Experiment No 2

Title: Logistic Regression Analysis in R

Theory

Logistic regression analysis studies the association between a categorical dependent variable and a set of independent (explanatory) variables. The name logistic regression is used when the dependent variable has only two values, such as 0 and 1 or Yes and No. The name multinomial logistic regression is usually reserved for the case when the dependent variable has three or more unique values, such as Married, Single, Divored, or Widowed. Although the type of data used for the dependent variable is different from that of multiple regression, the practical use of the procedure is similar.

Logistic regression competes with discriminant analysis as a method for analyzing categorical-response variables. Many statisticians feel that logistic regression is more versatile and better suited for modelling most situations than is discriminant analysis. This is because logistic regression does not assume that the independent variables are normally distributed, as discriminant analysis does.

We know that a liner model assumes that response variable is normally distributed. We have equation of the form $Yi = \beta 0 + \beta 1X + \varepsilon i$, where we predict the value of Y for some value of X. We know this is linear because for each unit change in X, it will affect the Y by some magnitude $\beta 1$. *Also,* the error term εi *is* assumed to be normally distributed and if that error term is added to each output of Y, then Y is also becoming normally distributed, which means that for each value of X we get Y and that Y is contributing to that normal distribution. Now, this is all good when the value of Y can be $-\infty$ to $+\infty$, but if the value needs to be TRUE or FALSE, 0 or 1, YES or No then our variables does not follow normal distribution pattern. All we have is the counts of 0s and 1s which is only useful to find probabilities for example say if you have five 0s and fifteen 1s then getting 0 has probability of 0.25 and getting 1 has the probability of 0.75. But how can we use that probability to make a kind of smooth distribution that fits a line (Not linear) as close as possible to all the points you have, given that those points are either 0 or 1.

To do that you have to imagine that the probability can only be between 0 and 1 and when you try to fit a line to those points, it cannot be a straight line but rather a S-shape curve.

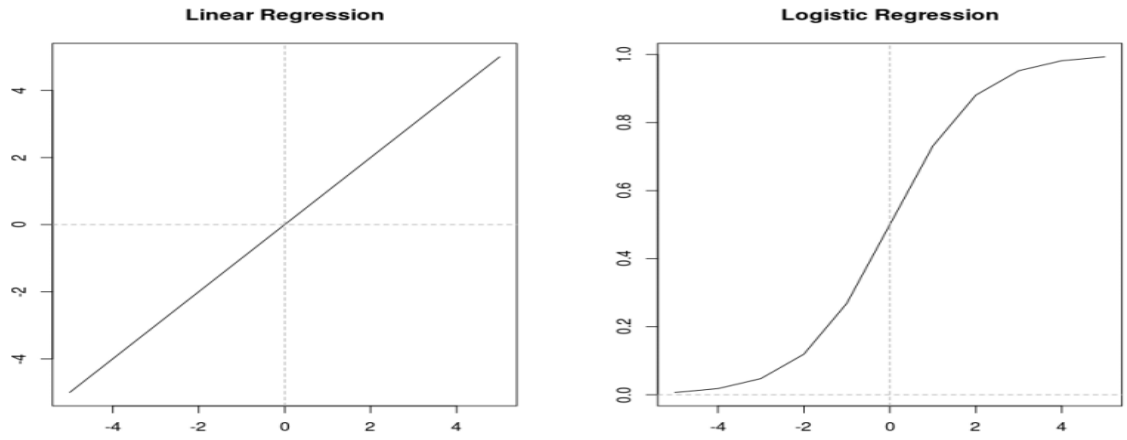

Fig. Linear vs Logistic

If you have a greater number of 1s then that S will be skewed upwards and if you have greater numbers of 0s then it will be skewed downwards. Note that the number 0 on Y-axis represents that half of the counts of total number is on left and half of total count is on right, but it cannot be the case always.

Now the question arises, how do we map binary information of 1s and 0s to regression model which uses continuous variables? The reason we do that mapping is because we want our model to be capable of finding the probability of desired outcome being true. Below I am going to describe how we do that mapping. Keep in mind that the main premise of logistic regression is still based upon a typical regression model with a few methodical changes.

Now, to find the probability of desired outcome, two things we must always be followed.

1. That the probability cannot be negative, so we introduce a term called exponential in our normal regression model to make it logistic regression.
2. Since the probability can never be greater than 1, we need to divide our outcome by something bigger than itself.

And based on those two things, our formula for logistic regression unfolds as following:

1. Regression formula give us Y using formula $Yi = \beta 0 + \beta 1X + \varepsilon i$.
2. We have to use exponential so that it does not become negative and hence we get $P = \exp(\beta 0 + \beta 1X + \varepsilon i)$.
3. We divide that P by something bigger than itself so that it remains less than one and hence we get $P = e(\beta 0 + \beta 1X + \varepsilon i)/e(\beta 0 + \beta 1X + \varepsilon i) + 1$.
4. After doing some calculations that formula in 3rd step can be re-written as $\log(p/(1-p)) = \beta 0 + \beta 1X + \varepsilon i$.
5. $\log(p/(1-p))$ is called the odds of probability. If you look closely it is the probability of desired outcome being true divided by the probability of desired outcome not being true and this is called logit function.

