Home | Assignment Regression

localhost:8888/notebooks/Assignment%20Regression.ipynb?

jupyter Assignment Regression Last Checkpoint: yesterday

File Edit View Run Kernel Settings Help

Trusted

Markdown

JupyterLab Python 3 (ipykernel)

REGRESSION

[ ]:
```
# The objective of this assignment is to evaluate your understanding of regression techniques in supervised learning by
# applying them to a real world dataset
```

[ ]:
```
# The dataset california housing available in the sklearn library contains information about various features of houses in
# California and their respective median prices
```

[6]:
```
import pandas as pd
from sklearn.datasets import fetch_california_housing
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
```

Loading and Preprocessing

[8]:
```
# Load the California Housing Dataset
Cal_Housing = fetch_california_housing()
```

[10]:
```
# Convert the dataset into a pandas Dataframe
df = pd.DataFrame(Cal_Housing.data, columns = Cal_Housing.feature_names)
df['Target']=Cal_Housing.target
```

[19]:
```
df
```

[19]:

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude	Target
0	8.3252	41.0	6.984127	1.023810	322.0	2.555556	37.88	-122.23	4.526
1	8.3014	21.0	6.238137	0.971880	2401.0	2.109842	37.86	-122.22	3.585
2	7.2574	52.0	8.288136	1.073446	496.0	2.802260	37.85	-122.24	3.521
3	5.6431	52.0	5.817352	1.073059	558.0	2.547945	37.85	-122.25	3.413

29°C Sunny

Search

ENG IN

11:31 05-01-2025

0	8.3252	41.0	6.984127	1.023810	322.0	2.555556	37.88	-122.23	4.526
1	8.3014	21.0	6.238137	0.971880	2401.0	2.109842	37.86	-122.22	3.585
2	7.2574	52.0	8.288136	1.073446	496.0	2.802260	37.85	-122.24	3.521
3	5.6431	52.0	5.817352	1.073059	558.0	2.547945	37.85	-122.25	3.413
4	3.8462	52.0	6.281853	1.081081	565.0	2.181467	37.85	-122.25	3.422
...	...	...	...	...	...	...	...	...	...
20635	1.5603	25.0	5.045455	1.133333	845.0	2.560606	39.48	-121.09	0.781
20636	2.5568	18.0	6.114035	1.315789	356.0	3.122807	39.49	-121.21	0.771
20637	1.7000	17.0	5.205543	1.120092	1007.0	2.325635	39.43	-121.22	0.923
20638	1.8672	18.0	5.329513	1.171920	741.0	2.123209	39.43	-121.32	0.847
20639	2.3886	16.0	5.254717	1.162264	1387.0	2.616981	39.37	-121.24	0.894

20640 rows × 9 columns

[21]:
```
df.head()
```

[21]:

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude	Target
0	8.3252	41.0	6.984127	1.023810	322.0	2.555556	37.88	-122.23	4.526
1	8.3014	21.0	6.238137	0.971880	2401.0	2.109842	37.86	-122.22	3.585
2	7.2574	52.0	8.288136	1.073446	496.0	2.802260	37.85	-122.24	3.521
3	5.6431	52.0	5.817352	1.073059	558.0	2.547945	37.85	-122.25	3.413
4	3.8462	52.0	6.281853	1.081081	565.0	2.181467	37.85	-122.25	3.422

3	5.6431	52.0	5.817352	1.073059	558.0	2.547945	37.85	-122.25	3.413
4	3.8462	52.0	6.281853	1.081081	565.0	2.181467	37.85	-122.25	3.422

[27]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 9 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   MedInc      20640 non-null  float64
 1   HouseAge    20640 non-null  float64
 2   AveRooms    20640 non-null  float64
 3   AveBedrms   20640 non-null  float64
 4   Population  20640 non-null  float64
 5   AveOccup    20640 non-null  float64
 6   Latitude    20640 non-null  float64
 7   Longitude   20640 non-null  float64
 8   Target      20640 non-null  float64
dtypes: float64(9)
memory usage: 1.4 MB
```

[12]:
```
# Handling missing values
print('Missing Values Count:',df.isnull().sum())
```

```
Missing Values Count: MedInc        0
HouseAge      0
AveRooms      0
AveBedrms     0
Population    0
AveOccup      0
Latitude      0
Longitude     0
Target        0
dtype: int64
```

Home | Assignment Regression

localhost:8888/notebooks/Assignment%20Regression.ipynb?

jupyter Assignment Regression Last Checkpoint: yesterday

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

[14]:
```
# Perform Feature Scaling using Standard Scaler
scaler = StandardScaler()
scaled_data = scaler.fit_transform(df[Cal_Housing.feature_names])
scaled_data
```

[14]:
```
array([[ 2.34476576,  0.98214266,  0.62855945, ..., -0.04959654,
         1.05254828, -1.32783522],
       [ 2.33223796, -0.60701891,  0.32704136, ..., -0.09251223,
         1.04318455, -1.32284391],
       [ 1.7826994 ,  1.85618152,  1.15562047, ..., -0.02584253,
         1.03850269, -1.33282653],
       ...,
       [-1.14259331, -0.92485123, -0.09031802, ..., -0.0717345 ,
         1.77823747, -0.8237132 ],
       [-1.05458292, -0.84539315, -0.04021111, ..., -0.09122515,
         1.77823747, -0.87362627],
       [-0.78012947, -1.00430931, -0.07044252, ..., -0.04368215,
         1.75014627, -0.83369581]])
```

[16]:
```
#Split the dataset into training and testing sets
X_train , X_test , Y_train , Y_test = train_test_split(df[Cal_Housing.feature_names],df['Target'],test_size=0.2, random_state = 42)
```

[18]:
```
print('Preprocessed DataFrame:')
print(df.head())
```

