DA Project

Rathnapriya Gopalakrishnan

4/21/2022

# Data Storage

## Importing Dataset

df = read.csv("C:/Users/Priya/Desktop/heart\_2020\_cleaned.csv")  
head(df)

## HeartDisease BMI Smoking AlcoholDrinking Stroke PhysicalHealth MentalHealth  
## 1 No 16.60 Yes No No 3 30  
## 2 No 20.34 No No Yes 0 0  
## 3 No 26.58 Yes No No 20 30  
## 4 No 24.21 No No No 0 0  
## 5 No 23.71 No No No 28 0  
## 6 Yes 28.87 Yes No No 6 0  
## DiffWalking Sex AgeCategory Race Diabetic PhysicalActivity GenHealth  
## 1 No Female 55-59 White Yes Yes Very good  
## 2 No Female 80 or older White No Yes Very good  
## 3 No Male 65-69 White Yes Yes Fair  
## 4 No Female 75-79 White No No Good  
## 5 Yes Female 40-44 White No Yes Very good  
## 6 Yes Female 75-79 Black No No Fair  
## SleepTime Asthma KidneyDisease SkinCancer  
## 1 5 Yes No Yes  
## 2 7 No No No  
## 3 8 Yes No No  
## 4 6 No No Yes  
## 5 8 No No No  
## 6 12 No No No

unalteredDf = df

# Data Cleaning

## Checking for missing values

sum(is.na(df))

## [1] 0

No missing values to handle

# Data Exploration

str(df)

## 'data.frame': 319795 obs. of 18 variables:  
## $ HeartDisease : chr "No" "No" "No" "No" ...  
## $ BMI : num 16.6 20.3 26.6 24.2 23.7 ...  
## $ Smoking : chr "Yes" "No" "Yes" "No" ...  
## $ AlcoholDrinking : chr "No" "No" "No" "No" ...  
## $ Stroke : chr "No" "Yes" "No" "No" ...  
## $ PhysicalHealth : num 3 0 20 0 28 6 15 5 0 0 ...  
## $ MentalHealth : num 30 0 30 0 0 0 0 0 0 0 ...  
## $ DiffWalking : chr "No" "No" "No" "No" ...  
## $ Sex : chr "Female" "Female" "Male" "Female" ...  
## $ AgeCategory : chr "55-59" "80 or older" "65-69" "75-79" ...  
## $ Race : chr "White" "White" "White" "White" ...  
## $ Diabetic : chr "Yes" "No" "Yes" "No" ...  
## $ PhysicalActivity: chr "Yes" "Yes" "Yes" "No" ...  
## $ GenHealth : chr "Very good" "Very good" "Fair" "Good" ...  
## $ SleepTime : num 5 7 8 6 8 12 4 9 5 10 ...  
## $ Asthma : chr "Yes" "No" "Yes" "No" ...  
## $ KidneyDisease : chr "No" "No" "No" "No" ...  
## $ SkinCancer : chr "Yes" "No" "No" "Yes" ...

summary(df)

## HeartDisease BMI Smoking AlcoholDrinking   
## Length:319795 Min. :12.02 Length:319795 Length:319795   
## Class :character 1st Qu.:24.03 Class :character Class :character   
## Mode :character Median :27.34 Mode :character Mode :character   
## Mean :28.33   
## 3rd Qu.:31.42   
## Max. :94.85   
## Stroke PhysicalHealth MentalHealth DiffWalking   
## Length:319795 Min. : 0.000 Min. : 0.000 Length:319795   
## Class :character 1st Qu.: 0.000 1st Qu.: 0.000 Class :character   
## Mode :character Median : 0.000 Median : 0.000 Mode :character   
## Mean : 3.372 Mean : 3.898   
## 3rd Qu.: 2.000 3rd Qu.: 3.000   
## Max. :30.000 Max. :30.000   
## Sex AgeCategory Race Diabetic   
## Length:319795 Length:319795 Length:319795 Length:319795   
## Class :character Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character Mode :character   
##   
##   
##   
## PhysicalActivity GenHealth SleepTime Asthma   
## Length:319795 Length:319795 Min. : 1.000 Length:319795   
## Class :character Class :character 1st Qu.: 6.000 Class :character   
## Mode :character Mode :character Median : 7.000 Mode :character   
## Mean : 7.097   
## 3rd Qu.: 8.000   
## Max. :24.000   
## KidneyDisease SkinCancer   
## Length:319795 Length:319795   
## Class :character Class :character   
## Mode :character Mode :character   
##   
##   
##

# Feature Engineering

## Extracting the continuous and categorical features

numeric\_df <- df[, sapply(df, class) == "numeric"]  
head(numeric\_df)

## BMI PhysicalHealth MentalHealth SleepTime  
## 1 16.60 3 30 5  
## 2 20.34 0 0 7  
## 3 26.58 20 30 8  
## 4 24.21 0 0 6  
## 5 23.71 28 0 8  
## 6 28.87 6 0 12

cat\_df <- df[, sapply(df, class) == "character"]  
head(cat\_df)

## HeartDisease Smoking AlcoholDrinking Stroke DiffWalking Sex AgeCategory  
## 1 No Yes No No No Female 55-59  
## 2 No No No Yes No Female 80 or older  
## 3 No Yes No No No Male 65-69  
## 4 No No No No No Female 75-79  
## 5 No No No No Yes Female 40-44  
## 6 Yes Yes No No Yes Female 75-79  
## Race Diabetic PhysicalActivity GenHealth Asthma KidneyDisease SkinCancer  
## 1 White Yes Yes Very good Yes No Yes  
## 2 White No Yes Very good No No No  
## 3 White Yes Yes Fair Yes No No  
## 4 White No No Good No No Yes  
## 5 White No Yes Very good No No No  
## 6 Black No No Fair No No No

df$Smoking = as.factor(df$Smoking)  
df$AlcoholDrinking = as.factor(df$Smoking)  
df$Stroke = as.factor(df$Stroke)  
df$DiffWalking = as.factor(df$DiffWalking)  
df$Sex = as.factor(df$Sex)  
df$AgeCategory = as.factor(df$AgeCategory)  
df$Race = as.factor(df$Race)  
df$Diabetic = as.factor(df$Diabetic)  
df$PhysicalActivity = as.factor(df$PhysicalActivity)  
df$GenHealth = as.factor(df$GenHealth)  
df$Asthma = as.factor(df$Asthma)  
df$KidneyDisease = as.factor(df$KidneyDisease)  
df$SkinCancer = as.factor(df$SkinCancer)  
#df$HD = ifelse (df$HeartDisease=="Yes",1,0)  
df$HeartDisease = factor(df$HeartDisease,levels = c("No","Yes"),labels = c(0,1))

head(df$HeartDisease)

## [1] 0 0 0 0 0 1  
## Levels: 0 1

str(df)

## 'data.frame': 319795 obs. of 18 variables:  
## $ HeartDisease : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 1 1 1 1 ...  
## $ BMI : num 16.6 20.3 26.6 24.2 23.7 ...  
## $ Smoking : Factor w/ 2 levels "No","Yes": 2 1 2 1 1 2 1 2 1 1 ...  
## $ AlcoholDrinking : Factor w/ 2 levels "No","Yes": 2 1 2 1 1 2 1 2 1 1 ...  
## $ Stroke : Factor w/ 2 levels "No","Yes": 1 2 1 1 1 1 1 1 1 1 ...  
## $ PhysicalHealth : num 3 0 20 0 28 6 15 5 0 0 ...  
## $ MentalHealth : num 30 0 30 0 0 0 0 0 0 0 ...  
## $ DiffWalking : Factor w/ 2 levels "No","Yes": 1 1 1 1 2 2 1 2 1 2 ...  
## $ Sex : Factor w/ 2 levels "Female","Male": 1 1 2 1 1 1 1 1 1 2 ...  
## $ AgeCategory : Factor w/ 13 levels "18-24","25-29",..: 8 13 10 12 5 12 11 13 13 10 ...  
## $ Race : Factor w/ 6 levels "American Indian/Alaskan Native",..: 6 6 6 6 6 3 6 6 6 6 ...  
## $ Diabetic : Factor w/ 4 levels "No","No, borderline diabetes",..: 3 1 3 1 1 1 1 3 2 1 ...  
## $ PhysicalActivity: Factor w/ 2 levels "No","Yes": 2 2 2 1 2 1 2 1 1 2 ...  
## $ GenHealth : Factor w/ 5 levels "Excellent","Fair",..: 5 5 2 3 5 2 2 3 2 3 ...  
## $ SleepTime : num 5 7 8 6 8 12 4 9 5 10 ...  
## $ Asthma : Factor w/ 2 levels "No","Yes": 2 1 2 1 1 1 2 2 1 1 ...  
## $ KidneyDisease : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 2 1 ...  
## $ SkinCancer : Factor w/ 2 levels "No","Yes": 2 1 1 2 1 1 2 1 1 1 ...

