Urban Informatics

Fall 2018

dr. federica bianco fbianco@nyu.edu

@fedhere

XII: decision trees

Recap:

- Good practices with data: falsifiability, reproducibility
- Basic data retrieving and munging: APIs, Data formats
- SQL
- Basic statistics: distributions and their moments
- Hypothesis testing: p-value, statistical significance
- Statistical and Systematic errors
- Visualizations
- Geospatial analysis
- OLS
- Goodness of fit tests
- Likelihood

Today:

- decision and regression trees (CART)

machine learning

models with parameters that are “learned” from the data

machine learning

models with parameters that are "learned" from the data

parameters that are optimized based on the data

machine learning

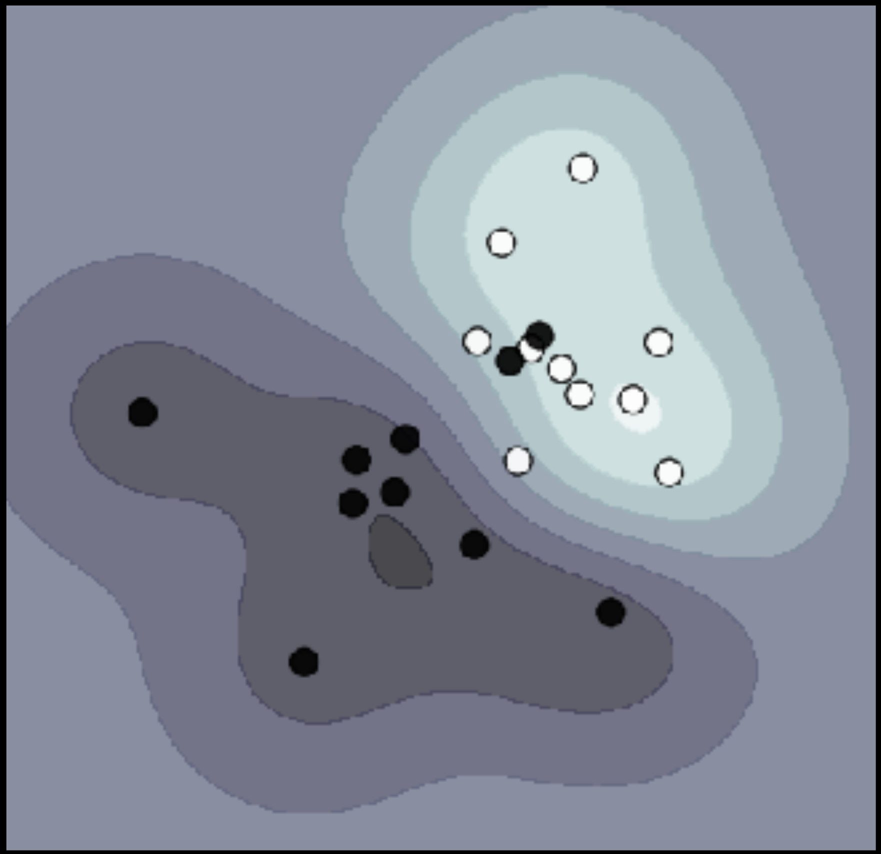
algorithms that can learn from and make predictions on data.

machine learning

algorithms that can learn from and make predictions on data.

supervised learning

extract features and create models that allow prediction where the correct answer is known for a subset of the data



machine learning

algorithms that can learn from and make predictions on data.

supervised learning

extract features and create models that allow prediction where the correct answer is known for a subset of the data

unsupervised learning

identify features and create models that allow to understand structure in the data

machine learning

algorithms that can learn from and make predictions on data.

supervised learning

classification

prediction

unsupervised learning

understanding structure

organizing + compressing data

(classification, feature learing)

machine learning

algorithms that can learn from and make predictions on data.

supervised learning

classification

prediction

LR, SVM
CART
DL

unsupervised learning

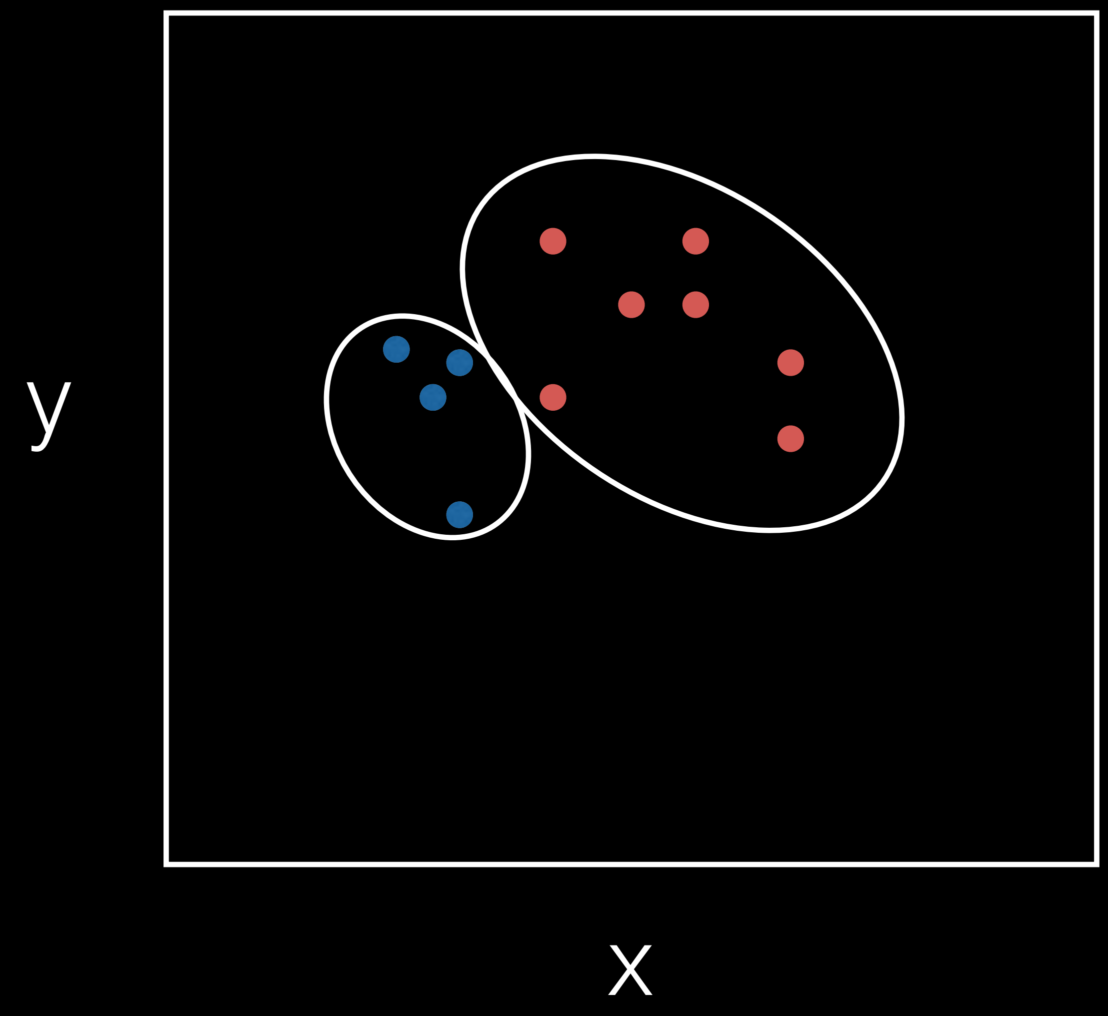
understanding structure

organizing + compressing data

(classification, feature learing)

