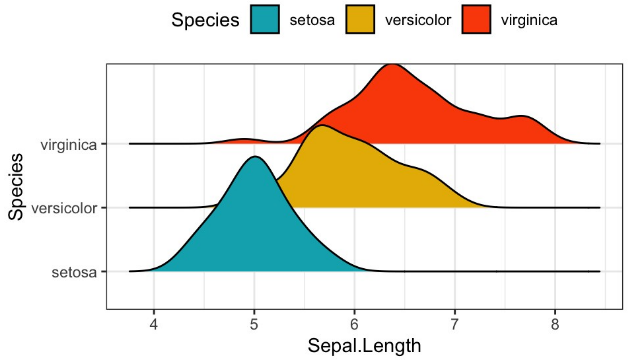
Illustrating expression data in R

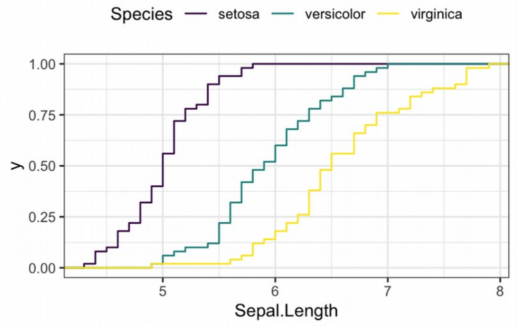
Bioinformatics workshops
Jakob Willforss
191023

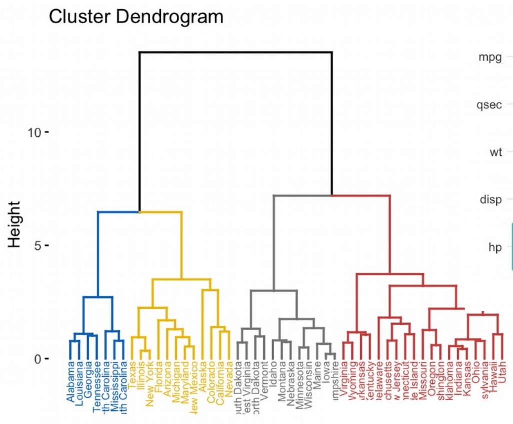
What is ggplot2?

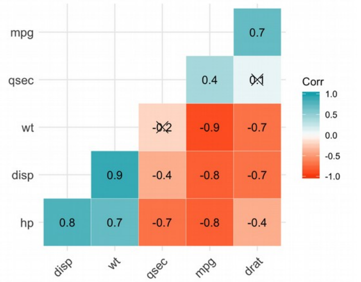
- Data visualization **package** in R
- Created by Hadley Wickham
- Part of a collection of R packages called **Tidyverse**
- Very flexible and powerful tool for visualizations
- Master a limited set of ideas and you can do a wide array of reproducible visualizations

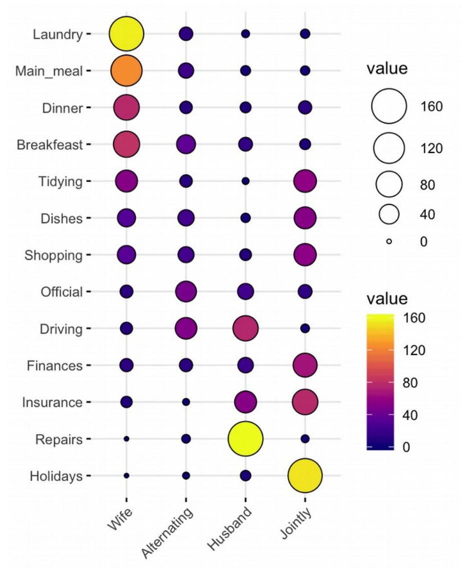
The difficult part when working with ggplot is not ggplot itself, but getting the input data in the correct format!

