## Question 1

You're to use the KC Weather Data ("kc_weather_srt.csv", available here: kc_weather_srt.csv ). The data has categorized the weather for each day into three categories ("Events": Rain, Rain_Thunderstorm, Snow) over the three years 2014, 2015, and 2016. You'll note that not all dates are listed because it's a filtered subset where other categories or no events are deleted to have a more manageable subset. The entire dataset has 366 entries. The column labels indicate the units as well such as Temp.F means temperature in Fahrenheit, Visibility.mi means Visibility in miles, etc.
You're to do two level of analysis

a. Consider first the subset that consists only of Rain and Snow. There are 226 entries with these two categories.
   i. Apply logistic regression, LDA, QDA, and knn on this dataset to determine the accuracy, precision, and recall of these models. You're to use randomly 180 days for the training set (approximately 80% of 226) and the rest for the test data set. Conduct your study over 100 replications, and summary the result of your analysis with your conclusion which models you'll recommend to use based on the metrics: accuracy, precision and recall.
   ii. Discuss and analyze in a systematic way you would consider eliminating some of the predictors and see if your accuracy, precision and recall improves.
b. Consider next the entire dataset consisting of 366 entries. Now logistics regression cannot be applied, but you can apply the rest of them. Repeat the above studies in i) and ii) with LDA, QDA, and knn on the entire data set (using 290 of them in a training set). Do not forget randomization and 100 replications for this study.

(ADDED NOTE):
For logistic regression, accuracy, precision etc may not be "directly" or "readily" available. So, I left this part open for you to figure out which metrics you'd use or how you'd choose response for creating metrics for logistic regression, or any steps you might take to obtain the standard metrics.
(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.

```r
#import the dataset and make some changes
library(readr)
kc_weather_srt <- read_csv("C:/Users/bvkka/Desktop/ISL-Deep Medhi/kc_weather_srt.csv")

kc_weather_srt=kc_weather_srt[,2:9] #selecting all the predictors and response column
```

| | Temp.F | Dew_Point.F | Humidity.percentage | Sea_Level_Press.in | Visibility.mi | Wind.mph | Precip.in | Events |
|---|---|---|---|---|---|---|---|---|
| 19 | 22 | 11 | 59 | 30.38 | 10 | 11 | 0.00 | Snow |
| 20 | 21 | 11 | 76 | 30.29 | 5 | 14 | 0.03 | Snow |
| 21 | 5 | -5 | 65 | 30.50 | 5 | 14 | 0.08 | Snow |
| 22 | 36 | 25 | 72 | 30.20 | 7 | 4 | 0.00 | Snow |
| 23 | 54 | 23 | 29 | 29.87 | 10 | 13 | 0.00 | Rain |
| 24 | 55 | 36 | 50 | 29.88 | 9 | 7 | 0.38 | Rain_Thunderstorm |
| 25 | 34 | 14 | 53 | 30.47 | 8 | 9 | 0.00 | Snow |
| 26 | 34 | 23 | 68 | 30.26 | 6 | 5 | 0.00 | Snow |
| 27 | 46 | 25 | 53 | 30.00 | 9 | 11 | 0.07 | Rain |
| 28 | 55 | 42 | 68 | 29.57 | 10 | 16 | 0.00 | Rain_Thunderstorm |
| 29 | 62 | 37 | 37 | 29.69 | 10 | 19 | 0.00 | Rain_Thunderstorm |
| 30 | 49 | 45 | 85 | 29.83 | 6 | 11 | 0.39 | Rain_Thunderstorm |
| 31 | 58 | 50 | 82 | 29.63 | 9 | 9 | 0.01 | Rain_Thunderstorm |
| 32 | 42 | 28 | 59 | 30.03 | 10 | 14 | 0.00 | Rain |
| 33 | 57 | 31 | 40 | 29.97 | 10 | 7 | 0.00 | Rain |
| 34 | 72 | 54 | 53 | 29.69 | 10 | 11 | 0.00 | Rain_Thunderstorm |
| 35 | 56 | 54 | 78 | 29.54 | 7 | 10 | 0.91 | Rain_Thunderstorm |

```
#subset that consists of only rain and snow
kc_weather_srt_without_rainthunderstorm<-kc_weather_srt[!grepl("Rain_Thunderstorm",kc_weather_srt$Events),]

dim(kc_weather_srt_without_rainthunderstorm) #dimensions
```

