



Intro to Rstudio

via “R” statistical computing language

(don't let the language scare you :)



Outline:

1. Our data
2. How do we get our data
3. How do we store our data
4. How do we test and visualize our data

Bonus: basic machine learning techniques via packages

Take Home:

- Show your friends/parents your analysis via cloud... & be able to explain!

1. **Code + data:**

<https://github.com/dcolinmorgan/r-jupyterlite-website/tree/main/lab/workspaces>

📄 910_example.Rmd

We will run this example

📄 910_example.html

Web version (with output)

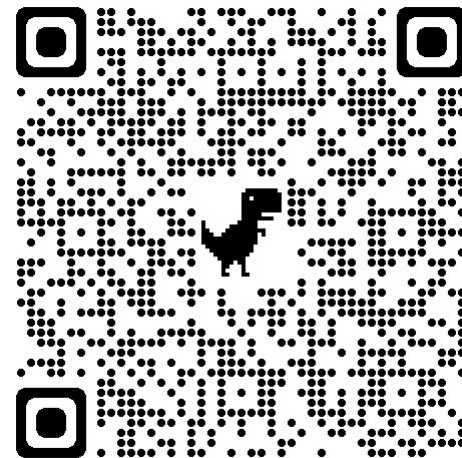
📄 910_example.ipynb

Python version (skip)

📄 example.ipynb

📄 heart_failure_clinical_records_dataset.csv

Dataset we will read in



2. **you can run Rstudio in cloud:**

posit.cloud



1. Our data: Cardiovascular diseases

- Cardiovascular diseases (CVDs) are the **number 1 cause of death globally**, taking an estimated **17.9 million lives each year**, which accounts for **31% of all deaths worldwide**.



1. Our data: Cardiovascular diseases

- Cardiovascular diseases (CVDs) are the **number 1 cause of death globally**, taking an estimated **17.9 million lives each year**, which accounts for **31% of all deaths worldwide**.
- Most cardiovascular diseases can be prevented by addressing behavioural risk factors such as tobacco use, unhealthy diet and obesity, physical inactivity and harmful use of alcohol using population-wide strategies.



1. Our data: Cardiovascular diseases

- Cardiovascular diseases (CVDs) are the **number 1 cause of death globally**, taking an estimated **17.9 million lives each year**, which accounts for **31% of all deaths worldwide**.
- Most cardiovascular diseases can be prevented by addressing behavioural risk factors such as tobacco use, unhealthy diet and obesity, physical inactivity and harmful use of alcohol using population-wide strategies.
- People with cardiovascular disease or who are at high cardiovascular risk (due to the presence of one or more risk factors such as hypertension, diabetes, hyperlipidaemia or already established disease) need **early detection** and management wherein a machine learning model can be of great help.
→ Therefore prediction of heart failure in patients is important



Research Article | [Open Access](#) | Published: 03 February 2020

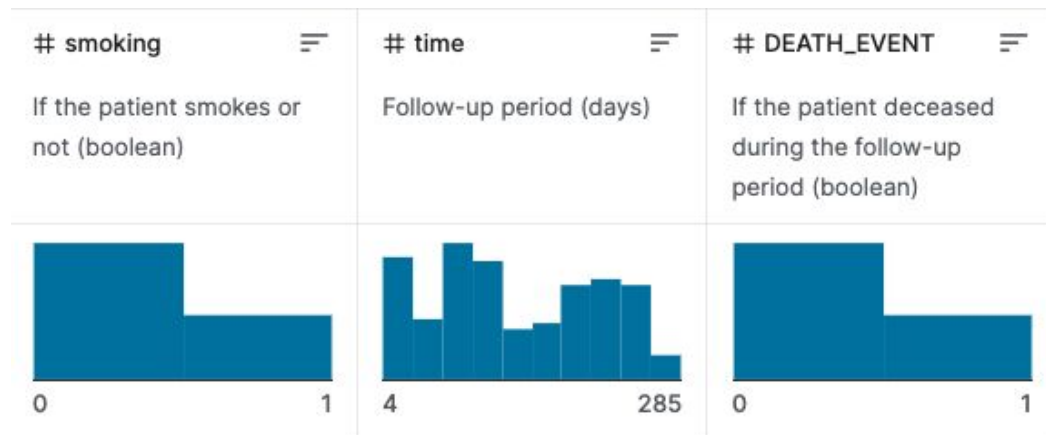
Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone