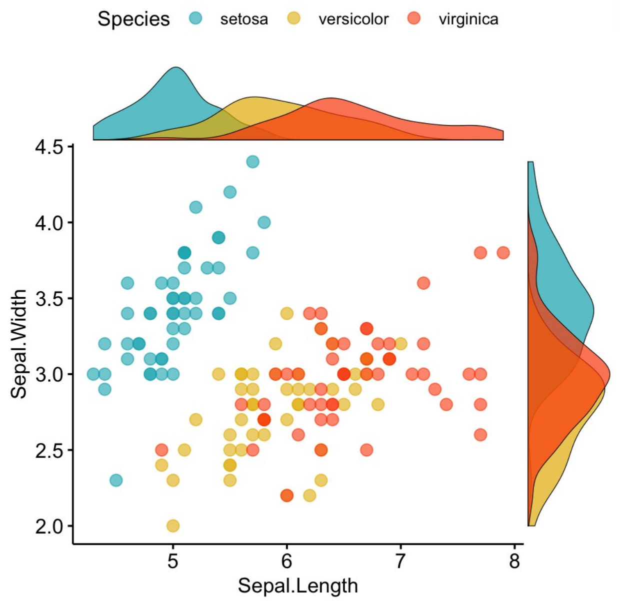


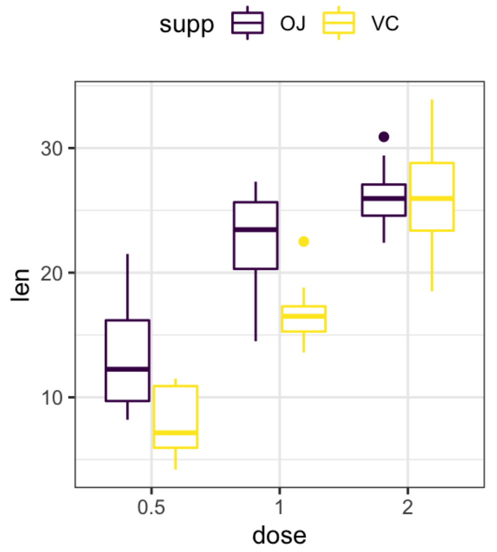






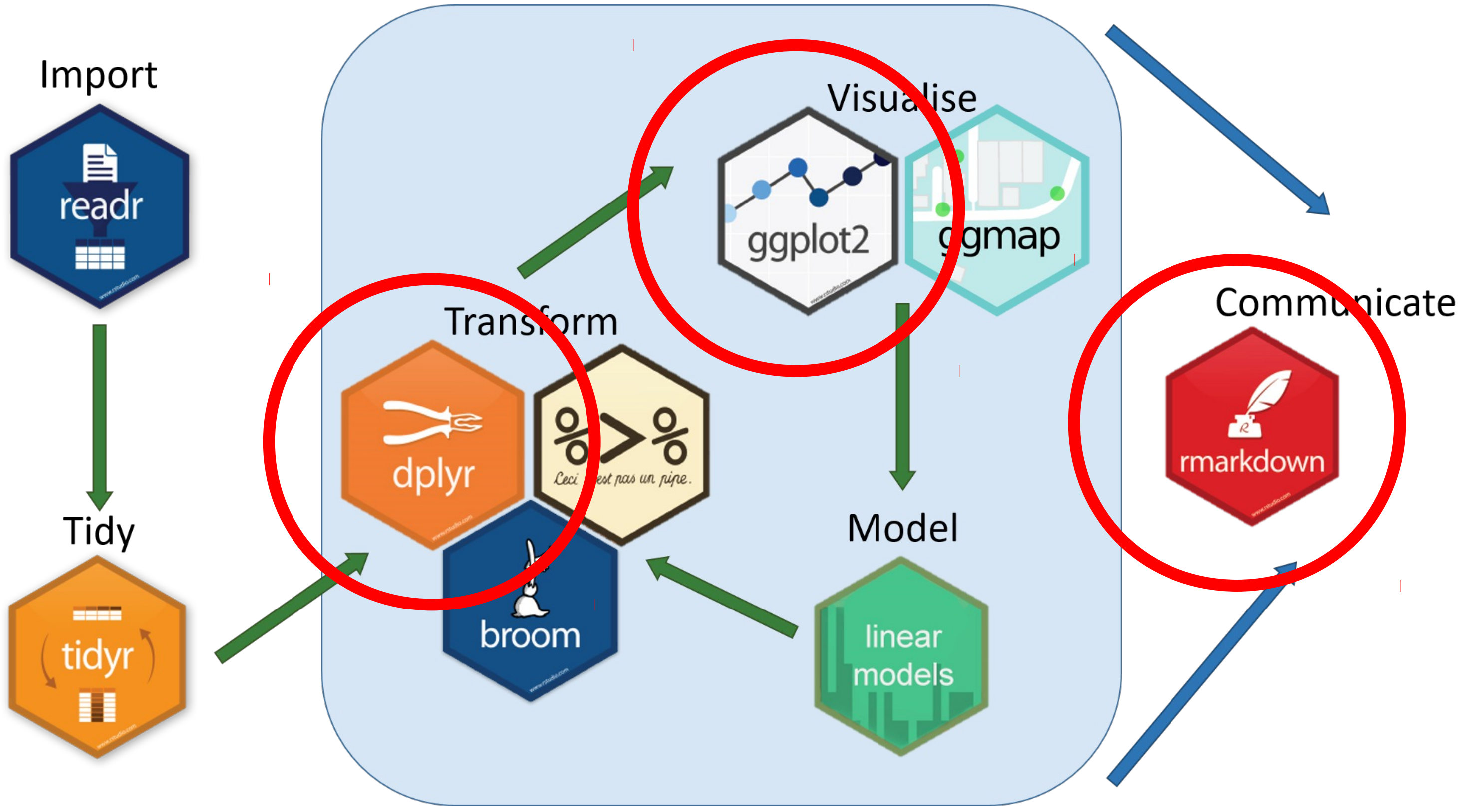






Some examples from: https://www.datanovia.com/en/blog/ggplot-examples-best-reference/
All on public datasets with runnable code available

Tidyverse: A collection of R packages



https://medium.com/@kadek/how-to-install-the-tidyverse-r-via-homebrew-macos-10-14-d749d2136cf1

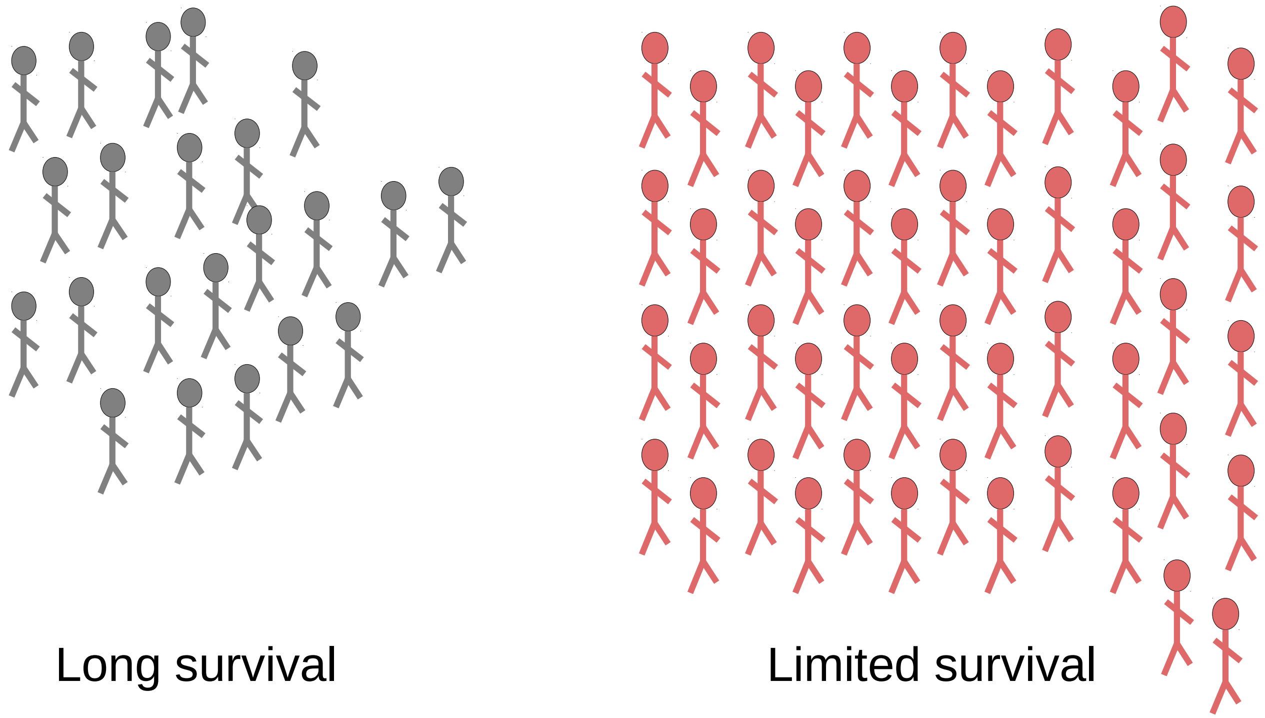
Demonstration example: Array data

Setup:

- 72 array samples
- Survival times
- MIPIcat levels

Preprocessing:

NormalyzerDE



Long survival

PFS = 1

Patients: 22

Limited survival

PFS = 0

Patients: 50

The data and the design

Design matrix

Sample information

sample	PFS	PFS_years	MIPIcat
id1	0	4.3	1
id2	1	9.3	2
id3	1	9.5	1
id4	0	2.3	3

Data matrix

Gene information

Gene	FDR	fold	id1	id2	id3	id4
A	0.1	1.2	7.5	6.4	4.5	6.5
B	0.9	-2.3	10.2	9.8	11.2	10.3

