CS 251 Statistical Computing

HOP 8: R for statistical project

8/10/2020 Developed by Aya Khalil

7/7/2019 Reviewed by

School of Technology and Computing @City University of Seattle (CityU)

**Before You Start**

* If you already finished this module through any CityU Technology Institute (TI) courses,  
  just skim this module and skip it.
* Version numbers may not match with the guide. But that should be fine.  
  If given the option to choose between stable release (long-term support) or most recent, please choose the stable release.
* This guide targets Windows OS users. So, MacOS users may have different commands to input in the shell/terminal.
* We cannot explain every step. **This cookbook always needs your own creative judgement.**
* **For your working directory, use your course number.** The hands-on tutorial may use a different course number as an example.

**Learning Outcomes**

* Logistic regression

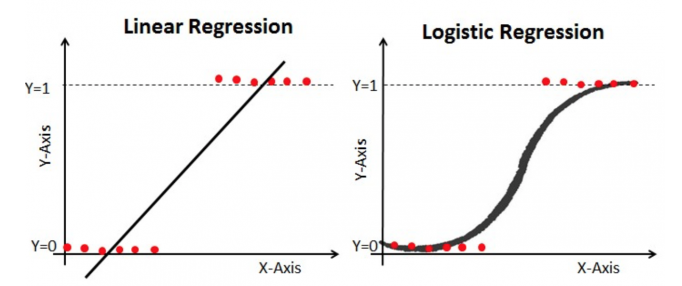
**Resource**

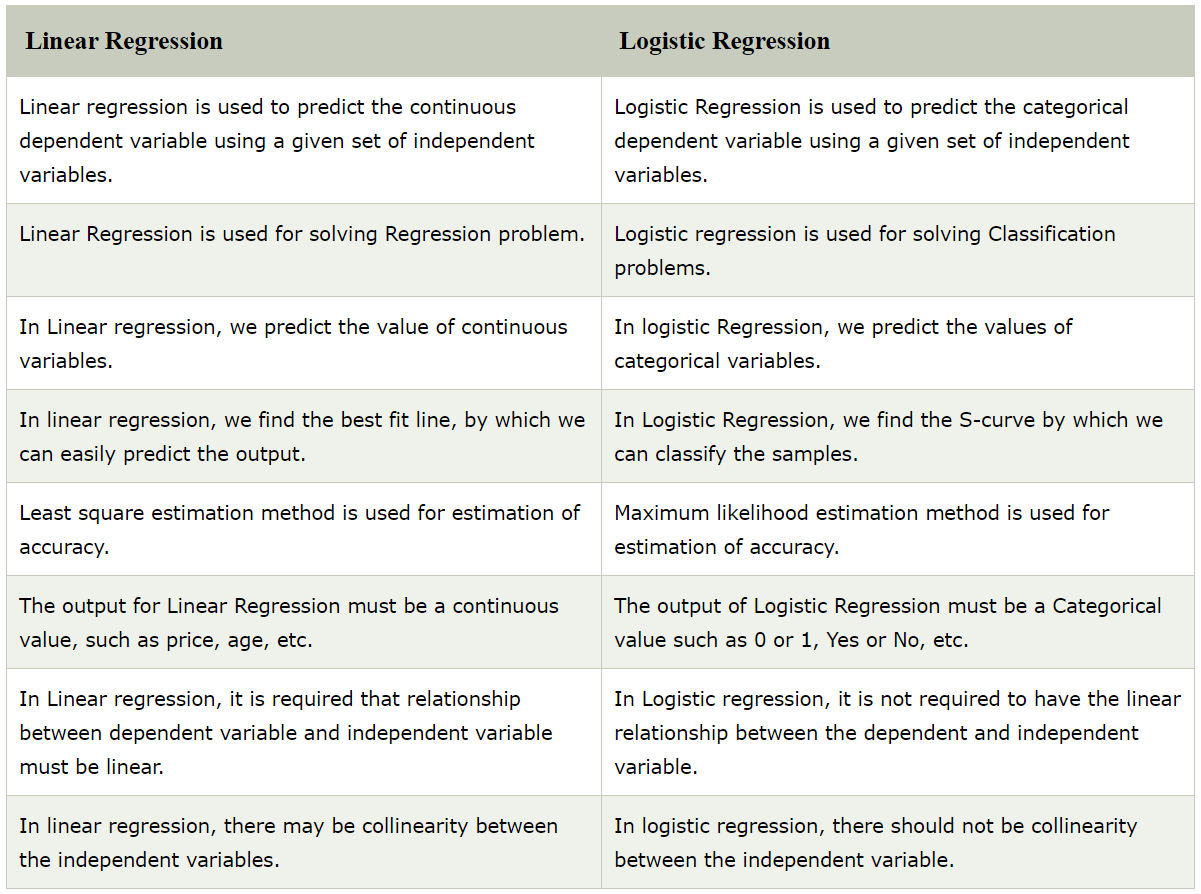
* Data Science and Machine Learning BootCamp with R online course
* Logistic Regression with Python: <https://medium.com/@ODSC/logistic-regression-with-python-ede39f8573c7>
* Linear Regression vs Logistic Regression: <https://www.javatpoint.com/linear-regression-vs-logistic-regression-in-machine-learning>

**Logarithm regression**

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model (a form of binary regression).

