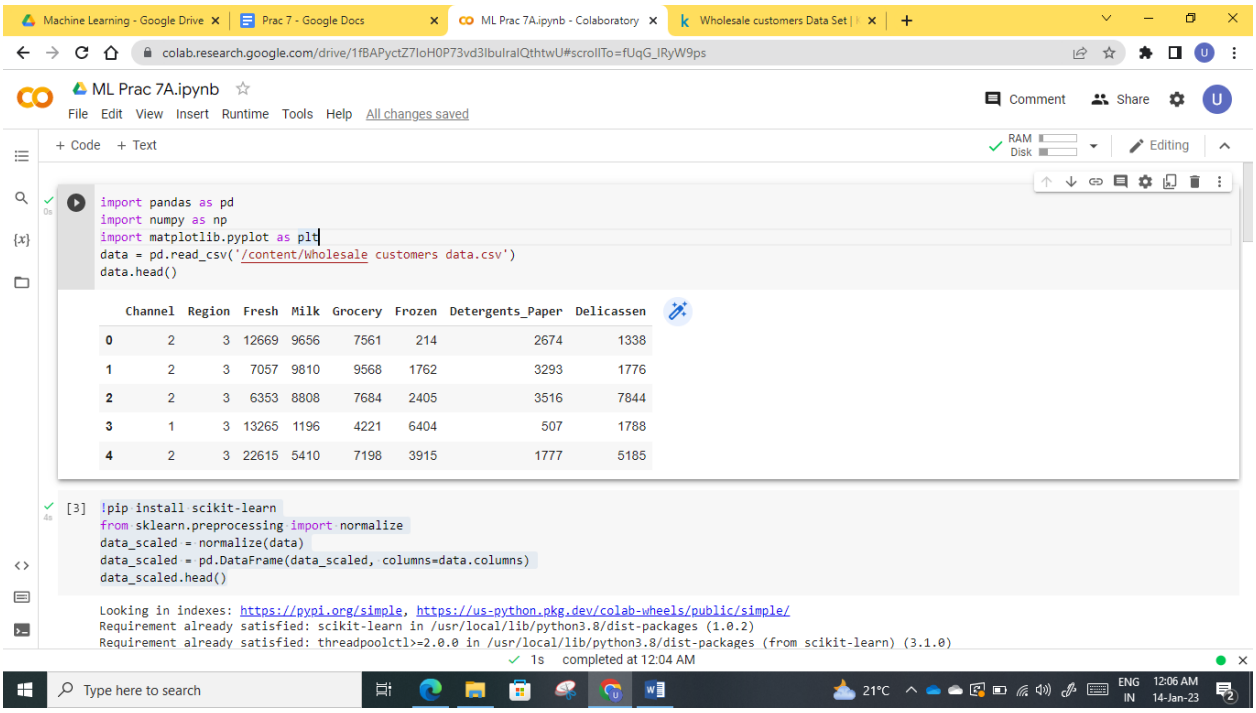
Implement the classification model using clustering for the following techniques with hierarchical clustering with Prediction, Test Score and Confusion Matrix

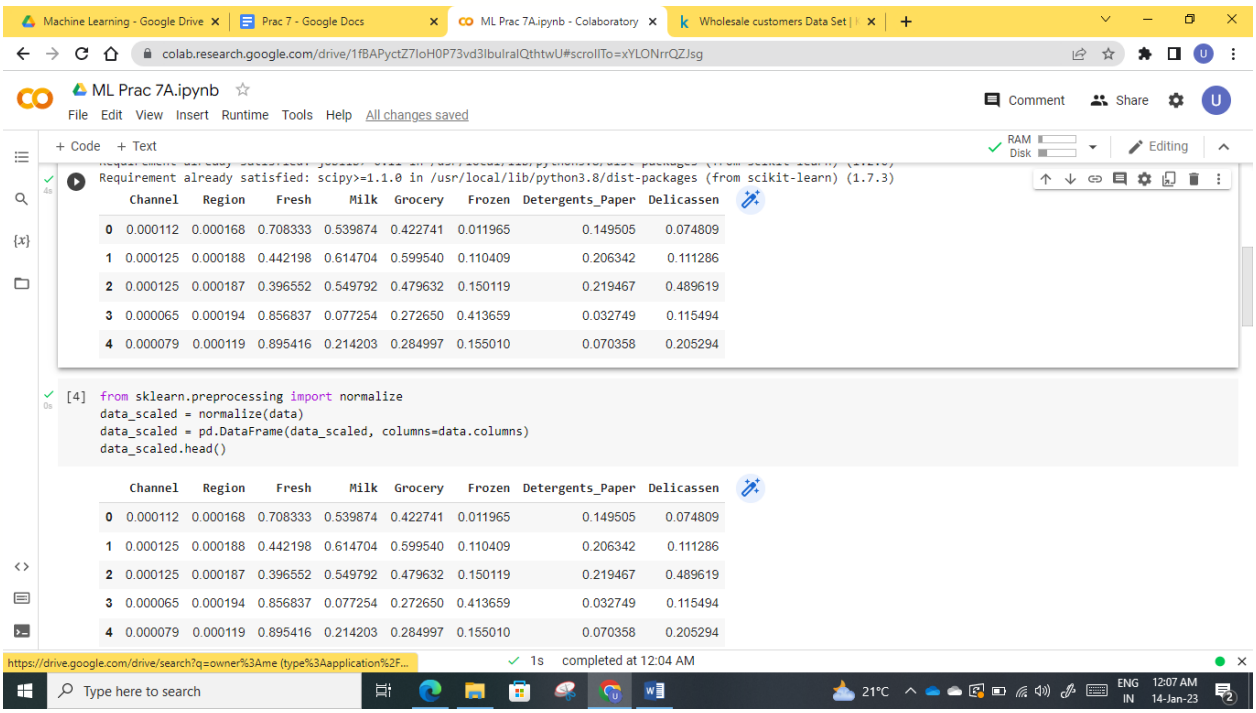
Code :

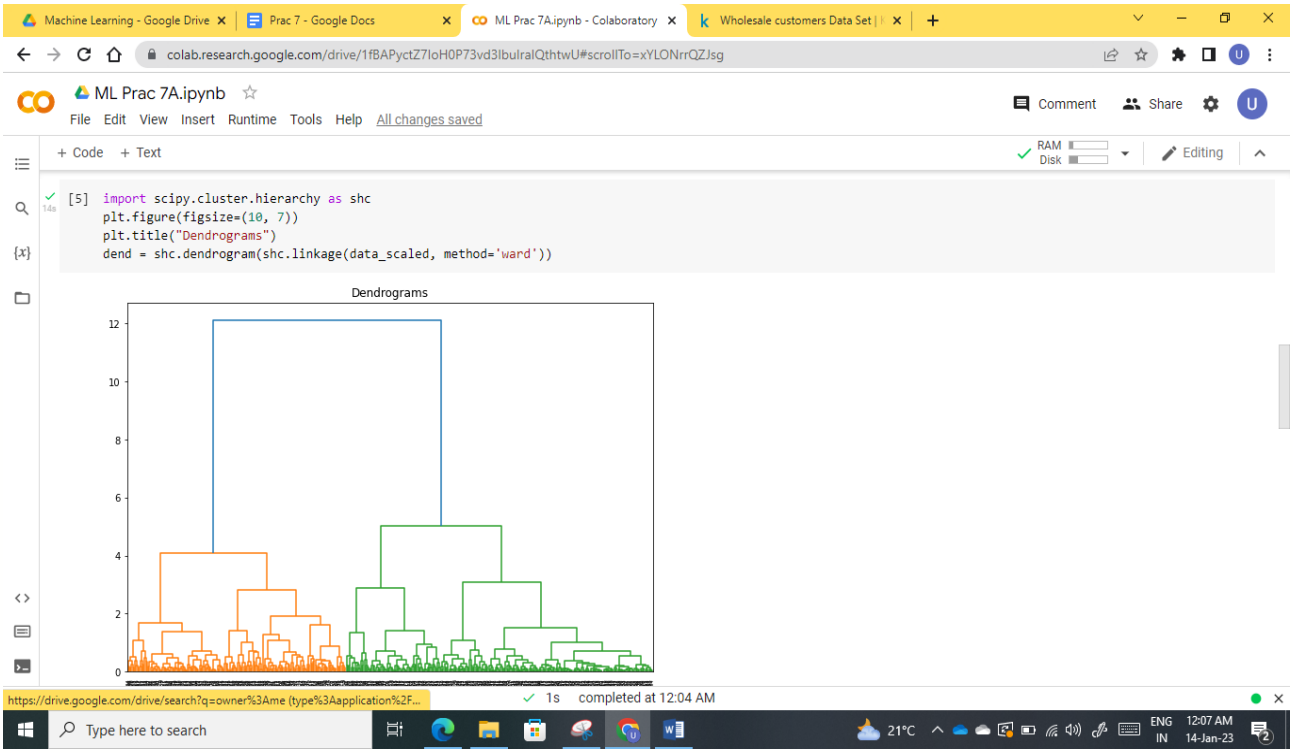
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
data = pd.read_csv('/content/Wholesale customers data.csv')
data.head()

!pip install scikit-learn
from sklearn.preprocessing import normalize
data_scaled = normalize(data)
data_scaled = pd.DataFrame(data_scaled, columns=data.columns)
data_scaled.head()

from sklearn.preprocessing import normalize
data_scaled = normalize(data)
data_scaled = pd.DataFrame(data_scaled, columns=data.columns)
data_scaled.head()

import scipy.cluster.hierarchy as shc
plt.figure(figsize=(10, 7))
plt.title("Dendrograms")
dend = shc.dendrogram(shc.linkage(data_scaled, method='ward'))

