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… & <u>be able to explain</u>!

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

BMC Medical Informatics and Decision Making

Home About Articles Submission Guidelines Collections Join The Board Submit manuscript

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

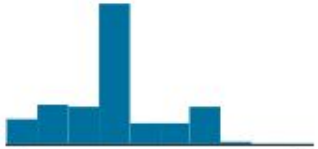
BMC Medical Informatics and Decision Making **20**, Article number: 16 (2020) | Cite this article

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.

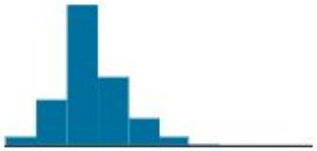
# smoking	# time	# DEATH_EVENT
If the patient smokes or not (boolean)	Follow-up period (days)	If the patient deceased during the follow-up period (boolean)

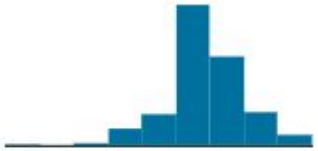
1. Our data: Cardiovascular diseases

# age	# anaemia	# creatinine_phosp...	# diabetes	# ejection_fraction
Age	Decrease of red blood cells or hemoglobin (boolean)	Level of the CPK enzyme in the blood (mcg/L)	If the patient has diabetes (boolean)	Percentage of blood leaving the heart at each contraction (percentage)
				
40 95	0 1	23 7861	0 1	14 80

1. Our data: Cardiovascular diseases

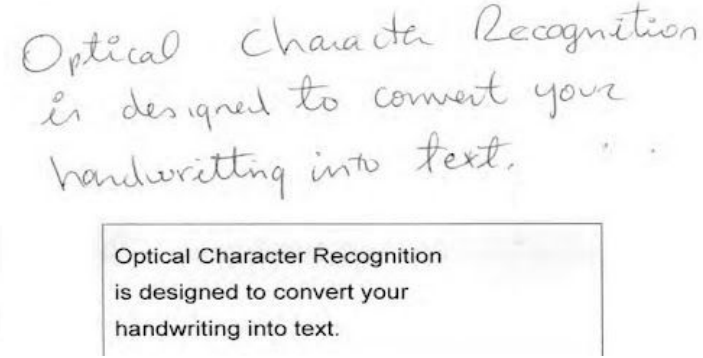
# high_blood_press...	# platelets	# serum_creatinine	# serum_sodium	# sex
If the patient has hypertension (boolean)	Platelets in the blood (kiloplatelets/mL)	Level of serum creatinine in the blood (mg/dL)	Level of serum sodium in the blood (mEq/L)	Woman or man (binary)





0. How do we get …

1. Our data: Cardiovascular diseases



Biologica types Information:

- 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

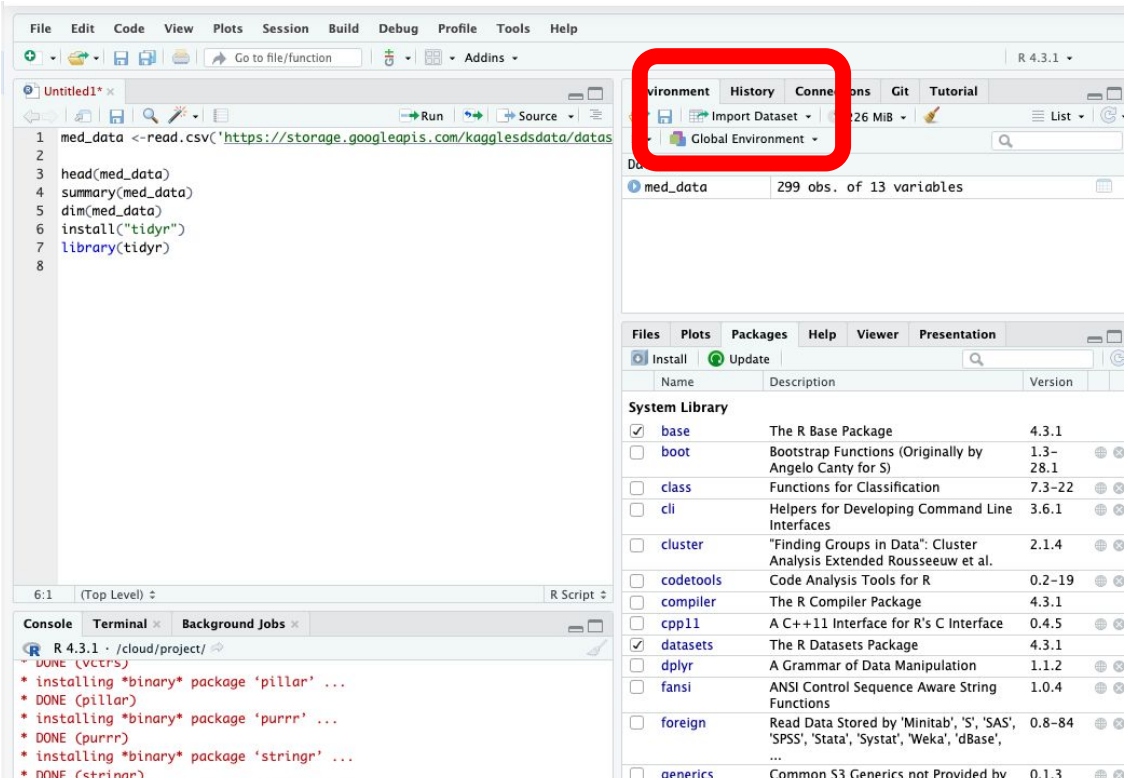
1. Our data: Cardiovascular diseases
2. How do we store our data
3. How do we test and visualize our data…

