lab8b

September 2, 2024

[135]:
```
import numpy as np# Summary of results
results = {
    'OLS': ols_mse,
    'Ridge': ridge_mse,
    'Lasso': lasso_mse
}

best_model = min(results, key=results.get)
print(f"The model with the highest accuracy (lowest MSE) is: {best_model}")

import pandas as pd
```

```
The model with the highest accuracy (lowest MSE) is: Lasso
```

[136]:
```
dataset = pd.read_csv("Hitters.csv")
df = pd.DataFrame(dataset)
df
```

[136]:
```
             Unnamed: 0  AtBat  Hits  HmRun  Runs  RBI  Walks  Years  CAtBat  \
0       -Andy Allanson    293    66      1    30   29     14      1     293
1          -Alan Ashby    315    81      7    24   38     39     14    3449
2         -Alvin Davis    479   130     18    66   72     76      3    1624
3        -Andre Dawson    496   141     20    65   78     37     11    5628
4    -Andres Galarraga    321    87     10    39   42     30      2     396
..                 …     …     …    …   …    …    …     …       …
317      -Willie McGee    497   127      7    65   48     37      5    2703
318   -Willie Randolph    492   136      5    76   50     94     12    5511
319     -Wayne Tolleson    475   126      3    61   43     52      6    1700
320     -Willie Upshaw    573   144      9    85   60     78      8    3198
321     -Willie Wilson    631   170      9    77   44     31     11    4908

     CHits  …  CRuns  CRBI  CWalks  League Division  PutOuts  Assists  \
0       66  …     30    29      14       A        E      446       33
1      835  …    321   414     375       N        W      632       43
2      457  …    224   266     263       A        W      880       82
3     1575  …    828   838     354       N        E      200       11
4      101  …     48    46      33       N        E      805       40
..     …   …    …    …     …       …       …      …       …
```

```
317   806  …   379   311     138        N        E     325        9
318  1511  …   897   451     875        A        E     313      381
319   433  …   217    93     146        A        W      37      113
320   857  …   470   420     332        A        E    1314      131
321  1457  …   775   357     249        A        W     408        4

     Errors  Salary  NewLeague
0        20     NaN          A
1        10   475.0          N
2        14   480.0          A
3         3   500.0          N
4         4    91.5          N
..      …       …        …
317       3   700.0          N
318      20   875.0          A
319       7   385.0          A
320      12   960.0          A
321       3  1000.0          A

[322 rows x 21 columns]
```

[137]:
```
df.dtypes
```

[137]:
```
Unnamed: 0     object
AtBat           int64
Hits            int64
HmRun           int64
Runs            int64
RBI             int64
Walks           int64
Years           int64
CAtBat          int64
CHits           int64
CHmRun          int64
CRuns           int64
CRBI            int64
CWalks          int64
League         object
Division       object
PutOuts         int64
Assists         int64
Errors          int64
Salary        float64
NewLeague      object
dtype: object
```

[138]:
```
df.pop('Unnamed: 0')
```

[138]:
```
0          -Andy Allanson
1             -Alan Ashby
2            -Alvin Davis
3           -Andre Dawson
4       -Andres Galarraga
              …
317         -Willie McGee
318      -Willie Randolph
319       -Wayne Tolleson
320        -Willie Upshaw
321        -Willie Wilson
Name: Unnamed: 0, Length: 322, dtype: object
```

[139]:
```
df.head()
```

[139]:
```
   AtBat  Hits  HmRun  Runs  RBI  Walks  Years  CAtBat  CHits  CHmRun  CRuns  \
0    293    66      1    30   29     14      1     293     66       1     30
1    315    81      7    24   38     39     14    3449    835      69    321
2    479   130     18    66   72     76      3    1624    457      63    224
3    496   141     20    65   78     37     11    5628   1575     225    828
4    321    87     10    39   42     30      2     396    101      12     48

   CRBI  CWalks League Division  PutOuts  Assists  Errors  Salary NewLeague
0    29      14      A        E      446       33      20     NaN         A
1   414     375      N        W      632       43      10   475.0         N
2   266     263      A        W      880       82      14   480.0         A
3   838     354      N        E      200       11       3   500.0         N
4    46      33      N        E      805       40       4    91.5         N
```

