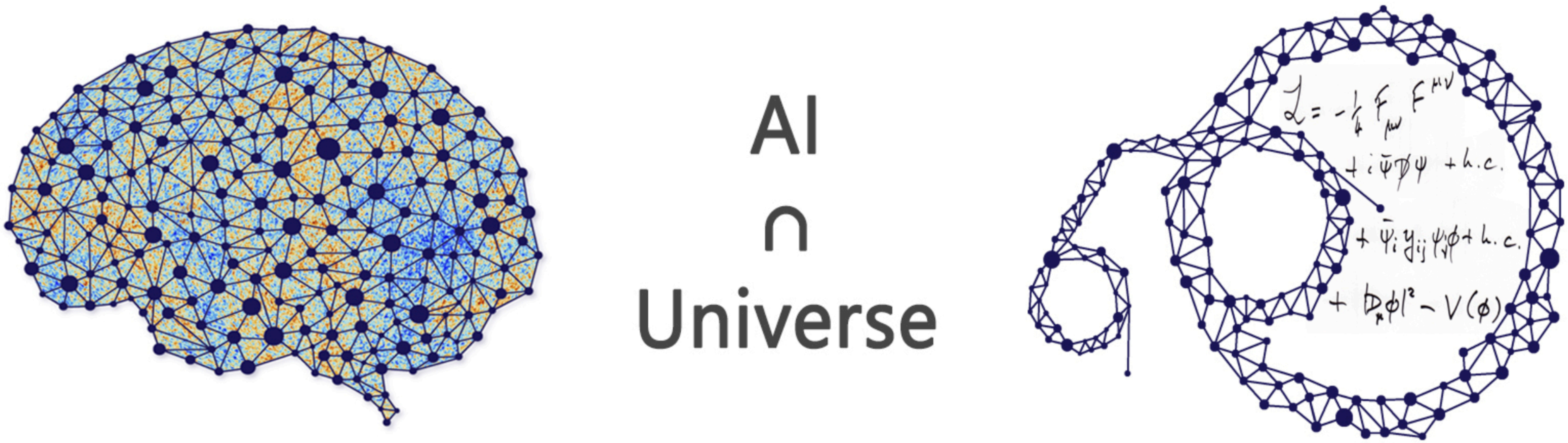
PHY 835: Machine Learning in Physics

Lecture 16: Decision Trees & kNN

March 14, 2024



Gary Shiu

Outline for today

- Many ML models are designed to solve **classification** or **regression** problems.

- Simple Classifiers:

    - Decision Trees

    - kNN: Finding neighbors

- Reference: “Machine Learning for Physics and Astronomy” by Viviana Acquaviva, Princeton University Press, Chapter 2.

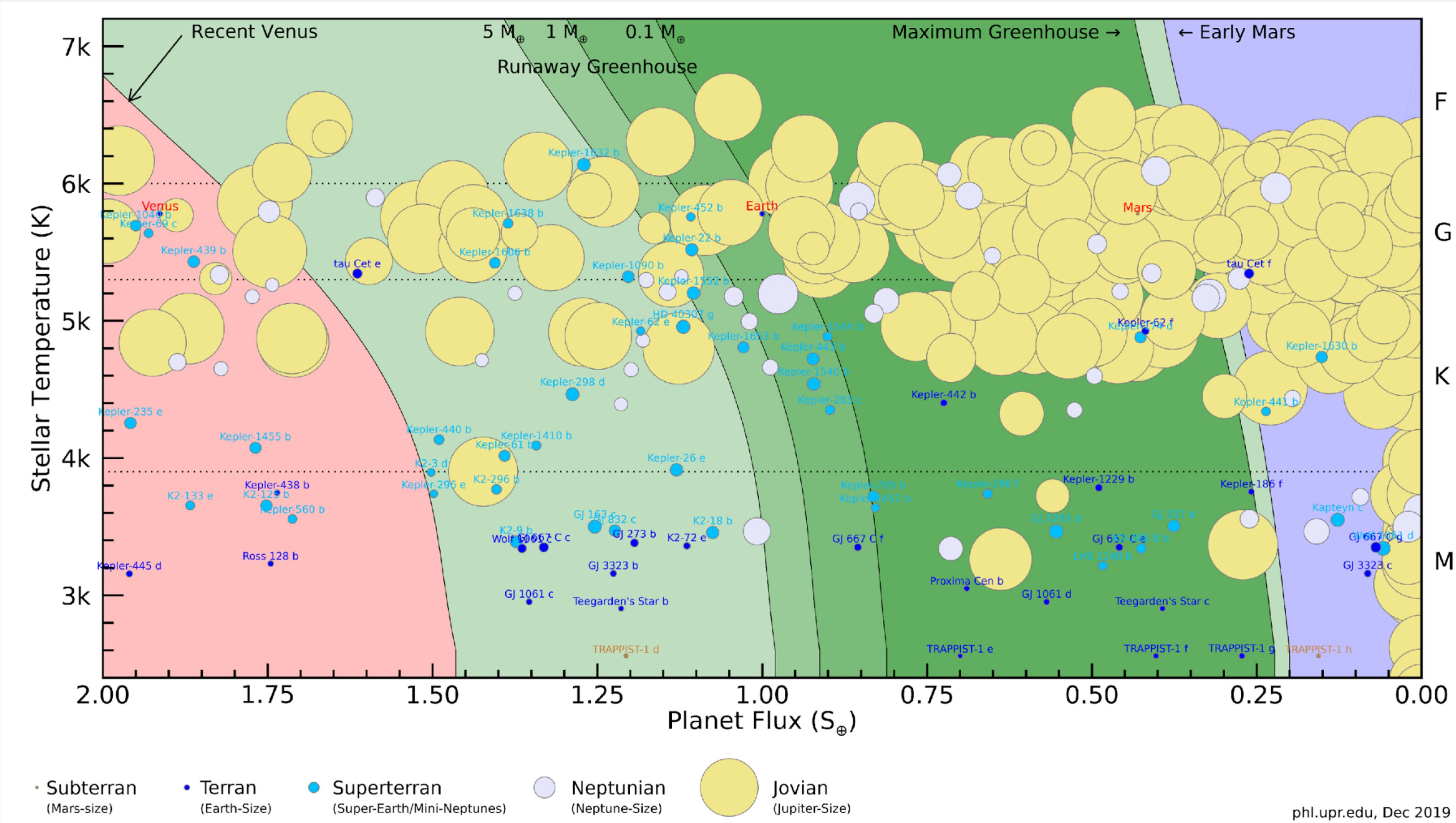
We can now design a decision tree
in a (astro)physics context.

