HW 2 Stats Learn, Date: 02/20/25

Steafan Steinocher ID 1554129

The following Question 1 is in regards to this table:

Obs	X1	X2	X3	Y
1	1	2	-1	Blue
2	1	0	3	Blue
3	0	3	0	Red
4	0	2	-1	Red
5	2	0	-1	Blue
6	1	4	1	Red

Question 1a:

The nearest neighbor to any K = 1 observation is the observation itself so in this case Blue with a misclassification rate of 0%, and this would be the case for each of the other observations.

Question 1b:

Where K = 3, there is a misclassification of 1/6.

Obs	X1	X2	X3	Y
1	Blue	Red	Red	Red
2	Blue	Blue	Red	Blue
3	Red	Red	Blue	Red
4	Red	Blue	Red	Red
5	Blue	Blue	Red	Blue
6	Red	Red	Blue	Red

Question 1c:

Question 1 c i:

Because our point of interest is (0,0,0) the distance of each observation from our point of interest is simply the magnitude of the observation.

Obs	Distance from (0,0,0)
1	$\sqrt{6}$
2	$\sqrt{10}$
3	3
4	$\sqrt{5}$
5	$\sqrt{5}$
6	$\sqrt{18}$

Question 1 c ii:

- For K = 1 the nearest neighbor prediction could be Red or Blue as they are equidistant from the initial point.

Question 1 c iii:

- For K = 3 the nearest neighbor prediction would be Blue as the majority of points nearest to our point of interest are Blue.

Question 1 c iv:

- For K = 5 the nearest neighbor prediction would be Blue because the majority of points nearest to our point of interest are Blue.

Question 2:

Question 2 a:

The validation set approach splits the data set into two parts.

- Training set which is often seen as 70% of the data set.
- Validation set, which is used to evaluate the models performance which is often roughly 30% of the data set.

A disadvantage to the validation set approach is Data inefficiency, not as much data is used to train a model that adopts this approach.

Question 2 b:

Leave one out cross validation works as follows:

- Treat that observation as the validation set.
- Train the model on the remaining (n-1) observations, hence leave one out.
- Evaluate the model's performance on the held-out observation.

One of the advantages of LOOCV is that because more data is used, there is lower variance.

Question 2 c:

The main disadvantage of LOOCV that I have seen in this HW is the amount of time for computer. I think on EXTREMLY LARGE data sets, compute time will be long.

Question 2 d:

Both validation set approach and LOOCV can be applied to any supervised learning algorithm, not just k-Nearest Neighbors. Like, linear regression.

