

AN INTRODUCTION TO DEEP LEARNING FOR ASTRONOMY

Marc Huertas-Company

EUCLID Summer School 2018



institut
universitaire
de France

REFERENCES

SEVERAL SLIDES / INFOS SHOWN HERE ARE INSPIRED/
TAKEN FROM OTHER WORKS / COURSES FOUND ONLINE

- Deep Learning: Do-It-Yourself! [Bursuc, Krzakala, Lelarge]
- DEEPMLEARNING.AI [COURSERA, Ng, Bensouda, Katanforoosh]
- MACHINE LEARNING LECTURES [Keck]
- EPFL DEEP LEARNING COURSE [Fleuret]

Thanks to all of them!

SOME PRELIMINARY NOTES

I AM NOT A MACHINE LEARNING RESEARCHER

SOME PRELIMINARY NOTES

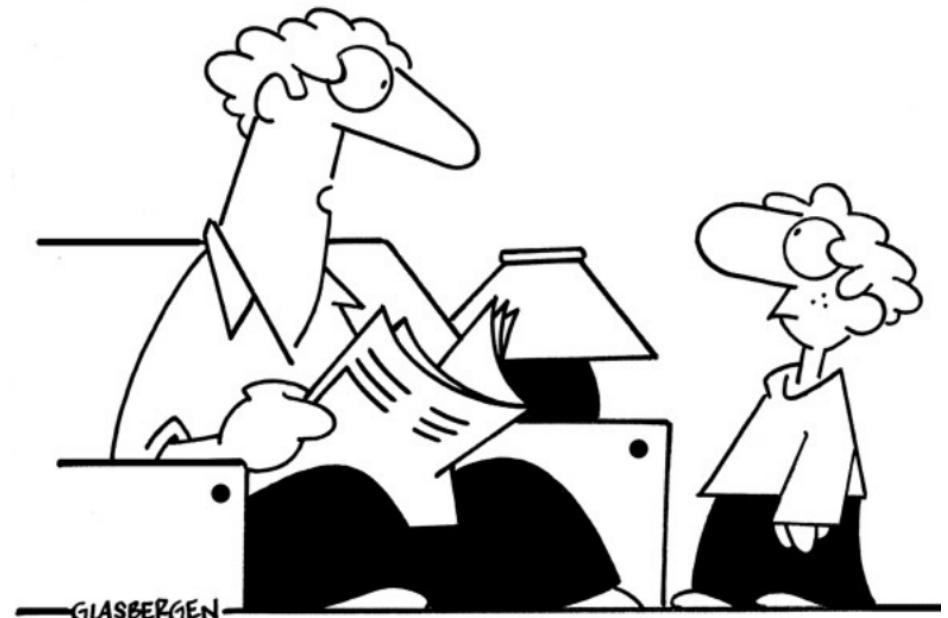
I AM NOT A MACHINE LEARNING RESEARCHER

ONLY AN ASTRONOMER WHO HAS BEEN USING MACHINE
LEARNING FOR THE LAST ~13 YEARS FOR MY RESEARCH

THIS LECTURE IS INTENDED TO PROVIDE A GLOBAL
UNDERSTANDING OF HOW AI TECHNIQUES WORK AND
ESPECIALLY HOW TO USE THEM FOR YOUR RESEARCH

WHAT ARE WE GOING TO LEARN?

data-science
pattern-recognition
artificial-intelligence
database
data
big-data machine
data-mining
learning
clustering



*"Artificial intelligence is when you get a college degree,
but you're still stupid when you graduate."*

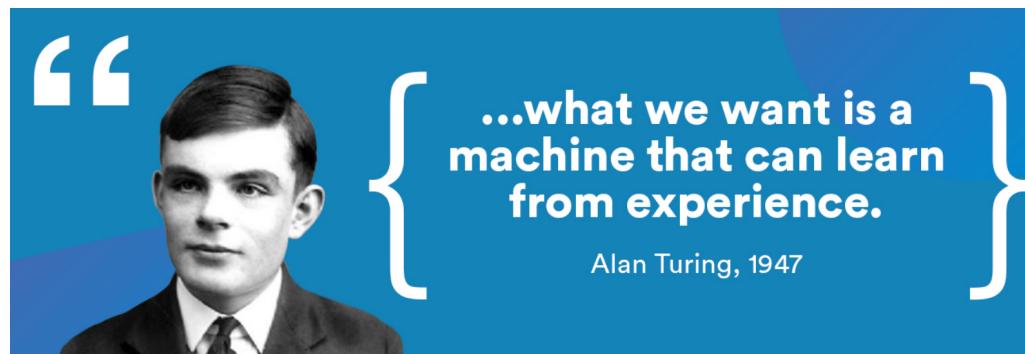
WHAT ARE WE GOING TO LEARN?

data-science
pattern-recognition
artificial-intelligence
database
da
big-data machine learning
data-mining
learning
clustering

A BUNCH OF
SOMETIMES
CONFUSING
TERMS...

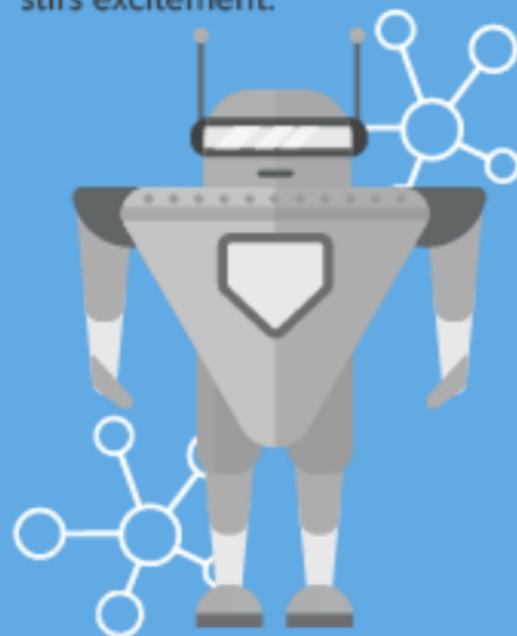


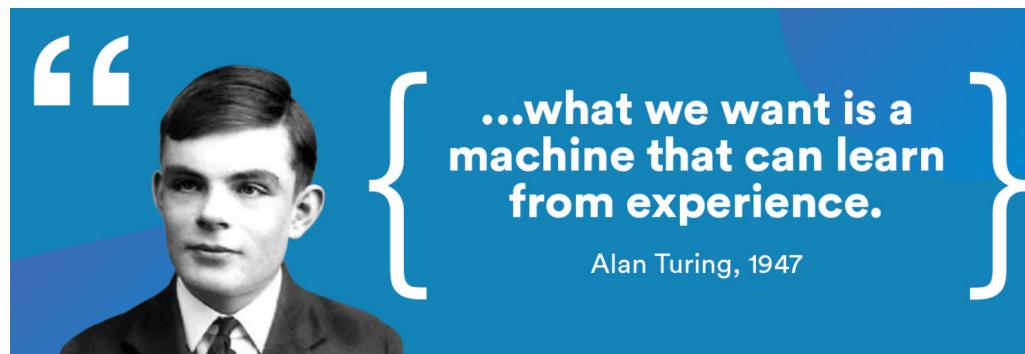
*"Artificial intelligence is when you get a college degree,
but you're still stupid when you graduate."*



ARTIFICIAL INTELLIGENCE

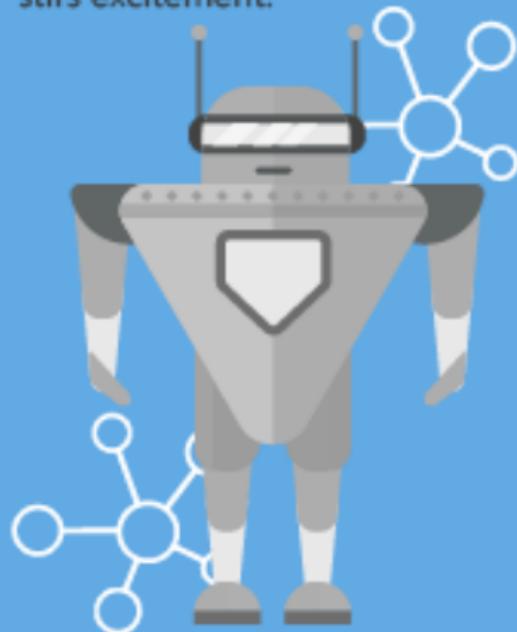
Early artificial intelligence stirs excitement.





ARTIFICIAL INTELLIGENCE

Early artificial intelligence stirs excitement.



1950's

1960's

1970's

1980's

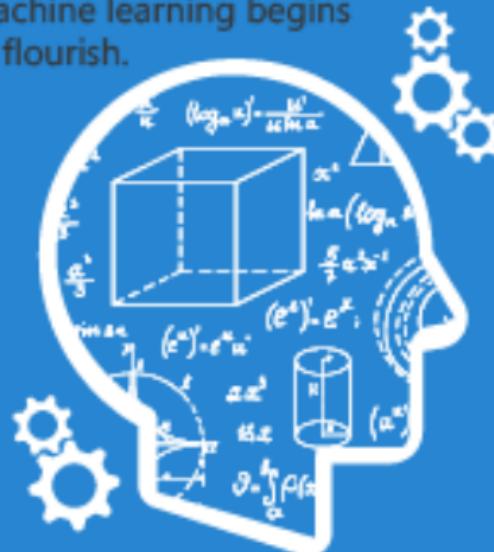
1990's

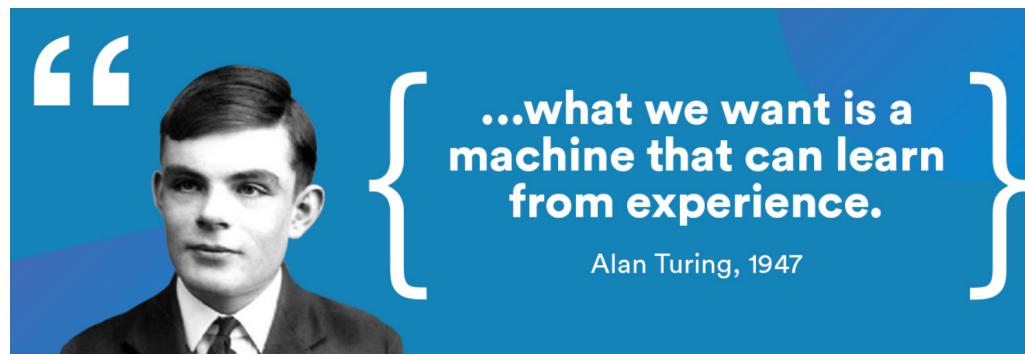
2000's

2010's

MACHINE LEARNING

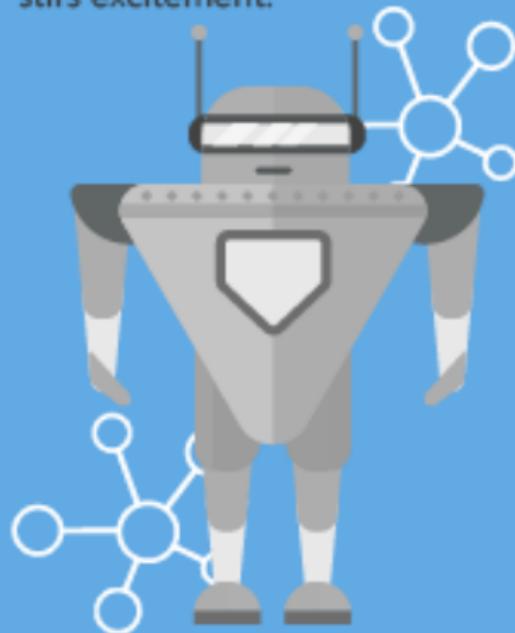
Machine learning begins to flourish.





ARTIFICIAL INTELLIGENCE

Early artificial intelligence stirs excitement.



1950's

1960's

1970's

1980's

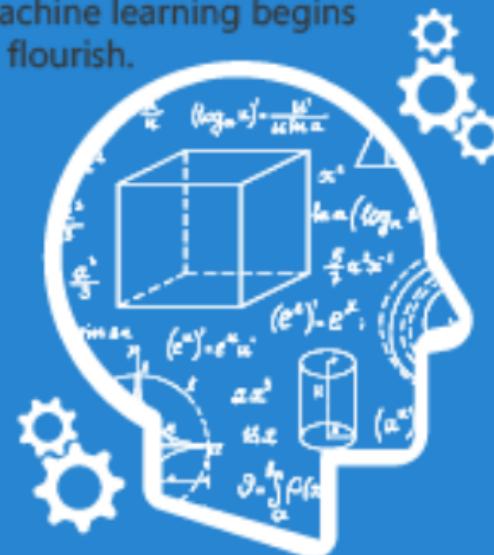
1990's

2000's

2010's

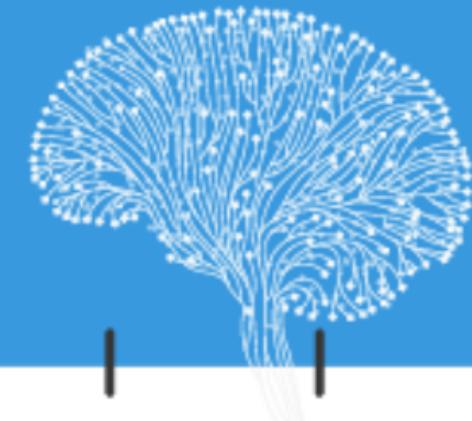
MACHINE LEARNING

Machine learning begins to flourish.

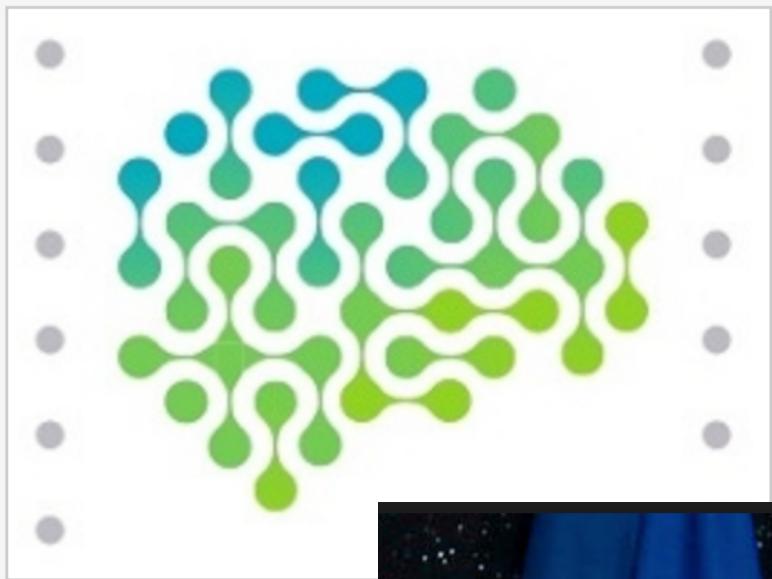


DEEP LEARNING

Deep learning breakthroughs drive AI boom.



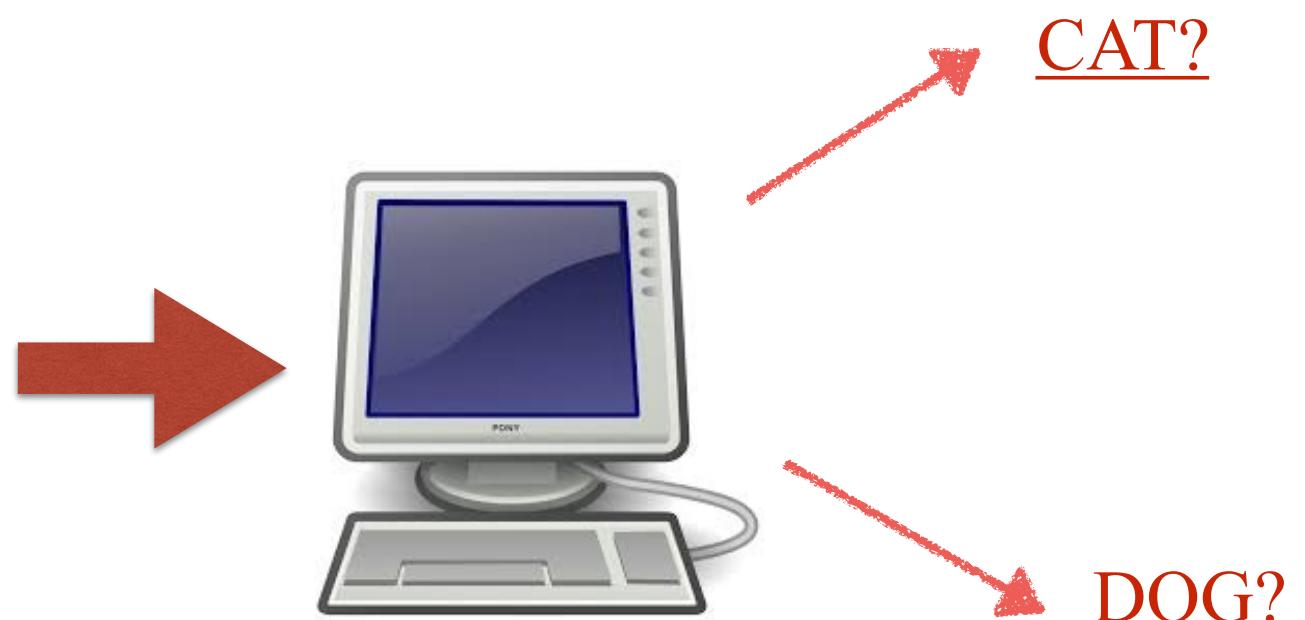
AN AMAZING MEDIA ATTENTION



Le CNRS, Inria, l'université PSL et les entreprises Amazon, Criteo, Facebook, Faurecia, Google, Microsoft, NAVER LABS, Nokia Bell Labs, le Groupe PSA, SUEZ et Valeo font converger intérêts académiques et industriels et s'unissent pour créer, à Paris, l'Institut PRAIRIE dont l'objectif est de devenir une référence internationale de l'intelligence artificielle.

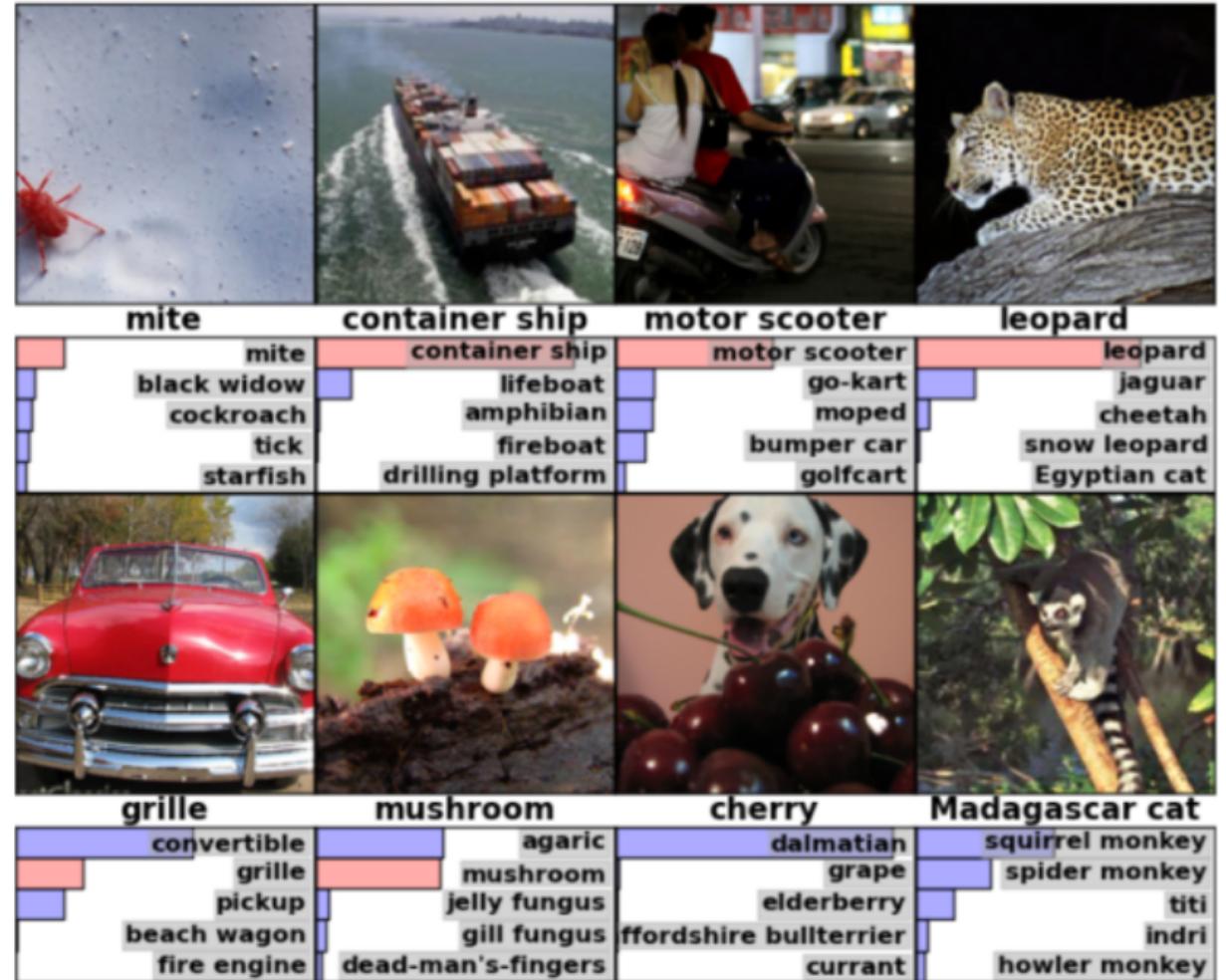
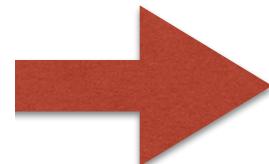


UNTIL 2012....



