Attrition_Stephen Tucker

November 2, 2022

1 People Analytics

1.1 Predicting Employee Attrition with R

In this notebook, we will walk through solving a classification problem. Classification is a type of Machine Learning where we identify which set of categories a person or observation might belong to. We use a set of "training" data containing observations whose category is known and then build a model to make judgements about what classification we might put other people (or instances) into. Classification is an example of pattern recognition, we look for patterns in the data to help us predict what future patterns might be.

1.1.1 Table of Contents

1.1.2 Getting Help

As you go through this assignment, you might have questions or technical errors come up. A few troubleshooting tips can help you solve most Jupyter problems:

1. **re-run your code cells**: if you get an "object '_____' not found" error under a code cell, you may have forgotten to run a cell above it. Try re-running previous cells in order, from top to bottom, or go to the Cell menu and click "Run all" or "Run all above".
2. **try a different browser**: either Firefox or Chrome should work with minimal issues.
3. **restart your notebook kernel**: the "kernel" is the program that runs Jupyter. Restart it by going to the Kernel menu and clicking "Restart". Note: this will make Jupyter "forget" any commands you previously gave it; you will need to re-run previously run code cells.

4. **restart your server**: if all else fails, try clicking "Control Panel" at the top right, then click "Stop my server" followed by "Start my server". Like restarting your kernel, this will require you to re-run any code cells you had already executed.

There are also several resources available to help you at any time:

- **Your classmates**: it's not uncommon for students to enter this course with some familiarity with programming, R, and/or Jupyter.
- **Your course reader/grader**: reach out to Walker Azam via email or Canvas with questions on programming and Jupyter.
- **The internet**: the first thing most professional data analysts do when they run into technical issues is to Google them. Jupyter and R are both widely-used and well-documented- the solution to your problem may already be online!

1.2 Case Study: Employee Attrition at IBM

Employee attrition refers to a decrease in employees at a company, caused by resignations, retirements, or the elimination of job positions. It is important for businesses to anticipate attrition so they can keep costs low and properly distribute workloads.

Today, we'll try to predict whether or not an employee will leave their job using a data set created by IBM. Because employee data is confidential, IBM's data set is fictional- it does not represent real employees. However, the data and the problem are structured very similarly to how actual companies such as IBM are approaching this very real problem. In analytics, the focus of the problem, study, or experiment is often called the response, explanatory, or the dependent variable. In this example, attrition is our response or dependent variable. Any variables that we may use to predict attrition would be our independent variable(s).

```
<b>NOTE</b>: we've made some changes to the data set to make this assignment simpler- things li
</div>
```

Run the following cell to load the R-packages we will use to analyze the data. These packages provide a starting point that will allow us to process data without having to write code from scratch.

```
<b>NOTE</b>: To run a code cell, you can either press the play button at the top bar of the no
</div>
```

[2]:
```
# Run this cell to install the necessary software
install.packages("fmsb")
install.packages("corrplot")
library(corrplot)
library(ggplot2)
library(plyr)
library(fmsb)
```

```
Updating HTML index of packages in '.Library'

Making 'packages.html' …
 done
```

```
Updating HTML index of packages in '.Library'

Making 'packages.html' …
 done
```

1.3 The Dataset

Run the following cell to load the data.

<b>NOTE</b>: To make sure the data is read, the name of the data file must be the same as the
</div>

[5]:
```r
# load the dataset
hr = read.csv("HR-Employee-Attrition.csv")

# show the first 6 rows of the dataset
head(hr)
```

A data.frame: 6 × 35

	Age	Attrition	BusinessTravel	DailyRate	Department	Dista
	<int>	<chr>	<chr>	<int>	<chr>	<int
1	41	Yes	Travel_Rarely	1102	Sales	1
2	49	No	Travel_Frequently	279	Research & Development	8
3	37	Yes	Travel_Rarely	1373	Research & Development	2
4	33	No	Travel_Frequently	1392	Research & Development	3
5	27	No	Travel_Rarely	591	Research & Development	2
6	32	No	Travel_Frequently	1005	Research & Development	2

This data set contains a wealth of information that could be used to predict attrition. In fact, there is so much information that the rows and columns have been truncated in the notebook.

We can see the dimensions of the dataset (number of rows and columns, respectively) with the `dim` function.

[6]:
```r
# show the size of the dataset
dim(hr)
```

1. 1470 2. 35

The first number is the number of rows and it shows how many employees are available on the dataset. The second number is the number of columns, which indicates how many variables we have data on for these employees. We can view all of the column names by using the `names` function.

[7]:
```r
# view all column names
names(hr)
```

1. 'Age' 2. 'Attrition' 3. 'BusinessTravel' 4. 'DailyRate' 5. 'Department' 6. 'DistanceFromHome' 7. 'Education' 8. 'EducationField' 9. 'EmployeeCount' 10. 'EmployeeNumber' 11. 'EnvironmentSatisfaction' 12. 'Gender' 13. 'HourlyRate' 14. 'JobInvolvement' 15. 'JobLevel' 16. 'JobRole' 17. 'JobSatisfaction' 18. 'MaritalStatus' 19. 'MonthlyIncome' 20. 'MonthlyRate' 21. 'NumCompaniesWorked' 22. 'Over18' 23. 'OverTime' 24. 'PercentSalaryHike' 25. 'PerformanceRating'

26. 'RelationshipSatisfaction' 27. 'StandardHours' 28. 'StockOptionLevel' 29. 'TotalWorkingYears' 30. 'TrainingTimesLastYear' 31. 'WorkLifeBalance' 32. 'YearsAtCompany' 33. 'YearsInCurrentRole' 34. 'YearsSinceLastPromotion' 35. 'YearsWithCurrManager'

The source website included the **data dictionary**: the information about each feature (column) contained in the data set. We've reproduced the data dictionary here.

