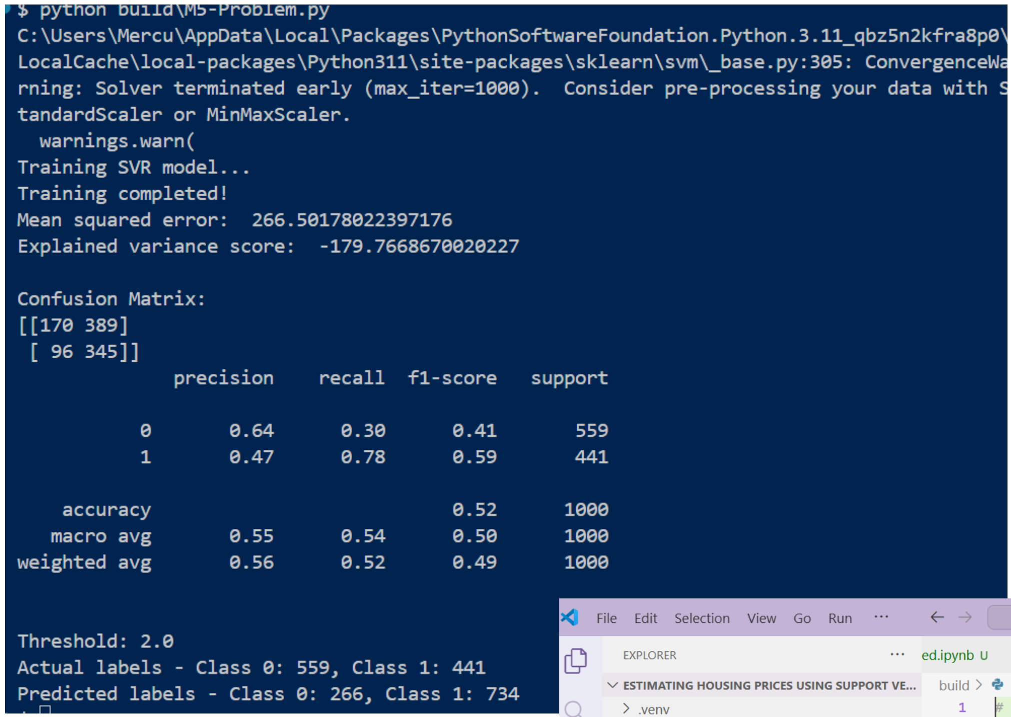
Name: **Serene Plummer**

Assignment: Week 8 Review Lab

Date Due: Saturday, October 11, 2025

Class: Computer Science II - CSCE 1040 Section 501
https://github.com/serene4444/Estimating-Housing-Prices-using-Support-Vector-Machine