plt.figure(figsize=(10, 7))
plt.title("Dendrograms")
dend = shc.dendrogram(shc.linkage(data_scaled, method='ward'))
plt.axhline(y=6, color='r', linestyle='--')
from sklearn.cluster import AgglomerativeClustering
cluster = AgglomerativeClustering(n_clusters=2, affinity='euclidean', linkage='ward')
```

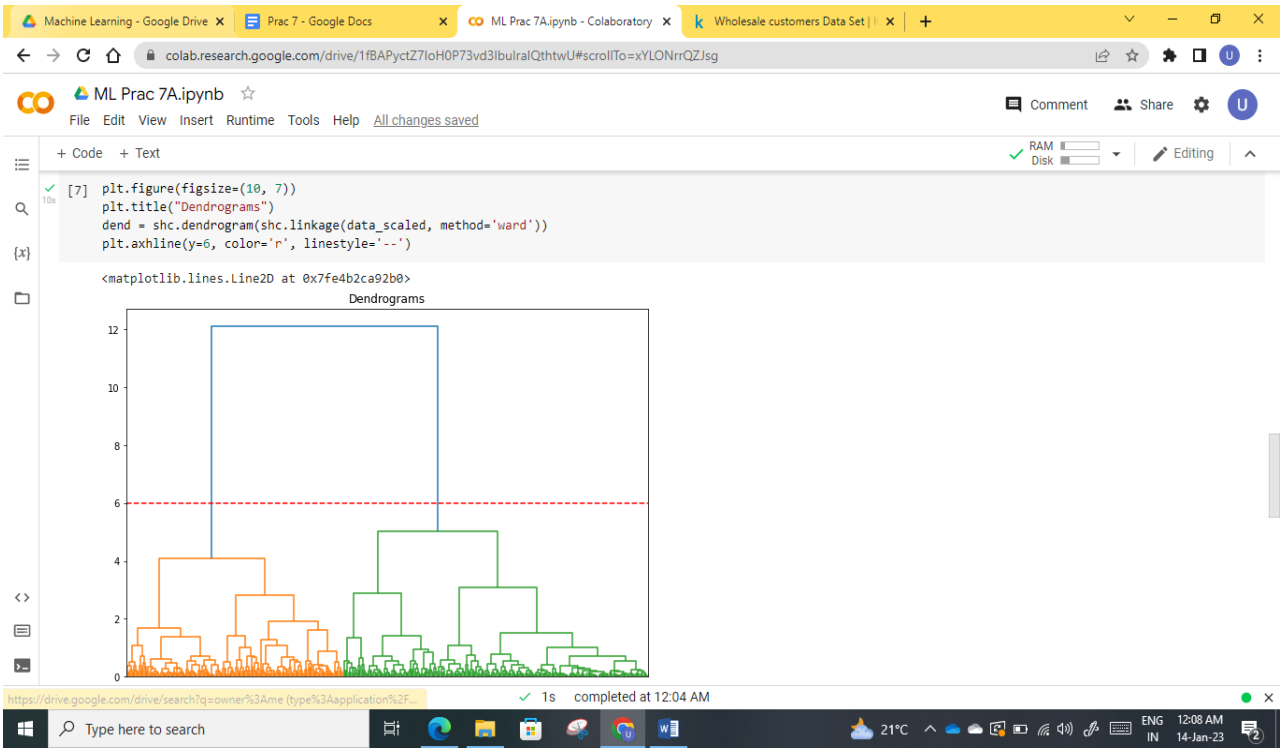
```
cluster.fit_predict(data_scaled)
plt.figure(figsize=(10, 7))
plt.scatter(data_scaled['Milk'], data_scaled['Grocery'], c=cluster.labels_)
```

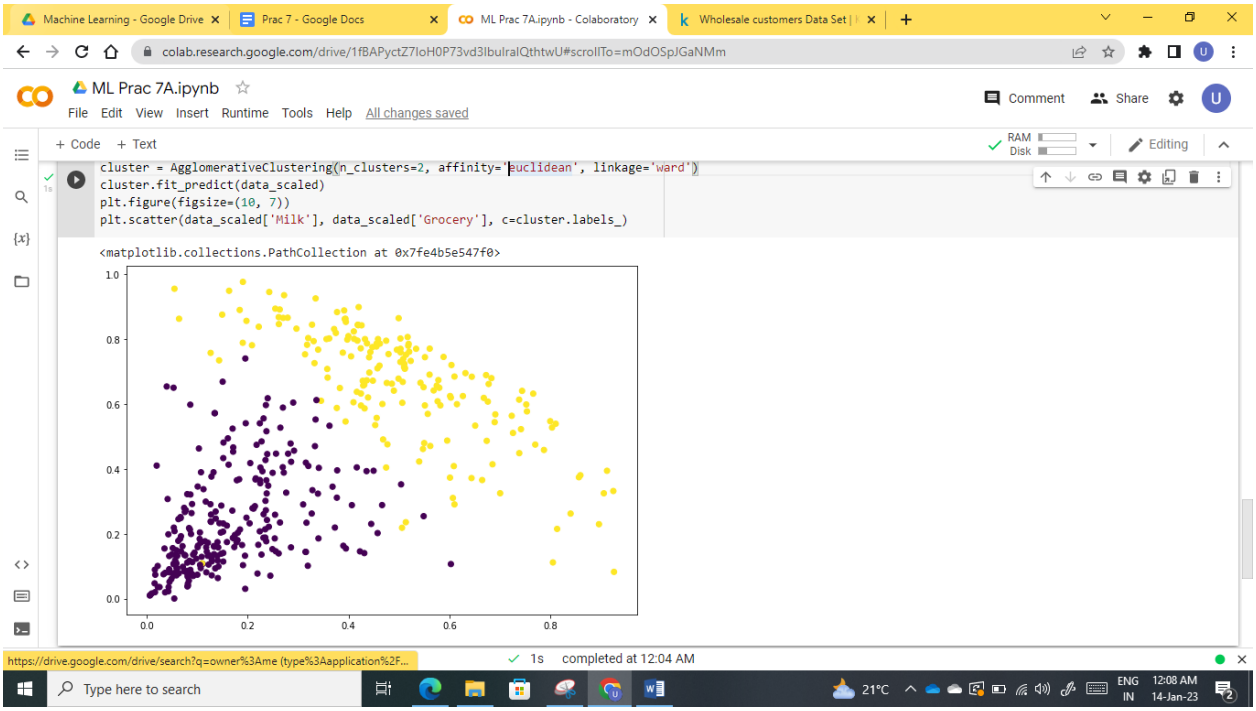
Output :






Machine Learning - Google Drive
Prac 7 - Google Docs
ML Prac 7A.ipynb - Colaboratory
Wholesale customers Data Set
colab.research.google.com/drive/1fBAPyctZ7IoH0P73vd3lbuIralQthtwU#scrollTo=xYLONrrQZJsg
ML Prac 7A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[5] import scipy.cluster.hierarchy as shc
plt.figure(figsize=(10, 7))
plt.title("Dendrograms")
dend = shc.dendrogram(shc.linkage(data_scaled, method='ward'))
Dendrograms
12
10
8
6
4
2
0
1s completed at 12:04 AM
Type here to search
21°C
ENG IN
12:07 AM 14-Jan-23


Machine Learning - Google Drive
Prac 7 - Google Docs
ML Prac 7A.ipynb - Colaboratory
Wholesale customers Data Set
colab.research.google.com/drive/1fBAPyctZ7IoH0P73vd3lbuIralQthtwU#scrollTo=xYLONrrQZJsg
ML Prac 7A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[7] plt.figure(figsize=(10, 7))
plt.title("Dendrograms")
dend = shc.dendrogram(shc.linkage(data_scaled, method='ward'))
plt.axhline(y=6, color='r', linestyle='--')
<matplotlib.lines.Line2D at 0x7fe4b2ca92b0>
Dendrograms
12
10
8
6
4
2
0
1s completed at 12:04 AM
Type here to search
21°C
ENG IN
12:08 AM 14-Jan-23


Machine Learning - Google Drive
Prac 7 - Google Docs
ML Prac 7A.ipynb - Colaboratory
Wholesale customers Data Set
colab.research.google.com/drive/1fBAPyctZ7IoH0P73vd3IbuIraIQthtwU#scrollTo=mOdOSpJGaNMm
ML Prac 7A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
cluster = AgglomerativeClustering(n_clusters=2, affinity='euclidean', linkage='ward')
cluster.fit_predict(data_scaled)
plt.figure(figsize=(10, 7))
plt.scatter(data_scaled['Milk'], data_scaled['Grocery'], c=cluster.labels_)
<matplotlib.collections.PathCollection at 0x7fe4b5e547f0>
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1s completed at 12:04 AM
Type here to search
21°C
ENG
IN
12:08 AM
14-Jan-23

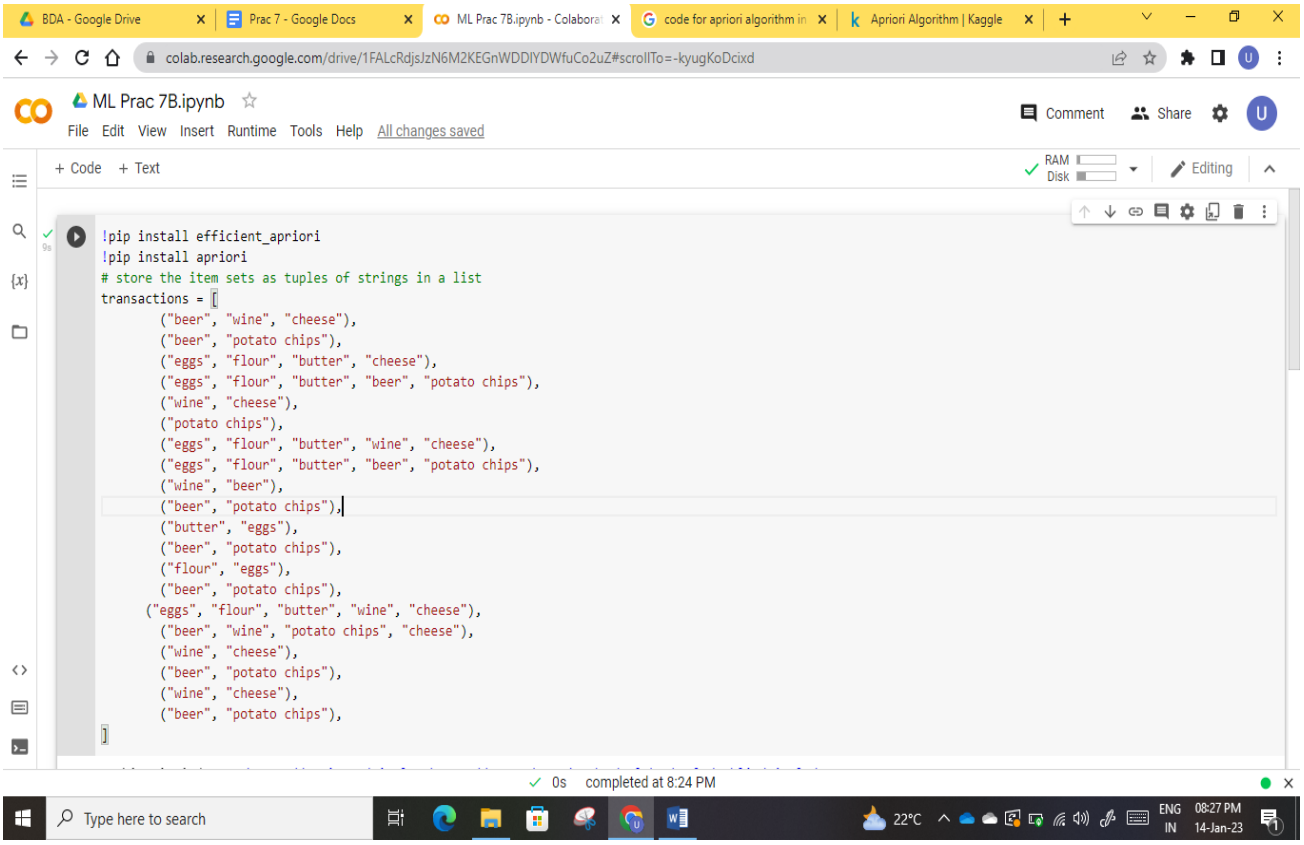
Implement the Rule based method and test the same (Apriori Algorithm)

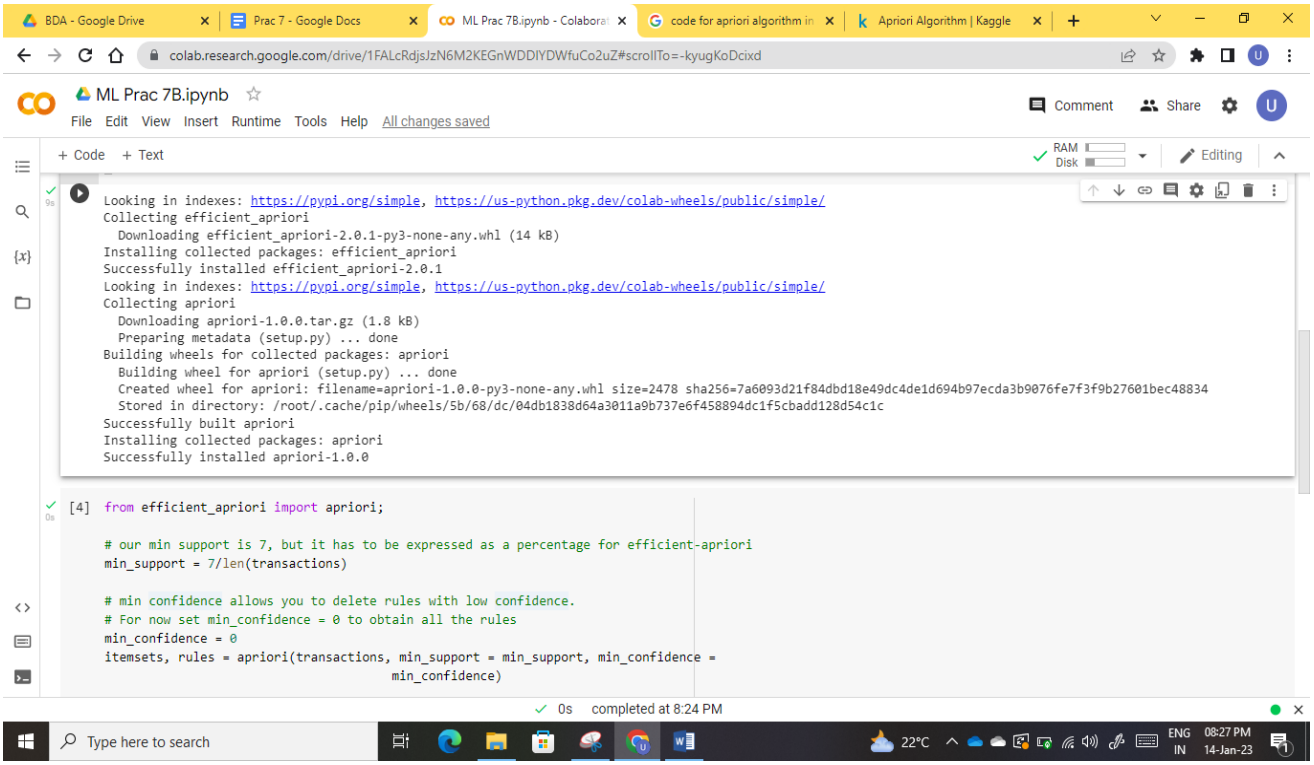
Code :

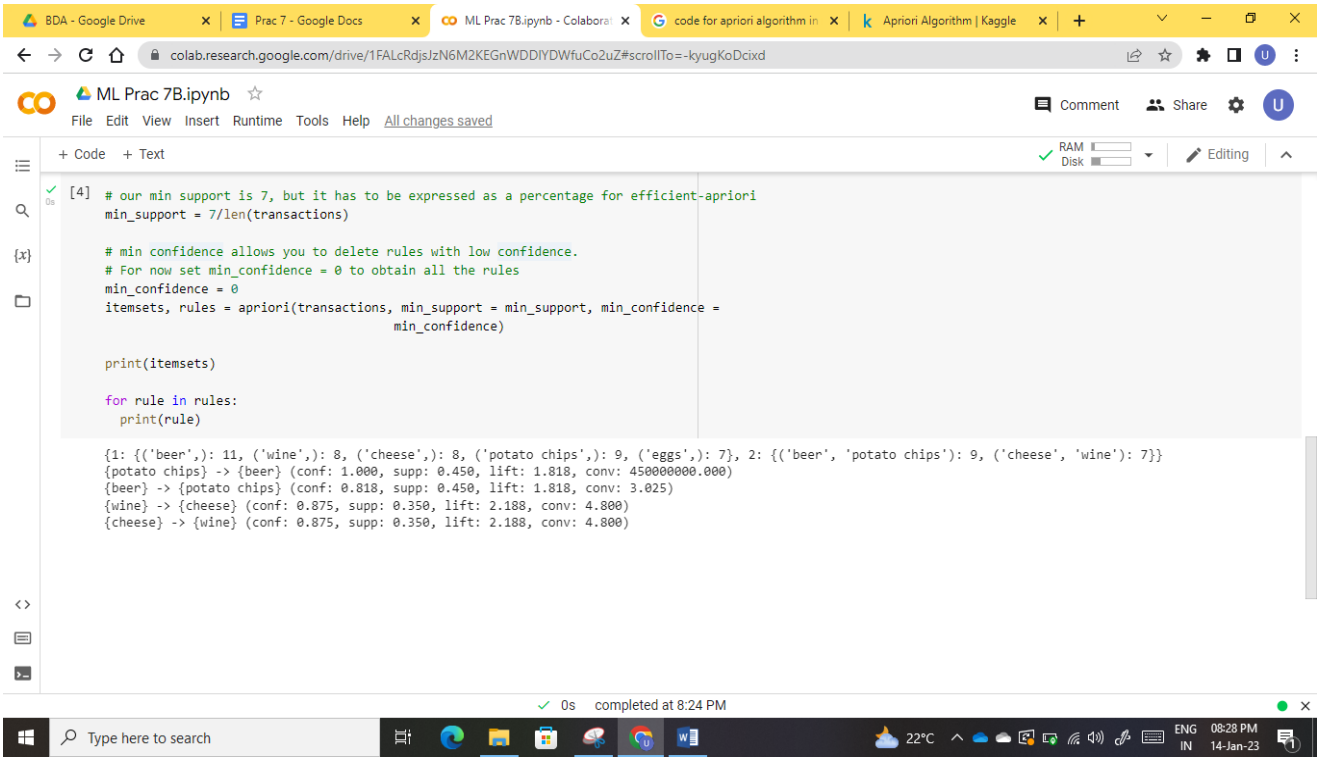
```
!pip install efficient_apriori
!pip install apriori
# store the item sets as tuples of strings in a list
transactions = [
                ("beer", "wine", "cheese"),
                ("beer", "potato chips"),
                ("eggs", "flour", "butter", "cheese"),
                ("eggs", "flour", "butter", "beer", "potato chips"),
                ("wine", "cheese"),
                ("potato chips"),
                ("eggs", "flour", "butter", "wine", "cheese"),
                ("eggs", "flour", "butter", "beer", "potato chips"),
                ("wine", "beer"),
                ("beer", "potato chips"),
                ("butter", "eggs"),
                ("beer", "potato chips"),
                ("flour", "eggs"),
                ("beer", "potato chips"),
                ("eggs", "flour", "butter", "wine", "cheese"),
                ("beer", "wine", "potato chips", "cheese"),
                ("wine", "cheese"),
                ("beer", "potato chips"),
                ("wine", "cheese"),
                ("beer", "potato chips"),
        ]

from efficient_apriori import apriori;
# our min support is 7, but it has to be expressed as a percentage for efficient-apriori
min_support = 7/len(transactions)
```

```
# min confidence allows you to delete rules with low confidence.
# For now set min_confidence = 0 to obtain all the rules
min_confidence = 0
itemsets, rules = apriori(transactions, min_support = min_support, min_confidence =
                          min_confidence)

print(itemsets)

for rule in rules:
        print(rule)
```

Output :




BDA - Google Drive
Prac 7 - Google Docs
ML Prac 7B.ipynb - Colaborat
code for apriori algorithm in
Apriori Algorithm | Kaggle
colab.research.google.com/drive/1FALcRdjsJzN6M2KEGnWDDIYDWfuCo2uZ#scrollTo=-kyugKoDcixd
ML Prac 7B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting efficient_apriori
Downloading efficient_apriori-2.0.1-py3-none-any.whl (14 kB)
Installing collected packages: efficient_apriori
Successfully installed efficient_apriori-2.0.1
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting apriori
Downloading apriori-1.0.0.tar.gz (1.8 kB)
Preparing metadata (setup.py) ... done
Building wheels for collected packages: apriori
Building wheel for apriori (setup.py) ... done
Created wheel for apriori: filename=apriori-1.0.0-py3-none-any.whl size=2478 sha256=7a6093d21f84dbd18e49dc4de1d694b97ecda3b9076fe7f3f9b27601bec48834
Stored in directory: /root/.cache/pip/wheels/5b/68/dc/04db1838d64a3011a9b737e6f458894dc1f5cbadd128d54c1c
Successfully built apriori
Installing collected packages: apriori
Successfully installed apriori-1.0.0
[4] from efficient_apriori import apriori;
# our min support is 7, but it has to be expressed as a percentage for efficient-apriori
min_support = 7/len(transactions)
# min confidence allows you to delete rules with low confidence.
# For now set min_confidence = 0 to obtain all the rules
min_confidence = 0
itemsets, rules = apriori(transactions, min_support = min_support, min_confidence =
min_confidence)
0s completed at 8:24 PM
Type here to search
22°C
ENG IN
08:27 PM
14-Jan-23

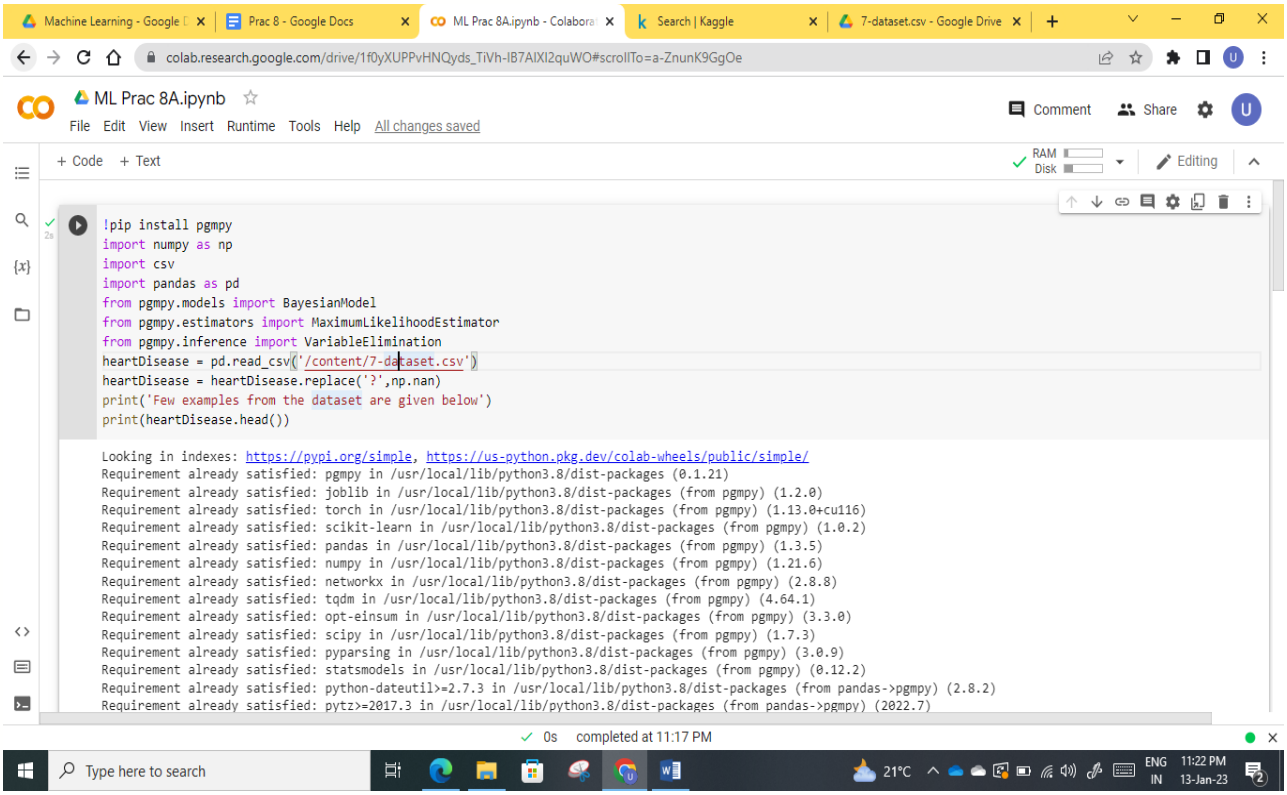

BDA - Google Drive
Prac 7 - Google Docs
ML Prac 7B.ipynb - Colaborat
code for apriori algorithm in
Apriori Algorithm | Kaggle
colab.research.google.com/drive/1FALcRdjsJzN6M2KEGnWDDIYDWfuCo2uZ#scrollTo=-kyugKoDcixd
ML Prac 7B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[4] # our min support is 7, but it has to be expressed as a percentage for efficient-apriori
min_support = 7/len(transactions)
# min confidence allows you to delete rules with low confidence.
# For now set min_confidence = 0 to obtain all the rules
min_confidence = 0
itemsets, rules = apriori(transactions, min_support = min_support, min_confidence =
min_confidence)
print(itemsets)
for rule in rules:
print(rule)
{1: {('beer',): 11, ('wine',): 8, ('cheese',): 8, ('potato chips',): 9, ('eggs',): 7}, 2: {('beer', 'potato chips'): 9, ('cheese', 'wine'): 7}}
{potato chips} -> {beer} (conf: 1.000, supp: 0.450, lift: 1.818, conv: 450000000.000)
{beer} -> {potato chips} (conf: 0.818, supp: 0.450, lift: 1.818, conv: 3.025)
{wine} -> {cheese} (conf: 0.875, supp: 0.350, lift: 2.188, conv: 4.800)
{cheese} -> {wine} (conf: 0.875, supp: 0.350, lift: 2.188, conv: 4.800)
0s completed at 8:24 PM
Type here to search
22°C
ENG IN
08:28 PM
14-Jan-23

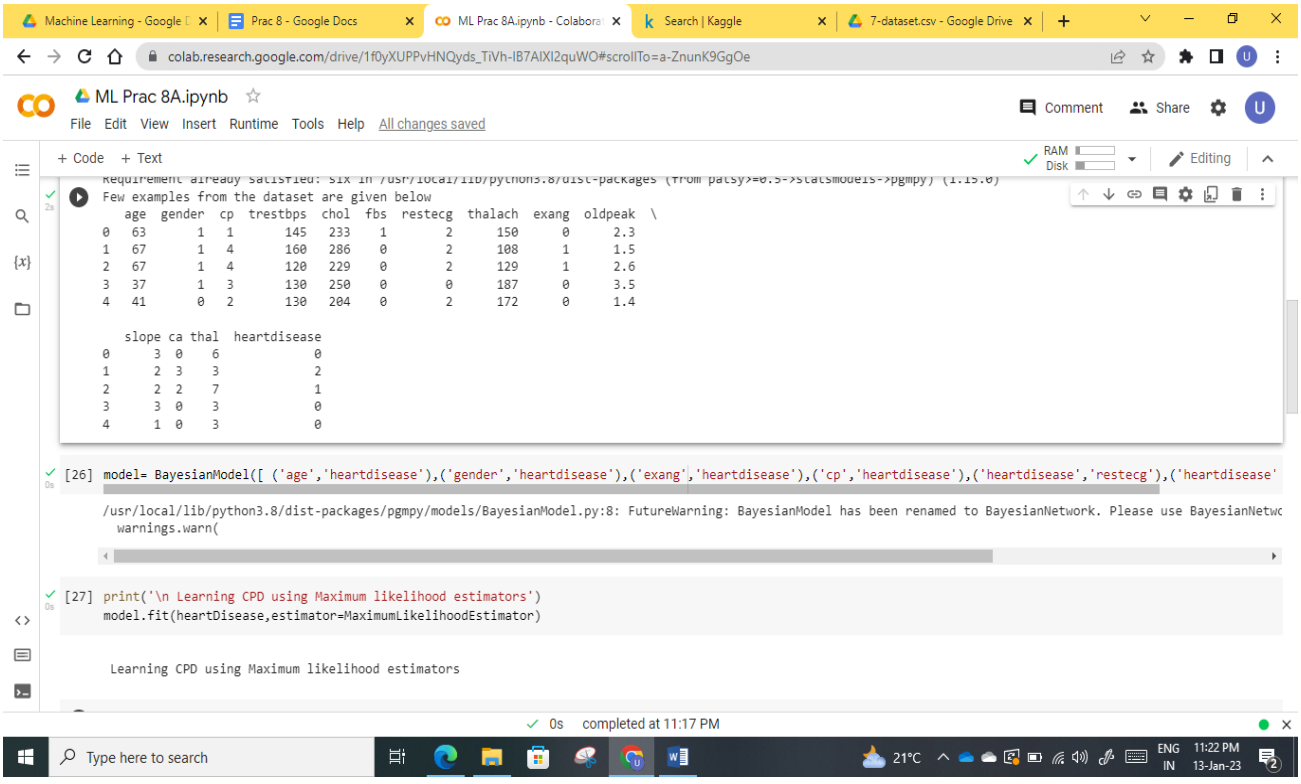
Write a program to construct a Bayesian network considering medical data. Use this model to demonstrate the diagnosis of heart patients using standard Heart Disease Data Set

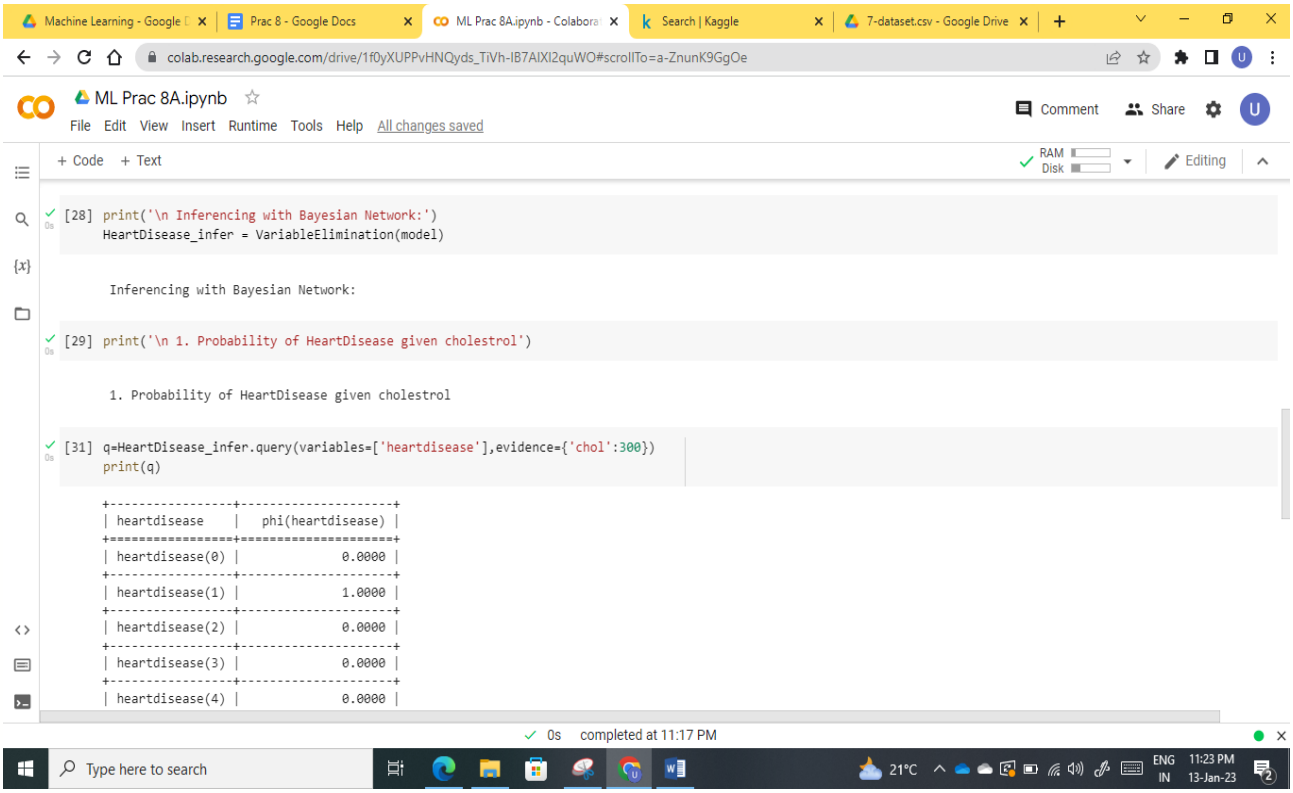
Code :