[Davide Chicco](#) & [Giuseppe Jurman](#)

BMC Medical Informatics and Decision Making 20, Article number: 16 (2020) | [Cite this article](#)

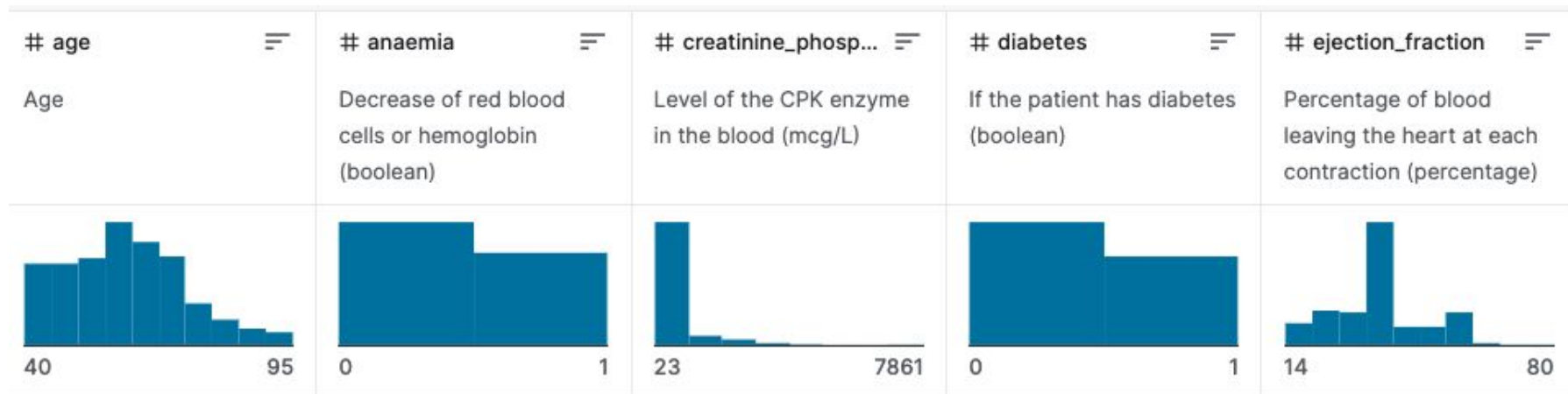
148k Accesses | 203 Citations | 26 Altmetric | [Metrics](#)

Heart failure is a common event caused by CVDs and this dataset contains 12 features that can be used to predict mortality by heart failure.



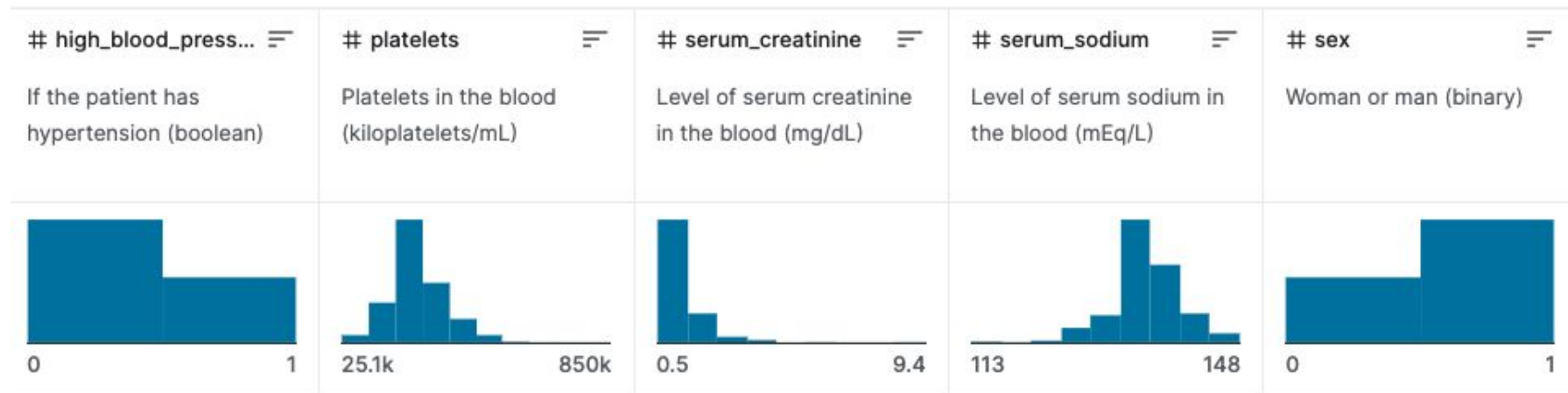


1. Our data: Cardiovascular diseases





1. Our data: Cardiovascular diseases





0. How do we get ...

1. Our data: Cardiovascular diseases

*Optical Character Recognition
is designed to convert your
handwriting into text.*

