Name: Alex Lundin

Assignment: Project 1

Data set 1 - heart disease

**Columns labeled** on graph and in notes

**5 R functions** (names, str, head, cor, summary)

**2 meaningful graphs**

–(Scatterplot Chest Pain ~ Heart Disease)

–(Density plot Chest Pain Type)

**Algorithms**

–Linear Regression

– knn clustering

**Metrics**

–r squared for Linear Regression

–predictions for knn clustering

**Analysis**

–this dataset works well with regression since it assigns a numeric value of heart disease diagnosis

Data set 2- Poker hand

**Columns labeled** on graph and in notes

**5 R functions** (names, str, head, cor, summary)

**2 meaningful graphs**

– (Scatterplot: Card 1 value ~ Hand Classfication)

– (pairs plot: all predictors)

– (Density Plot: Card 1 value)

– (Density Plot: Card 1 suit)

**Algorithms**

–Linear Regression

–Logistic Regression

**Metrics**

–r squared for Linear Regression

–accuracy for Logistic Regression

**Analysis**

–this data set works better with classification because it assigns each hand a class.

–poker hands are still subject to alot of radomness that is not very predictable

# Heart disease

This data set is from the following URL: <https://archive.ics.uci.edu/ml/datasets/Heart+Disease>

This study pulls various pieces of data from subjects to learn about factors related to heart disease.

14 attributes of the original 76 attributes were used in the processed data sets.

Complete attribute documentation:

(These are renumbered from the original website to match the trimed data)

1 age: age in years

2 sex: sex (1 = male; 0 = female)

3 cp: chest pain type

– Value 1: typical angina

– Value 2: atypical angina

– Value 3: non-anginal pain

– Value 4: asymptomatic

4 trestbps: resting blood pressure (in mm Hg on admission to the hospital)

5 chol: serum cholestoral in mg/dl

6 fbs: (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)

7 restecg: resting electrocardiographic results

8 thalach: maximum heart rate achieved

9 exang: exercise induced angina (1 = yes; 0 = no)

10 oldpeak = ST depression induced by exercise relative to rest

11 slope: the slope of the peak exercise ST segment – Value 1: upsloping – Value 2: flat – Value 3: downsloping

12 ca: number of major vessels (0-3) colored by flourosopy

13 thal: 3 = normal; 6 = fixed defect; 7 = reversable defect

14 num: diagnosis of heart disease (angiographic disease status) – Value 0: < 50% diameter narrowing – Value 1: > 50% diameter narrowing (in any major vessel: attributes 59 through 68 are vessels)

# load the project data for first set heart disease

# install and load packages  
if(!require('tinytex')){  
 install.packages('tinytex')   
 library('tinytex')  
}

## Loading required package: tinytex

# different file locations for the project  
dataPathHomeComputer <- "C:\\Users\\Alex\\Desktop\\Screen-Cleaner\\GitHub\\UTDSummer2018\\CS-4375.0U2-Machine-Learning\\Projects\\project1\\data\\processed.hungarian.data"  
  
dataPathSchoolComputer <- "H:\\GitHub\\UTDSummer2018\\CS-4375.0U2-Machine-Learning\\Projects\\project1\\data\\processed.hungarian.data"  
  
# create the dataframe for heart disease  
df\_hd <- read.table(dataPathHomeComputer)  
  
# sets the column names of the data frame with colnames function  
colnames(df\_hd) <- c("age", "sex", "cp", "trestbps", "chol", "fbs", "restecg", "thalach", "exang", "oldpeak", "slope", "ca", "thal", "num")  
  
  
  
# separate out the train and test dataframes following a similar naming convention of original frame  
  
# Set random seed to ensure reproducibility of the shuffle.  
set.seed(1958)  
  
  
# shuffle the df\_hd and store into a new df\_hd frame  
df\_hd\_numberOfRows <- nrow(df\_hd)  
shuf\_df\_hd <- df\_hd[sample(df\_hd\_numberOfRows), ]  
  
# set train\_data df\_hd with the train\_data indices  
train\_data\_indices <- 1:round(0.75 \* df\_hd\_numberOfRows)  
train\_data <- shuf\_df\_hd[train\_data\_indices, ]  
# set test\_data df\_hd with the test\_data indices  
test\_data\_indices <- (round(0.75 \* df\_hd\_numberOfRows) + 1):df\_hd\_numberOfRows  
test\_data <- shuf\_df\_hd[test\_data\_indices, ]

