COM662
Data Analytics

Dr Lu Bai
l.bai@ulster.ac.uk

Week 6 – Linear and Logistic Regression

ulster.ac.uk

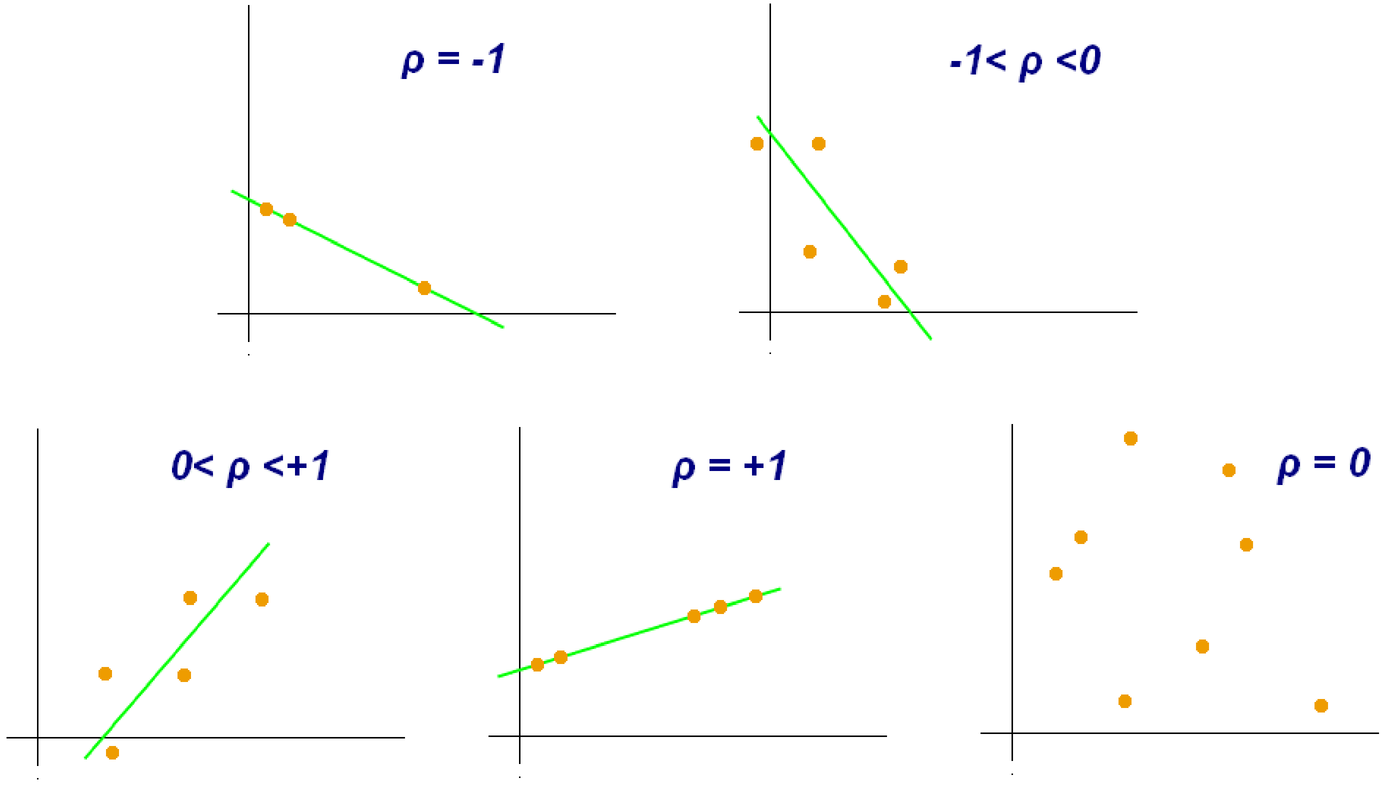
Week 6
Content

- Review correlation
- Linear regression
- Multiple regression
- Model fitness and significance
- Using regression as a predictive model
- Evaluating a linear regression model
- Logistic regression
- Application of logistic regression
  - Odds ratios
  - Binary classification/prediction

Ulster University

Correlation / association

- Correlation
  - Strength of association between variables



https://en.wikipedia.org/wiki/Pearson_correlation_coefficient

Regression
Definition

- Regression models the relationship between one or many independent variables (features) with one dependent/target variable.

- Typically used to model continuous variables

- Multiple regression vs. simple linear regression

https://en.wikipedia.org/wiki/Linear_regression

Regression

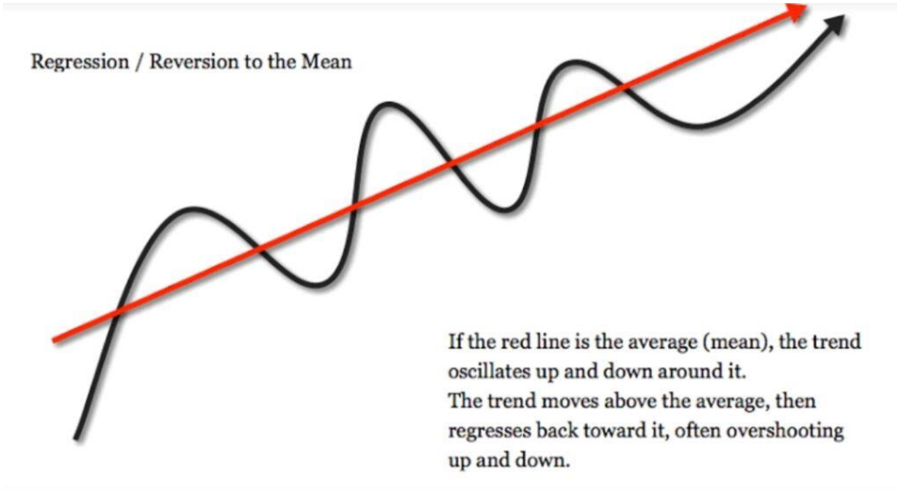
Regression to the Mean

- Heights of parents vs. heights of their children

"It appeared from these experiments that the offspring did not tend to resemble their parents in size, but always to be more mediocre than they – to be smaller than the parents, if the parents were large; to be larger than the parents, if the parents were small."

Sir Francis Galton

https://select-statistics.co.uk/blog/regression-to-the-mean-as-relevant-today-as-it-was-in-the-1900s/

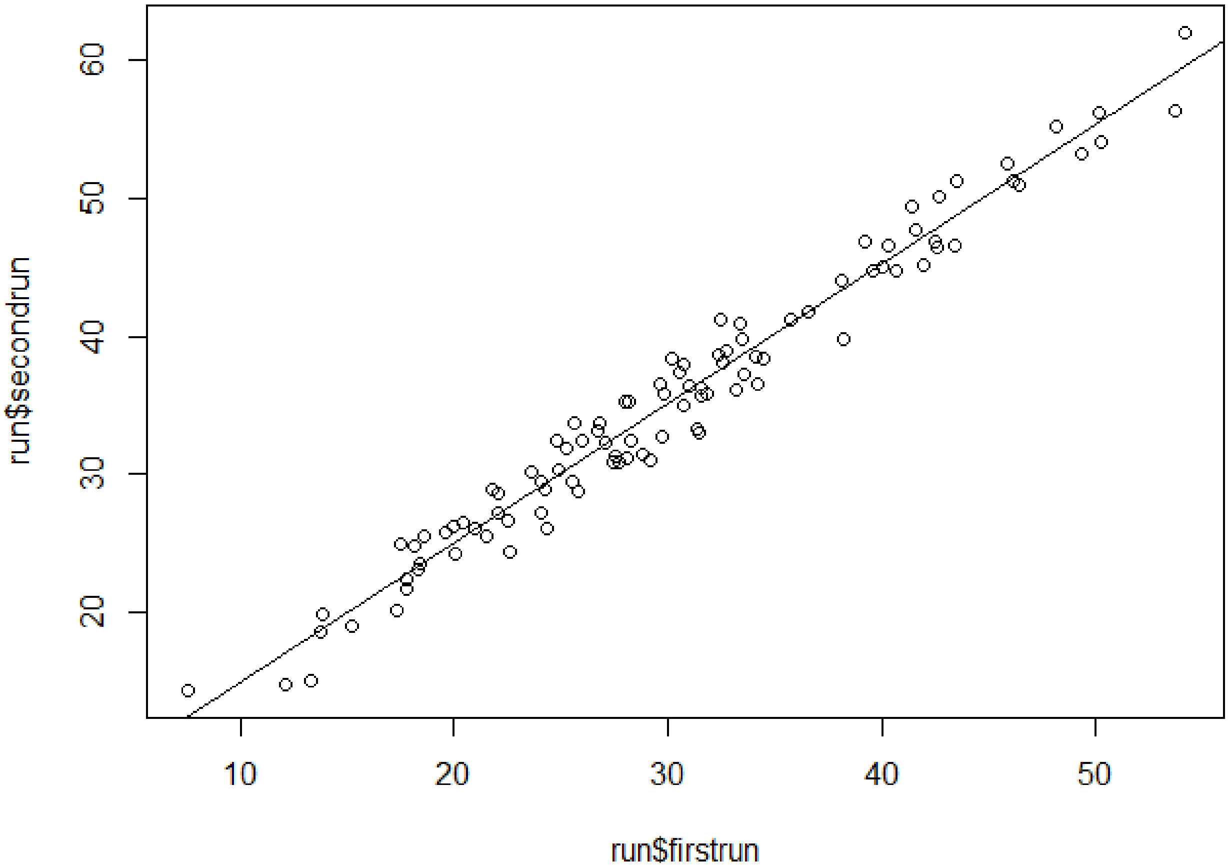


https://ridiculousdata.com/dc/regression-to-the-mean/

Ulster University

Regression

Simple Linear Regression



$$\hat{y} = b_0 + b_1 x$$

Regression

Multiple Linear Regression

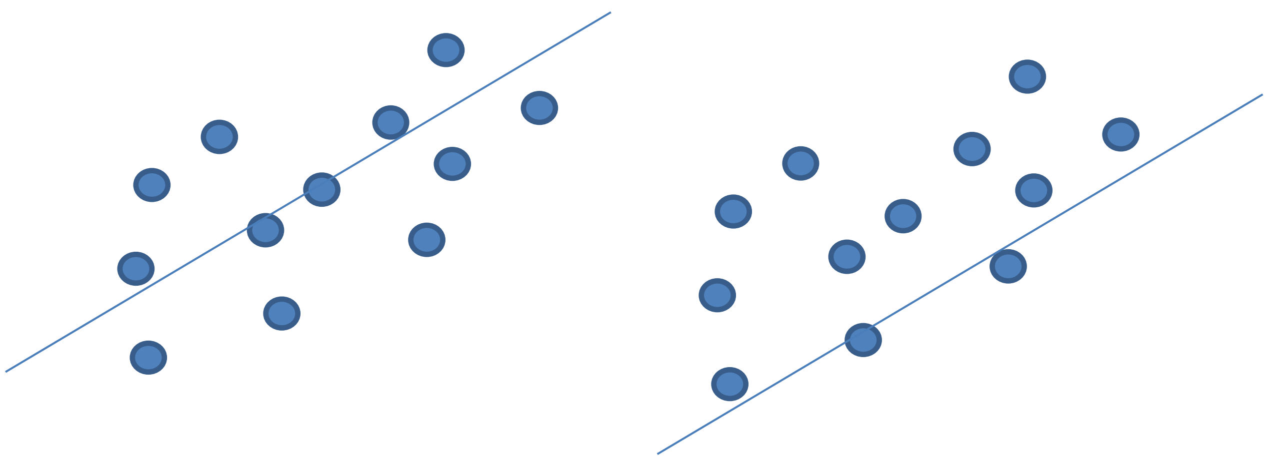
$$\hat{y} = bX + cY + dZ + a$$

$$\hat{y} = \sum_{i=1}^{n} b_i \cdot X_i + b_0$$

Regression

Ordinary Least Squares Regression

- Finds the regression line that has the least squared error between what is predicted and the actual values.
- OLS minimizes the sum of the squared **residuals**.
- It finds b_0 and b_1.

