API 222 Problem Set 4

Machine Learning and Big Data Analytics: Fall 2024

Donghun Kim

This problem set is worth 30 points in total. To get full credit, submit your code along with a write-up of your answers. This should either be done in R Markdown or Jupyter Notebook, submitted in one knitted PDF.

1. Support Vector Machines (SVMs) (3 pts)

(a) In 1-2 sentences, explain the main idea behind a support vector classifier (SVC) when the classes of the outcome can be perfectly separated by a separating hyperplane. (1 pt)

SVC's main purpose is to find the hyperplane that results in the biggest margin between the classes. The margin is the distance between each point and the hyperplane.

(b) In 2-3 sentences,explain one solution when the classes of the outcome cannot be perfectly separated by a separating hyperplane. (1 pt)

When the classes cannot be perfectly separated, we can use a soft margin classifier. This allows for some points to be on the wrong side of the margin, but penalizes them with a cost parameter. The goal is to minimize the sum of the margin and the penalty for misclassified points.

(c) When does SVM perform better than logistic regression? (1 pt)

SVM performs better than logistic regression when the data is not linearly separable. SVM can use the kernel trick to transform the data into a higher-dimensional space where it is linearly separable, while logistic regression cannot do this.

2. Principal Components Analysis (PCA) (3 pts)

(a) Name and describe the main plot used to determine the optimal number of components in PCA (i.e. what is plotted on each axis). What would be the visual indication of the optimal number of components? (2 pt)

WE use the proportion of variance explained (PVE) by each PC to determine the number of components in PCA. The x-axis would have the number of principal components and the y-axis would have the PVE. In most cases, the optimal number of components will be determined at the elbow of the plot where there seems to be a meaningful decrease in PVE but not a significant decrease after that.

(b) Can you use cross-validation to determine the number of components? Why or why not? (1 pt)

Cross-validation is not typically used to determine the number of components in PCA. This is because PCA is an unsupervised learning technique and does not have a target variable to predict. Cross-validation is used to evaluate the performance of a model on a test set, but PCA does not involve training a model on a test set.

3. Clustering methods (6 pts)

(a) In 2-3 sentences, describe the main differences between K-Means and hierarchical clustering. (2 pt)

K-Means clustering is a partitioning method that assigns each point to the nearest cluster center. It is a fast and scalable algorithm but requires the number of clusters to be specified in advance. Hierarchical clustering, on the other hand, builds a tree of clusters by merging or splitting them based on their similarity. It does not require the number of clusters to be specified in advance but can be computationally expensive for large datasets.

(b) K-Means clustering might find a local optimum but is not guaranteed to find a global optimum. How do you deal with this in practice, to find a good model using K-Means? (2 pt)

To find a good model using K-Means, we can run the algorithm multiple times with different initializations and choose the model with the lowest total within-cluster variation (TWCV).

(c) In the dendrogram below (Figure 1), is 2 closer to 1 or to 10? Why? (1 pt)

2 is closer to 10 because they are grouped together.

(d) Now consider all the points in the dendrogram. Which two points are closest? Why? (1 pt)

2 and 10 are the closest points because the vertical axis represents the distance between points, and the line that connects 2 and 10 is the shortest.

4. Neural Networks (NNs) (4 pts)

(a) In 1-2 sentences, explain the difference between Feed-forward NNs and Recurrent NNs.(1 pt)

Feed-forward neural networks are the most common type of neural network where the connections between nodes do not form a cycle. Recurrent neural networks, on the other hand, have connections that form a cycle, allowing them to retain information over time.

(b) Suppose someone gives you this visual representation of their neural network (Figure 2). Is this a feed-forward or recurrent neural network? (1 pt)

This is a feed-forward network.

(c) State what each of the colored layers represent in the neural network in Figure 2. (2 pt)

Yellow represents the input layer. Blue and Green represent the hidden layers. The red layer represents the output layer.

