NYCFlights: Arrival Delay Logictic Model

Saqib Ali

May 1st, 2017

Logsitic and Inverse Logistic Transformation

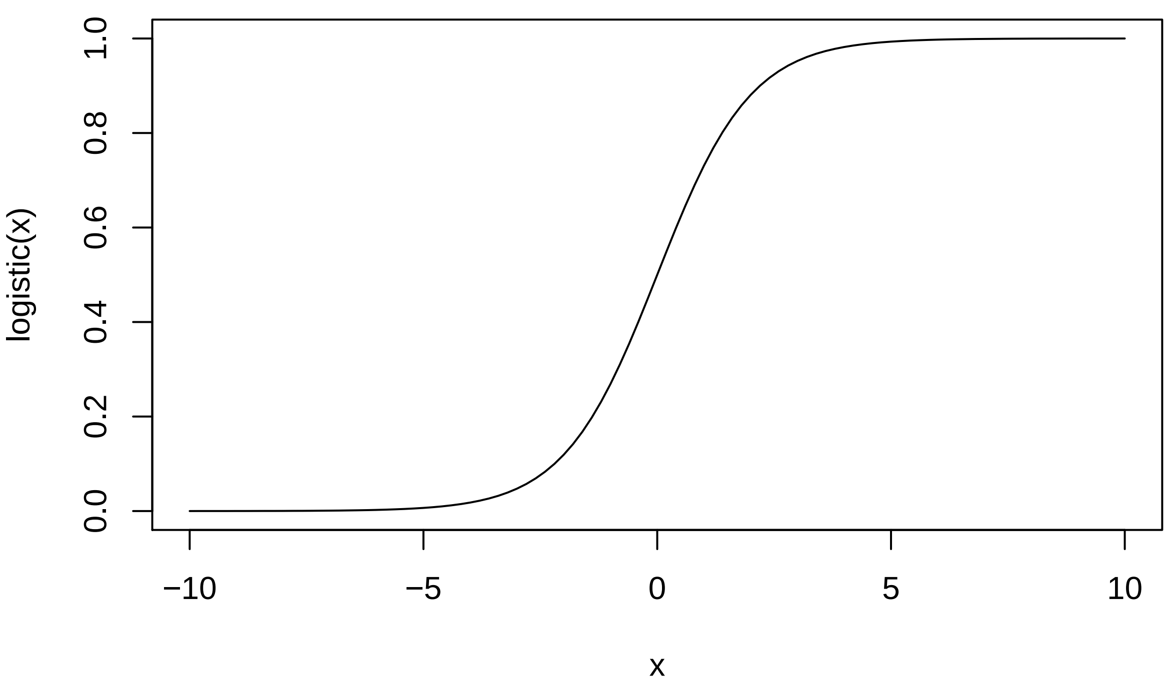
- Write an R function for the logistic function. The function should accept a `numeric` vector with values `[-Inf,Inf]` and produce a numeric vector in the the range `[0,1]`.
- Plot the logistic function from `[-10,10]`
- Write a R function for the inverse logistic function. The function should accept a `numeric` vector with values `[0,1]` and prodcuce a numeric vector in the range `[-Inf,Inf]`
- Plot the Inverse Logistic function from `[0,1]`

