**Lab File**

**Fundamentals of Machine Learning**

**(CSE 313)**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

A blue and yellow logo

Description automatically generated with low confidence

Submitted to: Submitted by:

Dr Shipra Shukla Shaina Mehta

Ast. Professor A2305219268

CSE Department, ASET B.tech. C.S.E.

6CSE-4Y

Amity School Of Engineering and technology

Amity University Uttar Pradesh

Noida -201301

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Exp No | Assignment  Category | Code | Name of Experiment | Date of Allotment | Date of Evaluation | Max  Marks | Marks  Obtained | Faculty  Sign |
| 1 | Mandatory  Experiment |  | Basic Python Programs | 11-01-2022 | 11-01-2022 |  |  |  |
| 2 |  | Intermediate Python Programs | 18-01-2022 | 18-01-2022 |  |  |  |
| 3 |  | To draw different kinds of plots | 25-01-2022 | 25-01-2022 |  |  |  |
| 4 |  | To implement linear regression model on housing dataset. | 08-02-2022 | 08-02-2022 |  |  |  |
| 5 |  | To implement logistic regression model on diabetes dataset and email dataset. | 15-02-2022 | 15-02-2022 |  |  |  |
| 6 |  | To implement naïve bayes classifier on advertisement and email dataset. | 08-03-2022 | 08-03-2022 |  |  |  |
| 7 |  | Use some function for neural networks, like Stochastic Gradient Descent or backpropagation - algorithm to predict the value of a variable based on the dataset. | 15-03-2022 | 15-03-2022 |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  | Viva | Viva |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Exp No | Assignment  Category | Code | Name of Experiment | Date of Allotment | Date of Evaluation | Max  Marks | Marks  Obtained | Faculty  Sign |
| 8 | Mandatory  Experiment |  | To implement and evaluate a classification using SVM. | 29-03-2022 | 29-03-2022 |  |  |  |
| 9 |  | To implement and evaluate a classification using K-means Algorithm. | 05-04-2022 | 05-04-2022 |  |  |  |
| 10 |  | Implement and evaluate a classification using Decision Tree. | 05-04-2022 | 05-04-2022 |  |  |  |
| 11 |  | Implement and evaluate a classification using Random Forest. | 05-04-2022 | 05-04-2022 |  |  |  |
| 12 |  | To implement Regularization on the dataset. | 05-04-2022 | 05-04-2022 |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  | Viva | Viva |  |  |  |  |  |  |

**Experiment 1**

**Aim:** Basic Python Programs

**Software Used:** Jupyter Notebook

**Code and Output:**

1. **Arithmetic Operators and Lists**

***Code:***

import numpy as np

# type(X) function

x = 5

print('datatype of x is:',type(x))

print('value of x is:',x)

x = "Shaina"

print('datatype of x is:',type(x))

print('value of x is:',x)

x = 'c'

print('datatype of x is:',type(x))

print('value of x is:',x)

x = True

print('datatype of x is:',type(x))

print('value of x is:',x)

x = 77.034

print('datatype of x is:',type(x))

print('value of x is:',x)

x = b'Shaina'

print('datatype of x is:',type(x))

print('value of x is:',x)

x = [23,'Mona',77.5]

print('datatype of x is:',type(x))

print('value of x is:',x)

x = (23,'Mona',77.5)

print('datatype of x is:',type(x))

print('value of x is:',x)

x = {23,'Mona',77.5}

print('datatype of x is:',type(x))

print('value of x is:',x)

x = {"a":12,"b":34}

print('datatype of x is:',type(x))

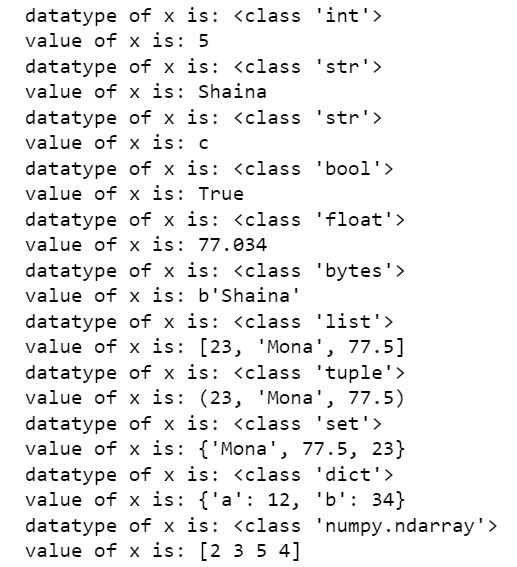
print('value of x is:',x)

x = np.array([2,3,5,4])

print('datatype of x is:',type(x))

print('value of x is:',x)

***Output:***



***Code:***

# Floor Divison

x = 123

y = 34

print(x//y)

***Output:***



***Code:***

# Using OR Operator

a = 30

b = 2

print((a%2)==0 or (b%2)==0)

***Output:***



***Code:***

# Slicing

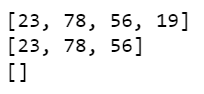
L = [23,45,23,78,56,19]

print(L[2:6])

print(L[-4:-1])

print(L[-2:0])

***Output:***



***Code:***

# Write a Python program to sum all the items in a list

import functools as f

l = [23,45,12,56,78]

add = f.reduce(lambda a,b: a+b,l)

print(add)

***Output:***



***Code:***

# Python program to find largest number in a list

maximum = max(l)

print(maximum)

***Output:***



***Code:***

# Python program to find smallest number in a list

minimum = min(l)

print(minimum)

***Output:***



***Code:***

# Write a Python program to multiplies all the items in a list

mul = f.reduce(lambda a,b: a\*b,l)

print(mul)

***Output:***



1. **Loops**

***Code:***

# while loop

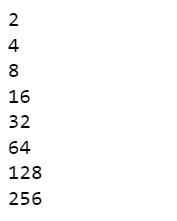
number = 1

while(number<200):

number = number\*2

print(number)

***Output:***



***Code:***

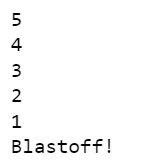
# for loop

for x in range(5,0,-1):

print(x)

print('Blastoff!')

***Output:***



1. **Dictionaries**

***Code:***

# Create a dictionary

di = {

"brand": "Ford",

"model": "Mustang",

"year": 1964

}

print(di)

***Output:***



***Code:***

# Accessing elements

# get the value of the model key

x = di.get('brand')

print(x)

# another Method

x = di['brand']

print(x)

***Ouput:***



***Code:***

# Change Values of a specific item by referring to its key name

di['year'] = 2018

print(di)

***Output:***



***Code:***

# Dictionart length using len() function

print(len(di))

***Output:***



***Code:***

#remove an item from a dictionary if it is present.

# check if item is present

if('year' in di):

print('Yes')

# removing item using pop()

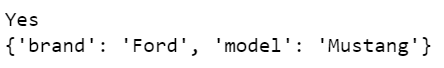
di.pop('year')

print(di)

else:

print('No. Key and item cannot be deleted')

***Output:***



***Code:***

# Add a key to a dictionary.

di['year'] = 2018

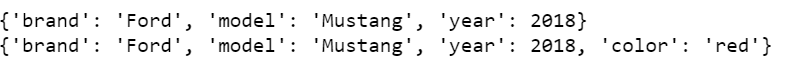
print(di)

# using update method

di.update({'color':'red'})

print(di)

***Output:***



***Code:***

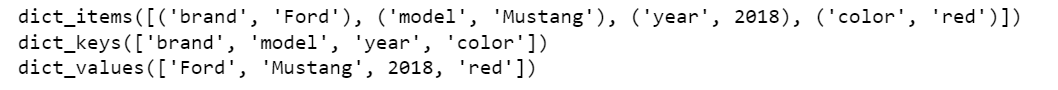
# getting the values and the keys of the dictionary

print(di.items())

print(di.keys())

print(di.values())

***Output:***



***Code:***

# clearing the items of the dictionary

di.clear()

print(di)

***Output:***



***Code:***

# del method to delete

del di

#print(di)

# This will give an error which is as follows:

#---------------------------------------------------------------------------

#NameError Traceback (most recent call last)

#C:\Users\SHAINA~1\AppData\Local\Temp/ipykernel\_16192/3110200438.py in <module>

# 1 # del method to delete

# 2 del di

#----> 3 print(di)

# NameError: name 'di' is not defined

***Code:***

# merging two dictionaries

di1 = {'a':10,'b':20}

di2 = {'c':30,'d':40}

print('dictionaries before updating')

print(di1)

print(di2)

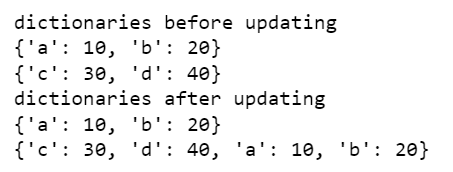
di2.update(di1)

print('dictionaries after updating')

print(di1)

print(di2)

***Output:***



**Conclusion:** Basic python programs has been done successfully.

|  |  |  |  |
| --- | --- | --- | --- |
| Internal Assessment (Mandatory Experiment) Sheet for Lab Experiment  Department of Computer Science & Engineering  ASET, Amity University Noida (U.P) | | | |
| Program | B. Tech CSE | Course Name | Fundamentals of Machine Learning |
| Course Code | CSE 313 | Semester | 6 |
| Student Name | Shaina Mehta | Enrolment no | A2305219268 |
| **Marking Criteria** | | | |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept(A) | 2 |  |  |
| Implementation(B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 |  |  |

**Experiment - 2**

**Aim:** Intermediate Python Programs.

**Software Used:** Jupyter Notebook

**Code and Output:**

1. **Loops**

***Code:***

# while loop

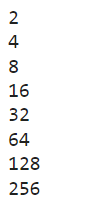
number = 1

while(number<200):

number=number\*2

print(number)

***Output:***



***Code:***

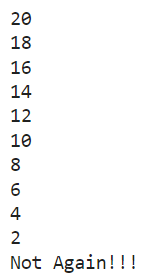
# for loop

for x in range(20,0,-2):

print(x)

print("Not Again!!!")

***Output:***



1. **Functions and Lambda Expressions**

***Code:***

# lambda functions

x = lambda a,b: a\*b

print(x(2,3))

***Output:***

******

***Code:***

# user defined function

def fun(x,y):

a=x+y

b=x-y

return a\*\*2,b\*\*2

print(fun(3,4))

***Output:***

******

1. **Sets**

***Code:***

# Create a set

thisset = {"apple", "banana","cherry",}

print(thisset)

***Output:***

******

***Code:***

# Add member(s) in the set

# add single element

thisset.add("orange")

print(thisset)

# add multiple element

thisset.update(["peach","mango"])

print(thisset)

***Output:***

******

***Code:***

# delete item(s) from set

# using discard

thisset.discard("orange")

print(thisset)

# using remove

thisset.remove("mango")

print(thisset)

***Output:***

******

***Code:***

# delete an item from a set if it is present in the set.

# to check if an item is present in the set

if("banana" in thisset):

thisset.discard("banana")

print(thisset)

else:

print("It cannot be discarded.")

***Output:***

******

***Code:***

# Remove the last item by using the pop()

X=thisset.pop()

print(X)

print(thisset)

***Output:***

******

***Code:***

# clear the set

thisset.clear()

print(thisset)

***Output:***

******

***Code:***

# delete a set object using del keyword

del thisset

# print(thisset)

# This will give the error if you try to print the set object

# ---------------------------------------------------------------------------

# NameError Traceback (most recent call last)

# C:\Users\SHAINA~1\AppData\Local\Temp/ipykernel\_12880/1209023853.py in <module>

# 1 #delete a set

# 2 del thisset

# ----> 3 print(thisset)

# NameError: name 'thisset' is not defined

1. **Matrix Operations**
2. **Perform basic operations on matrices (like addition, subtraction, multiplication) and display specific rows or columns of the matrix.**

***Code:***

# matrix addition

import numpy as np

import random as rn

mat1 = np.array([rn.randint(1,101) for i in range(9)]).reshape(3,3)

mat2 = np.array([rn.randint(1,101) for i in range(9)]).reshape(3,3)

print('matrix 1')

print(mat1)

print('matrix 2')

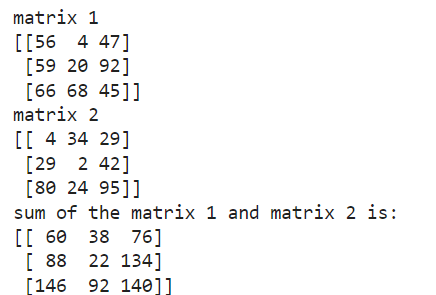
print(mat2)

mat3 = np.add(mat1,mat2)

print("sum of the matrix 1 and matrix 2 is: ")

print(mat3)

***Output:***

******

***Code:***

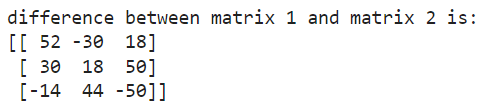
# subtraction of two matrices

mat4 = np.subtract(mat1,mat2)

print("difference between matrix 1 and matrix 2 is: ")

print(mat4)

***Output:***

******

***Code:***

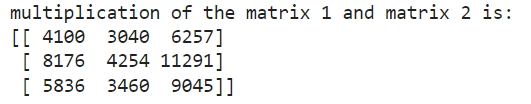
# multiplicaion of two matrices

mat5 = np.matmul(mat1,mat2)

print("multiplication of the matrix 1 and matrix 2 is: ")

print(mat5)

***Output:***

******

***Code:***

# transpose of a matrix

mat6 = np.array([rn.randint(1,101) for i in range(9)]).reshape(3,3)

print('matrix 6')

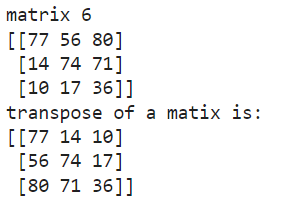
print(mat6)

mat7 = np.transpose(mat6)

print('transpose of a matix is: ')

print(mat7)

***Output:***

******

***Code:***

# displaying the specific rows and columns of a matrix

print('matrix')

print(mat6)

# 3rd row

print('3rd row')

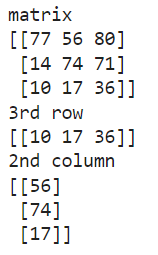
print(mat6[2:,])

# 2nd column

print('2nd column')

print(mat6[:,[1]])

***Output:***

******

1. **Perform other matrix operations like converting matrix data to absolute values, taking the negative of matrix values, additing/removing rows/columns from a matrix, finding the maximum or minimum values in a matrix or in a row/column, and finding the sum of some/all elements in a matrix.**

***Code:***

# converting the matrix into absolute values

arr = np.array([rn.randint(-10,11) for i in range(9)]).reshape(3,3)

print('matrix')

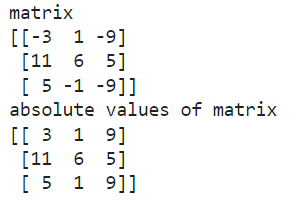
print(arr)

arr1 = np.abs(arr)

print('absolute values of matrix')

print(arr1)

***Output:***



***Code:***

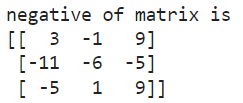
# taking negative of matrix values

arr2 = np.negative(arr)

print('negative of matrix is')

print(arr2)

***Output:***

******

***Code:***

# maxmimum element

# in a matrix

print("maxmimum element")

print('in a matrix')

print(np.max(arr))