5. Reinforcement Learning: State-Action Pairs and Decision Making (4 pts)

In reinforcement learning, state-action pairs (s, a) play a central role in determining an agent's decisions. Each pair corresponds to the reward r received by taking an action a in state s. Your goal is to learn the optimal policy $\pi(s)$, which maps each state to the action that maximizes the reward.

Suppose you are in an environment with the following state-action rewards for $s = \{1, 2\}$ and $a = \{0, 1\}$. The reward of an action can either be 10 or 5:

s	a	$P(r = 10\|s, a)$	$P(r = 5\|s, a)$
1	0	0.2	0.8
1	1	0.7	0.3
2	0	0.5	0.5
2	1	0.3	0.7

(a) Expected Reward (2 pts) Calculate the expected reward for each state-action pair (s, a).

r(s,a) = P(r=10|s,a) * 10 + P(r=5|s,a) * 5

r(1,0) = 0.2 * 10 + 0.8 * 5 = 6

r(1,1) = 0.7 * 10 + 0.3 * 5 = 8.5

r(2,0) = 0.5 * 10 + 0.5 * 5 = 7.5

r(2,1) = 0.3 * 10 + 0.7 * 5 = 6.5

(b) Exploration-Exploitation Trade-off (2 pts) Explain the concept of the exploration-exploitation trade-off in reinforcement learning.

The exploration-exploitation trade-off refers to the balance between exploring new actions to learn more about the environment (exploration) and exploiting known actions to maximize rewards (exploitation). Agents need to decide whether to try new actions to gather more information or stick to actions that have yielded high rewards in the past.

Data Questions

In this homework, you will learn how to perform k-means clustering on a dataset of gene expression levels for different tumor types. The dataset we will use is the **Khan** dataset from the `ISLR2` package. By the end of this assignment, you will have implemented k-means clustering, evaluated its performance, and visualized the results.

Part 1: Understanding the Dataset

1. Load the Dataset (0.5 pts)

The **Khan** dataset contains gene expression levels for multiple genes (`Khan$xtrain`) and the corresponding tumor types (`Khan$ytrain`).
- **Task**: Load the dataset using the `data()` function and examine its structure using the `dim()` function to check the number of rows and columns in the training data (`Khan$xtrain`).

```
# install.packages("ISLR2")
library(ISLR2)
data("Khan")

print(dim(Khan$xtrain)) # 63 * 2308
```

```
## [1]   63 2308
```

```
# this means there are 63 observations and 2308 features
# print(length(Khan$ytrain)) # this has 63 observations
```

Q2. Describe the Dataset (0.5 pts)

The dataset includes expression levels for over 2000 genes for 63 training observations. The labels (`Khan$ytrain`) indicate the tumor type for each observation.
- **Question**: How many tumor types are there in `Khan$ytrain`?

```
# Count the unique tumor types in Khan$ytrain
unique_tumor_types <- unique(Khan$ytrain)
print(length(unique_tumor_types))
```

```
## [1] 4
```

Part 2: Preprocessing

Q3. Normalize the Data (1 pts)

For k-means clustering, we need to normalize the data so that all features have equal weight. Use the `scale()` function to normalize the gene expression data.
- **Task**: Normalize `Khan$xtrain` by applying `scale()` to it. This standardizes each feature to have a mean of 0 and a standard deviation of 1.

```r
# Normalize the gene expression data
Khan$xtrain_norm <- scale(Khan$xtrain)
```

Part 3: K-Means Clustering

Q4. Run K-Means (2 pts)

Perform k-means clustering on the normalized data. Use `k = 4`, since there are 4 tumor types in the dataset. Set a random seed to 222 for reproducibility.
- **Task**: Use the `kmeans()` function to cluster the data into 4 groups. Store the result in a variable.

```r
# Set the random seed for reproducibility
set.seed(222)
result_kmeans <- kmeans(Khan$xtrain_norm, centers = 4)
```

Q5. Compare Clustering Results (1 pts)

