STATISTICAL LEARNING (AMI22T)

HOME EXERCISE - 2

Balaji Vijayaraj

v22balvi@du.se

TASK - 1

Introduction:

In this task, we are working with a bike rental dataset that contains information about bike rentals for approximately 11,000 individuals. The dataset consists of 12 variables, including a target variable, which provides valuable insights into weather conditions, seasons, and other factors related to bike rentals. The main objective is to develop regression algorithms to predict the number of bikes rented within each time slot. We have access to a training dataset that includes the target variable for model building, and a testing dataset where the target variable has been removed. Our task is to develop at least three models, including two based on Support Vector Machines (SVM) and a decision tree-based algorithm and one with linear regression. These models will be trained using the training dataset and evaluated using the testing dataset to assess their predictive performance. We will then analyze and discuss the consistency of predictions among the three algorithms, providing valuable insights for bike rental companies to optimize their operations and better understand the factors influencing bike rental demand.

Method:

1. **Loading the Data:**
   I started by loading the training and testing datasets using the read.csv() function. The training dataset is stored in the train_set variable, and the testing dataset is stored in the test_set variable.

2. **Data Preprocessing:**
   In this step, I performed some necessary data preprocessing tasks on the combined dataset (full_set). First, combine the relevant columns from the training and testing datasets to create the train_set variable. Then, I converted the weather variable to a factor variable using the as.factor() function. Then again I split the dataset.

3. **Feature Engineering:**
   I extracted additional temporal features from the datetime variable in both the training and testing datasets. Specifically, I extracted the year, month, day, hour, and minute using the year(), month(), day(), hour(), and minute() functions, respectively. I also converted these newly created variables to factor variables using the as.factor() function.

4. **Removing Unnecessary Variables:**
   I removed unnecessary variables from both the training and testing datasets to simplify the analysis. These variables include the datetime variable and some redundant variables that were created during the feature engineering process.

Random Forest model

I used Random Forest for this task because it offers a reliable and flexible approach for regression tasks. Its ability to handle non-linear relationships, high-dimensional data, and outliers are major reason for this. Also, I tried bagging and boosting, but random forest gave me better results.

1. **Setting the Seed:**
   To ensure the reproducibility of results, the random seed was set to 123.

2. **Hyperparameter Tuning:**
   The 'tune' function was employed to find the optimal values for the hyperparameters 'ntree' and 'mtry' of the randomForest algorithm. The parameter ranges considered were ntree = 100, 200, 300, 400, 500 and mtry = 3, 6, 8, 10, 12.

3. **Model Training:**
   The randomForest algorithm was trained on the training dataset (train_set) using the best hyperparameters identified from the tuning process (ntree = 500, mtry = 10).

4. **Prediction:**
   The trained random forest model (rf_model) was used to predict the target variable for the testing dataset (test_set). The predictions were stored in the 'rf_pred' column.

5. **Handling Negative Predictions:**
   A check was performed to identify if any negative values were present in the 'rf_pred' column. If any negative values were found, they were replaced with 0, ensuring the predictions are meaningful.

SVM (Support Vector Machine) model with the Radial kernel

I chose to use Support Vector Machines (SVM) with a radial kernel for my regression task for several reasons. Firstly, the radial kernel is known for its ability to handle non-linear relationships between variables, which is the case in this regression problem where the target variable does not have a linear relationship with the predictors.

Additionally, the radial kernel is well-suited for handling high-dimensional data, which is common in regression tasks where multiple predictors are involved. Furthermore, the radial kernel is robust to outliers, which can be a challenge in regression tasks where extreme points can heavily influence the model's performance.

Methods are like the previous model except for the 2 steps,

1. **Hyperparameter Tuning:**
   The 'tune' function was used to find the optimal values for the hyperparameters 'cost' and 'gamma' of the SVM model. The tuning was performed with the Radial kernel and the parameter ranges considered were cost = 0.1, 1, 5, 10, 100 and gamma = 0.001, 0.01, 0.1, 1, 10.

2. **Model Training:**
   An SVM model with the Radial kernel was trained on the training dataset (train_set) using the best hyperparameters obtained from the tuning process (cost = 10, gamma = 0.1).

Linear Regression model

After exploring different regression techniques such as lasso regression and generalized additive models (GAM), and experimenting with encoding categorical variables, I found that none of these approaches outperformed linear regression. Therefore, I decided to continue with linear regression and made some modifications to improve the model.

To begin, I examined the correlation between the predictors and discovered a high correlation of 0.97 between month and season, as well as a correlation of 0.98 between atemp and temp. Consequently, I decided to remove temp and season from the dataset to avoid multicollinearity.

Next, I employed backward selection to identify the most influential predictors for the model. After careful evaluation, I selected windspeed, humidity, weather, atemp, month, year, and hour as the key predictors.