The sample names is found in one column in the design and should be present as columns in the data

Loading design matrix

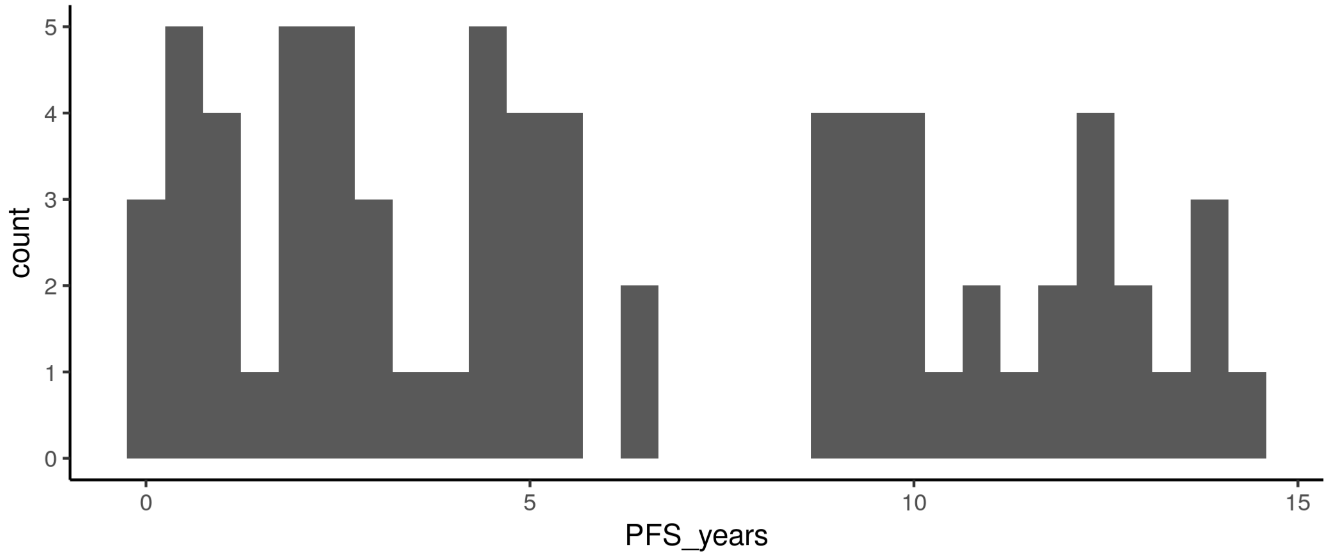
```
> design_fp ← ″joana_data/joana_design.tsv″
> design_df ← read.table(design_fp, sep=″\t″, header=TRUE, stringsAsFactors=FALSE)
> dim(design_df)
[1] 72 5
> colnames(design_df)
[1] "ID"         "PFS"        "PFS_years" "MIPIcat"   "array_id"
> head(design_df)
```

```
ID<chr> PFS<int>    PFS_years<dbl>  MIPIcat<int> array_id<chr>
1    MCL2_006   1    1.547     3    p1615_01_MCL2_006.CEL
2    MCL2_007   1    12.686    1    p1615_02_MCL2_007.CEL
3    MCL2_008   0    13.994    1    p1615_43_MCL2_008.CEL
4    MCL2_013   0    12.458    1    p1615_04_MCL2_013.CEL
5    MCL2_031   1    13.100    1    p1615_06_MCL2_031.CEL
6    MCL2_032   0    12.175    1    p1615_08_MCL2_032.CEL
```

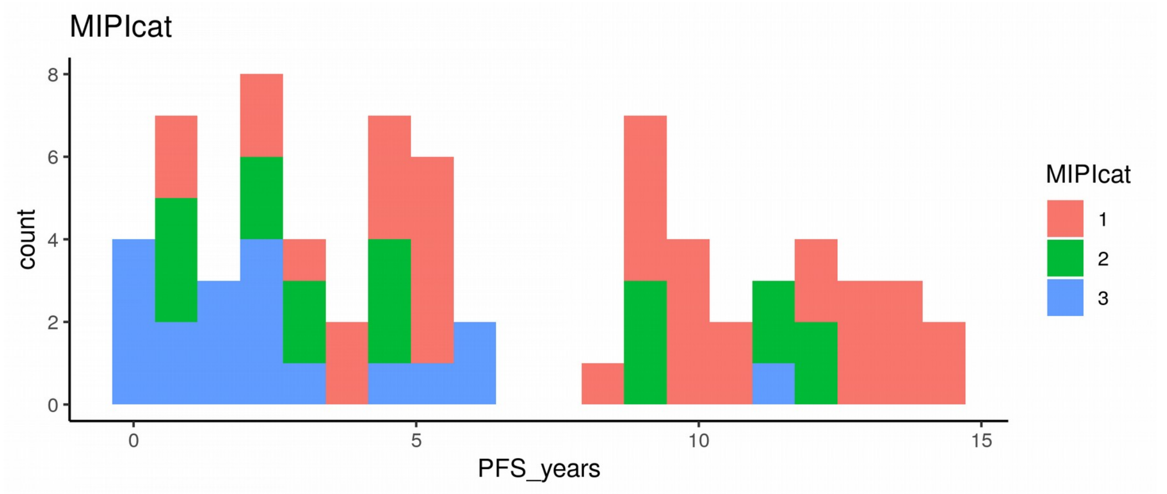
array_id column matches
sample names in data matrix

Visualizing PFS_years

```
> ggplot(design_df, aes(x=PFS_years)) + geom_histogram()
```

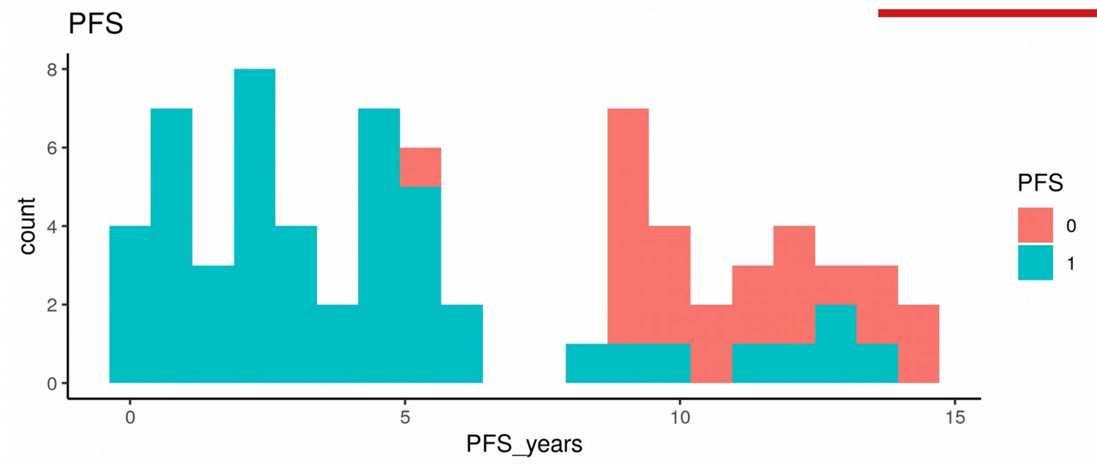


```
> ggplot(design_df, aes(x=PFS_years, fill=MIPIcat)) + geom_histogram()
```