Compare the cluster assignments to the actual tumor types in `Khan$ytrain`.

```r
cluster1 <- Khan$ytrain[result_kmeans$cluster == 1]
cluster2 <- Khan$ytrain[result_kmeans$cluster == 2]
cluster3 <- Khan$ytrain[result_kmeans$cluster == 3]
cluster4 <- Khan$ytrain[result_kmeans$cluster == 4]

# get the dominant tumor type in each cluster
for (i in 1:4) {
  cluster <- Khan$ytrain[result_kmeans$cluster == i]
  df <- table(cluster)
  # the dominant
  print(paste("The dominant tumor type in Cluster", i, "is", names(df)[which.max(df)], ", and it compris
  for (j in 1:length(df)) {
    print(paste("In Cluster", i, ":", "there are a total of ", df[j], "tumor types of ", names(df)[j]))
  }
  print("-----------------------------------------------------")
}
```

```
## [1] "The dominant tumor type in Cluster 1 is 3 , and it comprises  0.5 out of  18 observations."
## [1] "In Cluster 1 : there are a total of  4 tumor types of  1"
## [1] "In Cluster 1 : there are a total of  1 tumor types of  2"
## [1] "In Cluster 1 : there are a total of  9 tumor types of  3"
## [1] "In Cluster 1 : there are a total of  4 tumor types of  4"
## [1] "-----------------------------------------------------"
## [1] "The dominant tumor type in Cluster 2 is 2 , and it comprises  0.714285714285714 out of  7 obser
## [1] "In Cluster 2 : there are a total of  5 tumor types of  2"
## [1] "In Cluster 2 : there are a total of  2 tumor types of  4"
## [1] "-----------------------------------------------------"
## [1] "The dominant tumor type in Cluster 3 is 2 , and it comprises  0.529411764705882 out of  17 obse
## [1] "In Cluster 3 : there are a total of  9 tumor types of  2"
## [1] "In Cluster 3 : there are a total of  8 tumor types of  4"
## [1] "-----------------------------------------------------"
## [1] "The dominant tumor type in Cluster 4 is 2 , and it comprises  0.380952380952381 out of  21 obse
```