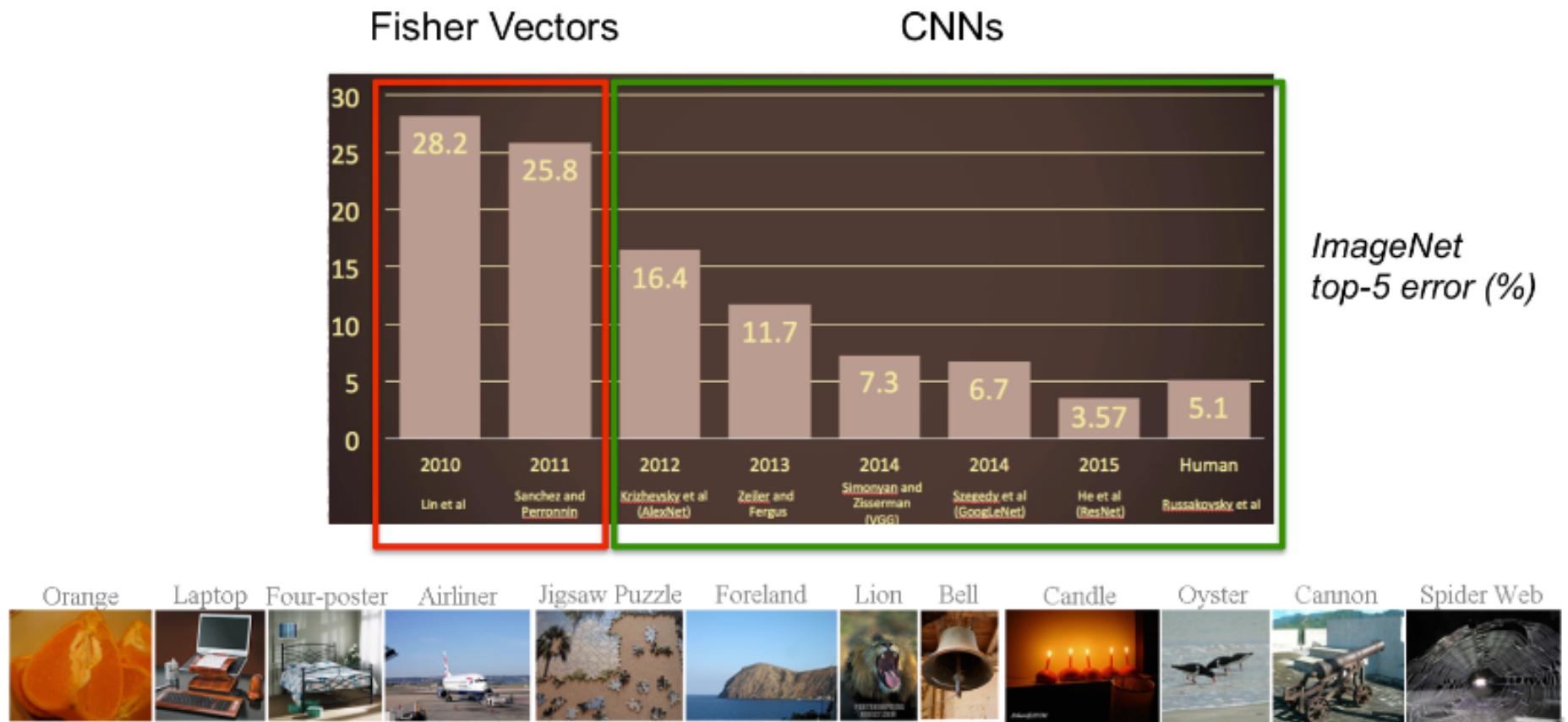
**TRIVIAL HUMAN TASKS REMAINED
CHALLENGING FOR COMPUTERS**

SINCE 2012



IT HAS BECOME TRIVIAL....

THIS IS A CHANGE OF PARADIGM!



ONE OF THE MAIN REASONS OF THIS
BREAKTHROUGH IS THE AVAILABILITY OF VERY
LARGE DATASETS TO LEARN



COMBINED WITH THE TECHNOLOGY TO
PROCESS ALL THIS DATA



ONE OF THE MAIN REASONS OF THIS
BREAKTHROUGH IS THE AVAILABILITY OF VERY
LARGE DATASETS TO LEARN

HOWEVER THERE HAS NOT BEEN A MAJOR
REVOLUTIONARY IDEA



WHAT ARE WE GOING TO LEARN?

BASICS OF CLASSICAL MACHINE LEARNING

BASICS OF DEEP LEARNING

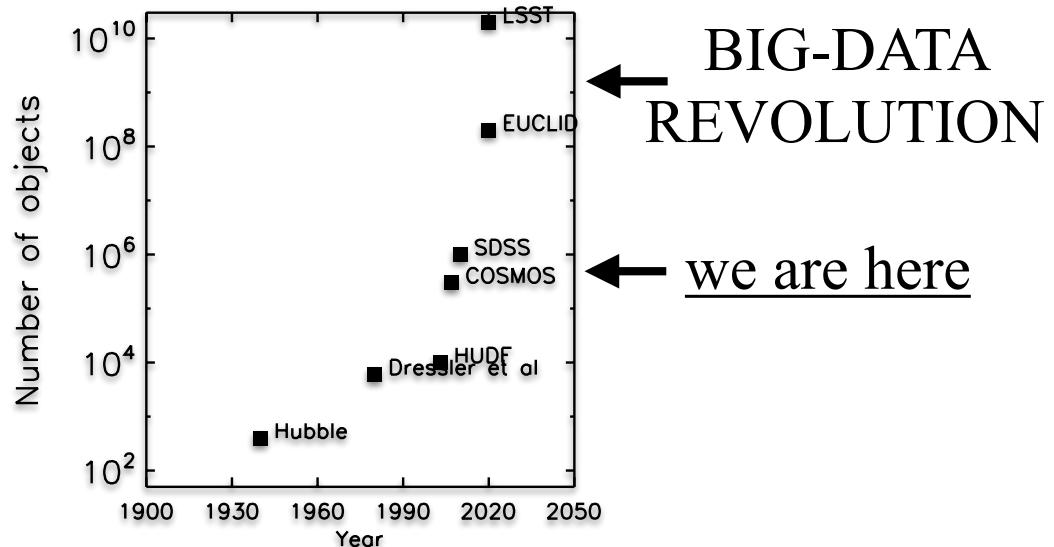
HOPING THAT THIS WOULD BE USEFUL FOR YOUR
RESEARCH!

(Apologies in advance for biases on Extra-Galactic Science +
imaging)

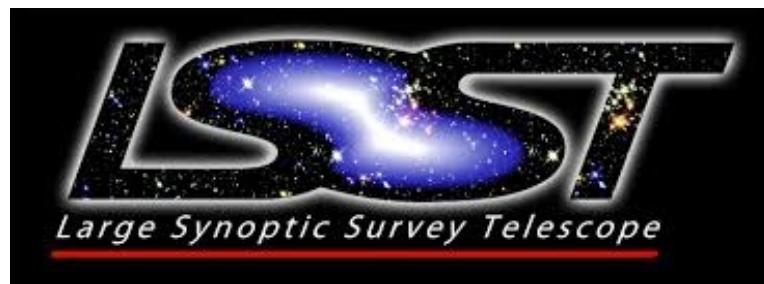
WHY DO WE NEED THESE TOOLS IN ASTRONOMY?

WHY DO WE NEED THESE TOOLS IN ASTRONOMY?

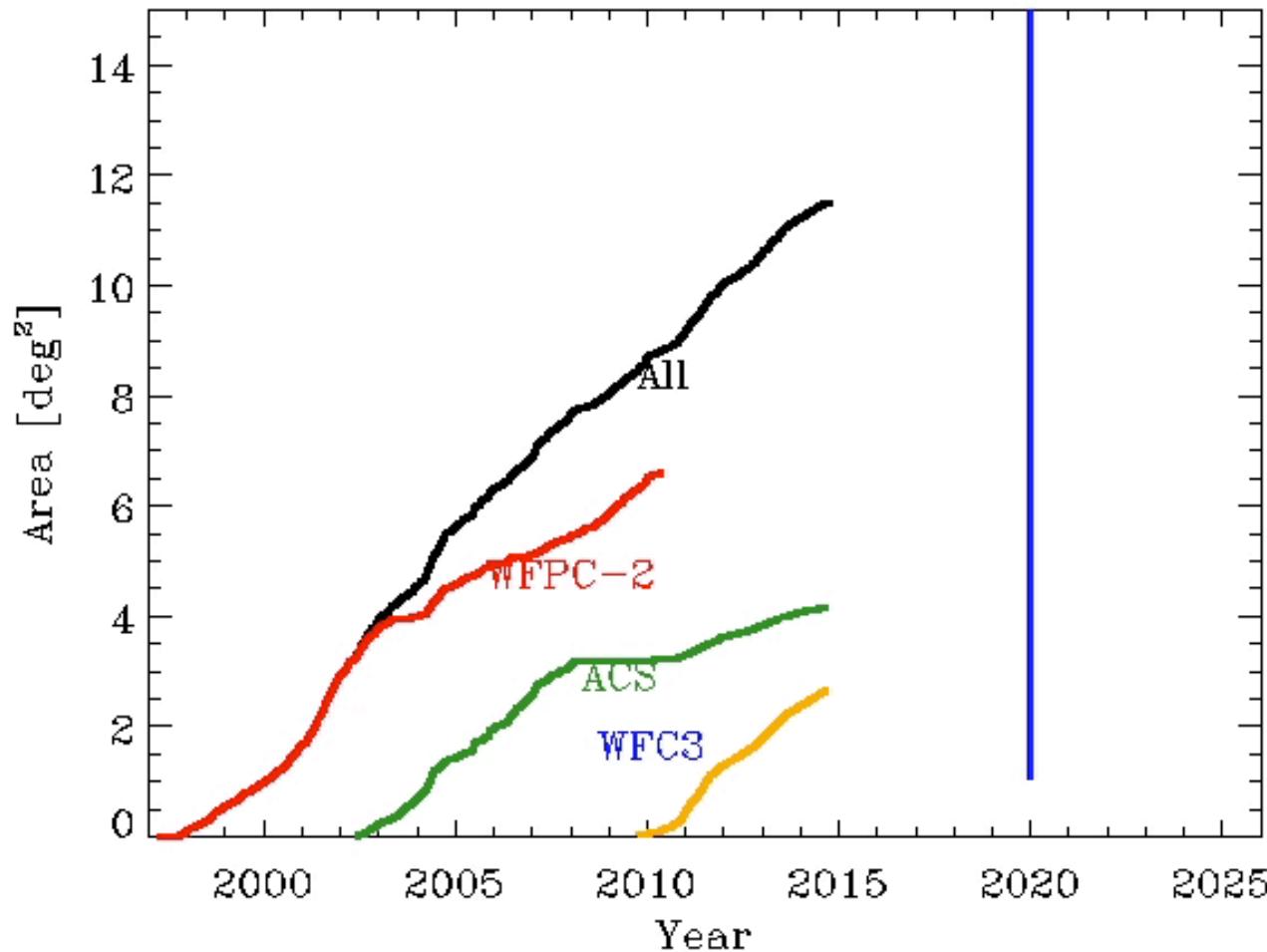
AS IN MANY OTHER DISCIPLINES THE BIG-DATA REVOLUTION HAS ARRIVED TO ASTRONOMY TOO



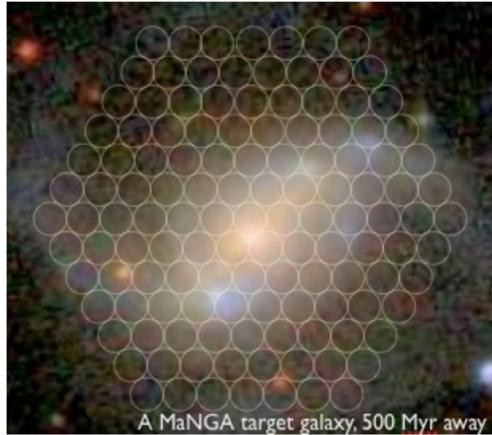
EXTREMELY LARGE
IMAGING SURVEYS
DELIVERING BILLIONS
OF OBJECTS IN 2-5 YEARS



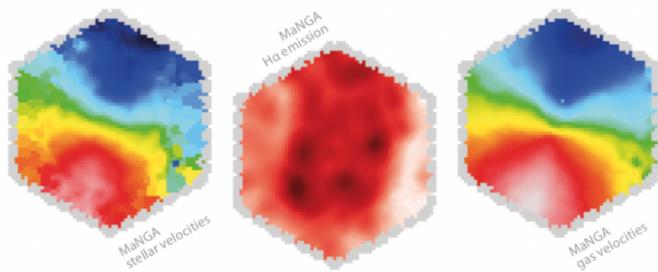
LSST simulation



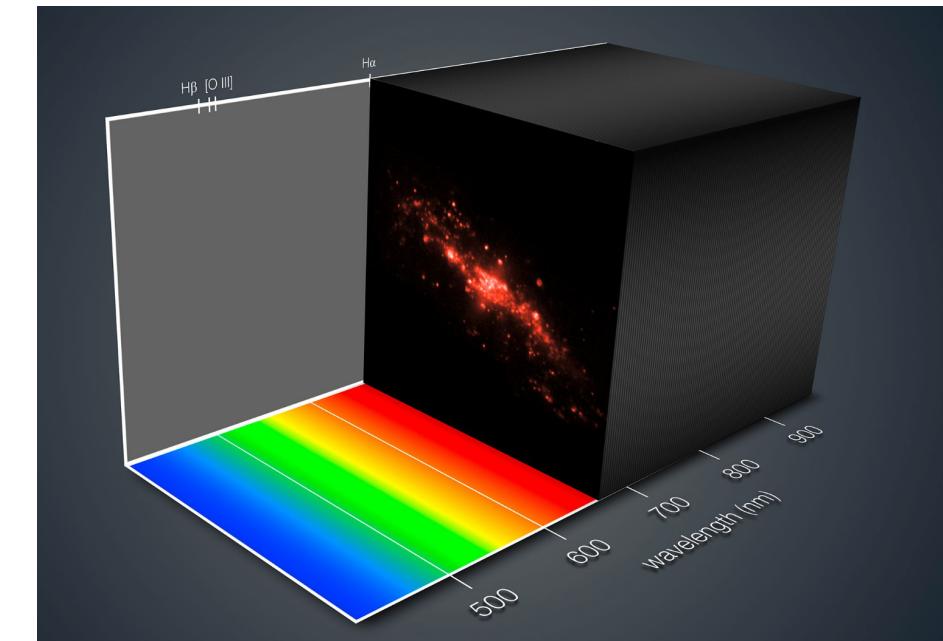
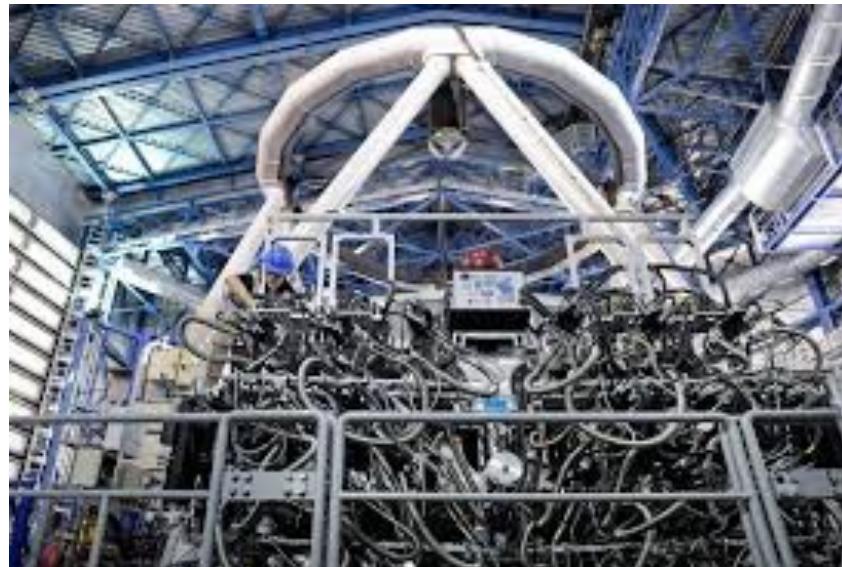
(Thanks to J. Brinchmann)



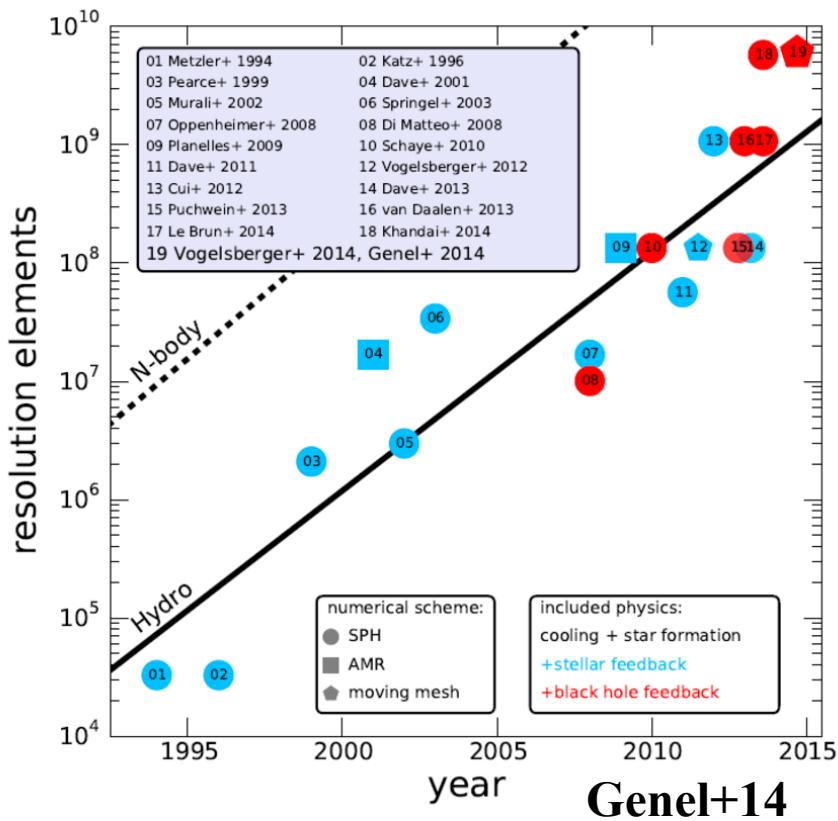
NOT ONLY VOLUME: AN INCREASING COMPLEXITY OF DATA



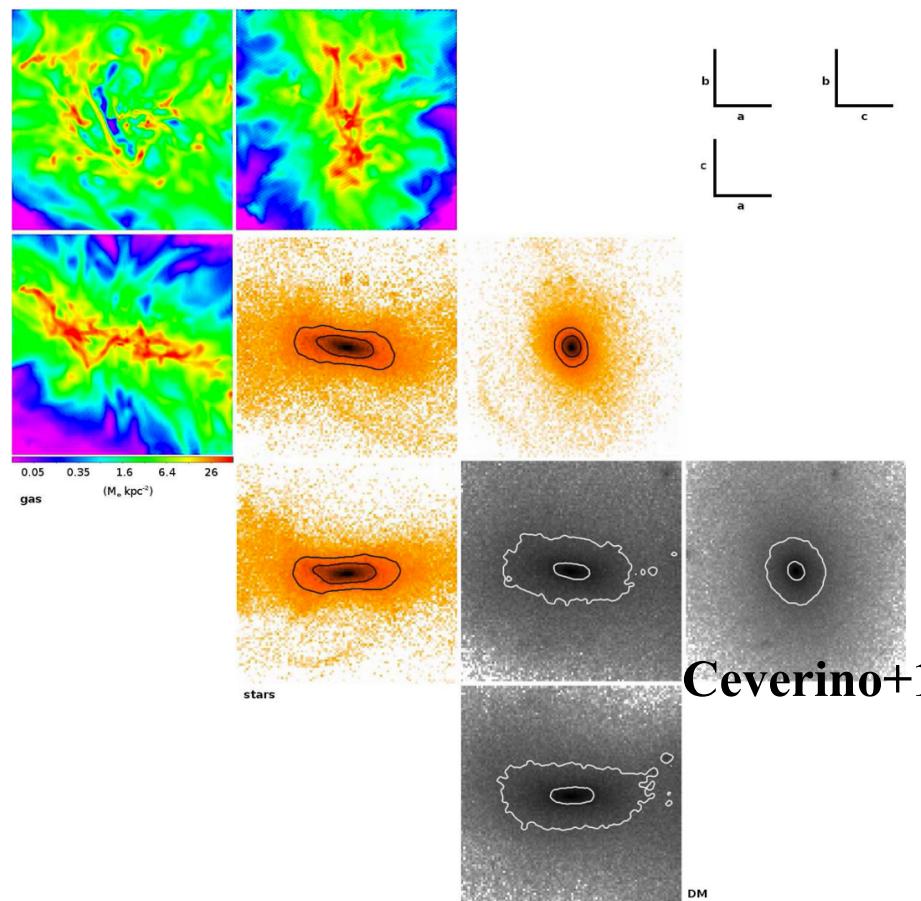
MANGA Survey



MUSE@VLT



AND ALSO SIMULATIONS!



PROGRAM FOR TODAY

- **PART I: AN INTRODUCTION TO ‘CLASSICAL’ MACHINE LEARNING**
 - UNSUPERVISED / SUPERVISED
 - GENERAL STEPS TO “TEACH A MACHINE”
 - SUPERVISED ALGORITHMS [CARTS, RF, SVMS, NN]
 - EVALUATION OF RESULTS [ROC, P-C..]

PROGRAM FOR TODAY

- PART II: FOCUS ON ‘SHALLOW’ NEURAL NETWORKS
 - PERCEPTRON, NEURON DEFINITION
 - LAYER OF NEURONS, HIDDEN LAYERS
 - ACTIVATION FUNCTIONS
 - OPTIMIZATION [GRADIENT DESCENT]
 - BACKPROPAGATION

PROGRAM FOR TODAY

- **PART III: CONVOLUTIONAL NEURAL NETWORKS**
 - CONVOLUTIONS AS NEURONS
 - CNNs [POOLING, DROPOUT]
 - VANISHING GRADIENT / BATCH NORMALIZATION

PROGRAM FOR TODAY

- PART IV: INTRODUCTION TO GENERATIVE MODELS
 - NETWORKS FOR IMAGE SEGMENTATION
 - AUTO-ENCODERS
 - GENERATIVE ADVERSARIAL NETWRKS

PROGRAM FOR TODAY

- **PART V: SOME PRACTICAL CONSIDERATIONS**
 - HOW DO I SETUP MY CNN?
 - HOW LARGE DO TRAINING SETS NEED TO BE?
 - OPTIMIZING YOUR NET: HYPER PARAMETER SEARCH
 - VISUALIZING CNNs [DECONVNETS, INCEPTIONISM, INTEGRATED GRADIENTS]

AFTERNOON HANDS-ON SESSION

WE WILL TRY TO IMPLEMENT SOME OF THE THINGS
LEARNED THIS MORNING

MORE PRECISELY WE WILL SET UP A DEEP NETWORK TO
MEASURE GALAXY ELLIPTICITIES

A GITHUB REPOSITORY INCLUDING DATA, NOTEBOOKS
AND ALSO THE SLIDES IS AVAILABLE HERE:

<https://github.com/mhuertascompany/deeplearningEUCLID>

LET'S TRY TO DISCUSS AS MUCH AS POSSIBLE!

