Question 2

Now consider, from the KC weather data set, just the predictors: Temp.F, Humidity.percentage, Precip.in. Categorize these three data sets into qualitative predictors. It is up to you to decide on the break points, but you must discuss a rationale for your breakpoints. Now apply, naive Bayes Classifier on the entire data set (with these three qualitative predictors), using 290 of them as a training data set randomly (and the rest as the test data set), over 100 replications. Report on accuracy, precision, and recall.

```
#import the dataset and make some changes

library(readr)

library(naivebayes)

kc_weather_srt <- read_csv("C:/Users/bvkka/Desktop/ISL-Deep
Medhi/kc_weather_srt.csv")

#consider only 3 predictors - Temp.F,Humidity.%,Precip.in.

kc_weather_srt=kc_weather_srt[,c(2,4,8,9)]
```

```
# A tibble: 6 x 4
  Temp.F Humidity.percentage Precip.in Events
   <int>               <int>     <dbl>  <chr>
1     26                  73      0.03   Snow
2     31                  68      0.01   Snow
3     10                  63      0.02   Snow
4     38                  90      0.00   Rain
5     40                  75      0.00   Rain
6     49                  51      0.00   Rain
> View(kc_weather_srt)
> View(kc_weather_srt)
```

```
head(kc_weather_srt) # to check the new data consisting of only three predictors
```

```
###*****categorization of three predictors****###
```

```r
mean(kc_weather_srt$Temp.F) ## we see 58.74044

mean(kc_weather_srt$Humidity.percentage)## we see 69.85246

mean(kc_weather_srt$Precip.in)##we see 0.1728415
```

```
> mean(kc_weather_srt$Temp.F)
[1] 0.5546448
> mean(kc_weather_srt$Humidity.percentage)
[1] 0.5601093
> mean(kc_weather_srt$Precip.in)
[1] 0.2677596
>
```

```r
##Now, categorize the temp.F based on mean of all the values that is values >58.74 as 1 and <58.74 as 0

##Now, categorize the Humidity% based on mean of all the values that is values >69.85246 as 1 and <69.85246 as 0

##Now, categorize the Precip.in based on mean of all the values that is values >0.1728415 as 1 and <0.1728415 as 0
```

```r
kc_weather_srt$Temp.F=ifelse(kc_weather_srt$Temp.F>58.74044,1,0)

kc_weather_srt$Humidity.percentage=ifelse(kc_weather_srt$Humidity.percentage>69.85246,1,0)

kc_weather_srt$Precip.in=ifelse(kc_weather_srt$Precip.in>0.1728415,1,0)
```

```r
head(kc_weather_srt)
```

```
# A tibble: 6 x 4
  Temp.F Humidity.percentage Precip.in Events
   <dbl>               <dbl>     <dbl>  <chr>
1      0                   1         0   Snow
2      0                   0         0   Snow
3      0                   0         0   Snow
4      0                   1         0   Rain
5      0                   1         0   Rain
6      0                   0         0   Rain
>
```

```
#replications

rep=100




# newly added


accuracy=dim(rep)



precision_snow=dim(rep)

precision_rain=dim(rep)

precision_rainThunderstorm=dim(rep)


recall_snow=dim(rep)

recall_rain=dim(rep)

recall_rainThunderstorm=dim(rep)



#splitting the dataset into training and test sets, also install caTools packages

#install.packages('caTools')

library(caTools)
```

```r
set.seed(123)

for(k in 1:rep)
{

  split=sample.split(kc_weather_srt$Events,SplitRatio = 0.7923)
  training_set=subset(kc_weather_srt,split==TRUE)
  test_set=subset(kc_weather_srt,split==FALSE)
```

Data	
kc_weather_srt	366 obs. of 4 variables
test_set	76 obs. of 4 variables
training_set	290 obs. of 4 variables
Values	
split	logi [1:366] TRUE TRUE TRUE TRUE TRU…

```r
#install.packages('e1071')
library(e1071)

Nb=naiveBayes(formula=Events~.,data=training_set)
summary(Nb)
#predicting the test set results
y_pred=predict(Nb,newdata=test_set[-4])
#making the confusion matrix
cm=table(y_pred,test_set[,4])
```

```r
accuracy[k]=mean(y_pred==test_set[,4])


precision=precision<-diag(cm)/colSums(cm)
precision_rainThunderstorm[k]=precision[3]
precision_snow[k]=precision[2]
precision_rain[k]=precision[1]


recall=recall<-diag(cm/rowSums(cm))
recall_rainThunderstorm[k]=recall[3]
recall_snow[k]=recall[2]
recall_rain[k]=recall[1]

}


mean(accuracy) ##0.5307895

mean(precision_snow)##0.7293103
mean(precision_rain)##0.2483784
mean(precision_rainThunderstorm)##1

mean(recall_snow)##0.7474423
mean(recall_rain)##0.6610176
mean(recall_rainThunderstorm)##0.2998356
```