# investigate the data with names, str, head, summary and cor

# look at the names of the data frame  
nameArray <- names(df\_hd)  
printString <- "The names of the columns are:"  
  
# print a useful message with a compact version of the columns using str  
print(printString)

## [1] "The names of the columns are:"

str(nameArray)

## chr [1:14] "age" "sex" "cp" "trestbps" "chol" "fbs" "restecg" ...

# show first 6 instances of the frame  
head(df\_hd)

## age sex cp trestbps chol fbs restecg thalach exang oldpeak slope ca thal  
## 1 28 1 2 130 132 0 2 185 0 0 0 0 0  
## 2 29 1 2 120 243 0 0 160 0 0 0 0 0  
## 3 29 1 2 140 0 0 0 170 0 0 0 0 0  
## 4 30 0 1 170 237 0 1 170 0 0 0 0 6  
## 5 31 0 2 100 219 0 1 150 0 0 0 0 0  
## 6 32 0 2 105 198 0 0 165 0 0 0 0 0  
## num  
## 1 0  
## 2 0  
## 3 0  
## 4 0  
## 5 0  
## 6 0

The 4 main columns that correlate highly with a diagonois of heart disease are:

3 // cp // chest pain

9 // exang // exercise induced angina

10 // oldpeak // ST depression induced by exercise relative to rest

11 // slope // slope of the peak exercise ST segment

all four of these have a correlation above .5 with the diagnois of heart disease in patients.

This section prints the correlations to screen

# store a summary of the data frame  
sm <- summary(df\_hd)  
  
# print the summary  
print(sm)

## age sex cp trestbps   
## Min. :28.00 Min. :0.0000 Min. :1.000 Min. : 0.0   
## 1st Qu.:42.00 1st Qu.:0.0000 1st Qu.:2.000 1st Qu.:120.0   
## Median :49.00 Median :1.0000 Median :3.000 Median :130.0   
## Mean :47.83 Mean :0.7245 Mean :2.983 Mean :132.1   
## 3rd Qu.:54.00 3rd Qu.:1.0000 3rd Qu.:4.000 3rd Qu.:140.0   
## Max. :66.00 Max. :1.0000 Max. :4.000 Max. :200.0   
## chol fbs restecg thalach   
## Min. : 0.0 Min. :0.00000 Min. :0.0000 Min. : 0.0   
## 1st Qu.:198.0 1st Qu.:0.00000 1st Qu.:0.0000 1st Qu.:122.0   
## Median :237.0 Median :0.00000 Median :0.0000 Median :140.0   
## Mean :231.2 Mean :0.06803 Mean :0.2177 Mean :138.7   
## 3rd Qu.:277.0 3rd Qu.:0.00000 3rd Qu.:0.0000 3rd Qu.:155.0   
## Max. :603.0 Max. :1.00000 Max. :2.0000 Max. :190.0   
## exang oldpeak slope ca   
## Min. :0.0000 Min. :0.0000 Min. :0.0000 Min. :0   
## 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:0   
## Median :0.0000 Median :0.0000 Median :0.0000 Median :0   
## Mean :0.3027 Mean :0.5861 Mean :0.6701 Mean :0   
## 3rd Qu.:1.0000 3rd Qu.:1.0000 3rd Qu.:2.0000 3rd Qu.:0   
## Max. :1.0000 Max. :5.0000 Max. :3.0000 Max. :0   
## thal num   
## Min. :0.0000 Min. :0.0000   
## 1st Qu.:0.0000 1st Qu.:0.0000   
## Median :0.0000 Median :0.0000   
## Mean :0.5374 Mean :0.3605   
## 3rd Qu.:0.0000 3rd Qu.:1.0000   
## Max. :7.0000 Max. :1.0000

# coerce all predictors and targets as numeric for correlation function  
  
xPredictor <- as.numeric(df\_hd$cp)  
yTarget <- as.numeric(df\_hd$num)  
  
print("Correlation of -- Chest pain and heart disease:")

## [1] "Correlation of -- Chest pain and heart disease:"

cor(xPredictor, yTarget)

## [1] 0.505864

xPredictor <- as.numeric(df\_hd$exang)  
print("Correlation of -- Exercise induced chest pain and heart disease:")

## [1] "Correlation of -- Exercise induced chest pain and heart disease:"

cor(xPredictor, yTarget)

## [1] 0.5845414

xPredictor <- as.numeric(df\_hd$oldpeak)  
print("Correlation of -- ST depression, aka irregular heart beat, due to exercise and heart disease:")

