Perform the Logistic Regression analysis of the challenger data

Step 1: Collecting data

```
launch <- read.csv("http://www.sci.csueastbay.edu/~esuess/classes/Statistics_6620/Presentations/ml10/challenger.csv")

str(launch)
```

```
## 'data.frame':    23 obs. of  4 variables:
##  $ distress_ct         : int  0 1 0 0 0 0 0 0 1 1 ...
##  $ temperature         : int  66 70 69 68 67 72 73 70 57 63 ...
##  $ field_check_pressure: int  50 50 50 50 50 50 100 100 200 200 ...
##  $ flight_num          : int  1 2 3 4 5 6 7 8 9 10 ...
```

Step 2: Exploring and preparing the data

- Fix the default variable to be 0 or 1

```
launch$distress_ct = ifelse(launch$distress_ct<1,0,1)
launch$distress_ct
```

```
##  [1] 0 1 0 0 0 0 0 0 1 1 1 0 0 1 0 0 0 0 0 0 1 0 1
```

- Set up trainning and test data sets

```
indx = sample(1:nrow(launch), as.integer(0.9*nrow(launch)))
indx
```

```
##  [1] 17 16  1 18 23 10  8  6  9 21  2 13  7  3  4 22 19 11 12 20
```

```
launch_train = launch[indx,]
launch_test = launch[-indx,]

launch_train_labels = launch[indx,1]
launch_test_labels = launch[-indx,1]
```

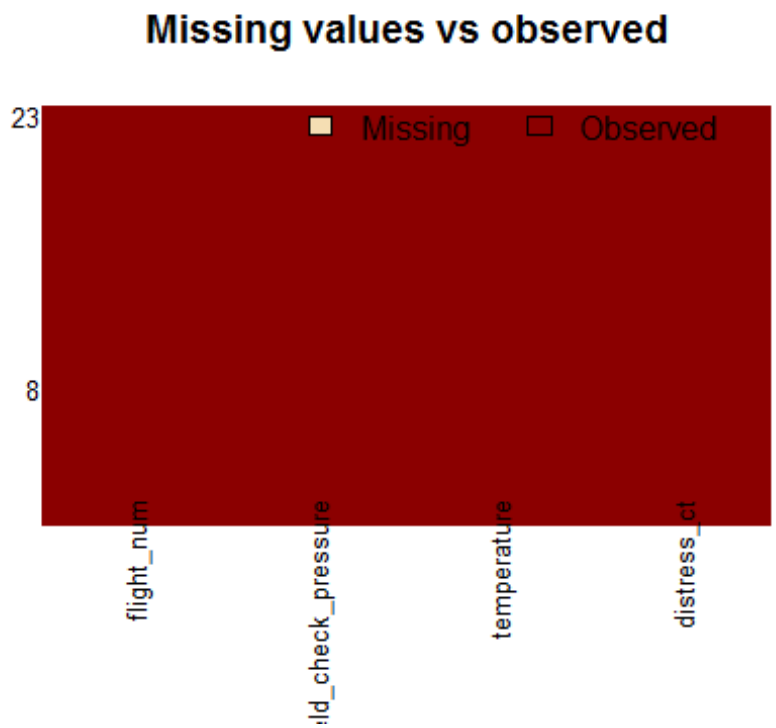
- Check if there are any missing values "Missing values vs observed"

```
library(Amelia)
```

```
## Loading required package: Rcpp
```

```
## ##
## ## Amelia II: Multiple Imputation
## ## (Version 1.7.4, built: 2015-12-05)
## ## Copyright (C) 2005-2017 James Honaker, Gary King and Matthew Blackwell
## ## Refer to http://gking.harvard.edu/amelia/ for more information
## ##
```

```
missmap(launch, main = "Missing values vs observed")
```



- Number of missing values in each column

```
sapply(launch,function(x) sum(is.na(x)))
```

```
##          distress_ct          temperature field_check_pressure
##                    0                    0                    0
##           flight_num
##                    0
```

- Number of unique values in each column

```
sapply(launch, function(x) length(unique(x)))
```

```
##          distress_ct          temperature field_check_pressure
##                    2                   16                    3
##           flight_num
##                   23
```