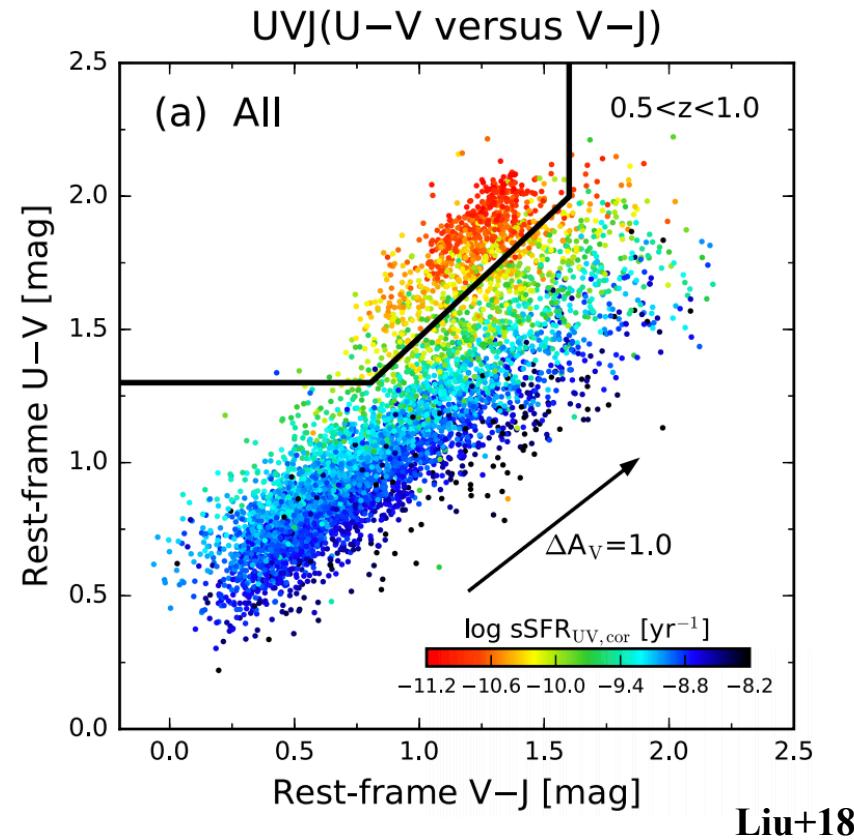
SOFTWARE REQUIREMENTS

- PYTHON 3 OR GREATER
- TENSORFLOW FOR DEEP LEARNING
- KERAS - HIGH LEVEL LIBRARY WHICH MAKES GPU CODING TRANSPARENT - SIMPLIFIES THINGS A LOT AND MOST OF THE TIME ENOUGH FOR OUR APPLICATIONS

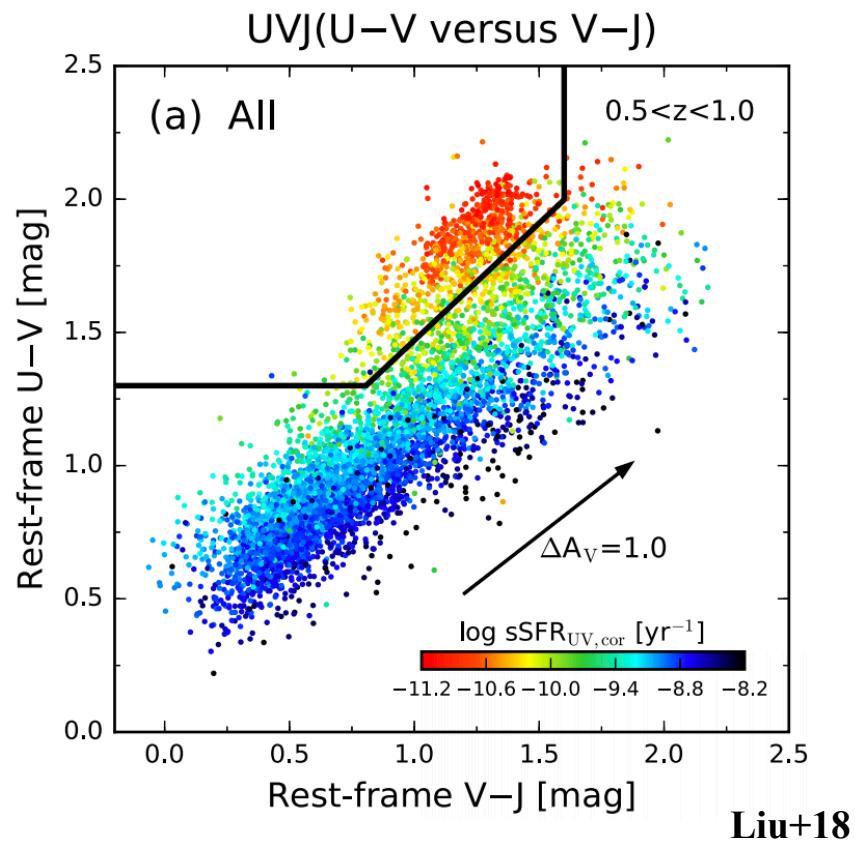
**ALL OF THIS SHOULD BE INSTALLED IN THE VIRTUAL
MACHINE**

PART I: AN INTRODUCTION TO
“CLASSICAL” MACHINE LEARNING

THRE IS NO MAGIC IN MACHINE LEARNING, AND IT IS ACTUALLY PRETTY SIMPLE

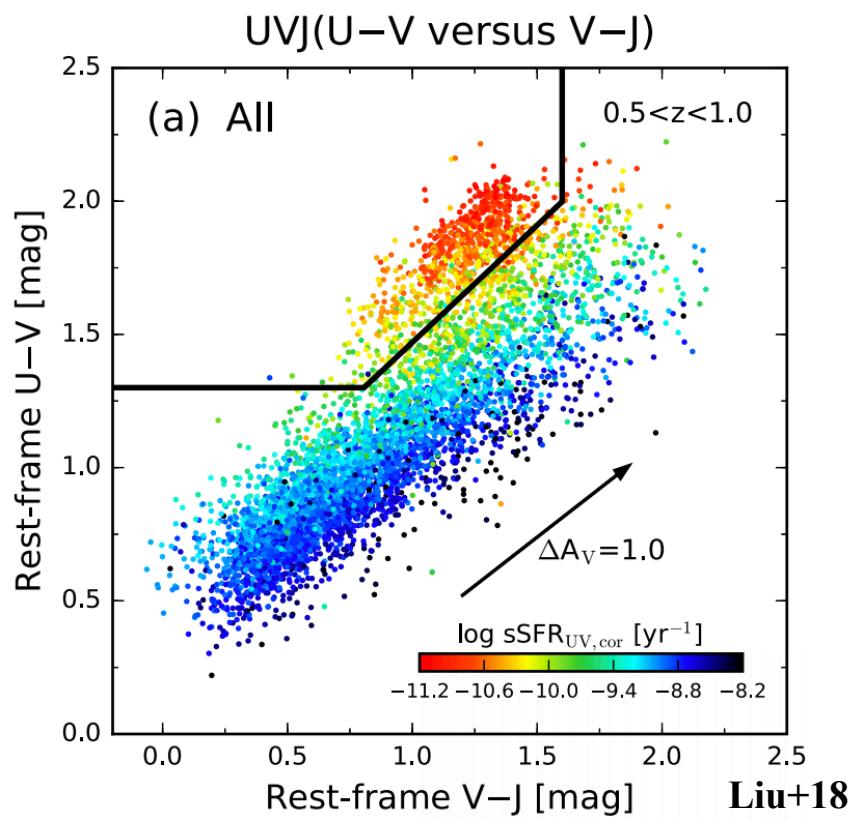


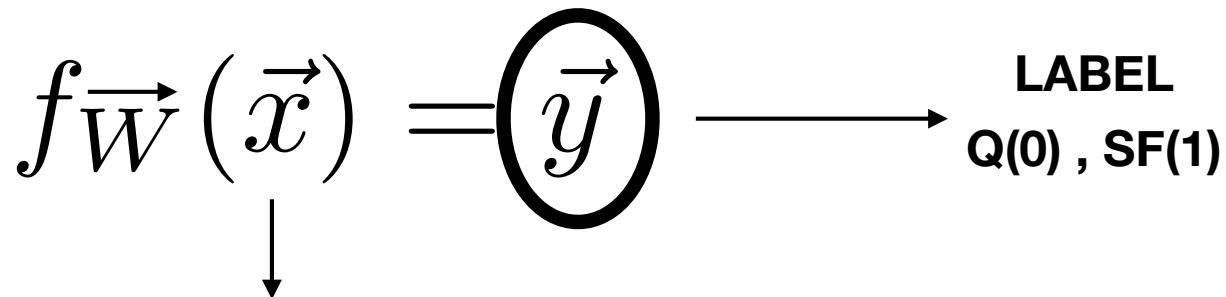
$$f_W(\vec{x}) = \vec{y}$$



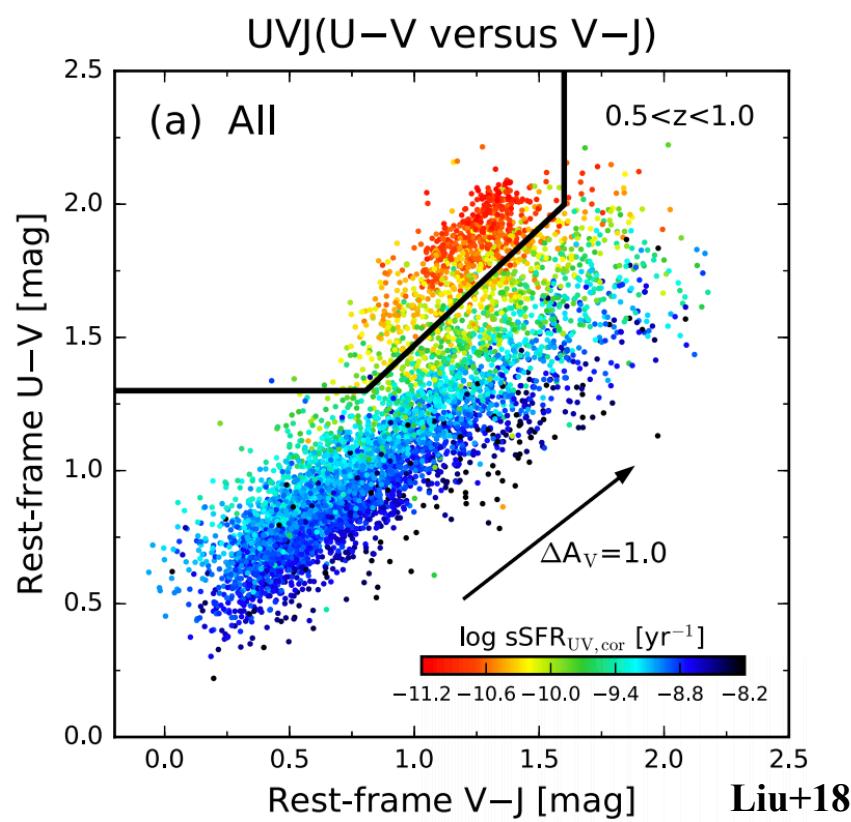
$$f_W(\vec{x}) = \vec{y}$$

LABEL
Q , SF





(U-V, V-J) FEATURES



$$f_{\vec{W}}(\vec{x}) = \vec{y} \longrightarrow \text{LABEL}$$

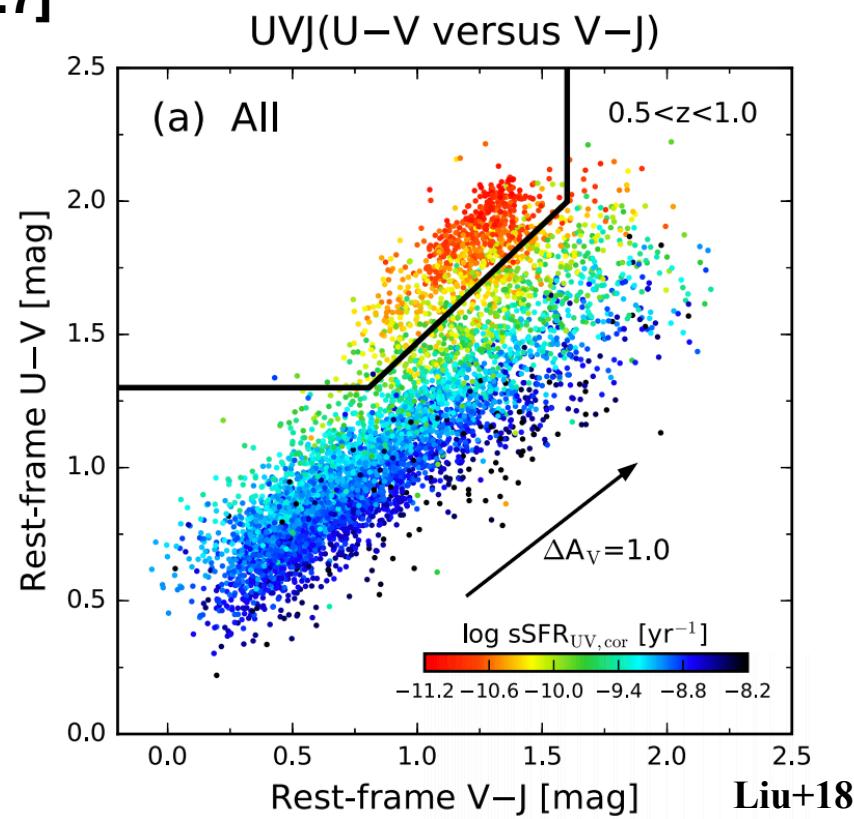
Q(0) , SF(1)

NETWORK FUNCTION

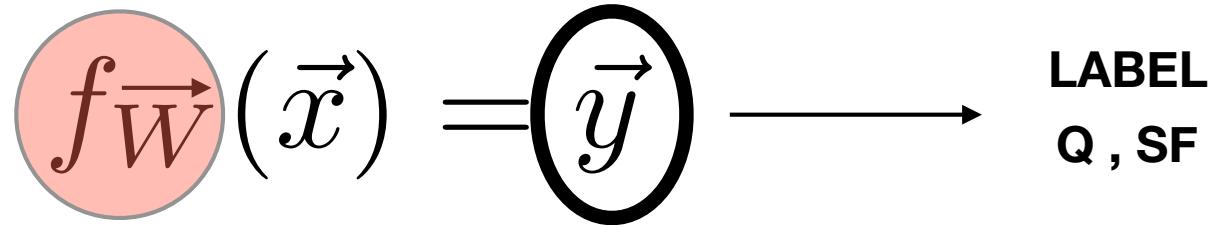
(U-V, V-J) FEATURES

$$\text{sgn}[(u-v)-0.8*(v-j)-0.7]$$

WEIGHTS



**“CLASSICAL”
MACHINE LEARNING**



$$\text{sgn}[(u-v)-W_1*(v-j)-W_2]$$



**REPLACE THIS BY A GENERAL
NON LINEAR FUNCTION WITH SOME PARAMETERS W**

WHAT DOES MACHINE LEARNING DO?



WHAT DOES MACHINE LEARNING DO?

SUPERVISED

Classification

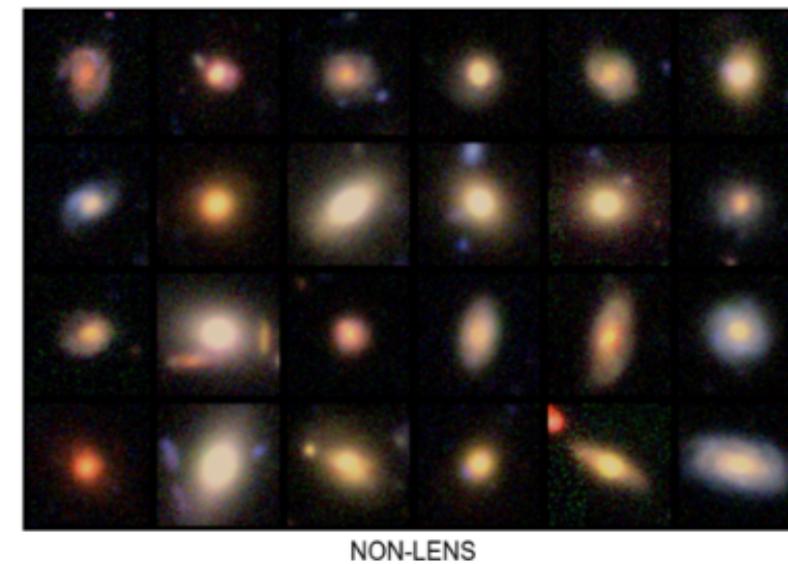
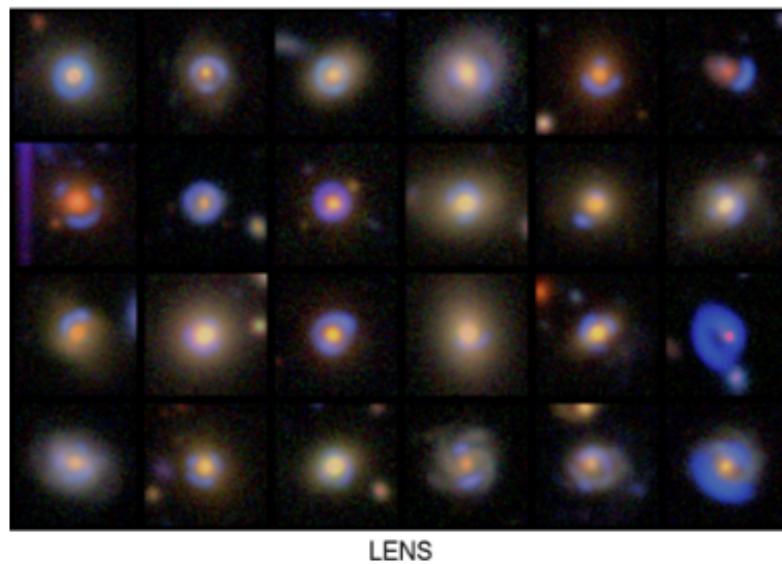
Regression

UN-SUPERVISED

Clustering

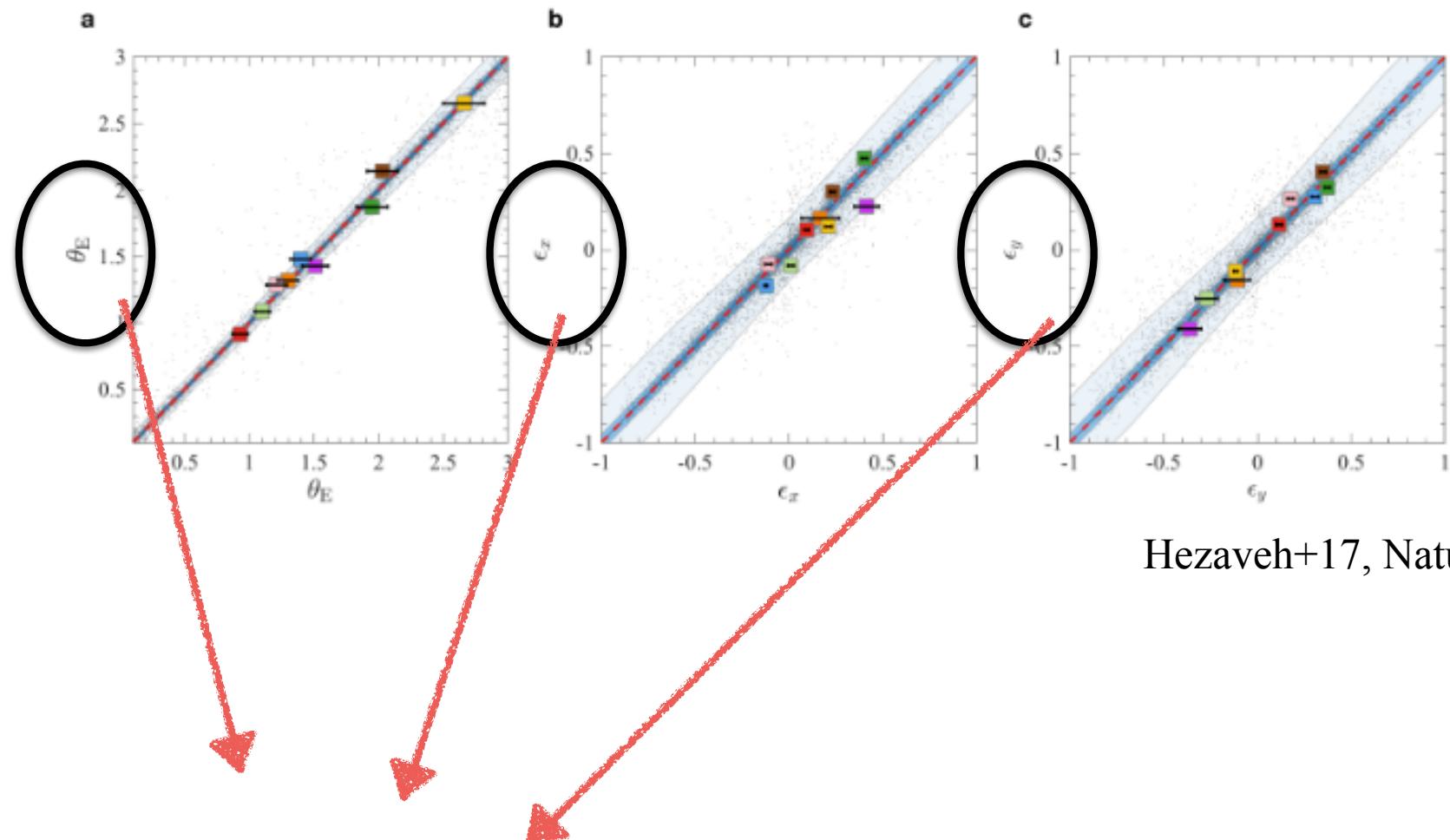
Generative

Classification



Jacobs+17

Regression

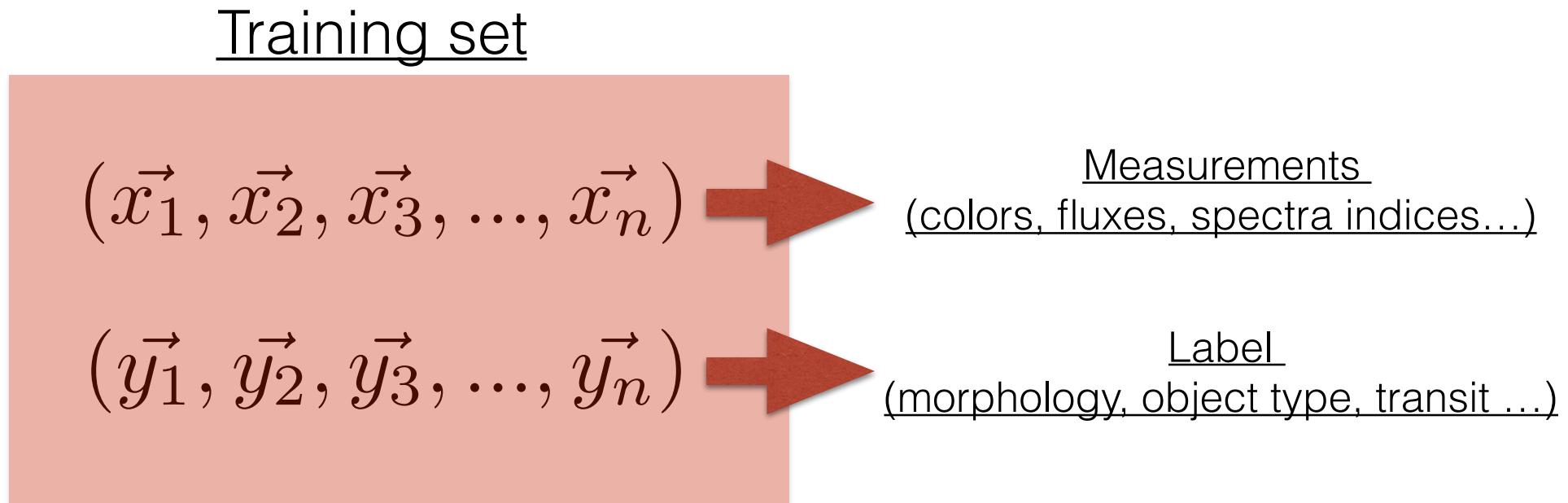


Hezaveh+17, Nature

REGRESSION ON
STRONG LENSES PARAMETERS

SUPERVISED LEARNING

Given a dataset with known labels (measurements) - find a function that can assign (predict) measurements for an unlabeled dataset



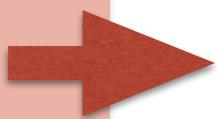
SUPERVISED LEARNING

Given a dataset with known labels (measurements) - find a function that can assign (predict) measurements for an unlabeled dataset

Training set

$$(\vec{x}_1, \vec{x}_2, \vec{x}_3, \dots, \vec{x}_n)$$

$$(\vec{y}_1, \vec{y}_2, \vec{y}_3, \dots, \vec{y}_n)$$

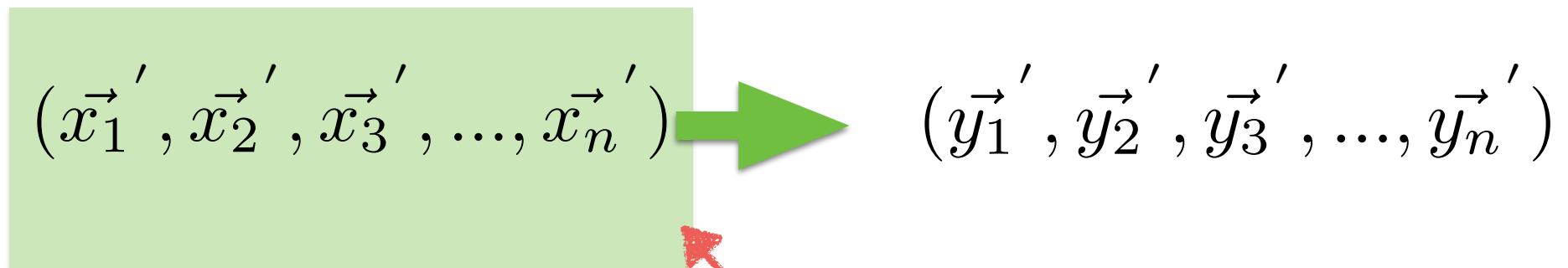


$$f_W(\vec{x}) = \vec{y}$$

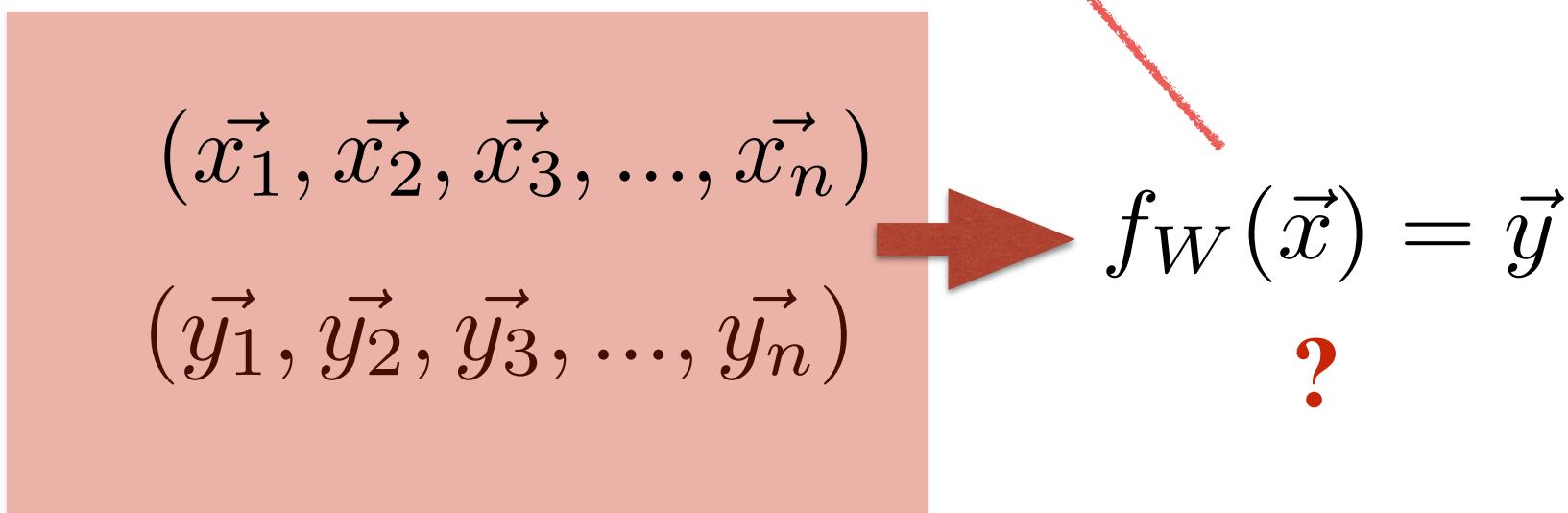
?

