CSPB 3202 Artificial Intelligence

Non-parametric models

Geena Kim

University of Colorado **Boulder**

Supervised Learning- model types

- Data: $\boldsymbol{X}$
- Target y
- Prediction $\hat{y}$
- Model $\mathcal{F}$
- Parameters $\boldsymbol{\theta}$
- Hyperparameters $\boldsymbol{\lambda}$
- Loss $\mathcal{L}$

$$\hat{y} = \mathcal{F}(\boldsymbol{X})$$ non-parametric

$$\hat{y} = \mathcal{F}(\boldsymbol{X}, \theta_1, \theta_2, \cdots, \theta_n)$$ → parametric

$$\hat{y} = \mathcal{F}(\boldsymbol{X}, \theta_1, \theta_2, \cdots, \theta_n, \lambda_1, \lambda_2, \cdots, \lambda_m)$$

$$\hat{y} = \mathcal{F}(\boldsymbol{X}, \lambda_1, \lambda_2, \cdots, \lambda_m)$$

"Regression" ← Linear regression

$y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots$

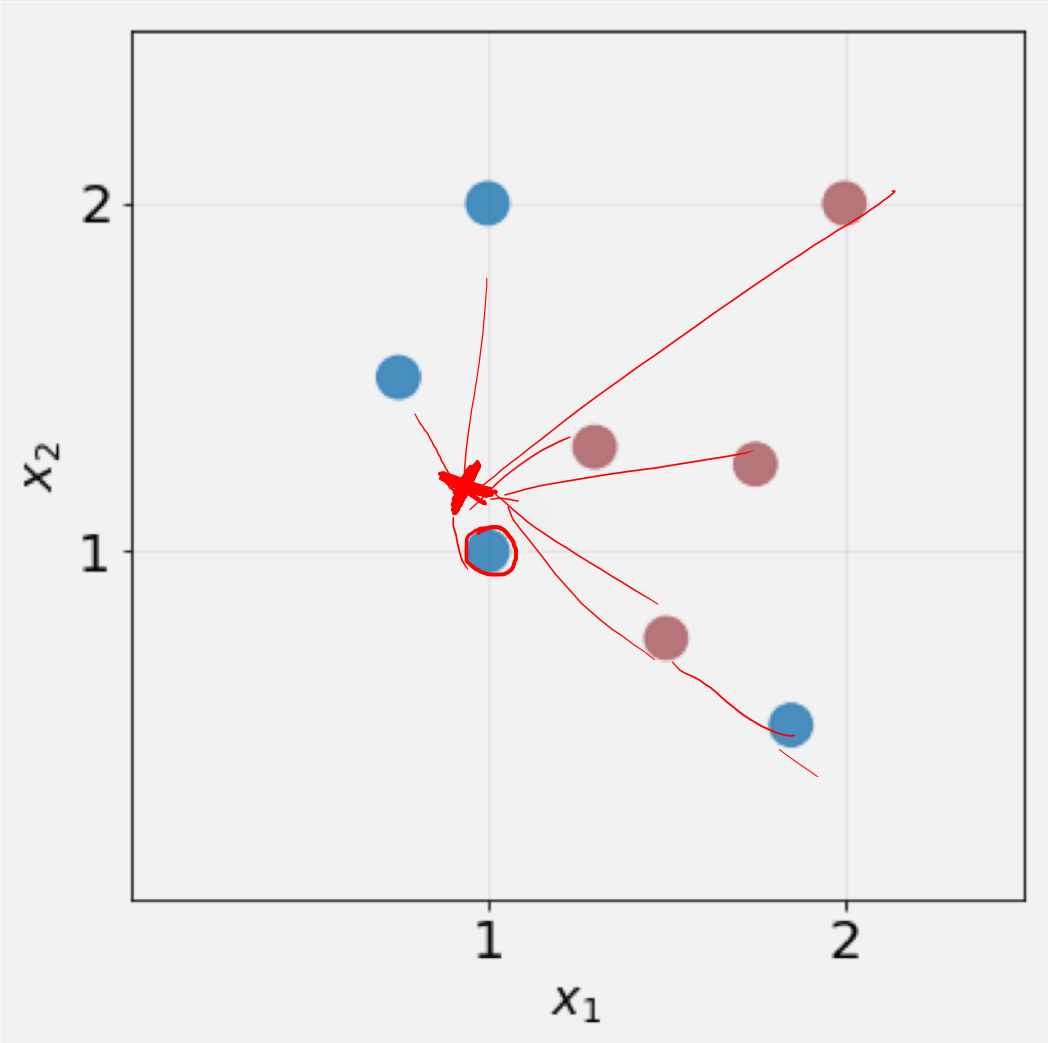