Biological Information:

Optical Character Recognition
is designed to convert your
handwriting into text.

- Sound (language)
- Touch (physical interactions)
- Taste/Smell (chemistry)
- Light (everything above... + basically everything)
 - Reflected light “digitizes” the molecules of life





1. Our data: Cardiovascular diseases
2. How do we store our data



1. Our data: Cardiovascular diseases
2. How do we store our data

Data and handling it in a coding environment:

- Excel table == 2D matrix

age	anaemia	creatinine	phosphokinase	diabetes	ejection_fraction
75	0		582	0	20
55	0		7861	0	38
65	0		146	0	20
50	1		111	0	20
65	1		160	1	20
90	1		47	0	40

- Can call these “data frames”
 - contain row and column info – nice metadata to have during analysis
 - **Rows** are generally subjects or samples or repeated measures
 - Various measures are represented in different **columns**



1. Our data: Cardiovascular diseases
2. How do we store our data
3. How do we visualize our data



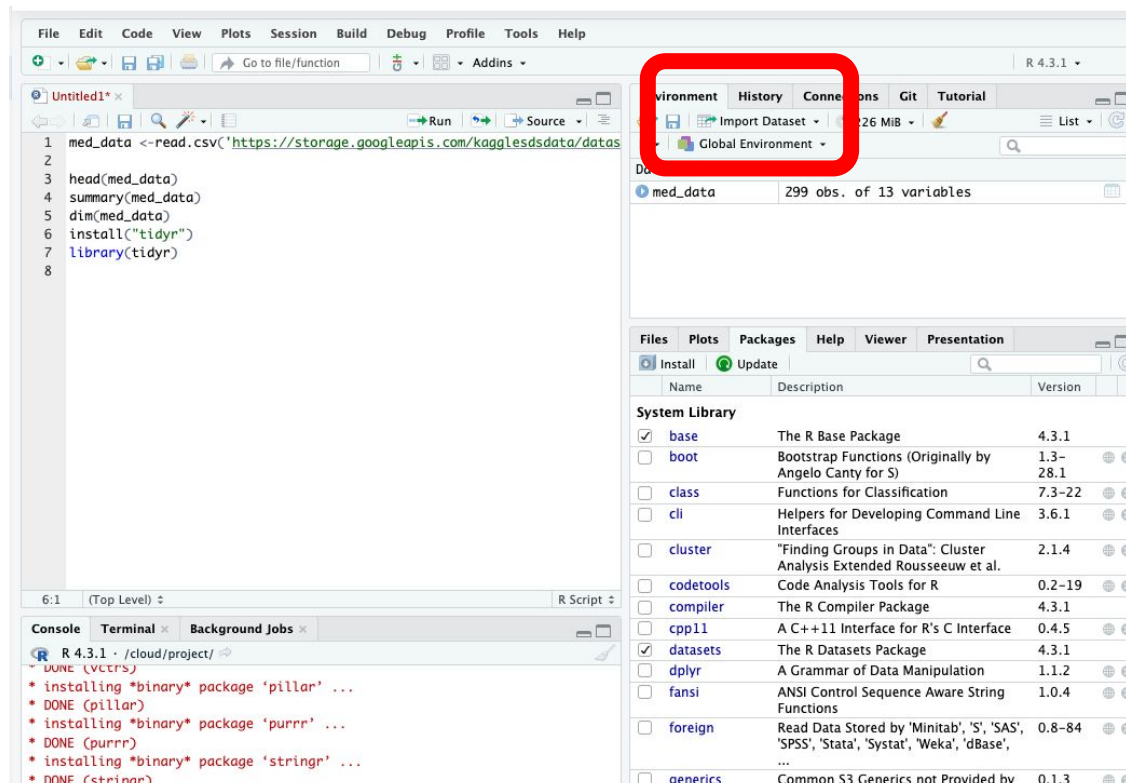
1. Our data: Cardiovascular diseases
2. How do we store our data
3. How do we test and visualize our data...