SUPERVISED LEARNING

Unlabeled set



Training set



$$(\vec{x}_1, \vec{x}_2, \vec{x}_3, \dots, \vec{x}_n)$$

$$\vec{x} \in \mathbb{R}^d$$

$$(\vec{y}_1, \vec{y}_2, \vec{y}_3, \dots, \vec{y}_n)$$

$$\vec{y} \in \mathbb{R} \quad \vec{y} \in \mathbb{N}$$

GENERAL GOAL: Find a (non-linear) function that outputs the correct class / measurement for a given input object:

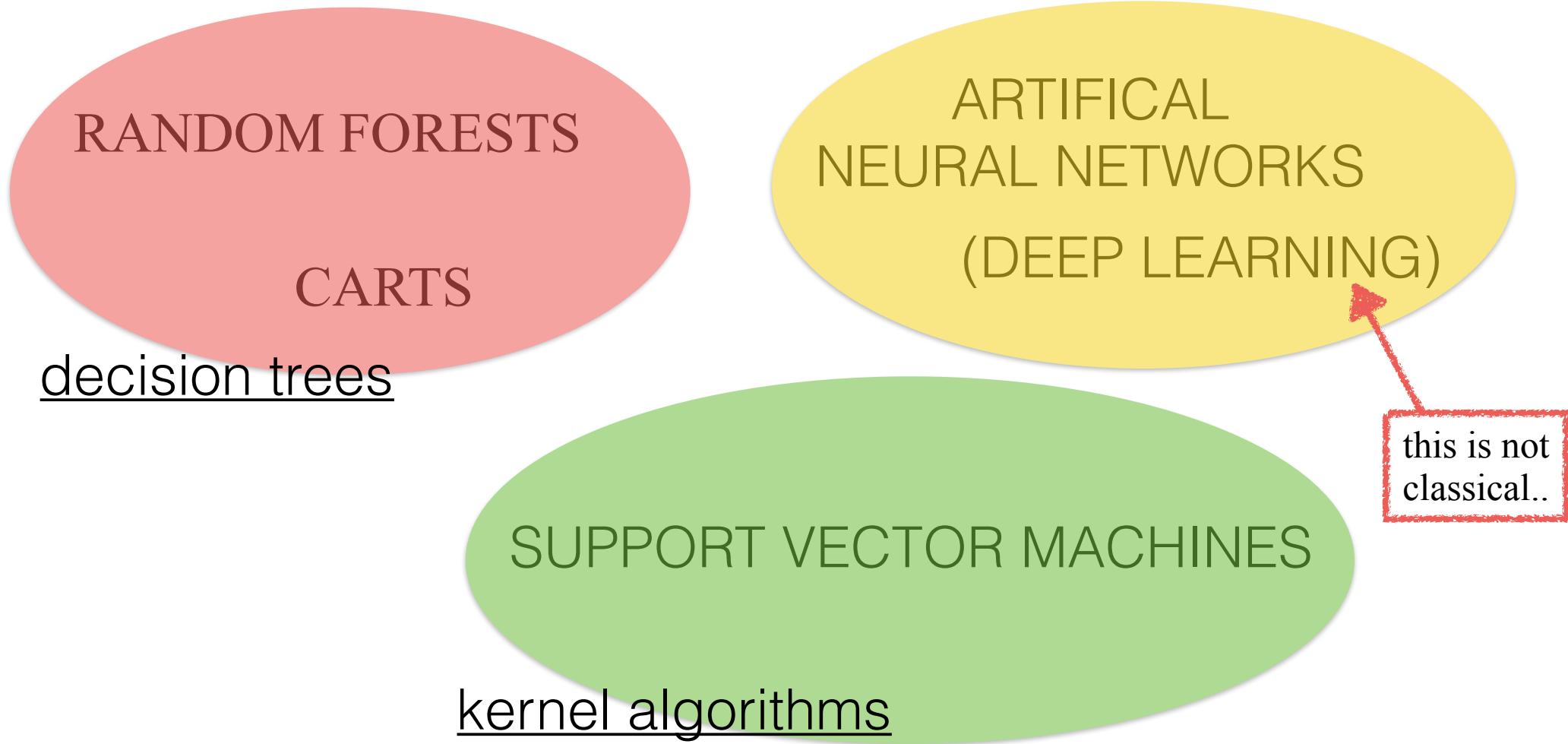
$$f_W(\vec{x})$$



Number of parameters - can be large

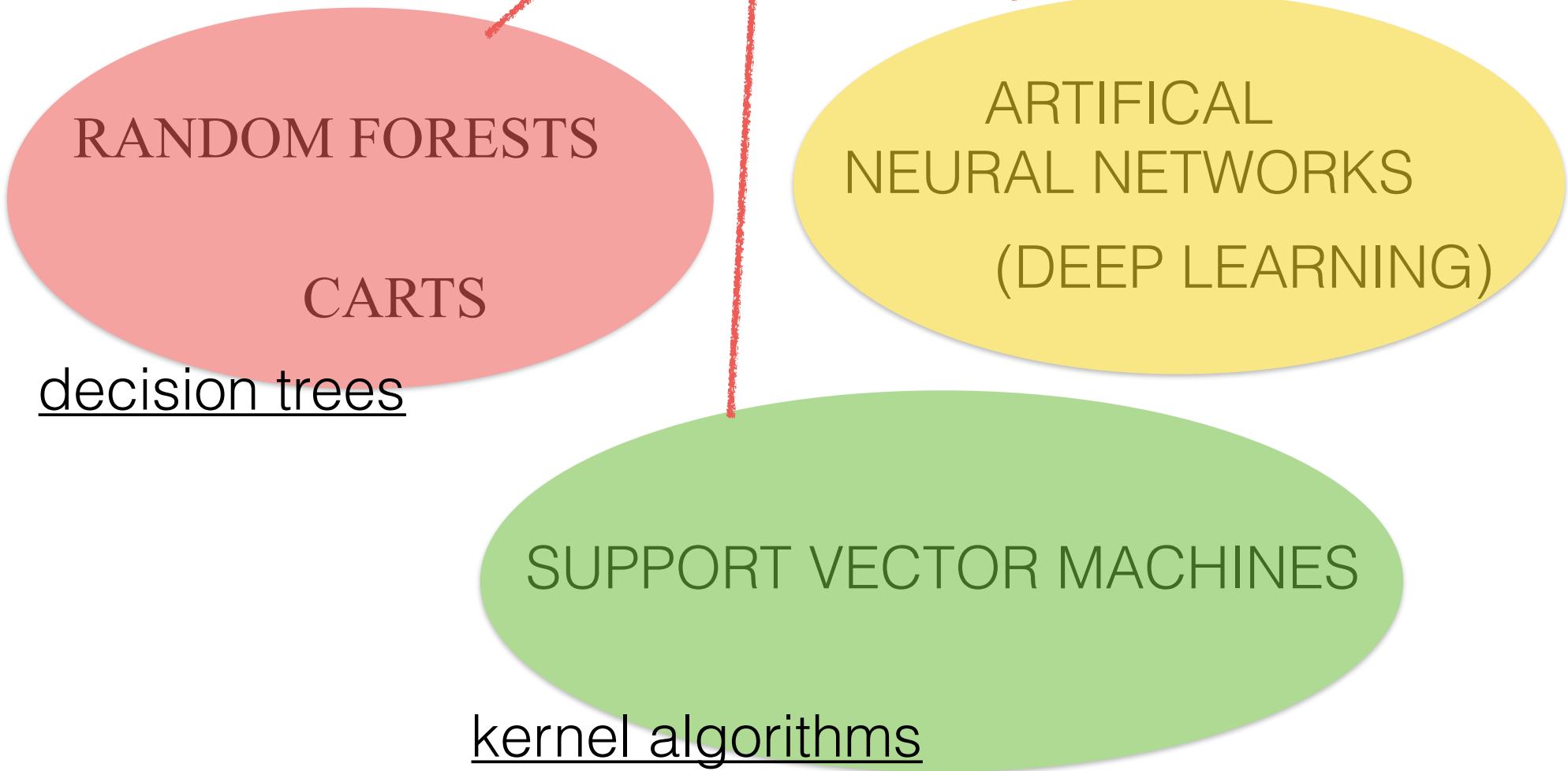
It is translated into a minimization problem : find \mathbf{W} such as the prediction error is minimal over all unseen vectors

Different “classical” supervised machine learning methods



The differences are
in the function
that is used

$$f_W(\vec{x})$$



We need two key elements

1. A LOSS FUNCTION

**2. A MINIMIZATION OR OPTIMIZATION
ALGORITHM**

We need two key elements

1. A LOSS FUNCTION

**2. A MINIMIZATION OR OPTIMIZATION
ALGORITHM**

**THIS IS COMMON TO ALL MACHINE LEARNING
ALGORITHMS**

1. DEFINE A LOSS FUNCTION

$$loss(F_W(.), \vec{x}_i, \vec{y}_i)$$

Typically: $(F_W(\vec{x}_i) - \vec{y}_i)^2$ Quadratic loss function

2. MINIMIZE THE EMPIRICAL RISK

$$\mathfrak{R}_{empirical}(W) = \frac{1}{N} \sum_i^N [loss(W, \vec{x}, \vec{y})]$$



MINIMIZE THE RISK

EMPIRICAL RISK?

$$\mathfrak{R}_{\text{empirical}}(W) = \frac{1}{N} \sum_i^N [\text{loss}(W, \vec{x}, \vec{y})]$$

WE ARE MINIMIZING WITH RESPECT TO A FINITE NUMBER OF OBSERVED EXAMPLES

EMPIRICAL RISK?

$$\mathfrak{R}_{\text{empirical}}(W) = \frac{1}{N} \sum_i^N [\text{loss}(W, \vec{x}, \vec{y})]$$

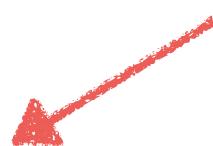
WE ARE MINIMIZING WITH RESPECT TO A FINITE NUMBER OF OBSERVED EXAMPLES

OBSERVED DATASET



EMPIRICAL RISK?

$$\mathfrak{R}_{\text{empirical}}(W) = \frac{1}{N} \sum_i^N [\text{loss}(W, \vec{x}, \vec{y})]$$



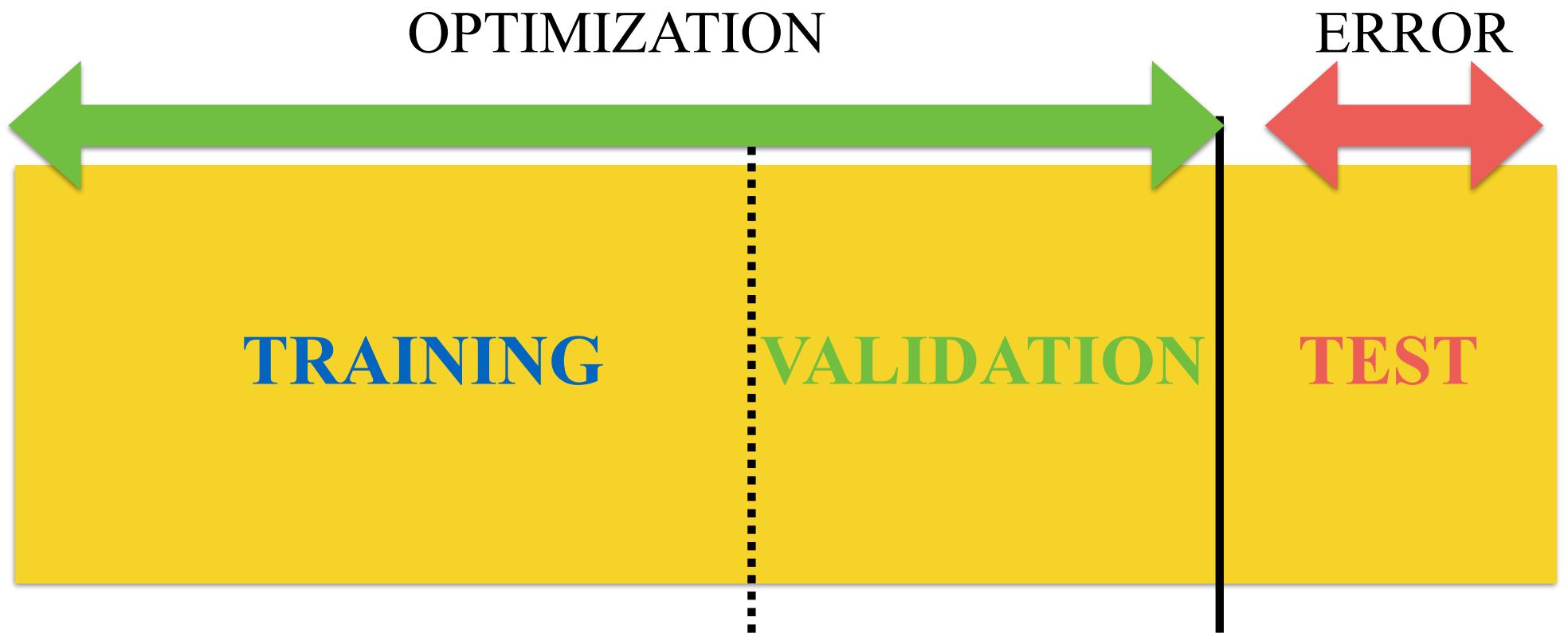
WE ARE MINIMIZING WITH RESPECT TO A FINITE NUMBER OF OBSERVED EXAMPLES

ALL “GALAXIES IN THE UNIVERSE”