Rstudio

Local computer

or

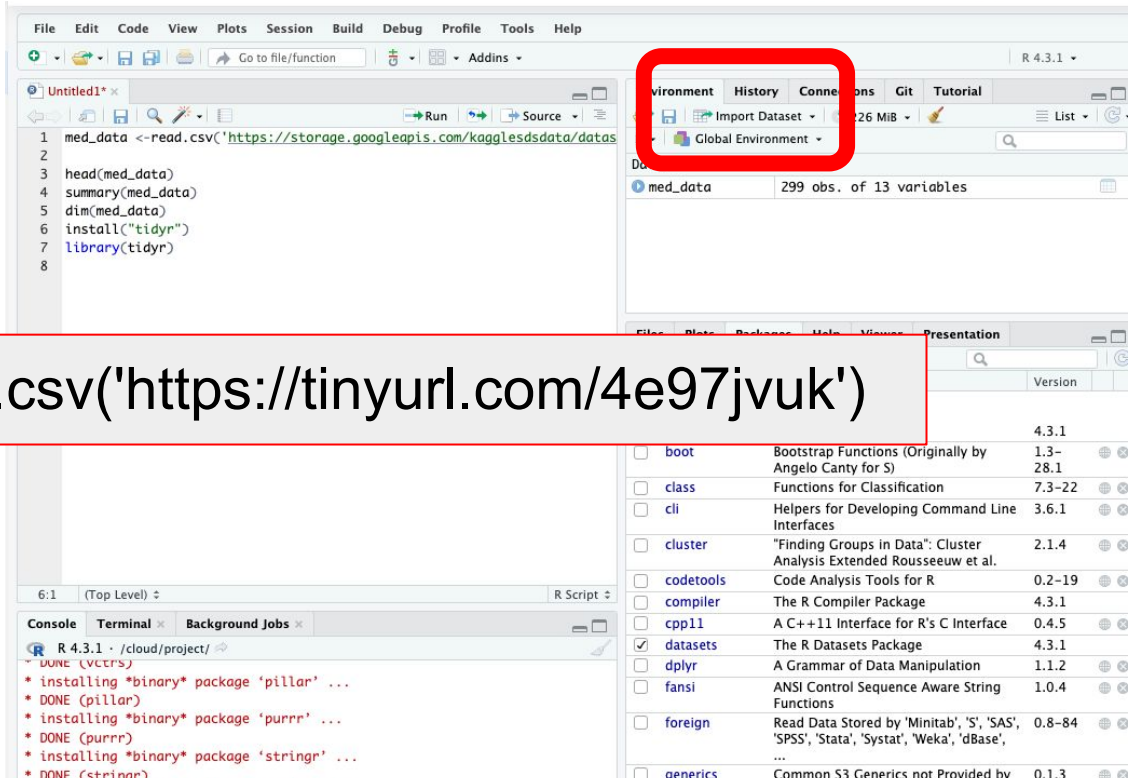
posit.cloud



Local computer

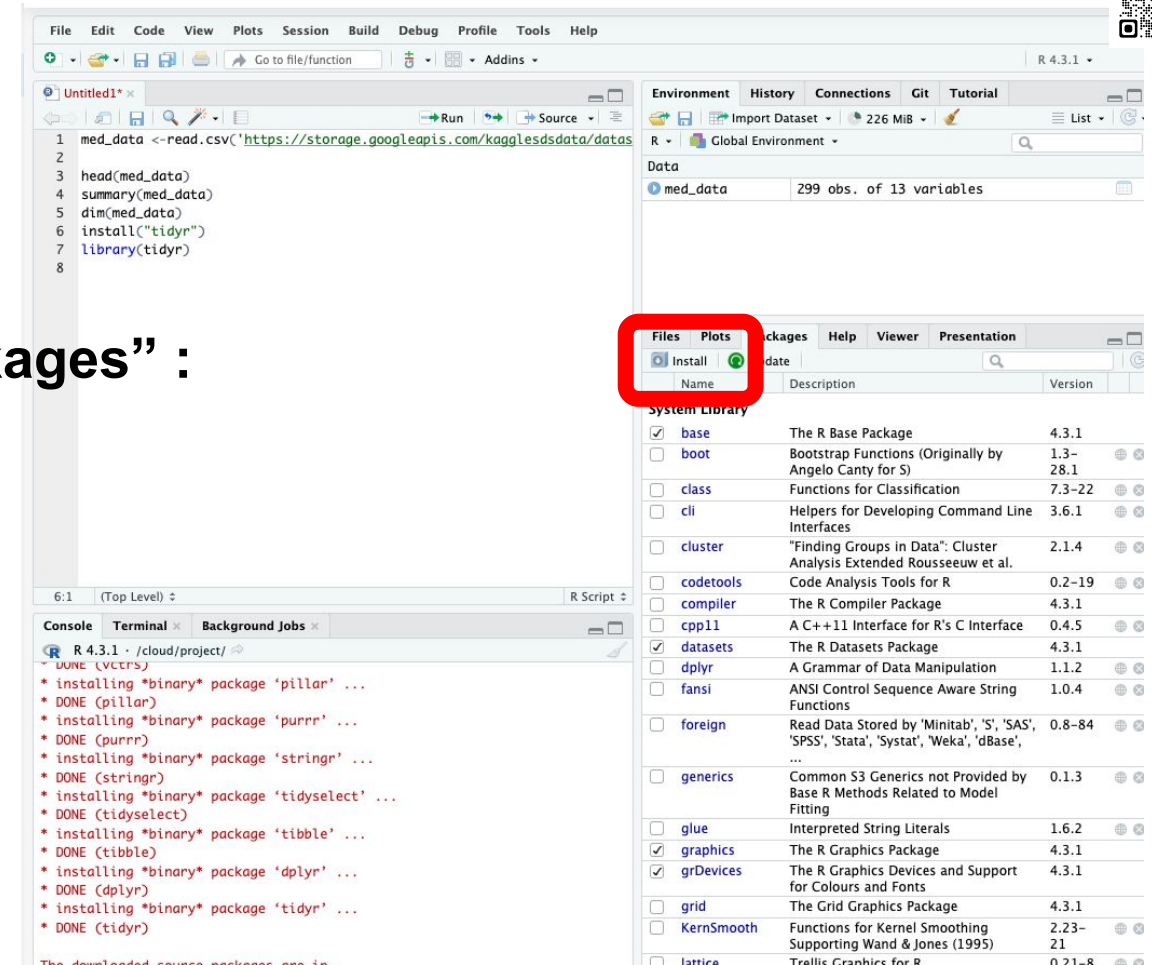
or

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



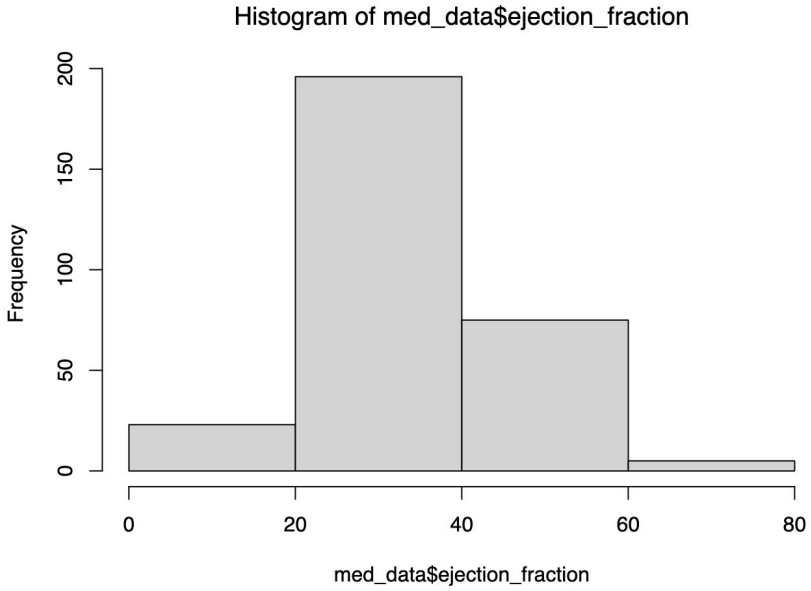
Using “library/ packages” :