```
!pip install pgmpy
import numpy as np
import csv
import pandas as pd
from pgmpy.models import BayesianModel
from pgmpy.estimators import MaximumLikelihoodEstimator
from pgmpy.inference import VariableElimination
heartDisease = pd.read_csv('/content/7-dataset.csv')
heartDisease = heartDisease.replace('?',np.nan)
print('Few examples from the dataset are given below')
print(heartDisease.head())

model= BayesianModel([('age','heartdisease'),('gender','heartdisease'), ('exang', 'heartdisease') ,
                      ('cp','heartdisease'),('heartdisease','restecg'),('heartdisease','chol')])
print('\n Learning CPD using Maximum likelihood estimators')
model.fit(heartDisease,estimator=MaximumLikelihoodEstimator)

print('\n Inferencing with Bayesian Network:')
HeartDisease_infer = VariableElimination(model)
print('\n 1. Probability of HeartDisease given cholestrol')
q=HeartDisease_infer.query(variables=['heartdisease'],evidence={'chol':300})
print(q)
print('\n 2. Probability of HeartDisease given  cp=2')
q=HeartDisease_infer.query(variables=['heartdisease'],evidence={'cp':2})
print(q)
```

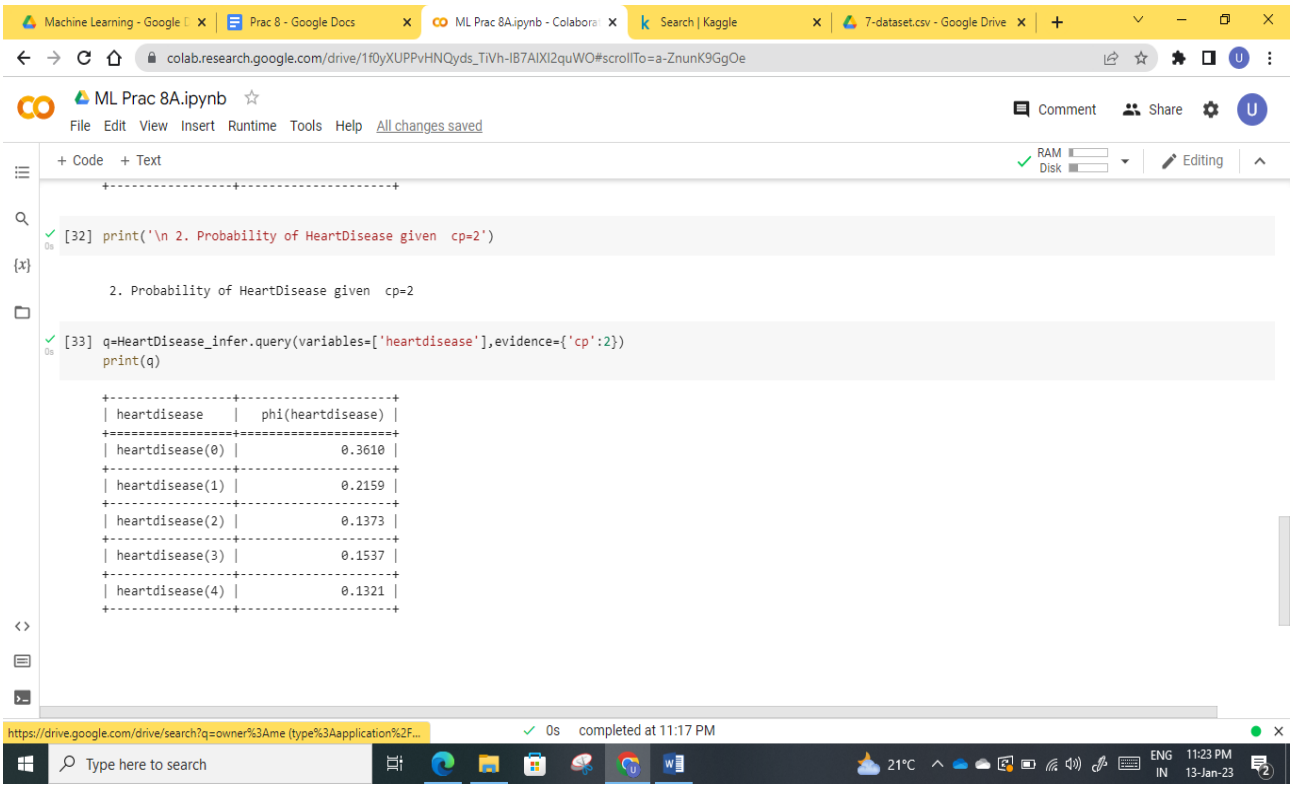
Output :


Machine Learning - Google
Prac 8 - Google Docs
ML Prac 8A.ipynb - Colabora
Search | Kaggle
7-dataset.csv - Google Drive
colab.research.google.com/drive/1f0yXUPPvHNQyds_TiVh-IB7AIXI2quWO#scrollTo=a-ZnunK9GgOe
ML Prac 8A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
!pip install pgmpy
import numpy as np
import csv
import pandas as pd
from pgmpy.models import BayesianModel
from pgmpy.estimators import MaximumLikelihoodEstimator
from pgmpy.inference import VariableElimination
heartDisease = pd.read_csv('/content/7-dataset.csv')
heartDisease = heartDisease.replace('?',np.nan)
print('Few examples from the dataset are given below')
print(heartDisease.head())
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: pgmpy in /usr/local/lib/python3.8/dist-packages (0.1.21)
Requirement already satisfied: joblib in /usr/local/lib/python3.8/dist-packages (from pgmpy) (1.2.0)
Requirement already satisfied: torch in /usr/local/lib/python3.8/dist-packages (from pgmpy) (1.13.0+cu116)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.8/dist-packages (from pgmpy) (1.0.2)
Requirement already satisfied: pandas in /usr/local/lib/python3.8/dist-packages (from pgmpy) (1.3.5)
Requirement already satisfied: numpy in /usr/local/lib/python3.8/dist-packages (from pgmpy) (1.21.6)
Requirement already satisfied: networkx in /usr/local/lib/python3.8/dist-packages (from pgmpy) (2.8.8)
Requirement already satisfied: tqdm in /usr/local/lib/python3.8/dist-packages (from pgmpy) (4.64.1)
Requirement already satisfied: opt-einsum in /usr/local/lib/python3.8/dist-packages (from pgmpy) (3.3.0)
Requirement already satisfied: scipy in /usr/local/lib/python3.8/dist-packages (from pgmpy) (1.7.3)
Requirement already satisfied: pyparsing in /usr/local/lib/python3.8/dist-packages (from pgmpy) (3.0.9)
Requirement already satisfied: statsmodels in /usr/local/lib/python3.8/dist-packages (from pgmpy) (0.12.2)
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.8/dist-packages (from pandas->pgmpy) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.8/dist-packages (from pandas->pgmpy) (2022.7)
0s completed at 11:17 PM
Type here to search
21°C
ENG IN
11:22 PM
13-Jan-23


Machine Learning - Google
Prac 8 - Google Docs
ML Prac 8A.ipynb - Colabora
Search | Kaggle
7-dataset.csv - Google Drive
colab.research.google.com/drive/1f0yXUPPvHNQyds_TiVh-IB7AIXI2quWO#scrollTo=a-ZnunK9GgOe
ML Prac 8A.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
Few examples from the dataset are given below
age gender cp trestbps chol fbs restecg thalach exang oldpeak \
0 63 1 1 145 233 1 2 150 0 2.3
1 67 1 4 160 286 0 2 108 1 1.5
2 67 1 4 120 229 0 2 129 1 2.6
3 37 1 3 130 250 0 0 187 0 3.5
4 41 0 2 130 204 0 2 172 0 1.4
slope ca thal heartdisease
0 3 0 6 0
1 2 3 3 2
2 2 2 7 1
3 3 0 3 0
4 1 0 3 0
[26] model= BayesianModel([('age','heartdisease'),('gender','heartdisease'),('exang','heartdisease'),('cp','heartdisease'),('heartdisease','restecg'),('heartdisease'
/usr/local/lib/python3.8/dist-packages/pgmpy/models/BayesianModel.py:8: FutureWarning: BayesianModel has been renamed to BayesianNetwork. Please use BayesianNetwo
warnings.warn(
[27] print('\n Learning CPD using Maximum likelihood estimators')
model.fit(heartDisease,estimator=MaximumLikelihoodEstimator)
Learning CPD using Maximum likelihood estimators
0s completed at 11:17 PM
Type here to search
21°C
ENG IN
11:22 PM
13-Jan-23



```
[28] print('\n Inferencing with Bayesian Network:')
     HeartDisease_infer = VariableElimination(model)

 Inferencing with Bayesian Network:

[29] print('\n 1. Probability of HeartDisease given cholestrol')

 1. Probability of HeartDisease given cholestrol

[31] q=HeartDisease_infer.query(variables=['heartdisease'],evidence={'chol':300})
     print(q)

+-----------------+---------------------+
| heartdisease    |   phi(heartdisease) |
+=================+=====================+
| heartdisease(0) |              0.0000 |
+-----------------+---------------------+
| heartdisease(1) |              1.0000 |
+-----------------+---------------------+
| heartdisease(2) |              0.0000 |
+-----------------+---------------------+
| heartdisease(3) |              0.0000 |
+-----------------+---------------------+
| heartdisease(4) |              0.0000 |
```



```
+-----------------+---------------------+

[32] print('\n 2. Probability of HeartDisease given  cp=2')

 2. Probability of HeartDisease given  cp=2

[33] q=HeartDisease_infer.query(variables=['heartdisease'],evidence={'cp':2})
     print(q)