# row-wisw

print('row-wise')

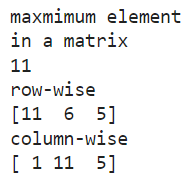
print(np.max(arr,axis=0))

# column-wise

print('column-wise')

print(np.max(arr,axis=1))

***Output:***

******

***Code:***

# minimum element

# in a matrix

print("minimum element")

print('in a matrix')

print(np.min(arr))

# row-wisw

print('row-wise')

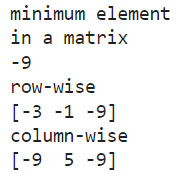
print(np.min(arr,axis=0))

# column-wise

print('column-wise')

print(np.min(arr,axis=1))

***Output:***

******

***Code:***

# element sum of a matrix

# all the elements

print('element sum of all the elements of a matrix')

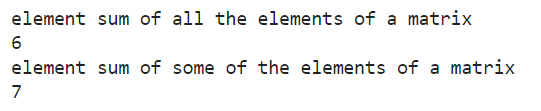
print(np.sum(arr))

# some of the elements

print('element sum of some of the elements of a matrix')

print(np.sum(arr[0:2,1:2]))

***Output:***

******

***Code:***

# deleting rows and columns of a matrix

# deleting 3rd row

arr3 = np.delete(arr,2,0)

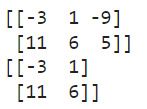
print(arr3)

# deleting 3rd column

arr4 = np.delete(arr3,2,1)

print(arr4)

***Output:***

******

***Code:***

# adding rows and columns of a matrix

# adding row

arr5 = np.vstack((arr4,np.array([2,3])))

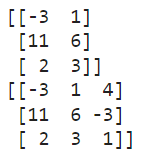
print(arr5)

# adding column

arr6 = np.append(arr5,np.array([4,-3,1]).reshape(3,1),axis=1)

print(arr6)

***Output:***

******

1. **Graphs**

***Code:***

# Basic Plot

import matplotlib.pyplot as plt

x = [rn.randint(-20,20) for i in range(10)]

y = [rn.randint(-20,20) for i in range(10)]

plt.plot(x,y)

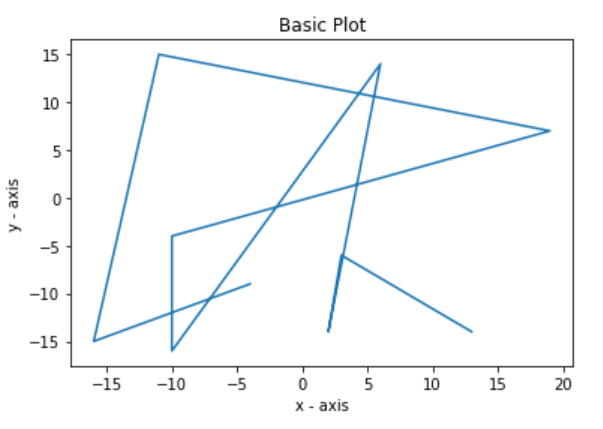
plt.xlabel('x - axis')

plt.ylabel('y - axis')

plt.title('Basic Plot')

plt.show()

***Output:***



***Code:***

# Bar Graph

city = ['Delhi','Mumbai','Chennai','Kolkata']

AQI = [345,123,83,230]

x\_pos = [i for i, \_ in enumerate(city)]

plt.bar(x\_pos,AQI,color='violet')

plt.xticks(x\_pos, city)

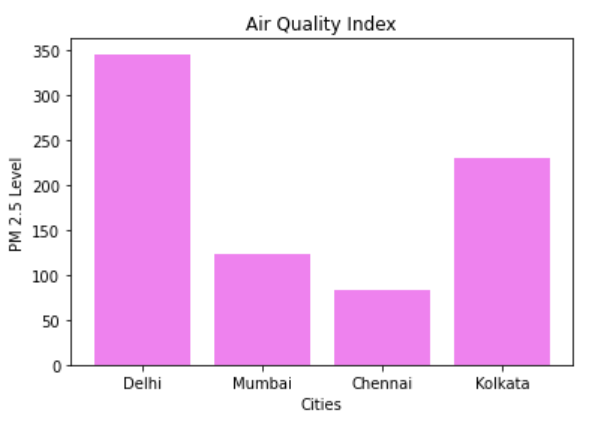
plt.xlabel('Cities')

plt.ylabel('PM 2.5 Level')

plt.title('Air Quality Index')

plt.show()

***Output:***

******

**Conclusion:** Intermediate python programs have been done successfully.

|  |  |  |  |
| --- | --- | --- | --- |
| Internal Assessment (Mandatory Experiment) Sheet for Lab Experiment  Department of Computer Science & Engineering  ASET, Amity University Noida (U.P) | | | |
| Program | B. Tech CSE | Course Name | Fundamentals of Machine Learning |
| Course Code | CSE 313 | Semester | 6 |
| Student Name | Shaina Mehta | Enrolment no | A2305219268 |
| **Marking Criteria** | | | |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept(A) | 2 |  |  |
| Implementation(B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 |  |  |

**Experiment 3**

**Aim:** To draw different kinds of plots

**Software Used:** Jupyter Notebook

**Code and Output:**

1. **Create various type of plots/charts like histograms, plot based on sine/cosine function based on data from a matrix. Further label different axes in a plot and data in a plot.**

***Code:***

import numpy as np

import matplotlib.pyplot as plt

x = np.arange(-np.pi,np.pi,np.pi/4)

y = np.sin(x)

plt.plot(x,y)

plt.title('Sine Function')

plt.xlabel('x')

plt.ylabel('Sin x')

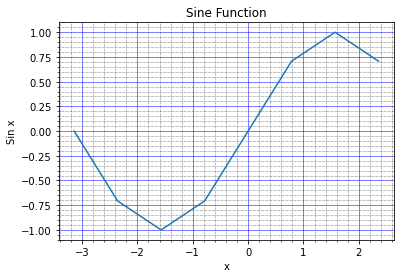
plt.minorticks\_on()

plt.grid(which='major',linestyle='-',linewidth='0.5',color='blue')

plt.grid(which='minor',linestyle=':',linewidth='0.5',color='black')

plt.show()

***Output:***



***Code:***

x = np.arange(-np.pi,np.pi,np.pi/4)

y = np.cos(x)

plt.plot(x,y)

plt.title('Cosine Function')

plt.xlabel('x')

plt.ylabel('Cos x')

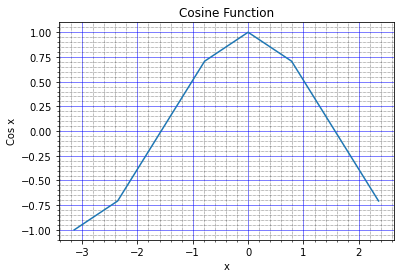
plt.minorticks\_on()

plt.grid(which='major',linestyle='-',linewidth='0.5',color='blue')

plt.grid(which='minor',linestyle=':',linewidth='0.5',color='black')

plt.show()

***Output:***



***Code:***

# x-coordinates of left sides of bars

left = [1, 2, 3, 4, 5]

# heights of bars

height = [10, 24, 36, 40, 5]

# labels for bars

tick\_label = ['one', 'two', 'three', 'four', 'five']

# plotting a bar chart

plt.bar(left,height,color=['red','green'])

# naming the x-axis

plt.xlabel('x - axis')

# naming the y-axis

plt.ylabel('y - axis')

# plot title

plt.title('My bar chart!')

plt.xticks(left,tick\_label)

plt.minorticks\_on()

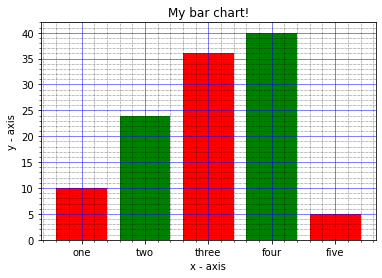
plt.grid(which='major',linestyle='-',linewidth='0.5',color='blue')

plt.grid(which='minor',linestyle=':',linewidth='0.5',color='black')

# function to show the plot

plt.show()

***Output:***



***Code:***

# frequencies

ages = [2,5,70,40,30,45,50,45,43,40,44,60,7,13,57,18,90,77,32,21,20,40]

# setting the ranges and no. of intervals

range = (0, 100)

bins = 10

# plotting a histogram

plt.hist(ages,bins,range,color='pink',histtype='bar',rwidth=0.8)

# x-axis label

plt.xlabel('age')

# frequency label

plt.ylabel('No. of people')

# plot title

plt.title('My histogram')

plt.minorticks\_on()

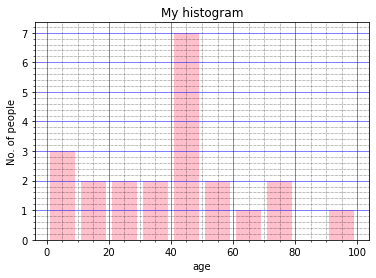
plt.grid(which='major',linestyle='-',linewidth='0.5',color='blue')

plt.grid(which='minor',linestyle=':',linewidth='0.5',color='black')

# function to show the plot

plt.show()

***Output:***



***Code:***

pl = ['Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++']

pop = [22.2, 17.6, 8.8, 8, 7.7, 6.7]

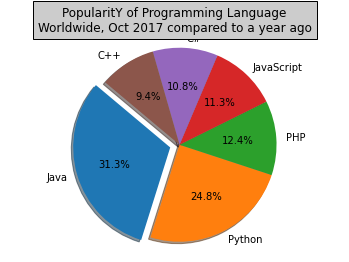
plt.pie(pop,explode=(0.1,0,0,0,0,0),labels=pl,startangle=140,shadow=True,autopct='%1.1f%%')

plt.axis('equal')

plt.title("PopularitY of Programming Language\n" + "Worldwide, Oct 2017 compared to a year ago",bbox={'facecolor':'0.8','pad':5})

plt.show()

***Output:***



1. **Visualization Using Iris Flower Dataset**

***Code:***

import pandas as pd

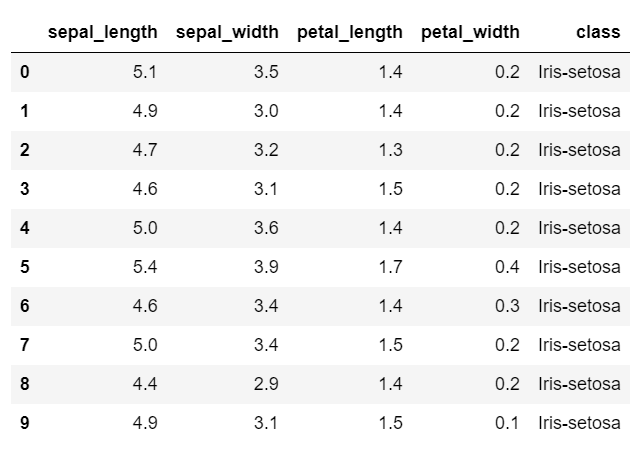
import seaborn as sns

%matplotlib inline

iris = pd.read\_csv('Iris.csv')

iris.head(10)

***Output:***

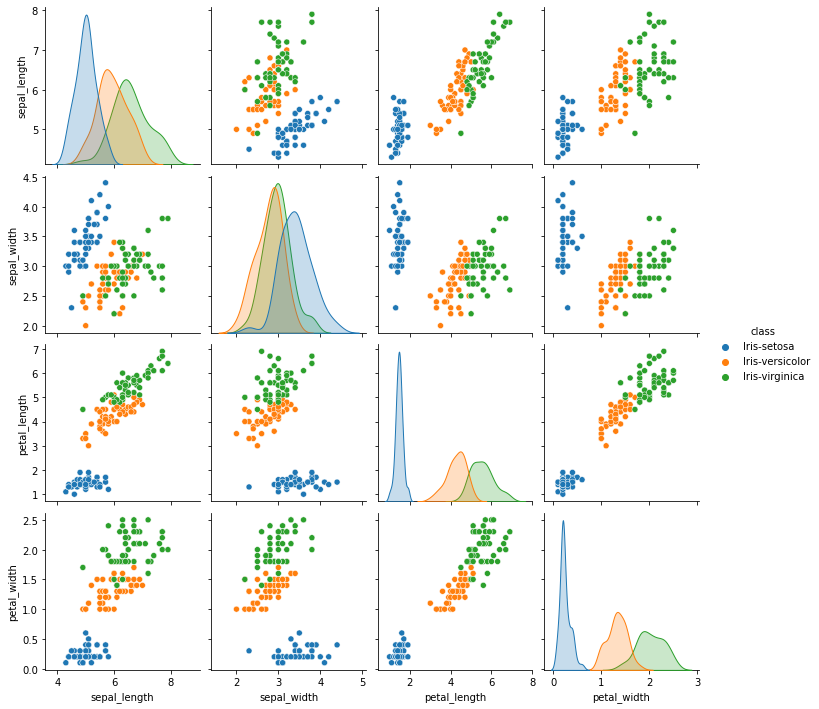


***Code:***

sns.pairplot(iris,hue='class')

plt.show()

***Output:***



**Conclusion:** Different kinds of plots have been plotted successfully.

|  |  |  |  |
| --- | --- | --- | --- |
| Internal Assessment (Mandatory Experiment) Sheet for Lab Experiment  Department of Computer Science & Engineering  ASET, Amity University Noida (U.P) | | | |
| Program | B. Tech CSE | Course Name | Fundamentals of Machine Learning |
| Course Code | CSE 313 | Semester | 6 |
| Student Name | Shaina Mehta | Enrolment no | A2305219268 |
| **Marking Criteria** | | | |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept(A) | 2 |  |  |
| Implementation(B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 |  |  |

**Experiment 4**

**Aim:** To implement linear regression model on housing dataset.

**Software Used:** Jupyter Notebook

**Code and Output:**

1. **Implement Linear Regression problem. For example, based on a dataset comprising of existing set of prices and area/size of the houses, predict the estimated price of a given house.**

***Code:***

import numpy as np

import pandas as pd

import seaborn as sns

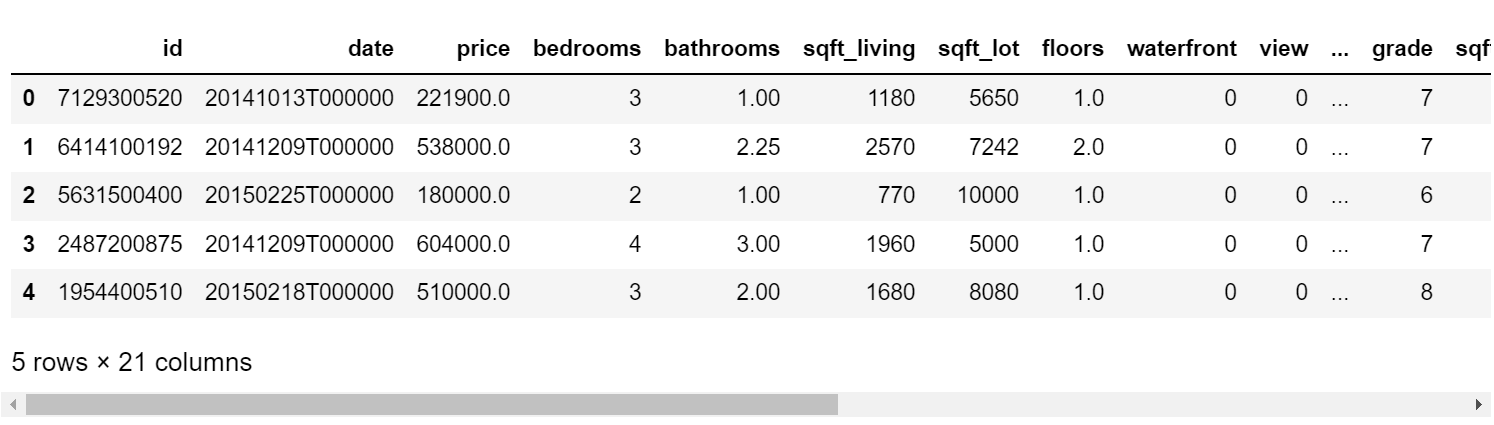
import matplotlib.pyplot as plt

%matplotlib inline

house= pd.read\_csv('kc\_house\_data.csv')

house.head()