OBSERVED DATASET



In practice

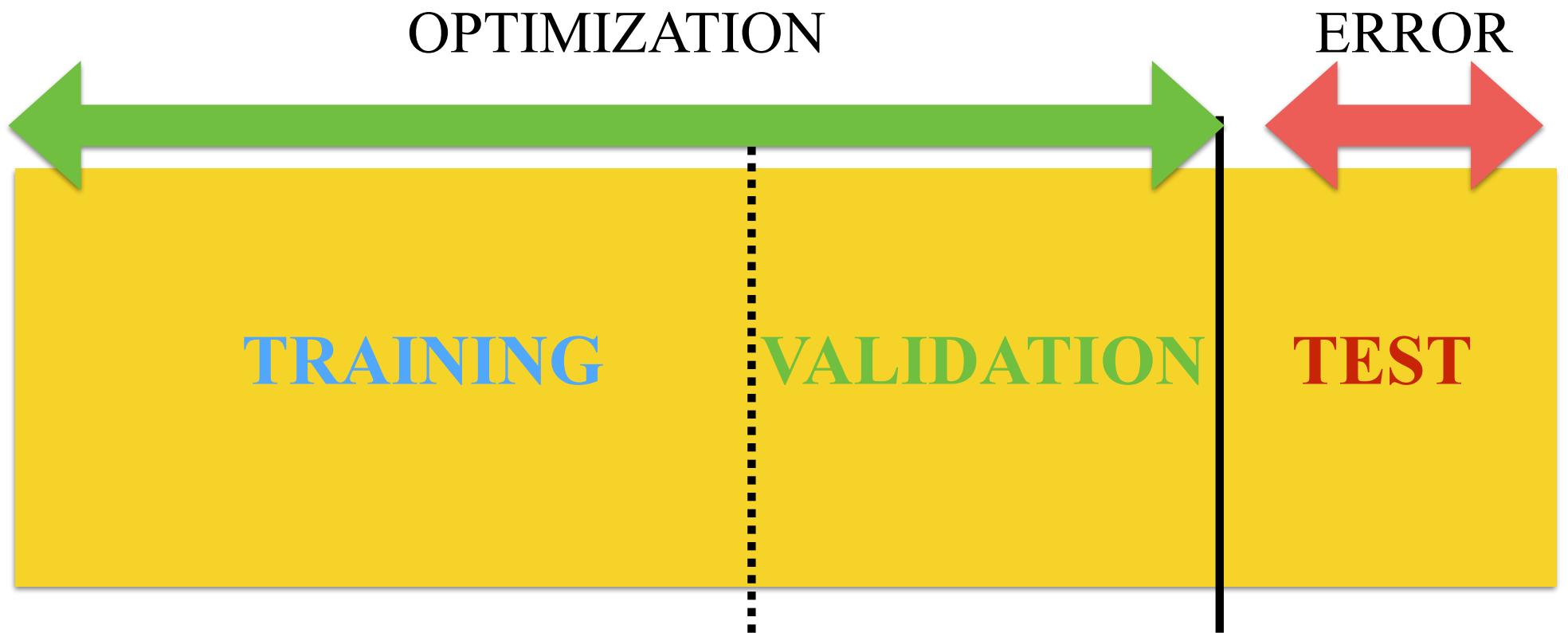


training set: use to train the classifier

validation set: use to monitor performance in real time - check
for overfitting

test set: use to train the classifier

In practice



**NO CHEATING! NEVER USE TRAINING TO VALIDATE
YOUR ALGORITHM!**

The algorithm used to minimize is
called OPTIMIZATION

THERE ARE SEVERAL OPTIMIZATION TECHNIQUES

Optimization

THERE ARE SEVERAL OPTIMIZATION TECHNIQUES

THEY DEPEND ON THE MACHINE LEARNING ALGORITHM

Optimization

THERE ARE SEVERAL OPTIMIZATION TECHNIQUES

THEY DEPEND ON THE MACHINE LEARNING ALGORITHM

NEURAL NETWORKS USE THE GRADIENT DESCENT AS WE
WILL SEE LATER

$$W_{t+1} = W_t - \lambda_h \nabla f(W_t)$$

weights to be learned



epoch



learning rate



DECISION TREES ALGORITHMS

DECISION TREES

CARTS

Classification Trees

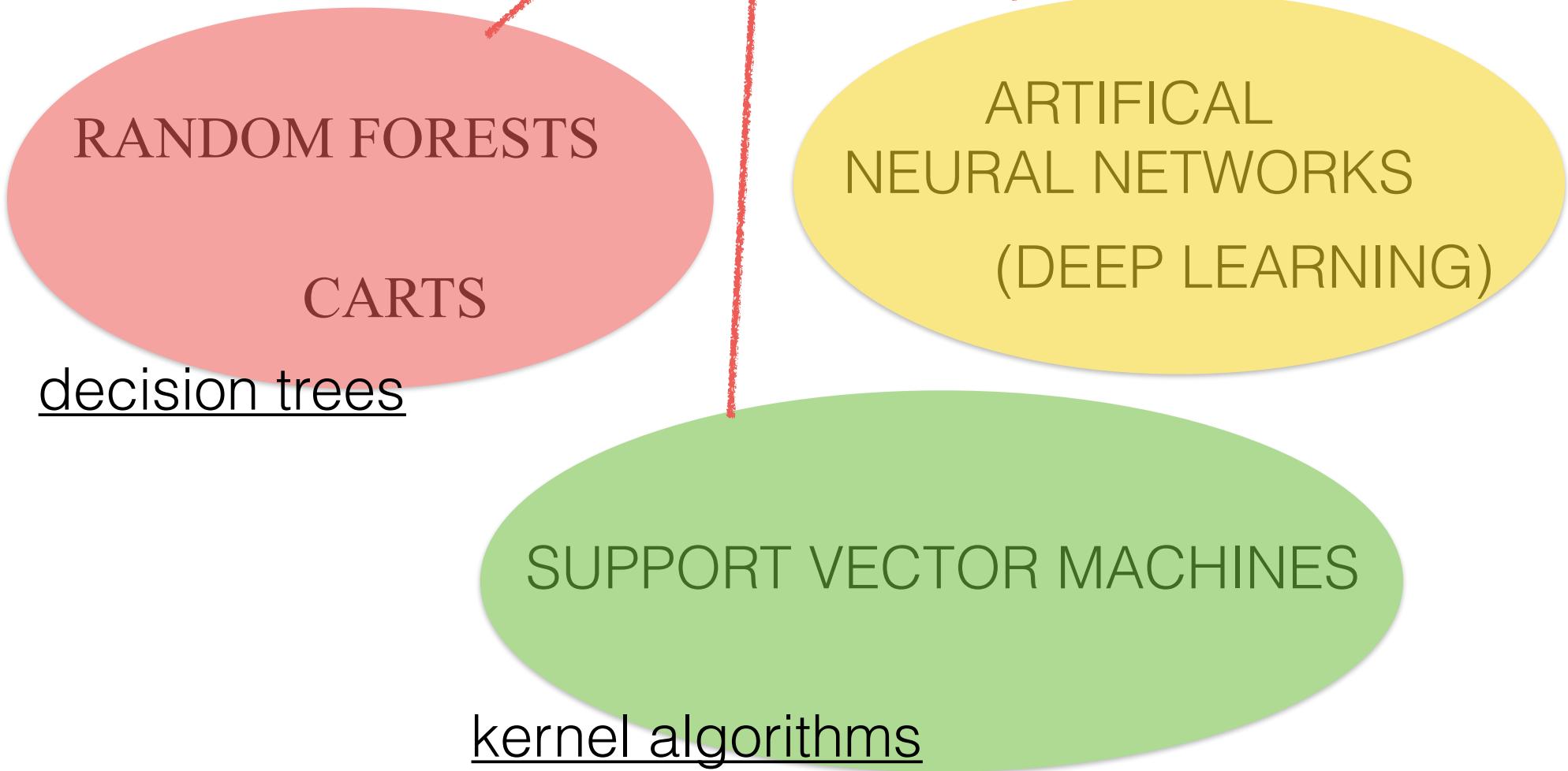
Regression Trees

BOOSTED TREES

Random Forests

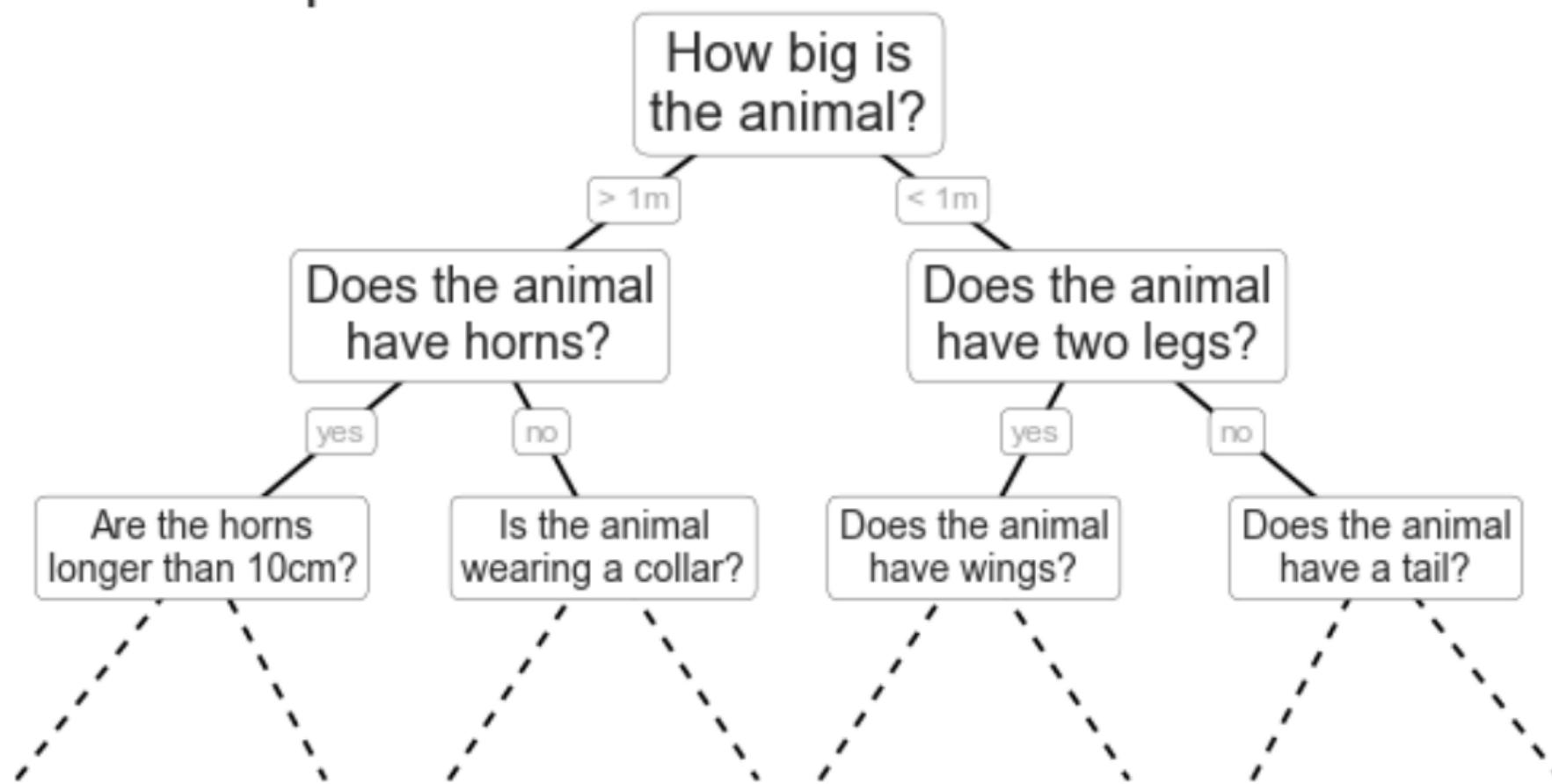
The differences are
in the function
that is used

$$f_W(\vec{x})$$

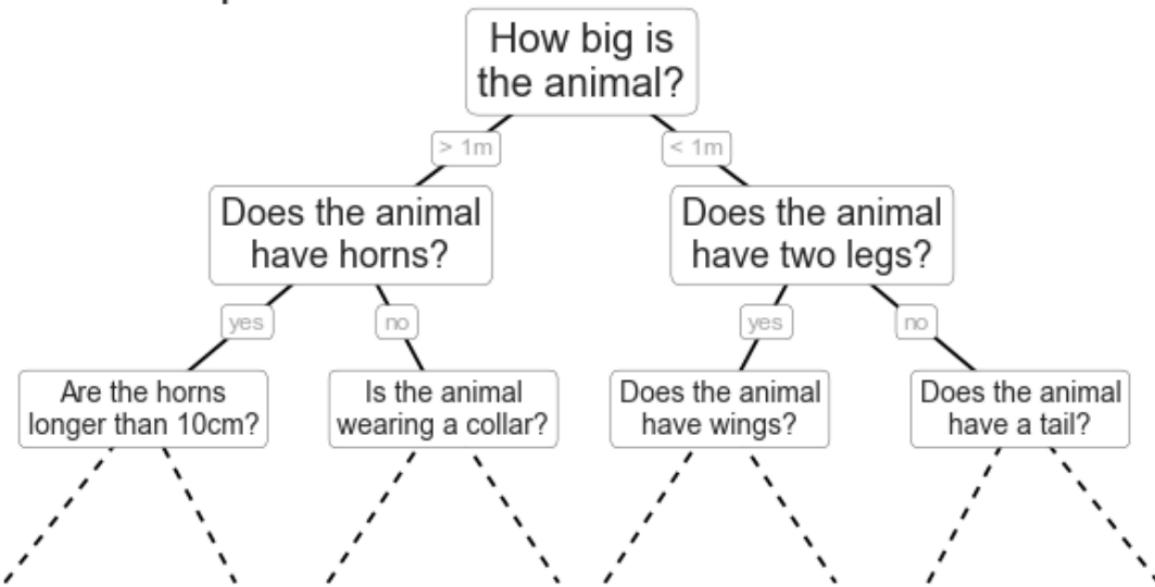


CLASSIFICATION AND REGRESSION TREES (CARTS)

THIS IS THE SIMPLEST AND MORE INTUITIVE MACHINE LEARNING ALGORITHM



DECISION TREES (CARTS)



IT IS BUILT IN AN ITERATIVE WAY

1. FROM THE INPUT PARAMETERS, FIRST FIND THE PROPERTY THAT BEST SEPARATE INTO 2 GROUPS [I.E. **MINIMIZES SOME LOSS FUNCTION**]
2. REPEAT STEP 1 WITH ANOTHER PARAMETER
3. AT THE END THERE IS A TREE WHERE, AT EACH POINT, ONE OF TWO DECISIONS CAN BE MADE

DECISION TREES (CARTS)

TYPICAL METRICS USED:

[THE IDEA IS TO FIND THE SPLITTING VALUE THAT PUTS ALL OBJECTS OF A GIVEN CLASS IN ONE LEAF]

DECISION TREES (CARTS)

TYPICAL METRICS USED:

GINI IMPURITY

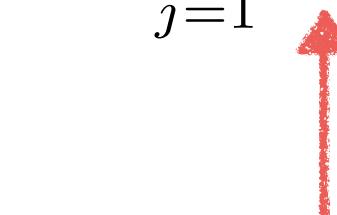
$$G = 1 - \sum_{j=1}^c p_j^2$$

DECISION TREES (CARTS)

TYPICAL METRICS USED:

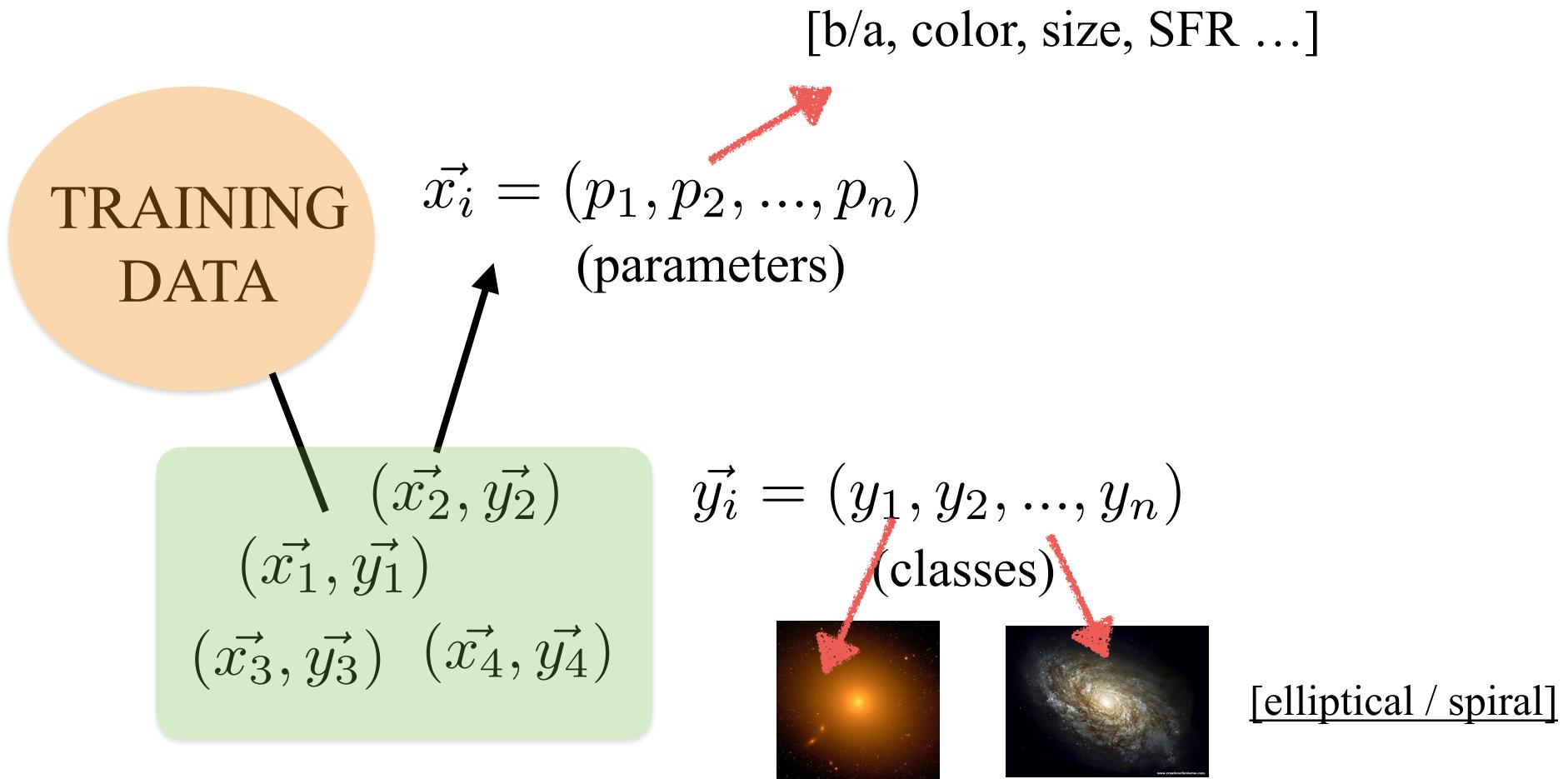
GINI IMPURITY

$$G = 1 - \sum_{j=1}^c p_j^2$$

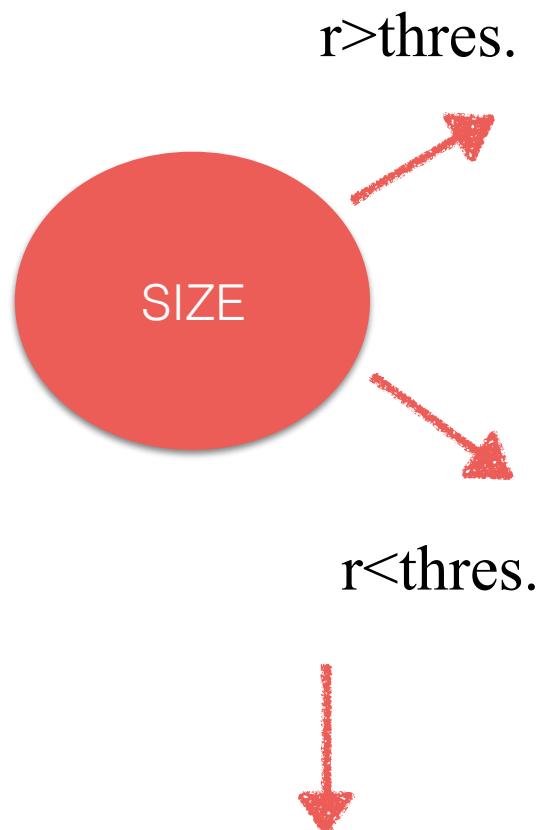


fraction of
objects in each class given
a split value

DECISION TREES (CARTS)



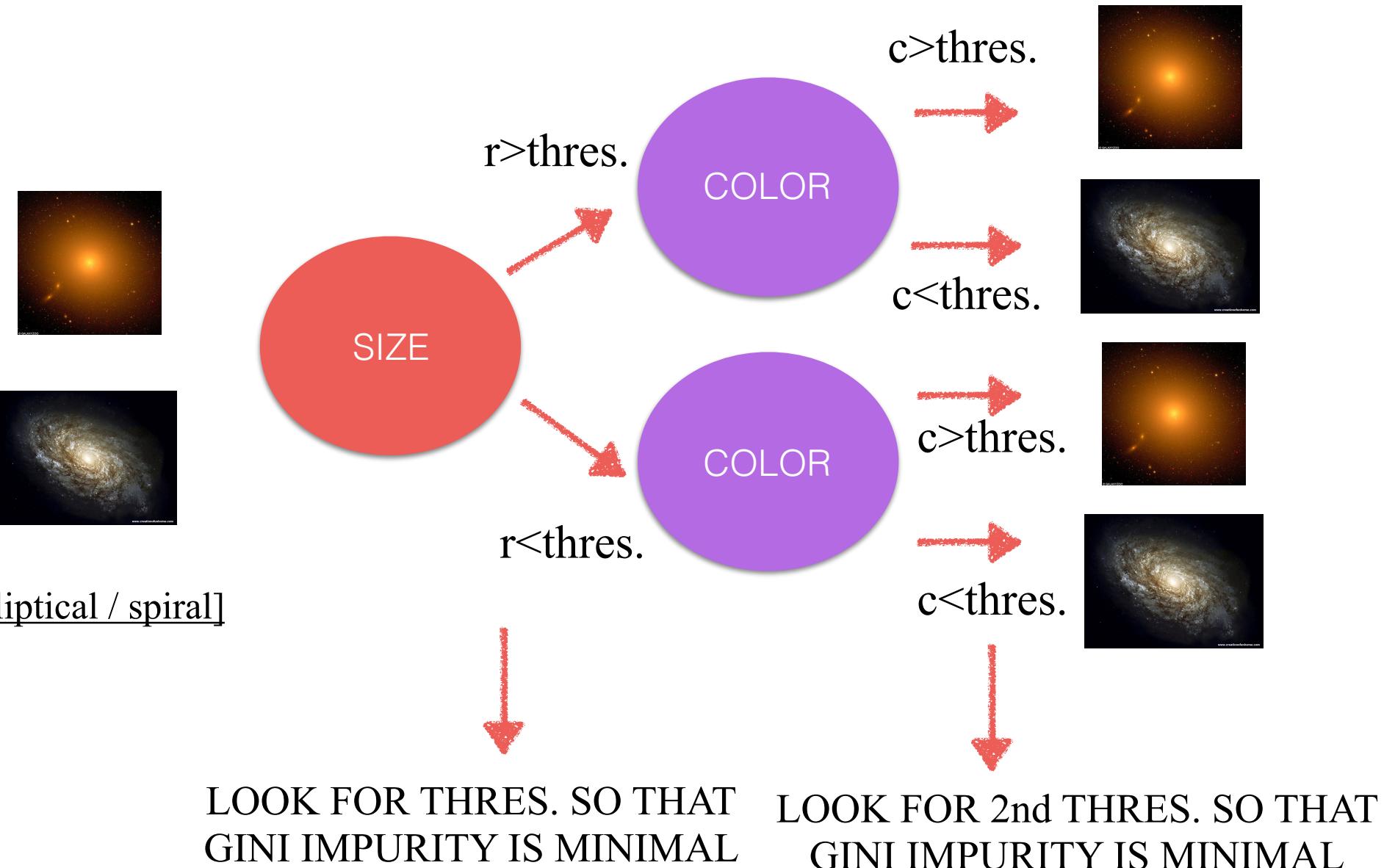
DECISION TREES (CARTS)



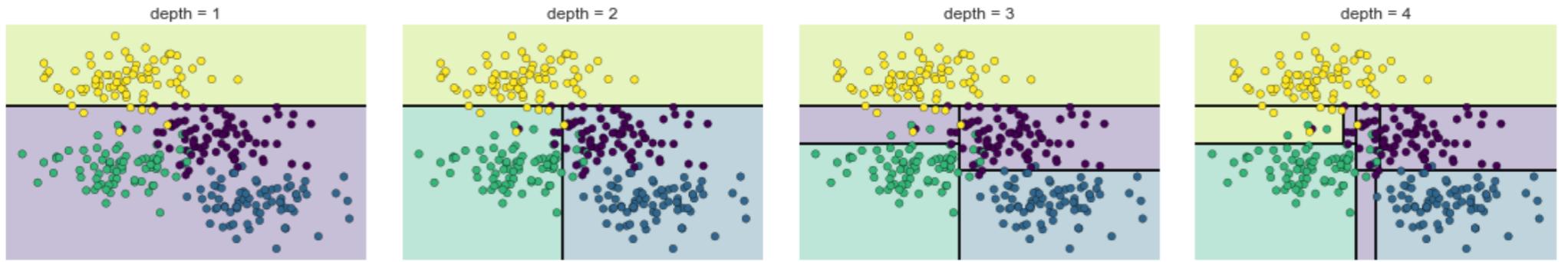
[elliptical / spiral]

LOOK FOR THRES. SO THAT
GINI IMPURITY IS MINIMAL

DECISION TREES (CARTS)



DECISION TREES (CARTS)



IT IS SIMPLY A PARTITION OF THE PARAMETER SPACE WITH
CONSTANT BOUNDARIES

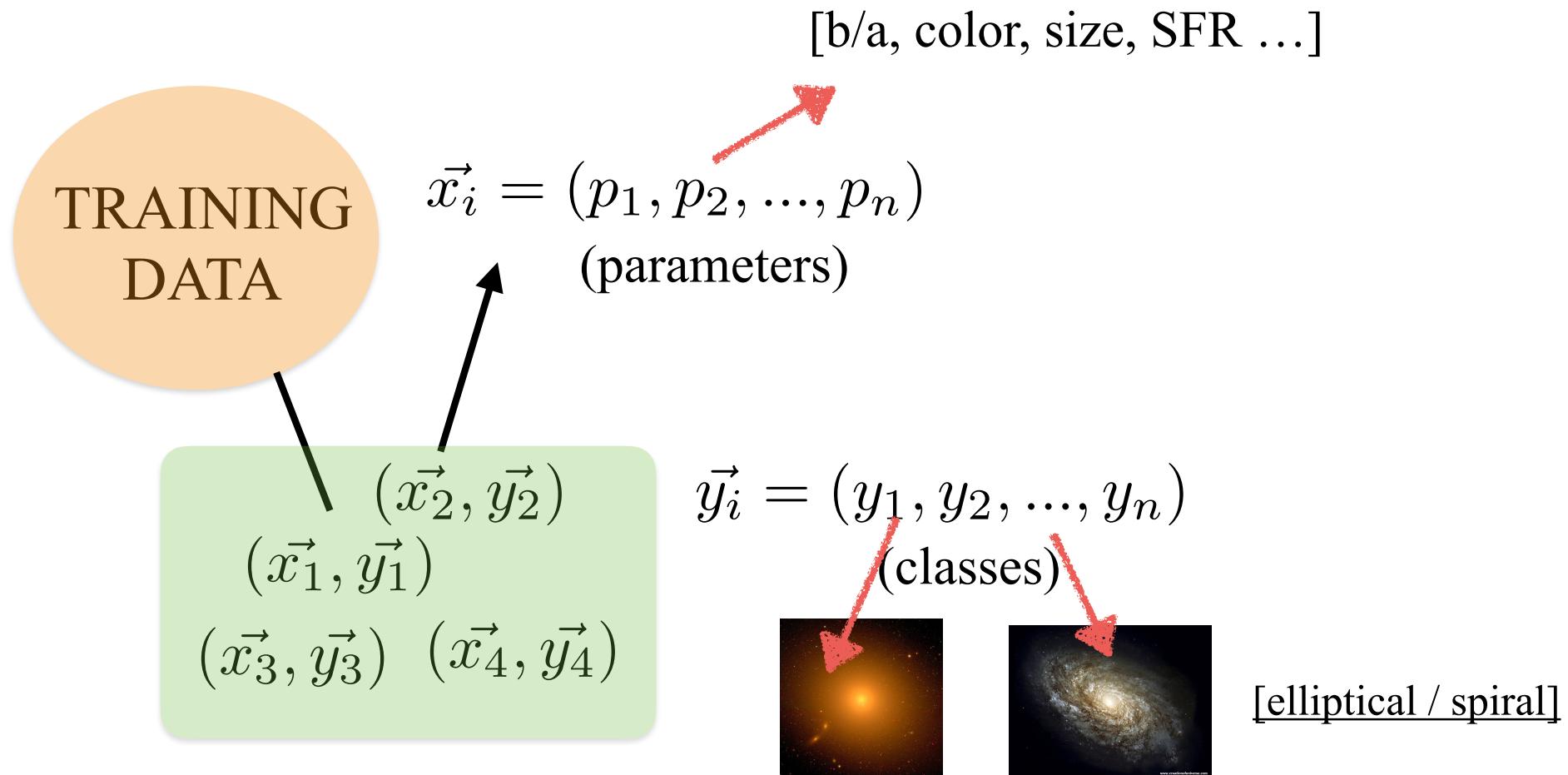
RANDOM FORESTS

ONE PROBLEM WITH CLASSIFICATION TREES IS THAT
THEY CAN EASILY OVERFIT

THE DECISIONS ARE VERY SPECIFIC TO THE TRAINING
SET AND NOT REPRESENTATIVE OF THE FULL
POPULATION

RANDOM FORESTS

RANDOM FORESTS TRY TO SOLVE THIS PROBLEM BY INTRODUCING SOME RANDOM INFORMATION IN THE TRAINING PROCESS



RANDOM FORESTS

(\vec{x}_2, \vec{y}_2)

