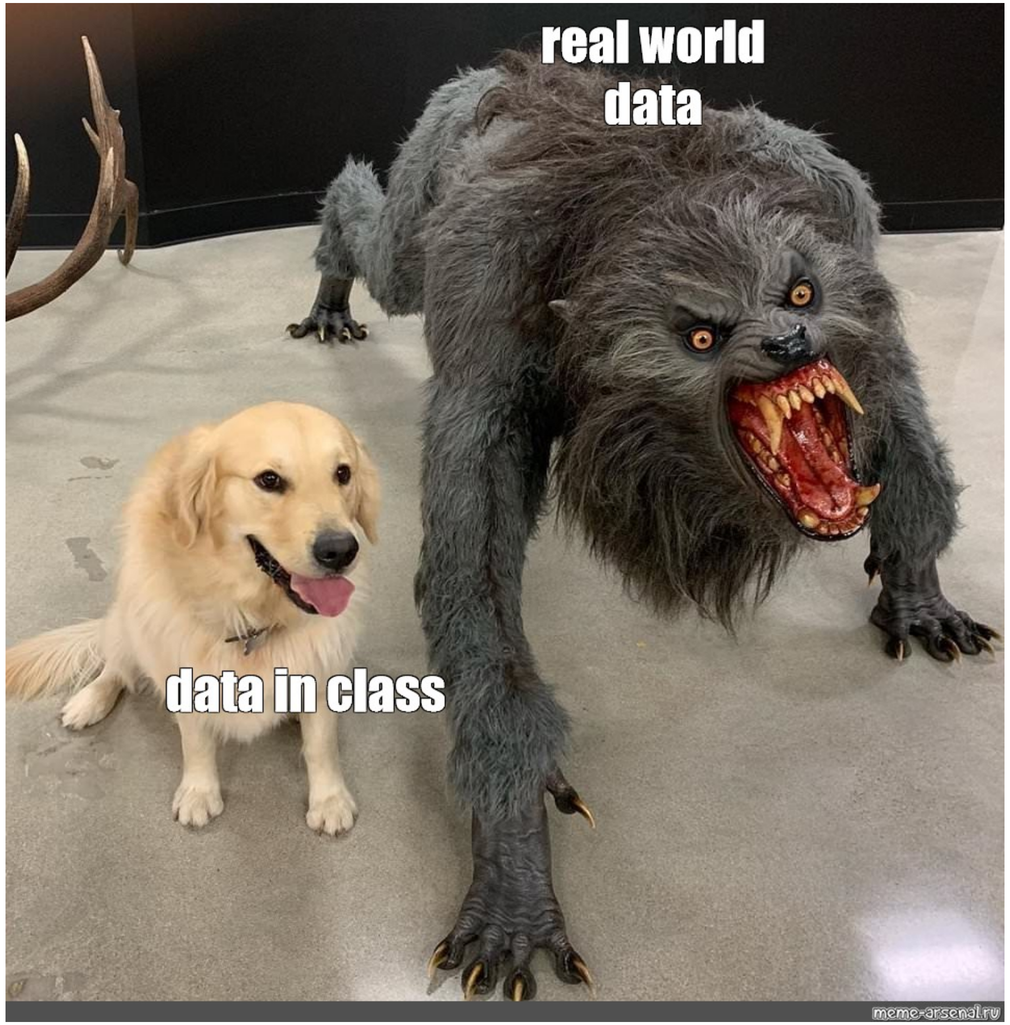
DATA3888: Data Science Capstone

Week 1 - Case study 2: Biomedical data

Andy Tran
22 February, 2023

THE UNIVERSITY OF SYDNEY

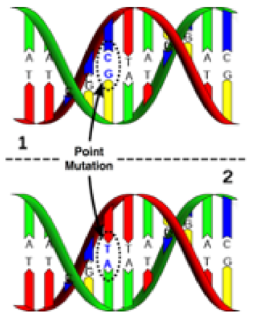

real world data
data in class
meme-arsenal.ru

Outline

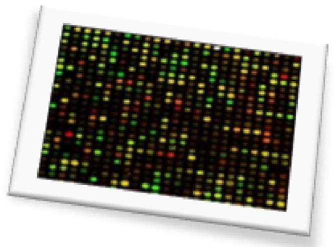
- Biomedical data
- Case study 2: Kidney graft rejection
- Feature Selection

What is molecular biomedical data?

