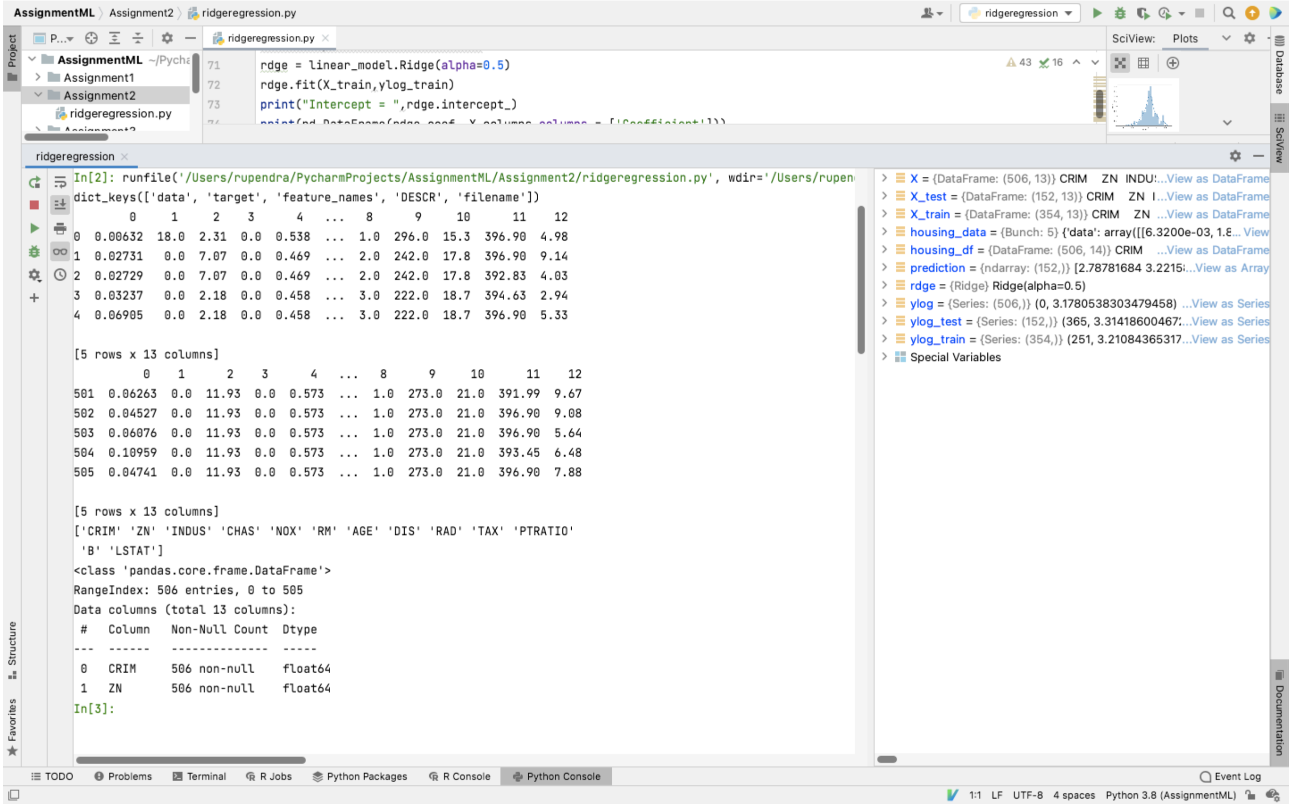
Assignment 3
Ridge Regression

In various types of regression models we are try to predict the numerical output based on some given input. Linear regression is a basic algorithm for regression, in which a linear relationship between the input and output data is formed. In other linear regression models, an extra penalty term is added with the loss function during training to encourages simpler models with having smaller coefficient values.
Ridge Regression is an example of regularized linear regression that includes an L2 penalty. This has the effect of shrinking the coefficients for those input variables that do not contribute much to the prediction task.

