Hannah Roach
3/7/2019

Lab Exercise 7: Logistic Regression

Purpose:	This lab is designed to investigate and practice the Logistic Regression method. After completing the tasks in this lab you should able to: • Use R functions for Logistic Regression – *also known as Logit*) • Predict the dependent variables based on the model • Investigate different statistical parameter tests that measure the effectiveness of the model

Tasks:	Tasks you will complete in this lab include: • Use R –Studio environment to code Logit models • Review the methodology to validate the model and predict the dependent variable for a set of given independent variables • Use R graphics functions to visualize the results generated with the model

References:	References used in this lab are located in your ***Student Resource Guide Appendix.***

Workflow Overview

1. Set the Working Directory
2. Define the problem and review input data
3. Read in and Examine the Data
4. Build and Review logistic regression Model
5. Review and interpret the coefficients
6. Visualize the Model Using the Plot Function
7. Use relevel Function to re-level the Price factor with value 30 as the base reference
8. Plot the ROC Curve
9. Predict Outcome given Age and Income
10. Predict outcome for a sequence of Age values at price 30 and income at its mean
11. Predict outcome for a sequence of income at price 30 and Age at its mean
12. Use Logistic regression as a classifier

LAB Instructions

Step	Action
1	Log in with GPADMIN credentials on to R-Studio.
2	**Set the Working Directory** Set the working directory to ~/LAB07/ by executing the command: `setwd("~/LAB07")` • (Or using the "Tools" option in the tool bar in the RStudio environment).

Step	Action

3	**Define the problem and review input data**
	
	Logistic Regression, also known as ***Logit***, is typically used in models where the dependent variables have a binary outcome (True/False, which is coded with 1/0). You model the log odds of the outcome as a linear combination of predictor variables).
	
	Marketing Survey Data
	In this lab you use hypothetical marketing survey data in which customers:
	• Responded to the question:
	  ○ Would you choose a product based on a "pricing" factor (three "Price" ranges 10, 20 and 30)?
	• Response options:
	  ○ "1" for "yes" and "0" for "no"
	• The survey also collected information such as "Age" and "Income" of the respondent.
	
	Business Need
	The marketing campaign team wants to send special offers to those respondents with the highest probability of purchase.
	
	This data file "survey.csv" is available in the folder ~\LAB07\survey.csv.
	
	1. Review the survey.csv file.
	2. How many responses to the survey does the file contain?
	750
	3. What is the main purpose of building this model?
	To determine which customers have higher probabilities of making a purchase

4	**Read in and Examine the Data:**
	
	1. The first step in the modeling process is to examine the data and determine if there are any outliers. To do this you must read in the survey data, use the following command:
	
	`Mydata <- read.csv("survey.csv",header=TRUE,sep=",")`
	
	2. With the following command, explore the data further:
	
	`> table(Mydata$MYDEPV)`
	How many are 0 and how many are 1?
	`> table(Mydata$MYDEPV)`
	
	`  0   1`
	`426 324`
	
	`> with(Mydata, table(Price,MYDEPV))`
	What does the matrix look like?
	`> with(Mydata, table(Price,MYDEPV))`
	`     MYDEPV`
	`Price   0   1`
	`   10 115 135`
	`   20 137 113`
	`   30 174  76`
	
	`> summary(Mydata$Age)`
	What is the median age? 32
	`> summary(Mydata$Age)`
	`   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.`
	`     18      32      32      36      43      66`
	`> cor.mat <- cor(Mydata[,-1])`
	`> cor.mat`
	3. Review the results on the console
	`> cor.mat <- cor(Mydata[,-1])`
	`> cor.mat`
	`        Price Income    Age`
	`Price      1 0.0000 0.0000`
	`Income     0 1.0000 0.0961`
	`Age        0 0.0961 1.0000`
	Note: The general rule **is not** to include variables in your model that are too highly correlated with other predictors. For example, including two variables that are correlated by 0.85 in your model may prevent the true contribution of each variable from being identified by the statistical algorithm. Confirm that the variables in our survey do not fall in this category.