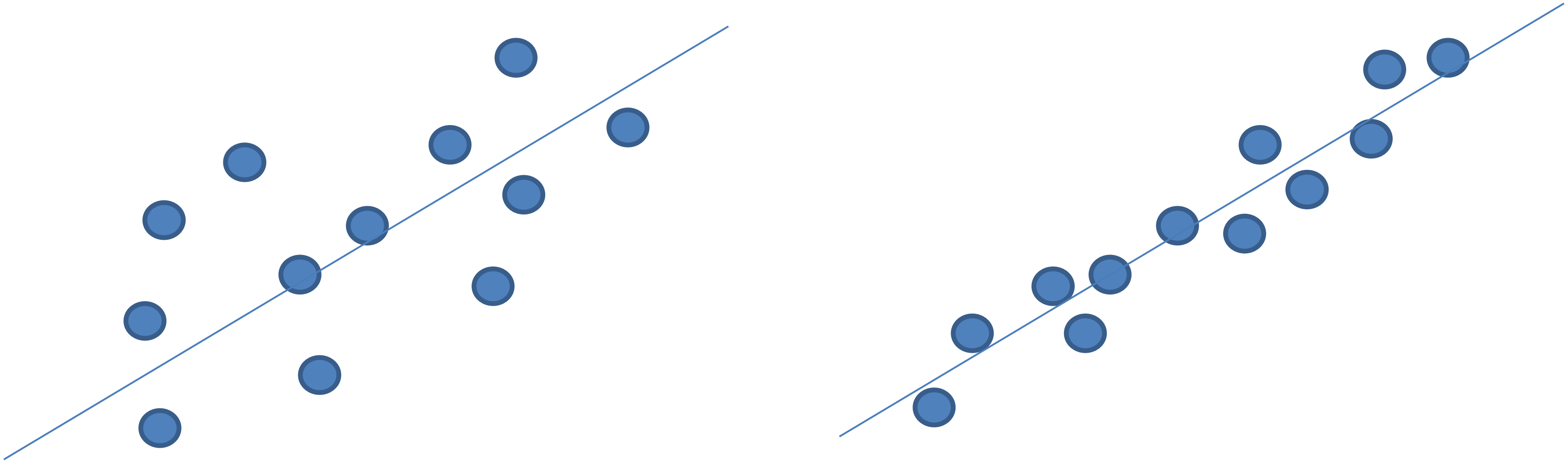


$$\hat{\beta} = \frac{\sum x_i y_i - \frac{1}{n}\sum x_i \sum y_i}{\sum x_i^2 - \frac{1}{n}(\sum x_i)^2} = \frac{\mathrm{Cov}[x, y]}{\mathrm{Var}[x]}$$

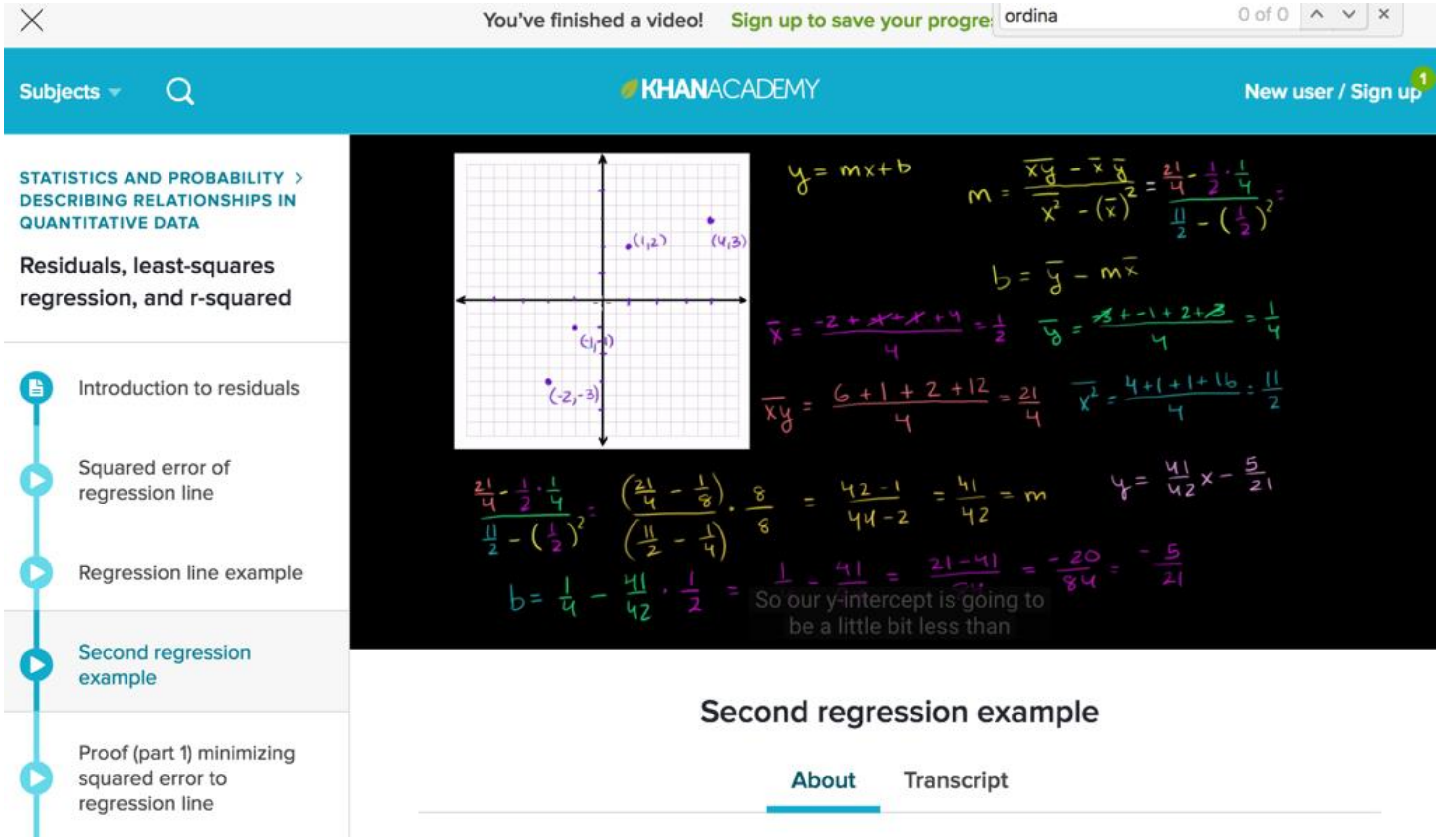
$$\hat{\alpha} = \overline{y} - \hat{\beta}\,\overline{x}$$

Regression
Goodness of Fit

- Definition: How the regression line fits the observed data



Regression
Exercise – watch online video



https://www.khanacademy.org/math/statistics-probability/describing-relationships-quantitative-data/residuals-least-squares-rsquared/v/second-regression-example

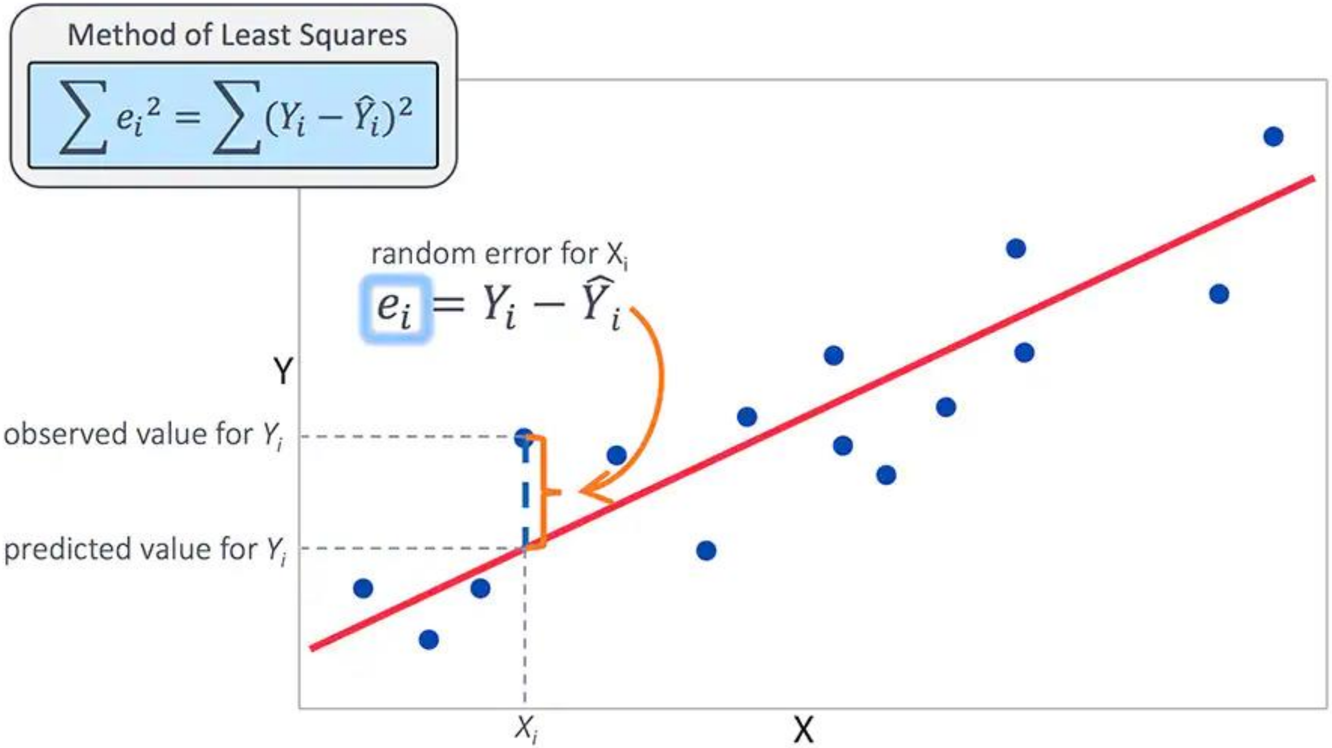
Ulster University

Regression
Goodness of Fit

- How the regression line fits the observed data
- **R Squared** (Coefficient of determination)
  - R-squared is the % of the total variation which the model explains
  - R-squared is always between 0 and 100% or 0 and 1
  - R-squared is a statistical measure of how close the data are to the fitted regression line.

http://blog.minitab.com/blog/adventures-in-statistics/regression-analysis-how-do-i-interpret-r-squared-and-assess-the-goodness-of-fit

Regression
Goodness of Fit - Residuals absolute error



https://www.jmp.com/en_nl/statistics-knowledge-portal/what-is-multiple-regression/fitting-multiple-regression-model.html

Regression

Statistical Significance in Regression - F Test

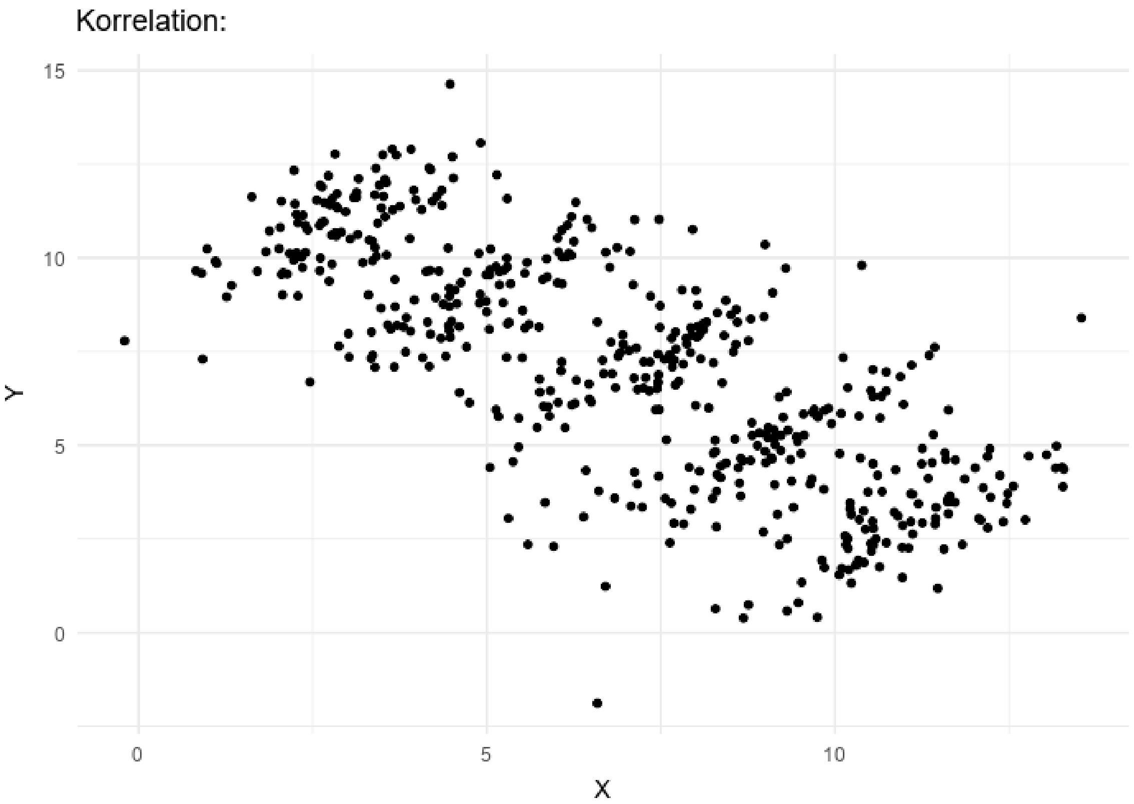
- Compares a model with no predictors (intercept-only model) to your model.

