Kickstarter Data

Columns Explanation

1) ID = internal kickstarter id
2) name = name of project - A project is a finite work with a clear goal that you'd like to bring to life. Think albums, books, or films.
3) category = category
4) main_category = category of campaign
5) currency = currency used to support
6) deadline = deadline for crowdfunding
7) goal = fundraising goal - The funding goal is the amount of money that a creator needs to complete their project.
8) launched = launched date launched
9) pledged = amount pledged by "crowd"
10) state = Current condition the project is in
11) backers = number of backers
12) country = country pledged from
13) usd pledged = amount of money pledged

Packages Needed to Download

```
#install.packages("ggplot2")
library(ggplot2)
library(doBy)
#install.packages("useful")
library(useful)
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loading required package: foreach
```

```
## Loaded glmnet 2.0-13
```

```
#install.packages('rpart')
library(rpart)
#install.packages("randomForest")
library("randomForest")
```

```
## randomForest 4.6-12
```

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

```
##
## Attaching package: 'randomForest'
```

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

```
set.seed(1861)
```

```
#reads in data into KickStar
KickStar <- read.csv("/Users/Cheddar3/Desktop/MGSC310/Group Projects/kickstarter-projects/ks-projects-2
#removing a few columns with null values
```

```r
KickStar <- subset(KickStar, select = -c(14,15,16,17))
#colSums(is.na(KickStar))
KickStar<- KickStar[!is.na(KickStar$name),]
#colSums(is.na(KickStar))
#cleaning up category column
KickStar$category <- factor(KickStar$category, levels = c("Zines","Young Adult", "World Music", "Worksh
#colSums(is.na(KickStar))
KickStar<- KickStar[!is.na(KickStar$category),]
#colSums(is.na(KickStar))
#cleaning main_category column
#table(KickStar$main_category)
KickStar$main_category <- factor(KickStar$main_category, levels = c("Theater", "Technology", "Publishin
#colSums(is.na(KickStar))
#clearning currency column
#table(KickStar$currency)
KickStar$currency <- factor(KickStar$currency, levels = c("USD", "SGD", "SEK", "NZD", "NOK", "MXN", "HK
#colSums(is.na(KickStar))
#cleaning goal column
KickStar$goal <- as.numeric(KickStar$goal)
#colSums(is.na(KickStar))
#cleaning pledge column
KickStar$pledged <- as.numeric(KickStar$pledged)
#cleaning state column
#table(KickStar$state)
KickStar$state <- factor(KickStar$state, levels = c("failed", "successful"))
#colSums(is.na(KickStar))
KickStar<- KickStar[!is.na(KickStar$state),]
#colSums(is.na(KickStar))
#cleaning backers column
KickStar$backers <- as.numeric(KickStar$backers)
#colSums(is.na(KickStar))
#cleaning country column
#table(KickStar$country)
KickStar$country <- factor(KickStar$country, levels = c("US", "SG", "SE", "NZ", "NO", "NL", "MX", "LU",
#colSums(is.na(KickStar))
KickStar<- KickStar[!is.na(KickStar$country),]
#colSums(is.na(KickStar))
#cleaning usd.pledged column
KickStar$usd.pledged <- as.numeric(KickStar$usd.pledged)
#colSums(is.na(KickStar))
#creates deadline and launched variable so its just year/month/day
KickStar$newDeadline <- as.Date(as.POSIXlt(KickStar$deadline, origin = "1582-10-14",tz = "GMT"))
```

```
## Warning in strptime(xx, f <- "%Y-%m-%d %H:%M:%OS", tz = tz): unknown
## timezone 'zone/tz/2018g.1.0/zoneinfo/America/Los_Angeles'
```

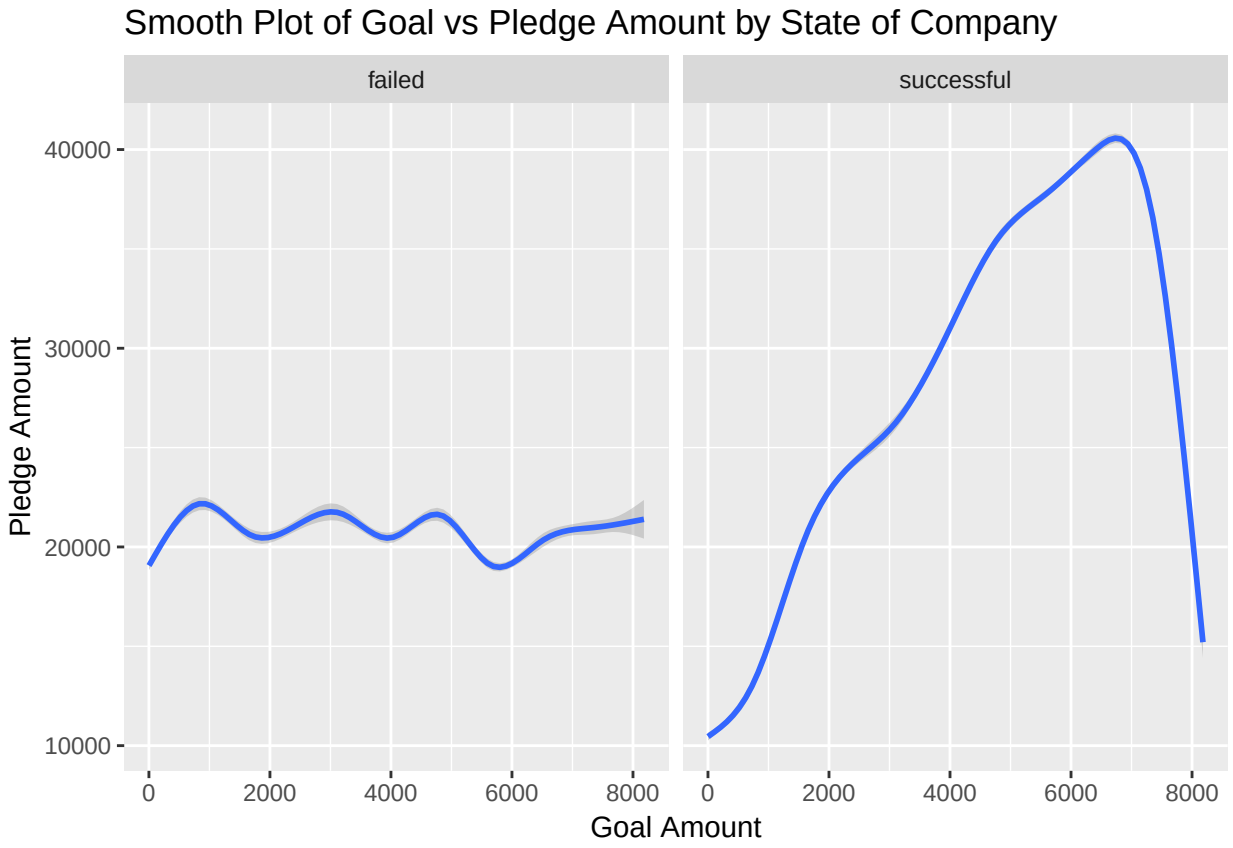
```r
KickStar$newLaunched <- as.Date(as.POSIXlt(KickStar$launched, origin = "1582-10-14",tz = "GMT"))
#difference between launched and deadline in days
KickStar$DateDiffDays <- KickStar$newDeadline - KickStar$newLaunched
#if successful, then 1, else 0
KickStar$binomState <- ifelse(KickStar$state == "successful",1,0)
trainSize <- 0.75
trainInd <- sample(1:nrow(KickStar), size = floor(nrow(KickStar) * trainSize))
KickTrain <- KickStar[trainInd, ]
```

```r
KickTest <- KickStar[-trainInd, ]
#gets summary of data
summary(KickStar)
```

```
##        ID                                     name
##  Min.   :5.971e+03   New EP/Music Development  :    13
##  1st Qu.:5.373e+08   New EP / Music Development:    10
##  Median :1.076e+09   Music Video               :     9
##  Mean   :1.075e+09   Reflections               :     9
##  3rd Qu.:1.611e+09   A Midsummer Night's Dream :     8
##  Max.   :2.147e+09   Pizza                     :     8
##                      (Other)                   :281034
##            category           main_category      currency
##  Product Design: 14515   Film & Video:51034   USD    :229365
##  Documentary   : 13346   Music       :40808   GBP    : 23820
##  Shorts        : 10775   Publishing  :30143   CAD    :  9899
##  Music         : 10724   Games       :22398   EUR    :  9156
##  Food          :  9517   Art         :21745   AUD    :  5183
##  Tabletop Games:  8819   Technology  :21405   SEK    :  1065
##  (Other)       :213395   (Other)     :93558   (Other):  2603
##                 deadline           goal                launched
##  2012-01-01 05:59:00:    46   Min.   :   1   2009-09-15 05:56:28:     2
##  2014-11-01 04:59:00:    34   1st Qu.:1257   2010-06-30 17:29:43:     2
##  2015-01-01 05:59:00:    34   Median :4145   2011-02-08 04:29:48:     2
##  2012-03-01 05:59:00:    32   Mean   :3693   2011-02-25 09:58:36:     2
##  2010-08-01 05:59:00:    28   3rd Qu.:5820   2011-03-07 17:11:18:     2
##  2010-09-01 05:59:00:    28   Max.   :8181   2011-06-07 06:42:01:     2
##  (Other)            :280889                  (Other)            :281079
##     pledged              state           backers        country
##  Min.   :    1   failed    :168115   Min.   :   1   US     :229365
##  1st Qu.: 5597   successful:112976   1st Qu.: 156   GB     : 23820
##  Median :23388                       Median :1395   CA     :  9899
##  Mean   :23310                       Mean   :1465   AU     :  5183
##  3rd Qu.:39532                       3rd Qu.:2537   DE     :  2130
##  Max.   :55597                       Max.   :3587   NL     :  1869
##                                                     (Other):  8825
##   usd.pledged     newDeadline          newLaunched
##  Min.   :    2   Min.   :2009-05-03   Min.   :2009-04-21
##  1st Qu.: 9838   1st Qu.:2012-12-26   1st Qu.:2012-11-22
##  Median :39322   Median :2014-08-11   Median :2014-07-11
##  Mean   :39382   Mean   :2014-04-13   Mean   :2014-03-10
##  3rd Qu.:66703   3rd Qu.:2015-08-07   3rd Qu.:2015-07-05
##  Max.   :94361   Max.   :2016-12-06   Max.   :2016-12-01
##
##  DateDiffDays        binomState
##  Length:281091     Min.   :0.0000
##  Class :difftime   1st Qu.:0.0000
##  Mode  :numeric    Median :0.0000
##                    Mean   :0.4019
##                    3rd Qu.:1.0000
##                    Max.   :1.0000
##
```

```
#ggplot of Goal Amount and Pledge Amount by state of company
ggplot(data = KickStar, aes(goal, pledged)) + geom_smooth() + facet_wrap(~state) + labs(title = "Smooth
```

