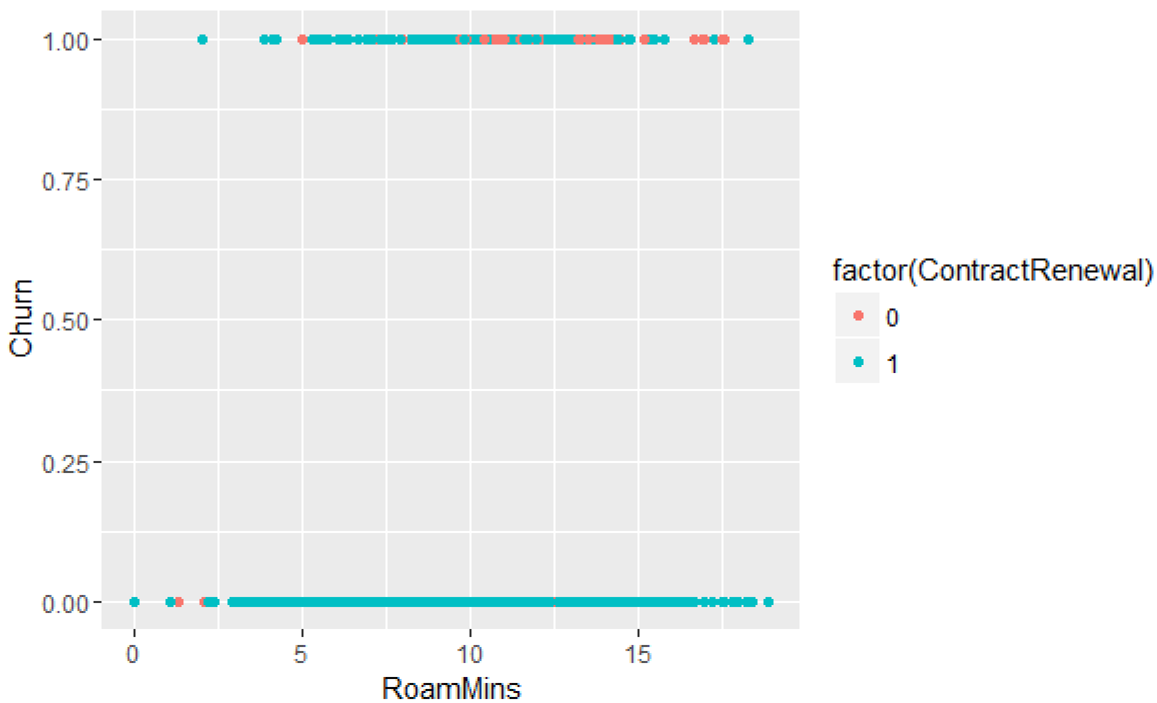
Assignment 3

a.

The overall churn rate is 14.49 % (0.14491449). This has been obtained using excel.

b. Using the training dataset, the scatterplot of Churn on the Y axis and RoamMin on the x axis has been generated by using ContractRenewal as a color.



c.

The confusion table is :

```
          Reference
Prediction    0    1
```

```
            0 1115  187
            1   11   20
```

Accuracy rate is 0.851

Misclassification error is 0.149

Sensitivity value is 0.097

d.

Contract renewal has the highest correlation value with churn , which is (-0.2718).

This means that if a contract is renewed, it will result in a decrease in the log odds of churn by 0.2718.

```
          Reference
Prediction    0    1
         0 1099  168
         1   27   39
```

The accuracy rate is 0.854

Misclassification error is 0.146

Sensitivity value is 0.188

e.

The confusion matrix is as follows :

```
          Reference
Prediction    0    1
         0 1098  166
         1   28   41
```

Accuracy rate is 0.854

Misclassification error is 0.146

Sensitivity value is 0.198

f.

Given average Net Loss Table is :

	Reference	
Prediction	0	1
0	$0	$200
1	$50	$0

For part c :

```
Reference
Prediction    0    1
         0 1115  187
         1   11   20
```

Average net loss = (187*200) + (11*50) = $37950

For part d:

```
          Reference
Prediction    0    1
         0 1099  168
         1   27   39
```