We can specify different columns to use for coloring

```
> ggplot(design_df, aes(x=PFS_years, fill=PFS)) + geom_histogram()
```



The ggplot2 command

The target data frame

```
ggplot(design_df, aes(x=PFS_years)) + geom_histogram()
```

Specify aesthetics columns
from the target data

geometric - how should the
data with aesthetics
be illustrated

The ggplot2 command

The target data frame

```
ggplot(design_df, aes(x=PFS_years)) + geom_histogram()
```

Specify aesthetics columns
from the target data

Common aesthetics are:
x, y, color, fill, label, shape

geometric - how should the
data with aesthetics
be illustrated

Common geoms are:
geom_histogram()
geom_point()
geom_line()
geom_col()
… and many many more

The data and the design

Design matrix

Sample information

sample	PFS	PFS_years	MIPIcat
id1	0	4.3	1
id2	1	9.3	2
id3	1	9.5	1
id4	0	2.3	3

Data matrix

Gene information

Gene	FDR	fold	id1	id2	id3	id4
A	0.1	1.2	7.5	6.4	4.5	6.5
B	0.9	-2.3	10.2	9.8	11.2	10.3

The sample names is found in one column in the design and should be present as columns in the data

Let’s demonstrate on the data

```
> data_fp ← ″joana_data/normalyzerde_stats.tsv″
> data_df ← read.table(data_fp, sep=″\t″, header=TRUE, stringsAsFactors=FALSE)
> dim(data_df)
[1] 9190 80
> colnames(data_df)
[1] "PROBEID"                "SYMBOL"                 "GENENAME"               "array.iD"
 [5] "P.Value"                "adj.P.Val"              "log2Fold"               "AvgExpr"
 [9] "p1615_01_MCL2_006.CEL"  "p1615_02_MCL2_007.CEL"  "p1615_43_MCL2_008.CEL"  "p1615_04_MCL2_013.CEL"
[13] "p1615_06_MCL2_031.CEL"  "p1615_08_MCL2_032.CEL"  "p1615_61_MCL2_038.CEL"  "p1615_52_MCL2_039.CEL"
[17] "p1615_10_MCL2_048.CEL"  "p1615_11_MCL2_054.CEL"  "p1615_12_MCL2_058.CEL"  "p1615_13_MCL2_059.CEL"
> head(data_df)
```

```
SYMBOL <chr>     P.Value <dbl> adj.P.Val <dbl> log2Fold <dbl> AvgExpr <dbl>  (then sample cols)
LOC100287497     0.86          0.98            0.01           7.00           6.48
LINC01128        0.80          0.97            0.01           4.29           4.44
SAMD11           0.83          0.97            0.01           6.34           6.14
KLHL17           0.29          0.86            -0.04          5.83           5.75
PLEKHN1          0.15          0.79            -0.08          6.34           6.10
ISG15            0.13          0.77            0.14           6.19           6.51
```

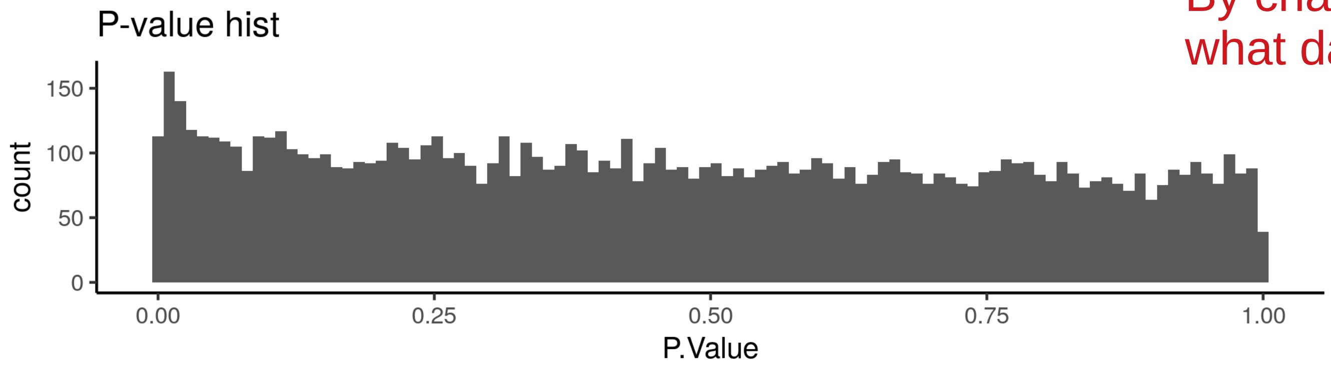
Specifying what features are ‘significant’

Returns a vector with TRUE and FALSE values, which is assigned to a new column

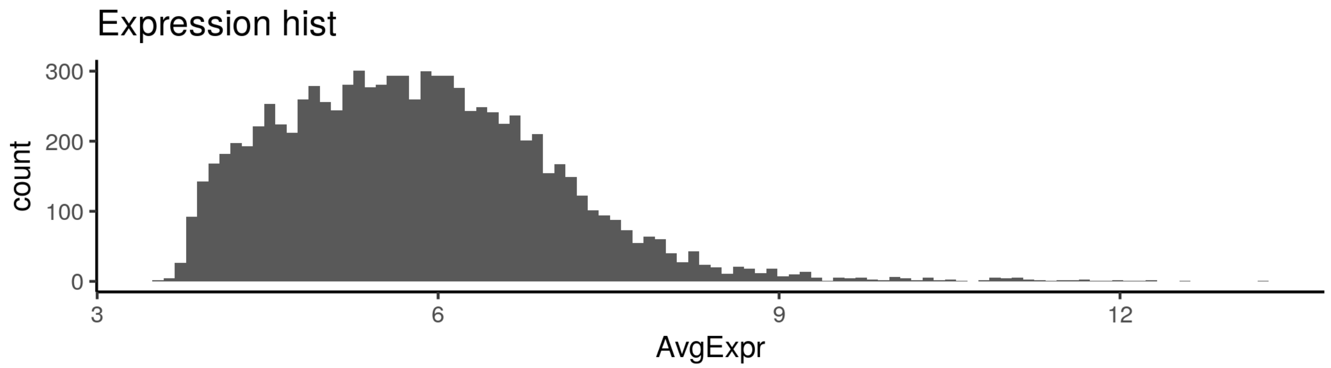
```
> fdr_thres ← 0.25
> p_thres ← 0.05
> data_df$IsPSig ← data_df$P.Value < p_thres
> data_df$IsFDRSig ← data_df$adj.P.Val < fdr_thres
> # head(data_df) to double check
> table(data_df$IsSig)
FALSE  TRUE
 8487   702
> table(data_df$IsFDRSig)
FALSE  TRUE
 9178    11
```

```
> ggplot(data_df, aes(x=P.Value)) + geom_histogram(bins=100) +
ggtitle("P-value hist")
```