Question 3:

```r
#|eval: false
#Needed libraries
library(mlbench)
library(ISLR)
library(caret)#For KNN
```

```
Loading required package: ggplot2
```

```
Loading required package: lattice
```

```r
library(lattice)#For visualizations also required by caret
library(ggplot2)#For graphs also required by caret
```

```r
#|eval: false
#Data initialization and preprocessing
data("Ionosphere")
summary(Ionosphere)
```

```
 V1       V2              V3                 V4                  V5          
 0: 38   0:351   Min.   :-1.0000   Min.   :-1.00000   Min.   :-1.0000  
 1:313           1st Qu.: 0.4721   1st Qu.:-0.06474   1st Qu.: 0.4127  
                 Median : 0.8711   Median : 0.01631   Median : 0.8092  
                 Mean   : 0.6413   Mean   : 0.04437   Mean   : 0.6011  
                 3rd Qu.: 1.0000   3rd Qu.: 0.19418   3rd Qu.: 1.0000  
                 Max.   : 1.0000   Max.   : 1.00000   Max.   : 1.0000  
       V6                V7                V8                 V9          
 Min.   :-1.0000   Min.   :-1.0000   Min.   :-1.00000   Min.   :-1.00000  
 1st Qu.:-0.0248   1st Qu.: 0.2113   1st Qu.:-0.05484   1st Qu.: 0.08711  
 Median : 0.0228   Median : 0.7287   Median : 0.01471   Median : 0.68421  
 Mean   : 0.1159   Mean   : 0.5501   Mean   : 0.11936   Mean   : 0.51185  
 3rd Qu.: 0.3347   3rd Qu.: 0.9692   3rd Qu.: 0.44567   3rd Qu.: 0.95324  
 Max.   : 1.0000   Max.   : 1.0000   Max.   : 1.00000   Max.   : 1.00000  
      V10                V11                V12                V13         
 Min.   :-1.00000   Min.   :-1.00000   Min.   :-1.00000   Min.   :-1.0000  
 1st Qu.:-0.04807   1st Qu.: 0.02112   1st Qu.:-0.06527   1st Qu.: 0.0000  
 Median : 0.01829   Median : 0.66798   Median : 0.02825   Median : 0.6441  
 Mean   : 0.18135   Mean   : 0.47618   Mean   : 0.15504   Mean   : 0.4008  
 3rd Qu.: 0.53419   3rd Qu.: 0.95790   3rd Qu.: 0.48237   3rd Qu.: 0.9555  
 Max.   : 1.00000   Max.   : 1.00000   Max.   : 1.00000   Max.   : 1.0000  
      V14                V15               V16                V17         
 Min.   :-1.00000   Min.   :-1.0000   Min.   :-1.00000   Min.   :-1.0000  
 1st Qu.:-0.07372   1st Qu.: 0.0000   1st Qu.:-0.08170   1st Qu.: 0.0000  
 Median : 0.03027   Median : 0.6019   Median : 0.00000   Median : 0.5909  
 Mean   : 0.09341   Mean   : 0.3442   Mean   : 0.07113   Mean   : 0.3819  
 3rd Qu.: 0.37486   3rd Qu.: 0.9193   3rd Qu.: 0.30897   3rd Qu.: 0.9357  
 Max.   : 1.00000   Max.   : 1.0000   Max.   : 1.00000   Max.   : 1.0000  
      V18                 V19               V20                V21         
 Min.   :-1.000000   Min.   :-1.0000   Min.   :-1.00000   Min.   :-1.0000  
 1st Qu.:-0.225690   1st Qu.: 0.0000   1st Qu.:-0.23467   1st Qu.: 0.0000  
 Median : 0.000000   Median : 0.5762   Median : 0.00000   Median : 0.4991  
 Mean   :-0.003617   Mean   : 0.3594   Mean   :-0.02402   Mean   : 0.3367  
 3rd Qu.: 0.195285   3rd Qu.: 0.8993   3rd Qu.: 0.13437   3rd Qu.: 0.8949  
 Max.   : 1.000000   Max.   : 1.0000   Max.   : 1.00000   Max.   : 1.0000  
      V22                 V23               V24                V25         
 Min.   :-1.000000   Min.   :-1.0000   Min.   :-1.00000   Min.   :-1.0000  
```

```
 1st Qu.:-0.243870   1st Qu.: 0.0000   1st Qu.:-0.36689   1st Qu.: 0.0000
 Median : 0.000000   Median : 0.5318   Median : 0.00000   Median : 0.5539
 Mean   : 0.008296   Mean   : 0.3625   Mean   :-0.05741   Mean   : 0.3961
 3rd Qu.: 0.188760   3rd Qu.: 0.9112   3rd Qu.: 0.16463   3rd Qu.: 0.9052
 Max.   : 1.000000   Max.   : 1.0000   Max.   : 1.00000   Max.   : 1.0000
      V26                 V27                V28                 V29
 Min.   :-1.00000   Min.   :-1.0000   Min.   :-1.00000   Min.   :-1.0000
 1st Qu.:-0.33239   1st Qu.: 0.2864   1st Qu.:-0.44316   1st Qu.: 0.0000
 Median :-0.01505   Median : 0.7082   Median :-0.01769   Median : 0.4966
 Mean   :-0.07119   Mean   : 0.5416   Mean   :-0.06954   Mean   : 0.3784
 3rd Qu.: 0.15676   3rd Qu.: 0.9999   3rd Qu.: 0.15354   3rd Qu.: 0.8835
 Max.   : 1.00000   Max.   : 1.0000   Max.   : 1.00000   Max.   : 1.0000
      V30                 V31                V32                  V33
 Min.   :-1.00000   Min.   :-1.0000   Min.   :-1.000000   Min.   :-1.0000
 1st Qu.:-0.23689   1st Qu.: 0.0000   1st Qu.:-0.242595   1st Qu.: 0.0000
 Median : 0.00000   Median : 0.4428   Median : 0.000000   Median : 0.4096
 Mean   :-0.02791   Mean   : 0.3525   Mean   :-0.003794   Mean   : 0.3494
 3rd Qu.: 0.15407   3rd Qu.: 0.8576   3rd Qu.: 0.200120   3rd Qu.: 0.8138
 Max.   : 1.00000   Max.   : 1.0000   Max.   : 1.000000   Max.   : 1.0000
      V34             Class
 Min.   :-1.00000   bad :126
 1st Qu.:-0.16535   good:225
 Median : 0.00000
 Mean   : 0.01448
 3rd Qu.: 0.17166
 Max.   : 1.00000
```

```
head(Ionosphere)
```

```
  V1 V2      V3       V4       V5       V6       V7       V8      V9      V10
1  1  0 0.99539 -0.05889  0.85243  0.02306  0.83398 -0.37708 1.00000  0.03760
2  1  0 1.00000 -0.18829  0.93035 -0.36156 -0.10868 -0.93597 1.00000 -0.04549
3  1  0 1.00000 -0.03365  1.00000  0.00485  1.00000 -0.12062 0.88965  0.01198
4  1  0 1.00000 -0.45161  1.00000  1.00000  0.71216 -1.00000 0.00000  0.00000
5  1  0 1.00000 -0.02401  0.94140  0.06531  0.92106 -0.23255 0.77152 -0.16399
6  1  0 0.02337 -0.00592 -0.09924 -0.11949 -0.00763 -0.11824 0.14706  0.06637
      V11      V12     V13      V14      V15      V16      V17      V18
1 0.85243 -0.17755 0.59755 -0.44945  0.60536 -0.38223  0.84356 -0.38542
2 0.50874 -0.67743 0.34432 -0.69707 -0.51685 -0.97515  0.05499 -0.62237
3 0.73082  0.05346 0.85443  0.00827  0.54591  0.00299  0.83775 -0.13644
4 0.00000  0.00000 0.00000  0.00000 -1.00000  0.14516  0.54094 -0.39330
5 0.52798 -0.20275 0.56409 -0.00712  0.34395 -0.27457  0.52940 -0.21780
```