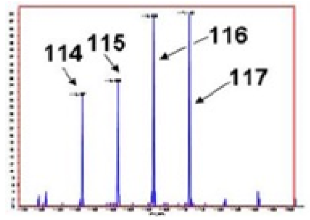
Genome sequencing and point mutations



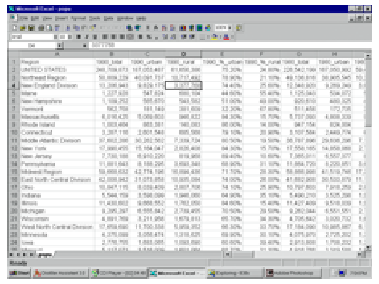
Gene expression
Microarrays
RNA-Seq

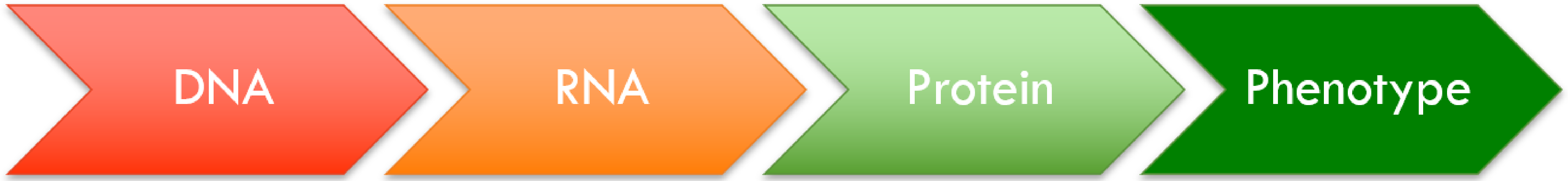


Protein expression
iTRAQ
mass spectrometry

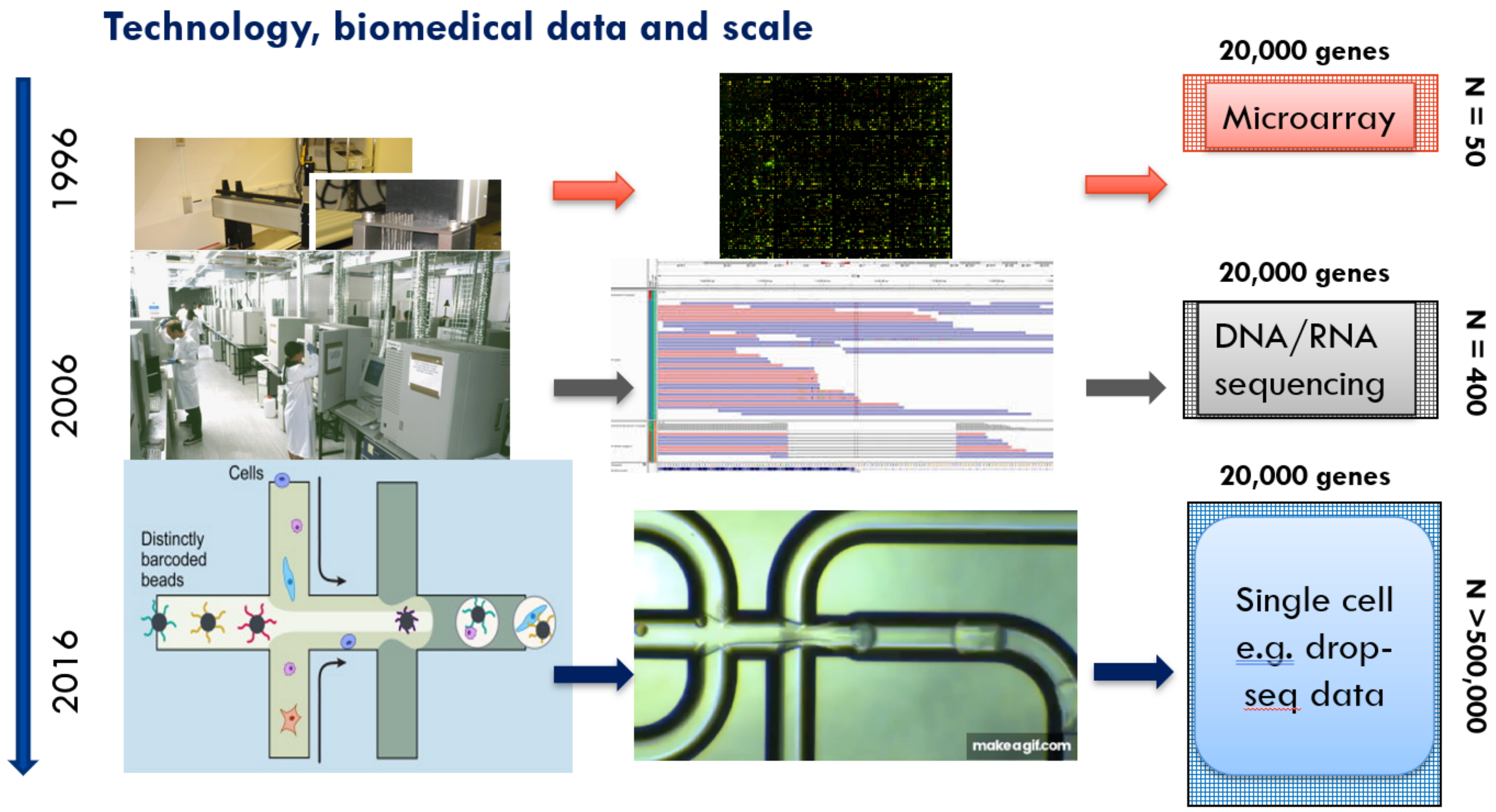


Patient clinical information

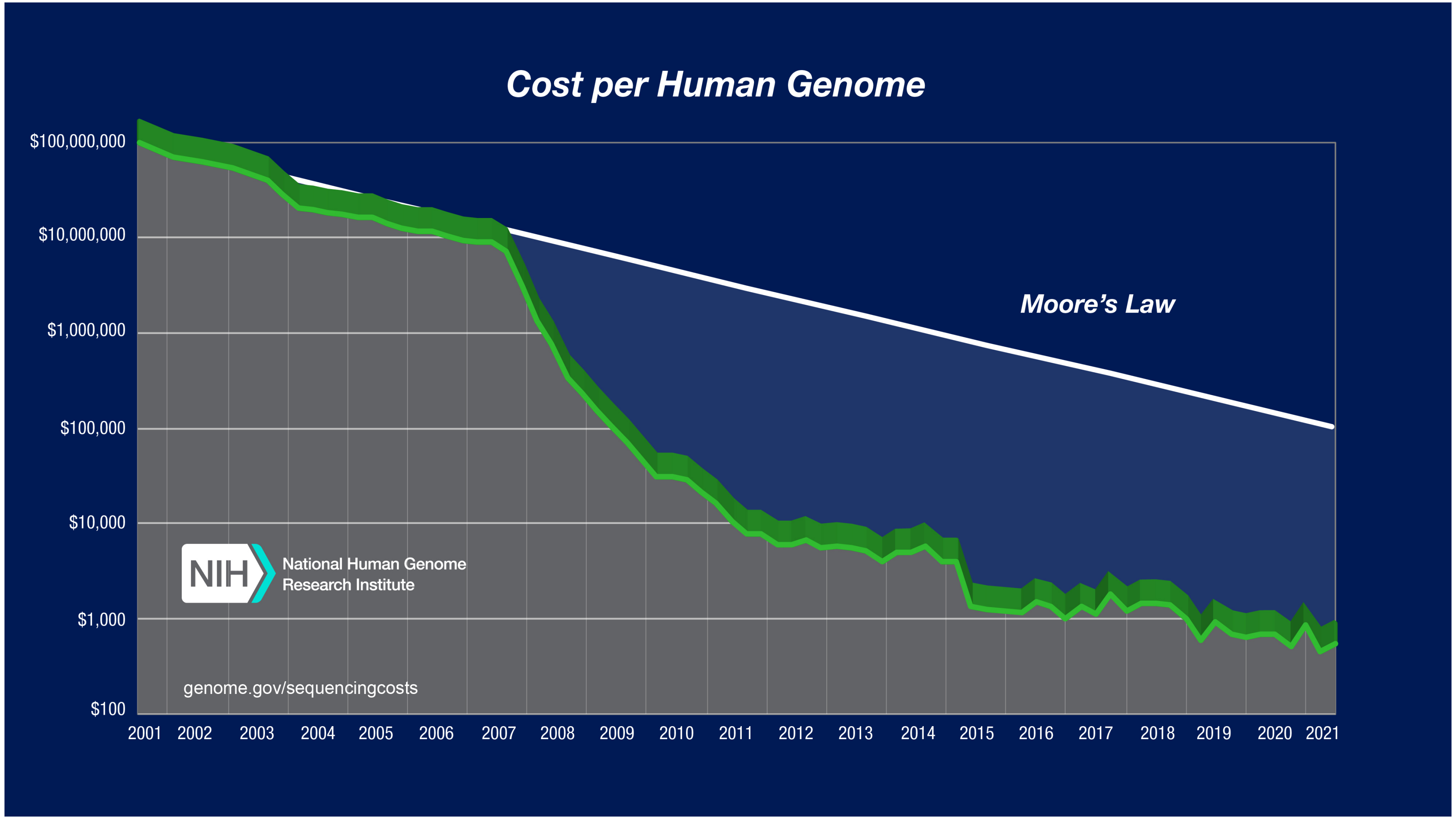




Omics data in recent decades



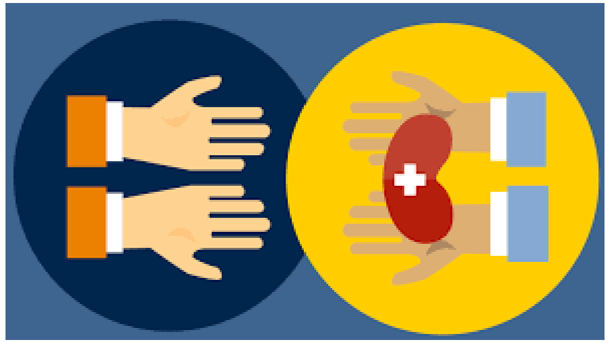
Explosion of omics data



Case study 2: Predicting kidney graft rejection

Case study: Predicting graft status - Kidney transplant

- Treatment of choice to people with end-stage kidney disease.
- Patients may develop graft rejection after kidney transplant.
- Examine public datasets on kidney transplant patients generated from RNA-sequencing.
- RNA-seq measures gene expression of patients' cells.
- Use the gene expression to predict the patient outcome (i.e., stable versus rejection)
    - Stable patients
    - Patients who experienced rejection.



Gene Expression Omnibus database (GEO)