(\vec{x}_1, \vec{y}_1)

(\vec{x}_3, \vec{y}_3) (\vec{x}_4, \vec{y}_4)

$\vec{x}_i = (p_1, p_2, \dots, p_n)$

RANDOM FORESTS

(\vec{x}_2, \vec{y}_2)
 (\vec{x}_1, \vec{y}_1)
 (\vec{x}_3, \vec{y}_3) (\vec{x}_4, \vec{y}_4)

$\vec{x}_i = (p_1, p_2, \dots, p_n)$



[elliptical / spiral]

SIZE

$r > \text{thres.}$

$r < \text{thres.}$



COLOR

$c > \text{thres.}$



$c < \text{thres.}$



$c > \text{thres.}$



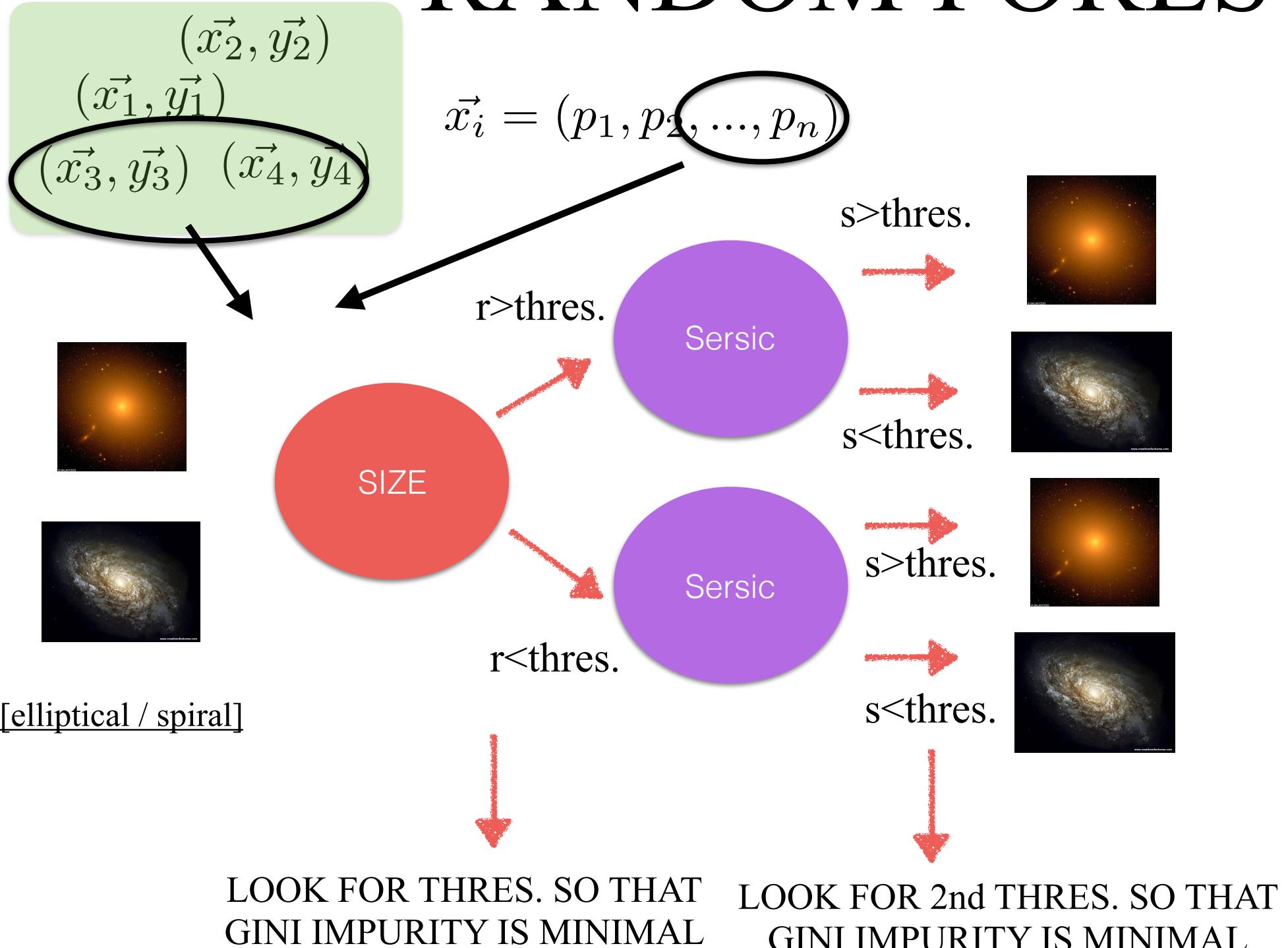
$c < \text{thres.}$



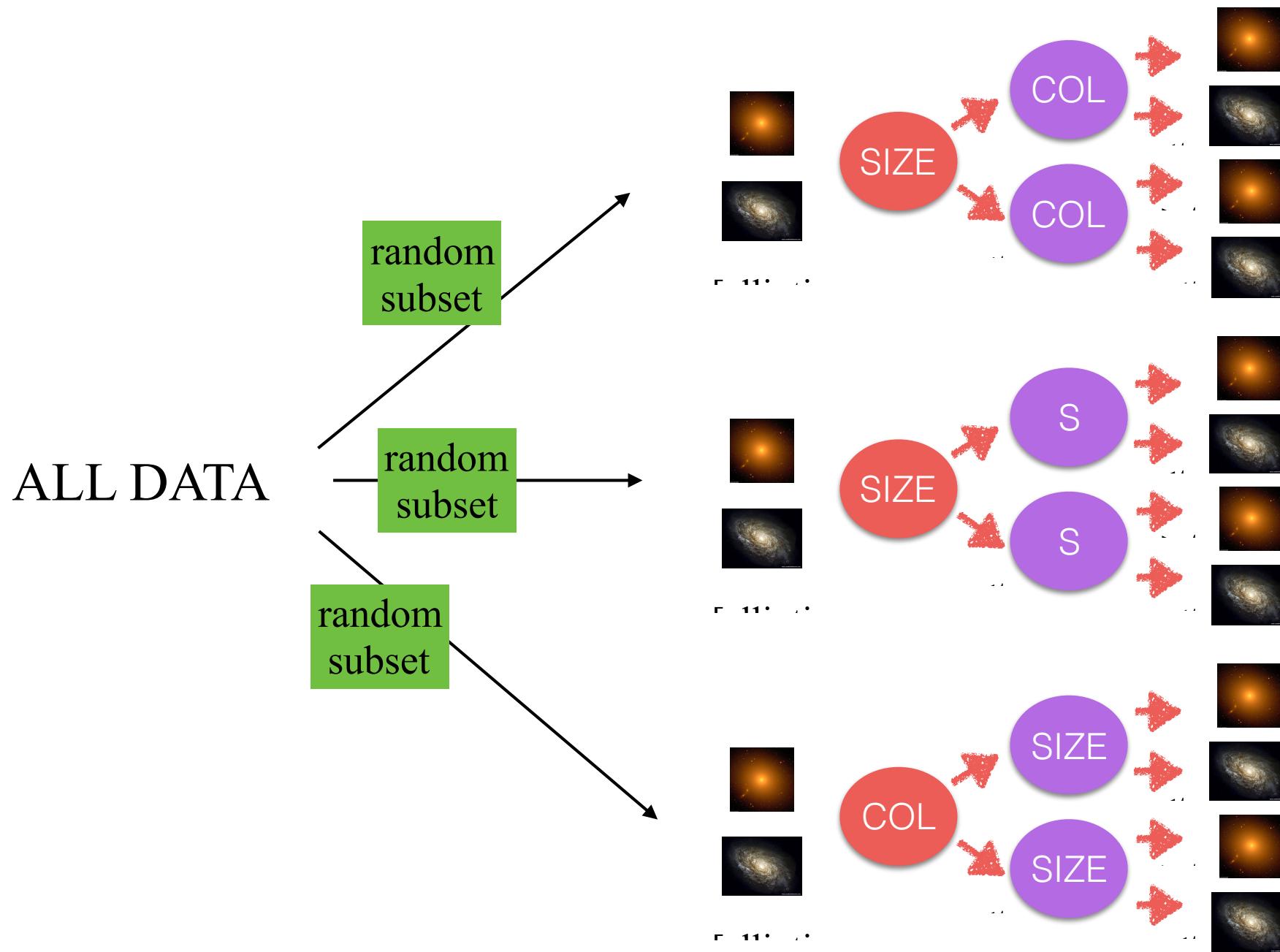
LOOK FOR THRES. SO THAT
GINI IMPURITY IS MINIMAL

LOOK FOR 2nd THRES. SO THAT
GINI IMPURITY IS MINIMAL

RANDOM FORESTS

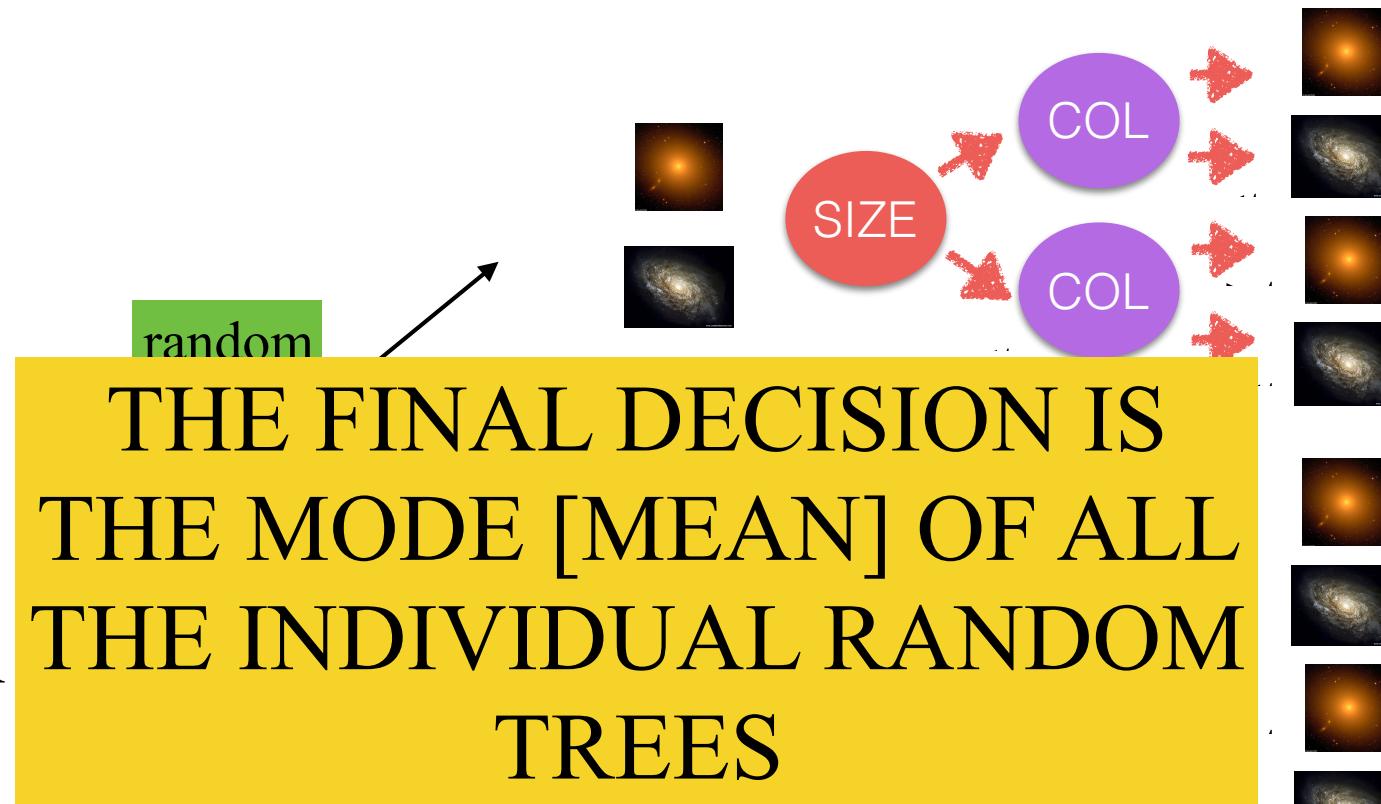


RANDOM FORESTS

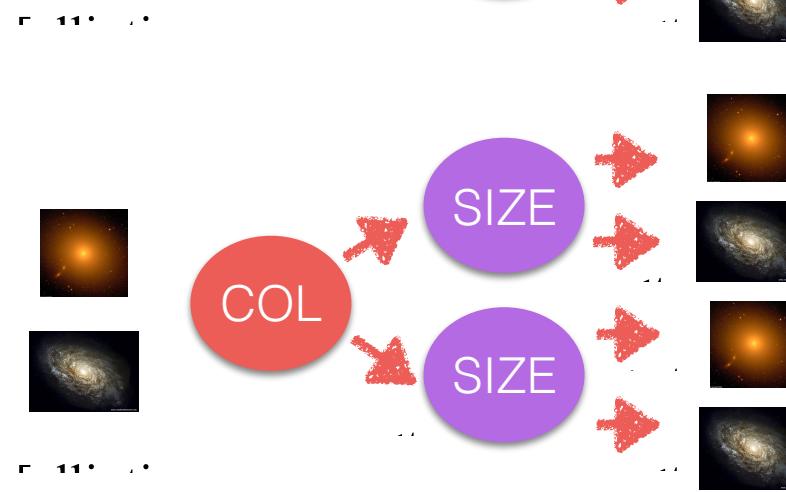


RANDOM FORESTS

ALL DATA



subset



RANDOM FORESTS

ONE KEY ADVANTAGE OF DECISION TREE ALGORITHMS IS THAT THEY ARE VERY EASY TO INTERPRET

ONE CAN EASILY DETERMINE THE MOST IMPORTANT FEATURES TO TAKE DECISIONS

| Rank | Property | AUC | Success label ^a |
|------|--------------------|-----------------|----------------------------|
| 1 | ALL | 0.9074 ± 0.0106 | Outstanding |
| 1 | CVD | 0.8559 ± 0.0039 | Excellent |
| 2 | M_{bulge} | 0.8335 ± 0.0060 | Excellent |
| 3 | B/T | 0.8267 ± 0.0028 | Excellent |
| 4 | M_{halo} | 0.7983 ± 0.0045 | Acceptable |
| 5 | M_* | 0.7819 ± 0.0025 | Acceptable |
| 6 | M_{disc} | 0.7124 ± 0.0016 | Acceptable |
| 7 | δ_5 | 0.5894 ± 0.0015 | Unacceptable |
| 8 | Re | 0.5599 ± 0.0013 | Unacceptable |

IMPORTANCE OF
PARAMETERS TO PREDICT
IF A GALAXY IS QUENCHED

PRACTICAL NOTE: *Python scikit learn*

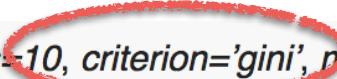
- All these different methods are standard and available in Python
- Very easy to use
- All info here

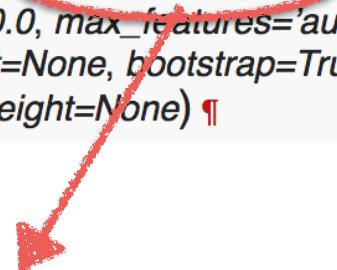
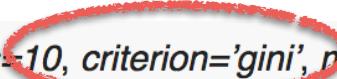
sklearn.ensemble.RandomForestClassifier

```
class sklearn.ensemble. RandomForestClassifier(n_estimators=10, criterion='gini', max_depth=None,  
min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto',  
max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, bootstrap=True, oob_score=False,  
n_jobs=1, random_state=None, verbose=0, warm_start=False, class_weight=None)  [source]
```

NUMBER OF RANDOM
TREES

sklearn.ensemble.RandomForestClassifier

```
class sklearn.ensemble. RandomForestClassifier (n_estimators=10, criterion='gini', max_depth=None,  
min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto',  
max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, bootstrap=True, oob_score=False,  
n_jobs=1, random_state=None, verbose=0, warm_start=False, class_weight=None)  [source]
```



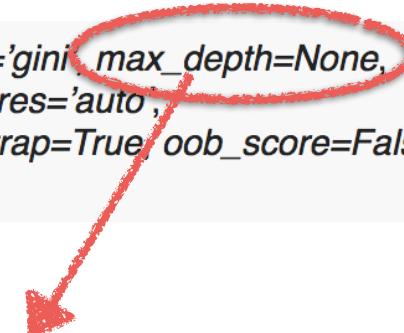
METRIC TO MINIMIZE

sklearn.ensemble.RandomForestClassifier

```
class sklearn.ensemble. RandomForestClassifier (n_estimators=10, criterion='gini', max_depth=None,  
min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto',  
max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, bootstrap=True, oob_score=False,  
n_jobs=1, random_state=None, verbose=0, warm_start=False, class_weight=None) ¶ [source]
```

MAXIMUM NUMBER OF SPLITS

sklearn.ensemble.RandomForestClassifier

```
class sklearn.ensemble.RandomForestClassifier(n_estimators=10, criterion='gini', max_depth=None,  
min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto',  
max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, bootstrap=True, oob_score=False,  
n_jobs=1, random_state=None, verbose=0, warm_start=False, class_weight=None)  [source]
```

MAXIMUM NUMBER OF SPLITS

```
>>> from sklearn.ensemble import RandomForestClassifier  
>>> from sklearn.datasets import make_classification  
>>>  
>>> X, y = make_classification(n_samples=1000, n_features=4,  
...                                n_informative=2, n_redundant=0,  
...                                random_state=0, shuffle=False)  
>>> clf = RandomForestClassifier(max_depth=2, random_state=0)  
>>> clf.fit(X, y)  
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',  
                      max_depth=2, max_features='auto', max_leaf_nodes=None,  
                      min_impurity_decrease=0.0, min_impurity_split=None,
```

DIFFERENT CLASSIFIERS ARE OBJECTS WITH METHODS TO
FIT, PREDICT ETC..

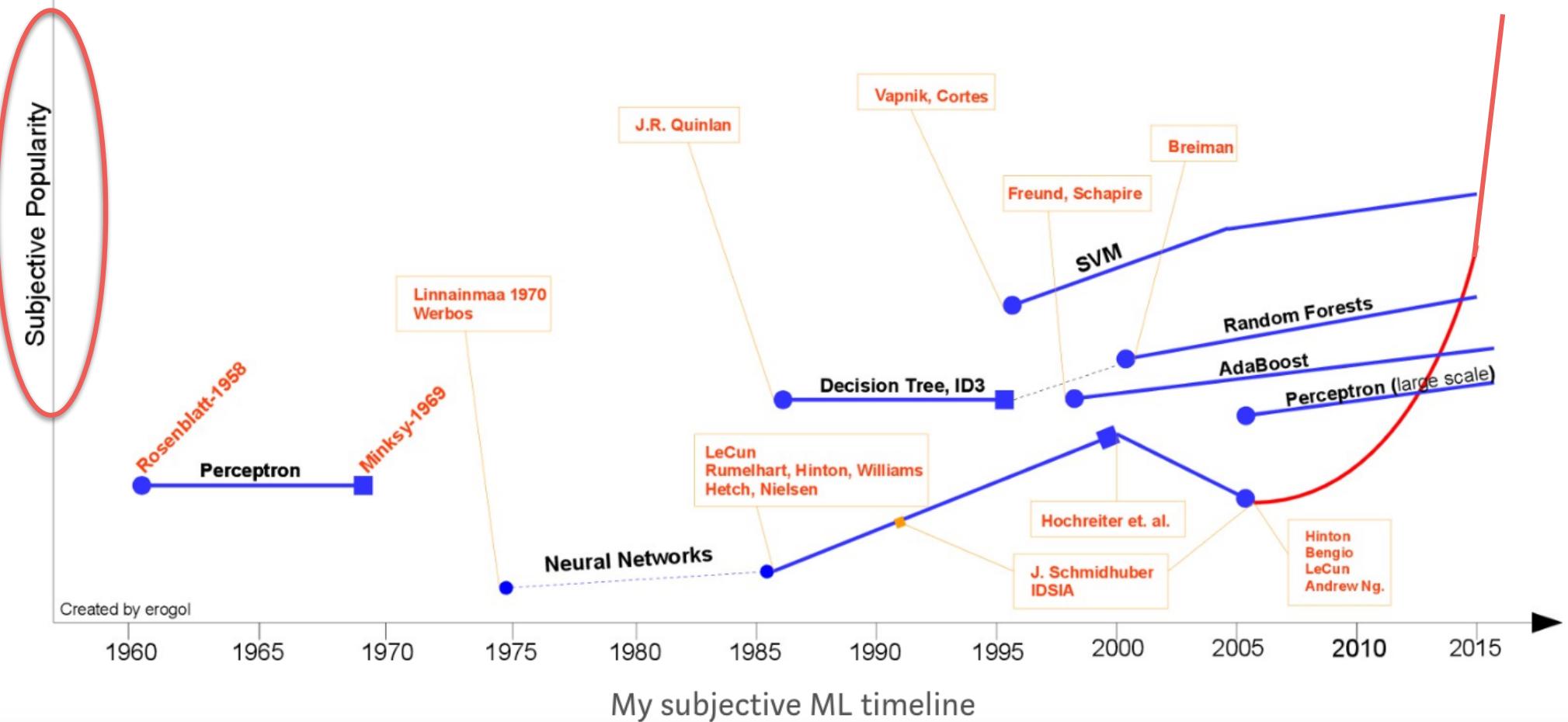
```
>>> print(clf.predict([[0, 0, 0, 0]]))  
[1]
```

HOW TO CHOOSE YOUR CLASSICAL CLASSIFIER?