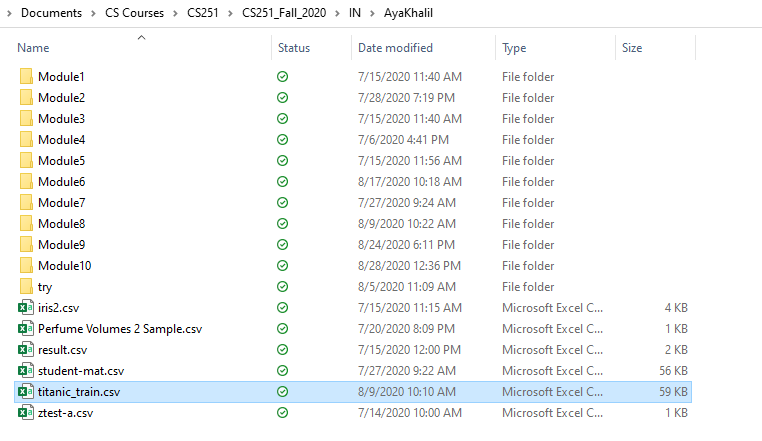
**Linear regression vs. logistic regression**





In this module we are going to use the Titanic data set from Kaggel.com and we are going to predict that whether someone survives or deceased in the Titanic disaster.

**Download the attached CSV file “titanic.train.csv” and make sure that you have it in your working directory like what we did in the previous HOP5A- Part1.**



**Setup Working Environment for Module8**

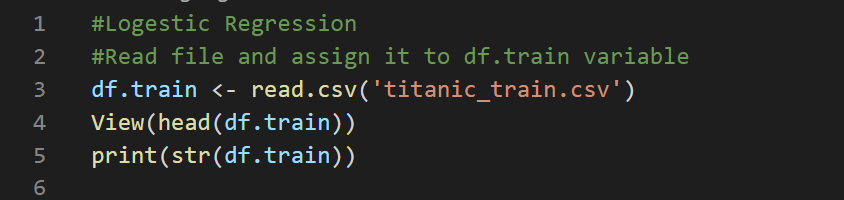
1. Open VS Code.

* **online student:** Open CS251 \_Fall\_2020/**ON**/FirstnameLastname /. ( File > Open )
* **onsite student:** Open CS251 \_ Fall \_2020/**IN**/FirstnameLastname. ( File > Open )

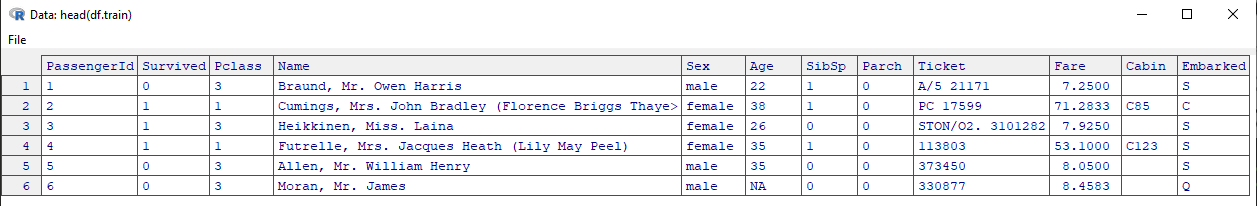
1. Then, create the “**Module8**” directory in the VSCode.

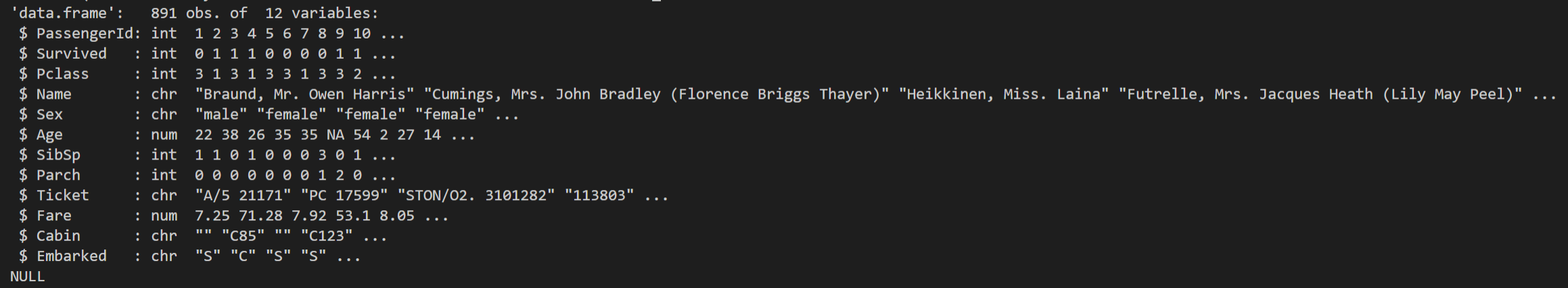
>>>mkdir Module8

* In module8 project folder, create new file LogReg.R
* Type the following code in LogReg.R file