115

Build and Review Logistic Regression Model:

1. Use the "glm" function for logit modeling. Type in the following command:

```
mylogit <- glm(MYDEPV ~ Income + Age + as.factor(Price)
,
               data=Mydata,family=binomial(link="logit"),
               na.action=na.pass)
```

2. Review the model by typing the "summary" and "plot" functions:

```
summary (mylogit)
```

Review the results of the summary command, for the fitted model, on the console. Results you should see:

- The first line provides the model you specified.
- Next, you should see the **deviance residuals**, which provide the measure of the model fit.
- The next part of the output shows the **coefficients,** their standard errors, the **z-statistic** (sometimes called a Wald z-statistic), and the associated **p-values.**
- Both **Income** and **Age** are statistically significant, as are the two terms for **Price**.
- The **logistic regression coefficients** show the change in the **log odds** of the outcome for a one unit increase in the predictor variable.
- Residual deviance: analogous to the Residual Sum of Squares of a linear model; that is, it is related to the "total error" of the fit. It is twice the negative log likelihood of the model.
- Null deviance: the deviance associated with the "null model" -- that is the model that returns just the global probability of TRUE for every x. The quantity 1 - (Residual deviance/Null deviance) is sometimes called "pseudo-R-squared"; you use it to evaluate goodness of fit in the same way that R-sqr is used for linear models.

The interpretation of the results are as follows:

1. Review the "Estimate" column. For every one unit change in **Income**, the log odds of Purchase (versus no-Purchase) increases by 0.12876.

2. Record the number that describes how much one unit increase in **Age** increases the log odds of purchase: The indicator variables for **Price** are interpreted differently. **For** example, Purchase decision at a **Price** of 20, compared with a **Price** of **10,** decreases the log odds of admission by 0.74418
   Age Estimate: 0.03506

3. Record the log odds at Price point 30 compared to Price point 20 below:

Step	Action
	The summary then shows the table of coefficients that are “fit indices”, including the null and deviance residuals and the AIC.

```
> summary(mylogit)

Call:
glm(formula = MYDEPV ~ Income + Age + as.factor(Price), family = bi
"logit"),
    data = Mydata, na.action = na.pass)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-3.039  -0.558  -0.243   0.418   3.238

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)        -6.02116    0.53244  -11.31  < 2e-16 ***
Income              0.12876    0.00923   13.95  < 2e-16 ***
Age                 0.03506    0.01179    2.97   0.0029 **
as.factor(Price)20 -0.74418    0.26439   -2.81   0.0049 **
as.factor(Price)30 -2.21028    0.31108   -7.11  1.2e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1025.81  on 749  degrees of freedom
Residual deviance:  534.17  on 745  degrees of freedom
AIC: 544.2

Number of Fisher Scoring iterations: 6
```

6

Review the results and interpret the coefficients

1. Use the "confint" function to obtain the confidence intervals of the coefficient estimates:

```
confint(mylogit)
```

**** printscreen**

```
> confint(mylogit)
Waiting for profiling to be done...
                        2.5 %  97.5 %
(Intercept)          -7.102 -5.0110
Income                0.111  0.1477
Age                   0.012  0.0583
as.factor(Price)20 -1.269 -0.2308
as.factor(Price)30 -2.841 -1.6184
>
```

2. Review the results on the console.

- You can also exponentiate the coefficients and interpret them as odds-ratios.
- To get the exponentiated coefficients, use (**exp( )**)
- The object you want to exponentiate is called coefficients and it is part of mylogit (**mylogit$coefficients**).

```
exp(mylogit$coefficients)
```

You can observe that for every unit change in income, the odd-ratio of Purchase increases by a multiplicative factor of 1.137 (and remember a multiplicative factor of 1 corresponds to no change).

This is actually a bit more intuitive than the log odds explanation you reviewed in the previous step. Observe that that Age does not appear to be a very strong factor in this model, and the price factor of 30 has a stronger effect than a price factor of 20.

