Homework 2

Question 3.1
Using the same data set (credit_card_data.txt or credit_card_data-headers.txt) as in Question 2.2, use the ksvm or kknn function to find a good classifier:
using cross-validation (do this for the k-nearest-neighbors model; SVM is optional); and
splitting the data into training, validation, and test data sets (pick either KNN or SVM; the other is optional).

Methodology

Part (a) - Cross-Validation Approach
- Split the data into two sets - 80% for cross-validation and training (data1), and 20% for testing (data2). I used 10-fold cross-validation on data1 to find the best hyperparameters
- Used kknn function for KNN - used 1 to 49 (odd only to avoid ties) as k, with Euclidean distance and optimal kernel weighting.
- Used ksvm with RBF kernel for SVM, tested C-values from 0.001 to 100, and sigma values from 0.001 to 10.
- After finding the best hyperparameters with cv, I tested the final model (caret automatically retrains on all data1) on data2 to get test accuracy

Part (b) - Regular Validation Approach
- Split the data into three sets - 60% for training, 20% for validation and 20% for testing
- Then I trained SVM models with RBF kernel on the training set for each combination of C and sigma values (used same test values as cv)
- I then predicted on the validation set to find the combination that yields best accuracy
- After finding the combination with best accuracy, I trained a model on the training set and tested on the test set

Results

Part (a)

KNN - best parameter was k = 9 with CV Accuracy 84.33% and Test Accuracy 83.85%

SVM - best parameters were C = 10, sigma = 0.001 with CV Accuracy 85.88% and Test Accuracy 87.69%

Part (b)

SVM - best parameters were C = 1, sigma = 0.01 with Validation Accuracy 89.31% and Test Accuracy 83.97%

Discussion

In part (a), SVM performed better than KNN on both cross-validation and test accuracy. The test accuracies were close to the CV accuracies, showing that CV gave a good estimation (smaller for SVM, bigger for KNN) of how the models would perform on test (unseen) data.

In part (b), the validation accuracy was much higher than the test accuracy (89.31% vs 83.97%) and this gap was not surprising because I chose the model that did best on the validation set and there is some overfitting to that data.

Looking at the results from part a and b, we could see that cross-validation method from part a gave more stable results. This is because cross-validation tests each hyperparameter across all the data through rotating folds (all 10 of them), giving more reliable estimate compared to testing on a single validation set.

Question 4.1
Describe a situation or problem from your job, everyday life, current events, etc., for which a clustering model would be appropriate. List some (up to 5) predictors that you might use.

I am currently consulting for a company in recycling industry and problem lies in sorting efficiency. sorting facility receives material batches from different sources everyday. Each barch has a different mix of materials, some are 90% cardboard, others are heavy in plastics, and some might even have hight contamination. Currently, the facility processes barches in the order they arrive, and sorting staff would constantly work in between materials. By using clustering model, they might be able to group similar batches together so they can work on similar materials at once, instead of processing in the order they arrive.

Some predictors could be

- Source type - residential, commercial, or industrial -> each of them have characteristics (for example commercial offices have more paper and such)
- Weight/Density - heavier batches could indicate dense materials like glass or metals
- Time of week - weekday vs weekend collections might have different compositions

Question 4.2
Use the R function kmeans to cluster the points as well as possible. Report the best combination of predictors, your suggested value of k, and how well your best clustering predicts flower type.

Methodology

- I used only the four predictor variables Sepal.Length, Sepal.Width, Petal.Length, and Petal.Width (not including the response variable, Species) to cluster the iris data using kmeans function
- To find the best combination of predictors, I test all 15 possible combinations (from single predictors, all the way to all four predictors).
- For each combination, I test k values from 2 to 7, and used nstart = 30 to test different starting points.
- Then I compared cluster assignments to the true Species labels to find the accuracy. Cluster comes out in numbers from 1 to 3, and they need to be matched to each species so I can calculate accuracy.