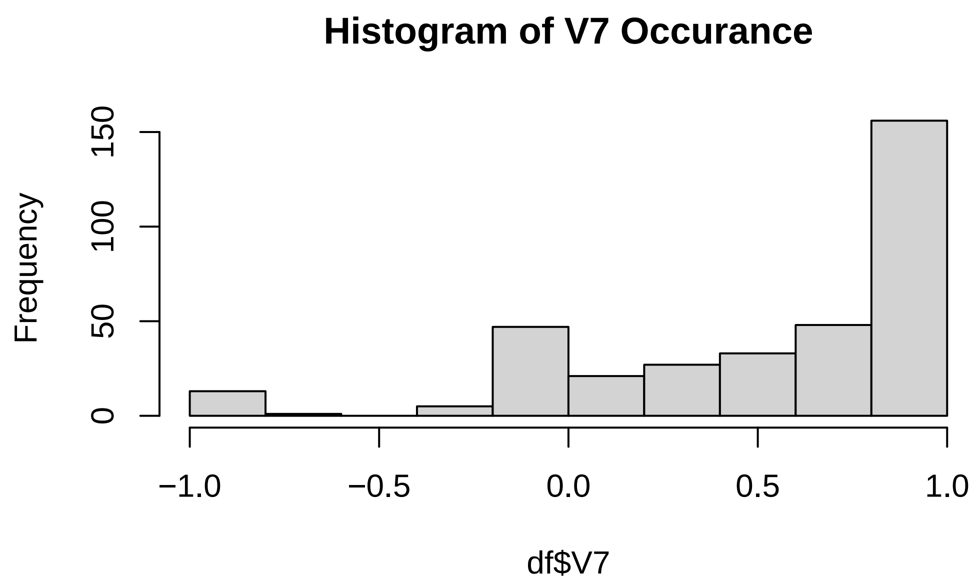
```
6 0.03786 -0.06302 0.00000  0.00000 -0.04572 -0.15540 -0.00343 -0.10196
       V19      V20      V21      V22      V23      V24      V25      V26
1  0.58212 -0.32192  0.56971 -0.29674  0.36946 -0.47357  0.56811 -0.51171
2  0.33109 -1.00000 -0.13151 -0.45300 -0.18056 -0.35734 -0.20332 -0.26569
3  0.75535 -0.08540  0.70887 -0.27502  0.43385 -0.12062  0.57528 -0.40220
4 -1.00000 -0.54467 -0.69975  1.00000  0.00000  0.00000  1.00000  0.90695
5  0.45107 -0.17813  0.05982 -0.35575  0.02309 -0.52879  0.03286 -0.65158
6 -0.11575 -0.05414  0.01838  0.03669  0.01519  0.00888  0.03513 -0.01535
       V27      V28      V29      V30      V31      V32      V33      V34 Class
1  0.41078 -0.46168  0.21266 -0.34090  0.42267 -0.54487  0.18641 -0.45300  good
2 -0.20468 -0.18401 -0.19040 -0.11593 -0.16626 -0.06288 -0.13738 -0.02447   bad
3  0.58984 -0.22145  0.43100 -0.17365  0.60436 -0.24180  0.56045 -0.38238  good
4  0.51613  1.00000  1.00000 -0.20099  0.25682  1.00000 -0.32382  1.00000   bad
5  0.13290 -0.53206  0.02431 -0.62197 -0.05707 -0.59573 -0.04608 -0.65697  good
6 -0.03240  0.09223 -0.07859  0.00732  0.00000  0.00000 -0.00039  0.12011   bad
```

```r
df <- Ionosphere
###About the data: 351 Observations and 34 Independent Variables(Removed one)
###Last column in the data is categorical variable called Class: good/bad
df <- subset(df, select = -V2) #Removed V2 bc all 0's
str(df)
```

