housing

```
mport numpy as np
list  =  [0,1,2,3,4]
arr  =  np.array(list)
print(type(arr))
print(arr)


import numpy as np
list  =  [0,1,2,3,4]
print(list)
li2  =  list+2
print(li2)


import numpy as np
list  =  [0,1,2,3,4]
arr  =  np.array(list)
print(arr)
arr  =arr+2
print(arr)


import numpy as np
list  =  [[0,1,2],[3,4,5],[6,7,8]]
arr  =  np.array(list)
print(arr)

import numpy as np
list  =  [[0,1,2],[3,4,5],[6,7,8]]
arr = np.array(list, dtype="float")
print(arr)

import numpy as np
list  =  [[0,1,2],[3,4,5],[6,7,8]]
arr = np.array(list, dtype="float")
print(arr)
arr1  =  arr.astype("int").astype("str")
print(arr1)

import numpy as np
arr1 = np.array([1,'a'], dtype="object")
print(arr1)

import numpy as np
arr  =  arr1.tolist()
print(arr)

import numpy as np
list  =  [[0,1,2],[3,4,5],[6,7,8],[9,10,11]]
```

```python
arr = np.array(list, dtype="float")
print("Shape:", arr.shape)
print("Data type:", arr.dtype)
print("Size:",  arr.size)
print("Num Diamention:",arr.ndim)

import numpy as np
list  = [[0,1,2],[3,4,5],[6,7,8]]
arr = np.array(list, dtype="float")
boo = arr>2
print(boo)

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
data=pd.read_csv("C://Users//DSU-CSE513-25//Downloads//Housing.csv")
print("\n sample Data")
print(data.head(10))

print("\n Sample data:")
print(data.tail(10))

print("basic Information:")
print(data.info())

data.dtypes

data.columns

data.shape

print("\n Summary Statistics:")
print(data.describe())

data.isnull()

data.isnull().sum()

data.dropna(inplace=True)
data.count()
duplicate_row_df  =  data[data.duplicated()]
print("No. of duplicate rows: ",duplicate_row_df.shape)
data.count()
data = data.drop_duplicates()
data.count()

features =data.columns
features
```

```
Zero_val_cols = (data[features]==0).sum()
Zero_val_cols

data.isnull().sum() / len(data) * 100

data[['area','bedrooms']] = data[['area','bedrooms']].replace(0,np.NaN)

data['area'] = data['area'].fillna (data.area.median())

one_hot_encoded = pd.get_dummies(data,columns=['mainroad'],prefix=['mainroad'])
print("One Hot Encoded Data:")
print(one_hot_encoded)

from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
data['Guestroom_labelEncoder'] = label_encoder.fit_transform(data['guestroom'])
print("\n Label Encoded data:")
print(data)


plt.figure(figsize=(8,6))
sns.histplot(data['price'], bins=20,
kde=True) plt.title("Distributon of House
Price") plt.xlabel("Price ($)")
plt.ylabel("Frequency")
plt.grid(True)
plt.show()

plt.figure(figsize=(8,6))
sns.boxplot(data=data, x='price')
plt.title("Outliers in House Price")
plt.xlabel("Price ($)")
plt.grid(True)
plt.show()

plt.figure(figsize=(8,6))
sns.scatterplot(data = data, x='area' , y='price')
plt.title("Area v/s Price")
plt.xlabel("Area")
plt.ylabel("Price
($)") plt.grid(True)
plt.show()
```

Salary

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
data={'Voltage': [0,1,2,3,4,5,6,7,8,9,10], 'Current':[0,1,2,3,4,5,6,7,8,9,10]}
df=pd.DataFrame(data)
 df
plt.scatter(data=df, x='Voltage', y='Current')
plt.title("Voltage v/s Current") plt.ylabel("Current")
plt.xlabel("Voltage")
plt.show()

salary_df=pd.read_csv("C://Users//DSU-CSE513-25//Downloads//Salary_Data.csv")
salary_df.shape
salary_df.info()
salary_df.head()
salary_df.describe()

plt.scatter(data=salary_df, x='YearsExperience', y='Salary')
plt.title("Salary based on years of experience") plt.xlabel("Years of
Experience")
plt.ylabel("Salary")
plt.show()

