Exercise 4:
Making Predictions

Image source: https://cmci.colorado.edu/classes/INFO-4604/fa17/

Try in R - Making Predictions

```
# Making predictions

  # "Predicting" the TRAINING data
  pred.train.lin = predict(glm.fit) # No data set is supplied to the predict() function: the probabilities are computed for the training data that was used to fit the logistic regression model.
                                    # Notice: Without the type option specified in predict we get the linear predictor scale (see plot below)
      pred.train.lin.df <- data.frame(balance=training.data$balance,pred.train.lin=pred.train.lin) # make it a data frame fro plotting
      ggplot() + geom_point(data = pred.train.lin.df, aes(x=balance, y=pred.train.lin, col=training.data$default)) + geom_hline(yintercept = 0) + geom_hline(yintercept = 1) + ylim(-15,2) # Plot.

  pred.train.probs = predict(glm.fit, type = "response") # With type = "response", we get the response variable scale, i.e., the probabilities.
      pred.train.probs.df <- data.frame(balance=training.data$balance,pred.train.probs=pred.train.probs) # make it a data frame fro plotting
      ggplot() + geom_point(data = pred.train.probs.df, aes(x=balance, y=pred.train.probs, col=training.data$default)) + geom_hline(yintercept = 0) + geom_hline(yintercept = 1) # Plot.
```

1. Try the code for yourself.
2. Interpret the plots.

Try in R - Making Predictions

```
# Predicting the TEST data PROBABILITIES
pred.test.probs = predict(glm.fit, test.data, type = "response")
    pred.test.probs.df <- data.frame(balance=test.data$balance,pred.test.probs=pred.test.probs) # make it a data frame fro plotting
    ggplot() + geom_point(data = pred.test.probs.df, aes(x=balance, y=pred.test.probs, col=test.data$default), size=5) + geom_hline(yintercept = 0) + geom_hline(yintercept = 1) + geom_hline(yintercept = 0.5, linetype="dashed") + ylim(0,1)

# Predicting the TEST data CLASSES
pred.test.classes = rep("No",nrow(test.data)) # In order to predict the classes, we must convert the predicted into class labels, Yes or No. We start by converting all to No.
pred.test.classes[pred.test.probs > 0.5] = "Yes"  # Now we set those to Yes whose proobability is greater than 0.5.
    pred.test.classes.df <- data.frame(balance=test.data$balance,pred.test.classes=pred.test.classes) # make it a data frame for plotting
    ggplot() + geom_point(data = pred.test.classes.df, aes(x=balance, y=pred.test.classes, col=test.data$default), size=5)


# Confusion matrix
table(test.data$default, pred.test.classes)

# Calculating the validation error rate (percentage of incorrectly classified samples) as an estimate of the test error rate
mean(pred.test.classes != test.data$default)

# Predicting probabilities and classes for a balance of 1000 and 2000 Dollars:
new.data <- data.frame(student = c("No", "No"), balance= c(1000, 2000), income=c(1000, 2000)) # student and income are arbitrarily set, since they will not be used by predict
predict(glm.fit, newdata = new.data, type = "response")
```

1. Try the code for yourself.
2. Interpret the plots of the test data predictions.
3. Interpret the results of the confusion matrix and of test error rate, and compare it with the outputs of your k-NN Models.