5

March 1, 2020

1 Assignment 5

1. Choose a regression dataset (bikeshare is allowed), perform a test/train split, and build a regression model (just like in assingnment 3), and calculate the
   - Training Error (MSE, MAE)
   - Testing Error (MSE, MAE)
2. Choose a classification dataset (not the adult.data set, The UCI repository has many datasets as well as Kaggle), perform test/train split and create a classification model (your choice but DecisionTree is fine). Calculate
   - Accuracy
   - Confusion Matrix
   - Classifcation Report
3. (Bonus) See if you can improve the classification model's performance with any tricks you can think of (modify features, remove features, polynomial features)

[114]:

```
import pandas as pd
import numpy as np
import matplotlib.pylab as plt

from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.model_selection import train_test_split
from sklearn.linear_model import Ridge
from sklearn.metrics import (accuracy_score,
                             classification_report,
                             confusion_matrix, auc, roc_curve,␣
 ↪mean_squared_error)
from sklearn.tree import DecisionTreeClassifier
```

2 Regression Data Set

[30]:

```
credit = pd.read_csv('../data/Credit.csv')
credit
```

[30]:

```
     Unnamed: 0   Income  Limit  Rating  Cards  Age  Education  Gender  \
0             1   14.891   3606     283      2   34         11    Male
1             2  106.025   6645     483      3   82         15  Female
```

```
2             3  104.593  7075     514       4  71          11    Male
3             4  148.924  9504     681       3  36          11  Female
4             5   55.882  4897     357       2  68          16    Male
..           …      …    …     …    …  …        …     …
395         396   12.096  4100     307       3  32          13    Male
396         397   13.364  3838     296       5  65          17    Male
397         398   57.872  4171     321       5  67          12  Female
398         399   37.728  2525     192       1  44          13    Male
399         400   18.701  5524     415       5  64           7  Female

    Student Married          Ethnicity  Balance
0        No     Yes          Caucasian      333
1       Yes     Yes              Asian      903
2        No      No              Asian      580
3        No      No              Asian      964
4        No     Yes          Caucasian      331
..       …     …               …      …
395      No     Yes          Caucasian      560
396      No      No   African American      480
397      No     Yes          Caucasian      138
398      No     Yes          Caucasian        0
399      No      No              Asian      966

[400 rows x 12 columns]
```

3 Classification Data Set

[78]:

```
cancer_data = pd.read_csv('../data/breast-cancer-wisconsin.data')
```

[79]:

```
cancer_data.columns = ['id',
                'clump_thickness',
                'cell_size',
                'cell_shape',
                'adhesion',
                'epithelial_cell_size',
                'bare_nucleoli',
                'bland_chromatin',
                'normal_nucleoli',
                'mitosis',
                'cell_class']
```

[80]:

```
cancer_data
```

[80]:

```
          id  clump_thickness  cell_size  cell_shape  adhesion  \
0    1002945                5          4           4         5
1    1015425                3          1           1         1
```

```
2    1016277                 6          8           8         1
3    1017023                 4          1           1         3
4    1017122                 8         10          10         8
..       …              …         …         …       …
693   776715                 3          1           1         1
694   841769                 2          1           1         1
695   888820                 5         10          10         3
696   897471                 4          8           6         4
697   897471                 4          8           8         5

     epithelial_cell_size bare_nucleoli   bland_chromatin  normal_nucleoli  \
0                       7            10                 3                2
1                       2             2                 3                1
2                       3             4                 3                7
3                       2             1                 3                1
4                       7            10                 9                7
..                    …           …              …              …
693                     3             2                 1                1
694                     2             1                 1                1
695                     7             3                 8               10
696                     3             4                10                6
697                     4             5                10                4

     mitosis  cell_class
0          1           2
1          1           2
2          1           2
3          1           2
4          1           4
..       …         …
693        1           2
694        1           2
695        2           4
696        1           4
697        1           4

[698 rows x 11 columns]
```

4 Question 1

Choose a regression dataset (bikeshare is allowed), perform a test/train split, and build a regression model (just like in assingnment 3), and calculate the + Training Error (MSE, MAE) + Testing Error (MSE, MAE)