"classification" Logistic regression

z

$P(y=1|x) = \frac{1}{1+e^{-\theta}}$

k-Nearest Neighbor

Nearest Neighbor → non-parametric



no parameters

Manhattan distance

$$||\vec{X}_1 - \vec{X}_2||_1$$

Euclidean distance

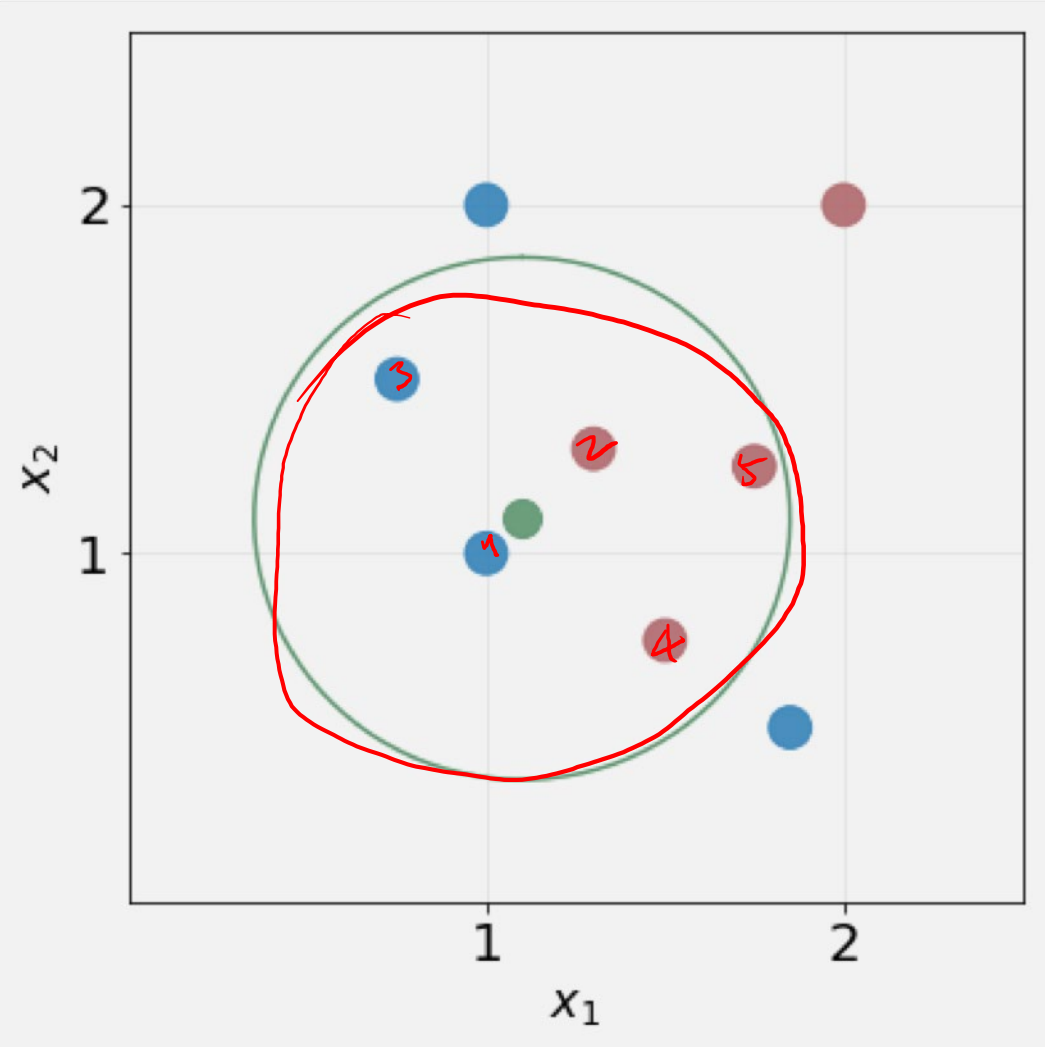
$$||\vec{X}_1 - \vec{X}_2||_2$$

$$\sqrt{\sum_{i=1}^{2} (X_{1,i} - X_{2,i})^2}$$

K-Nearest Neighbor

hyperparam

• No param

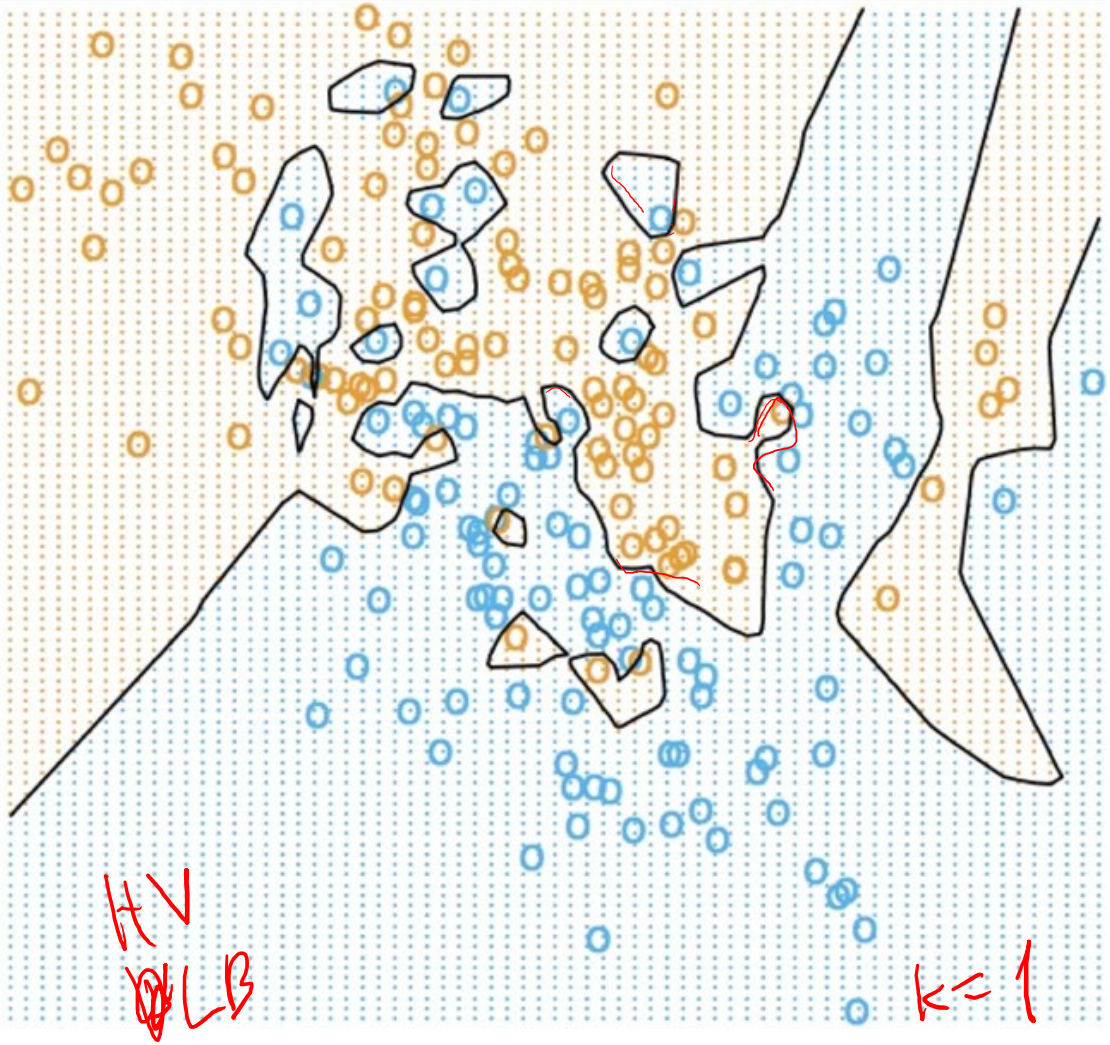


- Average /Majority vote
- k becomes a hyperparameter
  - 1-NN predicts ~~blue~~
  - 2-NN predicts ~~tie~~
  - 5-NN predicts ~~red~~

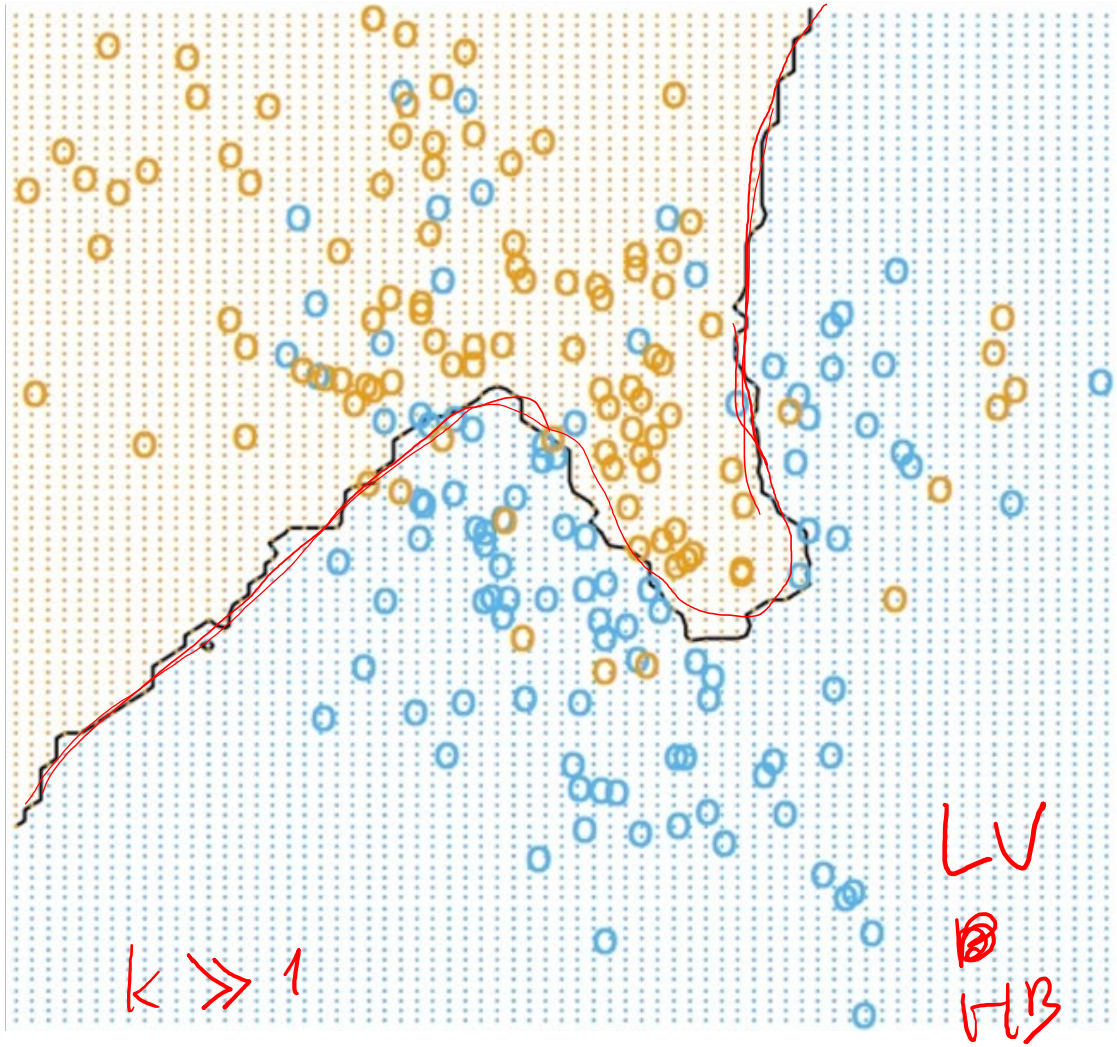
2 blue
3 red

$\hat{y} = f(X, k)$

How to choose k?

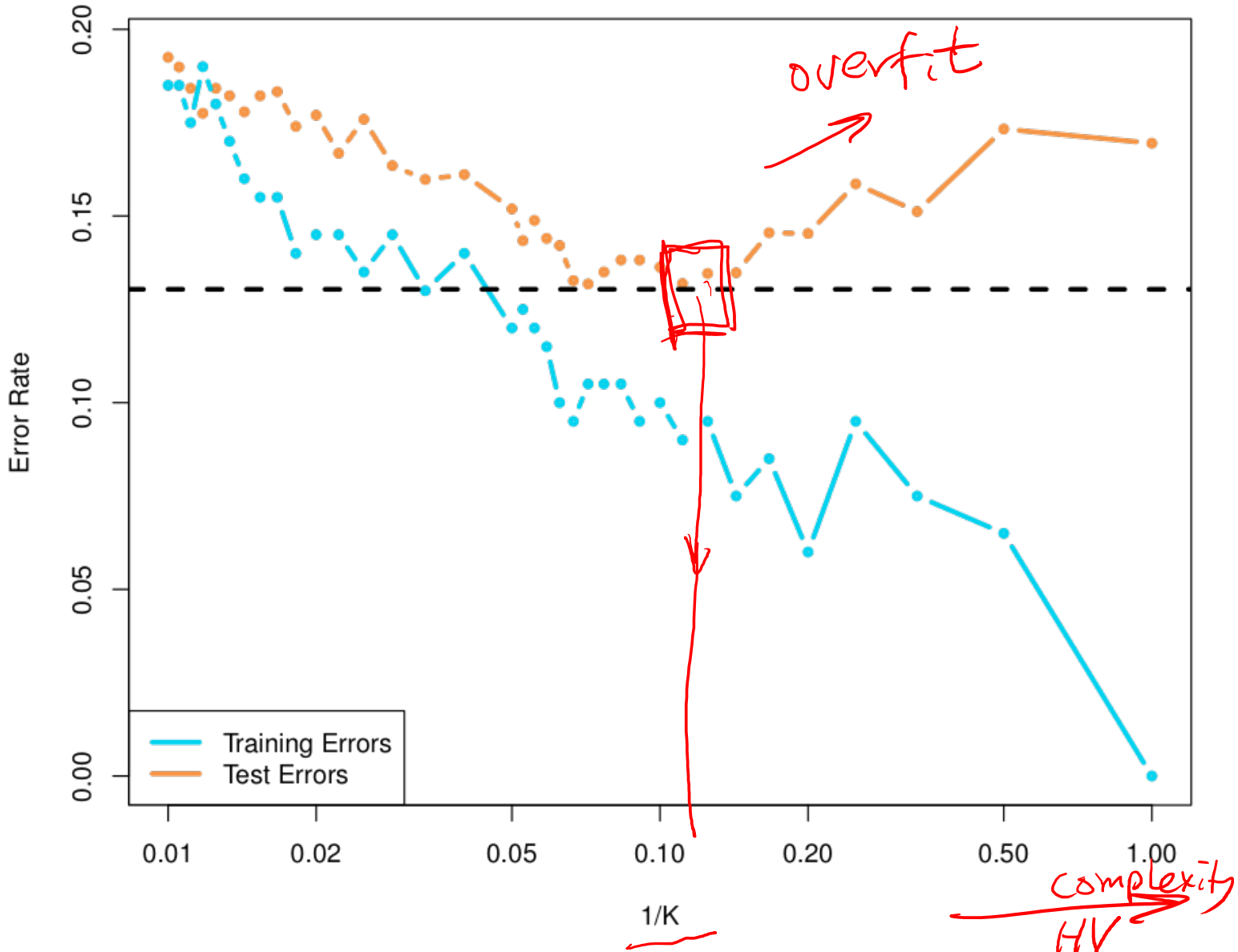


k=1



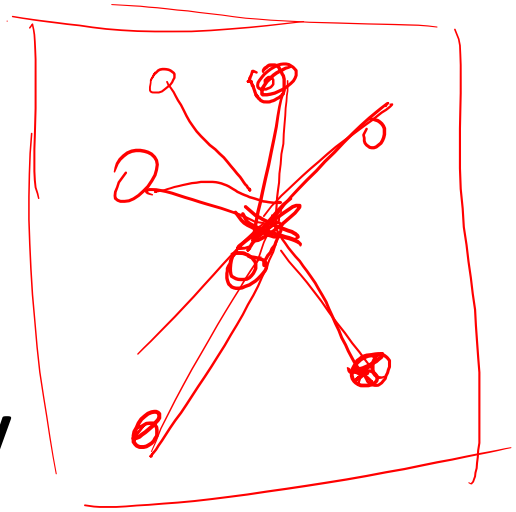
k=15

How to choose k?

