Homework 1

> 張祐嘉 41047055s

Problem 2.7 & 2.8

Data set

- Columns: I named the column as the following: state, index, vote, abbv The columns I will need is the index and vote column, which corresponds to the obesity index and the voting outcome

Preprocessing

Steps in Data Processing:

- Data Cleaning: Both datasets' index column were cleaned to remove percentage symbols and converted to float. Converted voting outcomes(D, R) into numerical formats(0, 1) for modeling.
- Feature Selection: The primary feature used was the obesity index, while the target variable was the voting outcome.

Model Estimation and Coding

- Model complexity range: The K range was set from 1-50
- Resampling error:
    - Since the ouput of the model is categorical, I choose to use 1 - accuracy to represent the resampling error
    - For each k value, we estimate models using LOOCV, getting the predicting results and the mean error rate of that k, after running all k values, I compared the error rates and select the one with the lowest error

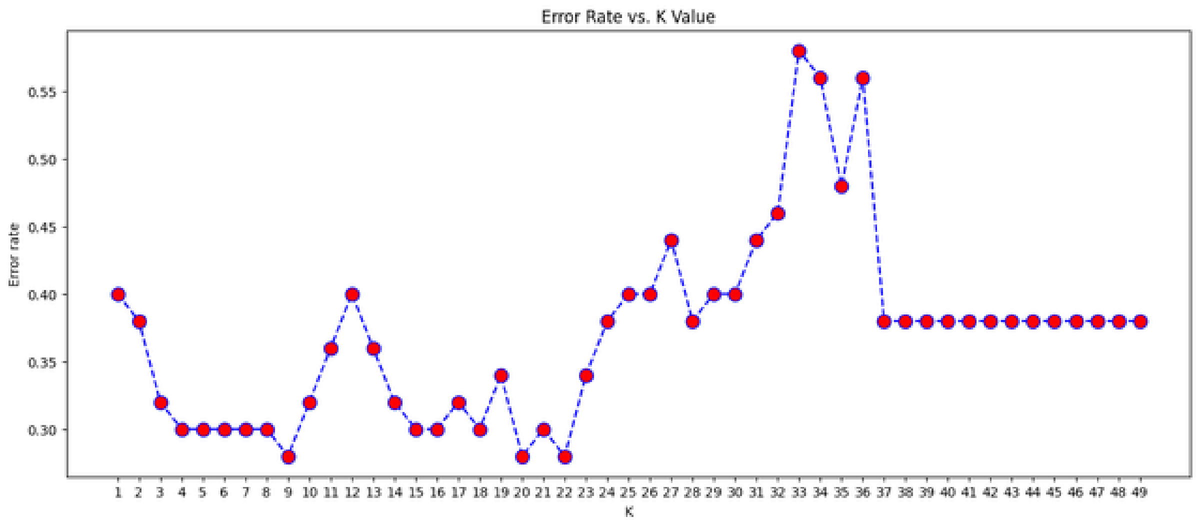
```
for k in k_vals:
    knn = neighbors.KNeighborsClassifier(n_neighbors=k)
    preds = []

    for i ,(train_index, test_index) in enumerate(loo.split(X)):
        X_train, X_test = X[train_index], X[test_index]
        y_train, y_test = y[train_index], y[test_index]

        knn.fit(X_train, y_train)
        pred = knn.predict(X_test)
        preds.append(pred)

    acc = accuracy_score(y, preds)
    acc_scores.append(acc)
```

Results

- Optimal k Value The optimal k was found by comparing the resampling errors, and this value was used to test the model's generalization capability on the 2000 data.