CLUSTERING

Supervised Learning

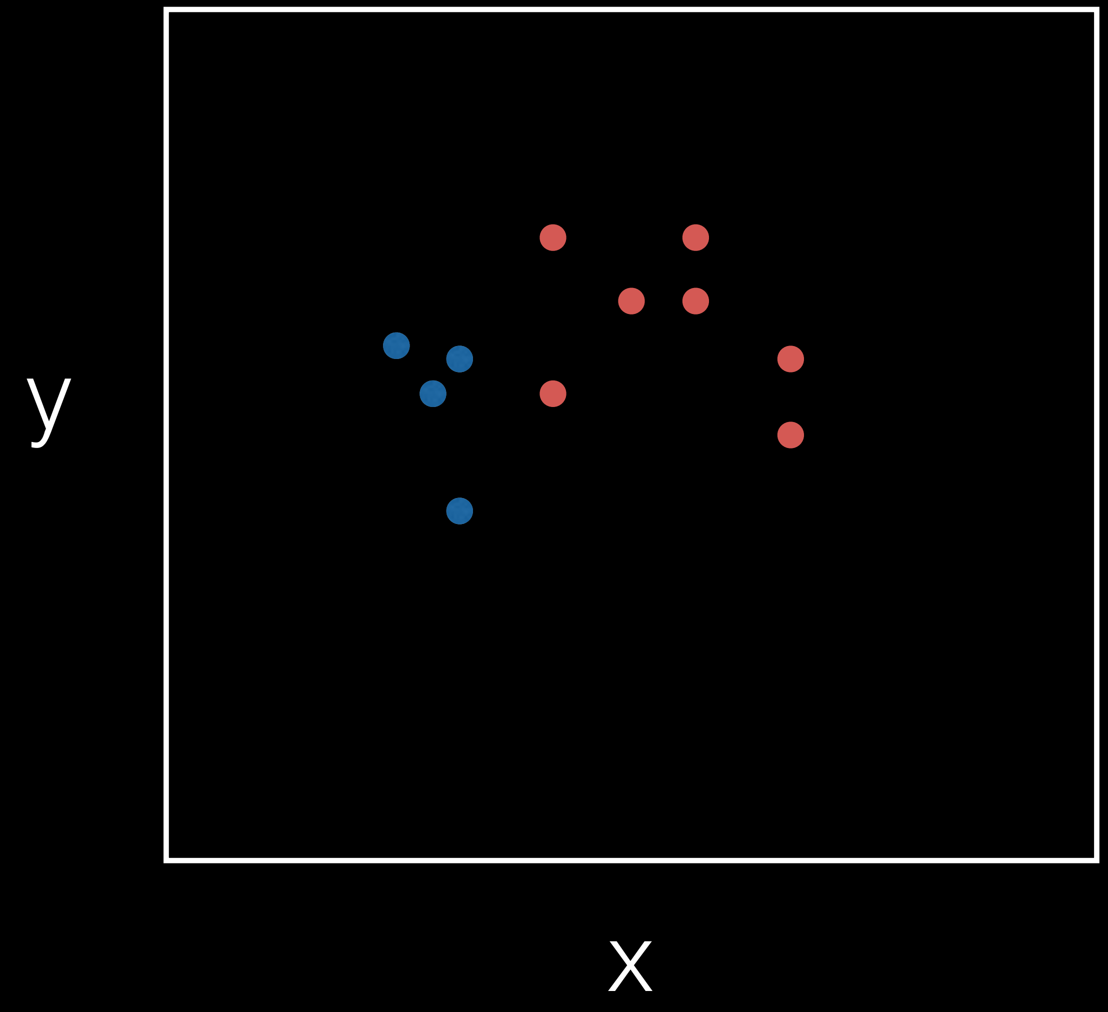


observed:
(x, y, color)

Partitioning methods: classifying (SVM, CART)

goal is to partition the space of observed variables
to separate the space of unobserved (target variables)

observed:
(x, y)

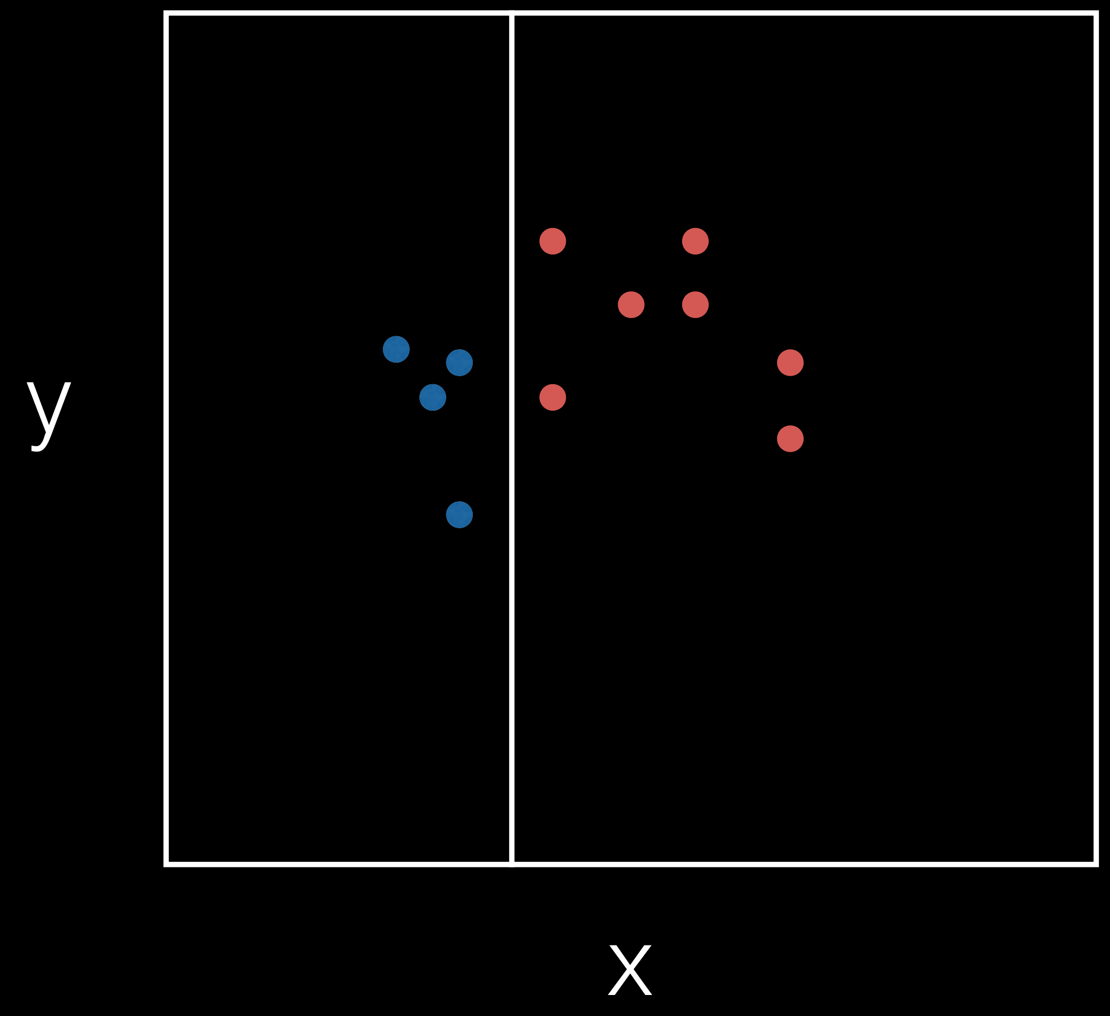


target:
(color)

Partitioning methods: classifying

goal is to partition the space of observed variables
to separate the space of unobserved (target variables)

observed:
(x, y)

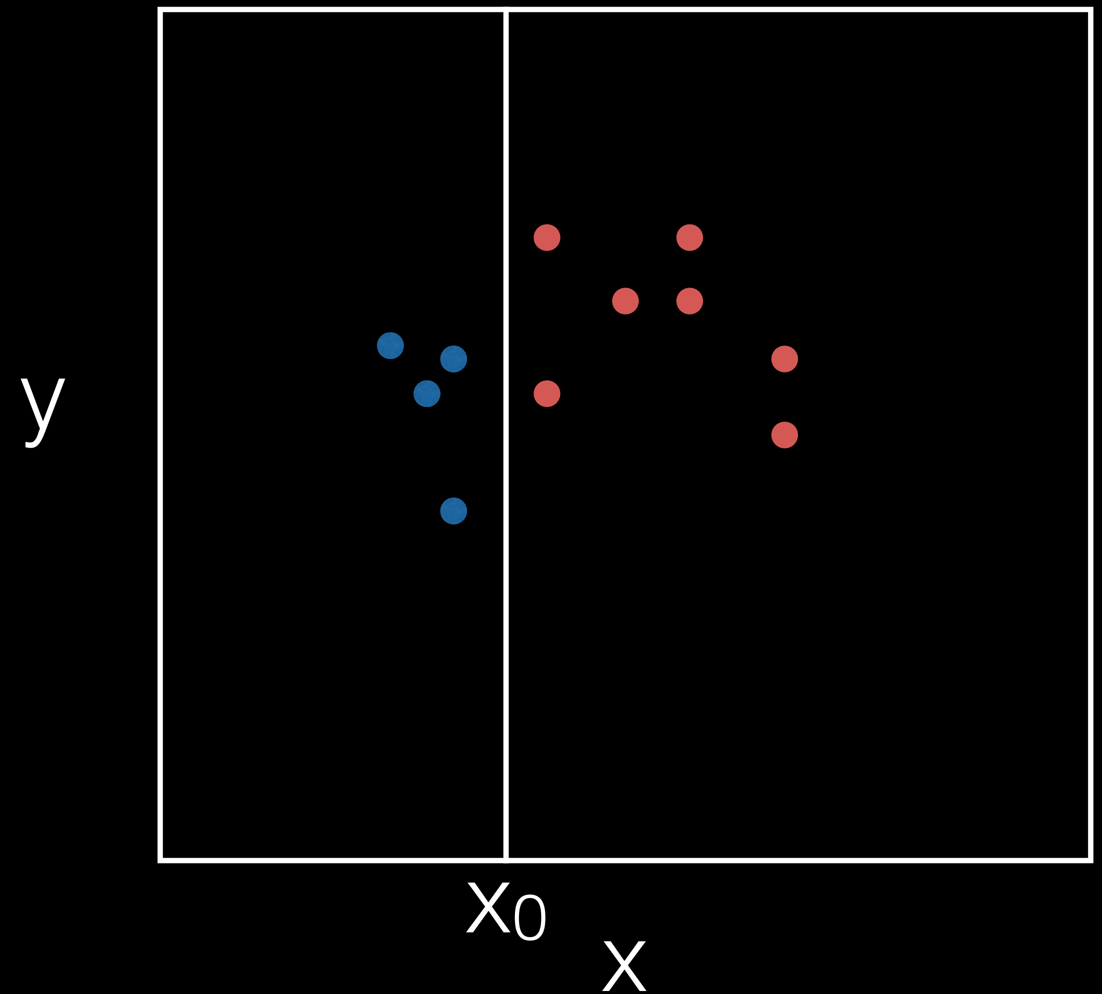


target:
(color)

Partitioning methods: classifying

goal is to partition the space of observed variables
to separate the space of unobserved (target variables)

observed:
(x, y)