Average net loss = (168*200) + (27*50) = $34950

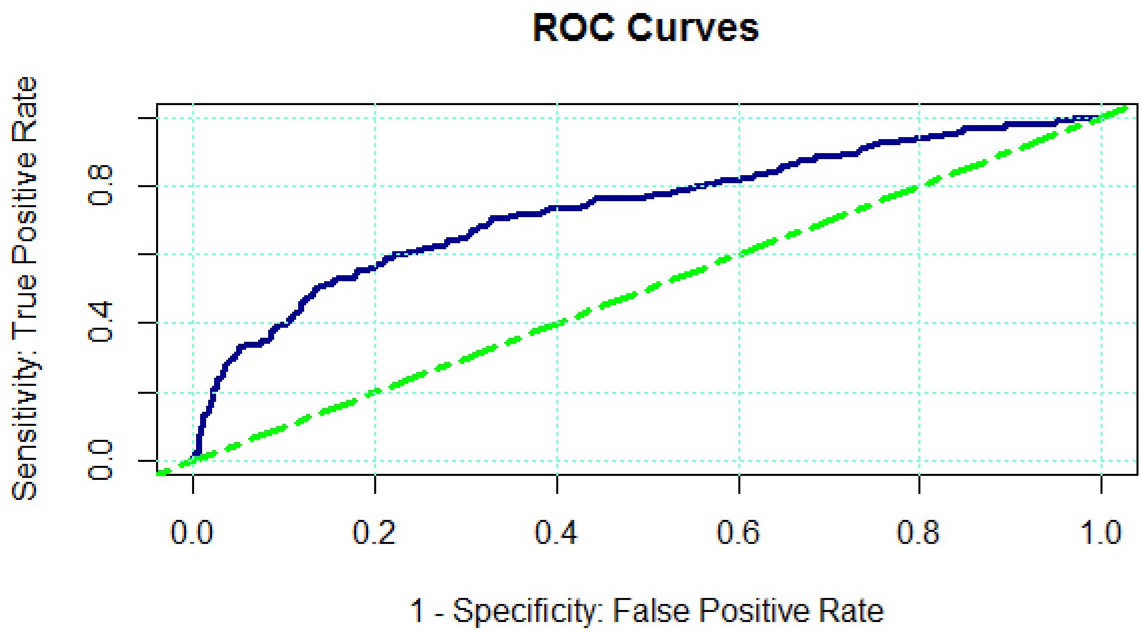
For Part e:

	Reference	
Prediction	0	1
0	1098	166
1	28	41

Average net loss = (166*200) + (28*50) = $34600

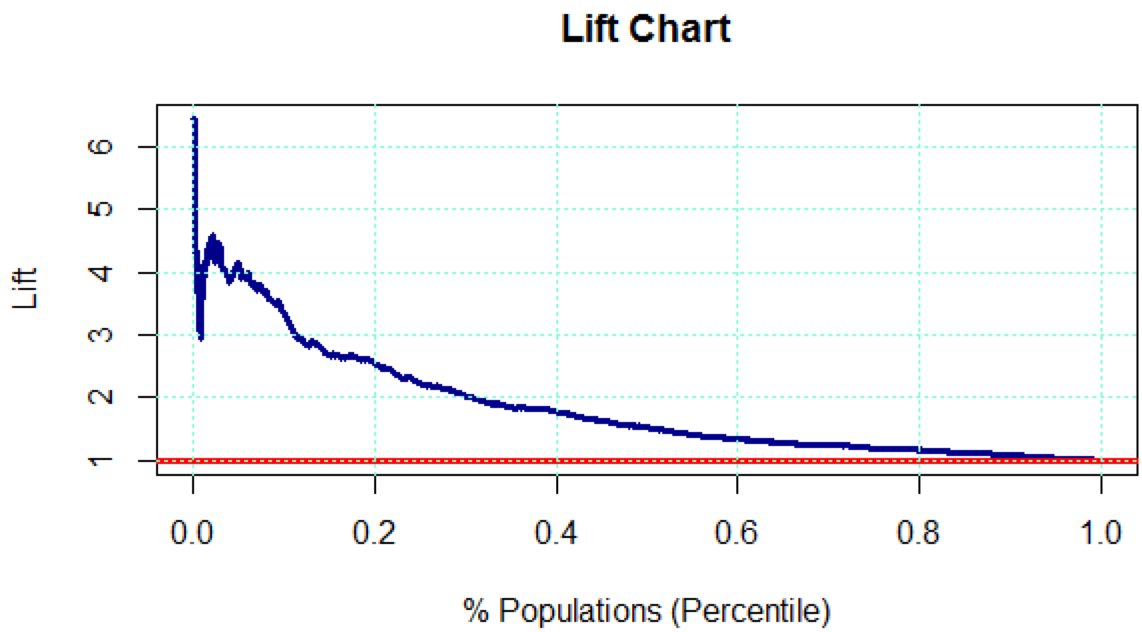
The loss is the least in the case of part e , therefore this is what we consider.