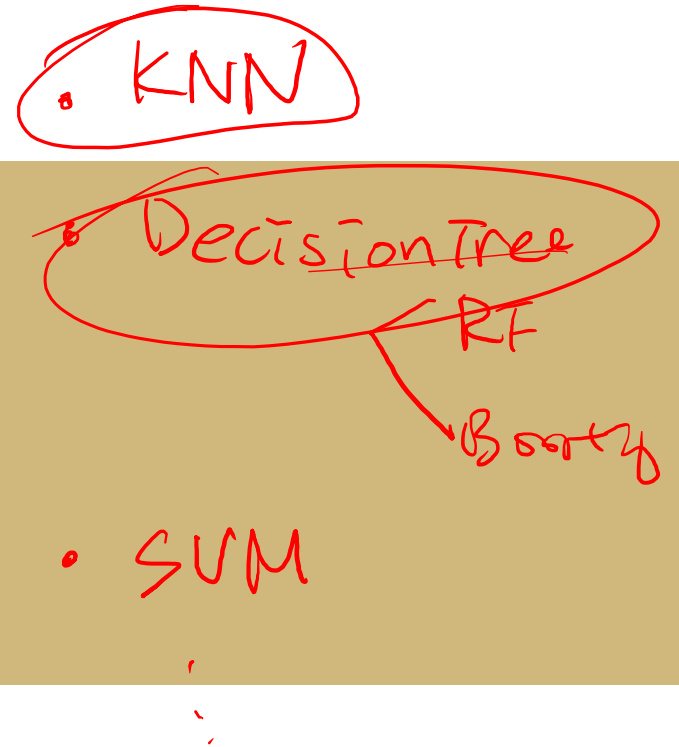


KNN Properties

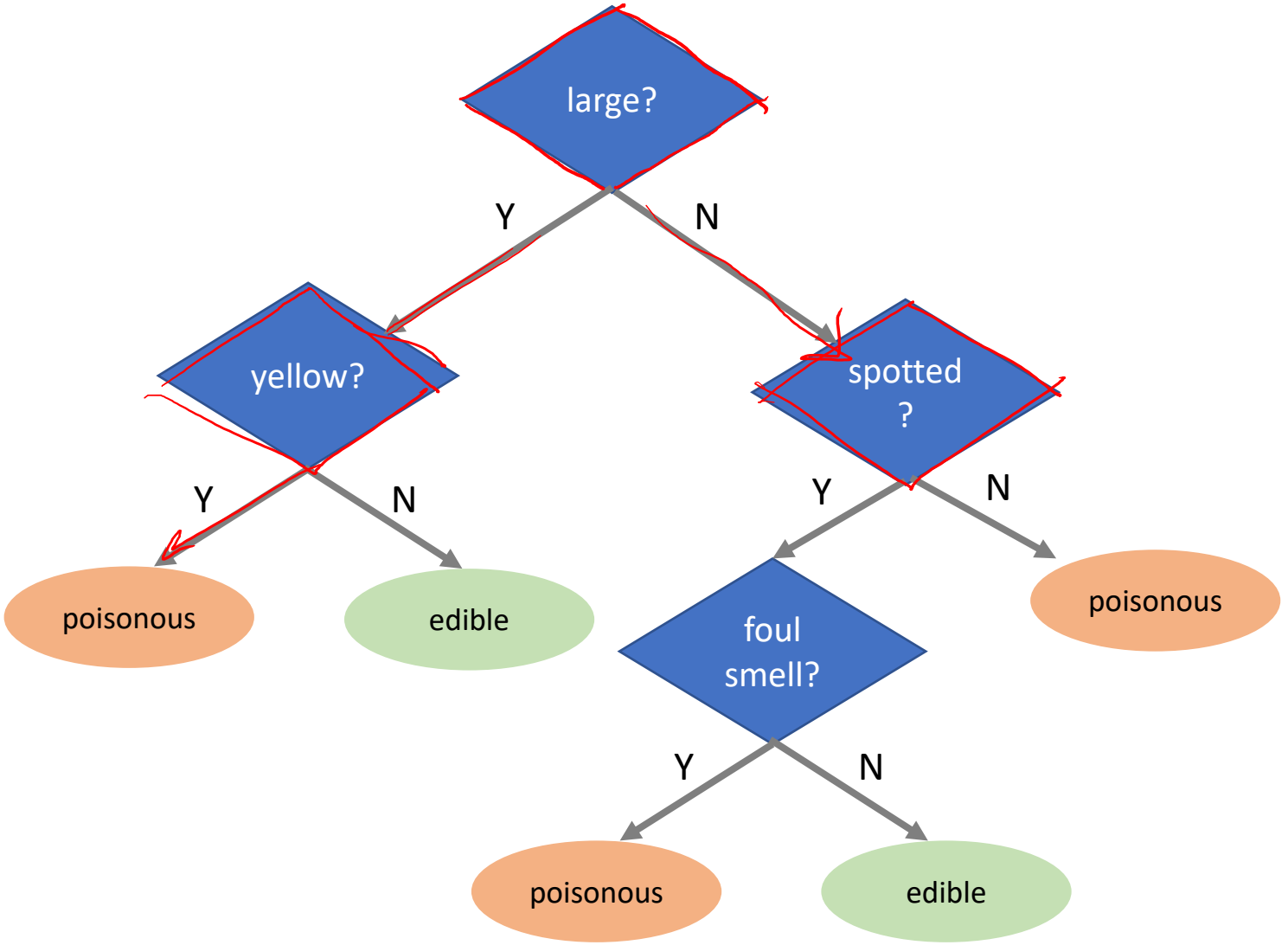
- Very Simple Algorithm
- Non-parameteric
- k is a Hyperparameter
- Very slow
- Curse of dimensionality



Tree Methods



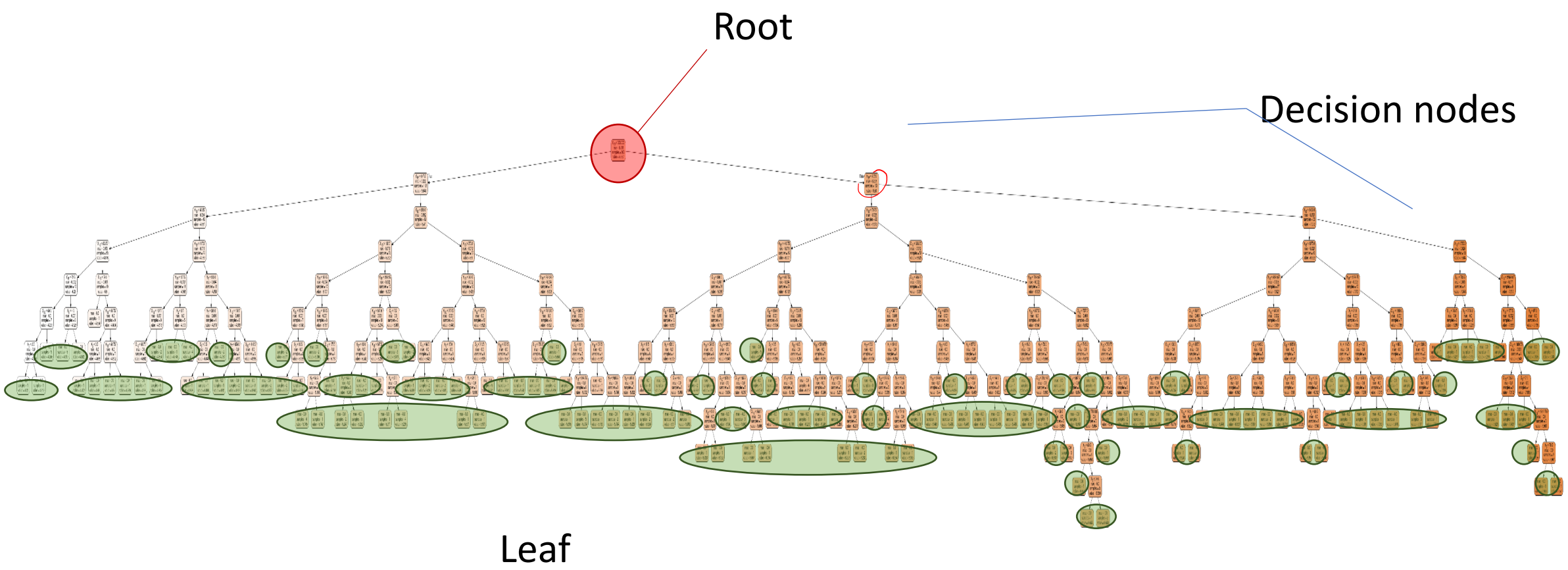
What is Decision Tree?