Rstudio

Local computer

or

posit.cloud



The screenshot displays the RStudio IDE interface. The main editor window shows an R script with the following code:

```
1 med_data <- read.csv('https://storage.googleapis.com/kagglestdsdata/datas
2
3 head(med_data)
4 summary(med_data)
5 dim(med_data)
6 install("tidyr")
7 library(tidyr)
8
```

The Environment pane on the right shows a variable named `med_data` with 299 observations and 13 variables. The Packages pane on the right lists installed and available packages. A red box highlights the 'Global Environment' button in the Environment pane.

Name	Description	Version
System Library		
<input checked="" type="checkbox"/> base	The R Base Package	4.3.1
<input type="checkbox"/> boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-28.1
<input type="checkbox"/> class	Functions for Classification	7.3-22
<input type="checkbox"/> cli	Helpers for Developing Command Line Interfaces	3.6.1
<input type="checkbox"/> cluster	"Finding Groups in Data": Cluster Analysis Extended Rousseeuw et al.	2.1.4
<input type="checkbox"/> codetools	Code Analysis Tools for R	0.2-19
<input type="checkbox"/> compiler	The R Compiler Package	4.3.1
<input type="checkbox"/> cpp11	A C++11 Interface for R's C Interface	0.4.5
<input checked="" type="checkbox"/> datasets	The R Datasets Package	4.3.1
<input type="checkbox"/> dplyr	A Grammar of Data Manipulation	1.1.2
<input type="checkbox"/> fansi	ANSI Control Sequence Aware String Functions	1.0.4
<input type="checkbox"/> foreign	Read Data Stored by 'Minitab', 'S', 'SAS', 'SPSS', 'Stata', 'Systat', 'Weka', 'dBase', ...	0.8-84
<input type="checkbox"/> generics	Common S3 Generics not Provided by	0.1.3

The Console pane at the bottom shows the output of the script execution:

```
R 4.3.1 > /cloud/project/
~ DONE (vctr)
* installing *binary* package 'pillar' ...
* DONE (pillar)
* installing *binary* package 'purrr' ...
* DONE (purrr)
* installing *binary* package 'stringr' ...
* DONE (stringr)
```





Local computer

or

```
med_data <- read.csv('https://tinyurl.com/4e97jvuk')
```

The screenshot displays the RStudio environment with the following components:

- Script Editor:** Contains the following R code:

```
1 med_data <- read.csv('https://storage.googleapis.com/kagglestdsdata/datas
2
3 head(med_data)
4 summary(med_data)
5 dim(med_data)
6 install("tidyr")
7 library(tidyr)
8
```
- Environment:** Shows the 'Global Environment' with a variable 'med_data' containing 299 observations of 13 variables.
- Console:** Displays the output of the R script, including the installation of several packages:

```
R 4.3.1 > /cloud/project/
- DONE (vctrs)
* installing *binary* package 'pillar' ...
* DONE (pillar)
* installing *binary* package 'purrr' ...
* DONE (purrr)
* installing *binary* package 'stringr' ...
* DONE (stringr)
```
- Package List:** A table of installed and available packages:

Package	Description	Version
boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-28.1
class	Functions for Classification	7.3-22
cli	Helpers for Developing Command Line Interfaces	3.6.1
cluster	"Finding Groups in Data": Cluster Analysis Extended Rousseeuw et al.	2.1.4
codetools	Code Analysis Tools for R	0.2-19
compiler	The R Compiler Package	4.3.1
cxx11	A C++11 Interface for R's C Interface	0.4.5
datasets	The R Datasets Package	4.3.1
dplyr	A Grammar of Data Manipulation	1.1.2
fansi	ANSI Control Sequence Aware String Functions	1.0.4
foreign	Read Data Stored by 'Minitab', 'S', 'SAS', 'SPSS', 'Stata', 'Systat', 'Weka', 'dBase', ...	0.8-84
generics	Common S3 Generics not Provided by	0.1.3



