Predicting Rainfall at Seattle Tacoma Airport - Logistic Regression

Priyaranjan Pattnayak

December 21, 2017

Logistic Regression to predict rain probability in SEATAC International Airport

Pre-requisites

Previous knowledge of logistic regression is not required. However, basic working knowledge of R commands, linear models and basic data analytics/statistics concepts are required.

We will only use ROCR, XLConnect, tidyverse, dplyr, plyr, Amelia, ModelMetrics and caTools.

What is Logistic Regression?

Logistic Regression is a classification algorithm. It is used to predict a categorical outcome (1 / 0, Yes / No, True / False) given a set of independent variables. To represent binary / categorical outcome, we use dummy variables. It can also be considered as a special case of linear regression when the outcome variable is categorical, where we are using log of odds as dependent variable. In simple words, it predicts the probability of occurrence of an event by fitting data to a logit function.

A classical example used in machine learning is email classification: given a set of attributes for each email such as number of words, links and pictures, the algorithm should decide whether the email is spam (1) or not (0).

Reason why logistic Regression is preferred over linear regression, when decision variable is categorical is[1]:

1. If we use linear regression, the predicted values will become greater than one and less than zero if we move far enough on the X-axis. Such values are theoretically inadmissible.
2. One of the assumptions of regression is that the variance of Y is constant across values of X (homoscedasticity). This cannot be the case with a binary variable, because the variance is PQ. When 50 percent of the people are 1s, then the variance is .25, its maximum value. As we move to more extreme values, the variance decreases. When P=.10, the variance is .1*.9 = .09, so as P approaches 1 or zero, the variance approaches zero.
3. The significance testing of the b weights rest upon the assumption that errors of prediction (Y-Y') are normally distributed. Because Y only takes the values 0 and 1, this assumption is pretty hard to justify, even approximately. Therefore, the tests of the regression weights are suspect if you use linear regression with a binary DV.

Scope of this Project

This project will provide a step-by-step guide for building a logistic regression model using R. Here, we will be using a **‘binomial logistic regression’**, as the decision variable can have only two values. However, we can also predict a decision variable with more than three vallues, using logistic regression. This type of regression is called **‘Multinomial Logistic Regression’**. In this project, we predict the probablity of rainfall at SEATAC airport in Seattle.

Data Source

National Climatic Data Center Link (https://www.ncdc.noaa.gov/cdo-web/datasets/GHCND/stations/GHCND:USW00024233/detail)

Read the data from the csv file (Remember to replace the filepath)

```
rain_data <- read.csv("G:\\RProject\\SeattleRain Logistic Regression\\RainSeattle2016.csv", header=T,na.strings = c(""))
```

The Dataset has below fields:

Rain is the column which we are trying to predict.

NAME is the place where the weather data was recorded on respective DATE.