## [1] "Correlation of -- ST depression, aka irregular heart beat, due to exercise and heart disease:"

cor(xPredictor, yTarget)

## [1] 0.5457004

xPredictor <- as.numeric(df\_hd$slope)  
print("Correlation of -- Slope of ST depression during peak exercise and heart disease:")

## [1] "Correlation of -- Slope of ST depression during peak exercise and heart disease:"

cor(xPredictor, yTarget)

## [1] 0.580153

# two informative graphs

The scatter plot shows the likelyhood of heart disease for each type of chest pain (which ranges from 1-4) The scatter plot function in R complains when there is a fixed number of observance values (chest pain from 1-4)

The density plots show the highest correlators do not have a normal distrubtion of occcurances in the data set. The predictors are very un normally distributed.

# scatterplot for data view  
scatter.smooth(x=df\_hd$cp, y=df\_hd$num, main="Scatterplot Chest Pain ~ Heart Disease", xlab="Chest Pain (types 1 through 4)", ylab="Heart disease (< .5 positive diagnosis)")

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =  
## FALSE, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =  
## FALSE, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =  
## FALSE, : reciprocal condition number -0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =  
## FALSE, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =  
## FALSE, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =  
## FALSE, : reciprocal condition number -0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =  
## FALSE, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =  
## FALSE, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =  
## FALSE, : reciprocal condition number -0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =  
## FALSE, : pseudoinverse used at 3

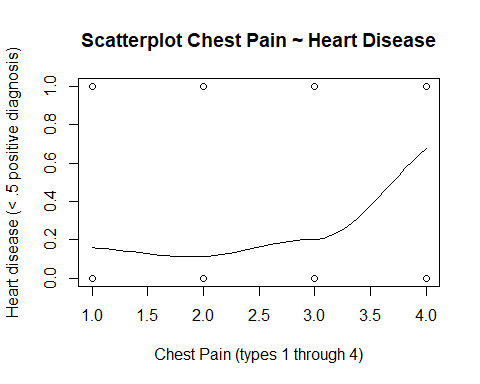
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =  
## FALSE, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =  
## FALSE, : reciprocal condition number -0

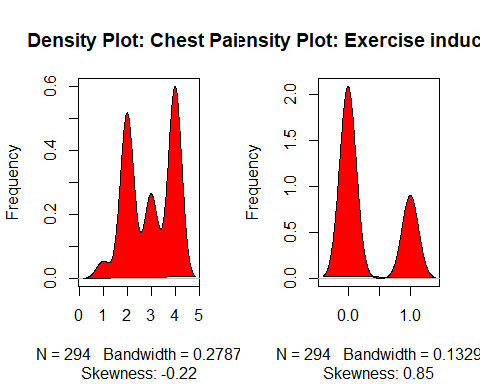
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =  
## FALSE, : pseudoinverse used at 3

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =  
## FALSE, : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =  
## FALSE, : reciprocal condition number -0



# # Density plot  
par(mfrow=c(1, 2)) # divide graph area in 2 columns  
  
plot(density(df\_hd$cp), main="Density Plot: Chest Pain", ylab="Frequency", sub=paste("Skewness:", round(e1071::skewness(df\_hd$cp), 2)))  
polygon(density(df\_hd$cp), col="red")  
  
plot(density(df\_hd$exang), main="Density Plot: Exercise induced CP", ylab="Frequency", sub=paste("Skewness:", round(e1071::skewness(df\_hd$exang), 2)))  
polygon(density(df\_hd$exang), col="red")



# first model for heart disease

# This model uses exercise induced angina as the predictor for the target of a heart disease diagnosis

The model has: Rsquared = .39 low pvalues for the predictor and target low pvalue for the f-stastic

