

# Tutorial 8A

## Cluster Analysis

```
Tutorial 8A DataMining.py
Project
  Tutorial 8A C:\Users\adhir\Documents\MIS502
    DataMining.py
    ESSData.csv
    single_family_home_values.csv
    Tutorial Eight-A.pdf
    Tutorial 8A.docx
    -Tutorial 8A.docx
  External Libraries
  Scratches and Consoles

DataMining.py
5 import seaborn as sns
6 import matplotlib.pyplot as plt
7 from sklearn.cluster import KMeans
8 desired_width = 400
9 pd.set_option('display.width', desired_width) # sets run screen width to 400
10 pd.set_option('display.max_columns', 20) # sets run screen column display to 20
11 df = pd.read_csv(r'single_family_home_values.csv') # reads Zillow file
12 df2 = df.drop('estimated_value', axis = 1) # any data frame column can be dropped
13 df3 = df2[['bedrooms', 'bathrooms', 'rooms', 'squareFootage', 'lotSize', 'yearBuilt', 'priorSaleAmount']] # reduced df
14 df3.fillna(0, inplace=True) # replaces the NaN in priorSaleAmount with 0 -- may get a warning, but better than NaN
15 print(df3.head(2)) # prints top two rows of df3
16 k_groups = KMeans(n_clusters=5, random_state=0).fit(df3) # computes data set into 5 distinguishable groups
```

```
Run: DataMining
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
df3.fillna(0, inplace=True) # replaces the NaN in priorSaleAmount with 0 -- may get a warning, but better than NaN
C:\Users\adhir\anaconda\envs\Tutorial 8A\lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value of 'n_init' will change from 10 to 'auto'
warnings.warn(
  bedrooms  bathrooms  rooms  squareFootage  lotSize  yearBuilt  priorSaleAmount
0          3          2.0      6          1378      9968      2003.0      165700.0
1          2          2.0      6          1653      6970      2004.0           0.0
C:\Users\adhir\Documents\MIS502\Tutorial 8A\DataMining.py:23: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
df3['cluster'] = k_groups.labels_ # add a new column to df3 called 'cluster', the k-group #
[1 1 1 ... 0 4 0]
15000 (15000, 7)
```

```
[1 1 1 ... 0 4 0]
15000 (15000, 7)
[[3.31226296e+00 3.83944374e+00 8.42730721e+00 2.69720607e+03
  6.97174968e+03 1.94200506e+03 7.43586930e+05]
 [2.64078392e+00 1.93525180e+00 5.86293724e+00 1.39300918e+03
  5.94409712e+03 1.93060060e+03 3.93563157e+04]
 [3.00000000e+00 4.50000000e+00 9.00000000e+00 3.74800000e+03
  8.59750000e+03 1.99800000e+03 1.37500550e+07]
 [3.73118280e+00 5.64516129e+00 1.04408602e+01 4.51996774e+03
  1.30122688e+04 1.96766667e+03 2.37729552e+06]
 [2.70373430e+00 2.27247191e+00 6.20290813e+00 1.47484848e+03
  5.39461203e+03 1.92551404e+03 2.93157062e+05]]
[[3.31226296e+00 3.83944374e+00 8.42730721e+00 2.69720607e+03
  6.97174968e+03 1.94200506e+03 7.43586930e+05]
```

