Urban Informatics

Fall 2018

dr. federica bianco fbianco@nyu.edu

@fedhere

X: 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)
- topics in Time Series Analysis

X: decision trees

machine learning

models with parameters that are “learned” from the data

CUSP

machine learning

models with parameters that are “learned” from the data

parameters that are optimized based on the data

machine learning

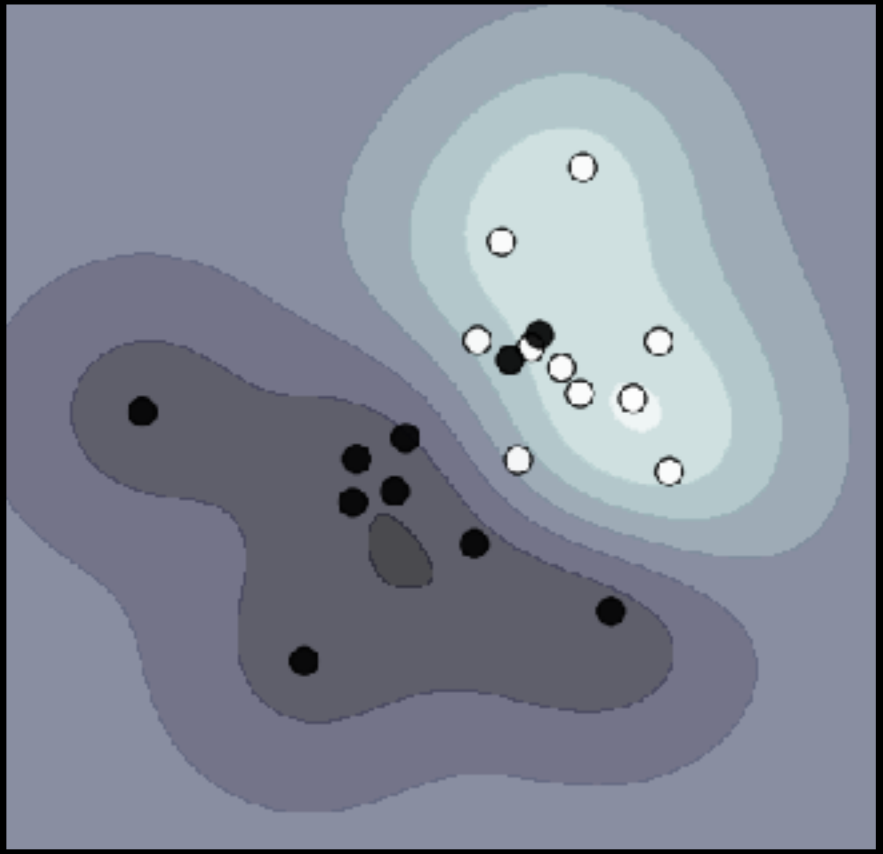
algorithms that can learn from and make predictions on data.

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

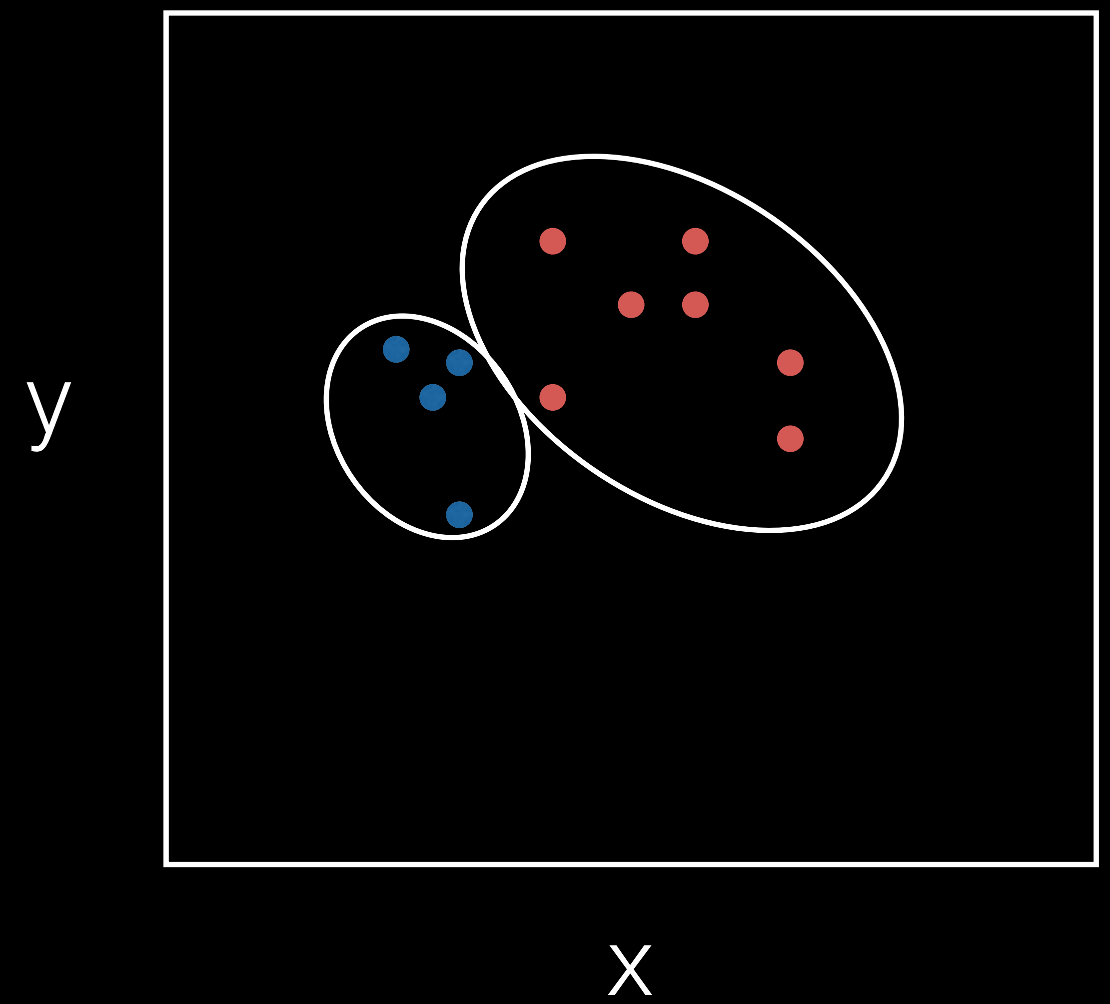
understanding structure

organizing + compressing data

(classification, feature learing)

CLUSTERING

Supervised Learning

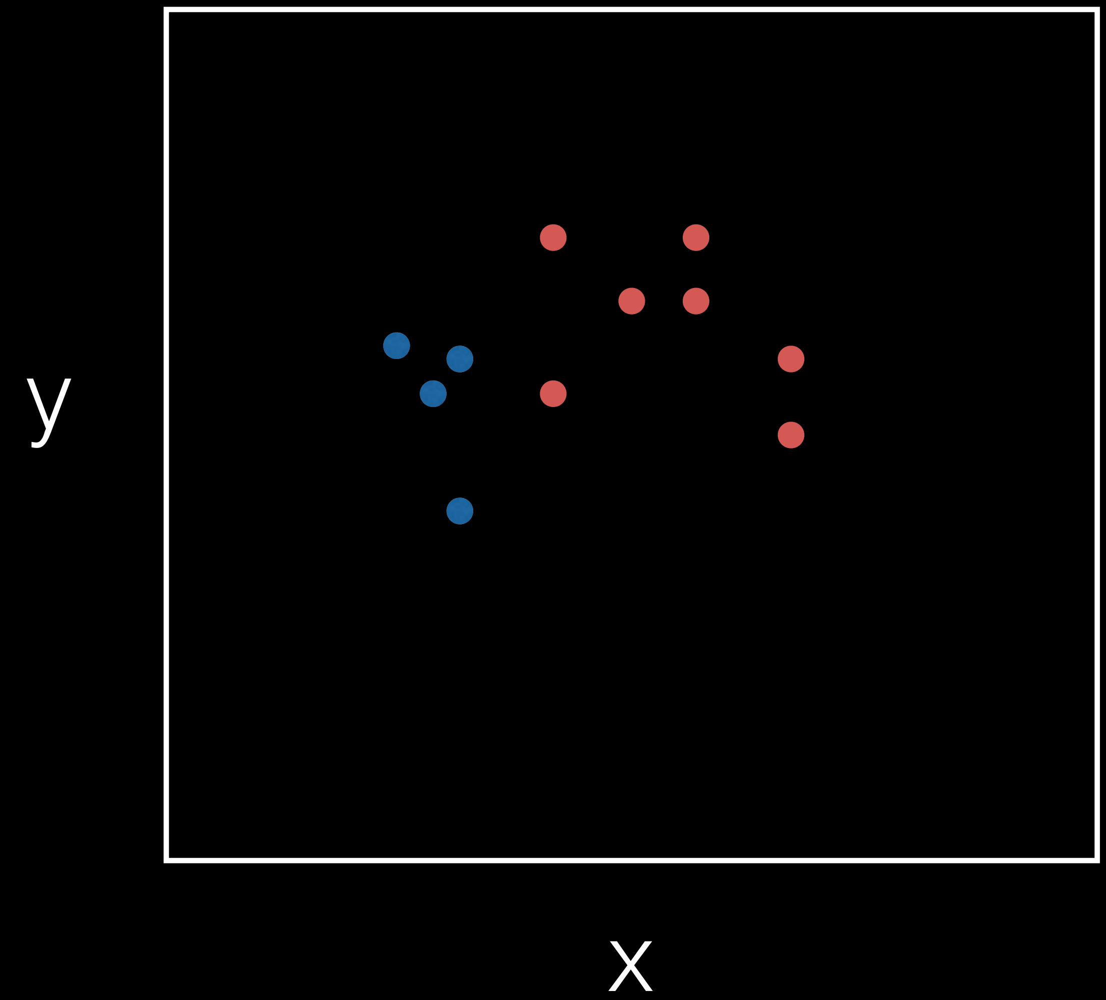


***observed**:*
(x, y, color)

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)

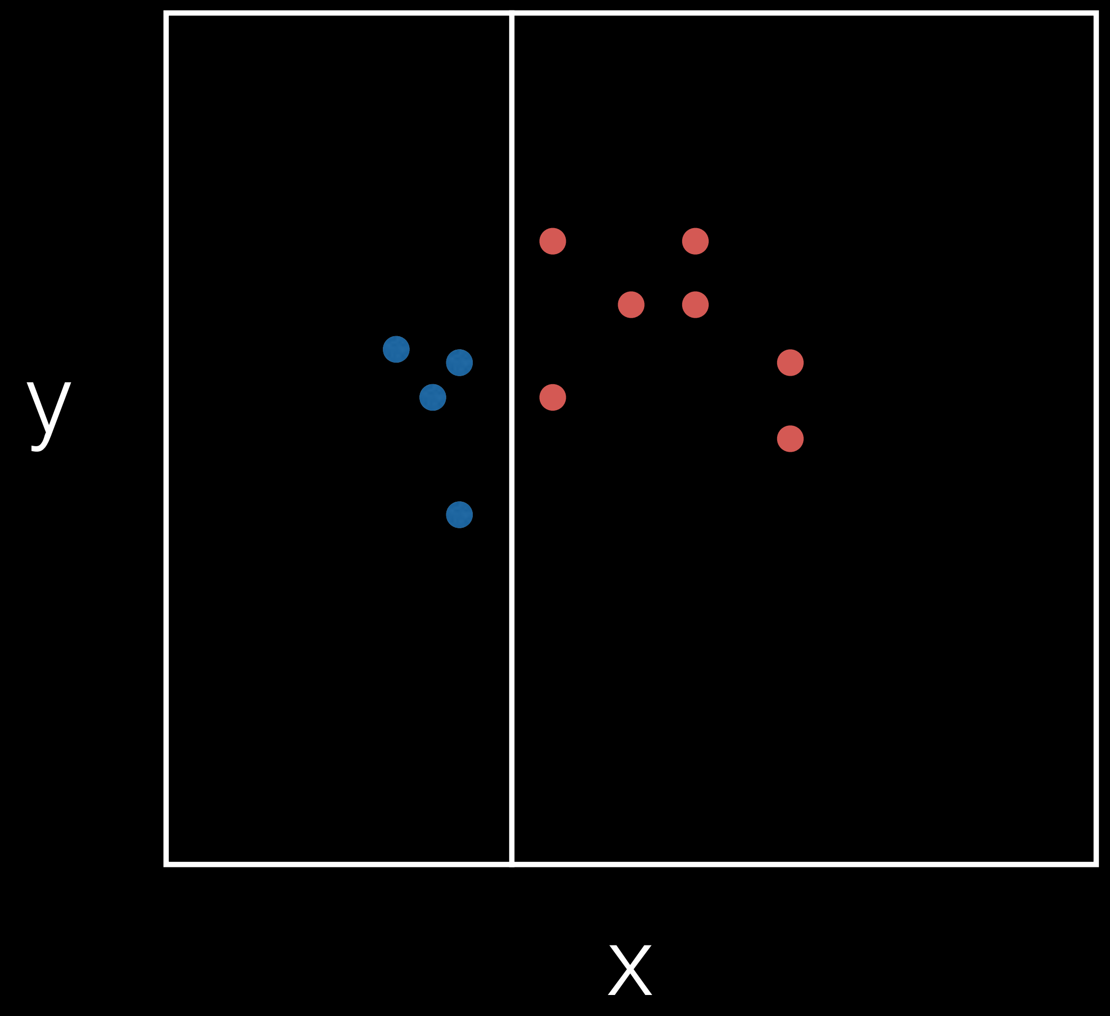


target:
(color)

Partitioning methods: classifying

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

observed:
(x, y)

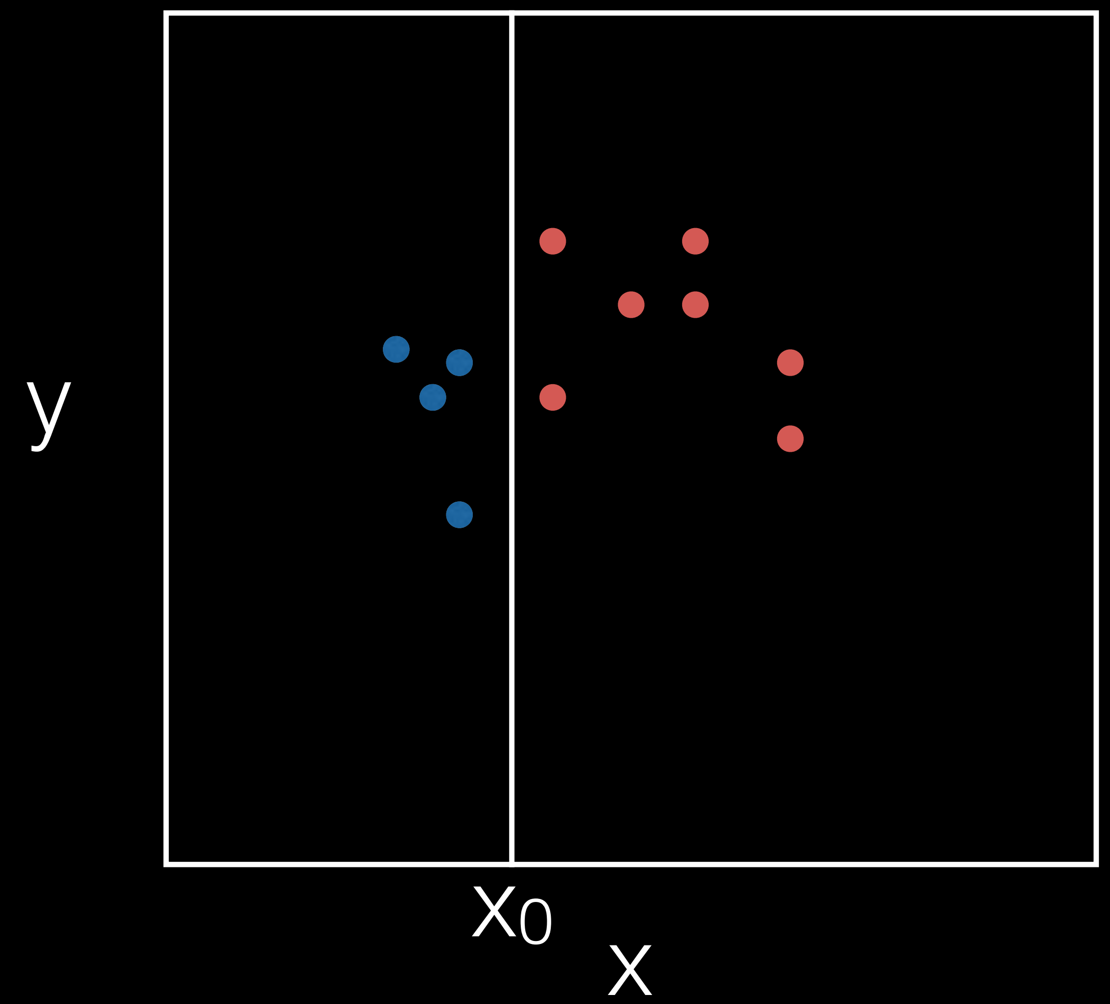


target:
(color)

Partitioning methods: classifying

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

***observed**:*
(x, y)



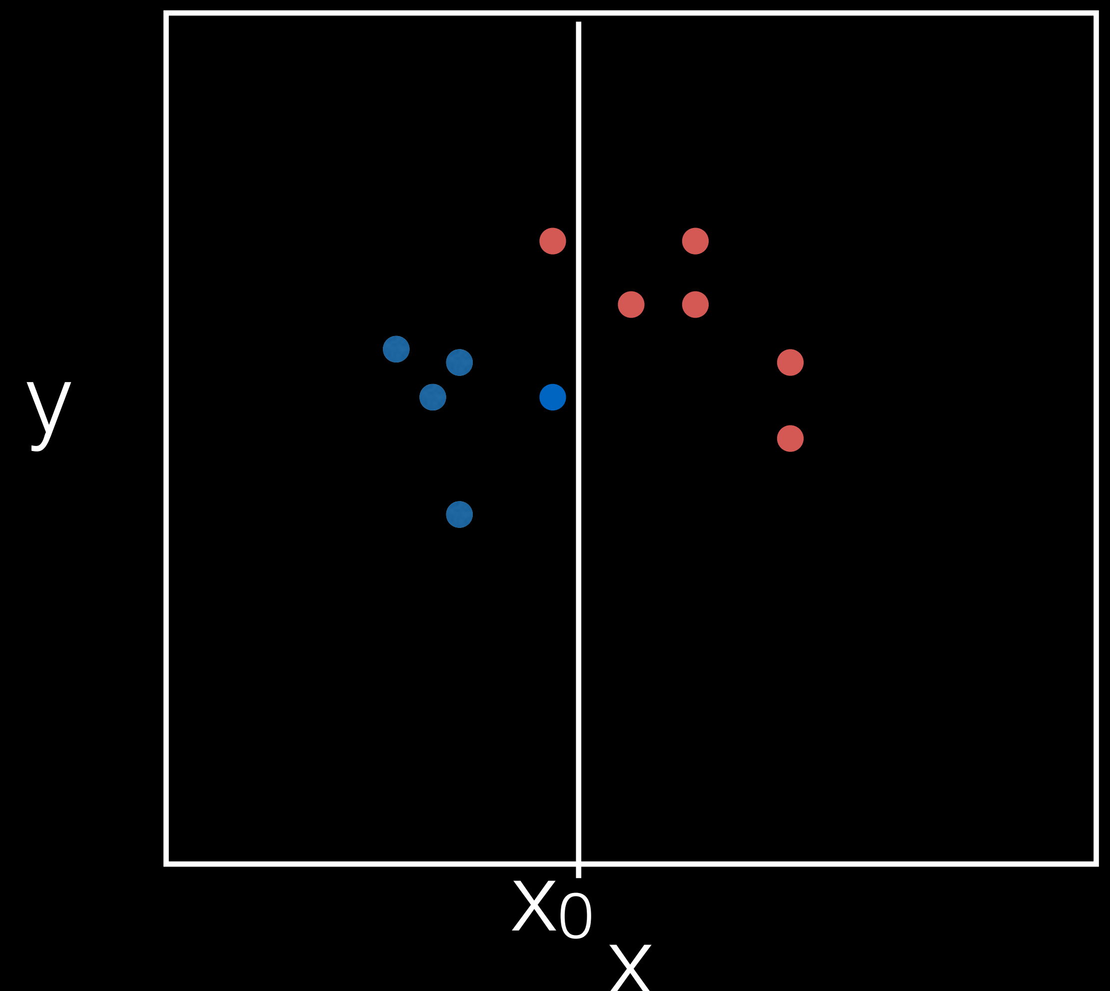
***target**:*
(color)

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)