Using “library/ packages” :

```
install("ggplot2")  
library(ggplot2)
```

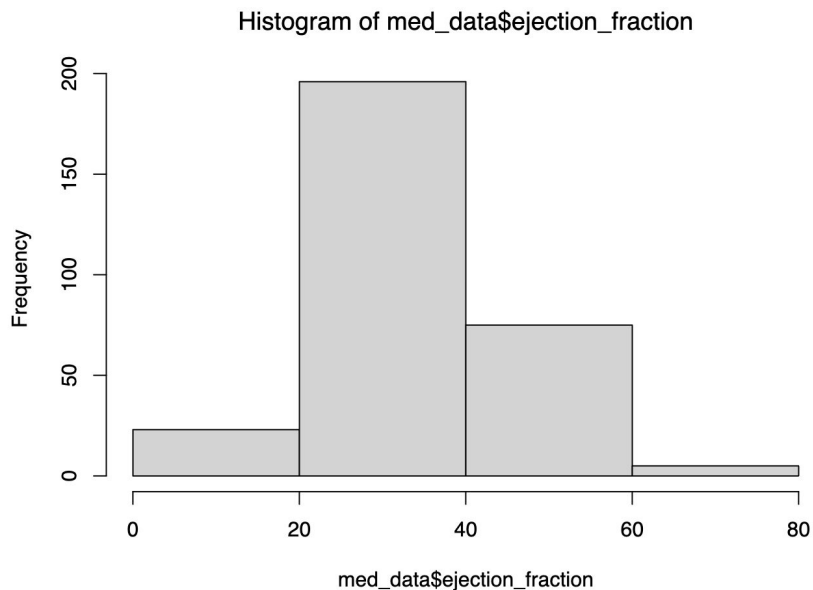
The screenshot displays the RStudio environment with the following components:

- Code Editor:** Contains R code for reading a CSV file from a Google Cloud Storage link, summarizing it, and installing the `tidyr` package.
- Environment:** Shows the `med_data` object with 299 observations and 13 variables.
- Files Plots Packages:** A red box highlights the **Install** button in the Packages pane.
- Console:** Displays the output of the executed code, including the installation of `tidyr` and other dependencies like `pillar`, `purrr`, `stringr`, `tidyselect`, `tibble`, `dplyr`, and `tidyr`.
- Package List:** A table of installed and available packages.

Name	Description	Version
System Library		
<input checked="" type="checkbox"/> base	The R Base Package	4.3.1
<input type="checkbox"/> boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-28.1
<input type="checkbox"/> class	Functions for Classification	7.3-22
<input type="checkbox"/> cli	Helpers for Developing Command Line Interfaces	3.6.1
<input type="checkbox"/> cluster	"Finding Groups in Data": Cluster Analysis Extended Rousseeuw et al.	2.1.4
<input type="checkbox"/> codetools	Code Analysis Tools for R	0.2-19
<input type="checkbox"/> compiler	The R Compiler Package	4.3.1
<input type="checkbox"/> cpp11	A C++11 Interface for R's C Interface	0.4.5
<input checked="" type="checkbox"/> datasets	The R Datasets Package	4.3.1
<input type="checkbox"/> dplyr	A Grammar of Data Manipulation	1.1.2
<input type="checkbox"/> fansi	ANSI Control Sequence Aware String Functions	1.0.4
<input type="checkbox"/> foreign	Read Data Stored by 'Minitab', 'S', 'SAS', 'SPSS', 'Stata', 'Systat', 'Weka', 'dBase', ...	0.8-84
<input type="checkbox"/> generics	Common S3 Generics not Provided by Base R Methods Related to Model Fitting	0.1.3
<input type="checkbox"/> glue	Interpreted String Literals	1.6.2
<input checked="" type="checkbox"/> graphics	The R Graphics Package	4.3.1
<input checked="" type="checkbox"/> grDevices	The R Graphics Devices and Support for Colours and Fonts	4.3.1
<input type="checkbox"/> grid	The Grid Graphics Package	4.3.1
<input type="checkbox"/> KernSmooth	Functions for Kernel Smoothing Supporting Wand & Jones (1995)	2.23-21
<input type="checkbox"/> lattice	Trellis Graphics for R	0.21-8