- `Age`: employee's age
- `Attrition`: the **response variable**. 'Yes' if employee left IBM due to normal life circumstances, 'No' if the employee stayed with IBM
- `BusinessTravel`: the amount of travel the employee does for business. 0 (no travel), 1 (rare travel), or 2 (frequent travel)
- `Department`: employee's home department. 'Sales', 'Research & Development', or 'Human Resources'
- `DistanceFromHome`: number of miles the employee has to travel to work
- `Education`: employee's highest level of education. 1 (below college), 2 (some college), 3 (bachelor's degree), 4 (master's degree), 5 (doctorate)
- `Gender`: 1 (female) or 0 (male)
- `JobRole`: 'Sales Executive', 'Research Scientist', 'Laboratory Technician', 'Manufacturing Director', 'Healthcare Representative', 'Manager', 'Sales Representative', 'Research Director', or 'Human Resources'
- `JobSatisfaction`: the employee's ranking of their satisfaction with their job. Minimum is 1, maximum is 4.
- `MaritalStatus`: 'Single', 'Married', or 'Divorced'
- `MonthlyIncome`: in dollars
- `NumCompaniesWorked`: the total number of companies the employee has worked at in their lifetime
- `OverTime`: whether or not the employee is eligible for overtime. 1 (yes) or 0 (no)
- `PercentSalaryHike`: for the employee's most recent salary increase, the percentage by which their salary increased
- `WorkLifeBalance`: employee's rating of their work-life balance. 1 (bad), 2 (good), 3 (better), 4 (best)
- `YearsAtCompany`: number of years employee has worked for IBM
- `YearsInCurrentRole`: number of years employee has worked for IBM in their current position

You saw in the data that our response variable for attrition used the words 'Yes' or 'No' to indicate whether the employee left the organization. In order to conduct our classification exercise we will need this data to be in the form of numbers. In fact, in the type of analysis we will be using today (logistic regression), we can only use variables that are numeric in value. So, we will need to 'transform' the attrition data into numbers and will only select the other variables that contain numbers.

In the following cell we will add a new column (`AttritionInt`) to the dataset: the Attrition variable stored as an integer (1 if the employee left IBM, 0 if the employee stayed). This will be the actual response variable our models will use to predict attrition.

[10]:
```
# add a data frame column with Attrition coded as integers: 1 for "Yes" and 0␣
↪for "No" i.e binary
# this will allow us to make plots
```

```
hr$AttritionInt = as.numeric(as.factor(hr$Attrition)) - 1

# select only columns with numeric data
numeric_data = hr[,sapply(hr,is.numeric)]
head(numeric_data)
```

A data.frame: 6 × 27

	Age <int>	DailyRate <int>	DistanceFromHome <int>	Education <int>	EmployeeCount <int>	EmployeeNu <int>
1	41	1102	1	2	1	1
2	49	279	8	1	1	2
3	37	1373	2	2	1	4
4	33	1392	3	4	1	5
5	27	591	2	1	1	7
6	32	1005	2	2	1	8

1.3.1 Exploratory Analysis

Exploratory Data Analysis (EDA) can help us get a sense of what data each column contains.

The `summary` function displays summary statistics for each column. For **quantitative variables**, `summary` will show the minimum, maximum, mean, and quartiles. For **categorical variables**, `summary` shows the possible values and frequencies for each.