Hint: For plotting curves see `?graphics::curve` or `?ggplot2::stat_function`

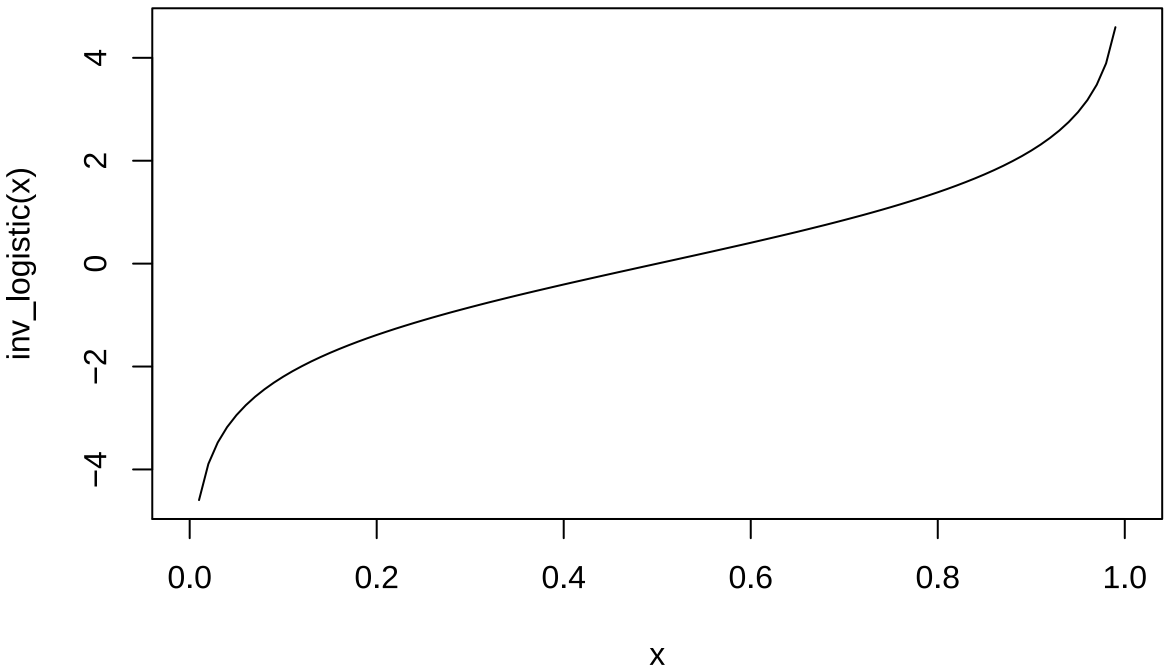
```
logistic <- function(x) {
  1/(1+exp(-x))
}

inv_logistic <- function(x) {
  -log((1-x)/x)
}

x <- -10:10
curve(logistic, -10, 10)
```




```
curve(inv_logistic, 0, 1)
```



NYCFlights Model

Using the rectangular data that you created from the earlier assignment and following theexample from the text and class, create a model for arr_delay >= 22 minutes. Describe/Explain each of the steps and show all work.

KNIT YOUR DOCUMENT AS *HTML* AND SUBMIT IT AND THE `Rmd` file to your repository.

Join the datasets and analyze the structure

```
YX <- flightsDT
YX %<>% merge( planesDT, all.x = TRUE, by='tailnum', suffixes=c('','.pl') )
YX %<>% merge( weatherDT, all.x = TRUE, by=c('origin','year','month','day','hour'), suffixes=c('','.we'
YX %<>% merge( airportsDT, all.x = TRUE, by.x='origin', by.y='faa', suffixes=c('','.orig') )
YX %<>% merge( airportsDT, all.x = TRUE, by.x='dest', by.y='faa', suffixes=c('','.dest') )
data.joined <- YX
```

Add a categorical variable for arr_delay >= 22 minutes. It is called arrival_delayed

```
data.joined$arrival_delayed <- ifelse(data.joined$arr_delay >= 22, 1,0)
```

Remove entries with NA values for Independant variables that we want to use in the model

```
data.joined <- data.joined  %>%
  filter(!is.na(dep_delay))   %>%
  filter(!is.na(dest))   %>%
  filter(!is.na(origin))   %>%
  filter(!is.na(year))   %>%
  filter(!is.na(month))   %>%
  filter(!is.na(day))   %>%
  filter(!is.na(hour))   %>%
  filter(!is.na(tailnum))   %>%
  filter(!is.na(sched_dep_time))   %>%
  filter(!is.na(sched_arr_time))   %>%
  filter(!is.na(flight))   %>%
  filter(!is.na(distance))   %>%
  filter(!is.na(year.pl))   %>%
  filter(!is.na(minute))   %>%
  filter(!is.na(year.pl))   %>%
  filter(!is.na(type))   %>%
  filter(!is.na(manufacturer))   %>%
  filter(!is.na(model))   %>%
  filter(!is.na(engines))   %>%
  filter(!is.na(seats))   %>%
  filter(!is.na(engine))   %>%
  filter(!is.na(temp))   %>%
  filter(!is.na(dewp))   %>%
  filter(!is.na(humid))   %>%
  filter(!is.na(wind_dir))   %>%
  filter(!is.na(wind_speed))   %>%
  filter(!is.na(wind_gust))   %>%
  filter(!is.na(precip))   %>%
  filter(!is.na(pressure))   %>%
  filter(!is.na(visib))   %>%
  filter(!is.na(name))   %>%
  filter(!is.na(lat))   %>%
  filter(!is.na(lon))   %>%
  filter(!is.na(tz))   %>%
  filter(!is.na(name.dest))   %>%
  filter(!is.na(lat.dest))   %>%
  filter(!is.na(lon.dest))   %>%
  filter(!is.na(alt.dest))   %>%
  filter(!is.na(tz.dest))
```