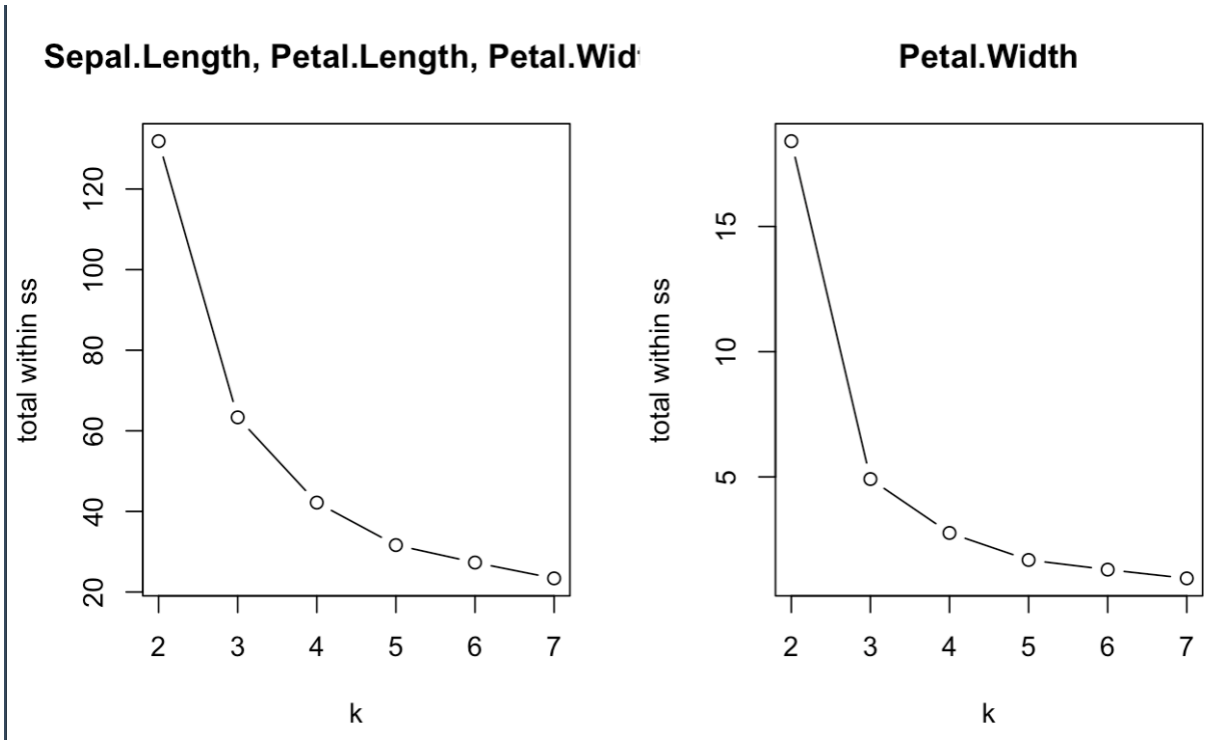
- After, I also created elbow plots to visualize how total within cluster cum of squares decreases as k increases, and observe the elbow (where decrease rate decreases)

Results

Best overall result - Predictors combination [Sepal.Length, Petal.Length, Petal.Width] with k = 6 or 7 gave us 98% accuracy.

Best result with k = 3 (since there are 3 species) - Predictor combinations were [Petal.Width] or [Petal.Length, Petal.Width], both with accuracy 96%

Elbow plot



Discussion

My suggested value of k is 3, which is actually the number of species in the data. It is appropriate for this case and we could see from Petal.Width elbow plot that supports this choice - the decrease in within-cluster sum of squares slows down significantly after k = 3.

For the best combination of predictors, I got Petal.Width alone or Petal.Length and Petal.Width together as tie, both achieving 96% accuracy with k = 3. If I had to suggest one, I would go with just using Petal.Width alone, because from the results, most of the combinations with high accuracy contained Petal.Width which means Petal.Width has great contribution, and using single predictor would be simpler, performing equally well.

From seeing that Sepal.Width alone only achieved 57.3% accuracy, we could assume that iris species are mainly distinguished by their petal characteristics.