***Output:***

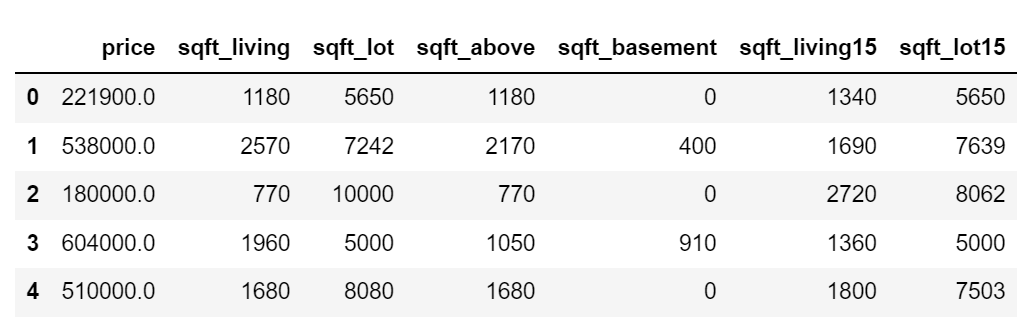


***Code:***

house=house.drop(['id','date','zipcode','bedrooms','bathrooms','floors','waterfront','view','condition','grade','yr\_built','yr\_renovated','lat','long'],axis=1)

house.head()

***Output:***

******

***Code:***

house.isnull().sum()

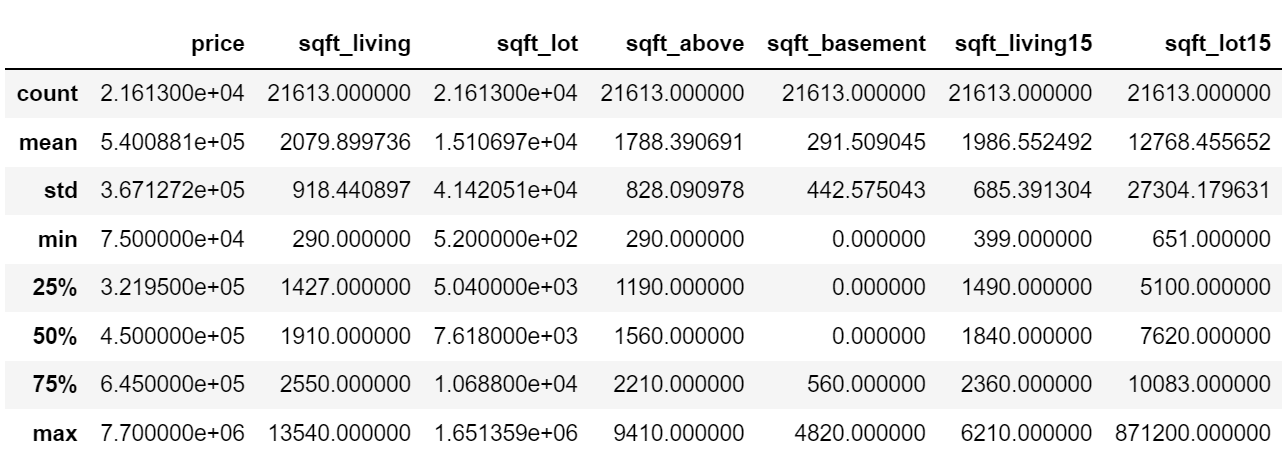
***Output:***

******

***Code:***

house.describe()

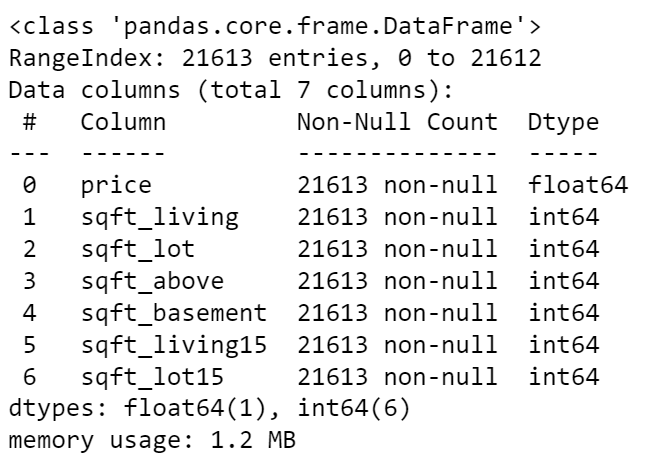
***Output:***

******

***Code:***

house.info()

***Output:***

******

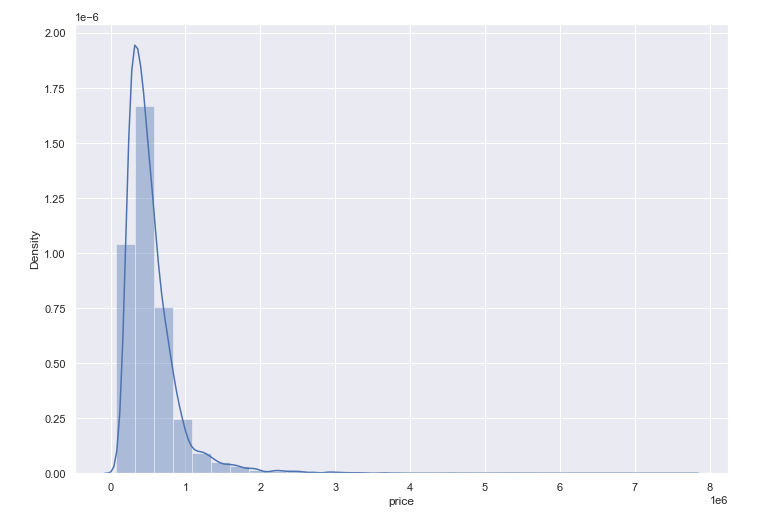
***Code:***

sns.set(rc={'figure.figsize':(11.7,8.27)})

sns.distplot(house['price'], bins=30)

plt.show()

***Output:***

******

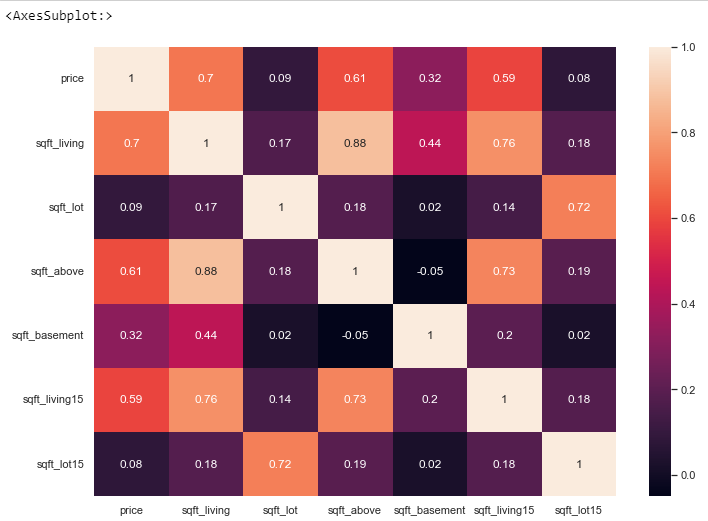
***Code:***

correlation\_matrix = house.corr().round(2)

# annot = True to print the values inside the square

sns.heatmap(data=correlation\_matrix, annot=True)

***Output:***

******

***Code:***

prices = house['price']

features = house.drop(['price'],axis=1)

plt.figure(figsize=(20, 20))

# i: index

for i, col in enumerate(features.columns):

plt.subplot(3, 3, i+1)

x = features[col]

y = prices

plt.plot(x, y, 'o')

# Create regression line

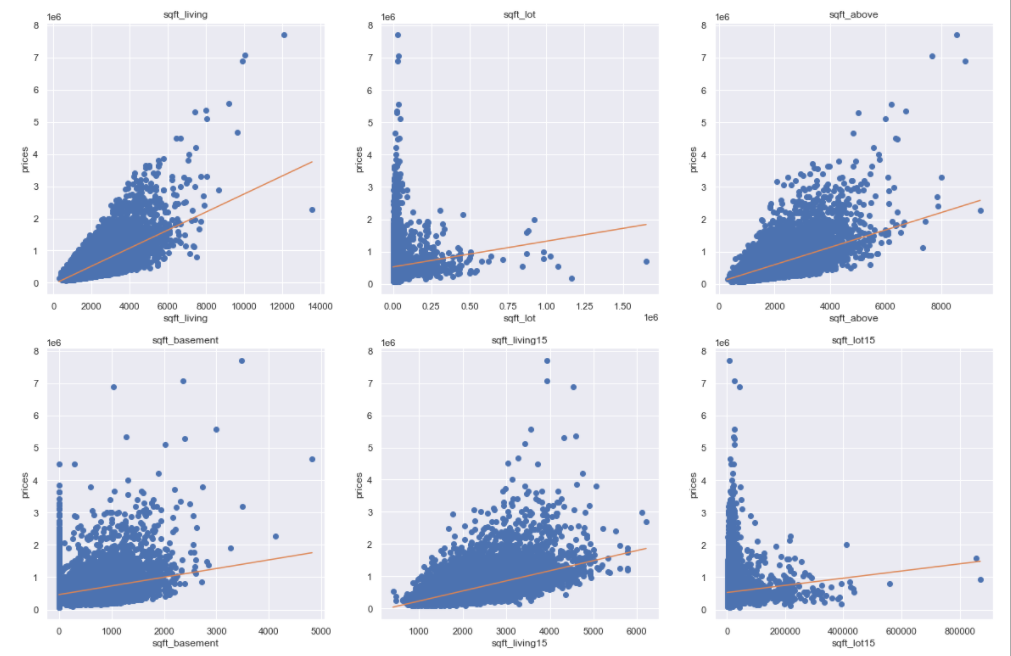
plt.plot(np.unique(x), np.poly1d(np.polyfit(x, y, 1))(np.unique(x)))

plt.title(col)

plt.xlabel(col)

plt.ylabel('prices')

***Output:***

******

***Code:***

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test=train\_test\_split(features,prices,test\_size=0.20,random\_state=10)

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error

lin\_model = LinearRegression()

lin\_model.fit(x\_train,y\_train)

price\_pred = lin\_model.predict(x\_test)

Mse = mean\_squared\_error(y\_test,price\_pred)

acc = lin\_model.score(x\_test,y\_test)

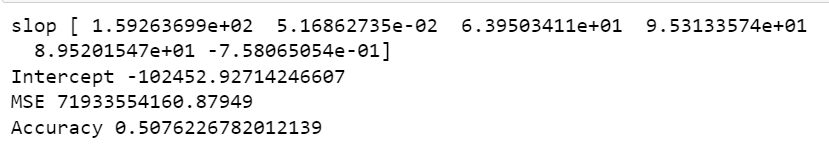
print('slop', lin\_model.coef\_)

print("Intercept", lin\_model.intercept\_)

print("MSE", Mse)

print('Accuracy',acc)

***Output:***

******

1. **Based on multiple features/variables perform Linear Regression. For example, based on a number of additional features like number of bedrooms, servant room, number of balconies, number of houses of years a house has been built – predict the price of a house.**

***Code:***

import numpy as np

import pandas as pd

import seaborn as sns

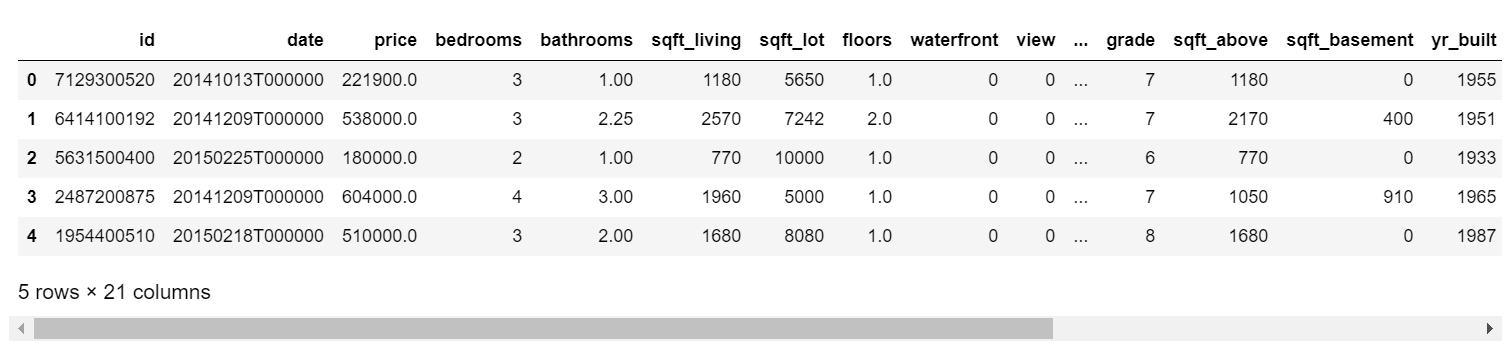
import matplotlib.pyplot as plt

%matplotlib inline

house= pd.read\_csv('kc\_house\_data.csv')

house.head()

***Output:***

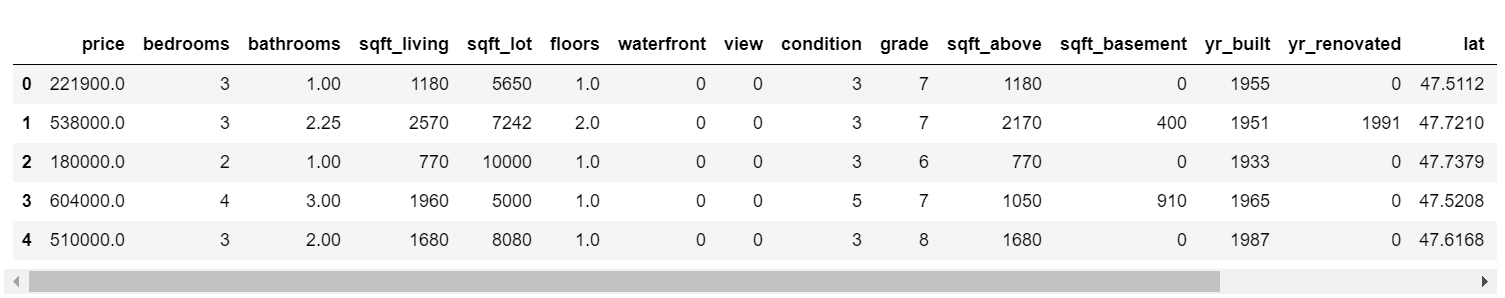
******

***Code:***

house = house.drop(['id','date','zipcode'],axis=1)

house.head()

