ASSIGNMENT 3

The Vehicles_Sales.csv dataset comprises detailed records of sales transactions, capturing a range of variables that provide insights into order performance and product sales. It includes information on order specifics, such as quantities and pricing, as well as metadata about the order's timing and status. By analyzing this dataset, one can gain a comprehensive understanding of sales patterns, evaluate product performance, and assess the overall effectiveness of sales strategies. Below is a description of each column in the dataset:

- **ORDERNUMBER**: Unique identifier for each order. This helps in tracking and referencing specific orders.
- **QUANTITY ORDERED**: The number of units ordered for each product. It reflects the volume of products purchased in each order.
- **PRICEEACH**: The price per unit of the product. This indicates the cost of one unit before any discounts or additional charges.
- **ORDERLINENUMBER**: The line item number within an order. Each order can have multiple line items, and this column identifies each line item uniquely within its order.
- **SALES**: Total sales amount for each line item.
- **STATUS (Target Variable)**: The current status of the order. It may include values such as 'Shipped'(1), and 'Cancelled'(0), representing the order's processing stage.
- **QTR_ID**: Quarter identifier for the order. It shows the quarter of the year in which the order was placed (e.g., Q1, Q2, Q3, Q4).
- **MONTH_ID**: Month identifier for the order. This column represents the month during which the order was placed (e.g., January, February, etc.).
- **YEAR_ID**: Year identifier for the order. It indicates the year in which the order was placed.
- **PRODUCTLINE**: Category or line of the product ordered. It provides information about the type or group of products.
- **MSRP**: Manufacturer's Suggested Retail Price. This is the recommended selling price of the product from the manufacturer's perspective.
- **DEALSIZE**: The size of the deal, which might indicate whether it was a small, medium, or large deal, potentially reflecting the total value or quantity of the order.

Useful/relevant libraries

library(class), library(caret), library(tinytex), library(e1071), library(readr)

Data Preparation

Follow the following steps; a few coding hints are also provided.

1) Read your file

```
Vehicles_data <- read.csv("C:/YOUR DIRECTORY/Vehicles data.csv")
```

2) Drop YEAR_ID and PRODUCTLINE variables.

```
Vehicles_data <- Vehicles_data[ ,-c(9,10)]
```

3) Transforming the categorical variables into dummy variables.

```
# Only DEALSIZE needs to be converted to factor
Vehicles_data$DEALSIZE <- as.factor(Vehicles_data$DEALSIZE)
# now, convert DEALSIZE to dummy variables
groups <- dummyVars(~., data = Vehicles_data)
# Create Dummy variable.names
Vehicles_data <- as.data.frame(predict(groups, Vehicles_data))
```

Questions

1. [20 Points] Partition the data into Training (50%), Validation (30%), and Testing (20%) sets.

Now, normalize the datasets. Consider the following code lines for help.

```
train.norm.df <- train.df[,-6] # Note that STATUS (response) is the 6th variable
valid.norm.df <- valid.df[,-6]
test.norm.df <- test.df[,-6]
norm.values <- preProcess(train.df[, -6], method=c("center", "scale"))
train.norm.df <- predict(norm.values, train.df[, -6])
valid.norm.df <- predict(norm.values, valid.df[, -6])
test.norm.df <- predict(norm.values, test.df[, -6])
```

2. [40 Points] Consider the following Vehicle,
   ORDERNUMBER=10322,
   QUANTITYORDERED=50,
   PRICEEACH=100,
   ORDERLINENUMBER=6,
   SALES=12536.5,
   QTR_ID=4,
   MONTH_ID=11,
   MSRP=127,
   DEALSIZE.Large=1,
   DEALSIZE.Small=0,
   DEALSIZE.Medium=0.

   Perform a k-NN classification using all predictors except for YEAR_ID and PRODUCTLINE, with k=1. Ensure that categorical predictors with more than two categories are converted into dummy variables before applying the classification. Set the class to 1 (representing 'Shipped') and use the default cutoff value of 0.5. How would KNN classify the vehicle? **Hint**: relevant commands are data.frame(), predict(), class::knn(). Do not forget to normalize the given Vehicle using predict() command before utilizing the KNN method.

3. [40 Points] What is a suitable *k* value that balances between overfitting and underfitting? **Hint**: which *k* gives you the highest accuracy based on the confusion matrix. Recall that hyperparameter, here *k*, optimization must be done based on Training and Validation datasets. A partial code view can be as

```
accuracy.df <- data.frame(k = seq(1, 15, 1), overallaccuracy = rep(0, 15))

for(i in 1:15) {
…
class::knn()

confusionMatrix()
… }
```

Note that there is no unique way to develop your code; provided partial codes (pseudocodes) can just serve you as hints and you don't need to use them.

- Submit your HTML file using R-Markdown.