  - “**Null hypothesis**: The fit of the intercept-only model and your model are equal.
  - **Alternative hypothesis**: The fit of the intercept-only model is significantly reduced compared to your model.”

http://blog.minitab.com/blog/adventures-in-statistics/what-is-the-f-test-of-overall-significance-in-regression-analysis

Regression

Simpsons Paradox



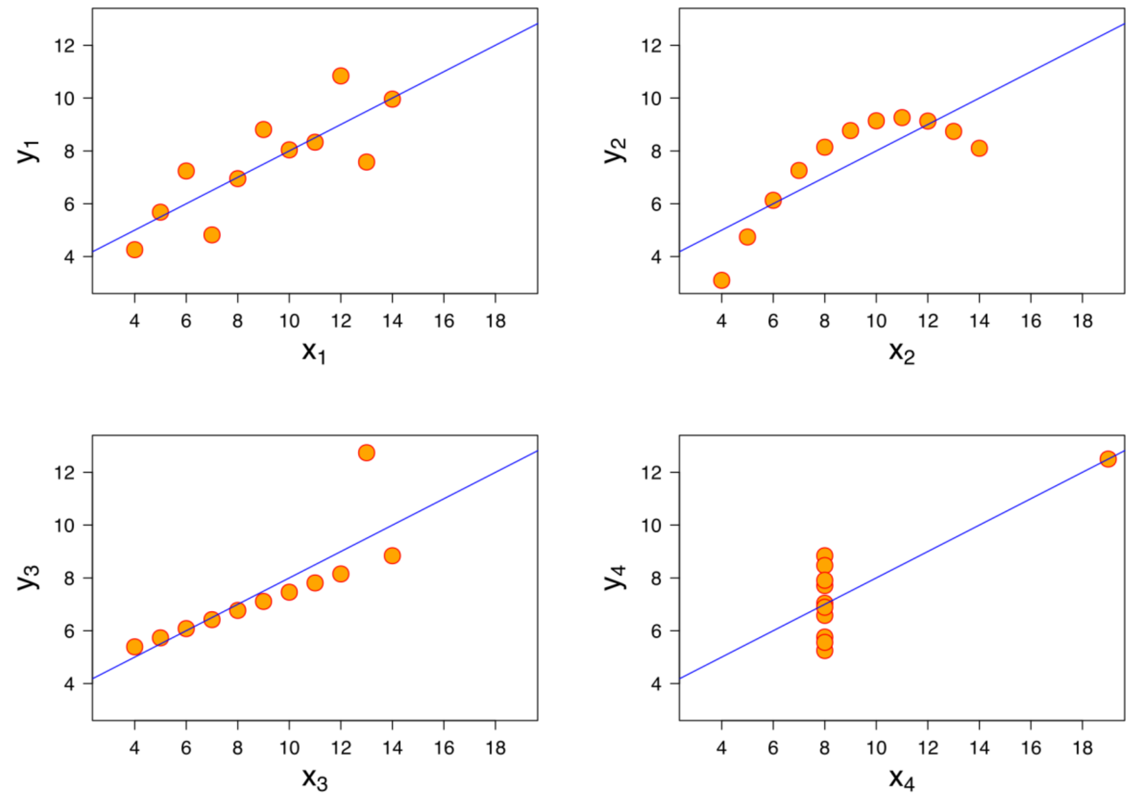
https://en.wikipedia.org/wiki/Simpson%27s_paradox

Regression

Anscombe's quartet

Anscombe's quartet

I		II		III		IV	
x	*y*	*x*	*y*	*x*	*y*	*x*	*y*
10.0	8.04	10.0	9.14	10.0	7.46	8.0	6.58
8.0	6.95	8.0	8.14	8.0	6.77	8.0	5.76
13.0	7.58	13.0	8.74	13.0	12.74	8.0	7.71
9.0	8.81	9.0	8.77	9.0	7.11	8.0	8.84
11.0	8.33	11.0	9.26	11.0	7.81	8.0	8.47
14.0	9.96	14.0	8.10	14.0	8.84	8.0	7.04
6.0	7.24	6.0	6.13	6.0	6.08	8.0	5.25
4.0	4.26	4.0	3.10	4.0	5.39	19.0	12.50
12.0	10.84	12.0	9.13	12.0	8.15	8.0	5.56
7.0	4.82	7.0	7.26	7.0	6.42	8.0	7.91
5.0	5.68	5.0	4.74	5.0	5.73	8.0	6.89



Anscombe, F. J. (1973). "Graphs in Statistical Analysis". American Statistician. 27 (1): 17–21. doi:10.1080/00031305.1973.10478966. JSTOR 2682899.

https://en.wikipedia.org/wiki/Anscombe%27s_quartet

Ulster University

Regression

Segmented regression

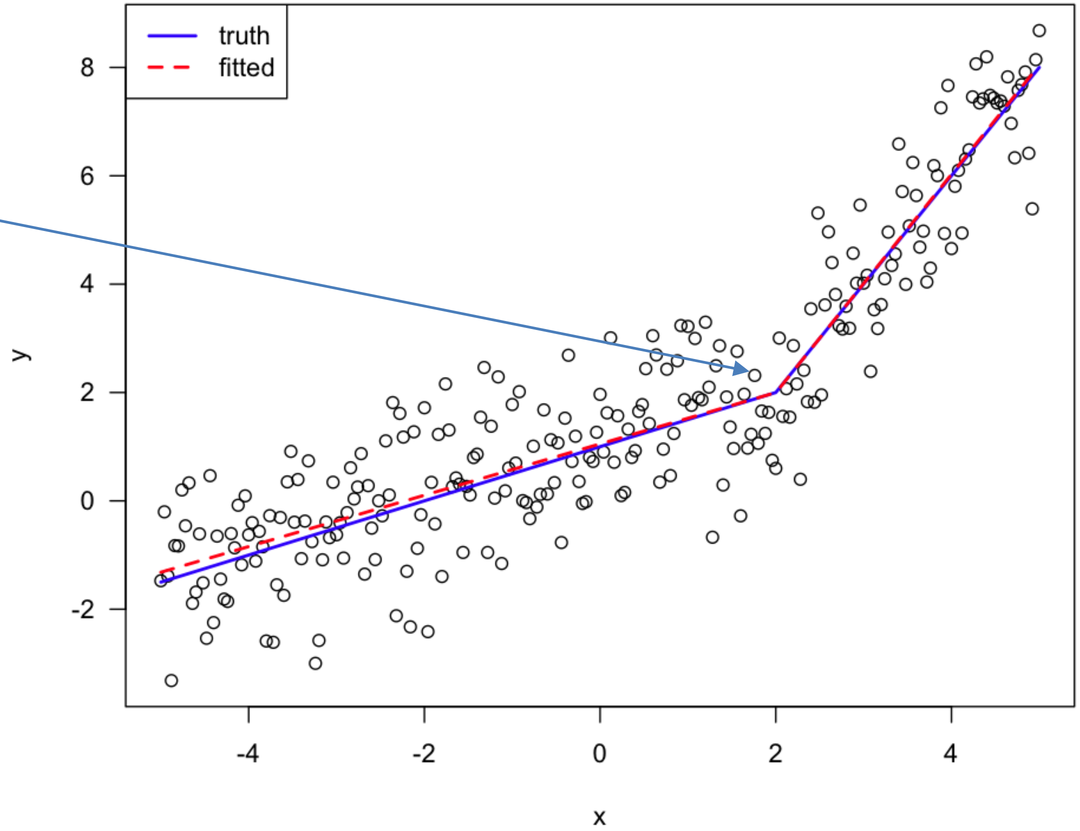
- AKA piecewise regression / "broken-stick regression"

When x < 2

$$y_i = \alpha + \beta x_i$$

When x > 2

$$y_i = \alpha + \beta x_i$$



https://stats.stackexchange.com/questions/14538/not-usual-piecewise-linear-regression

Regression
Building a model using Multiple Regression

- Stepwise regression
  - Use a test for each independent variable (correlation)
- Model building using iteration
  - Backwards elimination
  - Forward selection

Regression
Multicollinearity

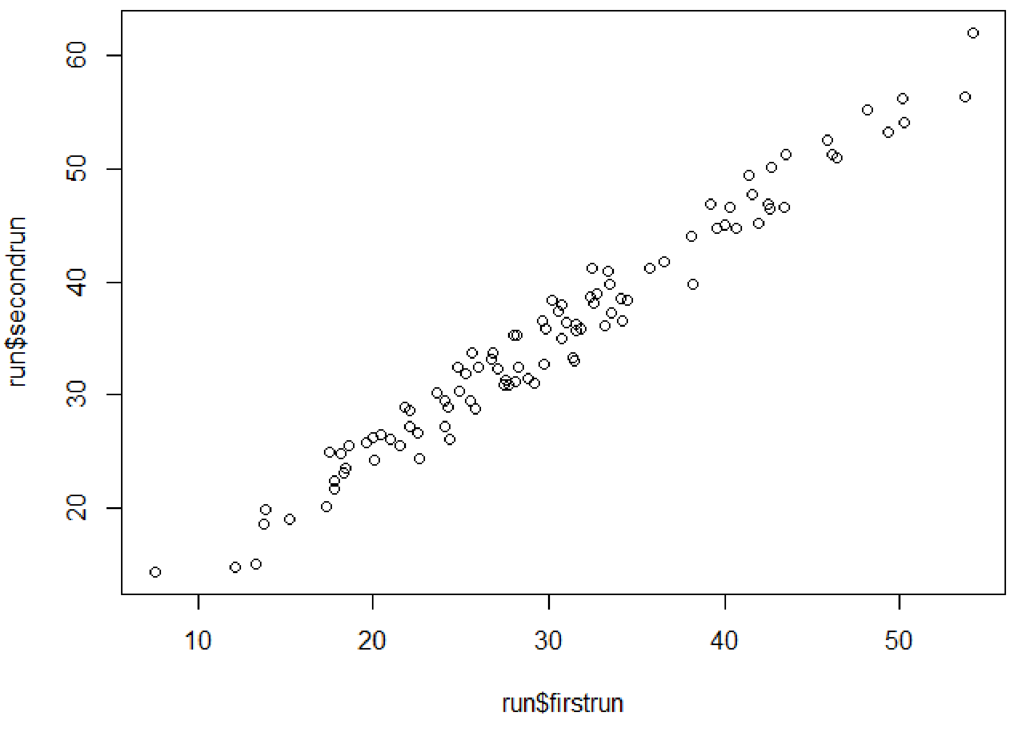
- This is when multiple predictors are highly correlated meaning that coefficients maybe less interpretable

Regression

Regression Tutorial

- Using run.csv

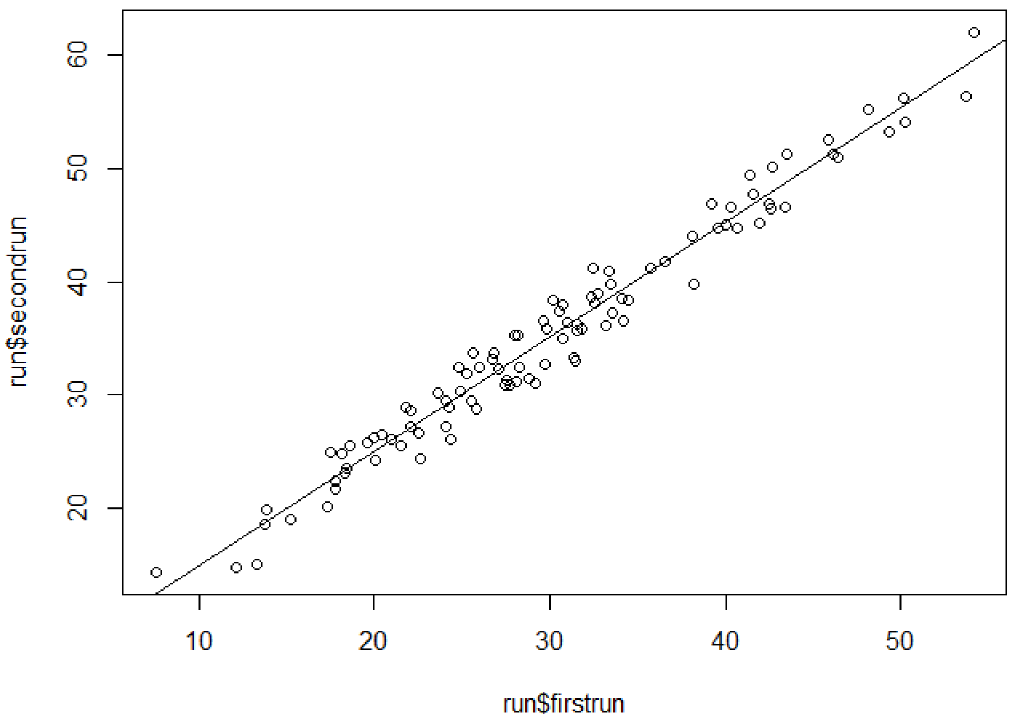
```
> run <- read.csv('run.csv')
> plot(run$firstrun,run$secondrun)
>
```