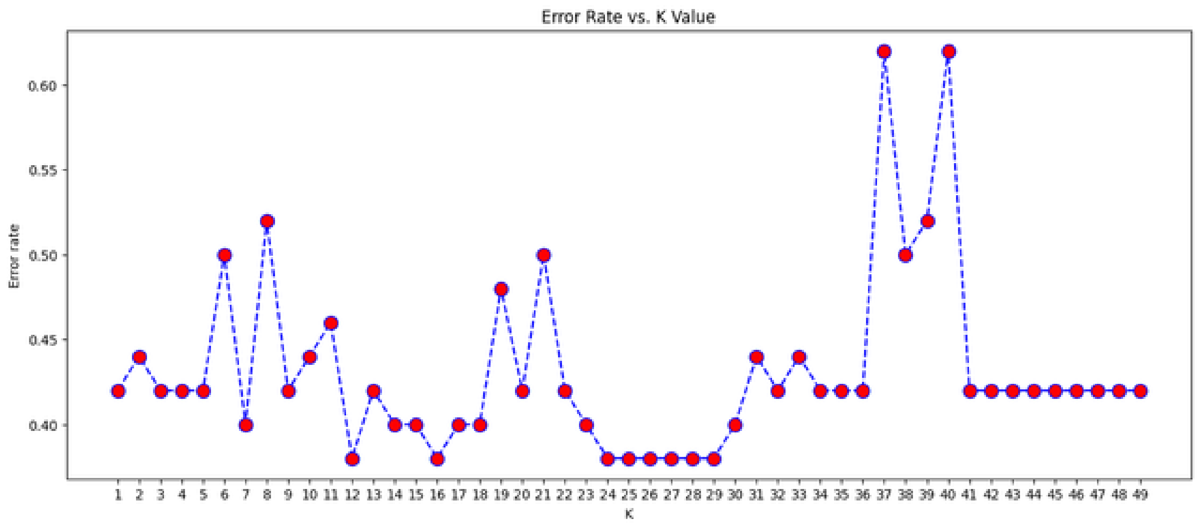
  Problem: Select only odd k(since when tie, will use nearest or random) Use k = 9

  - Problem 2.7: k = 9, 20, 22 have the lowest error rate of 28%



  If were to pick based on Ocams Razor on kNN, select the one with the largest K, since that is the simplest model

  - Problem 2.8: k = 12, 16, 24-29 have the lowest error rate of 38%



- Test Error: After identifying the optimal k, the model was evaluated on the test data to compute the test error, providing an estimate of the model's test error on new data.

- Since there are several lowest resampling error rates, we will test them all and see which has the lowest testing error rate.

  - Problem 2.7: k = 9 with 50% error rate(k=20,22 both got 54%)
    - A 50% error rate indicates that the model misclassified half of the instances in the 2000 test dataset. This suggests that the model may not be generalizing well to unseen data, or there could be insufficient correlation between the feature and the target variable in this dataset.
  - Problem 2.8: 36% error rate(except k = 12 got 38%)
    - On the other hand, using 2000 data as traing data seems to have a lower error rate of 36%, reflecting that it has learned more about the data.
    - And I have no idea why there are so many optimal k value, maybe the dataset is too small?

- Model Quality: The higher test error rate in Problem 2.7 could mean that the obesity index as feature used in this model were not effective in capturing the relationship between input feature and voting outcomes. Same as problem 2.8, but seems like it just happens to have a lower error rate. With only one feature to train model, I think it is quite clear that it isn't sufficient.

Problem 2.11

Data generation

- y = x^2 + 0.1x + noise and the noise has Gaussian distribution N(0, 0.25). The noise has variance 0.25 or standard deviation 0.5.

```
x = np.random.uniform(0, 1, 10)
noise = np.random.normal(0, np.sqrt(0.25), 10)
y = x**2 + 0.1 * x + noise
```

Model complexity

- Since there are only 10 instances in our generated dataset, I have restrained the m of both trigonometric and polynomial model to 8.
- For the parameters of Schwarz criterion
  - DoF: m + 1, where m is the complexity of the model
  - n: 10, sample size
  - Empirical risk: MSE, since this is a regression problem, we will use MSE as the metric to evaluate