Create Training and Test Samples

```
data.joined.training <- sample_frac(data.joined, .75)
data.joined.testing <- sample_frac(data.joined, .5)
```

Generate a Logistic Model

```
logit.fit <- glm(arrival_delayed ~  dep_delay + dest + origin + year + month + day + hour + sched_dep_ti
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
#summary(logit.fit)
```

Predict

```
prob = plogis(predict(logit.fit, data.joined.testing))
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading
```

Question:

Is this a good model? (Write your answer here.)

To answer this let's first calculate the misclassifications, Specificity and Sensitivity of the model.

Model Evaluation

Decide on optimal prediction probability cutoff for the model

Using the InformationValue::optimalCutoff we determine the optimal cut-off

```
optimalCutoff(data.joined.testing$arrival_delayed, prob)[1]
## [1] 0.51
optCutOff <- optimalCutoff(data.joined.testing$arrival_delayed, prob)[1]
```

Mis-classification rate

```
misClassError(data.joined.testing$arrival_delayed, prob, threshold = optCutOff)
## [1] 0.0724
```

Sensitivity

```
sensitivity(data.joined.testing$arrival_delayed, prob, threshold = optCutOff)
## [1] 0.6919365
```

Specificity

```
specificity(data.joined.testing$arrival_delayed, prob, threshold = optCutOff)
## [1] 0.9806628
```

Confusion Matrix

```
confusionMatrix(data.joined.testing$arrival_delayed, prob, threshold = optCutOff)
```

```
##       0     1
## 0 93415  6640
## 1  1842 14914
```

Conclusion

Based on the Confusion Matrix results, high Specificity and high Sensitivity and very low mis-classification errors this is a fairly good model, with high accuracy, sensitivity and specivity.

PART B:

Your model should be good at explaining tardiness. Now, assume that your job is to predict arrival delays a month in advance. You can no longer use all the features in your model. Retrain your model using only features that will be *known* only a month in advance of the departure time. Show all steps as above.

```
logit.fit <- glm(arrival_delayed ~  dest + origin + sched_dep_time + sched_arr_time + carrier + distanc
```

Model Evaluation

Let's see how good this model is using a Confusion Matrix.

Decide on optimal prediction probability cutoff for the model

Using the InformationValue::optimalCutoff we determine the optimal cut-off

```
prob = plogis(predict(logit.fit, data.joined.testing))
```

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

```
optimalCutoff(data.joined.testing$arrival_delayed, prob)[1]
```

```
## [1] 0.4935788
```

```
optCutOff <- optimalCutoff(data.joined.testing$arrival_delayed, prob)[1]
```

Mis-classification rate

```
misClassError(data.joined.testing$arrival_delayed, prob, threshold = optCutOff)
```

```
## [1] 0.1838
```

Sensitivity

Note that the **Sensitivity** is very low

```
sensitivity(data.joined.testing$arrival_delayed, prob, threshold = optCutOff)
```

```
## [1] 0.001716619
```

Specificity

```
specificity(data.joined.testing$arrival_delayed, prob, threshold = optCutOff)
```

```
## [1] 0.9996746
```

Confusion Matrix

```
confusionMatrix(data.joined.testing$arrival_delayed, prob, threshold = optCutOff)
```

```
##       0     1
## 0 95226 21517
## 1    31    37
```

Conclusion

Based on the Confusion Matrix results, and a very low Sensitivity, we can conclude that this is **NOT** a Model Fit.

```
#prob = predict(logit.fit, data.joined.sample, type="response")
#prob = plogis(predict(logit.fit, data.joined.testing))
#prob<-ifelse(prob> 0.5,1,0)
#data.joined.testing$prob = prob
```