[101]:
```
# separate groups from the data and only keep numerical data.
numerical_cats = credit.select_dtypes(include=['int64','float64'])
```

```
x_train, x_test, y_train, y_test = train_test_split(numerical_cats.
 ↪drop(['Rating'], axis=1),numerical_cats.Rating, test_size=.20)
x_train,y_train
```

```
[101]: (     Unnamed: 0   Income  Limit  Cards  Age  Education  Balance
        368          369   89.000   5759      3   37          6      345
        23            24   64.027   5179      5   48          8      411
        336          337   32.856   5884      4   68         13      926
        201          202   73.914   7333      6   67         15     1048
        36            37   62.413   6457      2   71         11      762
        ..           …        …     …      …   …          …       …
        322          323   27.229   3484      6   51         11      265
        378          379   19.349   4941      1   33         19      717
        339          340  149.316  10278      1   80         16     1107
        53            54   16.304   5466      4   66         10      957
        112          113   46.007   6637      4   42         14     1046

        [320 rows x 7 columns], 368    440
        23     398
        336    438
        201    529
        36     455
              …
        322    282
        378    366
        339    707
        53     413
        112    491
        Name: Rating, Length: 320, dtype: int64)
```

```
[111]: model = Ridge(alpha=.8)
       training_set = model.fit(x_train, y_train)
       training_set.coef_, training_set.intercept_

       training_test = model.predict(x_train)
       acc_score_train = accuracy_score(y_train, training_test.astype(int))

       print(acc_score_train,mean_squared_error(y_train, training_test.astype(int)))
```

```
0.04375 103.609375
```

```
[103]: # test the model to the testing data.
       prediction = model.predict(x_test)
       np.array(y_test),prediction.astype(int)

       # testing error and metrics.
       acc_score = accuracy_score(y_test,prediction.astype(int))
```

```
print(acc_score)

mean_squared_error(y_test, prediction.astype(int))
```

```
0.025
```

[103]:
```
98.475
```

5 Question 2

Choose a classification dataset (not the adult.data set, The UCI repository has many datasets as well as Kaggle), perform test/train split and create a classification model (your choice but DecisionTree is fine). Calculate + Accuracy + Confusion Matrix + Classifcation Report

[126]:
```
# Clean up data.
cancer_data.isin(['?']).any()
index_name = cancer_data[cancer_data['bare_nucleoli'] == '?'].index
cancer_data.drop(index_name, inplace=True)
```

[127]:
```
x_train, x_test, y_train, y_test = train_test_split(cancer_data.
 ↪drop(['cell_class'], axis=1),cancer_data.cell_class, test_size=.20)
```

[128]:
```
# decision tree classification of cancer cells.
model = DecisionTreeClassifier(criterion='entropy')
model.fit(x_train, y_train)
```

[128]:
```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='entropy',
                       max_depth=None, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=None, splitter='best')
```

[132]:
```
test_predictions = model.predict(x_test)
acc = accuracy_score(y_test, test_predictions)
conf_matrx = confusion_matrix(y_test, test_predictions)
report = classification_report(y_test, test_predictions)
print(acc)
print(conf_matrx)
print(report)
```

```
0.9562043795620438
[[89  4]
 [ 2 42]]
              precision    recall  f1-score   support

           2       0.98      0.96      0.97        93
           4       0.91      0.95      0.93        44

    accuracy                           0.96       137
   macro avg       0.95      0.96      0.95       137
weighted avg       0.96      0.96      0.96       137
```

6 Bonus

(Bonus) See if you can improve the classification model's performance with any tricks you can think of (modify features, remove features, polynomial features)

[156]:

```
# modify the alpha value to see an increase in score.
credit.Gender.replace('Male', 1)
credit.Gender.replace('Female',0)
credit.Student.replace('Yes',1)
credit.Student.replace('No',0)
credit.Married.replace('Yes',1)
credit.Married.replace('No',0)

numerical_cats = credit.select_dtypes(include=['int64','float64'])
x_train, x_test, y_train, y_test = train_test_split(numerical_cats.
 ↪drop(['Rating'], axis=1),numerical_cats.Rating, test_size=.20)
x_train,y_train

model = Ridge(alpha=.35)
training_set = model.fit(x_train, y_train)
training_set.coef_, training_set.intercept_

training_test = model.predict(x_train)
acc_score_train = accuracy_score(y_train, training_test.astype(int))

print(acc_score_train,mean_squared_error(y_train, training_test.astype(int)))

# test the model to the testing data.
prediction = model.predict(x_test)
np.array(y_test),prediction.astype(int)

# testing error and metrics.
acc_score = accuracy_score(y_test,prediction.astype(int))
print(acc_score)

mean_squared_error(y_test, prediction.astype(int))
```

```
0.05 103.775
0.0125
```

```
[156]: 98.125
```

```
[ ]:
```