PRCP and SNOW = Precipitation and Snow in inches

SNWD = Snow depth

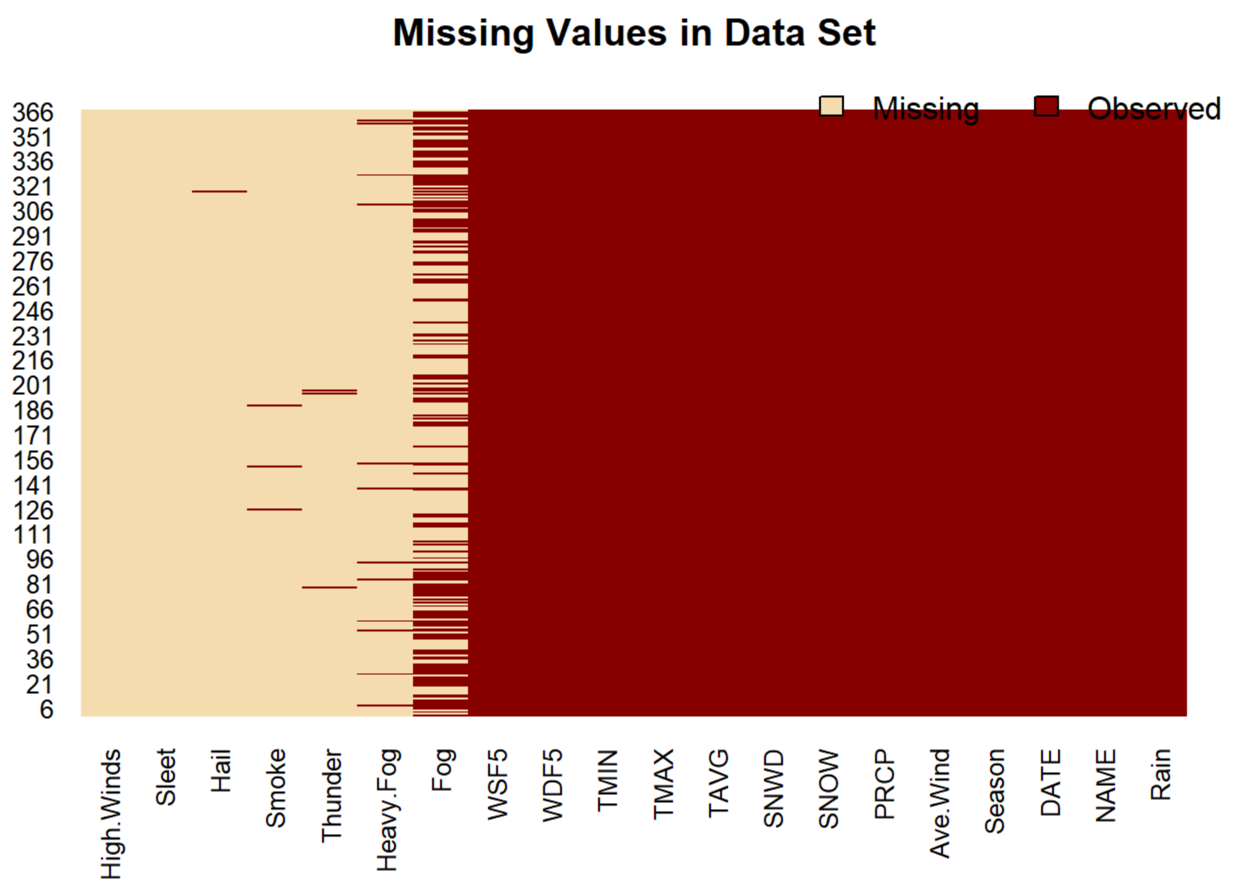
TAVG = Average Temperature in F

TMAX, TMIN = max and Min Temperature of the day in F.

WDF5, WSF5 - Direction of fastest 5-second wind in degree and Fastest 5-second wind speed respectively.

Next, we check for missing values. Amelia library gives us a missmap function that shows the missing details in a visual map.

```
missmap(rain_data,main="Missing Values in Data Set")
```



We have lots of missing values in Fog column. And high winds, sleet, hail, smoke, thunder and heavy fog are unuseable as they have almost all missing values. Therefore, we can exclude them from our model.

Name, Days, DATE, PRCP and SNWD can also be ignored. Only include the columns that we would be using for building a model.

In our case, we only use Rain, Season, Ave.wind, PRCP, TAVG, TMAX, TMIN, WDF5 and WSF5

```
rain_df <- subset(rain_data,select=c(1,3,4,5,7,8,9,10,11,12,13))
```

Sneak-peak into the data:

```
head(rain_df)
```

```
##   Rain     DATE Season Ave.Wind SNOW SNWD TAVG TMAX TMIN WDF5 WSF5
## 1    0 1/1/2016 Winter     4.92    0    0   36   46   28   70 13.0
## 2    0 1/2/2016 Winter     5.59    0    0   33   42   25  110 29.1
## 3    1 1/3/2016 Winter     9.17    0    0   37   40   31  110 36.9
## 4    1 1/4/2016 Winter     7.61    0    0   36   38   35   10 17.0
## 5    1 1/5/2016 Winter     5.37    0    0   38   46   36   80 18.1
## 6    0 1/6/2016 Winter     6.49    0    0   44   53   37  100 16.1
```

Before we proceed, let’s see if our subset has any NA values. In case missing values are observed, we need to account for them. One of the most common ways to exclude the rows containing NA values or fit missing values by inserting the median value of that column. One can also plot box-plots to find appropriate value for the missing value. Or use Knn/mice/Amelia imputation.

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

```
##     Rain     DATE   Season Ave.Wind     SNOW     SNWD     TAVG     TMAX 
##        0        0        0        0        0        0        0        0 
##     TMIN     WDF5     WSF5 
##        0        0        0
```

We **don't have missing** values.