By making these adjustments and refining the selection of predictors, I aimed to enhance the accuracy and predictive power of the linear regression model.

Methods are like the previous model except for the 2 steps,

1. **Training Control Parameters:**
   The training control parameters for cross-validation were defined using the 'trainControl' function. The method chosen was "cv" (cross-validation), and the number of folds was set to 10.

2. **Model Training:**
   The Linear Regression model was trained using the 'train' function with cross-validation. The formula used for training the model includes the above-mentioned predictors. These variables collectively explained approximately 70% of the variation in the response variable. The training was performed on the training dataset (train_set) with the specified formula, method as "lm" (Linear Regression), and the training control parameters defined earlier.

Discussion:

I checked the consistency between the algorithms by checking the mean of difference between the predicted count variable.

The average difference between Random Forest and SVM with Radial Kernel predictions: 26.96556

The average difference between Random Forest and Linear Regression predictions: 53.21993

The average difference between SVM with Radial Kernel and Linear Regression predictions: 60.49403

The average differences between the predictions of the algorithms show how similar or different their results are. A lower average difference means that the predictions are more similar, while a higher average difference means there is more variation.

Based on the results, we can say that the predictions of Random Forest and SVM with Radial Kernel are quite similar, with an average difference of around 26.97. This means that these two algorithms tend to give similar predictions for the target variable. On the other hand, the predictions of Random Forest and Linear Regression are more different, with an average difference of approximately 53.22. This indicates that the predictions from these two algorithms are not as consistent. Similarly, the predictions of SVM with Radial Kernel and Linear Regression have a higher average difference of about 60.49, showing less agreement between them.

TASK - 2

Introduction:

Clustering is a technique used to group similar data points together based on their characteristics, without any predefined labels. In this task, we will apply clustering to a combined dataset obtained by merging the training and testing files without the target variable. The dataset contains information from around 11,000 individuals and includes variables such as weather conditions, season, and day of the week. Our goal is to create clusters that effectively separate the dataset into homogeneous groups. We will analyze whether these clusters can effectively separate the data based on the target variable, which represents the number of bikes rented within each time slot.

Method:

1. Data Preprocessing:

To perform clustering on the dataset, I followed the following steps. First, I loaded the training and testing datasets using the read.csv() function. The training dataset was stored in the "train_set" object, and the testing dataset was stored in the "test_set" object.

Next, I combined the two datasets into one using the rbind() function. The combined dataset, "full_set", contained information from both the training and testing datasets.

I extracted relevant columns from the "full_set" dataset by selecting columns 1 to 10 using the indexing notation. This subset of columns contained the necessary variables for clustering.

I then created additional features by extracting the year, month, day, hour, and minute from the "datetime" column of the "full_set" dataset using the year(), month(), day(), hour(), and minute() functions, respectively. These additional features would provide more information for clustering.

After creating the additional features, I removed the "datetime" column from the "full_set" dataset as it was no longer needed.

2. One-hot encoding:
After creating separate datasets for numeric and categorical variables, I performed one-hot encoding on the categorical variables. Following that, I merged the encoded categorical variables with the numeric variables, resulting in a combined dataset. With the categorical variables now encoded, they can be treated as numeric variables in further analysis.

3. Principal Component Analysis:
After scaling all the variables in the combined dataset, I performed Principal Component Analysis (PCA) on the scaled data. PCA allows for dimensionality reduction by transforming the original variables into a new set of uncorrelated variables called principal components. In this case, I selected the first 52 components as they collectively explain 90% of the total variance in the dataset.

4. Clustering:
I choose kmeans clustering because the algorithm aims to minimize the within-cluster sum of squares, making it intuitive to understand and interpret the resulting clusters. Also, it allows for greater control over the number of clusters through the predefined "k" parameter. I also used hierarchical clustering with complete linkage to verify if the clustering is based on target variable.

After analyzing the elbow plot and considering the options of 4 or 11 clusters, I decided to proceed with 4 clusters. This choice was motivated by the simplicity and interpretability it offers compared to using 11 clusters. With 4 clusters, the resulting groups are more distinct and provide better separation of data points.

To further analyze the clusters, I added the cluster number as an additional variable to the original dataset. This allows for easy identification and analysis of the data points within each cluster. Next, I divided the dataset into separate training and testing sets. The purpose of this division is to verify if the clusters are indeed separating the data based on the target variable, which in this case is the "count" variable.

After splitting the dataset, I filtered the data based on the assigned clusters. This enables a more focused analysis within each cluster, examining the patterns, characteristics, and relationships specific to each group.