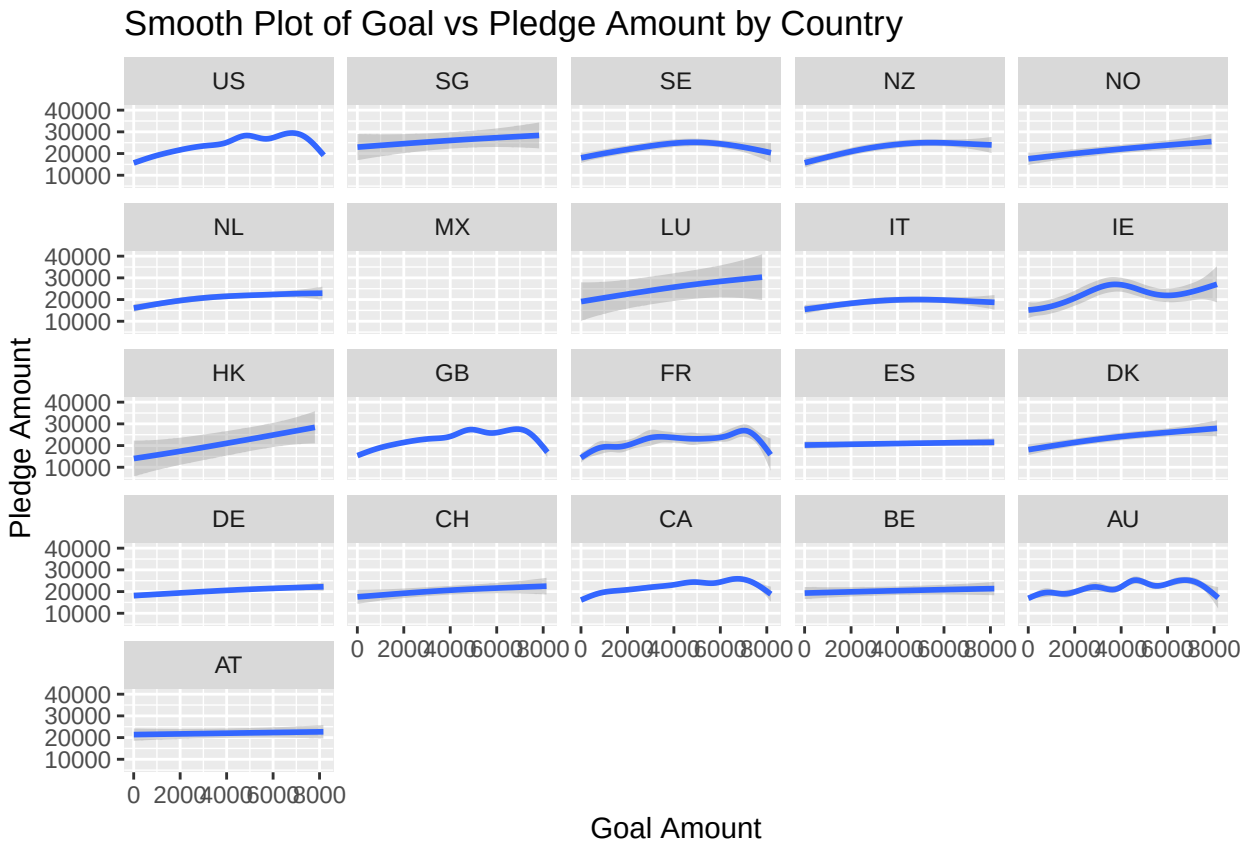
```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



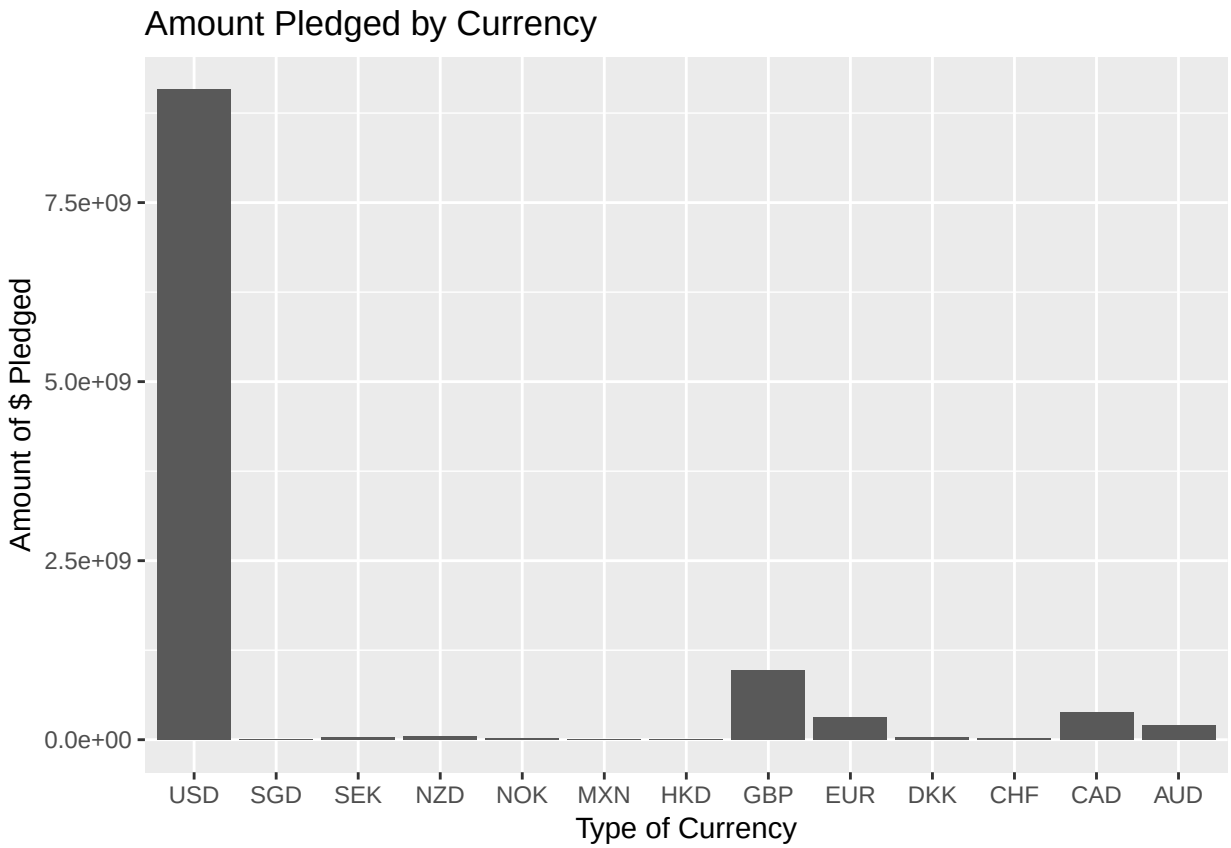

```
#ggplot of Goal Amount and Pledge Aamount by country
ggplot(data = KickStar, aes(goal, pledged)) + geom_smooth() + facet_wrap(~country) + labs(title = "Smoo
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