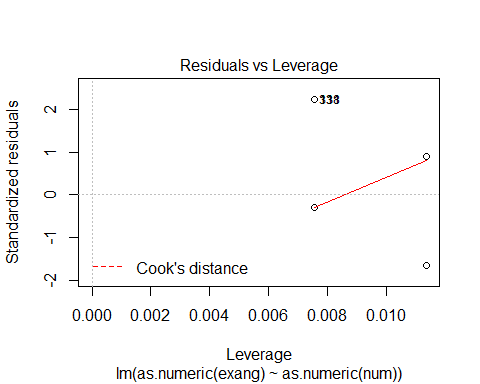
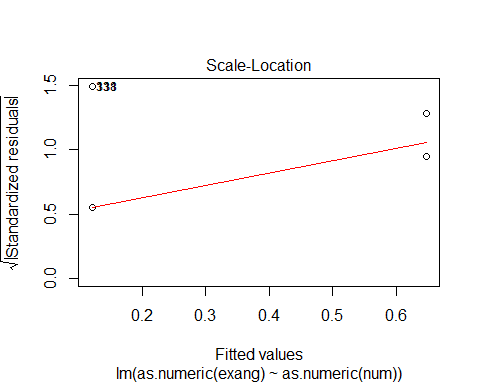
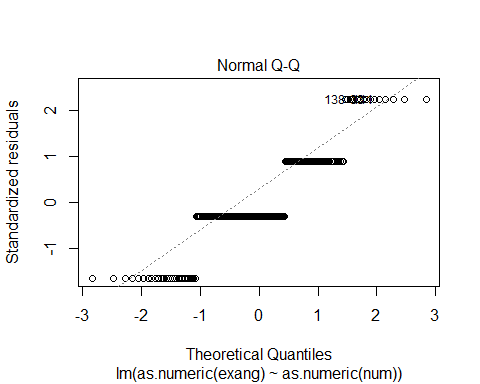
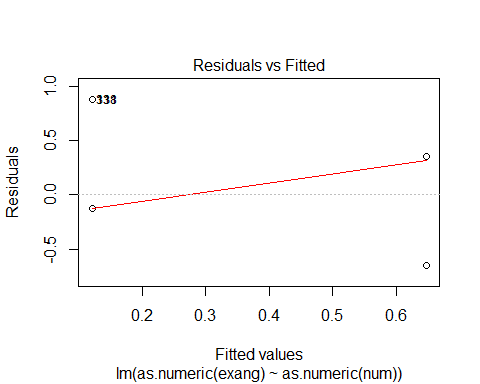
The rsquared value should be closer to 1, but this dataset is attempting to predict something with alot of factors. So the nature of this data explains why the rsquared value is low, it’s complex.

When checking how the model does predicting on the test data. There is a .69 correlation between the predicted value on the test data and the training model. This model is a good represenation, not super duper, but pretty good. I think this model performed well because the data set is suited for linear regression.

# create linear model on train\_data  
train\_lm1 <- lm(as.numeric(exang)~as.numeric(num), data=train\_data)  
print(train\_lm1 )

##   
## Call:  
## lm(formula = as.numeric(exang) ~ as.numeric(num), data = train\_data)  
##   
## Coefficients:  
## (Intercept) as.numeric(num)   
## 0.1212 0.5265

plot(train\_lm1)



# store a summary of the model created from the train data and print it  
lm1\_sm <- summary(train\_lm1)  
print(lm1\_sm)

##   
## Call:  
## lm(formula = as.numeric(exang) ~ as.numeric(num), data = train\_data)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.6477 -0.1212 -0.1212 0.3523 0.8788   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.12121 0.03444 3.519 0.000527 \*\*\*  
## as.numeric(num) 0.52652 0.05446 9.668 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3957 on 218 degrees of freedom  
## Multiple R-squared: 0.3001, Adjusted R-squared: 0.2969   
## F-statistic: 93.46 on 1 and 218 DF, p-value: < 2.2e-16

# attempt to predict the target y value of the test data with the train linear model  
pred <- predict(train\_lm1, newdata=test\_data)  
print("Correlation of -- Prediction of heart disease using training model with test data:")

## [1] "Correlation of -- Prediction of heart disease using training model with test data:"

cor(pred, as.numeric(test\_data$exang))

## [1] 0.6968762

# second model for heart disease

# this model uses knn clustering with columns 9 and 10

# exang and oldpeak to attempt to predict heart disease diagnosis based on those two predictors

correlation value of predictions vs actual: .66 this model was slightly less accurate than the regression model. This data set is more suitable towards linear regression models, so I think that’s why this model is less accurate since it uses classification when all of this data set is numeric type.

# install and load packages  
if(!require('caret')){  
 install.packages('caret')   
 library('caret')  
}

## Loading required package: caret

## Loading required package: lattice

## Loading required package: ggplot2

if(!require('DMwR')){  
 install.packages('DMwR')   
 library('DMwR')  
}

## Loading required package: DMwR

## Loading required package: grid

# create the second linear model for heart disease data  
train\_lm2 <- knnreg(train\_data[,9:10], train\_data[,14], k=3)  
  
# store a summary of the model created from the train data and print it  
lm2\_sm <- summary(train\_lm2)  
print(lm2\_sm)