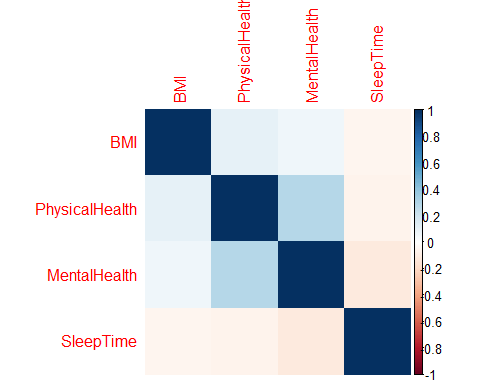
cor(numeric\_df)

## BMI PhysicalHealth MentalHealth SleepTime  
## BMI 1.00000000 0.10978754 0.06413057 -0.05182225  
## PhysicalHealth 0.10978754 1.00000000 0.28798667 -0.06138663  
## MentalHealth 0.06413057 0.28798667 1.00000000 -0.11971679  
## SleepTime -0.05182225 -0.06138663 -0.11971679 1.00000000

library(corrplot)

## corrplot 0.92 loaded

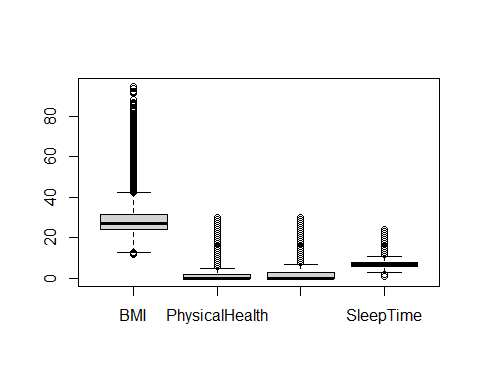
corrplot(cor(numeric\_df),method='color')



# Data Visualization

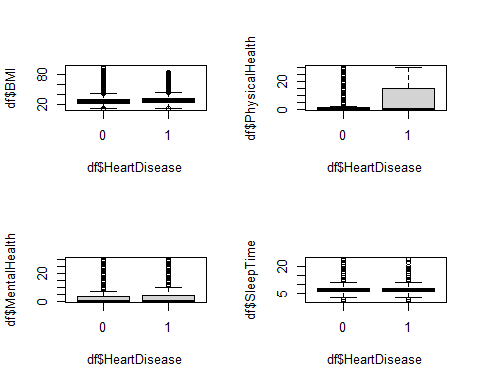
## For continuous variables

boxplot(numeric\_df)



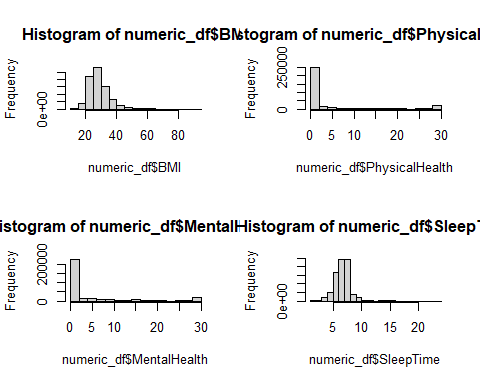
## How each continuous variable differs for different values of Heart Disease

par(mfrow=c(2,2))  
boxplot(df$BMI~ df$HeartDisease)  
boxplot(df$PhysicalHealth~ df$HeartDisease)  
boxplot(df$MentalHealth~ df$HeartDisease)  
boxplot(df$SleepTime~ df$HeartDisease)



## Histogram of continuous variables

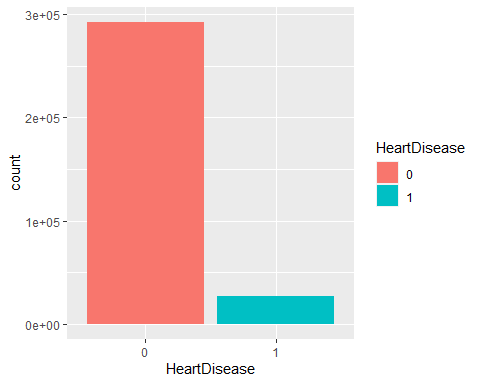
par(mfrow=c(2,2))  
hist(numeric\_df$BMI)  
hist(numeric\_df$PhysicalHealth)  
hist(numeric\_df$MentalHealth)  
hist(numeric\_df$SleepTime)



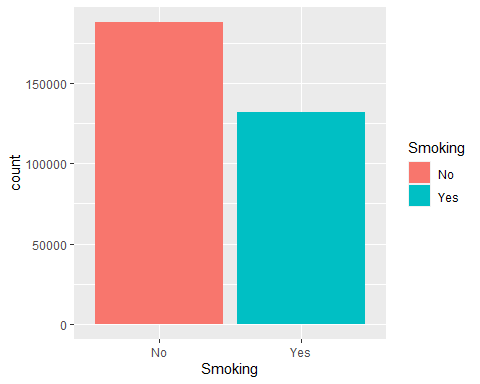
## For categorical variables

## Frequency of classes for each categorical variable

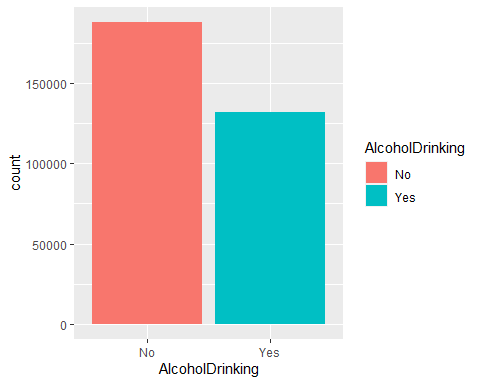
library(ggplot2)  
ggplot(df) + geom\_bar(aes(x = HeartDisease, fill=HeartDisease))



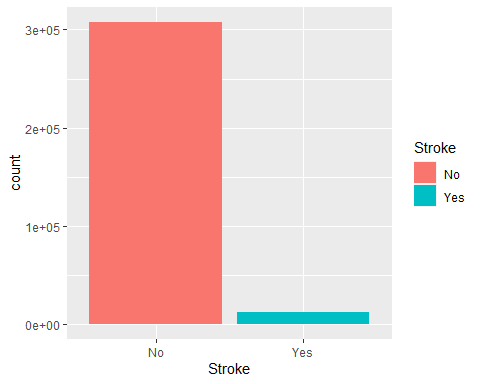
ggplot(df) + geom\_bar(aes(x = Smoking, fill=Smoking))



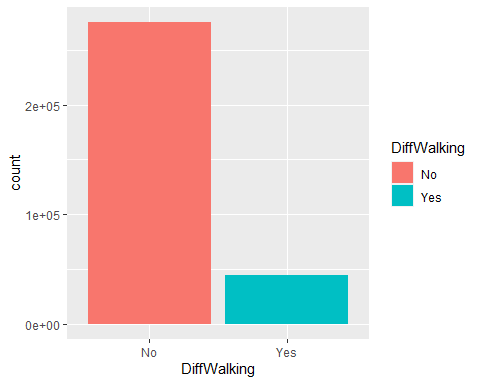
ggplot(df) + geom\_bar(aes(x = AlcoholDrinking, fill=AlcoholDrinking))



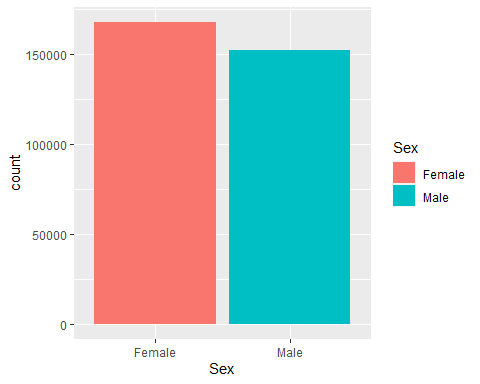
ggplot(df) + geom\_bar(aes(x = Stroke, fill=Stroke))



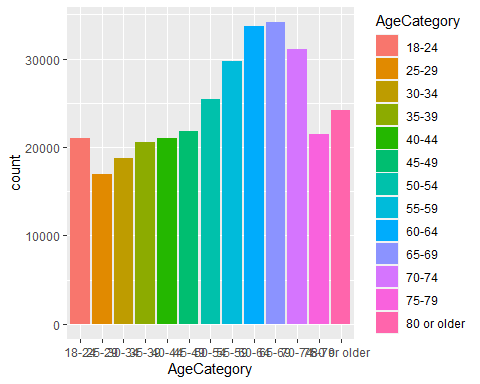
ggplot(df) + geom\_bar(aes(x = DiffWalking, fill=DiffWalking))



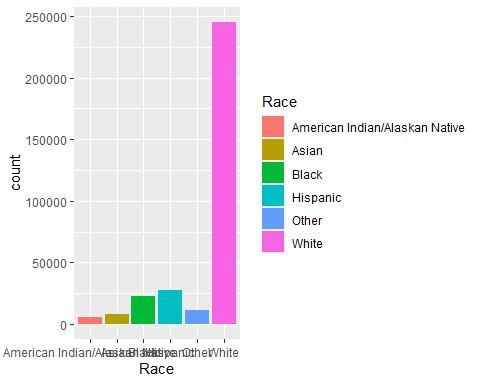
ggplot(df) + geom\_bar(aes(x = Sex, fill=Sex))