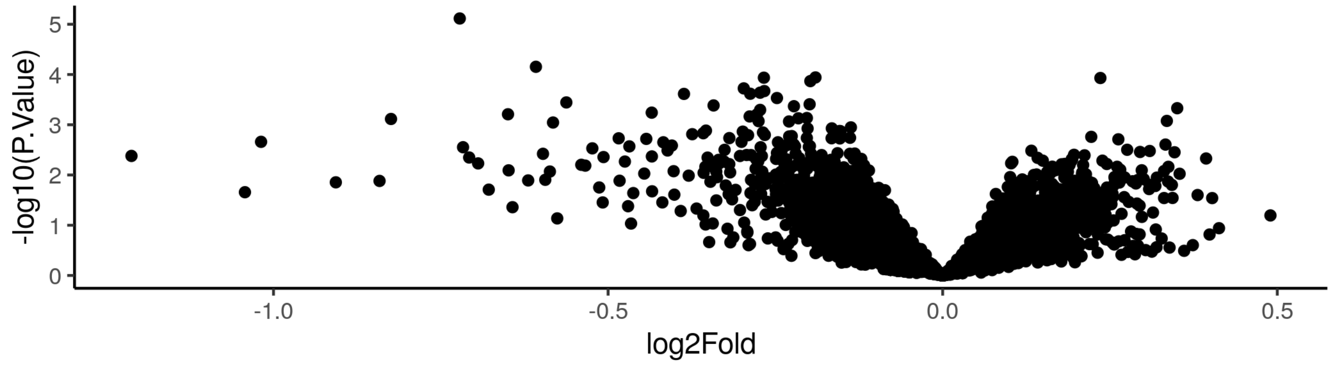
By changing x-aesthetics we change what data is shown as a histogram



```
> ggplot(data_df, aes(x=AveExpr)) + geom_histogram(bins=100) +
ggtitle("Expression hist")
```

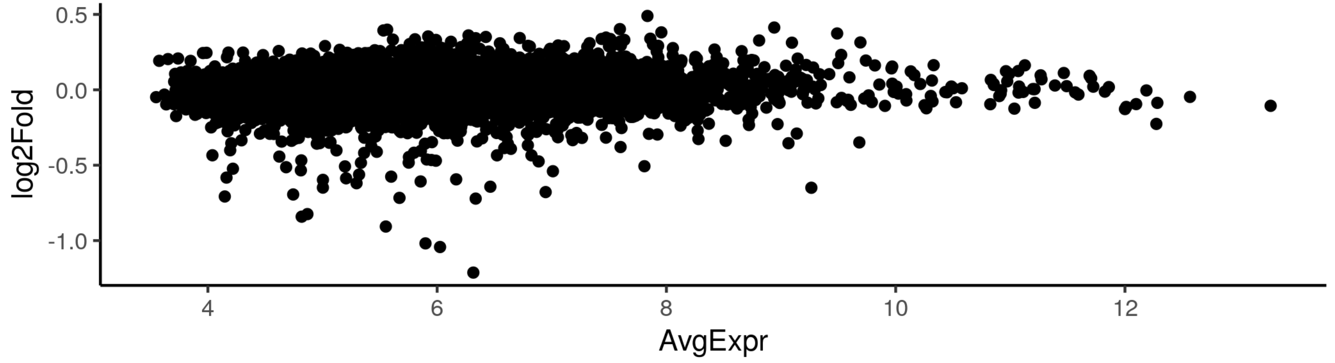


```
> ggplot(data_df, aes(x=log2Fold, y=-log10(P.Value))) + geom_point()
```