(ADDED)

naiveBayes implementation the R package e2071 allows predictors to be *quantitative* as well. Analyze Temp.F as quantitative predictor in naiveBayes. Also, consider all predictors as quantitative predictors and comment on how the results differ again single predictor Tem.F, and also against other models for Q-1.

(ADDED NOTE-2):

First include a text summarizing your KEY observations and any issues (this can be a page or so in single-space). Following this, include the output from R. From the text, you may include some pointers to the R output where your observation comes from.

#consider Temp.F as quantitative and other two predictors as qualitative

```
#import the dataset and make some changes

library(readr)

library(naivebayes)

library(e2071)


kc_weather_srt <- read_csv("C:/Users/bvkka/Desktop/ISL-Deep
Medhi/kc_weather_srt.csv")

#consider only 3 predictors - Temp.F,Humidity.%,Precip.in.

kc_weather_srt=kc_weather_srt[,c(2,4,8,9)]


head(kc_weather_srt) # to check the new data consisting of only three predictors
```

###**categorization of two predictors****###***

```
mean(kc_weather_srt$Humidity.percentage)## we see 69.85246
```

```
mean(kc_weather_srt$Precip.in)##we see 0.1728415
```

```
##Now, categorize the temp.F based on mean of all the values that is values >69.85246 as 1 and <69.85246 as 0

##Now, also categorize the Precip on mean of all the values that is values >0.1728415 as 1 and <0.1728415 as 0
```

```
kc_weather_srt$Humidity.percentage=ifelse(kc_weather_srt$Humidity.percentage>69.85246
,1,0)

kc_weather_srt$Precip.in=ifelse(kc_weather_srt$Precip.in>0.1728415,1,0)
```

```
head(kc_weather_srt)
```

```
> head(kc_weather_srt)
# A tibble: 6 x 4
  Temp.F Humidity.percentage Precip.in Events
   <int>               <dbl>     <dbl>  <chr>
1     26                   1         0   Snow
2     31                   0         0   Snow
3     10                   0         0   Snow
4     38                   1         0   Rain
5     40                   1         0   Rain
6     49                   0         0   Rain
> 
```

```
#replications

rep=100
```

```
# newly added
```

```
accuracy=dim(rep)
```

```r
precision_snow=dim(rep)
precision_rain=dim(rep)
precision_rainThunderstorm=dim(rep)

recall_snow=dim(rep)
recall_rain=dim(rep)
recall_rainThunderstorm=dim(rep)


#splitting the dataset into training and test sets, also install caTools packages
#install.packages('caTools')
library(caTools)
set.seed(123)

for(k in 1:rep)
{


  split=sample.split(kc_weather_srt$Events,SplitRatio = 0.7923)
  training_set=subset(kc_weather_srt,split==TRUE)
  test_set=subset(kc_weather_srt,split==FALSE)





  #install.packages('e1071')
  library(e1071)
```

```
  Nb=naiveBayes(formula=Events~.,data=training_set)

  summary(Nb)

  #predicting the test set results

  y_pred=predict(Nb,newdata=test_set[-4])

  #making the confusion matrix

  cm=table(y_pred,test_set[,4])


  accuracy[k]=mean(y_pred==test_set[,4])




  precision=precision<-diag(cm)/colSums(cm)

  precision_rainThunderstorm[k]=precision[3]

  precision_snow[k]=precision[2]

  precision_rain[k]=precision[1]




  recall=recall<-diag(cm/rowSums(cm))

  recall_rainThunderstorm[k]=recall[3]

  recall_snow[k]=recall[2]

  recall_rain[k]=recall[1]


}
```

```
mean(accuracy) ##0.6923684


mean(precision_snow)##0.7682759

mean(precision_rain)##0.5497292

mean(precision_rainThunderstorm)##1


mean(recall_snow)##0.7787254

mean(recall_rain)##0.7723869

mean(recall_rainThunderstorm)##0.4878522
```

(ADDED)

naiveBayes implementation the R package e2071 allows predictors to be *quantitative* as well. Analyze Temp.F as quantitative predictor in naiveBayes. Also, consider all predictors as quantitative predictors and comment on how the results differ again single predictor Tem.F, and also against other models for Q-1.