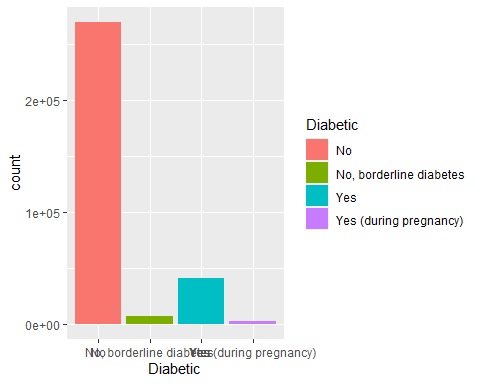
ggplot(df) + geom\_bar(aes(x = AgeCategory, fill=AgeCategory))



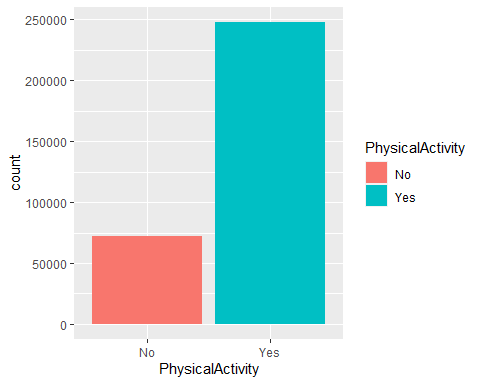
ggplot(df) + geom\_bar(aes(x = Race, fill=Race))



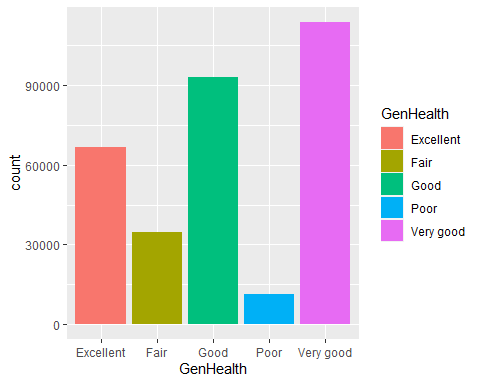
ggplot(df) + geom\_bar(aes(x = Diabetic, fill=Diabetic))



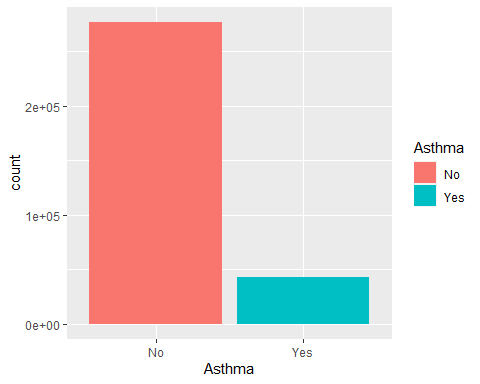
ggplot(df) + geom\_bar(aes(x = PhysicalActivity, fill=PhysicalActivity))



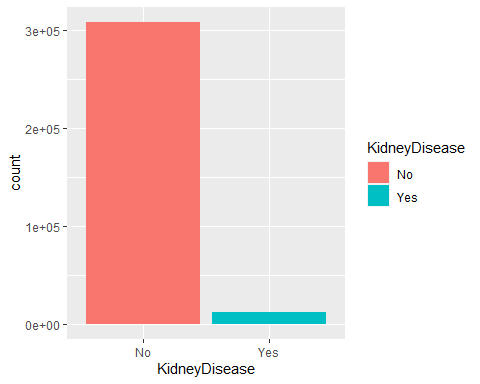
ggplot(df) + geom\_bar(aes(x = GenHealth, fill=GenHealth))



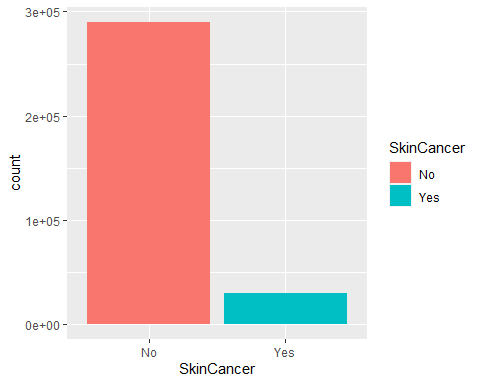
ggplot(df) + geom\_bar(aes(x = Asthma, fill=Asthma))



ggplot(df) + geom\_bar(aes(x = KidneyDisease, fill=KidneyDisease))

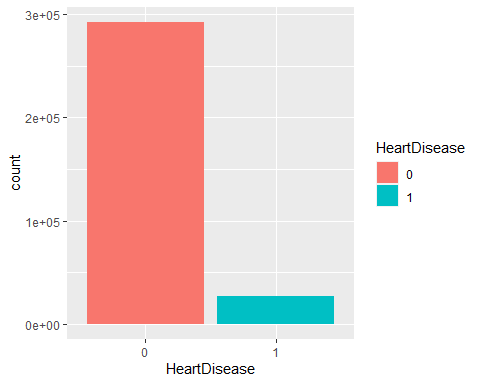


ggplot(df) + geom\_bar(aes(x = SkinCancer, fill=SkinCancer))

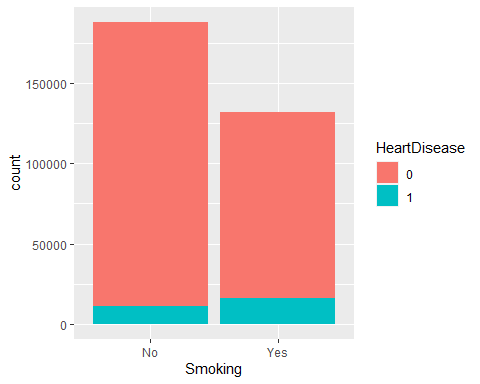


## Frequency of Heart Disease in each class of categorical variables

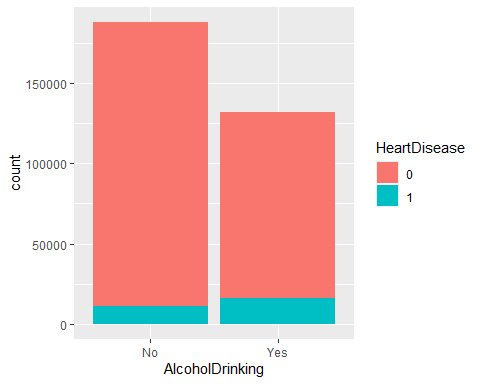
library(ggplot2)  
ggplot(df) + geom\_bar(aes(x = HeartDisease, fill=HeartDisease))



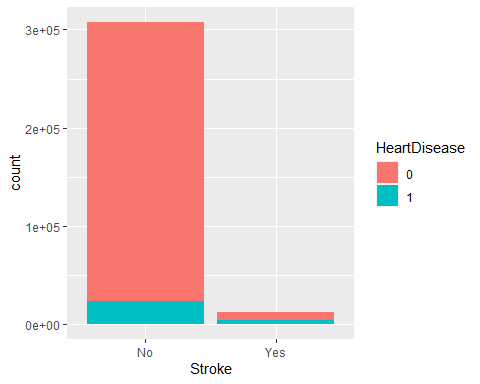
ggplot(df) + geom\_bar(aes(x = Smoking, fill=HeartDisease))



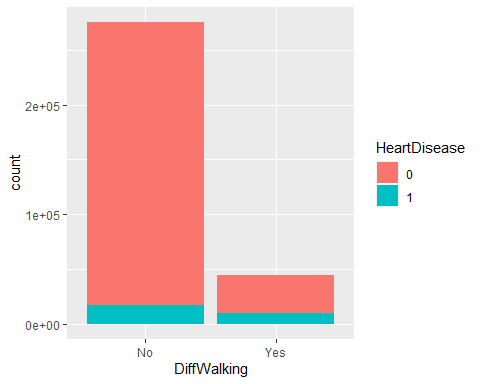
ggplot(df) + geom\_bar(aes(x = AlcoholDrinking, fill=HeartDisease))



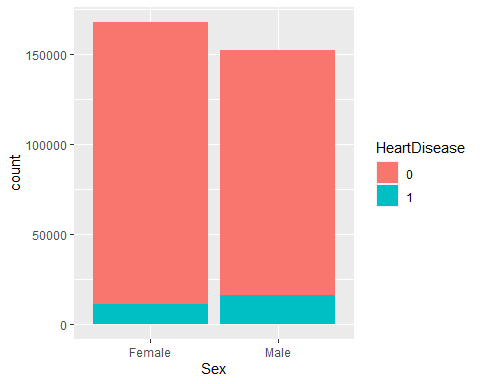
ggplot(df) + geom\_bar(aes(x = Stroke, fill=HeartDisease))



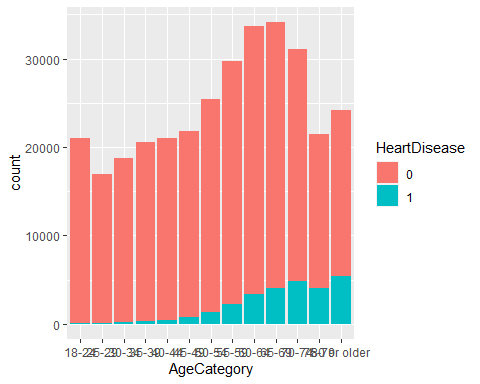
ggplot(df) + geom\_bar(aes(x = DiffWalking, fill=HeartDisease))



