INT 303 BIG DATA ANALYTICS

Lecture11: Decision Trees, Bagging and Random Forest

Pengfei FAN
pengfei.fan@xjtlu.edu.cn

Xi'an Jiaotong-Liverpool University
西交利物浦大學

Outline

- Decision Trees
- Bagging
- Random Forests
- Boosting

Decision Trees

- Motivation
- Decision Trees
- Classification Trees
- Regression Trees

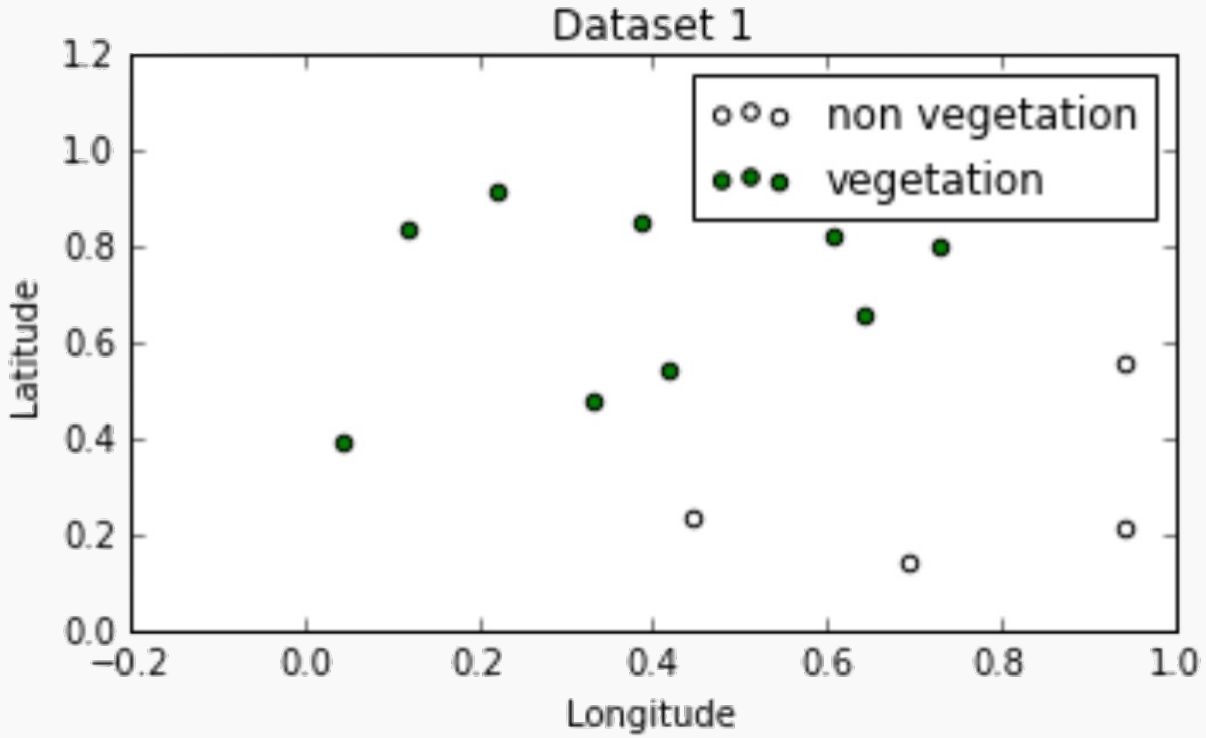
Text Reading: Section 8.1, p. 303-314 in the textbook.