Working

When you calculate total number of 1s and 0s you can calculate the value of $\log(p/(1-p))$ quite easily and we know that this value is equal to $\beta 0 + \beta 1X + \varepsilon i$. Now you can put that value into the formula $P = e(\beta 0 + \beta 1X + \varepsilon i)/e(\beta 0 + \beta 1X + \varepsilon i) + 1$ and get the value of P. That P will be the probability of your outcome being TRUE based on some given parameters.

From a different perspective, let's say you have your regression formula available with intercept and slope already given to you, you just need to put in the value of X to predict Y. But you know in logistic regression it doesn't work that way, that is why you put your X value here in this formula $P = e(\beta 0 + \beta 1X + \varepsilon i)/e(\beta 0 + \beta 1X + \varepsilon i) + 1$ and map the result on x-axis and y-axis. If the value is above 0.5 then you know it is towards the desired outcome (that is 1) and if it is below 0.5 then you know it is towards not-desired outcome (that is 0).

Dataset and Task

The source of the data is from UCLA which has 4 variable called admit, GRE score, GPA and rank of their undergrad school. Our aim is to build a model so that predict the probability of that student getting admit if we are given his profile.

R Program

```
df <- read.csv("D:/VinodData/Year 2019 20/Practicals/ML/2/binary.csv")

str(df)

sum(is.na(df))

summary(df)

xtabs(~ admit +rank ,data=df)

df$rank <- as.factor(df$rank)

logit <- glm(admit ~ gre+gpa+rank,data=df,family="binomial")

summary(logit)

x <- data.frame(gre=790,gpa=3.8,rank=as.factor(1))

p<- predict(logit,x)

p
```

Output of the program and analysis

```
> df <- read.csv("D:/VinodData/Year 2019 20/Practicals/ML/2/binary.csv")
> str(df)
'data.frame':        400 obs. of  4 variables:
 $ admit: int  0 1 1 1 0 1 1 0 1 0 ...
 $ gre  : int  380 660 800 640 520 760 560 400 540 700 ...
 $ gpa  : num  3.61 3.67 4 3.19 2.93 3 2.98 3.08 3.39 3.92 ...
 $ rank : int  3 3 1 4 4 2 1 2 3 2 ...
```

We see that variable are either integer or number

```
> sum(is.na(df))
[1] 0
```

No null values

```
> summary(df)
    admit          gre          gpa          rank
 Min.   :0.0000   Min.   :220.0   Min.   :2.260   Min.   :1.000
 1st Qu.:0.0000   1st Qu.:520.0   1st Qu.:3.130   1st Qu.:2.000
 Median :0.0000   Median :580.0   Median :3.395   Median :2.000
 Mean   :0.3175   Mean   :587.7   Mean   :3.390   Mean   :2.485
 3rd Qu.:1.0000   3rd Qu.:660.0   3rd Qu.:3.670   3rd Qu.:3.000
 Max.   :1.0000   Max.   :800.0   Max.   :4.000   Max.   :4.000
```

We can notice that there are a greater number of rejects than there are acceptance since the mean of variable admit is less than "0.5".

```
> xtabs(~ admit +rank ,data=df)
    rank
admit  1  2  3  4
   0 28 97 93 55
   1 33 54 28 12
```

We do this to check if the admits are distributed well enough in each category of rank. If let's say one rank has only 5 admit or reject information, then it will not be necessary to include that rank in analysis.

```
> df$rank <- as.factor(df$rank)
> logit <- glm(admit ~ gre+gpa+rank,data=df,family="binomial")
> summary(logit)

Call:
glm(formula = admit ~ gre + gpa + rank, family = "binomial",
   data = df)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.6268  -0.8662  -0.6388   1.1490   2.0790

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.989979   1.139951  -3.500 0.000465 ***
gre          0.002264   0.001094   2.070 0.038465 *
gpa          0.804038   0.331819   2.423 0.015388 *
rank2       -0.675443   0.316490  -2.134 0.032829 *
rank3       -1.340204   0.345306  -3.881 0.000104 ***
rank4       -1.551464   0.417832  -3.713 0.000205 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

(Dispersion parameter for binomial family taken to be 1)

```
    Null deviance: 499.98  on 399  degrees of freedom
Residual deviance: 458.52  on 394  degrees of freedom
AIC: 470.52

Number of Fisher Scoring iterations: 4
```

Now we run our logit function, but before that we convert rank variable from integer to factor.

1- Each one-unit change in gre will increase the log odds of getting admit by 0.002, and its p-value indicates that it is somewhat significant in determining the admit.
2- Each unit increase in GPA increases the log odds of getting admit by 0.80 and p-value indicates that it is somewhat significant in determining the admit.
3- The interpretation of rank is different from others, going to rank-2 college from rank-1 college will decrease the log odds of getting admit by -0.67. Going from rank-2 to rank-3 will decrease it by -1.340
4- The difference between Null deviance and Residual deviance tells us that the model is a good fit. Greater the difference better the model. Null deviance is the value when you only have intercept in your equation with no variables and Residual deviance is the value when you are taking all the variables into account. It makes sense to consider the model good if that difference is big enough.

```
> x <- data.frame(gre=790,gpa=3.8,rank=as.factor(1))
> p<- predict(logit,x)
> p
      1
0.85426
```

Let's say a student have a profile with 790 in GRE,3.8 GPA and he studied from a rank-1 college. Now you want to predict the chances of that boy getting admit in future