+-----------------+---------------------+
| heartdisease    |   phi(heartdisease) |
+=================+=====================+
| heartdisease(0) |              0.3610 |
+-----------------+---------------------+
| heartdisease(1) |              0.2159 |
+-----------------+---------------------+
| heartdisease(2) |              0.1373 |
+-----------------+---------------------+
| heartdisease(3) |              0.1537 |
+-----------------+---------------------+
| heartdisease(4) |              0.1321 |
+-----------------+---------------------+
```

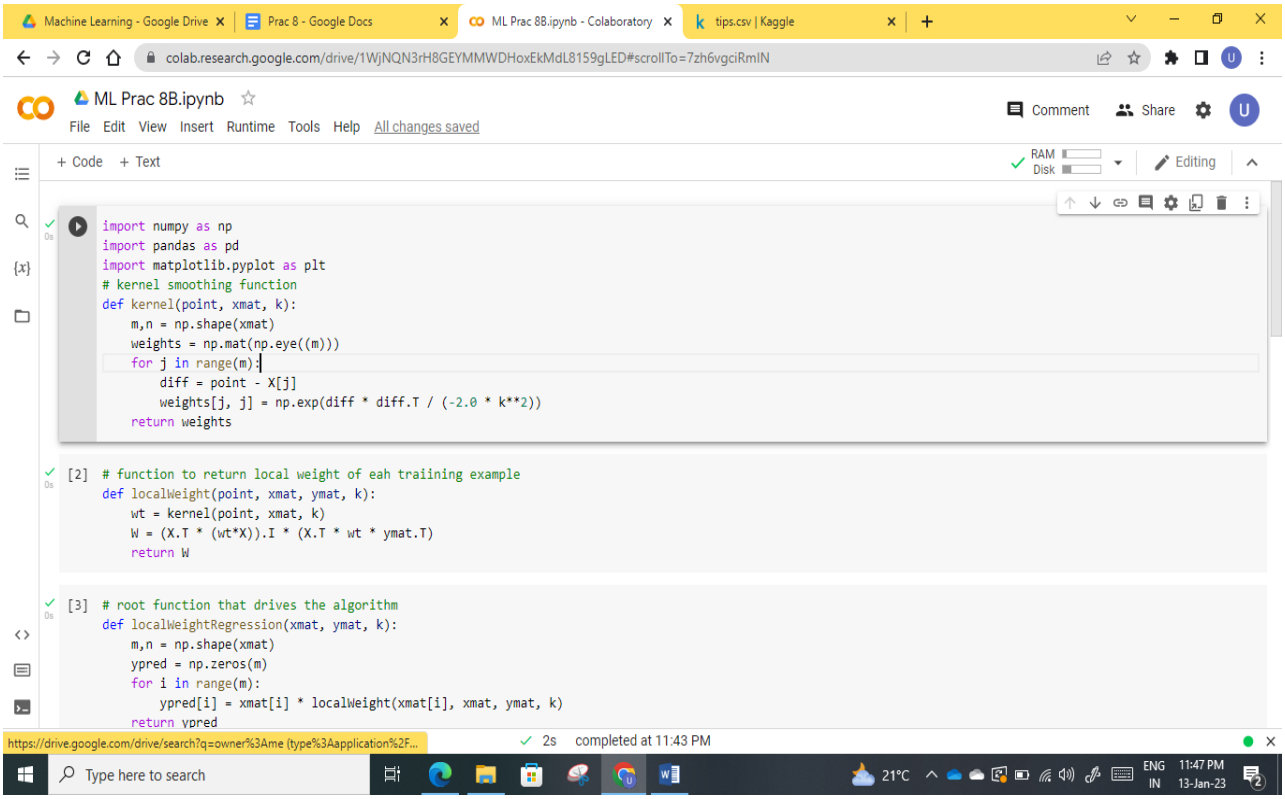
Implement the non-parametric Locally Weighted Regression algorithm in order to fit data points. Select appropriate data set for your experiment and draw graphs

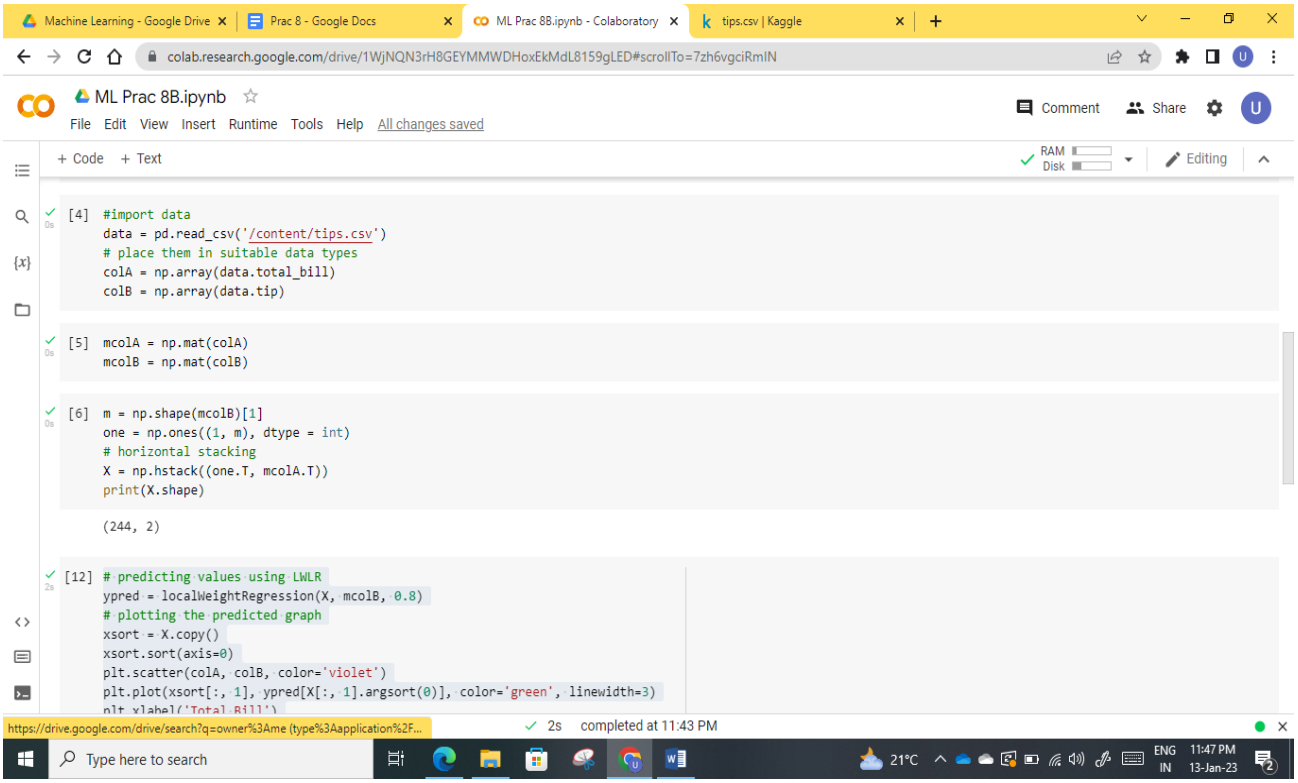
Code :

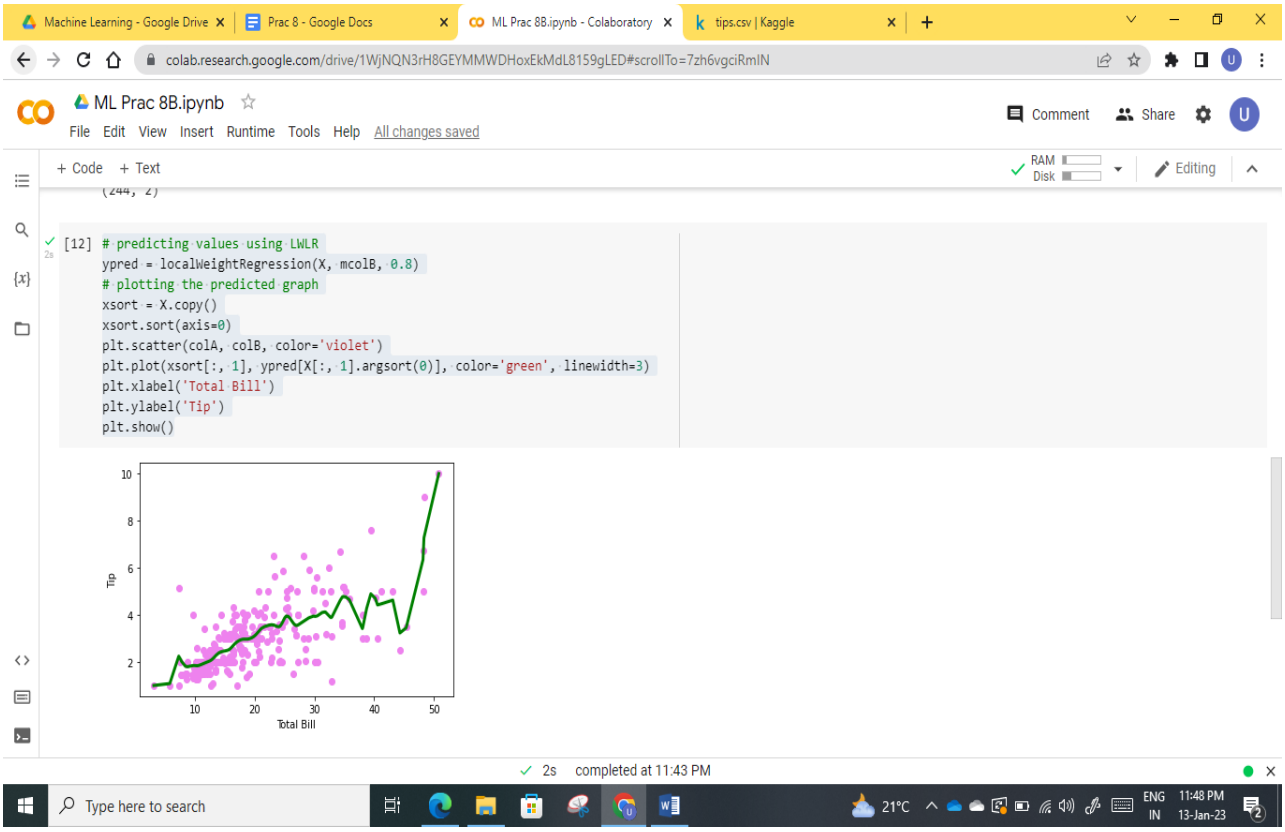
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
# kernel smoothing function
def kernel(point, xmat, k):
    m,n = np.shape(xmat)
    weights = np.mat(np.eye((m)))
    for j in range(m):
        diff = point - X[j]
        weights[j, j] = np.exp(diff * diff.T / (-2.0 * k**2))
    return weights

# function to return local weight of eah traiining example
def localWeight(point, xmat, ymat, k):
    wt = kernel(point, xmat, k)
    W = (X.T * (wt*X)).I * (X.T * wt * ymat.T)
    return W

# root function that drives the algorithm
def localWeightRegression(xmat, ymat, k):
    m,n = np.shape(xmat)
    ypred = np.zeros(m)
    for i in range(m):
        ypred[i] = xmat[i] * localWeight(xmat[i], xmat, ymat, k)
    return ypred

#import data
```

```
data = pd.read_csv('/content/tips.csv')
# place them in suitable data types
colA = np.array(data.total_bill)
colB = np.array(data.tip)

mcolA = np.mat(colA)
mcolB = np.mat(colB)
m = np.shape(mcolB)[1]
one = np.ones((1, m), dtype = int)
# horizontal stacking
X = np.hstack((one.T, mcolA.T))
print(X.shape)

# predicting values using LWLR
ypred = localWeightRegression(X, mcolB, 0.8)
# plotting the predicted graph
xsort = X.copy()
xsort.sort(axis=0)
plt.scatter(colA, colB, color='violet')
plt.plot(xsort[:, 1], ypred[X[:, 1].argsort(0)], color='green', linewidth=3)
plt.xlabel('Total Bill')
plt.ylabel('Tip')
plt.show()
```

Output :


Machine Learning - Google Drive
Prac 8 - Google Docs
ML Prac 8B.ipynb - Colaboratory
tips.csv | Kaggle
colab.research.google.com/drive/1WjNQN3rH8GEYMMWDHoxEkMdL8159gLED#scrollTo=7zh6vgciRmIN
ML Prac 8B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
# kernel smoothing function
def kernel(point, xmat, k):
m,n = np.shape(xmat)
weights = np.mat(np.eye((m)))
for j in range(m):
diff = point - X[j]
weights[j, j] = np.exp(diff * diff.T / (-2.0 * k**2))
return weights
[2] # function to return local weight of eah traiining example
def localWeight(point, xmat, ymat, k):
wt = kernel(point, xmat, k)
W = (X.T * (wt*X)).I * (X.T * wt * ymat.T)
return W
[3] # root function that drives the algorithm
def localWeightRegression(xmat, ymat, k):
m,n = np.shape(xmat)
ypred = np.zeros(m)
for i in range(m):
ypred[i] = xmat[i] * localWeight(xmat[i], xmat, ymat, k)
return ypred
2s completed at 11:43 PM
Type here to search
21°C
ENG IN
11:47 PM 13-Jan-23


Machine Learning - Google Drive
Prac 8 - Google Docs
ML Prac 8B.ipynb - Colaboratory
tips.csv | Kaggle
colab.research.google.com/drive/1WjNQN3rH8GEYMMWDHoxEkMdL8159gLED#scrollTo=7zh6vgciRmIN
ML Prac 8B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[4] #import data
data = pd.read_csv('/content/tips.csv')
# place them in suitable data types
colA = np.array(data.total_bill)
colB = np.array(data.tip)
[5] mcolA = np.mat(colA)
mcolB = np.mat(colB)
[6] m = np.shape(mcolB)[1]
one = np.ones((1, m), dtype = int)
# horizontal stacking
X = np.hstack((one.T, mcolA.T))
print(X.shape)
(244, 2)
[12] # predicting values using LWLR
ypred = localWeightRegression(X, mcolB, 0.8)
# plotting the predicted graph
xsort = X.copy()
xsort.sort(axis=0)
plt.scatter(colA, colB, color='violet')
plt.plot(xsort[:, 1], ypred[X[:, 1].argsort(0)], color='green', linewidth=3)
plt.xlabel('Total Bill')
2s completed at 11:43 PM
Type here to search
21°C
ENG IN
11:47 PM 13-Jan-23


Machine Learning - Google Drive
Prac 8 - Google Docs
ML Prac 8B.ipynb - Colaboratory
tips.csv | Kaggle
colab.research.google.com/drive/1WjNQN3rH8GEYMMWDHoxEkMdL8159gLED#scrollTo=7zh6vgciRmIN
ML Prac 8B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
(244, 2)
[12] # predicting values using LWLR
ypred = localWeightRegression(X, mcolB, 0.8)
# plotting the predicted graph
xsort = X.copy()
xsort.sort(axis=0)
plt.scatter(colA, colB, color='violet')
plt.plot(xsort[:, 1], ypred[X[:, 1].argsort(0)], color='green', linewidth=3)
plt.xlabel('Total Bill')
plt.ylabel('Tip')
plt.show()
Tip
Total Bill
2s completed at 11:43 PM
Type here to search
21°C
ENG
IN
11:48 PM
13-Jan-23

Build an Artificial Neural Network by implementing the Backpropagation algorithm and test the same using appropriate data sets

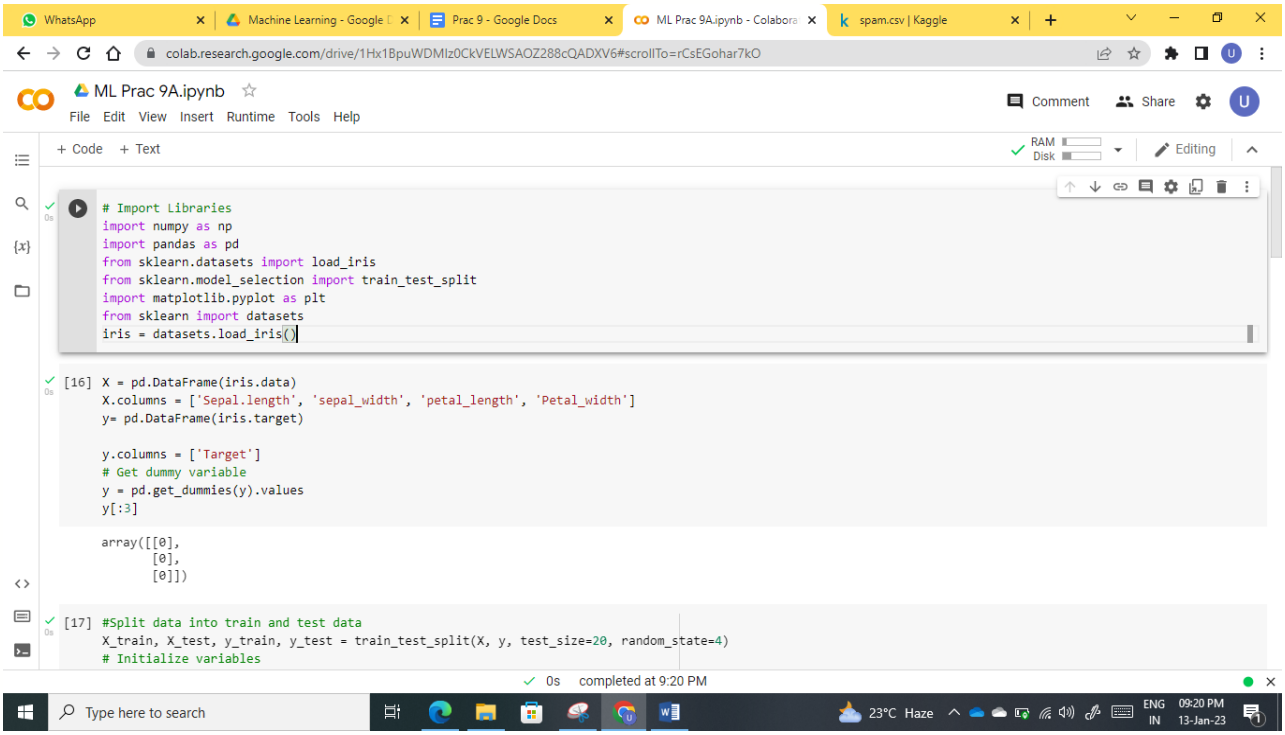
Code :

```
# Import Libraries
import numpy as np
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
from sklearn import datasets
iris = datasets.load_iris()

X = pd.DataFrame(iris.data)
X.columns = ['Sepal.length', 'sepal_width', 'petal_length', 'Petal_width']
y= pd.DataFrame(iris.target)

y.columns = ['Target']
# Get dummy variable
y = pd.get_dummies(y).values
y[:3]

#Split data into train and test data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=20, random_state=4)
# Initialize variables
learning_rate = 0.1
iterations = 5000
N = y_train.size
# number of input features
input_size = 4
# number of hidden layers neurons
hidden_size = 2
```

```
# number of neurons at the output layer
output_size = 3
results = pd.DataFrame(columns=["mse", "accuracy"])
# Initialize weights
np.random.seed(10)

# initializing weight for the hidden layer
W1 = np.random.normal(scale=0.5, size=(input_size, hidden_size))
# initializing weight for the output layer
W2 = np.random.normal(scale=0.5, size=(hidden_size , output_size))

def sigmoid(x):
        return 1 / (1 + np.exp(-x))
def mean_squared_error(y_pred, y_true):
        return ((y_pred - y_true)**2).sum() / (2*y_pred.size)
def accuracy(y_pred, y_true):
        acc = y_pred.argmax(axis=1) == y_true.argmax(axis=1)
        return acc.mean()
for itr in range(iterations):
        # feedforward propagation
        # on hidden layer
        Z1 = np.dot(X_train, W1)
        A1 = sigmoid(Z1)

        # on output layer
        Z2 = np.dot(A1, W2)
        A2 = sigmoid(Z2)

        # Calculating error
        mse = mean_squared_error(A2, y_train)
        acc = accuracy(A2, y_train)
        results=results.append({"mse":mse,"accuracy":acc},
                        ignore_index=True)
```

```
        # backpropagation
        E1 = A2 - y_train
        dW1 = E1 * A2 * (1 - A2)

        E2 = np.dot(dW1, W2.T)
        dW2 = E2 * A1 * (1 - A1)

        # weight updates
        W2_update = np.dot(A1.T, dW1) / N
        W1_update = np.dot(X_train.T, dW2) / N

        W2 = W2 - learning_rate * W2_update
        W1 = W1 - learning_rate * W1_update
results.mse.plot(title="Mean Squared Error")

results.accuracy.plot(title="Accuracy")

# feedforward
Z1 = np.dot(X_test, W1)
A1 = sigmoid(Z1)
Z2 = np.dot(A1, W2)
A2 = sigmoid(Z2)
acc = accuracy(A2, y_test)
print("Accuracy: {}".format(acc))
```