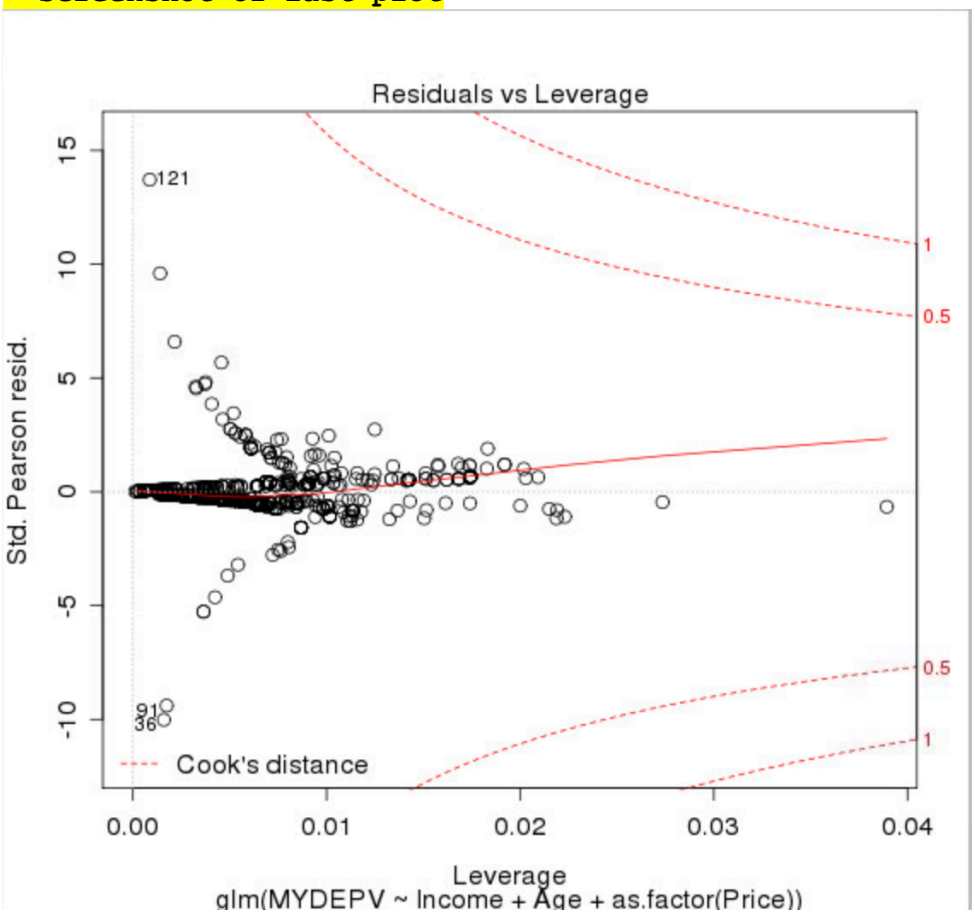
3. Pseduo R^2 with first obtaining the names of the class members of the model "mylogit" and then using the formula (1 – deviance/null deviance)

```
attributes(mylogit)
1- with(mylogit, deviance/null.deviance)
```

Step	Action

	`> attributes(mylogit)  # get me the names of the 'class members'`
	`$names`
	` [1] "coefficients"  "residuals"  "fitted.values"`
	` [4] "effects"  "R"  "rank"`
	` [7] "qr"  "family"  "linear.predictors"`
	`[10] "deviance"  "aic"  "null.deviance"`
	`[13] "iter"  "weights"  "prior.weights"`
	`[16] "df.residual"  "df.null"  "y"`
	`[19] "converged"  "boundary"  "model"`
	`[22] "call"  "formula"  "terms"`
	`[25] "data"  "offset"  "control"`
	`[28] "method"  "contrasts"  "xlevels"`
	
	`$class`
	`[1] "glm" "lm"`

Step	Action

7	**Visualize the Model Using the Plot Function:**
	
	`plot(mylogit)`
	`**screenshot of last plot`
	 
	
	You should see multiple plots generated on the graphics window.

Step	Action

8	**Use relevel Function to re-level the Price factor with value 30 as the base reference**.
	