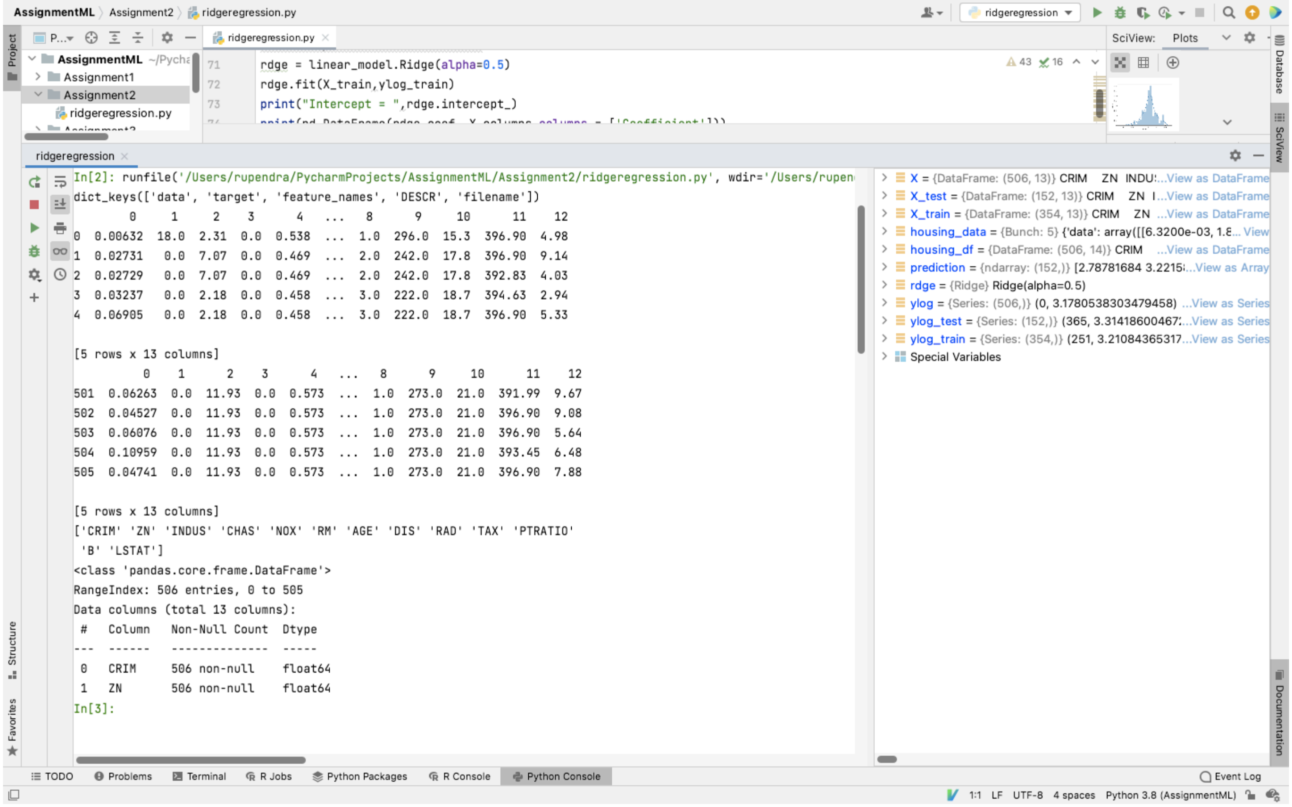
In this, I'm trying to perform the ridge regression on Boston Housing dataset.



```
rdge = linear_model.Ridge(alpha=0.5)
rdge.fit(X_train,ylog_train)
print("Intercept = ",rdge.intercept_)
print(pd.DataFrame(rdge.coef_,X.columns,columns = ['Coefficient']))
```

```
In[2]: runfile('/Users/rupendra/PycharmProjects/AssignmentML/Assignment2/ridgeregression.py', wdir='/Users/rupen
dict_keys(['data', 'target', 'feature_names', 'DESCR', 'filename'])
        0     1     2    3      4  ...    8      9    10      11    12
0  0.00632  18.0  2.31  0.0  0.538  ...  1.0  296.0  15.3  396.90  4.98
1  0.02731   0.0  7.07  0.0  0.469  ...  2.0  242.0  17.8  396.90  9.14
2  0.02729   0.0  7.07  0.0  0.469  ...  2.0  242.0  17.8  392.83  4.03
3  0.03237   0.0  2.18  0.0  0.458  ...  3.0  222.0  18.7  394.63  2.94
4  0.06905   0.0  2.18  0.0  0.458  ...  3.0  222.0  18.7  396.90  5.33

[5 rows x 13 columns]
           0    1      2    3      4  ...    8      9    10      11    12
501  0.06263  0.0  11.93  0.0  0.573  ...  1.0  273.0  21.0  391.99  9.67
502  0.04527  0.0  11.93  0.0  0.573  ...  1.0  273.0  21.0  396.90  9.08
503  0.06076  0.0  11.93  0.0  0.573  ...  1.0  273.0  21.0  396.90  5.64
504  0.10959  0.0  11.93  0.0  0.573  ...  1.0  273.0  21.0  393.45  6.48
505  0.04741  0.0  11.93  0.0  0.573  ...  1.0  273.0  21.0  396.90  7.88

[5 rows x 13 columns]
['CRIM' 'ZN' 'INDUS' 'CHAS' 'NOX' 'RM' 'AGE' 'DIS' 'RAD' 'TAX' 'PTRATIO'
 'B' 'LSTAT']
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 13 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   CRIM     506 non-null    float64
 1   ZN       506 non-null    float64
In[3]:
```