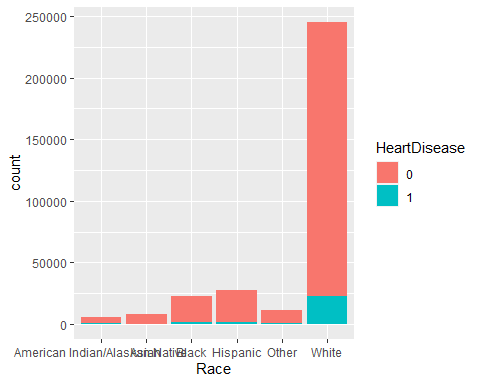
ggplot(df) + geom\_bar(aes(x = Sex, fill=HeartDisease))



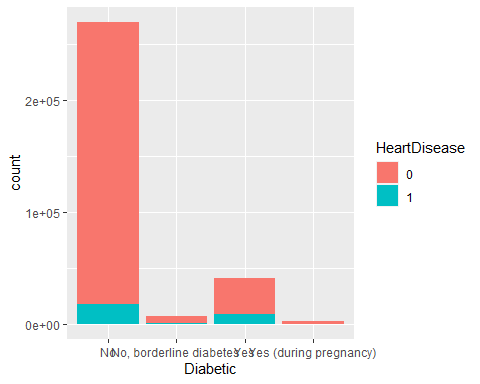
ggplot(df) + geom\_bar(aes(x = AgeCategory, fill=HeartDisease))



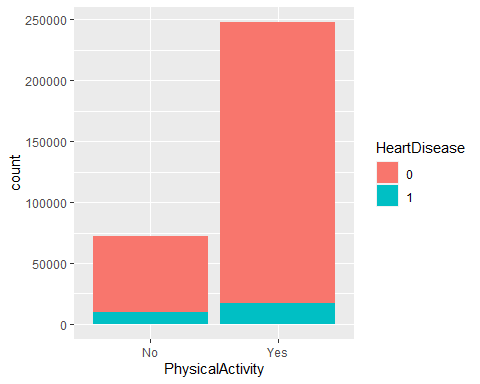
ggplot(df) + geom\_bar(aes(x = Race, fill=HeartDisease))



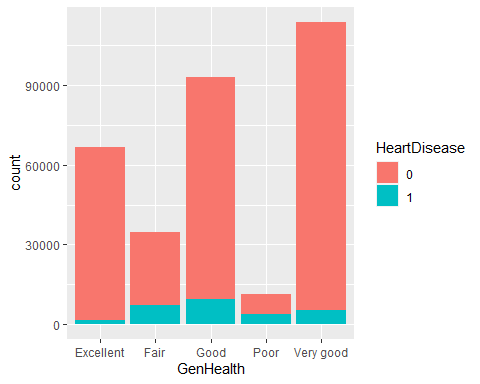
ggplot(df) + geom\_bar(aes(x = Diabetic, fill=HeartDisease))



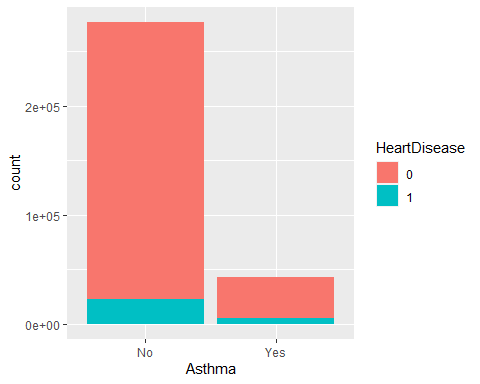
ggplot(df) + geom\_bar(aes(x = PhysicalActivity, fill=HeartDisease))



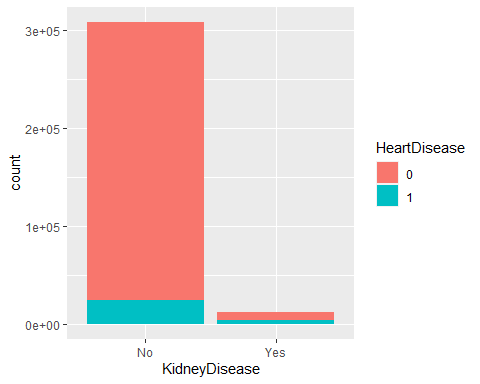
ggplot(df) + geom\_bar(aes(x = GenHealth, fill=HeartDisease))



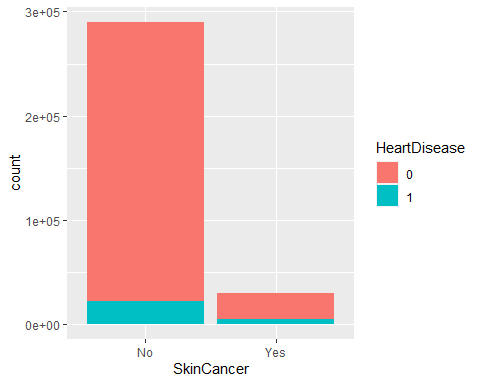
ggplot(df) + geom\_bar(aes(x = Asthma, fill=HeartDisease))



ggplot(df) + geom\_bar(aes(x = KidneyDisease, fill=HeartDisease))



ggplot(df) + geom\_bar(aes(x = SkinCancer, fill=HeartDisease))



# Sampling

## We have an imbalance in the data

No of records in each class of the target variable before sampling

table(df$HeartDisease)

##   
## 0 1   
## 292422 27373

## Performing Undersampling

library(ROSE)

## Loaded ROSE 0.0-4

library(TeachingDemos)  
set.seed(char2seed("Priya"))  
balanced\_df\_under = ovun.sample(HeartDisease~.,   
 data=df,  
 method = "under")$data

No of records in each class of the target variable after undersampling

table(balanced\_df\_under$HeartDisease)

##   
## 0 1   
## 27386 27373

## Performing Oversampling

library(TeachingDemos)  
set.seed(char2seed("Priya"))  
balanced\_df\_over = ovun.sample(HeartDisease~.,   
 data=df,  
 method = "over")$data

No of records in each class of the target variable after oversampling

table(balanced\_df\_over$HeartDisease)

##   
## 0 1   
## 292422 292248

MLdf=balanced\_df\_under  
  
  
# MLdf0 = subset(MLdf, HeartDisease==0)  
# MLdf1 = subset(MLdf, HeartDisease==1)  
# MLdf0 = MLdf0[1:500,]  
# MLdf1 = MLdf1[1:500,]  
#   
# MLdf = rbind(MLdf0, MLdf1)  
  
  
# MLdf=balanced\_df\_under[1:1000,]  
  
nrow(MLdf)

## [1] 54759

ncol(MLdf)

## [1] 18

# Machine Learning

## Forming train and test sets

library(TeachingDemos)  
set.seed(char2seed("Priya"))  
train = sample(nrow(MLdf), size = 0.8\*nrow(MLdf))  
head(MLdf)

## HeartDisease BMI Smoking AlcoholDrinking Stroke PhysicalHealth MentalHealth  
## 1 0 29.53 No No No 0 2  
## 2 0 32.45 No No No 2 5  
## 3 0 22.60 No No No 0 0  
## 4 0 25.79 Yes Yes No 0 7  
## 5 0 28.89 Yes Yes No 0 0  
## 6 0 25.40 No No No 0 0  
## DiffWalking Sex AgeCategory Race Diabetic PhysicalActivity GenHealth  
## 1 No Male 50-54 White No Yes Very good  
## 2 No Female 50-54 White No Yes Very good  
## 3 No Female 55-59 Black No Yes Excellent  
## 4 No Female 18-24 White No Yes Excellent  
## 5 No Male 35-39 Other No Yes Excellent  
## 6 No Female 25-29 Asian No Yes Excellent  
## SleepTime Asthma KidneyDisease SkinCancer  
## 1 6 No No No  
## 2 7 No No No  
## 3 7 No No No  
## 4 6 Yes No No  
## 5 8 Yes No No  
## 6 7 Yes No No

Removing HD column

#MLdf = subset(MLdf, select = -c(HD))

trainDf = MLdf[train,]  
testDf = MLdf[-train,]

Extracting Train and Test sets

trainMldf = MLdf[train,]  
testMldf = MLdf[-train,]  
Xtrain = trainDf[,-c(1)]  
Xtest = testDf[,-c(1)]  
ytrain = trainDf$HeartDisease  
ytest = testDf$HeartDisease

## Logistic Regression

head(MLdf)