- Public datasets are all taken from GEO database (https://www.ncbi.nlm.nih.gov/geo/).
- Download the dataset by looking up its GSE ID in the database.
- You can go to (https://github.com/seandavi/GEOquery) for installation details

```
library(GEOquery)
gse <- getGEO("GSE46474")
gse <- gse$GSE46474_series_matrix.txt.gz
```

Gene Expression Omnibus Data - GSE46474

- Focus on how to analyse the data from "GSE46474".
- This dataset contains the gene expression profiles of 40 blood samples.
    - 20 patients rejected their kidney
    - 20 had stable grafts and will be treated as controls.
- For Week 2 lab, you can load the pre-made `GSE46474.RData` file.

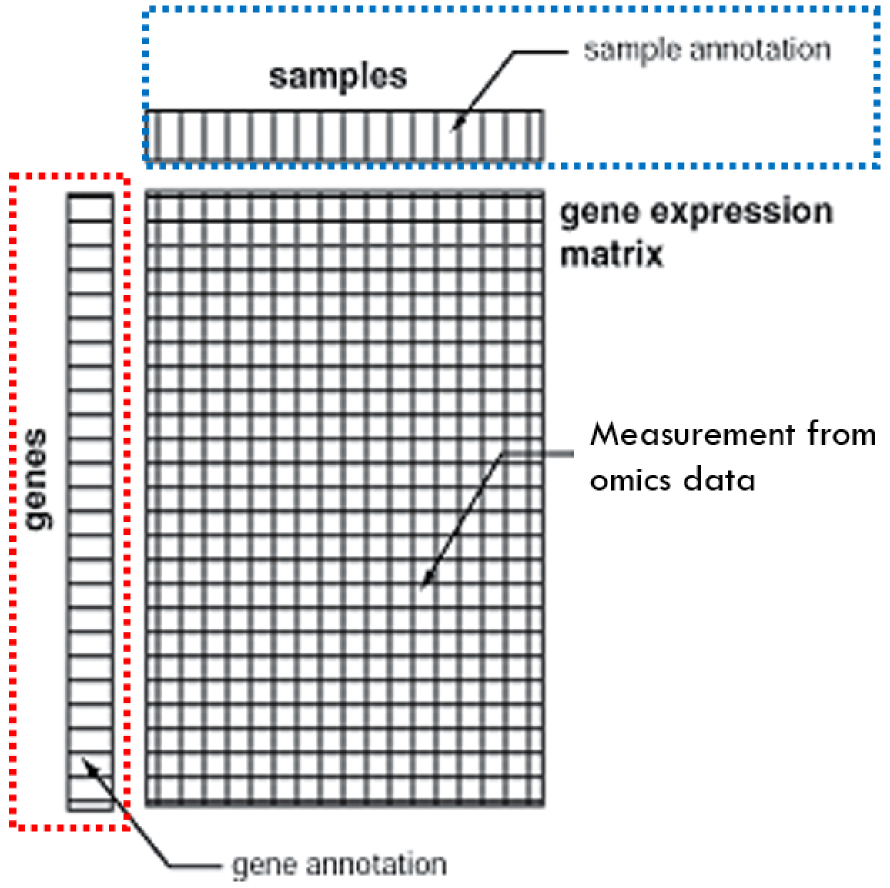
```
load("data/GSE46474.RData")
head(gse$title)
```

```
## [1] "WB_AR1" "WB_NR1" "WB_AR2" "WB_NR2" "WB_AR3" "WB_NR3"
```

```
Outcome <- ifelse(grepl("AR", gse$title), "Rejection", "Stable") #Tidy the title variable and call i
table(Outcome)
```

```
## Outcome
## Rejection    Stable
##        20        20
```

