https://github.com/YousefALH/MLHWCore/tree/main/Homework/Homework%204%20model

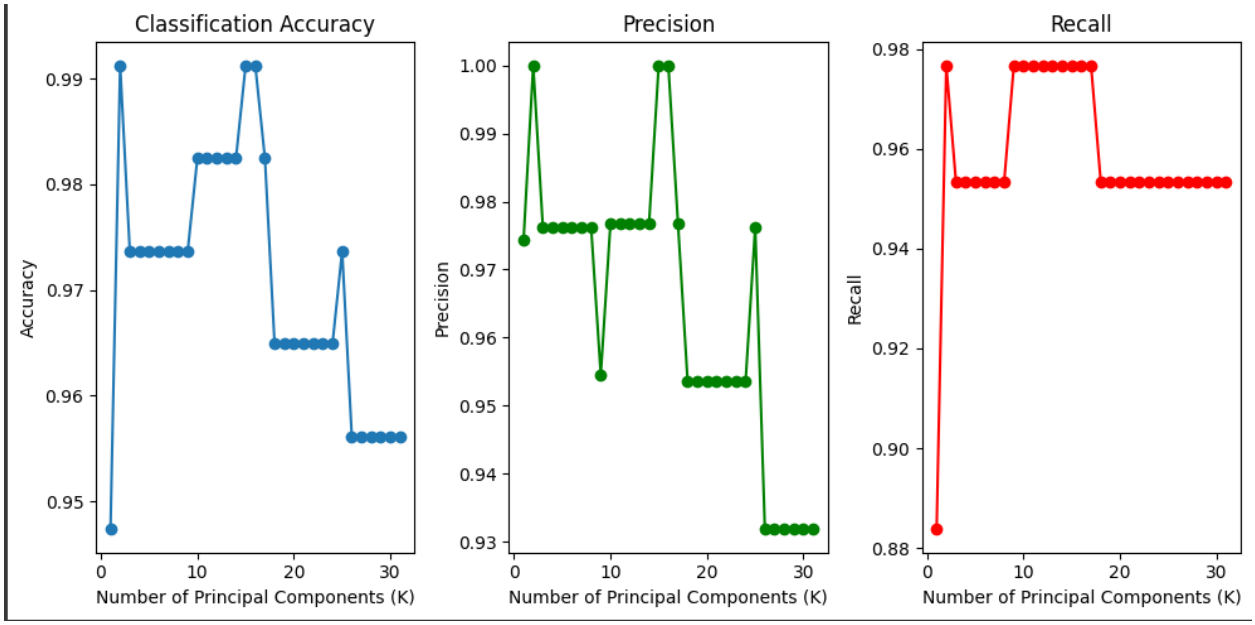
https://colab.research.google.com/drive/1YaUbDDEsmoWJj3Vq3t2ofjV-sISUv-zz?authuser=1#scrollTo=0kAiyWsNu1ub

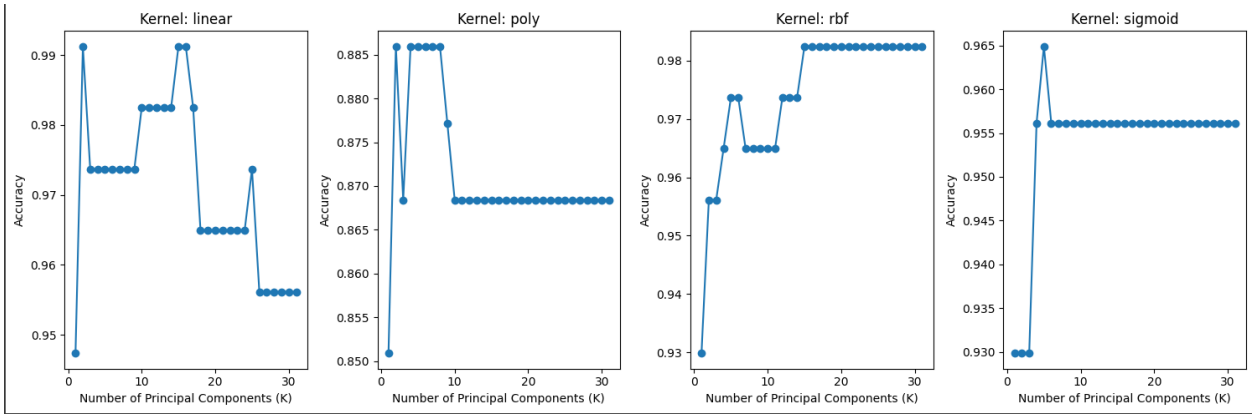
Problem 1 (50pts):