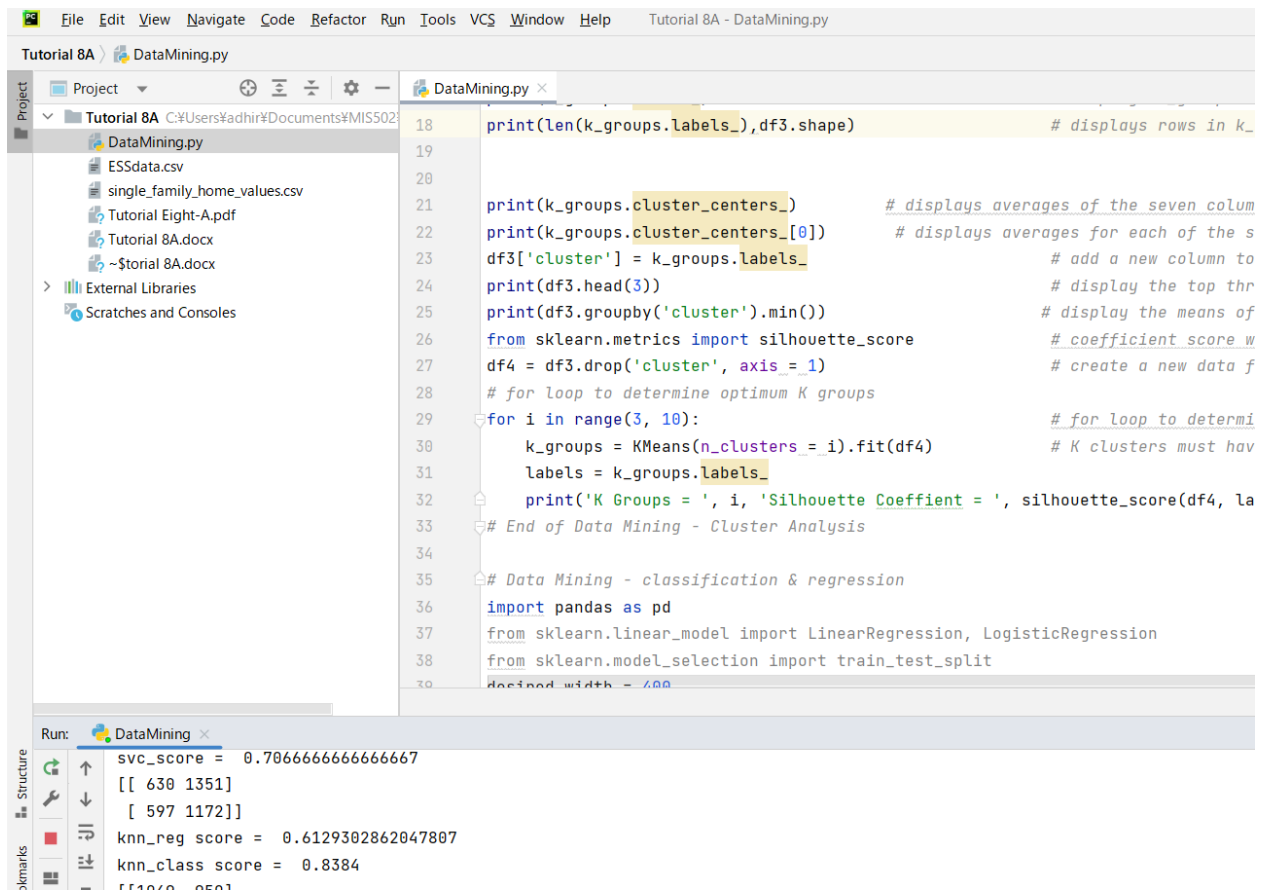
## Min cluster:

```
24 print(df3.head(3)) # display the top three rows of data frame df3
25 print(df3.groupby('cluster').min()) # display the means of the seven columns of data frame df3
26 from sklearn.metrics import silhouette_score # coefficient score where higher is better, 0 = cluster overlap

Run: DataMining
  bedrooms  bathrooms  rooms  squareFootage  lotSize  yearBuilt  priorSaleAmount  cluster
0          3          2.0      6          1378      9968      2003.0      165700.0      1
1          2          2.0      6          1653      6970      2004.0           0.0      1
2          3          1.0      0          1882      23875      1917.0           0.0      1
  bedrooms  bathrooms  rooms  squareFootage  lotSize  yearBuilt  priorSaleAmount
cluster
0          0          0.0      0           662      1626      1879.0      519000.0
1          0          0.0      0           350       278           0.0           0.0
2          3          4.0      8          3355      3916      1994.0      11500110.0
3          1          1.0      4           772      4078      1887.0      1580000.0
4          1          0.0      0           517      1175      1874.0      166331.0
C:\Users\adhir\Documents\MIS502\Tutorial 8A\DataMining.py:23: SettingWithCopyWarning:
```