Base R

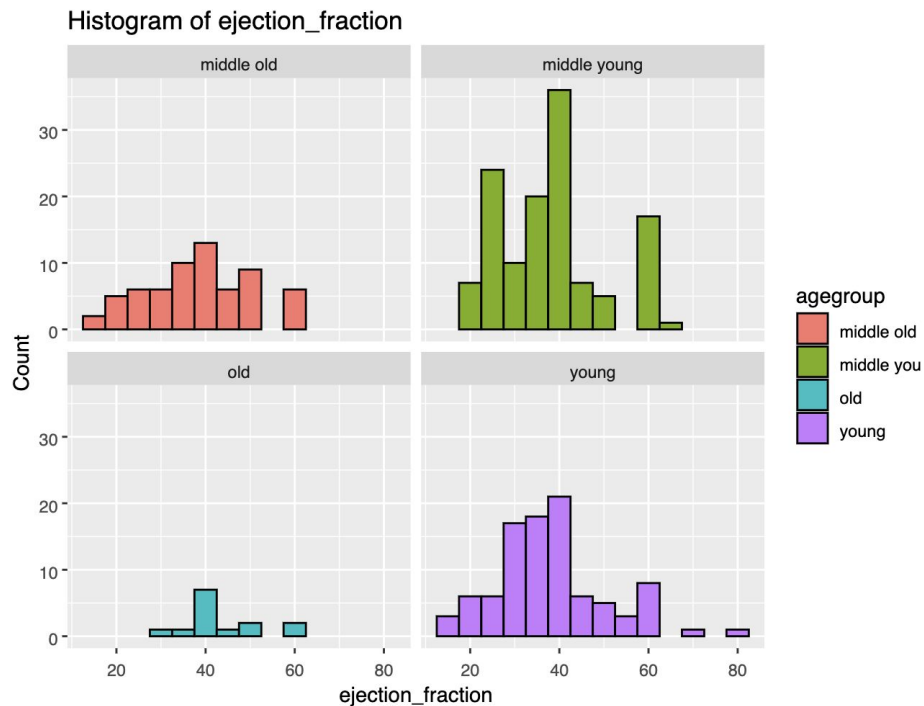
```
hist(med_data$ejestion_fraction, breaks=4)
```



ggplot library



```
ggplot(med_data, aes(x=ejestion_fraction, fill=agegroup)) +  
  geom_histogram(binwidth=5, color="black") +  
  facet_wrap(~agegroup) +  
  labs(x="ejestion_fraction", y="Count", title="Histogram of ejestion_fraction")
```





epilogue :

Machine learning / AI (*i.e.* fancy statistics)

Split data into testing and training (machine learning)

```
set.seed(0)
N <- nrow(med_data)
idx <- sample(1:N, N*2/3)
train <- med_data[idx,]
validation <- med_data[-idx,]
```

Why split?

**What would happen if you tested
on training data?**



epilogue :

Machine learning / AI (*i.e.* fancy statistics)

Next, we'll fit a logistic regression (logit) model using R's `glm()` function,

```
train$dead = train$DEATH_EVENT == "1"
model <- glm(dead ~ creatinine_phosphokinase + ejection_fraction + platelets + serum_creatinine + serum_sodiu
              data = train, family=binomial(link="logit"))
summary(model)
```



epilogue :

Machine learning / AI (*i.e.* fancy statistics)

Next, we'll fit a logistic regression (logit) model using R's `glm()` function,

```
train$dead = train$DEATH_EVENT == "1"
model <- glm(dead ~ creatinine_phosphokinase + ejection_fraction + platelets + serum_creatinine + serum_sodium,
             data = train, family = binomial(link = "logit"))
summary(model)
```

Outcome ~ input 1

+ input 2

+ input 3

+ input 4

+ input 5

TLDR

Regression is a basic machine learning / statistical function for determining the relationship between an input(s) and an output variable

Call:
 glm(formula = dead ~ creatinine_phosphokinase + ejection_fraction
 platelets + serum_creatinine + serum_sodium, family = binomia
 data = train)



Outcome ~ input 1 + input 2 + input 3 + input 4 + input 5

Deviance Residuals:

Min	1Q	Median	3Q	Max
-2.3159	-0.7850	-0.6069	0.9851	2.2909

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	1.005e+01	5.862e+00	1.715	0.086387 .
creatinine_phosphokinase	1.958e-04	1.354e-04	1.446	0.148268
ejection_fraction	-4.668e-02	1.658e-02	-2.815	0.004873 **
platelets	-2.763e-07	1.681e-06	-0.164	0.869491
serum_creatinine	7.033e-01	2.116e-01	3.324	0.000889 ***
serum_sodium	-7.490e-02	4.340e-02	-1.726	0.084383 .