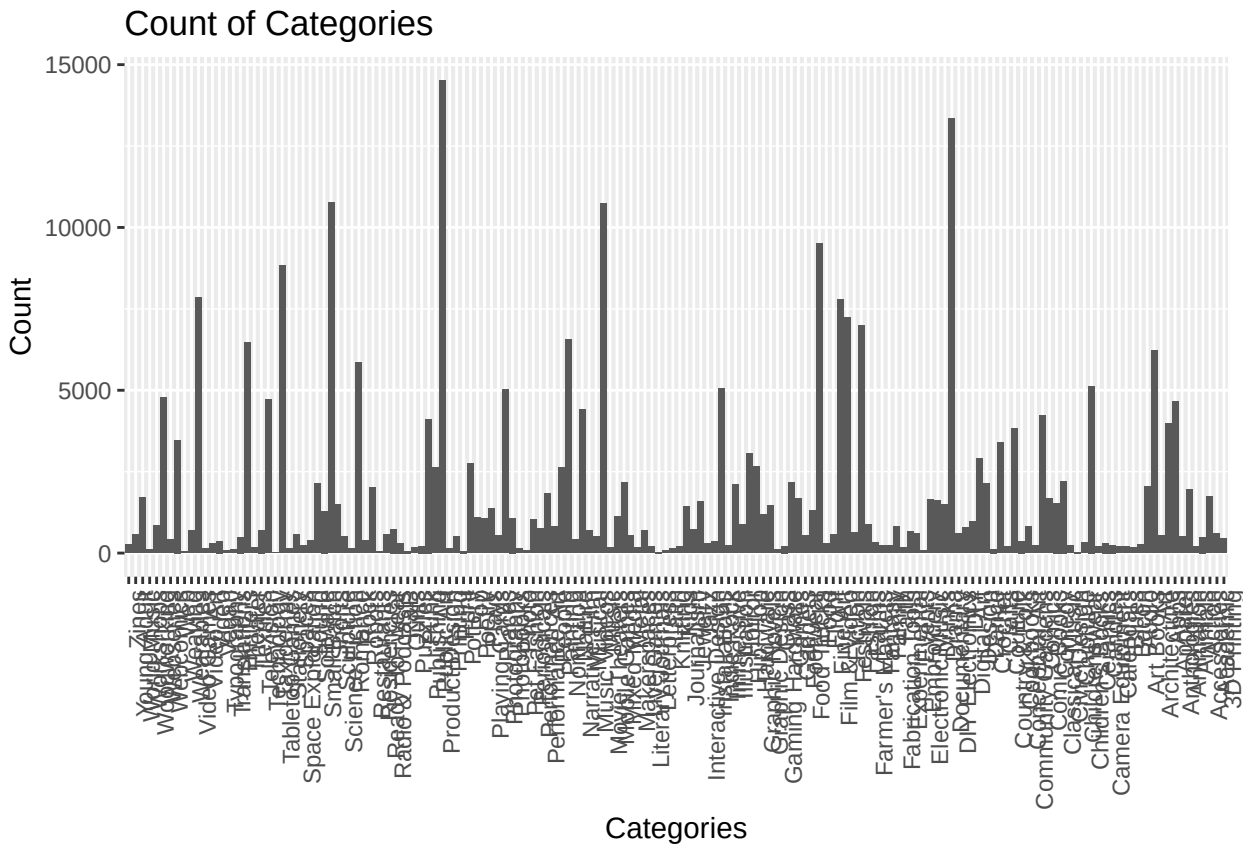
```
## Warning: Computation failed in `stat_smooth()`:
## x has insufficient unique values to support 10 knots: reduce k.
```



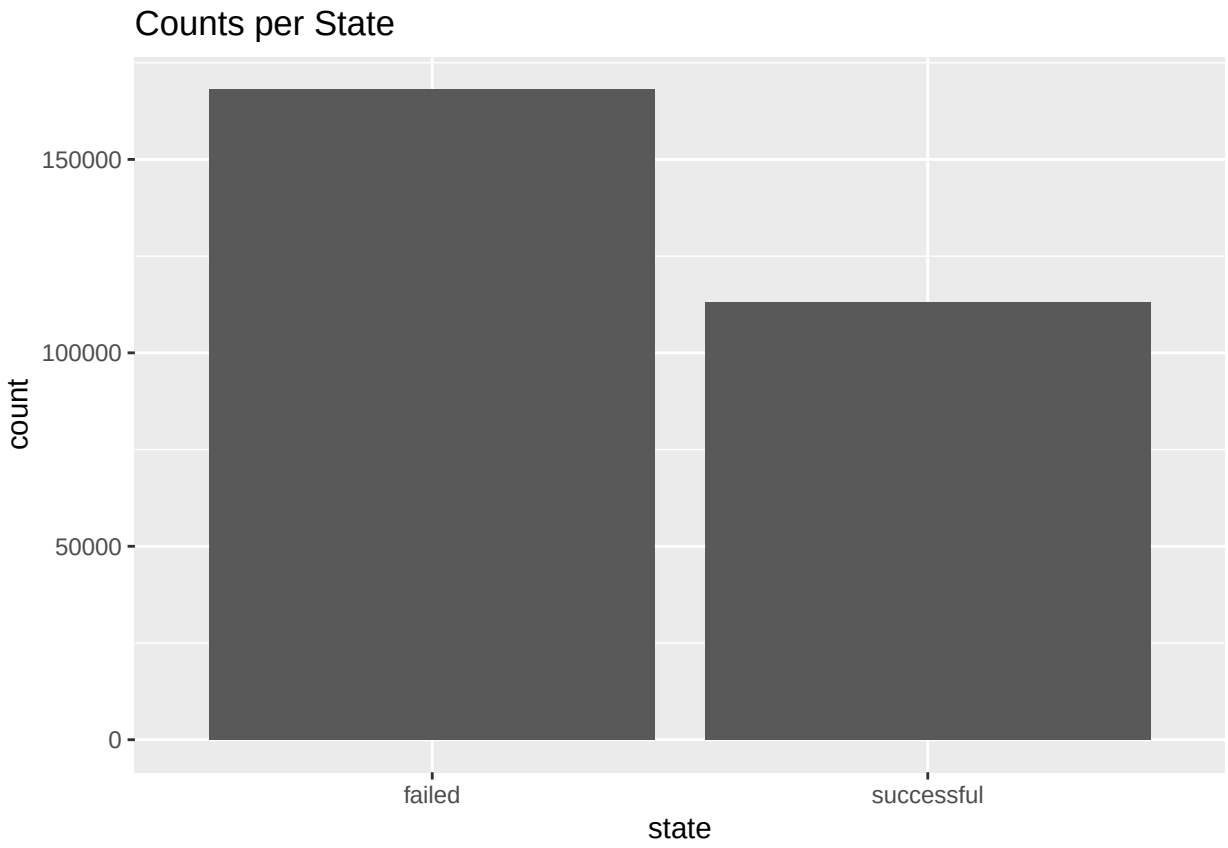

```
#ggplot of bar graph with type of currency and the amount pledged
ggplot(data = KickStar, aes(x = currency, y = usd.pledged)) + geom_bar(stat = "identity") + labs(x = "Ty
```




```
##ggplot of bar graph of count of each categories
ggplot(data = KickStar, aes(category)) + geom_bar() + theme(axis.text.x = element_text(angle = 90, hjus
```




```
#ggplot of bar graph of count by state
ggplot(data = KickStar, aes(state)) + geom_bar() + labs(title = "Counts per State")
```



Logistic Regression Models

Using the training data to create a model

Find summary statistics when the Kickstarter succeed (1) or did not (0)

```
summaryBy(goal + pledged + backers + currency ~ binomState, data = KickTrain, fun = c(mean,sd))
```

```
##   binomState goal.FUN1 pledged.FUN1 backers.FUN1 currency.FUN1
## 1          0  3686.835     20404.68     1196.640      2.714879
## 2          1  3703.354     27685.02     1867.916      2.236159
```

It appears that Kickstarters that had less funding, less pledges, less currency, and less backers failed.

Creating a glm model on the training data and summary

```
glmmodTrain <- glm(binomState ~ goal + pledged + backers + DateDiffDays + currency, data = KickTrain, fa
summary(glmmodTrain)
```

```
##
## Call:
## glm(formula = binomState ~ goal + pledged + backers + DateDiffDays +
##     currency, family = binomial, data = KickTrain)
##
```

```
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9011  -0.9887  -0.6527   1.1424   2.3405
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -6.615e-01  1.723e-02 -38.386  < 2e-16 ***
## goal         -3.085e-05  1.993e-06 -15.477  < 2e-16 ***
## pledged       1.750e-05  2.882e-07  60.735  < 2e-16 ***
## backers       4.481e-04  4.248e-06 105.480  < 2e-16 ***
## DateDiffDays -1.992e-02  3.884e-04 -51.274  < 2e-16 ***
## currencySGD  -3.643e-02  3.010e-01  -0.121 0.903672
## currencySEK  -4.454e-01  8.063e-02  -5.524 3.31e-08 ***
## currencyNZD  -4.454e-01  8.278e-02  -5.380 7.44e-08 ***
## currencyNOK  -5.789e-01  1.339e-01  -4.325 1.53e-05 ***
## currencyMXN  -8.475e-01  1.351e+00  -0.628 0.530318
## currencyHKD  -4.955e-01  3.848e-01  -1.288 0.197784
## currencyGBP  -1.113e-01  1.702e-02  -6.543 6.04e-11 ***
## currencyEUR  -7.401e-01  3.015e-02 -24.546  < 2e-16 ***
## currencyDKK  -3.311e-01  9.988e-02  -3.315 0.000917 ***
## currencyCHF  -8.959e-01  1.470e-01  -6.093 1.11e-09 ***
## currencyCAD  -4.174e-01  2.701e-02 -15.456  < 2e-16 ***
## currencyAUD  -5.758e-01  3.766e-02 -15.290  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 284073  on 210817  degrees of freedom
## Residual deviance: 259355  on 210801  degrees of freedom
## AIC: 259389
##
## Number of Fisher Scoring iterations: 4
```