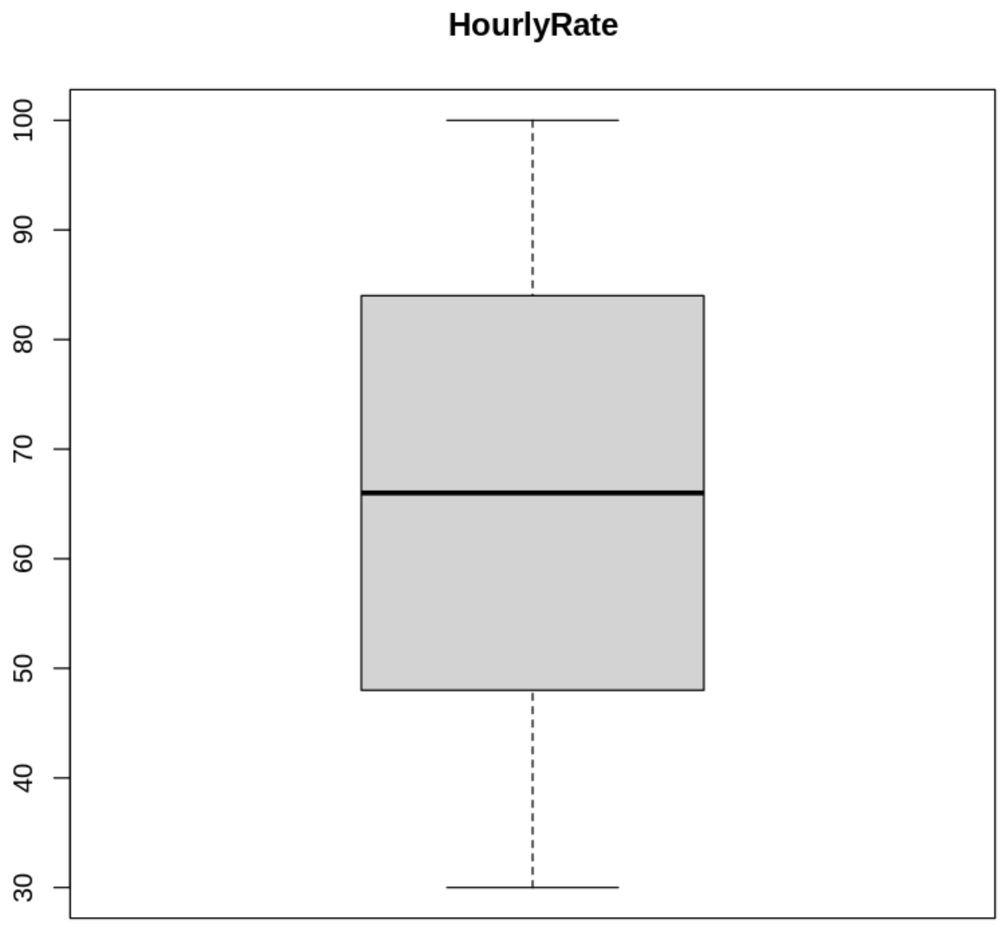
[11]:
```
# show summary statistics for the data
summary(hr)
```

```
     Age         Attrition         BusinessTravel       DailyRate
Min.   :18.00   Length:1470        Length:1470        Min.   : 102.0
1st Qu.:30.00   Class  :character  Class  :character  1st Qu.: 465.0
Median :36.00   Mode   :character  Mode   :character  Median : 802.0
Mean   :36.92                                         Mean   : 802.5
3rd Qu.:43.00                                         3rd Qu.:1157.0
Max.   :60.00                                         Max.   :1499.0
 Department        DistanceFromHome   Education     EducationField
Length:1470        Min.   : 1.000   Min.   :1.000   Length:1470
Class  :character  1st Qu.: 2.000   1st Qu.:2.000   Class  :character
Mode   :character  Median : 7.000   Median :3.000   Mode   :character
                   Mean   : 9.193   Mean   :2.913
                   3rd Qu.:14.000   3rd Qu.:4.000
                   Max.   :29.000   Max.   :5.000
EmployeeCount EmployeeNumber   EnvironmentSatisfaction    Gender
Min.   :1     Min.   :   1.0   Min.   :1.000           Length:1470
1st Qu.:1     1st Qu.: 491.2   1st Qu.:2.000           Class  :character
Median :1     Median :1020.5   Median :3.000           Mode   :character
Mean   :1     Mean   :1024.9   Mean   :2.722
3rd Qu.:1     3rd Qu.:1555.8   3rd Qu.:4.000
Max.   :1     Max.   :2068.0   Max.   :4.000
  HourlyRate     JobInvolvement     JobLevel       JobRole
Min.   : 30.00   Min.   :1.00   Min.   :1.000   Length:1470
```

```
1st Qu.: 48.00   1st Qu.:2.00   1st Qu.:1.000   Class :character
Median : 66.00   Median :3.00   Median :2.000   Mode  :character
Mean   : 65.89   Mean   :2.73   Mean   :2.064
3rd Qu.: 83.75   3rd Qu.:3.00   3rd Qu.:3.000
Max.   :100.00   Max.   :4.00   Max.   :5.000
JobSatisfaction MaritalStatus      MonthlyIncome    MonthlyRate
Min.   :1.000   Length:1470        Min.   : 1009   Min.   : 2094
1st Qu.:2.000   Class :character   1st Qu.: 2911   1st Qu.: 8047
Median :3.000   Mode  :character   Median : 4919   Median :14236
Mean   :2.729                      Mean   : 6503   Mean   :14313
3rd Qu.:4.000                      3rd Qu.: 8379   3rd Qu.:20462
Max.   :4.000                      Max.   :19999   Max.   :26999
NumCompaniesWorked    Over18            OverTime         PercentSalaryHike
Min.   :0.000      Length:1470        Length:1470        Min.   :11.00
1st Qu.:1.000      Class :character   Class :character   1st Qu.:12.00
Median :2.000      Mode  :character   Mode  :character   Median :14.00
Mean   :2.693                                            Mean   :15.21
3rd Qu.:4.000                                            3rd Qu.:18.00
Max.   :9.000                                            Max.   :25.00
PerformanceRating RelationshipSatisfaction StandardHours StockOptionLevel
Min.   :3.000     Min.   :1.000            Min.   :80    Min.   :0.0000
1st Qu.:3.000     1st Qu.:2.000            1st Qu.:80    1st Qu.:0.0000
Median :3.000     Median :3.000            Median :80    Median :1.0000
Mean   :3.154     Mean   :2.712            Mean   :80    Mean   :0.7939
3rd Qu.:3.000     3rd Qu.:4.000            3rd Qu.:80    3rd Qu.:1.0000
Max.   :4.000     Max.   :4.000            Max.   :80    Max.   :3.0000
TotalWorkingYears TrainingTimesLastYear WorkLifeBalance YearsAtCompany
Min.   : 0.00     Min.   :0.000         Min.   :1.000   Min.   : 0.000
1st Qu.: 6.00     1st Qu.:2.000         1st Qu.:2.000   1st Qu.: 3.000
Median :10.00     Median :3.000         Median :3.000   Median : 5.000
Mean   :11.28     Mean   :2.799         Mean   :2.761   Mean   : 7.008
3rd Qu.:15.00     3rd Qu.:3.000         3rd Qu.:3.000   3rd Qu.: 9.000
Max.   :40.00     Max.   :6.000         Max.   :4.000   Max.   :40.000
YearsInCurrentRole YearsSinceLastPromotion YearsWithCurrManager
Min.   : 0.000     Min.   : 0.000          Min.   : 0.000
1st Qu.: 2.000     1st Qu.: 0.000          1st Qu.: 2.000
Median : 3.000     Median : 1.000          Median : 3.000
Mean   : 4.229     Mean   : 2.188          Mean   : 4.123
3rd Qu.: 7.000     3rd Qu.: 3.000          3rd Qu.: 7.000
Max.   :18.000     Max.   :15.000          Max.   :17.000
 AttritionInt
Min.   :0.0000
1st Qu.:0.0000
Median :0.0000
Mean   :0.1612
3rd Qu.:0.0000
Max.   :1.0000
```