Gene Expression data structure



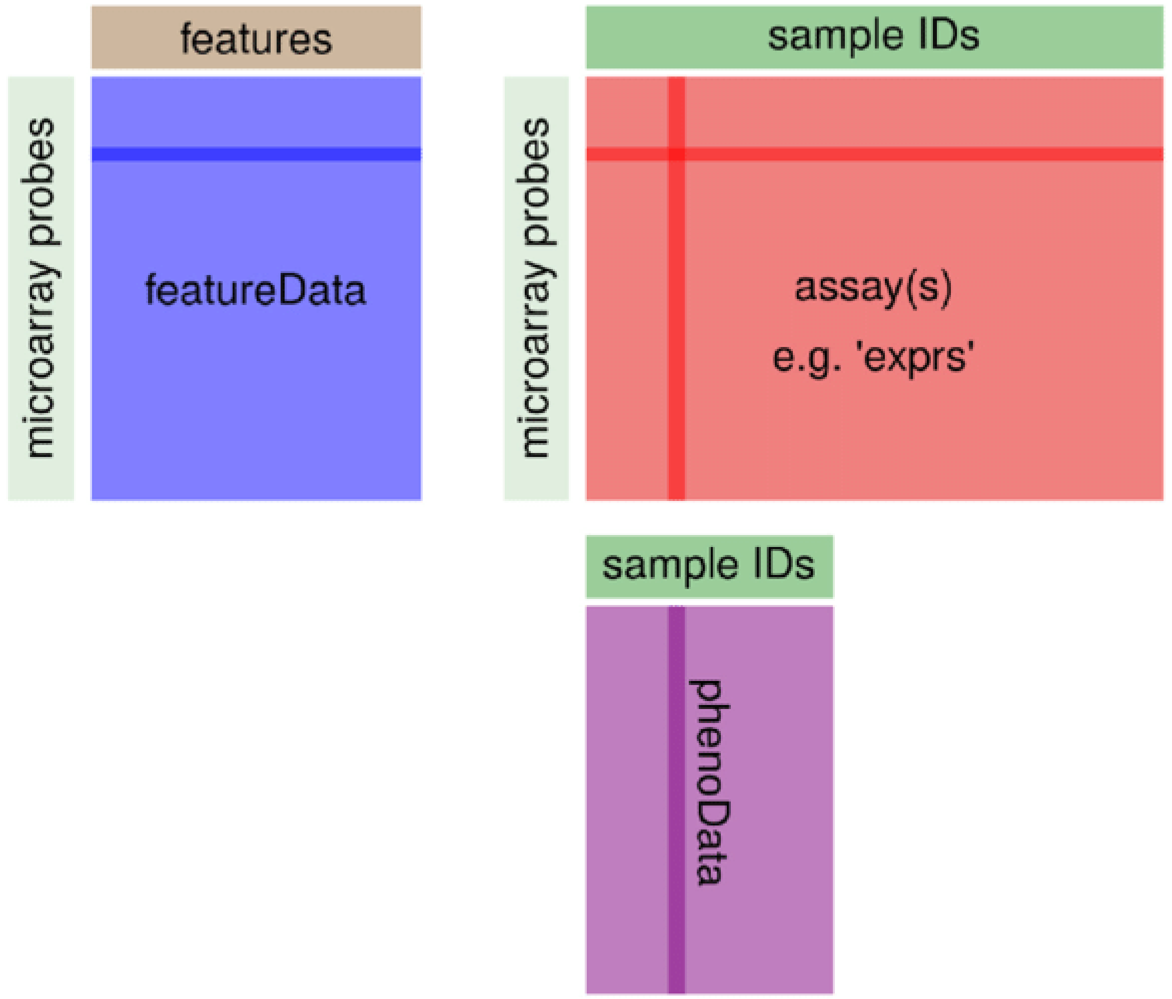
Data structure - ExpressionSet

```
slotNames(gse)
```

```
## [1] "experimentData"    "assayData"          "phenoData"
## [4] "featureData"       "annotation"         "protocolData"
## [7] ".__classVersion__"
```

- `gse` is an object of class *ExpressionSet* that has three main slots:
- [1] `assayData`: Contains a *matrix* of expression values for each gene (feature) measured.
- [2] `phenoData`: Contains a *data frame* of sample information. i.e Covariates.
- [3] `featureData`: Contains a *data frame* of gene (feature) information. There are other slots with other information
- `experimentData`: Contains *text* information about the design of the experiment.
- `annotation`: Contains a *Character* describing the type of platform the samples were sequenced on.

Data structure - ExpressionSet



ExpressionSet - gene expression

- Gene expression data can be retrieved by the `exprs` function from Biobase package. Notice how we have a matrix where our rows are probes which correspond to a gene of interest, and our columns correspond to a sample that has been sequenced.

```
eMat = exprs(gse)
eMat[1:4,1:3]
```

```
##           GSM1130812 GSM1130813 GSM1130814
## 1007_s_at   5.702192   6.454067   6.113615
## 1053_at     4.816291   4.918209   4.686569
## 117_at      7.950576   6.726106   7.437923
## 121_at      7.643679   7.193593   7.482420
```