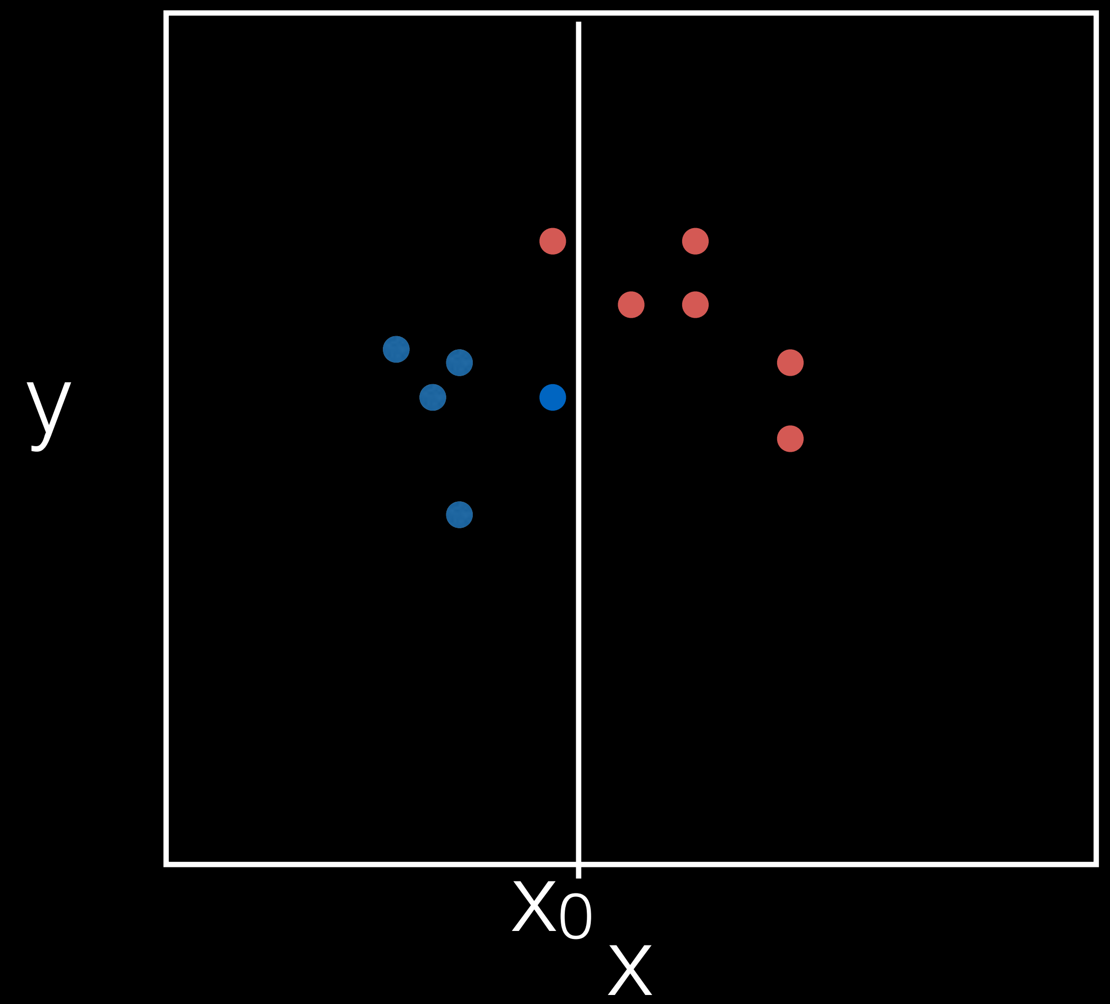
target:
(color)

if $x>x_0$ => ball is red

Partitioning methods: classifying

goal is to partition the space of observed variables to separate the space of unobserved (target variables)

observed:
(x, y)

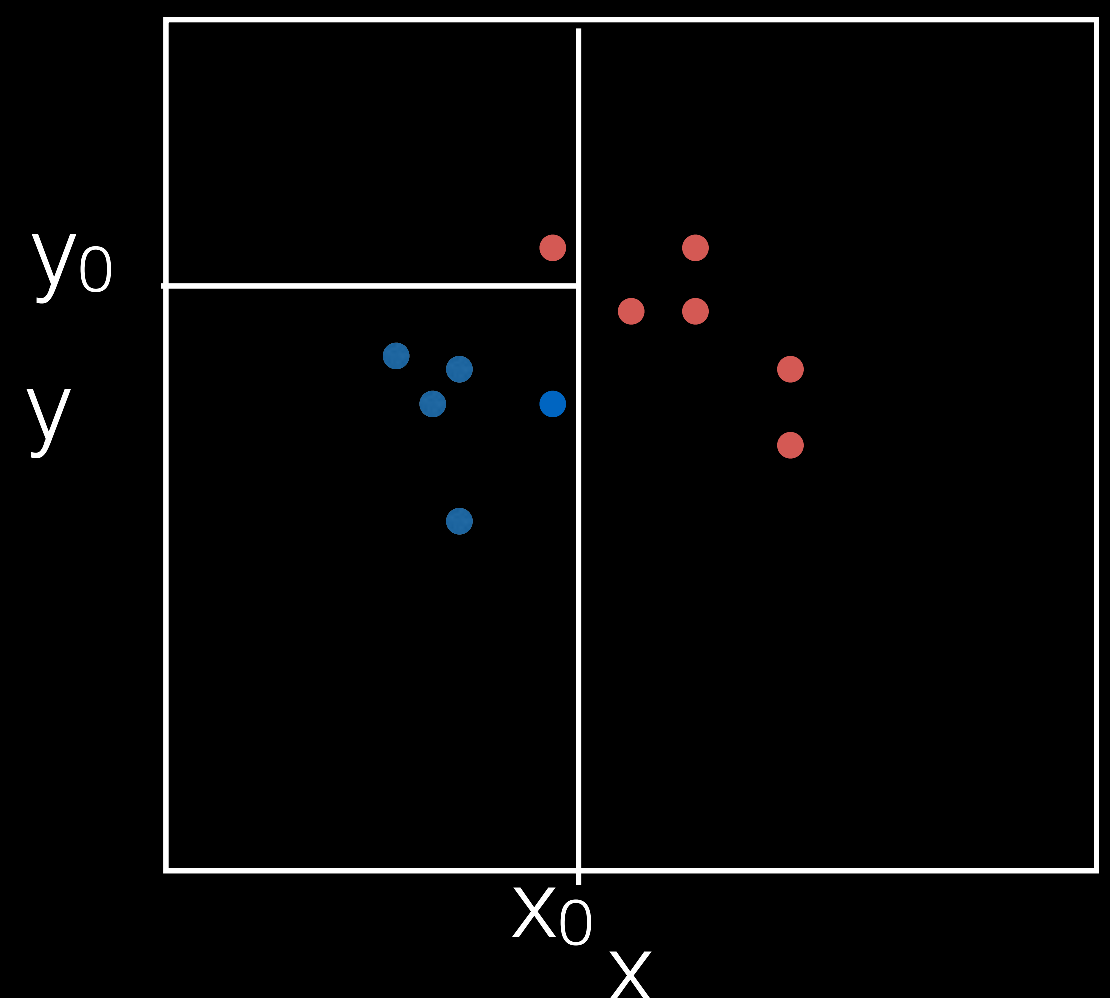


target:
(color)

Partitioning methods: classifying

goal is to partition the space of observed variables
to separate the space of unobserved (target variables)

observed:
(x, y)



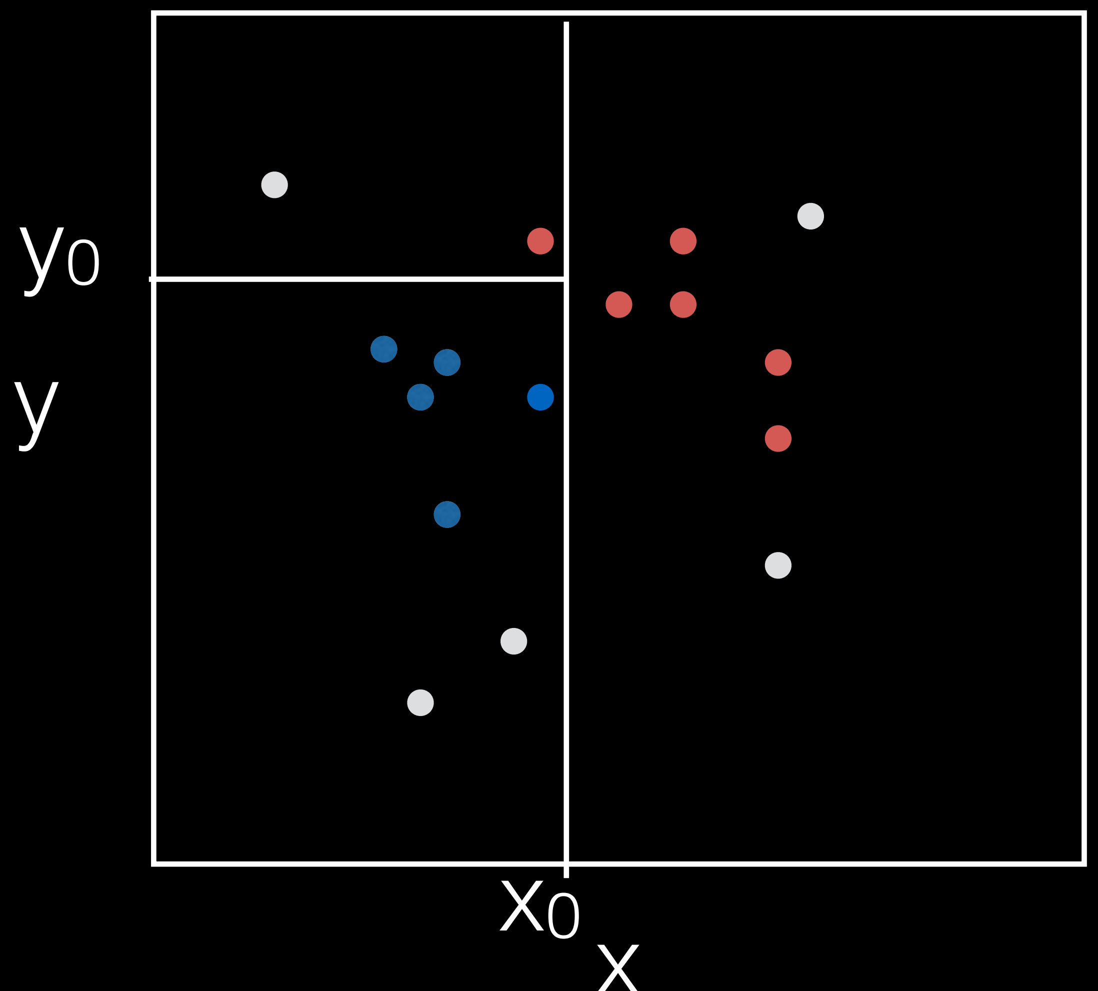
target:
(color)

if $x > x_0$ or $y > y_0$ => ball is red

Partitioning methods: classifying

goal is to partition the space of observed variables
to separate the space of unobserved (target variables)

observed:
(x, y)