Decision Trees

Geometry of Data

Recall:

logistic regression for classification works best when the classes are well-separated in the feature space



Geometry of Data

Recall:

the decision boundary is defined where the probability of being in class 1 and class 0 are equal, i.e.

$$P(Y = 1) = 1 - P(Y = 1) \quad \Rightarrow. \quad P(Y = 1) = 0.5,$$

Which is equivalent to when the `log-odds=0`:

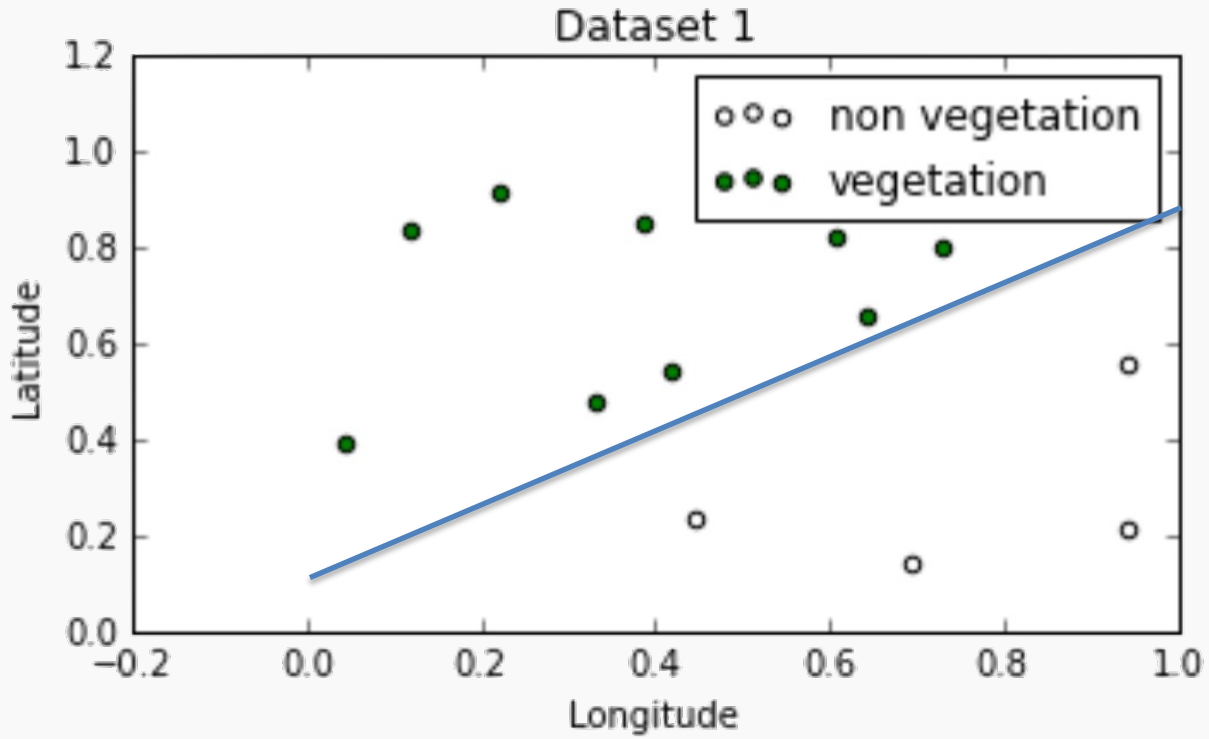
$$\boldsymbol{x}\beta = 0,$$

this equation defines a line or a hyperplane.