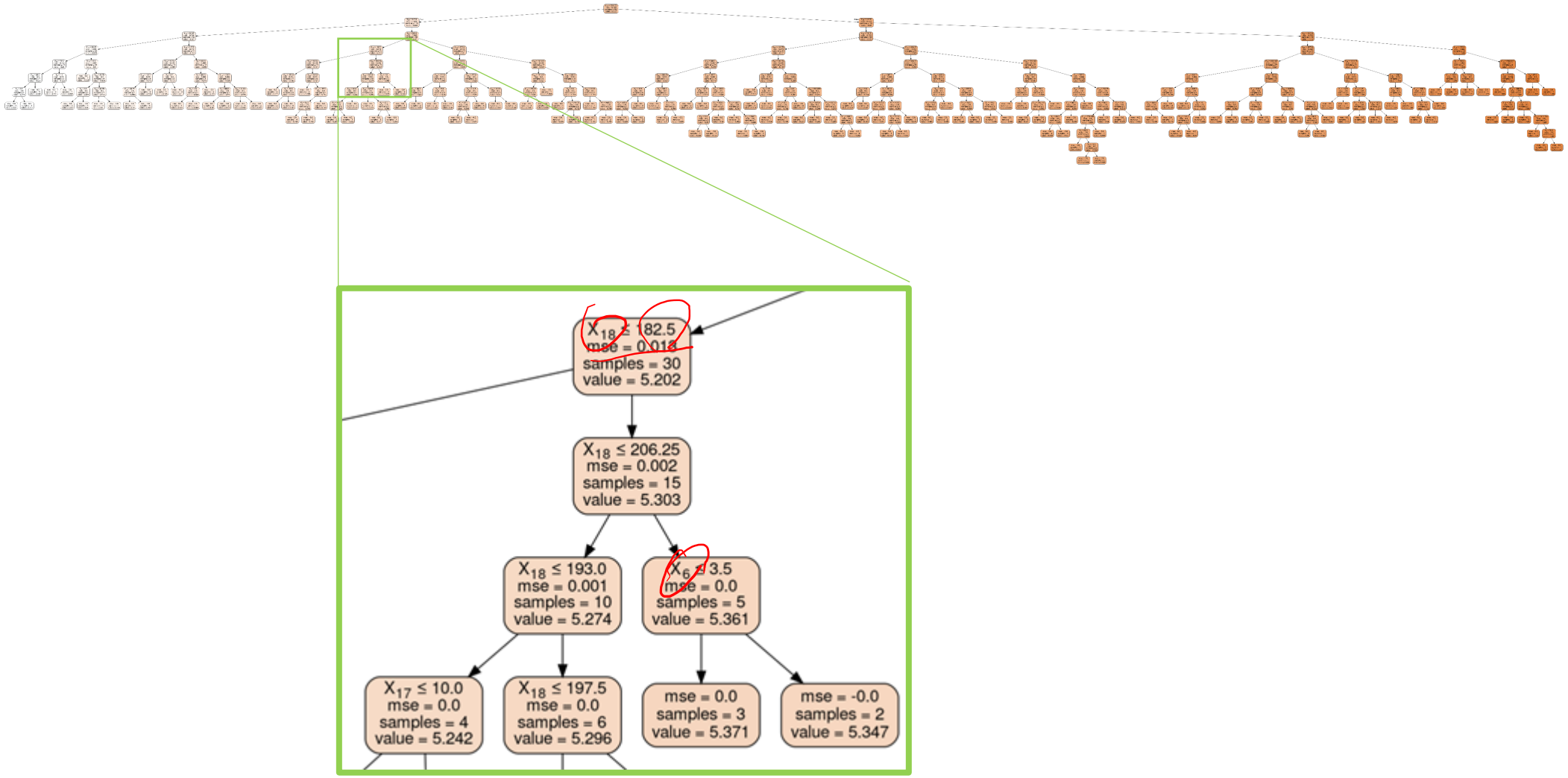
Caesar's mushroom

Death Cap

Decision Tree Nodes



Decision Nodes



Different kinds of models

Parametric vs. Non-parametric

Parameters vs. Hyperparameters

Linear Regression P

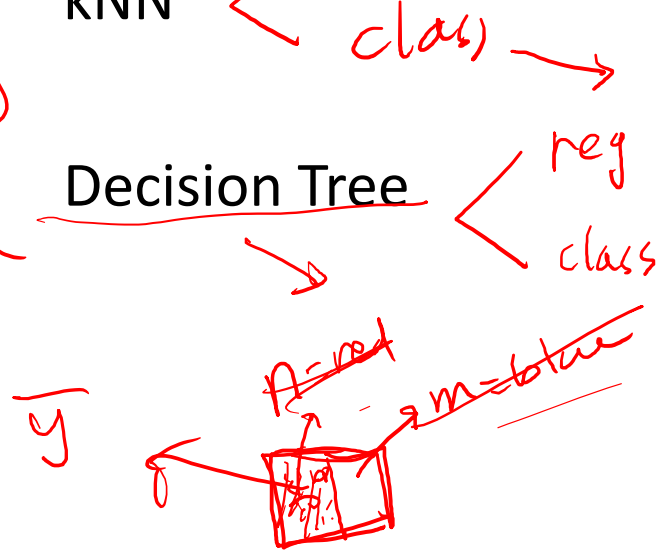
Logistic Regression P

kNN NP

Decision Tree NP

Optimization objective function

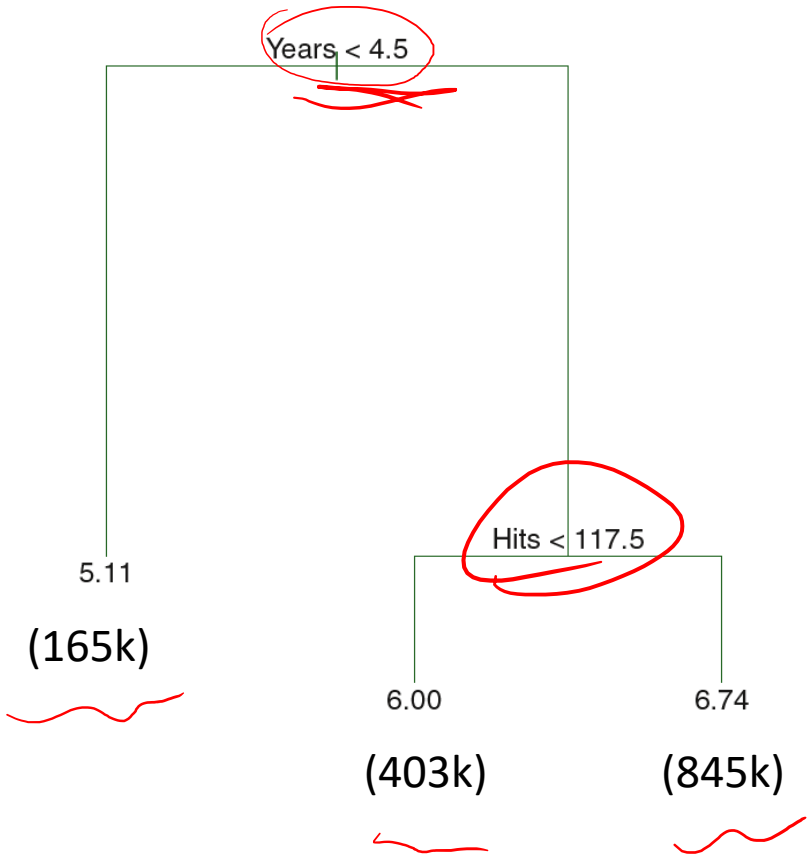
Linear Regression	Minimize MSE
Logistic Regression	Minimize Cross Entropy
kNN	No optimization, but uses distance
Decision Tree	Split to minimize MSE for Regression task and minimize Cross Entropy or Gini for Classification task

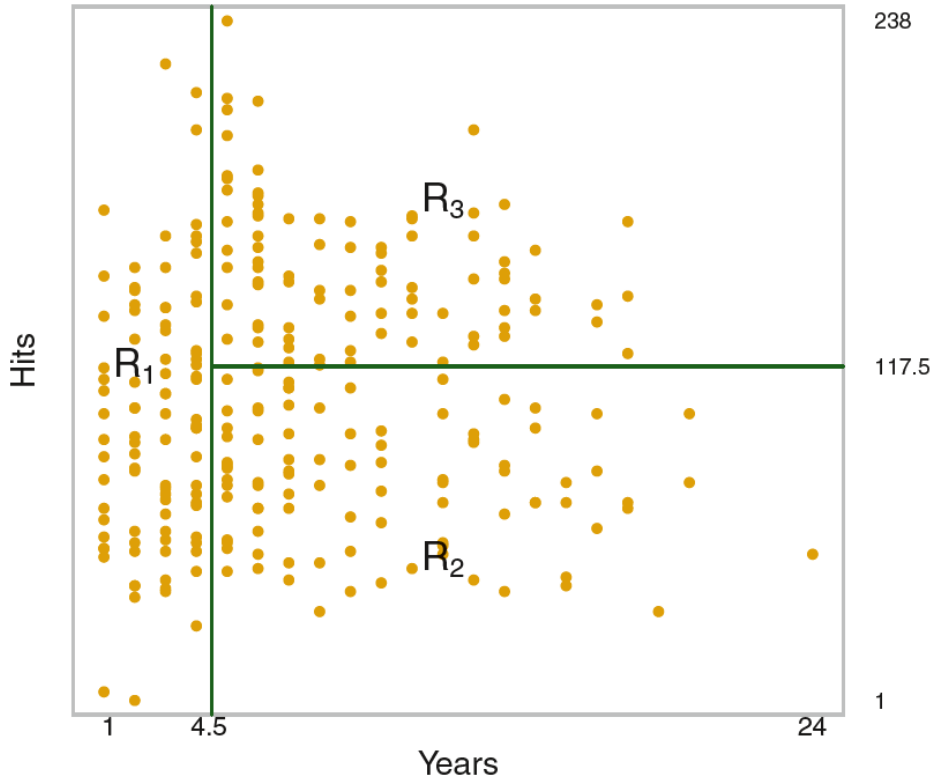


Decision Tree Regressor

Predicting Salary of Baseball players

- X1: number of years played in the major league
- X2: number of hits made in the last year
- y: log(salary)