Use the cancer dataset to build an SVM classifier to classify the type of cancer (Malignant vs. benign). Use the PCA feature extraction for your training. Perform N number of independent training (N=1, ..., K).

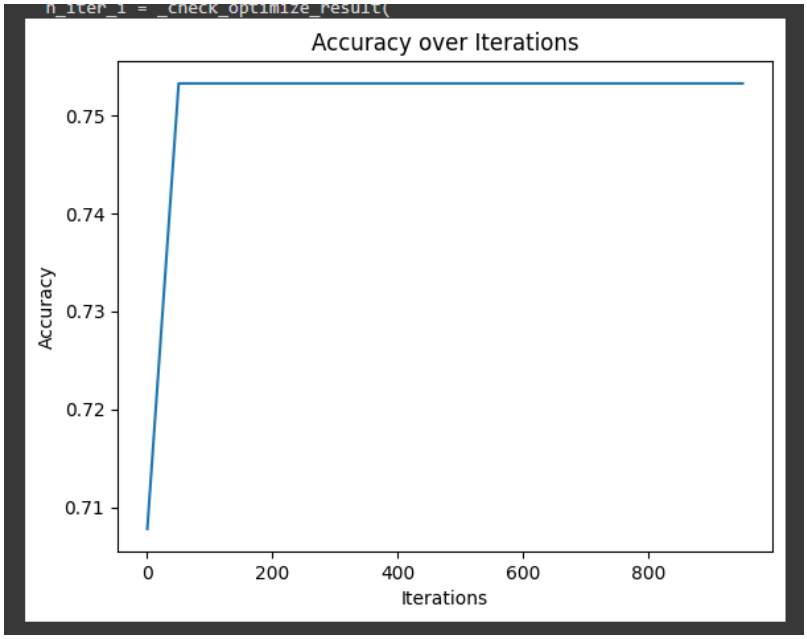
1. Identify the optimum number of K, principal components that achieve the highest classification accuracy.
2. Plot your classification accuracy, precision, and recall over a different number of Ks.
3. Explore different kernel tricks to capture non-linearities within your data. Plot the results and compare the accuracies for different kernels.
4. Compare your results against the logistic regression that you have done in homework 3.

1.
```
Optimum number of principal components: 2
Highest classification accuracy: 0.9912280701754386
```