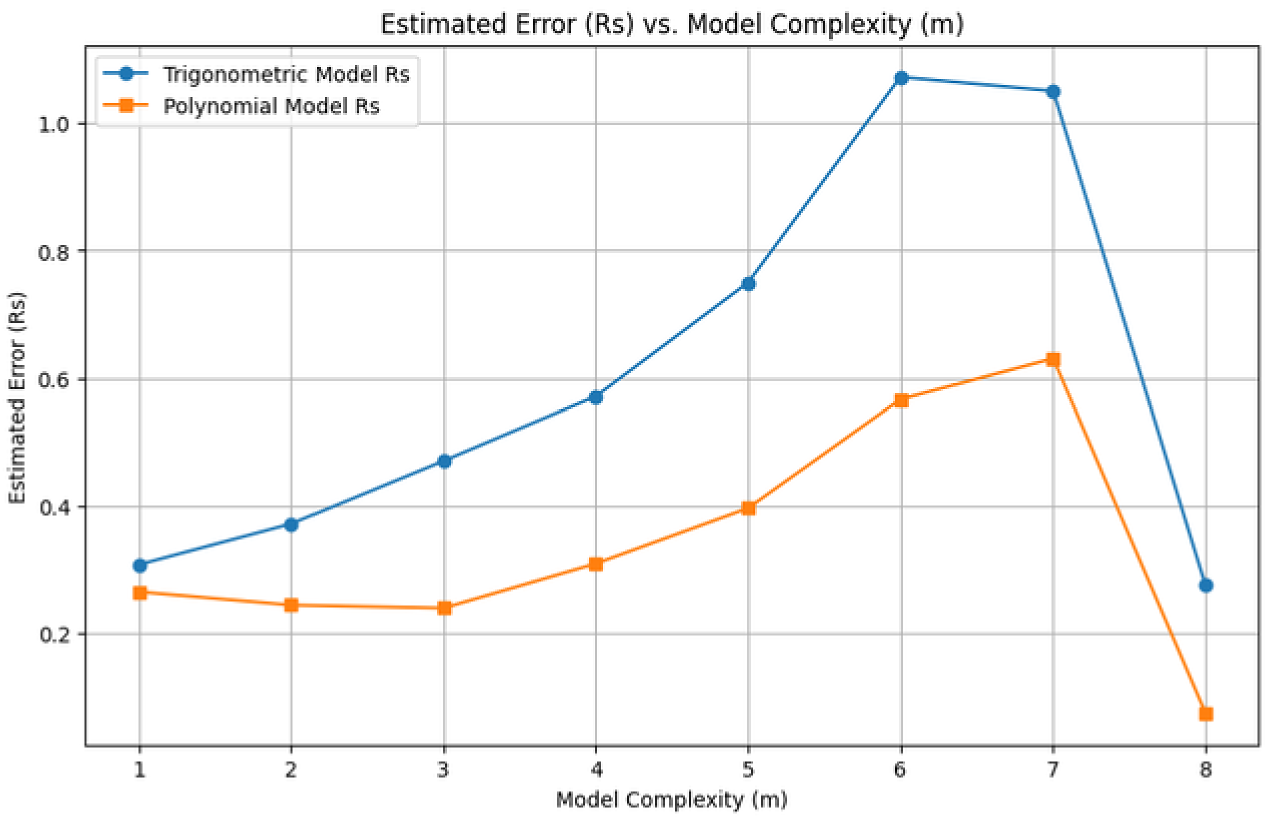
Results

- I only realized that the results vary a lot due to the noise, see Additional
- Trigonometric

m	Empirical Risk (MSE)	Schwarz Criterion	Estimated Error
1	0.194779	1.575646	0.306903
2	0.186832	1.986822	0.371201
3	0.185393	2.535057	0.469981
4	0.172985	3.302585	0.571297
5	0.168368	4.453878	0.749892
6	0.168365	6.372699	1.072941
7	0.102928	10.210340	1.050932
8	0.012647	21.723266	0.274730

- Polynomial

m	Empirical Risk (MSE)	Schwarz Criterion	Estimated Error
1	0.167775	1.575646	0.264354
2	0.122556	1.986822	0.243497
3	0.094154	2.535057	0.238685
4	0.093472	3.302585	0.308699
5	0.088902	4.453878	0.395960
6	0.088893	6.372699	0.566489
7	0.061753	10.210340	0.630518
8	0.003403	21.723266	0.073917



- The graph suggests that the models both have a substantial MSE drop around m=8, which could mean that it starts over-fitting.
- One thing I notice with the above tables is that even though the MSE has decreased a lot, Schwarz criterion still penalizes models with higher complexities hard, leading to the estimated error going up until the MSE is so small that it has a sudden drop.