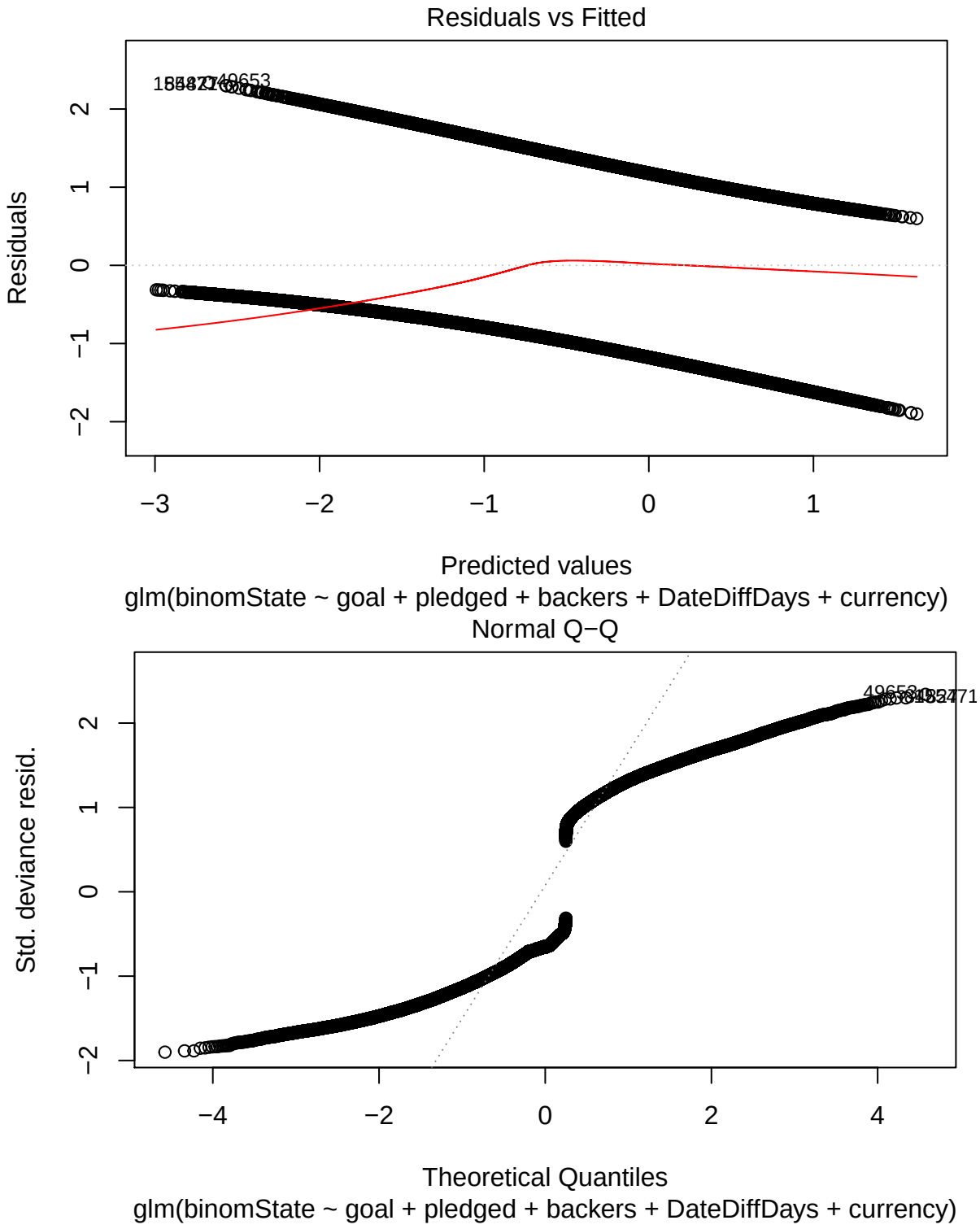
Find odds ratios

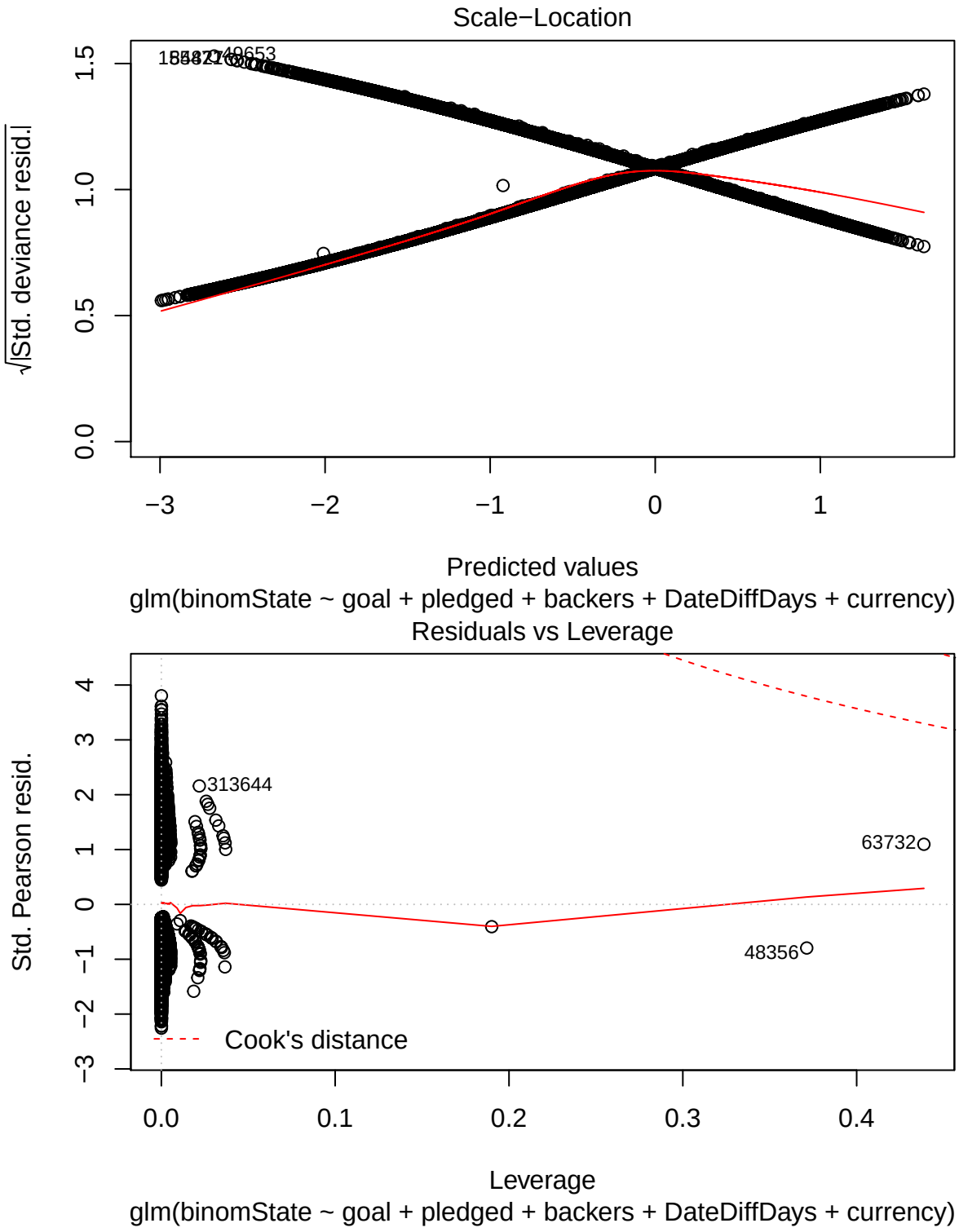
```
exp(glmmodTrain$coefficients)
```

```
##  (Intercept)         goal      pledged      backers DateDiffDays
##    0.5160980    0.9999691    1.0000175    1.0004482    0.9802806
##  currencySGD  currencySEK  currencyNZD  currencyNOK  currencyMXN
##    0.9642298    0.6405436    0.6405719    0.5605136    0.4284974
##  currencyHKD  currencyGBP  currencyEUR  currencyDKK  currencyCHF
##    0.6092383    0.8946348    0.4770469    0.7181443    0.4082378
##  currencyCAD  currencyAUD
##    0.6587364    0.5622580
```

Plotting the model

```
plot(glmmodTrain)
```


Residuals vs Fitted
Residuals
Predicted values
glm(binomState ~ goal + pledged + backers + DateDiffDays + currency)
Normal Q−Q
Std. deviance resid.
Theoretical Quantiles
glm(binomState ~ goal + pledged + backers + DateDiffDays + currency)



Creating probabilty of failure (1) or not (0)

```
glmTrainProbs <- predict.glm(glmmodTrain, type = "response")
hist(glmTrainProbs)
```


```
table(KickStar$binomState)
```

```
##
##      0      1
## 168115 112976
```

```
glmTrainPreds <- rep("Failed", nrow(KickTrain))
glmTrainPreds[glmTrainProbs > 0.5] <- "Succeed"
conftab <- table(glmTrainPreds, true = KickTrain$binomState)
print(conftab)
```

```
##              true
## glmTrainPreds     0     1
##       Failed  97362 48772
##       Succeed 28746 35938
```

```
TPtrain <- conftab[2, 2]/(conftab[1, 2] + conftab[2, 2])
TNtrain <- conftab[1, 1]/(conftab[1, 1] + conftab[2, 1])
FPtrain <- conftab[2, 1]/(conftab[2, 1] + conftab[1, 1])
FNtrain <- conftab[1, 2]/(conftab[1, 1] + conftab[2, 1])
```

Using the TESTING data to create a model

Find summary statistics when the Kickstarter failed (1) or did not fail (0)

```
summaryBy(goal + pledged + backers + currency + DateDiffDays ~ binomState, data = KickTest, fun = c(mea
```

```
##   binomState goal.FUN1 pledged.FUN1 backers.FUN1 currency.FUN1
## 1          0  3694.141     20310.86     1190.185      2.727950
## 2          1  3690.896     27616.60     1867.456      2.225748
```

```
##   DateDiffDays.FUN1
## 1          35.29374
## 2          32.34069
```

It appears that Kickstarters that had less funding, less pledges, less currency, and less backers failed.