*Note: If View(head(df.train)) doesn’t work, then use print(head(df.train)).*

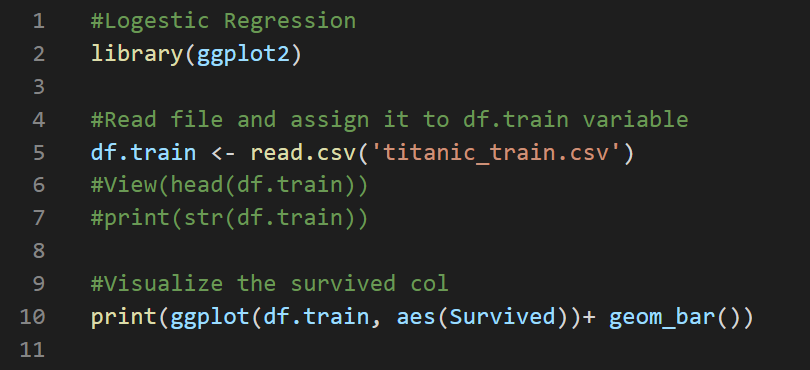
Output: 

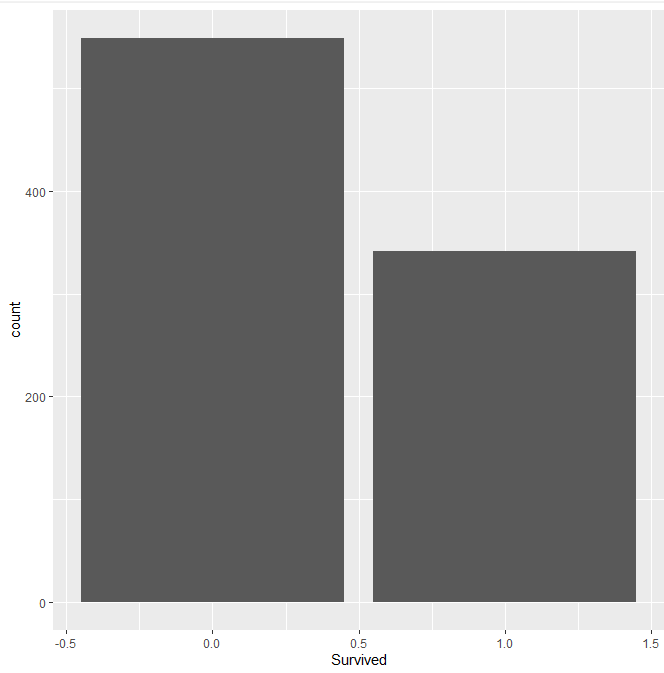


The output shows the columns name of the dataset and the structure.

Let’s continue by exploring the actual data through exploratory data analysis to have a better understanding about the data set.

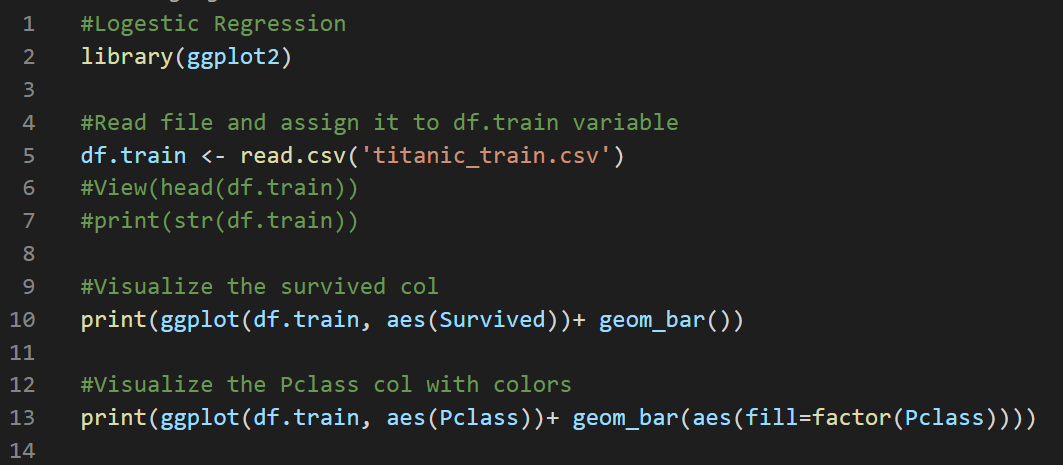
1. First, visualize the Survived columns
   * Type the following to update LogReg.R file

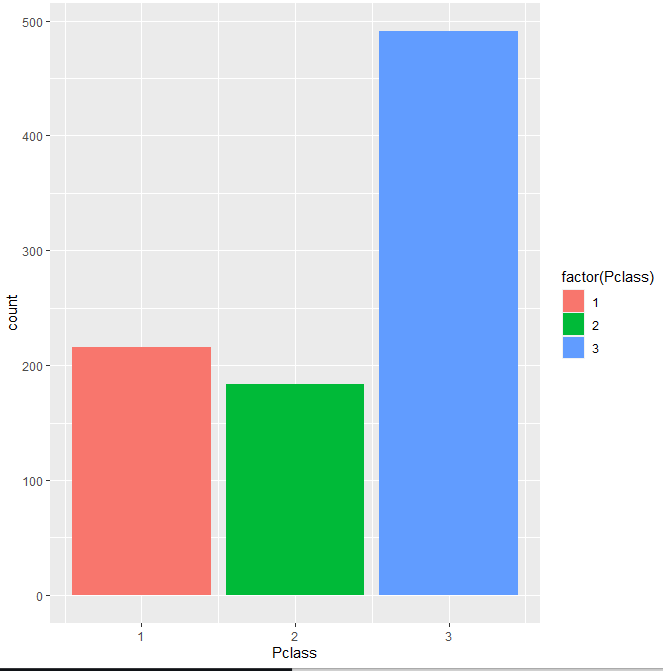


Output: 

The output shows that value of people who not survived and survived. Zero value means the didn’t survived and 1 means they survived.