Step 3: Training a model on the data

- Fit the logistic regression model, with all predictor variables
- Look at the model that gives the smallest AIC

```
model <- glm(distress_ct ~.,family=binomial(link='logit'),data=launch_train)
model
```

```
##
## Call:  glm(formula = distress_ct ~ ., family = binomial(link = "logit"),
##     data = launch_train)
##
```

```
## Coefficients:
##          (Intercept)           temperature  field_check_pressure
##              14.47537              -0.24308               0.01258
##           flight_num
##              -0.02179
##
## Degrees of Freedom: 19 Total (i.e. Null);  16 Residual
## Null Deviance:       24.43
## Residual Deviance: 16.66     AIC: 24.66
```

```
summary(model)
```

```
##
## Call:
## glm(formula = distress_ct ~ ., family = binomial(link = "logit"),
##     data = launch_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3608  -0.6150  -0.4591   0.3454   2.0432
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          14.47537    9.66367   1.498   0.1342
## temperature          -0.24308    0.13846  -1.756   0.0791 .
## field_check_pressure  0.01258    0.01820   0.691   0.4895
## flight_num           -0.02179    0.19230  -0.113   0.9098
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 24.435  on 19  degrees of freedom
## Residual deviance: 16.658  on 16  degrees of freedom
## AIC: 24.658
##
## Number of Fisher Scoring iterations: 5
```

```
anova(model, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: distress_ct
##
## Terms added sequentially (first to last)
##
##
##                      Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                                     19     24.435
```

```
## temperature           1   6.2933        18     18.141  0.01212 *
## field_check_pressure  1   1.4711        17     16.670  0.22516
## flight_num            1   0.0126        16     16.658  0.91078
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Drop the insignificant predictors, alpha = 0.10

```
model <- glm(distress_ct ~ temperature,family=binomial(link='logit'),data=lau
nch_train)
model

##
## Call:  glm(formula = distress_ct ~ temperature, family = binomial(link = "
logit"),
##     data = launch_train)
##
## Coefficients:
## (Intercept)  temperature
##     15.6117      -0.2369
##
## Degrees of Freedom: 19 Total (i.e. Null);  18 Residual
## Null Deviance:       24.43
## Residual Deviance: 18.14     AIC: 22.14

summary(model)

##
## Call:
## glm(formula = distress_ct ~ temperature, family = binomial(link = "logit")
,
##     data = launch_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.1673  -0.8013  -0.4182   0.4929   2.1286
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  15.6117     8.2684   1.888   0.0590 .
## temperature  -0.2369     0.1196  -1.981   0.0476 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 24.435  on 19  degrees of freedom
## Residual deviance: 18.141  on 18  degrees of freedom
## AIC: 22.141
##
## Number of Fisher Scoring iterations: 5
```

```
anova(model, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: distress_ct
##
## Terms added sequentially (first to last)
##
##
##             Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                           19     24.435
## temperature  1   6.2933        18     18.141  0.01212 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Step 4: Evaluating model performance

- Check Accuracy which is 100%. I think the data is to small to be considered reliable.

```
fitted.results <- predict(model,newdata=launch_test,type='response')
fitted.results <- ifelse(fitted.results > 0.5,1,0)

misClasificError <- mean(fitted.results != launch_test$distress_ct)
print(paste('Accuracy',1-misClasificError))
```

```
## [1] "Accuracy 1"
```

Step 5: Improving model performance

- Using ROC curve to improve the model
- This classifier has an AUC of 1 which shows that the classifier has done a pretty good job at classifying.
- Because this data set is so small, it is possible that the test data set does not contain both 0 and 1 values. If this happens the code will not run. And since the test data set is so small the ROC is not useful here.

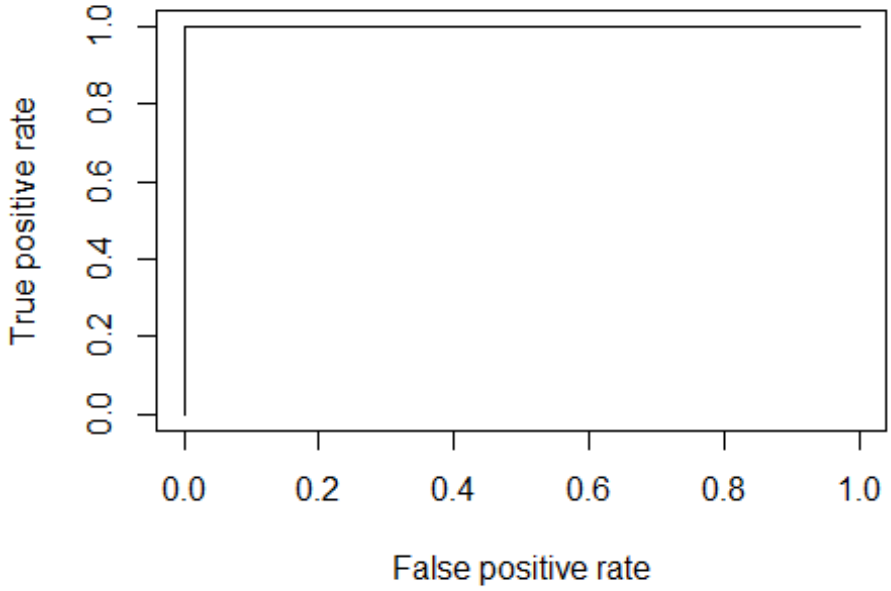
```
library(ROCR)
```

```
## Loading required package: gplots
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```
p <- predict(model, newdata=launch_test, type="response")
pr <- prediction(p, launch_test$distress_ct)
prf <- performance(pr, measure = "tpr", x.measure = "fpr")
plot(prf)
```




```
auc <- performance(pr, measure = "auc")
auc <- auc@y.values[[1]]
auc

## [1] 1
```