Now that we have the data in place, we need to modify the few columns from continuous type to categorical type. In many data analysis settings, it might be useful to break up a continuous variable such as age into a categorical variable. Or, one might want to classify a categorical variable like year into a larger bin, such as 1990-2000

Here, we categorize two columns in the dataset. First, Rain is converted to a factor with 2 levels i.e 0 and 1. 0 indicates no rain, whereas 1 indicates rainfall on that day.

```
rain_df$Rain <- factor(rain_df$Rain)
```

Next, we create an unordered categorical variable for Season.

```
rain_df$Season <- factor(rain_df$Season,ordered = FALSE)
```

Season is now a factorial variable with 4 levels i.e 'Fall', 'Soring', 'Summer', 'Winter'. Ordered = FALSE indicates R that the categories are not ordinal.

We need to convert DATE from character type to date type. We do that by using the as.Date() function.

```
rain_df$DATE <- as.Date(rain_df$DATE,'%m/%d/%Y')
```

Before we proceed, let's see how R is going to deal with the categorical variables. We can use the contrasts() function for this. This function shows us how the variables have been dummyfied by R and how to interpret them in a model. It shows us the reference values for each factorial variable.

```
contrasts(rain_df$Rain)
```

```
##   1
## 0 0
## 1 1
```

```
contrasts(rain_df$Season)
```

```
##        Spring Summer Winter
## Fall        0      0      0
## Spring      1      0      0
## Summer      0      1      0
## Winter      0      0      1
```

In Season category, class ‘Fall’ is used as reference.

The Generalized Linear Models in R internally encode categorical variables into n - 1 distinct levels. Thus, glm() in R will use one category of Season and use it as reference against remaining three categories in the model. Model statistics will indicate the significance of each of those three levels. Later, we will check if the categorical variable Season is statistically significant as a whole.

We split the data into training and test data set. Training set will be used to build the model And test data can be used to tes the model fit. One can also use n-fold cross validation for better results.

We will split 80% of data into training and 20% into test set.

```
set.seed(88)
split <- sample.split(rain_df$Rain,SplitRatio = 0.8)
```

Build the training and test chunks.

```
train_df <- subset(rain_df,split==TRUE)
test_df <- subset(rain_df,split==FALSE)
```

Let’s fit the model to our data. The code below estimates a logistic regression model using the glm (generalized linear model) function. Be sure to specify the parameter family=binomial in the glm() function.

```
glm1 <- glm(Rain ~ Season+Ave.Wind+SNOW+TAVG+TMAX+TMIN+WDF5+WSF5,family = binomial(link
= "logit"),data = train_df)
```

Model Statistics

```
summary(glm1)
```

```
##
## Call:
## glm(formula = Rain ~ Season + Ave.Wind + SNOW + TAVG + TMAX +
##     TMIN + WDF5 + WSF5, family = binomial(link = "logit"), data = train_df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9478  -0.5731  -0.1530   0.5889   2.5485
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -2.336e+00  1.487e+00  -1.570  0.11630
## SeasonSpring -3.826e-01  4.836e-01  -0.791  0.42889
## SeasonSummer -9.289e-01  5.964e-01  -1.558  0.11930
## SeasonWinter  8.826e-01  4.909e-01   1.798  0.07219 .
## Ave.Wind     -1.676e-01  1.063e-01  -1.576  0.11500
## SNOW          2.885e+01  2.056e+03   0.014  0.98880
## TAVG          5.374e-03  9.740e-02   0.055  0.95600
## TMAX         -2.902e-01  5.772e-02  -5.027 4.98e-07 ***
## TMIN          3.806e-01  7.992e-02   4.762 1.91e-06 ***
## WDF5         -1.013e-03  1.903e-03  -0.532  0.59447
## WSF5          1.567e-01  4.768e-02   3.287  0.00101 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 405.20  on 292  degrees of freedom
## Residual deviance: 235.56  on 282  degrees of freedom
## AIC: 257.56
##
## Number of Fisher Scoring iterations: 14
```

Interpreting the results of our regression model

The glm function internally encodes categorical variables into n - 1 distinct levels, as mentioned earlier. Here, the regression coefficients explain the change in log(odds) of the response variable for one unit change in the predictor variable. Std. Error represents the standard error associated with the regression coefficients. z value(sometimes called a Wald z-statistic) is analogous to t-statistics in multiple regression output. p value has the same interpretation as that in linear regression. With 95% confidence level, a variable having $p < 0.05$ is considered an important predictor.

In the above model(glm1), SNOW is least significant. And TMAX is the most significant. This means, with all other variables remaining constant, SNOW has the least effect on rain probablity and TMAX has highest effect. For every one unit change in TMAX, log odds of rain reduces by 2.902e-01 times.