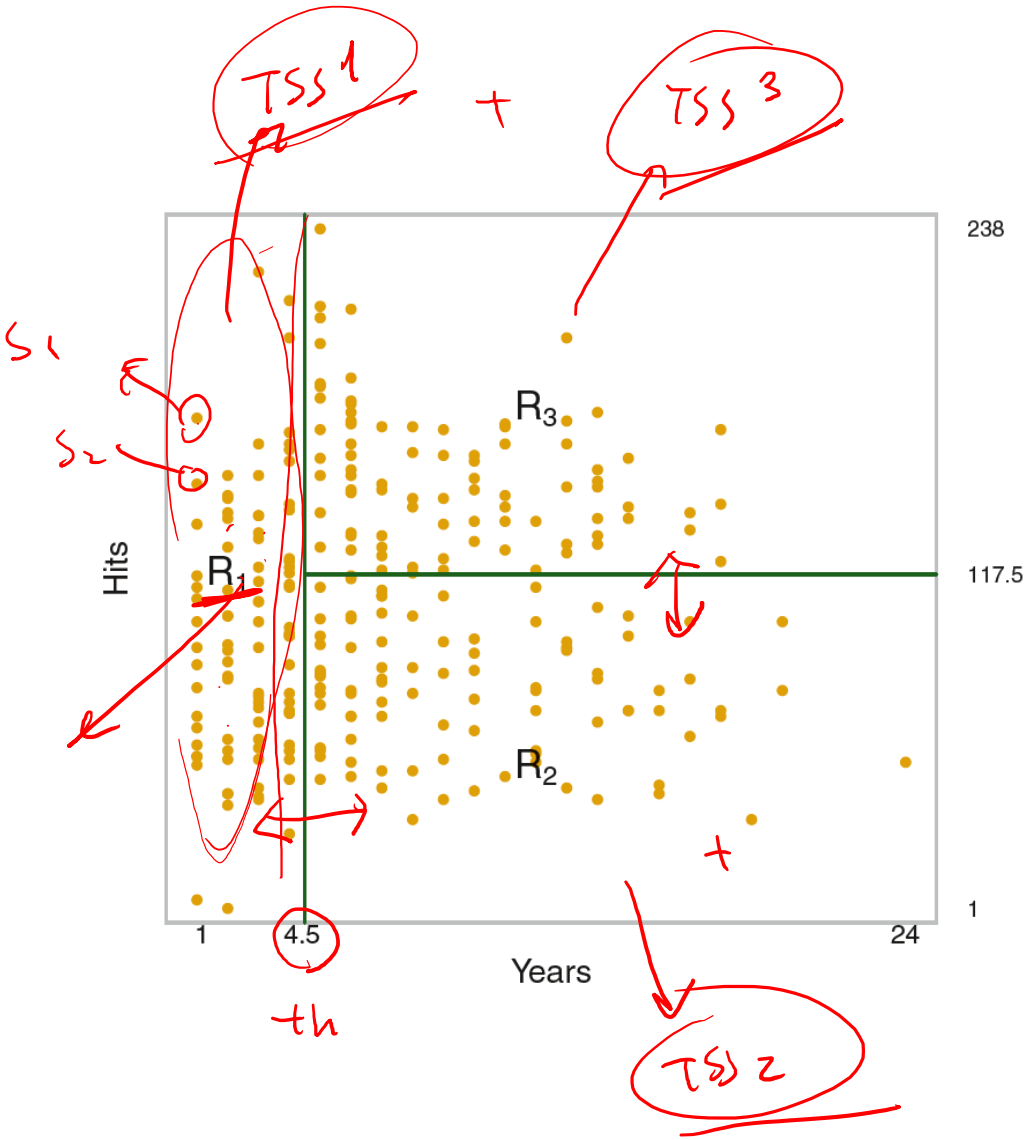


Decision Tree Regressor

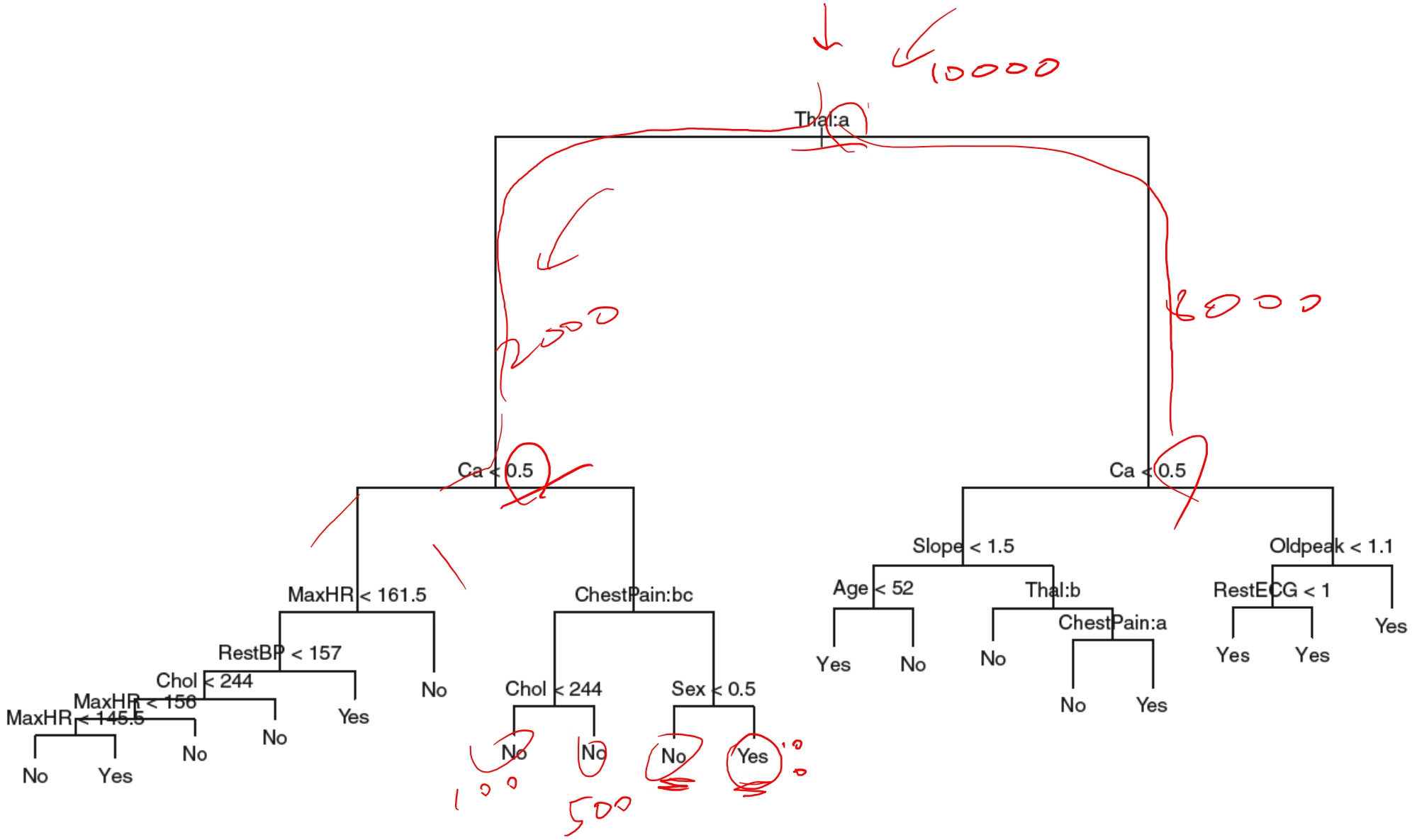
The goal is to find boxes $R_1 \sim R_J$ such that

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$$ is minimized.