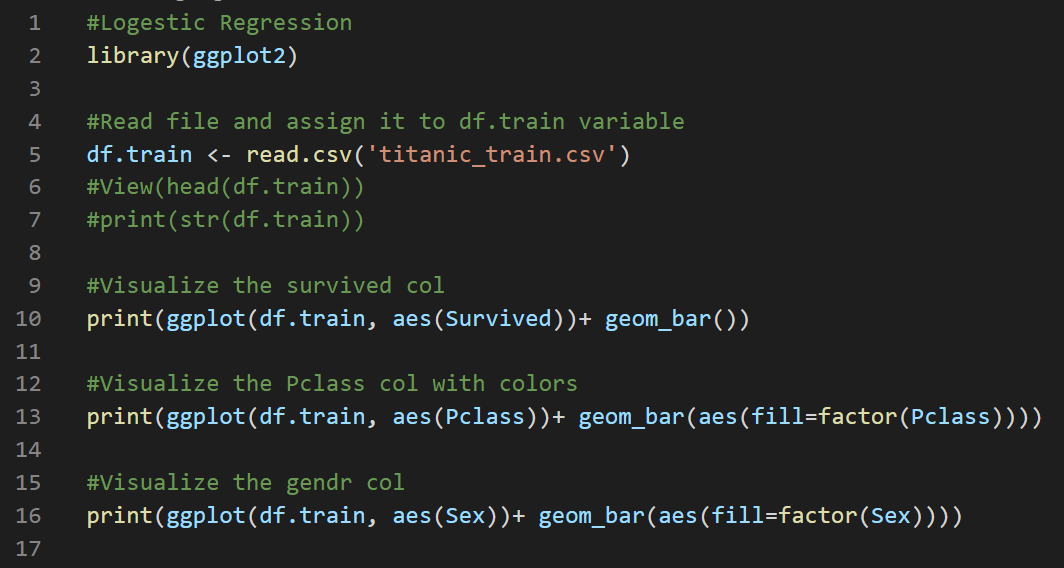
1. Visualize the Pclass. Pclass is the class of the passengers 1,2, & 3.
   * Type the following to update LogReg.R file

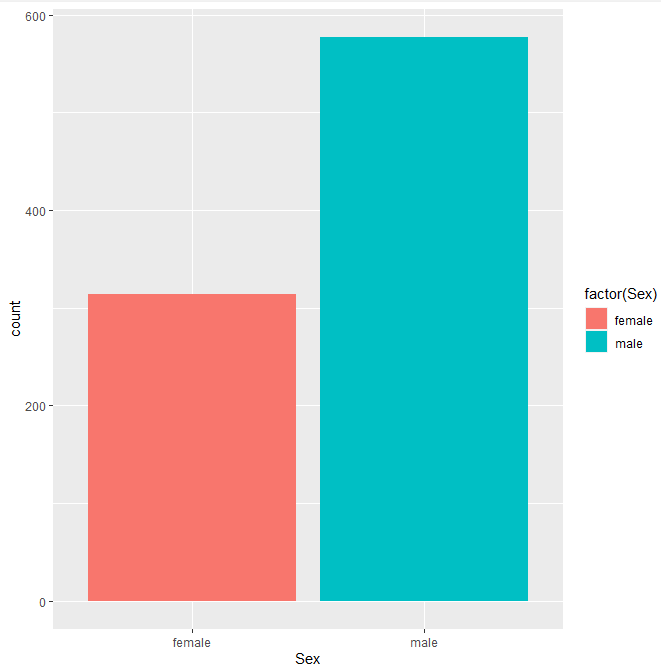


Output: 

The output shows that there was a lot more third-class passengers than either second class passengers or first-class passengers.

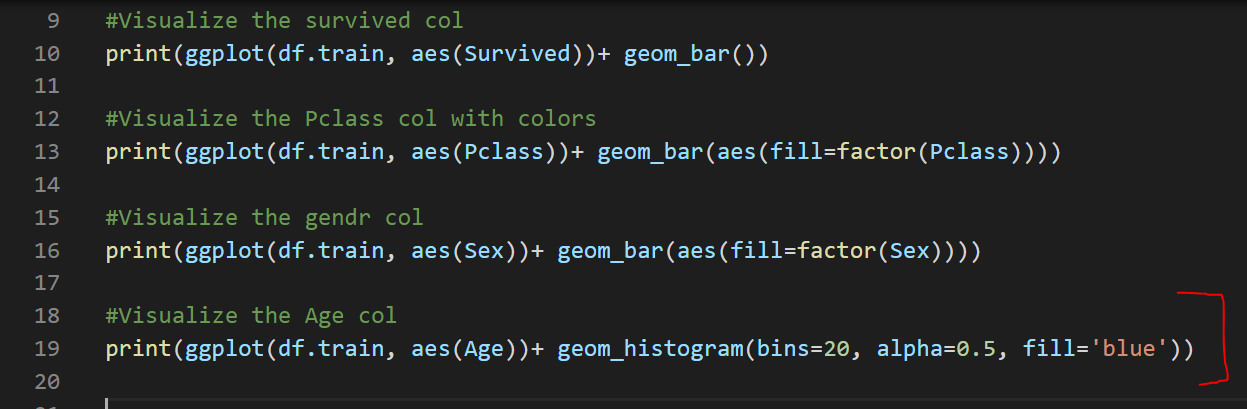
1. Visualize the gender of the passengers to get an idea if there were more men than women on board.
   * Type the following to update LogReg.R file