| | Temp.F | Dew_Point.F | Humidity.percentage | Sea_Level_Press.in | Visibility.mi | Wind.mph | Precip.in | Events |
|---|---|---|---|---|---|---|---|---|
| 16 | 55 | 28 | 38 | 29.82 | 10 | 12 | 0.00 | Rain |
| 18 | 44 | 20 | 41 | 29.88 | 10 | 11 | 0.00 | Rain |
| 19 | 22 | 11 | 59 | 30.38 | 10 | 11 | 0.00 | Snow |
| 20 | 21 | 11 | 76 | 30.29 | 5 | 14 | 0.03 | Snow |
| 21 | 5 | -5 | 65 | 30.50 | 5 | 14 | 0.08 | Snow |
| 22 | 36 | 25 | 72 | 30.20 | 7 | 4 | 0.00 | Snow |
| 23 | 54 | 23 | 29 | 29.87 | 10 | 13 | 0.00 | Rain |
| 25 | 34 | 14 | 53 | 30.47 | 8 | 9 | 0.00 | Snow |
| 26 | 34 | 23 | 68 | 30.26 | 6 | 5 | 0.00 | Snow |
| 27 | 46 | 25 | 53 | 30.00 | 9 | 11 | 0.07 | Rain |
| 32 | 42 | 28 | 59 | 30.03 | 10 | 14 | 0.00 | Rain |
| 33 | 57 | 31 | 40 | 29.97 | 10 | 7 | 0.00 | Rain |
| 36 | 38 | 25 | 64 | 30.03 | 9 | 16 | 0.00 | Snow |
| 37 | 56 | 29 | 37 | 29.94 | 10 | 18 | 0.00 | Rain |
| 38 | 70 | 50 | 51 | 30.04 | 10 | 8 | 0.00 | Rain |
| 39 | 69 | 54 | 63 | 29.97 | 10 | 7 | 0.00 | Rain |

```r
#first make the response column to 0 and 1 (qualitative)
#install.packages("plyr")
library(plyr)
kc_weather_srt_without_rainthunderstorm$Events <-
revalue(kc_weather_srt_without_rainthunderstorm$Events,c("Snow"=1))
kc_weather_srt_without_rainthunderstorm$Events <-
revalue(kc_weather_srt_without_rainthunderstorm$Events,c("Rain"=0))
```

| | Temp.F | Dew_Point.F | Humidity.percentage | Sea_Level_Press.in | Visibility.mi | Wind.mph | Precip.in | Events |
|---|---|---|---|---|---|---|---|---|
| 1 | 26 | 12 | 73 | 30.19 | 5 | 9 | 0.03 | 1 |
| 2 | 31 | 18 | 68 | 29.95 | 7 | 11 | 0.01 | 1 |
| 3 | 10 | 1 | 63 | 30.24 | 5 | 14 | 0.02 | 1 |
| 4 | 38 | 35 | 90 | 29.70 | 6 | 5 | 0.00 | 0 |
| 5 | 40 | 30 | 75 | 29.80 | 9 | 7 | 0.00 | 0 |
| 6 | 49 | 29 | 51 | 29.64 | 10 | 10 | 0.00 | 0 |
| 7 | 36 | 19 | 45 | 30.02 | 10 | 9 | 0.00 | 0 |
| 8 | 29 | 11 | 48 | 30.14 | 10 | 8 | 0.00 | 0 |
| 9 | 26 | 2 | 38 | 30.13 | 10 | 13 | 0.00 | 1 |
| 10 | 13 | -3 | 46 | 30.37 | 10 | 12 | 0.00 | 1 |
| 11 | 28 | 3 | 35 | 30.19 | 10 | 14 | 0.00 | 1 |

```r
#character to numeric
kc_weather_srt_without_rainthunderstorm$Events<-
as.numeric(as.character(kc_weather_srt_without_rainthunderstorm$Events))
```

```r
# number of replications
rep=100



# newly added

#snow=1 rain=0
accuracy=dim(rep)
accuracy1=dim(rep)
accuracy2=dim(rep)
accuracy3=dim(rep)

precision_snow=dim(rep)
precision_rain=dim(rep)
recall_snow=dim(rep)
recall_rain=dim(rep)

precision_snow1=dim(rep)
precision_rain1=dim(rep)
recall_snow1=dim(rep)
recall_rain1=dim(rep)

precision_snow2=dim(rep)
precision_rain2=dim(rep)
recall_snow2=dim(rep)
recall_rain2=dim(rep)


precision_snow3=dim(rep)
precision_rain3=dim(rep)
recall_snow3=dim(rep)
recall_rain3=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_without_rainthunderstorm$Events,SplitRatio = 0.8) #80% split ratio
training_set=subset(kc_weather_srt_without_rainthunderstorm,split==TRUE) #80% split into training set
test_set=subset(kc_weather_srt_without_rainthunderstorm,split==FALSE) #20% split into testing set
table(split)
dim(training_set)
```

| | |
|---|---|
| test_set | 45 obs. of 8 variables |
| training_set | 181 obs. of 8 variables |

```
#****Logistic Regression***#
#Fitting Logistic Regression to the Training set
model1<-glm(formula = Events ~ ., binomial(link="logit"),data =training_set)

#predicting the test set results
prob_pred=predict(model1,type='response',newdata = test_set[-8])   #for predicitng we only need predictors , but not response
summary(prob_pred)
y_pred=ifelse(prob_pred>0.5,1,0) #vector of predictions #Threshold value of 0.5
y_pred

#Making the confusion Matrix
cm=table(y_pred,test_set[,8])
cm

#accuracy
accuracy[k]=mean(y_pred==test_set[,8])
accuracy


#precision
```

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

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





#***LDA***#
#install.packages("MASS")
library(MASS)



lda=lda(formula=Events~.,data=training_set)
y_pred1=predict(lda,test_set)$class
cm1=table(y_pred1,test_set[,8])
cm1
accuracy1[k]=mean(y_pred1==test_set[,8])

precision1=precison1<-diag(cm1/colSums(cm1))
precision_snow1[k]=precison1[2]
precision_rain1[k]=precision1[1]
recall1=recall1<-diag(cm1/rowSums(cm1))
recall_snow1[k]=recall1[2]
recall_rain1[k]=recall1[1]





#### or use SVM (but not for this project)###
```

```
#lda=lda(formula=Events~.,data=training_set)
#training_set1=as.data.frame(predict(lda,training_set))
#training_set1=training_set1[c(4,1)]
#plot(lda)