Data visualization can also be helpful to find relationships. **Box plots** show the distribution of values within a particular column of numerical data through their quartiles (thin horizontal lines on the next graph). The spacings between the different parts of the box indicate the degree of dispersion in the data. Outliers may be plotted as individual points.

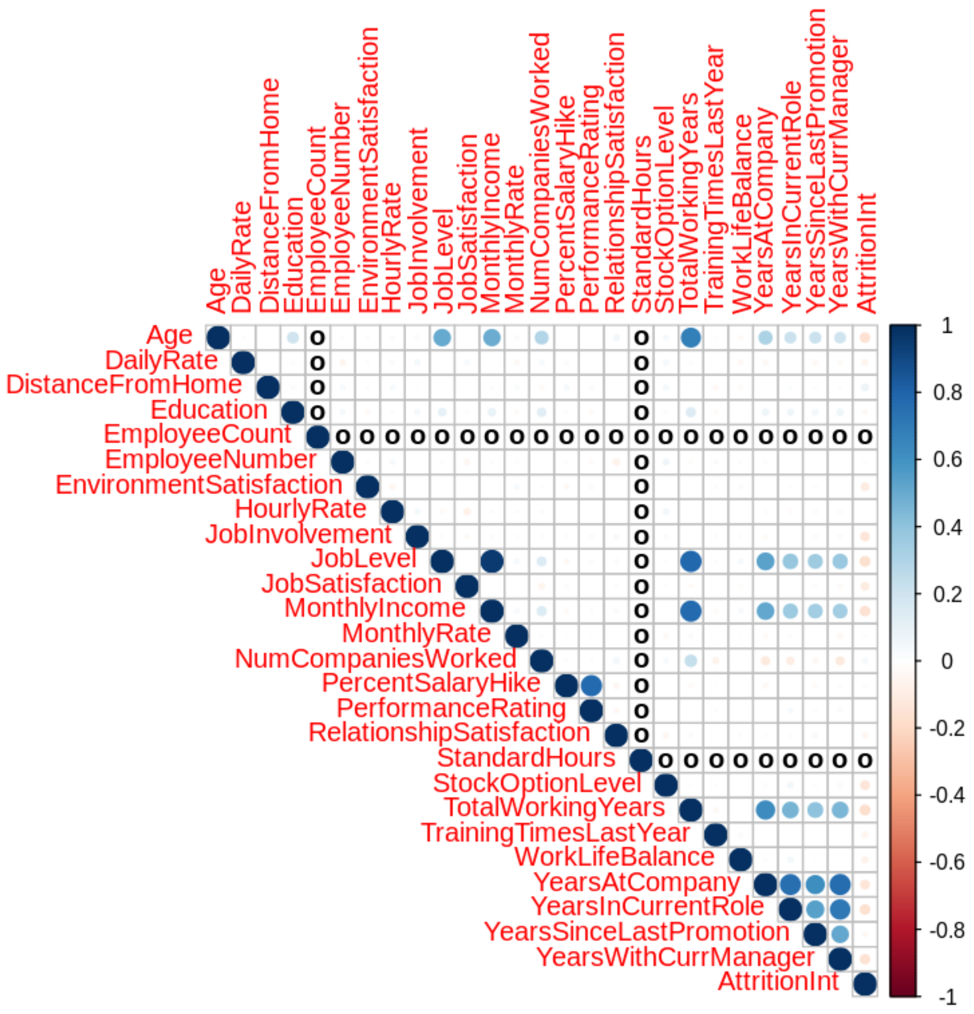
[13]:
```
# replace the ... with the name of the column you want to visualize
# capitalization counts!
my_column = "HourlyRate"

# run this cell to see the distribution of values for your column
boxplot(hr[,my_column],main=my_column)
```



Correlation plots show the strength of linear relationships between variables. Each cell on the plot shows the correlation between 2 variables: positive correlations are shown in blue and negative correlations are shown in red. The intensity of the color shows the strength of the correlation. Each element on the diagonal is the correlation of a variable with itself, which is always equal to 1. In this case we will use circles to highlight variables that have no correlation.

[15]:
```
# create a correlation plot for quantitative variables
correlations = cor(numeric_data)
corrplot(correlations, method="circle",type='upper',na.label = "o")
```

```
Warning message in cor(numeric_data):
"the standard deviation is zero"
```



```
<b>QUESTION</b>: Which variables do you think will be most useful to predict attrition, and wha
</div>
```

The variables which have will be the most useful to predict attrition are based on the data: years at company, years in current role, job level,total working hours, monthly income. Some possible souces of bias that may exist in the data are outliers, which can skew the mean,

1.4 Logistic Regression

Logistic Regression is used to model the probability associated to binary events, such as win/lose, pass/fail or attrition/no attrition. Logistic regressions take numeric variables as input and build a statistical model by expressing the dependent variable as a linear combination of the independent variables using the exponential function. In the following example, we will use a logistic model to calculate the probability of attrition for an employee.