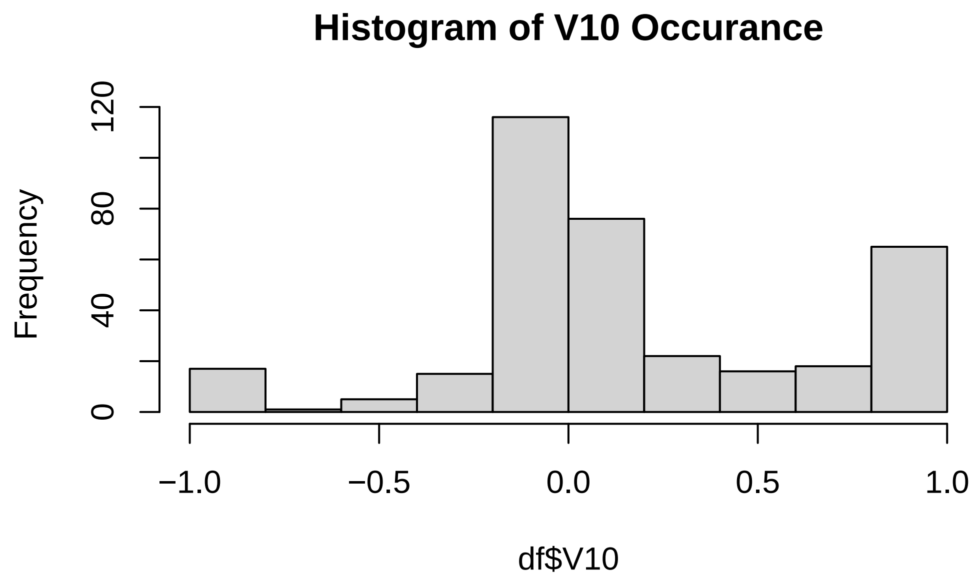
```
'data.frame':   351 obs. of  34 variables:
 $ V1   : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 1 2 2 ...
 $ V3   : num  0.995 1 1 1 1 ...
 $ V4   : num  -0.0589 -0.1883 -0.0336 -0.4516 -0.024 ...
 $ V5   : num  0.852 0.93 1 1 0.941 ...
 $ V6   : num  0.02306 -0.36156 0.00485 1 0.06531 ...
 $ V7   : num  0.834 -0.109 1 0.712 0.921 ...
 $ V8   : num  -0.377 -0.936 -0.121 -1 -0.233 ...
 $ V9   : num  1 1 0.89 0 0.772 ...
 $ V10  : num  0.0376 -0.0455 0.012 0 -0.164 ...
 $ V11  : num  0.852 0.509 0.731 0 0.528 ...
 $ V12  : num  -0.1776 -0.6774 0.0535 0 -0.2028 ...
 $ V13  : num  0.598 0.344 0.854 0 0.564 ...
 $ V14  : num  -0.44945 -0.69707 0.00827 0 -0.00712 ...
 $ V15  : num  0.605 -0.517 0.546 -1 0.344 ...
 $ V16  : num  -0.38223 -0.97515 0.00299 0.14516 -0.27457 ...
 $ V17  : num  0.844 0.055 0.838 0.541 0.529 ...
 $ V18  : num  -0.385 -0.622 -0.136 -0.393 -0.218 ...
 $ V19  : num  0.582 0.331 0.755 -1 0.451 ...
 $ V20  : num  -0.3219 -1 -0.0854 -0.5447 -0.1781 ...
```

```
 $ V21  : num  0.5697 -0.1315 0.7089 -0.6997 0.0598 ...
 $ V22  : num  -0.297 -0.453 -0.275 1 -0.356 ...
 $ V23  : num  0.3695 -0.1806 0.4339 0 0.0231 ...
 $ V24  : num  -0.474 -0.357 -0.121 0 -0.529 ...
 $ V25  : num  0.5681 -0.2033 0.5753 1 0.0329 ...
 $ V26  : num  -0.512 -0.266 -0.402 0.907 -0.652 ...
 $ V27  : num  0.411 -0.205 0.59 0.516 0.133 ...
 $ V28  : num  -0.462 -0.184 -0.221 1 -0.532 ...
 $ V29  : num  0.2127 -0.1904 0.431 1 0.0243 ...
 $ V30  : num  -0.341 -0.116 -0.174 -0.201 -0.622 ...
 $ V31  : num  0.4227 -0.1663 0.6044 0.2568 -0.0571 ...
 $ V32  : num  -0.5449 -0.0629 -0.2418 1 -0.5957 ...
 $ V33  : num  0.1864 -0.1374 0.5605 -0.3238 -0.0461 ...
 $ V34  : num  -0.453 -0.0245 -0.3824 1 -0.657 ...
 $ Class: Factor w/ 2 levels "bad","good": 2 1 2 1 2 1 2 1 2 1 ...
```