Below the table of coefficients are fit indices, including the null and deviance residuals and the AIC. AIC and the residuals are crucial in determining model fit. Model with lower AIC is almost always preferred.

Next step is to analyze the deviance table. We do that by doing a ANOVA Chi-square test to check the overall effect of variables on the dependent variable.

```
anova(glm1,test='Chisq')
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: Rain
##
## Terms added sequentially (first to last)
##
##
##          Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                       292     405.20
## Season    3   53.975       289     351.22 1.136e-11 ***
## Ave.Wind  1   19.569       288     331.65 9.702e-06 ***
## SNOW      1    0.620       287     331.03 0.4310916
## TAVG      1    1.021       286     330.01 0.3123916
## TMAX      1   57.057       285     272.96 4.234e-14 ***
## TMIN      1   24.779       284     248.18 6.429e-07 ***
## WDF5      1    0.013       283     248.16 0.9095634
## WSF5      1   12.600       282     235.56 0.0003857 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Null Deviance - Residual Deviance shows our model’s performance as compared to a model with no variables(only the intercept) a.k.a the null model. The wider the gap, the better is the model. In addition, lower null and residual deviance are preferred, whereever applicable. Every time a variable is added, deviance drops. We need to identify those variable that do not cause large drop in deviance. A large p-value here indicates that the model without the variable explains more or less the same amount of variation.

It’s interesting to notice that Season as a whole is highly significant, but at categorical level, Season(or rather Spring/Summer/Winter) are not significant, not at least in the first model. Here, variables WDF5, SNOW and TAVG are not significant. We can remove them sequentially and rebuild glm models and repeat the whole process until all variables are significant. AIC for glm1 was 257.56

After sequentially eliminating SNOW followed by WDF5 and TAVG, we come accross the below model.

```
glm2 <- glm(Rain ~ Season+Ave.Wind+TMAX+TMIN+WSF5,family = binomial(link = "logit"),data
= train_df)
summary(glm2)
```

```
##
## Call:
## glm(formula = Rain ~ Season + Ave.Wind + TMAX + TMIN + WSF5,
##     family = binomial(link = "logit"), data = train_df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9304  -0.5734  -0.1484   0.5987   2.5814
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -2.21596    1.46563  -1.512  0.13055
## SeasonSpring -0.42325    0.48146  -0.879  0.37934
## SeasonSummer -0.92002    0.59585  -1.544  0.12257
## SeasonWinter  0.82840    0.48800   1.698  0.08959 .
## Ave.Wind     -0.14870    0.10445  -1.424  0.15453
## TMAX         -0.28778    0.04120  -6.985 2.85e-12 ***
## TMIN          0.37669    0.05599   6.728 1.72e-11 ***
## WSF5          0.15205    0.04722   3.220  0.00128 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 405.20  on 292  degrees of freedom
## Residual deviance: 236.95  on 285  degrees of freedom
## AIC: 252.95
##
## Number of Fisher Scoring iterations: 5
```

Note that, the Ave.Wind variable is being considered as non-significant. Let’s see the ANOVA deviance table.

```
anova(glm2, test='Chisq')
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: Rain
##
## Terms added sequentially (first to last)
##
##
##          Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                       292     405.20
## Season    3   53.975       289     351.22 1.136e-11 ***
## Ave.Wind  1   19.569       288     331.65 9.702e-06 ***
## TMAX      1   16.737       287     314.92 4.294e-05 ***
## TMIN      1   65.914       286     249.00 4.709e-16 ***
## WSF5      1   12.056       285     236.95 0.0005162 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

ANOVA instead tells us that Ave.Wind is significant. It is important to note how anova performs the test. Sequentially compares the smaller model with the next more complex model by adding one variable in each step. Each of those comparisons is done via a likelihood ratio test.