ExpressionSet - phenotype data

Retrieve information on experimental phenotypes (i.e. covariates) by

```
pheno <- phenoData(gse)
colnames(pheno)
```

```
##  [1] "title"                     "geo_accession"
##  [3] "status"                    "submission_date"
##  [5] "last_update_date"          "type"
##  [7] "channel_count"             "source_name_ch1"
##  [9] "organism_ch1"              "characteristics_ch1"
## [11] "characteristics_ch1.1"     "characteristics_ch1.2"
## [13] "characteristics_ch1.3"     "characteristics_ch1.4"
## [15] "characteristics_ch1.5"     "characteristics_ch1.6"
## [17] "molecule_ch1"              "extract_protocol_ch1"
## [19] "label_ch1"                 "label_protocol_ch1"
## [21] "taxid_ch1"                 "hyb_protocol"
## [23] "scan_protocol"             "description"
## [25] "data_processing"           "data_processing.1"
## [27] "platform_id"               "contact_name"
## [29] "contact_email"             "contact_phone"
## [31] "contact_department"        "contact_institute"
## [33] "contact_address"           "contact_city"
## [35] "contact_state"             "contact_zip/postal_code"
## [37] "contact_country"           "supplementary_file"
## [39] "data_row_count"            "age_tx:ch1"
## [41] "collection_day_post_tx:ch1" "procedure status:ch1"
## [43] "race:ch1"                  "sample group:ch1"
## [45] "Sex:ch1"                   "tissue:ch1"
```

ExpressionSet - phenotype data

```
class(pheno)
```

```
## [1] "AnnotatedDataFrame"
## attr(,"package")
## [1] "Biobase"
```

```
pheno$`procedure status:ch1`[1:4]
```

```
## [1] "post-transplant acute rejection (AR)"
## [2] "post-transplant non-rejection (NR)"
## [3] "post-transplant acute rejection (AR)"
## [4] "post-transplant non-rejection (NR)"
```

```
table(pheno$`sample group:ch1`)
```

```
##
##   control discovery
##        20        20
```

```
pheno$`age_tx:ch1`
```

```
##  [1] "55.7" "50.3" "52.6" "44.8" "38.8" "51.1" "55.4" "35.5" "31.8" "55.2"
## [11] "37"   "58.4" "27"   "46.6" "50.3" "55.3" "58.7" "47.6" "57.5" "58.1"
## [21] "38"   "47.3" "45.1" "40.1" "42"   "56.8" "51.8" "54.6" "37.6" "50.8"
## [31] "49.7" "49.5" "43.4" "58.6" "50.2" "20.7" "20.3" "42.3" "71.8" "57.1"
```

ExpressionSet - feature data

The `fData` function retrieve information on features recorded, that is, gets the probes' annotation.

```
featureData <- fData(gse)
featureData[1:5, 1:5]
```

```
##                   ID GB_ACC SPOT_ID Species Scientific Name Annotation Date
## 1007_s_at 1007_s_at U48705      NA            Homo sapiens     Oct 6, 2014
## 1053_at     1053_at M87338      NA            Homo sapiens     Oct 6, 2014
## 117_at       117_at X51757      NA            Homo sapiens     Oct 6, 2014
## 121_at       121_at X69699      NA            Homo sapiens     Oct 6, 2014
## 1255_g_at 1255_g_at L36861      NA            Homo sapiens     Oct 6, 2014
```

Dimensions

- What are the dimensions of our data set?

```
eMat = exprs(gse)
dim(eMat)
```

```
## [1] 54613    40
```

- We have a $p > n$ problem! (p = parameters, n = sample size)
- Many machine learning methods assume or require $n > p$.
- In this case, we could even say $p >> n$.

Feature selection

Feature Selection

- When we have a $p > n$ problem, we may hypothesise that most of the features aren't useful for our task (eg. classification).
- Feature selection is the process of only selecting a (small) subset of features, so that we can apply our standard machine learning tools.
- The challenge here is how to identify the "useful features".
- For our task, useful features could be genes whose expression is different between the two groups (stable vs rejection).
- A naive approach is to perform a t-test!

Feature Selection: Gene 1

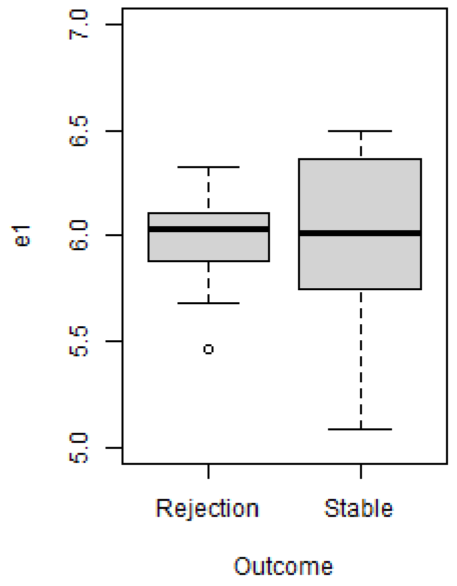
Looking at the first gene:

```r
e1 = eMat[1,]
t.test(e1 ~ Outcome)
```

```
##
##     Welch Two Sample t-test
##
## data:  e1 by Outcome
## t = -0.15104, df = 28.352, p-value = 0.881
## alternative hypothesis: true difference in means between group Rejection and group Stable is not equal
## 95 percent confidence interval:
##  -0.2211218  0.1907354
## sample estimates:
## mean in group Rejection    mean in group Stable
##                5.986109                6.001303
```

Feature Selection: Gene 1

```
par(mfrow=c(1,2))
boxplot(e1 ~ Outcome, ylim=c(5, 7))
```



- It seems like the first gene isn't that useful in differentiating the Rejection and Stable patients