OUR FIRST EXAMPLE WILL BE A SUPERVISED CLASSIFICATION PROBLEM,
IN WHICH WE ATTEMPT TO PREDICT
IF A PLANET IS HABITABLE, BASED ON ITS DISTANCE FROM PARENT STAR, MASS OF PARENT STAR, AND ORBITAL PERIOD.

Search for Habitable Planets

- **The problem:** search for intelligent life beyond Earth.
- More than 5000 exoplanets: https://exoplanets.nasa.gov/alien-worlds/historic-timeline/#first-transiting-exoplanet-observed
- Finding habitable planets (density and temperature conditions etc that are compatible with the development of life).
- Planet Habitability Laboratory website: https://phl.upr.edu/projects/habitable-exoplanets-catalog contains data for thousands of planets and collects a variety of features.
- We consider a learning set composed of 18 instances and their 3 features.

DATA FROM THE PLANET HABITABILITY LAB AT ARECIBO OBSERVATORY



Name	Stellar Mass ($M_\odot$)	Orbital Period (days)	Distance (AU)	Habitable?
Kepler-736 b	0.86	3.60	0.0437	0
Kepler-636 b	0.85	16.08	0.1180	0
Kepler-887 c	1.19	7.64	0.0804	0
Kepler-442 b	0.61	112.30	0.4093	1
Kepler-772 b	0.98	12.99	0.1074	0
Teegarden's Star b	0.09	4.91	0.0252	1
K2-116 b	0.69	4.66	0.0481	0
GJ 1061 c	0.12	6.69	0.035	1
HD 68402 b	1.12	1103	2.1810	0
Kepler-1544 b	0.81	168.81	0.5571	1
Kepler-296 e	0.5	34.14	0.1782	1
Kepler-705 b	0.53	56.06	0.2319	1
Kepler-445 c	0.18	4.87	0.0317	0
HD 104067 b	0.62	55.81	0.26	0
GJ 4276 b	0.41	13.35	0.0876	0
Kepler-296 f	0.5	63.34	0.2689	1
Kepler-63 b	0.98	9.43	0.0881	0
GJ 3293 d	0.42	48.13	0.1953	1

Table 2.1: Learning set for the habitable planets problem.

outlier

Name	Stellar Mass ($M_\odot$)	Orbital Period (days)	Distance (AU)	Habitable?
Kepler-736 b	0.86	3.60	0.0437	0
Kepler-636 b	0.85	16.08	0.1180	0
Kepler-887 c	1.19	7.64	0.0804	0
Kepler-442 b	0.61	112.30	0.4093	1
Kepler-772 b	0.98	12.99	0.1074	0
Teegarden's Star b	0.09	4.91	0.0252	1
K2-116 b	0.69	4.66	0.0481	0
GJ 1061 c	0.12	6.69	0.035	1
HD 68402 b	1.12	1103	2.1810	0
Kepler-1544 b	0.81	168.81	0.5571	1
Kepler-296 e	0.5	34.14	0.1782	1
Kepler-705 b	0.53	56.06	0.2319	1
Kepler-445 c	0.18	4.87	0.0317	0
HD 104067 b	0.62	55.81	0.26	0
GJ 4276 b	0.41	13.35	0.0876	0
Kepler-296 f	0.5	63.34	0.2689	1
Kepler-63 b	0.98	9.43	0.0881	0
GJ 3293 d	0.42	48.13	0.1953	1

Table 2.1: Learning set for the habitable planets problem.

outlier

Name	Stellar Mass ($M_\odot$)	Orbital Period (days)	Distance (AU)	Habitable?
Kepler-736 b	0.86	3.60	0.0437	0
Kepler-636 b	0.85	16.08	0.1180	0
Kepler-887 c	1.19	7.64	0.0804	0
Kepler-442 b	0.61	112.30	0.4093	1
Kepler-772 b	0.98	12.99	0.1074	0
Teegarden's Star b	0.09	4.91	0.0252	1
K2-116 b	0.69	4.66	0.0481	0
GJ 1061 c	0.12	6.69	0.035	1
HD 68402 b	1.12	1103	2.1810	0
Kepler-1544 b	0.81	168.81	0.5571	1
Kepler-296 e	0.5	34.14	0.1782	1
Kepler-705 b	0.53	56.06	0.2319	1
Kepler-445 c	0.18	4.87	0.0317	0
HD 104067 b	0.62	55.81	0.26	0
GJ 4276 b	0.41	13.35	0.0876	0
Kepler-296 f	0.5	63.34	0.2689	1
Kepler-63 b	0.98	9.43	0.0881	0
GJ 3293 d	0.42	48.13	0.1953	1