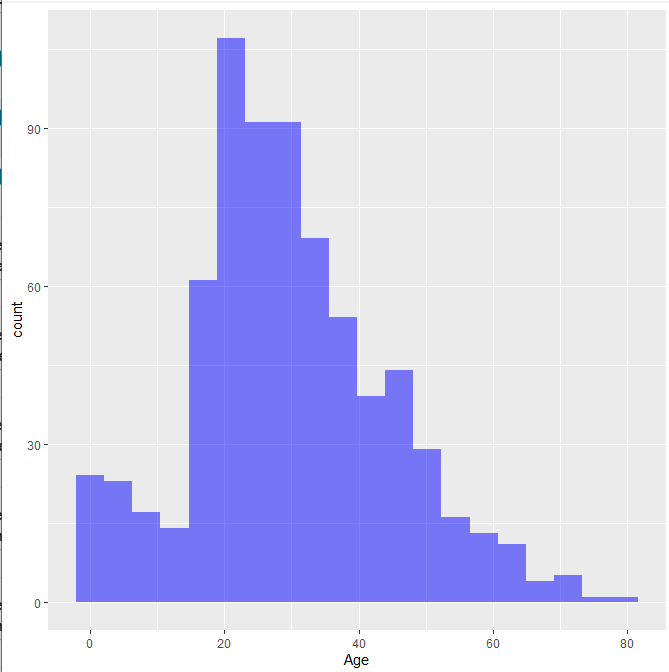


Output: 

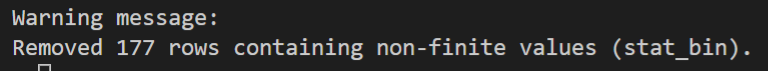
The output shows that there were more male than female on board.

1. Visualize the gender of the passengers
   * Type the following to update LogReg.R file



Output: 

The output shows that the ages of people on board. Also, there is a lot of missing data in the age and in the terminal, it gives you a warning message that there are 177 rows that containing non-finite values

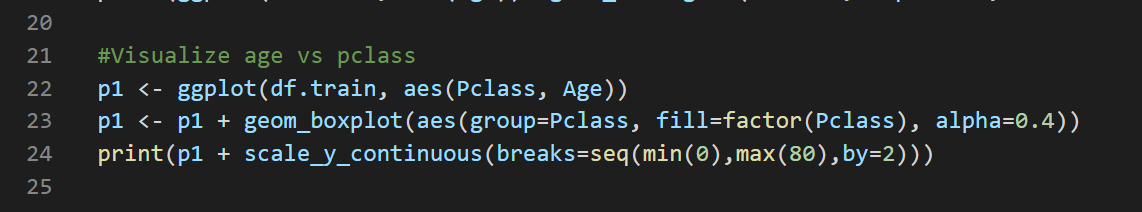


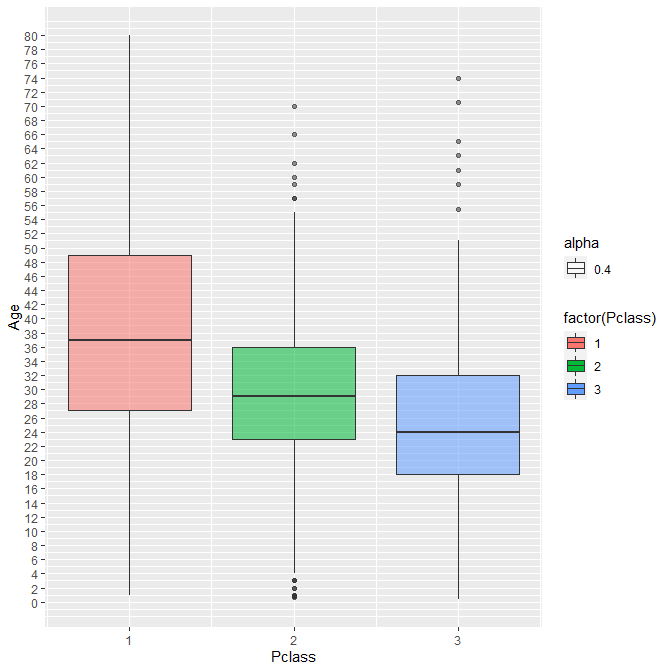
**So, we need to clean our data and fill the NA values.**

**We should make sure that the data is clean and tidy before we run it through our logistic regression model.**

You can clean your data by dropping the rows that contains NA values, however, in our case we have 177 rows of data that was missing out of around 890 total rows. So, it’s a little too much to get rid of all that data. So, we are going to fill in that data through imputation and we can do that by filling in the average age of the passengers by the passenger class.