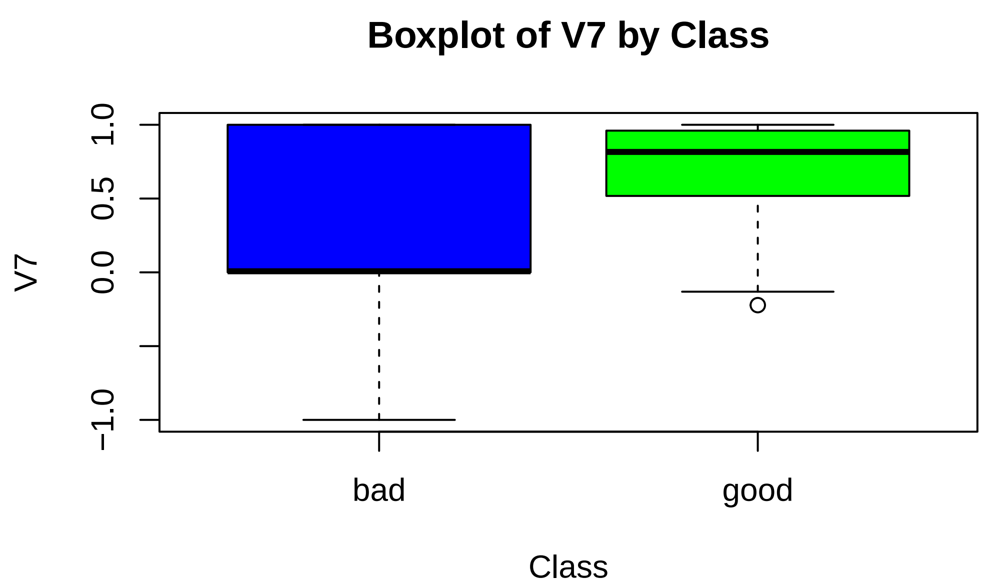
```r
#Here are some graphical summaries of the Ionosphere data
hist(df$V7, main = "Histogram of V7 Occurance")
```



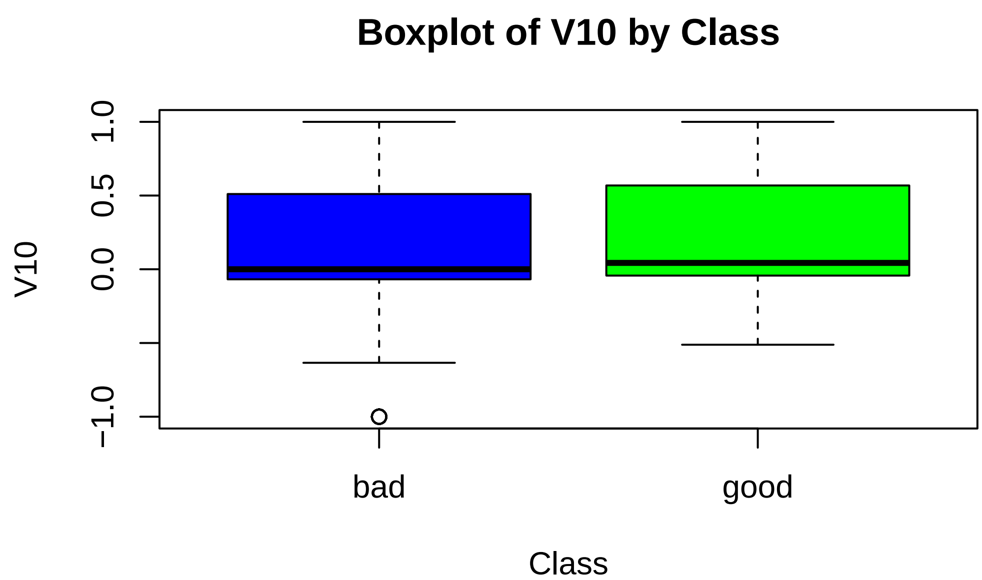

```r
hist(df$V10, main = "Histogram of V10 Occurance")
```




```r
boxplot(V7 ~ Class, data = df, col = c("blue", "green"), main = "Boxplot of V7 by Class")
```



```
boxplot(V10 ~ Class, data = df, col = c("blue", "green"), main = "Boxplot of V10 by Class")
```




```
#Here are the corresponding numerical summaries
summary(df$V7[df$Class == "good"])
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
-0.2222  0.5178  0.8159  0.7159  0.9603  1.0000 
```

```
summary(df$V7[df$Class == "bad"])
```

```
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max. 
-1.000000  0.000000  0.007185  0.253984  1.000000  1.000000 
```

```
summary(df$V10[df$Class == "good"])
```

```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
-0.51171 -0.04286  0.04317  0.22496  0.56830  1.00000 
```

```r
summary(df$V10[df$Class == "bad"])
```

```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
-1.00000 -0.06653  0.00000  0.10346  0.50075  1.00000 
```

There is some skewness in the histogram of V7 to the left direction.