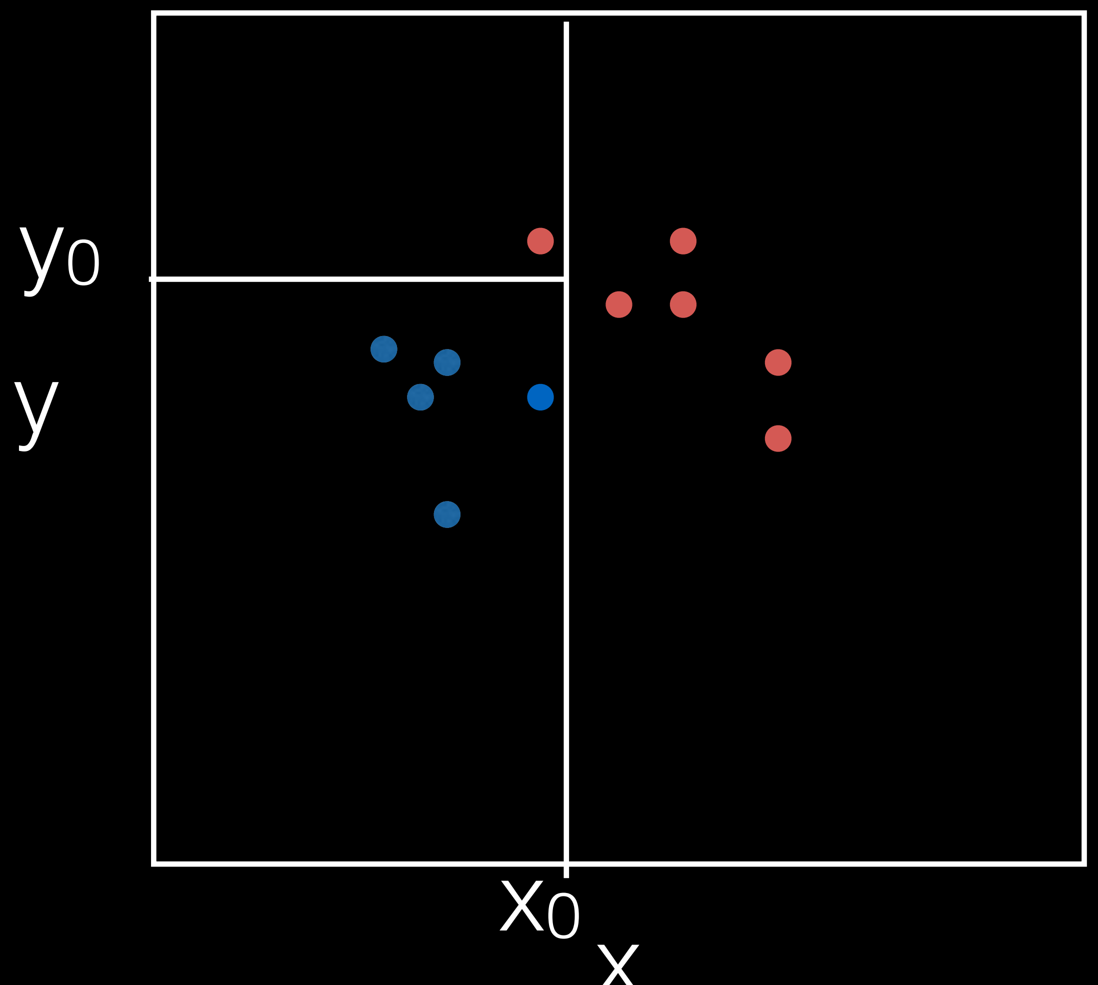


target:
(color)

Partitioning methods: classifying

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

observed:
(x, y)



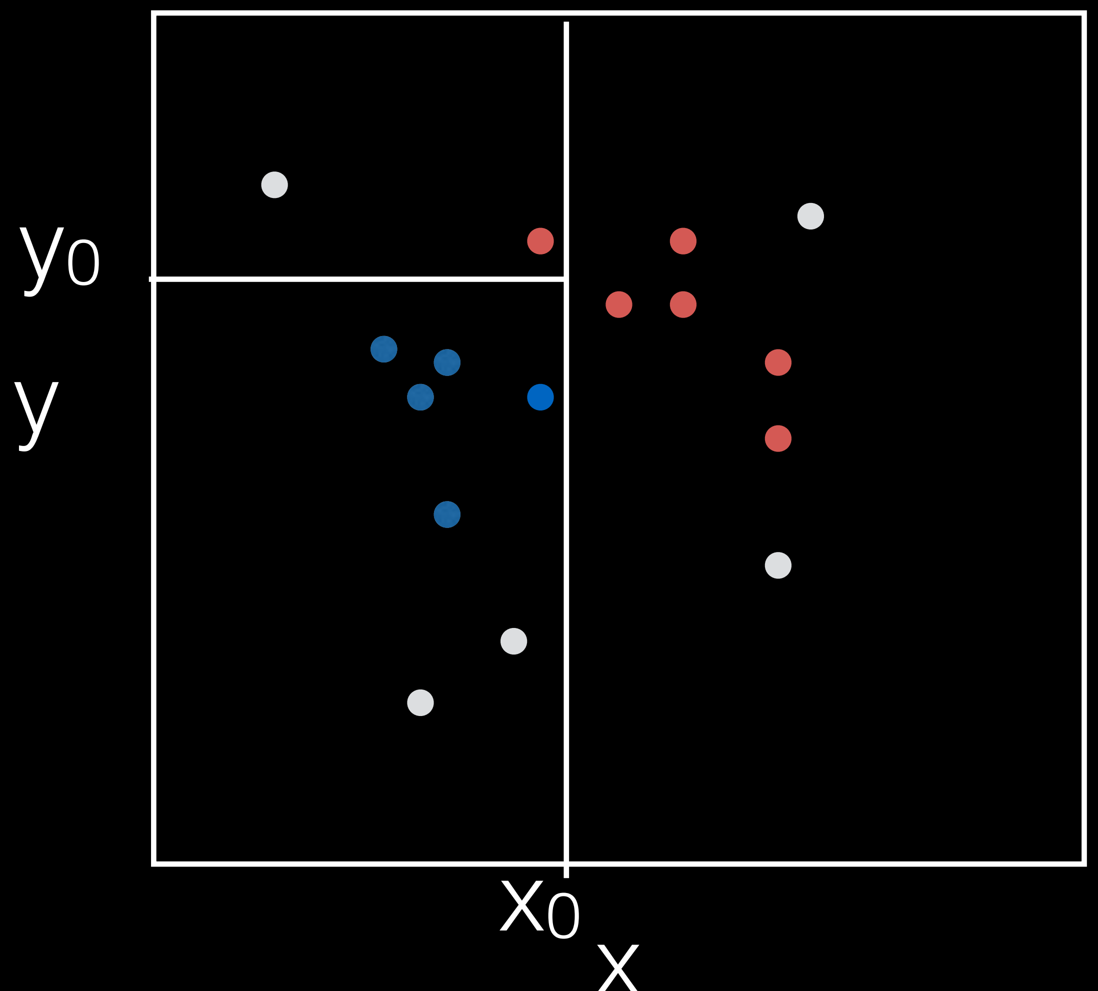
target:
(color)

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)



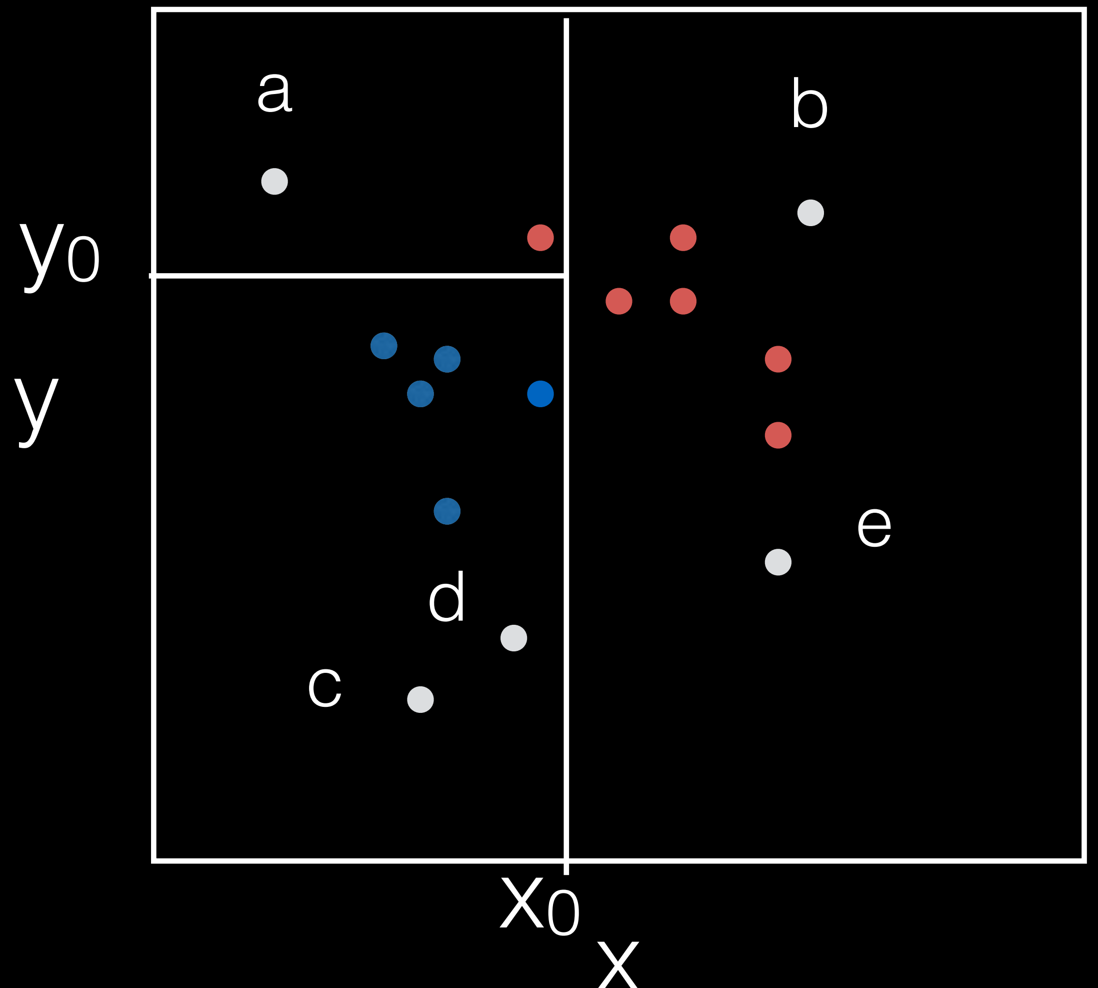
target:
(color)

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

X: decision trees

Partitioning methods: classifying

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



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

observed:
(x, y)

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

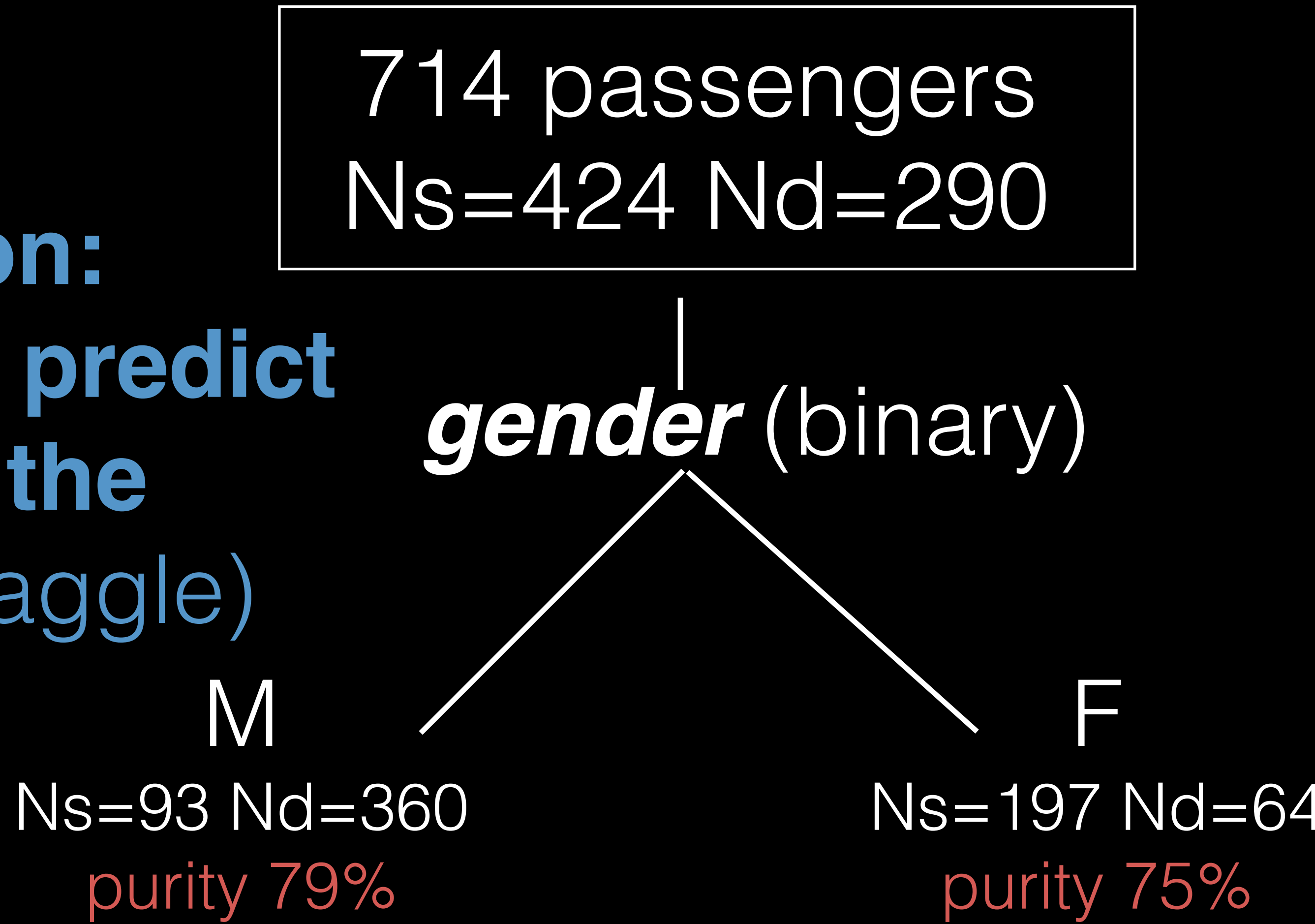
Application:
a robot to predict surviving the Titanic (Kaggle)

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)

714 passengers
Ns=424 Nd=290

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

class (categorical)

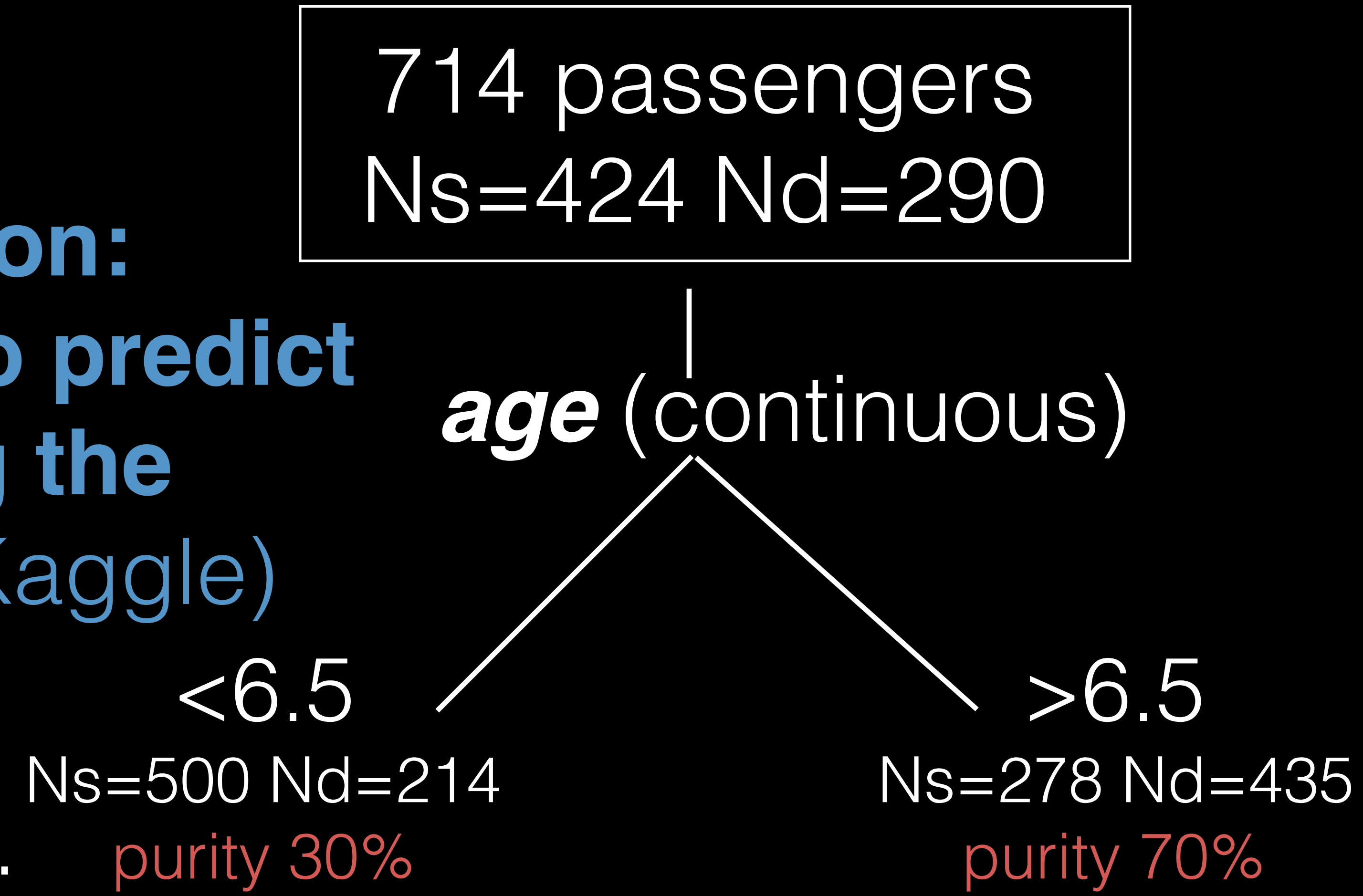
1st
Ns=471 Nd=242
purity 66%

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

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

target variable:
survival (y/n)

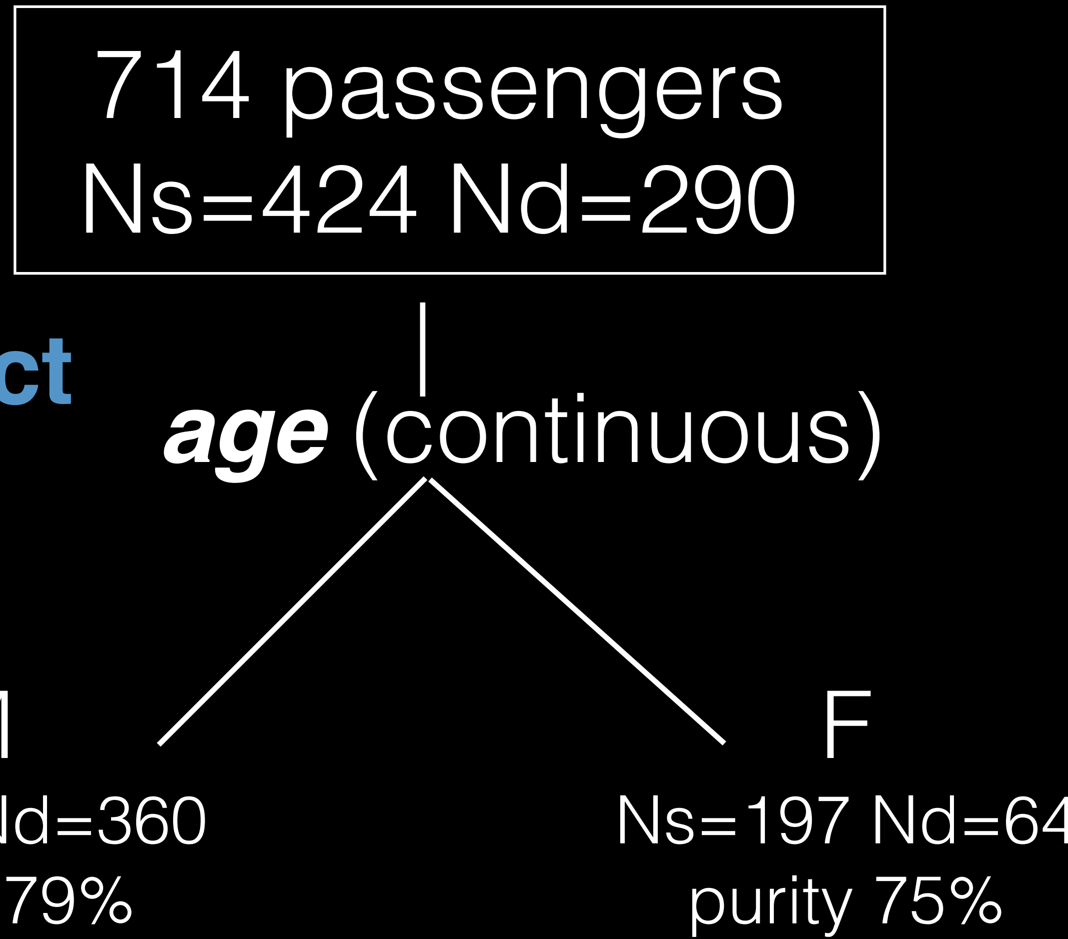
Application: a robot to predict surviving the Titanic (Kaggle)