Output :

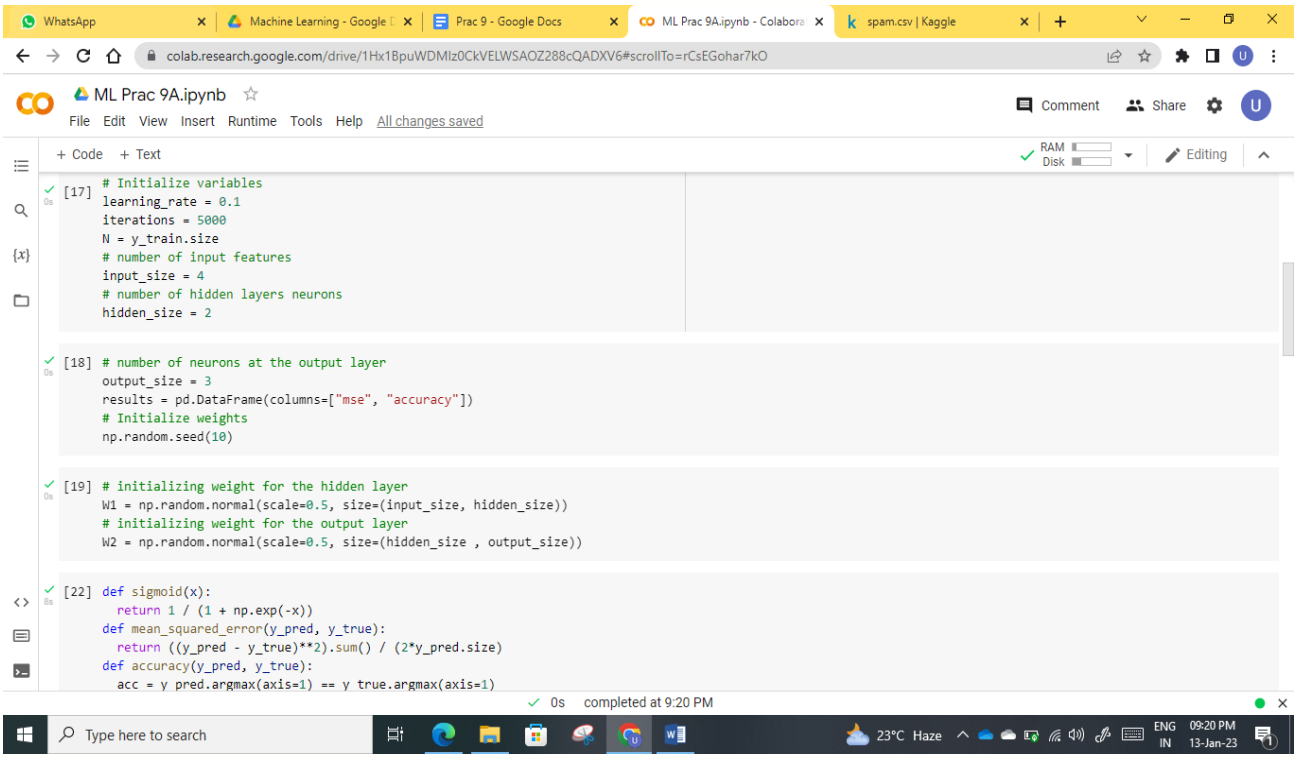


```
# Import Libraries
import numpy as np
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
from sklearn import datasets
iris = datasets.load_iris()
```

```
[16] X = pd.DataFrame(iris.data)
     X.columns = ['Sepal.length', 'sepal_width', 'petal_length', 'Petal_width']
     y= pd.DataFrame(iris.target)

     y.columns = ['Target']
     # Get dummy variable
     y = pd.get_dummies(y).values
     y[:3]

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

```
[17] #Split data into train and test data
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=20, random_state=4)
     # Initialize variables
```

0s completed at 9:20 PM



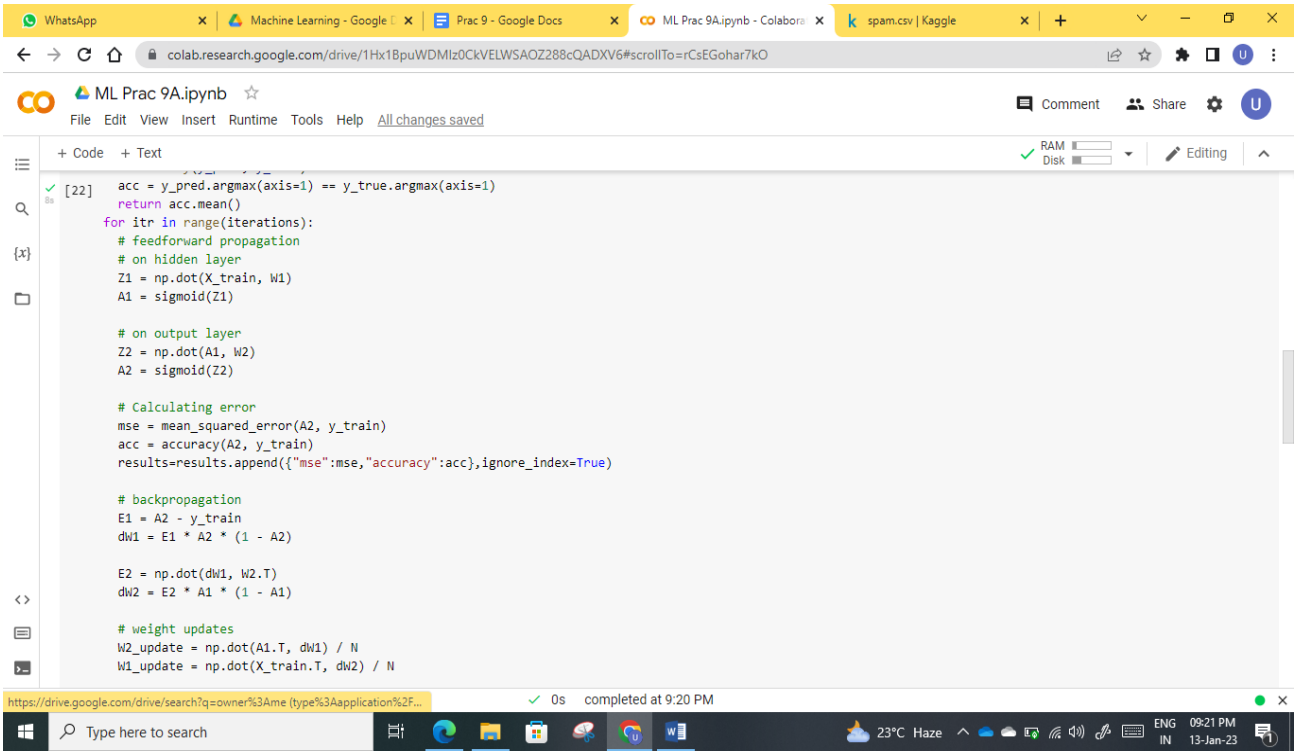
```
[17] # Initialize variables
     learning_rate = 0.1
     iterations = 5000
     N = y_train.size
     # number of input features
     input_size = 4
     # number of hidden layers neurons
     hidden_size = 2
```

```
[18] # number of neurons at the output layer
     output_size = 3
     results = pd.DataFrame(columns=["mse", "accuracy"])
     # Initialize weights
     np.random.seed(10)
```

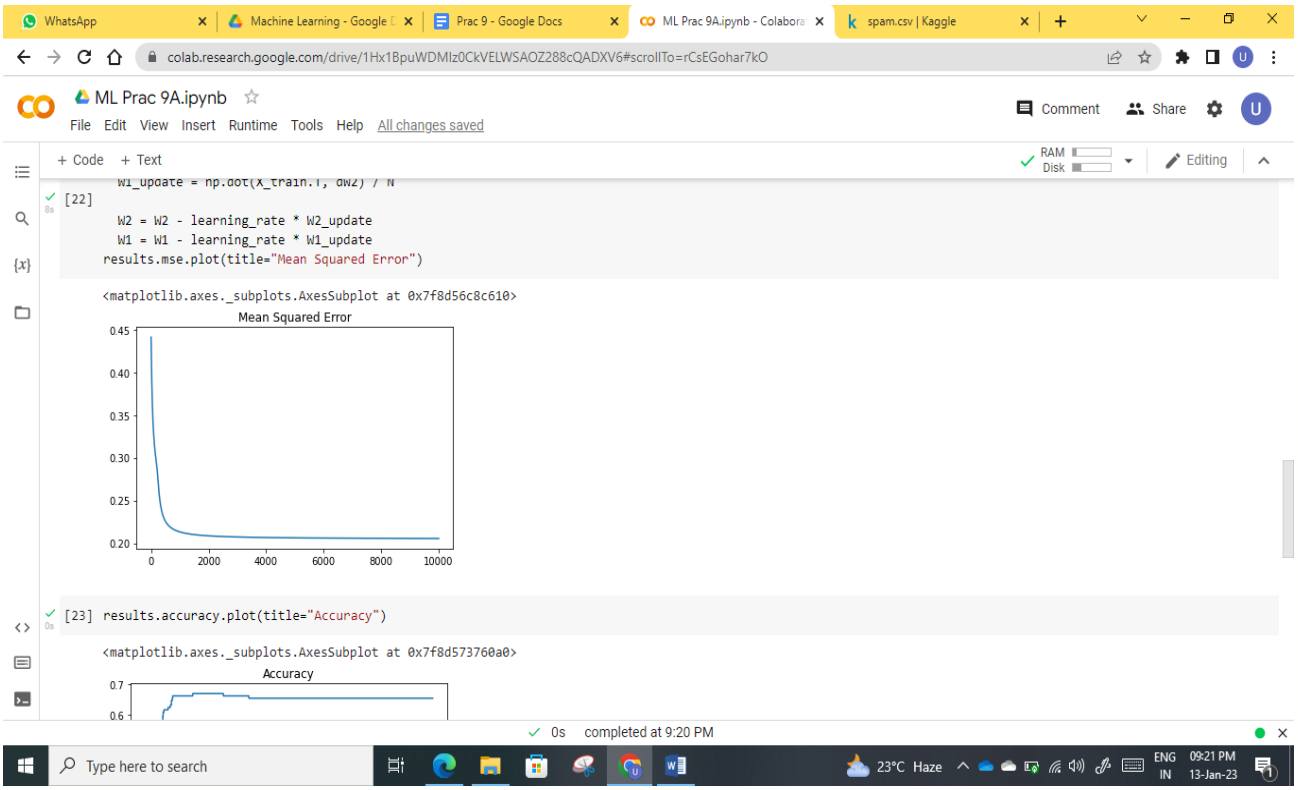
```
[19] # initializing weight for the hidden layer
     W1 = np.random.normal(scale=0.5, size=(input_size, hidden_size))
     # initializing weight for the output layer
     W2 = np.random.normal(scale=0.5, size=(hidden_size , output_size))
```

```
[22] def sigmoid(x):
       return 1 / (1 + np.exp(-x))
     def mean_squared_error(y_pred, y_true):
       return ((y_pred - y_true)**2).sum() / (2*y_pred.size)
     def accuracy(y_pred, y_true):
       acc = y_pred.argmax(axis=1) == y_true.argmax(axis=1)
```

0s completed at 9:20 PM



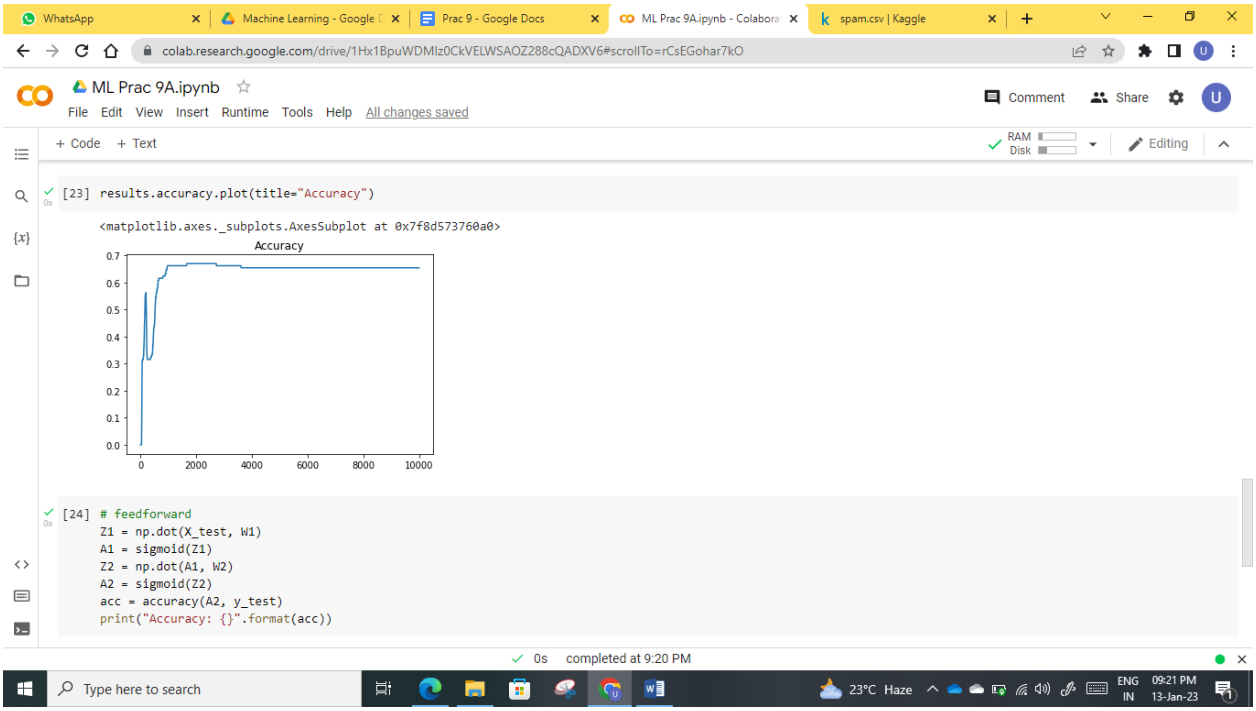
```
[22]    acc = y_pred.argmax(axis=1) == y_true.argmax(axis=1)
        return acc.mean()
    for itr in range(iterations):
      # feedforward propagation
      # on hidden layer
      Z1 = np.dot(X_train, W1)
      A1 = sigmoid(Z1)

      # on output layer
      Z2 = np.dot(A1, W2)
      A2 = sigmoid(Z2)

      # Calculating error
      mse = mean_squared_error(A2, y_train)
      acc = accuracy(A2, y_train)
      results=results.append({"mse":mse,"accuracy":acc},ignore_index=True)

      # backpropagation
      E1 = A2 - y_train
      dW1 = E1 * A2 * (1 - A2)

      E2 = np.dot(dW1, W2.T)
      dW2 = E2 * A1 * (1 - A1)

      # weight updates
      W2_update = np.dot(A1.T, dW1) / N
      W1_update = np.dot(X_train.T, dW2) / N
```



```
[22]    W2 = W2 - learning_rate * W2_update
        W1 = W1 - learning_rate * W1_update
    results.mse.plot(title="Mean Squared Error")

<matplotlib.axes._subplots.AxesSubplot at 0x7f8d56c8c610>
```



```
[23] results.accuracy.plot(title="Accuracy")

<matplotlib.axes._subplots.AxesSubplot at 0x7f8d573760a0>
```




```
[24] # feedforward
     Z1 = np.dot(X_test, W1)
     A1 = sigmoid(Z1)
     Z2 = np.dot(A1, W2)
     A2 = sigmoid(Z2)
     acc = accuracy(A2, y_test)
     print("Accuracy: {}".format(acc))
```