As for the box plot of V10 the two plots seem to be roughly identical and this follows its histogram which looks approximately normally distributed.

```r
set.seed(4323)
#Data slicing
intrainQ3 <- createDataPartition(y = df$Class, p= 0.7, list = FALSE)
trainingQ3 <-df[intrainQ3,]
testingQ3 <- df[-intrainQ3,]


#checking to see if our dimensions add up
dim(trainingQ3)
```

```
[1] 247  34
```

```r
dim(testingQ3)
```

```
[1] 104  34
```

```r
#
trControl <- trainControl(method  = "cv",
                          number  = 5)
fit <- train(Class ~ .,
             method     = "knn",
             trControl  = trControl,
             tuneGrid   = expand.grid(k = 1:10),
             data       = df)
fit
```

```
k-Nearest Neighbors 

351 samples
 33 predictor
```

```
  2 classes: 'bad', 'good'

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 281, 280, 281, 281, 281
Resampling results across tuning parameters:

  k   Accuracy   Kappa
   1  0.8461569  0.6409977
   2  0.8375855  0.6208973
   3  0.8375855  0.6155820
   4  0.8461167  0.6360559
   5  0.8462374  0.6359032
   6  0.8348491  0.6049333
   7  0.8319920  0.6004563
   8  0.8206036  0.5691947
   9  0.8320322  0.6002482
  10  0.8263179  0.5847849

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was k = 5.
```

```r
test_predictionQ3 <- predict(fit, newdata= testingQ3)
test_predictionQ3
```

```
  [1] good good good bad  good bad  good good good good good good good good good
 [16] good bad  good bad  good bad  bad  good good good good good bad  good good
 [31] good bad  bad  bad  good good good good bad  good good good bad  good good
 [46] bad  good good bad  good good bad  good bad  bad  good bad  good good bad
 [61] good good good good bad  good good bad  good bad  good bad  good good good
 [76] good good good good good good good good bad  good good good good good good
 [91] good good good good good good good good good good good good good good
Levels: bad good
```

Test error fo k = 5 was found to be the best at roughly 15.38% in comparison to k = 7, 16.8% and k = 1, 15.39%.

```r
confusionMatrix(test_predictionQ3, testingQ3$Class)
```

```
Confusion Matrix and Statistics
```

```
          Reference
Prediction bad good
      bad   22    2
      good  15   65

               Accuracy : 0.8365
                 95% CI : (0.7512, 0.9018)
    No Information Rate : 0.6442
    P-Value [Acc > NIR] : 1.185e-05

                  Kappa : 0.613

 Mcnemar's Test P-Value : 0.003609

            Sensitivity : 0.5946
            Specificity : 0.9701
         Pos Pred Value : 0.9167
         Neg Pred Value : 0.8125
             Prevalence : 0.3558
         Detection Rate : 0.2115
   Detection Prevalence : 0.2308
      Balanced Accuracy : 0.7824

       'Positive' Class : bad
```

What is the confusion matrix saying?

There is an 83.65% accuracy in this model.

The model hallucinated 15 bad as good and 2 good as bad, in which this model is better at determining what is good opposed to bad.

Question 4:

```r
#Data initialization and preprocessing
data("Auto")
summary(Auto)
```

```
      mpg          cylinders      displacement     horsepower        weight    
 Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0   Min.   :1613  
```

```
 1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0   1st Qu.:2225
 Median :22.75   Median :4.000   Median :151.0   Median : 93.5   Median :2804
 Mean   :23.45   Mean   :5.472   Mean   :194.4   Mean   :104.5   Mean   :2978
 3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:275.8   3rd Qu.:126.0   3rd Qu.:3615
 Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :230.0   Max.   :5140

  acceleration        year           origin                      name
 Min.   : 8.00   Min.   :70.00   Min.   :1.000   amc matador       :  5
 1st Qu.:13.78   1st Qu.:73.00   1st Qu.:1.000   ford pinto        :  5
 Median :15.50   Median :76.00   Median :1.000   toyota corolla    :  5
 Mean   :15.54   Mean   :75.98   Mean   :1.577   amc gremlin       :  4
 3rd Qu.:17.02   3rd Qu.:79.00   3rd Qu.:2.000   amc hornet        :  4
 Max.   :24.80   Max.   :82.00   Max.   :3.000   chevrolet chevette:  4
                                                 (Other)           :365
```

```
attach(Auto)
```

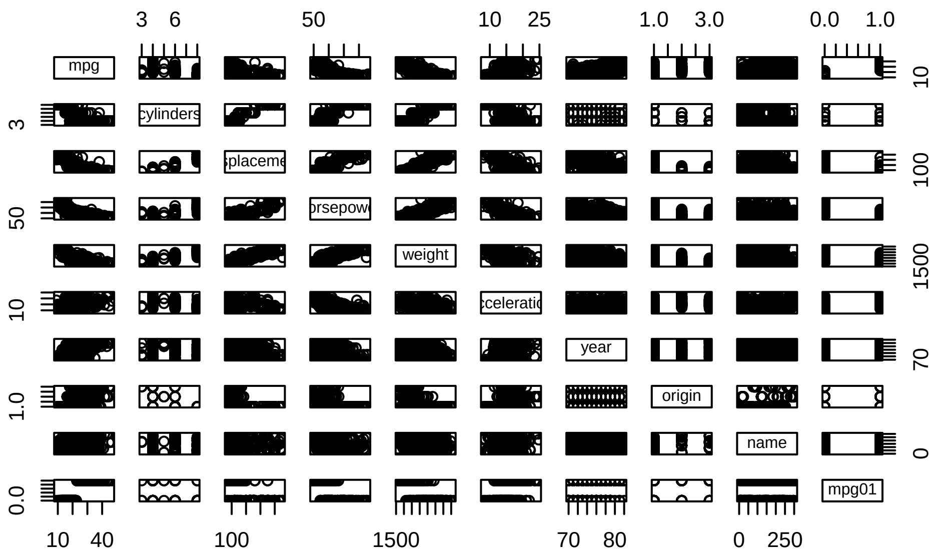
```
The following object is masked from package:ggplot2:

    mpg
```

```
mpg01 <- ifelse( mpg > median(mpg), yes = 1, no = 0)
newAuto <- data.frame(Auto, mpg01)
str(newAuto)
```

```
'data.frame':   392 obs. of  10 variables:
 $ mpg         : num  18 15 18 16 17 15 14 14 14 15 ...
 $ cylinders   : num  8 8 8 8 8 8 8 8 8 8 ...
 $ displacement: num  307 350 318 304 302 429 454 440 455 390 ...
 $ horsepower  : num  130 165 150 150 140 198 220 215 225 190 ...
 $ weight      : num  3504 3693 3436 3433 3449 ...
 $ acceleration: num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
 $ year        : num  70 70 70 70 70 70 70 70 70 70 ...
 $ origin      : num  1 1 1 1 1 1 1 1 1 1 ...
 $ name        : Factor w/ 304 levels "amc ambassador brougham",..: 49 36 231 14 161 141 54
 $ mpg01       : num  0 0 0 0 0 0 0 0 0 0 ...
```

```
#Scatter plot matrix
pairs(newAuto)
```