Regression

Regression Tutorial

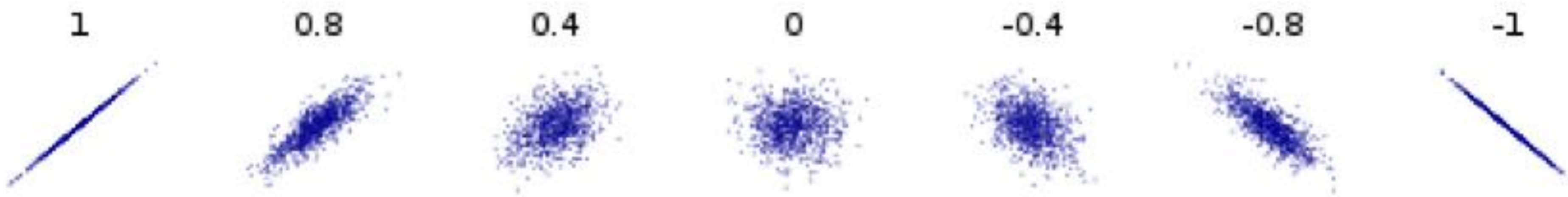
```
> plot(run$firstrun,run$secondrun)
> reg1 <- lm(run$secondrun ~ run$firstrun)
> abline(reg1)
>
```



Regression
Regression Tutorial

- Pearson Correlation Coefficient

```
> r <- cor(run$firstrun,run$secondrun)
> r
[1] 0.9843855
>
```



https://en.wikipedia.org/wiki/Correlation

Regression

Regression Tutorial

- Check if the Correlation is significant

```
> cor.test(run$firstrun,run$secondrun)

        Pearson's product-moment correlation

data:  x and y
t = 55.077, df = 97, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.9767932 0.9895071
sample estimates:
      cor
0.9843855
```

p-value < 0.05 indicates the correlation is statistically significant

Regression

Regression Tutorial

- Multiple Regression
- help("~") - tilde is used to separate the left- and right-hand sides in a model formula.

```
> lma <- lm(run$finalrun ~ firstrun + secondrun, data=run)
>
> summary(lma)

Call:
lm(formula = run$finalrun ~ firstrun + secondrun, data = run)

Residuals:
    Min      1Q  Median      3Q     Max
-9.1691 -1.5454  0.1766  1.6946  6.8319

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   6.1518     1.2025   5.116 1.60e-06 ***
firstrun      0.1764     0.1649   1.070    0.287
secondrun     0.8679     0.1607   5.400 4.82e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.873 on 96 degrees of freedom
Multiple R-squared:  0.9334,    Adjusted R-squared:  0.932
F-statistic: 672.6 on 2 and 96 DF,  p-value: < 2.2e-16

> |
```

Ulster University

Regression

Regression Tutorial

- Create a model and predict a value

```
> lma <- lm(finalrun ~ firstrun + secondrun, data=run)
> newdata <- data.frame(firstrun=22, secondrun=25)
> predict(lma,newdata)
       1
31.72882
> 
```

Regression

Regression Tutorial

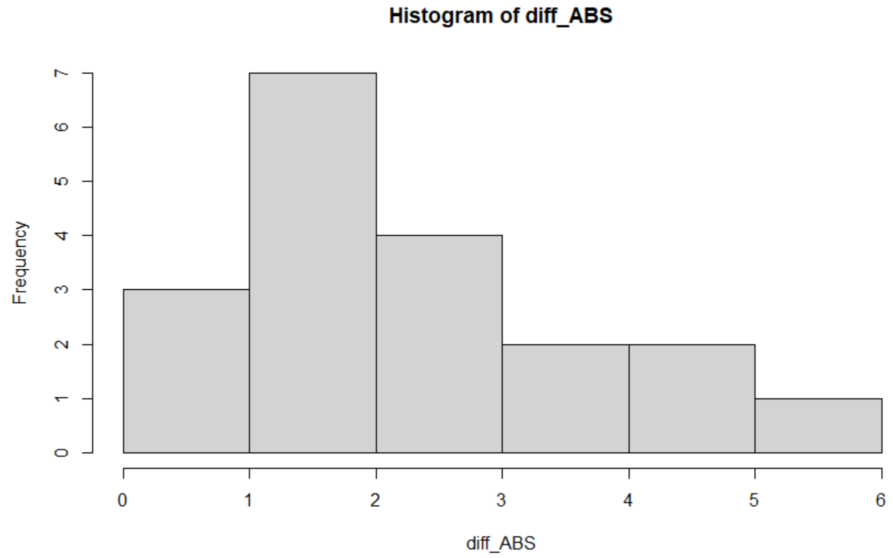
- Build model using 80% of the rows
- Test on 20% of the rows

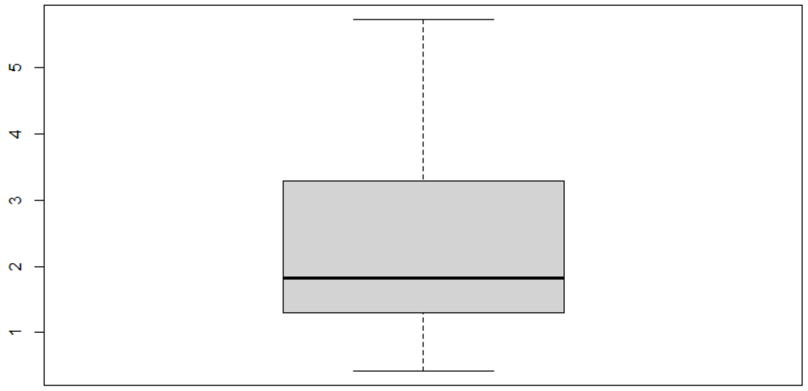
```
> set.seed(12345)
> run2 <- run[order(runif(99)),]
> lma <- lm(finalrun ~ firstrun + secondrun, data=run2[c(1:80),])
> testing <- run2[c(81:99),]
> predictions <- predict(lma,testing)
```

Regression

Regression Tutorial

```
> diff <- predictions - testing$finalrun
> diff_ABS <- abs(diff)
> mean(diff)
[1] -0.1306889
> hist(diff_ABS)
> boxplot(diff_ABS)
>
```





Regression
Regression Tutorial

- How many predicted distances of the final run were higher than the real distance of the final run?

```
> length(diff[diff>0])
[1] 10
```

- How many predicted distances were within 2km of the real distance of the final run?

```
> length(diff_ABS[diff_ABS<=2])
[1] 10
```

Ulster University

Regression

Regression Tutorial

```
> testing$prediction <- predictions
> testing$difference <- diff_ABS
> testing
    X firstrun secondrun finalrun prediction difference
92 92 32.37726  38.73920 44.88548   45.30739  0.4219054
87 87 17.52331  24.99804 29.37527   31.07447  1.6991964
52 52 41.91553  45.16510 51.27962   52.54004  1.2604169
37 37 32.42804  41.16914 50.93965   47.25107  3.6885802
81 81 50.28730  54.16673 66.55377   61.56143  4.9923423
2   2 41.60819  47.71729 55.34708   54.50125  0.8458281
4   4 38.18542  39.78412 46.55384   47.43082  0.8769718
38 38 31.48147  33.00327 39.31016   40.54661  1.2364473
43 43 34.14534  38.57678 47.85165   45.57167  2.2799835
91 91 49.39089  53.19785 62.11803   60.59146  1.5265686
70 70 17.74187  21.66239 22.73644   28.47046  5.7340216
67 67 26.02081  32.46219 42.79469   38.90118  3.8935087
20 20 25.81556  28.77266 34.49935   35.92144  1.4220917
77 77 42.71507  50.13416 58.74417   56.66955  2.0746192
51 51 17.27106  20.12811 22.97570   27.14557  4.1698691
23 23 45.89486  52.51884 60.62786   59.27353  1.3543277
65 65 25.63270  33.79786 38.04635   39.87699  1.8306355
63 63 12.11494  14.85375 24.69498   21.80381  2.8911652
10 10 32.69781  39.02388 43.19283   45.60511  2.4122794
>
```

Two new columns added

Regression

Regression Tutorial - Wine Quality Data Set

- https://archive.ics.uci.edu/ml/datasets/wine+quality
- Load the dataset

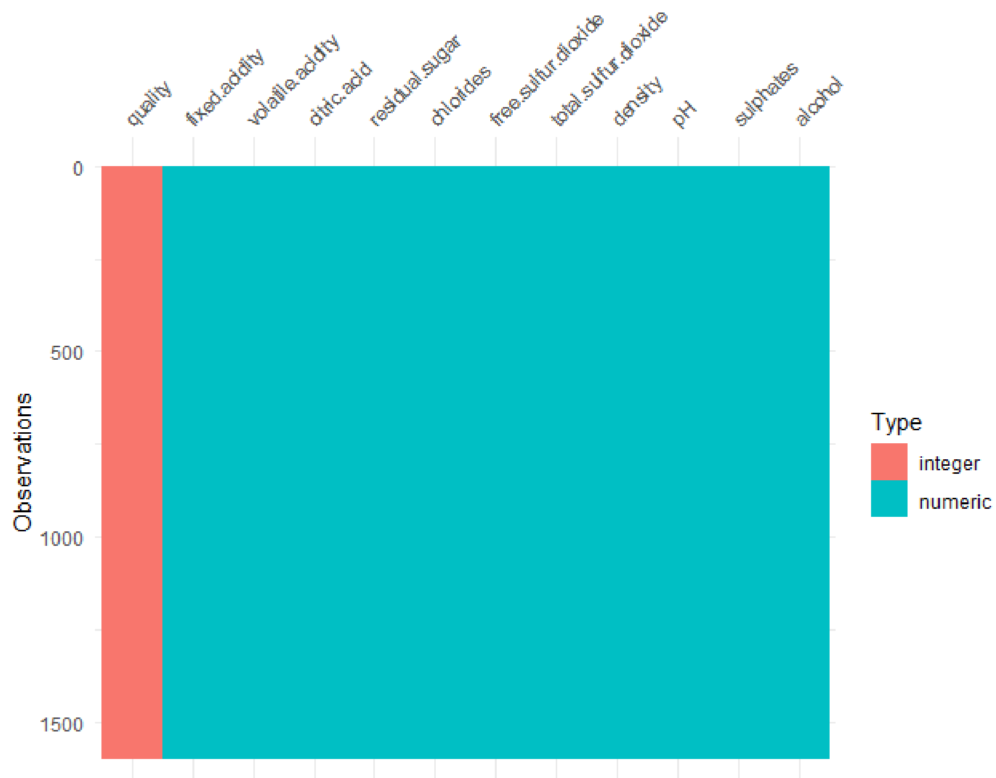
```
> df_red <- read.csv("winequality-red.csv", sep = ";")
> head(df_red)
  fixed.acidity volatile.acidity citric.acid residual.sugar chlorides free.sulfur.dioxide total.sulfur.dioxide density
1           7.4             0.70        0.00            1.9     0.076                  11                   34  0.9978
2           7.8             0.88        0.00            2.6     0.098                  25                   67  0.9968
3           7.8             0.76        0.04            2.3     0.092                  15                   54  0.9970
4          11.2             0.28        0.56            1.9     0.075                  17                   60  0.9980
5           7.4             0.70        0.00            1.9     0.076                  11                   34  0.9978
6           7.4             0.66        0.00            1.8     0.075                  13                   40  0.9978
    pH sulphates alcohol quality
1 3.51      0.56     9.4       5
2 3.20      0.68     9.8       5
3 3.26      0.65     9.8       5
4 3.16      0.58     9.8       6
5 3.51      0.56     9.4       5
6 3.51      0.56     9.4       5
> 
```

Regression

Regression Tutorial - Wine Quality Data Set

- Check the missing value

```
> # check missing value
> sum(is.na(df_red))
[1] 0
> # visulisation of missing values
> library('visdat')
> vis_dat(df_red)
> |
```