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 243.63 on 198 degrees of freedom
 Residual deviance: 209.60 on 193 degrees of freedom
 AIC: 221.6

Number of Fisher Scoring iterations: 4

Determining “machine intelligence”



What you predict (outputs)

		Predicted condition	
		Positive (PP)	Negative (PN)
Actual condition	Total population = P + N		
	Positive (P)	True positive (TP), hit	False negative (FN), type II error, miss, underestimation
	Negative (N)	False positive (FP), type I error, false alarm, overestimation	True negative (TN), correct rejection

What you see
(inputs)

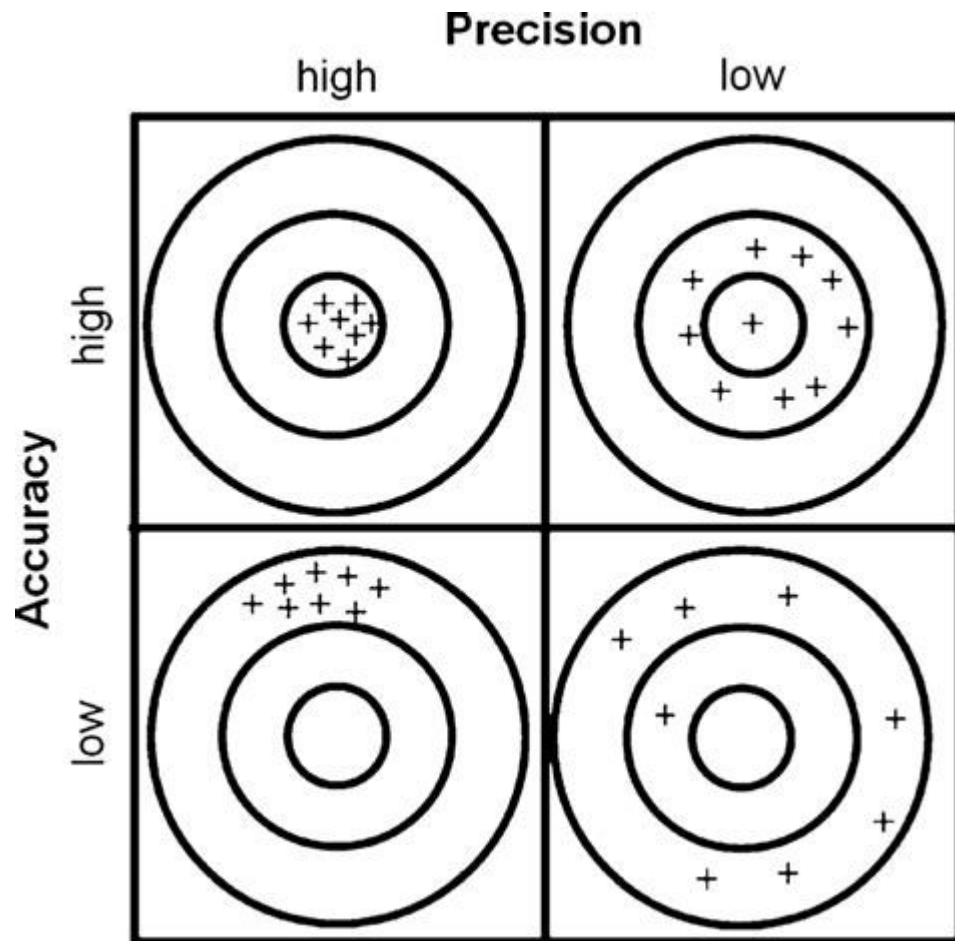
https://en.wikipedia.org/wiki/Confusion_matrix

What do you want to optimize for??