```
Accuracy: 0.85
```

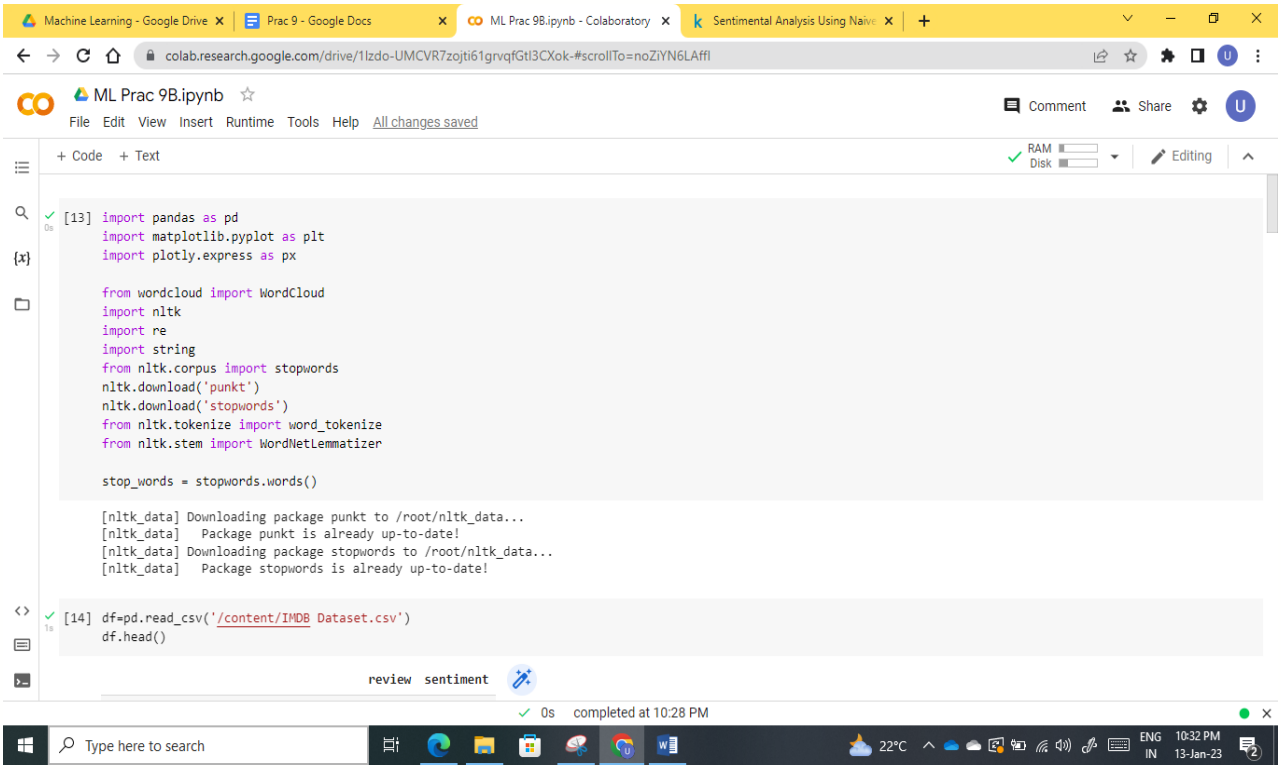
Assuming a set of documents that need to be classified, use the naïve Bayesian Classifier model to perform this task.

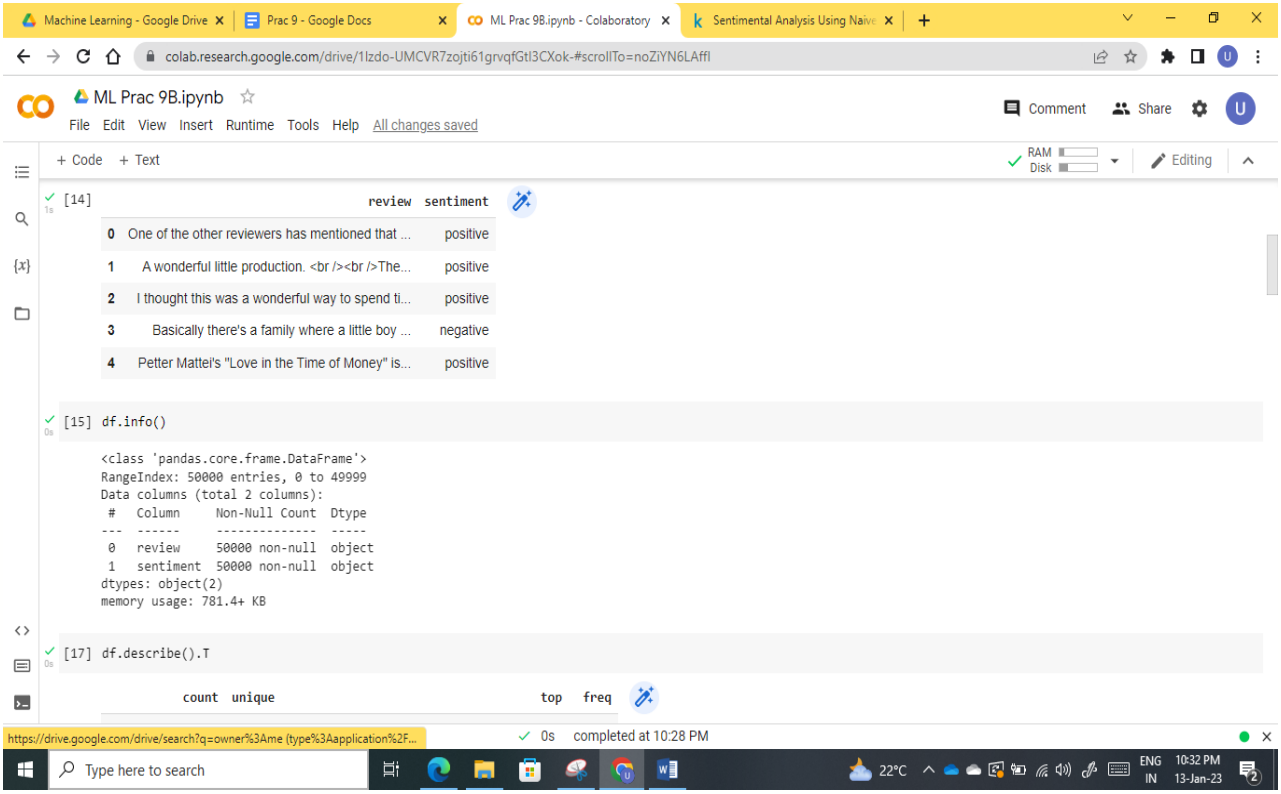
Code :

```
import pandas as pd
import matplotlib.pyplot as plt
import plotly.express as px

from wordcloud import WordCloud
import nltk
import re
import string
from nltk.corpus import stopwords
nltk.download('punkt')
nltk.download('stopwords')
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer

stop_words = stopwords.words()

df=pd.read_csv('/content/IMDB Dataset.csv')
df.head()

df.info()
df.describe().T

#sentiment count:
df['sentiment'].value_counts()

df['review'].str.len().hist()

fig,(ax1,ax2)=plt.subplots(1,2,figsize=(12,8))
```

```
ax1.hist(df[df['sentiment']=='positive']['review'].str.len())
ax1.set_title( 'Positive Reviews')
ax2.hist(df[df['sentiment']=='negative']['review'].str.len())
ax2.set_title( 'Negative Reviews')

text = " ".join(i for i in df[df['sentiment']=='positive']['review'])
wordcloud = WordCloud( background_color="white").generate(text)

plt.figure( figsize=(15,10))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title('wordcloud for positive review')
plt.show()

text = " ".join(i for i in df[df['sentiment']=='negative']['review'])
#stopwords = set(STOPWORDS)
wordcloud = WordCloud( background_color="white").generate(text)
#wordcloud = WordCloud(stopwords=stopwords, background_color="white").generate(text)
plt.figure( figsize=(15,10))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title('wordcloud for negative review')
plt.show()

df.rename(columns={'review':'text'}, inplace = True)
df
df['sentiment']
```

Output :


Machine Learning - Google Drive
Prac 9 - Google Docs
ML Prac 9B.ipynb - Colaboratory
Sentimental Analysis Using Naive
colab.research.google.com/drive/1lzdo-UMCVR7zojti61grvqfGtl3CXok-#scrollTo=noZiYN6LAffl
ML Prac 9B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[13] import pandas as pd
import matplotlib.pyplot as plt
import plotly.express as px
from wordcloud import WordCloud
import nltk
import re
import string
from nltk.corpus import stopwords
nltk.download('punkt')
nltk.download('stopwords')
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
stop_words = stopwords.words()
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data] Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data] Package stopwords is already up-to-date!
[14] df=pd.read_csv('/content/IMDB Dataset.csv')
df.head()
review sentiment
0s completed at 10:28 PM
Type here to search
22°C
ENG IN
10:32 PM 13-Jan-23


Machine Learning - Google Drive
Prac 9 - Google Docs
ML Prac 9B.ipynb - Colaboratory
Sentimental Analysis Using Naive
colab.research.google.com/drive/1lzdo-UMCVR7zojti61grvqfGtl3CXok-#scrollTo=noZiYN6LAffl
ML Prac 9B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[14]
review sentiment
0 One of the other reviewers has mentioned that ... positive
1 A wonderful little production.

The... positive
2 I thought this was a wonderful way to spend ti... positive
3 Basically there's a family where a little boy ... negative
4 Petter Mattei's "Love in the Time of Money" is... positive
[15] df.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 2 columns):
Column Non-Null Count Dtype
--- ------ -------------- -----
0 review 50000 non-null object
1 sentiment 50000 non-null object
dtypes: object(2)
memory usage: 781.4+ KB
[17] df.describe().T
count unique top freq
https://drive.google.com/drive/search?q=owner%3Ame (type%3Aapplication%2F...
0s completed at 10:28 PM
Type here to search
22°C
ENG IN
10:32 PM 13-Jan-23

Machine Learning - Google Drive | Prac 9 - Google Docs | ML Prac 9B.ipynb - Colaboratory | Sentimental Analysis Using Naive

colab.research.google.com/drive/1lzdo-UMCVR7zojti61grvqfGtl3CXok-#scrollTo=noZiYN6LAffl

ML Prac 9B.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing

```
[17] df.describe().T
```

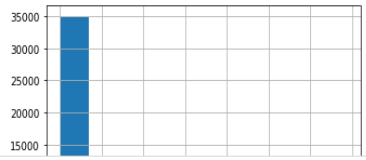
	count	unique	top	freq
review	50000	49582	Loved today's show!!! It was a variety and not...	5
sentiment	50000	2	positive	25000

```
[16] #sentiment count:
     df['sentiment'].value_counts()
```

```
positive    25000
negative    25000
Name: sentiment, dtype: int64
```

```
[18] df['review'].str.len().hist()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f561241b250>
```



0s completed at 10:28 PM

Type here to search 22°C ENG IN 10:33 PM 13-Jan-23

Machine Learning - Google Drive | Prac 9 - Google Docs | ML Prac 9B.ipynb - Colaboratory | Sentimental Analysis Using Naive

colab.research.google.com/drive/1lzdo-UMCVR7zojti61grvqfGtl3CXok-#scrollTo=noZiYN6LAffl

ML Prac 9B.ipynb

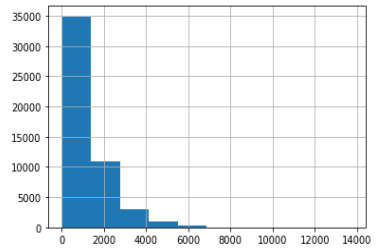
File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

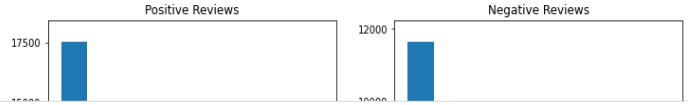
RAM Disk Editing

```
[18] <matplotlib.axes._subplots.AxesSubplot at 0x7f561241b250>
```



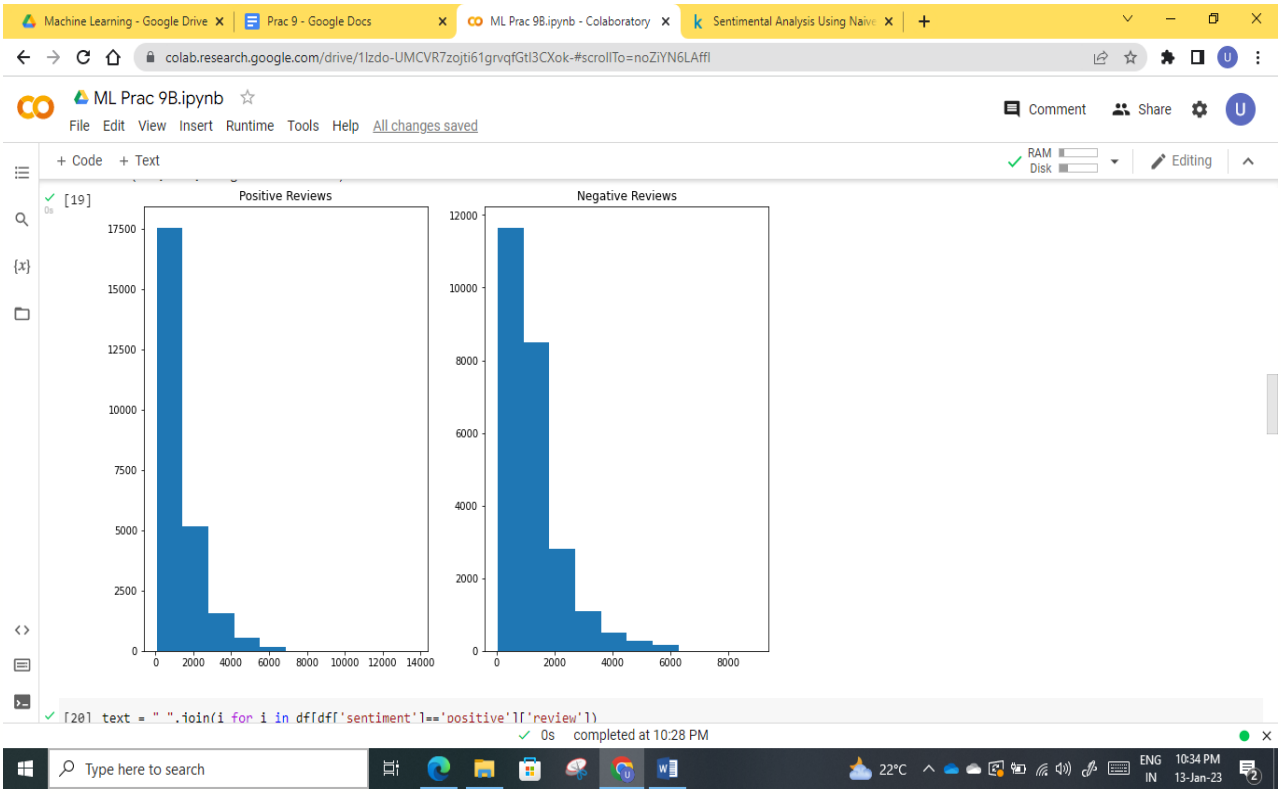