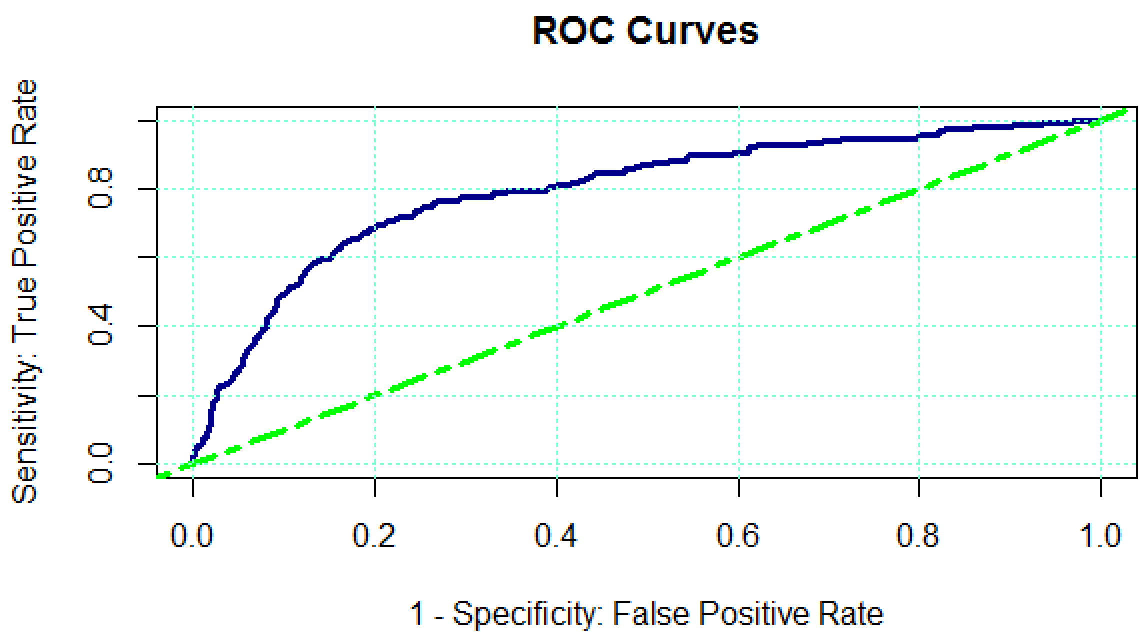
g.



ROC curve for part C

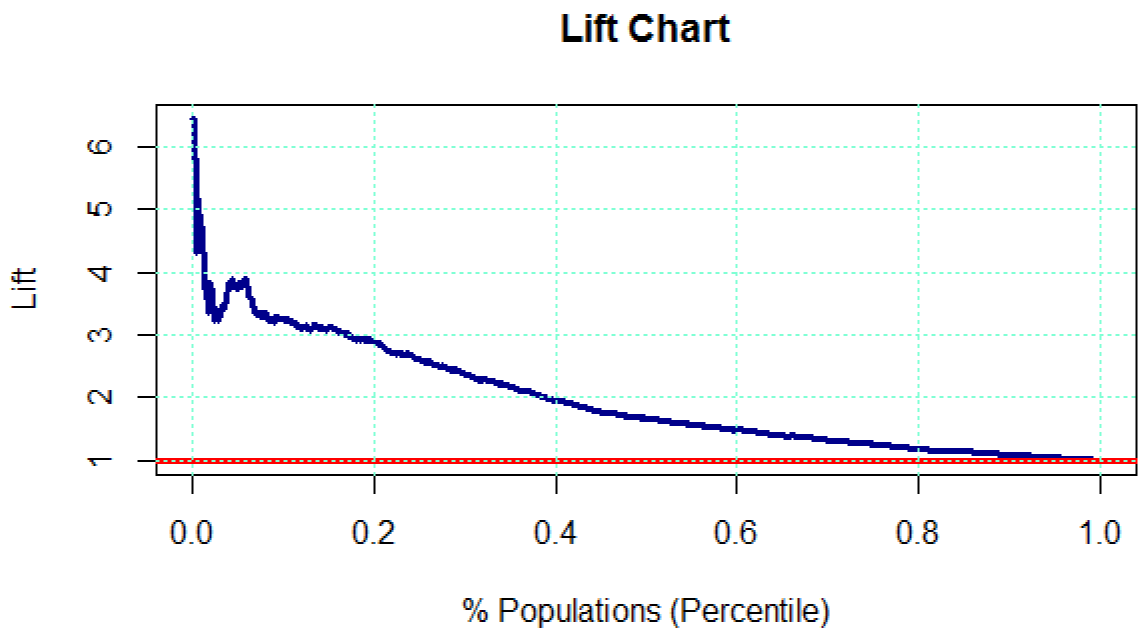
The AUC value is 0.7376 .