(ADDED NOTE-2):

First include a text summarizing your KEY observations and any issues (this can be a page or so in single-space). Following this, include the output from R. From the text, you may include some pointers to the R output where your observation comes from.

#consider all predictors as quantitative and use Naive Bayes Classifier and install package (e2071)

```
#import the dataset and make some changes

library(readr)

library(naivebayes)

library(e2071)


kc_weather_srt <- read_csv("C:/Users/bvkka/Desktop/ISL-Deep Medhi/kc_weather_srt.csv")

#consider only 3 predictors - Temp.F,Humidity.%,Precip.in.

kc_weather_srt=kc_weather_srt[,c(2,4,8,9)]


head(kc_weather_srt)
```

```
# A tibble: 6 x 4
  Temp.F Humidity.percentage Precip.in Events
   <int>               <int>     <dbl>  <chr>
1     26                  73      0.03   Snow
2     31                  68      0.01   Snow
3     10                  63      0.02   Snow
4     38                  90      0.00   Rain
5     40                  75      0.00   Rain
6     49                  51      0.00   Rain
> 
```

```
#replications
rep=100



# newly added

#snow=1 rain=0 thunderstorm=2
accuracy=dim(rep)



precision_snow=dim(rep)
precision_rain=dim(rep)
precision_rainThunderstorm=dim(rep)

recall_snow=dim(rep)
recall_rain=dim(rep)
recall_rainThunderstorm=dim(rep)



#splitting the dataset into training and test sets, also install caTools packages
#install.packages('caTools')
library(caTools)
set.seed(123)

for(k in 1:rep)
{
```

```r
split=sample.split(kc_weather_srt$Events,SplitRatio = 0.7923)
training_set=subset(kc_weather_srt,split==TRUE)
test_set=subset(kc_weather_srt,split==FALSE)




#install.packages('e1071')
library(e1071)




Nb=naiveBayes(formula=Events~.,data=training_set)
summary(Nb)
#predicting the test set results
y_pred=predict(Nb,newdata=test_set[-4])
#making the confusion matrix
cm=table(y_pred,test_set[,4])

accuracy[k]=mean(y_pred==test_set[,4])



precision=precision<-diag(cm)/colSums(cm)
precision_rainThunderstorm[k]=precision[3]
precision_snow[k]=precision[2]
precision_rain[k]=precision[1]



recall=recall<-diag(cm/rowSums(cm))
```

```r
  recall_rainThunderstorm[k]=recall[3]
  recall_snow[k]=recall[2]
  recall_rain[k]=recall[1]

}

mean(accuracy) ##0.7507895

mean(precision_snow)##0.73620689
mean(precision_rain)##0.7064865
mean(precision_rainThunderstorm)##0.957

mean(recall_snow)##0.7498165
mean(recall_rain)##0.766421
mean(recall_rainThunderstorm)##0.7367071
```

Summary (Naïve Bayes):

Naive Bayes	Accuracy	Precision Snow	Precision Rain	Precision Rain Thunderstorm	Recall Snow	Recall Rain	Recall Thunderstc
Three Qualitative Predictors	0.5307895	0.7293103	0.2483784	1	0.7474423	0.6610176	0.2998356
Only temperature Quantitative	0.6923684	0.7682759	0.5497292	1	0.7787254	0.7723869	0.4878522
All Three Quantitative predictors	0.7507895	0.73620689	0.7064865	0.957	0.7498165	0.766421	0.7367071

Text Summarization:

1. When all the three predictors i.e. Temperature, Humidity and Precipitation are categorized as Qualitative, we observe that the Accuracy using Naïve Bayes Model is very less. The Recall values are also less but the Precision values are good, and we see the Precision of Rain happening is 100%.
2. When only the temperature is quantitative, and rest two predictors are quantitative, the recall value is high compared to the remaining models and has better accuracy and a good precision values. So, when recall and accuracy are of higher importance then this model is the best fit.
3. When all three Predictors are Quantitative, we observe that the accuracy is increased (is best) and the Precision and recall values are also good. So, from above three summary, we can conclude that the predictors being quantitative is more better when we use Naïve Bayes compared to when we use this model for Qualitative Predictors.
4. When Naïve Bayes model results is compared to results obtained for LDA, QDA and KNN model the LDA, QDA model dominates the Naïve Bayes model in all the metrics accuracy, precision and recall values.
5. The Naïve Bayes model performs better than the KNN model (K=5) when at least one of the predictor is Quantitative.