## Regression and Classification

Knn :



```
File Edit View Navigate Code Refactor Run Tools VCS Window Help Tutorial 8A - DataMining.py

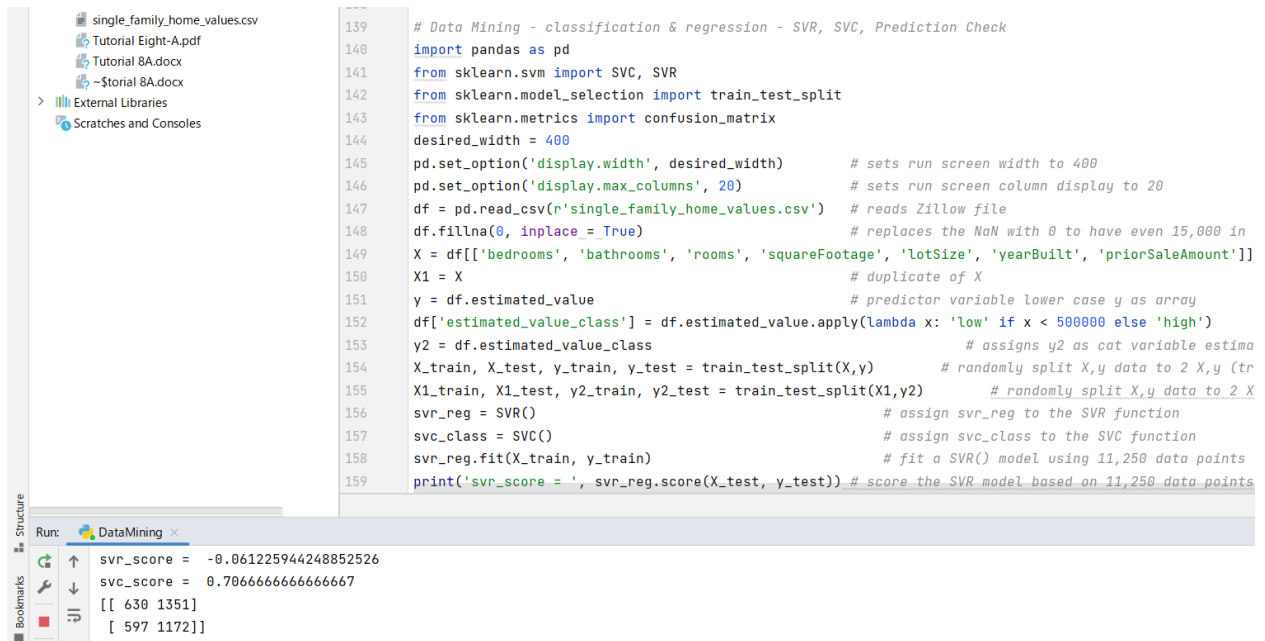
Tutorial 8A \ DataMining.py

Project
  Tutorial 8A C:\Users\adhir\Documents\MIS502
    DataMining.py
    ESSdata.csv
    single_family_home_values.csv
    Tutorial Eight-A.pdf
    Tutorial 8A.docx
    ~$torial 8A.docx
  External Libraries
  Scratches and Consoles

18 print(len(k_groups.labels_),df3.shape) # displays rows in k_
19
20
21 print(k_groups.cluster_centers_) # displays averages of the seven colum
22 print(k_groups.cluster_centers_[0]) # displays averages for each of the s
23 df3['cluster'] = k_groups.labels_ # add a new column to
24 print(df3.head(3)) # display the top thr
25 print(df3.groupby('cluster').min()) # display the means of
26 from sklearn.metrics import silhouette_score # coefficient score w
27 df4 = df3.drop('cluster', axis=1) # create a new data f
28 # for loop to determine optimum K groups
29 for i in range(3, 10): # for loop to determi
30     k_groups = KMeans(n_clusters=i).fit(df4) # K clusters must hav
31     labels = k_groups.labels_
32     print('K Groups = ', i, 'Silhouette Coefficient = ', silhouette_score(df4, la
33 # End of Data Mining - Cluster Analysis
34
35 # Data Mining - classification & regression
36 import pandas as pd
37 from sklearn.linear_model import LinearRegression, LogisticRegression
38 from sklearn.model_selection import train_test_split
39 decided_width = 600

Run: DataMining
svc_score = 0.7066666666666667
[[ 630 1351]
 [ 597 1172]]
knn_reg score = 0.6129302862047807
knn_class score = 0.8384
[[1000 0500]]
```