	In the original model that we fitted with the function call:
	
	`mylogit <- glm(MYDEPV ~ Income + Age + as.factor(Price) , + data= Mydata,family=binomial(link="logit"), + na.action=na.pass)`
	
	we obtained the results shown below:
	`Coefficients: Estimate Std. Error z value Pr(>|z|) (Intercept) -6.02116 0.53244 -11.309 < 2e-16 *** Income 0.12876 0.00923 13.950 < 2e-16 *** Age 0.03506 0.01179 2.974 0.00294 ** as.factor(Price)20 -0.74418 0.26439 -2.815 0.00488 ** as.factor(Price)30 -2.21028 0.31108 -7.105 1.2e-12 *** ---`
	
	What does this tell us?
	
	1. Now let's use 30 as the reference price, instead of 10. Type in the following:
	
	`Mydata$pricefactor = relevel(as.factor(Mydata$Price), "30")`

Step	Action

8 Cont.	Fit the Model Again (mylogit2) and Display the Summary:
	`mylogit2 <- glm(MYDEPV ~ Income + Age + pricefactor , data= Mydata,family=binomial(link="logit"), na.action=na.pass)`
	`summary(mylogit2)`
	
	You will see the results as follows:
	`Coefficients:`
	`Estimate Std. Error z value Pr(>|z|)`
	`(Intercept) -8.23144 0.66180 -12.438 < 2e-16 ***`
	`Income 0.12876 0.00923 13.950 < 2e-16 ***`
	`Age 0.03506 0.01179 2.974 0.00294 **`
	`pricefactor10 2.21028 0.31108 7.105 1.20e-12 ***`
	`pricefactor20 1.46610 0.29943 4.896 9.76e-07 ***`
	`---`
	
	Notice that the intercept has changed (because we changed the reference situation), but the coefficients for Income and Age are the same. The new model tells us that the odds of MYDEPV increase when price decreases from 30 to 10, and less so price decreases from 30 to 20.

9

Plot the ROC Curve:

1. Make sure you have the package ROCR installed and the library included

```
install.packages("ROCR", repos= "http://cran.r-project.org", lib="/home/gpadmin/R/library/")

library(ROCR)
```

2. First get all the probability scores on the training data

```
pred = predict(mylogit, type="response")
```

3. Every classifier evaluation using ROCR starts with creating a prediction object. This function is used to transform the input data (which can be in vector, matrix, data frame, or list form) into a standardized format.
   We create the prediction object needed for ROCR as follows:

```
predObj = prediction(pred, Mydata$MYDEPV)
```

4. All kinds of predictor evaluations are performed using the function "performance". Read and understand the parameters of the function with

```
?performance
```

5. We now create the ROC curve object and the AUC object with performance function

```
rocObj = performance(predObj, measure="tpr", x.measure="fpr")  # creates ROC curve obj
aucObj = performance(predObj, measure="auc")  # auc object
```