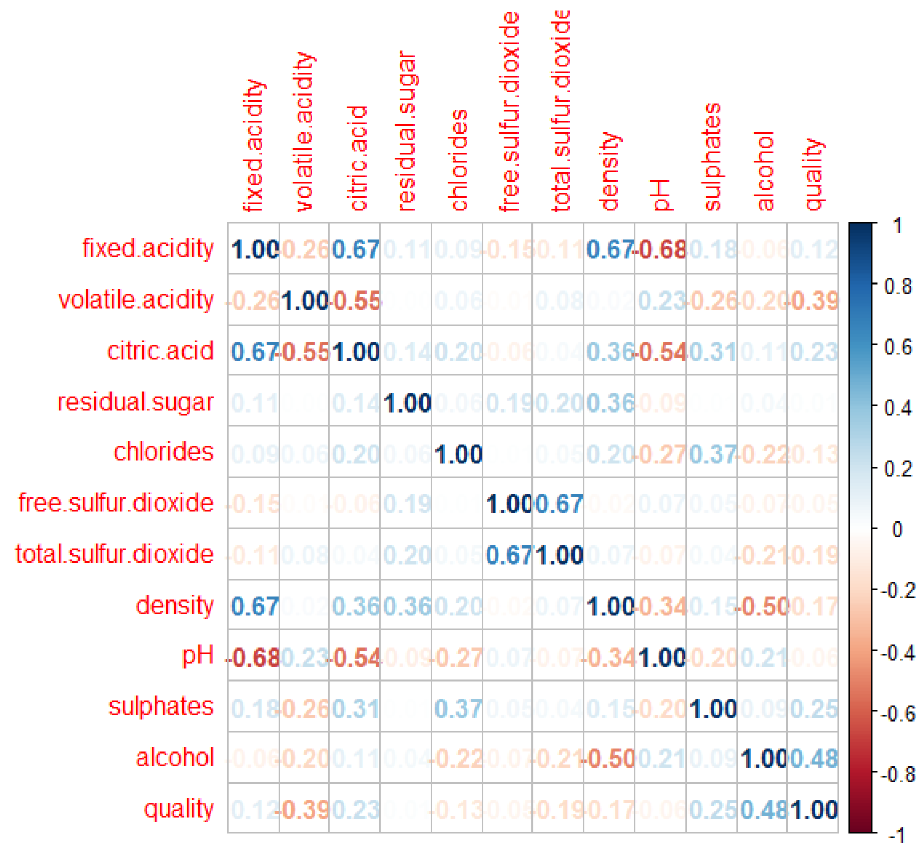
Ulster University

Regression

Regression Tutorial - Wine Quality Data Set

- Correlation analysis

```
> library('corrplot')
corrplot 0.92 loaded
> M <- cor(df_red)
> corrplot(M, method = 'number')
> |
```



Ulster University

Regression

Regression Tutorial - Wine Quality Data Set

- Modelling
  - Splitting Train Datasets and Test Datasets

```
> set.seed(1)
> sampleSize <- round(nrow(df_red)*0.8)
> idx <- sample(seq_len(sampleSize), size = sampleSize)
>
> train_red <- df_red[idx,]
> test_red <- df_red[-idx,]
> |
```

Regression

Regression Tutorial - Wine Quality Data Set

- Modelling
  - Create linear regression model using all the features

```
> model_red1 <- lm(quality ~ fixed.acidity + volatile.acidity + citric.acid + chlorides + free.sulfur.dioxide + total.sulfu
r.dioxide + density + pH + sulphates + alcohol, data = train_red)
> summary(model_red1)

Call:
lm(formula = quality ~ fixed.acidity + volatile.acidity + citric.acid +
    chlorides + free.sulfur.dioxide + total.sulfur.dioxide +
    density + pH + sulphates + alcohol, data = train_red)

Residuals:
     Min       1Q   Median       3Q      Max
-2.67536 -0.38553 -0.06879  0.45454  1.97578

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)           1.510e+01  1.912e+01   0.790 0.429935
fixed.acidity         1.547e-02  2.696e-02   0.574 0.566286
volatile.acidity     -1.057e+00  1.358e-01  -7.780 1.49e-14 ***
citric.acid          -1.774e-01  1.657e-01  -1.070 0.284621
chlorides            -1.779e+00  4.627e-01  -3.845 0.000126 ***
free.sulfur.dioxide   3.392e-03  2.480e-03   1.368 0.171691
total.sulfur.dioxide -3.645e-03  8.201e-04  -4.444 9.58e-06 ***
density              -1.102e+01  1.954e+01  -0.564 0.572664
pH                   -3.835e-01  2.010e-01  -1.908 0.056598 .
sulphates             7.945e-01  1.217e-01   6.527 9.67e-11 ***
alcohol               2.924e-01  2.462e-02  11.878  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6476 on 1268 degrees of freedom
Multiple R-squared:  0.3695,    Adjusted R-squared:  0.3645
F-statistic:  74.3 on 10 and 1268 DF,  p-value: < 2.2e-16

> |
```

Regression

Regression Tutorial - Wine Quality Data Set

- Modelling
  - Create linear regression model using only one feature

```
> model_red2 <- lm(quality ~ alcohol, data = train_red)
> summary(model_red2)

Call:
lm(formula = quality ~ alcohol, data = train_red)

Residuals:
    Min      1Q  Median      3Q     Max 
-2.8861 -0.4048 -0.1827  0.5582  2.5581 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1.81376    0.18792   9.652   <2e-16 ***
alcohol      0.37021    0.01797  20.601   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7041 on 1277 degrees of freedom
Multiple R-squared:  0.2494,	Adjusted R-squared:  0.2489 
F-statistic: 424.4 on 1 and 1277 DF,  p-value: < 2.2e-16

> |
```

Regression

Regression Tutorial - Wine Quality Data Set

- Modelling
  - Create linear regression model using top 5 most correlated features

```
> model_red3 <- lm(quality ~ alcohol + volatile.acidity + sulphates + citric.acid + density,
+                  data = train_red)
> summary(model_red3)

Call:
lm(formula = quality ~ alcohol + volatile.acidity + sulphates +
    citric.acid + density, data = train_red)

Residuals:
    Min      1Q  Median      3Q     Max
-2.7055 -0.4068 -0.0748  0.4949  2.2302

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)      -10.20026   13.12402  -0.777    0.437
alcohol            0.33403    0.02076  16.089  < 2e-16 ***
volatile.acidity  -1.26276    0.13187  -9.576  < 2e-16 ***
sulphates          0.57332    0.11255   5.094 4.03e-07 ***
citric.acid       -0.16117    0.13303  -1.212    0.226
density           12.75053   13.08709   0.974    0.330
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6601 on 1273 degrees of freedom
Multiple R-squared:  0.3423,    Adjusted R-squared:  0.3397
F-statistic: 132.5 on 5 and 1273 DF,  p-value: < 2.2e-16

> |
```

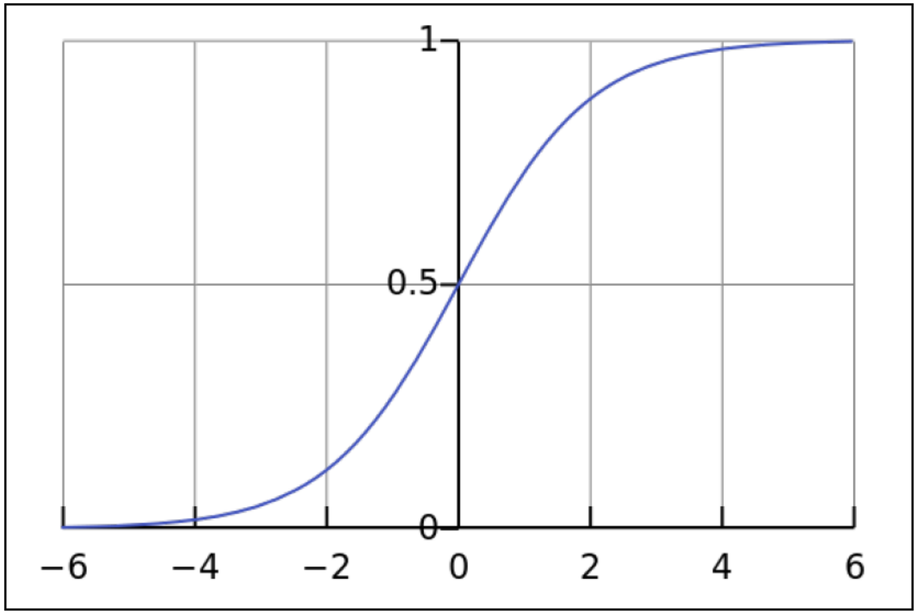
Logistic Regression

- A regression model where the dependent variable is nominal/categorical
  - Useful for feature engineering
  - Useful for binary classification
- Technology adoption
- Success/Unsuccess
- Pass/Fail
- Admitted or not admitted
- Useful for calculating the Odds Ratios for each independent variable, hence good for feature selection

Ulster University

Logistic Regression

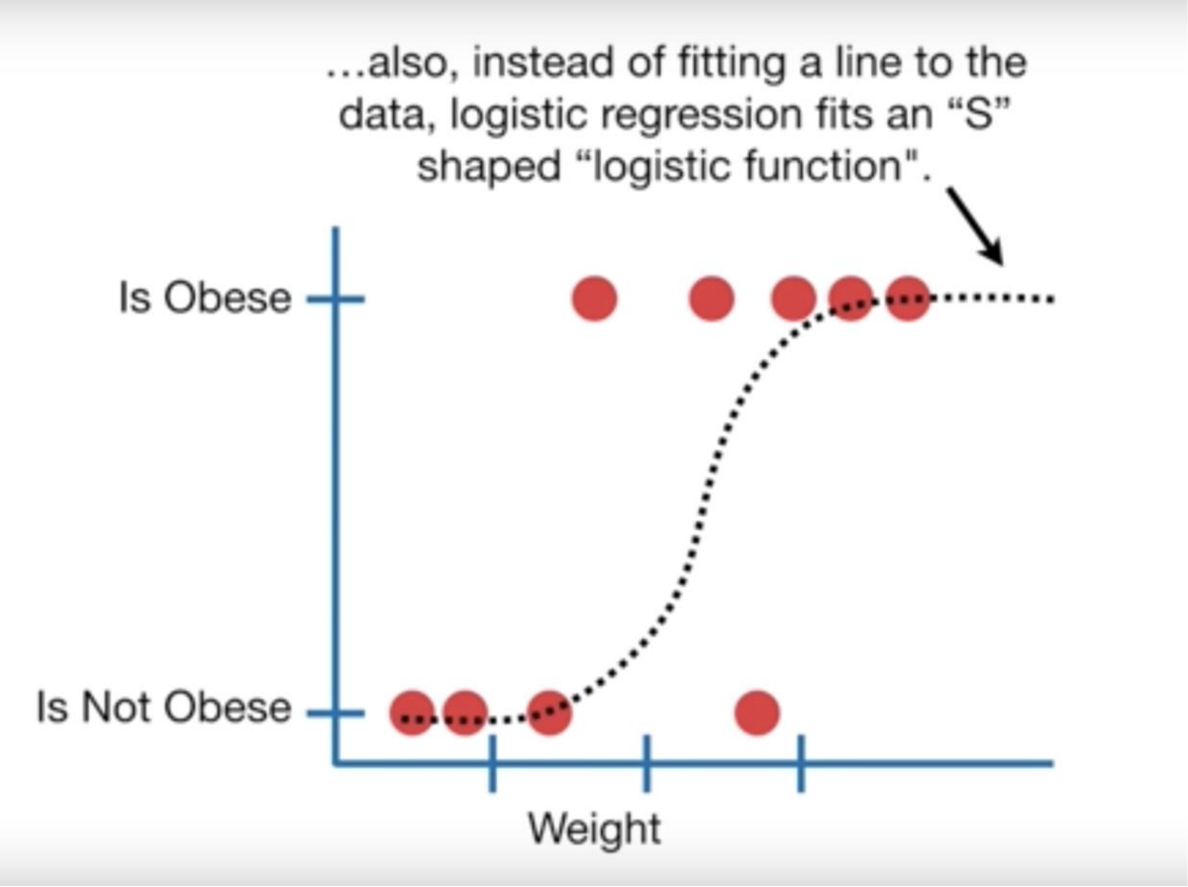
- **Input**: can take any value from a negative infinity to a positive infinity
- **Output**: values from 0 to 1 (a probability)



$$S(x) = \frac{1}{1 + e^{-x}}$$

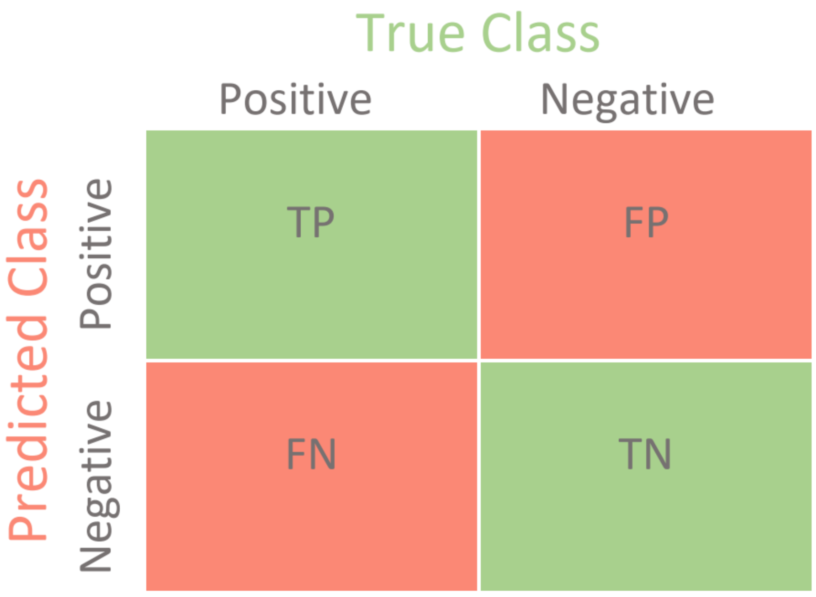
Logistic Regression
Online Exercise

- StatQuest: Logistic Regression.
- https://www.youtube.com/watch?v=yIYKR4sgzI8



Logistic Regression
Model evaluation

- Confusion matrix



https://towardsdatascience.com/confusion-matrix-for-your-multi-class-machine-learning-model-ff9aa3bf7826