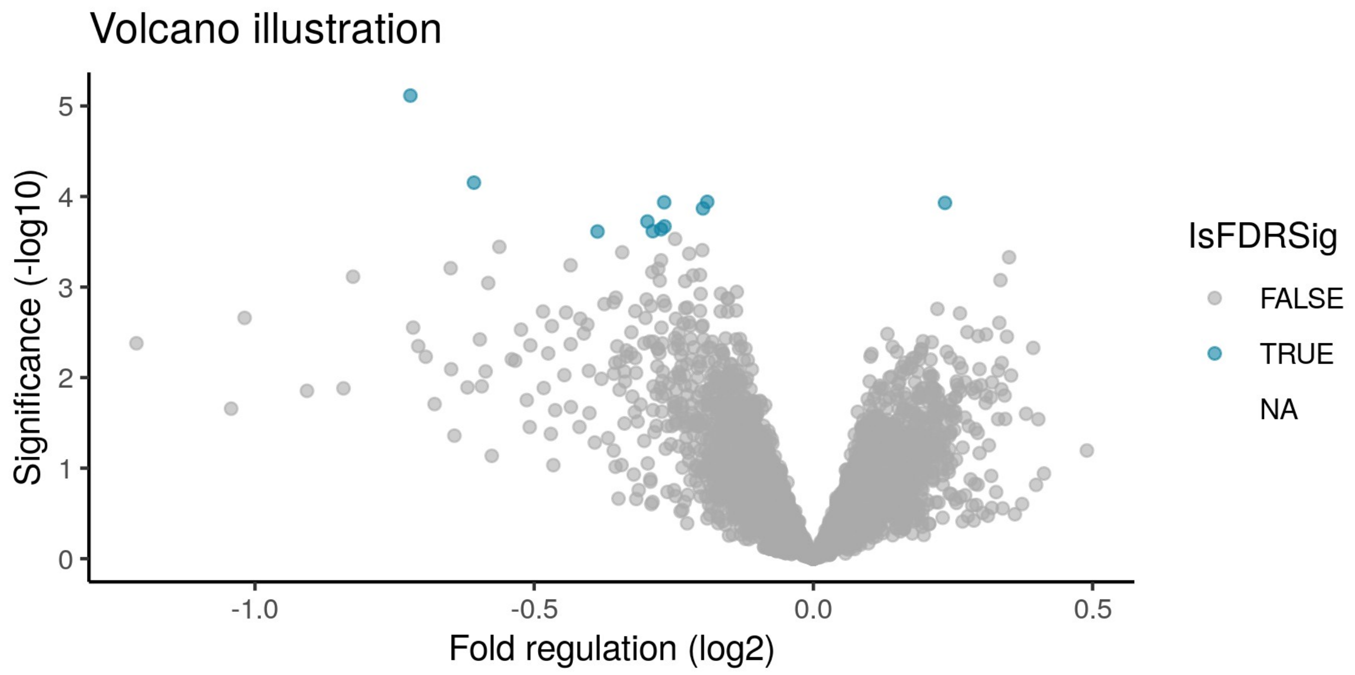


Again, we simply change the aesthetics to use different columns in data_df

```
> ggplot(data_df, aes(x=AvgExpr, y=log2Fold)) + geom_point()
```



```
gray_blue_colors ← c("#AAAAAA", "#0C81A2")
ggplot(full_data_df, aes(x=log2Fold, y=-log10(P.Value), color=IsFDRSig)) +
    geom_point(alpha=0.6) +
    ggtitle("Volcano illustration") +
    xlab("Expression level") +
    ylab("Fold change (log2)") +
    scale_color_manual(values=gray_blue_colors)
```



Hands-on time!

Single feature illustration

We want to: Illustrate the feature with lowest FDR

Single feature illustration

```
> target_cols ← c("SYMBOL", "GENENAME", "adj.P.Val", "log2Fold")
> top_hits ← head(arrange(data_df, P.Value), 5)
> top_hits[, target_cols]
```

```
SYMBOL <chr>   GENENAME <chr>                    adj.P.Val <dbl> log2Fold <dbl>
SH3BGRL2       SH3 domain binding glutamate ...  0.14            -0.72
KRT19          keratin 19                        0.21            -0.60
SLC2A10        solute carrier family 2 member 10 0.21            -0.19
PLEKHA5        pleckstrin homology domain ...    0.21            -0.26
TLR9           toll like receptor 9              0.21            0.23
```

arrange is a Tidyverse command organizing the data frame in rising order based on given column - note that the P.Value column is specified without quotes (""), same as within ggplot's aes

top_hits[, target_cols] slices out only the given columns

Getting the best hits

```
> best_hit ← data_df[data_df$SYMBOL == ″SH3BGRL2″, ]


> best_hit_vals <- best_hit[, design_df$array_id]


> best_hit_df <- cbind(value=unlist(best_hit_vals), design_df)
```

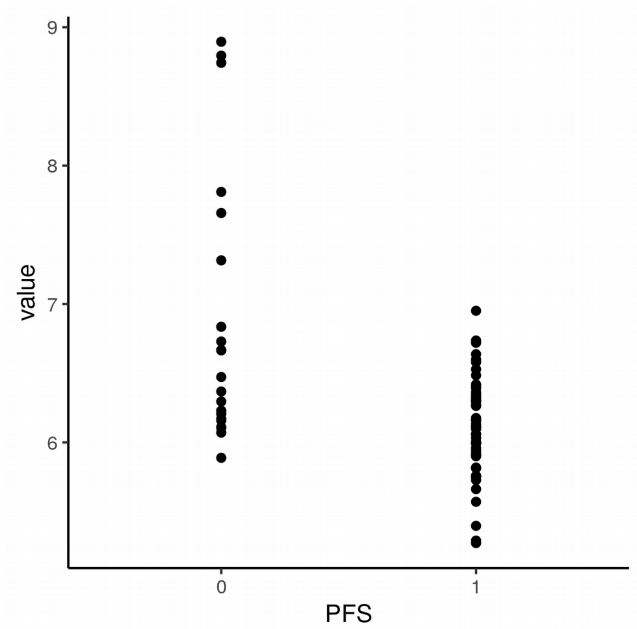
We bind the values from SH3BGRL2 to the design matrix, calling the new column ‘value’

```
> head(best_hit_df, 4)
value <dbl> ID <chr> PFS <fctr> PFS_years <dbl> MIPIcat <fctr> array_id <chr>
5.29       MCL2_006   1   1.547   3   p1615_01_MCL2_006.CEL
6.32       MCL2_007   1   12.686  1   p1615_02_MCL2_007.CEL
7.81       MCL2_008   0   13.994  1   p1615_43_MCL2_008.CEL
6.15       MCL2_013   0   12.458  1   p1615_04_MCL2_013.CEL
```

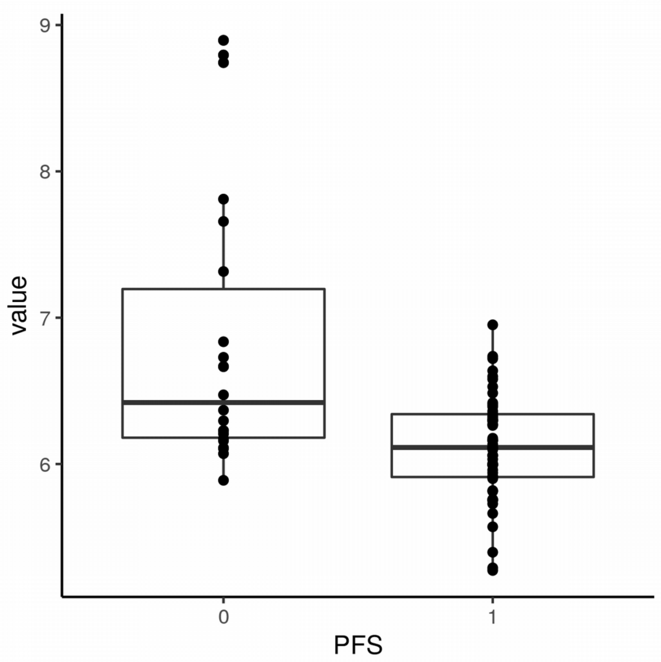
```
ggplot(best_hit_df, aes(x=PFS, y=value)) + geom_point()

ggplot(best_hit_df, aes(x=PFS, y=value)) + geom_boxplot() + geom_point()

ggplot(best_hit_df, aes(x=PFS, y=value)) + geom_violin() + geom_point()
```