Creating a glm model on the training data and summary

```
glmmodTest <- glm(binomState ~ goal + pledged + backers + DateDiffDays + currency, data = KickTest, fami
summary(glmmodTest)
```

```
##
## Call:
## glm(formula = binomState ~ goal + pledged + backers + DateDiffDays +
##     currency, family = binomial, data = KickTest)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8771  -0.9867  -0.6474   1.1399   2.3367
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -6.264e-01  2.982e-02 -21.003  < 2e-16 ***
## goal         -3.597e-05  3.461e-06 -10.395  < 2e-16 ***
## pledged       1.765e-05  4.997e-07  35.322  < 2e-16 ***
## backers       4.556e-04  7.380e-06  61.730  < 2e-16 ***
## DateDiffDays -2.069e-02  6.758e-04 -30.619  < 2e-16 ***
## currencySGD   4.072e-01  4.890e-01   0.833  0.40501
## currencySEK  -3.167e-01  1.372e-01  -2.308  0.02100 *
## currencyNZD  -2.281e-01  1.388e-01  -1.643  0.10031
## currencyNOK  -1.058e+00  2.362e-01  -4.481 7.44e-06 ***
## currencyMXN   8.574e+00  4.395e+01   0.195  0.84533
## currencyHKD   4.725e-03  6.527e-01   0.007  0.99422
## currencyGBP  -1.551e-01  2.944e-02  -5.268 1.38e-07 ***
## currencyEUR  -6.978e-01  5.171e-02 -13.494  < 2e-16 ***
## currencyDKK  -1.926e-01  1.663e-01  -1.158  0.24670
## currencyCHF  -7.671e-01  2.604e-01  -2.947  0.00321 **
## currencyCAD  -4.441e-01  4.686e-02  -9.477  < 2e-16 ***
## currencyAUD  -6.523e-01  6.748e-02  -9.666  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 94715  on 70272  degrees of freedom
## Residual deviance: 86265  on 70256  degrees of freedom
## AIC: 86299
##
## Number of Fisher Scoring iterations: 7
```