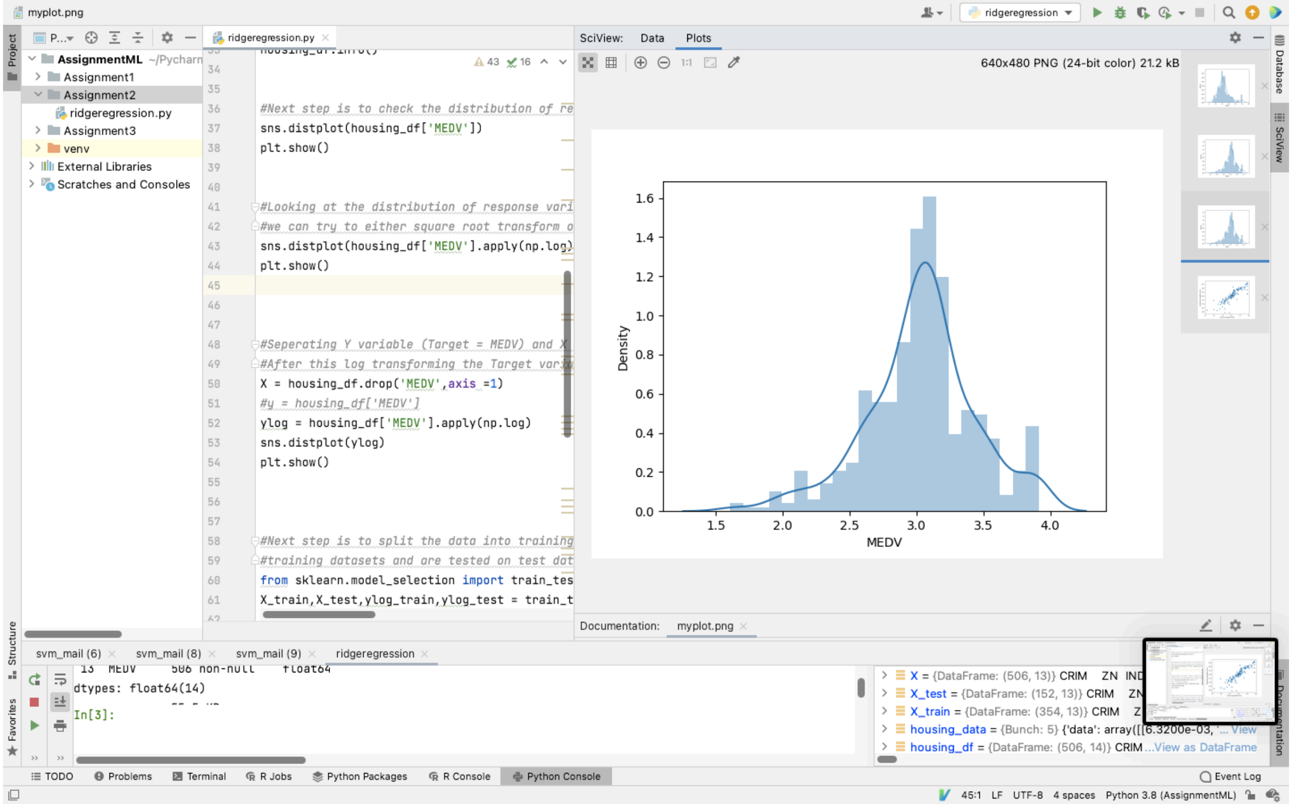
```
X = {DataFrame: (506, 13)} CRIM    ZN  INDU:...View as DataFrame
X_test = {DataFrame: (152, 13)} CRIM    ZN  I...View as DataFrame
X_train = {DataFrame: (354, 13)} CRIM    ZN  ...View as DataFrame
housing_data = {Bunch: 5} {'data': array([[6.3200e-03, 1.8... View
housing_df = {DataFrame: (506, 14)} CRIM    ...View as DataFrame
prediction = {ndarray: (152,)} [2.78781684 3.2215...View as Array
rdge = {Ridge} Ridge(alpha=0.5)
ylog = {Series: (506,)} (0, 3.1780538303479458) ...View as Series
ylog_test = {Series: (152,)} (365, 3.31418600467...View as Series
ylog_train = {Series: (354,)} (251, 3.21084365317...View as Series
Special Variables
```



```
rdge = linear_model.Ridge(alpha=0.5)
rdge.fit(X_train,ylog_train)
print("Intercept = ",rdge.intercept_)
```

```
In[2]: runfile('/Users/rupendra/PycharmProjects/AssignmentML/Assignment2/ridgeregression.py', wdir='/Users/rupen
dict_keys(['data', 'target', 'feature_names', 'DESCR', 'filename'])
         0     1     2    3      4  ...    8      9    10      11    12
0  0.00632  18.0  2.31  0.0  0.538  ...  1.0  296.0  15.3  396.90  4.98
1  0.02731   0.0  7.07  0.0  0.469  ...  2.0  242.0  17.8  396.90  9.14
2  0.02729   0.0  7.07  0.0  0.469  ...  2.0  242.0  17.8  392.83  4.03
3  0.03237   0.0  2.18  0.0  0.458  ...  3.0  222.0  18.7  394.63  2.94
4  0.06905   0.0  2.18  0.0  0.458  ...  3.0  222.0  18.7  396.90  5.33

[5 rows x 13 columns]
           0    1      2    3      4  ...    8      9    10      11    12
501  0.06263  0.0  11.93  0.0  0.573  ...  1.0  273.0  21.0  391.99  9.67
502  0.04527  0.0  11.93  0.0  0.573  ...  1.0  273.0  21.0  396.90  9.08
503  0.06076  0.0  11.93  0.0  0.573  ...  1.0  273.0  21.0  396.90  5.64
504  0.10959  0.0  11.93  0.0  0.573  ...  1.0  273.0  21.0  393.45  6.48
505  0.04741  0.0  11.93  0.0  0.573  ...  1.0  273.0  21.0  396.90  7.88

[5 rows x 13 columns]
['CRIM' 'ZN' 'INDUS' 'CHAS' 'NOX' 'RM' 'AGE' 'DIS' 'RAD' 'TAX' 'PTRATIO'
 'B' 'LSTAT']
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 13 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   CRIM     506 non-null    float64
 1   ZN       506 non-null    float64
In[3]:
```