6. Extract the value of AUC and display on the console:

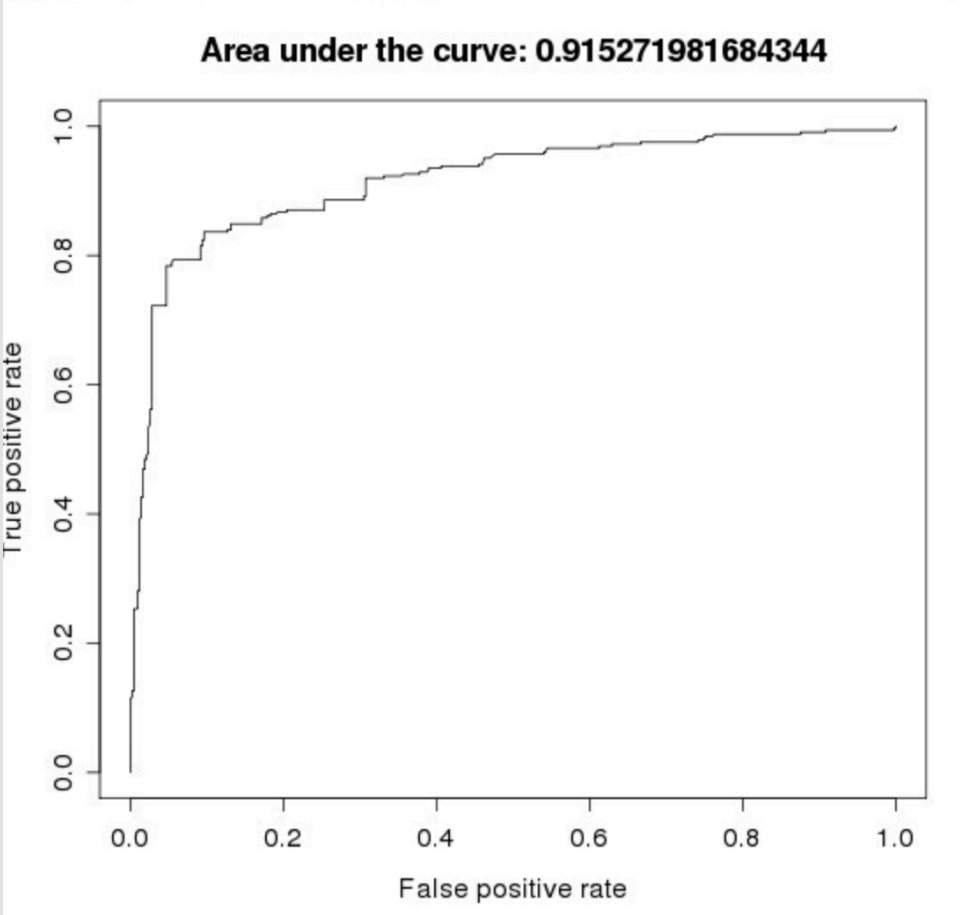
```
auc = aucObj@y.values[[1]]
auc
```

7. What is the value of AUC?
   0.915

8. We will plot the ROC curve now

```
plot(rocObj, main = paste("Area under the curve:", auc))

**show screenshot
```

Step	Action
	 9. Review the curve on the plot window. Review the discussions on ROC in the student resources guide. Record your observations below: It shows that this test is fairly accurate since the ROC value is close to 1. This means that there are the true positive rate is high. This is the number of times the model predicted true, and the result was true, over the number of true results.

Step	Action

lib10	**Predict Outcome given Age and Income:**
	
	1. Use the "predict" function to predict the probability of the purchase outcome given **Age** and **Income.** Start with predicting the probability of the purchase decision at different **Price** points (10, 20, and 30). Create a "data frame" called "newdata1" using the following commands:
	
	`Price <- c(10,20,30)`
	`Age <- c(mean(Mydata$Age))`
	`Income <- c(mean(Mydata$Income))`
	`newdata1 <- data.frame(Income,Age,Price)`
	`newdata1`
	
	You are predicting with **Income** and **Age** both set at their mean value and **Price** at 10, 20 and 30.
	
	Note: The values of the data frame "newdata1" displayed on the console. The predict function requires the variables to be named exactly as in the fitted model.
	
	**Show Screenshot
	`> newdata1 <- data.frame(Income,Age,Pri`
	`> newdata1`
	`  Income Age Price`
	`1   42.5  36    10`
	`2   42.5  36    20`
	`3   42.5  36    30`
	
	2. Create the fourth variable "PurchaseP".
	
	`newdata1$PurchaseP <- predict`
	`(mylogit,newdata=newdata1,type="response")`
	
	`newdata1`
	`> newdata1$PurchaseP <- predict (mylogit,newdata=newdata1,type`
	`> newdata1`
	`  Income Age Price PurchaseP`
	`1   42.5  36    10     0.671`
	`2   42.5  36    20     0.492`
	`3   42.5  36    30     0.183`
	`>`
	
	3. What is your observation on the probability of purchase at different **Price** levels?
	As the price decreases, the probability of purchase increases.

