balaji-creator / BLENDED_LEARNING_EXP4

Code Pull requests Actions Projects Wiki Security Insights Settings

Watch 0

BSD-3-Clause license

0 stars 148 forks 0 watching Branches Activity Tags

Public repository · Forked from AkilaMohan/BLENDED_LEARNING_Implementation-of-Stochastic-Gradient-Descent-SGD-Regressor

1 Branch 0 Tags Go to file t Go to file Add file Code

This branch is 1 commit ahead of AkilaMohan/BLENDED_LEARNING_Implementation-of-Stochastic-Gradient-Descent-SGD-Regressor:main .

Contribute Sync fork

balaji-creator success 4f9343d · 1 minute ago

LICENSE	Initial commit	2 years ago
README.md	success	1 minute ago
Screenshot 2026-02-19 091...	success	1 minute ago
Screenshot 2026-02-19 091...	success	1 minute ago
Screenshot 2026-02-19 091...	success	1 minute ago
Screenshot 2026-02-19 091...	success	1 minute ago
Screenshot 2026-02-19 091...	success	1 minute ago

README License

BLENDED_LEARNING

Implementation-of-Stochastic-Gradient-Descent-SGD-Regressor

AIM:

To write a program to implement Stochastic Gradient Descent (SGD) Regressor for linear regression and evaluate its performance.

Equipments Required:

1. Hardware – PCs
2. Anaconda – Python 3.7 Installation / Jupyter notebook

Algorithm

1. Import the necessary libraries.
2. Load the dataset.
3. Preprocess the data (handle missing values, encode categorical variables).
4. Split the data into features (X) and target (y).
5. Divide the data into training and testing sets.
6. Create an SGD Regressor model.
7. Fit the model on the training data.
8. Evaluate the model performance.
9. Make predictions and visualize the results.

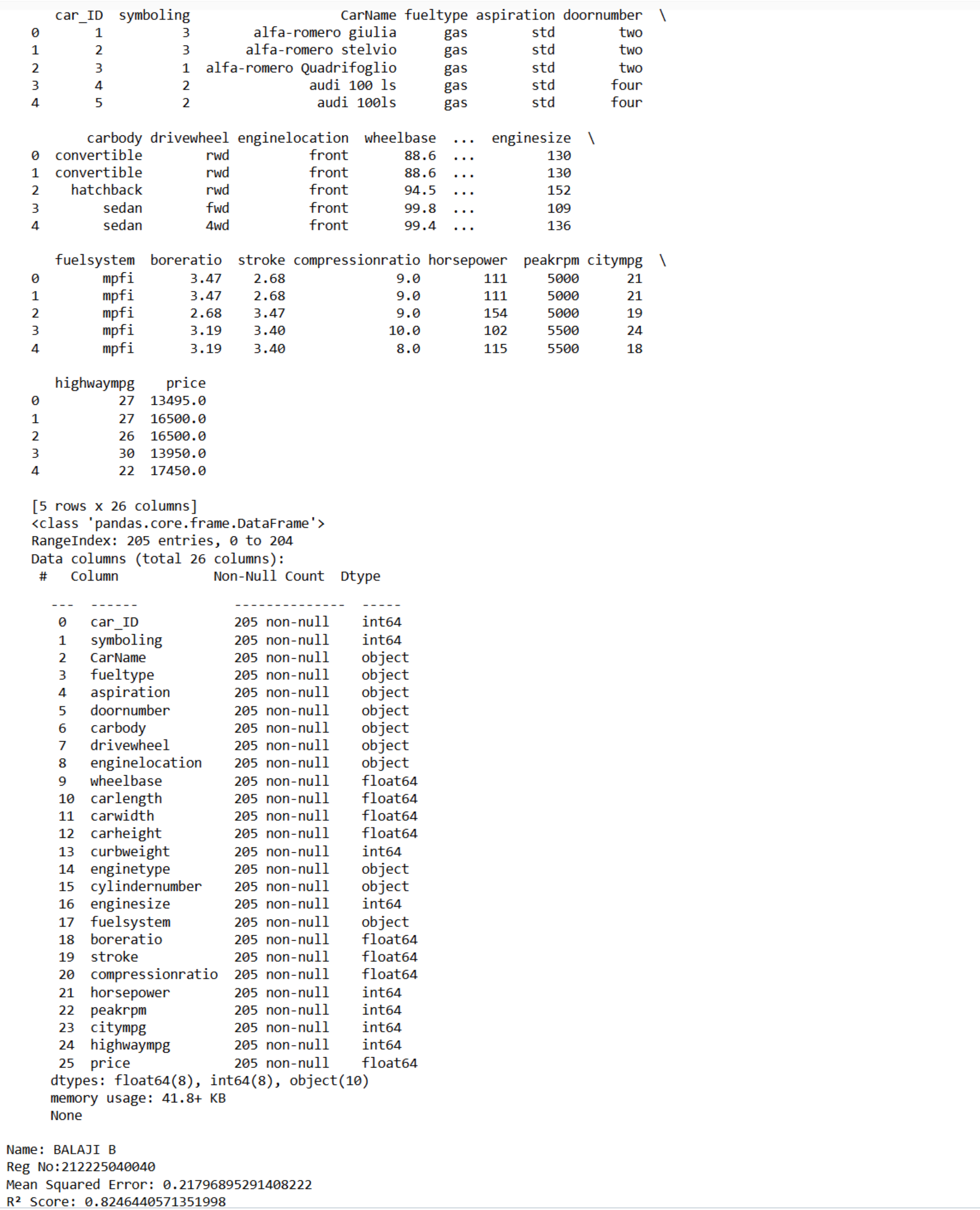
Program:

```
/*
Program to implement SGD Regressor for linear regression.
Developed by: Balaji B
RegisterNumber:  212225040040
*/
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import SGDRegressor
from sklearn.metrics import mean_squared_error,r2_score
from sklearn.preprocessing import StandardScaler

data=pd.read_csv("CarPrice_Assignment.csv")
print(data.head())
print(data.info())

data = data.drop(['CarName', 'car_ID'], axis=1)
data = pd.get_dummies(data, drop_first=True)

x = data.drop('price', axis=1)
y = data['price']

scaler = StandardScaler()
x = scaler.fit_transform(x)
y = scaler.fit_transform(np.array(y).reshape(-1, 1))

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)

sgd_model=SGDRegressor(max_iter=1000,tol=1e-3)

sgd_model.fit(x_train, y_train)
```

```
y_pred=sgd_model.predict(x_test)

mse=mean_squared_error(y_test,y_pred)
r2 = r2_score(y_test, y_pred)

print("Name: BALAJI B")
print("Reg No:212225040040")
print("Mean Squared Error:", mse)
print("R² Score:", r2)

print("Model Coefficients:")
print("Coefficients:", sgd_model.coef_)
print("Intercept:", sgd_model.intercept_)

plt.scatter(y_test, y_pred)
plt.xlabel("Actual Prices")
plt.ylabel("Predicted Prices")
plt.title("Actual vs Predicted Prices using SGD Regressor")
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='red')
plt.show()
```