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)



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%

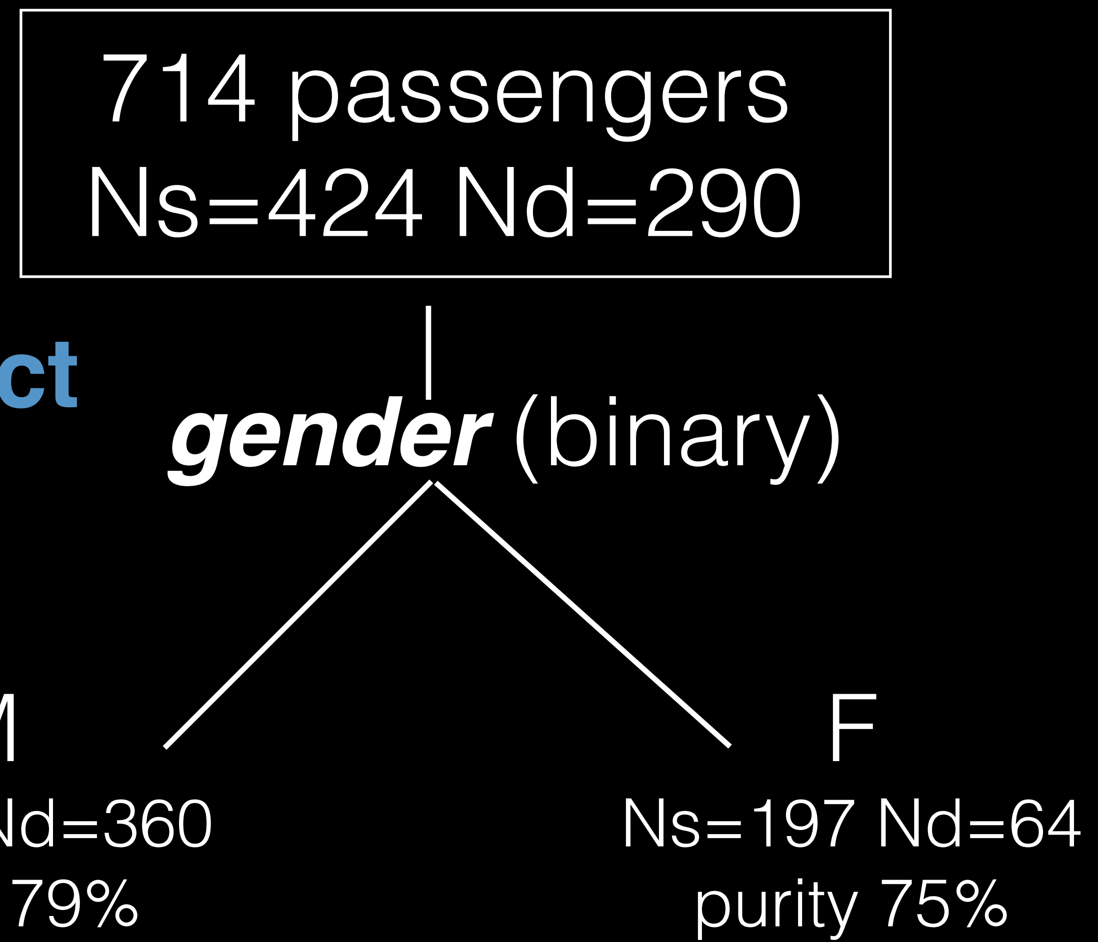
age: M 74/67% F 96/40%

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)

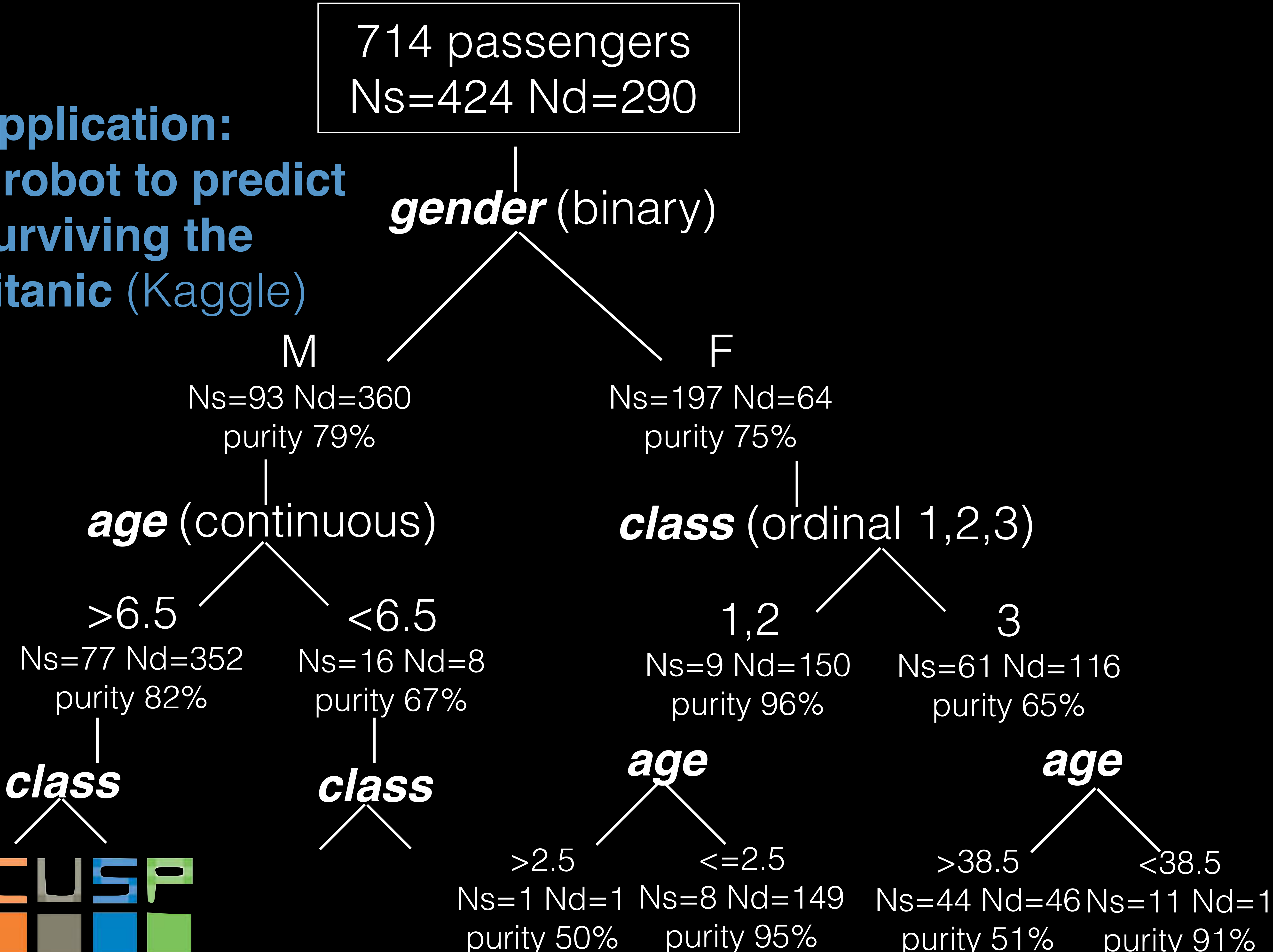
Application: a robot to predict surviving the Titanic (Kaggle)

714 passengers
Ns=424 Nd=290

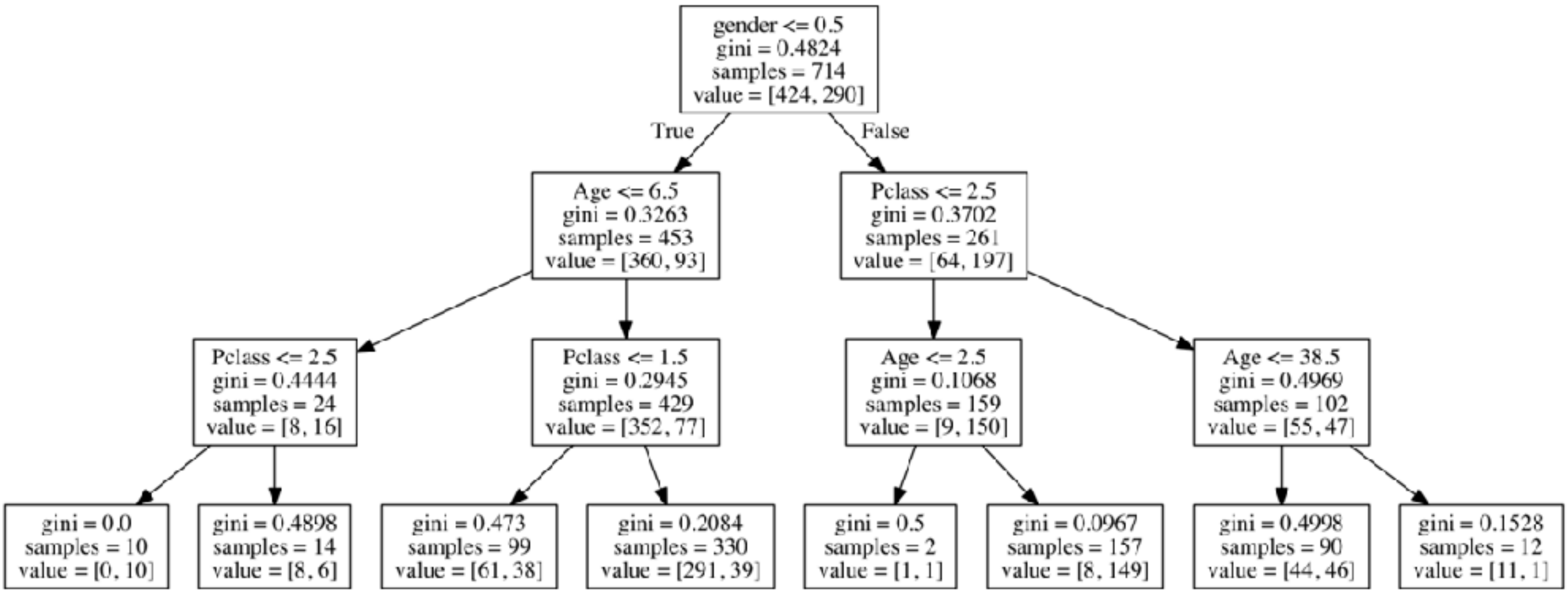
gender (binary)

- **M**
  Ns=93 Nd=360
  purity 79%
  - ***age*** (continuous)
    - **>6.5**
      Ns=77 Nd=352
      purity 82%
    - **<6.5**
      Ns=16 Nd=8
      purity 67%
- **F**
  Ns=197 Nd=64
  purity 75%
  - ***class*** (ordinal 1,2,3)
    - **1**
      Ns=82 Nd=3
      purity 96%
    - **2,3**
      Ns=114 Nd=62
      purity 65%

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

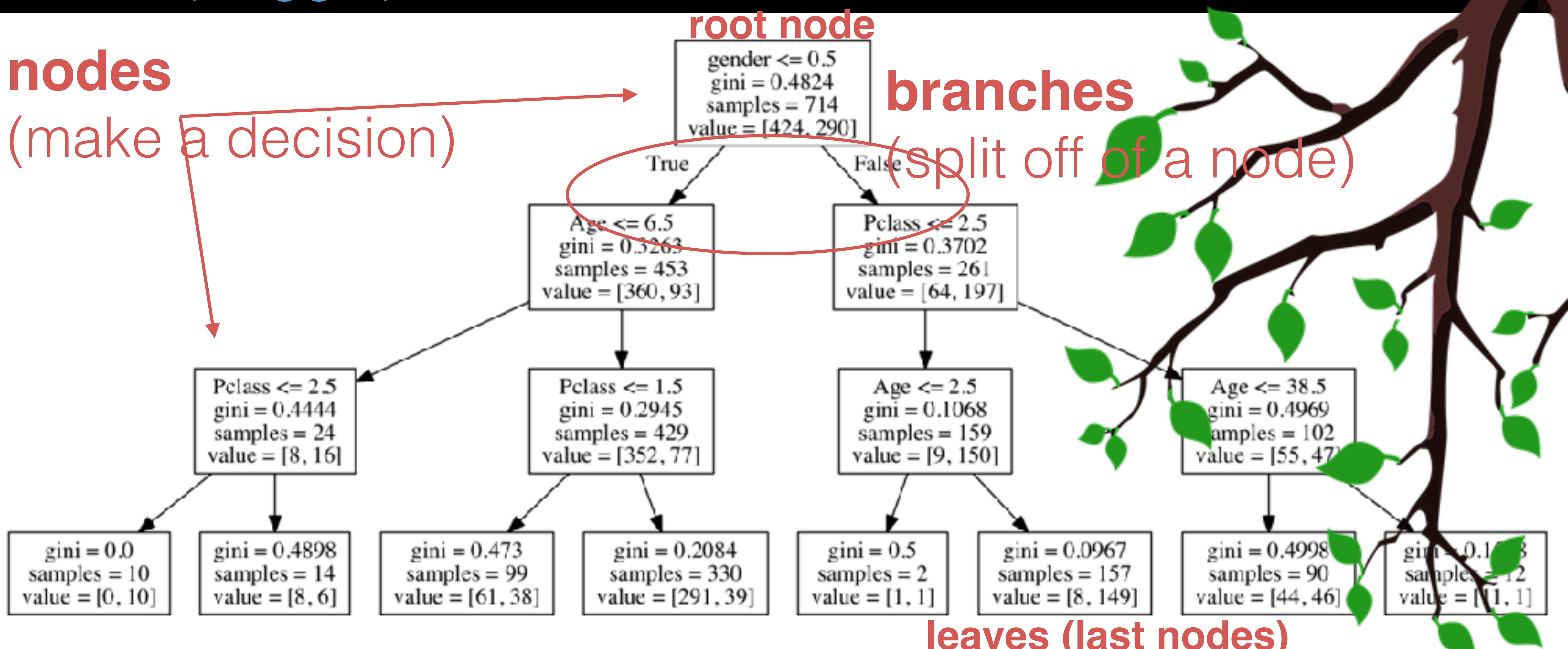


Application: a robot to predict surviving the Titanic (Kaggle)



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

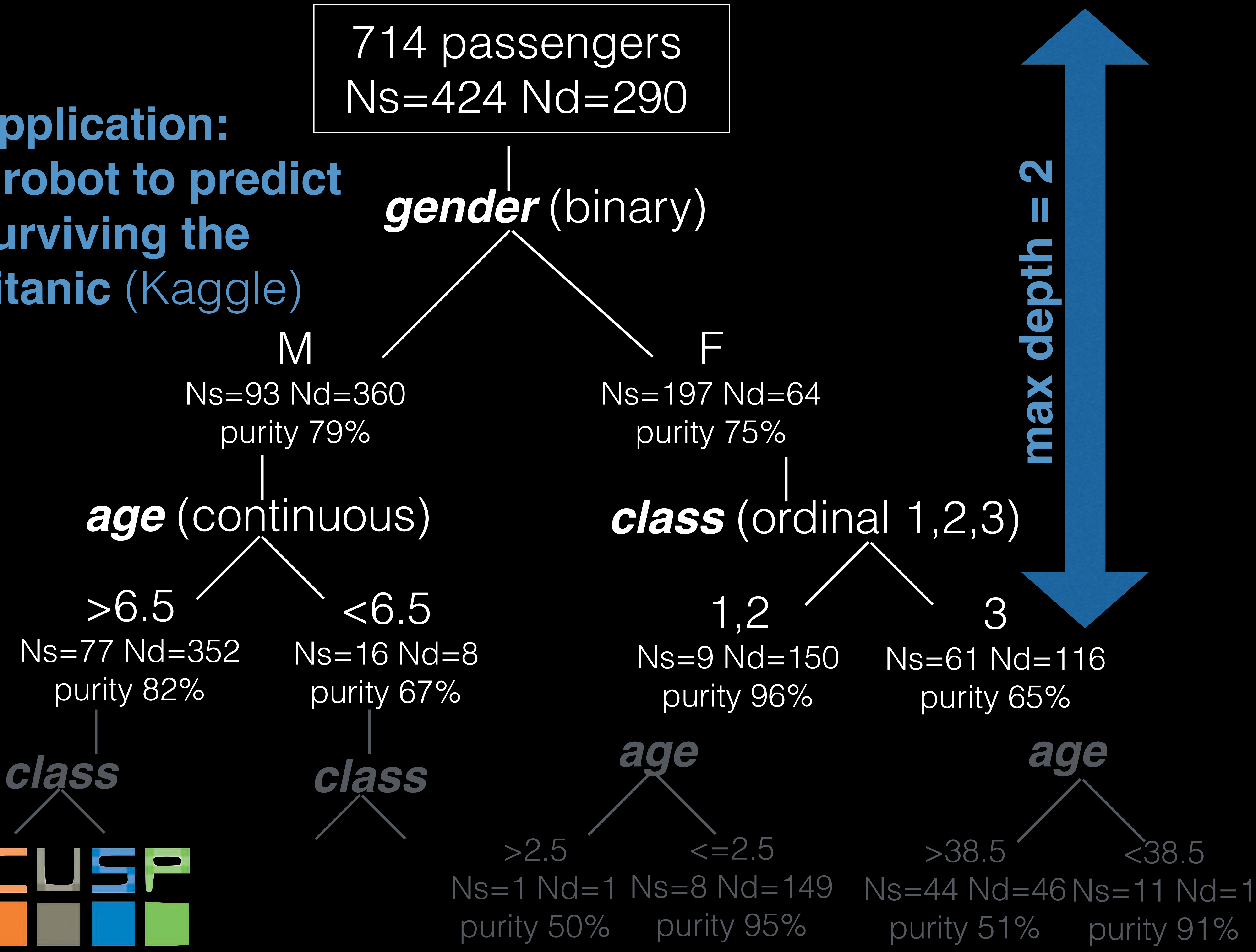
a single tree

parameters:

maximum depth (controls overfitting)

maximization scheme

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



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

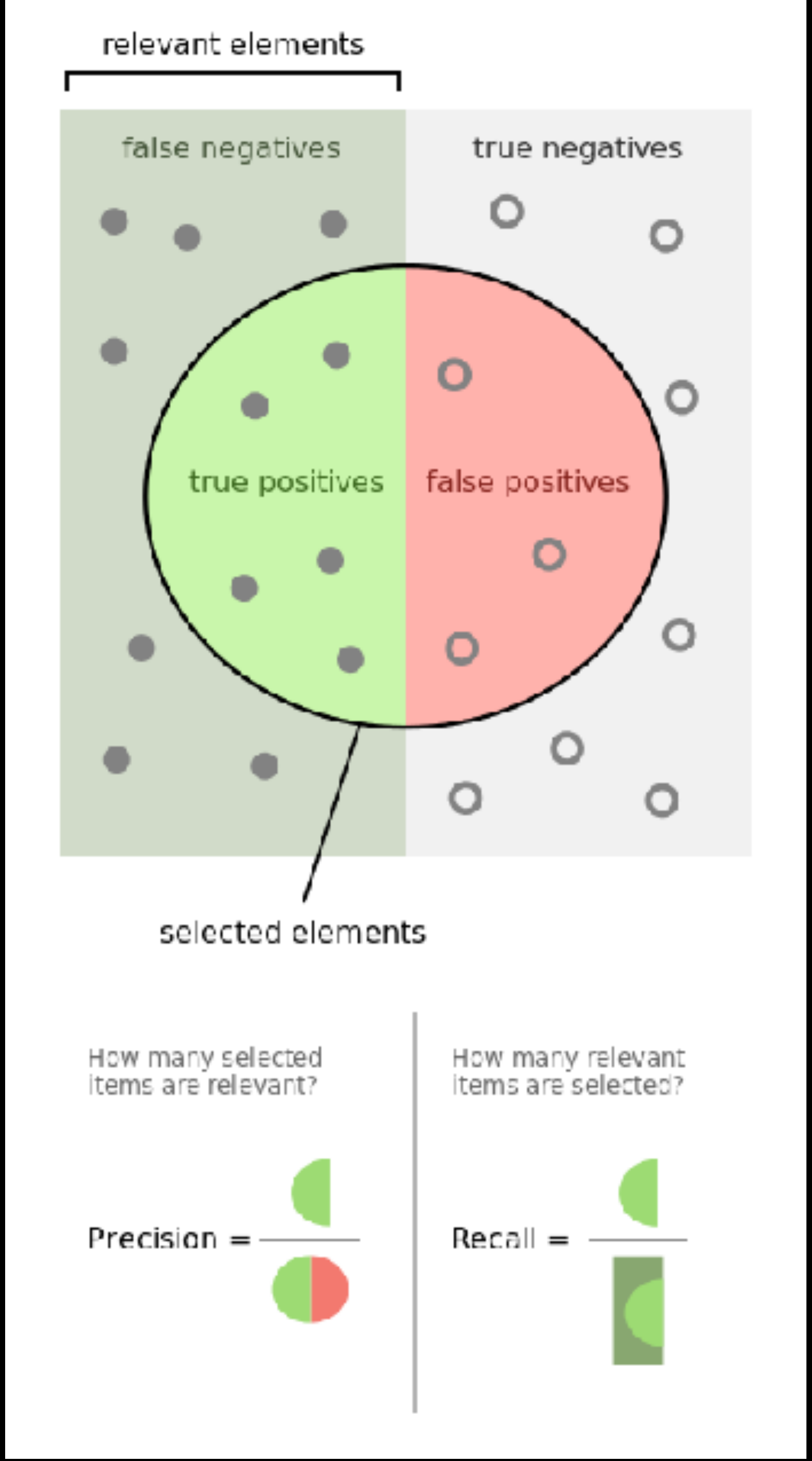
	H_0 is True	H_0 is False
H_0 is falsified	Type I error False Positive important message gets spammed	True Positive
H_0 is not falsified	True Negative	Type II error False negative Spam in your Inbox