11

Predict outcome for a sequence of Age values at price 30 and income at its mean:

1. Keep the **Price** at 30, **Income** at its mean value and select a sequence of values for **Age** starting at a minimum age, incrementing by 2 until the maximum age in our dataset:

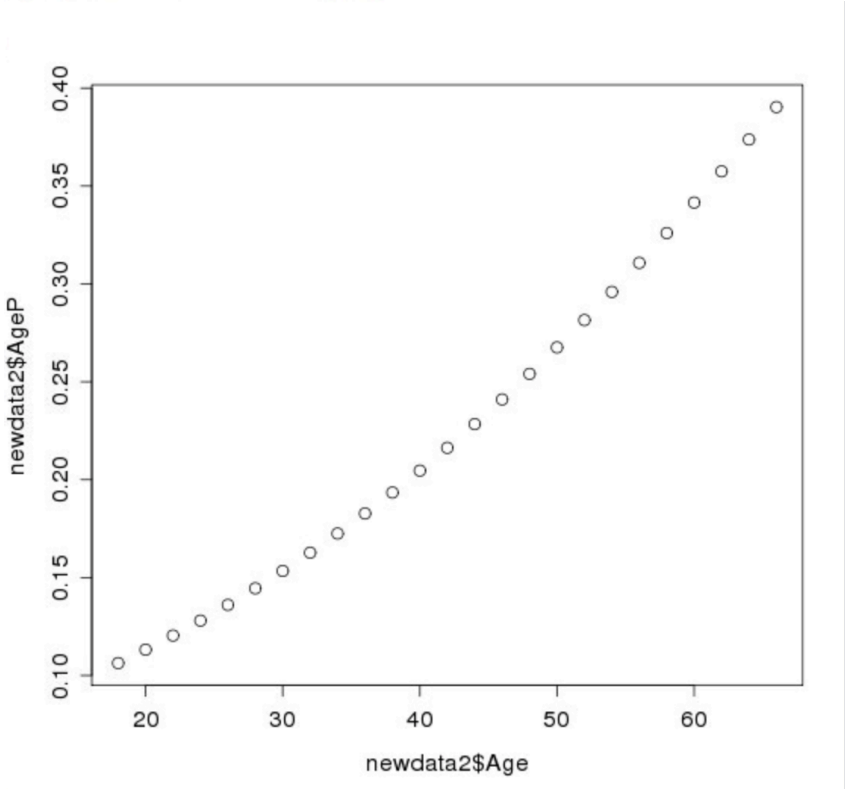
```
newdata2 <-
data.frame(Age=seq(min(Mydata$Age),max(Mydata$Age),2),

Income=mean(Mydata$Income),Price=30)
newdata2$AgeP<-
predict(mylogit,newdata=newdata2,type="response")
cbind(newdata2$Age,newdata2$AgeP)
```

Newdata2$AgeP stores the predicted variables and you just display the sequence for **Age** you generated and the corresponding probability of the purchase decision using the "cbind" function shown above.

2. Plot and visualize how the "purchase" probability varies with Age:

```
plot(newdata2$Age,newdata2$AgeP)
new**Show Screenshot
```