## HeartDisease BMI Smoking AlcoholDrinking Stroke PhysicalHealth MentalHealth  
## 1 0 29.53 No No No 0 2  
## 2 0 32.45 No No No 2 5  
## 3 0 22.60 No No No 0 0  
## 4 0 25.79 Yes Yes No 0 7  
## 5 0 28.89 Yes Yes No 0 0  
## 6 0 25.40 No No No 0 0  
## DiffWalking Sex AgeCategory Race Diabetic PhysicalActivity GenHealth  
## 1 No Male 50-54 White No Yes Very good  
## 2 No Female 50-54 White No Yes Very good  
## 3 No Female 55-59 Black No Yes Excellent  
## 4 No Female 18-24 White No Yes Excellent  
## 5 No Male 35-39 Other No Yes Excellent  
## 6 No Female 25-29 Asian No Yes Excellent  
## SleepTime Asthma KidneyDisease SkinCancer  
## 1 6 No No No  
## 2 7 No No No  
## 3 7 No No No  
## 4 6 Yes No No  
## 5 8 Yes No No  
## 6 7 Yes No No

str(MLdf)

## 'data.frame': 54759 obs. of 18 variables:  
## $ HeartDisease : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...  
## $ BMI : num 29.5 32.5 22.6 25.8 28.9 ...  
## $ Smoking : Factor w/ 2 levels "No","Yes": 1 1 1 2 2 1 2 1 1 2 ...  
## $ AlcoholDrinking : Factor w/ 2 levels "No","Yes": 1 1 1 2 2 1 2 1 1 2 ...  
## $ Stroke : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...  
## $ PhysicalHealth : num 0 2 0 0 0 0 0 1 1 0 ...  
## $ MentalHealth : num 2 5 0 7 0 0 0 15 0 0 ...  
## $ DiffWalking : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...  
## $ Sex : Factor w/ 2 levels "Female","Male": 2 1 1 1 2 1 1 1 2 1 ...  
## $ AgeCategory : Factor w/ 13 levels "18-24","25-29",..: 7 7 8 1 4 2 9 4 1 7 ...  
## $ Race : Factor w/ 6 levels "American Indian/Alaskan Native",..: 6 6 3 6 5 2 5 6 6 6 ...  
## $ Diabetic : Factor w/ 4 levels "No","No, borderline diabetes",..: 1 1 1 1 1 1 2 1 1 1 ...  
## $ PhysicalActivity: Factor w/ 2 levels "No","Yes": 2 2 2 2 2 2 2 2 2 2 ...  
## $ GenHealth : Factor w/ 5 levels "Excellent","Fair",..: 5 5 1 1 1 1 3 5 1 1 ...  
## $ SleepTime : num 6 7 7 6 8 7 6 7 8 8 ...  
## $ Asthma : Factor w/ 2 levels "No","Yes": 1 1 1 2 2 2 1 1 1 1 ...  
## $ KidneyDisease : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...  
## $ SkinCancer : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...

library(gbm)

## Loaded gbm 2.1.8

log\_mod = glm(HeartDisease~., data=trainMldf, family="binomial")  
summary(log\_mod)

##   
## Call:  
## glm(formula = HeartDisease ~ ., family = "binomial", data = trainMldf)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -3.0010 -0.7765 0.1448 0.8046 2.9514   
##   
## Coefficients: (1 not defined because of singularities)  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -4.133051 0.167785 -24.633 < 2e-16 \*\*\*  
## BMI 0.009287 0.002013 4.614 3.95e-06 \*\*\*  
## SmokingYes 0.353214 0.024376 14.490 < 2e-16 \*\*\*  
## AlcoholDrinkingYes NA NA NA NA   
## StrokeYes 1.198536 0.050800 23.593 < 2e-16 \*\*\*  
## PhysicalHealth 0.003877 0.001623 2.389 0.01690 \*   
## MentalHealth 0.006754 0.001590 4.247 2.17e-05 \*\*\*  
## DiffWalkingYes 0.220096 0.033625 6.546 5.92e-11 \*\*\*  
## SexMale 0.764987 0.024885 30.741 < 2e-16 \*\*\*  
## AgeCategory25-29 0.284680 0.152279 1.869 0.06156 .   
## AgeCategory30-34 0.634123 0.138224 4.588 4.48e-06 \*\*\*  
## AgeCategory35-39 0.648522 0.134765 4.812 1.49e-06 \*\*\*  
## AgeCategory40-44 1.066468 0.127706 8.351 < 2e-16 \*\*\*  
## AgeCategory45-49 1.359127 0.123984 10.962 < 2e-16 \*\*\*  
## AgeCategory50-54 1.805906 0.119728 15.083 < 2e-16 \*\*\*  
## AgeCategory55-59 2.071164 0.117803 17.582 < 2e-16 \*\*\*  
## AgeCategory60-64 2.397405 0.116635 20.555 < 2e-16 \*\*\*  
## AgeCategory65-69 2.650511 0.116289 22.792 < 2e-16 \*\*\*  
## AgeCategory70-74 2.963789 0.116614 25.415 < 2e-16 \*\*\*  
## AgeCategory75-79 3.145130 0.118228 26.602 < 2e-16 \*\*\*  
## AgeCategory80 or older 3.517134 0.118065 29.790 < 2e-16 \*\*\*  
## RaceAsian -0.557794 0.134501 -4.147 3.37e-05 \*\*\*  
## RaceBlack -0.301642 0.101097 -2.984 0.00285 \*\*   
## RaceHispanic -0.235162 0.101587 -2.315 0.02062 \*   
## RaceOther -0.121942 0.111674 -1.092 0.27486   
## RaceWhite -0.111475 0.091021 -1.225 0.22068   
## DiabeticNo, borderline diabetes 0.097056 0.072313 1.342 0.17954   
## DiabeticYes 0.553927 0.031396 17.643 < 2e-16 \*\*\*  
## DiabeticYes (during pregnancy) 0.178852 0.163547 1.094 0.27414   
## PhysicalActivityYes 0.005712 0.028480 0.201 0.84104   
## GenHealthFair 1.559150 0.050902 30.630 < 2e-16 \*\*\*  
## GenHealthGood 1.083397 0.042356 25.578 < 2e-16 \*\*\*  
## GenHealthPoor 1.919985 0.072772 26.383 < 2e-16 \*\*\*  
## GenHealthVery good 0.487483 0.042434 11.488 < 2e-16 \*\*\*  
## SleepTime -0.034666 0.007731 -4.484 7.32e-06 \*\*\*  
## AsthmaYes 0.292061 0.034464 8.474 < 2e-16 \*\*\*  
## KidneyDiseaseYes 0.608387 0.052157 11.665 < 2e-16 \*\*\*  
## SkinCancerYes 0.163776 0.035925 4.559 5.14e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 60729 on 43806 degrees of freedom  
## Residual deviance: 43017 on 43770 degrees of freedom  
## AIC: 43091  
##   
## Number of Fisher Scoring iterations: 5

Function to extract evaluation metrics

met <- function(cm){  
 print(cm)  
 met\_list = list( "Acc" =cm$overall['Accuracy'], "Spec" = cm$byClass['Specificity'], "Sens" = cm$byClass['Sensitivity'])  
 return (met\_list)  
}

proba = predict(log\_mod, testMldf, type = "response")

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :  
## prediction from a rank-deficient fit may be misleading

pred = ifelse(proba>0.5, 1,0)

mean(pred!=testMldf$HeartDisease)

## [1] 0.2376735

library(caret)

## Loading required package: lattice

pred = factor(pred, levels=c(0,1), labels=c("X0","X1"))  
testMldf$HeartDisease = factor(testMldf$HeartDisease, levels=c(0,1), labels=c("X0","X1"))  
  
cm\_glm= confusionMatrix(pred,testMldf$HeartDisease, positive='X1' )  
eval\_glm = met(cm\_glm)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction X0 X1  
## X0 4133 1208  
## X1 1395 4216  
##   
## Accuracy : 0.7623   
## 95% CI : (0.7542, 0.7703)  
## No Information Rate : 0.5047   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.5248   
##   
## Mcnemar's Test P-Value : 0.0002667   
##   
## Sensitivity : 0.7773   
## Specificity : 0.7476   
## Pos Pred Value : 0.7514   
## Neg Pred Value : 0.7738   
## Prevalence : 0.4953   
## Detection Rate : 0.3850   
## Detection Prevalence : 0.5123   
## Balanced Accuracy : 0.7625   
##   
## 'Positive' Class : X1   
##

print(cm\_glm)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction X0 X1  
## X0 4133 1208  
## X1 1395 4216  
##   
## Accuracy : 0.7623   
## 95% CI : (0.7542, 0.7703)  
## No Information Rate : 0.5047   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.5248   
##   
## Mcnemar's Test P-Value : 0.0002667   
##   
## Sensitivity : 0.7773   
## Specificity : 0.7476   
## Pos Pred Value : 0.7514   
## Neg Pred Value : 0.7738   
## Prevalence : 0.4953   
## Detection Rate : 0.3850   
## Detection Prevalence : 0.5123   
## Balanced Accuracy : 0.7625   
##   
## 'Positive' Class : X1   
##

library(pROC)