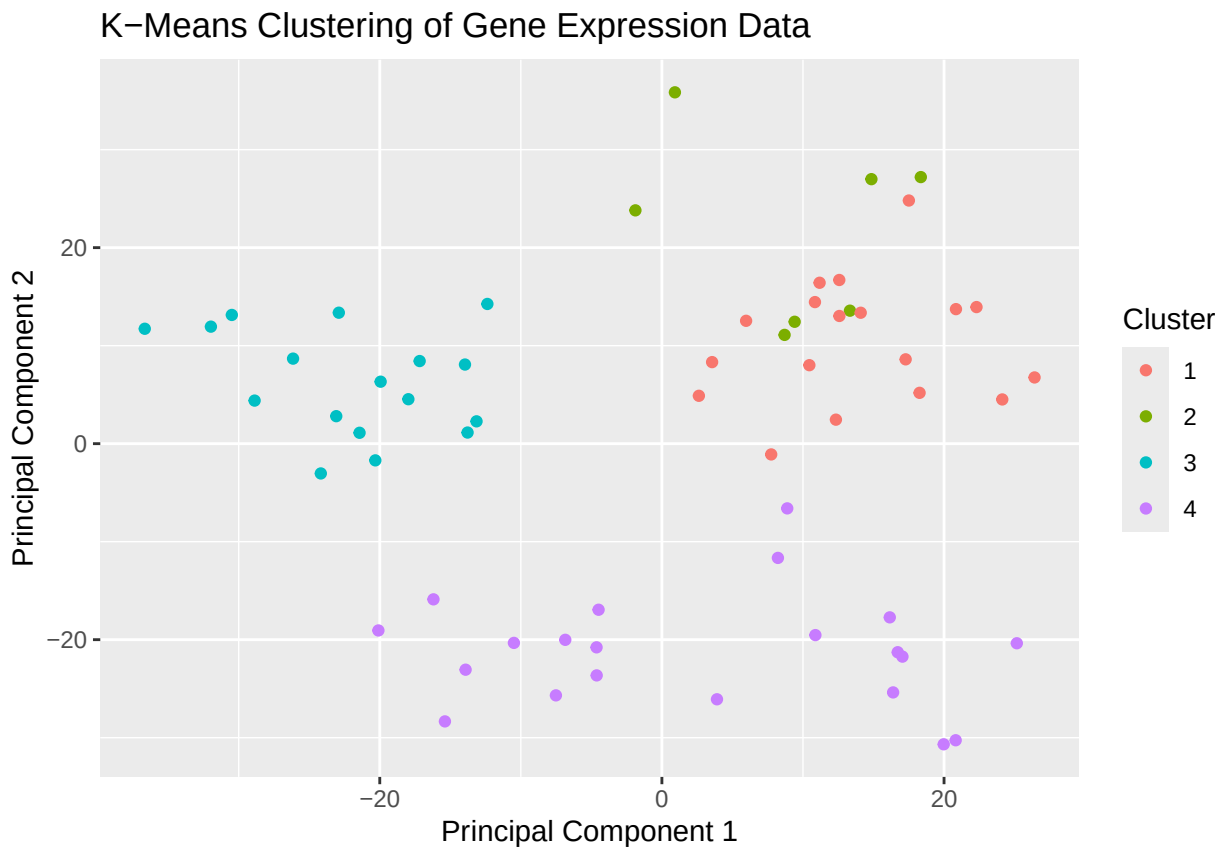
```
## [1] "In Cluster 4 : there are a total of  4 tumor types of  1"
## [1] "In Cluster 4 : there are a total of  8 tumor types of  2"
## [1] "In Cluster 4 : there are a total of  3 tumor types of  3"
## [1] "In Cluster 4 : there are a total of  6 tumor types of  4"
## [1] "-------------------------------------------------"
```

Q6. Visualize the Clusters (2 pts)

Use Principal Component Analysis (PCA) to reduce the data to two dimensions and plot the clusters.
- **Task**: Perform PCA on the normalized data and create a scatter plot of the first two principal components. Color the points based on their k-means cluster assignments.

```r
# Perform PCA on the normalized data
library(ggplot2)
pca_result <- prcomp(Khan$xtrain_norm)
# summary(pca_result)
pc1 <- pca_result$x[, 1]
pc2 <- pca_result$x[, 2]

#print(result_kmeans$cluster)
#print(paste(dim(pc1), dim(pc2), dim(cluster)))
#print(length(pc1))
#print(length(pc2))
# Create a scatter plot using pc1 and pc2
plot_data <- data.frame(pc1, pc2, cluster = as.factor(result_kmeans$cluster))
ggplot(plot_data, aes(x = pc1, y = pc2, color = cluster)) +
  geom_point() +
  labs(title = "K-Means Clustering of Gene Expression Data",
       x = "Principal Component 1",
       y = "Principal Component 2",
       color = "Cluster")
```



Part 4: Reflection (3 pts)

Q7. Interpret the Results

Answer the following questions based on the k-means results: 1. How well do the k-means clusters align with the actual tumor types?
2. What might explain any mismatches between the clusters and the tumor types?
3. How could clustering techniques like this be useful in healthcare or policy applications?

1. The k-means clusters do not align perfectly with the actual tumor types, as there are mismatches between the clusters and the tumor types.
2. The mismatches could be due to the complexity of gene expression data and the presence of overlapping patterns across different tumor types.
3. Clustering techniques like k-means can be useful in healthcare to identify subgroups of patients with similar gene expression profiles, which can help in personalized treatment strategies and disease diagnosis. In policy applications, clustering can be used to identify patterns in population health data to inform public health interventions and resource allocation.