## Svc:



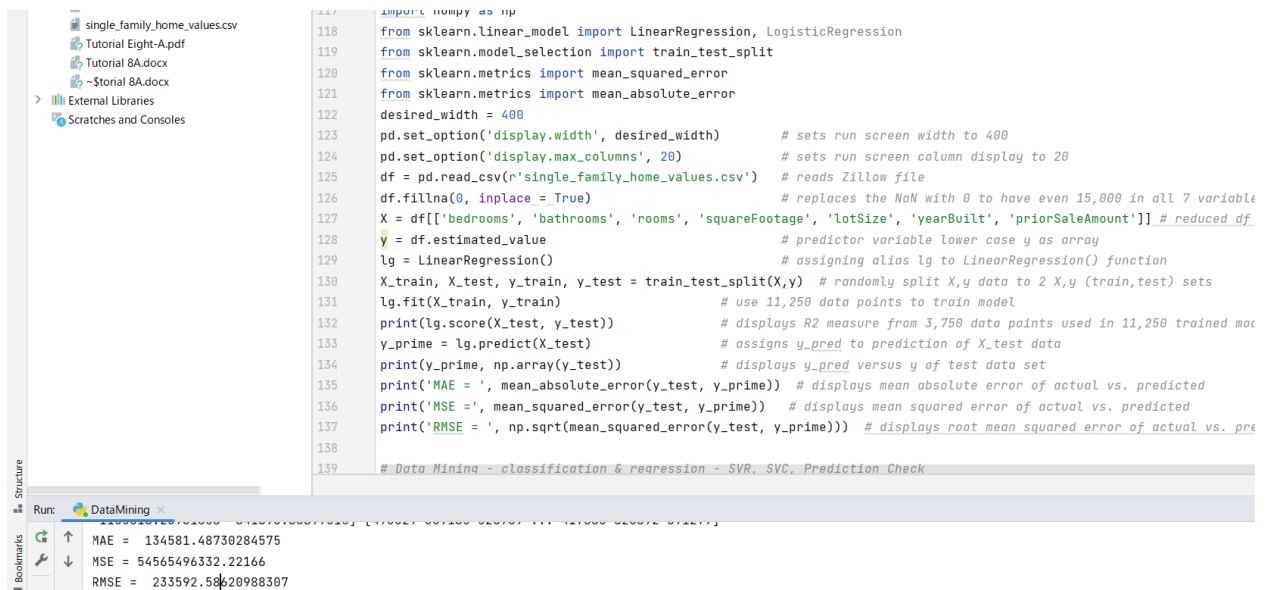
The screenshot shows a Jupyter Notebook interface with a file explorer on the left, a code editor in the center, and a console output at the bottom. The file explorer shows files like 'single\_family\_home\_values.csv', 'Tutorial Eight-A.pdf', and 'Tutorial 8A.docx'. The code editor contains Python code for data loading, preprocessing, and training an SVC model. The console output shows the results of the training and evaluation.

```
139 # Data Mining - classification & regression - SVR, SVC, Prediction Check
140 import pandas as pd
141 from sklearn.svm import SVC, SVR
142 from sklearn.model_selection import train_test_split
143 from sklearn.metrics import confusion_matrix
144 desired_width = 400
145 pd.set_option('display.width', desired_width) # sets run screen width to 400
146 pd.set_option('display.max_columns', 20) # sets run screen column display to 20
147 df = pd.read_csv(r'single_family_home_values.csv') # reads Zillow file
148 df.fillna(0, inplace = True) # replaces the NaN with 0 to have even 15,000 in
149 X = df[['bedrooms', 'bathrooms', 'rooms', 'squareFootage', 'lotSize', 'yearBuilt', 'priorSaleAmount']]
150 X1 = X # duplicate of X
151 y = df.estimated_value # predictor variable lower case y as array
152 df['estimated_value_class'] = df.estimated_value.apply(lambda x: 'low' if x < 500000 else 'high')
153 y2 = df.estimated_value_class # assigns y2 as cat variable estima
154 X_train, X_test, y_train, y_test = train_test_split(X,y) # randomly split X,y data to 2 X,y (tr
155 X1_train, X1_test, y2_train, y2_test = train_test_split(X1,y2) # randomly split X,y data to 2 X
156 svr_reg = SVR() # assign svr_reg to the SVR function
157 svc_class = SVC() # assign svc_class to the SVC function
158 svr_reg.fit(X_train, y_train) # fit a SVR() model using 11,250 data points
159 print('svr_score = ', svr_reg.score(X_test, y_test)) # score the SVR model based on 11,250 data points
```