Geometry of Data

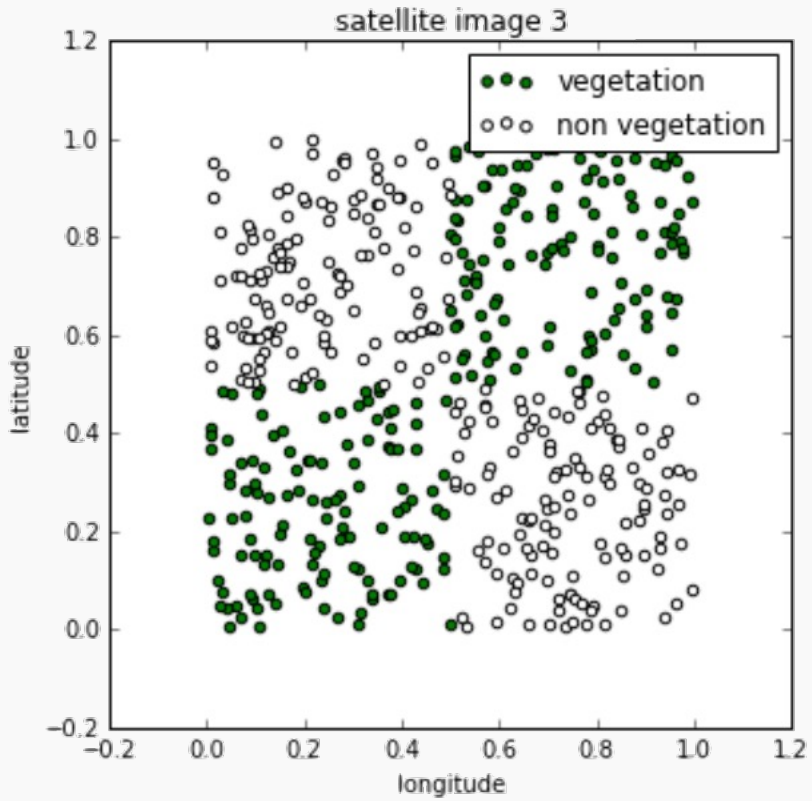
Can you guess the equation that defines the decision boundary below?

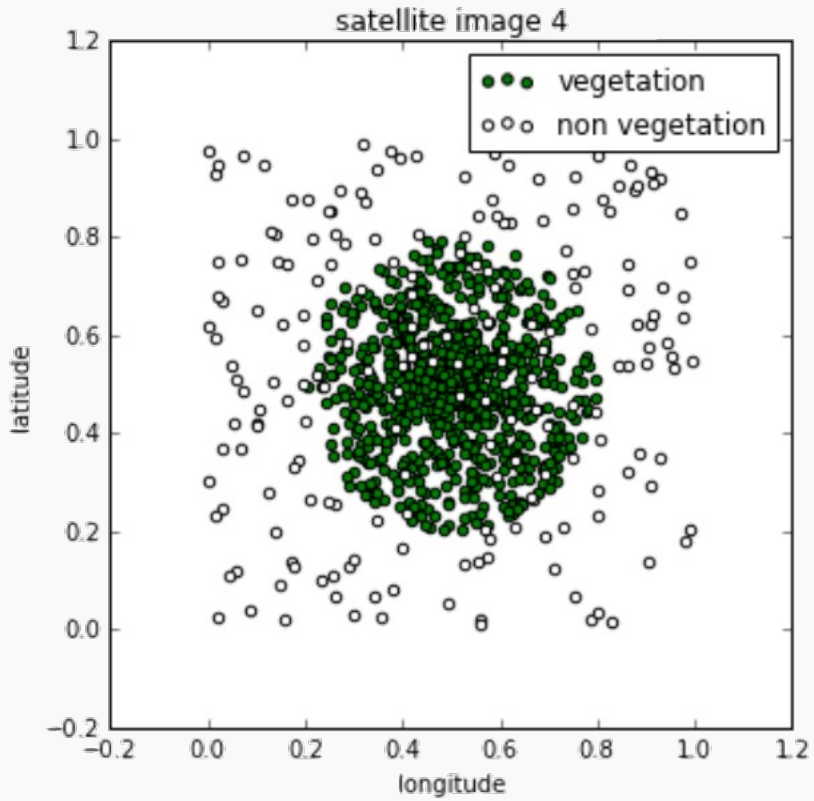
$$-0.8x_1 + x_2 = 0 \implies x_2 = 0.8x_1 \Rightarrow Latitude = 0.8\, Lon$$