## Length Class Mode   
## learn 2 -none- list   
## k 1 -none- numeric  
## theDots 0 -none- list

# correlate how well the model did  
# do this by comparing the performace of the model using test data for columns 9 and 10  
# which are exang and oldpeak  
# the model used those to predict heart disease based on those factors  
predictions <- predict(train\_lm2, test\_data[,9:10])  
# now correlate the predictions against the actual values in the test data  
print("Correlation of -- Prediction of heart disease using training model with test data:")

## [1] "Correlation of -- Prediction of heart disease using training model with test data:"

cor(predictions, test\_data$num)

## [1] 0.6620159

# Poker Hands

This data set is from the following URL: <https://archive.ics.uci.edu/ml/datasets/Heart+Disease>

1. S1 “Suit of card #1” Ordinal (1-4) representing {Hearts, Spades, Diamonds, Clubs}
2. C1 “Rank of card #1” Numerical (1-13) representing (Ace, 2, 3, … , Queen, King)
3. S2 “Suit of card #2” Ordinal (1-4) representing {Hearts, Spades, Diamonds, Clubs}
4. C2 “Rank of card #2” Numerical (1-13) representing (Ace, 2, 3, … , Queen, King)
5. S3 “Suit of card #3” Ordinal (1-4) representing {Hearts, Spades, Diamonds, Clubs}
6. C3 “Rank of card #3” Numerical (1-13) representing (Ace, 2, 3, … , Queen, King)
7. S4 “Suit of card #4” Ordinal (1-4) representing {Hearts, Spades, Diamonds, Clubs}
8. C4 “Rank of card #4” Numerical (1-13) representing (Ace, 2, 3, … , Queen, King)
9. S5 “Suit of card #5” Ordinal (1-4) representing {Hearts, Spades, Diamonds, Clubs}
10. C5 “Rank of card 5” Numerical (1-13) representing (Ace, 2, 3, … , Queen, King)
11. CLASS “Poker Hand” Ordinal (0-9)

0: Nothing in hand; not a recognized poker hand 1: One pair; one pair of equal ranks within five cards 2: Two pairs; two pairs of equal ranks within five cards 3: Three of a kind; three equal ranks within five cards 4: Straight; five cards, sequentially ranked with no gaps 5: Flush; five cards with the same suit 6: Full house; pair + different rank three of a kind 7: Four of a kind; four equal ranks within five cards 8: Straight flush; straight + flush 9: Royal flush; {Ace, King, Queen, Jack, Ten} + flush

# load the project data for second set, poker hands (ph)

# different file locations for the project  
dataPathHomeComputer <- "C:\\Users\\Alex\\Desktop\\Screen-Cleaner\\GitHub\\UTDSummer2018\\CS-4375.0U2-Machine-Learning\\Projects\\project1\\data\\processed.pokerhand.data"  
  
dataPathSchoolComputer <- "H:\\GitHub\\UTDSummer2018\\CS-4375.0U2-Machine-Learning\\Projects\\project1\\data\\processed.pokerhand.data"  
  
# create the dataframe for heart disease  
df\_ph <- read.table(dataPathHomeComputer)  
  
# sets the column names of the data frame with colnames function  
colnames(df\_ph) <- c("S1","C1","S2","C2","S3","C3","S4","C4","S5","C5","CLASS")  
  
  
  
# separate out the train and test dataframes following a similar naming convention of original frame  
  
# Set random seed to ensure reproducibility of the shuffle.  
set.seed(1958)  
  
  
# shuffle the df\_ph and store into a new df\_ph frame  
df\_ph\_numberOfRows <- nrow(df\_ph)  
shuf\_df\_ph <- df\_ph[sample(df\_ph\_numberOfRows), ]  
  
# set train\_ph\_data df\_ph with the train\_ph\_data indices  
train\_ph\_data\_indices <- 1:round(0.75 \* df\_ph\_numberOfRows)  
train\_ph\_data <- shuf\_df\_ph[train\_ph\_data\_indices, ]  
# set test\_ph\_data df\_ph with the test\_ph\_data indices  
test\_ph\_data\_indices <- (round(0.75 \* df\_ph\_numberOfRows) + 1):df\_ph\_numberOfRows  
test\_ph\_data <- shuf\_df\_ph[test\_ph\_data\_indices, ]

# investigate the data with names, str, head, summary and cor

# look at the names of the data frame  
nameArray <- names(df\_ph)  
printString <- "The names of the columns are:"  
  
# print a useful message with a compact version of the columns using str  
print(printString)

## [1] "The names of the columns are:"

str(nameArray)