Results and Discussion:

Boxplot of Clusters vs Count

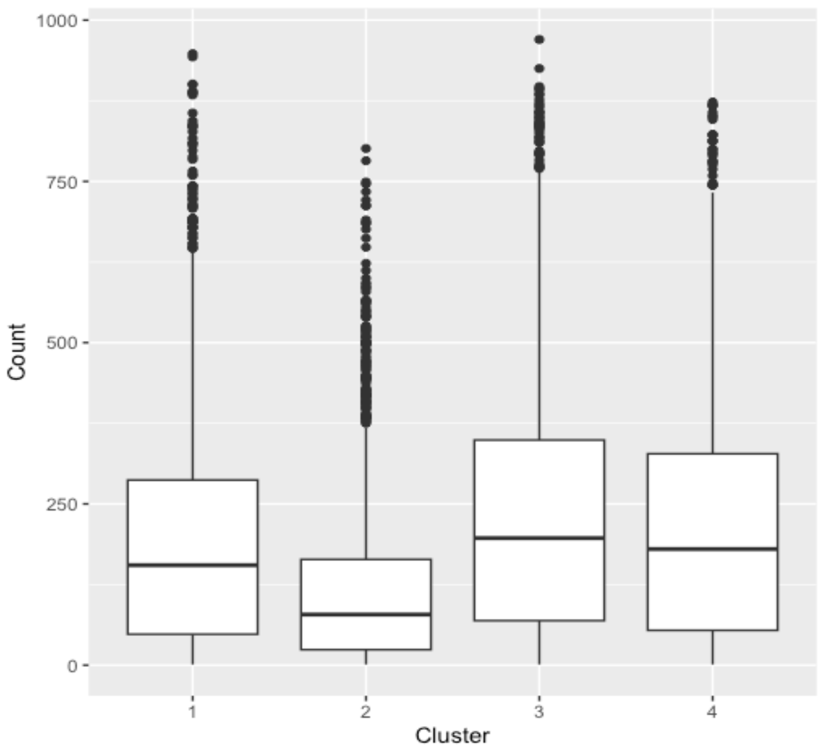

Figure 1

Based on the information provided by the boxplot from Figure 1, it is difficult to distinguish the clusters based on the count variable alone. Only cluster 2 exhibits an average count of 117, which differs from the other clusters. However, clusters 3 and 4 display similar average counts. Consequently, we can conclude that the clusters are not distinctly separated based on the target variable.

Boxplot of Clusters vs Season

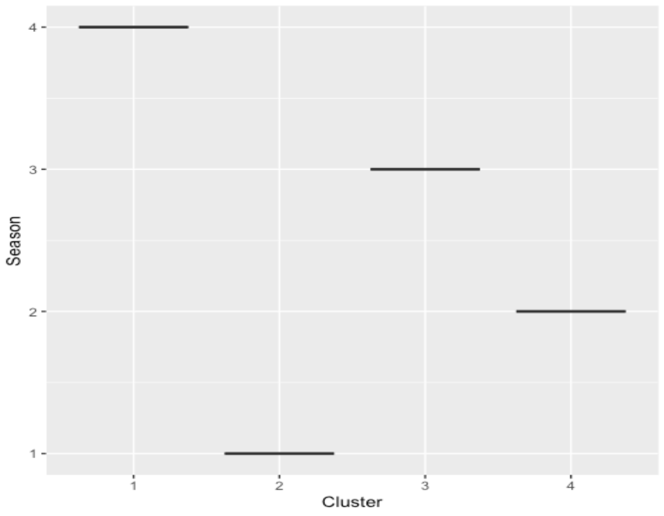

Figure 2

One noteworthy observation from figure 2 is that the clusters are clearly separated based on the season. The above plot clearly indicates the distinction between clusters based on the seasons.

Boxplot of Clusters vs Month

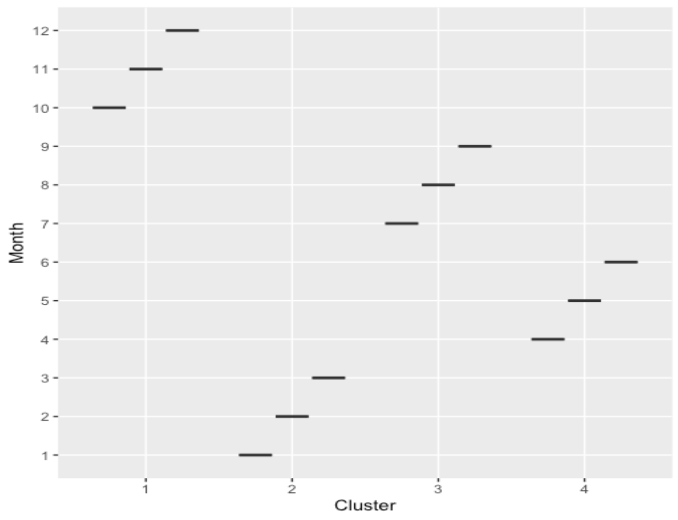

Figure 3

Figure 3 demonstrates a distinct separation of the clusters based on the months. It is evident that there is a similarity between the plot based on months and the plot based on seasons.