Output:



```
      car_ID  symboling                   CarName fueltype aspiration doornumber  \
   0       1          3        alfa-romero giulia      gas        std        two
   1       2          3       alfa-romero stelvio      gas        std        two
   2       3          1  alfa-romero Quadrifoglio      gas        std        two
   3       4          2               audi 100 ls      gas        std       four
   4       5          2                audi 100ls      gas        std       four

          carbody drivewheel enginelocation  wheelbase  ...  enginesize  \
   0  convertible        rwd          front       88.6  ...         130
   1  convertible        rwd          front       88.6  ...         130
   2    hatchback        rwd          front       94.5  ...         152
   3        sedan        fwd          front       99.8  ...         109
   4        sedan        4wd          front       99.4  ...         136

      fuelsystem  boreratio  stroke compressionratio horsepower  peakrpm citympg  \
   0        mpfi       3.47    2.68              9.0        111     5000      21
   1        mpfi       3.47    2.68              9.0        111     5000      21
   2        mpfi       2.68    3.47              9.0        154     5000      19
   3        mpfi       3.19    3.40             10.0        102     5500      24
   4        mpfi       3.19    3.40              8.0        115     5500      18

      highwaympg    price
   0          27  13495.0
   1          27  16500.0
   2          26  16500.0
   3          30  13950.0
   4          22  17450.0

   [5 rows x 26 columns]
   <class 'pandas.core.frame.DataFrame'>
   RangeIndex: 205 entries, 0 to 204
   Data columns (total 26 columns):
    #   Column            Non-Null Count  Dtype

     ---  ------            --------------  -----
      0   car_ID            205 non-null    int64
      1   symboling         205 non-null    int64
      2   CarName           205 non-null    object
      3   fueltype          205 non-null    object
      4   aspiration        205 non-null    object
      5   doornumber        205 non-null    object
      6   carbody           205 non-null    object
      7   drivewheel        205 non-null    object
      8   enginelocation    205 non-null    object
      9   wheelbase         205 non-null    float64
      10  carlength         205 non-null    float64
      11  carwidth          205 non-null    float64
      12  carheight         205 non-null    float64
      13  curbweight        205 non-null    int64
      14  enginetype        205 non-null    object
      15  cylindernumber    205 non-null    object
      16  enginesize        205 non-null    int64
      17  fuelsystem        205 non-null    object
      18  boreratio         205 non-null    float64
      19  stroke            205 non-null    float64
      20  compressionratio  205 non-null    float64
      21  horsepower        205 non-null    int64
      22  peakrpm           205 non-null    int64
      23  citympg           205 non-null    int64
      24  highwaympg        205 non-null    int64
      25  price             205 non-null    float64
     dtypes: float64(8), int64(8), object(10)
     memory usage: 41.8+ KB
     None

Name: BALAJI B
Reg No:212225040040
Mean Squared Error: 0.21796895291408222
R² Score: 0.8246440571351998
```

Releases

No releases published
Create a new release

Packages

No packages published

Publish your first package