the mean of the data in the box

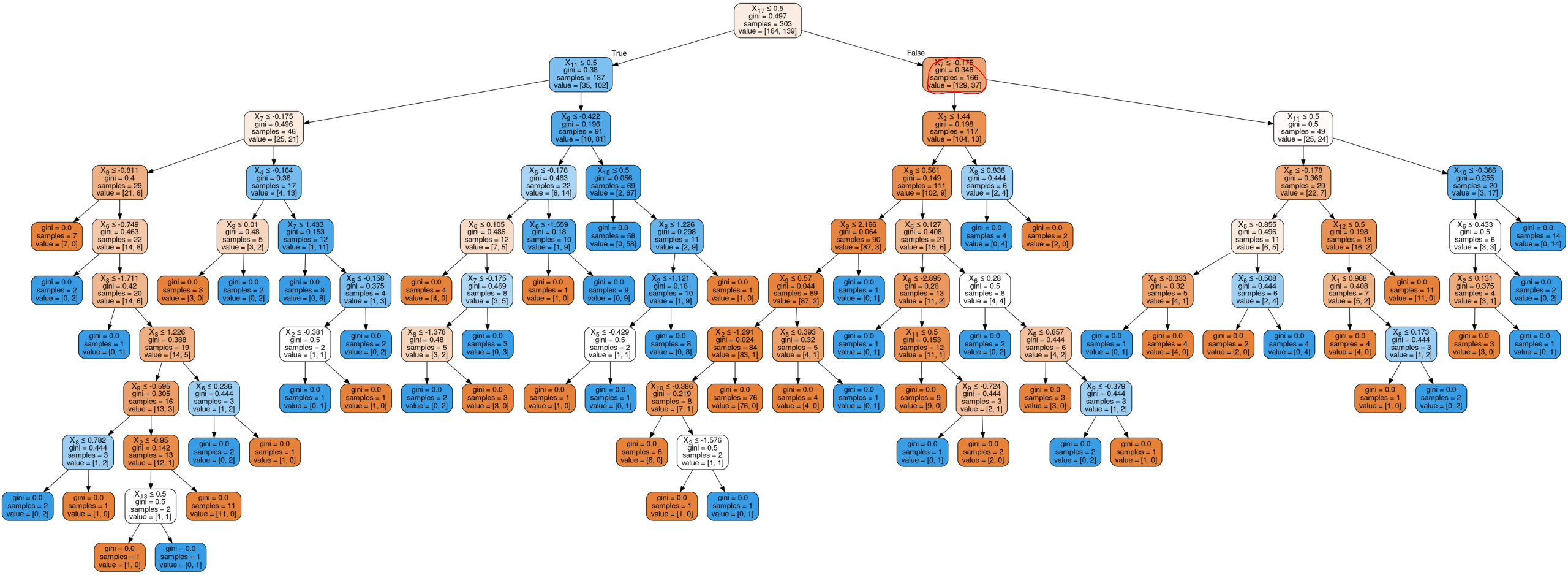


Decision Tree Classifier



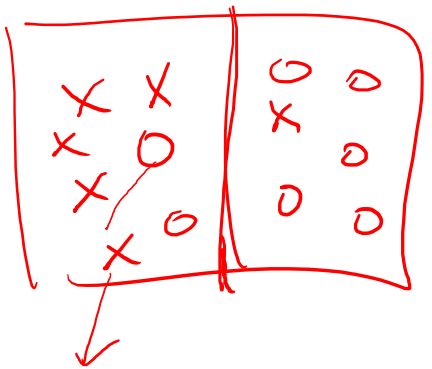
Decision Tree Classifier

sklearn

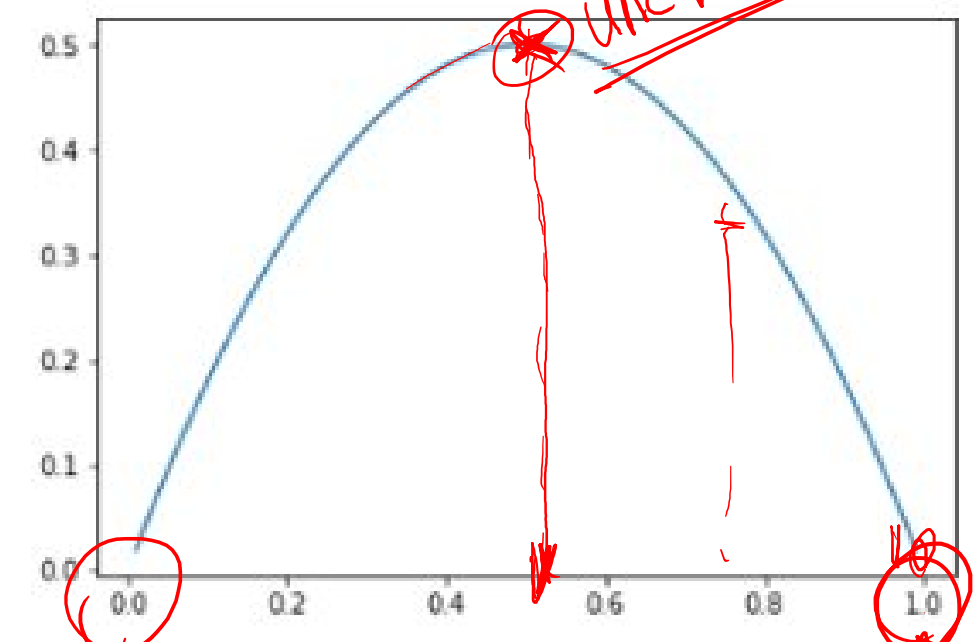


Split criterion- Gini index

measure of impurity



```
a = np.arange(0.01,1,0.01)
plt.plot(a,2*a*(1-a));
```



uncertain

pure

50:50 impure ↑

pure state

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

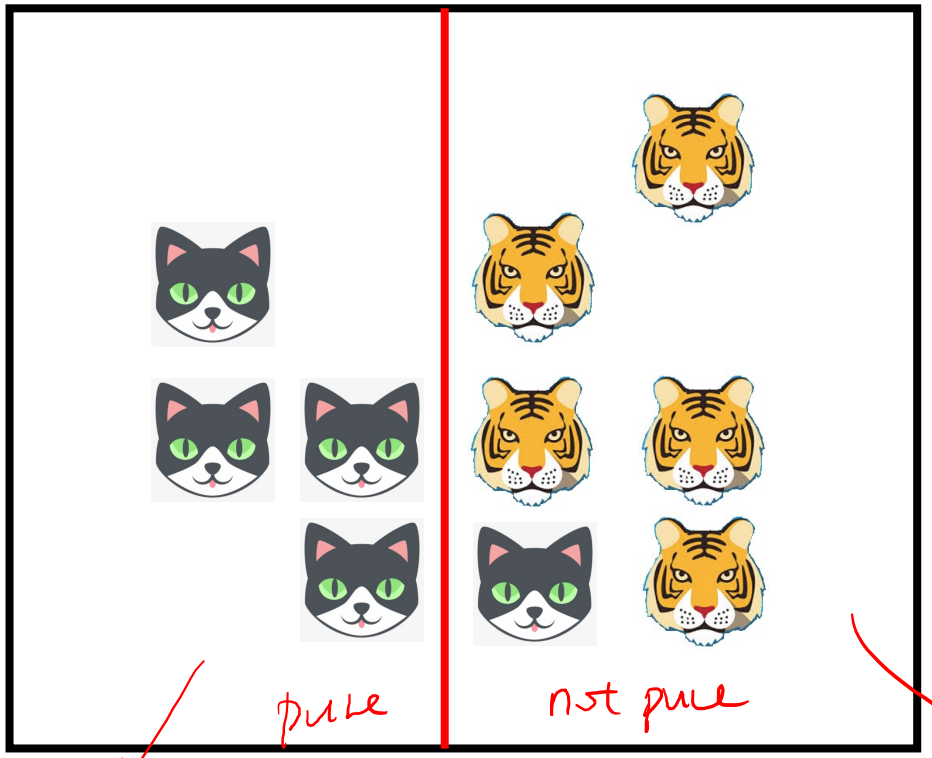
box indx

class index

What is the Gini of this box?

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

H_0 before $= 0{,}5$



Impurity < gini / Entropy

$\frac{N_L}{N} H_L + \frac{N_R}{N} H_R$

$N_L + N_R$

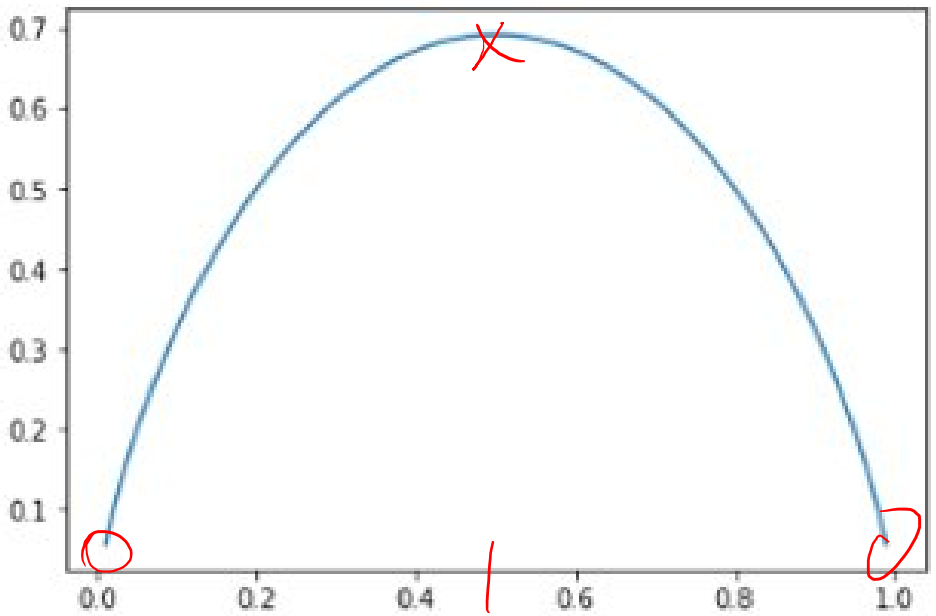
$H_L = 0$

$H_R = \frac{1}{6}\cdot\frac{5}{6} + \frac{5}{6}\cdot\frac{1}{6}$

$= 2 \cdot \frac{5}{36} = \frac{5}{18}$

Split criterion- Entropy

```
a = np.arange(0.01,1,0.01)
plt.plot(a,-a*np.log(a)-(1-a)*np.log(1-a));
```