Run: DataMining

```
svr_score = -0.061225944248852526
svc_score = 0.7066666666666667
[[ 630 1351]
 [ 597 1172]]
```

## Prediction check:



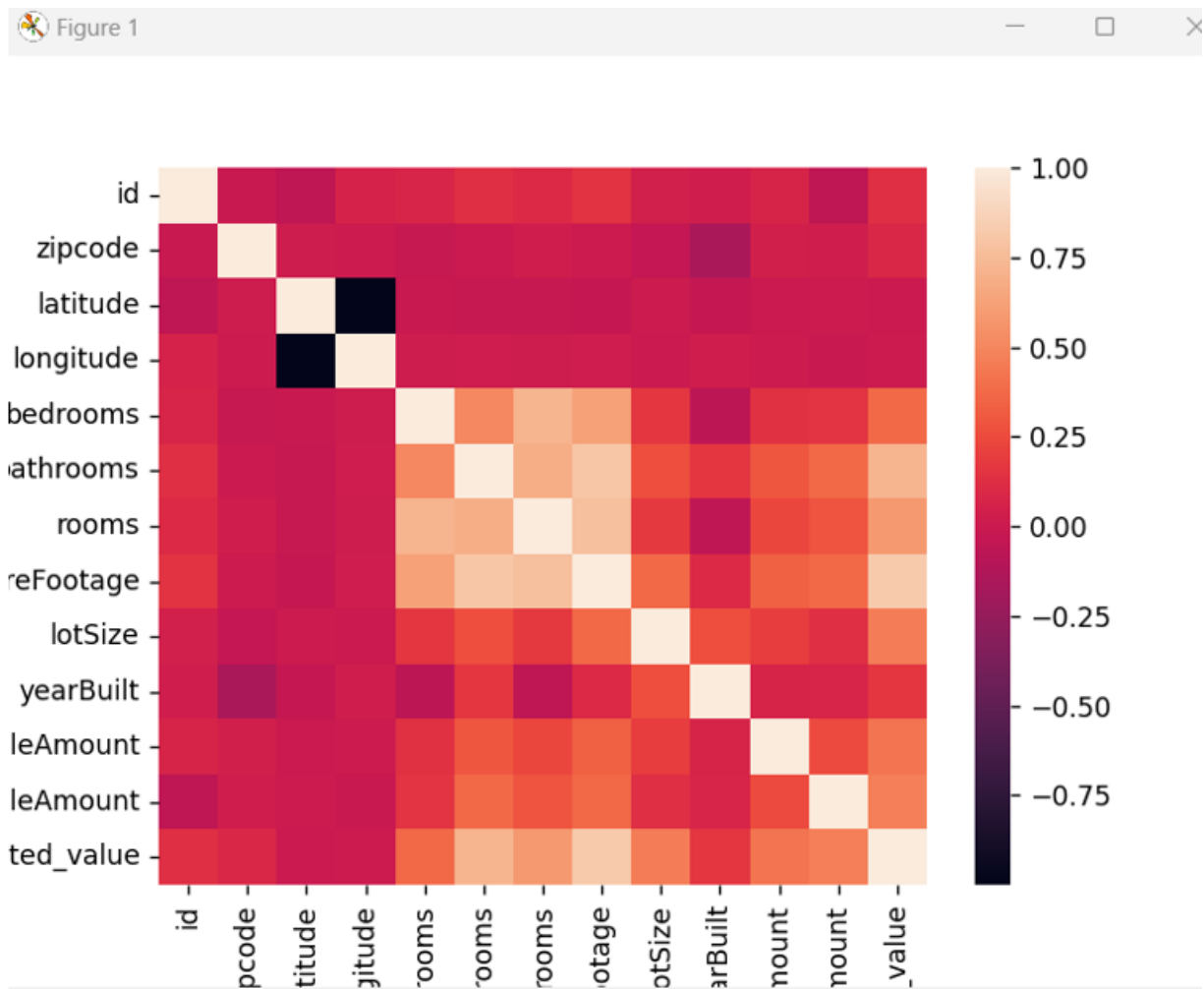
The screenshot shows a Jupyter Notebook interface with a file explorer on the left, a code editor in the center, and a console output at the bottom. The file explorer shows files like 'single\_family\_home\_values.csv', 'Tutorial Eight-A.pdf', and 'Tutorial 8A.docx'. The code editor contains Python code for data loading, preprocessing, and training a LinearRegression model. The console output shows the results of the training and evaluation.