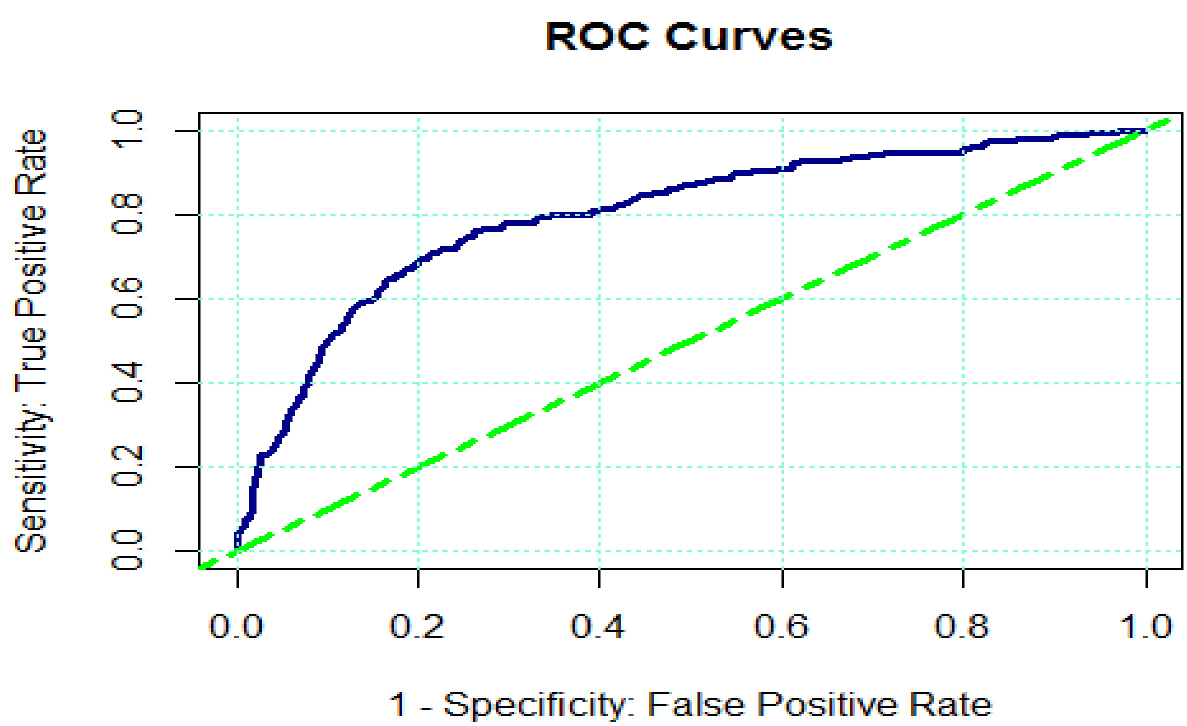


ROC curve for part D

The AUC value is 0.795.