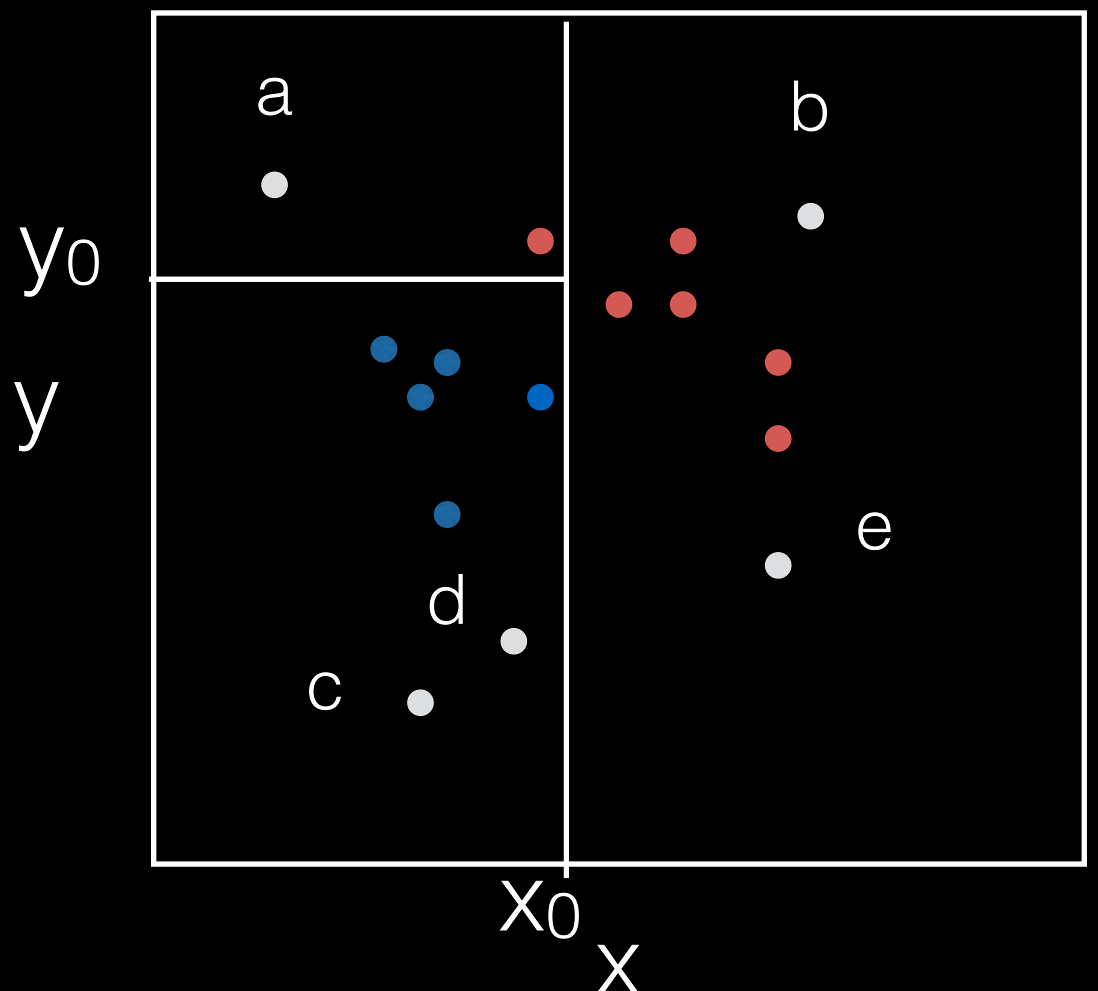
target:
(color)

if $x>x_0$ or $y>y_0$ => ball is red

Partitioning methods: classifying

goal is to partition the space of observed variables to separate the space of unobserved (target variables)

observed:
(x, y)



a: red
b: red
c: blue
d: blue
e: red

target:
(color)

if $x>x_0$ or $y>y_0$ => ball is red

Decision Trees

The good

- Non-Parametric
- White-box: can be easily interpreted
- Works with any feature type and mixed feature types
- Works with missing data
- Robust to outliers

The bad

- High variability (-> use *ensamble* methods)
- Tendency to overfit
- (not as easily interpretable after all…)

a single tree

714 passengers
Ns=424 Nd=290

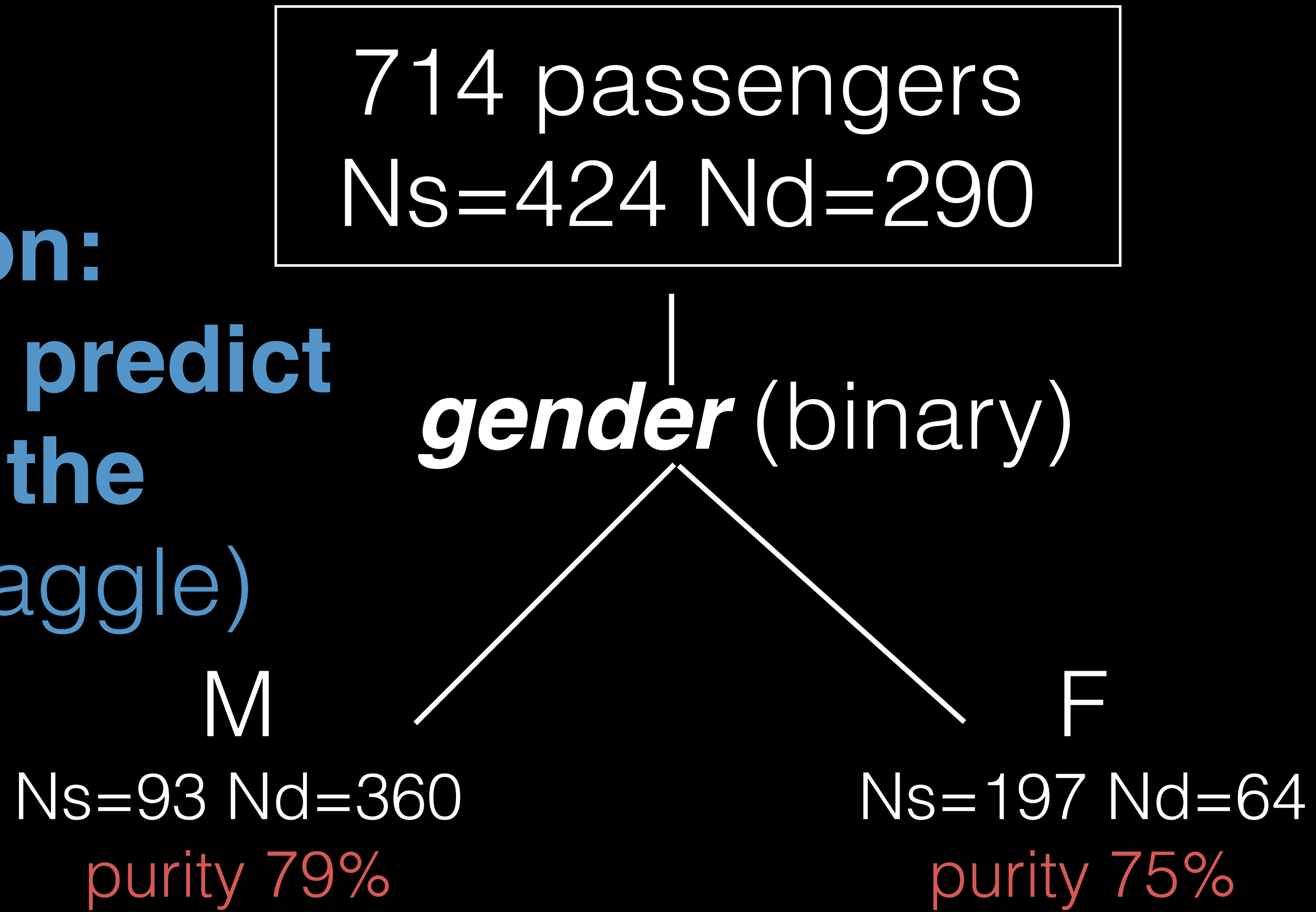
**Application:
a robot to predict
surviving the
Titanic** (Kaggle)

features:
gender
ticket class
age

target variable:
survival (y/n)

Ns: survived
Nd: died

Application: a robot to predict surviving the Titanic (Kaggle)



features:
gender 79/75%
ticket class
age

target variable:
survival (y/n)