x = salary_df['YearsExperience'].values y =
salary_df['Salary'].values
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)
x_train.shape, x_test.shape, y_train.shape, y_test.shape
type(x_train)
from sklearn.linear_model import LinearRegression
reg_model = LinearRegression()
reg_model.fit(x_train.reshape(-1,1),  y_train.reshape(-1,1))

reg_model.coef_
reg_model.intercept_
y_predicted = reg_model.predict(x_test.reshape(-1, 1))
y_predicted
y_test

from sklearn.metrics import mean_squared_error, r2_scorer_square=
r2_score(y_test, y_predicted)
r_square
rmse = mean_squared_error(y_test, y_predicted)rmse

plt.scatter(x=x_test, y=y_test, color='red')
plt.scatter(x=x_test, y=y_test, color='green')
plt.title("SalaryTest v/s Predicted")
plt.xlabel("Yearsof Experience")
plt.ylabel("Salary")
plt.show()

homeprice_df  =  pd.read_csv("C:\\Users\\DSU-CSE513-25\\Downloads\\Housing.csv")

homeprice_df.shape
homeprice_df.shape

homeprice_df.drop(columns=['stories'] , inplace= True)

 homeprice_df.head()

homeprice_df.
drop(columns=['guestroom','basement','hotwaterheating','airconditioning','parking'prefare’]

homeprice_df.shape

homeprice_df.head()

 homeprice_df.info()

 homeprice_df.duplicated().sum()

 homeprice_df['bedrooms'].skew()

 homeprice_df['bedrooms'].fillna(homeprice_df['bedrooms'].mean(), inplace = True)

 homeprice_df.isnull().sum()

 sns.pairplot(data=homeprice_df)
 plt.show()
```

insurance

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

plt.rcParams['figure.figsize']=[19,8]

import warnings

warnings.filterwarnings('ignore')

insurance_df=pd.read_csv("C:/Users/DSU-CSE513-16/Downloads/insurance_data  (1).
  csv")

insurance_df.shape

insurance_df.head()

plt.scatter(data=insurance_df,x='age',y='bought_insurance',marker='*',s=300,color='bla
ck')
plt.grid()
plt.xlabel("age")
plt.ylabel("brought insurance")
plt.show()

sns.regplot(data=insurance_df,x='age',y='bought_insurance',marker='*',color='black')
plt.grid()
plt.show()

sns.regplot(data=insurance_df,x='age',y='bought_insurance',marker='*',logistic=True
,color='black)
plt.grid()
plt.show()

sns.countplot(data=insurance_df,x='bought_insurance',color='black')    plt.title('Tagret
data  Distribution')
plt.show()

x=insurance_df['age'].values.reshape(-1,1)
y=insurance_df['bought_insurance'].values.reshape(-1,1)

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=1)

from sklearn.linear_model import LogisticRegression
```

```
logit_model=LogisticRegression()

logit_model.fit(x_train,y_train)

LogisticRegression()

logit_model.score(x_train,y_train)

logit_model.score(x_test,y_test)

x_test

y_test

y_predict =logit_model.predict(x_test)
y_predict

logit_model.predict_proba(x_test)

logit_model.predict([[36]])

logit_model.predict([[63]])

from sklearn.metrics import confusion_matrix

confusion_matrix=confusion_matrix(y_test,y_predict)
 confusion_matrix

print('test data:',y_test.reshape(-1,))
print("Predicted:",y_predict)

sns.heatmap(confusion_matrix,annot=True,cbar=False,annot_kws={"fontsize":20})
sns.set(font_scale=2)
plt.xlabel("PREDICTED  VALUE",fontsize=20)
plt.ylabel("ACTUAL  VALUE",fontsize=20)
plt.show()
```

Heart

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
heart_df=pd.read_csv('Heart.csv')
```

```
heart_df.shape
```

pd.DataFrame({
'Column': ['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach', 'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
'Description':['age in years',
'(1 = male; 0 = female)',
'chest pain type',
"resting blood pressure (in mm Hg on admission to the hospital)",
'serum cholestoral in mg/dl',
'(fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)',
'resting electrocardiographic results',
'maximum heart rate achieved',
'exercise induced angina (1 = yes; 0 = no)',
'ST depression induced by exercise relative to rest',
'the slope of the peak exercise ST segment',
'number of major vessels (0-3) colored by flourosopy', '0 = normal; 1 = fixed defect; 2 = reversable defect', '1 or 0'
]
})