* + Add the following to update LogReg.R file

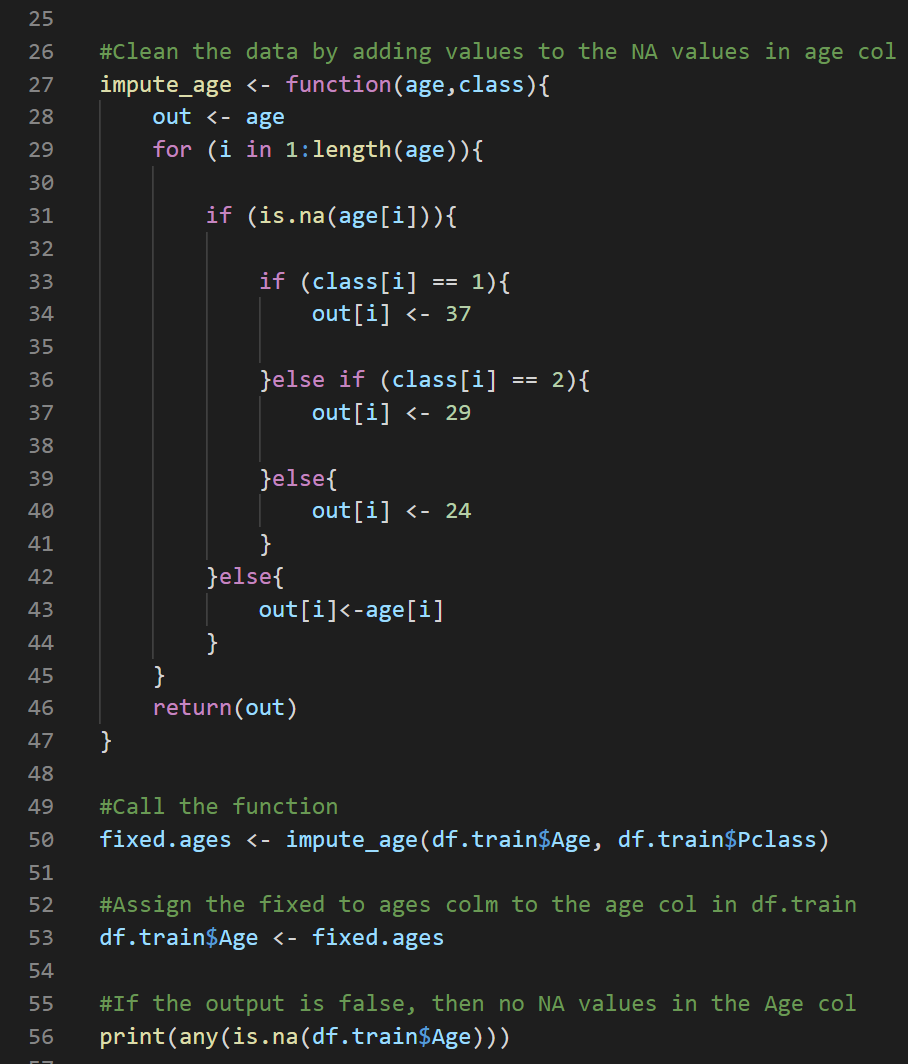


Output: 

The output shows that the average age of passengers of each class. The average age for first class passengers is around 37, so we will set the missing age for first class passengers to 37, for the second class to 29, and third class to 24.

So, we will create a function to do that.

* + Add the following to update LogReg.R file



Output: 

You will still see the warning message because we visualize the Age column before executing the impute\_age function. So, if you try to visualize the age after the function, you will not see the warning message.

Code explanation:

1. We created a function named impute\_age and this function takes two argument age and class. You may consider the age and class as two lists.
2. We created a variable named out which is equal to the age
3. We created a for loop and its length is equal to the age list length
4. For each element in the age list, if there is no value, then we will check the class if the class is equal to 1 then we will assign the age to 37. If the class is 2, then the value will be 24, and if it 3 then the age is 34.
5. If the there is a value not NA, then copy it to the out list.

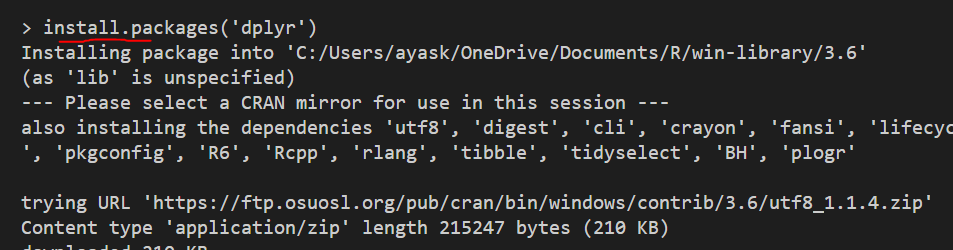
One final cleaning that we should do to the data before build the logistic regression model is removing the columns that we are not going to use which are the passenger ID, Name, ticket, and cabin

To do that, we are going to use **dplyr** package.