install("ggplot2")
library(ggplot2)



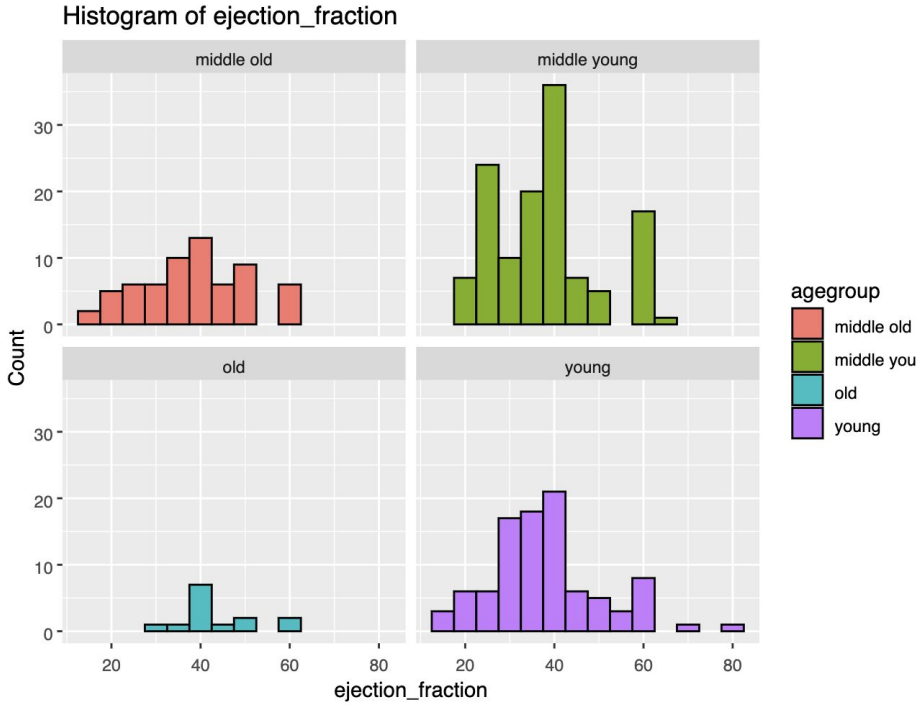
Base R

```r
hist(med_data$ejection_fraction,breaks=4)
```



ggplot library

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

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

Determining “machine intelligence”

What you predict (outputs)

What you see (inputs)

		Predicted condition	
	Total population = P + N	**Positive (PP)**	**Negative (PN)**
Actual condition	**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