```
heart_df.info()
```

```
heart_df.describe()
```

```
lt.hist(heart_df['age'], bins=[10, 20, 30,
40,50, 60, 70, 80])
plt.title("Age wise Frequency Distribution")
plt.xlabel('age')
plt.ylabel('count')
```

```
sns.countplot(data=heart_df
, x='cp')
plt.title("Chest  Pain  Type
count")
plt.show()
```

```
sns.countplot(data=hear
t_df, x='sex')
plt.title("Gender wise
count") plt.show()
```

```
sns.countplot(data=heart_df
, x='target')
plt.title("Count Plot for
target variable")
plt.show()

sns.countplot(data=heart_df, x='target',
hue='sex') plt.title("Count Plot for target
variable w.r.t gender") plt.show()

heart_df.duplicated().sum()

heart_df.drop_duplicates(inplace=True)

heart_df.shape

X =
heart_df.drop('targe
t', axis=1) y =
heart_df['target']

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

X = scaler.fit_transform(X)

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=42)

from sklearn.linear_model import
LogisticRegression logistic_model =
LogisticRegression()
logistic_model.fit(X_train, y_train)

logistic_model.score(X_train, y_train)

logistic_model.score(X_test, y_test)
 y_predict = logistic_model.predict(X_test)
from sklearn.metrics import confusion_matrix, accuracy_score,
recall_score,precision_score, f1_score
confusion_matrix(y_test, y_predict)
accuracy_score(y_test, y_predict)
precision_score(y_test, y_predict)
recall_score(y_test, y_predict)

f1_score(y_test, y_predict)
```

svm

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

plt.rcParams['figure.figsize']=[19,8]

import warnings
warnings.filterwarnings('ignore')

import warnings
warnings.filterwarnings('ignore')

from sklearn.datasets import load_iris

iris=load_iris()

dir(iris)

iris_df=pd.DataFrame(data=iris.data,columns=iris.feature_names)
iris_df.head()

iris_df['target'] = iris.target

iris_df.head()

iris_df.info()

ris.target_names

iris_df.duplicated().sum()

iris_df.drop_duplicates(inplace=True)

iris_df.duplicated().sum()

sns.countplot(data=iris_df, x='target')
plt.title("Count of target values")

plt.show()

iris_setosa = iris_df.loc[iris_df['target'] == 0, :] iris_versicolor =
iris_df.loc[iris_df['target'] == 1, :] iris_virginica =
iris_df.loc[iris_df['target'] == 2, :]
```

```
sns.scatterplot(data=iris_setosa, x='petal length (cm)', y='petal width (cm)', ␣
  ↪s=150,)
sns.scatterplot(data=iris_versicolor, x='petal length (cm)', y='petal width ␣
  ↪(cm)', s=150,)
sns.scatterplot(data=iris_virginica, x='petal length (cm)', y='petal width ␣
  ↪(cm)', s=150,)
plt.legend(['Setosa', 'Versicolor', 'Virginica'], loc='lower right') plt.xlabel('Petal Length
(cm)')
plt.ylabel('Petal Width (cm)')
plt.show()

sns.scatterplot(data=iris_setosa, x='sepal length (cm)', y='sepal width (cm)', ␣
  ↪s=150, label='Setosa')
sns.scatterplot(data=iris_versicolor, x='sepal length (cm)', y='sepal width ␣
  ↪(cm)', s=150, label='Versicolor')
sns.scatterplot(data=iris_virginica, x='sepal length (cm)', y='sepal width ␣
  ↪(cm)', s=150, label='Virginica')
plt.xlabel('Sepal Length (cm)')
plt.ylabel('Sepal Width (cm)')
plt.legend(loc='lower right') plt.show()

sns.boxplot(iris_df)
plt.show()

ul =Q3 + 1.5*IQR
ll =Q1-1.5*IQR

iris_df= iris_df[~((iris_df <ll) |(iris_df> ul)).any(axis=1)]

sns.boxplot(iris_df)
plt.show()

X =iris_df.loc[:,:'petal width (cm)'].values y
=iris_df.loc[:, 'target'].values

Y

from sklearn.preprocessing import StandardScalerscaler =
StandardScaler()
X  =  scaler.fit_transform(X)

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=1)

from sklearn.svm import SVC

model = SVC(kernel='linear')
model.fit(X_train, y_train)

model.score(X_train, y_train)
```

```
y_predict = model.predict(X_test)

y_predict

y_test

from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_test, y_predict)

cm
```