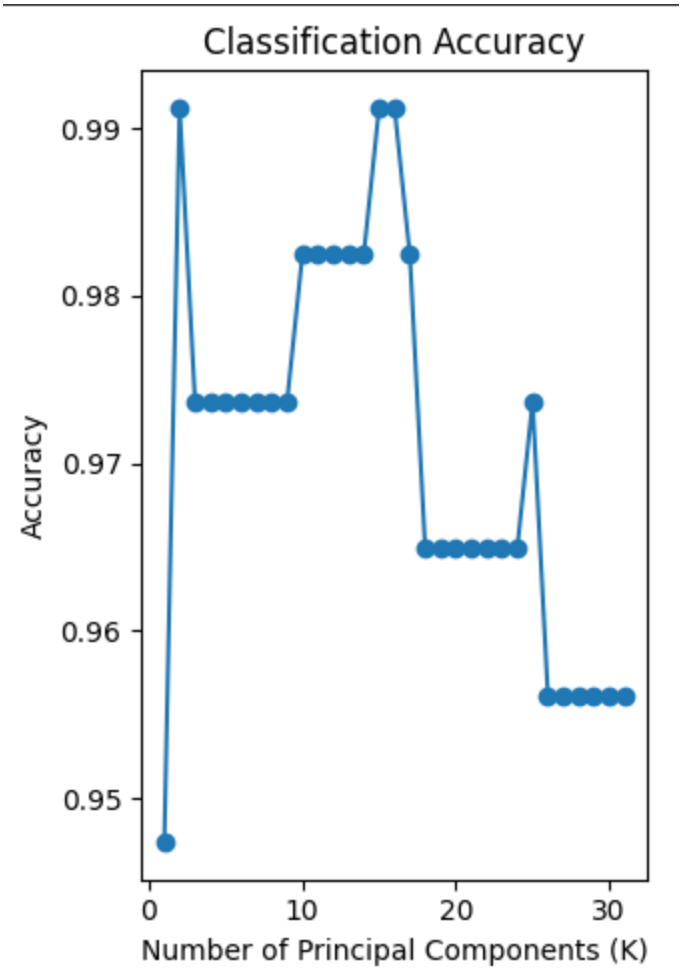
2.


3. 
Kernel: linear
Kernel: poly
Kernel: rbf
Kernel: sigmoid
Accuracy
Number of Principal Components (K)

4. HW3


Accuracy over Iterations
Accuracy
Iterations

HW4



For the plot between the 2 of accuracy for the given graph we can see how the plot runs there are two factors that is intact in these; it can be see the for the 2nd plot the the optimal K value was for the accuracy 0.9912280701754386 and the other plot 0.9473684210526315 (that is but the optimized value) on the homework 3. Leading the kernel linear value to be a better approach.

Problem 2 (50pts):

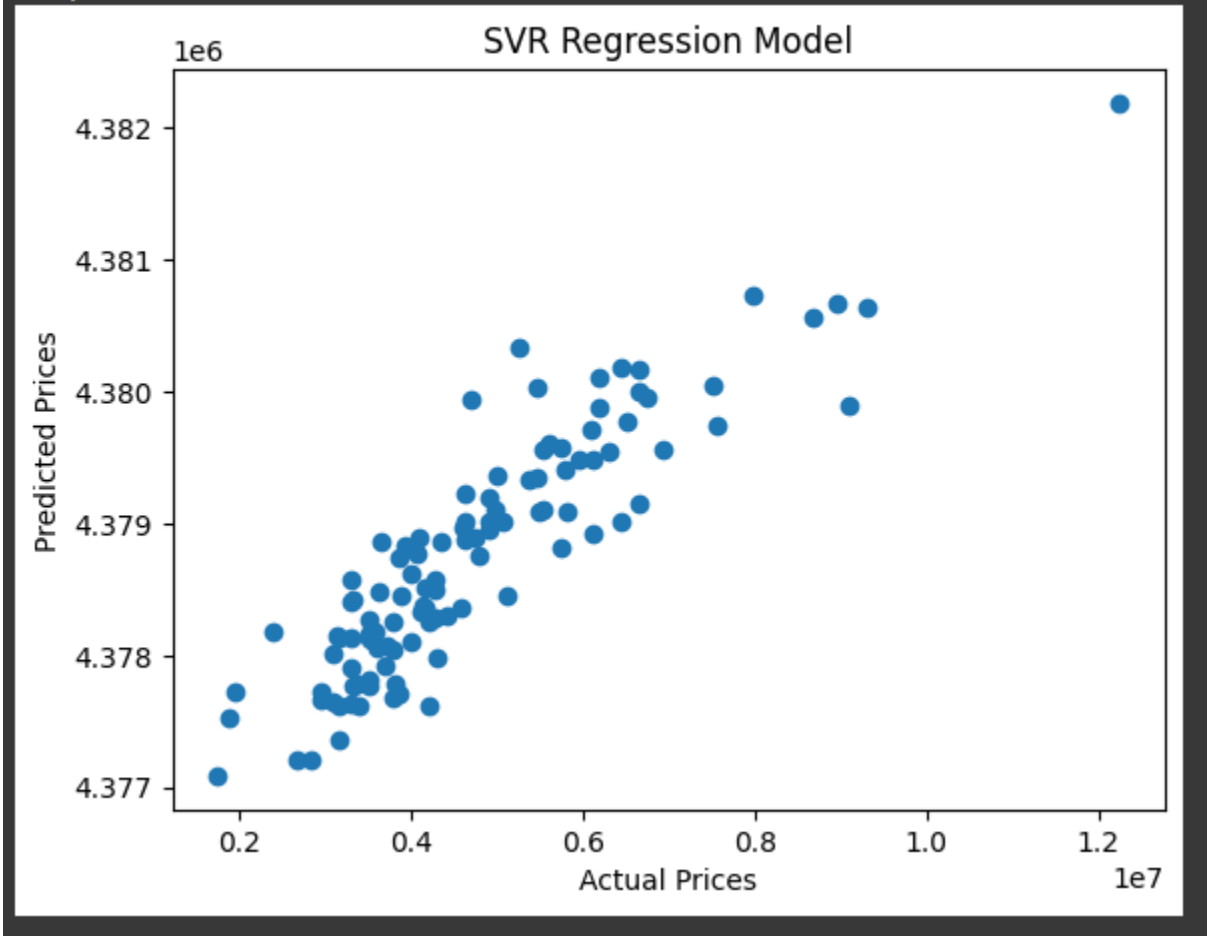
Develop a SVR regression model that predicts housing price based on the following input variables:

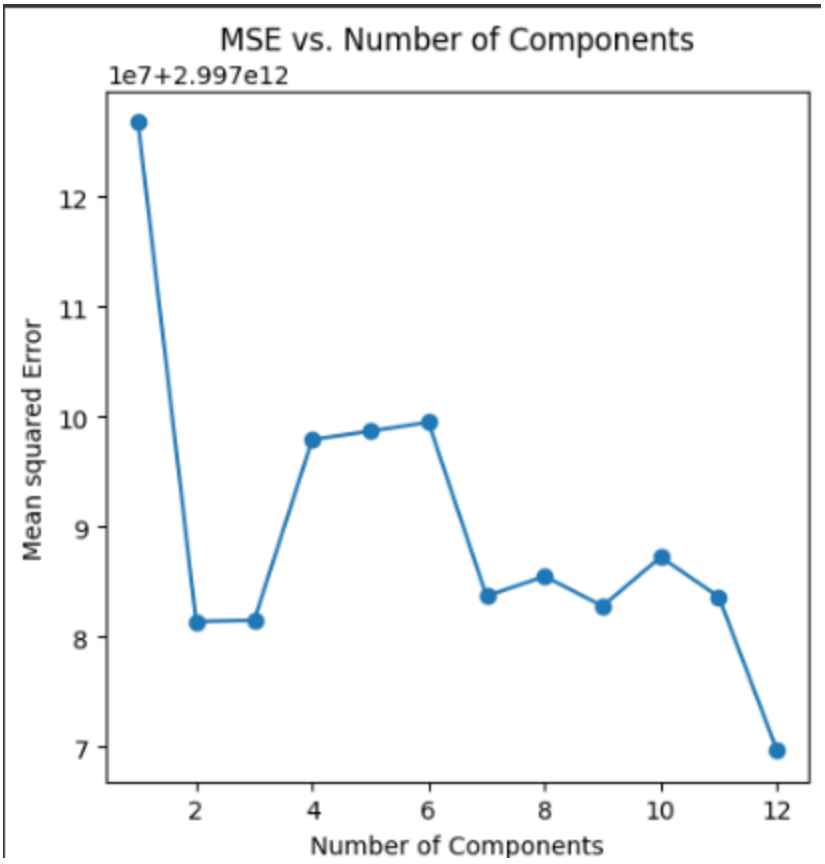
Area, bedrooms, bathrooms, stories, mainroad, guestroom, basement, hotwaterheating, airconditioning, parking, prefarea