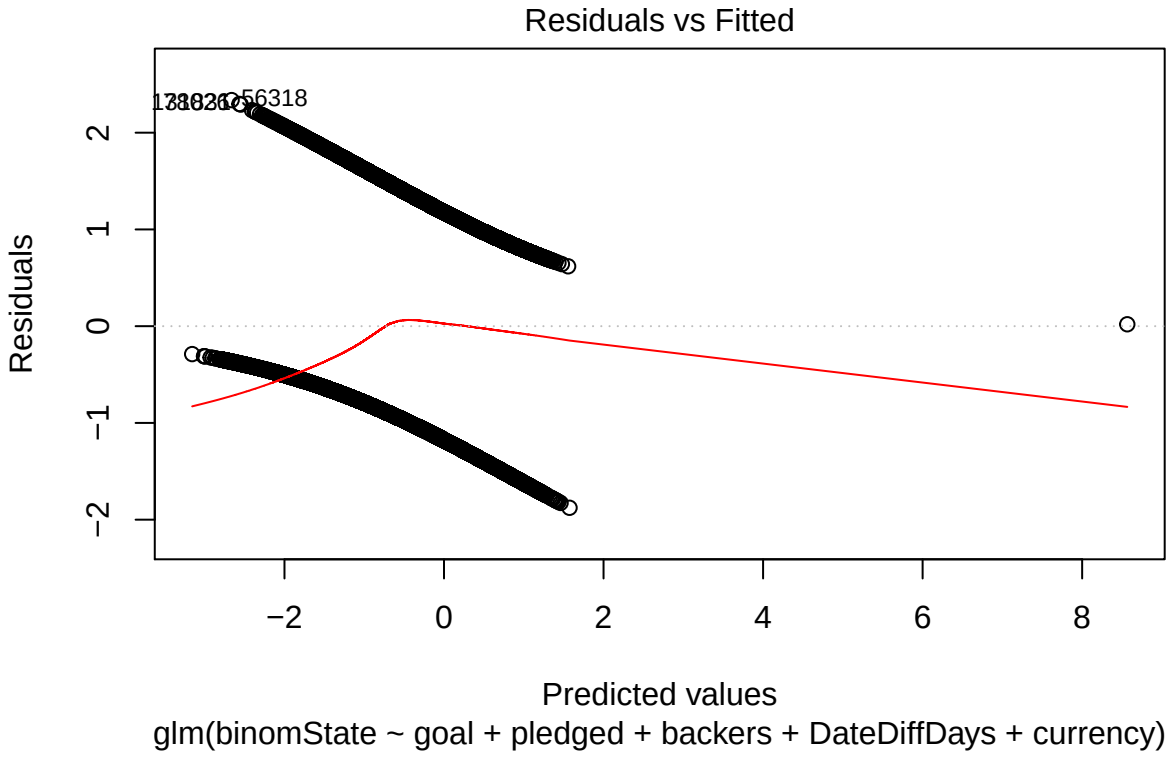
Find odds ratios

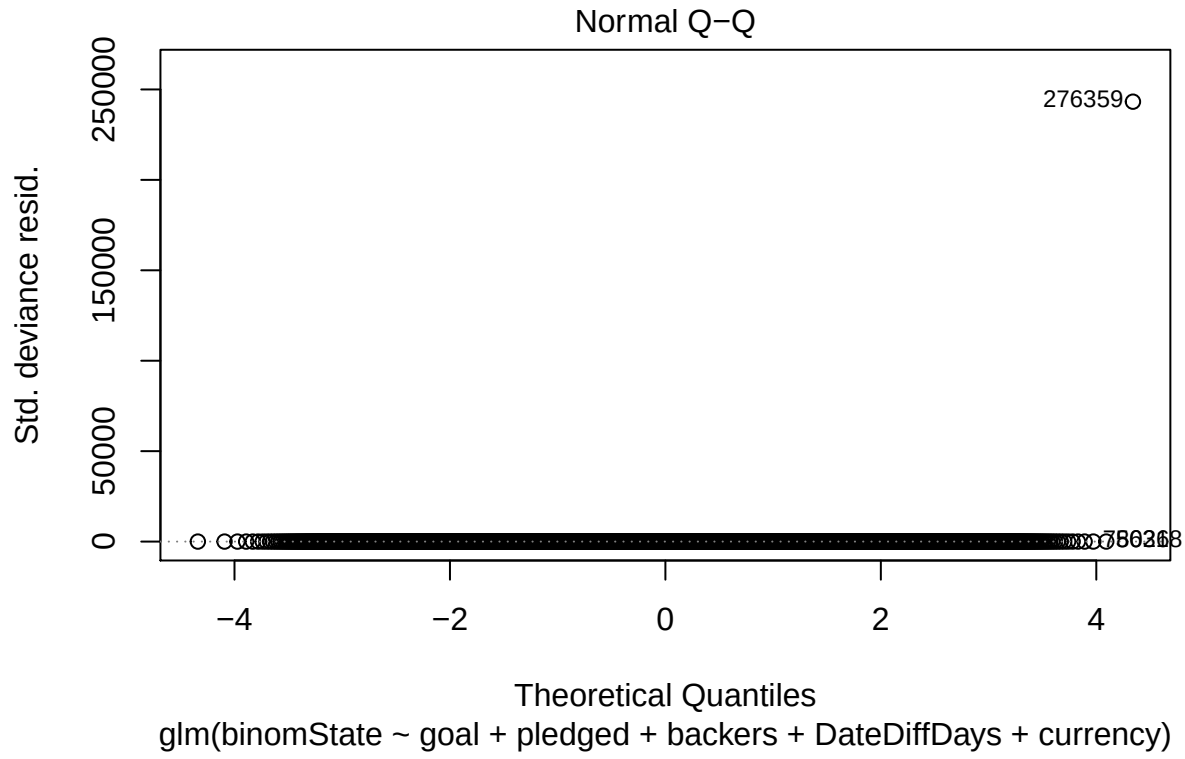
```
exp(glmmodTest$coefficients)
```

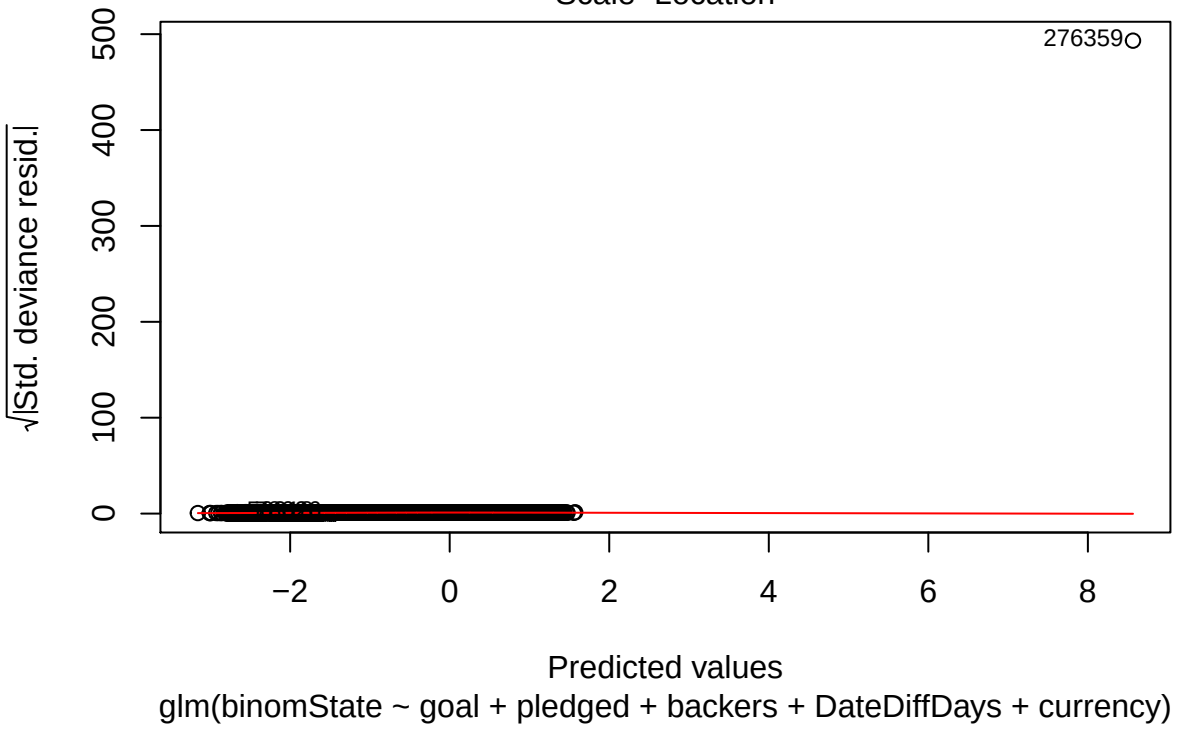
```
##  (Intercept)         goal      pledged      backers DateDiffDays
##    0.5345184    0.9999640    1.0000176    1.0004557    0.9795194
##  currencySGD  currencySEK  currencyNZD  currencyNOK  currencyMXN
##    1.5025825    0.7285280    0.7960272    0.3470471 5294.1754676
##  currencyHKD  currencyGBP  currencyEUR  currencyDKK  currencyCHF
##    1.0047366    0.8563176    0.4977022    0.8247978    0.4643376
##  currencyCAD  currencyAUD
##    0.6414080    0.5208611
```

Plotting the model

```
plot(glmmodTest)
```

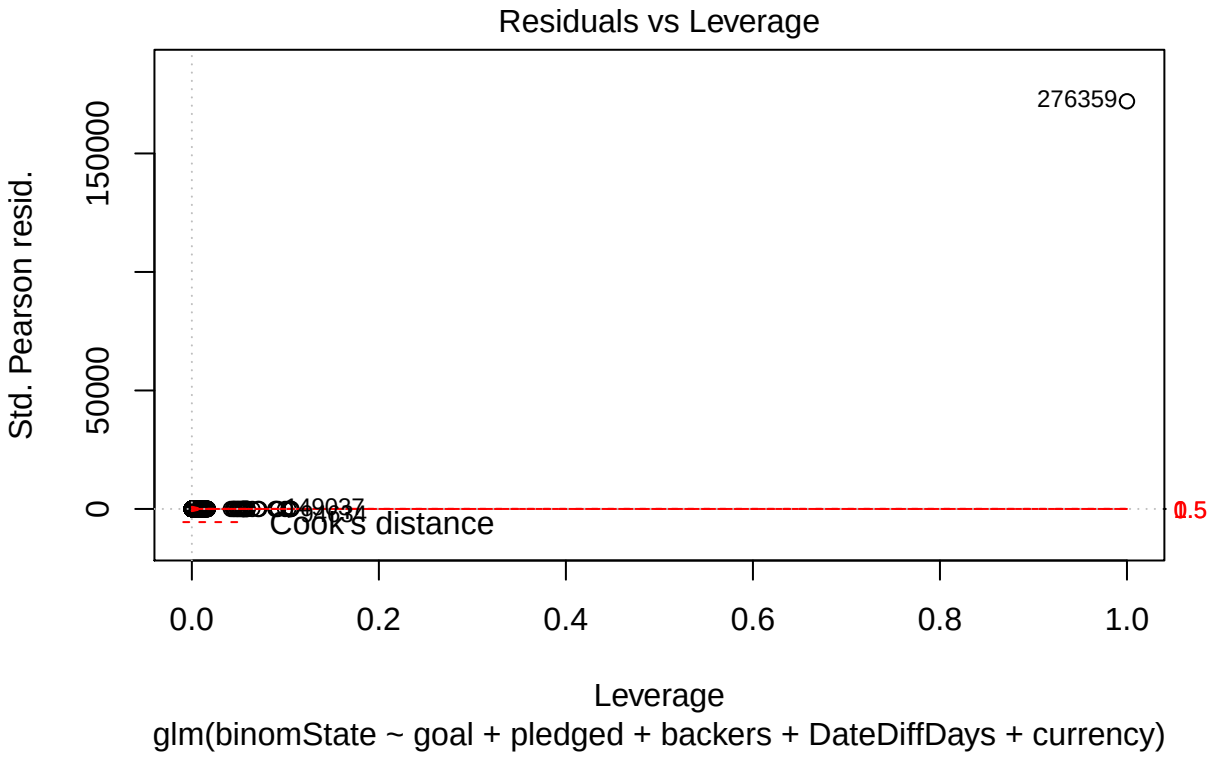






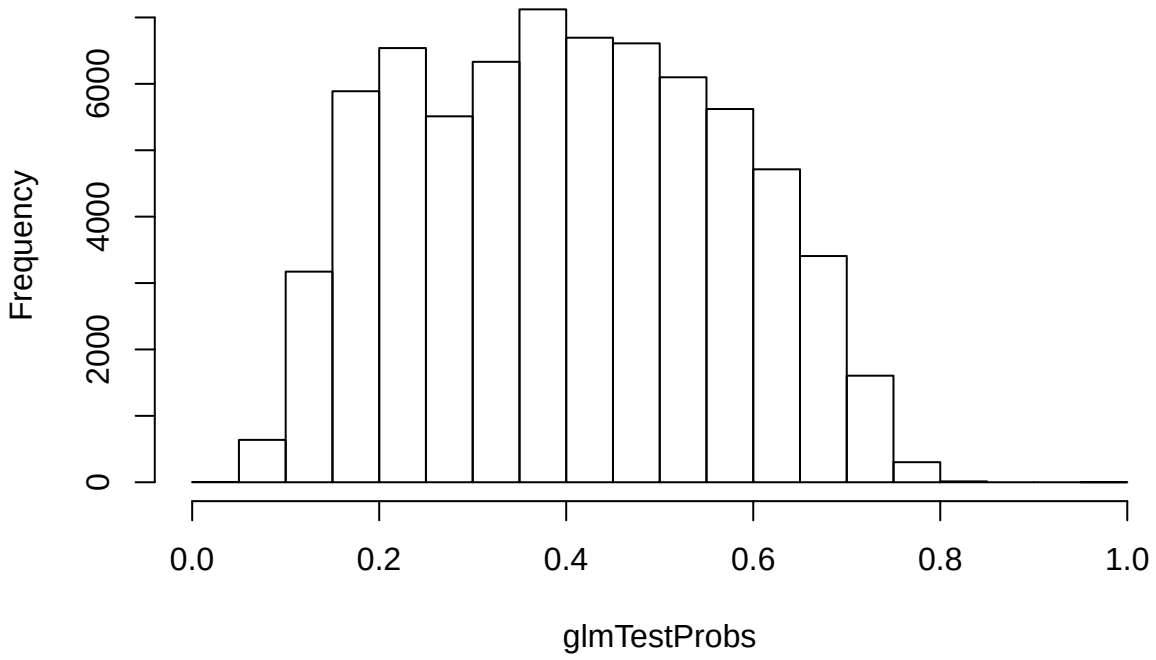

```
## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced

## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced
```



Creating probabilty of failure (1) or not (0)

```
glmTestProbs <- predict.glm(glmmodTest, type = "response")
hist(glmTestProbs)
```




```
glmTestPreds <- rep("Failed", nrow(KickTest))
glmTestPreds[glmTestProbs > 0.5] <- "Succeed"
conftabtest <- table(glmTestPreds, true = KickTest$binomState)
print(conftabtest)
```

```
##             true
## glmTestPreds     0     1
##      Failed  32373 16139
##      Succeed  9634 12127
```

```
TPtest <- conftabtest[2, 2]/(conftabtest[1, 2] + conftabtest[2, 2])
TNtest <- conftabtest[1, 1]/(conftabtest[1, 1] + conftabtest[2, 1])
FPtest <- conftabtest[2, 1]/(conftabtest[2, 1] + conftabtest[1, 1])
FNtest <- conftabtest[1, 2]/(conftabtest[1, 1] + conftabtest[2, 1])
#TRAINING RESULTS
print(TPtrain)
```

```
## [1] 0.4242474
```

```
print(TNtrain)
```

```
## [1] 0.7720525
```

```
print(FPtrain)
```

```
## [1] 0.2279475
```

```
print(FNtrain)
```

```
## [1] 0.3867479
```

```
#TESTING RESULTS
print(TPtest)
```

```
## [1] 0.4290313
```

```
print(TNtest)
```

```
## [1] 0.7706573
```

```
print(FPtest)
```

```
## [1] 0.2293427
```

```
print(FNtest)
```

```
## [1] 0.3841979
```

