DWDM LAB

Commands :-

```
        Sudo apt -get install python3 -venv
         python3 -m venv dwdmlab
          source dwdmlab/bin/activate
            cd dwdmlab
            Source bin/active
              pip install numpy
              pip install sklearn
              pip install pandas.
```

1. #exp1.py:-

```
from sklearn import datasets
from sklearn.datasets import fetch_california_housing
housing = fetch_california_housing(as_frame=TRUE)
    x = housing.data
     print(x)
```

#exp2.py:-

```
from sklearn import datasets
 from sklearn.datasets import fetch_california_housing
  housing_data = fetch_california_housing()
    x = housing_data.data
     print(housing_data.data)
```

```
# dataset·csv

MARKS1, MARKS2, MARKS3, GRADE
   100, 90, 80, "Good"
   70, 60, 70, "Average"
    40, 45, 50, "fair"
```

```
# exp3·py

    import numpy as np
    import pandas as pd
    df = pd.read_csv("dataset.csv")
     print(df)
```

* Handling missing values
* Handling categorial data:

DATA IMPUTATION:-

1. Replacing the missing values with median.
2. Most frequent value.
3. Replace with a fixed value.

4.

```
    import numpy as np
    import pandas as pd
     df = pd.read_csv('dataset.csv')
       print(df)
df['MARKS 1'] = df['MARKS2'].fillna(df['MARKSi'].mean())
    print('After DATA IMPUTATION')
     print(df)
     print(df['MARKS 1'][0])
```

5.
```
  import numpy as np
  import pandas as pd
from sklearn.impute import SimpleImputer
 df = pd.read_csv('dataset.csv')
   print(df)
imputer = SimpleImputer(Strategy= 'mean', missing_values
                                            = np.nan)
imputer = imputer.fit(df[['MARKS1']])
df['MARKS1'] = imputer.transform(df[['MARKS1']])
    print(df)
```

6.
```
  import pandas as pd
  import numpy as np
  import copy
  df_flights = pd.read_csv("flights.csv")
   print(df_flights.info())
cat_df_flights = df_flights.select_dtypes(include =
                                   ["object"]).copy()
Print(cat_df_flights['UniqueCarrier'])
# Label Encoding
cat_df_flights_lc = cat_flights.copy()
cat_df_flights_lc['UniqueCarrier'] = cat_df_flights_lc
                        ['UniqueCarrier'].astype(category)
print(cat_df_flights_lc.dtypes)
cat_df_flights_lc['UniqueCarrier'] = cat_df_flights_lc
                        ['UniqueCarrier'].cat.codes
Print(cat_df_flights_lc)
```

7. cat2.py

```
import pandas as pd
import numpy as np
 import copy
from sklearn.preprocessing import LabelEncoder
df_flights= pd.read_csv("flights.csv")
 lbobj = LabelEncoder()
 Print(df_flights.info())
cat_sklearn_flights = df_flights.copy()
cat_sklearn_flights['UniqueCarrier']= lbobj.fit_
                 transform(df_flights['UniqueCarrier'])
print(cat_sklearn_flights[~~'UniqueCarrier'~~])
```

8. cat3.py

```
import pandas as pd
import numpy as np
 import copy
df_flights = pd.read_csv ("flights.csv")
Cat_onehot= df_flights. copy()
cat_onehot= pd.get_dummies (cat_onehot,
        columns=['UniqueCarrier'],prefix=['UniqueCarrier'])
print(cat_onehot)
```

9.
```
import pandas as pd
import numpy as np
import copy
df_flights= pd.read_csv("flights.csv")
cat_onehot = df_flights.copy()
lb_obj = LabelBinarizer()
lb_results = lb_obj.fit_transform(df_flights
                                   ['UniqueCarrier'])
Print(lb_results).
```

* To install jupyter notebook
  → pip install jupyter notebook.
* jupyter notebook.

```
python3 -m pip install --upgrade pip
python3 -m pip install --upgrade pillow.
```