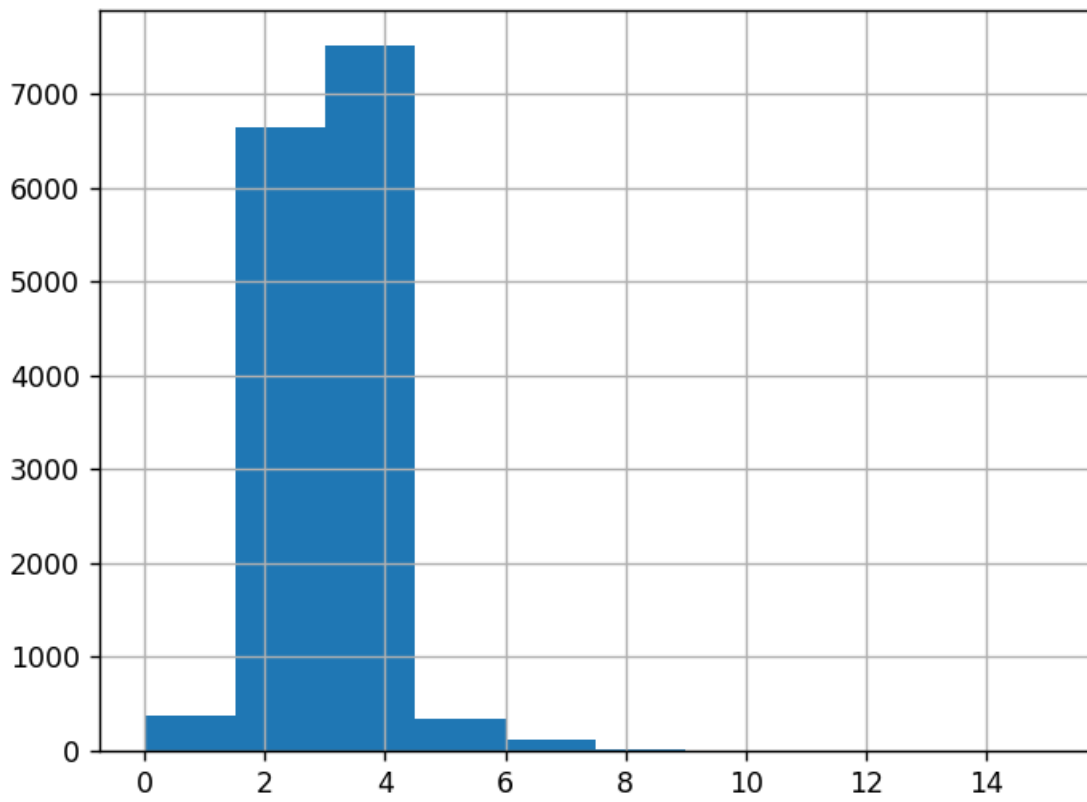
```
118 from sklearn.linear_model import LinearRegression, LogisticRegression
119 from sklearn.model_selection import train_test_split
120 from sklearn.metrics import mean_squared_error
121 from sklearn.metrics import mean_absolute_error
122 desired_width = 400
123 pd.set_option('display.width', desired_width) # sets run screen width to 400
124 pd.set_option('display.max_columns', 20) # sets run screen column display to 20
125 df = pd.read_csv(r'single_family_home_values.csv') # reads Zillow file
126 df.fillna(0, inplace = True) # replaces the NaN with 0 to have even 15,000 in all 7 variable
127 X = df[['bedrooms', 'bathrooms', 'rooms', 'squareFootage', 'lotSize', 'yearBuilt', 'priorSaleAmount']] # reduced df
128 y = df.estimated_value # predictor variable lower case y as array
129 lg = LinearRegression() # assigning alias lg to LinearRegression() function
130 X_train, X_test, y_train, y_test = train_test_split(X,y) # randomly split X,y data to 2 X,y (train,test) sets
131 lg.fit(X_train, y_train) # use 11,250 data points to train model
132 print(lg.score(X_test, y_test)) # displays R2 measure from 3,750 data points used in 11,250 trained moc
133 y_prime = lg.predict(X_test) # assigns y_pred to prediction of X_test data
134 print(y_prime, np.array(y_test)) # displays y_pred versus y of test data set
135 print('MAE = ', mean_absolute_error(y_test, y_prime)) # displays mean absolute error of actual vs. predicted
136 print('MSE = ', mean_squared_error(y_test, y_prime)) # displays mean squared error of actual vs. predicted
137 print('RMSE = ', np.sqrt(mean_squared_error(y_test, y_prime))) # displays root mean squared error of actual vs. pre
138
139 # Data Mining - classification & regression - SVR, SVC, Prediction Check
```