Ulster University

Logistic Regression
Exercise

Load data

```
> run <- read.csv('run.csv')
> head(run)
  X firstrun secondrun finalrun
1 1 46.47100  51.04981 62.91495
2 2 41.60819  47.71729 55.34708
3 3 40.29477  46.64771 56.34791
4 4 38.18542  39.78412 46.55384
5 5 34.50337  38.33918 45.68743
6 6 39.25199  46.94684 52.29005
> run$superrun = ifelse(run$finalrun >= 45,1,0)
> head(run)
  X firstrun secondrun finalrun superrun
1 1 46.47100  51.04981 62.91495        1
2 2 41.60819  47.71729 55.34708        1
3 3 40.29477  46.64771 56.34791        1
4 4 38.18542  39.78412 46.55384        1
5 5 34.50337  38.33918 45.68743        1
6 6 39.25199  46.94684 52.29005        1
```

Split training and test data

```
> # split train and test data
> set.seed(1)
> sampleSize <- round(nrow(run)*0.8)
> idx <- sample(seq_len(sampleSize), size = sampleSize)
>
> train_run <- run[idx,]
> test_run <- run[-idx,]
>
```

Logistic Regression
Exercise

Create logistic regression model

```
> # create logistic regression model
> logitModel <- glm(superrun ~ firstrun + secondrun, data=train_run, family = binomial(link = 'logit'))
> summary(logitModel)

Call:
glm(formula = superrun ~ firstrun + secondrun, family = binomial(link = "logit"),
    data = train_run)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.63898  -0.15492  -0.00658   0.02821   1.89502

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -25.9130     8.4084  -3.082  0.00206 **
firstrun      0.5608     0.3281   1.709  0.08746 .
secondrun     0.2117     0.2727   0.776  0.43760
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 104.903  on 78  degrees of freedom
Residual deviance:  24.515  on 76  degrees of freedom
AIC: 30.515

Number of Fisher Scoring iterations: 8
```

Logistic Regression

Exercise

```
> probabilities
          80           81           82           83           84           85           86 
9.999993e-01 9.999989e-01 1.876316e-03 4.603853e-03 3.790518e-01 1.787480e-04 4.775197e-02 
          87           88           89           90           91           92           93 
2.050390e-05 9.968462e-01 6.054316e-03 9.476916e-01 9.999978e-01 6.091173e-01 1.165638e-03 
          94           95           96           97           98           99 
1.892691e-02 1.762211e-01 8.104753e-01 3.039719e-01 3.043883e-01 7.838931e-05 
> 
```

```
> predicted <- as.numeric(probabilities > 0.5)
> predicted
 [1] 1 1 0 0 0 0 0 0 1 0 1 1 1 0 0 0 1 0 0 0
> 
```

```
> predicted <- as.factor(predicted)
> test_run$superrun <- as.factor(test_run$superrun)
> 
```

Logistic Regression
Exercise

- Model evaluation using confusion matrix

```
> library(caret)
Loading required package: lattice
> confusionMatrix(predicted, test_run$superrun)
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 11  2
         1  1  6

               Accuracy : 0.85
                 95% CI : (0.6211, 0.9679)
    No Information Rate : 0.6
    P-Value [Acc > NIR] : 0.01596

                  Kappa : 0.6809

 Mcnemar's Test P-Value : 1.00000

            Sensitivity : 0.9167
            Specificity : 0.7500
         Pos Pred Value : 0.8462
         Neg Pred Value : 0.8571
             Prevalence : 0.6000
         Detection Rate : 0.5500
   Detection Prevalence : 0.6500
      Balanced Accuracy : 0.8333

       'Positive' Class : 0

> |
```

Logistic Regression

Exercise

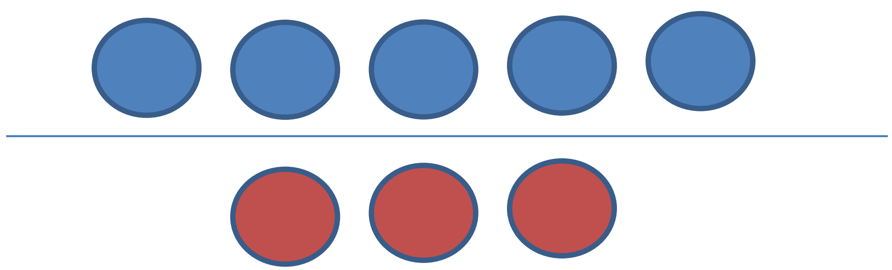
- exp(coef(logitModel)) ## odds ratios only
- exp(cbind(OR = coef(logitModel), confit(logitModel)

  ## odds ratios and 95% CI

```
> exp(coef(logitModel))
 (Intercept)    firstrun    secondrun
4.071837e-12 1.541070e+00 1.394437e+00
> exp(cbind(OR = coef(logitModel), confint(logitModel)))
Waiting for profiling to be done...
                      OR        2.5 %       97.5 %
(Intercept) 4.071837e-12 3.583676e-20 3.304195e-07
firstrun    1.541070e+00 9.353618e-01 2.853630e+00
secondrun   1.394437e+00 8.928344e-01 2.392165e+00
```

Logistic Regression
Another example

- What are the odds of winning the next football game?
  - From the last 8 games, if you won 5 and lost 3 then what are the odds of winning the next game?



5/3 = 1.66… times

StatQuest: Odds and Log(Odds): https://www.youtube.com/watch?v=ARfXDSkQf1Y&t=86s

Logistic Regression
Odds ratios

- How do I interpret odds ratios in logistic regression?
  - https://stats.oarc.ucla.edu/stata/faq/how-do-i-interpret-odds-ratios-in-logistic-regression/

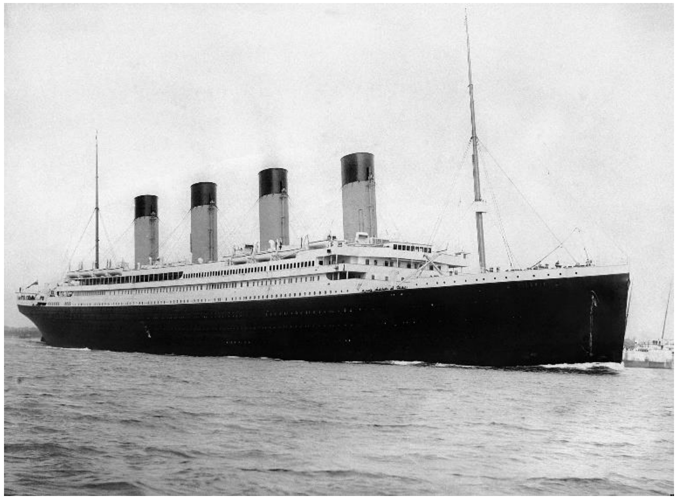
Logistic Regression
To learn more - watch

- Stats quest:
  - Logistic regression overview
    - https://www.youtube.com/watch?v=yIYKR4sgzI8
  - Logistic regression coefficients
    - https://www.youtube.com/watch?v=vN5cNN2-HWE
  - Maximum likelihood
    - https://www.youtube.com/watch?v=BfKanl1aSG0
  - R squared and p values for logistic regression
    - https://www.youtube.com/watch?v=xxFYro8QuXA
  - Odds ratios from logit
    - https://www.youtube.com/watch?v=8nm0G-1uJzA

Ulster University

Logistic Regression
Exercise

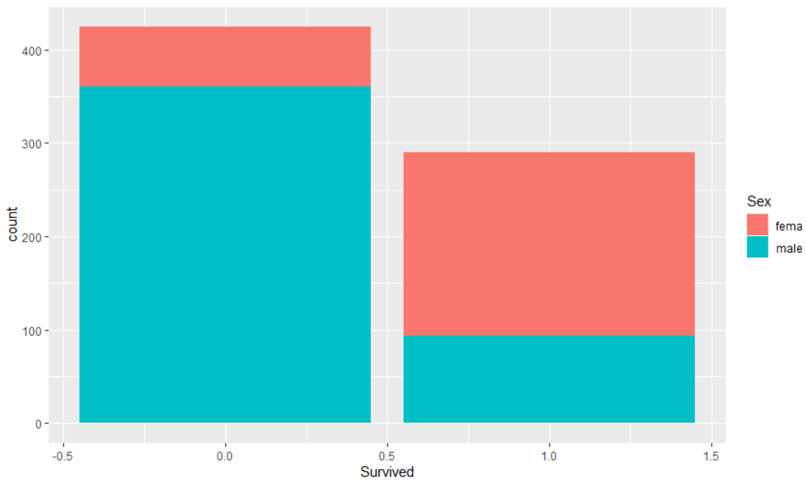
- Studying the Titanic passenger survival
- Titanic dataset from Kaggle
  https://www.kaggle.com/c/titanic
- build a logistic regression model for survival

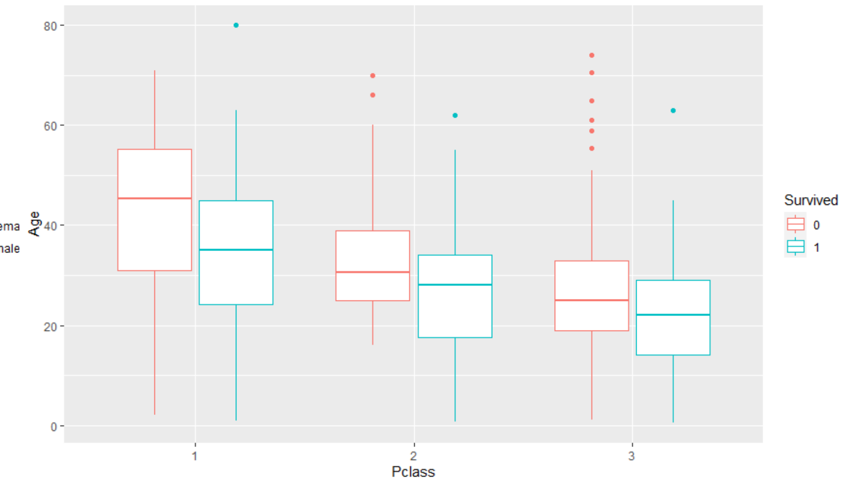


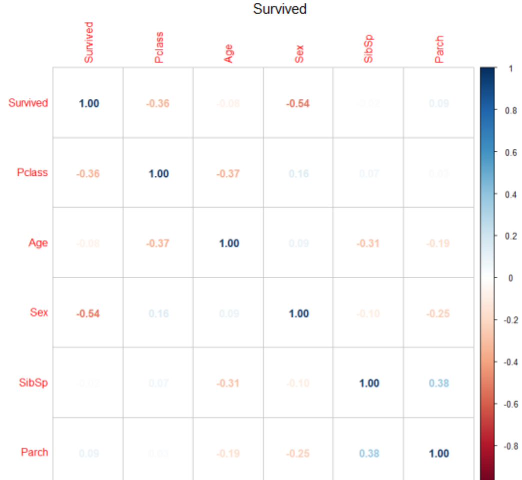
RMS Titanic, By F.G.O. Stuart (1843–1923)