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

	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

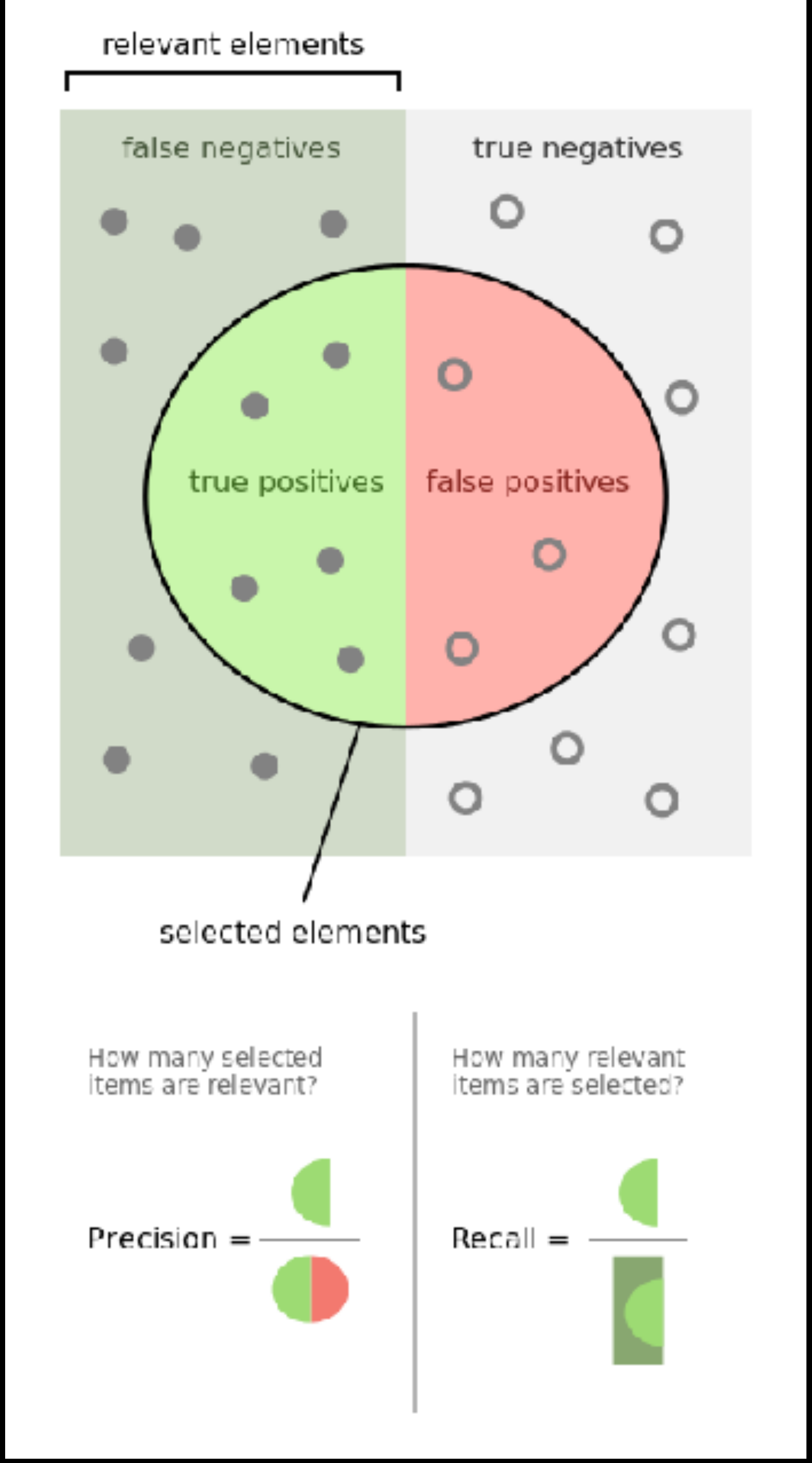
Precision = (TP)/(TP + FP)

Recall = (TP)/(TP + FN)



Precision = (TP)/(TP + FP)

Recall = (TP)/(TP + FN)



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

1.10.7.1. Classification criteria

If a target is a classification outcome taking on values 0,1,...,K-1, for node m, representing a region R_m with N_m observations, let

$$p_{mk} = 1/N_m \sum_{x_i \in R_m} I(y_i = k)$$

Common measures of impurity are Gini

$$H(X_m) = \sum_k p_{mk}(1 - p_{mk})$$

Cross-Entropy

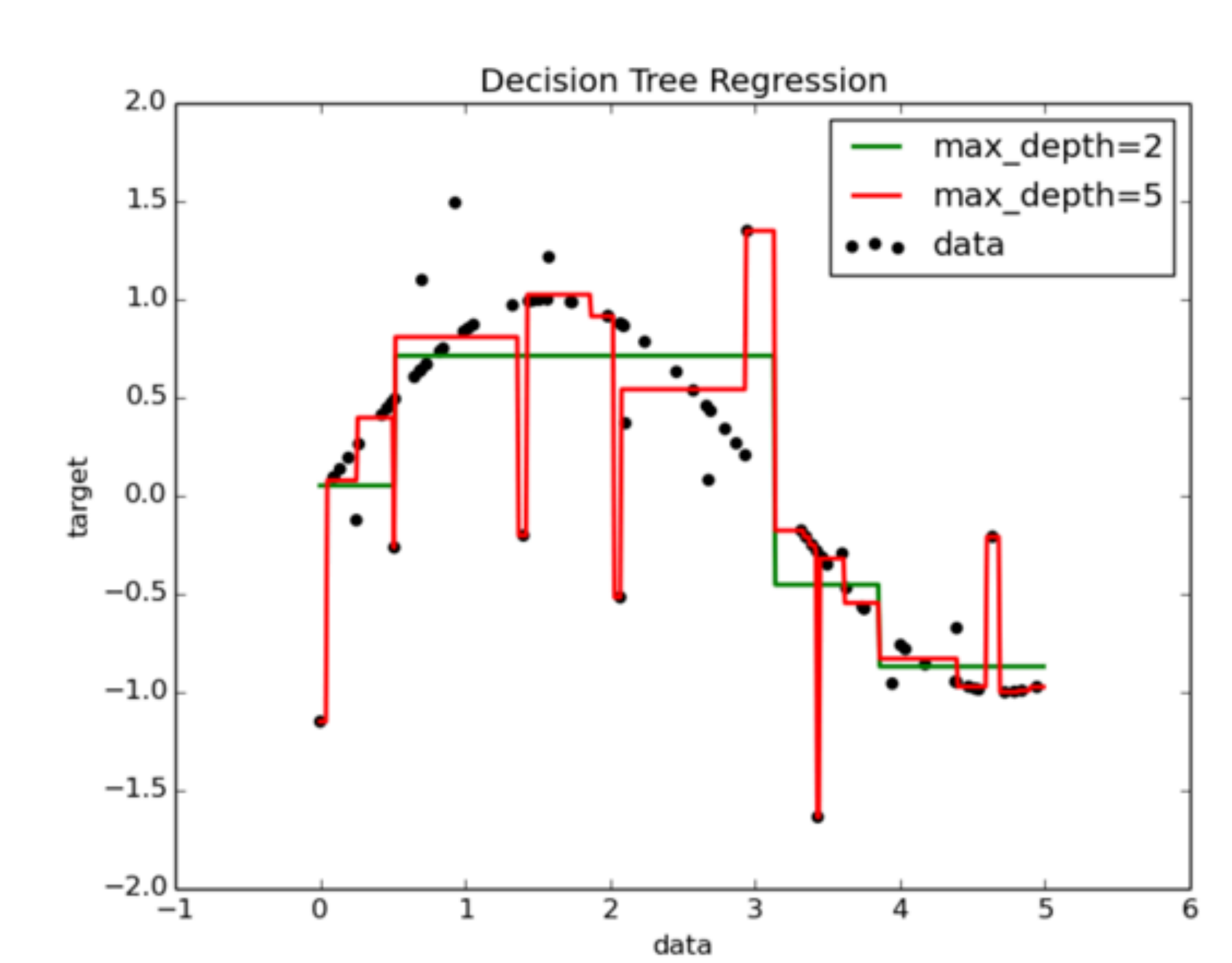
$$H(X_m) = \sum_k p_{mk} \log(p_{mk})$$

and Misclassification

$$H(X_m) = 1 - \max(p_{mk})$$

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

Regression Trees

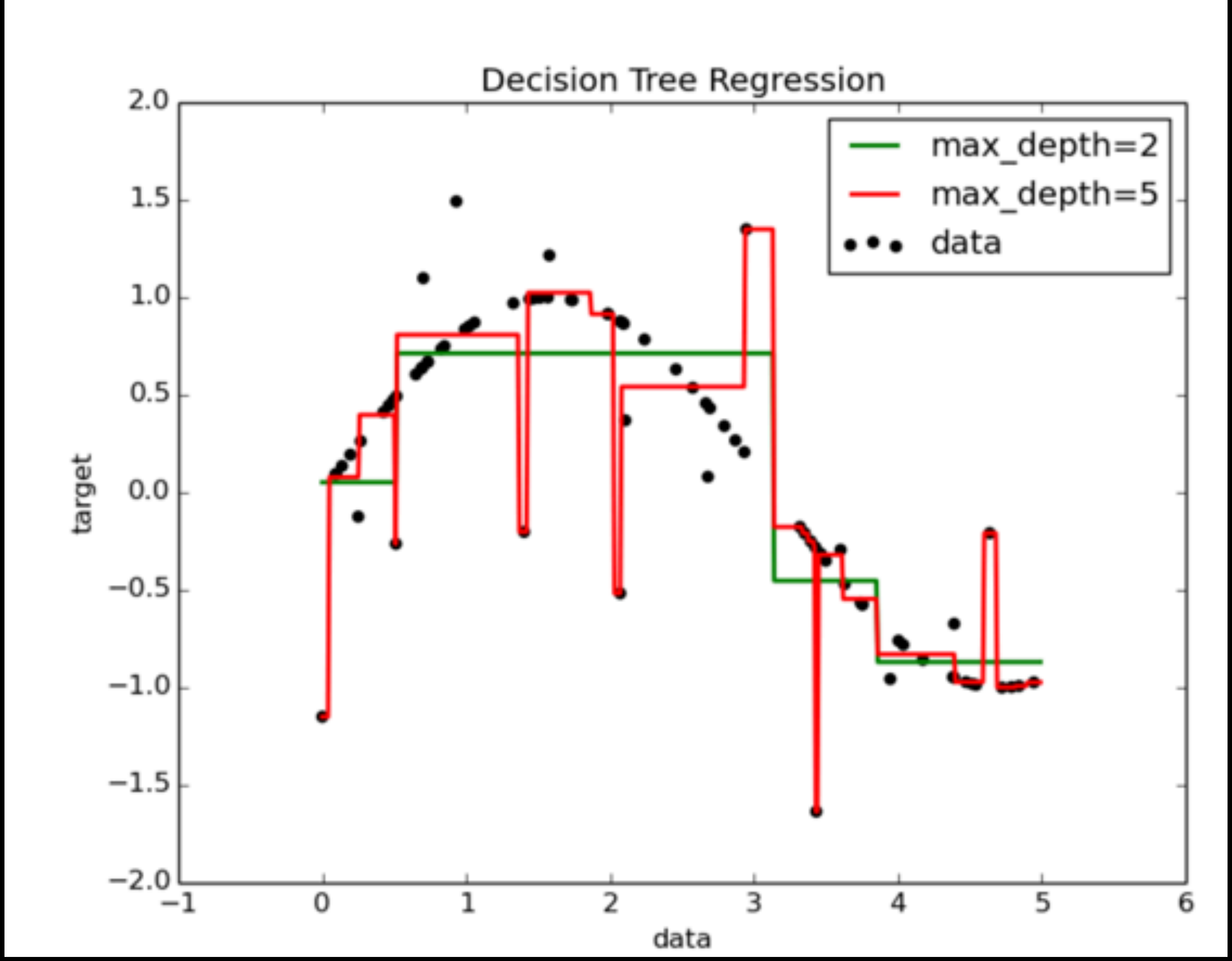


http://scikit-learn.org/0.16/modules/tree.html#tree-algorithms-id3-c4-5-c5-0-and-cart

Regression Trees

$$c_m = \frac{1}{N_m} \sum_{i \in N_m} y_i$$

$$H(X_m) = \frac{1}{N_m} \sum_{i \in N_m} (y_i - c_m)^2$$



How good is my model?

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

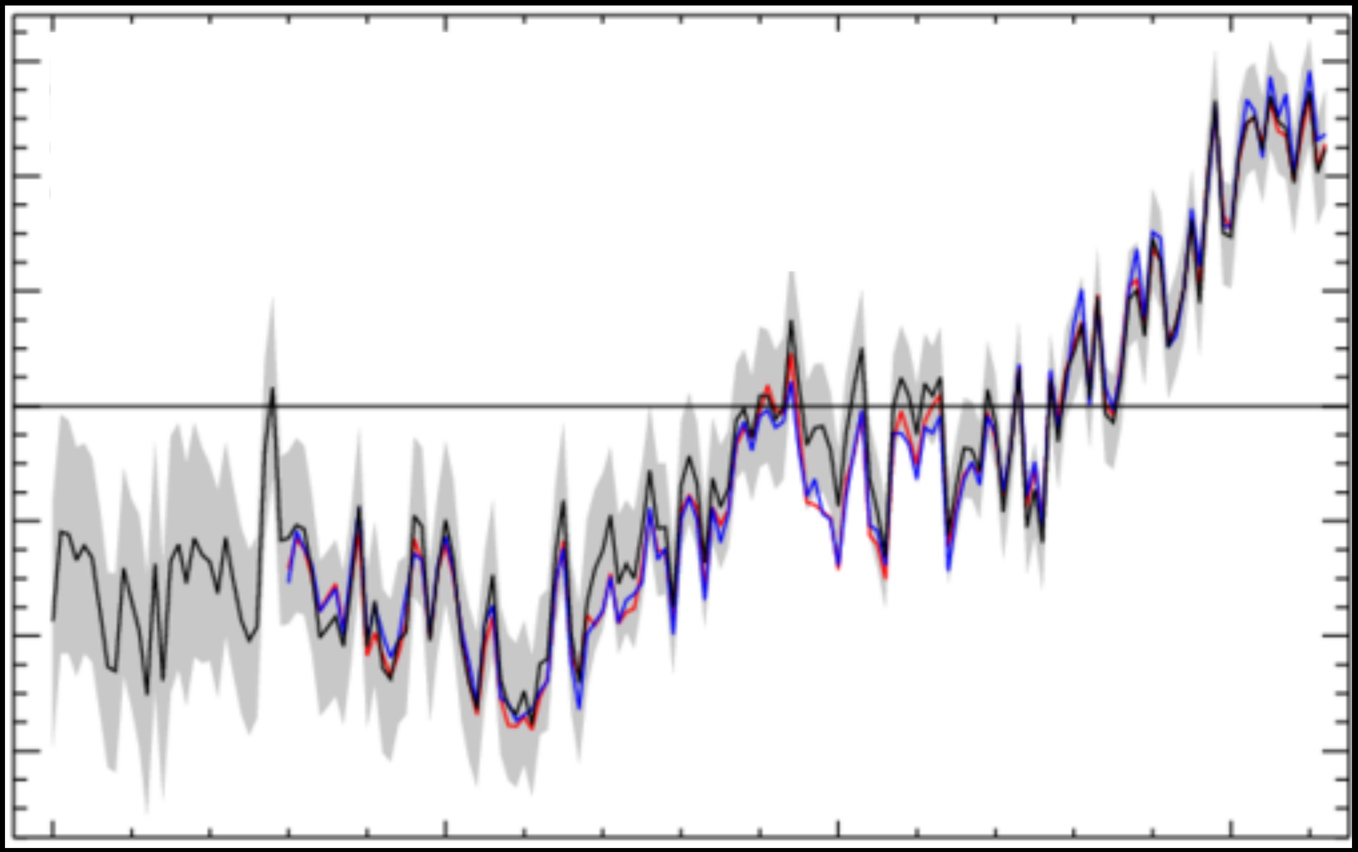
Is my model overfitting?