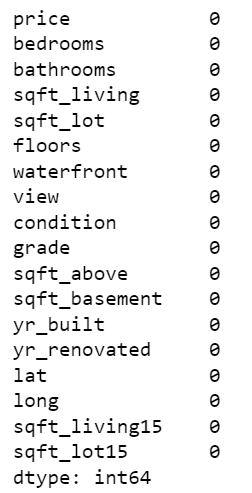
***Output:***

******

***Code:***

house.isnull().sum()

***Output:***

******

***Code:***

house.describe()

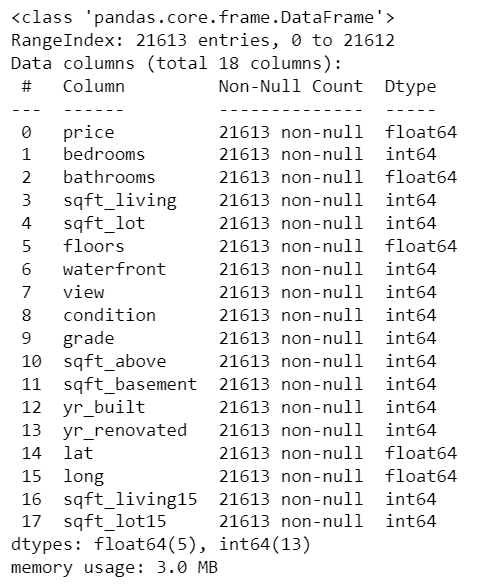
***Output:***

******

***Code:***

house.info()

***Output:***

******

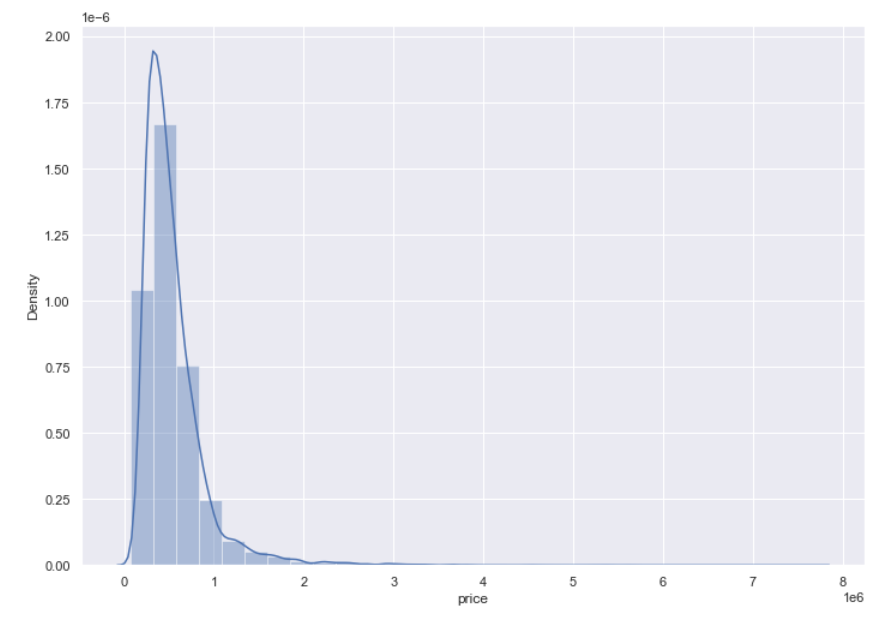
***Code:***

sns.set(rc={'figure.figsize':(11.7,8.27)})

sns.distplot(house['price'], bins=30)

plt.show()

***Output:***



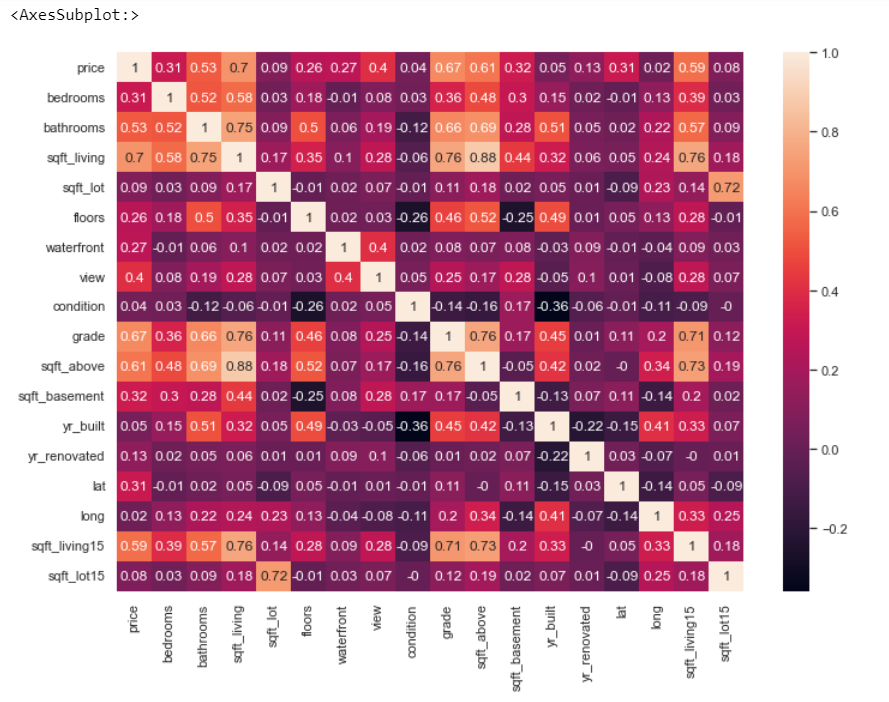
***Code:***

correlation\_matrix = house.corr().round(2)

# annot = True to print the values inside the square

sns.heatmap(data=correlation\_matrix, annot=True)

***Output:***

******

***Code:***

prices = house['price']

features = house.drop(['price'],axis=1)

plt.figure(figsize=(20, 20))

# i: index

for i, col in enumerate(features.columns):

plt.subplot(4, 5, i+1)

x = features[col]

y = prices

plt.plot(x, y, 'o')

# Create regression line

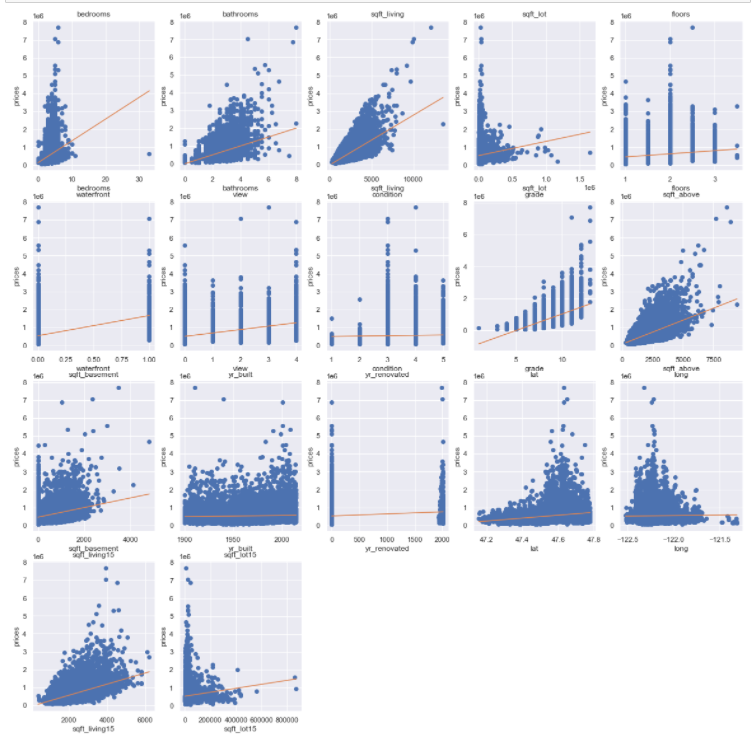
plt.plot(np.unique(x), np.poly1d(np.polyfit(x, y, 1))(np.unique(x)))

plt.title(col)

plt.xlabel(col)

plt.ylabel('prices')

***Output:***

******

***Code:***

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test=train\_test\_split(features,prices,test\_size=0.20,random\_state=10)

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error

lin\_model = LinearRegression()

lin\_model.fit(x\_train,y\_train)

price\_pred = lin\_model.predict(x\_test)

Mse = mean\_squared\_error(y\_test,price\_pred)

acc = lin\_model.score(x\_test,y\_test)

print('slop', lin\_model.coef\_)

print("Intercept", lin\_model.intercept\_)

print("MSE", Mse)

print('Accuracy',acc)

***Output:***

******

**Conclusion:** Implementation of linear regression on house data set has been done successfully.

|  |  |  |  |
| --- | --- | --- | --- |
| Internal Assessment (Mandatory Experiment) Sheet for Lab Experiment  Department of Computer Science & Engineering  ASET, Amity University Noida (U.P) | | | |
| Program | B. Tech CSE | Course Name | Fundamentals of Machine Learning |
| Course Code | CSE 313 | Semester | 6 |
| Student Name | Shaina Mehta | Enrolment no | A2305219268 |
| **Marking Criteria** | | | |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept(A) | 2 |  |  |
| Implementation(B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 |  |  |

**Experiment 5**

**Aim:** To implement logistic regression model on diabetes dataset and email dataset.

**Software Used:** Jupyter Notebook

**Code and Output:**

1. **Using Diabetes Dataset**

***Code:***

import numpy as np

import pandas as pd

import seaborn as sns

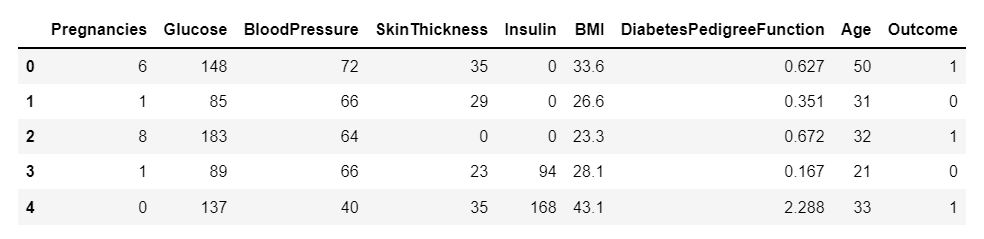
import matplotlib.pyplot as plt

%matplotlib inline

data = pd.read\_csv('diabetes.csv')

data.head()

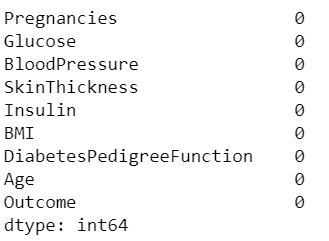
***Output:***

****

***Code:***

data.isnull().sum()

***Output:***

****

***Code:***

data.info()

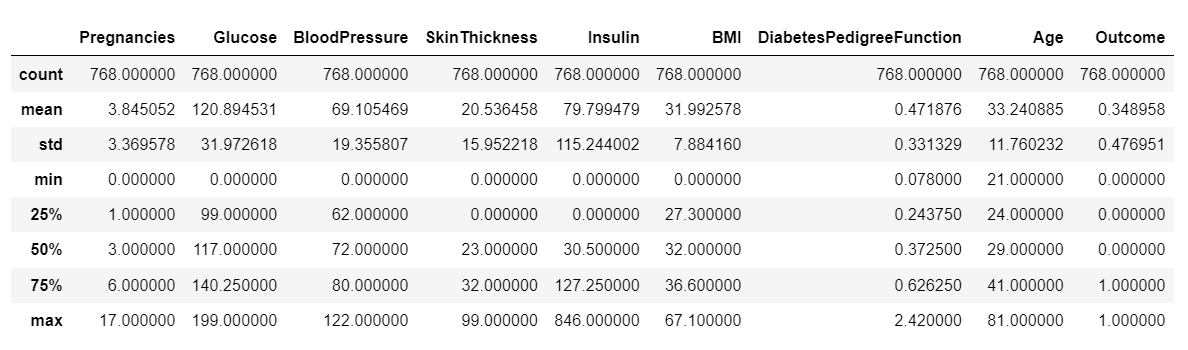
***Output:***

****

***Code:***

data.describe()

***Output:***

****

***Code:***

sns.set(rc={'figure.figsize':(11.7,8.27)})

correlation\_matrix = data.corr().round(2)

# annot = True to print the values inside the square

sns.heatmap(data=correlation\_matrix, annot=True, cmap='BuPu')

***Output:***

****

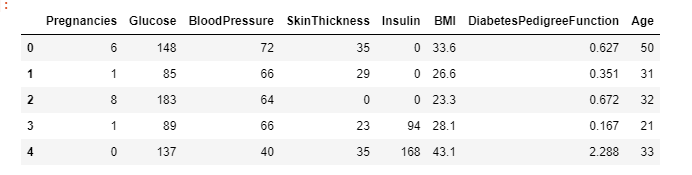
***Code:***

target = data['Outcome']

inp = data.drop(['Outcome'],axis=1)

inp.head()

***Output:***

****

***Code:***

plt.figure(figsize=(20, 20))

col = list(inp.columns)

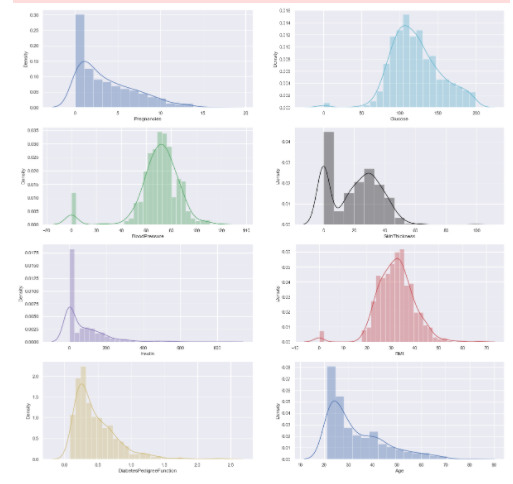
color = ['b','c','g','k','m','r','y','b']

for i in range(len(col)):

plt.subplot(4,2,i+1)

sns.distplot(inp[col[i]],hist=True,color=color[i])

***Output:***

****

***Code:***

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test=train\_test\_split(inp,target,test\_size=0.25,random\_state=0)

from sklearn.linear\_model import LogisticRegression

#from sklearn.preprocessing import StandardScalar

logreg = LogisticRegression()

logreg.fit(x\_train,y\_train)

y\_pred=logreg.predict(x\_test)

***Output:***

****

***Code:***

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, confusion\_matrix

cnf\_matrix = confusion\_matrix(y\_test, y\_pred)

cnf\_matrix

***Output:***

****

***Code:***

class\_names=[0,1] # name of classes

fig, ax = plt.subplots()

tick\_marks = np.arange(len(class\_names))

plt.xticks(tick\_marks, class\_names)

plt.yticks(tick\_marks, class\_names)

# create heatmap

sns.heatmap(pd.DataFrame(cnf\_matrix), annot=True, cmap="PuBu\_r" ,fmt='g')

ax.xaxis.set\_label\_position("top")

plt.tight\_layout()