The clusters can be identified as follows:

Cluster 1: Months 10, 11, and 12, corresponding to Season 4.
Cluster 2: Months 1, 2, and 3, corresponding to Season 1.
Cluster 3: Months 7, 8, and 9, corresponding to Season 3.
Cluster 4: Months 4, 5, and 6, corresponding to Season 2.

Boxplot of Clusters vs Temperature

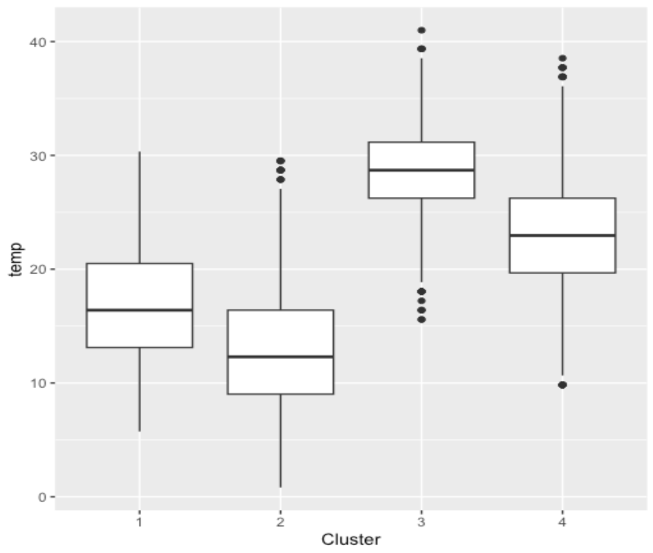

Figure 4

Figure 4 provides a clear indication that the clusters exhibit distinct differences in terms of temperature. Each cluster demonstrates a noticeable disparity in the mean temperature. However, it is worth noting that the distribution of temperature appears similar in clusters 1 and 2, while it is completely different in clusters 3 and 4.

Boxplot of Clusters vs humidity in Hierarchical clustering

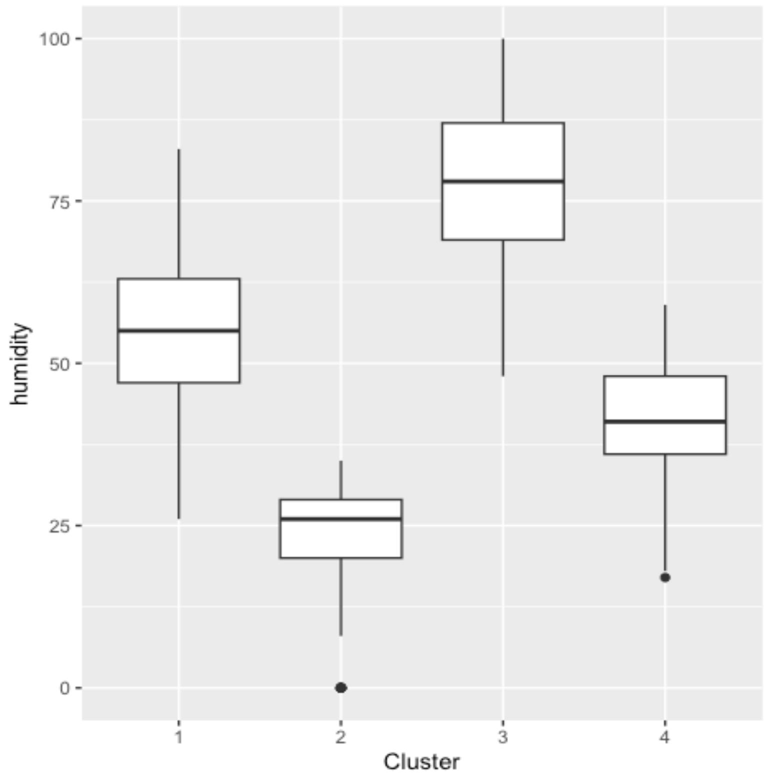

Figure 5

When I used hierarchical clustering the dataset clustered based on the humidity as shown in Figure 5, but not based on the target variable. Also, this method did not cluster based on the season.

Conclusion:

Based on the previous analysis, it is evident that the clusters do not exhibit separation based on the target variable (count). However, they do demonstrate distinct separation based on the season and month variables in K-means clustering and humidity in hierarchical clustering with complete linkage. Therefore, the clusters can be effectively differentiated by considering the seasonal, monthly and humidity patterns rather than the target variable.

Clustering does not explicitly consider the target variable during the clustering process. Instead, they focus on finding patterns and similarities within the feature space. Therefore, it is not guaranteed that the resulting clusters will align with the target variable.