We need to build two models, one with Ave.wind and another without Ave.WInd and then compare the two models and pick the better among them. Note the AIC of glm2 = 252.95 Remove Ave.Wind and build a model

```
glm3 <- glm(Rain ~ Season+TMAX+TMIN+WSF5,family = binomial(link = "logit"),data = train_
df)
summary(glm3)
```

```
## 
## Call:
## glm(formula = Rain ~ Season + TMAX + TMIN + WSF5, family = binomial(link = "logit"), 
##     data = train_df)
## 
## Deviance Residuals: 
##     Min       1Q   Median       3Q      Max  
## -2.8205  -0.5872  -0.1463   0.6026   2.6164  
## 
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)    
## (Intercept)  -2.35037    1.44588  -1.626 0.104043    
## SeasonSpring -0.59325    0.46777  -1.268 0.204705    
## SeasonSummer -1.06437    0.58534  -1.818 0.069006 .  
## SeasonWinter  0.89304    0.48347   1.847 0.064724 .  
## TMAX         -0.27727    0.03988  -6.953 3.57e-12 ***
## TMIN          0.36667    0.05496   6.671 2.53e-11 ***
## WSF5          0.09959    0.02774   3.590 0.000331 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 405.20  on 292  degrees of freedom
## Residual deviance: 239.02  on 286  degrees of freedom
## AIC: 253.02
## 
## Number of Fisher Scoring iterations: 5
```

```
anova(glm3, test='Chisq')
```

```
## Analysis of Deviance Table
## 
## Model: binomial, link: logit
## 
## Response: Rain
## 
## Terms added sequentially (first to last)
## 
## 
##        Df Deviance Resid. Df Resid. Dev  Pr(>Chi)    
## NULL                     292     405.20              
## Season  3   53.975       289     351.22 1.136e-11 ***
## TMAX    1   22.694       288     328.53 1.899e-06 ***
## TMIN    1   73.795       287     254.73 < 2.2e-16 ***
## WSF5    1   15.718       286     239.02 7.353e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Compare the two models in anova:

Our NULL hypothesis is that co-efficient of Ave.Wind is 0. Therefore, if p <0.05, we can reject NULL hypothesis and say that the additional variable must remain in the final model.

```
anova(glm2,glm3,test='Chisq')
```

```
## Analysis of Deviance Table
##
## Model 1: Rain ~ Season + Ave.Wind + TMAX + TMIN + WSF5
## Model 2: Rain ~ Season + TMAX + TMIN + WSF5
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       285     236.95
## 2       286     239.01 -1  -2.0695   0.1503
```

P = 0.15 and we stick with NULL hypothesis i.e co-efficient of Ave.Wind is 0. Therefore, we pick glm3 to proceed.

Looking back at coefficients of glm3, we can see Season categorical classes, individually, may or may not be significant. In glm(),for each coefficient of every level of the categorical variable, a Wald test is performed to test whether the pairwise difference between the coefficient of the reference class and the other class is different from zero or not.

If only one categorical class is insignificant, this does not imply that the whole variable is meaningless and should be removed from the model. Moreover, the categorical class coefficients denote the difference of each class to the reference class. Therefore, they only tell us about the pairwise differences between the levels. To test whether the categorical predictor, as a whole, is significant is equivalent to testing whether there is any heterogeneity in the means of the levels of the predictor. Therefore, we need to create two models, one with Season and another without. Then compare the models using ANOVA and pick one from them.

```
glm4 <- glm(Rain ~ TMAX+TMIN+WSF5, family = binomial(link = "logit"),data = train_df)
summary(glm4)
```

```
## 
## Call:
## glm(formula = Rain ~ TMAX + TMIN + WSF5, family = binomial(link = "logit"), 
##     data = train_df)
## 
## Deviance Residuals: 
##     Min       1Q   Median       3Q      Max  
## -2.4396  -0.6167  -0.1505   0.6410   2.6617  
## 
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)    
## (Intercept)  0.04771    1.17190   0.041    0.968    
## TMAX        -0.28930    0.03848  -7.518 5.56e-14 ***
## TMIN         0.32532    0.05150   6.317 2.66e-10 ***
## WSF5         0.10294    0.02622   3.926 8.64e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 405.2  on 292  degrees of freedom
## Residual deviance: 248.4  on 289  degrees of freedom
## AIC: 256.4
## 
## Number of Fisher Scoring iterations: 5
```

Let’s compare the above two models(one with Seasons and other without) using Anova.

```
anova(glm3, glm4, test='Chisq')
```

```
## Analysis of Deviance Table
## 
## Model 1: Rain ~ Season + TMAX + TMIN + WSF5
## Model 2: Rain ~ TMAX + TMIN + WSF5
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)  
## 1       286     239.01                       
## 2       289     248.40 -3  -9.3861  0.02457 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