1. Plot your regression model for SVR similar to the sample code provided on Canvas.
2. Compare your results against linear regression with regularization loss that you already did in homework1.
3. Use the PCA feature extraction for your training. Perform N number of independent training (N=1, ..., K). Identify the optimum number of K, principal components that achieve the highest regression accuracy.
4. Explore different kernel tricks to capture non-linearities within your data. Plot the results and compare the accuracies for different kernels.

1.

```
Mean Squared Error: 2997270112857.621
R-Squared: -0.04017047124631157
```

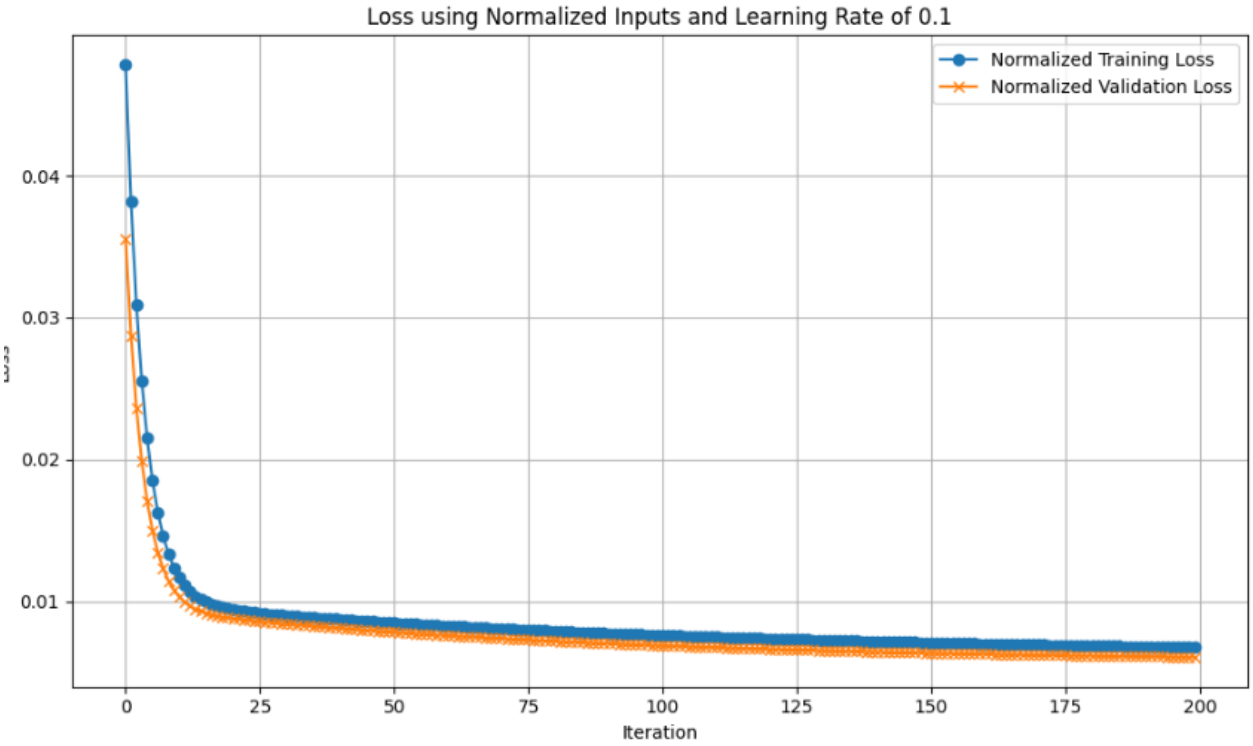



MSE vs. Number of Components
1e7+2.997e12
Mean squared Error
Number of Components
12
11
10
9
8
7
2
4
6
8
10
12

2.