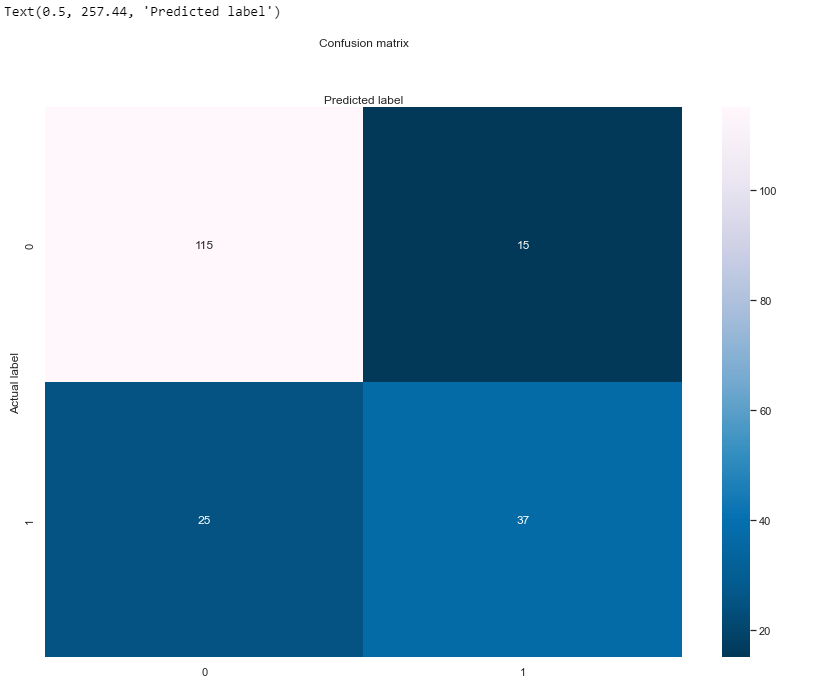
plt.title('Confusion matrix', y=1.1)

plt.ylabel('Actual label')

plt.xlabel('Predicted label')

plt.Text(0.5,257.44,'Predicted label')

***Output:***

****

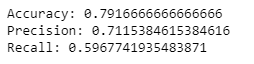
***Code:***

print("Accuracy:",accuracy\_score(y\_test, y\_pred))

print("Precision:",precision\_score(y\_test, y\_pred))

print("Recall:",recall\_score(y\_test, y\_pred))

***Output:***

****

1. **Using Emails Dataset**

***Code:***

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

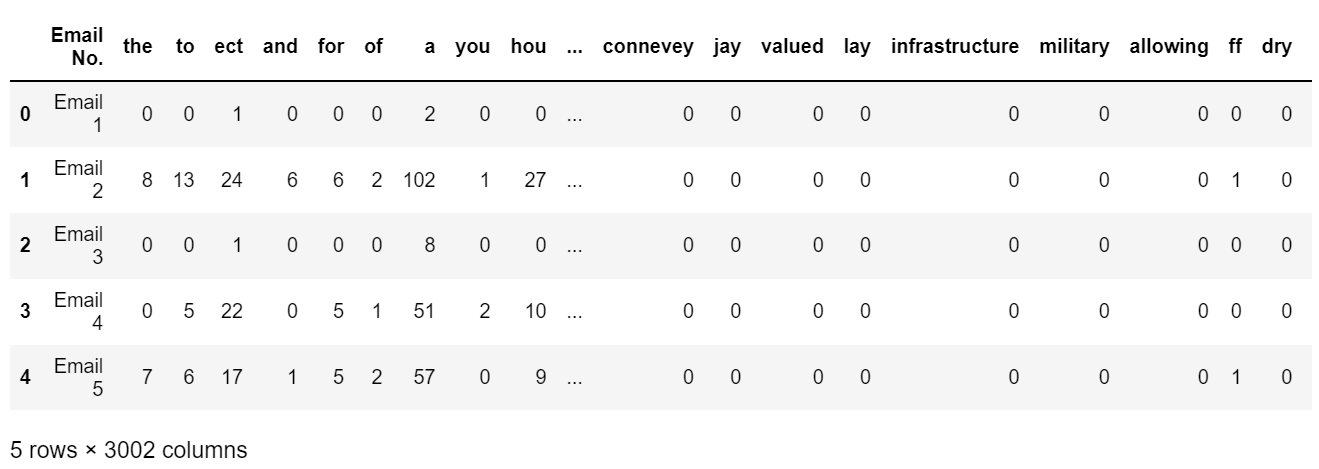
import seaborn as sns

%matplotlib inline

data = pd.read\_csv('emails.csv')

data.head()

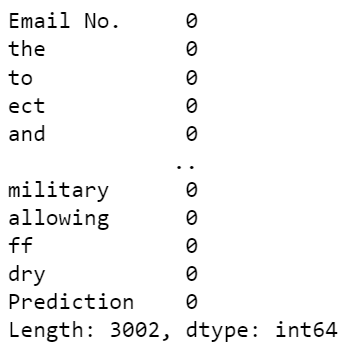
***Output:***

****

***Code:***

data.isnull().sum()

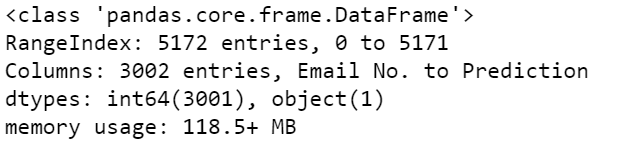
***Output:***

****

***Code:***

data.info()

***Output:***

****

***Code:***

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LogisticRegression

from sklearn.pipeline import make\_pipeline

from sklearn.preprocessing import MinMaxScaler

features = data.iloc[:, 1:-1].values

target = data.iloc[:, -1].values

x\_train,x\_test,y\_train,y\_test=train\_test\_split(features,target,test\_size=0.25,random\_state=101)

print(features.shape[0])

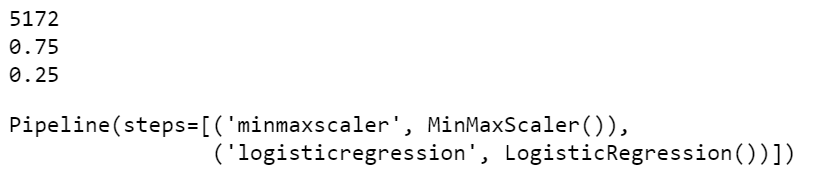
print(float(x\_train.shape[0]) / float(features.shape[0]))

print(float(x\_test.shape[0]) / float(features.shape[0]))

logmodel= make\_pipeline(MinMaxScaler(),LogisticRegression())

logmodel.fit(x\_train,y\_train)

***Output:***

****

***Code:***

y\_pred = logmodel.predict(x\_test)

y\_pred

***Output:***

****

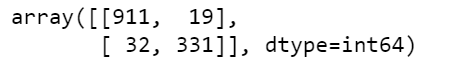
***Code:***

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, confusion\_matrix

cnf\_matrix = confusion\_matrix(y\_test, y\_pred)

cnf\_matrix

***Output:***

****

***Code:***

class\_names = [0,1]

fig,ax = plt.subplots()

tick\_marks = np.arange(len(class\_names))

plt.xticks(tick\_marks,class\_names)

plt.yticks(tick\_marks,class\_names)

sns.heatmap(pd.DataFrame(cnf\_matrix),annot=True,cmap='YlOrRd',fmt='g')

ax.xaxis.set\_label\_position("top")

plt.tight\_layout()

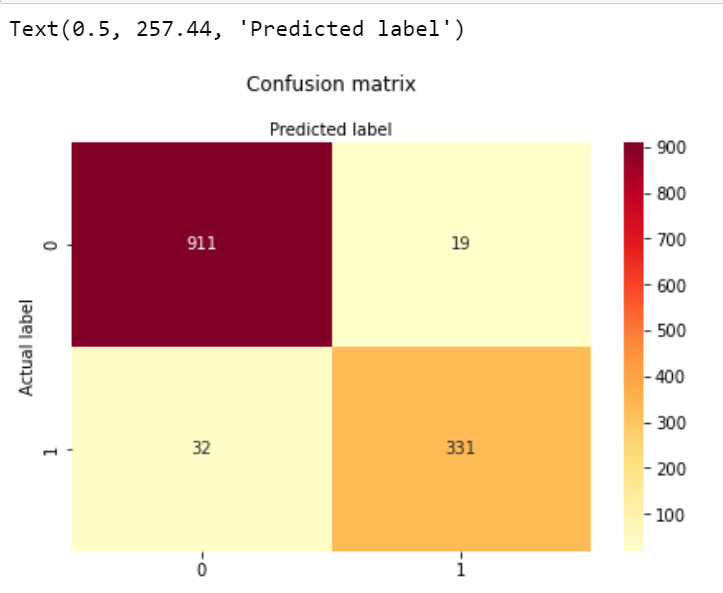
plt.title('Confusion matrix', y=1.1)

plt.ylabel('Actual label')

plt.xlabel('Predicted label')

plt.Text(0.5,257.44,'Predicted label')

***Output:***

****

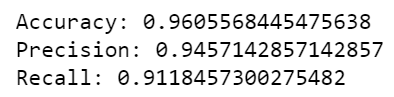
***Code:***

print("Accuracy:",accuracy\_score(y\_test, y\_pred))

print("Precision:",precision\_score(y\_test, y\_pred))

print("Recall:",recall\_score(y\_test, y\_pred))

***Output:***

****

**Conclusion:** Implementation of logistic regression on diabetes and email data set has been done successfully.

|  |  |  |  |
| --- | --- | --- | --- |
| Internal Assessment (Mandatory Experiment) Sheet for Lab Experiment  Department of Computer Science & Engineering  ASET, Amity University Noida (U.P) | | | |
| Program | B. Tech CSE | Course Name | Fundamentals of Machine Learning |
| Course Code | CSE 313 | Semester | 6 |
| Student Name | Shaina Mehta | Enrolment no | A2305219268 |
| **Marking Criteria** | | | |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept(A) | 2 |  |  |
| Implementation(B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 |  |  |

**Experiment 6**

**Aim:** To implement naïve bayes classifier on advisement and email dataset.

**Software Used:** Google Colab.

**Code and Output:**

1. **Using Advertisement Dataset**

***Code:***

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

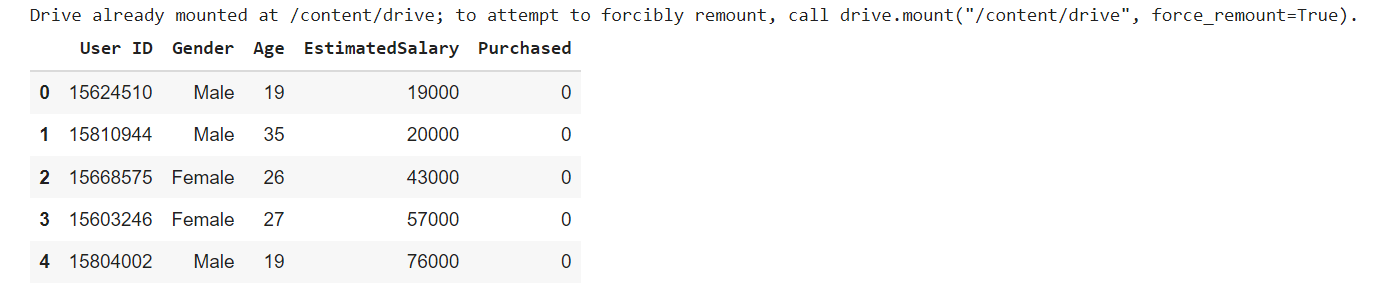
from google.colab import drive

drive.mount('/content/drive')

data = pd.read\_csv('/content/drive/My Drive/Colab Notebooks/Social\_Network\_Ads.csv')

data.head()

***Output:***

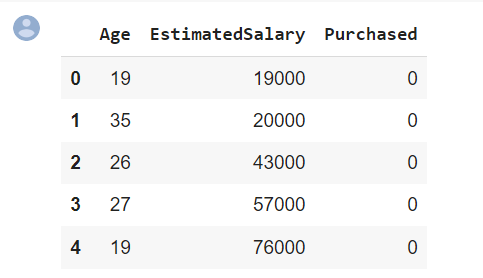
******

***Code:***

data.drop(['User ID','Gender'],axis=1,inplace=True)

data.head()

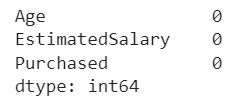
***Output:***

******

***Code:***

data.isnull().sum()

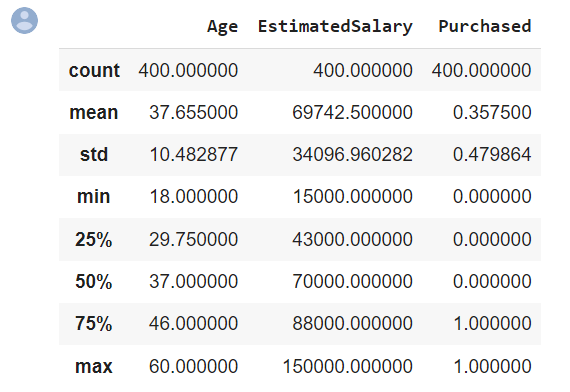
***Output:***



***Code:***

data.describe()

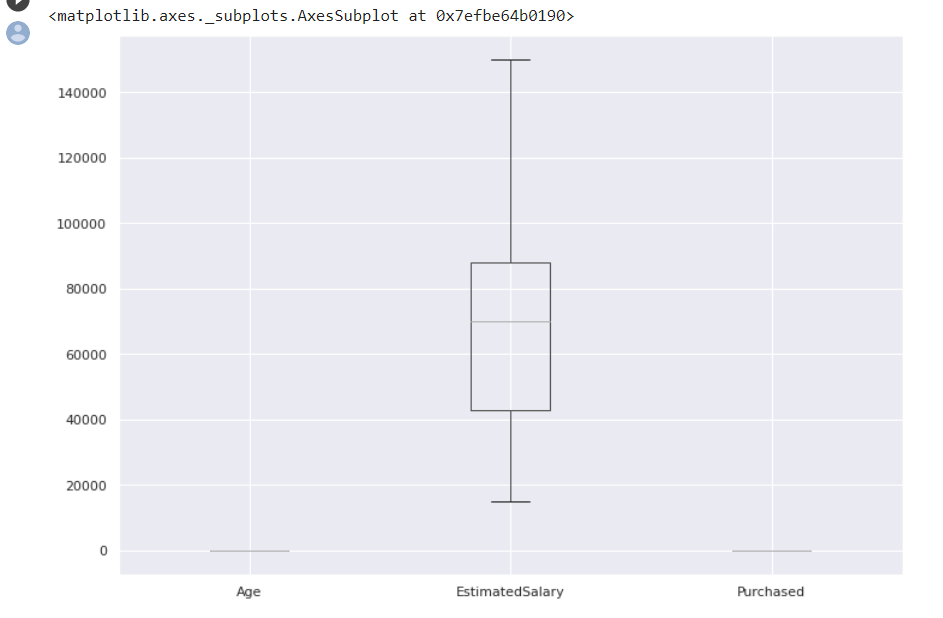
***Output:***

******

***Code:***

data.boxplot()