NO RULE OF THUMB - REALLY DEPENDS ON APPLICATION

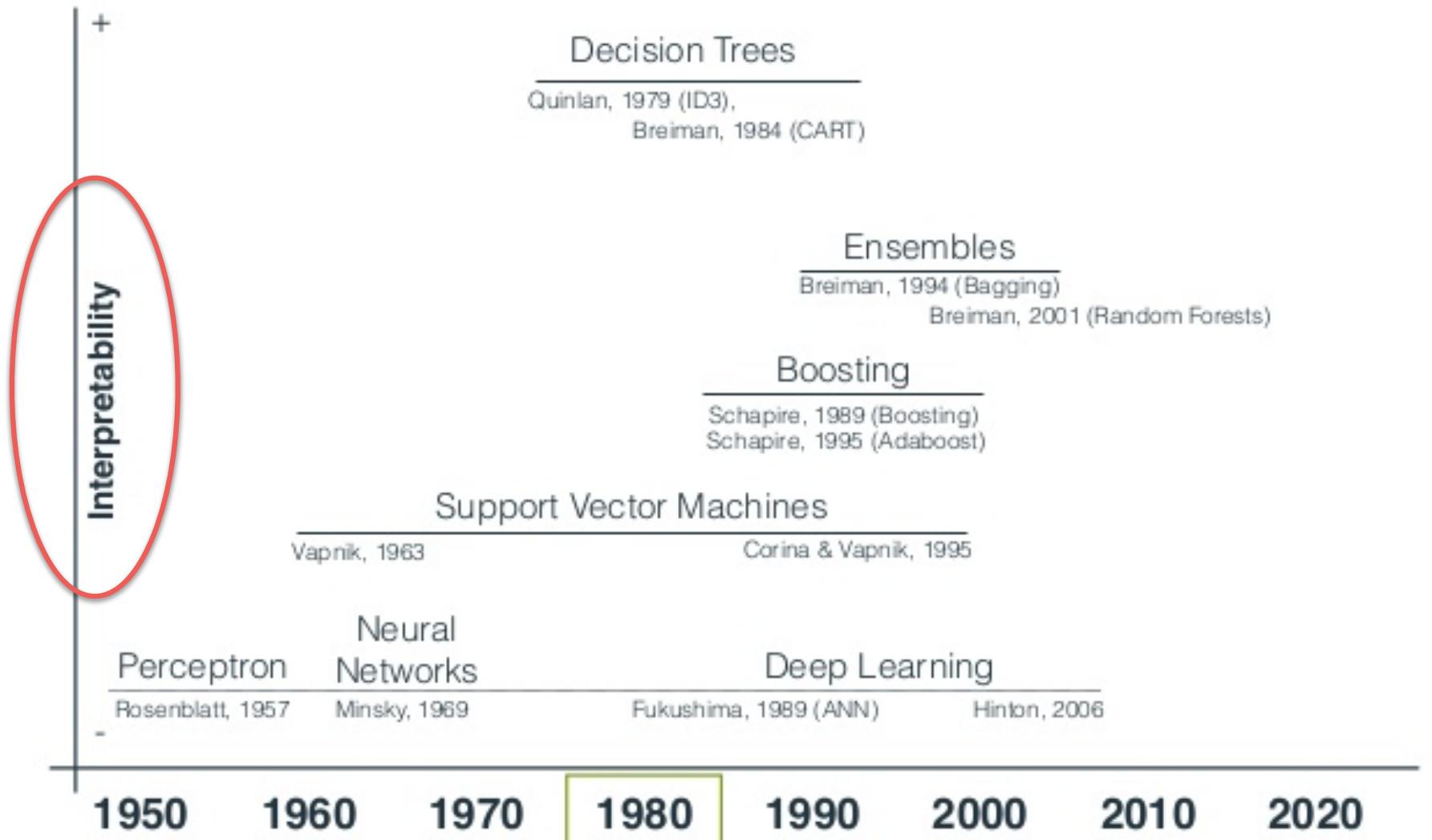
| ML METHOD | ++ | - | Python |
|------------------------------|---|---|---|
| CARTS / RANDOM FOREST | Easy to interpret (“White box”) Litte data preparation Both numerical + categorical | Over-complex trees Unstable Biased trees if some classes dominate | sklearn.ensemble.RandomForestClassifier sklearn.ensemble.RandomForestRegressor |
| SVM | Easy to interpret + Fast Kernel trick allows no linear problems | not very well suited to multi-class problems | sklearn.svm sklearn.svc |
| NN | seed of deep-learning very efficient with large amount of data as we will see | more difficult to interpret computing intensive | sklearn.neural_network.MLPClassifier sklearn.neural_network.MLPRegressor |

ALSO INFLUENCED BY “MAINSTREAM” TRENDS



Source

CAN DEPEND ON YOUR MAIN INTEREST



credit

HOW DO I KNOW THAT MY MACHINE LEARNING ALGORITHM IS WORKING?

- The way results are evaluated depends strongly on the type of problem
 - Binary Classification: ACC, ROC, P-C
 - Multi-Class: Confusion Matrix
 - Regression: RMSE, Bias, Scatter etc..

Evaluation of results [binary class.]

THE MOST STRAIGHTFORWARD WAY IS TO EVALUATE THE
TOTAL ACCURACY

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

true positives true negatives



MEASURES HOW MANY OBJECTS ARE CORRECTLY
CLASSIFIED

Evaluation of results [binary class.]

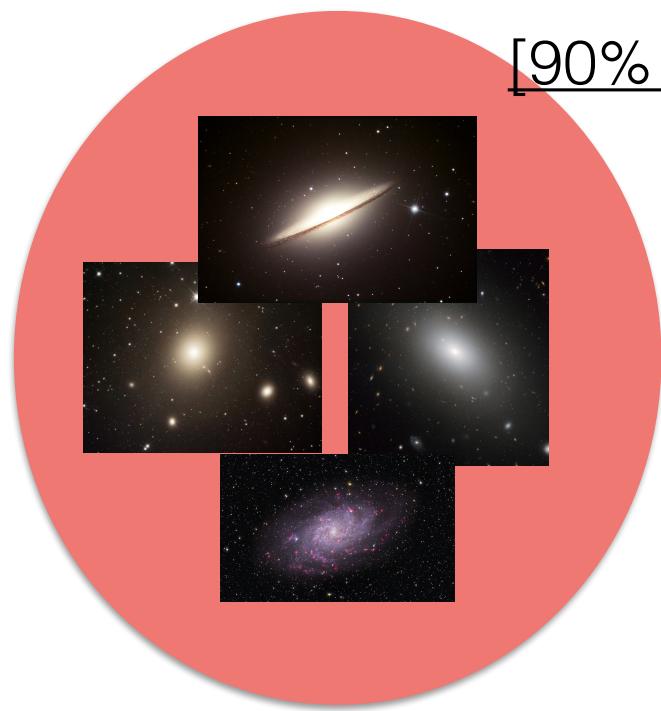
THE MOST STRAIGHTFORWARD WAY IS TO EVALUATE THE
TOTAL ACCURACY

NOT VERY
INFORMATIVE IF
UNBALANCED
CLASSES

$$ACC = \frac{\text{true positives} + \text{true negatives}}{TP + TN + FP + FN}$$

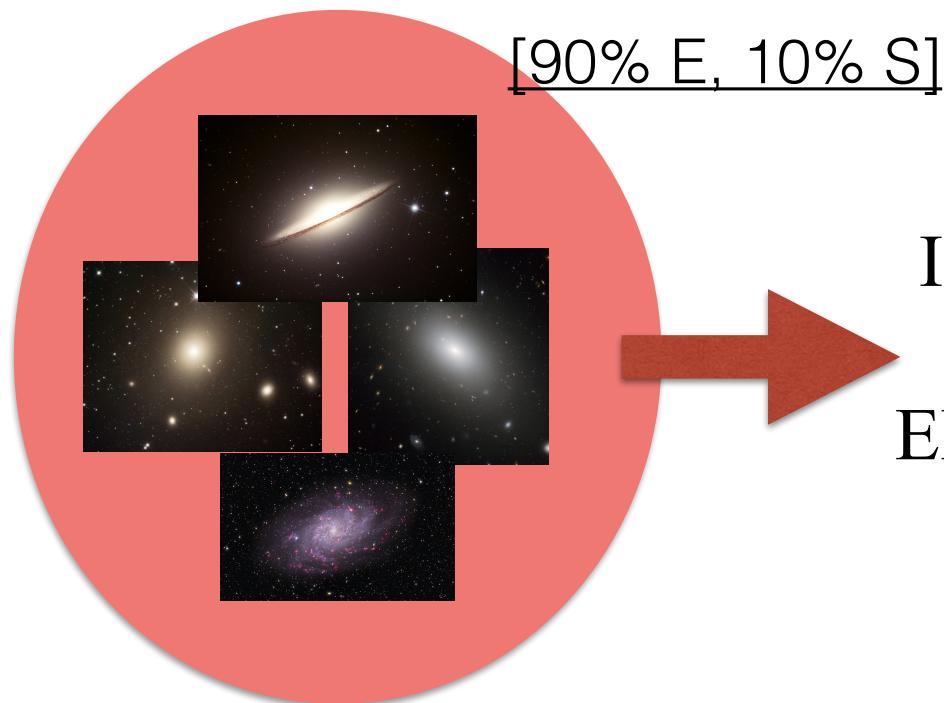
MEASURES HOW MANY OBJECTS ARE CORRECTLY
CLASSIFIED

IMAGINE AN EXTREME CASE WITH VERY UNBALANCED
DATA...



[90% E, 10% S]

IMAGINE AN EXTREME CASE WITH VERY UNBALANCED
DATA...



IF I SAY THAT ALL GALAXIES
IN THE UNIVERSE ARE
ELLIPTICALS I WILL BE RIGHT
90% OF THE TIMES

Evaluation of results [binary class.]

THE ROC CURVE (Receiver Operating Characteristic)

IT IS BASED ON TWO VERY SIMPLE PARAMETERS

1.
$$TPR = \frac{TP}{TP + FN}$$
 [Also called Sensitivity, Completeness]

“Fraction of positive examples classified correctly”

Evaluation of results [binary class.]

THE ROC CURVE (Receiver Operating Characteristic)

IT IS BASED ON TWO VERY SIMPLE PARAMETERS

1.

$$TPR = \frac{TP}{TP + FN}$$

TRUE POSITIVE RATE
[Also called Sensitivity, Completeness]

“Fraction of positive examples classified correctly”

2.

$$FPR = \frac{FP}{FP + TN}$$

FALSE POSITIVE RATE
[Also called Specificity, Contamination]

“Fraction of negative examples classified as positive”

Evaluation of results [binary class.]

- YOU WANT THIS TO BE AS BIG AS POSSIBLE
- Over Operating Characteristic)

IT IS BASED ON TWO VERY SIMPLE PARAMETERS

1.

$$TPR = \frac{TP}{TP + FN}$$

TRUE POSITIVE RATE
[Completeness]

YOU WANT THIS TO BE AS SMALL AS POSSIBLE

“Fraction of positive examples classified correctly”

2.

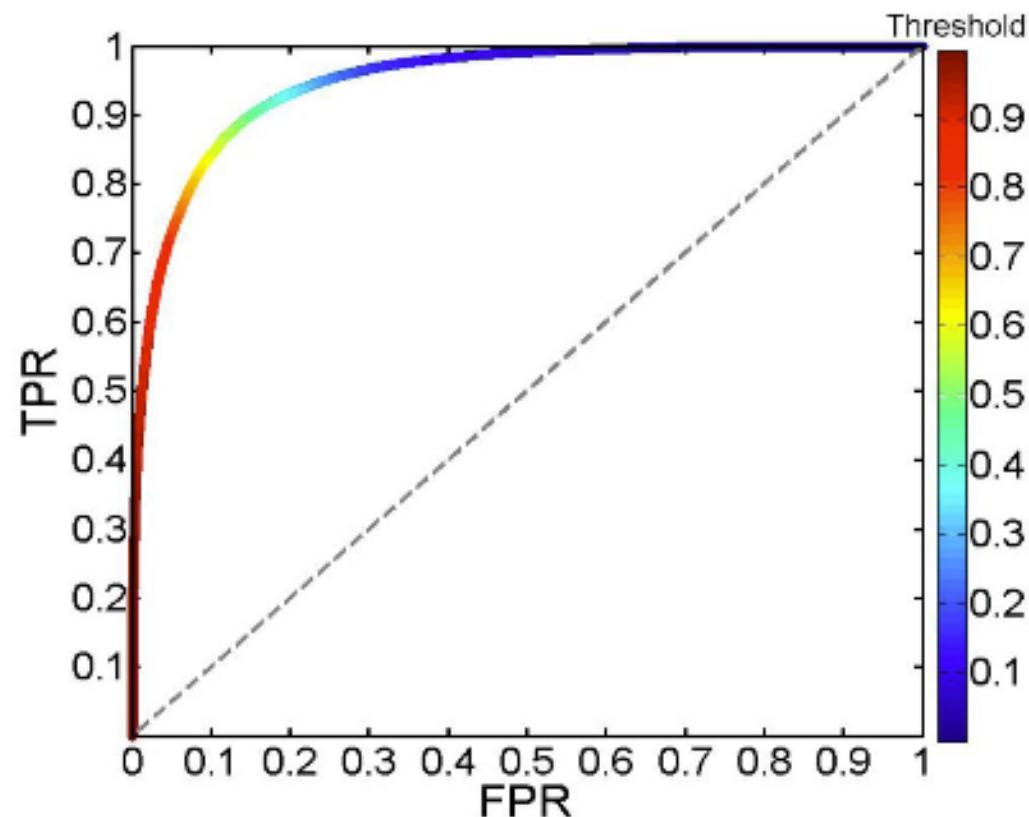
$$FPR = \frac{FP}{FP + TN}$$

FALSE POSITIVE RATE
[Also called Specificity, Contamination]

“Fraction of negative examples classified as positive”

IF YOUR CLASSIFIER OUTPUTS A SORT OF PROBABILITY,
TPR AND FPR CAN BE PLOTTED ONE AGAINST THE OTHER

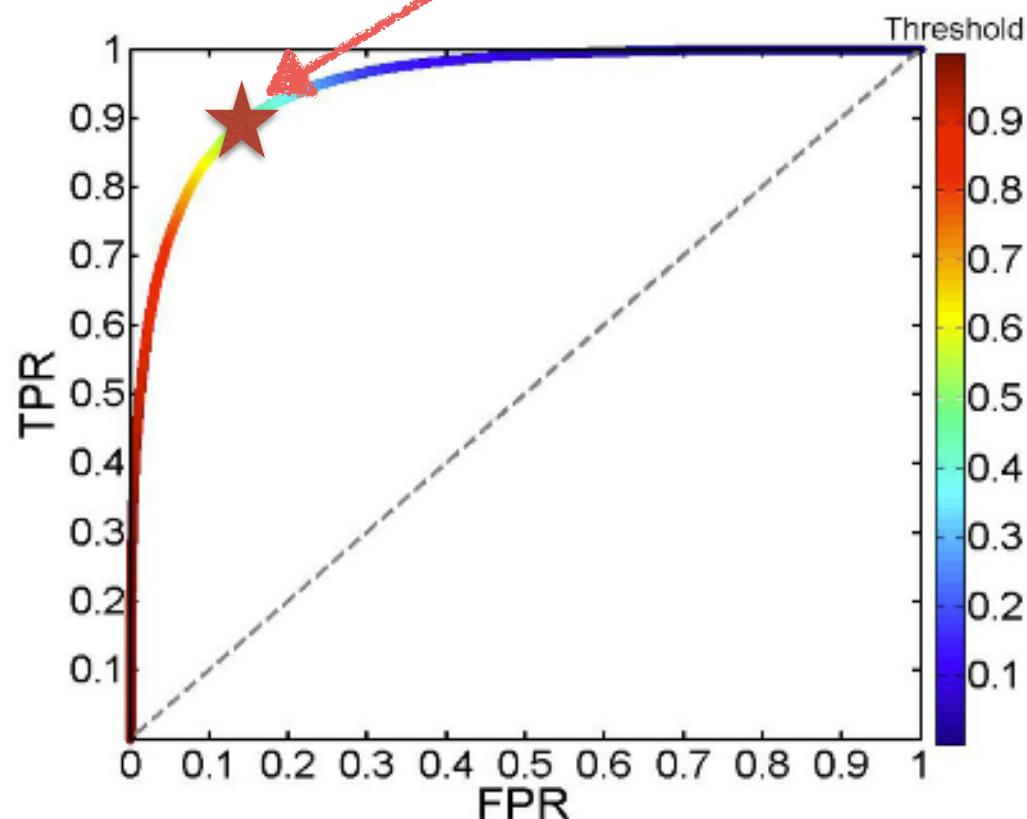
ROC CURVE



IF YOUR CLASSIFIER OUTPUTS A SORT OF PROBABILITY,
TPR AND FPR CAN BE PLOTTED ONE AGAINST THE OTHER

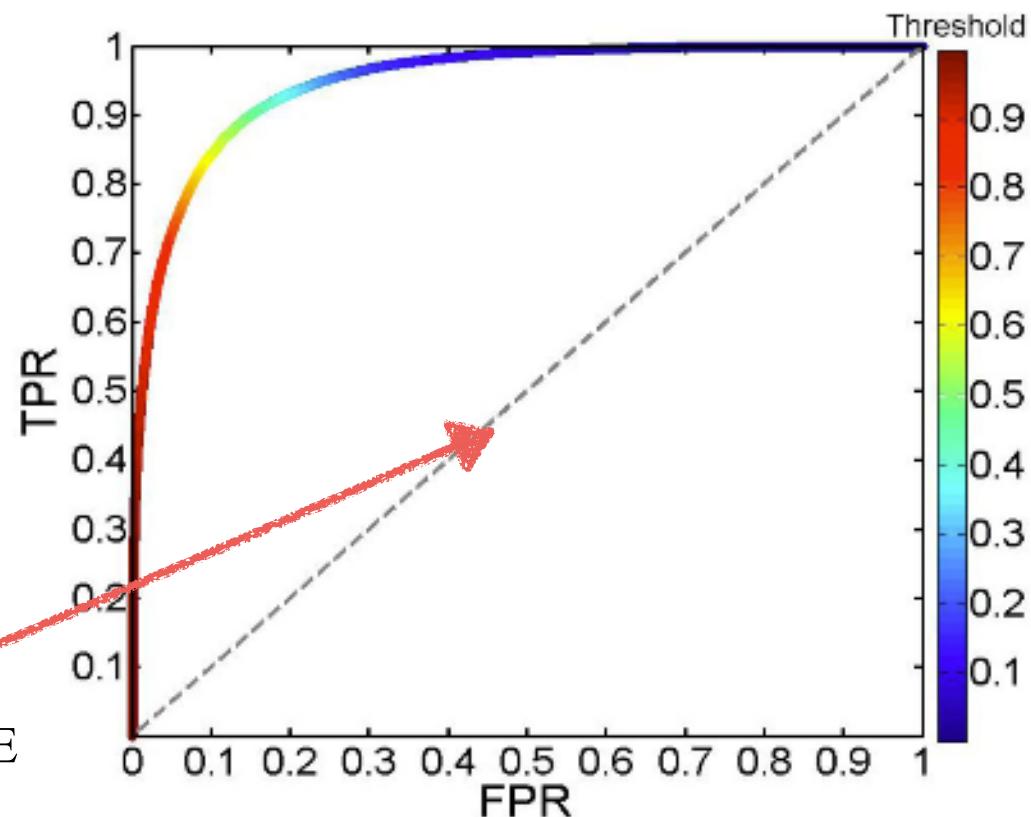
EACH POINT HERE SHOWS THE VALUES OF TPR AND
FPR FOR A GIVEN THRESHOLD

ROC CURVE



IF YOUR CLASSIFIER OUTPUTS A SORT OF PROBABILITY,
TPR AND FPR CAN BE PLOTTED ONE AGAINST THE OTHER

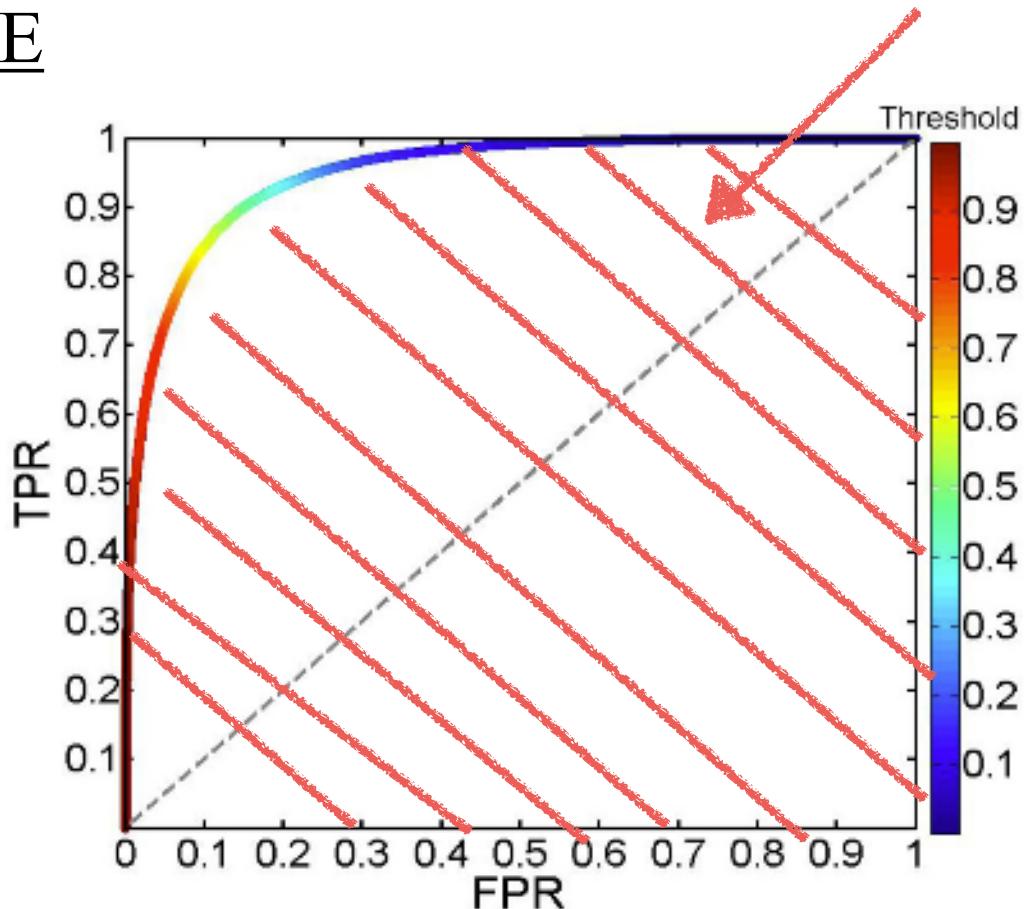
ROC CURVE



THE ONE-TO-ONE
LINE IS A
RANDOM
CLASSIFICATION

IF YOUR CLASSIFIER OUTPUTS A SORT OF PROBABILITY,
TPR AND FPR CAN BE PLOTTED ONE AGAINST THE OTHER

ROC CURVE



THE AREA UNDER THE
CURVE AUC ALSO
MEASURES THE
GLOBAL ACCURACY

SUMMARY OF DIFFERENT ACCURACY TRACERS

| | | True condition | | | |
|---------------------|--|---|--|---|--|
| Total population | Condition positive | Condition negative | Prevalence = $\frac{\sum \text{Condition positive}}{\sum \text{Total population}}$ | Accuracy (ACC) = $\frac{\sum \text{True positive} + \sum \text{True negative}}{\sum \text{Total population}}$ | |
| Predicted condition | Predicted condition positive | True positive, Power | False positive, Type I error | Positive predictive value (PPV), Precision = $\frac{\sum \text{True positive}}{\sum \text{Predicted condition positive}}$ | False discovery rate (FDR) = $\frac{\sum \text{False positive}}{\sum \text{Predicted condition positive}}$ |
| | Predicted condition negative | False negative, Type II error | True negative | False omission rate (FOR) = $\frac{\sum \text{False negative}}{\sum \text{Predicted condition negative}}$ | Negative predictive value (NPV) = $\frac{\sum \text{True negative}}{\sum \text{Predicted condition negative}}$ |
| | True positive rate (TPR), Recall, Sensitivity, probability of detection = $\frac{\sum \text{True positive}}{\sum \text{Condition positive}}$ | False positive rate (FPR), Fall-out, probability of false alarm = $\frac{\sum \text{False positive}}{\sum \text{Condition negative}}$ | Positive likelihood ratio (LR+) = $\frac{\text{TPR}}{\text{FPR}}$ | Diagnostic odds ratio (DOR) = $\frac{\text{LR+}}{\text{LR-}}$ | $F_1 \text{ score} = \frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}$ |
| | False negative rate (FNR), Miss rate = $\frac{\sum \text{False negative}}{\sum \text{Condition positive}}$ | True negative rate (TNR), Specificity (SPC) = $\frac{\sum \text{True negative}}{\sum \text{Condition negative}}$ | Negative likelihood ratio (LR-) = $\frac{\text{FNR}}{\text{TNR}}$ | | |

SOURCE

SUMMARY OF DIFFERENT ACCURACY TRACERS

| | | True condition | | | |
|---------------------|---|---|--|--|--|
| Total population | Condition positive | Condition negative | Prevalence = $\frac{\sum \text{Condition positive}}{\sum \text{Total population}}$ | Accuracy (ACC) = $\frac{\sum \text{True positive} + \sum \text{True negative}}{\sum \text{Total population}}$ | |
| Predicted condition | Predicted condition positive | True positive, Power | False positive, Type I error | Positive predictive value (PPV), Precision = $\frac{\sum \text{True positive}}{\sum \text{Predicted condition positive}}$ | False discovery rate (FDR) = $\frac{\sum \text{False positive}}{\sum \text{Predicted condition positive}}$ |
| | Predicted condition negative | False negative, Type II error | True negative | False omission rate (FOR) = $\frac{\sum \text{False negative}}{\sum \text{Predicted condition negative}}$ | Negative predictive value (NPV) = $\frac{\sum \text{True negative}}{\sum \text{Predicted condition negative}}$ |
| | True positive rate (TPR), Recall, Sensitivity, probability of detection = $\frac{\sum \text{True positive}}{\sum \text{Condition positive}}$ | False positive rate (FPR), Fall-out, probability of false alarm $= \frac{\sum \text{False positive}}{\sum \text{Condition negative}}$ | Positive likelihood ratio (LR+) = $\frac{\text{TPR}}{\text{FPR}}$ | Diagnostic odds ratio (DOR) = $\frac{\text{LR+}}{\text{LR-}}$ | $F_1 \text{ score} = \frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}$ |
| | False negative rate (FNR), Miss rate $= \frac{\sum \text{False negative}}{\sum \text{Condition positive}}$ | True negative rate (TNR), Specificity (SPC) $= \frac{\sum \text{True negative}}{\sum \text{Condition negative}}$ | Negative likelihood ratio (LR-) = $\frac{\text{FNR}}{\text{TNR}}$ | | |

THE F1 SCORE
COMBINES BOTH INFORMATIONS IN ONE VALUE

[SOURCE](#)

ALL THESE ARE INCLUDED IN SKLEARN

AND ARE VETY EASY TO USE. NO NEED OF CODING THEM AGAIN!

```
sklearn.metrics. precision_recall_curve (y_true, probas_pred, pos_label=None, sample_weight=None) ¶
```

[\[source\]](#)

Compute precision-recall pairs for different probability thresholds

Note: this implementation is restricted to the binary classification task.