[140]:
```
df.describe()
```

[140]:
```
            AtBat        Hits       HmRun        Runs         RBI       Walks  \
count  322.000000  322.000000  322.000000  322.000000  322.000000  322.000000
mean   380.928571  101.024845   10.770186   50.909938   48.027950   38.742236
std    153.404981   46.454741    8.709037   26.024095   26.166895   21.639327
min     16.000000    1.000000    0.000000    0.000000    0.000000    0.000000
25%    255.250000   64.000000    4.000000   30.250000   28.000000   22.000000
50%    379.500000   96.000000    8.000000   48.000000   44.000000   35.000000
75%    512.000000  137.000000   16.000000   69.000000   64.750000   53.000000
max    687.000000  238.000000   40.000000  130.000000  121.000000  105.000000

            Years       CAtBat        CHits      CHmRun       CRuns  \
count  322.000000    322.00000   322.000000  322.000000  322.000000
mean     7.444099   2648.68323   717.571429   69.490683  358.795031
std      4.926087   2324.20587   654.472627   86.266061  334.105886
min      1.000000     19.00000     4.000000    0.000000    1.000000
25%      4.000000    816.75000   209.000000   14.000000  100.250000
```

```
50%        6.000000   1928.00000   508.000000   37.500000   247.000000
75%       11.000000   3924.25000  1059.250000   90.000000   526.250000
max       24.000000  14053.00000  4256.000000  548.000000  2165.000000

              CRBI       CWalks      PutOuts     Assists      Errors  \
count   322.000000   322.000000   322.000000  322.000000  322.000000
mean    330.118012   260.239130   288.937888  106.913043    8.040373
std     333.219617   267.058085   280.704614  136.854876    6.368359
min       0.000000     0.000000     0.000000    0.000000    0.000000
25%      88.750000    67.250000   109.250000    7.000000    3.000000
50%     220.500000   170.500000   212.000000   39.500000    6.000000
75%     426.250000   339.250000   325.000000  166.000000   11.000000
max    1659.000000  1566.000000  1378.000000  492.000000   32.000000

            Salary
count   263.000000
mean    535.925882
std     451.118681
min      67.500000
25%     190.000000
50%     425.000000
75%     750.000000
max    2460.000000
```

[141]:
```
df.isnull().sum()
```

[141]:
```
AtBat         0
Hits          0
HmRun         0
Runs          0
RBI           0
Walks         0
Years         0
CAtBat        0
CHits         0
CHmRun        0
CRuns         0
CRBI          0
CWalks        0
League        0
Division      0
PutOuts       0
Assists       0
Errors        0
Salary       59
NewLeague     0
dtype: int64
```

[142]:
```
df['Salary'].fillna(value=df['Salary'].mean(),inplace=True)
```

[143]:
```
from scipy import stats
import numpy as np

# Calculate Z-scores
z_scores = np.abs(stats.zscore(df.select_dtypes(include=np.number)))

threshold = 3
outliers = (z_scores > threshold)

print(np.where(outliers)[0])

df = df.drop(index=np.where(outliers)[0])
```

```
[ 30  32  73  73  80  82  82  84  96 100 112 113 114 121 136 163 163 179
 179 180 189 217 229 235 236 236 236 236 236 248 249 249 249 249 260 272
 274 276 278 292 302 302 302 302 302 306 310 313 315 320]
```

[144]:
```
df.dtypes
```

[144]:
```
AtBat          int64
Hits           int64
HmRun          int64
Runs           int64
RBI            int64
Walks          int64
Years          int64
CAtBat         int64
CHits          int64
CHmRun         int64
CRuns          int64
CRBI           int64
CWalks         int64
League        object
Division      object
PutOuts        int64
Assists        int64
Errors         int64
Salary       float64
NewLeague     object
dtype: object
```