## Type 'citation("pROC")' for a citation.

##   
## Attaching package: 'pROC'

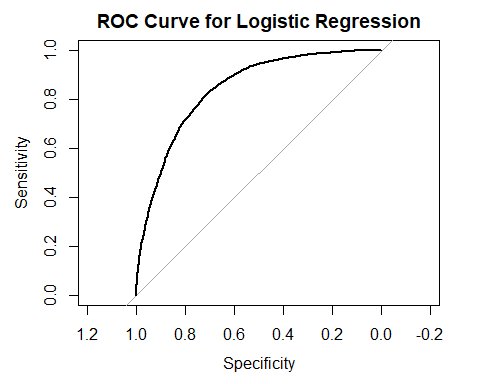
## The following objects are masked from 'package:stats':  
##   
## cov, smooth, var

roc\_glm = roc(testMldf$HeartDisease , proba)

## Setting levels: control = X0, case = X1

## Setting direction: controls < cases

plot(roc\_glm, main = "ROC Curve for Logistic Regression")



## Random Forest

Converting train and test categorical variables to factors

levels(trainMldf$HeartDisease) <- make.names(levels(factor(trainMldf$HeartDisease)))  
  
levels(trainMldf$Smoking) <- make.names(levels(factor(trainMldf$Smoking)))  
  
levels(trainMldf$AlcoholDrinking) <- make.names(levels(factor(trainMldf$AlcoholDrinking)))  
  
levels(trainMldf$Stroke) <- make.names(levels(factor(trainMldf$Stroke)))  
  
levels(trainMldf$DiffWalking) <- make.names(levels(factor(trainMldf$DiffWalking)))  
  
levels(trainMldf$Sex) <- make.names(levels(factor(trainMldf$Sex)))  
  
levels(trainMldf$AgeCategory) <- make.names(levels(factor(trainMldf$AgeCategory)))  
  
levels(trainMldf$Race) <- make.names(levels(factor(trainMldf$Race)))  
  
levels(trainMldf$Diabetic) <- make.names(levels(factor(trainMldf$Diabetic)))  
  
levels(trainMldf$PhysicalActivity) <- make.names(levels(factor(trainMldf$PhysicalActivity)))  
  
levels(trainMldf$GenHealth) <- make.names(levels(factor(trainMldf$GenHealth)))  
  
levels(trainMldf$Asthma) <- make.names(levels(factor(trainMldf$Asthma)))  
  
levels(trainMldf$KidneyDisease) <- make.names(levels(factor(trainMldf$KidneyDisease)))  
  
levels(trainMldf$SkinCancer) <- make.names(levels(factor(trainMldf$SkinCancer)))

levels(testMldf$HeartDisease) <- make.names(levels(factor(testMldf$HeartDisease)))  
  
levels(testMldf$Smoking) <- make.names(levels(factor(testMldf$Smoking)))  
  
levels(testMldf$AlcoholDrinking) <- make.names(levels(factor(testMldf$AlcoholDrinking)))  
  
levels(testMldf$Stroke) <- make.names(levels(factor(testMldf$Stroke)))  
  
levels(testMldf$DiffWalking) <- make.names(levels(factor(testMldf$DiffWalking)))  
  
levels(testMldf$Sex) <- make.names(levels(factor(testMldf$Sex)))  
  
levels(testMldf$AgeCategory) <- make.names(levels(factor(testMldf$AgeCategory)))  
  
levels(testMldf$Race) <- make.names(levels(factor(testMldf$Race)))  
  
levels(testMldf$Diabetic) <- make.names(levels(factor(testMldf$Diabetic)))  
  
levels(testMldf$PhysicalActivity) <- make.names(levels(factor(testMldf$PhysicalActivity)))  
  
levels(testMldf$GenHealth) <- make.names(levels(factor(testMldf$GenHealth)))  
  
levels(testMldf$Asthma) <- make.names(levels(factor(testMldf$Asthma)))  
  
levels(testMldf$KidneyDisease) <- make.names(levels(factor(testMldf$KidneyDisease)))  
  
levels(testMldf$SkinCancer) <- make.names(levels(factor(testMldf$SkinCancer)))

library(randomForest)

## randomForest 4.7-1

## Type rfNews() to see new features/changes/bug fixes.

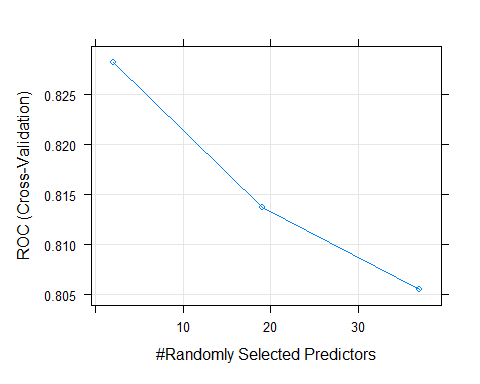
##   
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':  
##   
## margin

library(caret)  
  
library(TeachingDemos)  
set.seed(char2seed("Priya"))  
  
x = trainControl(method = "cv", number = 3, classProbs = TRUE, summaryFunction = twoClassSummary)  
  
rf\_mod\_1 <- train(HeartDisease~. , data = trainMldf, method = "rf", trControl = x, metric = "ROC")  
print(rf\_mod\_1)

## Random Forest   
##   
## 43807 samples  
## 17 predictor  
## 2 classes: 'X0', 'X1'   
##   
## No pre-processing  
## Resampling: Cross-Validated (3 fold)   
## Summary of sample sizes: 29205, 29205, 29204   
## Resampling results across tuning parameters:  
##   
## mtry ROC Sens Spec   
## 2 0.8281935 0.7284747 0.7840907  
## 19 0.8137359 0.7108153 0.7764818  
## 37 0.8055409 0.7026718 0.7676887  
##   
## ROC was used to select the optimal model using the largest value.  
## The final value used for the model was mtry = 2.

plot(rf\_mod\_1)



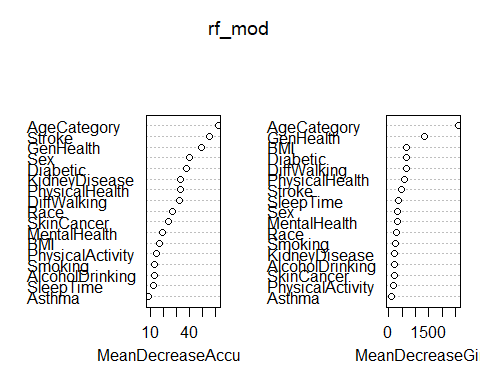
mtry = rf\_mod\_1$bestTune[1,]  
rf\_mod <- randomForest(HeartDisease ~ . , data= trainMldf, mtry=mtry, importance=TRUE)  
summary(rf\_mod)

## Length Class Mode   
## call 5 -none- call   
## type 1 -none- character  
## predicted 43807 factor numeric   
## err.rate 1500 -none- numeric   
## confusion 6 -none- numeric   
## votes 87614 matrix numeric   
## oob.times 43807 -none- numeric   
## classes 2 -none- character  
## importance 68 -none- numeric   
## importanceSD 51 -none- numeric   
## localImportance 0 -none- NULL   
## proximity 0 -none- NULL   
## ntree 1 -none- numeric   
## mtry 1 -none- numeric   
## forest 14 -none- list   
## y 43807 factor numeric   
## test 0 -none- NULL   
## inbag 0 -none- NULL   
## terms 3 terms call

varImp(rf\_mod)

## X0 X1  
## BMI 9.676276 9.676276  
## Smoking 11.663359 11.663359  
## AlcoholDrinking 11.041563 11.041563  
## Stroke 36.166450 36.166450  
## PhysicalHealth 17.622826 17.622826  
## MentalHealth 9.779616 9.779616  
## DiffWalking 19.241616 19.241616  
## Sex 32.156792 32.156792  
## AgeCategory 56.644695 56.644695  
## Race 15.060795 15.060795  
## Diabetic 25.494013 25.494013  
## PhysicalActivity 7.123029 7.123029  
## GenHealth 38.371869 38.371869  
## SleepTime 7.051907 7.051907  
## Asthma 5.198280 5.198280  
## KidneyDisease 12.734428 12.734428  
## SkinCancer 15.535930 15.535930

varImpPlot(rf\_mod)



# testMldf$HeartDisease = factor(testMldf$HeartDisease, levels = levels(trainMldf$HeartDisease))  
levels(testMldf$HeartDisease) = levels(trainMldf$HeartDisease)  
yPredRf\_proba\_old = predict(rf\_mod, newdata = testMldf, type='prob')  
  
yPredRf\_proba = yPredRf\_proba\_old[,1]  
yPredRf = rep(1,length(yPredRf\_proba))  
yPredRf[yPredRf\_proba > 0.5] = 0  
length(yPredRf\_proba)

## [1] 10952

# err\_table\_rf = table(testMldf$HeartDisease, yPredRf)  
# print(err\_table\_rf)  
# test\_err\_rf = err\_table\_rf[2,2]/(err\_table\_rf[1,2]+err\_table\_rf[2,2])  
yPredRf = factor(yPredRf, levels=c(0,1), labels=c("X0","X1"))  
mean(yPredRf!=testMldf$HeartDisease)