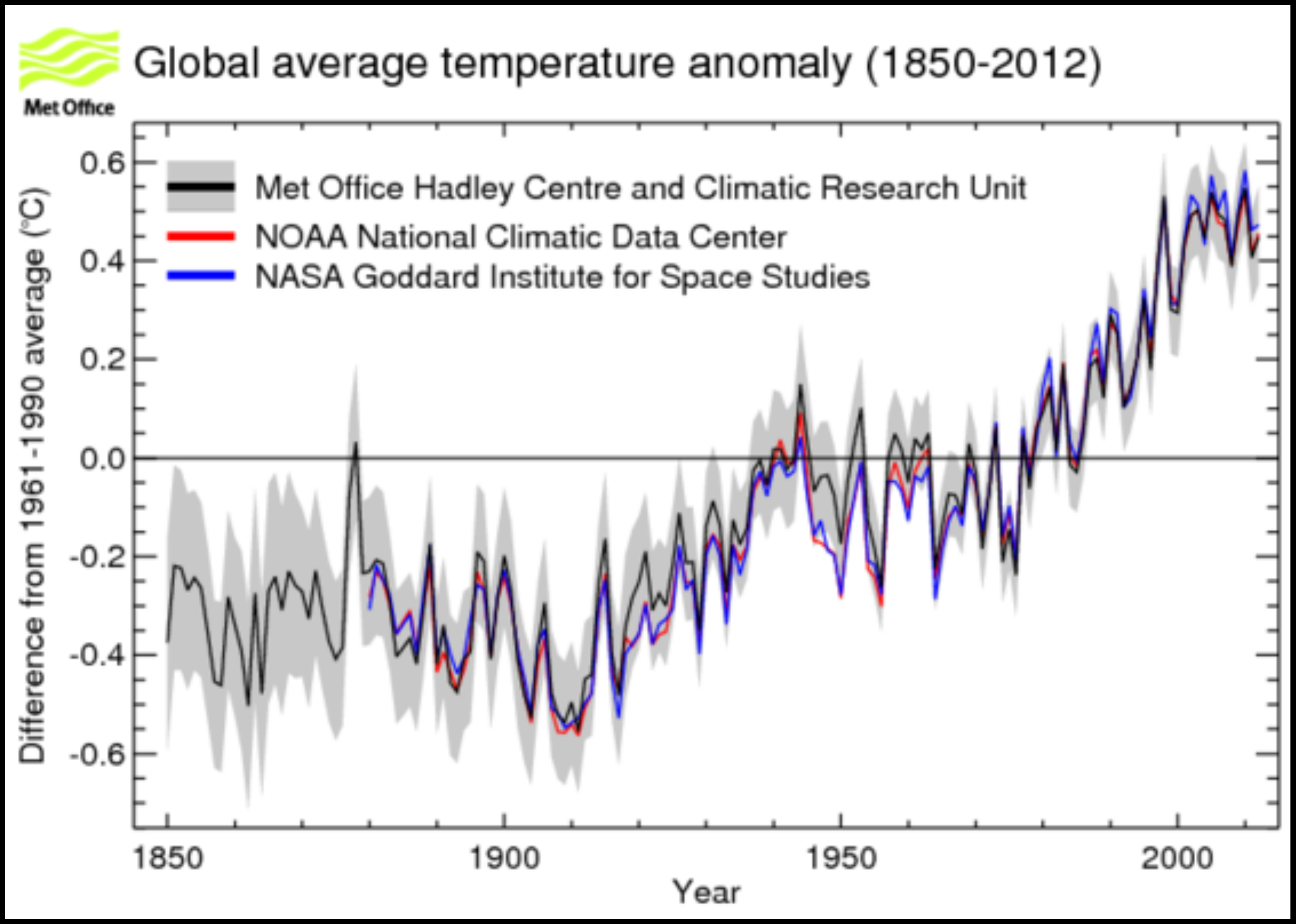
cross validation:

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

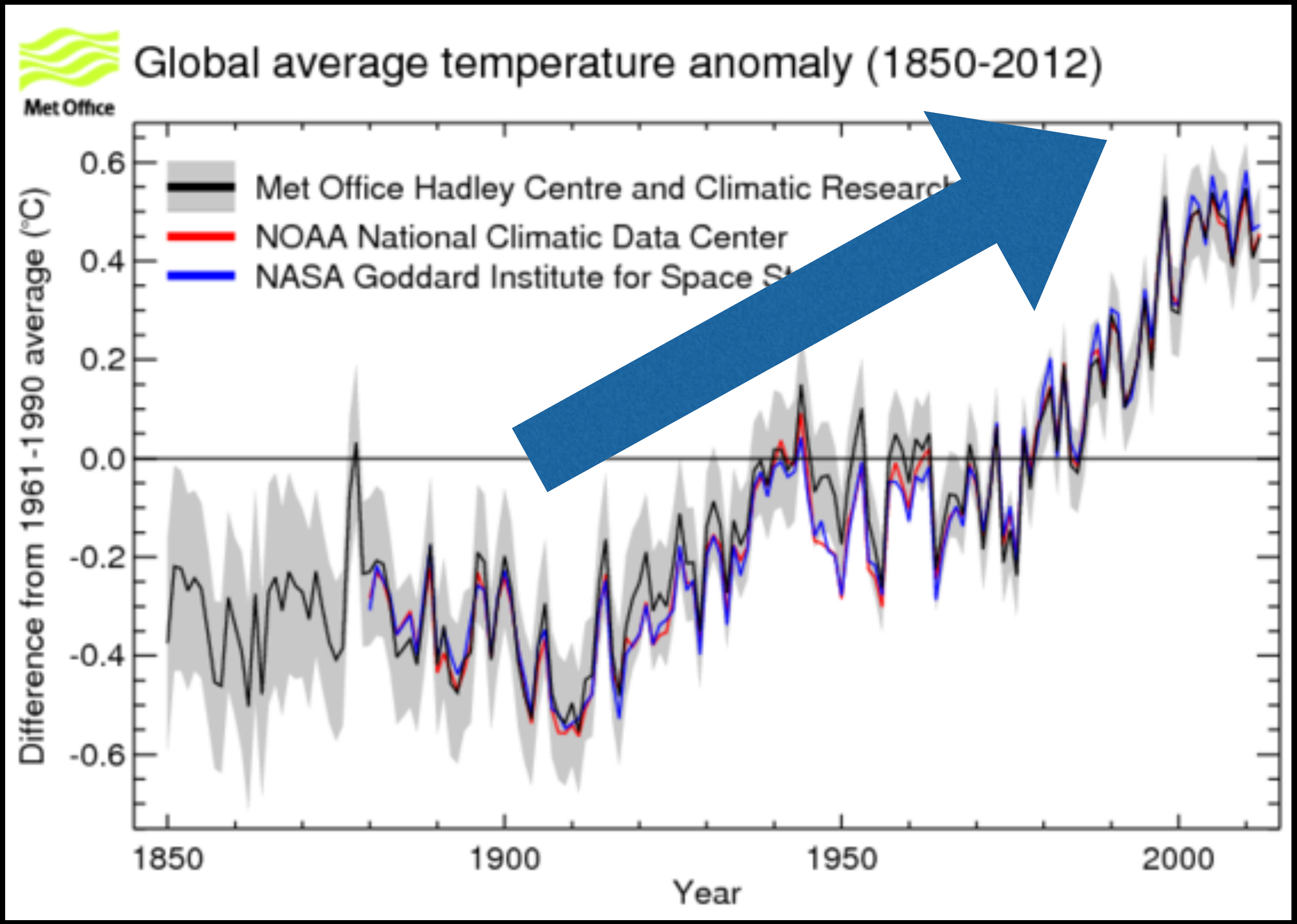
Topics in (time) series analysis

- smoothing
- de-trending
- event detection
- period finding (Fourier analysis)
- clustering (including anomaly detection)

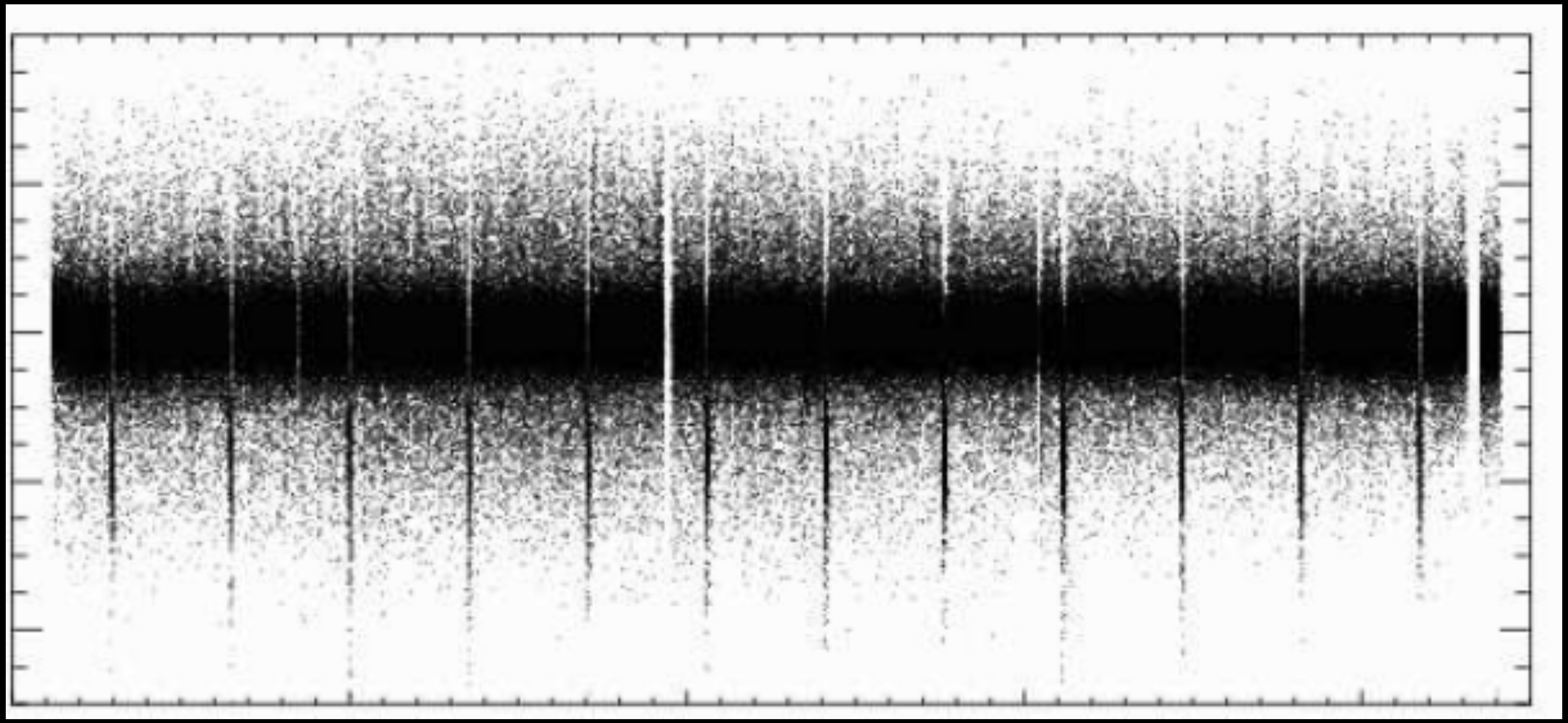


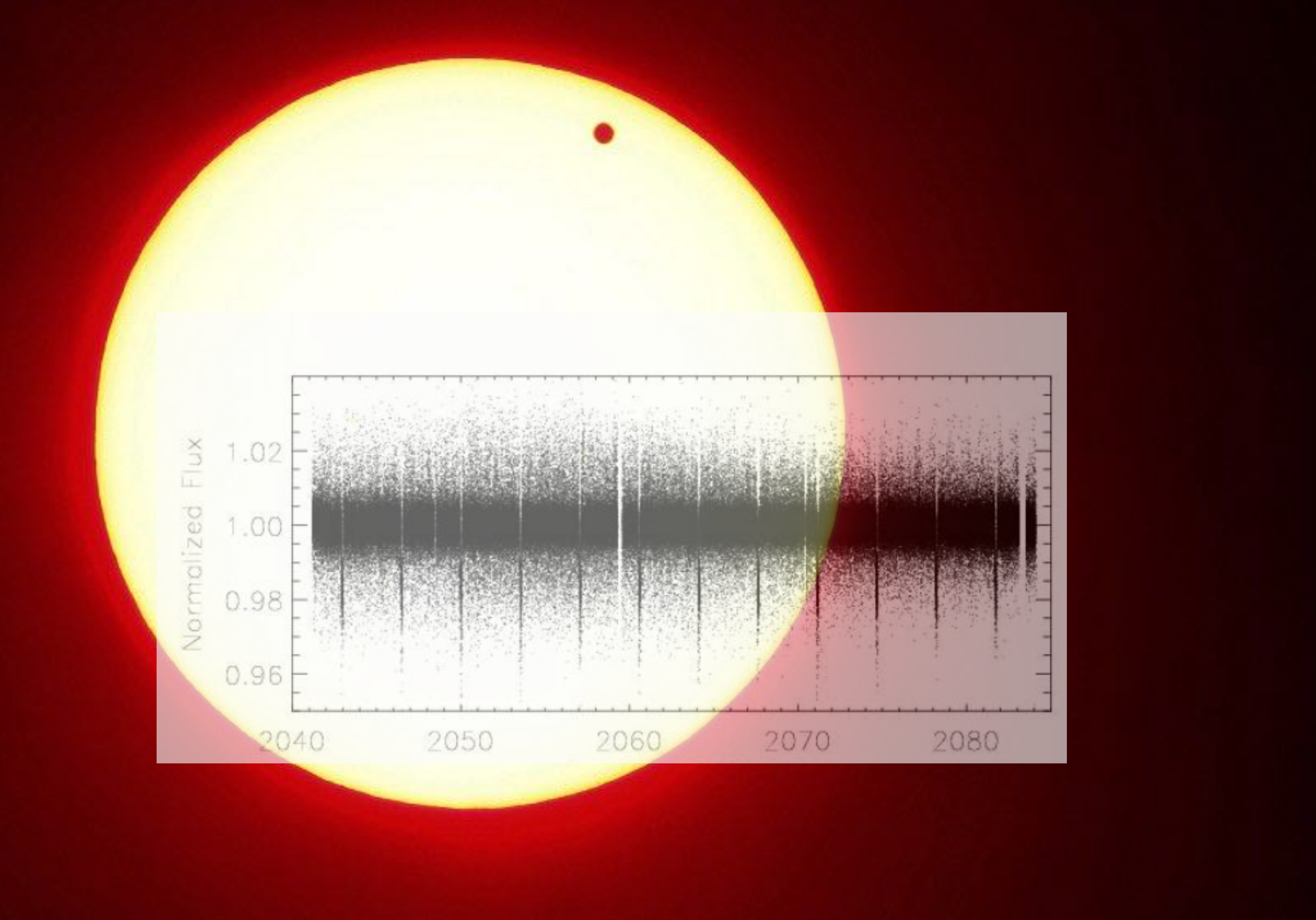

Global average temperature anomaly (1850-2012)
Met Office
Met Office Hadley Centre and Climatic Research Unit
NOAA National Climatic Data Center
NASA Goddard Institute for Space Studies
Difference from 1961-1990 average (°C)
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
1850
1900
1950
2000
Year

Trend


Global average temperature anomaly (1850-2012)
Met Office
Met Office Hadley Centre and Climatic Researc
NOAA National Climatic Data Center
NASA Goddard Institute for Space St
Difference from 1961-1990 average (°C)
Year
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
1850
1900
1950
2000

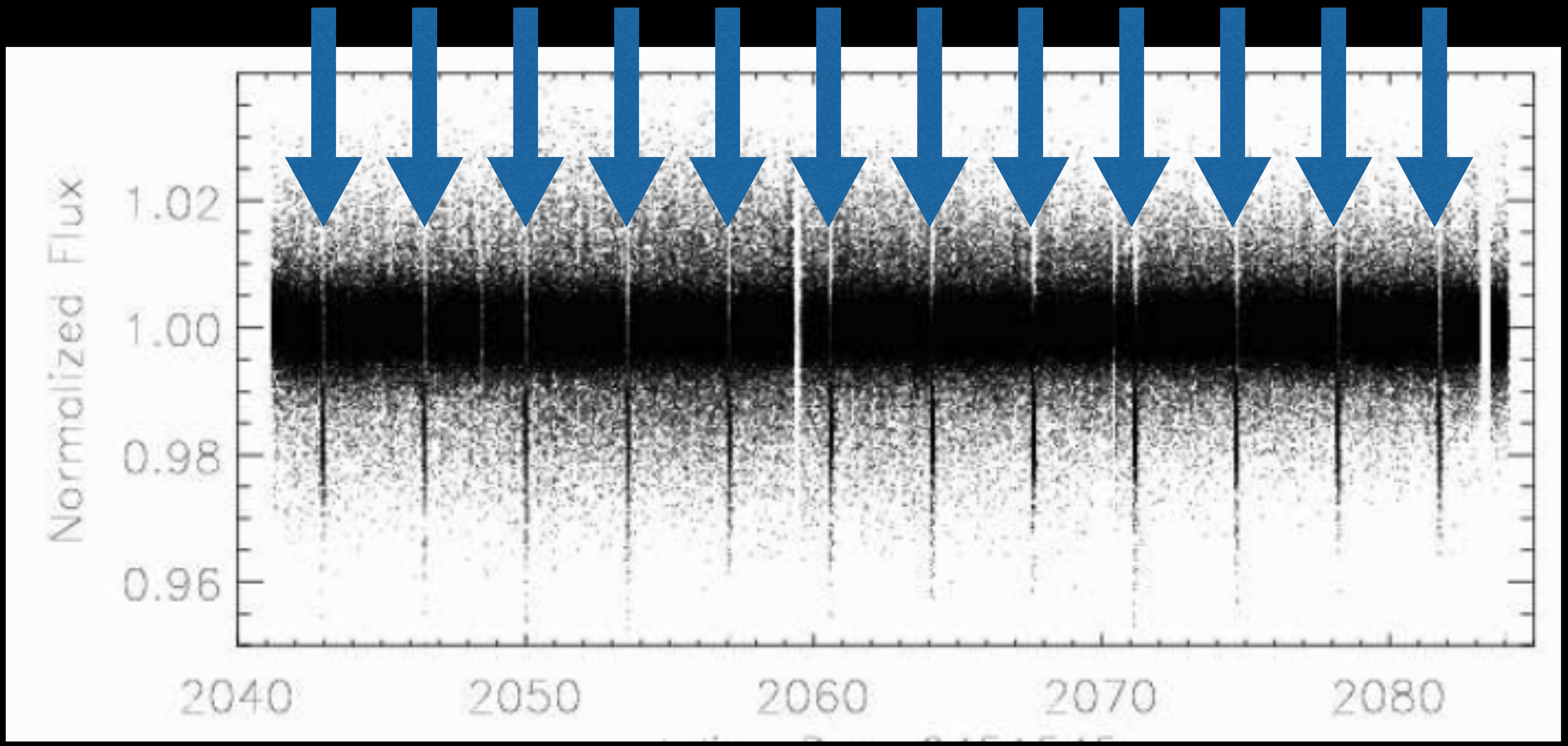
Trends





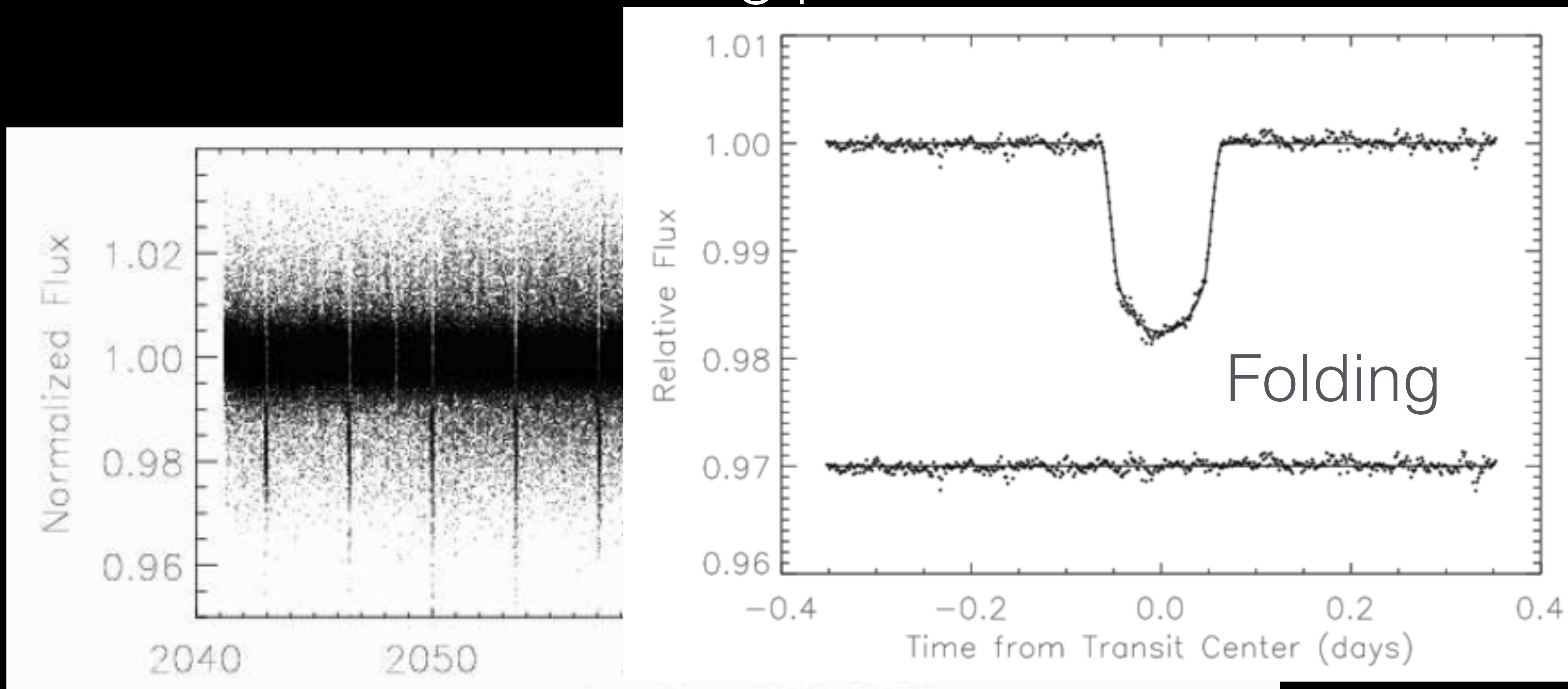
don't forget to vote!

HD 209458, the first transiting planet to be discovered.