Application: a robot to predict surviving the Titanic (Kaggle)

714 passengers
Ns=424 Nd=290

class (categorical)

1st
Ns=471 Nd=242
purity 66%

2nd,3rd
Ns=335 Nd=378
purity 44%

features:
gender 79/75%
ticket class 66/44%
age

target variable:
survival (y/n)

Application:
a robot to predict surviving the Titanic (Kaggle)

714 passengers
Ns=424 Nd=290

age (continuous)

<6.5
Ns=500 Nd=214
purity 30%

>6.5
Ns=278 Nd=435
purity 70%

features:
gender 79/75%
ticket class 66/44%
age 30/70%

target variable:
survival (y/n)

Application: a robot to predict surviving the Titanic (Kaggle)

714 passengers
Ns=424 Nd=290

age (continuous)

M
Ns=93 Nd=360
purity 79%

F
Ns=197 Nd=64
purity 75%

features:
gender 79/75%
age 66/44%
ticket class 30/70%

target variable:
survival (y/n)

Application: a robot to predict surviving the Titanic (Kaggle)

714 passengers
Ns=424 Nd=290

gender (binary)

M
Ns=93 Nd=360
purity 79%

F
Ns=197 Nd=64
purity 75%

features:
gender 79/75%

target variable:
survival (y/n)

Application: a robot to predict surviving the Titanic (Kaggle)

714 passengers
Ns=424 Nd=290

gender (binary)

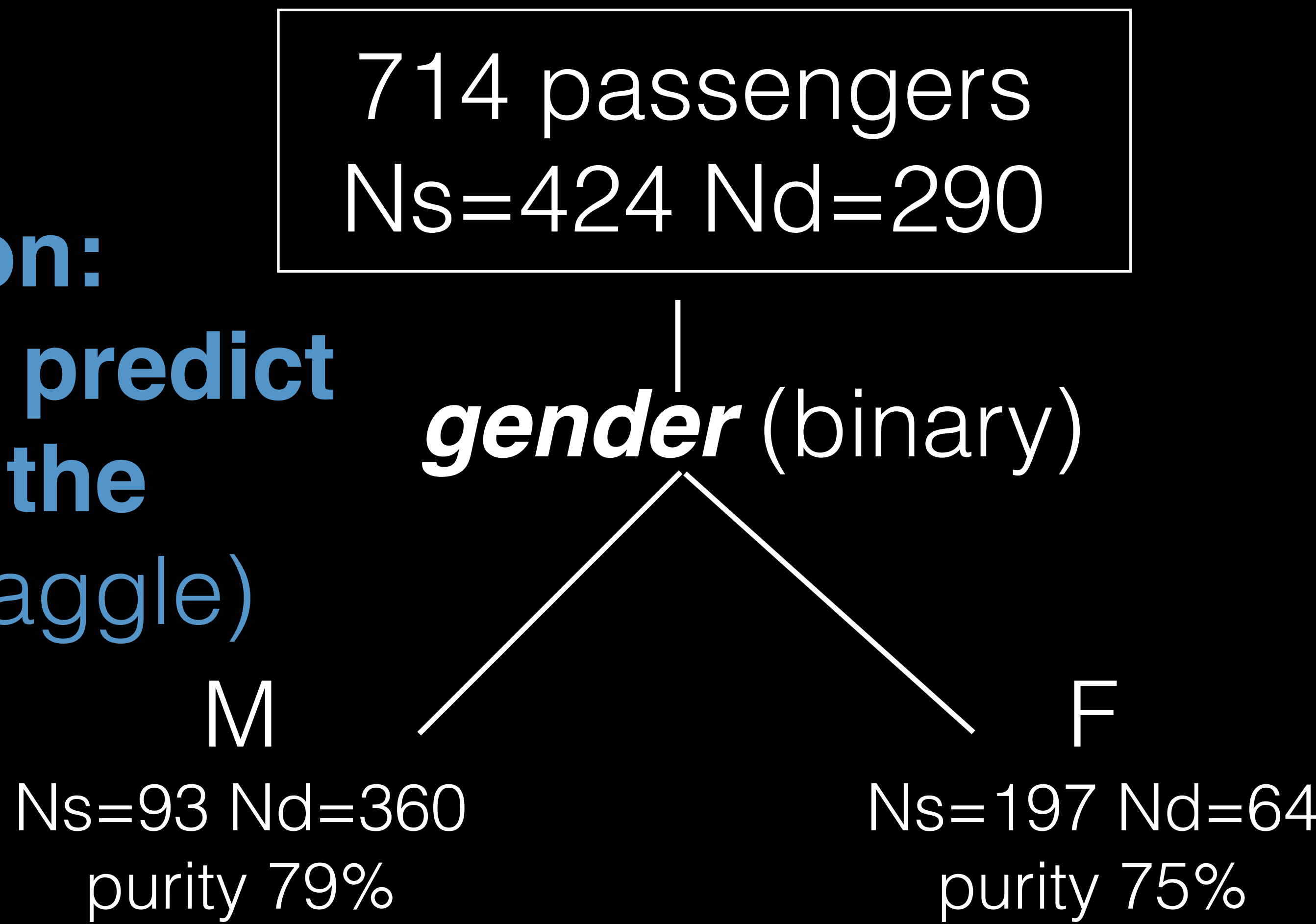
M
Ns=93 Nd=360
purity 79%

F
Ns=197 Nd=64
purity 75%

features:
gender 79/75%
age: M 74/67% F 96/40%
ticket class: M 40/15% F 96/65%

target variable:
survival (y/n)

Application: a robot to predict surviving the Titanic (Kaggle)



features:

