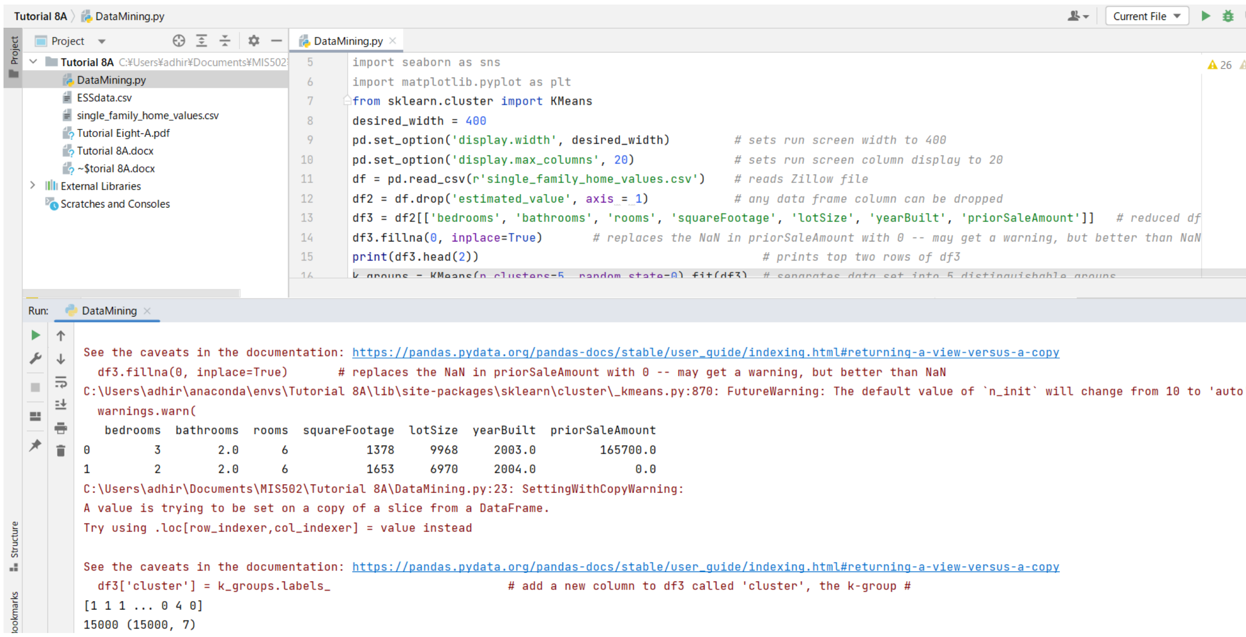
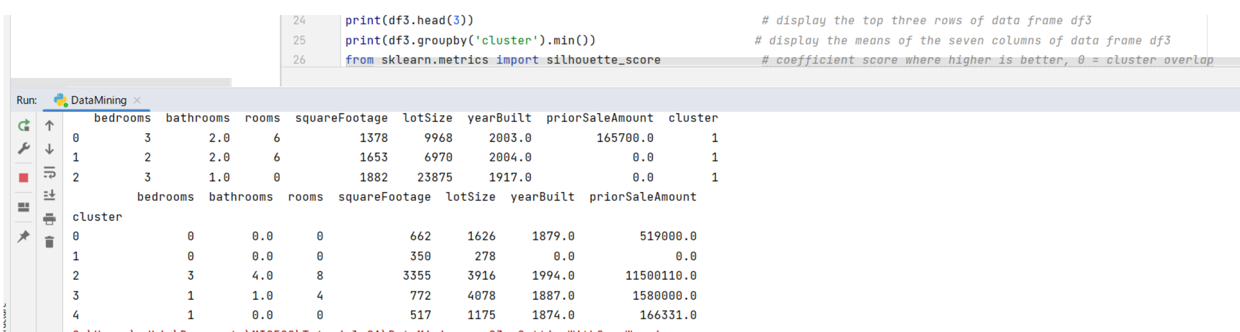
Tutorial 8A

Cluster Analysis



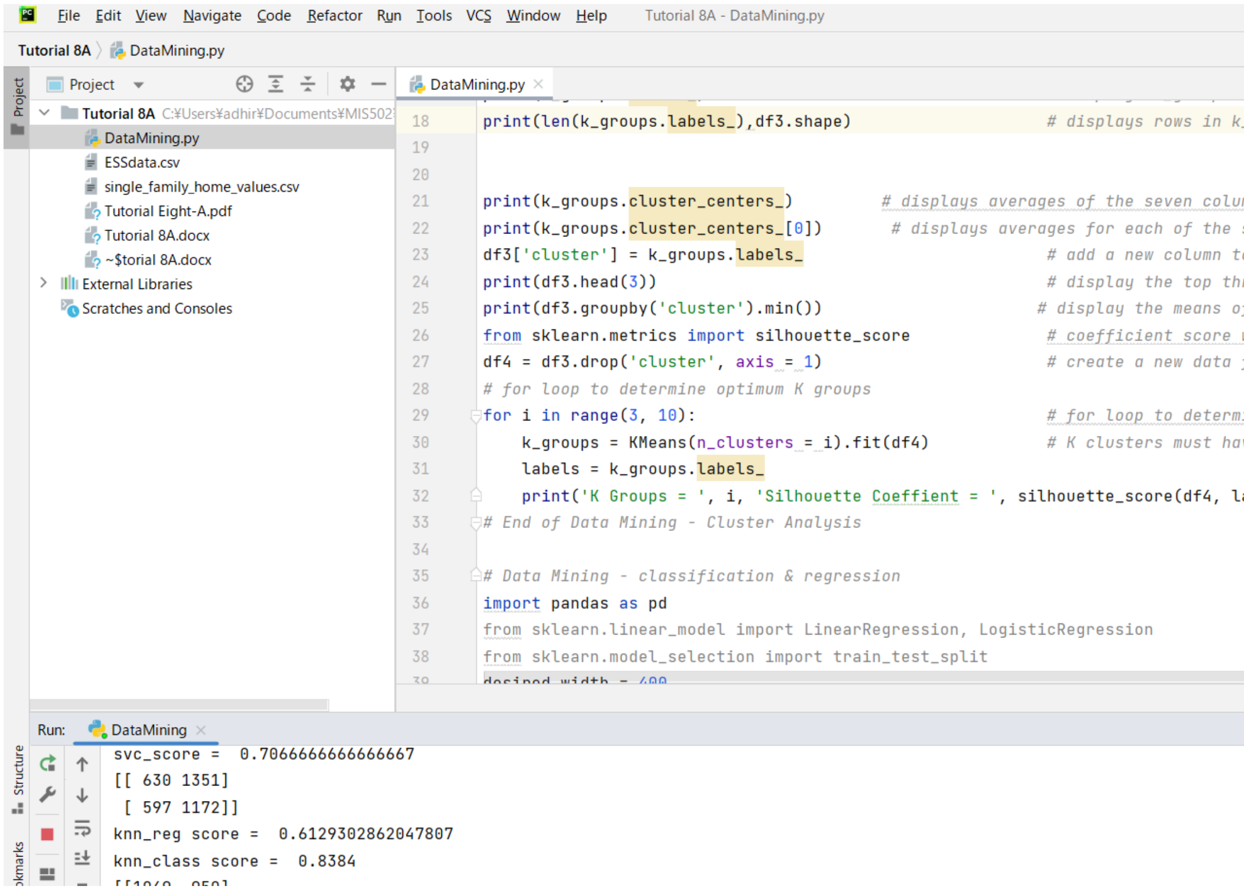

```
[1 1 1 ... 0 4 0]
15000 (15000, 7)
[[3.31226296e+00 3.83944374e+00 8.42730721e+00 2.69720607e+03
  6.97174968e+03 1.94200506e+03 7.43586930e+05]
 [2.64078392e+00 1.93525180e+00 5.86293724e+00 1.39300918e+03
  5.94409712e+03 1.93060060e+03 3.93563157e+04]
 [3.00000000e+00 4.50000000e+00 9.00000000e+00 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:



Regression and Classification

Knn :



Svc:

```