***Output:***

******

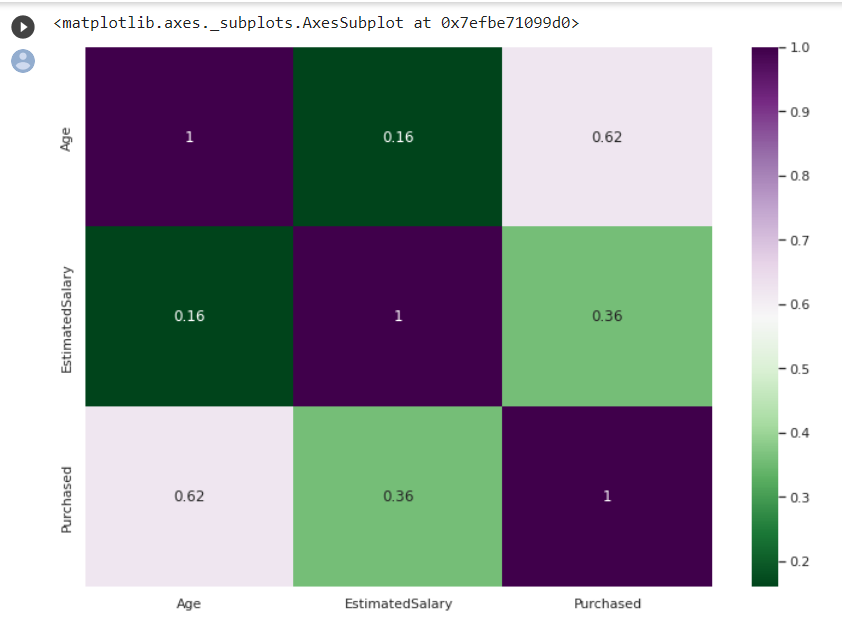
***Code:***

sns.set(rc={'figure.figsize':(11.7,8.27)})

corrMat = data.corr().round(2)

sns.heatmap(data=corrMat,annot=True,cmap='PRGn\_r')

***Output:***

******

***Code:***

sns.pairplot(data,hue='Purchased')

plt.show()

***Output:***

******

***Code:***

features = data.iloc[:,[0,1]].values

target = data.iloc[:,-1].values

***Output:***

******

***Code:***

from sklearn.model\_selection import train\_test\_split

x\_train, x\_test, y\_train, y\_test = train\_test\_split(features,target,test\_size=0.2,random\_state=0)

from sklearn.naive\_bayes import GaussianNB

from sklearn.pipeline import make\_pipeline

from sklearn.preprocessing import MinMaxScaler

nbmodel= make\_pipeline(MinMaxScaler(),GaussianNB())

nbmodel.fit(x\_train,y\_train)

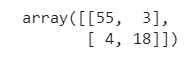
y\_pred = nbmodel.predict(x\_test)

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, confusion\_matrix

cnf\_matrix = confusion\_matrix(y\_test, y\_pred)

cnf\_matrix

***Output:***

******

***Code:***

class\_names = [0,1]

fig,ax = plt.subplots()

tick\_marks = np.arange(len(class\_names))

plt.xticks(tick\_marks,class\_names)

plt.yticks(tick\_marks,class\_names)

sns.heatmap(pd.DataFrame(cnf\_matrix),annot=True,cmap='spring',fmt='g')

ax.xaxis.set\_label\_position("top")

plt.tight\_layout()

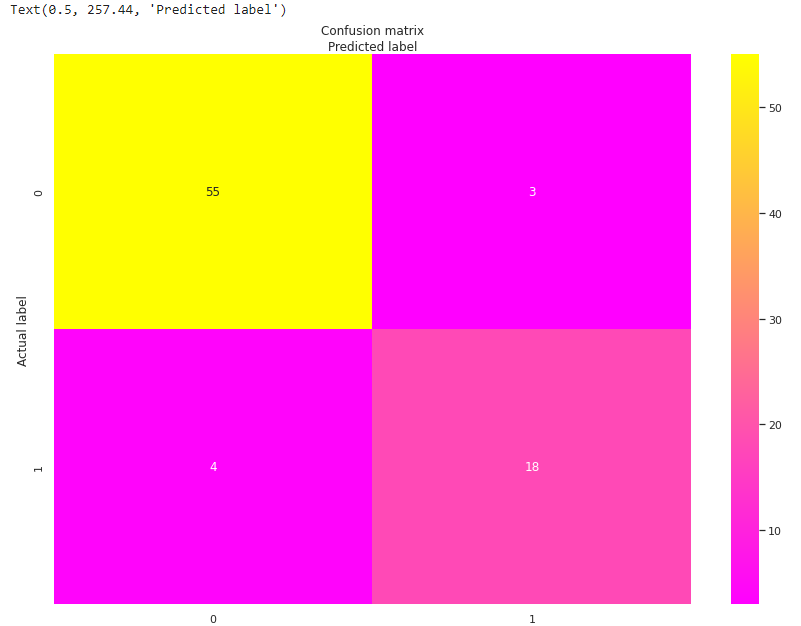
plt.title('Confusion matrix', y=1.1)

plt.ylabel('Actual label')

plt.xlabel('Predicted label')

plt.Text(0.5,257.44,'Predicted label')

***Output:***

******

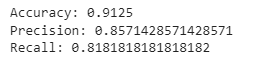
***Code:***

print("Accuracy:",accuracy\_score(y\_test, y\_pred))

print("Precision:",precision\_score(y\_test, y\_pred))

print("Recall:",recall\_score(y\_test, y\_pred))

***Output:***

******

1. **Using Emails Dataset**

***Code:***

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

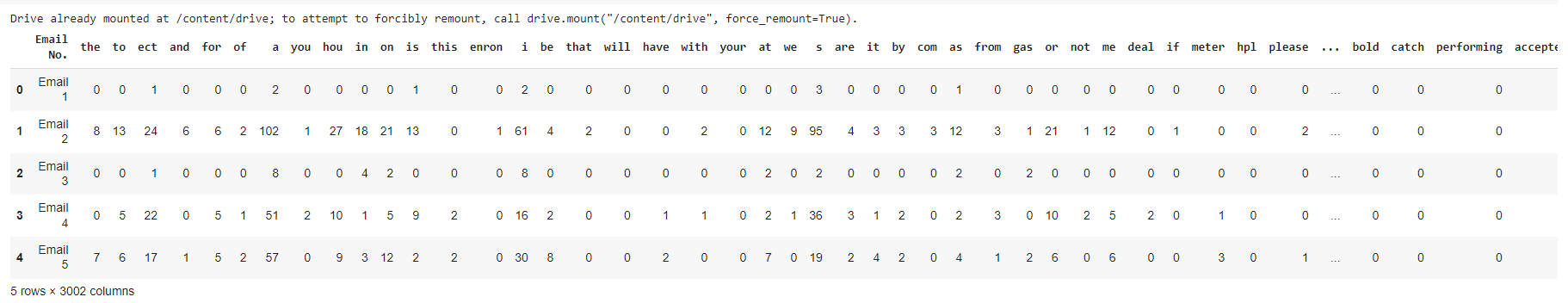
from google.colab import drive

drive.mount('/content/drive')

data = pd.read\_csv('/content/drive/My Drive/Colab Notebooks/emails.csv')

data.head()

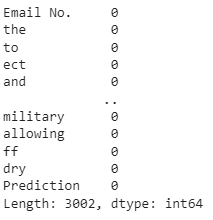
***Output:***

******

***Code:***

data.isnull().sum()

***Output:***

******

***Code:***

from sklearn.model\_selection import train\_test\_split

features = data.iloc[:,1:-1].values

target = data.iloc[:,-1].values

x\_train,x\_test,y\_train,y\_test=train\_test\_split(features,target,test\_size=0.25,random\_state=101)

from sklearn.naive\_bayes import GaussianNB

print(features.shape[0])

print(float(x\_train.shape[0]) / float(features.shape[0]))

print(float(x\_test.shape[0]) / float(features.shape[0]))

gnb = GaussianNB()

gnb.fit(x\_train,y\_train)

y\_pred = gnb.predict(x\_test)

***Output:***

******

***Code:***

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, confusion\_matrix

cnf\_matrix = confusion\_matrix(y\_test, y\_pred)

cnf\_matrix

***Output:***

******

***Code:***

class\_names = [0,1]

fig,ax = plt.subplots()

tick\_marks = np.arange(len(class\_names))

plt.xticks(tick\_marks,class\_names)

plt.yticks(tick\_marks,class\_names)

sns.heatmap(pd.DataFrame(cnf\_matrix),annot=True,cmap='GnBu',fmt='g')

ax.xaxis.set\_label\_position("top")

plt.tight\_layout()

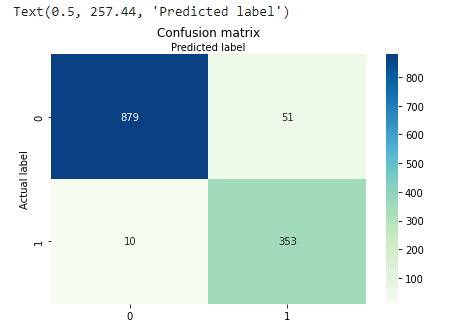
plt.title('Confusion matrix', y=1.1)

plt.ylabel('Actual label')

plt.xlabel('Predicted label')

plt.Text(0.5,257.44,'Predicted label')

***Output:***

******

***Code:***

print("Accuracy:",accuracy\_score(y\_test, y\_pred))

print("Precision:",precision\_score(y\_test, y\_pred))

print("Recall:",recall\_score(y\_test, y\_pred))

***Output:***

******

**Conclusion:** Implementation of naïve bayes classifier on advertisement and email data set has been done successfully.

|  |  |  |  |
| --- | --- | --- | --- |
| Internal Assessment (Mandatory Experiment) Sheet for Lab Experiment  Department of Computer Science & Engineering  ASET, Amity University Noida (U.P) | | | |
| Program | B. Tech CSE | Course Name | Fundamentals of Machine Learning |
| Course Code | CSE 313 | Semester | 6 |
| Student Name | Shaina Mehta | Enrolment no | A2305219268 |
| **Marking Criteria** | | | |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept(A) | 2 |  |  |
| Implementation(B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 |  |  |

**Experiment 7**

**Aim:** Use some function for neural networks, like Stochastic Gradient Descent or backpropagation - algorithm to predict the value of a variable based on the dataset.

**Software Used:** Jupyter Notebook.

**Code and Output:**

***Code:***

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

%matplotlib inline

from sklearn.datasets import load\_boston

data = load\_boston()

f = data.data

t = data.target.reshape(506,1)

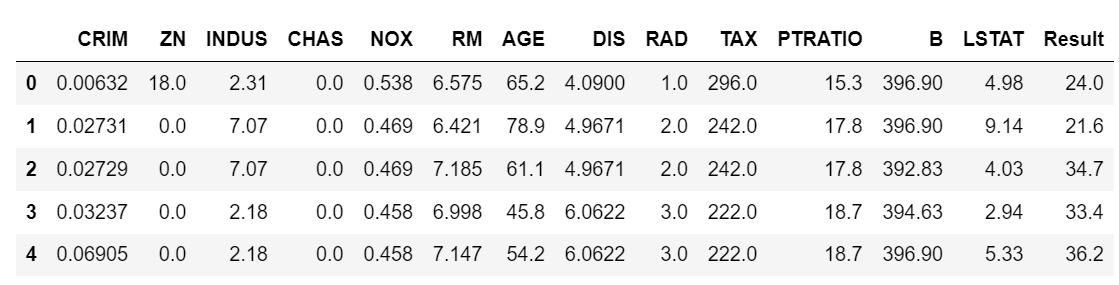
col = np.append(data.feature\_names,'Result')

d = np.hstack((f,t))

house = pd.DataFrame(d,columns=col)

house.head()

***Output:***

******

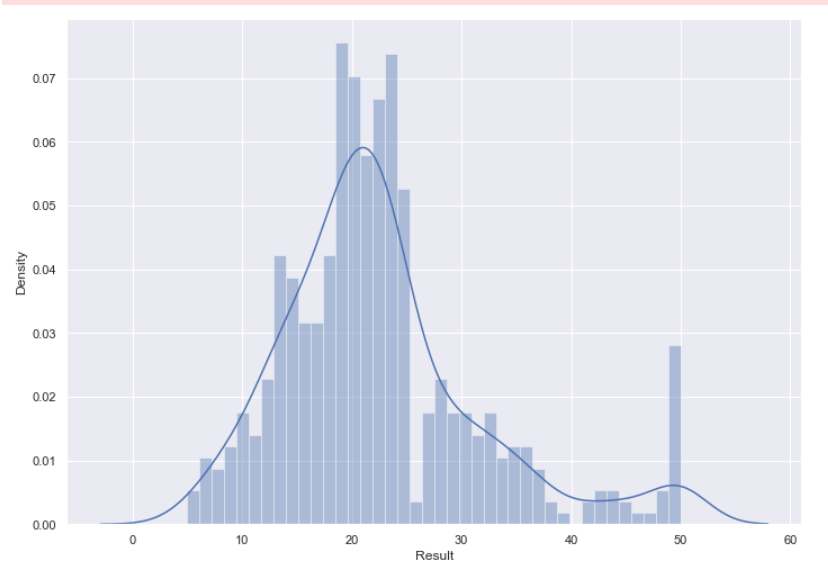
***Code:***

sns.set(rc={'figure.figsize':(11.7,8.27)})

sns.distplot(house['Result'], bins=40)

plt.show()

***Output:***



***Code:***

correlation\_matrix = house.corr().round(2)

# annot = True to print the values inside the square

sns.heatmap(data=correlation\_matrix, annot=True)

***Output:***



***Code:***

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

x\_train,x\_test,y\_train,y\_test=train\_test\_split(data.data,data.target,test\_size=0.20,random\_state=10)

from sklearn.linear\_model import SGDRegressor

clf = SGDRegressor(max\_iter=1000, tol=1e-3)

clf.fit(x\_train, y\_train)

y\_pred = clf.predict(x\_test)

from matplotlib.pyplot import figure

from sklearn.metrics import mean\_squared\_error

plt.figure(figsize=(25,6))

plt.plot(y\_test, label='Actual')

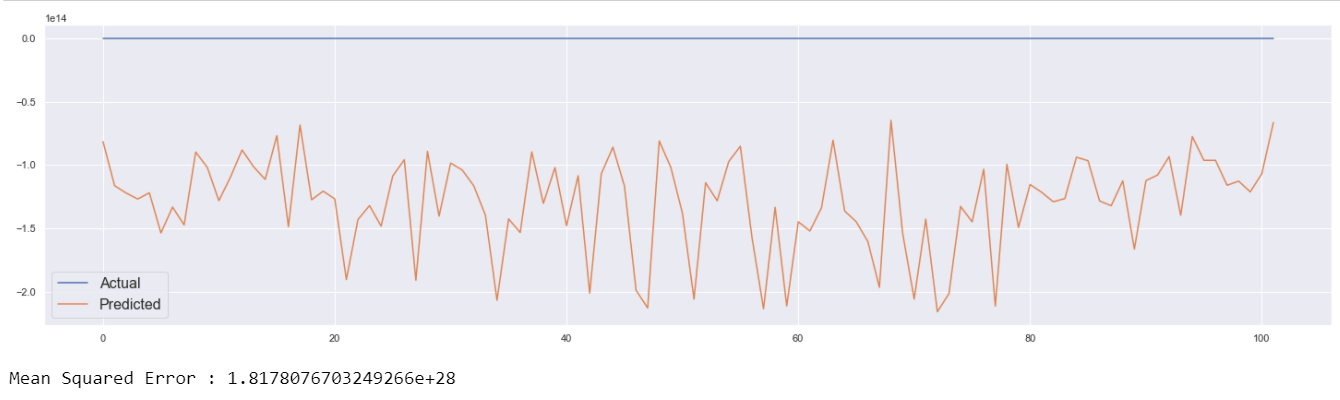
plt.plot(y\_pred, label='Predicted')

plt.legend(prop={'size': 16})

plt.show()

print('Mean Squared Error :',mean\_squared\_error(y\_test, y\_pred))