gender 79/75%

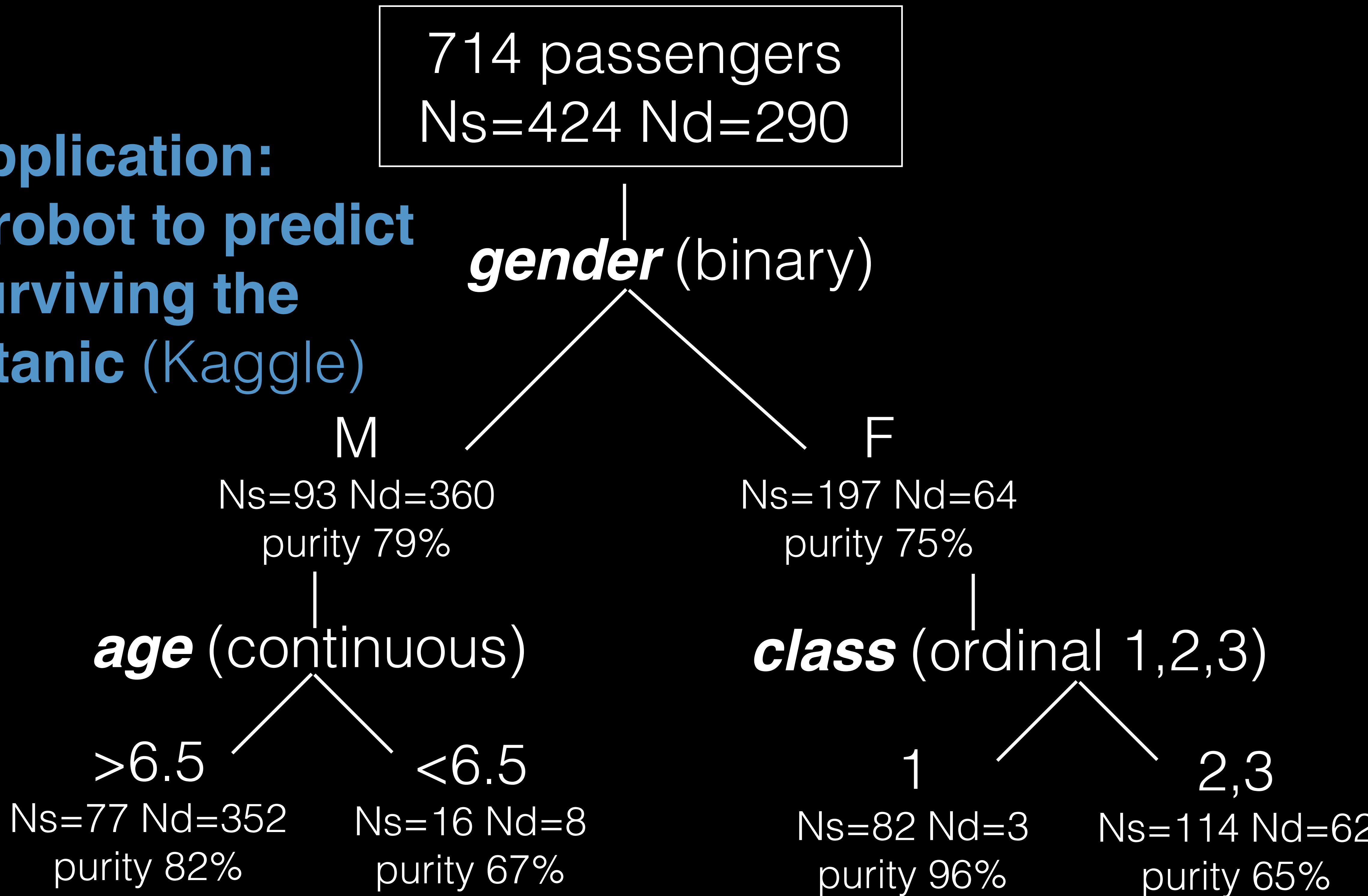
age: M 67/82% F 74/76%

ticket class: M 40/15% F 96/65%

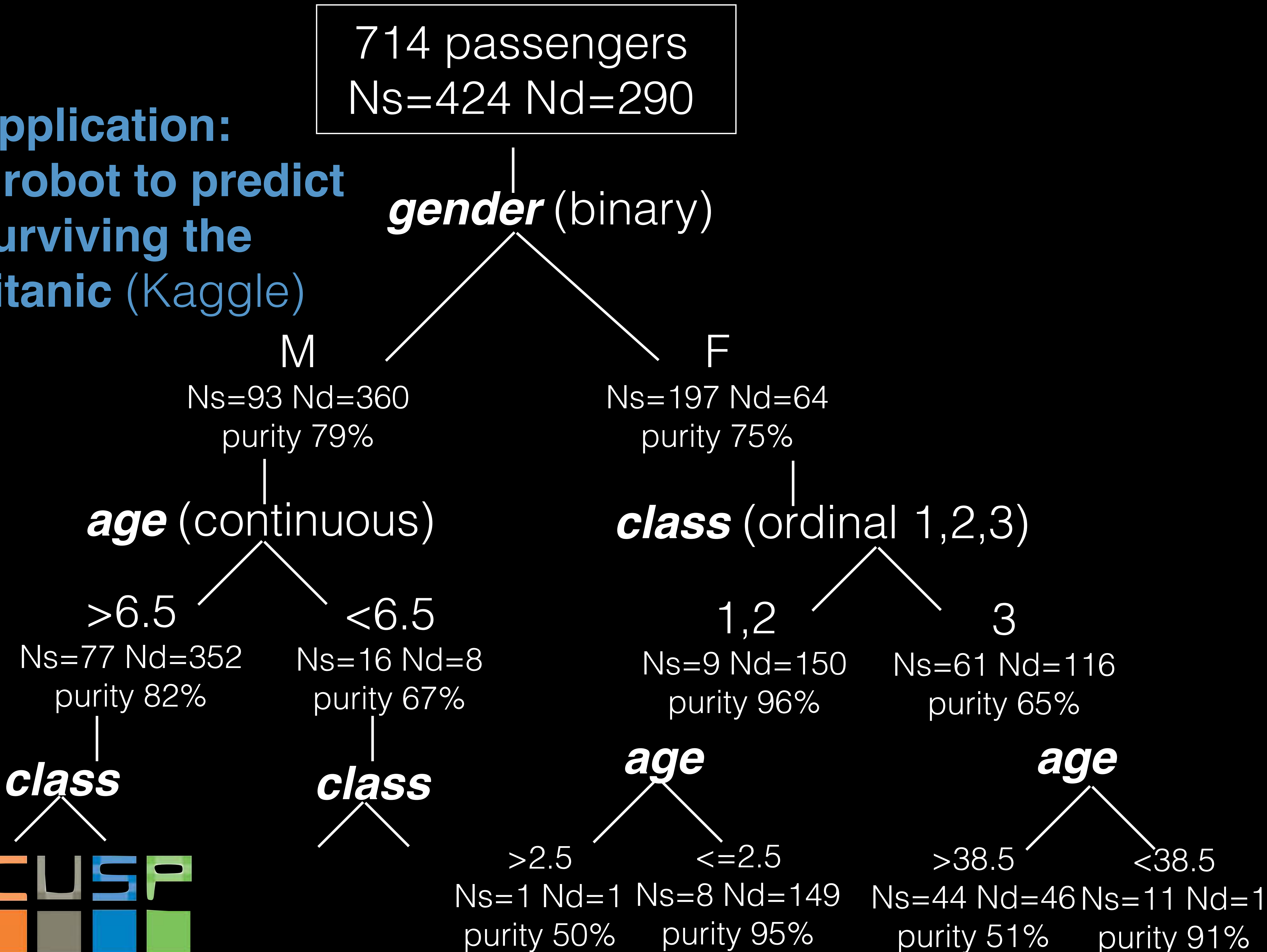
target variable:

survival (y/n)

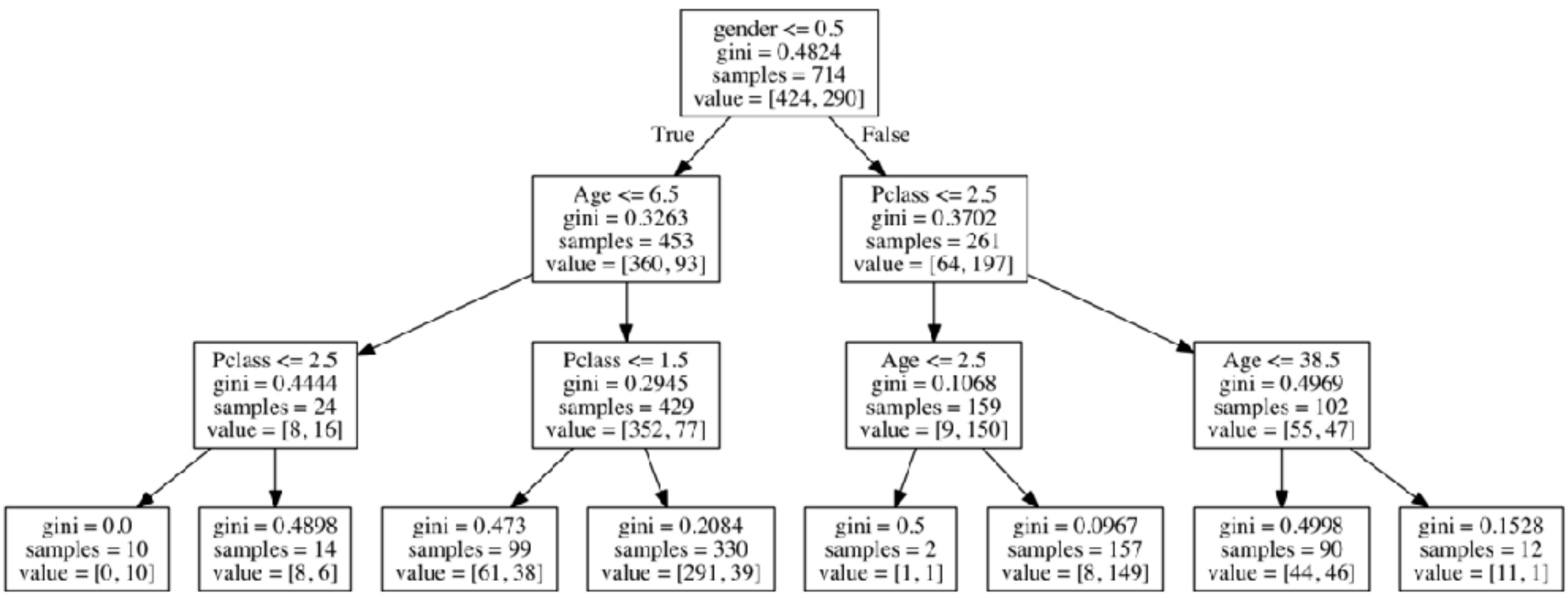
Application: a robot to predict surviving the Titanic (Kaggle)



Application: a robot to predict surviving the Titanic (Kaggle)

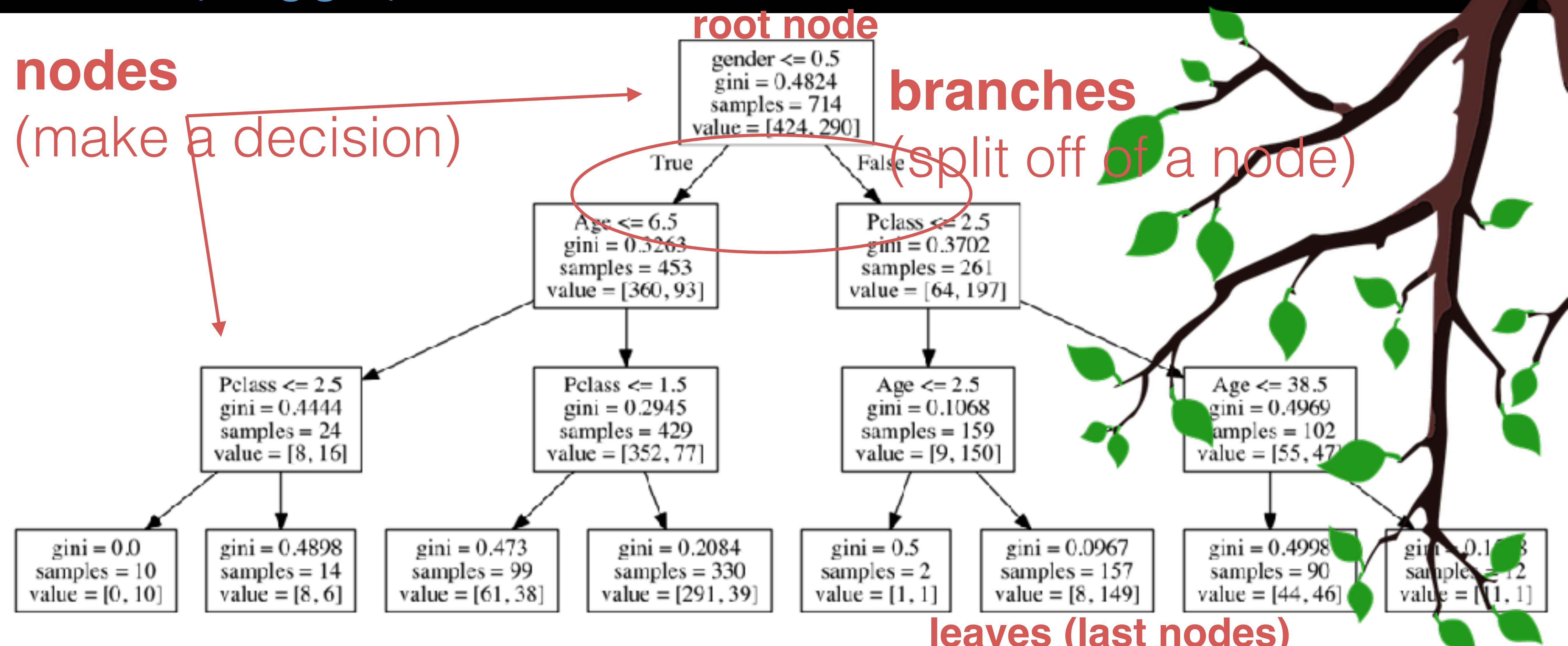


Application: a robot to predict surviving the Titanic (Kaggle)



https://github.com/fedhere/PUI2017_fb55/blob/master/Lab12_fb55/TitanicByCART.ipynb