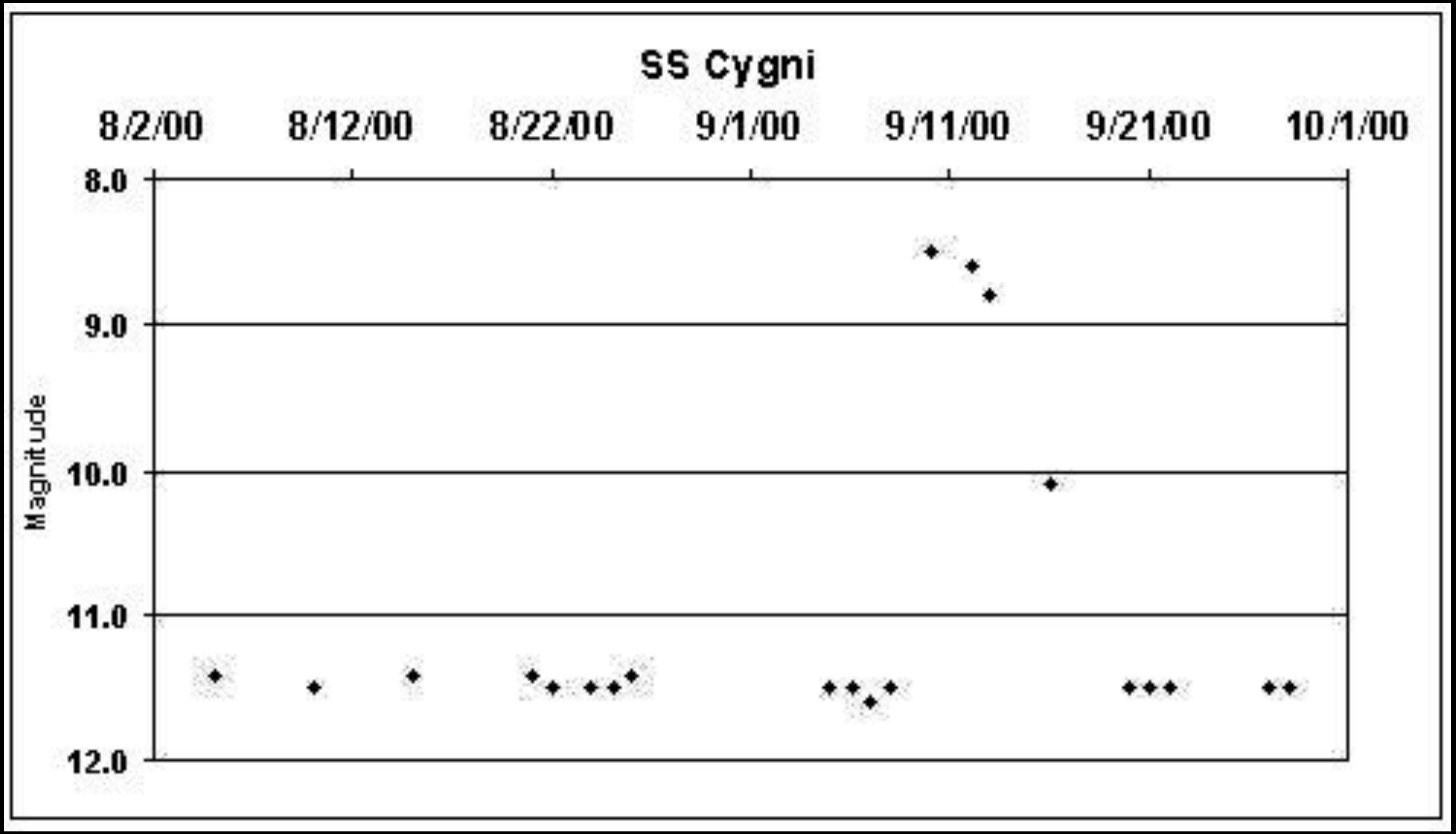


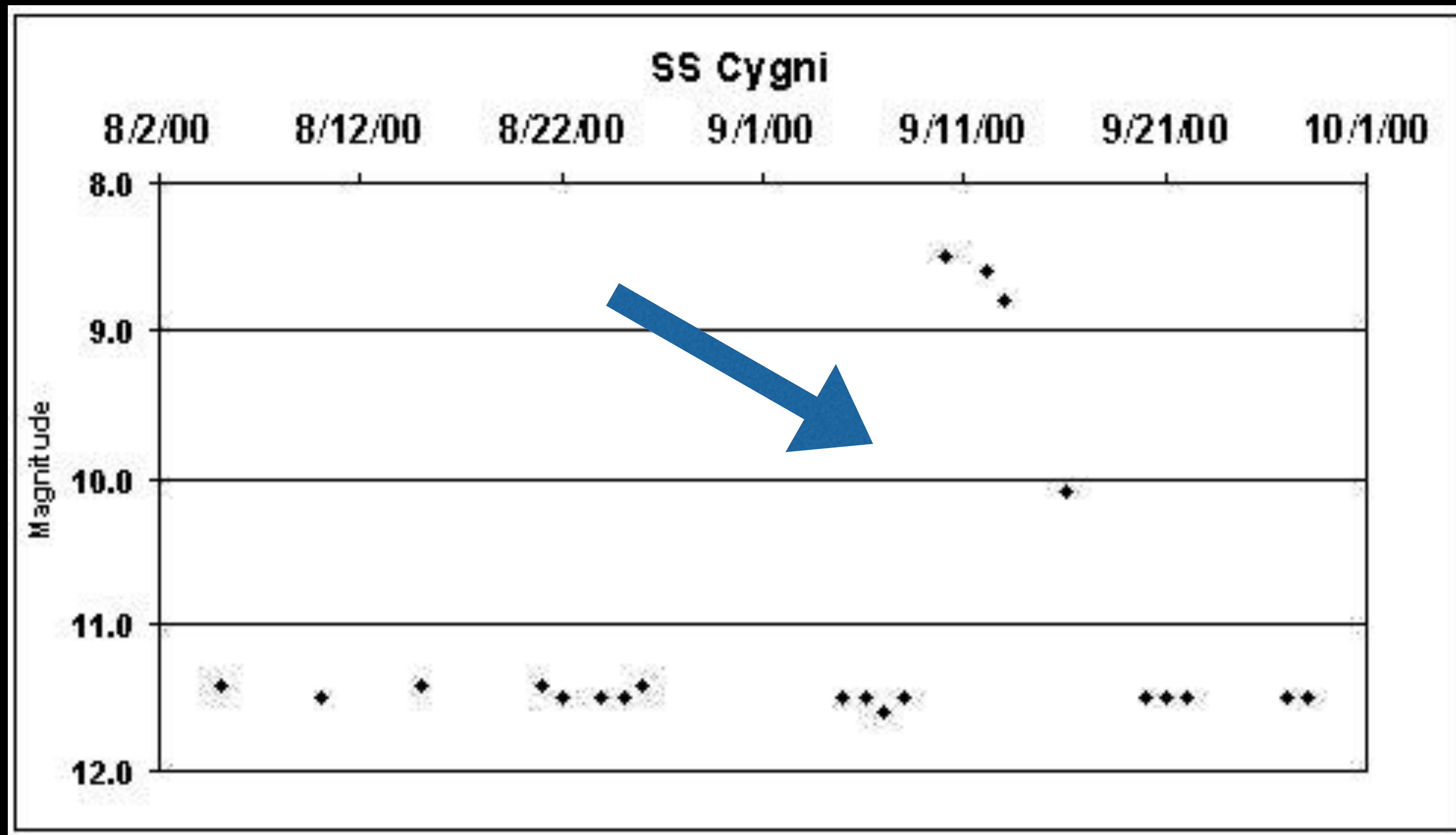
Periodicity

HD 209458, the first transiting planet to be discovered.

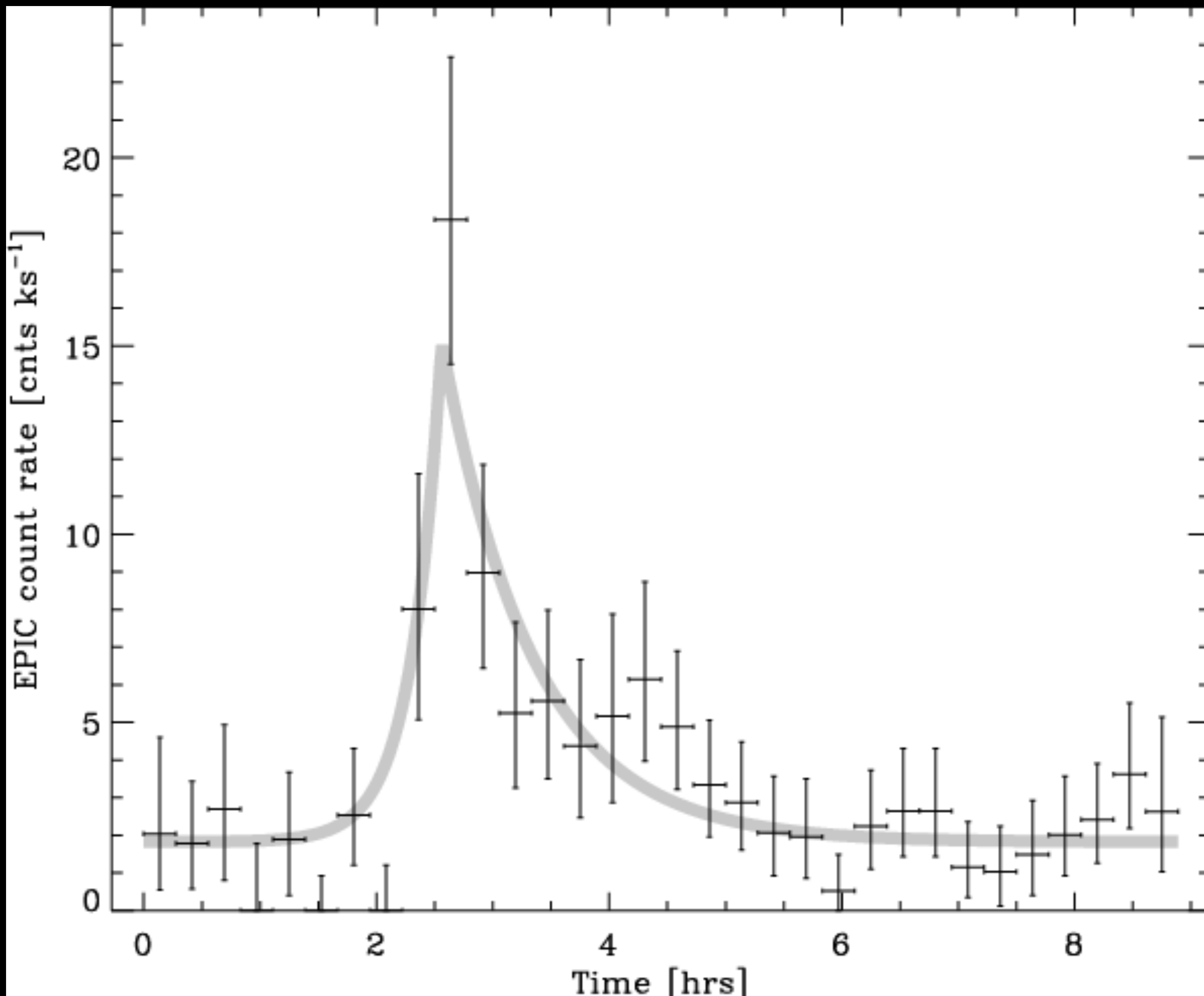


Periodicity

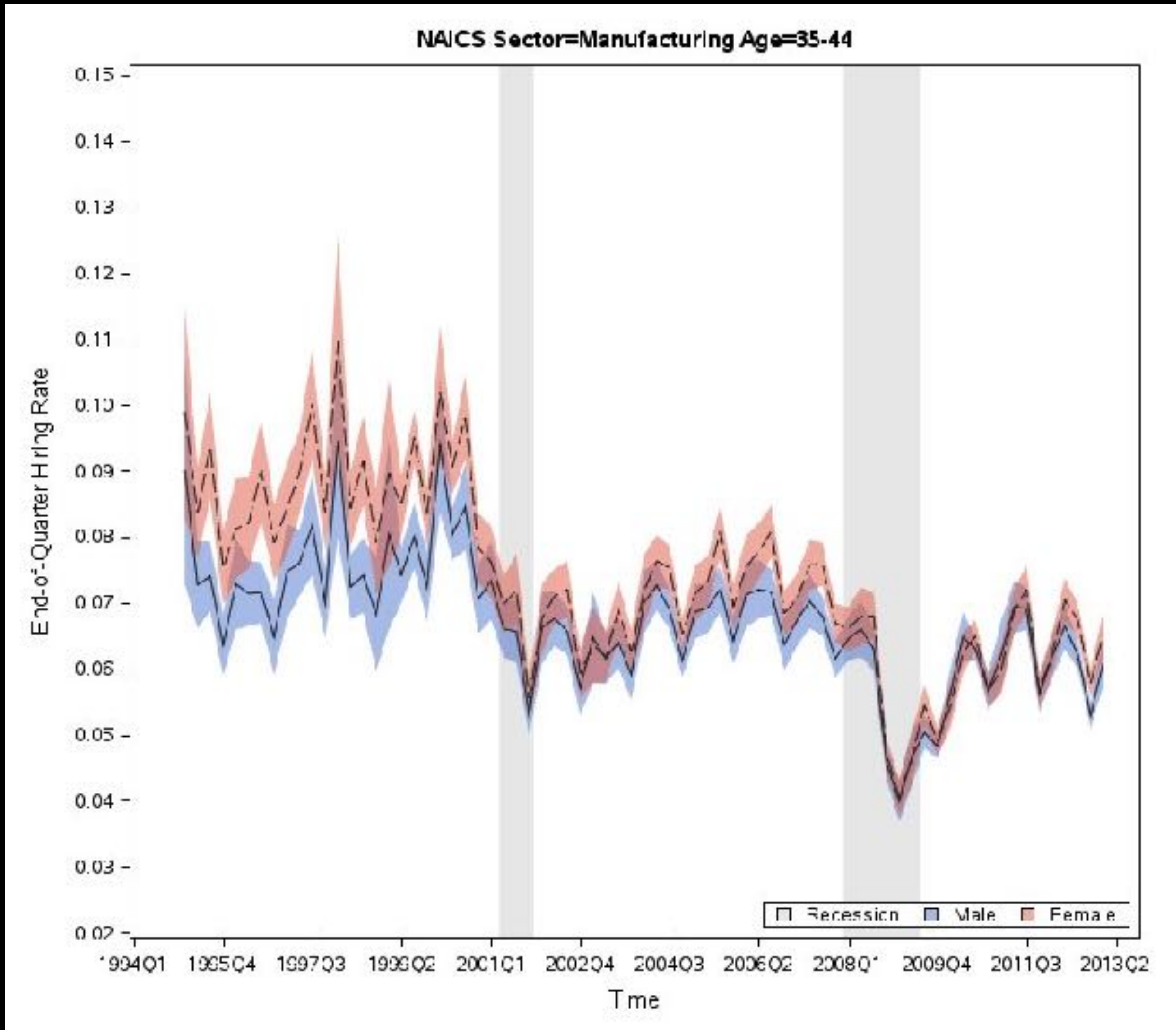

SS Cygni
8/2/00
8/12/00
8/22/00
9/1/00
9/11/00
9/21/00
10/1/00
8.0
9.0
10.0
11.0
12.0
Magnitude



event detection

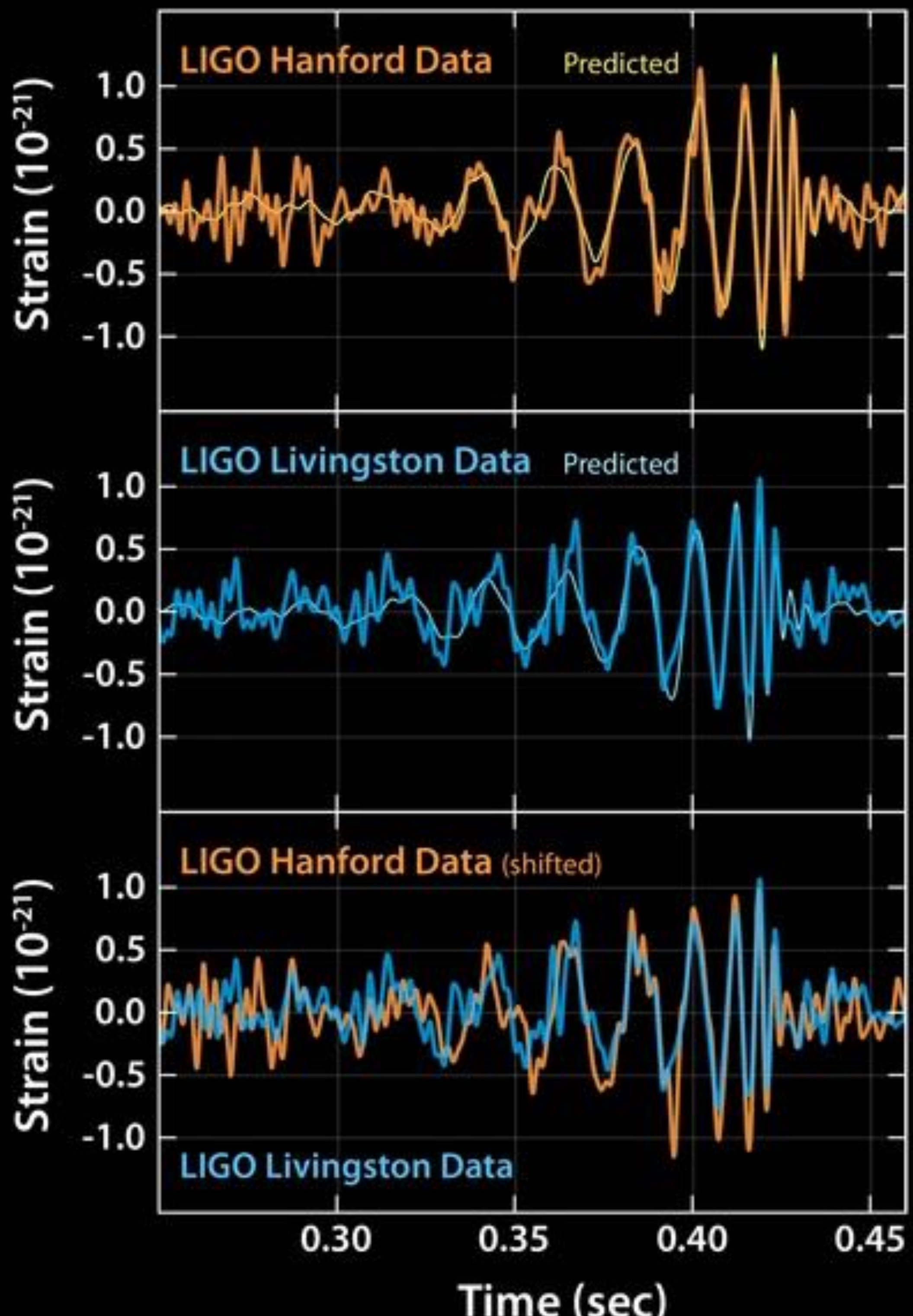


event detection



event detection

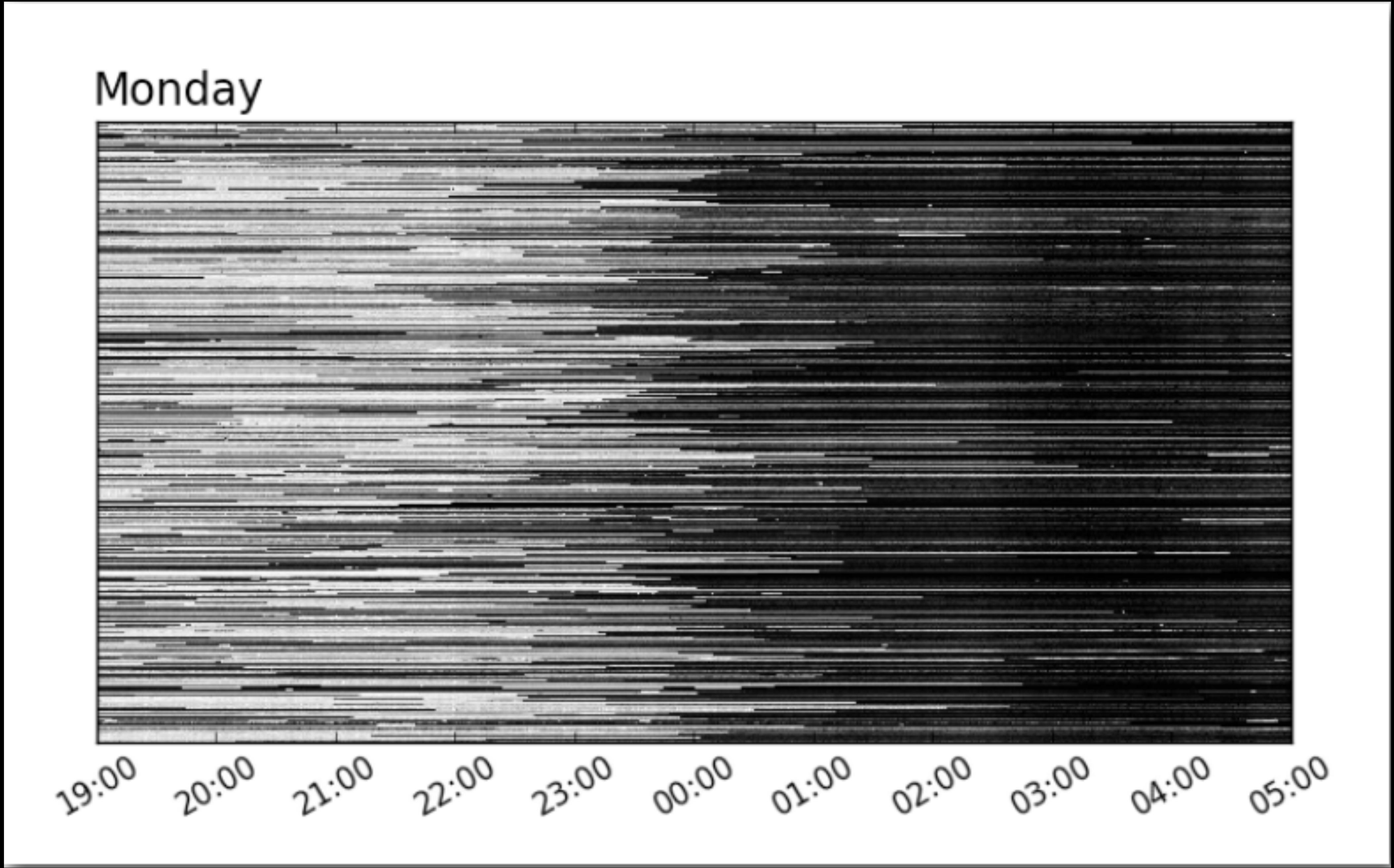
LEHD data (Prof. Julia Lane)