Geometry of Data

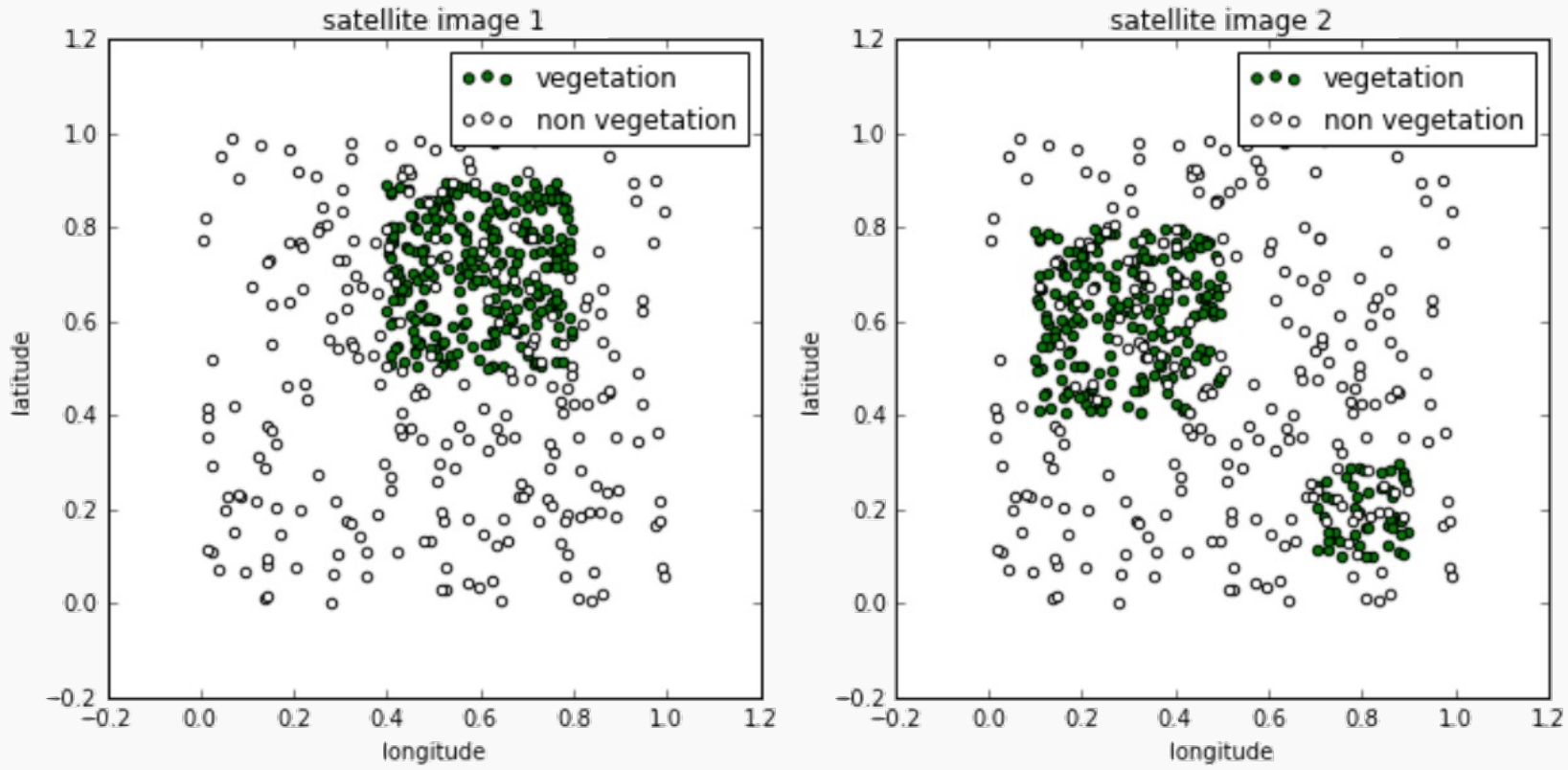
Question: How about these?