Application:
a robot to predict surviving the Titanic (Kaggle)



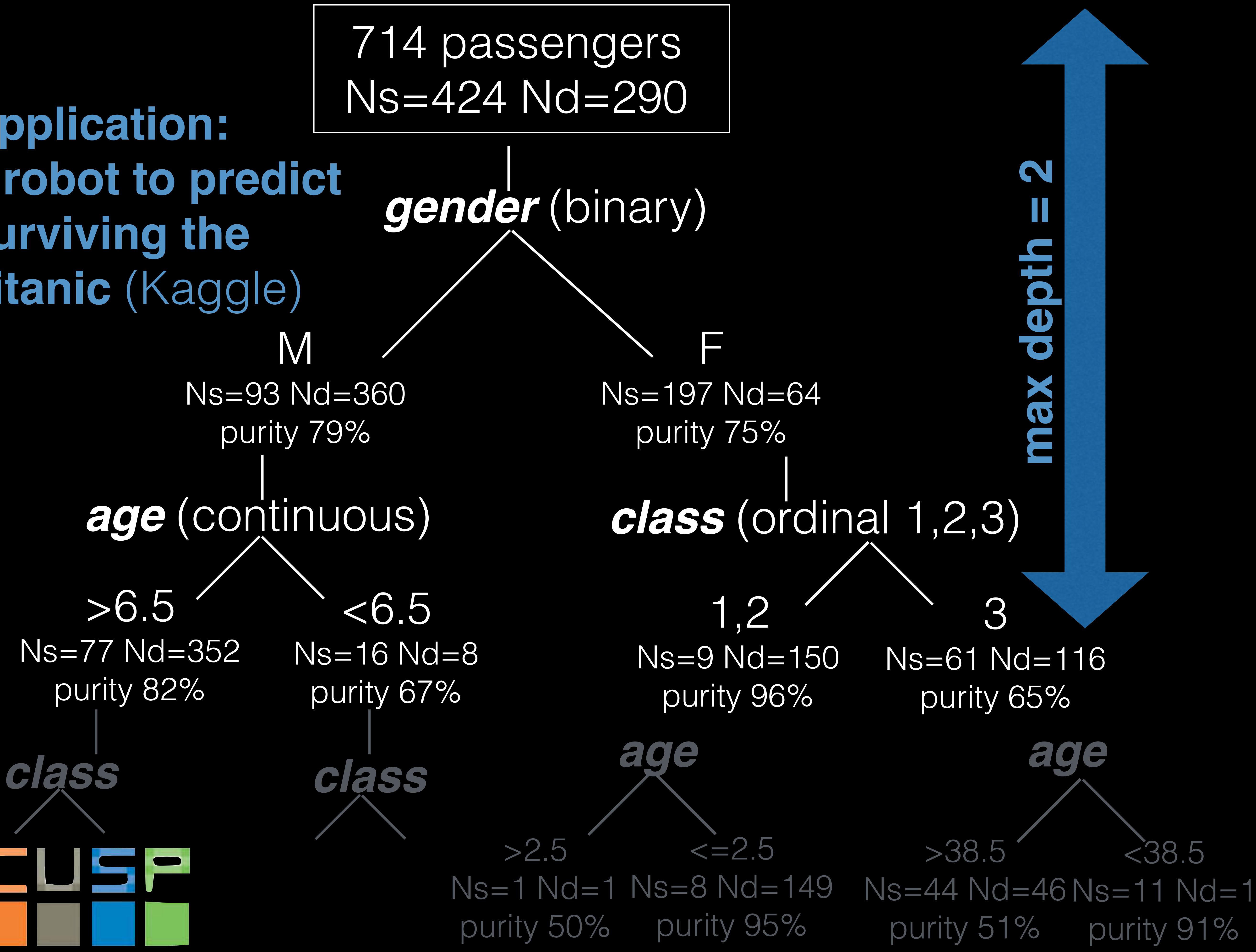
https://github.com/fedhere/PUI2017_fb55/blob/master/Lab12_fb55/TitanicByCART.ipynb a single tree

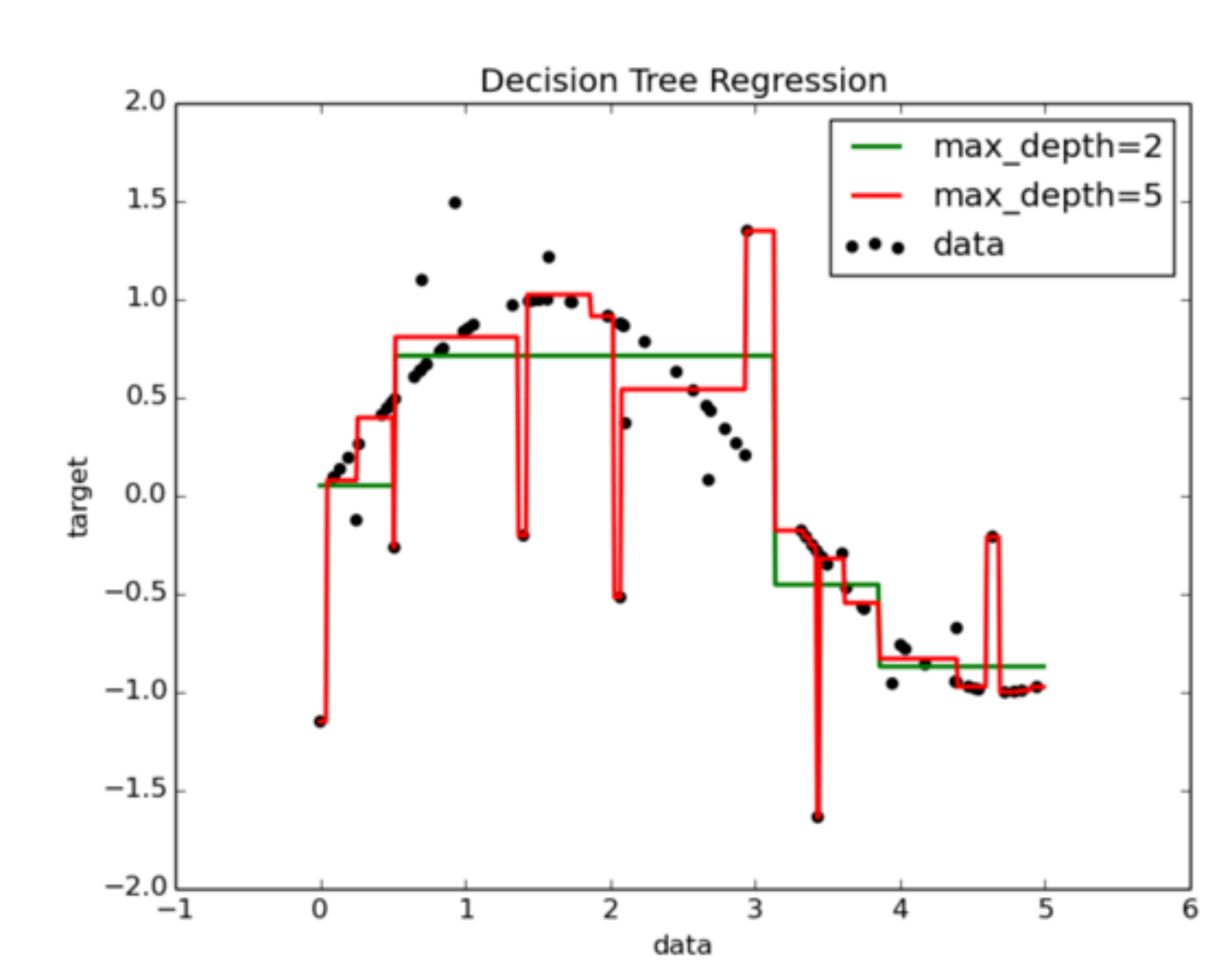
parameters:

maximum depth (controls overfitting)

maximization scheme

Application: a robot to predict surviving the Titanic (Kaggle)





http://scikit-learn.org/0.16/modules/tree.html#tree-algorithms-id3-c4-5-c5-0-and-cart

parameters:
maximum depth (controls overfitting)
maximization scheme

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

a single tree

parameters:
maximum depth (controls overfitting)
maximization scheme

gini, entropy (information content), variance…

a single tree

issues **:**

variance - different trees lead to different results

solution **: a forest**

a single tree

issues **:**

variance - different trees lead to different results

solution **: a forest**

- run many tree models,
- look at the ensemble result

Ensemble methods:

Random forest:

- trees run in parallel (independently of each other)
- each tree uses a random subset of observations/features (boostrap - bagging)
- class predicted by *majority vote:* what class do most trees think a point belong to?