Feature Selection: Gene 23127

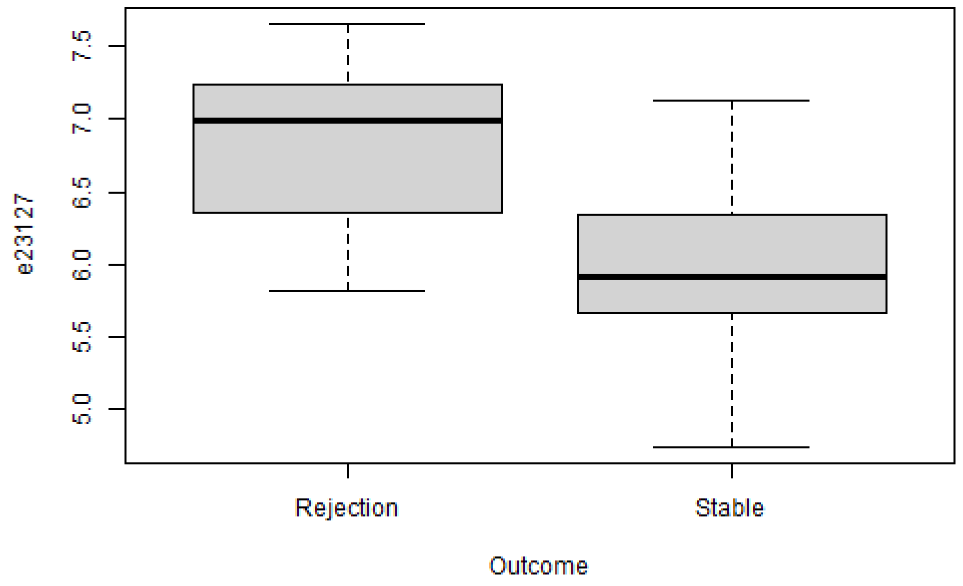
Lets repeat this again with another gene. Lets us select the 23127th gene and have a look.

```
e23127 = eMat[23127,]
e23127_t = t.test(e23127 ~ Outcome)
e23127_t
```

```
##
##      Welch Two Sample t-test
##
## data:  e23127 by Outcome
## t = 4.7974, df = 37.595, p-value = 2.55e-05
## alternative hypothesis: true difference in means between group Rejection and group Stable is not equal
## 95 percent confidence interval:
##  0.4914719 1.2095060
## sample estimates:
## mean in group Rejection    mean in group Stable
##                6.809849                5.959360
```

Feature Selection: Gene 23127

```
boxplot(e23127 ~ Outcome)
```



- It seems like the 23127th gene could be useful in differentiating the Rejection and Stable patients!
- But how do we perform feature selection to find the most useful features?

Using linear regression

Compare linear regression versus t-test

```
fit = lm(e23127 ~ factor(Outcome))
summary(fit)
```

```
##
## Call:
## lm(formula = e23127 ~ factor(Outcome))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.2139 -0.3792  0.1260  0.4068  1.1738
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)             6.8098     0.1254  54.323  < 2e-16 ***
## factor(Outcome)Stable  -0.8505     0.1773  -4.797  2.5e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5606 on 38 degrees of freedom
## Multiple R-squared:  0.3772,    Adjusted R-squared:  0.3608
## F-statistic: 23.01 on 1 and 38 DF,  p-value: 2.495e-05
```

Using linear regression

Compare linear regression versus t-test

```r
e23127 = eMat[23127,]
e23127_t = t.test(e23127 ~ Outcome, var.equal = TRUE)
e23127_t
```

```
##
##      Two Sample t-test
##
## data:  e23127 by Outcome
## t = 4.7974, df = 38, p-value = 2.495e-05
## alternative hypothesis: true difference in means between group Rejection and group Stable is not equal
## 95 percent confidence interval:
##  0.4915989 1.2093790
## sample estimates:
## mean in group Rejection    mean in group Stable
##                6.809849                5.959360
```

Finding differentially expressed genes

Which genes will be the most useful for classification? That is, which genes have the most different expression between stable and rejection patients.

- Use a `for` loop to perform a series of t-tests.
- Faster options with `limma` R package - speeds up this process.
- Fit a series of linear models

$$Y = X\beta$$

  where X is known as the `design` matrix.
- The theory and methods are covered in STAT3022 - Applied linear models.

DE genes using limma

```
design <- model.matrix(~Outcome)
design[1:5,]
```

```
##   (Intercept) OutcomeStable
## 1           1             0
## 2           1             1
## 3           1             0
## 4           1             1
## 5           1             0
```

```
Outcome[1:5]
```

```
## [1] "Rejection" "Stable"    "Rejection" "Stable"    "Rejection"
```

DE genes using limma

```r
library(limma)
fit <- lmFit(eMat, design) ## Fitting linear model
fit <- eBayes(fit) ## Calculate moderated t-stats and p-value
topTable(fit, n = 5) %>% signif(3) ## Output top ranking genes
```

```
## Removing intercept from test coefficients

##                logFC AveExpr     t  P.Value adj.P.Val    B
## NA.8986       -1.060    8.87 -7.10 1.12e-08  0.000614 9.11
## 210686_x_at -0.756    9.10 -6.79 3.11e-08  0.000849 8.25
## 202028_s_at -0.893    8.33 -6.35 1.31e-07  0.002280 7.01
## NA.4677      -0.688    6.87 -6.28 1.67e-07  0.002280 6.81
## NA.4200      -1.420    7.32 -5.96 4.86e-07  0.005300 5.88
```

- `logFC` = log(Fold Change), `AveExpr` = Average Expression.
- Which columns are useful to us?

DE genes using limma

- Adding gene symbols to a table

```
topTable(fit, genelist=featureData[,"Gene Symbol"], n = 5)
```