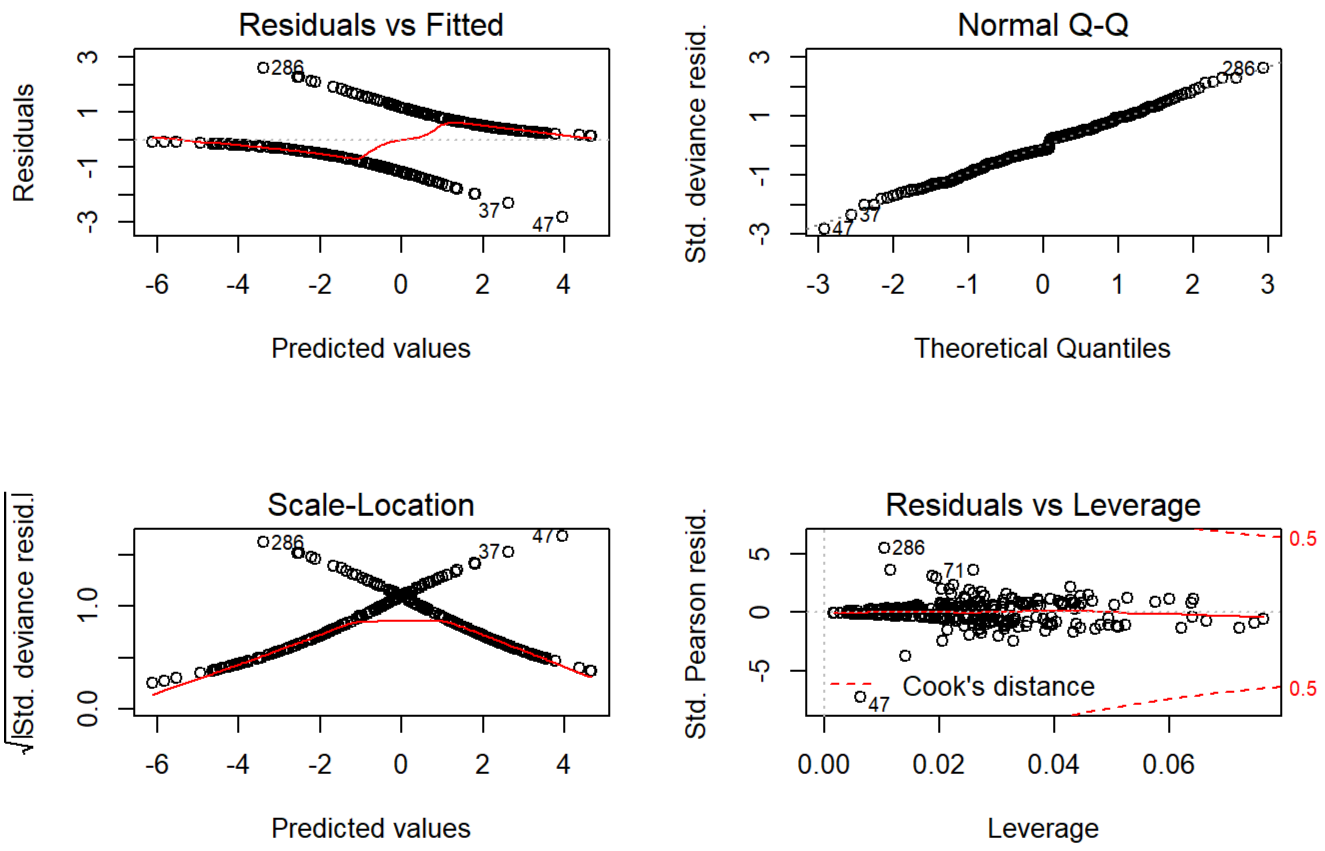
P value is significant and we reject the NULL hypothesis(Season’s coefficient is zero). Therefore, Season must remain in the model. AIC for glm4 is 253.02, whereas AIC for glm3 is 256.4. It is another indication that glm4 is a better model.

We find the odds ratios and 95% CI:

```
exp(cbind(OR = coef(glm3), confint(glm3)))
```

```
## Waiting for profiling to be done...
##                      OR       2.5 %    97.5 %
## (Intercept)  0.09533432 0.005274486 1.5684833
## SeasonSpring 0.55252833 0.219343670 1.3827518
## SeasonSummer 0.34494633 0.107891608 1.0808541
## SeasonWinter 2.44254122 0.966109119 6.4986820
## TMAX         0.75785084 0.696781552 0.8152555
## TMIN         1.44291970 1.303597237 1.6186522
## WSF5         1.10471805 1.049134105 1.1699394
```

Residual Plot for GLM models are not as useful as the ones in LM.



Now that we have the model, let’s go ahead and use this model to predict on the test data set.