single_family_home_values.csv
Tutorial Eight-A.pdf
Tutorial 8A.docx
~$torial 8A.docx
> External Libraries
Scratches and Consoles
```

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

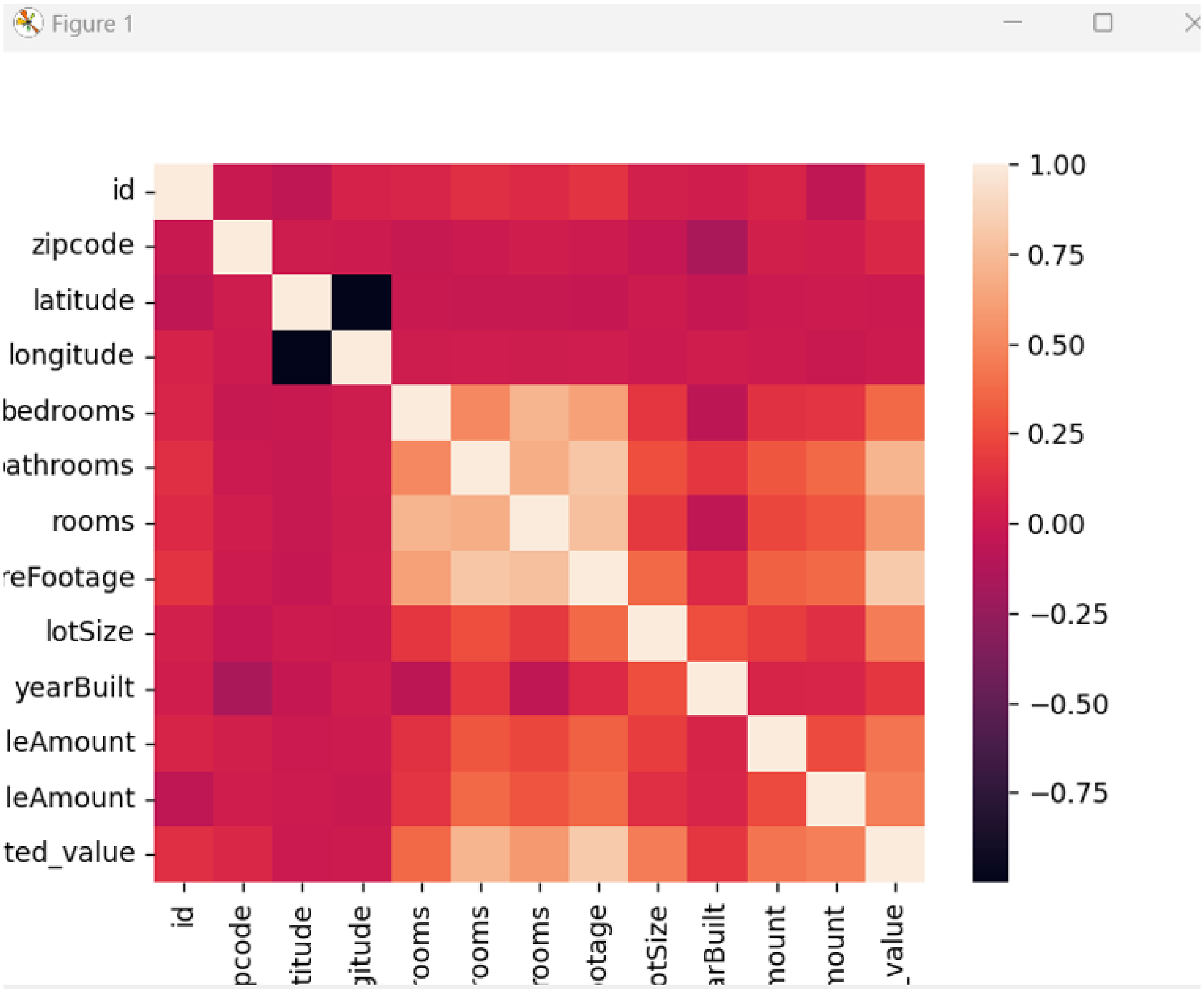
```
single_family_home_values.csv
Tutorial Eight-A.pdf
Tutorial 8A.docx
~$torial 8A.docx
> External Libraries
Scratches and Consoles
```

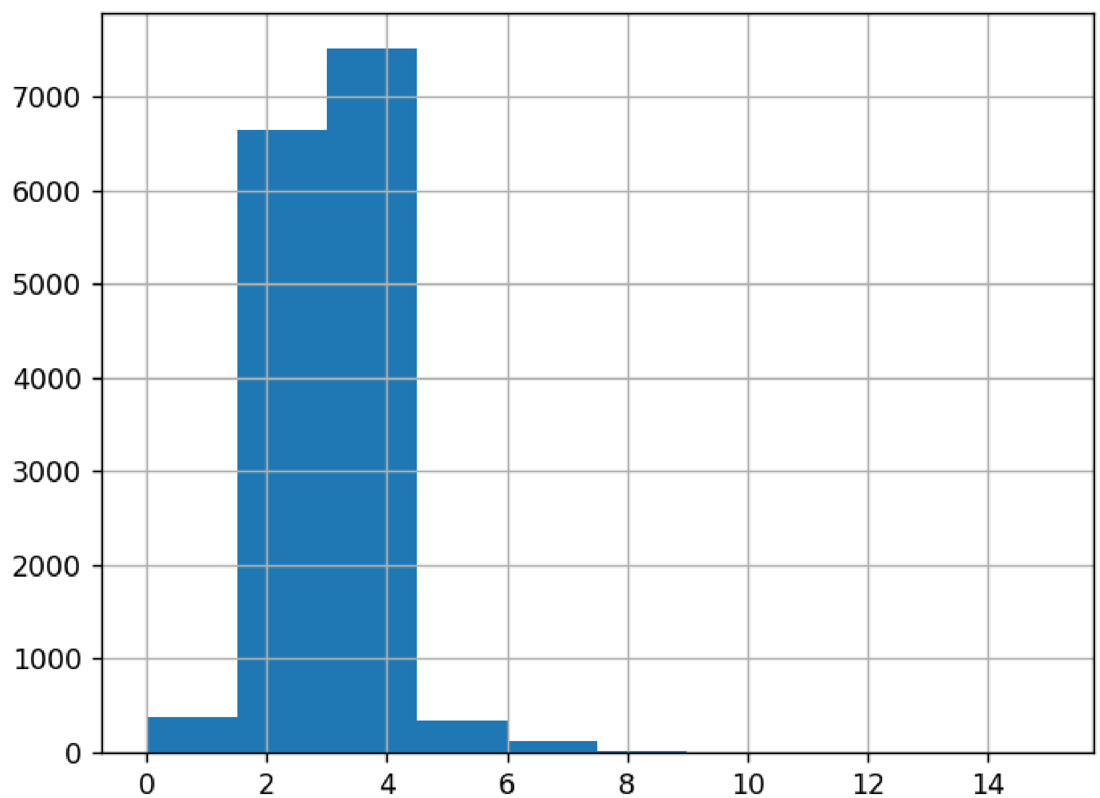
```python
import numpy as np
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
desired_width = 400
pd.set_option('display.width', desired_width)        # sets run screen width to 400