In a perfect world we want both to be high

Balance & Application dependent:

Do not want to scare people by saying they have disease when they do not



Determining “machine intelligence”



What you predict (outputs)

Sources: [22][23][24][25][26][27][28][29][30] [view](#) · [talk](#) · [edit](#)

		Predicted condition			
		Positive (PP)	Negative (PN)	Informedness, bookmaker informedness (BM) $= \text{TPR} + \text{TNR} - 1$	Prevalence threshold (PT) $= \frac{\sqrt{\text{TPR} \times \text{FPR}} - \text{FPR}}{\text{TPR} - \text{FPR}}$
Actual condition	Total population $= P + N$				
	Positive (P)	True positive (TP), hit	False negative (FN), type II error, miss, underestimation	True positive rate (TPR), recall, sensitivity (Sen) probability of detection, hit rate, power $= \frac{\text{TP}}{P} = 1 - \text{FNR}$	False negative rate (FNR), miss rate $= \frac{\text{FN}}{P} = 1 - \text{TPR}$
	Negative (N)	False positive (FP), type I error, false alarm, overestimation	True negative (TN), correct rejection	False positive rate (FPR), probability of false alarm, fall-out $= \frac{\text{FP}}{N} = 1 - \text{TNR}$	True negative rate (TNR), specificity (SPC), selectivity $= \frac{\text{TN}}{N} = 1 - \text{FPR}$
What you see (inputs)	Prevalence $= \frac{P}{P + N}$	Positive predictive value (PPV), precision $= \frac{\text{TP}}{\text{TP} + \text{FP}}$	False omission rate (FOR) $= \frac{\text{FN}}{\text{PN}} = 1 - \text{NPV}$	Positive likelihood ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$	Negative likelihood ratio (LR-) $= \frac{\text{FNR}}{\text{TNR}}$
	Accuracy (ACC) $= \frac{\text{TP} + \text{TN}}{P + N}$	False discovery rate (FDR) $= \frac{\text{FP}}{\text{PP}} = 1 - \text{PPV}$	Negative predictive value (NPV) $= \frac{\text{TN}}{\text{PN}}$ $= 1 - \text{FOR}$	Markedness (MK), deltaP (Δp) $= \text{PPV} + \text{NPV} - 1$	Diagnostic odds ratio (DOR) $= \frac{\text{LR}+}{\text{LR}-}$
	Balanced accuracy (BA) $= \frac{\text{TPR} + \text{TNR}}{2}$	F ₁ score $= \frac{2 \text{PPV} \times \text{TPR}}{\text{PPV} + \text{TPR}} = \frac{2 \text{TN} \times \text{FPR}}{2 \text{TN} + \text{FP} + \text{FN}}$	Fowlkes–Mallows index (FM) $= \sqrt{\text{PPV} \times \text{TPR}}$	Matthews correlation coefficient (MCC) $= \frac{\sqrt{\text{TPR} \times \text{TNR} \times \text{PPV} \times \text{NPV}}}{-\sqrt{\text{FNR} \times \text{FPR} \times \text{FOR} \times \text{FDR}}}$	Threat score (TS), critical success index (CSI), Jaccard index $= \frac{\text{TP}}{\text{TP} + \text{FN} + \text{FP}}$

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

Next, we'll fit a logistic regression (logit) model

```
train$dead = train$DEATH_EVENT == "1"
model <- glm(dead ~ creatinine_phosphokinase + ejection_fraction +
              platelets + serum_creatinine + serum_sodium, family = binomial,
              data = train)
summary(model)
```

```
prec <- precision(table_mat)
prec
rec <- recall(table_mat)
rec
```

```
[1] 0.6944444
[1] 0.2873563
```

```
f1 <- 2 * ((prec * rec) / (prec + rec))
f1
```

```
[1] 0.4065041
```

Call:

```
glm(formula = dead ~ creatinine_phosphokinase + ejection_fraction +
     platelets + serum_creatinine + serum_sodium, family = binomial,
     data = train)
```

Deviance Residuals:

Min	1Q	Median	3Q	Max
-2.3159	-0.7850	-0.6069	0.9851	2.2909

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	1.005e+01	5.862e+00	1.715	0.086387 .
creatinine_phosphokinase	1.958e-04	1.354e-04	1.446	0.148268
ejection_fraction	-4.668e-02	1.658e-02	-2.815	0.004873 **
platelets	-2.763e-07	1.681e-06	-0.164	0.869491
serum_creatinine	7.033e-01	2.116e-01	3.324	0.000889 ***
serum_sodium	-7.490e-02	4.340e-02	-1.726	0.084383 .

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 243.63 on 198 degrees of freedom
Residual deviance: 209.60 on 193 degrees of freedom
AIC: 221.6

Number of Fisher Scoring iterations: 4



sodium