event detection

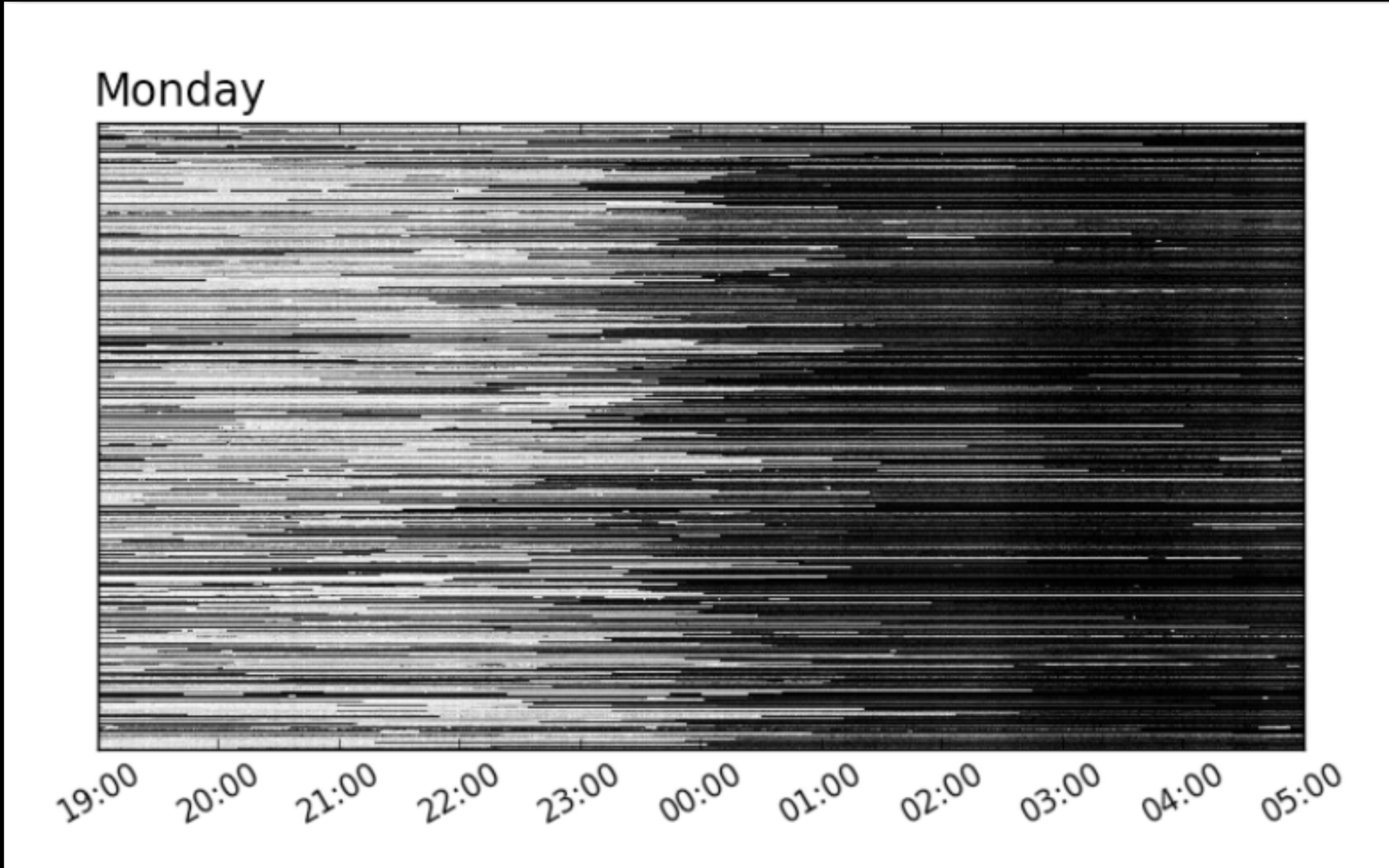
LIGO gravitational wave detection

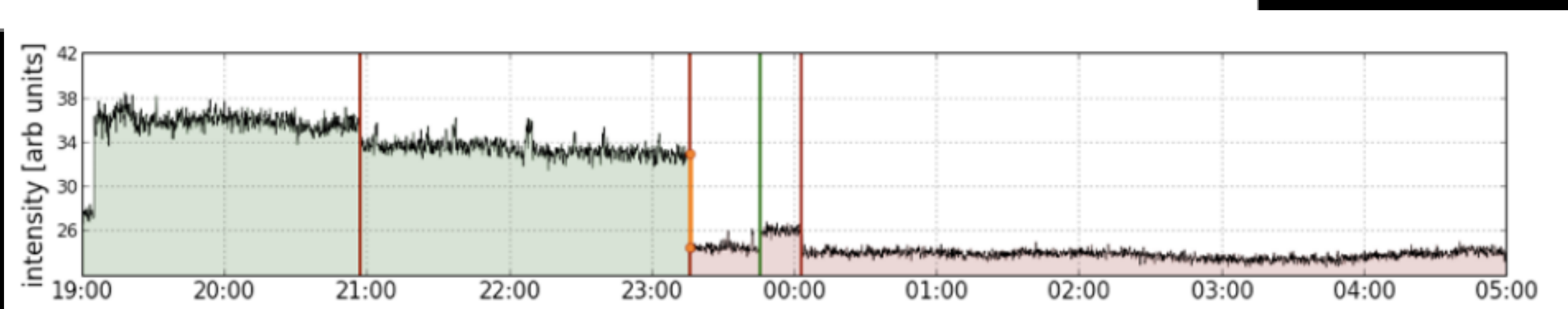
Abbott et al. Physical Review Letters 116, 061102 (2016)

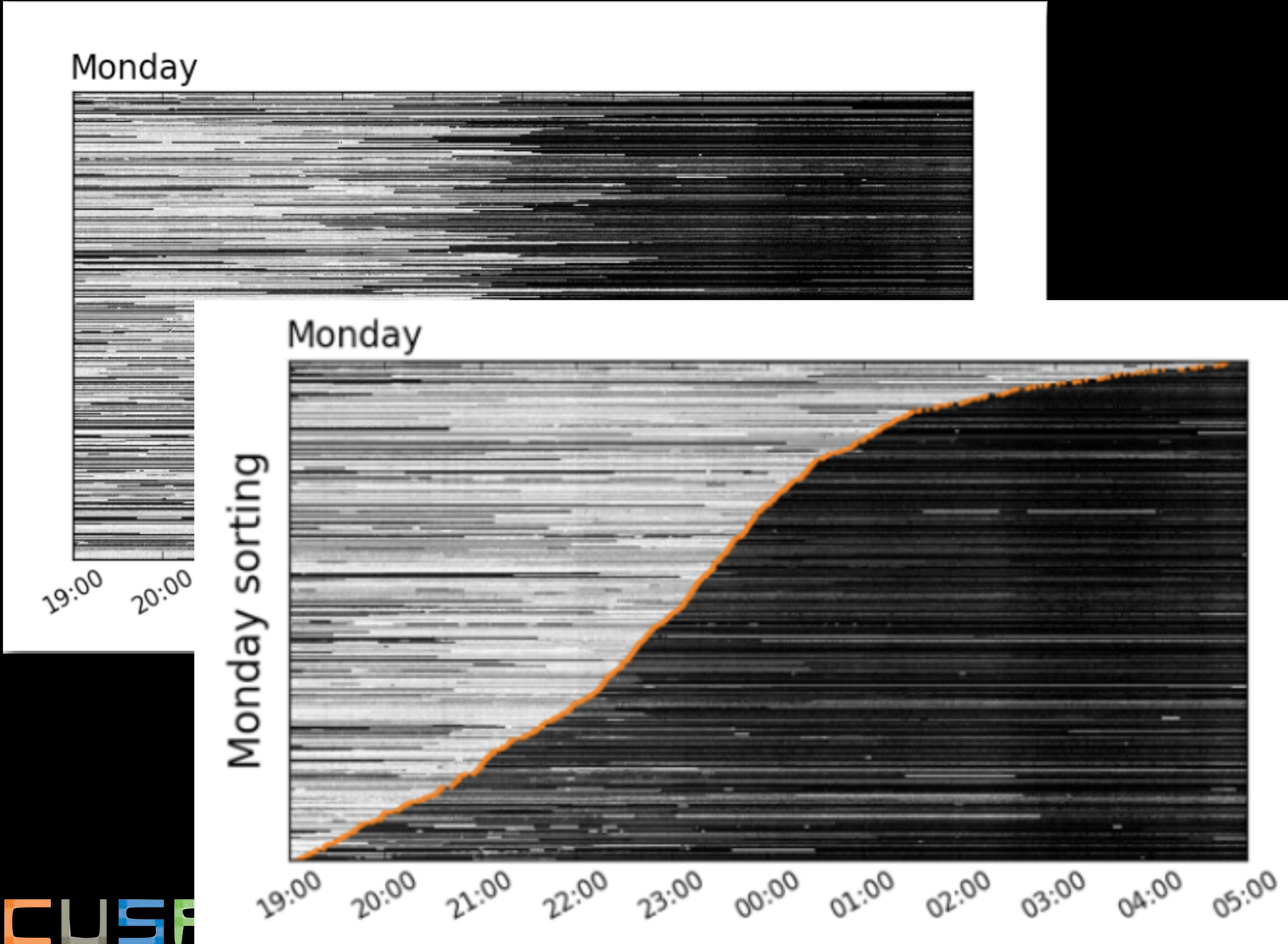


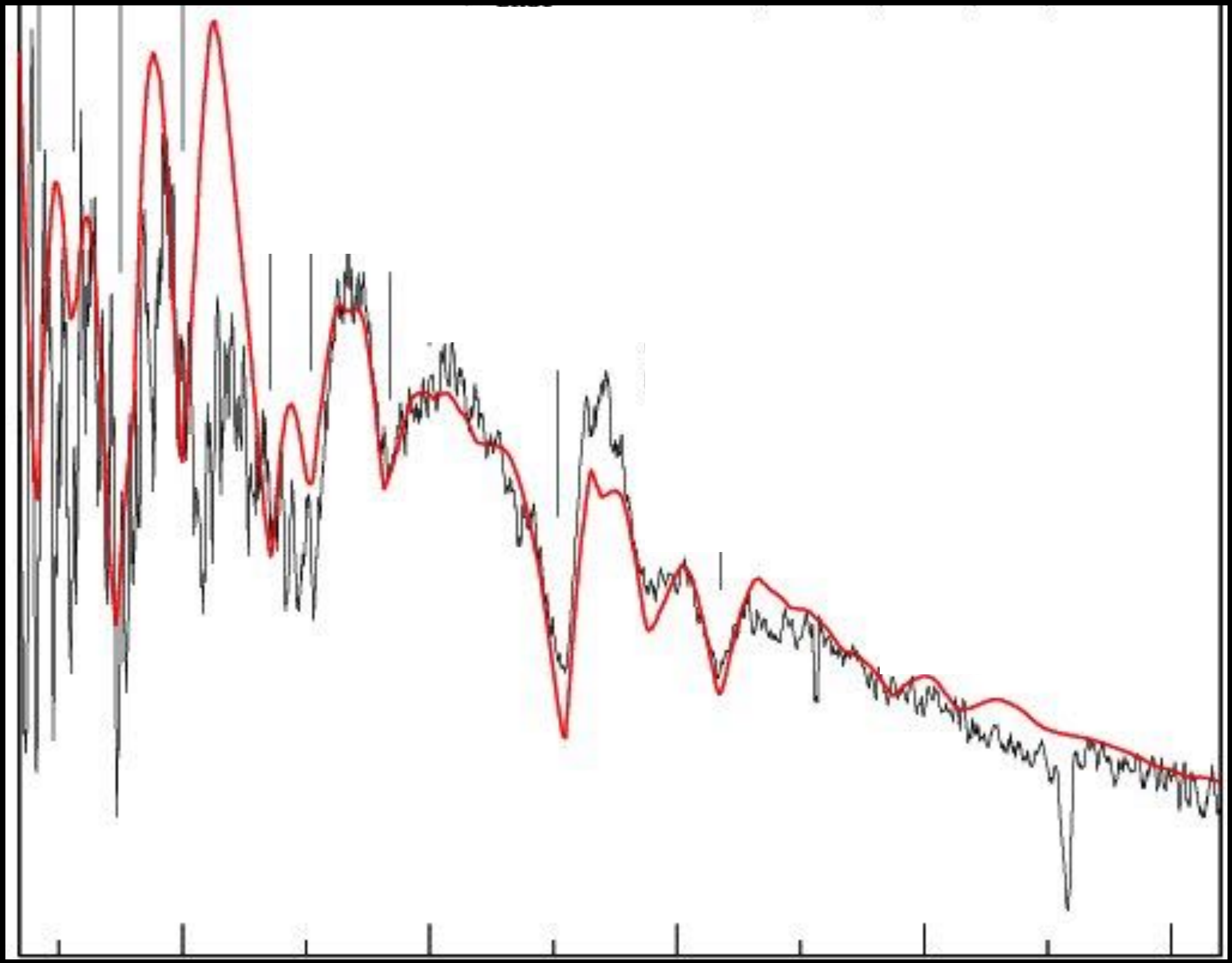
http://www.sciencedirect.com/science/article/pii/S0306437915001167

CUSP-UO

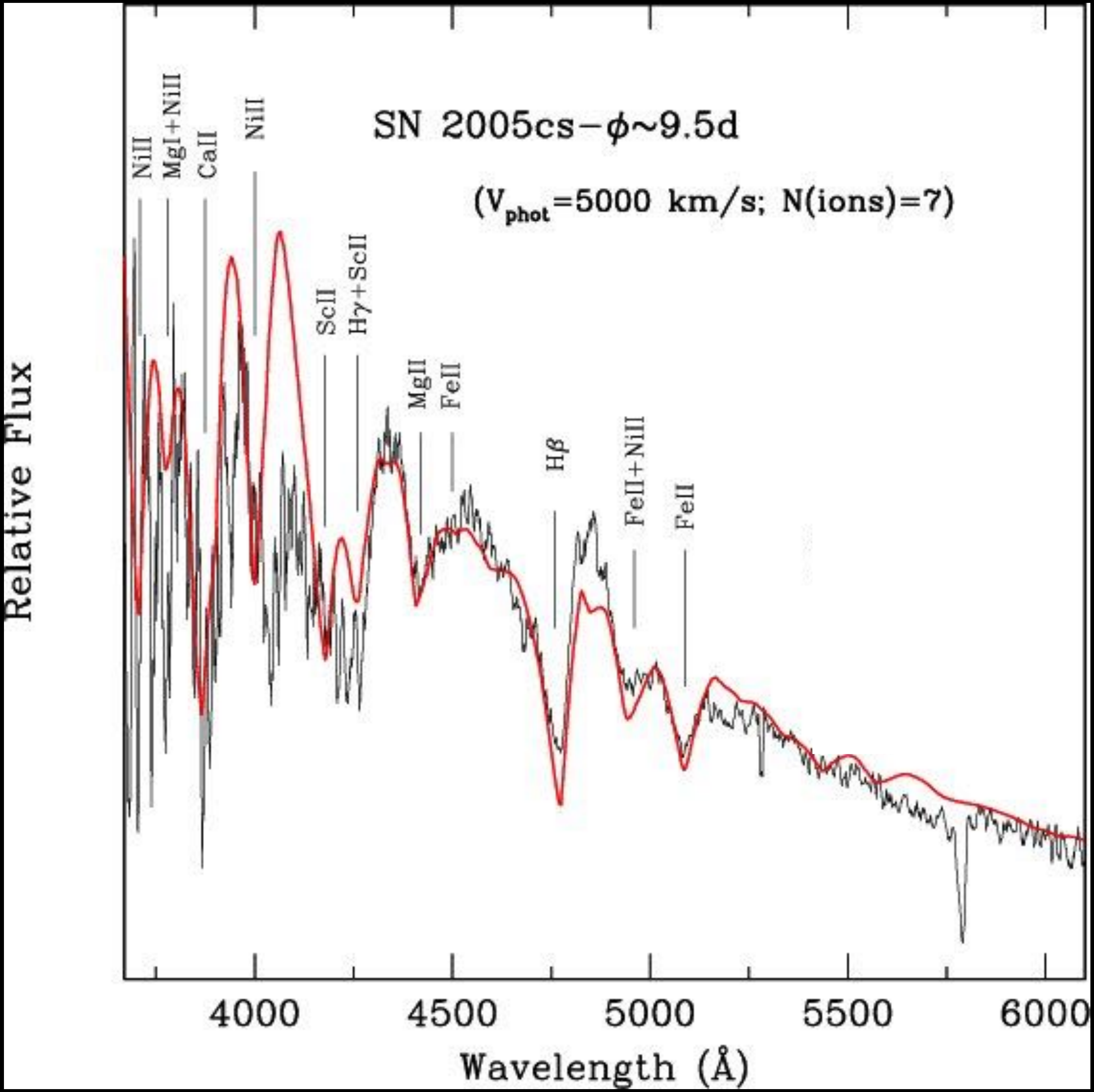

Monday
19:00
20:00
21:00
22:00
23:00
00:00
01:00
02:00
03:00
04:00
05:00


intensity [arb units]
42
38
34
30
26
19:00
20:00
21:00
22:00
23:00
00:00
01:00
02:00
03:00
04:00
05:00

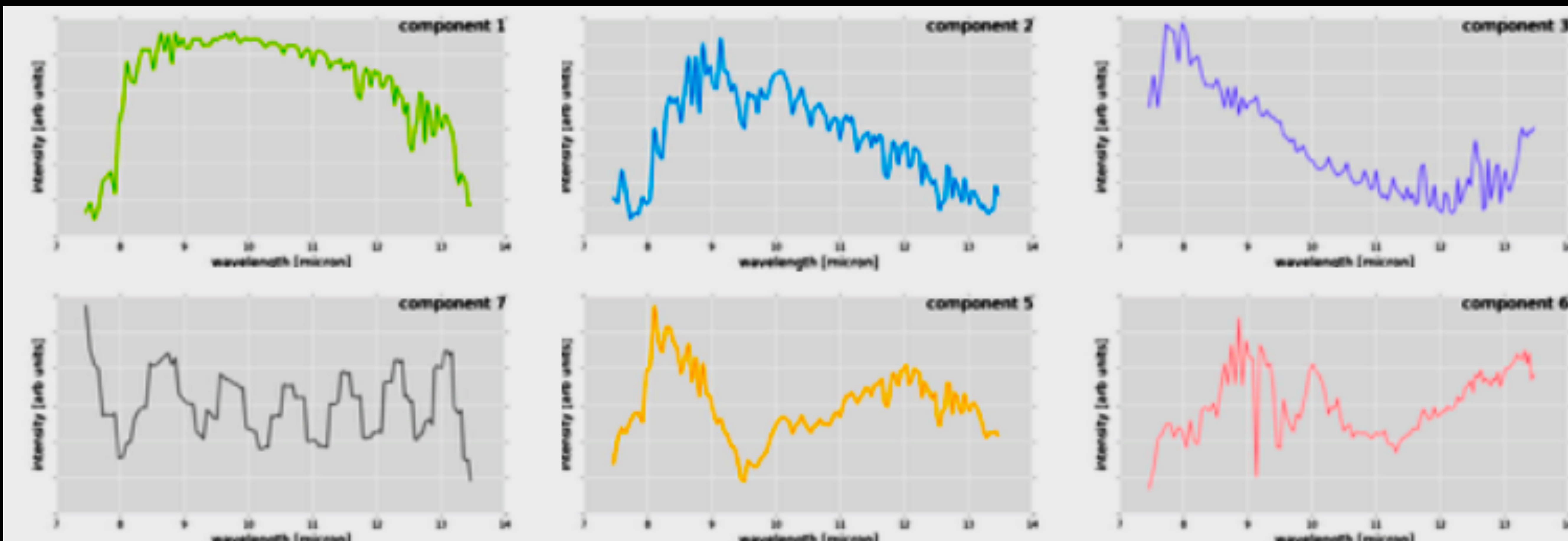

Monday
19:00
20:00
Monday
Monday sorting
19:00
20:00
21:00
22:00
23:00
00:00
01:00
02:00
03:00
04:00
05:00



they do not have to be *TIME* series!