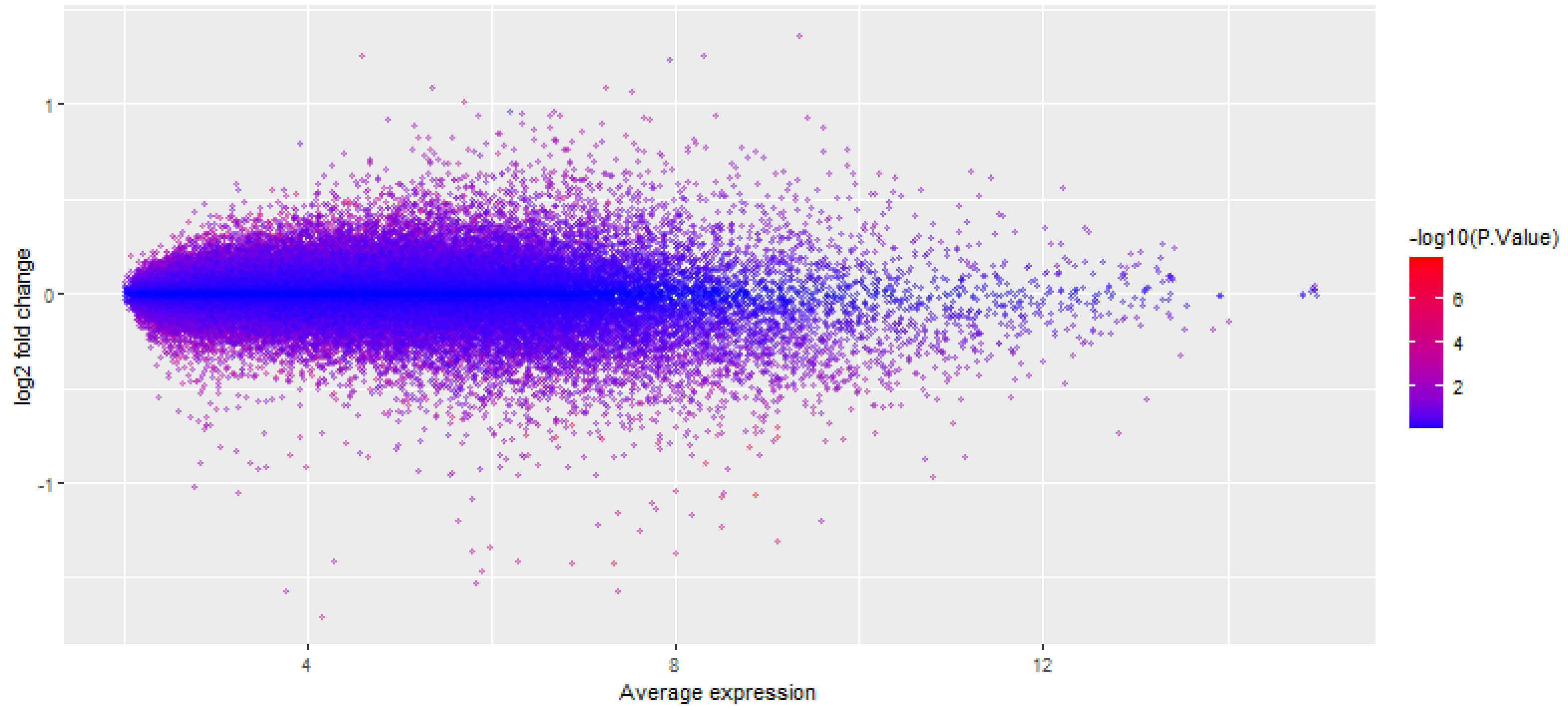
```
## Removing intercept from test coefficients

##                   ID      logFC  AveExpr         t      P.Value    adj.P.Val
## NA.8986         <NA> -1.0601112 8.866138 -7.103210 1.123539e-08 0.0006135986
## 210686_x_at SLC25A16 -0.7555083 9.095867 -6.791756 3.108334e-08 0.0008487773
## 202028_s_at    RPL38 -0.8933696 8.326946 -6.353216 1.311499e-07 0.0022774794
## NA.4677         <NA> -0.6881895 6.871418 -6.280092 1.668086e-07 0.0022774794
## NA.4200         <NA> -1.4234446 7.323691 -5.955004 4.860532e-07 0.0052974747
##                    B
## NA.8986     9.111266
## 210686_x_at 8.245293
## 202028_s_at 7.012077
## NA.4677     6.805216
## NA.4200     5.882729
```

Limitations

- Does the **p-value** tell you the whole story?
- Not necessarily!
- We often want to consider the **effect size** as well (here, `logFC`). Small effect sizes may not be meaningful.
- We may also want to consider the **magnitude** as well (here, `AveExpr`). Lowly expressed genes may not be interesting.
- Let us try visualising our features!

MA-plot

- This plots the `logFC` against `AveExpr`, often coloured by the "significance" (here, -log10(P.value))
- The "M" stands for "Minus" and the "A" stands for "Add".
- Where are the important features located on this plot?

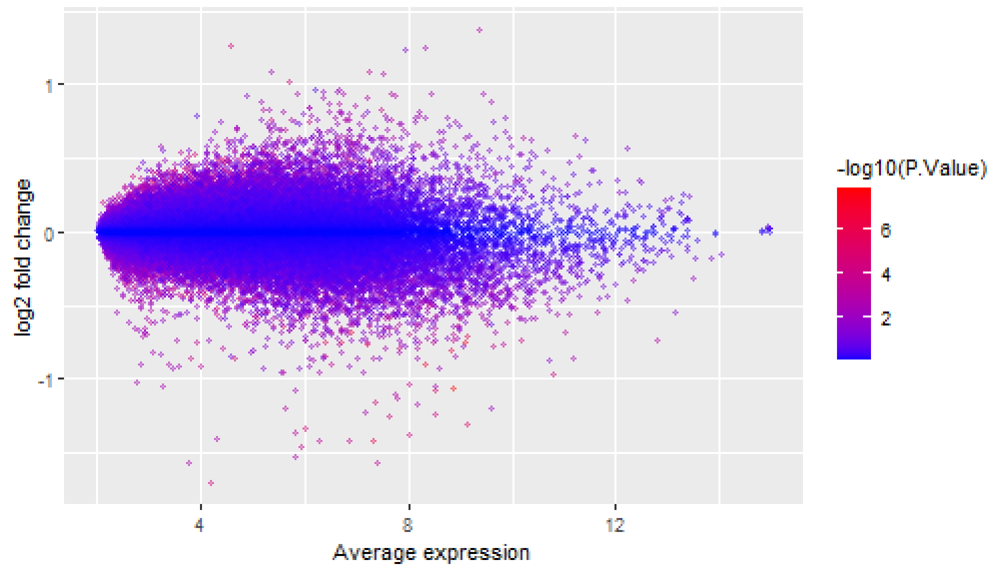


MA-plot code

```
library(ggplot2)
df<- topTable(fit, number=nrow(fit), genelist=rownames(gse))
```