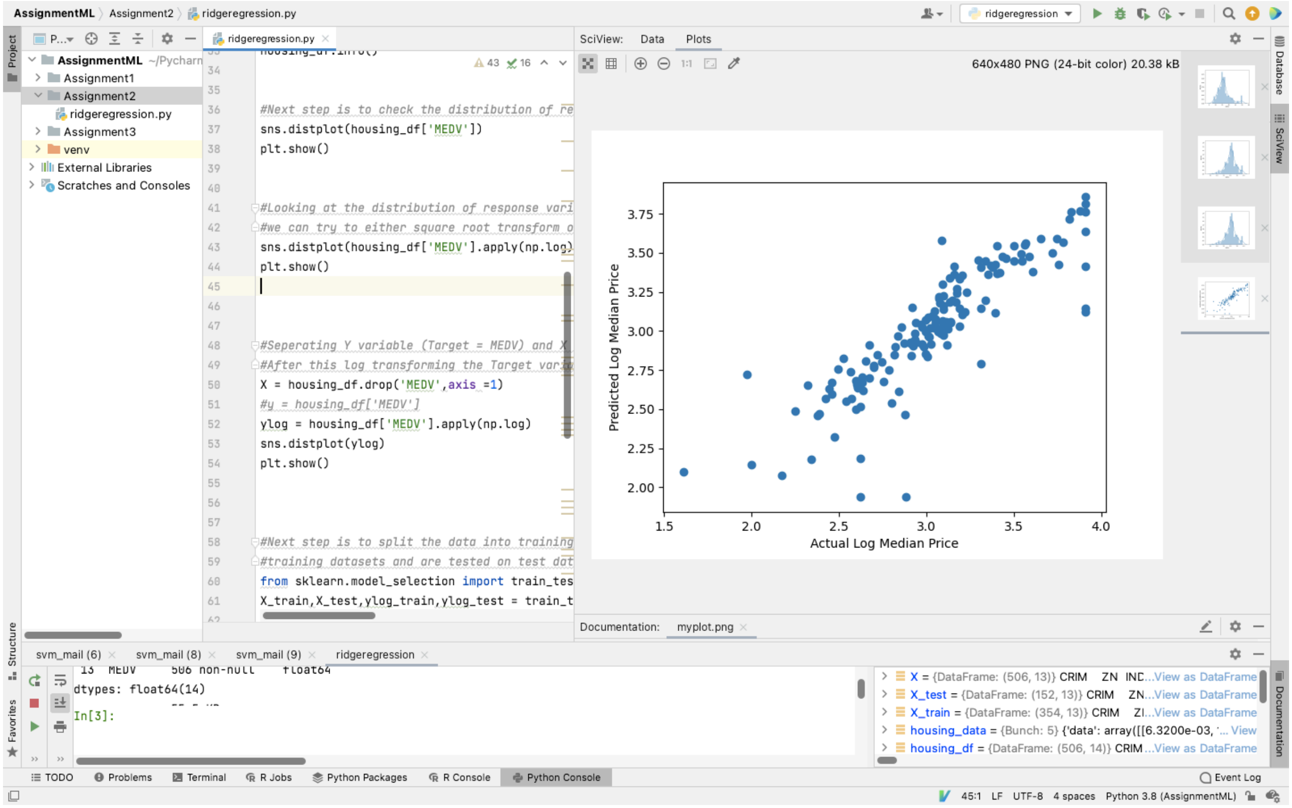
```
X = {DataFrame: (506, 13)} CRIM    ZN  INDU...View as DataFrame
X_test = {DataFrame: (152, 13)} CRIM    ZN  I...View as DataFrame
X_train = {DataFrame: (354, 13)} CRIM    ZN  ...View as DataFrame
housing_data = {Bunch: 5} {'data': array([[6.3200e-03, 1.8... View
housing_df = {DataFrame: (506, 14)} CRIM    ...View as DataFrame
prediction = {ndarray: (152,)} [2.78781684 3.2215...View as Array
rdge = {Ridge} Ridge(alpha=0.5)
ylog = {Series: (506,)} (0, 3.1780538303479458) ...View as Series
ylog_test = {Series: (152,)} (365, 3.31418600467...View as Series
ylog_train = {Series: (354,)} (251, 3.21084365317...View as Series
Special Variables
```



```
#Next step is to check the distribution of re
sns.distplot(housing_df['MEDV'])
plt.show()


#Looking at the distribution of response vari
#we can try to either square root transform o
sns.distplot(housing_df['MEDV'].apply(np.log)
plt.show()



#Seperating Y variable (Target = MEDV) and X
#After this log transforming the Target vari
X = housing_df.drop('MEDV',axis =1)
#y = housing_df['MEDV']
ylog = housing_df['MEDV'].apply(np.log)
sns.distplot(ylog)
plt.show()



#Next step is to split the data into training
#training datasets and are tested on test dat
from sklearn.model_selection import train_tes
X_train,X_test,ylog_train,ylog_test = train_t
```

```
 13  MEDV     506 non-null    float64