## chr [1:11] "S1" "C1" "S2" "C2" "S3" "C3" "S4" "C4" "S5" "C5" "CLASS"

# show first 6 instances of the frame  
head(df\_ph)

## S1 C1 S2 C2 S3 C3 S4 C4 S5 C5 CLASS  
## 1 1 1 1 13 2 4 2 3 1 12 0  
## 2 3 12 3 2 3 11 4 5 2 5 1  
## 3 1 9 4 6 1 4 3 2 3 9 1  
## 4 1 4 3 13 2 13 2 1 3 6 1  
## 5 3 10 2 7 1 2 2 11 4 9 0  
## 6 1 3 4 5 3 4 1 12 4 6 0

None of the indivudal columns correlated well with the Royal Flush hand classification: all were under .015

# store a summary of the data frame  
sm <- summary(df\_ph)  
  
# print the summary  
print(sm)

## S1 C1 S2 C2   
## Min. :1.0 Min. : 1.000 Min. :1.000 Min. : 1.000   
## 1st Qu.:2.0 1st Qu.: 4.000 1st Qu.:1.000 1st Qu.: 4.000   
## Median :2.0 Median : 7.000 Median :2.000 Median : 7.000   
## Mean :2.5 Mean : 6.935 Mean :2.471 Mean : 7.024   
## 3rd Qu.:3.0 3rd Qu.:10.000 3rd Qu.:3.000 3rd Qu.:10.000   
## Max. :4.0 Max. :13.000 Max. :4.000 Max. :13.000   
## S3 C3 S4 C4   
## Min. :1.000 Min. : 1.000 Min. :1.000 Min. : 1.000   
## 1st Qu.:2.000 1st Qu.: 4.000 1st Qu.:2.000 1st Qu.: 4.000   
## Median :3.000 Median : 7.000 Median :3.000 Median : 7.000   
## Mean :2.512 Mean : 6.945 Mean :2.506 Mean : 6.933   
## 3rd Qu.:4.000 3rd Qu.:10.000 3rd Qu.:4.000 3rd Qu.:10.000   
## Max. :4.000 Max. :13.000 Max. :4.000 Max. :13.000   
## S5 C5 CLASS   
## Min. :1.000 Min. : 1.000 Min. :0.0000   
## 1st Qu.:2.000 1st Qu.: 4.000 1st Qu.:0.0000   
## Median :3.000 Median : 7.000 Median :0.0000   
## Mean :2.522 Mean : 6.994 Mean :0.6244   
## 3rd Qu.:4.000 3rd Qu.:10.000 3rd Qu.:1.0000   
## Max. :4.000 Max. :13.000 Max. :7.0000

# coerce all predictors and targets as numeric for correlation function  
  
xPredictor <- as.numeric(df\_ph$C1)  
yTarget <- as.numeric(df\_ph$CLASS)  
  
print("Correlation of -- Card 1 value and Hand Classfication:")

## [1] "Correlation of -- Card 1 value and Hand Classfication:"

cor(xPredictor, yTarget)

## [1] -0.01236034

xPredictor <- as.numeric(df\_ph$S1)  
print("Correlation of -- Card 1 suit and Hand Classfication:")

## [1] "Correlation of -- Card 1 suit and Hand Classfication:"

cor(xPredictor, yTarget)

## [1] -0.01133808

xPredictor <- as.numeric(df\_ph$C3)  
print("Correlation of -- Card 3 and Hand Classfication:")

## [1] "Correlation of -- Card 3 and Hand Classfication:"

cor(xPredictor, yTarget)

## [1] 0.01405256

xPredictor <- as.numeric(df\_ph$S3)  
print("Correlation of -- Card 3 suit and Hand Classfication:")

## [1] "Correlation of -- Card 3 suit and Hand Classfication:"

cor(xPredictor, yTarget)

## [1] 0.004862796

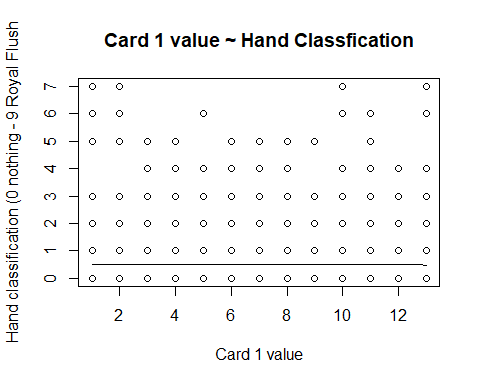
# two informative graphs

The scatter plot shows the likelyhood of hand classfication based on card 1 value The scatter plot shows that there is no correlation between the two