pd.set_option('display.max_columns', 20)             # sets run screen column display to 20
df = pd.read_csv(r'single_family_home_values.csv')   # reads Zillow file
df.fillna(0, inplace = True)                         # replaces the NaN with 0 to have even 15,000 in all 7 variable
X = df[['bedrooms', 'bathrooms', 'rooms', 'squareFootage', 'lotSize', 'yearBuilt', 'priorSaleAmount']] # reduced df
y = df.estimated_value                               # predictor variable lower case y as array
lg = LinearRegression()                              # assigning alias lg to LinearRegression() function
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
lg.fit(X_train, y_train)                     # use 11,250 data points to train model
print(lg.score(X_test, y_test))              # displays R2 measure from 3,750 data points used in 11,250 trained moc
y_prime = lg.predict(X_test)                 # assigns y_pred to prediction of X_test data
print(y_prime, np.array(y_test))             # displays y_pred versus y of test data set
print('MAE = ', mean_absolute_error(y_test, y_prime))  # displays mean absolute error of actual vs. predicted
print('MSE =', mean_squared_error(y_test, y_prime))   # displays mean squared error of actual vs. predicted
print('RMSE = ', np.sqrt(mean_squared_error(y_test, y_prime)))  # displays root mean squared error of actual vs. pre

# Data Mining - classification & regression - SVR, SVC, Prediction Check
```

Run: DataMining

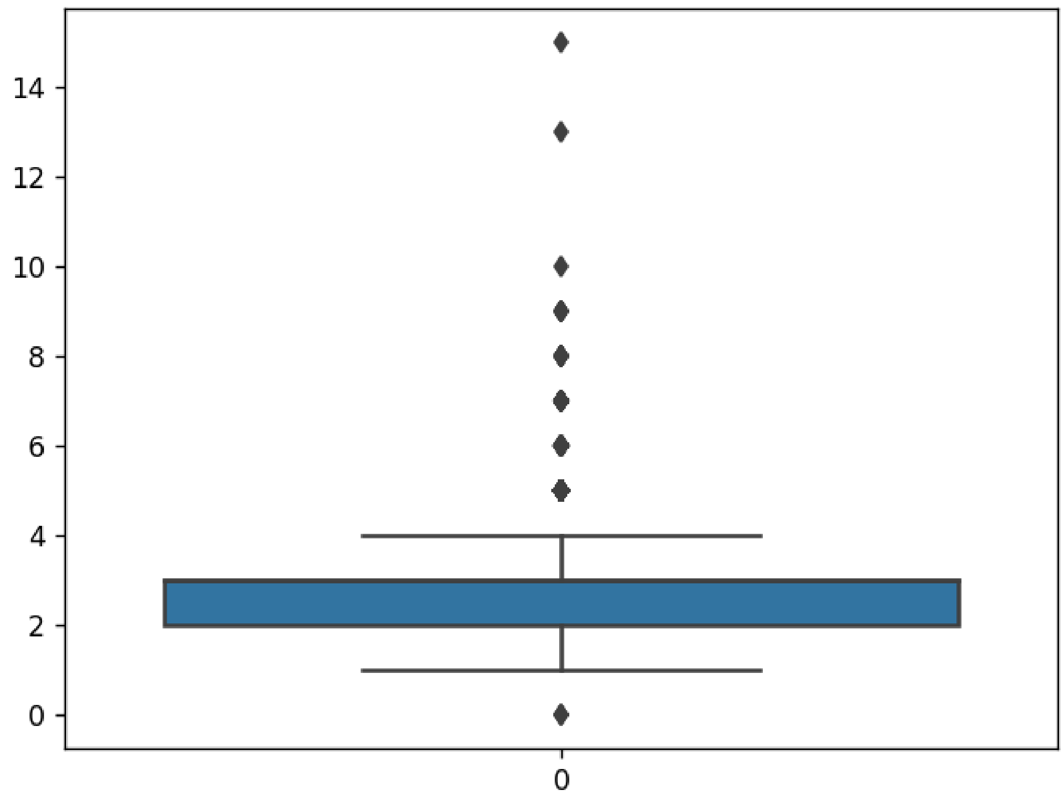
```
MAE =  134581.48730284575
MSE = 54565496332.22166
RMSE =  233592.58620988307
```

Association and Correlation





7000
6000
5000
4000
3000
2000
1000
0
0
2
4
6
8
10
12
14



Dimensionality Reduction