Scatter plot matrix:

Of the variables in the data set Auto, I found cylinders, weight, displacement, horsepower, acceleration and the age of the car to be among the most influencing of Mpg01.

```
#Here we standardized the data, since knn works on distance we don't want anything skewed
#cbind makes a matrix, apply, applies the scale (standardize) to the columns 2 not the rows 1
newAuto <- data.frame(mpg01, apply(cbind(cylinders, weight, displacement, horsepower, accele
str(newAuto)
```

```
'data.frame':   392 obs. of  7 variables:
 $ mpg01       : num  0 0 0 0 0 0 0 0 0 0 ...
 $ cylinders   : num  1.48 1.48 1.48 1.48 1.48 ...
 $ weight      : num  0.62 0.842 0.54 0.536 0.555 ...
 $ displacement: num  1.08 1.49 1.18 1.05 1.03 ...
 $ horsepower  : num  0.663 1.573 1.183 1.183 0.923 ...
 $ acceleration: num  -1.28 -1.46 -1.65 -1.28 -1.83 ...
 $ year        : num  70 70 70 70 70 70 70 70 70 70 ...
```

```
#Splitting the data 70:30
intrainQ4 <- createDataPartition(y = mpg01, p= 0.7, list = FALSE)
trainingQ4 <-newAuto[intrainQ4,]
testingQ4 <- newAuto[-intrainQ4,]
```

```r
dim(trainingQ4)
```

```
[1] 276   7
```

```r
dim(testingQ4)
```

```
[1] 116   7
```

```r
set.seed(1)

trControlQ4 <- trainControl(method  = "cv",
                            number  = 5)
fitQ4 <- train(as.factor(mpg01) ~ .,
             method     = "knn",
             trControl  = trControlQ4,
             tuneGrid   = expand.grid(k = 1:10),
             data       = newAuto)
fitQ4
```

```
k-Nearest Neighbors

392 samples
  6 predictor
  2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 314, 313, 314, 313, 314
Resampling results across tuning parameters:

  k   Accuracy   Kappa
   1  0.9209672  0.8419133
   2  0.9210321  0.8420609
   3  0.9260954  0.8522152
   4  0.9108731  0.8217583
   5  0.9133723  0.8268230
   6  0.9159364  0.8319125
   7  0.9185329  0.8371063
   8  0.9185005  0.8370503
```

```
   9  0.9236611  0.8473627
  10  0.9236611  0.8473529

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was k = 3.
```

```
test_predictionQ4 <- predict(fitQ4, newdata= testingQ4)
test_predictionQ4
```

```
  [1] 0 0 0 1 0 0 1 1 1 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 1 0
 [38] 1 0 0 1 1 0 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 1 1 1 1 0 1 0 1 0 0 1 1 0 1 0 0
 [75] 1 0 0 0 1 1 1 1 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
[112] 1 1 1 1 1
Levels: 0 1
```

Among the best K-Values, k = 3 was found to have a (1-0.926) 7.4% error rate.

I unfortunately already scaled my data but here is why: Because KNN works on distance, certain variables will have disproportionate weight. We needed to standardize the variables so that the distance between data points is not skewed. So scaling each feature to have a mean of 0 and a standard deviation of 1 should help.

```
confusionMatrix(test_predictionQ4, as.factor(testingQ4$mpg01))
```

```
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 54  1
         1  4 57

               Accuracy : 0.9569
                 95% CI : (0.9023, 0.9859)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9138

 Mcnemar's Test P-Value : 0.3711

            Sensitivity : 0.9310
```

```
            Specificity : 0.9828
         Pos Pred Value : 0.9818
         Neg Pred Value : 0.9344
             Prevalence : 0.5000
         Detection Rate : 0.4655
   Detection Prevalence : 0.4741
      Balanced Accuracy : 0.9569

       'Positive' Class : 0
```