HW4^^^

HW2


Loss using Normalized Inputs and Learning Rate of 0.1
Normalized Training Loss
Normalized Validation Loss
0.04
0.03
0.02
0.01
0
25
50
75
100
125
150
175
200
Iteration

The difference between the 2 one has a gradient descent and the other does not however the HW4 graph does have a similarity to the output with the GD. Leading to the fact to the concept of how if we expand on the number of K values we could get a similar result as a by product.

3.

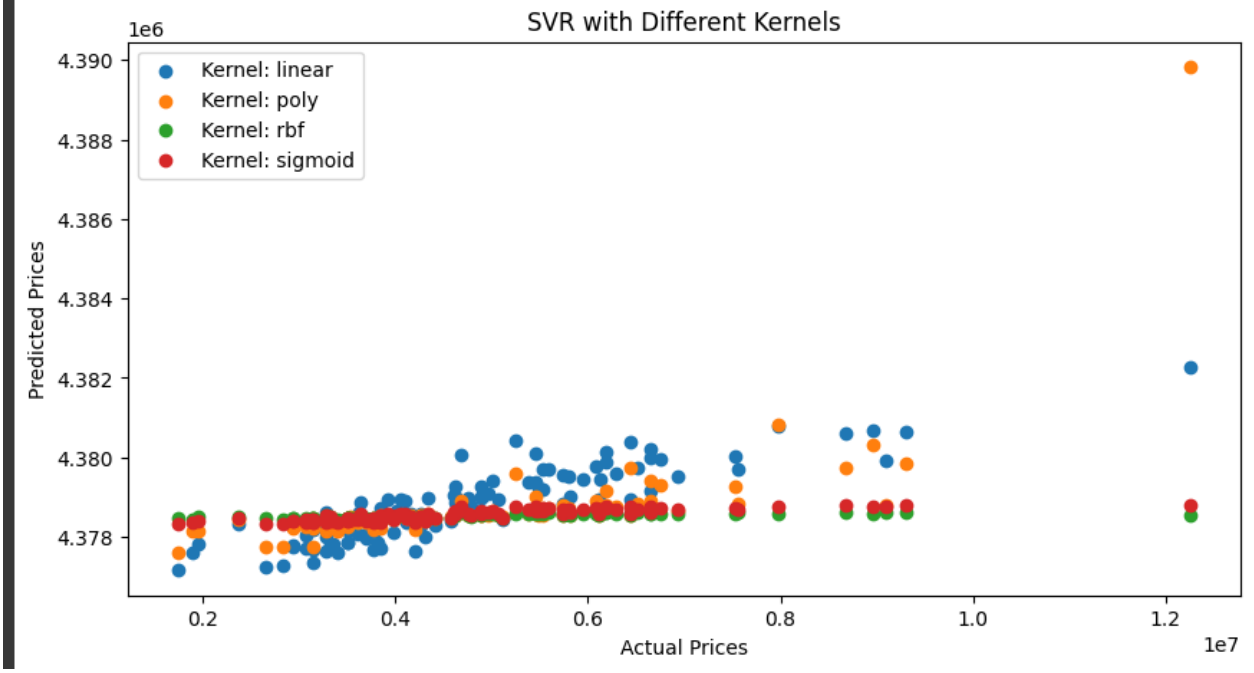
```
Optimum number of components: 12
```

```
Optimum Number of Principal Components: 12
Best R-Squared: -0.04017047124622697
```

I did 2 model to confirm the results and I got 1 as a optimum number of Principal Components: 12 as my optimal;

4.

```
Kernel: linear, R-Squared: -0.040174867073101694
Kernel: poly, R-Squared: -0.040257272942260114
Kernel: rbf, R-Squared: -0.0411545276816907
Kernel: sigmoid, R-Squared: -0.041068444241664004
```



Comparing sigmoid is the best out of the rest because it has a straight line close to the value.