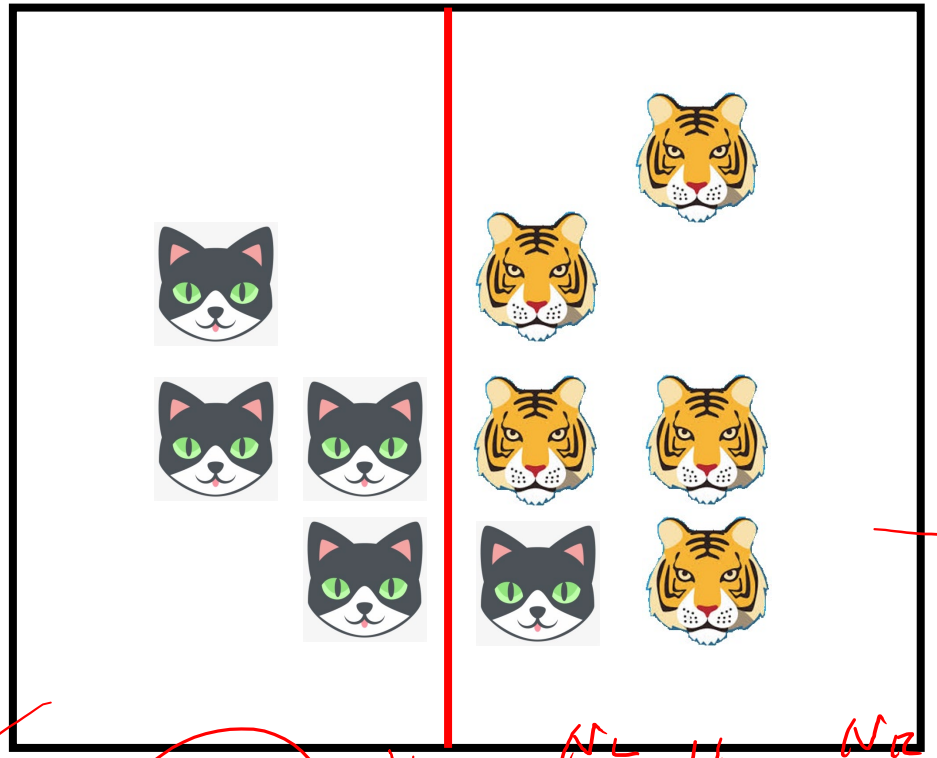


$$H(X_m) = -\sum_k p_{mk} \log(p_{mk})$$

Split criterion- Information gain

Information Gain = Reduction in Entropy

$$-\left(\frac{1}{2}\log_2\frac{1}{2}+\frac{1}{2}\log_2\frac{1}{2}\right)=1$$

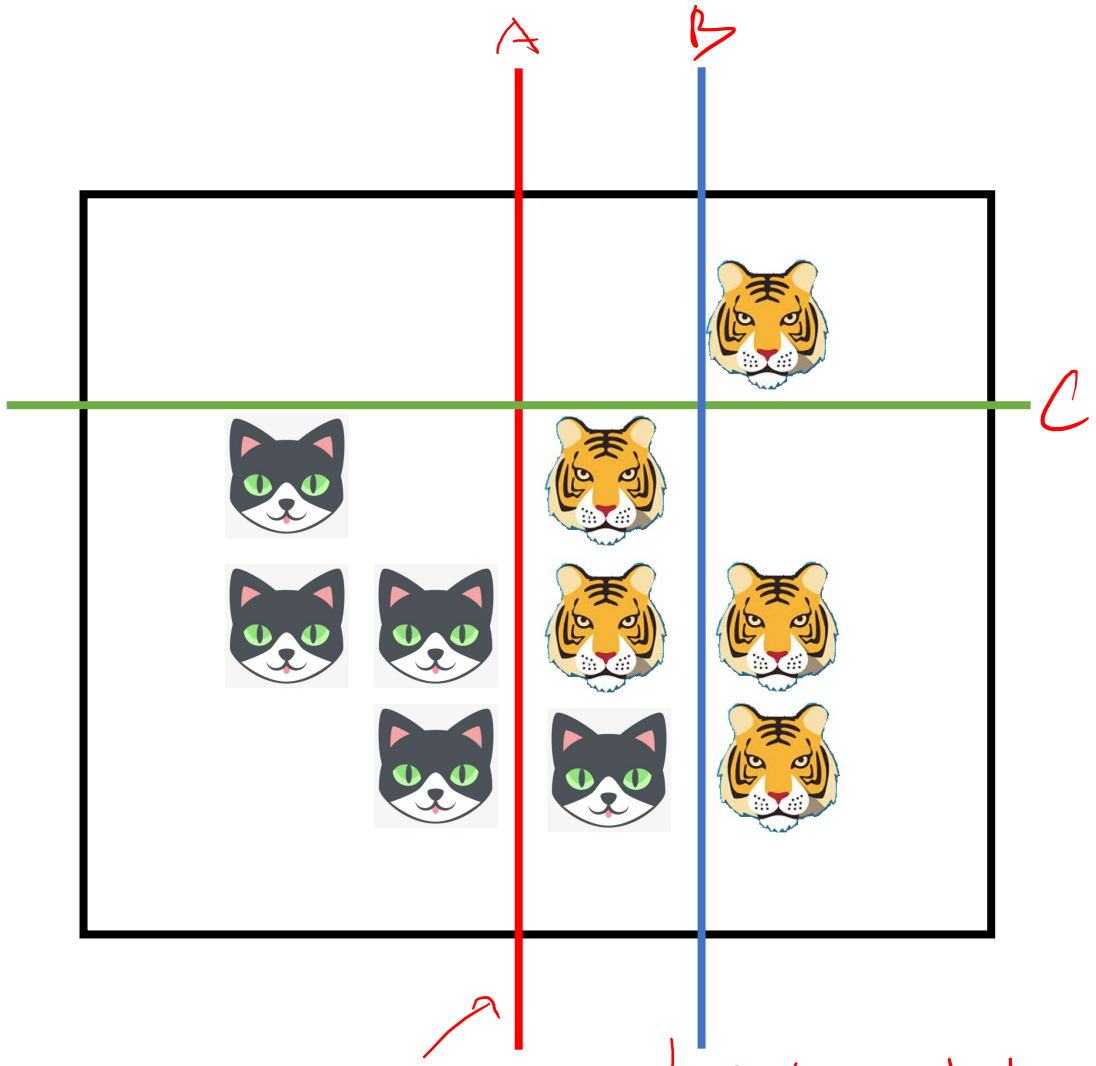


0

$$-\left(\frac{1}{6}\log_2\frac{1}{6}+\frac{5}{6}\log_2\frac{5}{6}\right)=0.65$$

Information Gain = 1 - 0.4*0 - 0.6*0.65 = 0.61

Which split gives the maximum information gain?



best split = a split that makes maximum info gain

Decision Tree Split Criteria

Regression Tree

MSE

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

MAE

$$H(X_m) = \frac{1}{N_m} \sum_{i \in N_m} |y_i - \bar{y}_m|$$

Classification Tree

Gini

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

Entropy

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

Information Gain

$$IG = E_{parent} - \frac{N_L}{N} E_L - \frac{N_R}{N} E_R$$

Decision Tree – When to stop split?

max_depth The maximum depth of the tree

min_samples_split The minimum number of samples required to split an internal node

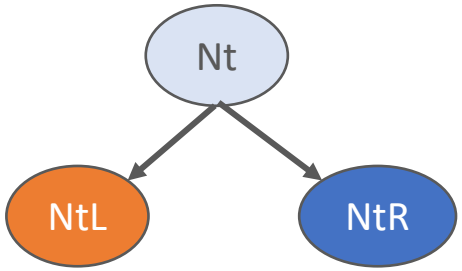
min_samples_leaf The minimum number of samples required to be at a leaf node

max_features The number of features to consider when looking for the best split

min_impurity_decrease A node will be split if this split induces a decrease of the impurity greater than or equal to this value

The weighted impurity decrease equation is the following:

```
N_t / N * (impurity - N_t_R / N_t * right_impurity
                    - N_t_L / N_t * left_impurity)
```



Hyperparameter search

Grid Search Tip

- Give a range of values for each hyperparameter
- Measure a training time for one, then estimate how long for the loop
- Adjust number of values, range, or hyperparameters to include

max_depth

min_samples_split

min_samples_leaf

max_features

min_impurity_decrease

Decision Tree Pros and Cons

Trees are easy to understand

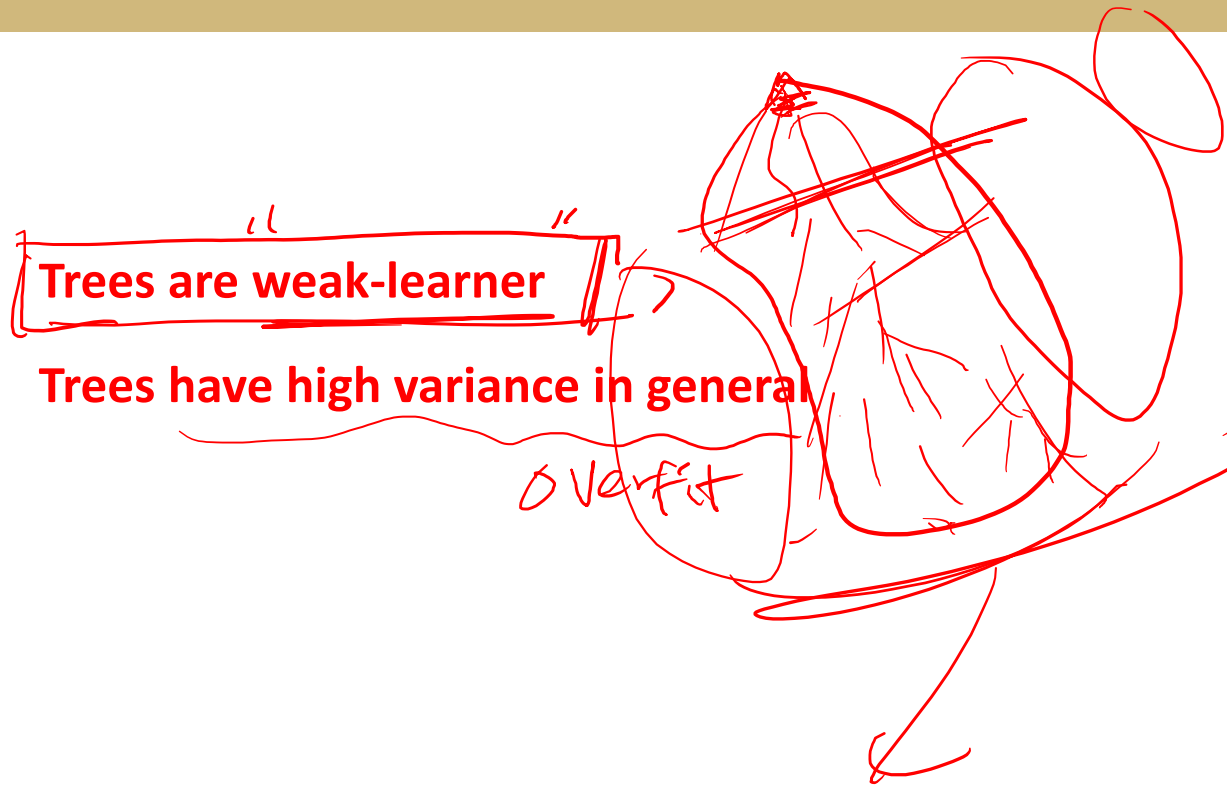
Trees don't suffer collinearity

Trees are good for non-linear features

Trees handle categorical variables easily

Trees are weak-learner

Trees have high variance in general



Linear regression is a better choice if features are linear

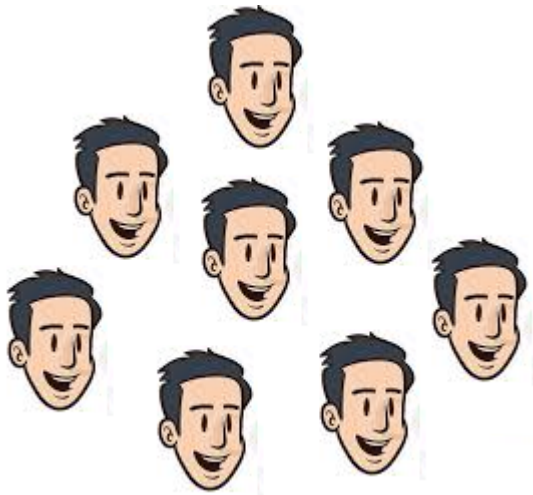
Tree's performance can be greatly improved when ensembled