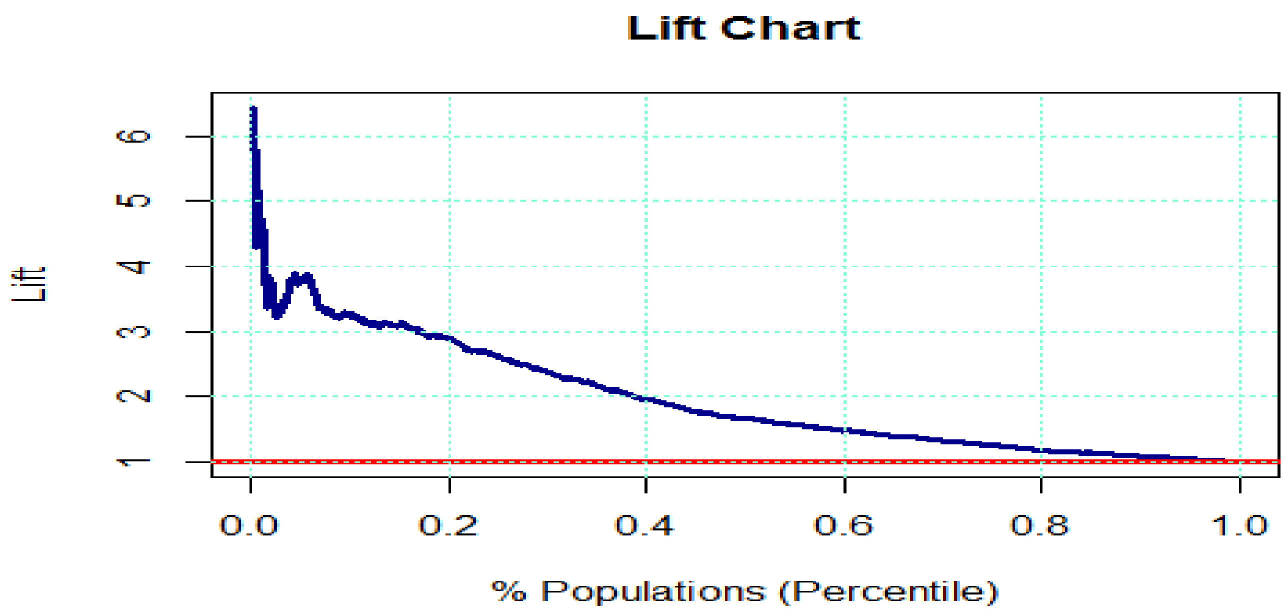


e



ROC curve for part E .

AUC value is 0.7984



The model used in Part (e) has the highest AUC value, hence that is the best model.

h.

Accuracy for 0.1 to 0.9 :

```
[1,] 0.6654164
[2,] 0.8087022
[3,] 0.8402101
[4,] 0.8469617
[5,] 0.8544636
[6,] 0.8462116
[7,] 0.8477119
[8,] 0.8492123
[9,] 0.8469617
```

The maximum accuracy is for cut off value of 0.5 , which makes it the best cutoff value.