## [1] 0.2362126

cm\_rf = confusionMatrix(yPredRf,testMldf$HeartDisease,positive='X1' )  
eval\_rf = met(cm\_rf)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction X0 X1  
## X0 3962 1021  
## X1 1566 4403  
##   
## Accuracy : 0.7638   
## 95% CI : (0.7557, 0.7717)  
## No Information Rate : 0.5047   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.528   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.8118   
## Specificity : 0.7167   
## Pos Pred Value : 0.7376   
## Neg Pred Value : 0.7951   
## Prevalence : 0.4953   
## Detection Rate : 0.4020   
## Detection Prevalence : 0.5450   
## Balanced Accuracy : 0.7642   
##   
## 'Positive' Class : X1   
##

print(cm\_rf)

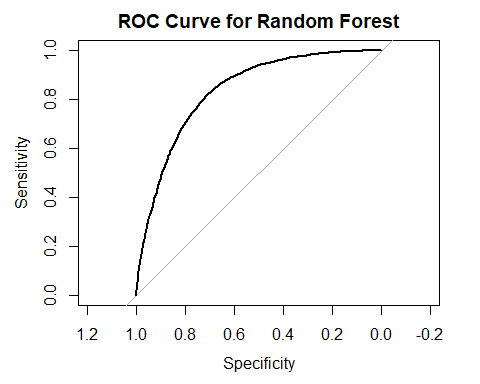
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction X0 X1  
## X0 3962 1021  
## X1 1566 4403  
##   
## Accuracy : 0.7638   
## 95% CI : (0.7557, 0.7717)  
## No Information Rate : 0.5047   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.528   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.8118   
## Specificity : 0.7167   
## Pos Pred Value : 0.7376   
## Neg Pred Value : 0.7951   
## Prevalence : 0.4953   
## Detection Rate : 0.4020   
## Detection Prevalence : 0.5450   
## Balanced Accuracy : 0.7642   
##   
## 'Positive' Class : X1   
##

library(pROC)  
roc\_rf = roc(testMldf$HeartDisease , yPredRf\_proba\_old[,2])

## Setting levels: control = X0, case = X1

## Setting direction: controls < cases

plot(roc\_rf, main = "ROC Curve for Random Forest")

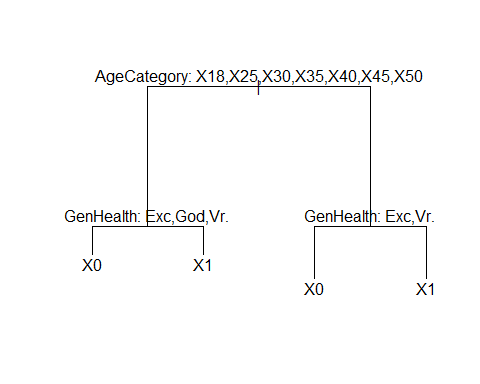


## Decision Tree

library(tree)  
r\_tree = tree(HeartDisease~., data =trainMldf)  
summary(r\_tree)

##   
## Classification tree:  
## tree(formula = HeartDisease ~ ., data = trainMldf)  
## Variables actually used in tree construction:  
## [1] "AgeCategory" "GenHealth"   
## Number of terminal nodes: 4   
## Residual mean deviance: 1.124 = 49230 / 43800   
## Misclassification error rate: 0.2828 = 12387 / 43807

plot(r\_tree)  
text(r\_tree, pretty=3)



yPredTree\_proba\_old = predict(r\_tree, newdata = testMldf)  
yPredTree\_proba = yPredTree\_proba\_old  
yPredTree\_proba = yPredTree\_proba[,1]  
yPredTree = rep(1,length(yPredTree\_proba))  
yPredTree[yPredTree\_proba > 0.5] = 0  
# length(yPredTree\_proba)  
#   
# err\_table\_tree = table(testMldf$HeartDisease, yPredTree)  
# print(err\_table\_tree)  
# test\_err\_tree = err\_table\_tree[2,2]/(err\_table\_tree[1,2]+err\_table\_tree[2,2])  
# print(test\_err\_tree)  
mean(yPredTree!=testMldf$HeartDisease)

## [1] 1

yPredTree = factor(yPredTree, levels=c(0,1), labels=c("X0","X1"))

cm\_dt = confusionMatrix(yPredTree,testMldf$HeartDisease )  
eval\_dt = met(cm\_dt)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction X0 X1  
## X0 4085 1596  
## X1 1443 3828  
##   
## Accuracy : 0.7225   
## 95% CI : (0.714, 0.7309)  
## No Information Rate : 0.5047   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.4448   
##   
## Mcnemar's Test P-Value : 0.005829   
##   
## Sensitivity : 0.7390   
## Specificity : 0.7058   
## Pos Pred Value : 0.7191   
## Neg Pred Value : 0.7262   
## Prevalence : 0.5047   
## Detection Rate : 0.3730   
## Detection Prevalence : 0.5187   
## Balanced Accuracy : 0.7224   
##   
## 'Positive' Class : X0   
##

print(cm\_dt)

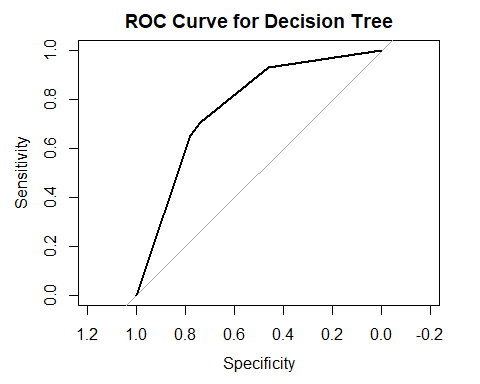
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction X0 X1  
## X0 4085 1596  
## X1 1443 3828  
##   
## Accuracy : 0.7225   
## 95% CI : (0.714, 0.7309)  
## No Information Rate : 0.5047   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.4448   
##   
## Mcnemar's Test P-Value : 0.005829   
##   
## Sensitivity : 0.7390   
## Specificity : 0.7058   
## Pos Pred Value : 0.7191   
## Neg Pred Value : 0.7262   
## Prevalence : 0.5047   
## Detection Rate : 0.3730   
## Detection Prevalence : 0.5187   
## Balanced Accuracy : 0.7224   
##   
## 'Positive' Class : X0   
##

library(pROC)  
roc\_tree = roc(testMldf$HeartDisease , yPredTree\_proba\_old[,2])

## Setting levels: control = X0, case = X1

## Setting direction: controls < cases

plot(roc\_tree, main = "ROC Curve for Decision Tree")



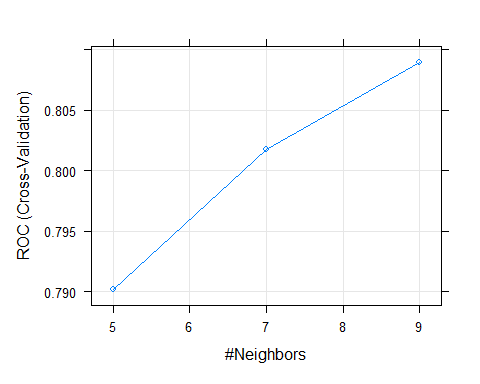
## KNN

library(caret)  
  
Xtrain = unalteredDf[,-c(1)]  
Xtest = unalteredDf[,-c(1)]  
#unalteredDf$HeartDisease = as.factor(unalteredDf$HeartDisease)  
ytrain = unalteredDf[,1]  
ytest = unalteredDf[,1]

library(caret)  
  
library(TeachingDemos)  
set.seed(char2seed("Priya"))  
x = trainControl(method = "cv", number = 3, classProbs = TRUE, summaryFunction = twoClassSummary)  
  
knn\_mod <- train(HeartDisease~. , data = trainMldf, method = "knn", preProcess = c("center","scale"), trControl = x, metric = "ROC")  
  
print(knn\_mod)

## k-Nearest Neighbors   
##   
## 43807 samples  
## 17 predictor  
## 2 classes: 'X0', 'X1'   
##   
## Pre-processing: centered (37), scaled (37)   
## Resampling: Cross-Validated (3 fold)   
## Summary of sample sizes: 29205, 29205, 29204   
## Resampling results across tuning parameters:  
##   
## k ROC Sens Spec   
## 5 0.7901926 0.7313112 0.7350678  
## 7 0.8017604 0.7340562 0.7413552  
## 9 0.8089575 0.7360234 0.7491005  
##   
## ROC was used to select the optimal model using the largest value.  
## The final value used for the model was k = 9.

plot(knn\_mod)



yPredKNN\_proba\_old <- predict(knn\_mod,testMldf, type = "prob")  
yPredKNN\_proba <- yPredKNN\_proba\_old  
  
yPredKNN\_proba = yPredKNN\_proba[,1]  
yPredKNN = rep(1,length(yPredKNN\_proba))  
yPredKNN[yPredKNN\_proba > 0.5] = 0  
# length(yPredKNN\_proba)  
#   
# err\_table\_KNN = table(testMldf$HeartDisease, yPredKNN)  
# print(err\_table\_KNN)  
# test\_err\_knn = err\_table\_KNN[2,2]/(err\_table\_KNN[1,2]+err\_table\_KNN[2,2])  
# print(test\_err\_knn)  
mean(yPredKNN!=testMldf$HeartDisease)