CUSP-UO spectra of urban lights for light technology assessment



- event detection

- event detection
- identification of trends

- event detection
- identification of trends
- periodicity detection

- event detection
- identification of trends
- periodicity detection
- prediction

- event detection
- identification of trends
- periodicity detection
- prediction
- classification (clustering)

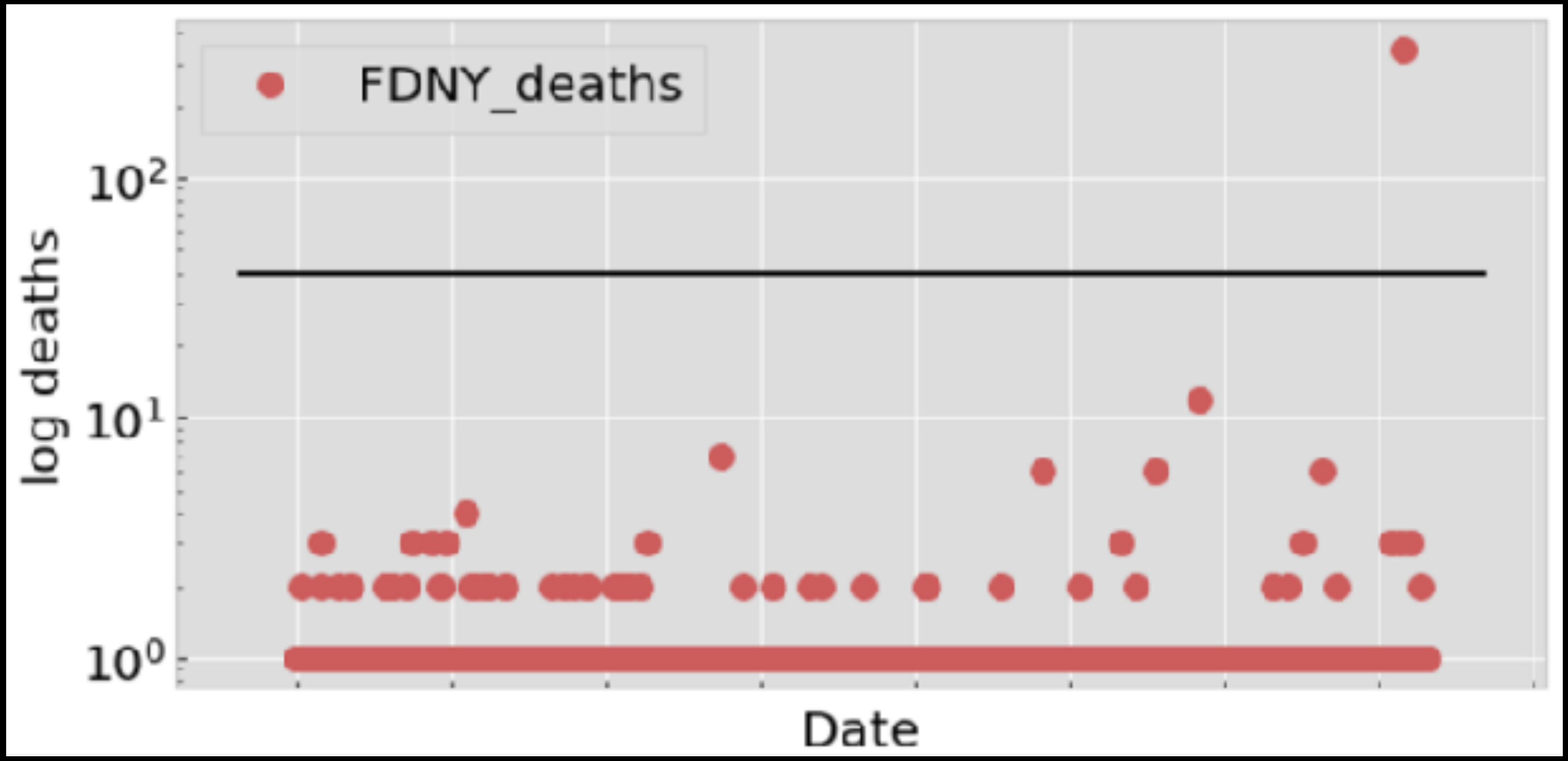
- event detection

Thresholding

jupyter

- event detection

Thresholding



https://github.com/fedhere/UInotebooks/blob/master/FDNYdeaths.ipynb

- event detection

Thresholding

jupyter

- take the mean (possibly a local mean)
- take the standard deviation (possibly a local stdev)
- find points that deviate from the mean by more than N standard deviation

https://github.com/fedhere/UInotebooks/blob/master/FDNYdeaths.ipynb

CUSP

- event detection

Point of change

jupyter

https://github.com/fedhere/UInotebooks/blob/master/timeseries/pointOfChange.ipynb

CUSP

- event detection
- identification of trends

Stationary data
Smoothing (Rolling mean)
ADFuller test for unit root (for non-stationarity)

jupyter

https://github.com/fedhere/UInotebooks/blob/master/timeseries/stationarity_macroeconomicData.ipynb

https://github.com/fedhere/UInotebooks/blob/master/timeseries/stationarity_syntheticData.ipynb

CUSP

- event detection
- identification of trends
- periodicity detection

ARMA/ARIMA

jupyter

http://www.statsref.com/HTML/index.html?arima.html

http://www.econ.ohio-state.edu/dejong/note2.pdf

CUSP

ARIMA

Autoregression

$$x(t) = a_1 x(t-1) + \epsilon_t$$

ARIMA

Autoregression

$$x(t)=a_1x(t-1)+\epsilon_t$$

$$x(t)=a_1x(t-1)+a_2x(t-2)+\ldots+a_nx(t-n)+\epsilon_t$$

Integration

$$x'(t) = x(t) - x(t-i)$$

ARIMA

Autoregression

$$x(t) = \sum_{i=1}^{p} a_i x_{t-i} + \varepsilon_t$$

Moving Average Model

$$x(t) = \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} + \varepsilon_t$$

ARIMA

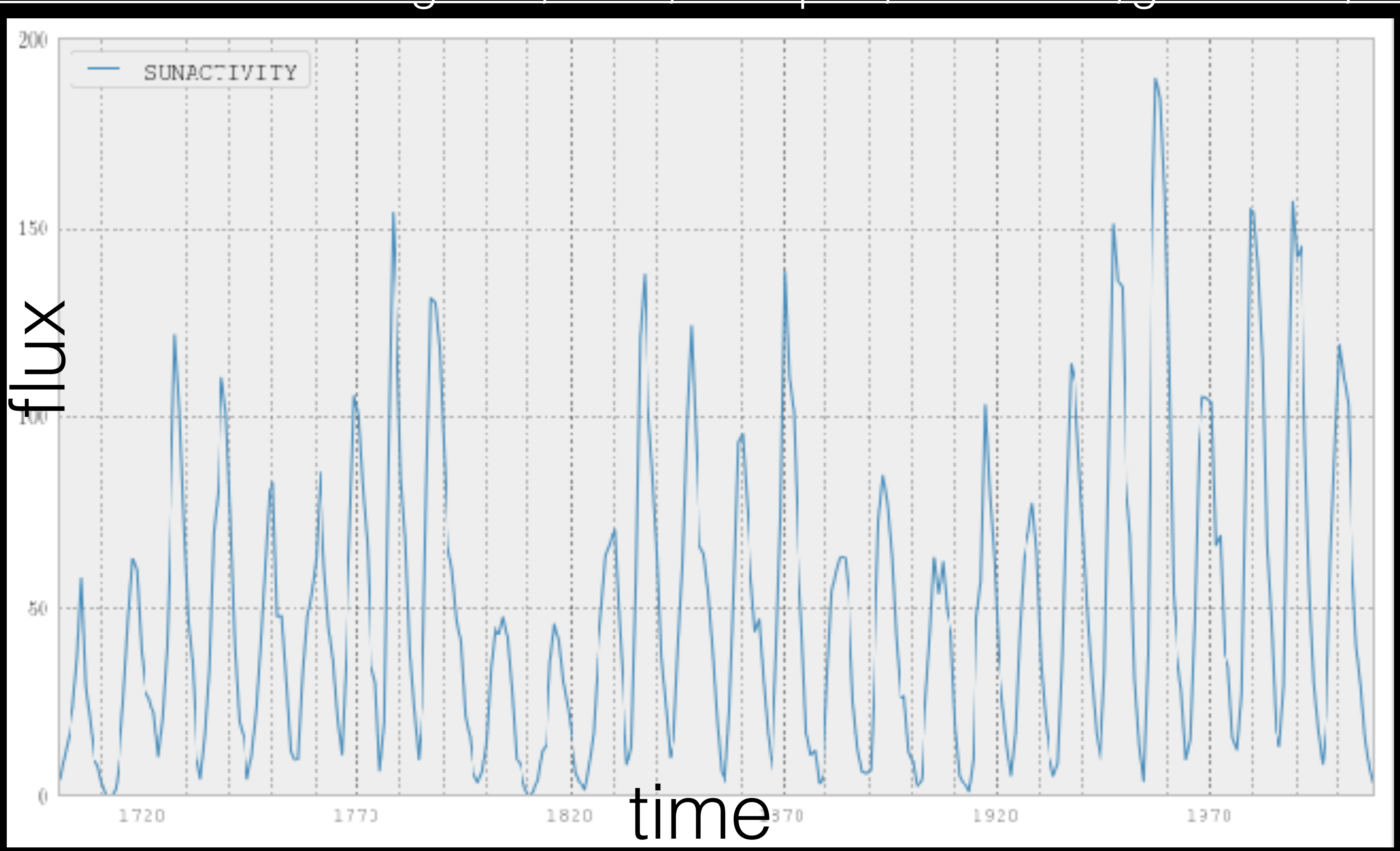
Autoregression

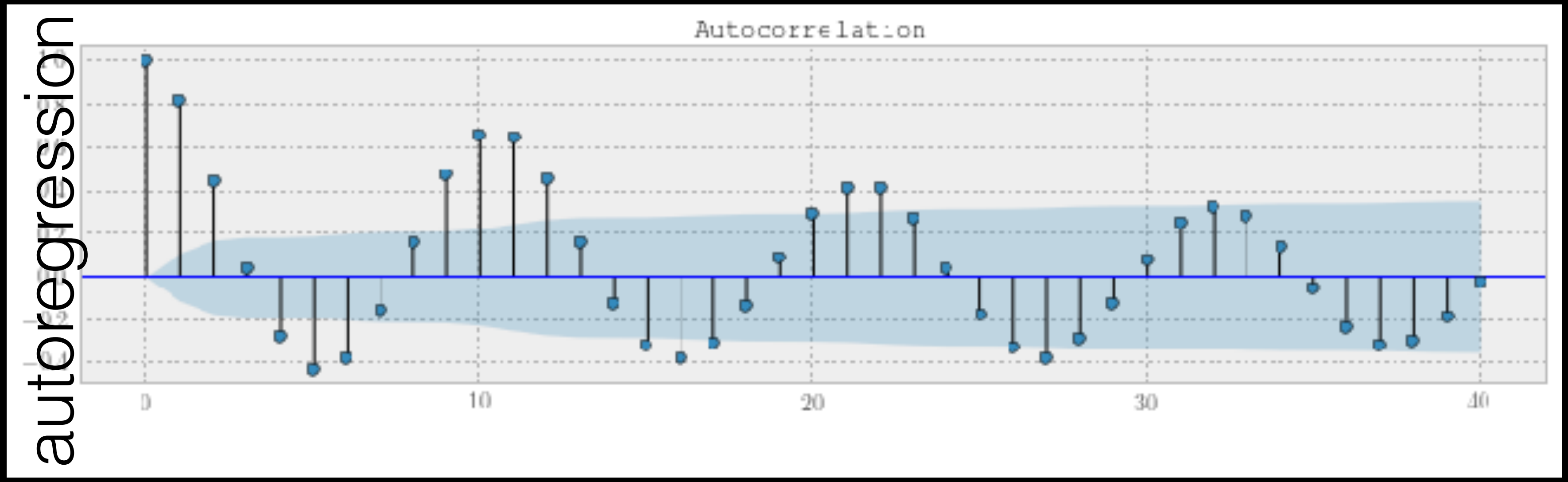
$$x(t)=\sum_{i=1}^{p} a_i x_{t-i} + \varepsilon_t$$

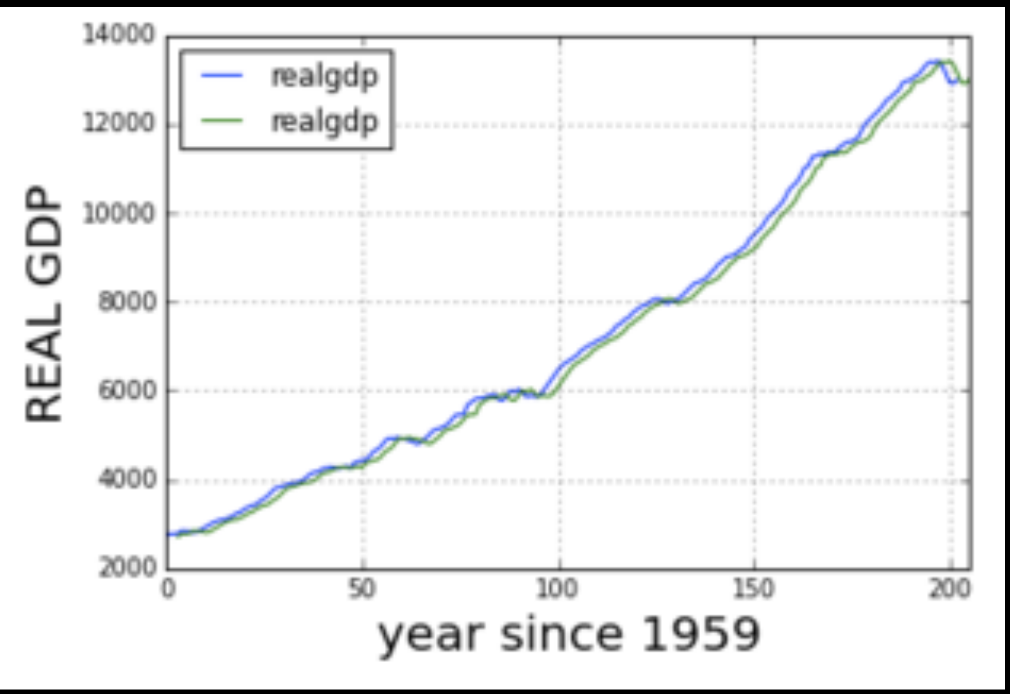
Moving Average Model

$$x(t)=\sum_{i=1}^{q} \theta_i \varepsilon_{t-i} + \varepsilon_t$$

http://statsmodels.sourceforge.net/devel/examples/notebooks/generated/tsa_arma_0.html



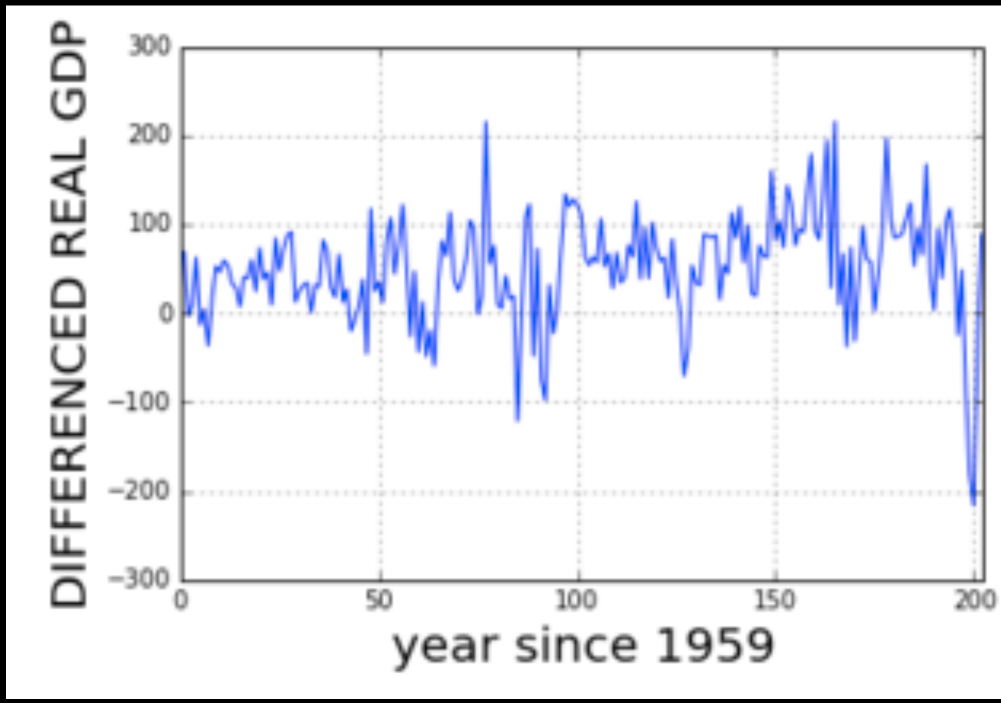




Integration

$$x'(t)=x(t)-x(t-i)$$

ARIMA



Autoregression

$$x(t)=\sum_{i=1}^{p} a_i x_{t-i}+\varepsilon_t$$

Moving Average Model

$$x(t)=\sum_{i=1}^{q} \theta_i \varepsilon_{t-i}+\varepsilon_t + \mu$$

jupyter

https://github.com/fedhere/UInotebooks/blob/master/ARMA_microdata.ipynb

CUSP

Key points:

- Time series analysis may be done for a number of purposes: classification, prediction, event detection, period finding
- smoothing, binning, detrending (difference, regression)
- prediction tools: autoregression, ARMA, ARIMA

Homework:

Technical reading on SM time analysis tools. Get through ARMA

http://conference.scipy.org/proceedings/scipy2011/pdfs/statsmodels.pdf

Reading: an excellent analysis of time series
by Jake Vander Plas
(UW e-science center)

https://jakevdp.github.io/blog/2014/06/10/is-seattle-really-seeing-an-uptick-in-cycling/

Homework:

Data:

MTA subway fares. It is a complete dataset of rides logged by card swipes for 600 Manhattan stations.

It contains 23 different subway card types (e.g. monthly pass, daily pass, Act for Disability pass… i will give you this as a list)

Each time series (per station, per ticket type) contains the number of swipes per week for 194 weeks from 05/21/2010 to 02/21/2014.

it is given to you as a python data cube.
you can load it as np.load("MTA_Fare.npy") and you will end up with a python numpy array of shape (600,23,194)

Homework:

Goal 1:

Event detection: Identify the most prominent event. There is a very significant drop (>3-sigma) in *all* time series.
Identify it and figure out what it is due to.

Goal 2:

Some of the time series are stationary, some show a downward trend: Identify the time series with the most prominent downward trend.

Goal 3:

Build a classified that assigns a card type to a time series based on time series features

Homework Hints:

Goal 1:

Some of the time series are stationary, some show a downward trend: Identify the time series with the most prominent downward trend.

work with all time series individually. you can use the rolling mean to find trends: compare rolling mean near beginning and end of time series.

Goal 2:

Event detection: Identify the most prominent event. There is a very significant drop (>3-sigma) in *all* time series.
Identify it and figure out what it is due to.

Since I am telling you the event is in all time series you can work with averages: for example average over all rise types per station. Since i am telling you it is a highly significant event you can find it by thresholding

CUSP

Homework Hints:

Goal 3:

Build a classified that assigns a card type to a time series based on time series features

- Clean the data from missing values (drop time series with NaNs)
- Used all the time series, the ticket type as a label.
- Calculate the mean, standard deviation, and by station and use t following features:

1,2 line fit coefficient to the reduced time series
(time series - mean_by_station)/ stdev_of_station
3 mean_of_station
4 stdev_of_station

- Split the training and test data
- Build and test a random forest model that predicts the ticket typ based on these 4 features.
- Build and test a random forest model that predicts the ticket typ based on all datapoint in the time series (194 features)
- Plot a confusion matrix for each model (discuss)
- Compare the models w a sklearn.metrics classification_report
- Find the 2 most important features in each model

References ok 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/

dont just do linear regression!

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

X: decision trees

References ok Decision trees:

Statistical Analysis Handbook
http://www.statsref.com/HTML/index.html

Stationary and non stationary time series
http://www.cas.usf.edu/~cconnor/geolsoc/html/chapter11.pdf

ARMA & ARIMA
http://www.econ.ohio-state.edu/dejong/note2.pdf

Time series classification in python, not covered but you should read about it!
http://alexminnaar.com/time-series-classification-and-clustering-with-python.html

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