Table 2.1: Learning set for the habitable planets problem.

outlier

NAME	Stellar Mass ($M_\odot$)	Orbital Period (days)	Distance (AU)	Habitable?
Kepler-736 b	0.86	3.60	0.0437	0
Kepler-636 b	0.85	16.08	0.1180	0
Kepler-887 c	1.19	7.64	0.0804	0
Kepler-442 b	0.61	112.30	0.4093	1
Kepler-772 b	0.98	12.99	0.1074	0
Teegarden's Star b	0.09	4.91	0.0252	1
K2-116 b	0.69	4.66	0.0481	0
GJ 1061 c	0.12	6.69	0.035	1
HD 68402 b	1.12	1103	2.1810	0
Kepler-1544 b	0.81	168.81	0.5571	1
Kepler-296 e	0.5	34.14	0.1782	1
Kepler-705 b	0.53	56.06	0.2319	1
Kepler-445 c	0.18	4.87	0.0317	0
HD 104067 b	0.62	55.81	0.26	0
GJ 4276 b	0.41	13.35	0.0876	0
Kepler-296 f	0.5	63.34	0.2689	1
Kepler-63 b	0.98	9.43	0.0881	0
GJ 3293 d	0.42	48.13	0.1953	1

Table 2.1: Learning set for the habitable planets problem.

Seems reasonable from Kepler's laws

Name	Stellar Mass ($M_\odot$)	Orbital Period (days)	Distance (AU)	Habitable?
Kepler-736 b	0.86	3.60	0.0437	0
Kepler-636 b	0.85	16.08	0.1180	0
Kepler-887 c	1.19	7.64	0.0804	0
Kepler-442 b	0.61	112.30	0.4093	1
Kepler-772 b	0.98	12.99	0.1074	0
Teegarden's Star b	0.09	4.91	0.0252	1
K2-116 b	0.69	4.66	0.0481	0
GJ 1061 c	0.12	6.69	0.035	1
outlier HD 68402 b	1.12	1103	2.1810	0
Kepler-1544 b	0.81	168.81	0.5571	1
Kepler-296 e	0.5	34.14	0.1782	1
Kepler-705 b	0.53	56.06	0.2319	1
Kepler-445 c	0.18	4.87	0.0317	0
HD 104067 b	0.62	55.81	0.26	0
GJ 4276 b	0.41	13.35	0.0876	0
Kepler-296 f	0.5	63.34	0.2689	1
Kepler-63 b	0.98	9.43	0.0881	0
GJ 3293 d	0.42	48.13	0.1953	1

Table 2.1: Learning set for the habitable planets problem.

Comments on the dataset

- Well balanced: 10 examples in one category (not habitable) and 8 in the other (habitable).
- A factor that determines whether a planet is habitable is temperature, which likely depends on the energy it receives from its parent star.
- The planet's temperature therefore depends on the star's luminosity and its **distance** from the planet.
- **Mass** of a star is a decent tracer of its luminosity (as is the case for main sequence stars).
- Reality is more complicated: the energy budget of each planet also depends on other features, e.g., properties of its atmosphere, & whether it has an internal energy source.
- Mass/luminosity relationship is monotonic only for main sequence stars, which make up only about 90% of the total.

Predicting planet habitability

- On canvas, under DT-kNN-Notebooks, you will find a Jupyter Notebook: Intro_DT_HabPlanets.ipynb and a dataset: HPLearningSet.csv.
- Split into a training and a test set:

```
LearningSet = pd.read_csv (`HPLearningSet.csv’)
TrainSet = LearningSet.iloc[;13,:]
TestSet = LearningSet.iloc[13:,:]
```