Here, we illustrate the data differently simply by changing the geom-layers

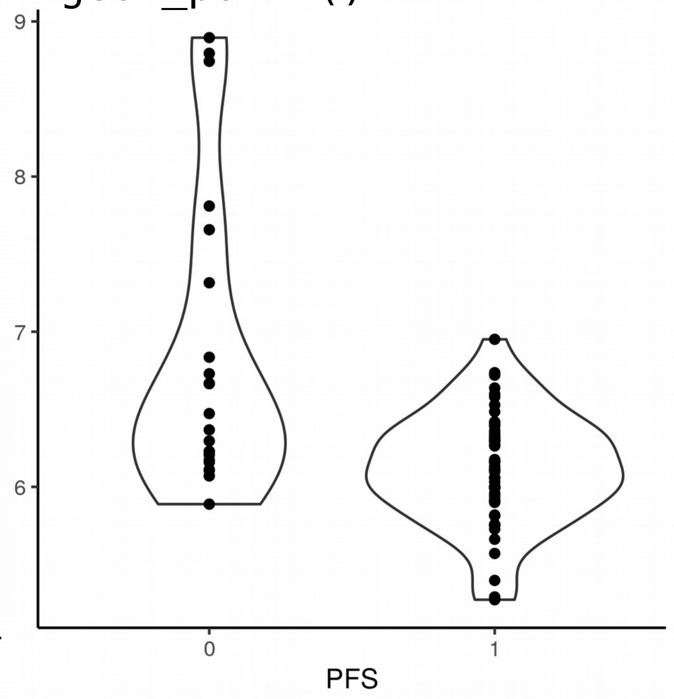
`geom_point()`



`geom_boxplot() + geom_point()`



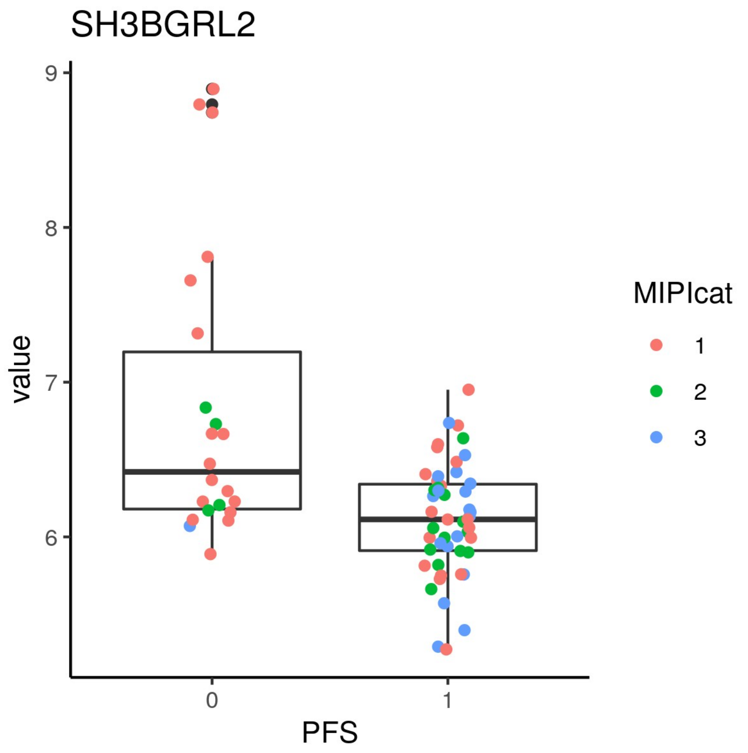
`geom_violin() + geom_point()`



```
ggplot(best_hit_df, aes(x=PFS, y=value)) +
    geom_boxplot() +
    geom_point(position=position_jitter(0.1), aes(color=MIPIcat)) +
    ggtitle("SH3BGRL2")
```

Jitter ‘shakes up’ the positions of the dots

If we only want to color one layer, we specify the aesthetic within it



Hands-on time!

Wide versus long format

- Expression data is commonly found in ″wide″ format
- ggplot expects the input to be in ″long″ format

Wide format

Gene	ID1	ID2	ID3
gene_A	9.4	7.5	8.4
gene_B	6.5	6.7	7.3

Long format

Gene	ID	value
gene_A	ID1	9.4
gene_A	ID2	7.5
gene_A	ID3	8.4
gene_B	ID1	6.5
gene_B	ID2	6.7
gene_B	ID3	7.3

The data and the design

Design matrix

Sample information

sample	PFS	PFS_years	MIPIcat
id1	0	4.3	1
id2	1	9.3	2
id3	1	9.5	1
id4	0	2.3	3

Data matrix

Gene information

Gene	FDR	fold	id1	id2	id3	id4
A	0.1	1.2	7.5	6.4	4.5	6.5
B	0.9	-2.3	10.2	9.8	11.2	10.3

The sample names is found in one column in the design and should be present as columns in the data

Converting to long format

```
> dim(data_df)
9190 82
> long_df ← pivot_longer(data_df, design_df$sample,
names_to="array_id", values_to="value")
> dim(long_df)
661680 12
> head(long_df)[, c(″SYMBOL″, ″sample″, ″value″)]
SYMBOL <chr>   array_id <chr>           value <dbl>
LOC100287497   p1615_01_MCL2_006.CEL   6.48
LOC100287497   p1615_02_MCL2_007.CEL   6.87
LOC100287497   p1615_43_MCL2_008.CEL   8.49
LOC100287497   p1615_04_MCL2_013.CEL   7.47
LOC100287497   p1615_06_MCL2_031.CEL   7.05
LOC100287497   p1615_08_MCL2_032.CEL   6.81
```

Now we have only one column with values, with the ″array_id″ column specifying from where each value comes