1.4.1 Example 1: predict by monthly income rate (univariate logistic regression)

Let's try predicting attrition based on the employee's monthly income.

The first task is to write out the formula and save it as a variable. In this example, the name of the formula variable is `univar_formula`. The formula itself is to the right of the `=` and has the syntax:

$$\text{response variable} \sim \text{predictor variable}$$

That is, the name of the variable we want to predict is on the left of the tilde, and the name of the variable we're using to predict attrition is on the right.

[16]:
```
# write the formula for univariate logistic regression
univar_formula = AttritionInt ~ MonthlyIncome
```

R provides a function called `glm` (for Generalized Linear Model) that will fit the data. The function takes three arguments: * the formula (symbolic description of the model) * `data` = the name of the data frame that holds our data * `family` = a description of the error distribution and link function to be used in the model. For logistic regression, we want `binomial`

Run the following cell to fit the model.

[17]:
```
# fit the model
logreg_single = glm(univar_formula,
              data = hr, # the dataset
              family = binomial) # the type of model to use
```

The `summary` function displays some information about the fit model. We will focus on the coefficients associated with the intercept and the independent variable. Similarly to the linear regression case, the intercept allows us to calculate the expected value of the probability when the independent variables are equal to zero and the coefficient of the independent variable shows us how much the outcome probability changes when the independent variable changes. The `summary` provides the following information: * **Estimate**: value of the coefficient obtained after training the model. * **Standard Error**: summarizes the deviation between the predicted values and the real observations. It can be used to build confidence intervals for the predictions. * **Z value**: normalized value used to run a hypothesis and check if there is a statistically significant relationship between the independent and dependent variable. The significance (next bullet) will allow us to check this relationship more directly. * **Significance (Pr(>|z|))**: obtained by checking the Z value on a standard normal table. If this value is below 5%, then we can affirm we are 95% confident that the coefficient of the independent variable is different from zero (i.e. there is a relationship between dependent and independent variables).

[18]:
```
# show some summary information about the fit model
summary(logreg_single)$coeff
```

A matrix: 2 × 4 of type dbl

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-0.9291087486	0.1292021393	-7.191125	6.425956e-13
MonthlyIncome	-0.0001271042	0.0000216188	-5.879336	4.119147e-09

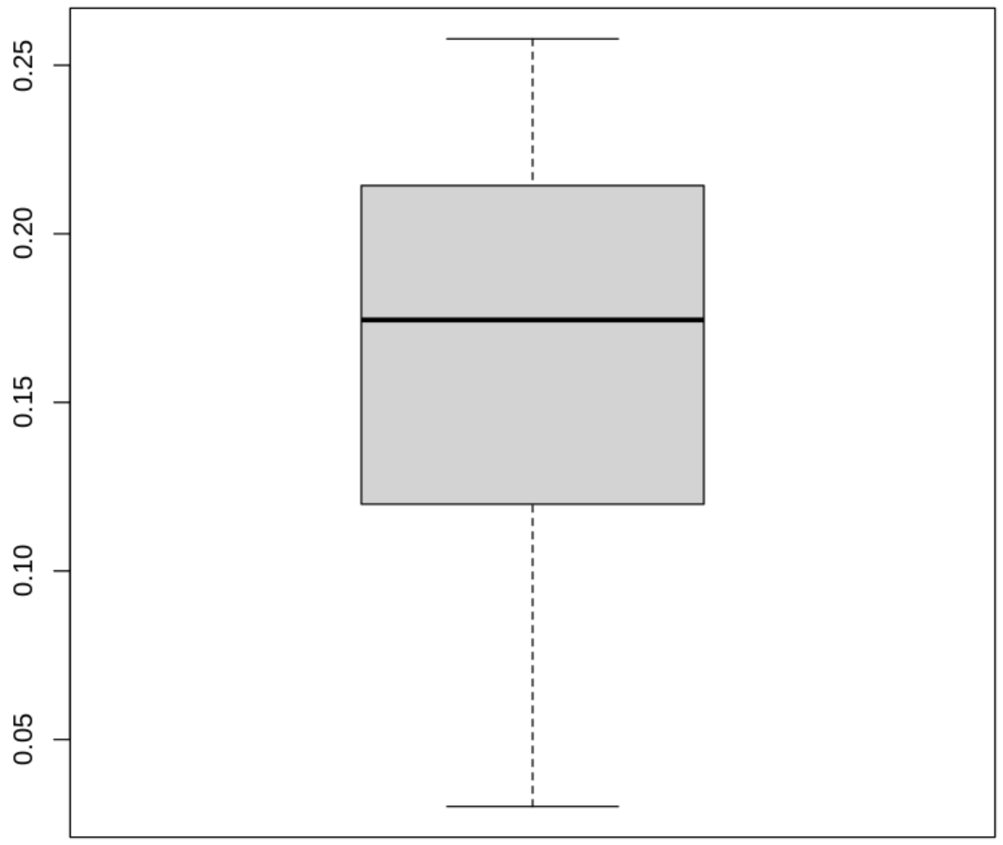
The `predict` function makes predictions using our fitted model. It has two arguments: * the fitted logistic regression model * `type` = the format of our predictions. By selecting "response", we will get back a probability between 0 (i.e. no predicted chance of attrition) and 1 (100% predicted chance of attrition)

[20]:
```
# use the model to predict attrition
single_probs = predict(logreg_single, hr, type = "response")

# show the probabilities of attrition for the first six employees in the dataset
head(single_probs)
```

1 0.155664874420095 **2** 0.170632232501251 **3** 0.232409326214141 **4** 0.214357819747282 **5** 0.202635033357612 **6** 0.21097400561898

We can view the range of predicted probabilities with a box plot.