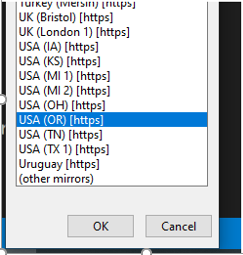
**dplyr** is a powerful R-package to transform and summarize data with rows and columns.The package contains a set of functions (or “verbs”) that perform common data manipulation operations such as filtering for rows, selecting specific columns, re-ordering rows, adding new columns and summarizing data.

**To Install the dyplyr package, type the following in R console**

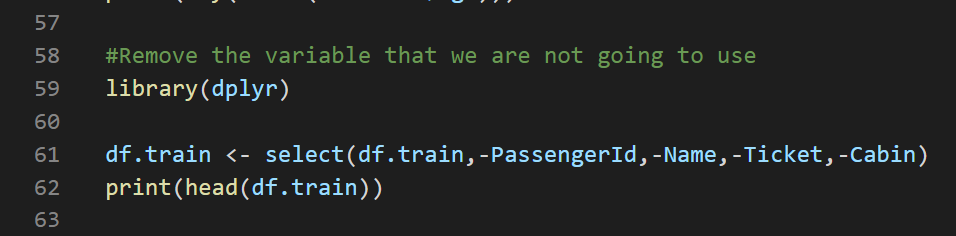
**>install.packages('dplyr')**

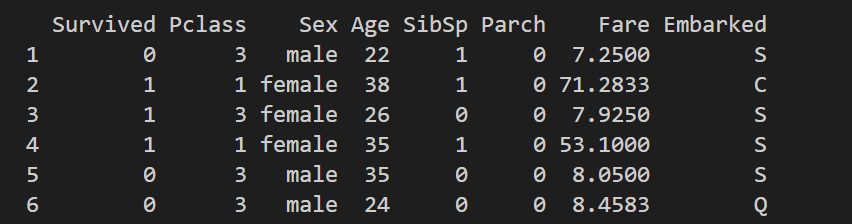


If you have the pop-up window that ask you to choose the CRANS Mirrors to download from, then choose any of the USA and select OK



* + Add the following to update LogReg.R file



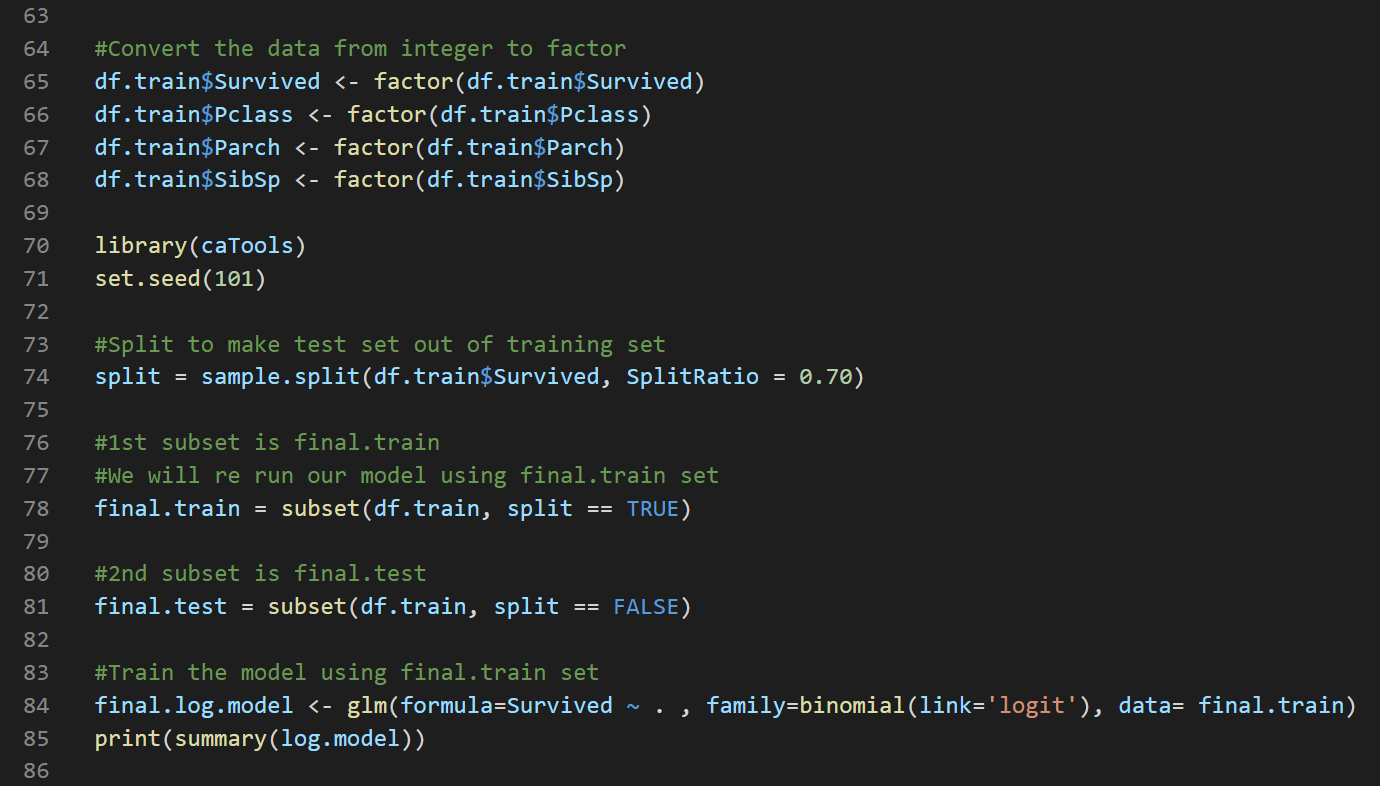
Output: 