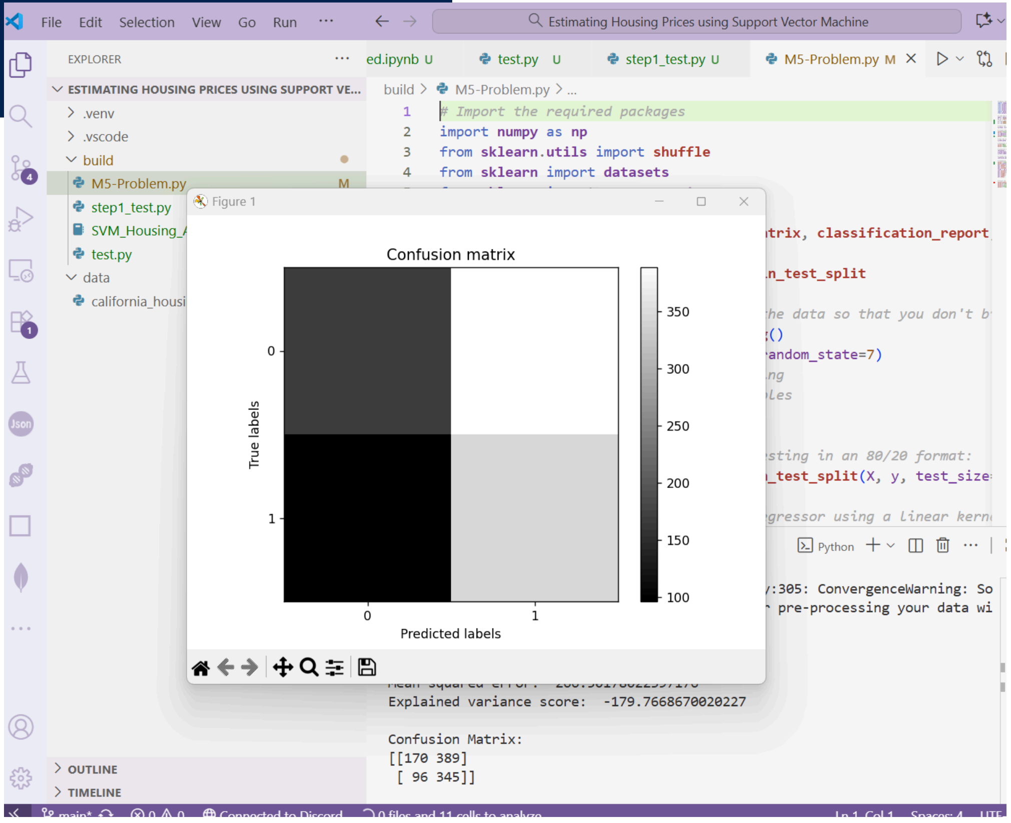
```
$ python build\M5-Problem.py
C:\Users\Mercu\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\
LocalCache\local-packages\Python311\site-packages\sklearn\svm\_base.py:305: ConvergenceWa
rning: Solver terminated early (max_iter=1000).  Consider pre-processing your data with S
tandardScaler or MinMaxScaler.
  warnings.warn(
Training SVR model...
Training completed!
Mean squared error:  266.50178022397176
Explained variance score:  -179.7668670020227

Confusion Matrix:
[[170 389]
 [ 96 345]]
              precision    recall  f1-score   support

           0       0.64      0.30      0.41       559
           1       0.47      0.78      0.59       441

    accuracy                           0.52      1000
   macro avg       0.55      0.54      0.50      1000
weighted avg       0.56      0.52      0.49      1000


Threshold: 2.0
Actual labels - Class 0: 559, Class 1: 441
Predicted labels - Class 0: 266, Class 1: 734
```



```python
# Import the required packages
import numpy as np
from sklearn.utils import shuffle
from sklearn import datasets
```

```
trix, classification_report
n_test_split
he data so that you don't b
()
andom_state=7)
ng
les
sting in an 80/20 format:
_test_split(X, y, test_size
gressor using a linear kern
```

```
:305: ConvergenceWarning: So
 pre-processing your data wi
Explained variance score:  -179.7668670020227

Confusion Matrix:
[[170 389]
 [ 96 345]]
```

```python
# Import the required packages
import numpy as np
from sklearn.utils import shuffle
from sklearn import datasets
from sklearn import preprocessing
from sklearn.svm import SVR
from sklearn.metrics import confusion_matrix, classification_report,
mean_squared_error, explained_variance_score
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split

# Load the housing dataset and shuffle the data so that you don't bias
your analysis.
data = datasets.fetch_california_housing()
X, y = shuffle(data.data, data.target, random_state=7)
# Use a smaller subset for faster training
X = X[:5000]  # Use only first 5000 samples
y = y[:5000]

# Split the dataset into training and testing in an 80/20 format:
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=7)

# Create and train the Support Vector Regressor using a linear kernel.
print("Training SVR model...")
sv_regressor = SVR(kernel='linear', C=1.0, max_iter=1000)  # Add max_iter
for faster training
sv_regressor.fit(X_train, y_train)
print("Training completed!")

# Run the regressor on the testing data and predict the output (predicted
labels).
y_test_pred = sv_regressor.predict(X_test)

# Evaluate the performance of the regressor and print the initial metrics.
print("Mean squared error: ", mean_squared_error(y_test, y_test_pred))
print("Explained variance score: ", explained_variance_score(y_test,
y_test_pred))
```

```python
# binarize the predicted values & the actual values using threshold of
25.00.
threshold = 2.0
y_pred_label = (y_test_pred > threshold).astype(int)
y_test_label = (y_test > threshold).astype(int)

# Create the confusion matrix using the predicted labels and the actual
labels.
confusion_mat = confusion_matrix(y_test_label, y_pred_label)

# Visualize the confusion matrix.
print("\nConfusion Matrix:")
print(confusion_mat)
plt.imshow(confusion_mat, interpolation='nearest', cmap=plt.cm.gray)
plt.title('Confusion matrix')
plt.colorbar()
ticks = np.arange (2)
plt.xticks(ticks, ticks)
plt.yticks(ticks, ticks)
plt.ylabel('True labels')
plt.xlabel('Predicted labels')
plt.show()


# Print the classification report based on the confusion matrix.
print(classification_report(y_test_label, y_pred_label))
print(f"\nThreshold: 2.0")
print(f"Actual labels - Class 0: {sum(y_test_label == 0)}, Class 1:
{sum(y_test_label == 1)}")
print(f"Predicted labels - Class 0: {sum(y_pred_label == 0)}, Class 1:
{sum(y_pred_label == 1)}")
```

Explanation and Approach

1. Data loading and shuffling
   - Uses `sklearn.datasets.fetch_california_housing()` to load the California Housing dataset.
   - Shuffles the dataset to avoid ordering bias using `sklearn.utils.shuffle`.

2. Subsampling for speed
   - For faster experimentation, the script reduces the dataset to the first 5,000 samples (this keeps runtime small during local testing).

3. Train / Test split
   - Splits the data into training and testing sets with an 80/20 ratio using `train_test_split`.

4. Model training
   - Trains a Support Vector Regressor (`sklearn.svm.SVR`) with a linear kernel and C=1.0.
   - `max_iter` is set to 1000 in the script to limit training time when experimenting locally.

5. Regression evaluation
   - Evaluates the regressor using Mean Squared Error (MSE) and Explained Variance Score (EVS).
   - Prints the regression metrics.

6. Binarization and classification analysis (post-hoc)
   - Binarizes continuous predictions and ground-truth values with a threshold (default `2.0`, meaning $200k as the scale is in hundreds of thousands).
   - Produces predicted labels and actual labels: 0 = low price (<= threshold), 1 = high price (> threshold).
   - Computes a confusion matrix and prints it.
   - Visualizes the confusion matrix (grayscale heatmap) and prints a classification report (precision / recall / f1-score).