[21]:
```
# show the range of predictions
boxplot(single_probs)
```



Having a predicted probability is interesting, but often we want our predictions to be in the same form as our original `Attrition` variable: either "Yes" or "No" values. This allows us to directly measure the accuracy of our model by comparing our predictions with the true outcomes.

There are a few ways to transform our probabilities into "Yes"/"No" values:

- We might decide that any probablity above a certain threshold will be coded as "Yes". For example, if the predicted probability is at least 0.24, we would predict attrition.
- If we believe that a certain proportion of staff are likely to leave the company, we might decide to code the top *n* percent of predictions as "Yes". For example, if we know that last year we lost 5% of employees to attrition, we could predict attrition this year for the 5% of employees with the highest predicted probability.

```
<b>QUESTION</b>: What method do you think is best for interpreting the probabilities, and why?
</div>
```

I believe they work in unison they being the boxplot and the confusion matrix. I like the Box plot becaus it shows the min - max - mean values as well as the majority. Where the confusion matrix gives us more striaghtforward information in a cumlative format to which is helpful as well.

In this example, we'll try coding any employee with a probability of at least 0.22 as likely to leave.

[22]:
```
# create a categorical variable: "Yes" if probability of attrition is > 0.22,␣
↪"No" otherwise
single_preds = ifelse(single_probs > 0.22, "Yes", "No")

# show the first six items in the probability and prediction vectors for our␣
↪single variable model
head(data.frame(single_probs, single_preds))
```

A data.frame: 6 × 2

	single_probs <dbl>	single_preds <chr>
1	0.1556649	No
2	0.1706322	No
3	0.2324093	Yes
4	0.2143578	No
5	0.2026350	No
6	0.2109740	No

One way to check our accuracy is to calculate how many predictions were correct out of the total number of predictions. This will take two steps: 1. for each prediction, check if it equals the actual value (`single_preds == hr[, "Attrition"]`) 2. add up the number of correct predictions and divide it by the total predictions (`mean`)

The resulting number is the proportion of correct predictions.

[24]:
```
# show the proportion of correct predictions
mean(single_preds == hr[, "Attrition"])
```

0.769387755102041

It can also be useful to see in what ways our model is correct or incorrect. We can visualize this with a **confusion matrix**. Each row shows the instances of the predicted attrition, and each column shows the instances of the actual attrition. The matrix values are the counts. So: - the value in row one, column one is the *true negatives*: the number of employees who were predicted to stay at the company and actually did stay - the value in row one, column two is the *false negatives*: number of employees who were predicted to stay and actually left - the value in row two, column one is the *false positives*: employees who were predicted to leave but didn't - the value in row two, column two is the *true positives*: employees who were predicted to leave and did

[23]:
```
# create a confusion matrix for our predictions (rows) vs the actual values␣
↪(columns)
table(single_preds, hr[, "Attrition"])
```

```
single_preds   No  Yes
```

```
No   1045  151
Yes   188   86
```

Another metric often used to evaluate how good the model fits the data is the R-squared (R^2), which summarizes the proportion of variance in the dependent variable associated with the independent variables. Larger R-squared values indicate that more of the variation is explained by the model, to a maximum of 1. R-squared is generally used on continuous data, however there is a modified version of R-squared (Nagelkerke) that we can use to evaluate our binary dependent variable in the logistic regression model.

[24]:
```
NagelkerkeR2(logreg_single)$R2
```

0.0519408366804252

```
<b>QUESTION</b>: How would you rate the accuracy of this model? Would you recommend this model
```

I would say that this is a poor fitting model because it only captures 0.052 of the variance of the data Therefore it does not capture much which means another model may be more appropriate.

1.4.2 Example 2: Predict By Monthly Income, Work-Life Balance, and Years at Company (multivariate logistic regression)

Perhaps we could do better if we used more predictor variables. In this example, we'll attempt to predict attrition based on the employee's monthly income, work-life balance, and number of years with the company.

The nice thing about using a programming language like R is that the large majority of the code used for the univariate model will work for a multivariate model. The main thing that needs to change is the initial formula fed into the model.

As before, our first step is to write the formula and give it a name. Here, the name of the formula variable is `multivar_formula`, and the additional predictor variables are added using `+`. So, for a formula with n predictor variables, the formula would look like this:

$$\text{response variable} \sim \text{predictor variable}_1 + \text{predictor variable}_2 + ... + \text{predictor variable}_n$$

[28]:
```
multi_formula = AttritionInt ~ MonthlyIncome + WorkLifeBalance + YearsAtCompany
```

The rest of the steps are almost identical to our single variable model:

1. **fit the model** to the data using the multivariate logistic regression formula
2. use the fitted model to **predict** attrition for each employee
3. **evaluate the model** by looking at summary data, the proportion of correct predictions, the confusion matrix, R-squared and plots of the results

[29]:
```
# fit the model
logreg_multi = glm(multi_formula,
              data = hr, # the dataset
              family = binomial) # the type of model to use

# use the model to predict attrition
```

```
multi_probs = predict(logreg_multi, hr, type = "response")

# create a categorical variable: "Yes" if probability of attrition is > 0.22,␣
↪"No" otherwise
multi_preds = ifelse(multi_probs > 0.22, "Yes", "No")
```

[30]:
```
# show the proportion of correct predictions
mean(multi_preds == hr[, "Attrition"])
```

0.75578231292517

[32]:
```
NagelkerkeR2(logreg_multi)$R2
```

0.0664375895144758

`<b>QUESTION</b>: Why wouldn't we want to just use all the variables as predictors? </div>`

not all the variables apply to the question your answering. and having to many prdictors can make a bad model because it will be to broad of focus.