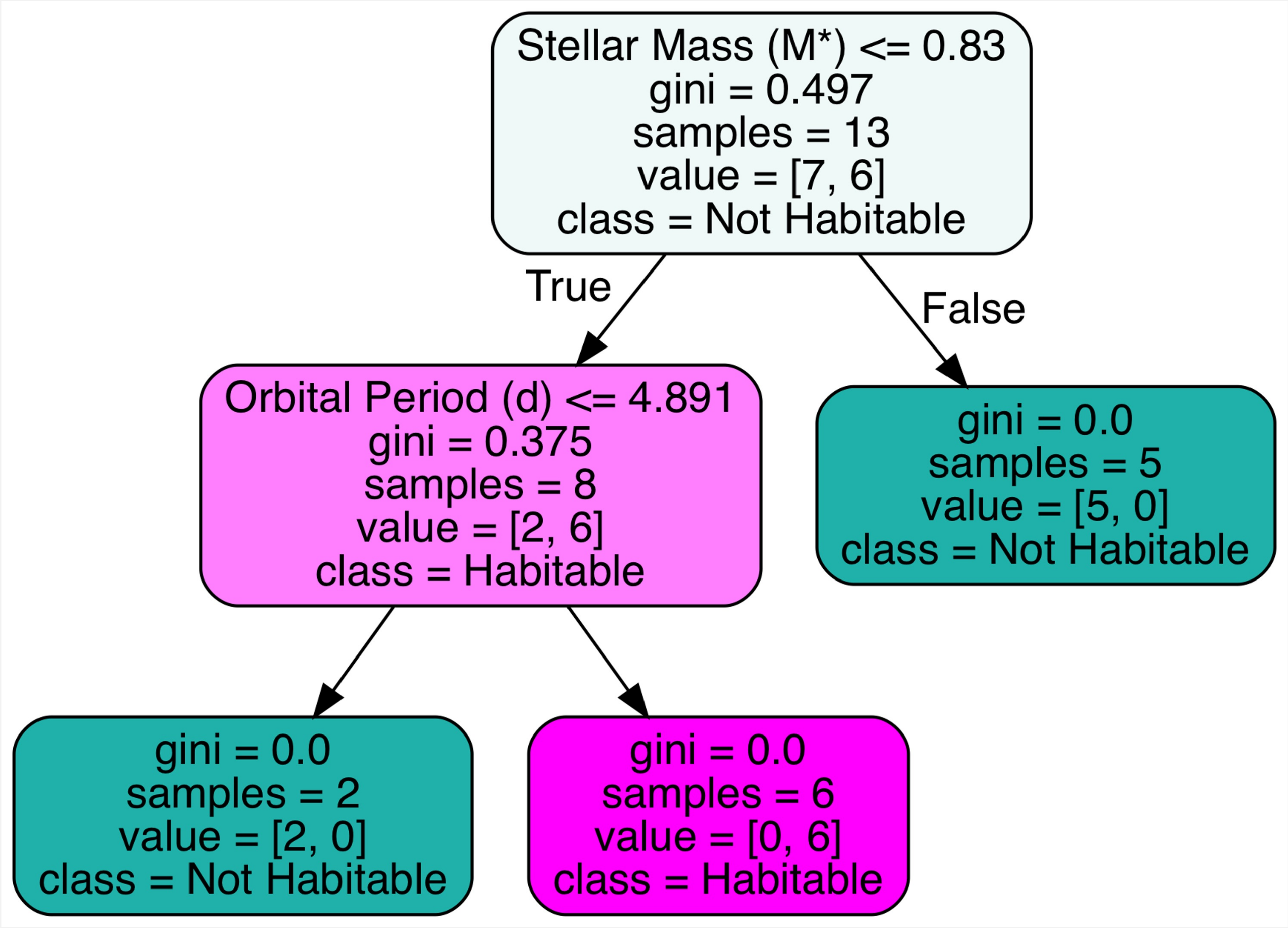
- The file contains both features & labels, separate them into 4 arrays:

```
Xtrain = TrainSet.drop([`P_NAME’,`P_HABITABLE’], axis = 1)
XTest = TestSet.drop([`P_NAME’,`P_HABITABLE’], axis = 1)
ytrain = TrainSet.P_HABITABLE
ytest = TestSet.P_HABITABLE
```

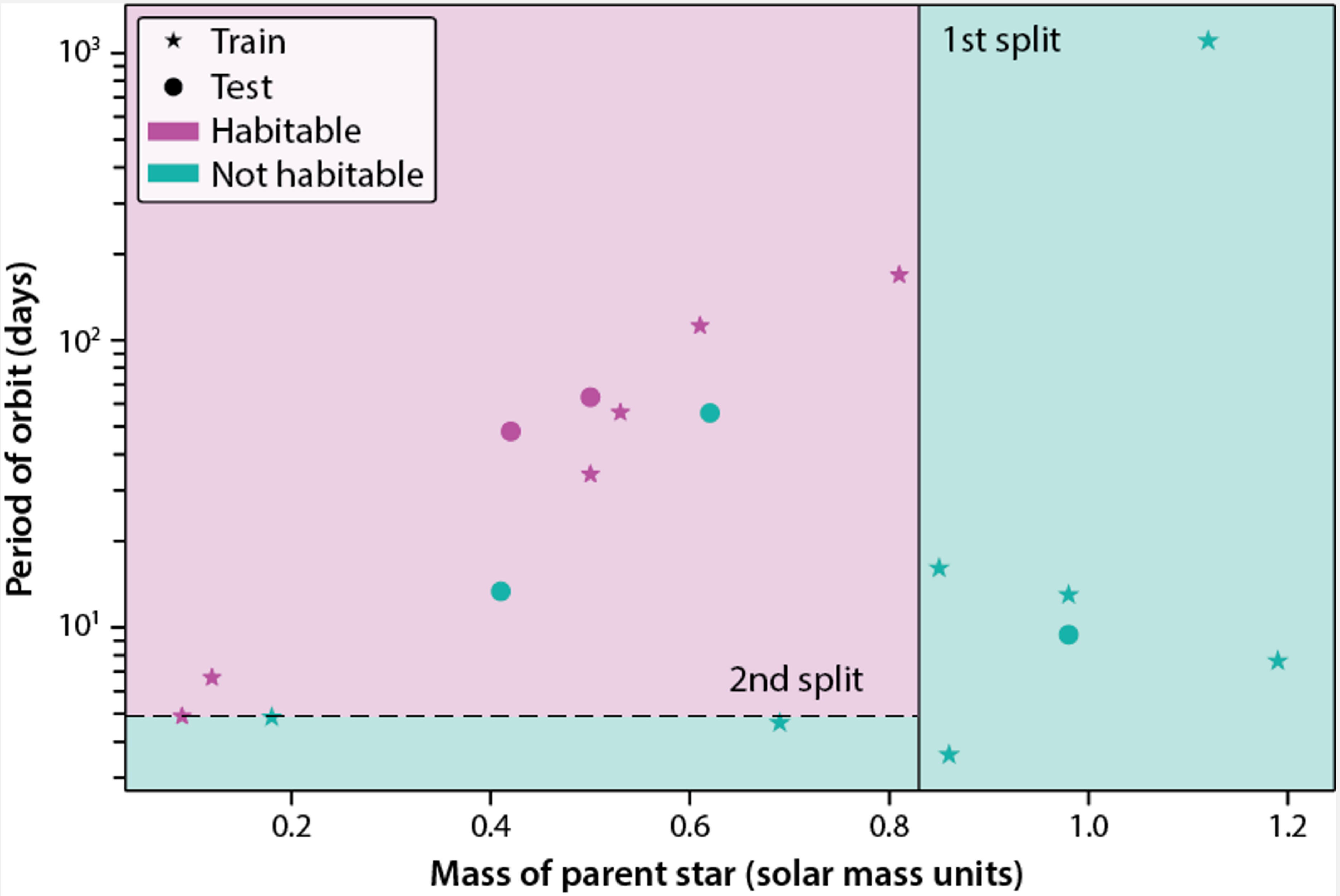
- Import the Decision Tree Classifier from sklearn and build the decision tree using the “fit” method.

```
model = DecisionTreeClassifier (random_state = 3)
model.fit(Xtrain, ytrain)
```