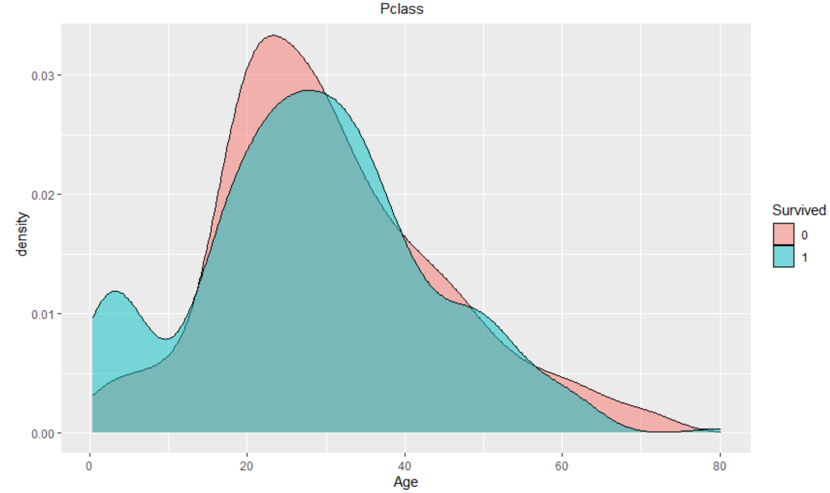
Logistic Regression
Exercise

- Review data exploratory analysis in Week 3









Logistic Regression
Exercise

- Load the data

```
> df <- read.csv('train.csv', na.strings = '') # empty values are read as NA values
> head(df)
  PassengerId Survived Pclass                                                Name    Sex Age SibSp Parch           Ticket    Fare Cabin Embarked
1           1        0      3                             Braund, Mr. Owen Harris   male  22     1     0        A/5 21171  7.2500  <NA>        S
2           2        1      1 Cumings, Mrs. John Bradley (Florence Briggs Thayer) female  38     1     0         PC 17599 71.2833   C85        C
3           3        1      3                              Heikkinen, Miss. Laina female  26     0     0 STON/O2. 3101282  7.9250  <NA>        S
4           4        1      1        Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35     1     0           113803 53.1000  C123        S
5           5        0      3                            Allen, Mr. William Henry   male  35     0     0           373450  8.0500  <NA>        S
6           6        0      3                                    Moran, Mr. James   male  NA     0     0           330877  8.4583  <NA>        Q
> |
```

- Check missing values

```
> # check missing values
> sum(is.na(df$Cabin))
[1] 687
```

Ulster University

Logistic Regression
Exercise

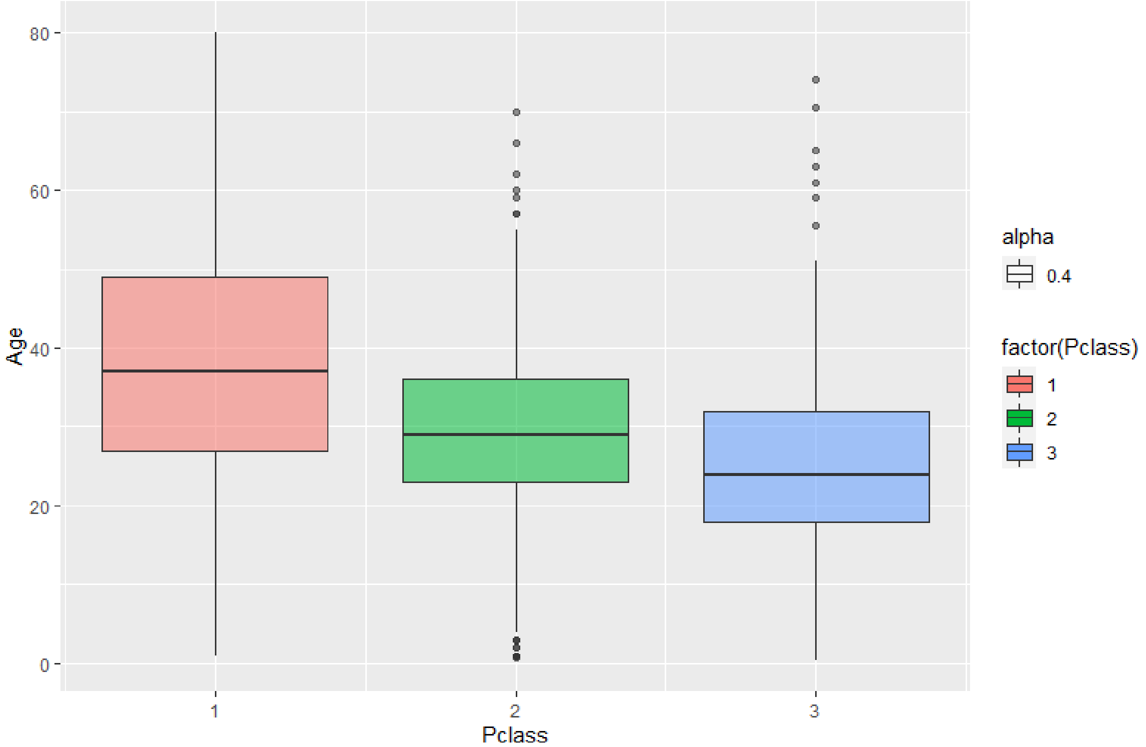
- Missing values

```
> library('visdat')
> vis_dat(df)
>
```

Logistic Regression
Exercise

- Age vs Pclass

```
> library(ggplot2)
> ggplot(df,aes(Pclass,Age)) + geom_boxplot(aes(group=Pclass,fill=factor(Pclass),alpha=0.4))
Warning message:
Removed 177 rows containing non-finite values (stat_boxplot).
> 
```



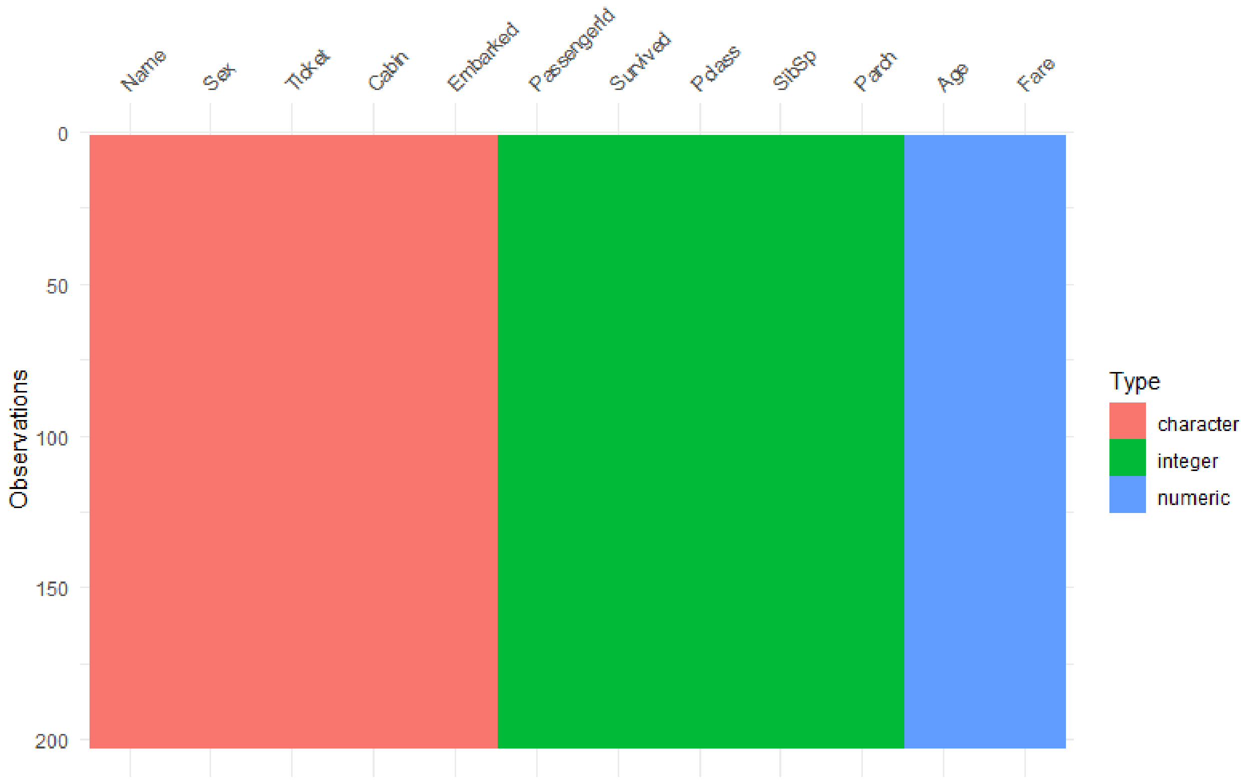
Logistic Regression
Exercise

- Impute missing age values

```
> # calculate average age for different Pclass
> avg_P1 <- round(mean(df[df$Pclass==1,]$Age, na.rm = TRUE))
> avg_P2 <- round(mean(df[df$Pclass==2,]$Age, na.rm = TRUE))
> avg_P3 <- round(mean(df[df$Pclass==3,]$Age, na.rm = TRUE))
> impute_age <- function(age,class){
+     out <- age
+     for (i in 1:length(age)){
+
+         if (is.na(age[i])){
+
+             if (class[i] == 1){
+                 out[i] <- avg_P1
+
+             }else if (class[i] == 2){
+                 out[i] <- avg_P2
+
+             }else{
+                 out[i] <- avg_P3
+             }
+         }else{
+             out[i]<-age[i]
+         }
+     }
+     return(out)
+ }
> fixed.ages <- impute_age(df$Age,df$Pclass)
> df$Age <- fixed.ages
> df <- na.omit(df) #Drop the rows with NA values
```

Logistic Regression
Exercise

```
> # check if the imputation worked
> vis_dat(df)
> 
```



Logistic Regression
Exercise

- Select features

```
> df$Survived <- factor(df$Survived)
> df$Pclass <- factor(df$Pclass)
> df$Parch <- factor(df$Parch)
> df$SibSp <- factor(df$SibSp)
> 
```

```
> library(dplyr)

Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union

> df <- select(df,-PassengerId,-Name,-Ticket,-Cabin)
> str(df)
'data.frame':	202 obs. of  8 variables:
 $ Survived: int  1 1 0 1 1 1 1 0 1 1 ...
 $ Pclass  : int  1 1 1 3 1 2 1 1 1 1 ...
 $ Sex     : chr  "female" "female" "male" "female" ...
 $ Age     : num  38 35 54 4 58 34 28 19 38 49 ...
 $ SibSp   : int  1 1 0 1 0 0 0 3 1 1 ...
 $ Parch   : int  0 0 0 1 0 0 0 2 0 0 ...
 $ Fare    : num  71.3 53.1 51.9 16.7 26.6 ...
 $ Embarked: chr  "C" "S" "S" "S" ...
 - attr(*, "na.action")= 'omit' Named int [1:689] 1 3 5 6 8 9 10 13 14 15 ...
  ..- attr(*, "names")= chr [1:689] "1" "3" "5" "6" ...
> 
```

Logistic Regression

Exercise

- Build logistic regression model to predict survival

```
> log.model <- glm(formula=Survived ~ . , family = binomial(link='logit'),data = df)
> summary(log.model)

Call:
glm(formula = Survived ~ ., family = binomial(link = "logit"),
    data = df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.7909  -0.7906   0.2852   0.5425   2.0974

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)   4.485693   0.969299   4.628 3.70e-06 ***
Pclass2       0.118245   0.845046   0.140   0.8887
Pclass3      -1.478862   0.935923  -1.580   0.1141
Sexmale      -3.012731   0.506304  -5.950 2.67e-09 ***
Age          -0.038646   0.015338  -2.520   0.0118 *
SibSp1        0.426687   0.417689   1.022   0.3070
SibSp2        0.886218   1.761204   0.503   0.6148
SibSp3       -0.906767   1.728729  -0.525   0.5999
Parch1       -0.319603   0.545424  -0.586   0.5579
Parch2       -0.652582   0.731659  -0.892   0.3724
Parch4      -13.825696 882.743802  -0.016   0.9875
Fare          0.001273   0.002925   0.435   0.6633
EmbarkedQ    -1.878968   1.639963  -1.146   0.2519
EmbarkedS    -0.501598   0.434676  -1.154   0.2485
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 258.07  on 201  degrees of freedom
Residual deviance: 173.92  on 188  degrees of freedom
AIC: 201.92

Number of Fisher Scoring iterations: 13

> |
```

We can see clearly that Sex, Age are the most significant features. Which makes sense given the women and children first policy.