#test_set1=as.data.frame(predict(lda,test_set))
#test_set1=test_set1[c(4,1)]

#fitting SVM to the training set
#install.packages('e1071')
#library(e1071)
#classifier=svm(formula=class~.,data=training_set1,type='C-classification',kernel='linear')
#predicting the test set results
#y_pred1=predict(classifier,newdata = test_set1[-2])
#making the confusion matrix
#cm1=table(y_pred1,test_set1[,2])
#cm1 #we see TP+TN=33+12=45 true predictions and FP+FN=0+0=0 False predictions, therefore it is 100% accurate
#accuracy1[k]=mean(y_pred1==test_set1[,2])
#accuracy1



#***QDA***#
qda=qda(formula=Events~.,data=training_set)
y_pred2=predict(qda,test_set)$class
cm2=table(y_pred2,test_set[,8])
cm2
accuracy2[k]=mean(y_pred2==test_set[,8])

precision2=precison2<-diag(cm2/colSums(cm2))
precision_snow2[k]=precison2[2]
precision_rain2[k]=precision2[1]
recall2=recall2<-diag(cm2/rowSums(cm2))
recall_snow2[k]=recall2[2]
recall_rain2[k]=recall2[1]
```

```
#***KNN***#


#install.packages('class')
#fitting KNN to the training set and predicting the test set results
library(class)
y_pred3=knn(train=training_set[,-8],test=test_set[,-8],cl=training_set[,8],k=5)

#making the cm
cm3=table(y_pred3,test_set[,8])
cm3
accuracy3[k]=mean(y_pred3==test_set[,8])

precision3=precision3<-diag(cm3/colSums(cm3))
precision_snow3[k]=precision3[2]
precision_rain3[k]=precision3[1]

recall3=recall3<-recall3<-diag(cm3/rowSums(cm3))
recall_snow3[k]=recall3[2]
recall_rain3[k]=recall3[1]
}


###*******MEAN VALUES*****#####

#****Logestic regression***#
mean(accuracy)###0.9531111
mean(precision_snow)###0.889
mean(precision_rain)###0.9714286

mean(recall_snow) ### 0.9075805
mean(recall_rain) ### 0.9688803



#****LDA***#
```

```
mean(accuracy1)###0.936222
mean(precision_snow1)###0.865
mean(precision_rain1)###0.9565714

mean(recall_snow1) ### 0.8595878
mean(recall_rain1) ### 0.9618026



#****QDA***#
mean(accuracy2)###0.9346667
mean(precision_snow2)###0.952
mean(precision_rain2)###0.9297143

mean(recall_snow2) ### 0.8043408
mean(recall_rain2) ### 0.985939



#****KNN***#
mean(accuracy3)###0.9486667
mean(precision_snow3)###0.878
mean(precision_rain3)###0.9688571

mean(recall_snow3) ### 0.89958469
mean(recall_rain3) ### 0.9658621
```

```
+ cm3=table(y_pred3,test_set[,8])
+ cm3
+ accuracy3[k]=mean(y_pred3==test_set[,8])
+
+ precision3=precision3<-diag(cm3/colSums(cm3))
+ precision_snow3[k]=precision3[2]
+ precision_rain3[k]=precision3[1]
+
+ recall3=recall3<-recall3<-diag(cm3/rowSums(cm3))
+ recall_snow3[k]=recall3[2]
+ recall_rain3[k]=recall3[1]
+ }
There were 50 or more warnings (use warnings() to see the first 50)
> accuracy
  [1] 1.0000000 1.0000000 0.9555556 1.0000000 0.9777778 0.9777778 0.9111111 1.0000000 0.9333333 0.9333333 0.9555556 1.0000000 0.9555556 0.9333333 0.9555556 0.9333333 0.8888889
 [18] 0.8888889 0.9777778 0.9555556 0.9777778 0.9555556 0.9111111 0.9777778 0.9777778 0.9555556 0.9555556 0.9333333 0.9333333 0.9777778 0.8888889 0.8666667 0.9333333 0.9111111
 [35] 0.9555556 0.9555556 0.9777778 0.9333333 0.9777778 0.8444444 0.9555556 1.0000000 0.9555556 0.9111111 0.9111111 0.9777778 0.9777778 0.9777778 0.9777778 0.9555556 0.9777778
 [52] 1.0000000 0.9111111 0.9777778 0.9333333 0.9777778 0.9777778 0.9111111 0.9333333 1.0000000 0.9777778 0.9777778 0.9777778 1.0000000 1.0000000 0.9555556 0.9333333 0.9333333
 [69] 0.9111111 0.9111111 0.9111111 0.9333333 0.9777778 0.9333333 0.9777778 0.8888889 0.9555556 1.0000000 0.9777778 0.9333333 0.9777778 0.8666667 0.9333333 1.0000000 1.0000000
 [86] 0.9555556 0.9555556 0.9777778 0.9777778 0.9555556 0.9777778 0.9777778 0.9555556 0.9555556 0.9333333 0.9555556 0.9111111 0.9555556 0.9555556 0.9333333
> mean(accuracy)
[1] 0.9531111
> mean(precision_snow)
[1] 0.889
> mean(precision_rain)
[1] 0.9714286
>
```

**Summary:**

| Model | Accuracy | Precision Snow | Precision Rain | Recall Snow | Recall Rain |
|---|---|---|---|---|---|
| **Logistic Regression** | 0.9531111 | 0.889 | 0.9714286 | 0.9075805 | 0.9688803 |
| **LDA** | 0.936222 | 0.865 | 0.9565714 | 0.8595878 | 0.9618026 |
| **QDA** | 0.9346667 | 0.952 | 0.9297143 | 0.8043408 | 0.985939 |
| **KNN (K=5)** | 0.9486667 | 0.878 | 0.9688571 | 0.89958469 | 0.9658621 |

**Text Summarization:**