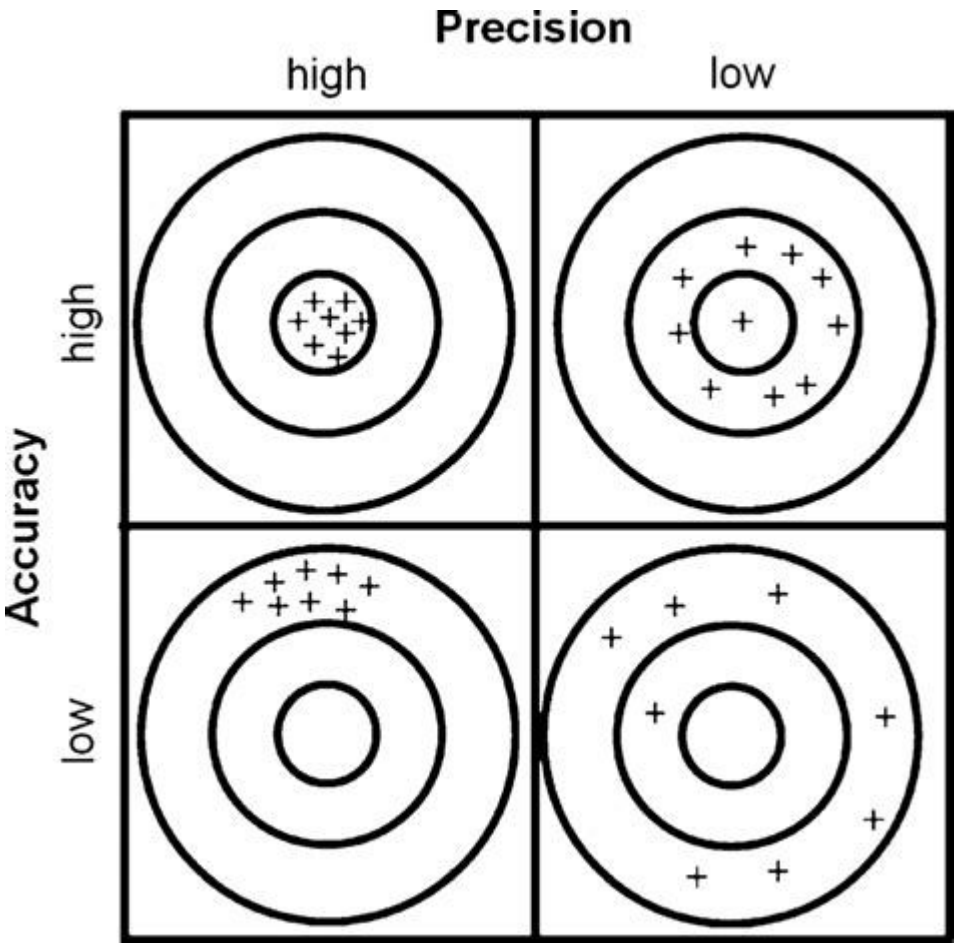
https://en.wikipedia.org/wiki/Confusion_matrix

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)

What you see (inputs)

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

		Predicted condition			
	Total population $= P + N$	**Positive (PP)**	**Negative (PN)**	Informedness, bookmaker informedness (BM) $= TPR + TNR - 1$	Prevalence threshold (PT) $= \frac{\sqrt{TPR \times FPR} - FPR}{TPR - FPR}$
Actual condition	**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 $= \frac{TP}{P} = 1 - FNR$	False negative rate (FNR), miss rate $= \frac{FN}{P} = 1 - TPR$
	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 $= \frac{FP}{N} = 1 - TNR$	True negative rate (TNR), specificity (SPC), selectivity $= \frac{TN}{N} = 1 - FPR$
	Prevalence $= \frac{P}{P + N}$	Positive predictive value (PPV), precision $= \frac{TP}{PP}$	False omission rate (FOR) $= \frac{FN}{PN} = 1 - NPV$	Positive likelihood ratio (LR+) $= \frac{TPR}{FPR}$	Negative likelihood ratio (LR−) $= \frac{FNR}{TNR}$
	Accuracy (ACC) $= \frac{TP + TN}{P + N}$	False discovery rate (FDR) $= \frac{FP}{PP} = 1 - PPV$	Negative predictive value (NPV) $= \frac{TN}{PN} = 1 - FOR$	Markedness (MK), deltaP (Δp) $= PPV + NPV - 1$	Diagnostic odds ratio (DOR) $= \frac{LR+}{LR-}$
	Balanced accuracy (BA) $= \frac{TPR + TNR}{2}$	F_1 score $= \frac{2PPV \times TPR}{PPV + TPR} = \frac{2TP}{2TP + FP + FN}$	Fowlkes–Mallows index (FM) $= \sqrt{PPV \times TPR}$	Matthews correlation coefficient (MCC) $= \sqrt{TPR \times TNR \times PPV \times NPV} - \sqrt{FNR \times FPR \times FOR \times FDR}$	Threat score (TS), critical success index (CSI), Jaccard index $= \frac{TP}{TP + FN + FP}$

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

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

```
train$dead = train$DEATH_EVENT == "1
model <- glm(dead ~ creatinine_phosp
             data = train, family=bi
summary(model)
```

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

sodiu

```
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
```