Logistic Regression
Exercise

- Predicting using Test cases
  - Training and test split

```
> library(caTools)
> set.seed(101)
>
> split = sample.split(df$Survived, SplitRatio = 0.70)
>
> df_train = subset(df, split == TRUE)
> df_test = subset(df, split == FALSE)
>
```

https://stackoverflow.com/questions/17200114/how-to-split-data-into-training-testing-sets-using-sample-function

Ulster University

Logistic Regression
Exercise

- Build the model using training dataset

```
> # build the model using training dataset
> log.model1 <- glm(formula=Survived ~ . , family = binomial(link='logit'),data = df_train)
> summary(log.model1)

Call:
glm(formula = Survived ~ ., family = binomial(link = "logit"),
    data = df_train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.6092  -0.7143   0.2133   0.4894   2.3432

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.624e+00  1.192e+00   3.880 0.000104 ***
Pclass2      1.841e+00  1.351e+00   1.362 0.173074
Pclass3     -2.175e+00  1.186e+00  -1.834 0.066589 .
Sexmale     -3.347e+00  6.903e-01  -4.849 1.24e-06 ***
Age         -3.213e-02  1.930e-02  -1.664 0.096066 .
SibSp1       4.689e-01  4.931e-01   0.951 0.341620
SibSp2       1.472e+01  2.011e+03   0.007 0.994161
SibSp3      -1.690e+01  3.956e+03  -0.004 0.996592
Parch1      -7.259e-01  6.480e-01  -1.120 0.262605
Parch2      -1.322e+00  9.968e-01  -1.326 0.184757
Parch4      -1.724e+01  3.956e+03  -0.004 0.996523
Fare         2.899e-03  3.408e-03   0.851 0.394898
EmbarkedQ   -1.170e+00  3.197e+00  -0.366 0.714289
EmbarkedS   -7.758e-01  5.488e-01  -1.414 0.157502
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 181.68  on 141  degrees of freedom
Residual deviance: 112.12  on 128  degrees of freedom
AIC: 140.12

Number of Fisher Scoring iterations: 16

> |
```

Logistic Regression
Exercise

- Check the prediction accuracy

```
> # Now check the prediction accuracy!
> probabilities <- predict(log.model1,newdata=df_test,type='response')
> probabilities
          63           89           97          111          124          125          129          167          210          225          231 
4.417443e-01 5.875553e-07 2.882180e-01 2.976527e-01 9.908272e-01 1.498200e-01 8.105386e-01 8.870230e-01 5.203135e-01 5.024311e-01 9.687646e-01 
         246          264          285          293          306          310          312          326          332          340          342 
9.999988e-01 3.133768e-01 3.441760e-01 8.805875e-01 5.143617e-01 9.786482e-01 1.000000e+00 9.793761e-01 2.934999e-01 3.010955e-01 5.689787e-07 
         346          378          391          395          446          450          457          458          461          488          493 
9.930039e-01 4.256300e-01 2.384780e-01 4.081726e-01 3.290564e-01 2.532270e-01 1.808523e-01 9.625446e-01 2.759944e-01 3.773614e-01 2.354377e-01 
         497          499          505          506          516          517          524          537          540          572          578 
9.730052e-01 9.328377e-01 9.730056e-01 8.150051e-01 2.868709e-01 9.903094e-01 9.341561e-01 2.956631e-01 9.392611e-01 1.000000e+00 9.618014e-01 
         582          586          592          631          648          708          711          725          731          752          760 
9.687893e-01 8.983150e-01 9.746426e-01 1.210536e-01 3.965905e-01 3.159569e-01 9.819478e-01 5.636863e-01 9.715035e-01 7.197754e-02 9.542876e-01 
         764          766          773          790          840 
8.990071e-01 9.480654e-01 9.799236e-01 5.070712e-01 5.353868e-01 
> # Now let's calculate from the predicted values:
> results <- ifelse(probabilities > 0.5,1,0)
> results
 63  89  97 111 124 125 129 167 210 225 231 246 264 285 293 306 310 312 326 332 340 342 346 378 391 395 446 450 457 458 461 488 493 497 499 505 506 
  0   0   0   0   1   0   1   1   1   1   1   1   0   0   1   1   1   1   1   0   0   0   1   0   0   0   0   0   0   1   0   0   0   1   1   1   1 
516 517 524 537 540 572 578 582 586 592 631 648 708 711 725 731 752 760 764 766 773 790 840 
  0   1   1   0   1   1   1   1   1   1   0   0   0   1   1   1   0   1   1   1   1   1   1 
> |
```

Ulster University

Logistic Regression
Exercise

- Check the prediction accuracy

```
> misClasificError <- mean(results != df_test$Survived)
> print(paste('Accuracy',1-misClasificError))
[1] "Accuracy 0.716666666666667"
>
> table(df_test$Survived, probabilities > 0.5)

    FALSE TRUE
  0    14    6
  1    11   29
>
```

Logistic Regression

Exercise

- Confusion matrix

```
> predicted <- as.numeric(probabilities > 0.5)
> predicted <- as.factor(predicted)
> library(caret)
> confusionMatrix(predicted, df_test$Survived)
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 14 11
         1  6 29

               Accuracy : 0.7167
                 95% CI : (0.5856, 0.8255)
    No Information Rate : 0.6667
    P-Value [Acc > NIR] : 0.2495

                  Kappa : 0.4

 Mcnemar's Test P-Value : 0.3320

            Sensitivity : 0.7000
            Specificity : 0.7250
         Pos Pred Value : 0.5600
         Neg Pred Value : 0.8286
             Prevalence : 0.3333
         Detection Rate : 0.2333
   Detection Prevalence : 0.4167
      Balanced Accuracy : 0.7125

       'Positive' Class : 0

> 
```

Ulster University

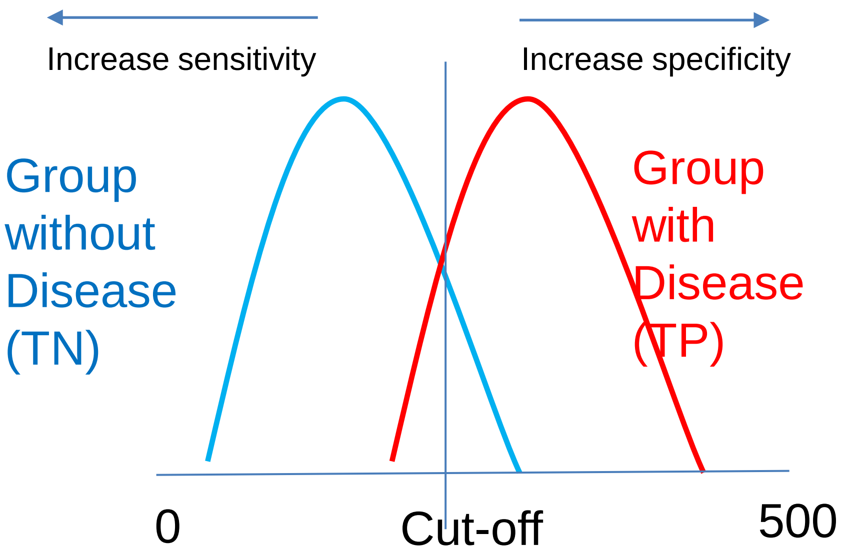
Statistical Model vs. ML algorithm
What is the difference?

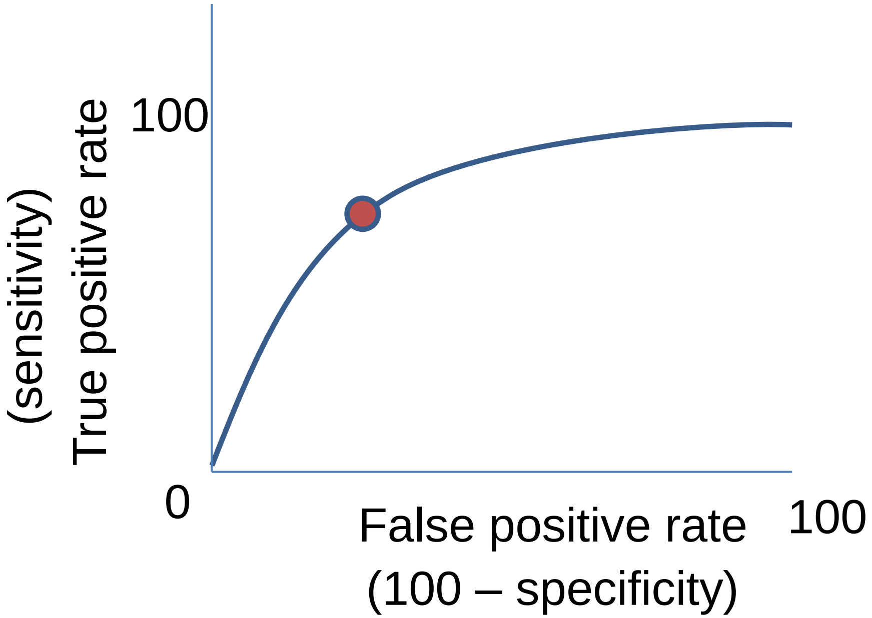
- Read:
  - https://www.analyticsvidhya.com/blog/2015/07/difference-machine-learning-statistical-modeling/

Ulster University

ROC curve

- An example of Receiver Operator Characteristic or ROC curve





$$\text{sensitivity} = \frac{\text{number of true positives}}{\text{number of true positives} + \text{number of false negatives}}$$

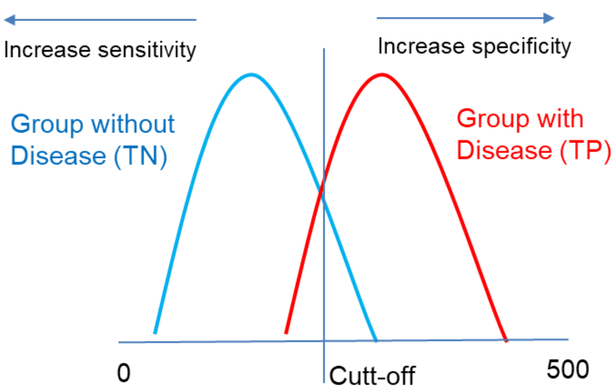
$$\text{specificity} = \frac{\text{number of true negatives}}{\text{number of true negatives} + \text{number of false positives}}$$

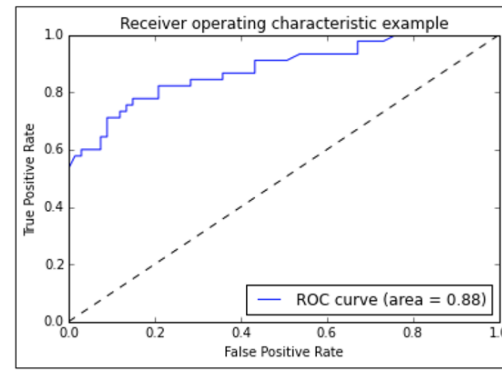
Ulster University

https://en.wikipedia.org/wiki/Sensitivity_and_specificity

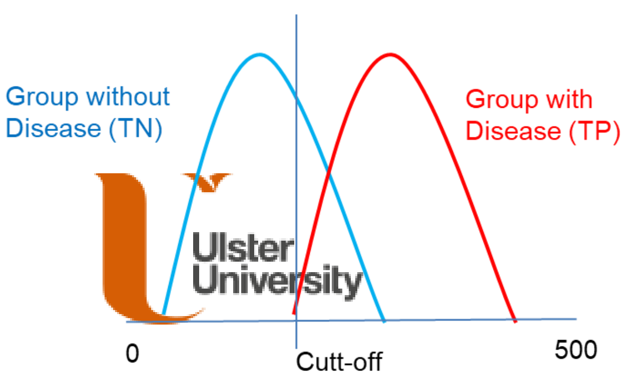
ROC curve

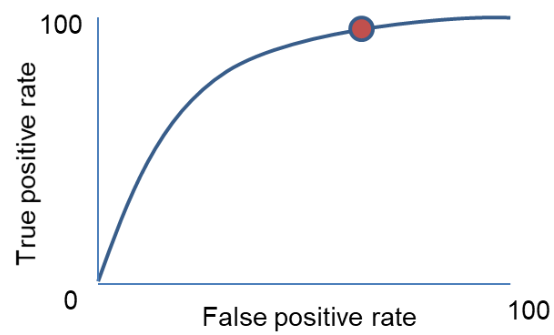
- An example of Receiver Operator Characteristic or ROC curve





Range	Category
0.90-1	This refers to excellent (A)
0.80-0.90	This refers to good (B)
0.70-0.80	This refers to fair (C)
0.60-0.70	This refers to poor (D)
0.50-0.60	This refers to fail (F)





Type 1 error

ROC curve

Receiver Operator Characteristic – Optional Exercise / Reading

- Play:
  - http://www.navan.name/roc/
- Watch:
  - https://www.youtube.com/watch?v=OAl6eAyP-yo
- Read:
  - http://people.inf.elte.hu/kiss/13dwhdm/roc.pdf

Ulster University

Further Reading

- Gareth, J., Daniela, W., Trevor, H. and Robert, T., 2013. An introduction to statistical learning: with applications in R. Springer. (Chapter 3)
- Lantz, Brett. Machine learning with R. Packt Publishing Ltd, 2013 (Chapter 6)
- https://stats.idre.ucla.edu/r/dae/logit-regression/
- https://www.scribbr.com/statistics/linear-regression-in-r/
- https://www.machinelearningplus.com/machine-learning/complete-introduction-linear-regression-r/

Ulster University