LET'S SEE WHAT SKLEARN SAYS

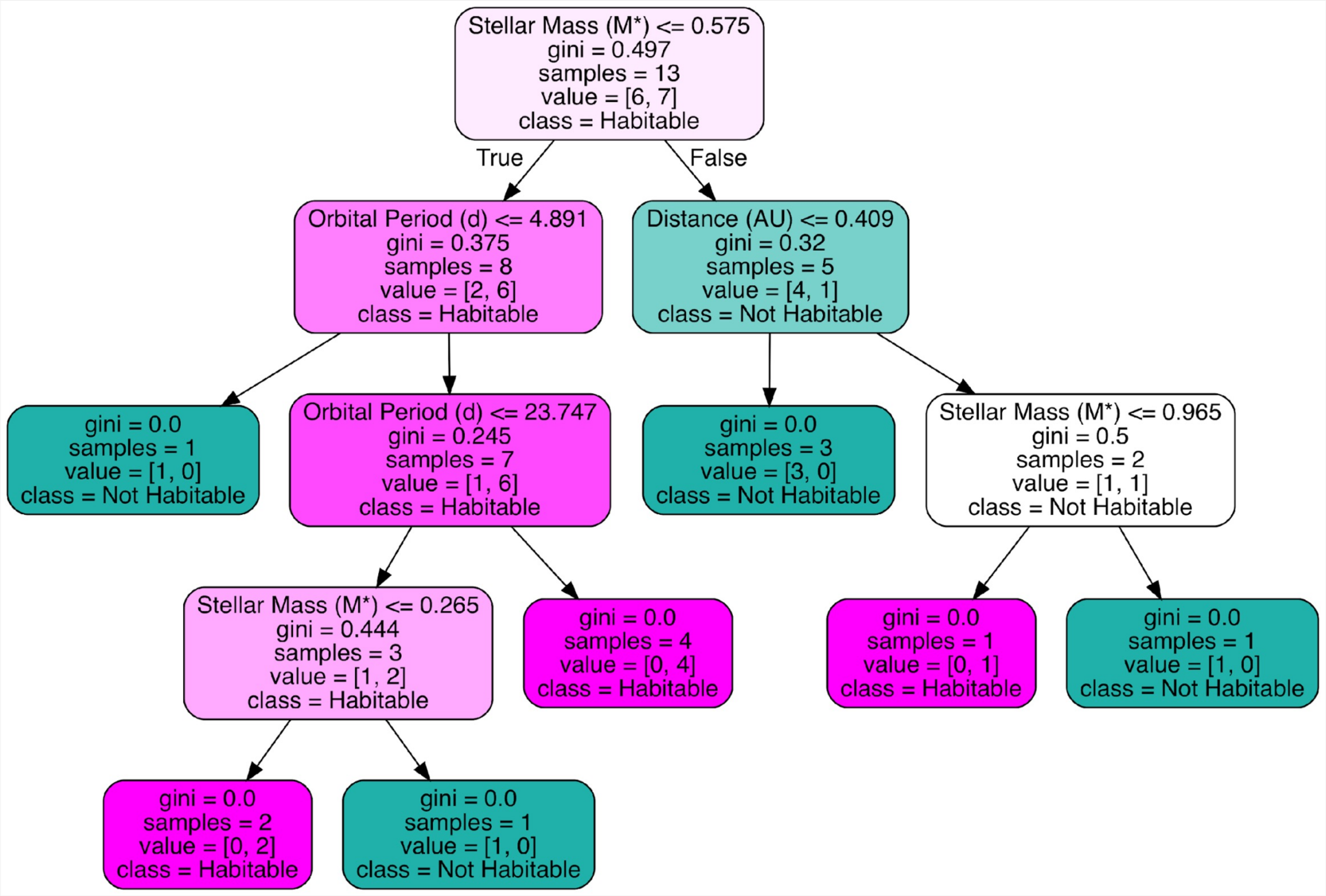


AND VISUALIZE THE CRITERIA THAT WE FOUND!



Can you figure out the accuracy on the test set?

NOTE (AND WE'LL SEE CODE FOR IT): IF YOU USE THE LAST 13 ROWS FOR TRAINING AND THE FIRST 5 FOR TESTING, YOU GET THIS TREE:



and 100% accuracy on test set

Morale: Different train/test split might give significantly different performances, especially when data sets are small.

KNN ALGORITHM
(K NEAREST NEIGHBORS)

Simple, yet powerful!