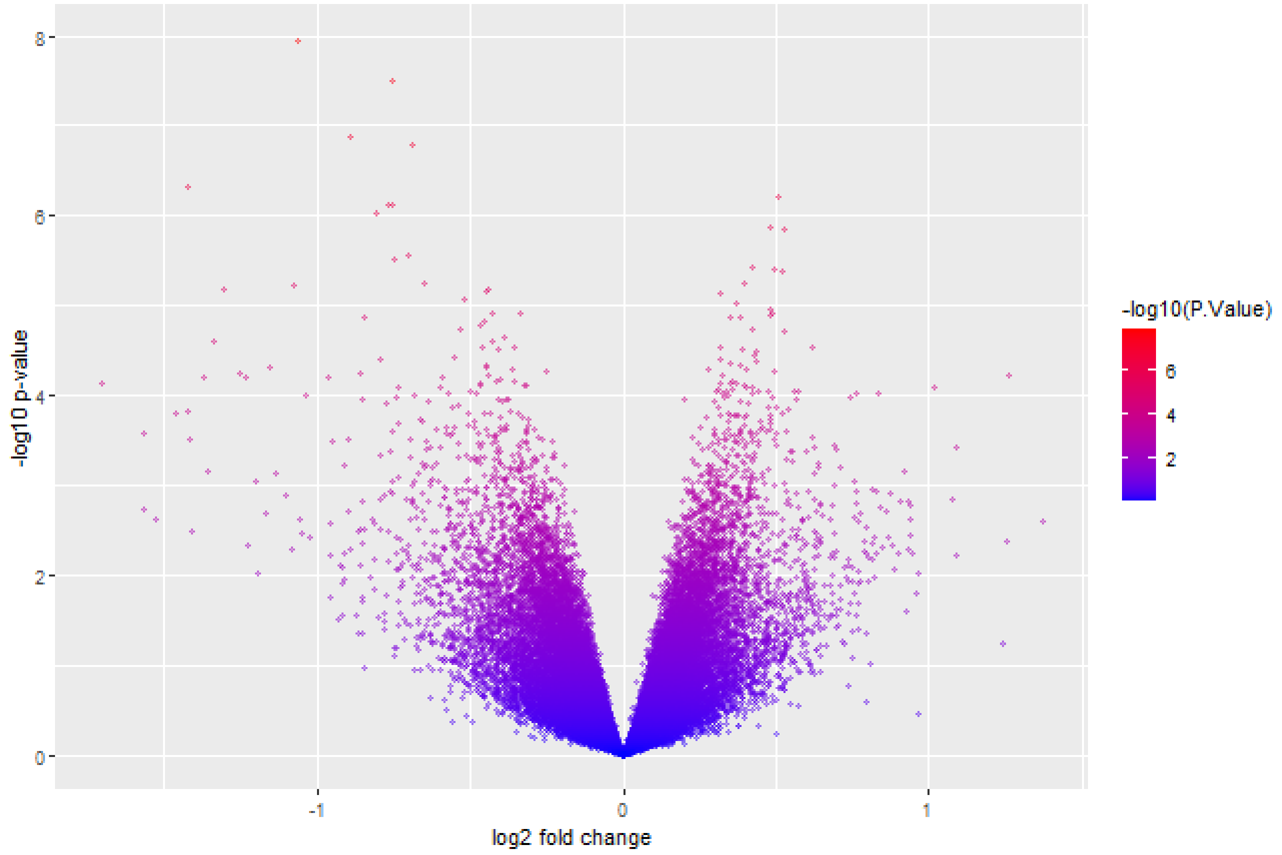
```
## Removing intercept from test coefficients
```

```
ggplot(df, aes(x = AveExpr, y = logFC))+
    geom_point(aes(colour=-log10(P.Value)), alpha=1/3, size=1) +
    scale_colour_gradient(low="blue",high="red")+
    ylab("log2 fold change") + xlab("Average expression")
```



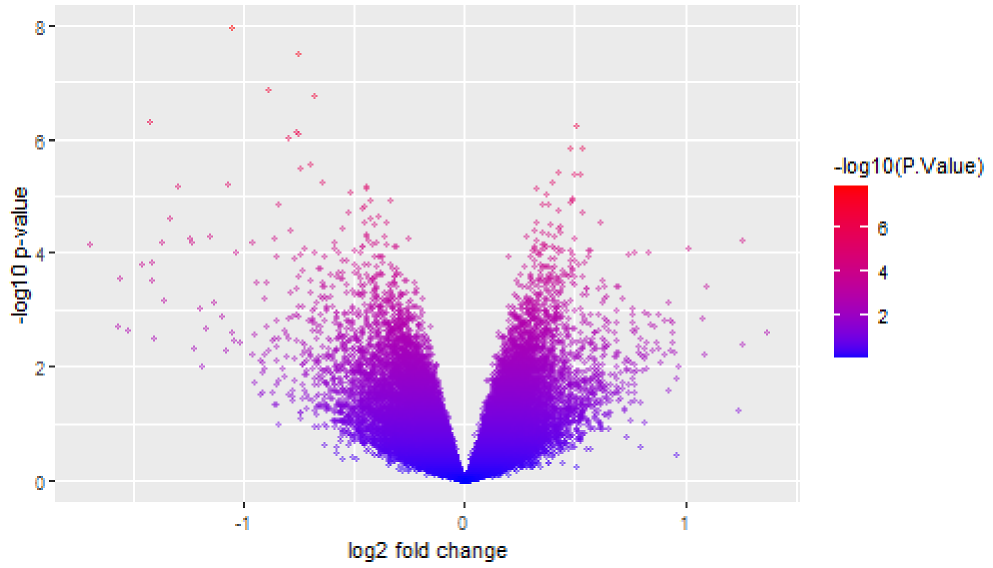
Volcano plot

- This plots the `logFC` against against the -log10(P.value).
- We generally observe a volcano-like pattern.
- Where are the important features located on this plot?



Volcano plot code

```
p <- ggplot(df, aes(logFC,-log10(P.Value)))+
    geom_point(aes(colour=-log10(P.Value)), alpha=1/3, size=1) +
    scale_colour_gradient(low="blue",high="red")+
    xlab("log2 fold change") + ylab("-log10 p-value")
p
```



Summary

- **Feature selection** is a useful tool, especially for $p > n$ problems.
- One method for feature selection (for a classification task) is to identify features that are different among the two groups of interest.
- We can visualise the features with an MA-plot and volcano plot.