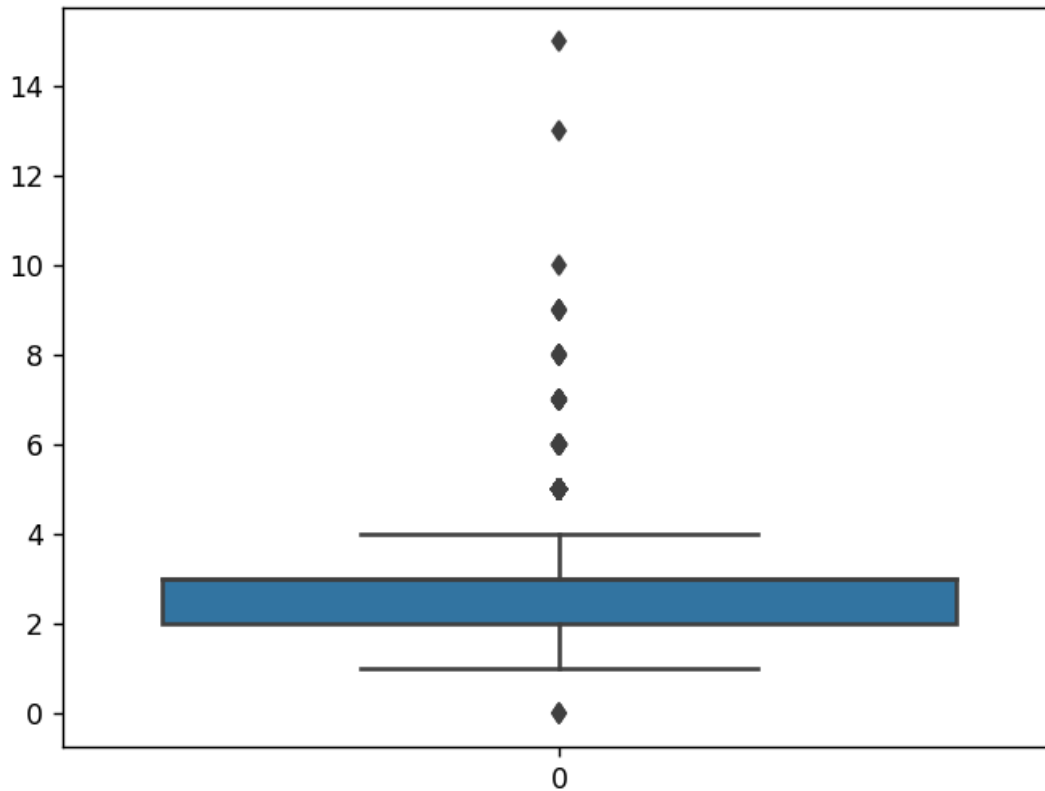
Run: DataMining

```
MAE = 134581.48730284575
MSE = 54565496332.22166
RMSE = 233592.58620988307
```