Only one parameter: k (a small integer)

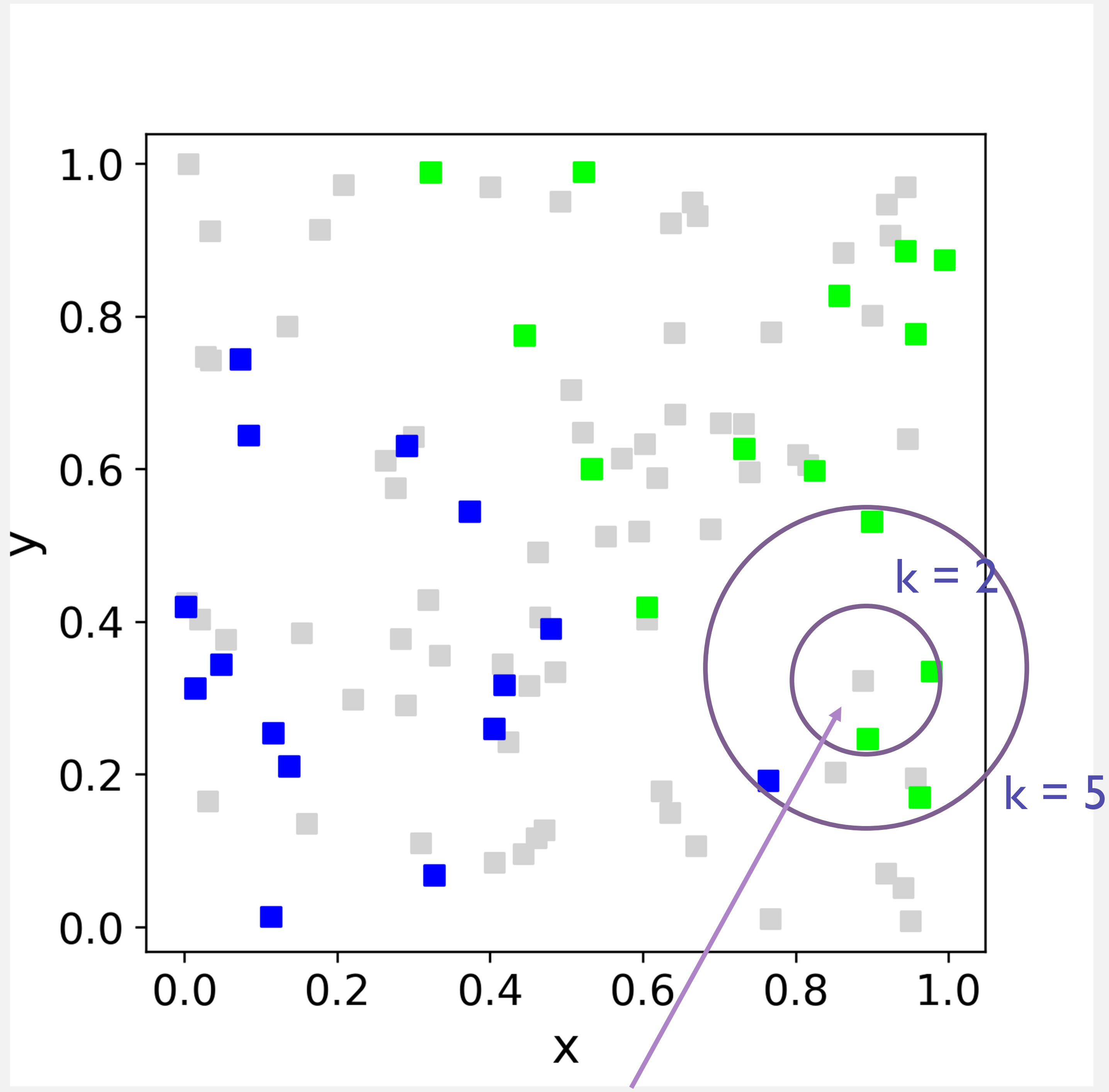
To make a prediction for a new object, find k closest examples in training set

For classification problems, output the majority class

For regression problems, output the mean of the target property

LET'S USE OUR OLD EXAMPLE.

WHAT VALUE SHOULD WE USE FOR K?