Gradient boosted trees:

- trees run in series (one after the other)
- each tree uses different weights for the features learning the weighs from the previous tree
- the last tree has the prediction

Ensemble methods:

Random forest:

- trees run in parallel (independently of each other)
- each tree uses a random subset of observations/features (boostrap - bagging)
- class predicted by *majority vote:* what class do most trees think a point belong to?

Gradient boosted trees:

- trees run in series (one after the other)
- each tree uses different weights for the features learning the weighs from the previous tree
- the last tree has the prediction

More parameters:

- BOTH: depth, criterion, min sample to split, min sample in leaf
- RF: number of trees, number of features/tree
- GB: loss function, learning rate, number of boosts

How good is my model?

https://scikit-learn.org/stable/modules/model_evaluation.html

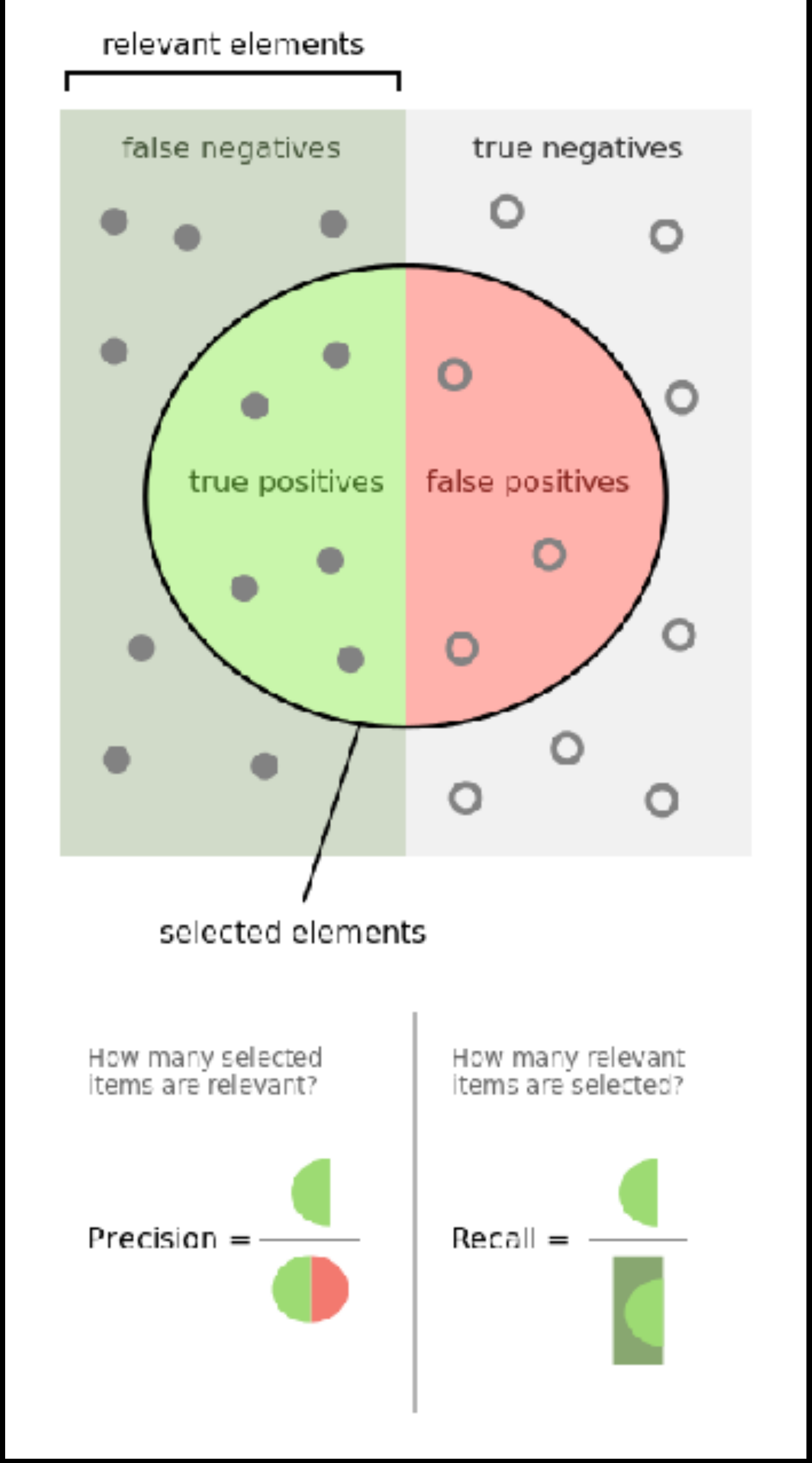
	H_0 is True	H_0 is False
H_0 is falsified	Type I error False Positive important message gets spammed	True Positive
H_0 is not falsified	True Negative	Type II error False negative Spam in your Inbox

$$LR = \frac{\text{False Negative}}{\text{True Negative}}$$

	H_0 is True	H_0 is False
H_0 is falsified	Type I error False Positive important message gets spammed	True Positive
H_0 is not falsified	True Negative	Type II error False negative Spam in your Inbox

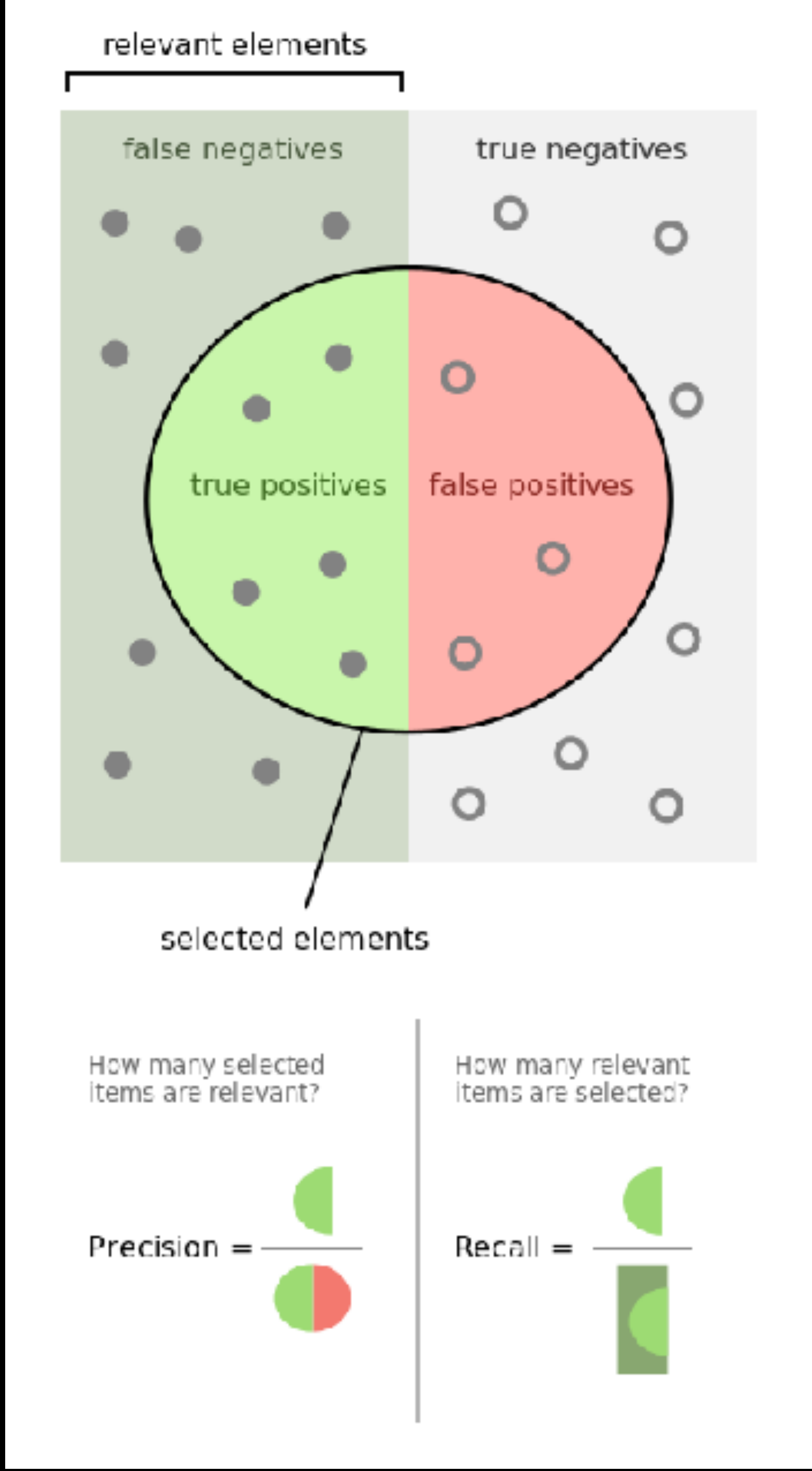
Precision = (TP)/(TP + FP)

Recall = (TP)/(TP + FN)



Precision = (TP)/(TP + FP)

Recall = (TP)/(TP + FN)



Accuracy = (TP+TN)/(TP+TN+FP+FN)

How good is my model?

https://scikit-learn.org/stable/modules/model_evaluation.html

Is my model overfitting?

cross validation:

super important missing topic:
pruning!
when is my tree overfitting?

dont just do linear regression!

http://scikit-learn.org/0.16/modules/tree.html#tree-algorithms-id3-c4-5-c5-0-and-cart

Reading:

An excellent use of viz for data exploration and transition to inferential analysis

https://blog.data.gov.sg/how-we-caught-the-circle-line-rogue-train-with-data-79405c86ab6a#.iz1r655xo

Lee Shangqian, Daniel Sim & Clarence Ng

Decision trees:

http://what-when-how.com/artificial-intelligence/decision-tree-applications-for-data-modelling-artificial-intelligence/

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4466856/

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4380222/