## Association and Correlation







## Dimensionality Reduction

```

Tutorial 8A - DataMining.py
File Edit View Navigate Code Refactor Run Tools VCS Window Help Tutorial 8A - DataMining.py

Project
  Tutorial 8A C:\Users\adhir\Documents\MIS502
    DataMining.py
    ESSdata.csv
    single_family_home_values.csv
    Tutorial Eight-A.pdf
    Tutorial 8A.docx
    -Storial 8A.docx
  External Libraries
  Scratches and Consoles

DataMining.py
254 desired_width = 400
255 pd.set_option('display.width', desired_width) # sets run screen width to 400
256 pd.set_option('display.max_columns', 20) # sets run screen column display to 20
257 df = pd.read_csv(r'single_family_home_values.csv') # reads Zillow file
258 df.fillna(0, inplace = True) # replaces the NaN with 0 to have even 15,000 in all 7 variables
259 X = df[['bedrooms', 'bathrooms', 'rooms', 'squareFootage', 'lotSize', 'yearBuilt', 'priorSaleAmount']] # reduced df as upper ci
260 X1 = X
261 pca = PCA(4)
262 X_transformed = pca.fit_transform(X)
263 y = df.estimated_value # predictor variable lower case y as array
264 y1 = df.estimated_value
265 lg = LinearRegression()
266 X_train, X_test, y_train, y_test = train_test_split(X_transformed, y) # randomly split X,y data to 2 X,y (train,test) se
267 X1_train, X1_test, y1_train, y1_test = train_test_split(X1, y1) # randomly split X,y data to 2 X,y (train,test) sets
268 print(X_transformed.shape)
269 print(X_train.shape, y_train.shape, X_test.shape, y_test.shape) # we used default settings 80% is train to 20% test
270 lg.fit(X_train, y_train) # Using 11,250 data points to train model
271 print('PCA = ', lg.score(X_test, y_test)) # Using 3,750 data points to test/evaluate R2 of model
272 lg.fit(X1_train, y1_train) # Using 11,250 data points to train model
273 print('non-PCA = ', lg.score(X1_test, y1_test)) # Using 3,750 data points to test/evaluate R2 of model

Run: DataMining
"C:\Users\adhir\anaconda\envs\Tutorial 8A\python.exe" "C:\Users\adhir\Documents\MIS502\Tutorial 8A\DataMining.py"
(15000, 4)
(11250, 4) (11250,) (3750, 4) (3750,)
PCA = 0.6967741049937116
non-PCA = 0.7645075484296683

```