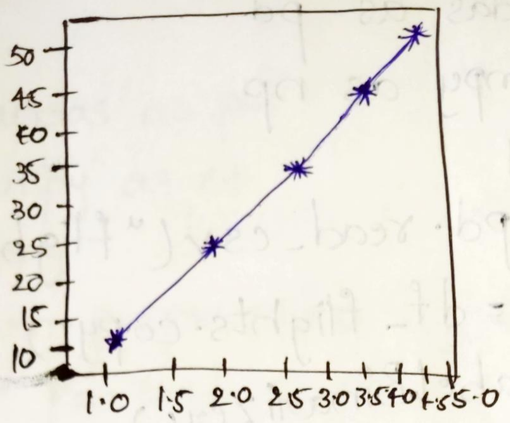
* To install matplotlib use

```
pip install matplotlib
```

Graph1:-

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
x = [1, 2, 3, 4, 5]
y = [10, 20, 30, 40, 50]
plt.plot(x, y; linewidth=1.5, color='purple',
                                    marker='*')
plt.show()
```

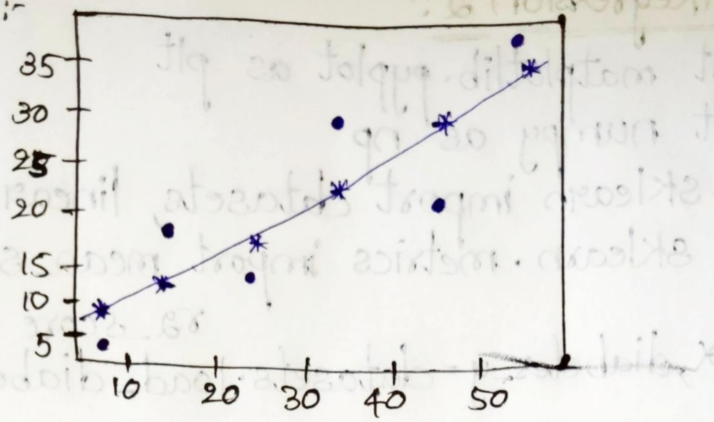
Output:-



Linear Regression:-

```
    import numpy as np
   import matplotlib.pyplot as plt
 from sklearn.linear_model import LinearRegression
x = np.array([5,15,25,35,45,55]).reshape(-1,1)
y = np.array([5,20,14,32,22,38])
  print(x,y)
model = LinearRegression()
model.fit(x, y)
result = model.score(x,y)
print("Score: ", result)
 print("Intercept: ", model.intercept_)
 print("Slope: ", model.coef_)
  y_pred = model.predict(x)
  print('Actual values of y:', y)
  print ('Predicted values of y:', y_pred)
  plt.scatter(x,y, color="black')
  plt.plot(x, y_pred, color='purple', marker='*',
                      markerfacecolor='red')
```

Output:-



Linear Regression of dataset:-

```
   import numpy as np
   import pandas as pd
   import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
n=pd.read_csv('dataset.csv')
x= np.array(n['MARKS1']).reshape(-1,1)
y = np.array(n['MARKS2'])
model= LinearRegression()
model.fit(x,y)
r= model.score(x,y)
print(r)
print("Intercept:", model.intercept_)
print("Slope:", model.coef_)
y_pred= model.predict(x)
print('Actual values of y:', y)
print('Predicted values of y:', y_pred)
 plt.scatter(x,y, color='black')
plt.plot(x, y-pred, color='purple', linewidth=2, marker='*',
                         markerfacecolor='red')
plt.title("LinearRegression")
```

```
#LinearRegression 2:-
   import matplotlib.pyplot as plt
   import numpy as np
   from sklearn import datasets, linear_model
   from sklearn.metrics import mean_squared_error,
                                   r2_score
diabetes_x, diabetes_y = datasets.load_diabetes(return_x_
                                                    y=True)
 diabetes_x = diabetes_x[:, np.newaxis, 2]
  diabetes_x_train = diabetes_x[:-20]
  diabetes_x_test = diabetes_x[-20:]
  diabetes-y-train = diabetes_y[:-20]
  diabetes_y_test = diabetes_y[-20:]
  regr = linear_model.LinearRegression()
  regr.fit(diabetes_x_train, diabetes_y-train)
 diabetes_y_pred = regr.predict(diabetes_x_test)
 print(diabetes_y_pred)
  plt.scatter(diabetes_x_test, diabetes_y_test,
                                   color='black')
plt.plot(diabetes_x_test, diabetes_y_pred, color='blue',
                          linewidth=2)
print("Mean Squared error: %2f" % r2_score(diabetes
              y_test, diabetes_y_pred))
```