```
#As you can see accuracy of LR>KNN>LDA>QDA, we should opt for Logistic Regression

#As you can see Precision of Snow, QDA>LR>KNN>LDA
#As you can see Precision of rain, LR>KNN>LDA>QDA
```

#As you can see Recall of Snow, LR>KNN>LDA>QDA
#As you can see Recall of rain, LR>KNN>LDA>QDA

###decision boundary is not very highly non-linear in this case, so we see Logistic Regression dominating KNN and LDA.

###KNN is a completely non-parametric approach: no assumptions are made about the shape of the decision boundary. Therefore, we can expect this approach to dominate LDA and logistic regression when the decision boundary is highly non-linear. On the other hand, KNN does not tell us which predictors are important; we don’t get a table of coefficients with p-values

###QDA serves as a compromise between KNN, LDA and LR, In this case there are more number of training observations, so QDA doesnt perform well

###***So based on results, Logestic regression performs better in accuracy, precision and recall, so we choose Logestic Regression model on this dataset****###

## Question 1

You're to use the KC Weather Data ("kc_weather_srt.csv", available here: kc_weather_srt.csv ). The data has categorized the weather for each day into three categories ("Events": Rain, Rain_Thunderstorm, Snow) over the three years 2014, 2015, and 2016. You'll note that not all dates are listed because it's a filtered subset where other categories or no events are deleted to have a more manageable subset. The entire dataset has 366 entries. The column labels indicate the units as well such as Temp.F means temperature in Fahrenheit, Visibility.mi means Visibility in miles, etc.
You're to do two level of analysis

a. Consider first the subset that consists only of Rain and Snow. There are 226 entries with these two categories.

ii. Discuss and analyze in a systematic way you would consider eliminating some of the predictors and see if your accuracy, precision and recall improves.

```
Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.70252  -0.00785  -0.00003   0.00000   1.48115

Coefficients:
                     Estimate Std. Error z value Pr(>|z|)
(Intercept)         -211.2210   120.0506  -1.759   0.0785 .
Temp.F                 0.4339     0.3522   1.232   0.2180
Dew_Point.F           -0.9378     0.4648  -2.018   0.0436 *
Humidity.percentage    0.2926     0.1880   1.556   0.1196
Sea_Level_Press.in     6.5689     3.9826   1.649   0.0991 .
Visibility.mi         -0.1174     0.5849  -0.201   0.8410
Wind.mph               0.2153     0.1633   1.319   0.1873
Precip.in           -150.0516    80.0575  -1.874   0.0609 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 191.195  on 180  degrees of freedom
Residual deviance:  26.556  on 173  degrees of freedom
AIC: 42.556

Number of Fisher Scoring iterations: 12

> |
```

*From the first problem, when I performed Summary of Logistic Regression, I found that p values of Temp.f, humidity%, visibility and wind are high. So, I removed these predictors and just took Dewpoint, SealevelPress, and Precip.in predictors since their p values are considerably lower.*

*###Removing some predictors after noticing p-values from summary of Logistic Regression table###*

```r
###Taking only Dewpoint, SeaLevel and Precipitation predictors, removing rest. Let's hope for the best results.
```

```r
#import the dataset and make some changes
library(readr)
kc_weather_srt <- read_csv("C:/Users/bvkka/Desktop/ISL-Deep Medhi/kc_weather_srt.csv")

kc_weather_srt=kc_weather_srt[c(2,4,7,8,9)]
```

| | Dew_Point.F | Sea_Level_Press.in | Precip.in | Events |
|---|---|---|---|---|
| 8 | 11 | 30.14 | 0.00 | Rain |
| 9 | 2 | 30.13 | 0.00 | Snow |
| 10 | -3 | 30.37 | 0.00 | Snow |
| 11 | 3 | 30.19 | 0.00 | Snow |
| 12 | 14 | 29.82 | 0.00 | Snow |
| 13 | 14 | 30.04 | 0.00 | Snow |
| 14 | 0 | 30.50 | 0.00 | Snow |
| 15 | 12 | 30.31 | 0.01 | Snow |
| 16 | 28 | 29.82 | 0.00 | Rain |
| 17 | 33 | 29.81 | 0.15 | Rain_Thunderstorm |
| 18 | 20 | 29.88 | 0.00 | Rain |