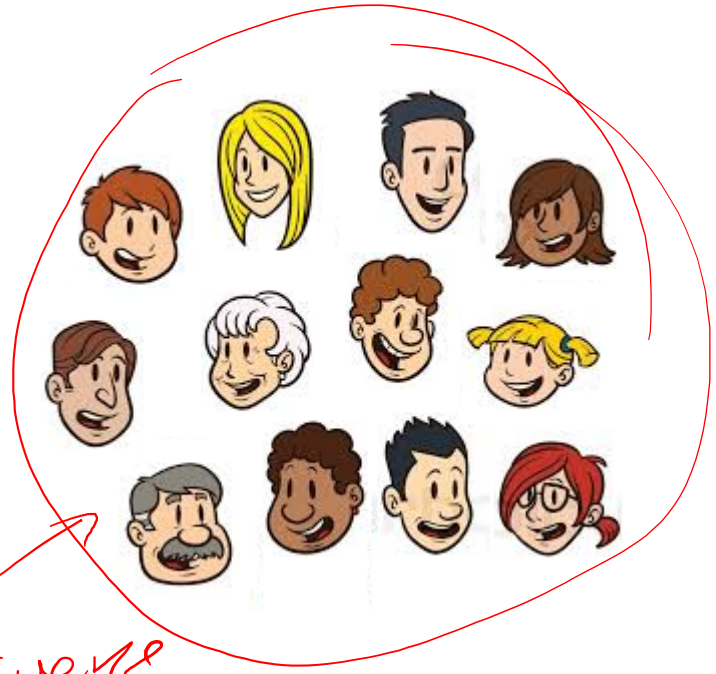
Ensemble method

What is an Ensemble?

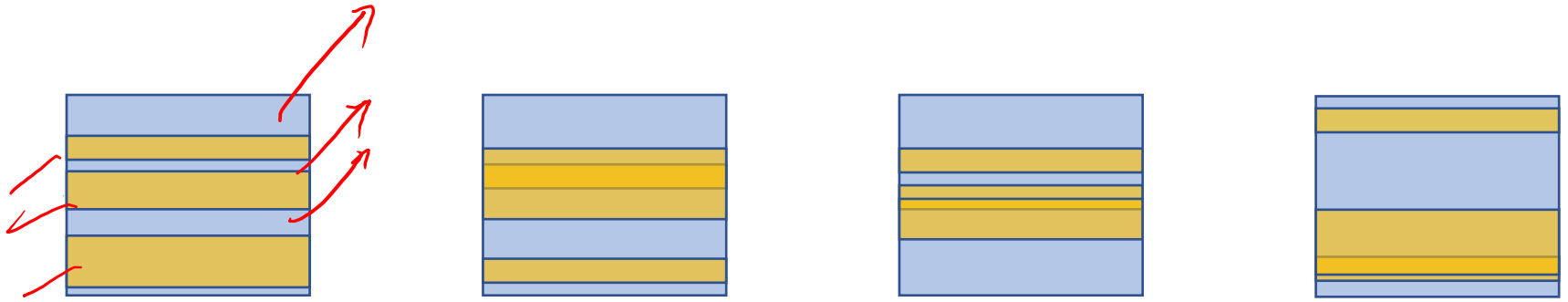
Ensemble method

Diversity matters





Bagging (Bootstrap-Aggregation)



STEP1: Randomly sample a subset of training data with replacement (Bootstrap)

STEP2: Grow a tree (without pruning) on the subset of data

STEP3: Ensemble the result (regression : average, classification : vote)

Out of Bag error (OOB) : test the grown tree on the rest of data, then average

Random Forest

"decorrelated" trees

Bagging : random sampling of data

+

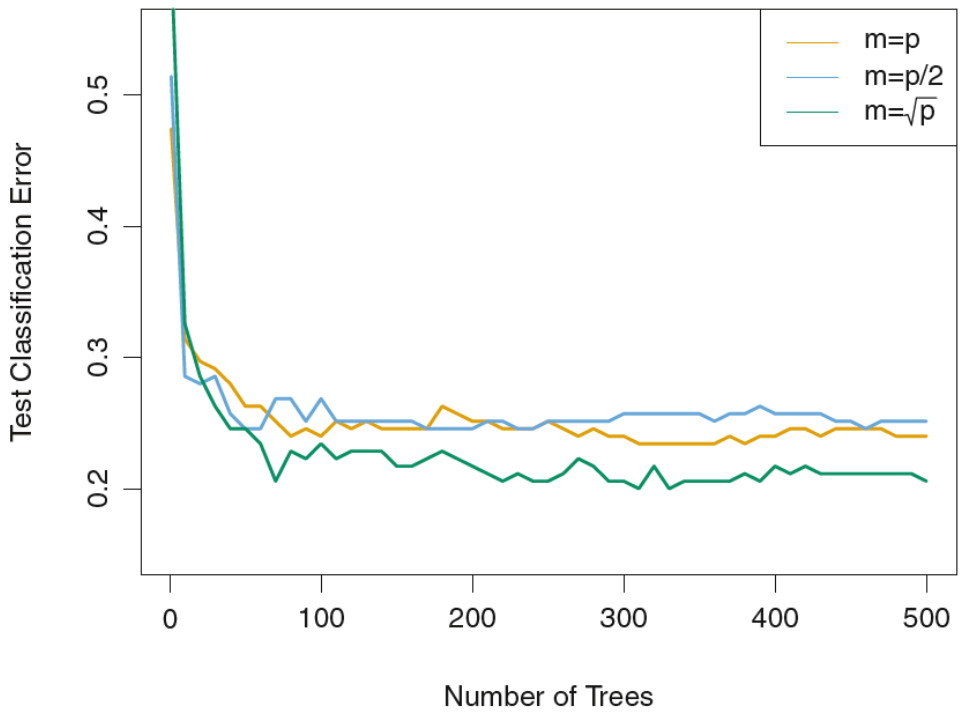
Decorrelation : random sampling of features

=

Random Forest

How do we sample features?

-> Rule of thumb : $\sqrt{n}$



Ensemble method- overview

Problem: Trees are weak learner and trees overfit

Idea 1: Let's average them (Ensemble)

Idea 2: Let's make decorrelated trees (samples, features)

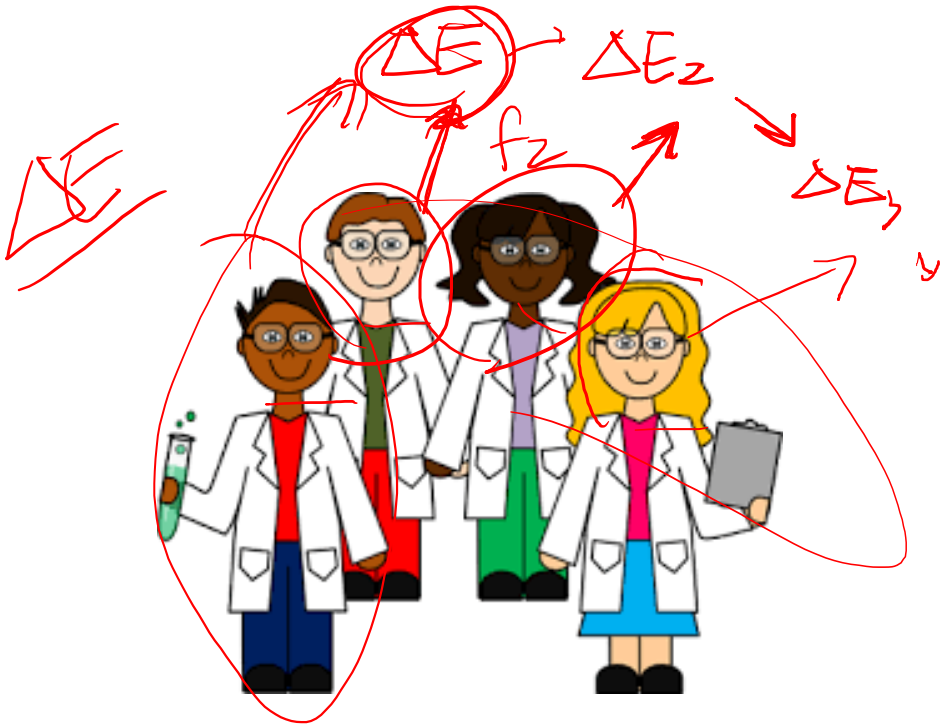
Random Forest

Ensemble method- review

Problem: Trees are weak learner and trees overfit

Idea 3: Let's make the trees a strong learner

How: Grow a small tree (stump) to fit residual sequentially



Boosting

Boosting

1. Set $\hat{f}(x) = 0$ and $r_i = y_i$ for all i in the training set.

2. For $b = 1, 2, \ldots, B$, repeat:

   (a) Fit a tree $\hat{f}^b$ with d splits ($d+1$ terminal nodes) to the training data (X, r).

   (b) Update $\hat{f}$ by adding in a shrunken version of the new tree:

$$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x).$$

   $\lambda \ll 1$

   (c) Update the residuals,

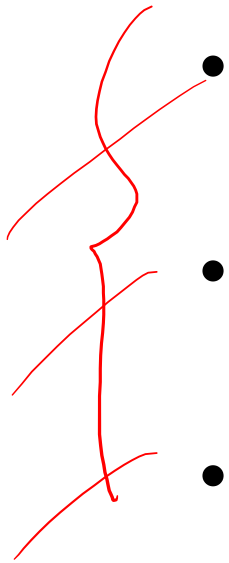
$$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i).$$

3. Output the boosted model,

$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x).$$

ISLR. *p.323*

Popular Boosted Tree Methods



- AdaBoost (Adaptive Boosting)
- GBM (Gradient Boosting Machine)
- XGBoost (Extreme Gradient Boosting)