dtypes: float64(14)
In[3]:
```

```
X = {DataFrame: (506, 13)} CRIM    ZN  IND
X_test = {DataFrame: (152, 13)} CRIM    ZN
X_train = {DataFrame: (354, 13)} CRIM    Z
housing_data = {Bunch: 5} {'data': array([[6.3200e-03, ... View
housing_df = {DataFrame: (506, 14)} CRIM ...View as DataFrame
```



```
housing_df.info()


#Next step is to check the distribution of re
sns.distplot(housing_df['MEDV'])
plt.show()


#Looking at the distribution of response vari
#we can try to either square root transform o
sns.distplot(housing_df['MEDV'].apply(np.log)
plt.show()



#Seperating Y variable (Target = MEDV) and X
#After this log transforming the Target vari
X = housing_df.drop('MEDV',axis =1)
#y = housing_df['MEDV']
ylog = housing_df['MEDV'].apply(np.log)
sns.distplot(ylog)
plt.show()



#Next step is to split the data into training
#training datasets and are tested on test dat
from sklearn.model_selection import train_tes
X_train,X_test,ylog_train,ylog_test = train_t
```

```
 13  MEDV     506 non-null    float64
dtypes: float64(14)
In[3]:
```

X = {DataFrame: (506, 13)} CRIM   ZN INC...View as DataFrame

X_test = {DataFrame: (152, 13)} CRIM   ZN...View as DataFrame

X_train = {DataFrame: (354, 13)} CRIM   ZI...View as DataFrame

housing_data = {Bunch: 5} {'data': array([[6.3200e-03, '... View

housing_df = {DataFrame: (506, 14)} CRIM ...View as DataFrame