***Output:***

******

**Conclusion:** Implementation of some functions for neural networks, like Stochastic Gradient Descent has been done successfully.

|  |  |  |  |
| --- | --- | --- | --- |
| Internal Assessment (Mandatory Experiment) Sheet for Lab Experiment  Department of Computer Science & Engineering  ASET, Amity University Noida (U.P) | | | |
| Program | B. Tech CSE | Course Name | Fundamentals of Machine Learning |
| Course Code | CSE 313 | Semester | 6 |
| Student Name | Shaina Mehta | Enrolment no | A2305219268 |
| **Marking Criteria** | | | |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept(A) | 2 |  |  |
| Implementation(B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 |  |  |

**Experiment 8**

**Aim:** To implement and evaluate a classification using SVM.

**Software Used:** Jupyter Notebook.

**Code and Output:**

***Code:***

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

%matplotlib inline

data = pd.read\_csv('spam.csv',encoding='Windows-1252')

data.head()

***Output:***

******

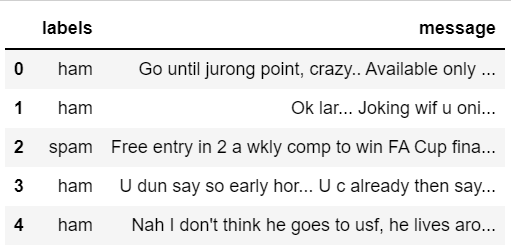
***Code:***

data = data.iloc[:,0:2]

data = data.rename(columns={'v1':'labels','v2':'message'})

data.head()

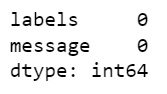
***Output:***

******

***Code:***

data.isnull().sum()

***Output:***

******

***Code:***

data.dtypes

***Output:***

******

***Code:***

sns.heatmap(data.isnull(),cbar='plasma')

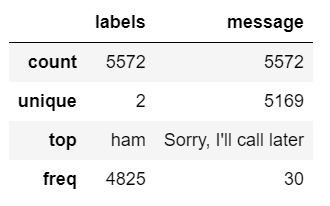
***Output:***

******

***Code:***

data.describe()

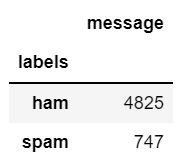
***Output:***

******

***Code:***

data.groupby('labels').count()

***Output:***

******

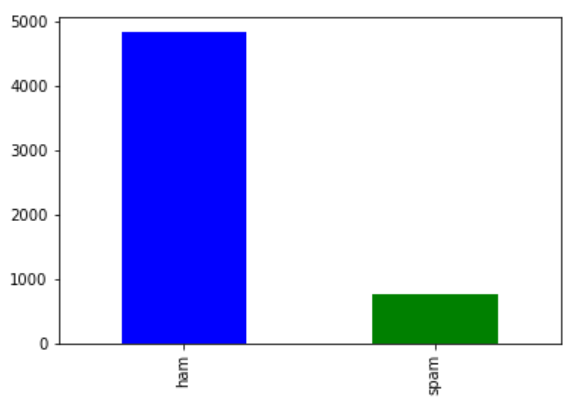
***Code:***

dat = pd.value\_counts(data['labels'],sort=True)

dat.plot(kind='bar',color=['blue','green'])

plt.show()

***Output:***

******

***Code:***

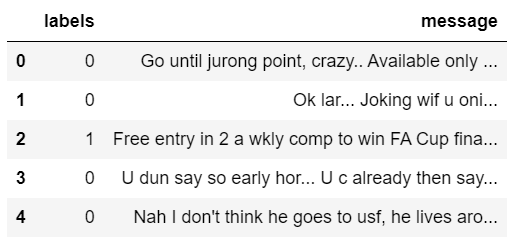
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()

data['labels'] = le.fit\_transform(data['labels'])

data.head()

***Output:***

******

***Code:***

from sklearn.feature\_extraction.text import CountVectorizer

from collections import Counter

count1 = Counter(" ".join(data[data['labels']==0]['message']).split()).most\_common(20)

df1 = pd.DataFrame.from\_dict(count1)

df1 = df1.rename(columns={0:"words in non-spam",1:"count"})

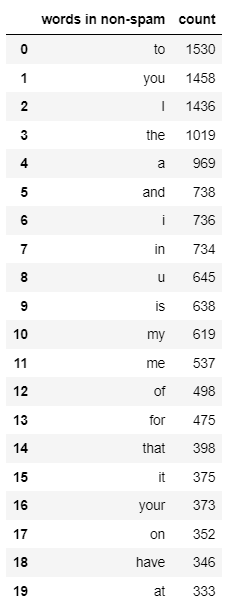
count2 = Counter(" ".join(data[data['labels']==1]['message']).split()).most\_common(20)

df2 = pd.DataFrame.from\_dict(count2)

df2 = df2.rename(columns={0:"words in spam",1:"count"})

df1

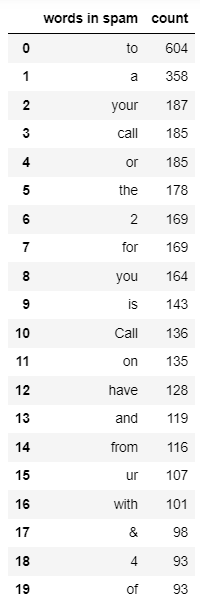
***Output:***

******

***Code:***

df2

***Output:***



***Code:***

df1.plot.bar(legend=False,color='violet')

y\_pos = np.arange(len(df1['words in non-spam']))

plt.xticks(y\_pos,df1['words in non-spam'])

plt.title('More frequent words in non - spam messages')

plt.xlabel('words')

plt.ylabel('numbers')

plt.show()

***Output:***



***Code:***

df2.plot.bar(legend=False,color='orange')

y\_pos = np.arange(len(df2['words in spam']))

plt.title('most frequent words in spam message')

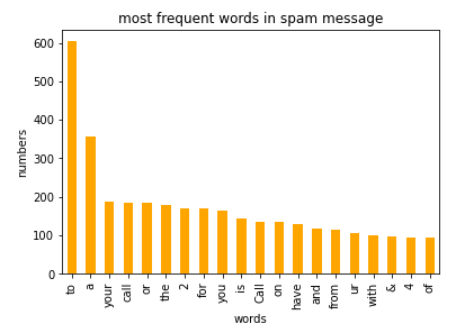
plt.xticks(y\_pos,df2['words in spam'])

plt.ylabel('numbers')

plt.xlabel('words')

plt.show()

***Output:***



***Code:***

# count vector will remove all the text

count\_vector = CountVectorizer(stop\_words='english')

x = count\_vector.fit\_transform(data['message'])

np.shape(x)

***Output:***

******

***Code:***

from sklearn.model\_selection import train\_test\_split

from sklearn.svm import SVC

x\_train,x\_test,y\_train,y\_test = train\_test\_split(x,data['labels'],test\_size=0.2,random\_state=10)

svm\_model = SVC(kernel='linear',C=1)

svm\_model.fit(x\_train,y\_train)

y\_pred = svm\_model.predict(x\_test)

cnf\_matrix = confusion\_matrix(y\_test,y\_pred)

class\_names = ['ham','spam']

fig, ax = plt.subplots()

tick\_marks = np.arange(len(class\_names))

plt.xticks(tick\_marks,class\_names)

plt.yticks(tick\_marks,class\_names)

sns.heatmap(pd.DataFrame(cnf\_matrix),annot=True,cmap='Blues\_r',fmt='g')

ax.xaxis.set\_label\_position("top")

plt.tight\_layout()

plt.title('Confusion matrix', y=1.1)

plt.ylabel('Actual label')

plt.xlabel('Predicted label')

plt.Text(0.5,257.44,'Predicted label')

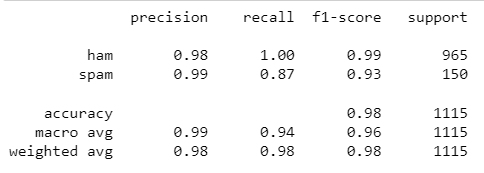
***Output:***



***Code:***

print(classification\_report(y\_test,y\_pred,target\_names=class\_names))

***Output:***

******

***Code:***

print(accuracy\_score(y\_test,y\_pred\_test))

***Output:***

******

**Conclusion:** Implementation and evaluation of SVM classifier on spam data set has been done successfully.

|  |  |  |  |
| --- | --- | --- | --- |
| Internal Assessment (Mandatory Experiment) Sheet for Lab Experiment  Department of Computer Science & Engineering  ASET, Amity University Noida (U.P) | | | |
| Program | B. Tech CSE | Course Name | Fundamentals of Machine Learning |
| Course Code | CSE 313 | Semester | 6 |
| Student Name | Shaina Mehta | Enrolment no | A2305219268 |
| **Marking Criteria** | | | |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept(A) | 2 |  |  |
| Implementation(B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 |  |  |

**Experiment 9**

**Aim:** To implement and evaluate a classification using the K-means algorithm.

**Software Used:** Jupyter Notebook.

**Code and Output:**

1. **Recognising Digits using K-Means Clustering**

***Code:***

from sklearn.datasets import load\_digits

from sklearn.cluster import KMeans

import numpy as np

import matplotlib.pyplot as plt

digits = load\_digits()

digits.data.shape

***Output:***

******

***Code:***

digits.target.shape

***Output:***

******

***Code:***

kmeans = KMeans(n\_clusters=10,random\_state=10)

clusters = kmeans.fit\_predict(digits.data)

kmeans.cluster\_centers\_.shape

***Output:***

******

***Code:***

fig, ax = plt.subplots(2,5,figsize=(8,3))

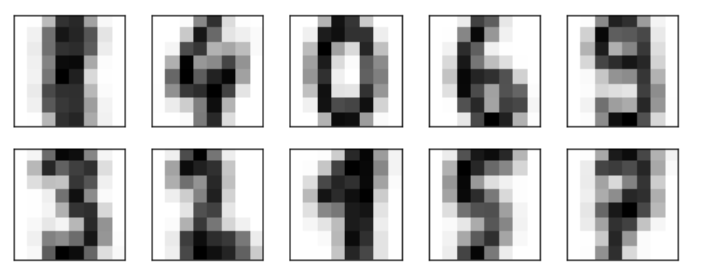
centers = kmeans.cluster\_centers\_.reshape(10,8,8)

for axi, center in zip(ax.flat,centers):

axi.set(xticks=[], yticks=[])

axi.imshow(center, interpolation='nearest', cmap=plt.cm.binary)

***Output:***

******

***Code:***

from scipy.stats import mode

labels = np.zeros\_like(clusters)

for i in range(10):

mask = (clusters==i)

labels[mask] = mode(digits.target[mask])[0]

from sklearn.metrics import accuracy\_score

accuracy\_score(digits.target, labels)

***Output:***

******

***Code:***

from sklearn.metrics import confusion\_matrix

mat = confusion\_matrix(digits.target,labels)

sns.heatmap(mat.T,annot=True,fmt='d',cbar=False,cmap='Pastel2\_r',

xticklabels=digits.target\_names,yticklabels=digits.target\_names)

plt.title('Confusion Matrix')

plt.xlabel('true label')

plt.ylabel('predicted label');

plt.show()

***Output:***

******

1. **Image Compression Using K=Means Clustering**

***Code:***

from skimage import io

from sklearn.cluster import KMeans

import numpy as np

image = io.imread('Lenna.png')

io.imshow(image)

io.show()

***Output:***

******

***Code:***

rows = image.shape[0]

columns = image.shape[1]

image = image.reshape(rows\*columns,3)

kmeans = KMeans(n\_clusters=16)

kmeans.fit(image)

compressed\_image = kmeans.cluster\_centers\_[kmeans.labels\_]

#print(compressed\_image)

compressed\_image = np.clip(compressed\_image.astype('uint8'),0,255)

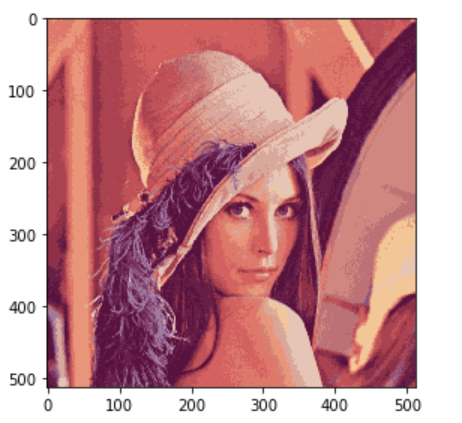
compressed\_image = compressed\_image.reshape(rows,columns,3)

io.imsave('lenna\_compressed.png',compressed\_image)

io.imshow(compressed\_image)

io.show()

***Output:***

******

**Conclusion:** Implementation and evaluation of the classification K-means algorithm have been done successfully.

|  |  |  |  |
| --- | --- | --- | --- |
| Internal Assessment (Mandatory Experiment) Sheet for Lab Experiment  Department of Computer Science & Engineering  ASET, Amity University Noida (U.P) | | | |
| Program | B. Tech CSE | Course Name | Fundamentals of Machine Learning |
| Course Code | CSE 313 | Semester | 6 |
| Student Name | Shaina Mehta | Enrolment no | A2305219268 |
| **Marking Criteria** | | | |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept(A) | 2 |  |  |
| Implementation(B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 |  |  |

**Experiment 10**

**Aim:** Implement and evaluate a classification using Decision Tree.

**Software Used:** Jupyter Notebook.

**Code and Output:**

***Code:***

import numpy as np

import pandas as pd

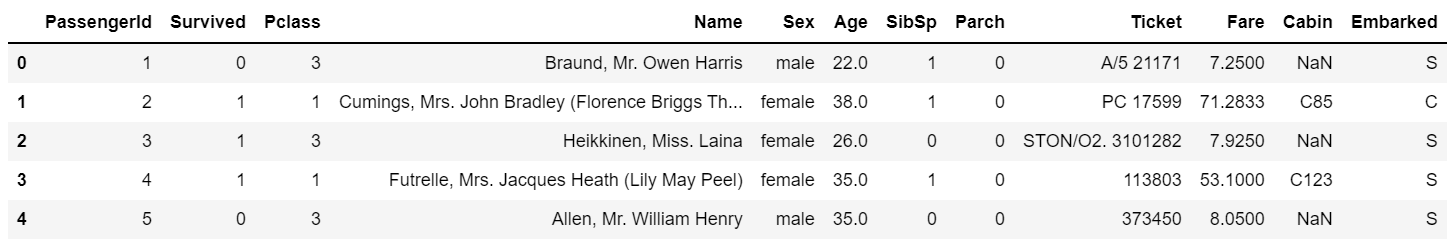
import matplotlib.pyplot as plt

import seaborn as sns

%matplotlib inline

data = pd.read\_csv('titanic.csv')

data.head()

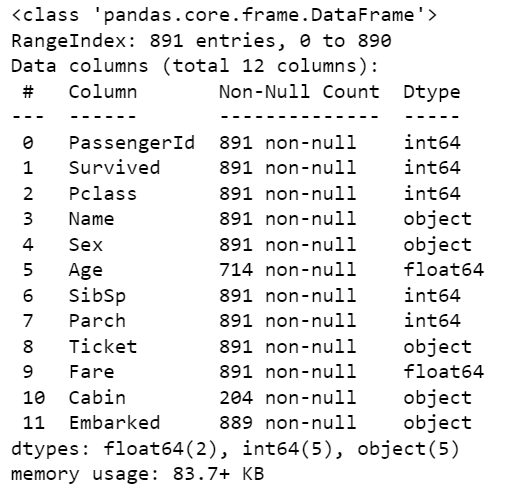


***Output:***

***Code:***

data.info()