By setting the parameter type=‘response’, R will output probabilities in the form of P(y=1|X)

Predicted values are in probablity. If probability of Rain > 0.5, we assume that it will rain that day and if probablity < 0.5, we can assume it won’t rain that day. This threshold can be changed based on subject knowledge.

Print the predictions

```
data.frame(Date = test_df$DATE,Rain_Actual = test_df$Rain, Predicted_Rain = predicted)
```

```
##           Date Rain_Actual Predicted_Rain
## 5   2016-01-05           1              1
## 8   2016-01-08           0              0
## 9   2016-01-09           0              0
## 16  2016-01-16           1              1
## 22  2016-01-22           1              1
## 25  2016-01-25           0              0
## 36  2016-02-05           1              1
## 49  2016-02-18           1              1
## 50  2016-02-19           1              1
## 52  2016-02-21           1              1
## 64  2016-03-04           1              1
## 68  2016-03-08           1              0
## 78  2016-03-18           0              1
## 79  2016-03-19           0              1
## 82  2016-03-22           0              1
## 88  2016-03-28           0              0
## 94  2016-04-03           1              0
## 95  2016-04-04           1              1
## 98  2016-04-07           0              0
## 101 2016-04-10           0              0
## 102 2016-04-11           0              1
## 104 2016-04-13           1              1
## 105 2016-04-14           1              0
## 117 2016-04-26           0              0
## 123 2016-05-02           0              0
## 150 2016-05-29           0              1
## 159 2016-06-07           0              0
## 160 2016-06-08           1              1
## 164 2016-06-12           0              0
## 170 2016-06-18           0              1
## 174 2016-06-22           0              0
## 179 2016-06-27           0              0
## 181 2016-06-29           0              0
## 184 2016-07-02           0              0
## 186 2016-07-04           0              0
## 189 2016-07-07           1              1
## 196 2016-07-14           0              0
## 198 2016-07-16           0              0
## 201 2016-07-19           0              0
## 204 2016-07-22           1              1
## 209 2016-07-27           0              0
## 215 2016-08-02           1              0
## 219 2016-08-06           0              0
## 222 2016-08-09           0              1
## 228 2016-08-15           0              0
## 229 2016-08-16           0              0
## 230 2016-08-17           0              0
## 235 2016-08-22           0              0
## 239 2016-08-26           0              0
## 241 2016-08-28           0              0
## 243 2016-08-30           0              0
## 245 2016-09-01           1              1
## 247 2016-09-03           0              0
## 267 2016-09-23           1              1
## 278 2016-10-04           1              1
## 284 2016-10-10           0              0
## 287 2016-10-13           1              1
## 292 2016-10-18           1              1
## 298 2016-10-24           1              1
## 299 2016-10-25           0              1
## 300 2016-10-26           1              1
## 307 2016-11-02           1              1
## 310 2016-11-05           1              1
## 319 2016-11-14           1              1
## 321 2016-11-16           1              1
## 329 2016-11-24           1              1
## 336 2016-12-01           1              1
## 338 2016-12-03           1              1
## 351 2016-12-16           0              0
## 355 2016-12-20           1              1
## 357 2016-12-22           1              1
## 360 2016-12-25           0              0
## 365 2016-12-30           1              0
```

One of the most important performance indicator of a model is the confusion matrix. It’s a tabular representation of Actual vs Predicted values. This helps us to find the accuracy of the model and avoid overfitting.

```
table(test_df$Rain, predicted)
```

```
##    predicted
##      0  1
##   0 31  8
##   1  5 29
```

Now, we calculate the accuracy of our prediction.

```
accuracy <- 1-mean(predicted != test_df$Rain)
accuracy
```

```
## [1] 0.8219178
```