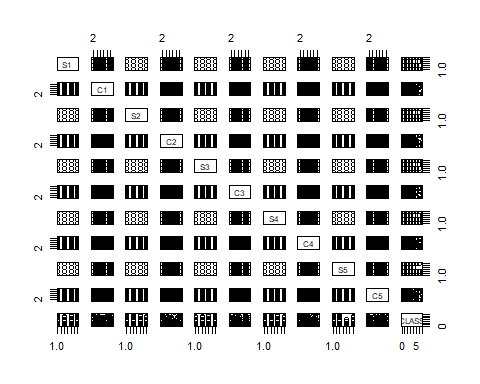
The pairs graph gives a broad view of the entire data set

The density plots show the distribution of certain predictors in the data set. In this case I chose card 1 value and card 1 suit. The predictors are very normally distributed.

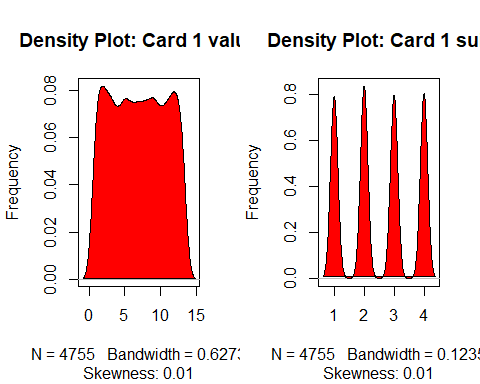
# scatterplot for data view  
scatter.smooth(x=df\_ph$C1, y=df\_ph$CLASS, main="Card 1 value ~ Hand Classfication", xlab="Card 1 value", ylab="Hand classification (0 nothing - 9 Royal Flush")



#pairs plot  
pairs(df\_ph)



# # Density plot  
par(mfrow=c(1, 2)) # divide graph area in 2 columns  
  
plot(density(df\_ph$C1), main="Density Plot: Card 1 value", ylab="Frequency", sub=paste("Skewness:", round(e1071::skewness(df\_ph$C1), 2)))  
polygon(density(df\_ph$C1), col="red")  
  
plot(density(df\_ph$S1), main="Density Plot: Card 1 suit", ylab="Frequency", sub=paste("Skewness:", round(e1071::skewness(df\_ph$S1), 2)))  
polygon(density(df\_ph$S1), col="red")



# first model for poker hands

# This model uses card 1 type to attempt to predict entire hand classification

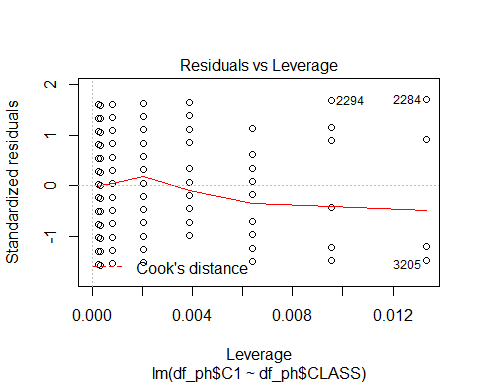
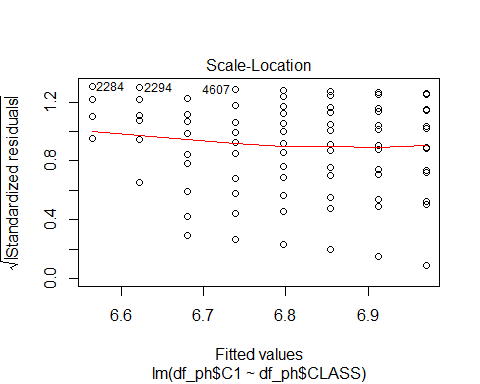
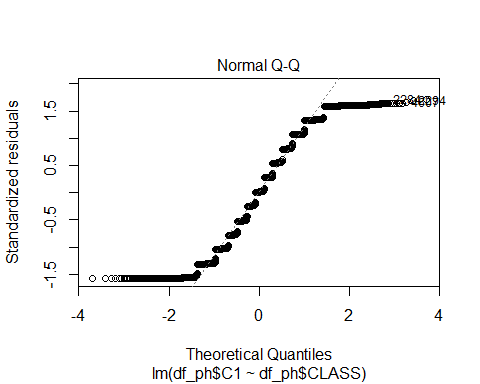
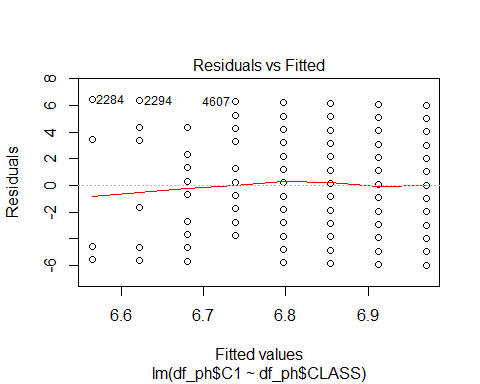
The model has: Rsquared = .000 low pvalues for the predictor high pvalue for the target low pvalue for the f-stastic