```r
#subset that consists of only rain and snow
kc_weather_srt_without_rainthunderstorm<-kc_weather_srt[!grepl("Rain_Thunderstorm",kc_weather_srt$Events),]

dim(kc_weather_srt_without_rainthunderstorm)
```

| | Dew_Point.F | Sea_Level_Press.in | Precip.in | Events |
|---|---|---|---|---|
| 7 | 19 | 30.02 | 0.00 | Rain |
| 8 | 11 | 30.14 | 0.00 | Rain |
| 9 | 2 | 30.13 | 0.00 | Snow |
| 10 | -3 | 30.37 | 0.00 | Snow |
| 11 | 3 | 30.19 | 0.00 | Snow |
| 12 | 14 | 29.82 | 0.00 | Snow |
| 13 | 14 | 30.04 | 0.00 | Snow |
| 14 | 0 | 30.50 | 0.00 | Snow |
| 15 | 12 | 30.31 | 0.01 | Snow |
| 16 | 28 | 29.82 | 0.00 | Rain |
| 17 | 20 | 29.88 | 0.00 | Rain |
| 18 | 11 | 30.38 | 0.00 | Snow |
| 19 | 11 | 30.29 | 0.03 | Snow |
| 20 | -5 | 30.50 | 0.08 | Snow |

```
#first make the response column to 0 and 1
#install.packages("plyr")
library(plyr)
kc_weather_srt_without_rainthunderstorm$Events <-
revalue(kc_weather_srt_without_rainthunderstorm$Events,c("Snow"=1))
kc_weather_srt_without_rainthunderstorm$Events <-
revalue(kc_weather_srt_without_rainthunderstorm$Events,c("Rain"=0))
```

|  | Dew_Point.F | Sea_Level_Press.in | Precip.in | Events |
|---|---|---|---|---|
| 7 | 19 | 30.02 | 0.00 | 0 |
| 8 | 11 | 30.14 | 0.00 | 0 |
| 9 | 2 | 30.13 | 0.00 | 1 |
| 10 | -3 | 30.37 | 0.00 | 1 |
| 11 | 3 | 30.19 | 0.00 | 1 |
| 12 | 14 | 29.82 | 0.00 | 1 |
| 13 | 14 | 30.04 | 0.00 | 1 |
| 14 | 0 | 30.50 | 0.00 | 1 |
| 15 | 12 | 30.31 | 0.01 | 1 |
| 16 | 28 | 29.82 | 0.00 | 0 |
| 17 | 20 | 29.88 | 0.00 | 0 |
| 18 | 11 | 30.38 | 0.00 | 1 |

```
#character to numeric
kc_weather_srt_without_rainthunderstorm$Events<-
as.numeric(as.character(kc_weather_srt_without_rainthunderstorm$Events))



# number of replications
rep=100



# newly added

#snow=1 rain=0
accuracy=dim(rep)
accuracy1=dim(rep)
accuracy2=dim(rep)
accuracy3=dim(rep)

precision_snow=dim(rep)
precision_rain=dim(rep)
recall_snow=dim(rep)
recall_rain=dim(rep)
```

```
precision_snow1=dim(rep)
precision_rain1=dim(rep)
recall_snow1=dim(rep)
recall_rain1=dim(rep)

precision_snow2=dim(rep)
precision_rain2=dim(rep)
recall_snow2=dim(rep)
recall_rain2=dim(rep)


precision_snow3=dim(rep)
precision_rain3=dim(rep)
recall_snow3=dim(rep)
recall_rain3=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_without_rainthunderstorm$Events,SplitRatio = 0.8) #80% split ratio
  training_set=subset(kc_weather_srt_without_rainthunderstorm,split==TRUE) #80% split into training set
  test_set=subset(kc_weather_srt_without_rainthunderstorm,split==FALSE) #20% split into testing set
  table(split)
  dim(training_set)
```

```
#****Logistic Regression***#
#Fitting Logistic Regression to the Training set
model1<-glm(formula = Events ~ ., binomial(link="logit"),data =training_set)

#predicting the test set results
prob_pred=predict(model1,type='response',newdata = test_set[-4])   #for predicitng we only need predictors , but
not response
summary(prob_pred)
y_pred=ifelse(prob_pred>0.5,1,0) #vector of predictions
y_pred