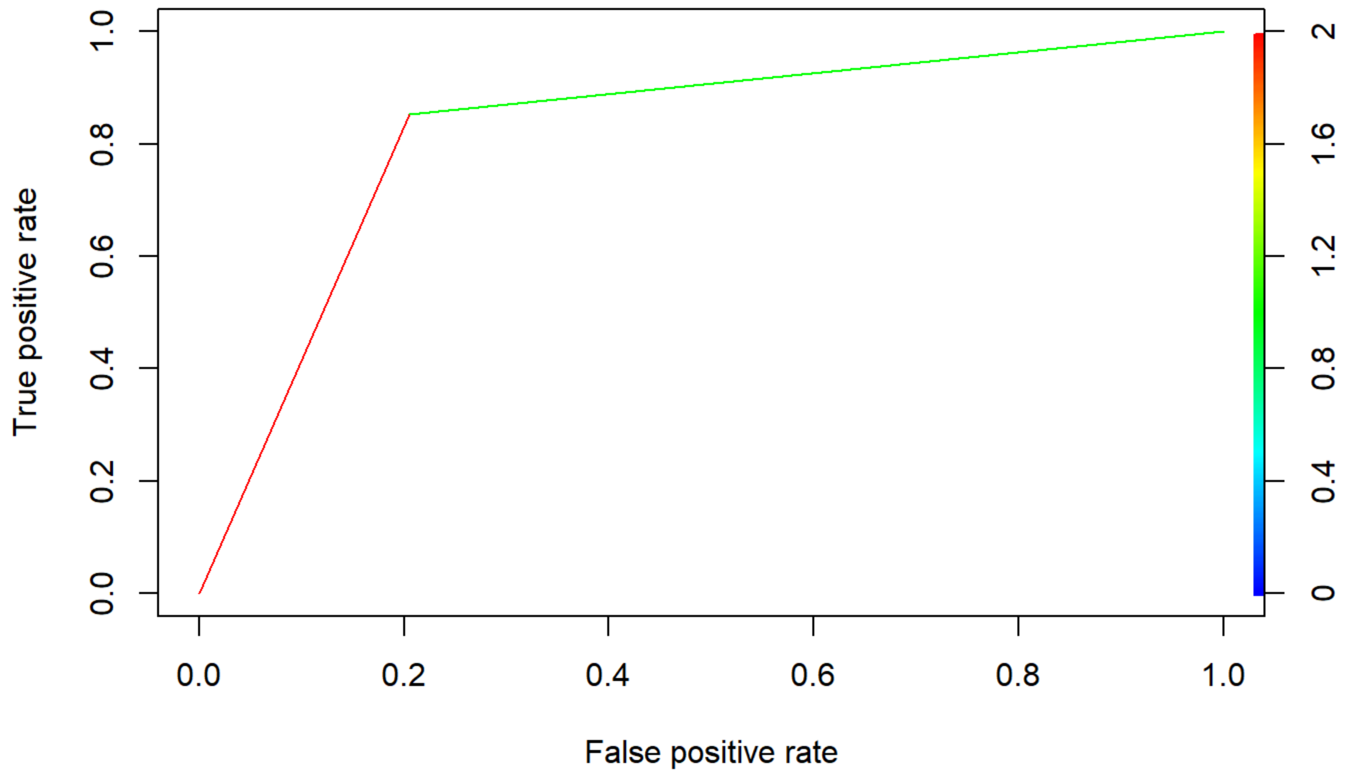
Accuracy for our model is **0.8219 or 82.2%**. This is fair value for our model. Accuracy can be increased by checking the ROCR curve and making necessary adjustment in the probablity threshold.

Final step is to plot the **ROC(Receiver Operating Characteristic)** curve and find out the **AUC(area under curve).** ROC summarizes the model’s performance by evaluating the trade offs between true positive rate (TPR or sensitivity) and false positive rate(1- specificity). The area under curve (AUC), referred to as index of accuracy(A) or concordance index, is a perfect performance metric for ROC curve. Higher the area under curve, better the prediction power of the model. Below is a sample ROC curve. As a rule of thumb, a model with good predictive ability should have an AUC closer to 1 (=1 ideally) than to 0.5.

```
ROCRPred <- prediction(predicted,test_df$Rain)
ROCRperf <- performance(ROCRPred, measure='tpr', x.measure='fpr')
```

Plot the ROC curve:

```
plot(ROCRperf, colorize=TRUE)
```



Calculate the Area under curve(AUC)

```
auc(test_df$Rain,predicted)
```

```
## [1] 0.8239065
```

AUC is 0.8239 which is an indication that our model is a good one.

Summary

It’s a good practice to start any given classification problem with logistic regression and go from there. It sort of lays the benchmark accuracy for other non-linear models that one can try afterwards.

Logistic Regression is one of the most widely used classification algorithms today. It has managed to survive Support Vector Machines, Random Forests and Boosting. It is also a crucial part of modern neural networks. As Andrew Ng says, Neural networks are nothing but a stack of multiple logistic regression models.

Here’s the link (https://github.com/ppattnayak/SeattleRainLogisticRegression) to the code and data file.

Citation:

[1] *http://faculty.cas.usf.edu/mbrannick/regression/Logistic.html* (http://faculty.cas.usf.edu/mbrannick/regression/Logistic.html*)