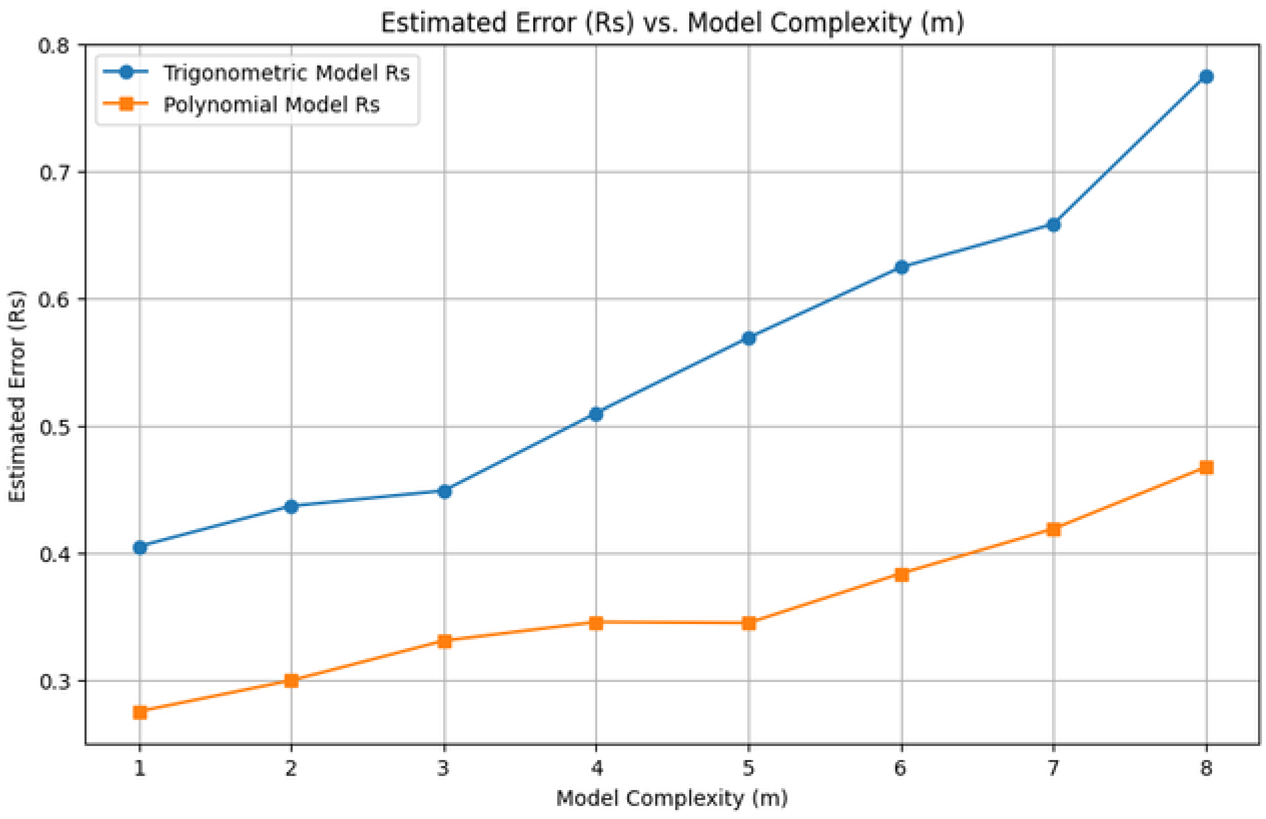
Optimal Value

- Trigonometric: m=1
- Polynomila: m=3

- The reason I picked these values are because they have the lowest estimated risk before the models have the tendency of over-fitting. But I think it aslo depends on the data set size and model complexity, since we only have 10 instances in our dataset here, if I picked the one with the lowest estiamted error as the optimal m value, it would be m=8, but I don't think that would be a good predcitive model.

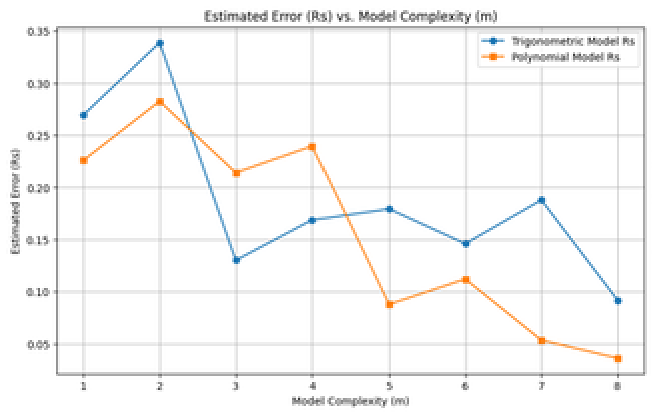
Additional

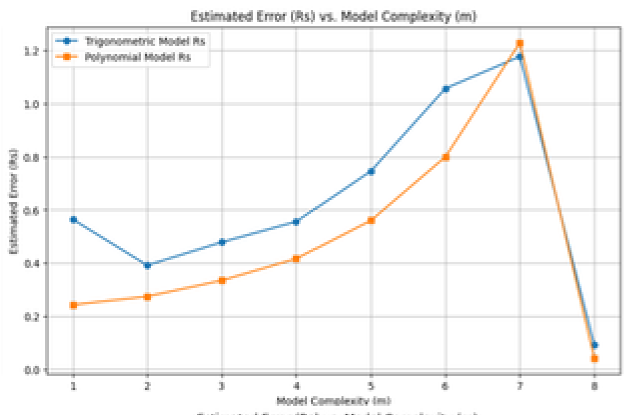
- I realized that the noise seems to vary a lot, so I decided to generated the data 100 times, and see how the models would perform on average:

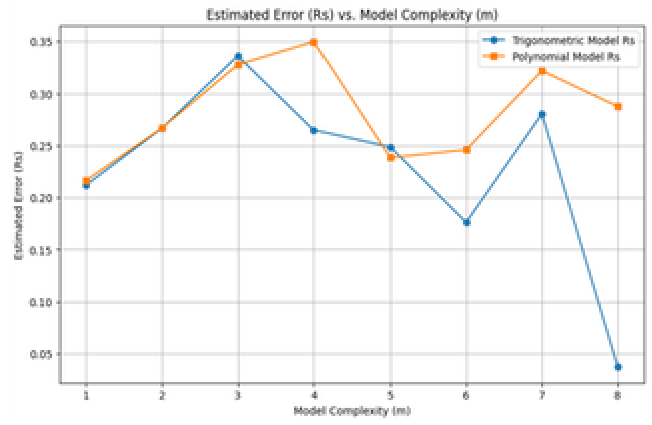


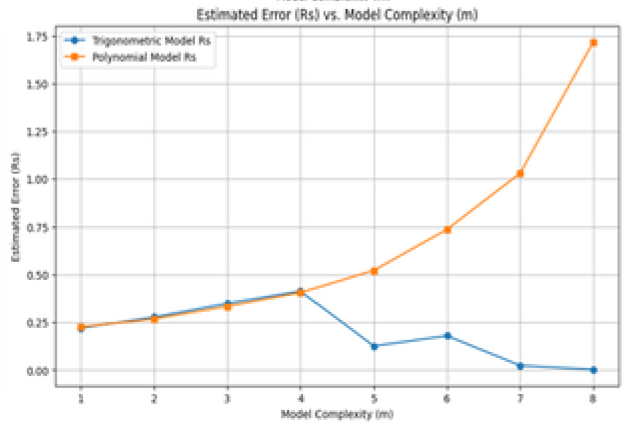
- After doing this for 100 times and averaging the estimated errors, this is the graph I got. It seems like the polynomial one does perform better on estimating the function, and the estimated error seems to have a acending trend as a result of the penalization of Schwarz criterion.

- Here are some graph of the results of the random data:









> - Question Is it possible to choose the best predictive model, trigonometric or polynomial, using only the results of model selection for each method?

- The polynomial one definitely has a lower estimated risk overall, so if we need to pick one model that suits this scernario where we only have 10 noisy data points, I'll say polynomial. If the datasets are larger or even if the model compelxity is larger for one model but has a lower estimated risk, I think we need to take in other considerations as well.

Problem 2.12

Procedure

- I seperated the data into different testing and training datasets with 5-fold cross validation. For each fold I use Schwarz criterion to calculate the estimated risk of different model complexities, after selecting the best model, we predict the testing data and check its prediction accuracy with mse.

K-fold

- Seperate the data using 5-fold, which means for every fold, we will have 8 instances as training data and 2 as testing.

Schwarz criterion

- For each fold we will set the model complexity(m) range from 1 to 7
- Parameters
  - DoF: m + 1
  - n: 8

  - Empirical risk: MSE
- Another thing about the best model selection, since there are so much to compare, 5 folds and 7 models each, I selected the ones with the lowest estimated risk as our best model, even though the best one may be over-fitting.(And seems like this affected the result)

Prediction accuracy

- After selecting the optimal m value of each fold, I estimated models with each folds best m and tested it, obtaining 5 MSE results.

Results

- Trigonometric

Fold	Optimal m	Prediction Accuracy (MSE)
1	1	0.147605
2	1	0.223362
3	6	91907.326815
4	6	835.810471
5	1	0.760901

average MSE: 18548.85383079157

- Polynomial

Fold	Optimal m	Prediction Accuracy (MSE)
1	2	0.948172
2	3	7.551578
3	3	33.896750
4	6	65944808.447251
5	1	0.636960

average MSE: 13188970.296142247

- As the above table shows, the one with optimal m=6 has a terrible prediction accuracy compared with other m values, suggesting that the model complexity is too high and the model is over-fitting, causing it to perform bad on unseen data.