#Making the confusion Matrix
cm=table(y_pred,test_set[,4])
cm

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


#precision

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

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

```
#***LDA***#
#install.packages("MASS")
library(MASS)



lda=lda(formula=Events~.,data=training_set)
y_pred1=predict(lda,test_set)$class
cm1=table(y_pred1,test_set[,4])
cm1
accuracy1[k]=mean(y_pred1==test_set[,4])

precision1=precison1<-diag(cm1/colSums(cm1))
precision_snow1[k]=precison1[2]
precision_rain1[k]=precision1[1]
recall1=recall1<-diag(cm1/rowSums(cm1))
recall_snow1[k]=recall1[2]
recall_rain1[k]=recall1[1]









#a=lda(formula=Events~.,data=training_set)
#training_set1=as.data.frame(predict(lda,training_set))
#training_set1=training_set1[c(4,1)]
#plot(lda)

#test_set1=as.data.frame(predict(lda,test_set))
#test_set1=test_set1[c(4,1)]

#fitting SVM to the training set
#install.packages('e1071')
#library(e1071)
#classifier=svm(formula=class~.,data=training_set1,type='C-classification',kernel='linear')
```

```
#predicting the test set results
#y_pred1=predict(classifier,newdata = test_set1[-2])
#making the confusion matrix
#cm1=table(y_pred1,test_set1[,2])
#cm1 #we see TP+TN=33+12=45 true predictions and FP+FN=0+0=0 False predictions, therefore it is 100% accurate
#accuracy1[k]=mean(y_pred1==test_set1[,2])
#accuracy1




#***QDA***#
qda=qda(formula=Events~.,data=training_set)
y_pred2=predict(qda,test_set)$class
cm2=table(y_pred2,test_set[,4])
cm2
accuracy2[k]=mean(y_pred2==test_set[,4])

precision2=precison2<-diag(cm2/colSums(cm2))
precision_snow2[k]=precison2[2]
precision_rain2[k]=precision2[1]
recall2=recall2<-diag(cm2/rowSums(cm2))
recall_snow2[k]=recall2[2]
recall_rain2[k]=recall2[1]




#***KNN***#
#install.packages('class')
#fitting KNN to the training set and predicting the test set results
library(class)
y_pred3=knn(train=training_set[,-4],test=test_set[,-4],cl=training_set[,4],k=5)

#making the cm
cm3=table(y_pred3,test_set[,4])
cm3
accuracy3[k]=mean(y_pred3==test_set[,4])
```

```
  precision3=precision3<-diag(cm3/colSums(cm3))
  precision_snow3[k]=precision3[2]
  precision_rain3[k]=precision3[1]

  recall3=recall3<-recall3<-diag(cm3/rowSums(cm3))
  recall_snow3[k]=recall3[2]
  recall_rain3[k]=recall3[1]
}



###*******MEAN VALUES*****#####

#****Logestic regression***#
mean(accuracy)###0.95022
mean(precision_snow)###0.891
mean(precision_rain)###0.967

mean(recall_snow) ### 0.8958132
mean(recall_rain) ### 0.9697611



#****LDA***#
mean(accuracy1)###0.95
mean(precision_snow1)###0.9
mean(precision_rain1)###0.9642857

mean(recall_snow1) ### 0.866313
mean(recall_rain1) ### 0.971967
```

```r
#****QDA***#
mean(accuracy2)###0.9088889
mean(precision_snow2)###0.938
mean(precision_rain2)###0.9005714

mean(recall_snow2) ### 0.7419847
mean(recall_rain2) ### 0.9813024



#****KNN***#
mean(accuracy3)###0.94355556
mean(precision_snow3)###0.8666
mean(precision_rain3)###0.9657143

mean(recall_snow3) ### 0.8895922
mean(recall_rain3) ### 0.96284
```

```
> mean(recall_snow1)
[1] 0.8866313
> mean(recall_rain1)
[1] 0.971967
> mean(accuracy2)
[1] 0.9088889
> mean(precision_snow2)
[1] 0.938
> mean(precision_rain2)
[1] 0.9005714
> mean(recall_snow2)
[1] 0.7419847
> mean(recall_rain2)
[1] 0.9813024
> mean(accuracy3)
[1] 0.9435556
> mean(precision_snow3)
[1] 0.866
> mean(precision_rain3)
[1] 0.9657143
> mean(recall_snow3)
[1] 0.8895922
> mean(recall_rain3)
[1] 0.96284
> mean(accuracy1)
[1] 0.95
>
```

```
Global Environment
prob_pred     Named num [1:45] 0.09659 0.99972 0.9...
qda           List of 10
recall        Named num [1:2] 0.971 0.9
recall_rain   num [1:100] 0.971 0.943 0.971 0.971 ...
recall_rain1  num [1:100] 0.971 0.943 0.971 0.971 ...
recall_rain2  num [1:100] 1 0.969 1 0.97 0.968 ...
recall_rain3  num [1:100] 0.972 0.943 0.971 1 0.94...
recall_snow   num [1:100] 0.9 0.8 0.9 0.9 0.875 ...
recall_snow1  num [1:100] 0.9 0.8 0.818 0.9 0.875 ...
recall_snow2  num [1:100] 0.769 0.692 0.667 0.75 0...
recall_snow3  num [1:100] 1 0.8 0.9 0.909 0.889 ...
recall1       Named num [1:2] 0.971 0.9
recall2       Named num [1:2] 0.968 0.643
recall3       Named num [1:2] 0.944 0.889
rep           100
split         logi [1:226] TRUE TRUE TRUE TRUE FAL...
y_pred        Named num [1:45] 0 1 1 0 0 0 0 0 0 0...
```

**Summary:**

| Model | Accuracy | Precision Snow | Precision Rain | Recall Snow | Recall Rain |
|---|---|---|---|---|---|
| **Logistic Regression** | 0.95022 | 0.891 | 0.967 | 0.8958132 | 0.9697611 |
| **LDA** | 0.95 | 0.9 | 0.9642857 | 0.866313 | 0.971967 |
| **QDA** | 0.9088889 | 0.938 | 0.9005714 | 0.7419847 | 0.9813024 |
| **KNN (K=5)** | 0.94355556 | 0.86666 | 0.9657143 | 0.8895922 | 0.96284 |

**Text Summarization:**

*#As you can see accuracy of LDA=LR>KNN>QDA, we should opt for LR*

*#As you can see Precision of Snow , QDA>LDA>LR>KNN  QDA gives highest*
*#As you can see Precision of rain , KNN=LR>LDA >QDA   LR gives highest*

*#As you can see Recall of Snow , KNN~=LR>LDA>QDA   LR gives highest*
*#As you can see Recall of rain , QDA>LDA>LR>KNN  ,,,QDA gives highest, outperforms LR, KNN and LR*

*## QDA can perform better in the presence of a limited number of training observations because it does make some assumptions about the form of the decision boundary*

*###***So based on results, QDA AND LR are close but we choose QDA model on this dataset****###*