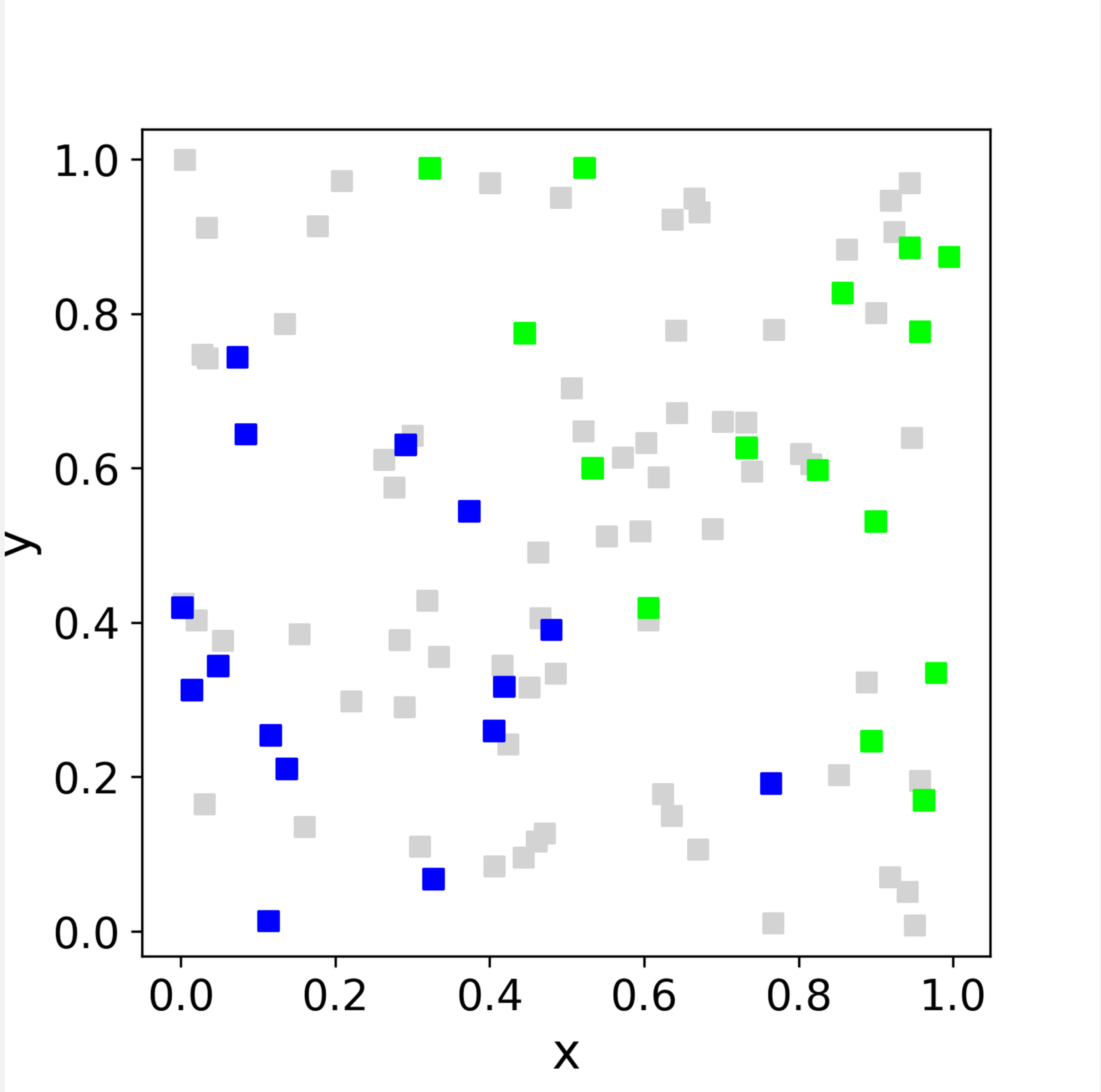
odd is better!

TWEAKABLES!

1) Choose neighborhood radius instead of k

2) Weigh different objects according to distance (inverse-distance weighing)

- Any insights on the effects of 1) and 2)?



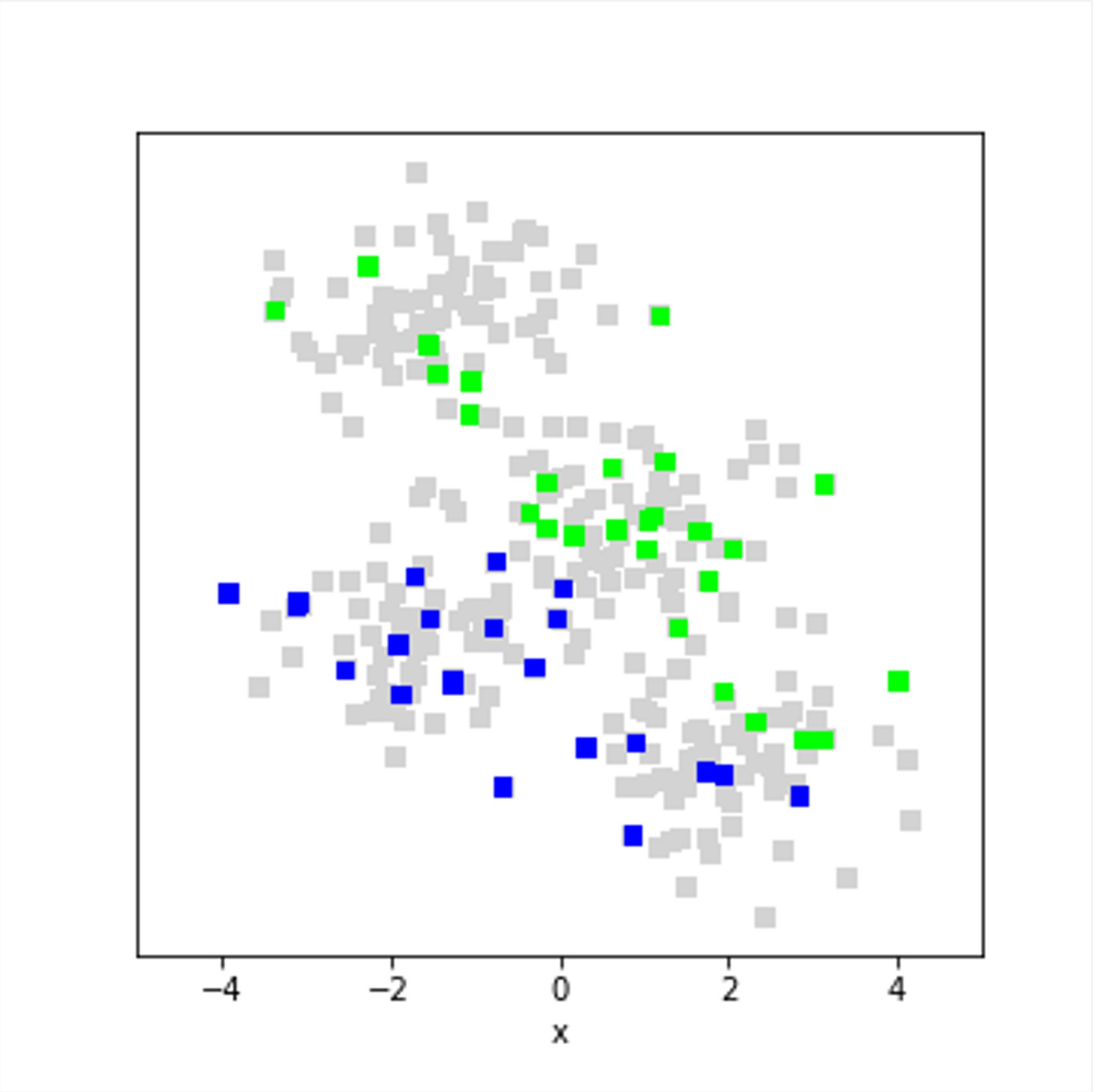
TWEAKABLES!