[145]:
```
# Convert categorical variables into dummy/indicator variables
df = pd.get_dummies(df, columns=['Division', 'League',␣
 ↪'NewLeague'],drop_first=True)
```

```
[146]: df
```

```
[146]:      AtBat  Hits  HmRun  Runs  RBI  Walks  Years  CAtBat  CHits  CHmRun  \
       0      293    66      1    30   29     14      1     293     66       1
       1      315    81      7    24   38     39     14    3449    835      69
       2      479   130     18    66   72     76      3    1624    457      63
       3      496   141     20    65   78     37     11    5628   1575     225
       4      321    87     10    39   42     30      2     396    101      12
       ..     …    …     …    …  …     …     …     …     …     …
       316    221    53      2    21   23     22      8    1063    283      15
       317    497   127      7    65   48     37      5    2703    806      32
       318    492   136      5    76   50     94     12    5511   1511      39
       319    475   126      3    61   43     52      6    1700    433       7
       321    631   170      9    77   44     31     11    4908   1457      30

            CRuns  CRBI  CWalks  PutOuts  Assists  Errors       Salary  Division_W  \
       0       30    29      14      446       33      20   535.925882       False
       1      321   414     375      632       43      10   475.000000        True
       2      224   266     263      880       82      14   480.000000        True
       3      828   838     354      200       11       3   500.000000       False
       4       48    46      33      805       40       4    91.500000       False
       ..     …    …     …      …      …         …       …
       316    107   124     106      325       58       6   535.925882       False
       317    379   311     138      325        9       3   700.000000       False
       318    897   451     875      313      381      20   875.000000       False
       319    217    93     146       37      113       7   385.000000        True
       321    775   357     249      408        4       3  1000.000000        True

            League_N  NewLeague_N
       0       False        False
       1        True         True
       2       False        False
       3        True         True
       4        True         True
       ..        …           …
       316      True         True
       317      True         True
       318     False        False
       319     False        False
       321     False        False

       [287 rows x 20 columns]
```

```
[147]: from sklearn.model_selection import train_test_split
       from sklearn.linear_model import LinearRegression, Ridge, Lasso
       from sklearn.metrics import mean_squared_error
```

```
# Prepare features and target variable
X = df.drop('Salary', axis=1)  # Assuming 'Salary' is the target variable
y = df['Salary']

# Split data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

# OLS Regression
ols_model = LinearRegression()
ols_model.fit(X_train, y_train)
ols_predictions = ols_model.predict(X_test)
ols_mse = mean_squared_error(y_test, ols_predictions)

# Ridge Regression
ridge_model = Ridge(alpha=1.0)
ridge_model.fit(X_train, y_train)
ridge_predictions = ridge_model.predict(X_test)
ridge_mse = mean_squared_error(y_test, ridge_predictions)

# Lasso Regression
lasso_model = Lasso(alpha=0.1)
lasso_model.fit(X_train, y_train)
lasso_predictions = lasso_model.predict(X_test)
lasso_mse = mean_squared_error(y_test, lasso_predictions)

print(f"OLS Mean Squared Error: {ols_mse}")
print(f"Ridge Mean Squared Error: {ridge_mse}")
print(f"Lasso Mean Squared Error: {lasso_mse}")
```

```
OLS Mean Squared Error: 59666.2269633068
Ridge Mean Squared Error: 59599.27851989654
Lasso Mean Squared Error: 59594.05119739496
```

```
/opt/anaconda3/lib/python3.11/site-
packages/sklearn/linear_model/_coordinate_descent.py:631: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 5.386e+06, tolerance: 2.068e+03
  model = cd_fast.enet_coordinate_descent(
```

[148]:
```
# Summary of results
results = {
    'OLS': ols_mse,
    'Ridge': ridge_mse,
    'Lasso': lasso_mse
}
```

```python
best_model = min(results, key=results.get)
print(f"The model with the highest accuracy (lowest MSE) is: {best_model}")
```

```
The model with the highest accuracy (lowest MSE) is: Lasso
```

[ ]:

```python

```