**Question 1**

You're to use the KC Weather Data ("kc_weather_srt.csv", available here: kc_weather_srt.csv ). The data has categorized the weather for each day into three categories ("Events": Rain, Rain_Thunderstorm, Snow) over the three years 2014, 2015, and 2016. You'll note that not all dates are listed because it's a filtered subset where other categories or no events are deleted to have a more manageable subset. The entire dataset has 366 entries. The column labels indicate the units as well such as Temp.F means temperature in Fahrenheit, Visibility.mi means Visibility in miles, etc.
You're to do two level of analysis

b. Consider next the entire dataset consisting of 366 entries. Now logistics regression cannot be applied, but you can apply the rest of them. Repeat the above studies in i) and ii) with LDA, QDA, and knn on the entire data set (using 290 of them in a training set). Do not forget randomization and 100 replications for this study.

```
#import the dataset and make some changes
library(readr)
kc_weather_srt <- read_csv("C:/Users/bvkka/Desktop/ISL-Deep Medhi/kc_weather_srt.csv")

kc_weather_srt=kc_weather_srt[,2:9]
```

| | Temp.F | Dew_Point.F | Humidity.percentage | Sea_Level_Press.in | Visibility.mi | Wind.mph | Precip.in | Events |
|---|---|---|---|---|---|---|---|---|
| 17 | 44 | 33 | 66 | 29.81 | 10 | 5 | 0.15 | Rain_Thunderstorm |
| 18 | 44 | 20 | 41 | 29.88 | 10 | 11 | 0.00 | Rain |
| 19 | 22 | 11 | 59 | 30.38 | 10 | 11 | 0.00 | Snow |
| 20 | 21 | 11 | 76 | 30.29 | 5 | 14 | 0.03 | Snow |
| 21 | 5 | -5 | 65 | 30.50 | 5 | 14 | 0.08 | Snow |
| 22 | 36 | 25 | 72 | 30.20 | 7 | 4 | 0.00 | Snow |
| 23 | 54 | 23 | 29 | 29.87 | 10 | 13 | 0.00 | Rain |
| 24 | 55 | 36 | 50 | 29.88 | 9 | 7 | 0.38 | Rain_Thunderstorm |
| 25 | 34 | 14 | 53 | 30.47 | 8 | 9 | 0.00 | Snow |
| 26 | 34 | 23 | 68 | 30.26 | 6 | 5 | 0.00 | Snow |
| 27 | 46 | 25 | 53 | 30.00 | 9 | 11 | 0.07 | Rain |
| 28 | 55 | 42 | 68 | 29.57 | 10 | 16 | 0.00 | Rain_Thunderstorm |

```
#first make the response column to 0-snow, 1-rain and 2-rain_thunderstorm
```

```
#install.packages("plyr")
library(plyr)
kc_weather_srt$Events <- revalue(kc_weather_srt$Events,c("Snow"=1))
kc_weather_srt$Events <- revalue(kc_weather_srt$Events,c("Rain"=0))
kc_weather_srt$Events <- revalue(kc_weather_srt$Events,c("Rain_Thunderstorm"=2))
```

| | Temp.F | Dew_Point.F | Humidity.percentage | Sea_Level_Press.in | Visibility.mi | Wind.mph | Precip.in | Events |
|---|---|---|---|---|---|---|---|---|
| 16 | 55 | 28 | 38 | 29.82 | 10 | 12 | 0.00 | 0 |
| 17 | 44 | 33 | 66 | 29.81 | 10 | 5 | 0.15 | 2 |
| 18 | 44 | 20 | 41 | 29.88 | 10 | 11 | 0.00 | 0 |
| 19 | 22 | 11 | 59 | 30.38 | 10 | 11 | 0.00 | 1 |
| 20 | 21 | 11 | 76 | 30.29 | 5 | 14 | 0.03 | 1 |
| 21 | 5 | -5 | 65 | 30.50 | 5 | 14 | 0.08 | 1 |
| 22 | 36 | 25 | 72 | 30.20 | 7 | 4 | 0.00 | 1 |
| 23 | 54 | 23 | 29 | 29.87 | 10 | 13 | 0.00 | 0 |
| 24 | 55 | 36 | 50 | 29.88 | 9 | 7 | 0.38 | 2 |
| 25 | 34 | 14 | 53 | 30.47 | 8 | 9 | 0.00 | 1 |
| 26 | 34 | 23 | 68 | 30.26 | 6 | 5 | 0.00 | 1 |
| 27 | 46 | 25 | 53 | 30.00 | 9 | 11 | 0.07 | 0 |
| 28 | 55 | 42 | 68 | 29.57 | 10 | 16 | 0.00 | 2 |

```
#small changes to Events column , making it to numeric from character
kc_weather_srt$Events<-as.numeric(as.character(kc_weather_srt$Events))

#replications
rep=100



# newly added

#snow=1 rain=0 thunderstorm=2
accuracy1=dim(rep)
accuracy2=dim(rep)
accuracy3=dim(rep)


precision_snow1=dim(rep)
```

```
precision_rain1=dim(rep)
precision_rainThunderstorm1=dim(rep)

recall_snow1=dim(rep)
recall_rain1=dim(rep)
recall_rainThunderstorm1=dim(rep)

precision_snow2=dim(rep)
precision_rain2=dim(rep)
precision_rainThunderstorm2=dim(rep)
recall_snow2=dim(rep)
recall_rain2=dim(rep)
recall_rainThunderstorm2=dim(rep)

precision_snow3=dim(rep)
precision_rain3=dim(rep)
precision_rainThunderstorm3=dim(rep)
recall_snow3=dim(rep)
recall_rain3=dim(rep)
recall_rainThunderstorm3=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)
```