```
Preprocessed DataFrame:
   MedInc  HouseAge  AveRooms  AveBedrms  Population  AveOccup  Latitude  \
0  8.3252      41.0  6.984127   1.023810       322.0  2.555556     37.88
1  8.3014      21.0  6.238137   0.971880      2401.0  2.109842     37.86
2  7.2574      52.0  8.288136   1.073446       496.0  2.802260     37.85
3  5.6431      52.0  5.817352   1.073059       558.0  2.547945     37.85
4  3.8462      52.0  6.281853   1.081081       565.0  2.181467     37.85

   Longitude  Target
0    -122.23   4.526
1    -122.22   3.585
2    -122.24   3.521
```

29°C Sunny

Search

ENG IN 11:32 05-01-2025

```
       1.75014627, -0.83369581]])
```

[16]:
```
#Split the dataset into training and testing sets
X_train , X_test , Y_train , Y_test = train_test_split(df[Cal_Housing.feature_names],df['Target'],test_size=0.2, random_state = 42)
```

[18]:
```
print('Preprocessed DataFrame:')
print(df.head())
```

```
Preprocessed DataFrame:
   MedInc  HouseAge  AveRooms  AveBedrms  Population  AveOccup  Latitude  \
0  8.3252      41.0  6.984127   1.023810       322.0  2.555556     37.88
1  8.3014      21.0  6.238137   0.971880      2401.0  2.109842     37.86
2  7.2574      52.0  8.288136   1.073446       496.0  2.802260     37.85
3  5.6431      52.0  5.817352   1.073059       558.0  2.547945     37.85
4  3.8462      52.0  6.281853   1.081081       565.0  2.181467     37.85

   Longitude  Target
0    -122.23   4.526
1    -122.22   3.585
2    -122.24   3.521
3    -122.25   3.413
4    -122.25   3.422
```

[ ]:
```
#Explain the preprocessing steps and justify why they are necessary
```

```
Preprocessing steps Performed
```

```
1. Load the dataset using fetch_california_housing functiom
2. Convert to pandas DataFrame for easier handling and manipulation
3. Checking for missing Values using isnull method
   Handling Missing Values: Although there are no missing values in this dataset,Handling missing values is an essential preprocessing and
   it can properly improve the accuracy and reliability of our result.
4. Feature Scaling by using StandardScaler , This is necessary because some features have different units or scales which affect the performance
   of machine learning models.By scaling these features to have zero mean and unit variance ,we can improve the stability and accuracy of the model.
```

Regression Algorithm Implementation

Linear Regression

[22]:
```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(X_train , Y_train)
y_predict = model.predict(X_test)
```

[26]:
```
y_predict
```

[26]:
```
array([0.71912284, 1.76401657, 2.70965883, ..., 4.46877017, 1.18751119,
       2.00940251])
```

Linear Regression is a linear model that predicts continuous output variable based on one or more input features.It works by learning the coefficients of a linear equation that minimizes the difference between predicted and actual values.
Linear Regression is suitable for this dataset because it's a simple and interpretable model that can capture linear relationships between features and the target variables.It may not perform well if the relationships are linear.

Model Evaluation and Comparison

[45]:
```
from sklearn.metrics import mean_squared_error , mean_absolute_error , r2_score
```

[47]:
```
#Performance of Linear Regression algorithm using metrics
MSE = mean_squared_error (Y_test,y_predict)
MAE = mean_absolute_error (Y_test,y_predict)
r2 = r2_score (Y_test,y_predict)

print("Linear Regression Performance Metrics:")
print(f"Mean Squared Error (MSE): {MSE:.2f}")
print(f"Mean Absolute Error (MAE) : {MAE:.2f}")
print(f"R-Squared Score (r2) : {r2:.2f}")
```

```
Linear Regression Performance Metrics:
```

Home | Assignment Regression

localhost:8888/notebooks/Assignment%20Regression.ipynb?

jupyter **Assignment Regression** Last Checkpoint: yesterday

Trusted

File Edit View Run Kernel Settings Help

Code | JupyterLab | Python 3 (ipykernel)

[22]:
```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(X_train , Y_train)
y_predict = model.predict(X_test)
```

[26]:
```
y_predict
```

[26]:
```
array([0.71912284, 1.76401657, 2.70965883, ..., 4.46877017, 1.18751119,
       2.00940251])
```

Linear Regression is a linear model that predicts continuous output variable based on one or more input features.It works by learning the coefficients of a linear equation that minimizes the difference between predicted and actual values.
Linear Regression is suitable for this dataset because it's a simple and interpretable model that can capture linear relationships between features and the target variables.It may not perform well if the relationships are linear.

Model Evaluation and Comparison

[45]:
```
from sklearn.metrics import mean_squared_error , mean_absolute_error , r2_score
```

[47]:
```
#Performance of Linear Regression algorithm using metrics
MSE = mean_squared_error (Y_test,y_predict)
MAE = mean_absolute_error (Y_test,y_predict)
r2 = r2_score (Y_test,y_predict)

print("Linear Regression Performance Metrics:")
print(f"Mean Squared Error (MSE): {MSE:.2f}")
print(f"Mean Absolute Error (MAE) : {MAE:.2f}")
print(f"R-Squared Score (r2) : {r2:.2f}")
```

```
Linear Regression Performance Metrics:
Mean Squared Error (MSE): 0.56
Mean Absolute Error (MAE) : 0.53
R-Squared Score (r2) : 0.58
```

29°C Sunny | Search | ENG IN | 11:33 05-01-2025