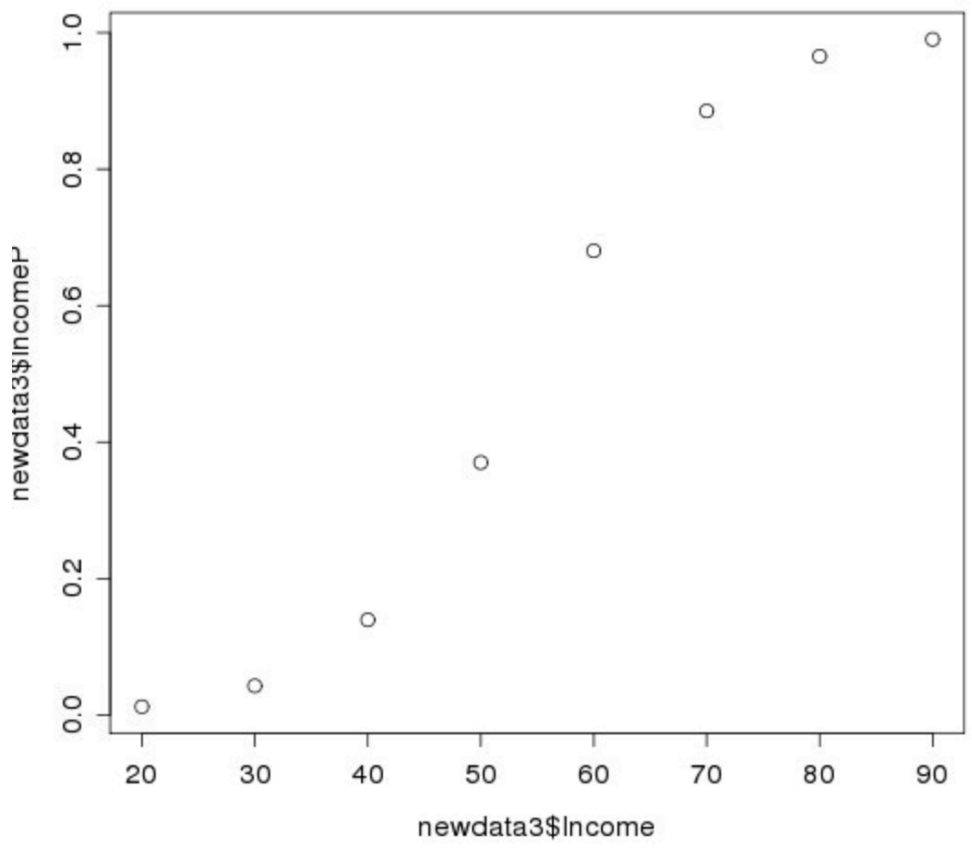
Predict outcome for a sequence of income at price 30 and Age at its mean:

1. Using the same methodology, create a data frame newdata3 with the following characteristics:
   - **Income** is a sequence from 20 to 90 in steps of 10
   - **Age** is the mean value for the dataset Mydata
   - **Price** point at 30

2. Predict **newdata3$IncomeP** and display the Income sequence along with the predicted probabilities.

```
> #Prediction - 3
> newdata3 <- data.frame(Income= seq(20,90,10),Age=mean(Mydata$Age
> newdata3$IncomeP<-predict(mylogit,newdata=newdata3,type="respons
> cbind(newdata3$Income,newdata3$IncomeP)
     [,1]   [,2]
[1,]   20 0.0122
[2,]   30 0.0428
[3,]   40 0.1395
[4,]   50 0.3700
[5,]   60 0.6804
[6,]   70 0.8853
[7,]   80 0.9655
[8,]   90 0.9902
> plot(newdata3$Income,newdata3$IncomeP)
>
```

3. Plot the results. **ShowScreenshot

Step	Action
	 newdata3$Income 0.0 0.2 0.4 0.6 0.8 1.0 20 30 40 50 60 70 80 90 newdata3$Income

13

Use Logistic regression as a classifier:

Recall the problem statement in Step 3, the marketing campaign team wants to send special offers to those respondents with the highest probability of purchase. They have established a threshold of 0.5 and they want to target customers whose probability of purchase are greater than 0.5.

Note: We are assuming that age and income are uniformly distributed in our customer base, and the price factors of our products are also uniformly distributed. Typically in order to run a scenario like this you should understand the demographic distribution of the customers (and the price distribution of the products).

1. You want an idea of how many offers will be sent out, using this threshold, so you test it on a 'random' set of data. First, generate this random set using "runif" functions:

```
newdata4 <- data.frame (
          Age=
round(runif(10,min(Mydata$Age),max(Mydata$Age))),

Income=round(runif(10,min(Mydata$Income),max(Mydata$Incom
e))),
          Price = round((runif(10,10,30)/10))*10)
newdata4$Prob <-
predict(mylogit,newdata=newdata4,type="response")
newdata4
```

2. How many samples in your random selection qualify for special offers?

8

```
> newdata4
   Age Income Price  Prob
1   63     85    20 0.998
2   24     88    30 0.981
3   25     65    30 0.734
4   46     38    30 0.151
5   62     77    20 0.995
6   60     72    20 0.990
7   58     49    20 0.829
8   35     58    20 0.873
9   63     71    20 0.990
10  54     28    20 0.220
>
```

End of Lab Exercise