***Output:***

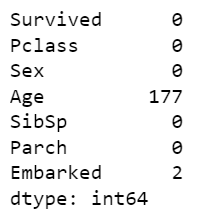


***Code:***

data = data.drop(['PassengerId','Name','Ticket','Fare','Cabin'],axis=1)

data.isnull().sum()

***Output:***

******

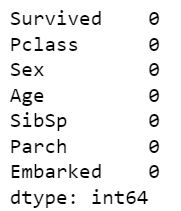
***Code:***

data.interpolate(method ='linear', limit\_direction ='forward', inplace = True)

data=data.dropna()

data.isnull().sum()

***Output:***

******

***Code:***

from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()

data['Sex'] = le.fit\_transform(data['Sex'])

data['Embarked'] = le.fit\_transform(data['Embarked'])

data.head()

***Output:***

******

***Code:***

target = data.iloc[:,0:1].values

features = data.iloc[:,1:].values

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test = train\_test\_split(features,target,test\_size=0.2,random\_state=10)

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

# Decision Tree Classifier

clf = DecisionTreeClassifier()

clf.fit(x\_train,y\_train)

y\_pred = clf.predict(x\_test)

cnf\_matrix = confusion\_matrix(y\_test,y\_pred)

class\_names = ['0','1']

fig, ax = plt.subplots()

tick\_marks = np.arange(len(class\_names))

plt.xticks(tick\_marks,class\_names)

plt.yticks(tick\_marks,class\_names)

sns.heatmap(pd.DataFrame(cnf\_matrix),annot=True,cmap='Blues\_r',fmt='g')

ax.xaxis.set\_label\_position("top")

plt.tight\_layout()

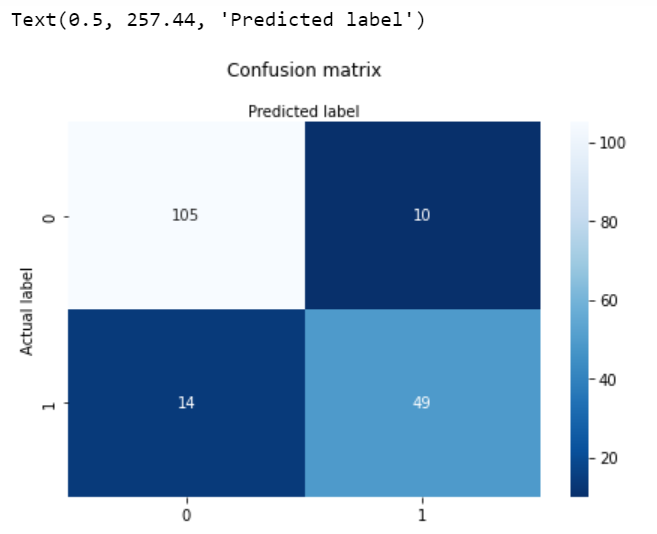
plt.title('Confusion matrix', y=1.1)

plt.ylabel('Actual label')

plt.xlabel('Predicted label')

plt.Text(0.5,257.44,'Predicted label')

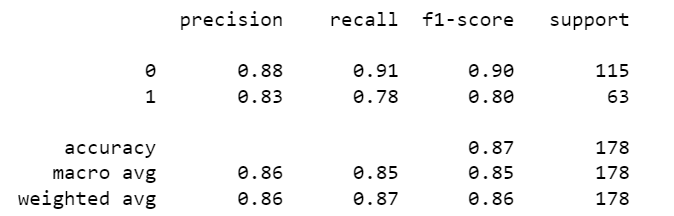
***Output:***

******

***Code:***

print(classification\_report(y\_test,y\_pred,target\_names=class\_names))

***Output:***

******

***Code:***

print(accuracy\_score(y\_test,y\_pred))

***Output:***

******

**Conclusion:** Implementation and evaluation of decision tree classifier has been done successfully.

|  |  |  |  |
| --- | --- | --- | --- |
| Internal Assessment (Mandatory Experiment) Sheet for Lab Experiment  Department of Computer Science & Engineering  ASET, Amity University Noida (U.P) | | | |
| Program | B. Tech CSE | Course Name | Fundamentals of Machine Learning |
| Course Code | CSE 313 | Semester | 6 |
| Student Name | Shaina Mehta | Enrolment no | A2305219268 |
| **Marking Criteria** | | | |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept(A) | 2 |  |  |
| Implementation(B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 |  |  |

**Experiment 11**

**Aim:** Implement and evaluate a classification using Random Forest.

**Software Used:** Jupyter Notebook.

**Code and Output:**

***Code:***

import numpy as np

import pandas as pd

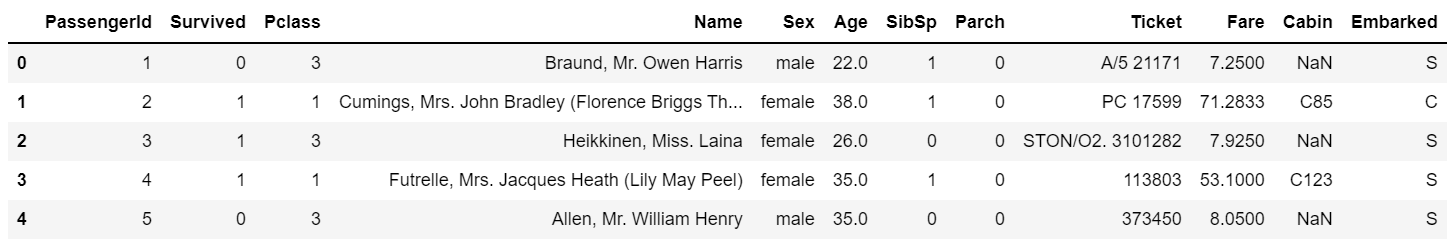
import matplotlib.pyplot as plt

import seaborn as sns

%matplotlib inline

data = pd.read\_csv('titanic.csv')

data.head()

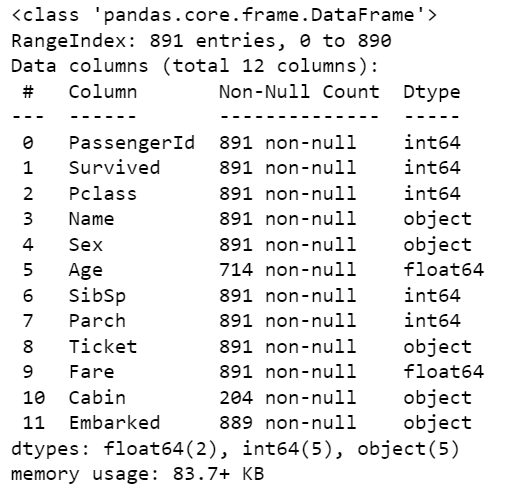


***Output:***

***Code:***

data.info()

***Output:***

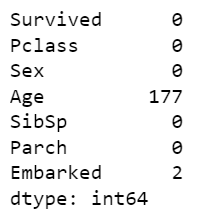


***Code:***

data = data.drop(['PassengerId','Name','Ticket','Fare','Cabin'],axis=1)

data.isnull().sum()

***Output:***

******

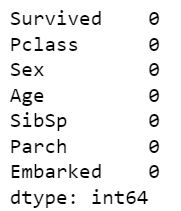
***Code:***

data.interpolate(method ='linear', limit\_direction ='forward', inplace = True)

data=data.dropna()

data.isnull().sum()

***Output:***

******

***Code:***

from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()

data['Sex'] = le.fit\_transform(data['Sex'])

data['Embarked'] = le.fit\_transform(data['Embarked'])

data.head()

***Output:***

******

***Code:***

target = data.iloc[:,0:1].values

features = data.iloc[:,1:].values

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test = train\_test\_split(features,target,test\_size=0.2,random\_state=10)

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

# Random Forest Classifier

clf = RandomForestClassifier(max\_depth=3, random\_state=0)

clf.fit(x\_train,y\_train.ravel())

y\_pred = clf.predict(x\_test)

cnf\_matrix = confusion\_matrix(y\_test,y\_pred)

class\_names = ['0','1']

fig, ax = plt.subplots()

tick\_marks = np.arange(len(class\_names))

plt.xticks(tick\_marks,class\_names)

plt.yticks(tick\_marks,class\_names)

sns.heatmap(pd.DataFrame(cnf\_matrix),annot=True,cmap='Blues\_r',fmt='g')

ax.xaxis.set\_label\_position("top")

plt.tight\_layout()

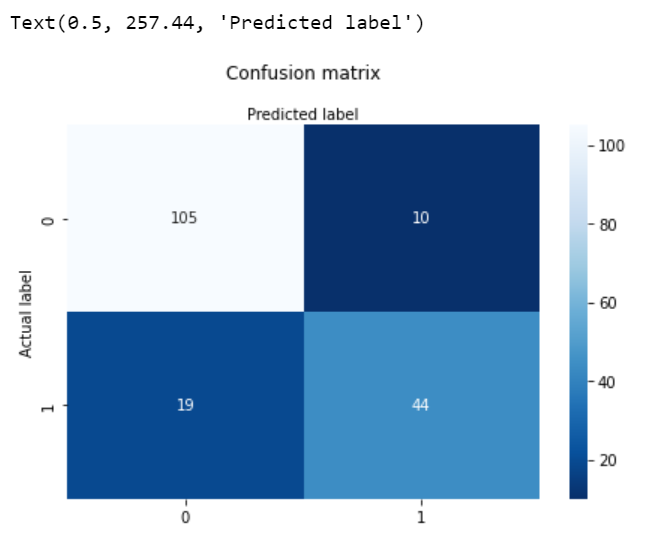
plt.title('Confusion matrix', y=1.1)

plt.ylabel('Actual label')

plt.xlabel('Predicted label')

plt.Text(0.5,257.44,'Predicted label')

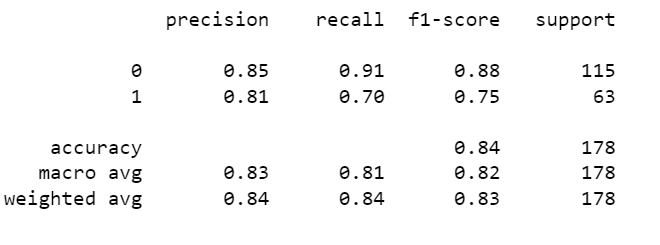
***Output:***

******

***Code:***

print(classification\_report(y\_test,y\_pred,target\_names=class\_names))

***Output:***

******

***Code:***

print(accuracy\_score(y\_test,y\_pred))

***Output:***

******

**Conclusion:** Implementation and evaluation of random forest classifier has been done successfully

|  |  |  |  |
| --- | --- | --- | --- |
| Internal Assessment (Mandatory Experiment) Sheet for Lab Experiment  Department of Computer Science & Engineering  ASET, Amity University Noida (U.P) | | | |
| Program | B. Tech CSE | Course Name | Fundamentals of Machine Learning |
| Course Code | CSE 313 | Semester | 6 |
| Student Name | Shaina Mehta | Enrolment no | A2305219268 |
| **Marking Criteria** | | | |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept(A) | 2 |  |  |
| Implementation(B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 |  |  |

**Experiment 12**

**Aim:** To implement Regularization on the dataset.

**Software Used:** Jupyter Notebook.

**Code and Output:**

***Code:***

import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

data = pd.read\_csv('melb\_data.csv')

data.head()

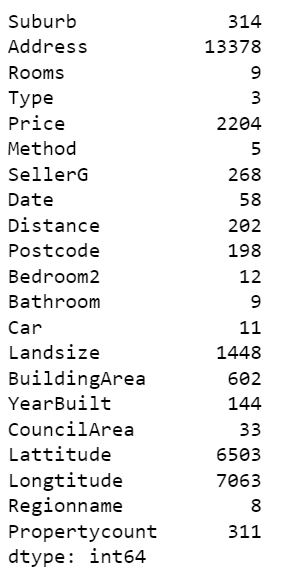
***Output:***

****

***Code:***

data.nunique()

***Output:***

****

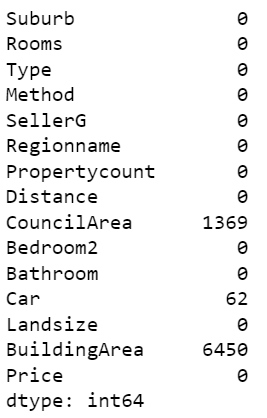
***Code:***

cols\_to\_use=['Suburb','Rooms', 'Type','Method', 'SellerG', 'Regionname', 'Propertycount', 'Distance', 'CouncilArea', 'Bedroom2', 'Bathroom', 'Car', 'Landsize','BuildingArea', 'Price']

data=data[cols\_to\_use]

data.isnull().sum()

***Output:***

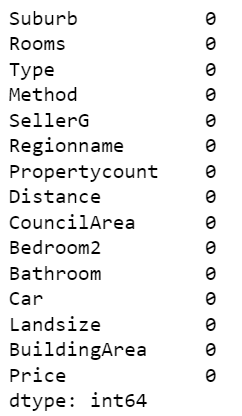
****

***Code:***

data.dropna(inplace=True)

data.isnull().sum()

***Output:***

****

***Code:***

data=pd.get\_dummies(data)

x=data.drop('Price', axis=1)

y=data['Price']

from sklearn.model\_selection import train\_test\_split

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x,y,test\_size=0.3,random\_state=0)

from sklearn.linear\_model import LinearRegression

reg = LinearRegression()

reg.fit(x\_train,y\_train)

***Output:***

****

***Code:***

reg.score(x\_test,y\_test)

***Output:***

****

***Code:***

reg.score(x\_train,y\_train)

***Output:***

****

***Code:***

from sklearn import linear\_model

lasso\_reg = linear\_model.Lasso(alpha=50,max\_iter=1000,tol=0.1)

lasso\_reg.fit(x\_train,y\_train)

***Output:***

****

**Code:**

lasso\_reg.score(x\_test, y\_test)

***Output:***

****

***Code:***

lasso\_reg.score(x\_train, y\_train)

***Output:***

****

***Code:***

from sklearn.linear\_model import Ridge

ridge\_reg=linear\_model.Ridge(alpha=50, max\_iter=1000, tol=0.1)

ridge\_reg.fit(x\_train, y\_train)

***Output:***

****

***Code:***

ridge\_reg.score(x\_test, y\_test)

***Output:***

****

***Code:***

ridge\_reg.score(x\_train, y\_train)

***Output:***

****

**Conclusion:** Implementation of Regularization on the dataset has been done successfully.

|  |  |  |  |
| --- | --- | --- | --- |
| Internal Assessment (Mandatory Experiment) Sheet for Lab Experiment  Department of Computer Science & Engineering  ASET, Amity University Noida (U.P) | | | |
| Program | B. Tech CSE | Course Name | Fundamentals of Machine Learning |
| Course Code | CSE 313 | Semester | 6 |
| Student Name | Shaina Mehta | Enrolment no | A2305219268 |
| **Marking Criteria** | | | |
| **Criteria** | **Total Marks** | **Marks Obtained** | **Comments** |
| Concept(A) | 2 |  |  |
| Implementation(B) | 2 |  |  |
| Performance (C) | 2 |  |  |
| Total | 6 |  |  |