1.5 4. Make Your Own Model

Now that you've seen some examples, try making and evaluating your own model. Most of the code has been provided for you (you might notice that it looks extremely similar to the univariate model). However, you will need to do the following:

1. Choose at least one variable as a predictor (and justify your choice)
2. Run the cells to train the model and generate probablities
3. Choose a method for interpreting attrition probabilities (and justify your choice)
4. Run the cells to generate model metrics and visualizations, and analyze your model

1.5.1 Write the formula

Start by writing the formula for your logistic regression model.

Remember, the syntax for the formula is:

$$\text{response variable} \sim \text{predictor variable}_1 + \text{predictor variable}_2 + ... + \text{predictor variable}_n$$

where the predictor variables are the names of the columns you want to use (spelling and capitalization count!)

You may want to revisit the Exploratory Analysis section to refresh your memory on the values and relationships of the possible predictor variables. You can also quickly see all possible variable names by running the following cell:

[ ]:
```
# Run this cell if you want a list of all possible variable names
EnvironmentSatisfaction
```

```
JobInvolvement
YearsInCurrentRole
YearsAtCompany
YearsSinceLastPromotion
YearsWithCurrManager
```

<b>QUESTION</b>: Complete the formula with your chosen predictor variables.
</div>

[32]:
```
# write the formula for your logistic regression model by replacing the ...
formula_1 = AttritionInt ~ YearsInCurrentRole + YearsWithCurrManager +␣
 ↪YearsAtCompany
```

<b>QUESTION</b>: Which predictor variables did you choose, and why?
</div>

I choose Years in Current Role, Years with Current manager, and Years at company The reasoon i chose these three are that they had shown up in the corr plot as well as alot of people say people dont quit jobs they quit managers so i felt that would be appropriate to variable to analyze.

1.5.2 Fit the model

Once your formula is complete, run the following cell to fit your model.

If you get an error, double-check that you've run the cell that contains your formula, and that all the predictor variable names are spelled correctly.

[33]:
```
# fit the model
logreg_mine = glm(formula_1,
              data = hr, # the dataset
              family = binomial) # the type of model to use
summary(logreg_mine)
```

```
Call:
glm(formula = formula_1, family = binomial, data = hr)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.7998  -0.6622  -0.5662  -0.3916   2.6040

Coefficients:
                     Estimate Std. Error z value Pr(>|z|)
(Intercept)          -1.05705    0.10877  -9.719  < 2e-16 ***
YearsInCurrentRole   -0.10219    0.03738  -2.733  0.00627 **
YearsWithCurrManager -0.08579    0.03820  -2.246  0.02472 *
YearsAtCompany        0.01355    0.02431   0.557  0.57725
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1298.6  on 1469  degrees of freedom
Residual deviance: 1250.2  on 1466  degrees of freedom
AIC: 1258.2

Number of Fisher Scoring iterations: 5
```

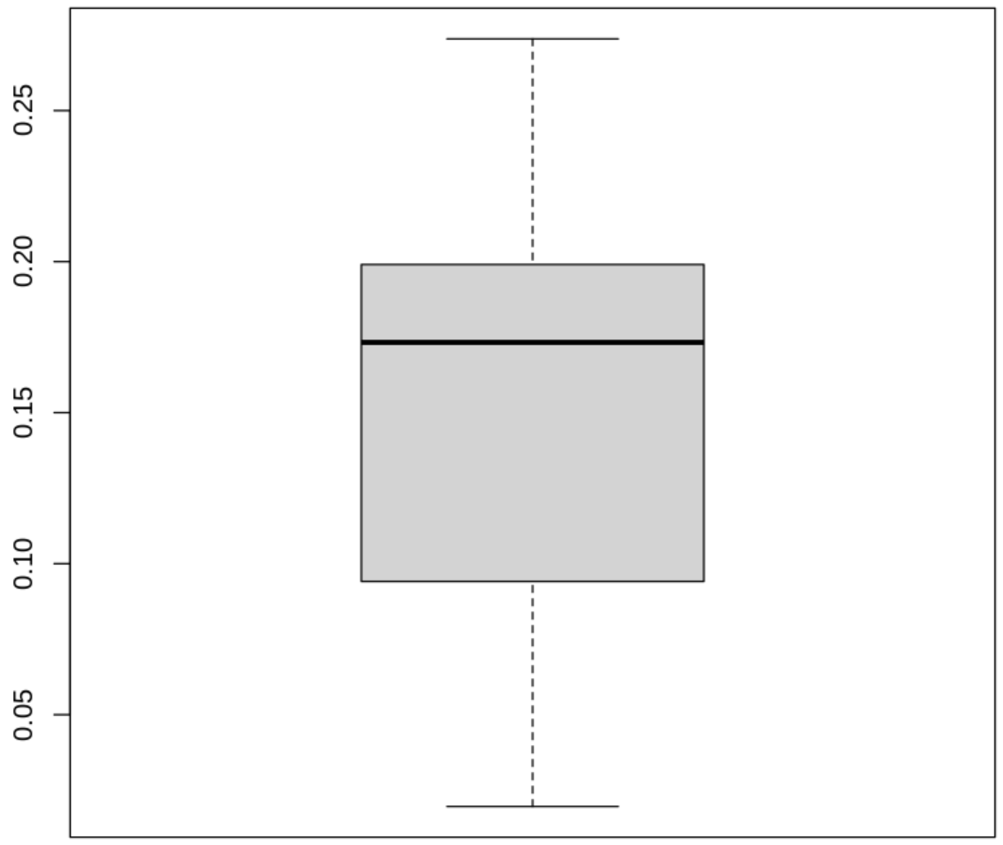
1.5.3 Make predictions

[34]:
```
# use the model to predict attrition
my_probs = predict(logreg_mine, hr, type = "response")

# show the probabilities of attrition for the first six employees in the dataset
head(my_probs)
```