The precision is the ratio $\text{tp} / (\text{tp} + \text{fp})$ where tp is the number of true positives and fp the number of false positives. The precision is intuitively the ability of the classifier not to label as positive a sample that is negative.

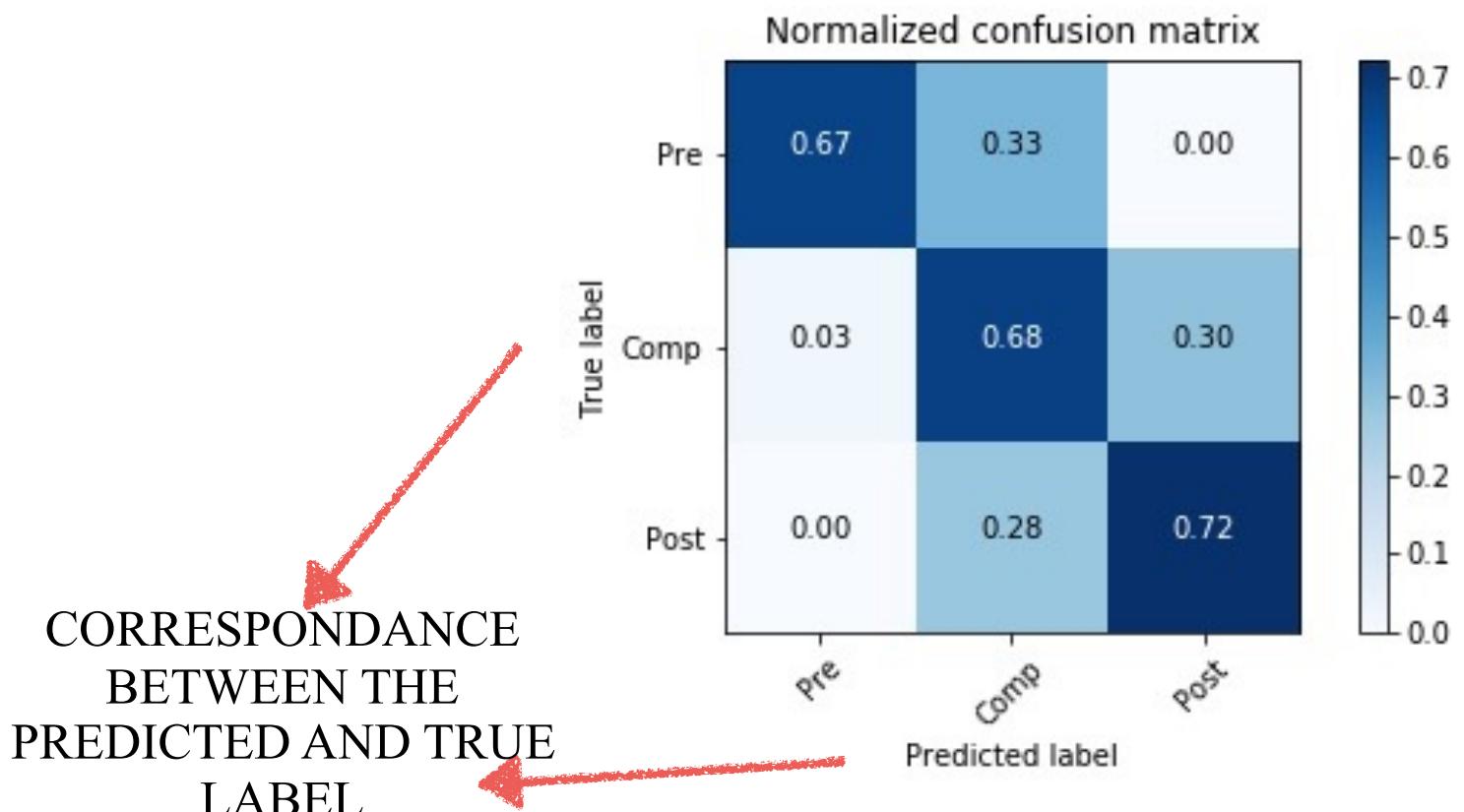
The recall is the ratio $\text{tp} / (\text{tp} + \text{fn})$ where tp is the number of true positives and fn the number of false negatives. The recall is intuitively the ability of the classifier to find all the positive samples.

The last precision and recall values are 1. and 0. respectively and do not have a corresponding threshold. This ensures that the graph starts on the x axis.

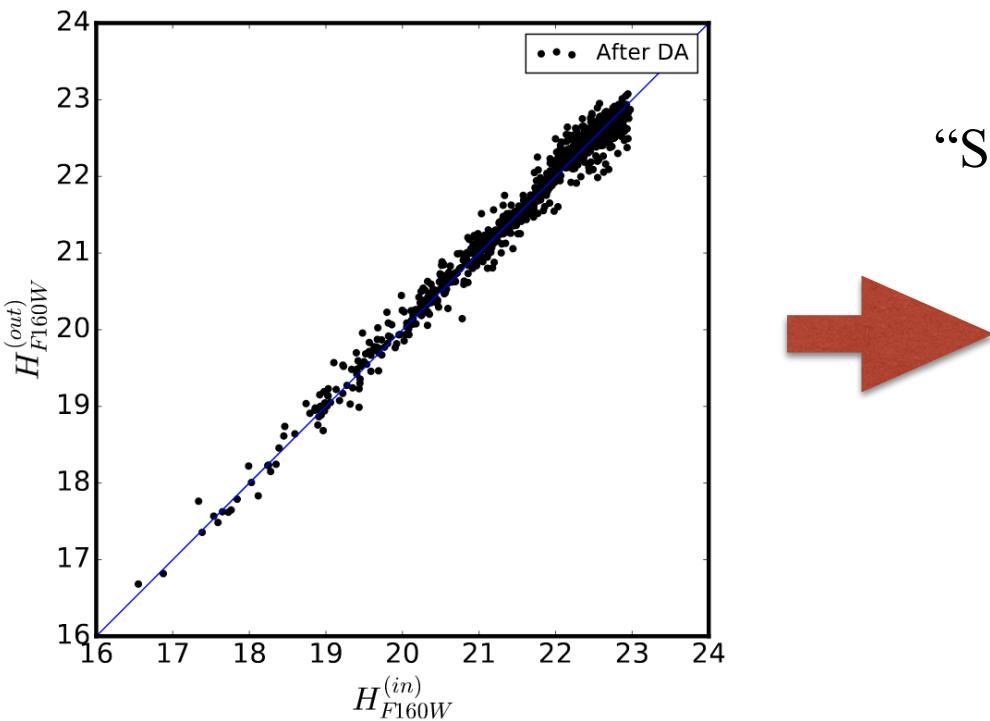
Evaluation of results

[multi-class]

CONFUSION MATRIX

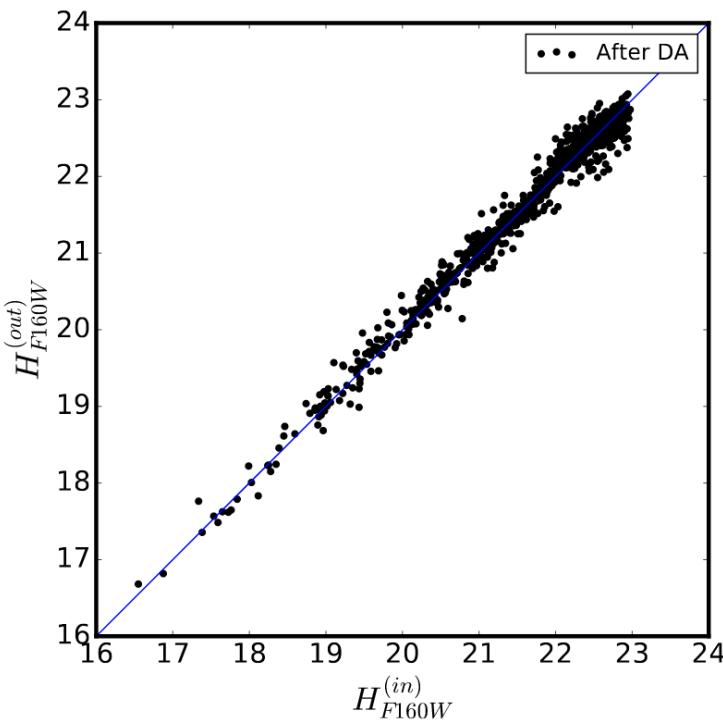


Evaluation of results [regression]

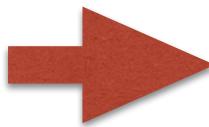


FOR REGRESSIONS, SIMPLY USE
“STANDARD ACCURACY MEASUREMENTS

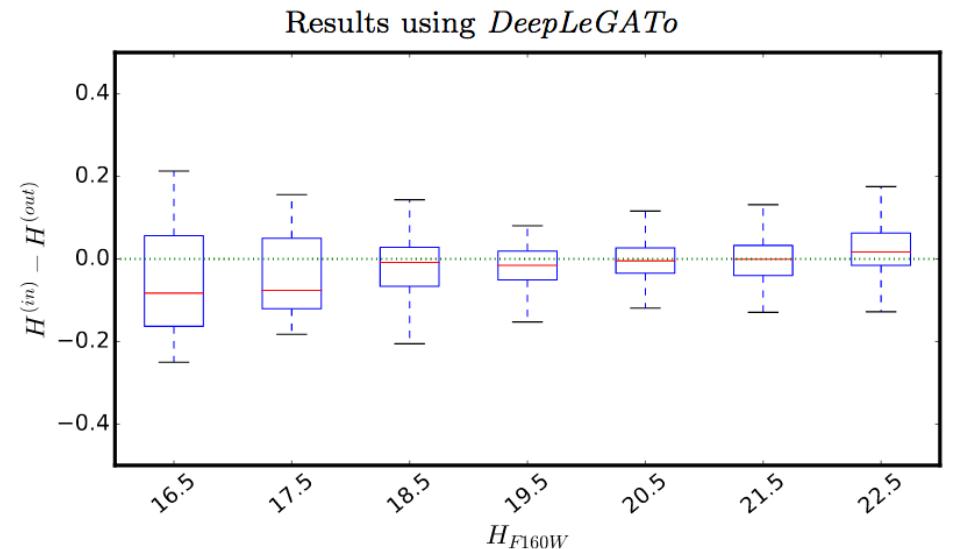
Evaluation of results [regression]



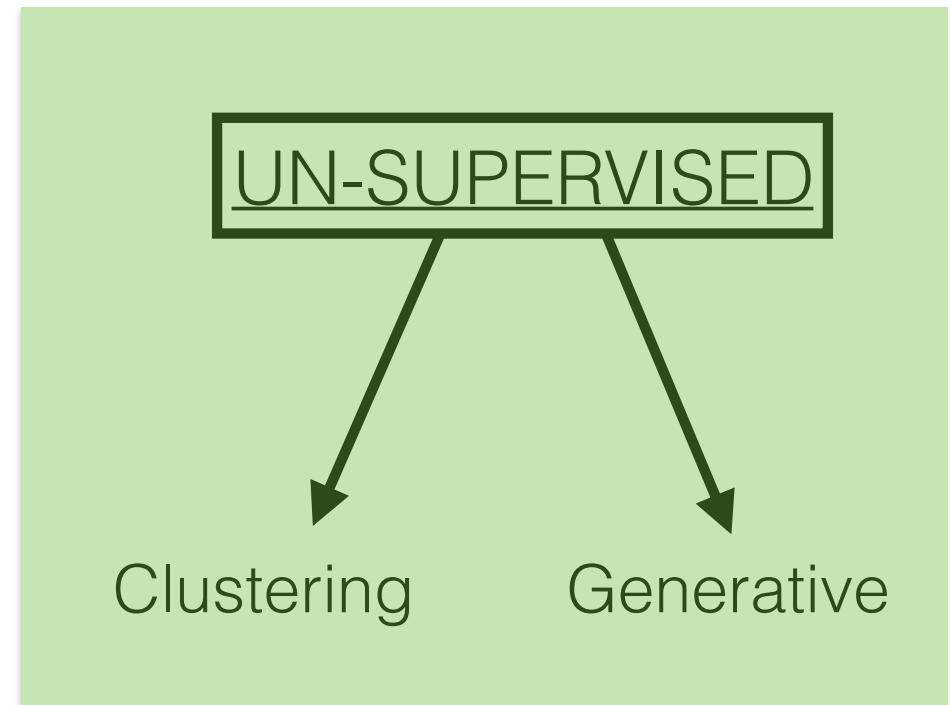
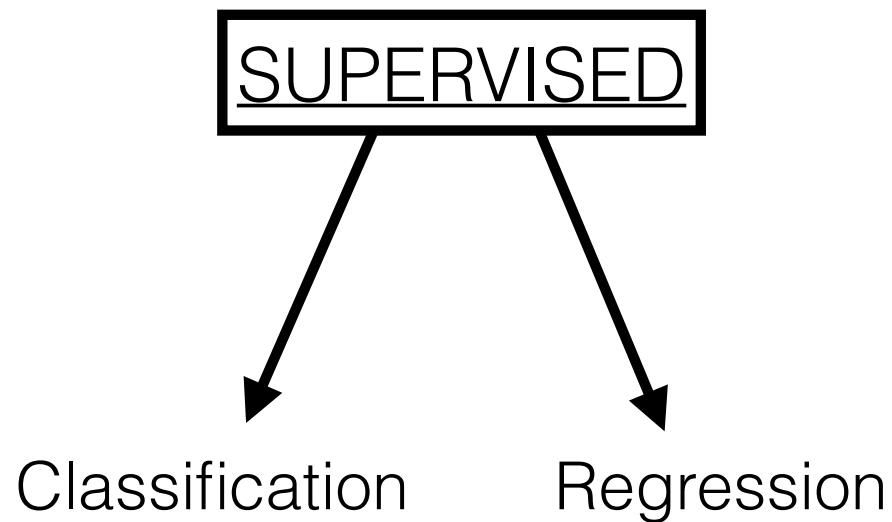
FOR REGRESSIONS, SIMPLY USE
“STANDARD ACCURACY MEASUREMENTS”



BIAS, SCATTER ... YOU KNOW!

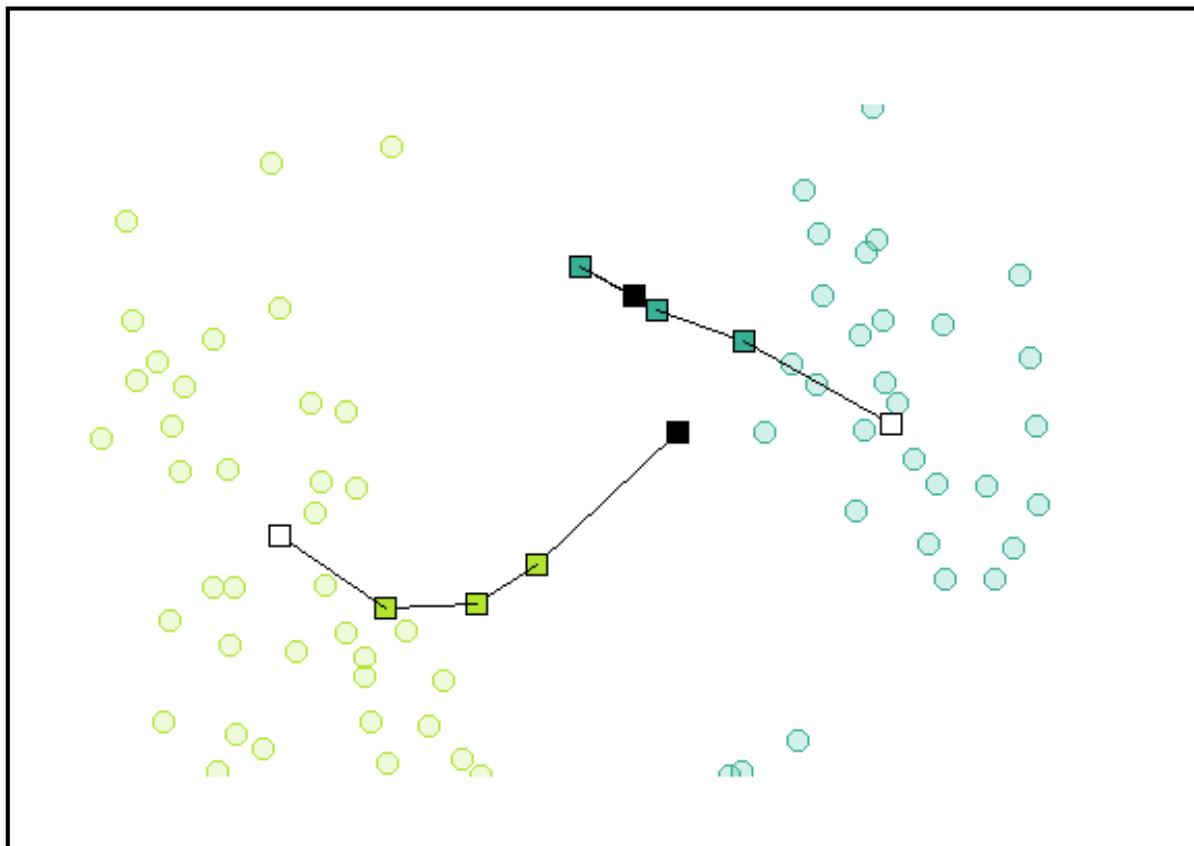


WHAT DOES MACHINE LEARNING DO?



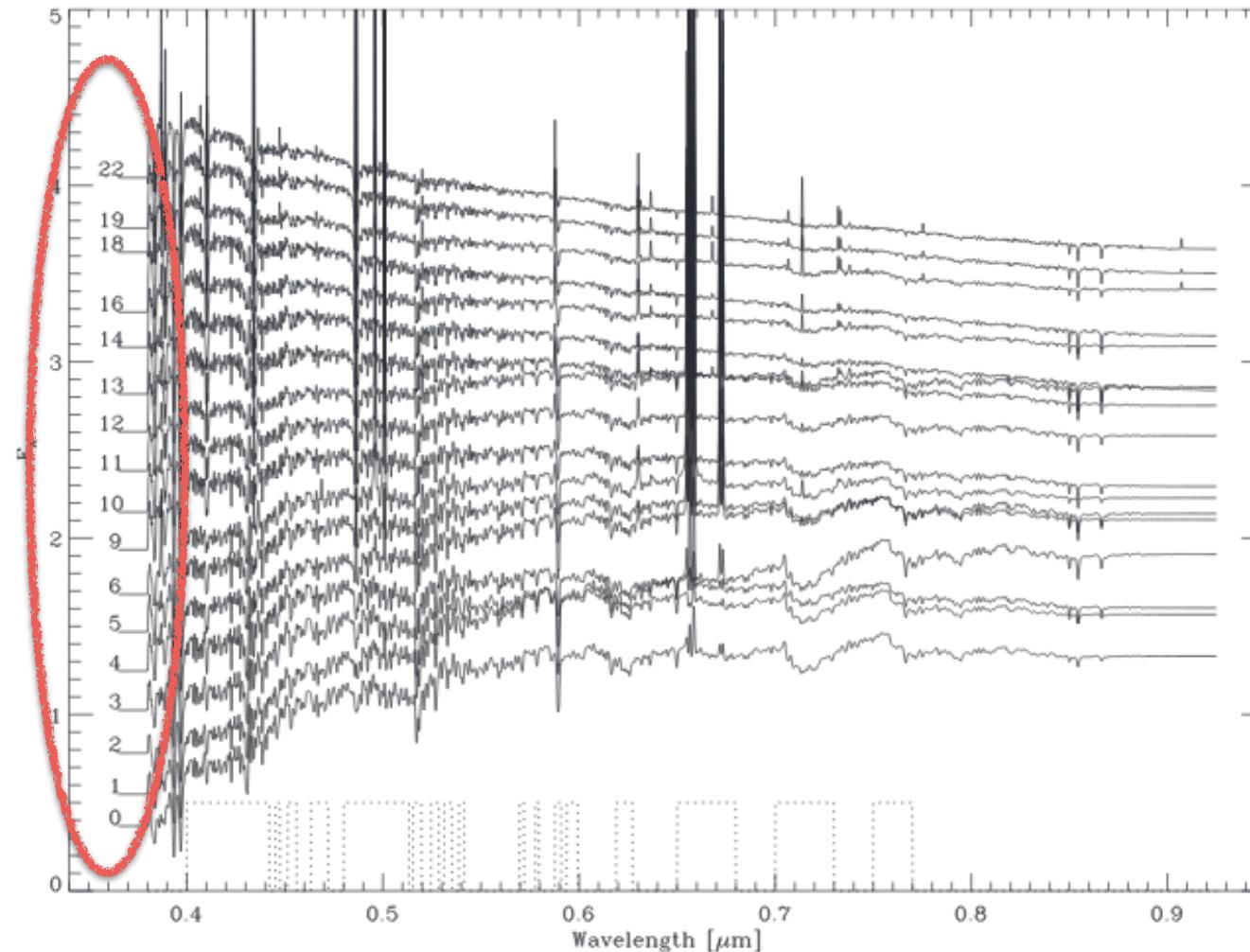
MOST DATA ARE ACTUALLY UNLABELLED!

K-MEANS CLUSTERING



K-MEANS APPLIED TO ALL SDSS SPECTRA

NUMBER
OF CLASSES
FOUND



KEY ISSUES WITH K-MEANS CLUSTERING

NUMBER OF CLASSES / CENTERS?

ARE THEY SIGNIFICANT?

INITIALIZATION

WHERE DO I CHOOSE THE CLUSTER CENTERS?