Appendix

```
rm(list = ls())

install.packages('caret', dependencies = TRUE)

install.packages('ggplot2', dependencies = TRUE)

install.packages("ROCR", dependencies = TRUE)

library(ggplot2)

library(caret)

library(ROCR)



data=read.csv('C://Users//SOURAV//Desktop//Churn.csv')



#problem b

set.seed(12345)

n=nrow(data)
```

```
trainindex <- sample(n, round(0.6*n), replace=FALSE)

training=data[trainindex,]

validation=data[-trainindex,]

ggplot(training,aes(x=RoamMins,y=Churn))+geom_point(aes(color = factor(ContractRenewal)))



#problem c

#to get numerical rows

num_data=data.frame(training)

num_data <- subset(num_data, select = -c( ContractRenewal,          DataPlan) )



#logistic regression fit line

fit_c=glm(Churn~.,family= binomial, data=num_data)

summary(fit_c)



#for the validation data
```

```
val<-predict(fit_c,validation,type="response")

validationdf <-cbind(validation,val)

validationdf$response <- as.factor(ifelse(validationdf$val>0.5, 1, 0))

confmatrix=confusionMatrix(data=factor(validationdf$response),
reference=factor(validationdf$Churn), positive='1')


#confusion table

confmatrix$table


#accuracy rate

round(confmatrix$overall[1],3)


# Misclassification error

round(1-confmatrix$overall[1],3)
```

```
#sensitivity value

round(confmatrix$byClass[1],3)




#problem d



fit_d=glm(Churn~AccountWeeks + ContractRenewal + DataPlan +

DataUsage + CustServCalls +DayMins + DayCalls + MonthlyCharge

+ OverageFee + RoamMins,family= binomial, data=training)

summary(fit_d)

cor(training)



val<-predict(fit_d,validation,type="response")

mydf <-cbind(validation,val)

mydf$response <- as.factor(ifelse(mydf$val>0.5, 1, 0))
```

```
confmatrix_d=confusionMatrix(data=factor(mydf$response),       reference=factor(mydf$Churn),
positive='1')


#confusion matrix

confmatrix_d$table

#Overall accuracy rate

round(confmatrix_d$overall[1],3)

#Misclassification error

round(1-confmatrix_d$overall[1],3)

#Sensitivity Value

confmatrix_d$byClass[1]



#problem E
```

```
fit_e=step(fit_d,direction="backward")

summary(fit_e)


val<-predict(fit_e,validation,type="response")

mydf_e <-cbind(validation,val)

mydf_e$response <- as.factor(ifelse(mydf_e$val>0.5, 1, 0))

cm_E=confusionMatrix(data=factor(mydf_e$response),            reference=factor(mydf_e$Churn),
positive='1')


#Confusion Table

cm_E$table


#Overall accuracy rate

round(cm_E$overall[1],3)

#Misclassification error
```

```
round(1-cm_E$overall[1],3)

#Sensitivity Value

round(cm_E$byClass[1],3)

# problem g

# for part c

logit_scores <- prediction(predictions=validationdf$val, labels=validationdf$Churn)

#PLOT ROC CURVE

logit_perf <- performance(logit_scores, "tpr", "fpr")
```

```
plot(logit_perf,

     main="ROC Curves",

     xlab="1 - Specificity: False Positive Rate",

     ylab="Sensitivity: True Positive Rate",

     col="darkblue",  lwd = 3)

abline(0,1, lty = 300, col = "green",  lwd = 3)

grid(col="aquamarine")




# AREA UNDER THE CURVE

logit_auc <- performance(logit_scores, "auc")

as.numeric(logit_auc@y.values)  ##AUC Value
```

```
##############################################################################
#################



# Getting Lift Charts in R

# For getting Lift Chart in R, use measure="lift", x.measure="rpp" in the performance function.

# Get data for ROC curve and create lift chart values

logit_lift <- performance(logit_scores, measure="lift", x.measure="rpp")

plot(logit_lift,

     main="Lift Chart",

     xlab="% Populations (Percentile)",

     ylab="Lift",

     col="darkblue", lwd = 3)

abline(1,0,col="red",  lwd = 3)

grid(col="aquamarine")
```

```
# for part d

##########################################################

logit_scores <- prediction(predictions=mydf$val, labels=mydf$Churn)

#PLOT ROC CURVE

logit_perf <- performance(logit_scores, "tpr", "fpr")

plot(logit_perf,
     main="ROC Curves",
     xlab="1 - Specificity: False Positive Rate",
     ylab="Sensitivity: True Positive Rate",
     col="darkblue",  lwd = 3)
abline(0,1, lty = 300, col = "green",  lwd = 3)
```

```
grid(col="aquamarine")



# AREA UNDER THE CURVE

logit_auc <- performance(logit_scores, "auc")

as.numeric(logit_auc@y.values)  ##AUC Value


##############################################################################
#################



# Getting Lift Charts in R

# For getting Lift Chart in R, use measure="lift", x.measure="rpp" in the performance function.

# Get data for ROC curve and create lift chart values

logit_lift <- performance(logit_scores, measure="lift", x.measure="rpp")

plot(logit_lift,
```

```
     main="Lift Chart",
     xlab="% Populations (Percentile)",
     ylab="Lift",
     col="darkblue", lwd = 3)
abline(1,0,col="red",  lwd = 3)
grid(col="aquamarine")



# for part e
################################################


logit_scores <- prediction(predictions=mydf_e$val, labels=mydf_e$Churn)


#PLOT ROC CURVE
logit_perf <- performance(logit_scores, "tpr", "fpr")
```

```
plot(logit_perf,
     main="ROC Curves",
     xlab="1 - Specificity: False Positive Rate",
     ylab="Sensitivity: True Positive Rate",
     col="darkblue",  lwd = 3)
abline(0,1, lty = 300, col = "green",  lwd = 3)
grid(col="aquamarine")


# AREA UNDER THE CURVE
logit_auc <- performance(logit_scores, "auc")
as.numeric(logit_auc@y.values)  ##AUC Value
```

```
##################################################################################
#################



# Getting Lift Charts in R

# For getting Lift Chart in R, use measure="lift", x.measure="rpp" in the performance function.

# Get data for ROC curve and create lift chart values

logit_lift <- performance(logit_scores, measure="lift", x.measure="rpp")

plot(logit_lift,

     main="Lift Chart",

     xlab="% Populations (Percentile)",

     ylab="Lift",

     col="darkblue", lwd = 3)

abline(1,0,col="red",  lwd = 3)

grid(col="aquamarine")
```

```
# problem h



temp = matrix(data=NA, nrow = 9)

for (i in seq(0.1,0.9,0.1)) {


  mydf_e <-cbind(validation,val)

  mydf_e$response <- as.factor(ifelse(mydf_e$val>i, 1, 0))

  cmat = confusionMatrix(mydf_e$response,mydf_e$Churn)

  temp[i*10] = cmat$overall[1]

}


temp
```