Question 5:

```r
head(Auto)
```

```
  mpg cylinders displacement horsepower weight acceleration year origin
1  18         8          307        130   3504         12.0   70      1
2  15         8          350        165   3693         11.5   70      1
3  18         8          318        150   3436         11.0   70      1
4  16         8          304        150   3433         12.0   70      1
5  17         8          302        140   3449         10.5   70      1
6  15         8          429        198   4341         10.0   70      1
                       name
1 chevrolet chevelle malibu
2         buick skylark 320
3        plymouth satellite
4             amc rebel sst
5               ford torino
6          ford galaxie 500
```

```r
summary(Auto)
```

```
      mpg          cylinders      displacement     horsepower        weight    
 Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0   Min.   :1613  
 1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0   1st Qu.:2225  
 Median :22.75   Median :4.000   Median :151.0   Median : 93.5   Median :2804  
 Mean   :23.45   Mean   :5.472   Mean   :194.4   Mean   :104.5   Mean   :2978  
 3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:275.8   3rd Qu.:126.0   3rd Qu.:3615  
 Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :230.0   Max.   :5140  
```

```
  acceleration        year           origin                      name    
 Min.   : 8.00   Min.   :70.00   Min.   :1.000   amc matador       :  5  
 1st Qu.:13.78   1st Qu.:73.00   1st Qu.:1.000   ford pinto        :  5  
 Median :15.50   Median :76.00   Median :1.000   toyota corolla    :  5  
 Mean   :15.54   Mean   :75.98   Mean   :1.577   amc gremlin       :  4  
 3rd Qu.:17.02   3rd Qu.:79.00   3rd Qu.:2.000   amc hornet        :  4  
 Max.   :24.80   Max.   :82.00   Max.   :3.000   chevrolet chevette:  4  
                                                 (Other)           :365  
```

```r
set.seed(1)
```

```r
trControlQ5 <- trainControl(method  = "LOOCV", number  = 5)
newAutoQ5 <- data.frame(Auto, mpg01)

fitQ5 <- train(as.factor(mpg01) ~ .,
               method     = "knn",
               trControl  = trControlQ5,
               tuneGrid   = expand.grid(k = 1:10),
               data       = newAutoQ5)
fitQ5
```

```
k-Nearest Neighbors 

392 samples
  9 predictor
  2 classes: '0', '1' 

No pre-processing
Resampling: Leave-One-Out Cross-Validation 
Summary of sample sizes: 391, 391, 391, 391, 391, 391, ... 
Resampling results across tuning parameters:

  k   Accuracy   Kappa    
   1  0.8698980  0.7397959
   2  0.8724490  0.7448980
   3  0.8877551  0.7755102
   4  0.8801020  0.7602041
   5  0.8826531  0.7653061
   6  0.8750000  0.7500000
   7  0.8724490  0.7448980
   8  0.8877551  0.7755102
   9  0.8750000  0.7500000
```

```
  10  0.8673469  0.7346939

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was k = 8.
```

Before the scaling, the approach towards validation that has proven to be better is a scaled k-cross validation approach. The accuracy of the best performing k values from each approach were ascertain, and k-cross validation prevailed, with a test error rate of (1-.926) = 7.4% in comparison to a non scaled LOOCV test error rate where k = 8 of(1 - .888) 11.2%

```r
set.seed(1)
newAutoQ5Scaled <- data.frame(mpg01, apply(cbind(cylinders, weight, displacement, horsepower

fitQ5 <- train(as.factor(mpg01) ~ .,
               method     = "knn",
               trControl  = trControlQ5,
               tuneGrid   = expand.grid(k = 1:10),
               data       = newAutoQ5Scaled)
fitQ5
```

```
k-Nearest Neighbors

392 samples
  6 predictor
  2 classes: '0', '1'

No pre-processing
Resampling: Leave-One-Out Cross-Validation
Summary of sample sizes: 391, 391, 391, 391, 391, 391, ...
Resampling results across tuning parameters:

  k   Accuracy   Kappa
   1  0.9234694  0.8469388
   2  0.9081633  0.8163265
   3  0.9209184  0.8418367
   4  0.9107143  0.8214286
   5  0.9132653  0.8265306
   6  0.9158163  0.8316327
   7  0.9158163  0.8316327
   8  0.9234694  0.8469388
   9  0.9234694  0.8469388
  10  0.9183673  0.8367347
```

```
Accuracy was used to select the optimal model using the largest value.
The final value used for the model was k = 9.
```

After having scaled the data it seems that LOOCV performed better than its un-scaled version with a k value = 9 yielding a test error rate of (1-.9235) 7.7% but exceptionally close to a scaled version of k - cross validation with a difference of approximately 0.3%.

In this case it would be better to do the k - cross validation approach as there is less compute time, as far as trusting which validation approach more, that would also be k-cross validation but it depends on the use case of your machine learning model.

PS: I couldnt figure out how to display the code but prevent evaluation, I am sorry.