Test vs Training MSE

```
#Might not need to do this because "prediction from a rank-deficient fit may be misleadingprediction fr
predictKickTrainLogistic <- predict(glmmodTrain, newdata = KickTrain, type = "response")
KickTrainLogisticMSE = mean((KickTrain$binomState - predictKickTrainLogistic)^2)
predictKickTestLogistic <- predict(glmmodTest, newdata = KickTest, type = "response")
KickTestLogisticMSE = mean((KickTest$binomState - predictKickTestLogistic)^2)
print(KickTrainLogisticMSE)
```

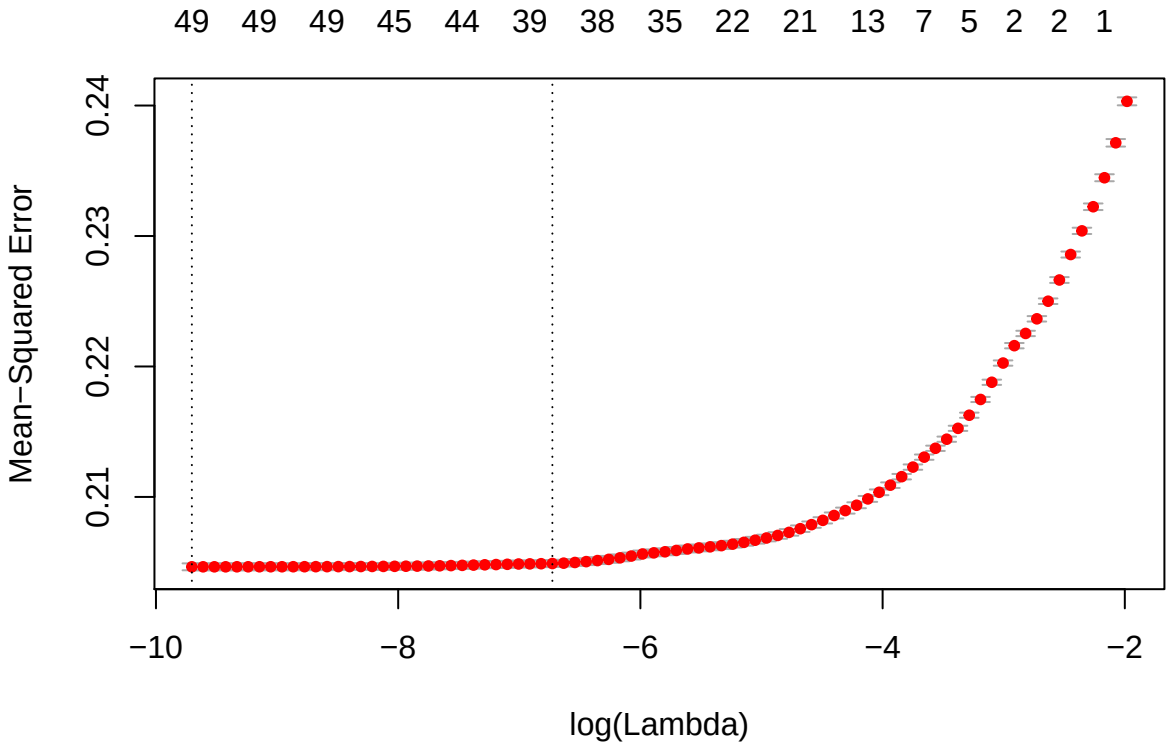
```
## [1] 0.2151755
```

```
print(KickTestLogisticMSE)
```

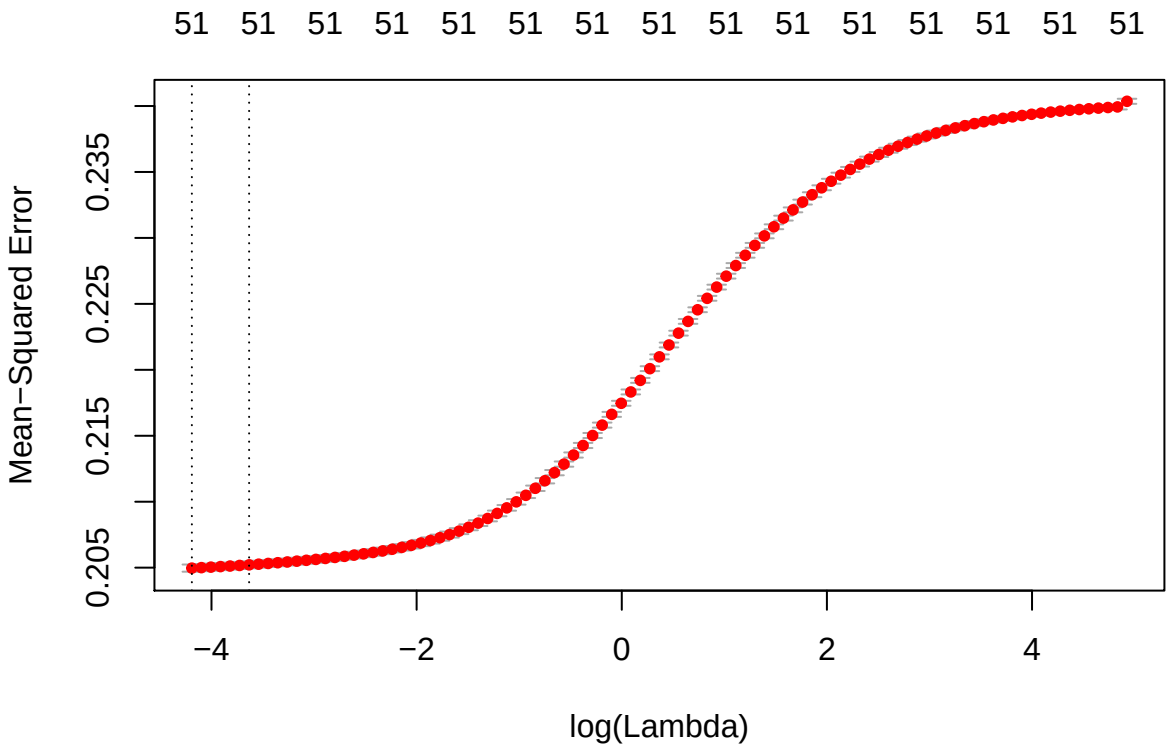
```
## [1] 0.2146105
```

Lasso, Ridge, Elastic Net Models

```
myFormula <- as.formula(binomState ~ main_category + currency + goal + pledged + backers + country + usd
Xvar <- build.x(myFormula, KickTrain)
Yvar <- build.y(myFormula, KickTrain)
LassoMod <- cv.glmnet(x = Xvar, y = Yvar, alpha = 1, nfolds = 10)
plot(LassoMod)
```