**Building a Logistic Regression Model**

**Predicting using Test Cases**

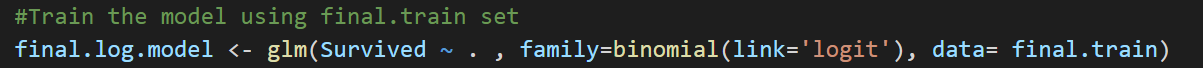
In this part we will make a test set out of our training set, train on the smaller version of our training set and check it against the test subset. We can do this easily using the caTools library

* + Add the following to update LogReg.R file

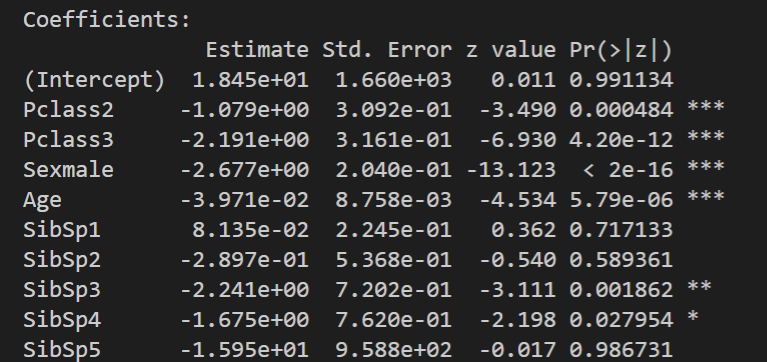


Code Explanation

1. We divided our data set df.train into two parts training data *final.train* and test data *final.test* then we run our logistic regression model.



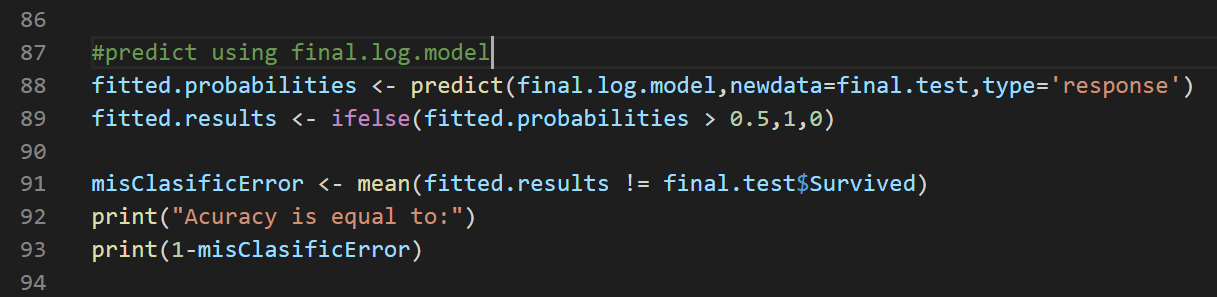
1. In this line we created a variable final.log.model which is equal to the logistic regression model. glm stands to generalized linear model then we pass in the formula and in this case, we want to predict the survived column based of everything or every column in our data frame, so we wrote Survived ~ . then we wrote family which is the description of the error distribution, so we used binomial(link=’logit’), data=final.train and this basically says use the logistic regression function on the final.train data.

Output: 

The number of stars indicate the significant value of the coefficient. As you can see the Pclass, sex and age are important coefficients to survive in our model.

**Check the accuracy of our model**

* + Add the following to update LogReg.R file



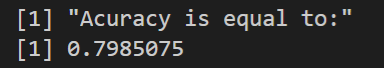
Code explanation

In line 88: in the predict function, you pass in your model which is *final.log.model* , your test data, and type=”response” tells r to output probabilities of the form P(Y=1|X).

In line 89: we created *fitted.result* and it will be equal to 1 if fitted.probabilities >0.5 otherwise it will be equal to 0.

In line91: We calculated the misclassification error which equal to the average of fitted.result that not equal to the survived col in final.test data

In line 93: we printed the accuracy which is equal to 1-misClassifierError

Output: 

The output means that we have around 80% predicted accuracy of our model.

**Push your work to GitHub**

**Make sure you are in**

Onsite students: CS251\_ Fall \_2020/**IN**/FirstnameLastname

Online students: CS251\_ Fall \_2020/**ON**/FirstnameLastname

Run the following commands to push your work to the GitHub repository:

Open the terminal from the VSCode by hit the **control + ~** key and type the following command:

>>> git add .

>>> git commit -m “Submission for Module 8”

>>> git push origin YOUR\_BRANCH\_NAME

Note: you should change the YOUR\_BRANCH\_NAME to your own branch name. It should be firstname-lastname (e.g. maria-gracia).