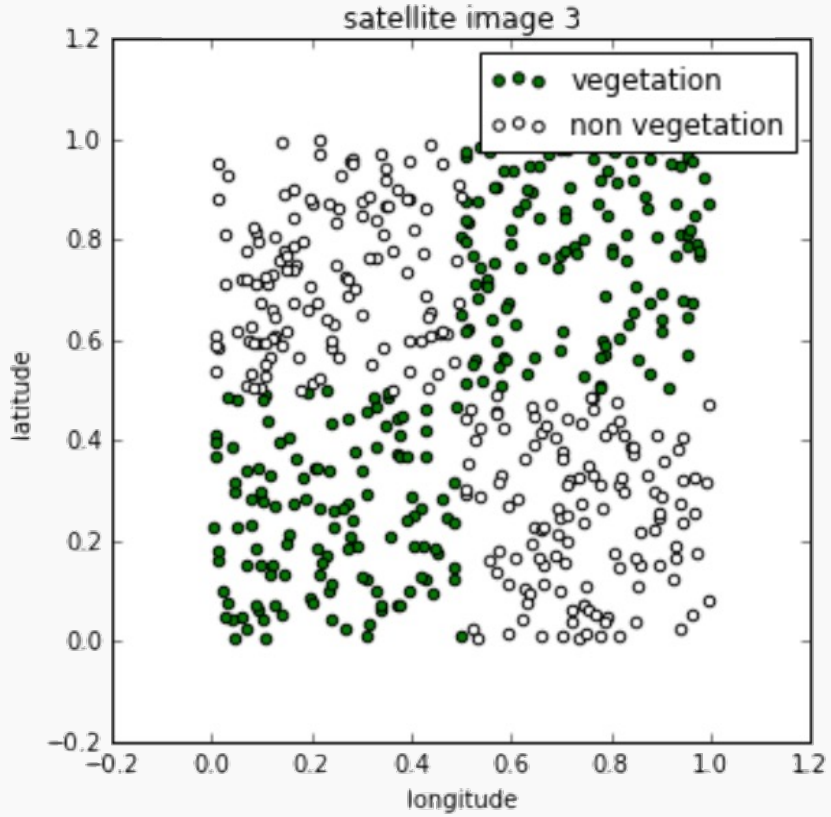
Geometry of Data

Question: Or these?



Geometry of Data

Notice that in all of the datasets the classes are still well-separated in the feature space, but ***the decision boundaries cannot be described by single equations***:



Geometry of Data

While logistic regression models with linear boundaries are intuitive to interpret by examining the impact of each predictor on the log-odds of a positive classification, it is less straightforward to interpret nonlinear decision boundaries in context:

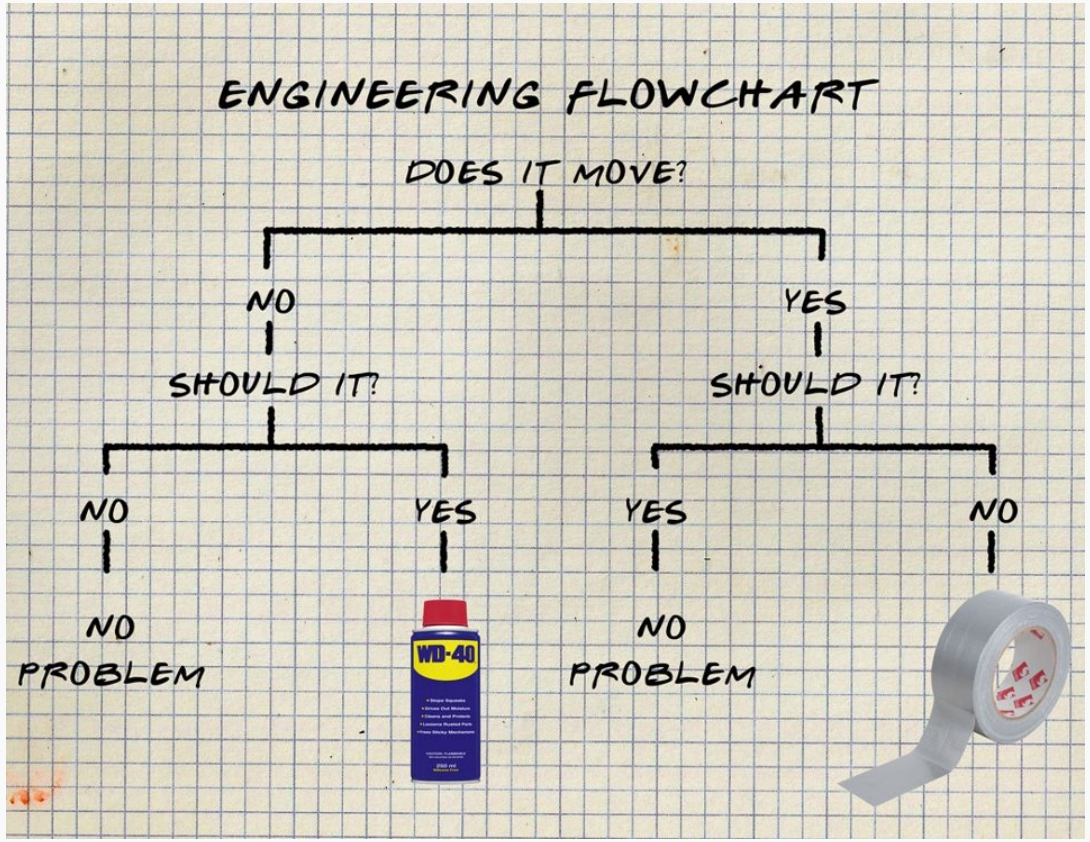
$$(x_3+2x_2) - x_1^2 + 10 = 0$$

It would be desirable to build models that:

1. allow for ***complex decision boundaries***.
2. are also ***easy to interpret***.