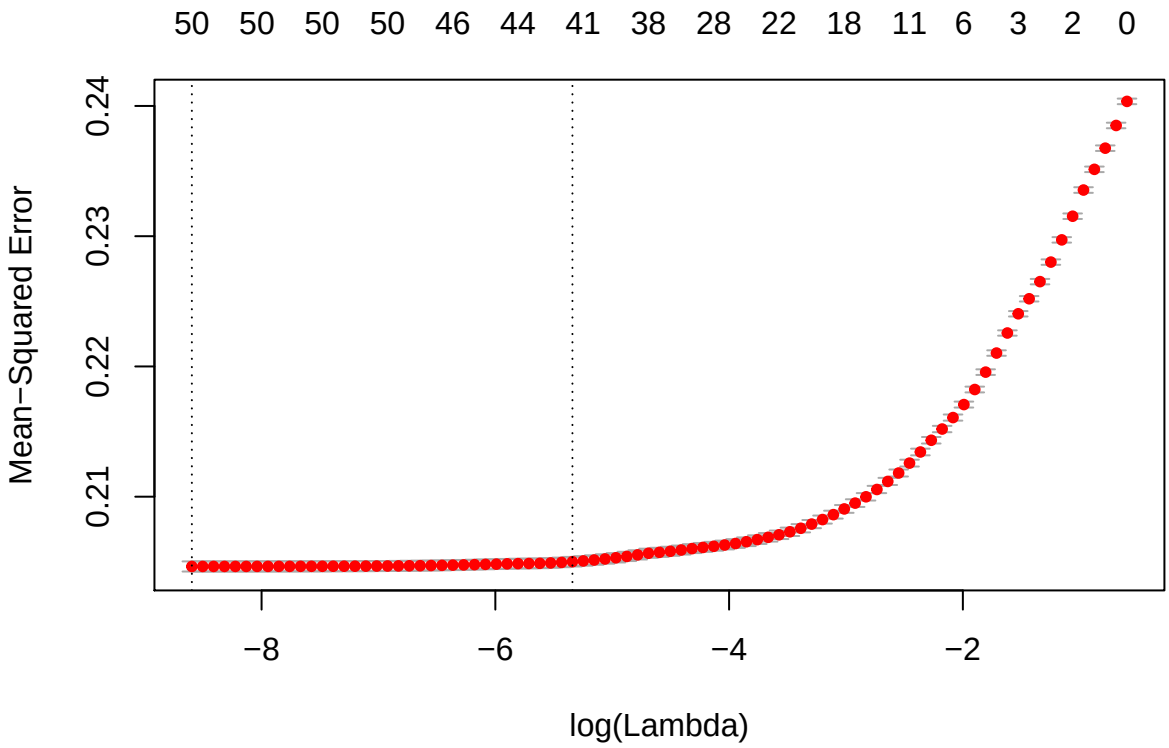



```
#Ridge Mod.
RidgeMod <- cv.glmnet(x = Xvar, y = Yvar, alpha = 0, nfolds = 10)
plot(RidgeMod)
```




```
#Elastic Net
ElasticNet1 <- cv.glmnet(x = Xvar, y = Yvar, alpha = .25, nfolds = 10)
ElasticNet2 <- cv.glmnet(x = Xvar, y = Yvar, alpha = .50, nfolds = 10)
```

```
ElasticNet3 <- cv.glmnet(x = Xvar, y = Yvar, alpha = .75, nfolds = 10)
plot(ElasticNet1)
```




```
#MSE
MSE <- function(true, preds) { mean((true - preds)^2) }
MSE(KickTrain$binomState, predict(LassoMod, newx = Xvar))
```

```
## [1] 0.2048256
```

```
MSE(KickTrain$binomState, predict(RidgeMod, newx = Xvar))
```

```
## [1] 0.2051462
```

```
MSE(KickTrain$binomState, predict(ElasticNet1, newx = Xvar))
```

```
## [1] 0.2049371
```

```
MSE(KickTrain$binomState, predict(ElasticNet2, newx = Xvar))
```

```
## [1] 0.2048069
```

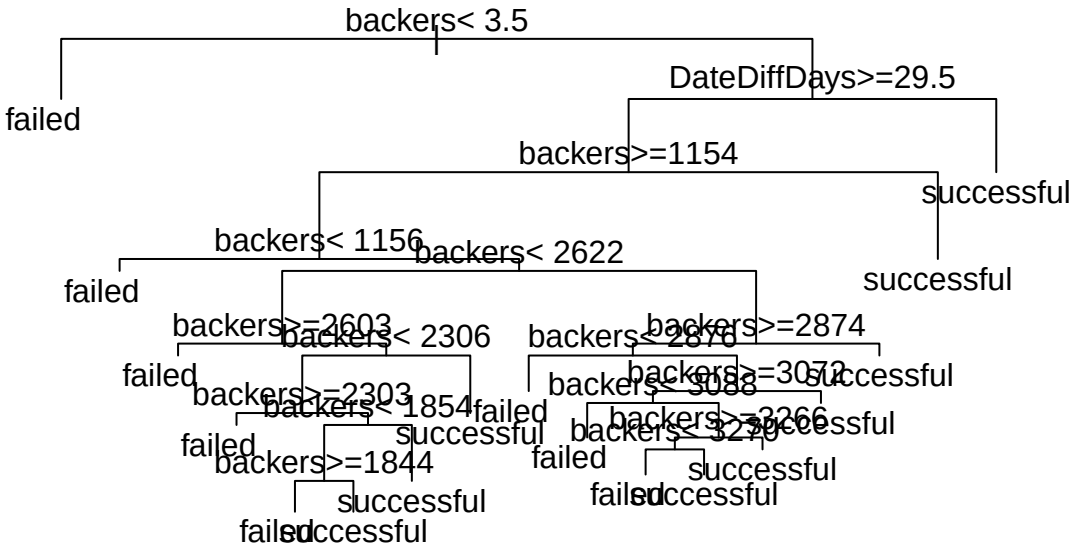
```
MSE(KickTrain$binomState, predict(ElasticNet3, newx = Xvar))
```

```
## [1] 0.2049251
```

Tree Based Methods

```
newKickStar <- subset(KickStar, select = -c(1,2,3,4,5,6,8,12,14,15))
trainSize <- .02
trainInd <- sample(1:nrow(newKickStar), size = floor(nrow(newKickStar) * trainSize))
newKickTrain <- newKickStar[trainInd, ]
newKickTest <- newKickStar[-trainInd, ]
```

```
#README!!! make sure you click the green arrow to run this properly
#method = class for classification problem
tree <- rpart(state ~ .-binomState, data = newKickTrain, method = "class", control = rpart.control(minsp
par(xpd=TRUE)
plot(tree); text(tree, pretty = 0)
```



Pruning Tree

```
# HAVENT TOUCHED THIS YET
# predict using the test data
tree.pred <- predict(tree, newKickTrain,type="class")
# note with() command evaluates an R expression in an environment
# constructed from the data,
with(newKickTrain,table(tree.pred,state))
```

```
##             state
## tree.pred    failed successful
##   failed       2167         62
##   successful   1187       2205
```

```
#install.packages('tree')
require('ISLR')
```

```
## Loading required package: ISLR
```

```
require('tree')
```

```
## Loading required package: tree
```

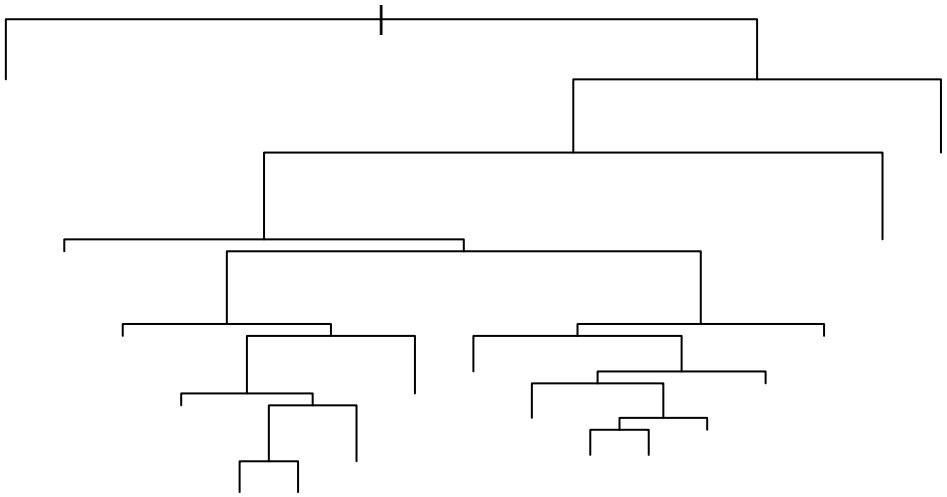
```
## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'tree'
```

```
# now do some cross-validating
printcp(tree)
```

```
##
## Classification tree:
```

```
## rpart(formula = state ~ . - binomState, data = newKickTrain,
##     method = "class", control = rpart.control(minsplit = 30,
##         cp = 0.01))
##
## Variables actually used in tree construction:
## [1] backers      DateDiffDays
##
## Root node error: 2267/5621 = 0.40331
##
## n= 5621
##
##         CP nsplit rel error  xerror     xstd
## 1 0.061903      0   1.00000 1.00000 0.016224
## 2 0.030657      3   0.81429 0.82047 0.015561
## 3 0.024261      5   0.75298 0.77944 0.015354
## 4 0.023599      7   0.70446 0.71725 0.014995
## 5 0.014998     10   0.63123 0.65417 0.014575
## 6 0.014557     12   0.60124 0.60432 0.014199
## 7 0.010587     14   0.57212 0.57389 0.013948
## 8 0.010000     16   0.55095 0.55668 0.013800
```

```
pruned.tree <- prune(tree, cp = 0.01)
preds <- predict(pruned.tree)
plot(pruned.tree)
```



```
help(rpart)
#cv.kick = cv.tree(tree,FUN=prune.misclass)
#cv.carseats
#plot(cv.carseats)
#prune.carseats=prune.misclass(tree.carseats,best=13)
#plot(prune.carseats);text(prune.carseats,pretty=0)
# now let's evaluate this pruned tree on the dataset
#tree.pred = predict(prune.carseats, Carseats[-train,],type="class")
#with(Carseats[-train,],table(tree.pred,High))
#(72+32)/150

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

WORK ON THIS LATER

```
#MSE <- function(truth, predict) {mean((truth - predict)^2)}
#predsTrain <- predict(tree, newdata = newKickTrain)
#predsValidate <- predict(tree, newdata = newKickTest)
#MSE(newKickStar$binomState, predsTrain)
#MSE(newKickStar$binomState, predsValidate)
```

Random Forest

WEBSITE USED: https://www.statmethods.net/advstats/cart.html

```
randTree <- randomForest(state ~ ., data=newKickTrain, mtry=3, method = "class")
#importance(randTree)
#plot(randTree)

randForPredictTrain <- predict(randTree, newdata = newKickTrain, mtry = 3)
#randForTrainMSE = mean((newKickTrain$state - randForPredictTrain)^2)
#randForPredictTest <- predict(randFor, newdata = newKickTest, mtry = 3)
#randForTestMSE = mean((AutoTest$mpg - randForPredictTest)^2)
#print(randForTrainMSE)
#print(randForTestMSE)
```