1) Choose neighborhood radius instead of k

2) Weigh different objects according to distance (inverse-distance weighing)

- Any insights on the effects of 1) and 2)?

- If the data have non-uniform density, different choices will have different effects! Needs to be chosen via cross-validation

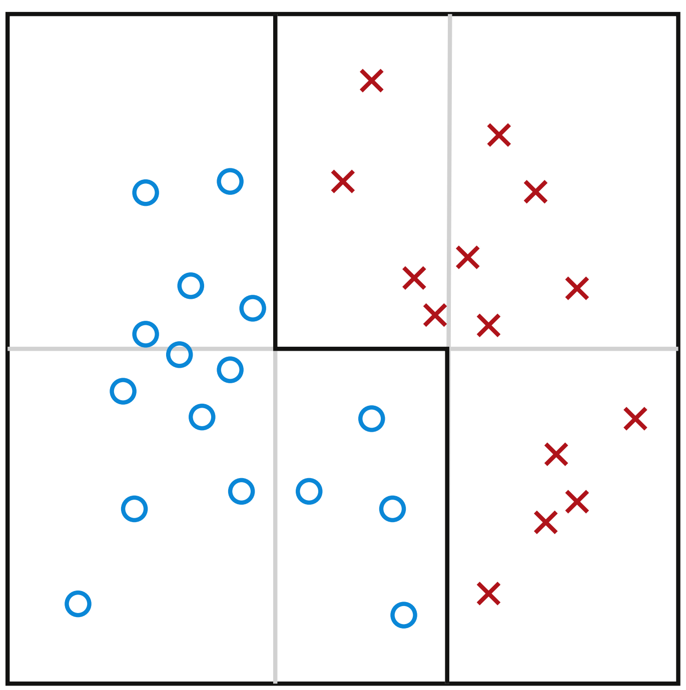


Ensemble Methods

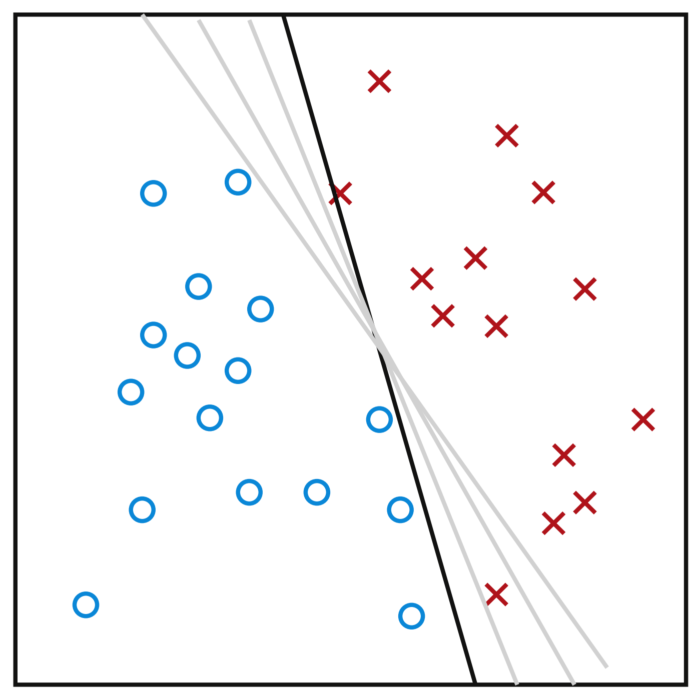
Two heads are better than one, or「三個臭皮匠，勝過一個諸葛亮」

Why Ensembles?

- **Statistical:** Multiple minima with same performance (training set too small). Choosing average reduces risk of wrong hypothesis choice.
- **Computational:** get stuck in local minima; results (e.g. decision tree structure + classification) vary strongly depending on training set.
- **Representational:** more expressive than single predictor, e.g.,



Aggregating different linear hypotheses



Linear perceptron hypothesis

Ensemble Methods

- Consider binary classification. Suppose we have N classifiers, each with accuracy p. If the models are independent of each other, the probability that k classifiers are correct:

$$P(k, N, p) = \frac{N!}{k!(N-k)!} p^k (1-p)^{N-k}$$

- Say the ensemble model classifies by a majority vote, i.e., $k \geq N/2$

$$p(\text{combined model}) = \sum_{i=N/2}^{N} \frac{N!}{i!(N-i)!} p^i (1-p)^{N-i}$$

- $p(\text{combined model}) > p$ if $p > 0.5$. For example, if $N = 11, p = 0.6$, you are encouraged to check that $p(\text{combined model}) = 0.75$.

- Ensemble methods include **Bagging**, **Boosting**, and **Random Forest**.