The rsquared value shows this regression technique explains none of the variance in the predictors. So the nature of this data explains why the rsquared value is low. This data set is not intended for regression

# create linear model on train\_ph\_data  
train\_lm1\_ph <- lm(df\_ph$C1~df\_ph$CLASS, data=train\_ph\_data)  
print(train\_lm1\_ph)

##   
## Call:  
## lm(formula = df\_ph$C1 ~ df\_ph$CLASS, data = train\_ph\_data)  
##   
## Coefficients:  
## (Intercept) df\_ph$CLASS   
## 6.97082 -0.05802

plot(train\_lm1\_ph)



# store a summary of the model created from the train data and print it  
lm1\_ph\_sm <- summary(train\_lm1\_ph)  
print(lm1\_ph\_sm)

##   
## Call:  
## lm(formula = df\_ph$C1 ~ df\_ph$CLASS, data = train\_ph\_data)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -5.9708 -2.9708 0.0292 3.0872 6.4353   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.97082 0.06948 100.322 <2e-16 \*\*\*  
## df\_ph$CLASS -0.05802 0.06808 -0.852 0.394   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 3.79 on 4753 degrees of freedom  
## Multiple R-squared: 0.0001528, Adjusted R-squared: -5.758e-05   
## F-statistic: 0.7263 on 1 and 4753 DF, p-value: 0.3941

# second model for poker card hand

# this model uses logistic regression to attempt to predict hand classification from card values and suits

the accuracy is .5 which is alot higher than the regression used in the previous model

this data set is more geared towards classification, so that explains why logistic regression did better.

# scale the value which determines the hand classification so it ranges between .1 and .9  
# this allows glm to work on the y value  
train\_ph\_data[,11]<-train\_ph\_data[,11]\*.1  
test\_ph\_data[,11]<-test\_ph\_data[,11]\*.1  
  
train\_lm2\_ph <- glm(CLASS~., data=train\_ph\_data, family="binomial")

## Warning in eval(family$initialize): non-integer #successes in a binomial  
## glm!

# store a summary of the model created from the train data and print it  
lm2\_ph\_sm <- summary(train\_lm2\_ph)  
print(lm2\_ph\_sm)

##   
## Call:  
## glm(formula = CLASS ~ ., family = "binomial", data = train\_ph\_data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -0.4002 -0.3585 -0.3297 0.1483 1.6094   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -2.3799397 0.4655810 -5.112 3.19e-07 \*\*\*  
## S1 -0.0219364 0.0624395 -0.351 0.725   
## C1 -0.0036185 0.0182600 -0.198 0.843   
## S2 -0.0307575 0.0620979 -0.495 0.620   
## C2 -0.0122942 0.0185530 -0.663 0.508   
## S3 0.0022338 0.0616806 0.036 0.971   
## C3 0.0030229 0.0184489 0.164 0.870   
## S4 -0.0014471 0.0619946 -0.023 0.981   
## C4 -0.0044036 0.0184200 -0.239 0.811   
## S5 -0.0296658 0.0621843 -0.477 0.633   
## C5 -0.0008516 0.0188071 -0.045 0.964   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 389.06 on 3565 degrees of freedom  
## Residual deviance: 387.93 on 3555 degrees of freedom  
## AIC: 546.36  
##   
## Number of Fisher Scoring iterations: 5

probs <- predict(train\_lm2\_ph, newdata=test\_ph\_data)  
pred <- ifelse(probs>0.0, 0.9, 0)  
acc <- mean(pred==test\_ph\_data$CLASS)  
print(paste("accuracy = ", acc))

## [1] "accuracy = 0.507148864592094"

table(pred, test\_ph\_data$CLASS)

##   
## pred 0 0.1 0.2 0.3 0.4 0.5 0.7  
## 0 603 488 58 27 11 1 1