1 0.140220485052553 **2** 0.0964419670819599 **3** 0.257872653590623 **4** 0.159233796608387 **5** 0.196879580333782 **6** 0.100447598229003

View the range of predicted probabilities with a box plot.

[35]:
```
# show the range of predictions
boxplot(my_probs)
```



```
<b>QUESTION</b>: Decide how to turn your probabilities into "Yes"/"No" predictions.
<ul>
    <li> If you want to use the threshold method, set the <code>method</code> variable to "thre
    <li> If you want to use the percentile method, set the <code>method</code> variable to "per
</ul>
</div>
```

[41]:
```
# replace the ... with either "threshold" or "percentile"
method = 0.25

# if you're using the threshold method, replace the ... with the
# cutoff probability (a number between 0 and 1)
cutoff = 0.4
```

```
# if you're using the percentile method, replace the ... with
# the dividing percentile (a number between 0 and 1)
percentile = 0.95

# create a categorical variable: "Yes" if probability of attrition is > 0.23,␣
↪"Yes" otherwise
if (method == 0.25) {
    my_preds = ifelse(my_probs > cutoff, "Yes", "No")
} else if (method == 0.95) {
    pct_cutoff = quantile(my_probs, percentile)
    my_preds = ifelse(my_probs > pct_cutoff, "Yes", "No")
} else {
    stop("method variable must be either \"threshold\" or \"percentile\"")
}

# show the first six items in the probability and prediction vectors for our␣
↪single variable model
head(data.frame(my_probs, my_preds))
tail(data.frame(my_probs, my_preds))
```

A data.frame: 6 × 2

	my_probs	my_preds
	<dbl>	<chr>
1	0.14022049	No
2	0.09644197	No
3	0.25787265	No
4	0.15923380	No
5	0.19687958	No
6	0.10044760	No

A data.frame: 6 × 2

	my_probs	my_preds
	<dbl>	<chr>
1465	0.23019364	No
1466	0.18984000	No
1467	0.09295674	No
1468	0.19193313	No
1469	0.09668966	No
1470	0.18527733	No

[42]:
```
# show some summary information about the fit model
summary(logreg_mine)
```

```
Call:
glm(formula = formula_1, family = binomial, data = hr)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
```

```
-0.7998  -0.6622  -0.5662  -0.3916   2.6040

Coefficients:
                     Estimate Std. Error z value Pr(>|z|)
(Intercept)          -1.05705    0.10877  -9.719  < 2e-16 ***
YearsInCurrentRole   -0.10219    0.03738  -2.733  0.00627 **
YearsWithCurrManager -0.08579    0.03820  -2.246  0.02472 *
YearsAtCompany        0.01355    0.02431   0.557  0.57725
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1298.6  on 1469  degrees of freedom
Residual deviance: 1250.2  on 1466  degrees of freedom
AIC: 1258.2

Number of Fisher Scoring iterations: 5
```

Calculate the proportion of correct predictions.

[43]:
```
# show the proportion of correct predictions
mean(my_preds == hr[, "Attrition"])
```

0.838775510204082

Create the confusion matrix. Remember, the format is:

[44]:
```
# create a confusion matrix for our predictions (rows) vs the actual values␣
↪(columns)
table(my_preds, hr[, "Attrition"])
```

```
my_preds   No  Yes
      No 1233  237
```

[45]:
```
NagelkerkeR2(logreg_mine)$R2
```

0.0551416108583594

```
<b>QUESTION</b>: How would you rate the accuracy of this model? Would you recommend this model
</div>
```

The accuracy of the model is poor because it doesnt capture enough data withing the model to be reliable. I would judge my model on R2 because you want to make sure you capture enough data and if i have an decent R2 then i can trust my results are not skewed or faulty. I like Example 2 because it appeared to capture more varience in the model.

1.6 Submitting your assignment

After you have finished the notebook, you will download it as a pdf file and upload it to the Canvas. To download notebooks as pdfs:

1. Make sure you've saved your work by clicking the floppy disc icon at the top left or going to the "File" menu and clicking "Save and Checkpoint".
2. Once you've saved your notebook, go to the "File" menu at the top left of the notebook screen.
3. Click "Save and Export Notebook As…", then click "PDF".
4. Depending on how your internet browser is set up to handle PDF files, your download may start automatically or you may be asked where you want to save the file.

Once the file is downloaded to your computer, you may upload it to Canvas.

NOTE: Make sure to check that your answers appear in the pdf file (you can quickly find the places where you had to fill in text by using control+F (Windows) or command + F (Mac) to search for "Question"). If your answers don't appear, you may not have saved your notebook before downloading. If you run into other issues, don't hesitate to ask your classmates or Walker for help!

Notebook created by Heather Whiteman, Walker Azam, Keeley Takimoto and Ignacio Solis

Some text adapted with permission from materials made for Haas Executive Education's Data Science Online course by Keeley Takimoto.