```
[19] fig,(ax1,ax2)=plt.subplots(1,2,figsize=(12,8))
     ax1.hist(df[df['sentiment']=='positive']['review'].str.len())
     ax1.set_title( 'Positive Reviews')
     ax2.hist(df[df['sentiment']=='negative']['review'].str.len())
     ax2.set_title( 'Negative Reviews')
```

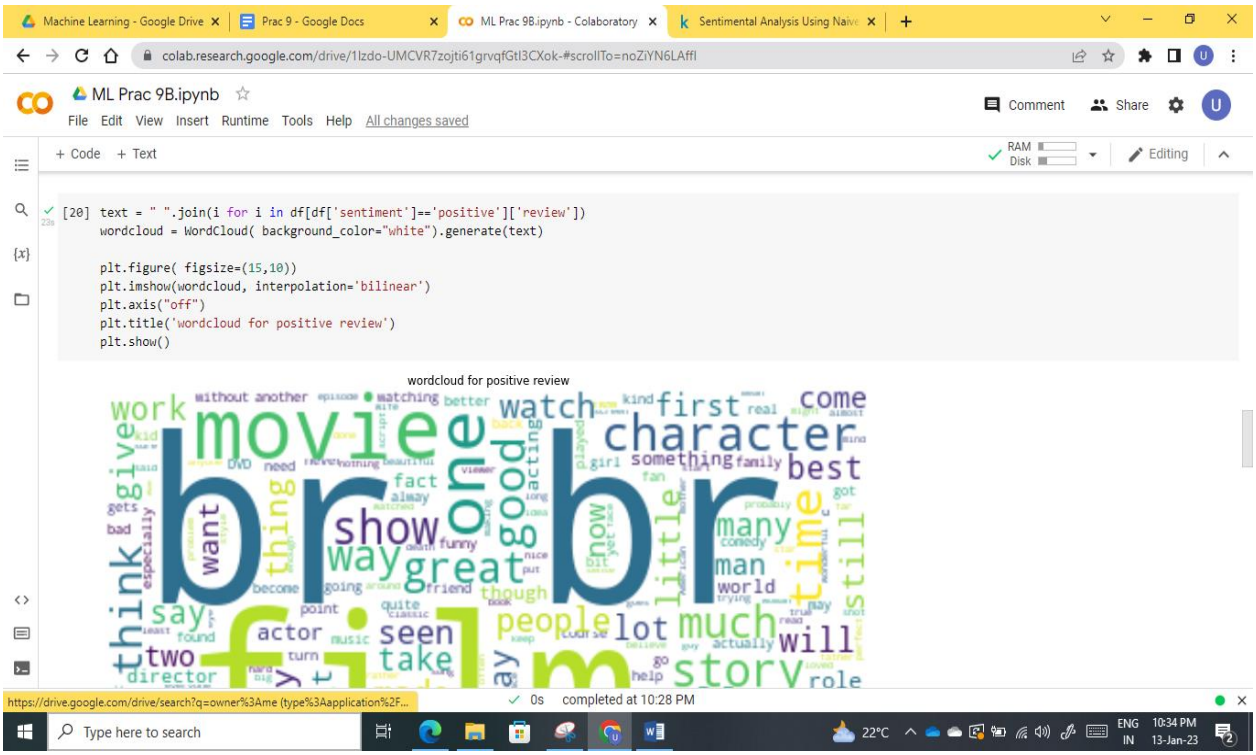
```
Text(0.5, 1.0, 'Negative Reviews')
```

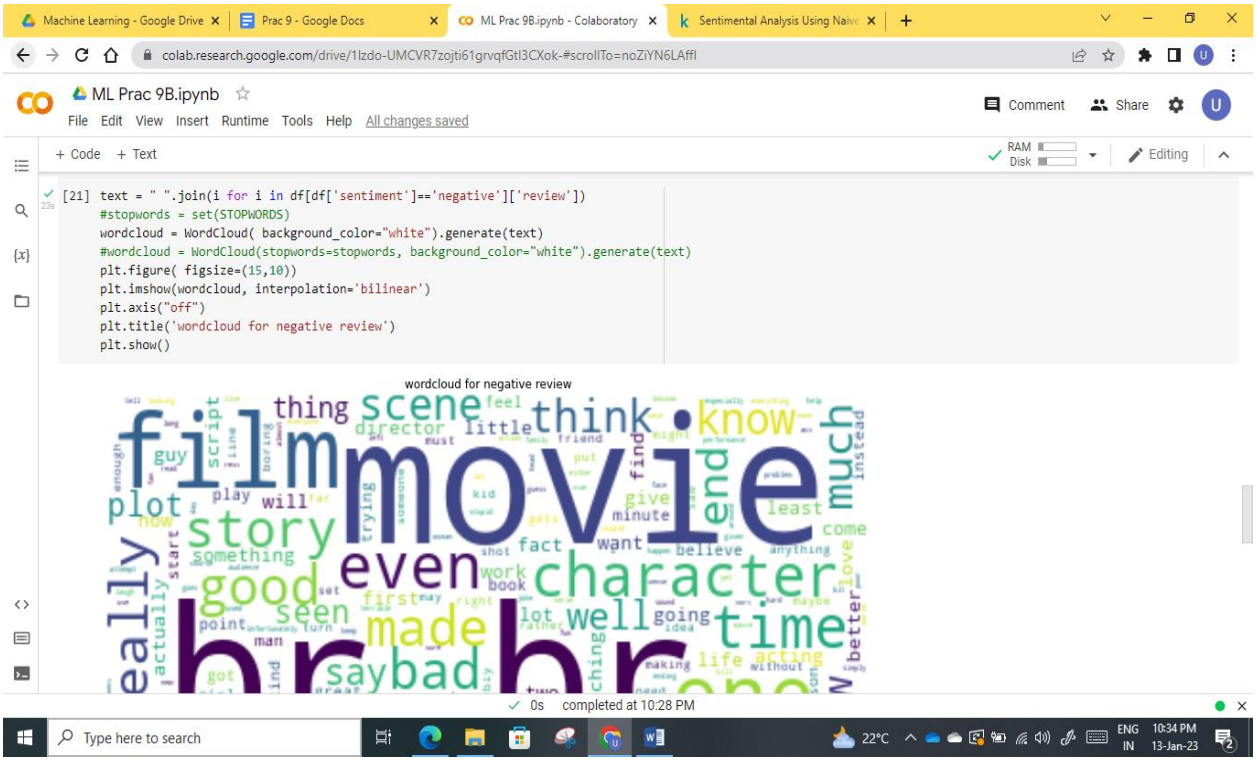


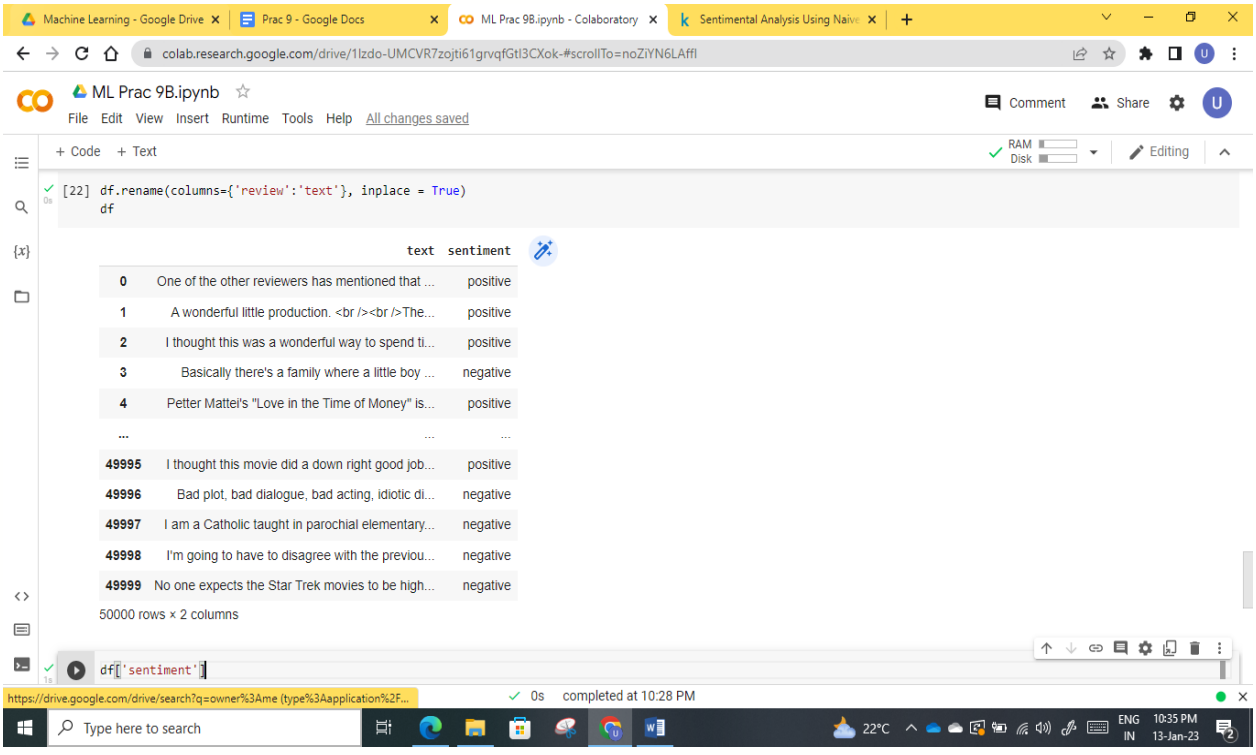
0s completed at 10:28 PM

Type here to search 22°C ENG IN 10:33 PM 13-Jan-23

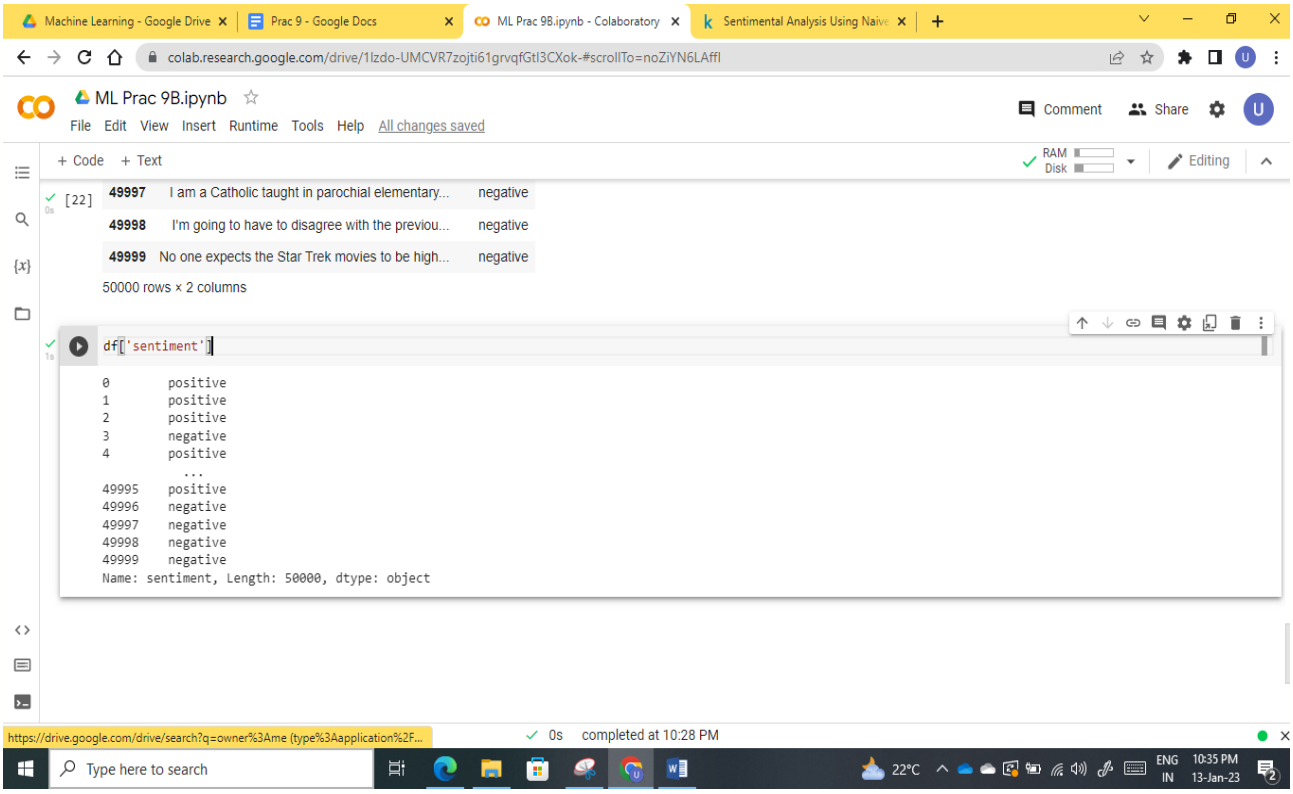

Machine Learning - Google Drive
Prac 9 - Google Docs
ML Prac 9B.ipynb - Colaboratory
Sentimental Analysis Using Naive
colab.research.google.com/drive/1lzdo-UMCVR7zojti61grvqfGtl3CXok-#scrollTo=noZiYN6LAffl
ML Prac 9B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[19]
Positive Reviews
Negative Reviews
[20] text = " ".join(i for i in df[df['sentiment']=='positive']['review'])
0s completed at 10:28 PM
Type here to search
22°C
ENG
IN
10:34 PM
13-Jan-23


Machine Learning - Google Drive
Prac 9 - Google Docs
ML Prac 9B.ipynb - Colaboratory
Sentimental Analysis Using Naive
colab.research.google.com/drive/1lzdo-UMCVR7zojti61grvqfGtl3CXok-#scrollTo=noZiYN6LAffl
ML Prac 9B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[20] text = " ".join(i for i in df[df['sentiment']=='positive']['review'])
wordcloud = WordCloud(background_color="white").generate(text)
plt.figure(figsize=(15,10))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title('wordcloud for positive review')
plt.show()
wordcloud for positive review
0s completed at 10:28 PM
Type here to search
22°C
ENG
IN
10:34 PM
13-Jan-23


Machine Learning - Google Drive
Prac 9 - Google Docs
ML Prac 9B.ipynb - Colaboratory
Sentimental Analysis Using Naive
colab.research.google.com/drive/1lzdo-UMCVR7zojti61grvqfGtl3CXok-#scrollTo=noZiYN6LAffl
ML Prac 9B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[21] text = " ".join(i for i in df[df['sentiment']=='negative']['review'])
#stopwords = set(STOPWORDS)
wordcloud = WordCloud(background_color="white").generate(text)
#wordcloud = WordCloud(stopwords=stopwords, background_color="white").generate(text)
plt.figure(figsize=(15,10))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title('wordcloud for negative review')
plt.show()
wordcloud for negative review
film
movie
br
0s completed at 10:28 PM
Type here to search
22°C
ENG IN
10:34 PM
13-Jan-23


Machine Learning - Google Drive
Prac 9 - Google Docs
ML Prac 9B.ipynb - Colaboratory
Sentimental Analysis Using Naive
colab.research.google.com/drive/1lzdo-UMCVR7zojti61grvqfGtl3CXok-#scrollTo=noZiYN6LAffl
ML Prac 9B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[22] df.rename(columns={'review':'text'}, inplace = True)
df
text sentiment
0 One of the other reviewers has mentioned that ... positive
1 A wonderful little production.

The... positive
2 I thought this was a wonderful way to spend ti... positive
3 Basically there's a family where a little boy ... negative
4 Petter Mattei's "Love in the Time of Money" is... positive
... ...
49995 I thought this movie did a down right good job... positive
49996 Bad plot, bad dialogue, bad acting, idiotic di... negative
49997 I am a Catholic taught in parochial elementary... negative
49998 I'm going to have to disagree with the previou... negative
49999 No one expects the Star Trek movies to be high... negative
50000 rows × 2 columns
df['sentiment']
0s completed at 10:28 PM
Type here to search
22°C
ENG IN
10:35 PM
13-Jan-23


Machine Learning - Google Drive
Prac 9 - Google Docs
ML Prac 9B.ipynb - Colaboratory
Sentimental Analysis Using Naive
colab.research.google.com/drive/1lzdo-UMCVR7zojti61grvqfGtl3CXok-#scrollTo=noZiYN6LAffI
ML Prac 9B.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing
[22]
49997 I am a Catholic taught in parochial elementary... negative
49998 I'm going to have to disagree with the previou... negative
49999 No one expects the Star Trek movies to be high... negative
50000 rows × 2 columns
df['sentiment']
0 positive
1 positive
2 positive
3 negative
4 positive
...
49995 positive
49996 negative
49997 negative
49998 negative
49999 negative
Name: sentiment, Length: 50000, dtype: object
0s completed at 10:28 PM
Type here to search
22°C
ENG
IN
10:35 PM
13-Jan-23

Perform Text pre-processing, Text clustering, classification with Prediction, Test Score and Confusion Matrix

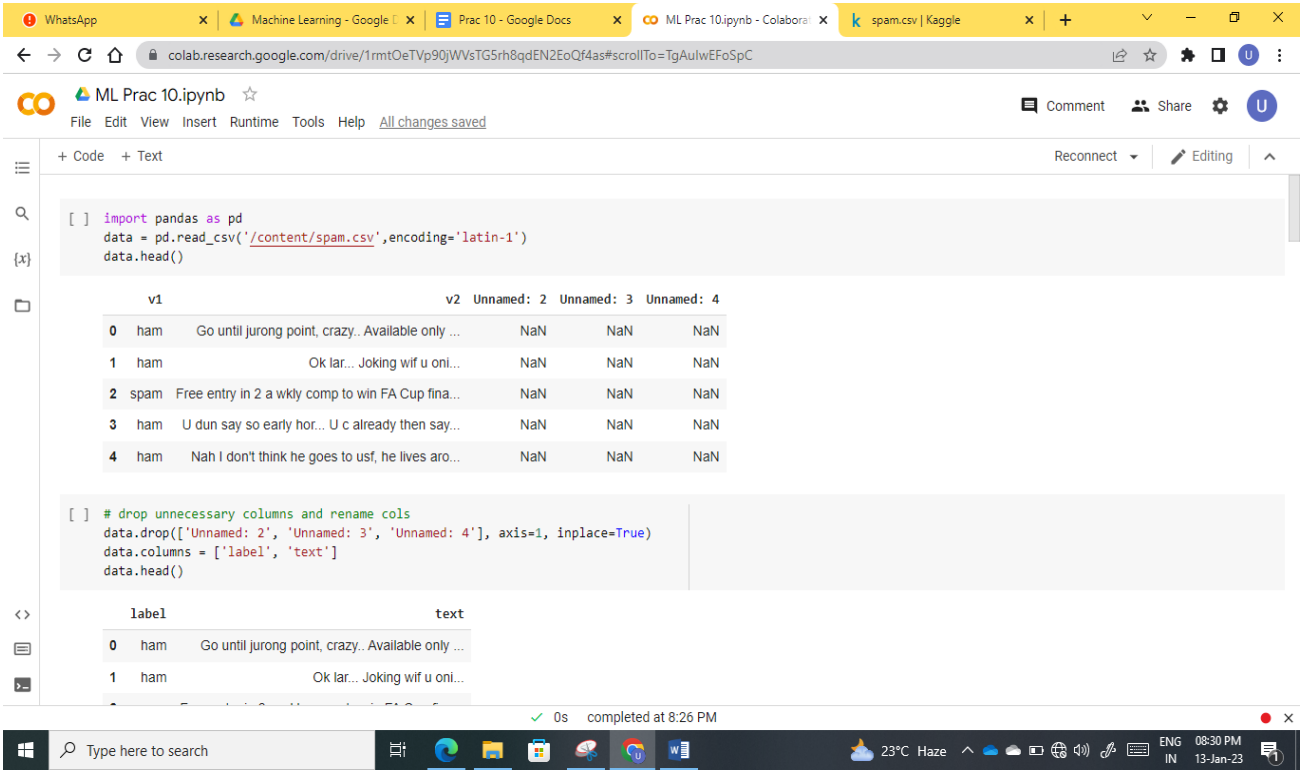
Code :

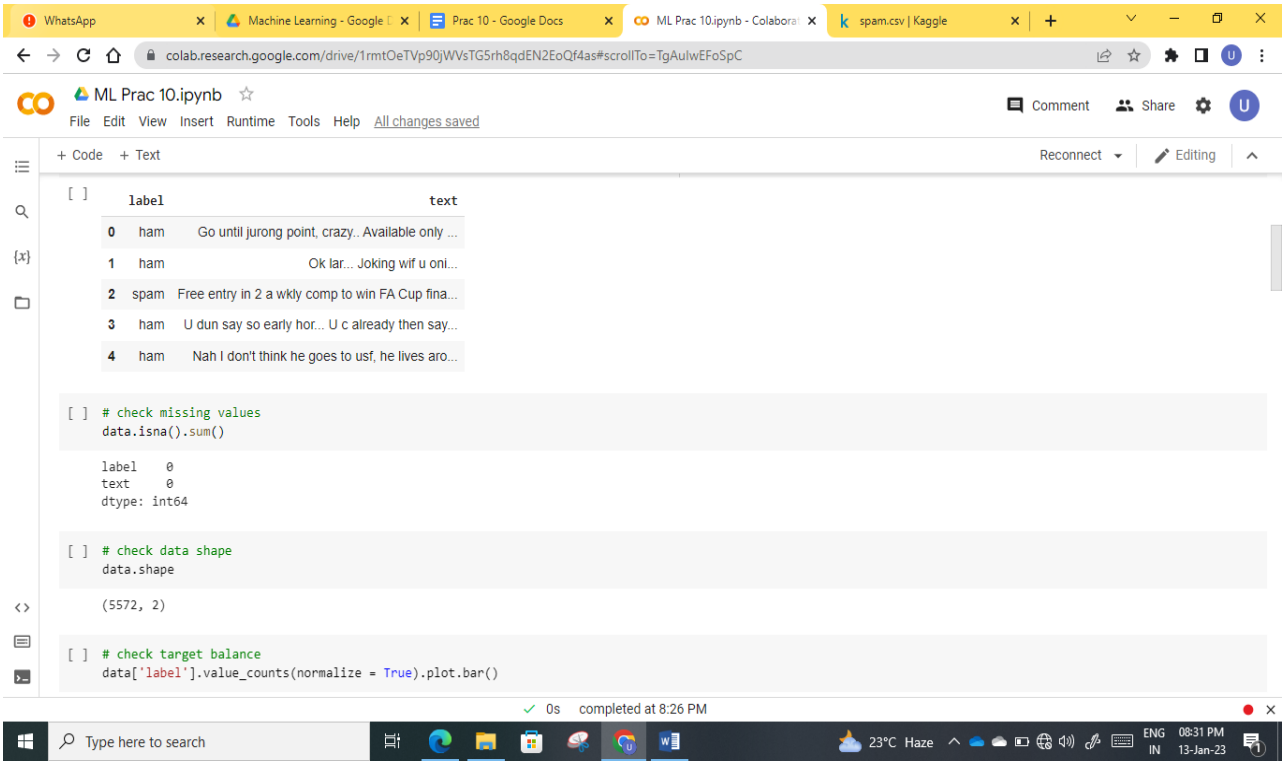
```
import pandas as pd
data = pd.read_csv('/content/spam.csv',encoding='latin-1')
data.head()
# drop unnecessary columns and rename cols
data.drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'], axis=1, inplace=True)
data.columns = ['label', 'text']
data.head()