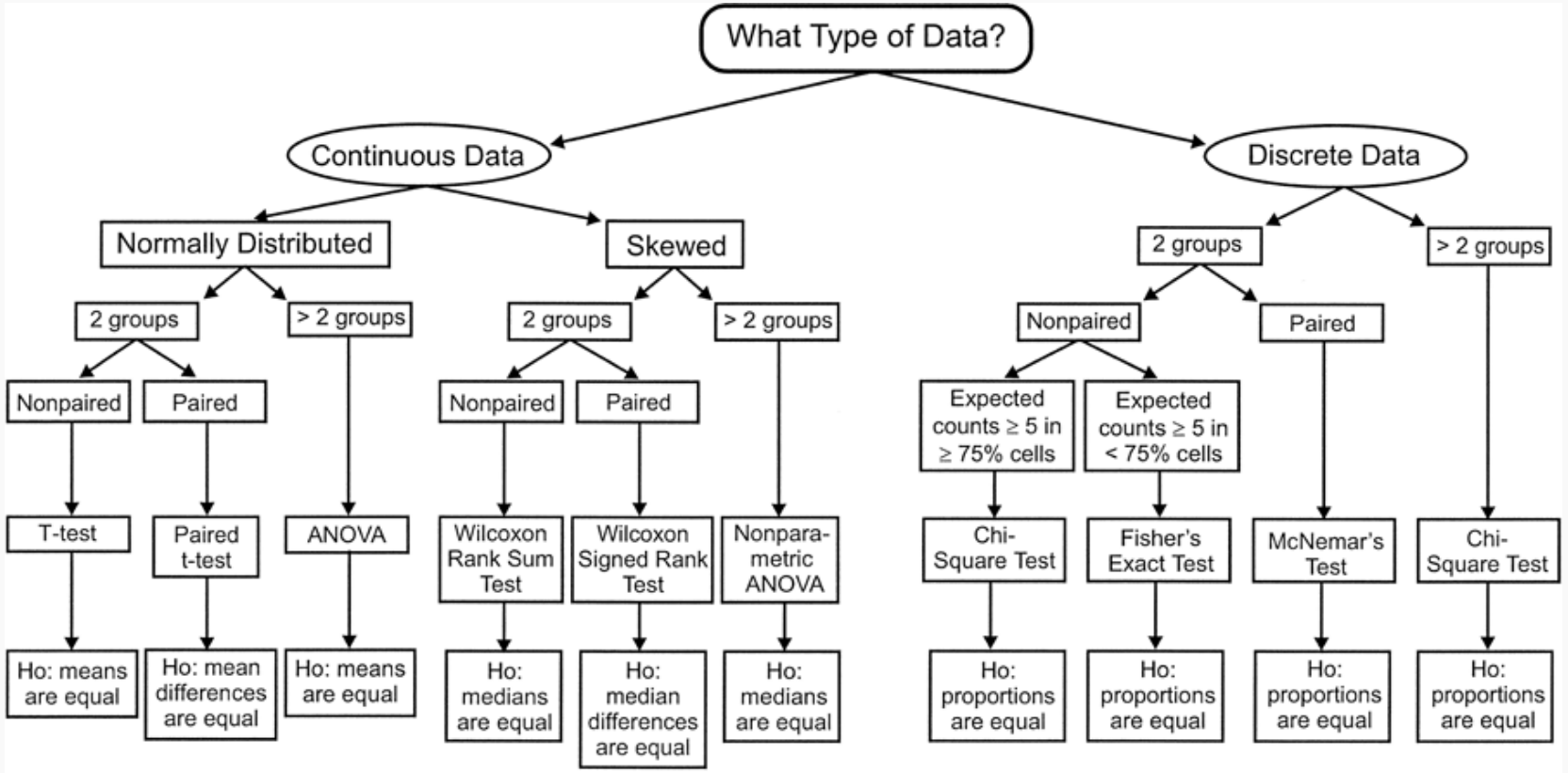
Interpretable Models

But people in every walk of life have long been using interpretable models for differentiating between classes of objects and phenomena:



Interpretable Models

But people in every walk of life have long been using interpretable models for differentiating between classes of objects and phenomena:



Source: Waning B, Montagne M: *Pharmacoepidemiology: Principles and Practice*: http://www.accesspharmacy.com

Decision Trees

It turns out that the simple flow charts in our examples can be formulated as mathematical models for classification and these models have the properties we desire; they are:

1. interpretable by humans
2. have sufficiently complex decision boundaries
3. the decision boundaries are locally linear, each component of the decision boundary is simple to describe mathematically.

The Geometry of Flow Charts

Flow charts whose graph is a tree (connected and no cycles) represents a model called a ***decision tree***.

Formally, a ***decision tree model*** is one in which the final outcome of the model is based on a series of comparisons of the values of predictors against threshold values.