## [1] 1

yPredKNN = factor(yPredKNN, levels=c(0,1), labels=c("X0","X1"))

cm\_knn = confusionMatrix(yPredKNN,testMldf$HeartDisease, positive='X1' )  
eval\_knn = met(cm\_knn)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction X0 X1  
## X0 4070 1346  
## X1 1458 4078  
##   
## Accuracy : 0.744   
## 95% CI : (0.7357, 0.7521)  
## No Information Rate : 0.5047   
## P-Value [Acc > NIR] : < 2e-16   
##   
## Kappa : 0.488   
##   
## Mcnemar's Test P-Value : 0.03606   
##   
## Sensitivity : 0.7518   
## Specificity : 0.7363   
## Pos Pred Value : 0.7366   
## Neg Pred Value : 0.7515   
## Prevalence : 0.4953   
## Detection Rate : 0.3724   
## Detection Prevalence : 0.5055   
## Balanced Accuracy : 0.7440   
##   
## 'Positive' Class : X1   
##

print(cm\_knn)

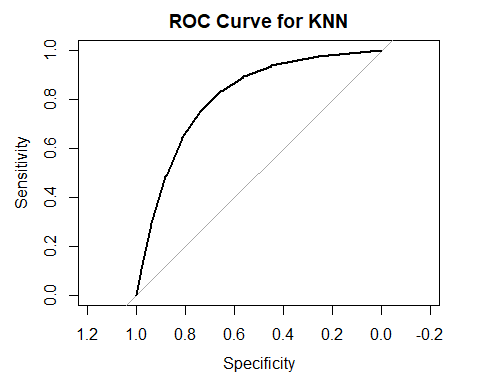
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction X0 X1  
## X0 4070 1346  
## X1 1458 4078  
##   
## Accuracy : 0.744   
## 95% CI : (0.7357, 0.7521)  
## No Information Rate : 0.5047   
## P-Value [Acc > NIR] : < 2e-16   
##   
## Kappa : 0.488   
##   
## Mcnemar's Test P-Value : 0.03606   
##   
## Sensitivity : 0.7518   
## Specificity : 0.7363   
## Pos Pred Value : 0.7366   
## Neg Pred Value : 0.7515   
## Prevalence : 0.4953   
## Detection Rate : 0.3724   
## Detection Prevalence : 0.5055   
## Balanced Accuracy : 0.7440   
##   
## 'Positive' Class : X1   
##

library(pROC)  
roc\_knn = roc(testMldf$HeartDisease , yPredKNN\_proba\_old[,2])

## Setting levels: control = X0, case = X1

## Setting direction: controls < cases

plot(roc\_knn, main = "ROC Curve for KNN")



## SVM

library(e1071)  
  
svm\_mod = svm(HeartDisease ~ .,data = trainMldf,kernel = 'linear', probability=TRUE)  
summary(svm\_mod)

##   
## Call:  
## svm(formula = HeartDisease ~ ., data = trainMldf, kernel = "linear",   
## probability = TRUE)  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: linear   
## cost: 1   
##   
## Number of Support Vectors: 24239  
##   
## ( 12118 12121 )  
##   
##   
## Number of Classes: 2   
##   
## Levels:   
## X0 X1

yPredSVM = predict(svm\_mod, newdata = testMldf, probability=TRUE)  
yPredSVm\_proba\_old = attr(yPredSVM, "probabilities")[,2]  
# err\_table\_SVM = table(testMldf$HeartDisease, yPredSVM)  
# print(err\_table\_SVM)  
#   
# test\_err\_svm = err\_table\_SVM[2,2]/(err\_table\_SVM[1,2]+err\_table\_SVM[2,2])  
# print(test\_err\_svm)  
mean(yPredSVM!=testMldf$HeartDisease)

## [1] 0.239317

cm\_svm = confusionMatrix(yPredSVM,testMldf$HeartDisease, positive = "X1" )  
eval\_svm = met(cm\_svm)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction X0 X1  
## X0 4145 1238  
## X1 1383 4186  
##   
## Accuracy : 0.7607   
## 95% CI : (0.7526, 0.7686)  
## No Information Rate : 0.5047   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.5214   
##   
## Mcnemar's Test P-Value : 0.004912   
##   
## Sensitivity : 0.7718   
## Specificity : 0.7498   
## Pos Pred Value : 0.7517   
## Neg Pred Value : 0.7700   
## Prevalence : 0.4953   
## Detection Rate : 0.3822   
## Detection Prevalence : 0.5085   
## Balanced Accuracy : 0.7608   
##   
## 'Positive' Class : X1   
##

print(cm\_svm)

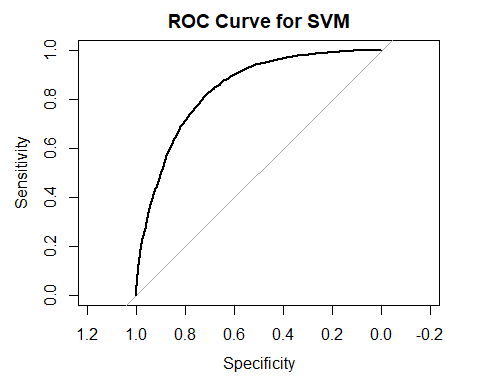
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction X0 X1  
## X0 4145 1238  
## X1 1383 4186  
##   
## Accuracy : 0.7607   
## 95% CI : (0.7526, 0.7686)  
## No Information Rate : 0.5047   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.5214   
##   
## Mcnemar's Test P-Value : 0.004912   
##   
## Sensitivity : 0.7718   
## Specificity : 0.7498   
## Pos Pred Value : 0.7517   
## Neg Pred Value : 0.7700   
## Prevalence : 0.4953   
## Detection Rate : 0.3822   
## Detection Prevalence : 0.5085   
## Balanced Accuracy : 0.7608   
##   
## 'Positive' Class : X1   
##

library(pROC)  
SVMROC <- roc(testMldf$HeartDisease,yPredSVm\_proba\_old)

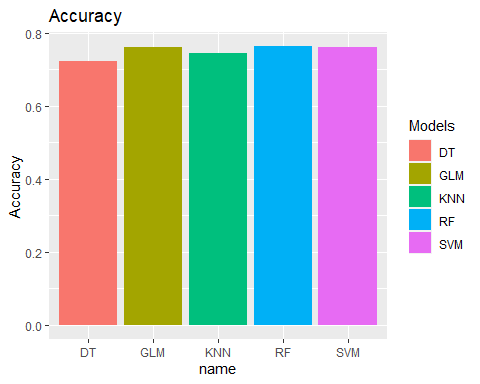
## Setting levels: control = X0, case = X1

## Setting direction: controls < cases

plot(SVMROC, main = "ROC Curve for SVM")



eval\_acc = c(eval\_glm$Acc, eval\_rf$Acc, eval\_dt$Acc, eval\_knn$Acc, eval\_svm$Acc)  
models = c("GLM","RF","DT","KNN","SVM")  
plot\_df = data.frame(  
name=models,  
value = eval\_acc  
)  
ggplot(plot\_df, aes(x=name, y=value, fill=name)) +   
 geom\_bar(stat = "identity")+  
 labs(title = "Accuracy", y = "Accuracy", fill = "Models")



eval\_sens = c(eval\_glm$Sens, eval\_rf$Sens, eval\_dt$Sens, eval\_knn$Sens, eval\_svm$Sens)  
eval\_spec = c(eval\_glm$Spec, eval\_rf$Spec, eval\_dt$Spec, eval\_knn$Spec, eval\_svm$Spec)  
  
plot\_eval\_df = data.frame(  
Models = models,  
Accuracy=eval\_acc,  
Sensitivity = eval\_sens,  
Specificity = eval\_spec  
)  
# ggplot(plot\_eval\_df, aes(fill=models, y=value, x=models)) +   
# geom\_bar(position="dodge", stat="identity")

plot\_eval\_data <- data.frame(Value = c(eval\_acc, eval\_sens, eval\_spec), # Create example data  
 Model = rep(c("GLM",  
 "RF",  
 "DT",  
 "KNN",  
 "SVM"  
 ),  
 times = 3),  
 Metric = c(rep(c("Accuracy"),5),  
 rep(c("Sensitivity"),5),  
 rep(c("Specificity"),5)  
 )  
)  
  
  
  
  
print(plot\_eval\_data)

## Value Model Metric  
## 1 0.7623265 GLM Accuracy  
## 2 0.7637874 RF Accuracy  
## 3 0.7225164 DT Accuracy  
## 4 0.7439737 KNN Accuracy  
## 5 0.7606830 SVM Accuracy  
## 6 0.7772861 GLM Sensitivity  
## 7 0.8117625 RF Sensitivity  
## 8 0.7389653 DT Sensitivity  
## 9 0.7518437 KNN Sensitivity  
## 10 0.7717552 SVM Sensitivity  
## 11 0.7476483 GLM Specificity  
## 12 0.7167149 RF Specificity  
## 13 0.7057522 DT Specificity  
## 14 0.7362518 KNN Specificity  
## 15 0.7498191 SVM Specificity

ggplot(plot\_eval\_data, aes(fill=Metric, y=Value, x=Model)) +   
 geom\_bar(position="dodge", stat="identity")+labs(title = "Evaluation Metrics")