Data

| | |
|---|---|
| kc_weather_srt | 366 obs. of 8 variables |
| test_set | 76 obs. of 8 variables |
| training_set | 290 obs. of 8 variables |

Values

```r
#***LDA***#
#install.packages("MASS")
library(MASS)
lda=lda(formula=Events~.,data=training_set)
training_set1=as.data.frame(predict(lda,training_set))
training_set1=training_set1[c(4,1)]
plot(lda)

test_set1=as.data.frame(predict(lda,test_set))
test_set1=test_set1[c(4,1)]

#fitting SVM to the training set
#install.packages('e1071')
library(e1071)
classifier=svm(formula=class~.,data=training_set1,type='C-classification',kernel='linear')
#predicting the test set results
y_pred1=predict(classifier,newdata = test_set1[-2])
#making the confusion matrix
cm1=table(y_pred1,test_set1[,2])

accuracy1[k]=mean(y_pred1==test_set1[,2])


precision1=precision1<-diag(cm1)/colSums(cm1)
precision_rainThunderstorm1[k]=precision1[3]
precision_snow1[k]=precision1[2]
precision_rain1[k]=precision1[1]
```

```
recall1=recall1<-diag(cm1/rowSums(cm1))
recall_rainThunderstorm1[k]=recall1[3]
recall_snow1[k]=recall1[2]
recall_rain1[k]=recall1[1]





#***QDA***#

qda=qda(formula=Events~.,data=training_set)
y_pred2=predict(qda,test_set)$class
cm2=table(y_pred2,test_set[,8])

accuracy2[k]=mean(y_pred2==test_set[,8])

precision2=precison2<-diag(cm2/colSums(cm2))
precision_rainThunderstorm2[k]=precison2[3]
precision_snow2[k]=precison2[2]
precision_rain2[k]=precision2[1]
recall2=recall2<-diag(cm2/rowSums(cm2))
recall_rainThunderstorm2[k]=recall2[3]
recall_snow2[k]=recall2[2]
recall_rain2[k]=recall2[1]





#***KNN***#
#install.packages('class')
#fitting KNN to the training set and predicting the test set results
library(class)
y_pred3=knn(train=training_set[,-8],test=test_set[,-8],cl=training_set[,8],k=5)

#making the cm
cm3=table(y_pred3,test_set[,8])

accuracy3[k]=mean(y_pred3==test_set[,8])
```

```
precision3=precision3<-diag(cm3/colSums(cm3))
precision_rainThunderstorm3[k]=precision3[3]
precision_snow3[k]=precision3[2]
precision_rain3[k]=precision3[1]

recall3=recall3<-recall3<-diag(cm3/rowSums(cm3))
recall_rainThunderstorm3[k]=recall3[3]
recall_snow3[k]=recall3[2]
recall_rain3[k]=recall3[1]
}

###*******MEAN VALUES*****#####


#****LDA***#
mean(accuracy1)###0.9026316
mean(precision_snow1)###0.6407459
mean(precision_rain1)###0.9115871
mean(precision_rainThunderstorm1)###0.9906168
mean(recall_snow1) ### 0.911675
mean(recall_rain1) ### 0.902705
mean(recall_rainThunderstorm1) ### 0.9906152






#****QDA***#
mean(accuracy2)###0.7389474
mean(precision_snow2)###0.945
mean(precision_rain2)###0.697027
mean(precision_rainThunderstorm2)###0.7213793
mean(recall_snow2) ### 0.7950919
mean(recall_rain2) ### 0.7514844
mean(recall_rainThunderstorm2) ### 0.7115056
```

```
#****KNN***#
mean(accuracy3)###0.745
mean(precision_snow3)###0.895
mean(precision_rain3)###0.7305405
mean(precision_rainThunderstorm3)###0.7117241
mean(recall_snow3) ### 0.9098042
mean(recall_rain3) ### 0.7444542
mean(recall_rainThunderstorm3) ### 0.701065
```

**Summary:**

| Model | Accuracy | Precision Snow | Precision Rain | Precision Rain Thunderstorm | Recall Snow | Recall Rain | Recall ThunderStorm |
|---|---|---|---|---|---|---|---|
| **LDA** | 0.9026316 | 0.6407459 | 0.9115871 | 0.9906168 | 0.911675 | 0.902705 | 0.9906152 |
| **QDA** | 0.7389474 | 0.945 | 0.697027 | 0.7213793 | 0.7950919 | 0.7514844 | 0.7115056 |
| **KNN (K=5)** | 0.745 | 0.895 | 0.7305405 | 0.7117241 | 0.9098042 | 0.7444542 | 0.701065 |

**Text Summarization:**

```
#As you can see accuracy of LDA>KNN>QDA , we should opt for LDA
```

```
#As you can see Precision of Snow , LDA gives highest
#As you can see Precision of rain , LDA gives highest, outperforms KNN and QDA
#As you can see Precision of rainthunderstorm , LDA gives highest, outperforms
KNN and QDA

#As you can see Recall of Snow , LDA gives highest
#As you can see Recall of rain , LDA gives highest, outperforms KNN and QDA
#As you can see Recall of rainthunderstorm , LDA gives highest, outperforms KNN
and QDA

###***So based on results, we choose LDA model on this dataset****###
```