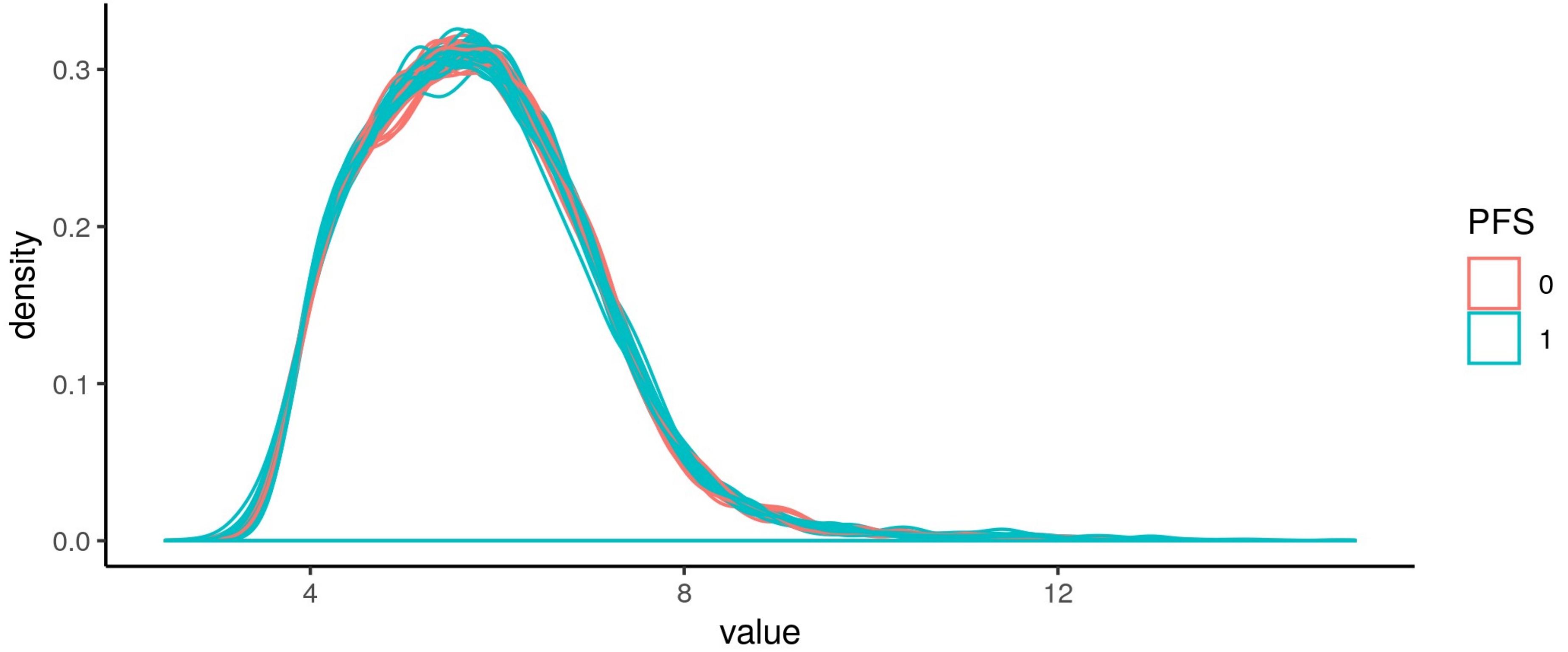
Adding sample annotations

```
> annot_long_df ← left_join(long_df, design_df, by=
″array_id″)
> head(annot_long_df)
ID <chr> PFS <fctr> PFS_years <dbl> MIPIcat <fctr>
p1615_01_MCL2_006.CEL   6.48     MCL2_006 1  1.54     3
p1615_02_MCL2_007.CEL   6.87     MCL2_007 1  12.68    1
p1615_43_MCL2_008.CEL   8.49     MCL2_008 0  13.99    1
p1615_04_MCL2_013.CEL   7.47     MCL2_013 0  12.45    1
p1615_06_MCL2_031.CEL   7.05     MCL2_031 1  13.10    1
p1615_08_MCL2_032.CEL   6.81     MCL2_032 0  12.17    1
```

″left_join″ from Tidyverse lets us merge information from another matrix by specifying a common column. Here, the design-matrix information is added based on the array IDs. Now we have everything we need for sample-wide illustrations.

The density plot

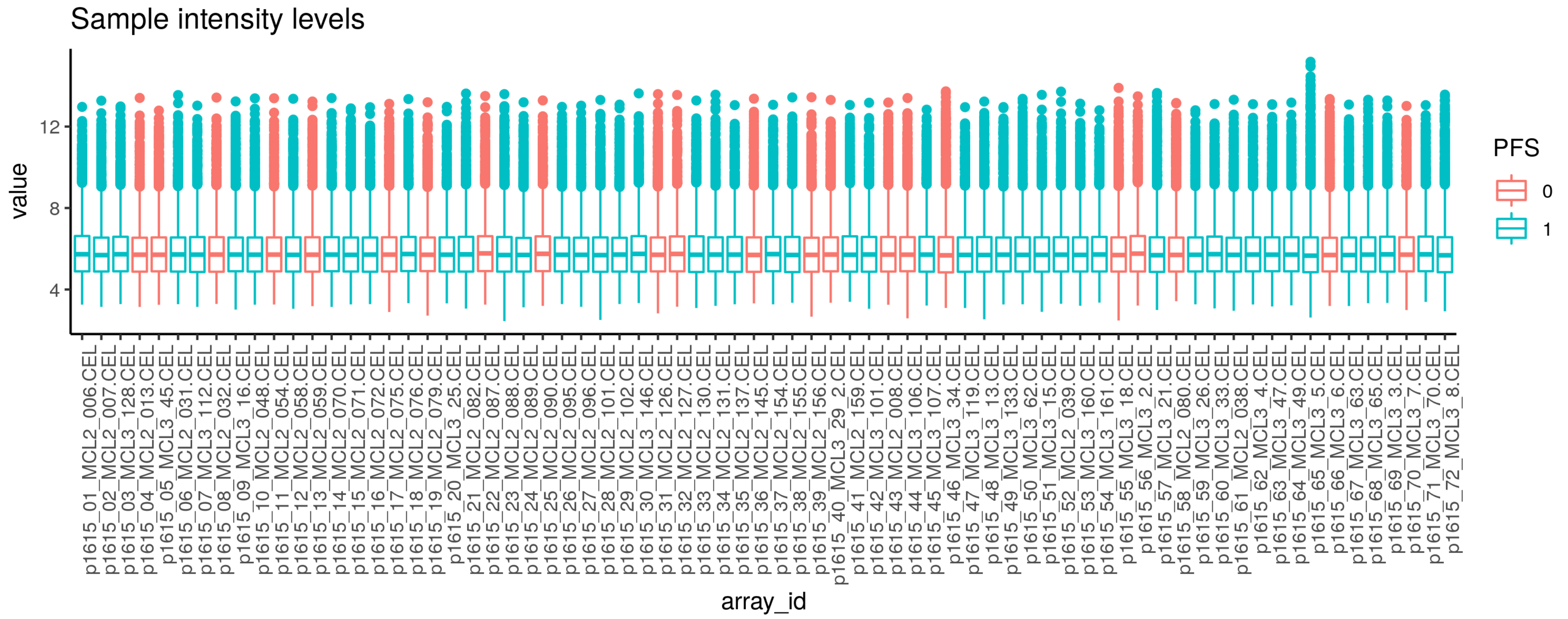
```
> ggplot(annot_long_data, aes(x=value, group=array_id,
color=PFS)) + geom_density()
```



Sample boxplots

```
> ggplot(annot_long_data, aes(x=array_id, y=value,
color=PFS)) + geom_boxplot() + theme(axis.text.x =
element_text(angle=90, hjust=1)) +
ggtitle("Sample intensity levels")
```

Rotate axis-x labels



Saving your plots

```
> plt_obj ← ggplot(annot_long_data, aes(x=value,
group=array_id, color=PFS)) + geom_density()

> ggsave(plt_obj, filename=″plots/density.png″, width=7,
height=7, dpi=300)
```

- plt_obj - An object containing the plot
- filename - Where to write the new plot. Note that the file ending is important - here we generate a figure in PNG format
- width / height - Size specified in inches
- dpi - The resolution of the written figure

Investigating further plots

Chord diagrams
https://www.r-graph-gallery.com/chord-diagram.html

Survival curves
https://rpkgs.datanovia.com/survminer/index.html

Enrichment illustrations
https://yulab-smu.github.io/clusterProfiler-book/

The End