# check missing values
data.isna().sum()
# check data shape
data.shape
# check target balance
data['label'].value_counts(normalize = True).plot.bar()
# text preprocessing
# download nltk
import nltk
nltk.download('all')

# create a list text
text = list(data['text'])

# preprocessing loop
import re
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
corpus = []
```

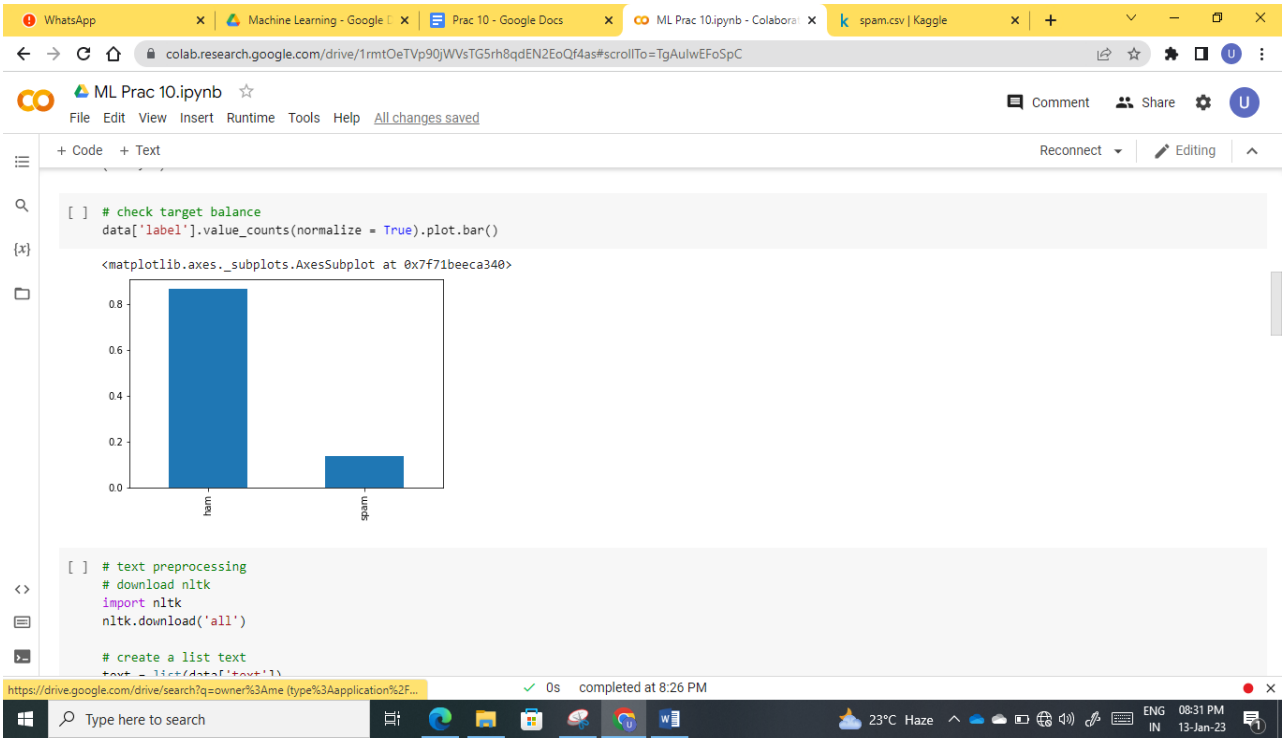
```
for i in range(len(text)):
        r = re.sub('[^a-zA-Z]', ' ', text[i])
        r = r.lower()
        r = r.split()
        r = [word for word in r if word not in stopwords.words('english')]
        r = [lemmatizer.lemmatize(word) for word in r]
        r = ' '.join(r)
corpus.append(r)
# Create Feature and Label sets
X = data['text']
y = data['label']

# train test split (66% train - 33% test)
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=123)
print('Training Data :', X_train.shape)
print('Testing Data : ', X_test.shape)

# Train Bag of Words model
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer()
X_train_cv = cv.fit_transform(X_train)
X_train_cv.shape

#Training Logistic Regression model
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()
lr.fit(X_train_cv, y_train)
# transform X_test using CV
X_test_cv = cv.transform(X_test)

# generate predictions
predictions = lr.predict(X_test_cv)
```

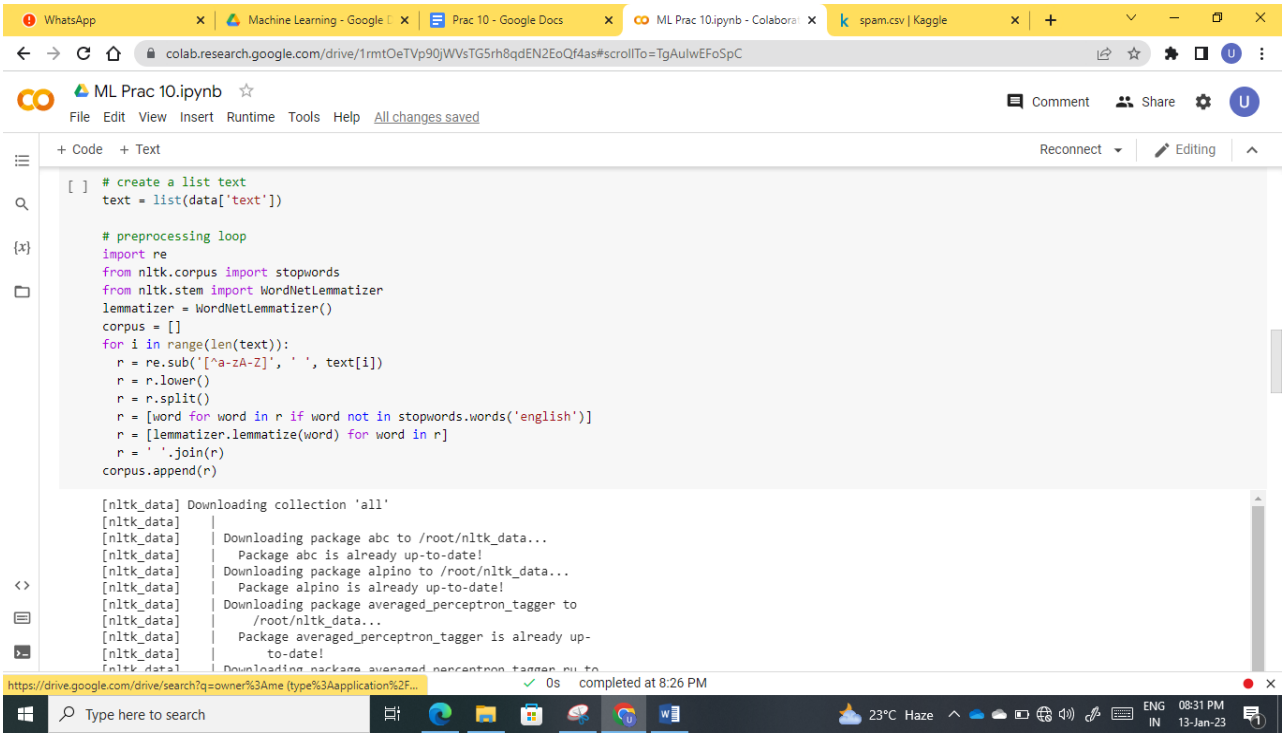
```
predictions
# confusion matrix
import pandas as pd
from sklearn import metrics
df = pd.DataFrame(metrics.confusion_matrix(y_test,predictions), index=['ham','spam'],
columns=['ham','spam'])
df
```

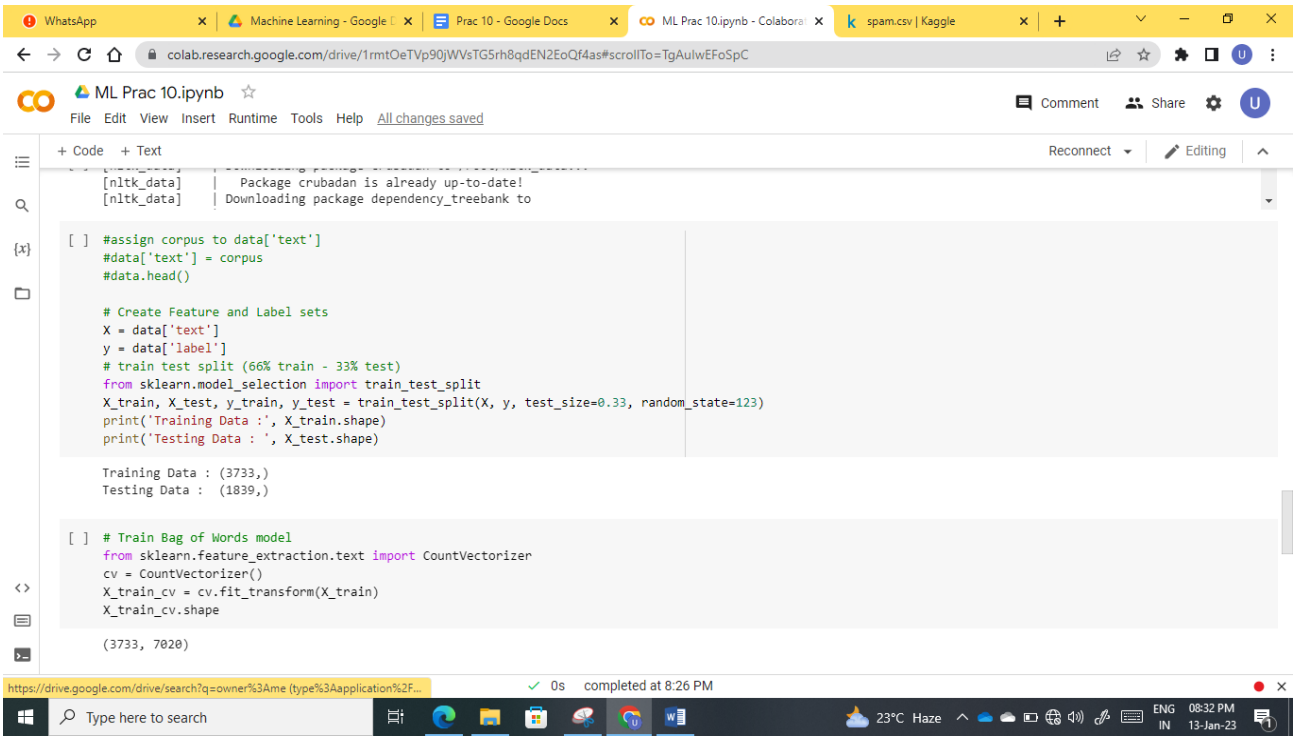
Output :

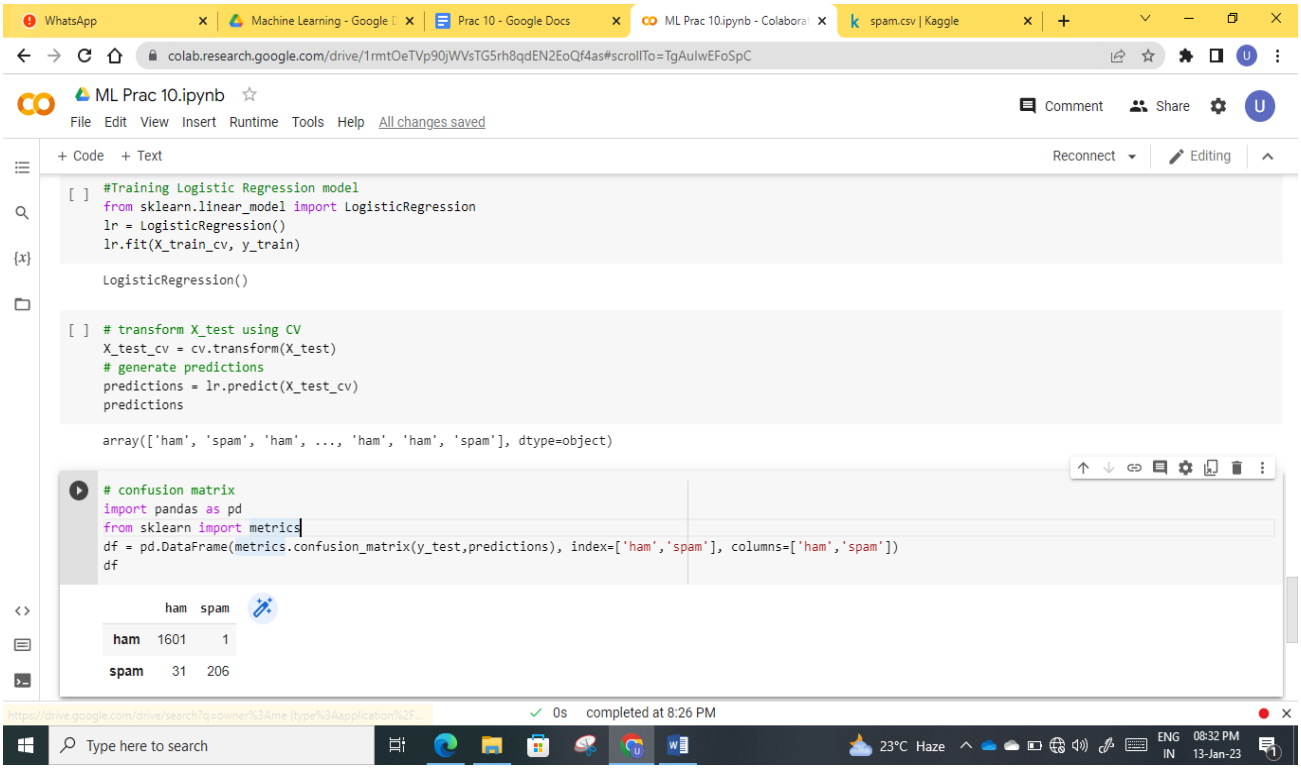



WhatsApp
Machine Learning - Google
Prac 10 - Google Docs
ML Prac 10.ipynb - Colabora
spam.csv | Kaggle
colab.research.google.com/drive/1rmtOeTVp90jWVsTG5rh8qdEN2EoQf4as#scrollTo=TgAulwEFoSpC
ML Prac 10.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Reconnect
Editing
label text
0 ham Go until jurong point, crazy.. Available only ...
1 ham Ok lar... Joking wif u oni...
2 spam Free entry in 2 a wkly comp to win FA Cup fina...
3 ham U dun say so early hor... U c already then say...
4 ham Nah I don't think he goes to usf, he lives aro...
# check missing values
data.isna().sum()
label 0
text 0
dtype: int64
check data shape
data.shape
(5572, 2)
check target balance
data['label'].value_counts(normalize = True).plot.bar()
0s completed at 8:26 PM
Type here to search
23°C Haze
ENG IN
08:31 PM
13-Jan-23


WhatsApp
Machine Learning - Google
Prac 10 - Google Docs
ML Prac 10.ipynb - Colabora
spam.csv | Kaggle
colab.research.google.com/drive/1rmtOeTVp90jWVsTG5rh8qdEN2EoQf4as#scrollTo=TgAulwEFoSpC
ML Prac 10.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Reconnect
Editing
check target balance
data['label'].value_counts(normalize = True).plot.bar()
<matplotlib.axes._subplots.AxesSubplot at 0x7f71beeca340>
0.8
0.6
0.4
0.2
0.0
ham
spam
# text preprocessing
# download nltk
import nltk
nltk.download('all')
# create a list text
0s completed at 8:26 PM
Type here to search
23°C Haze
ENG IN
08:31 PM
13-Jan-23


ML Prac 10.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
Reconnect Editing
[ ] # create a list text
text = list(data['text'])
# preprocessing loop
import re
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
corpus = []
for i in range(len(text)):
r = re.sub('[^a-zA-Z]', ' ', text[i])
r = r.lower()
r = r.split()
r = [word for word in r if word not in stopwords.words('english')]
r = [lemmatizer.lemmatize(word) for word in r]
r = ' '.join(r)
corpus.append(r)
[nltk_data] Downloading collection 'all'
[nltk_data] |
[nltk_data] | Downloading package abc to /root/nltk_data...
[nltk_data] | Package abc is already up-to-date!
[nltk_data] | Downloading package alpino to /root/nltk_data...
[nltk_data] | Package alpino is already up-to-date!
[nltk_data] | Downloading package averaged_perceptron_tagger to
[nltk_data] | /root/nltk_data...
[nltk_data] | Package averaged_perceptron_tagger is already up-
[nltk_data] | to-date!
0s completed at 8:26 PM
Type here to search
23°C Haze
ENG IN 08:31 PM 13-Jan-23


ML Prac 10.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
Reconnect Editing
[nltk_data] | Package crubadan is already up-to-date!
[nltk_data] | Downloading package dependency_treebank to
[ ] #assign corpus to data['text']
#data['text'] = corpus
#data.head()
# Create Feature and Label sets
X = data['text']
y = data['label']
# train test split (66% train - 33% test)
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=123)
print('Training Data :', X_train.shape)
print('Testing Data : ', X_test.shape)
Training Data : (3733,)
Testing Data : (1839,)
[ ] # Train Bag of Words model
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer()
X_train_cv = cv.fit_transform(X_train)
X_train_cv.shape
(3733, 7020)
0s completed at 8:26 PM
Type here to search
23°C Haze
ENG IN 08:32 PM 13-Jan-23


WhatsApp
Machine Learning - Google
Prac 10 - Google Docs
ML Prac 10.ipynb - Colabora
spam.csv | Kaggle
colab.research.google.com/drive/1rmtOeTVp90jWVsTG5rh8qdEN2EoQf4as#scrollTo=TgAuIwEFoSpC
ML Prac 10.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Reconnect
Editing
[ ] #Training Logistic Regression model
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()
lr.fit(X_train_cv, y_train)
LogisticRegression()
[ ] # transform X_test using CV
X_test_cv = cv.transform(X_test)
# generate predictions
predictions = lr.predict(X_test_cv)
predictions
array(['ham', 'spam', 'ham', ..., 'ham', 'ham', 'spam'], dtype=object)
# confusion matrix
import pandas as pd
from sklearn import metrics
df = pd.DataFrame(metrics.confusion_matrix(y_test,predictions), index=['ham','spam'], columns=['ham','spam'])
df
ham spam
ham 1601 1
spam 31 206
0s completed at 8:26 PM
Type here to search
23°C Haze
ENG IN
08:32 PM
13-Jan-23