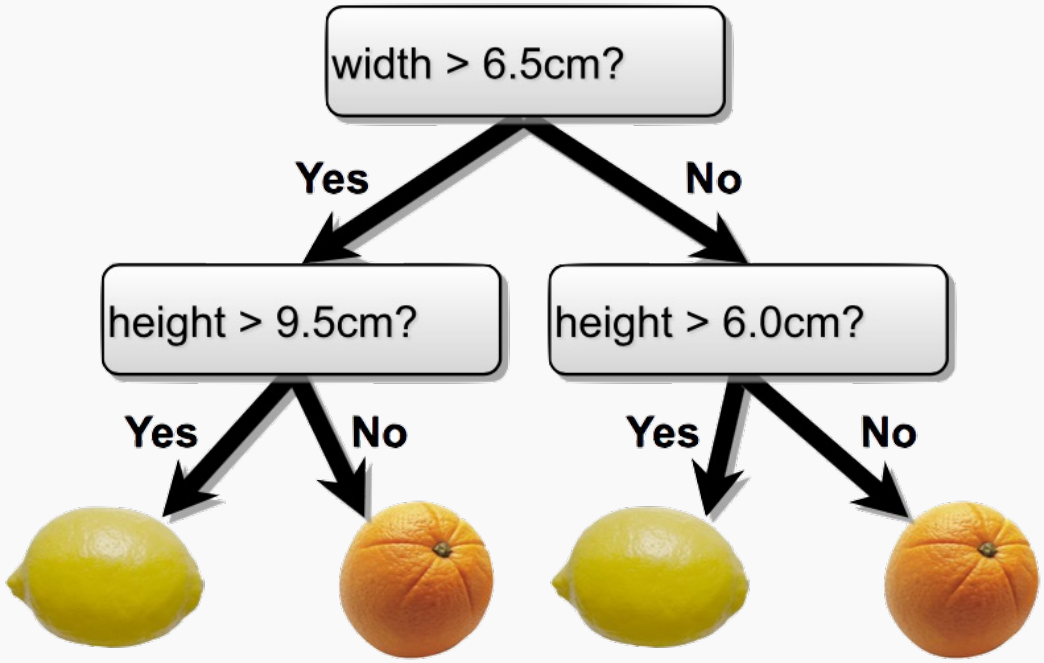
In a graphical representation (flow chart),

- the internal nodes of the tree represent attribute testing
- branching in the next level is determined by attribute value
- leaf nodes represent class assignments

The Geometry of Flow Charts

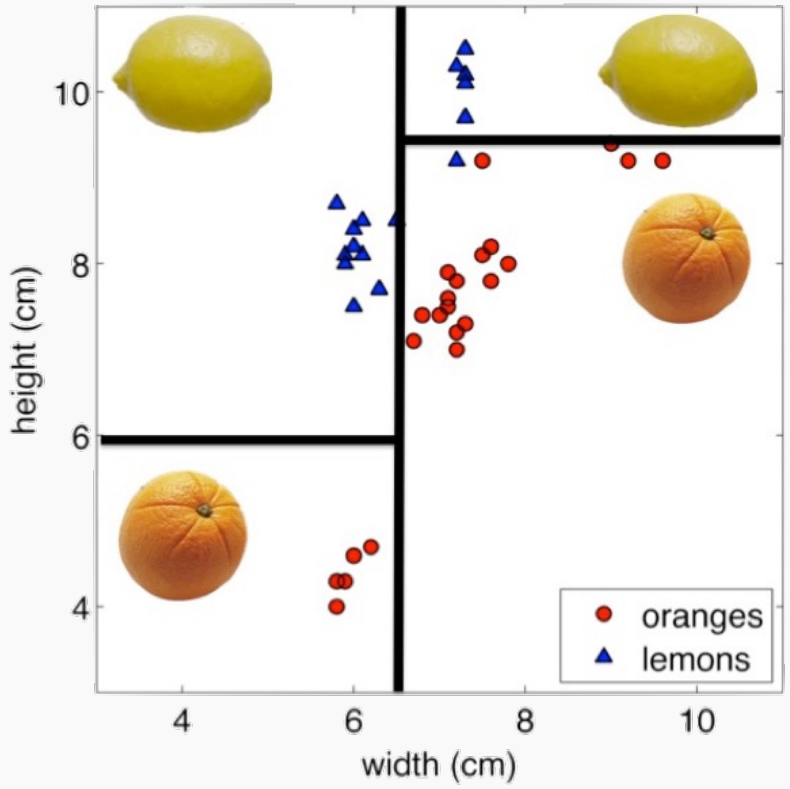
Flow charts whose graph is a tree (connected and no cycles) represents a model called a ***decision tree***.

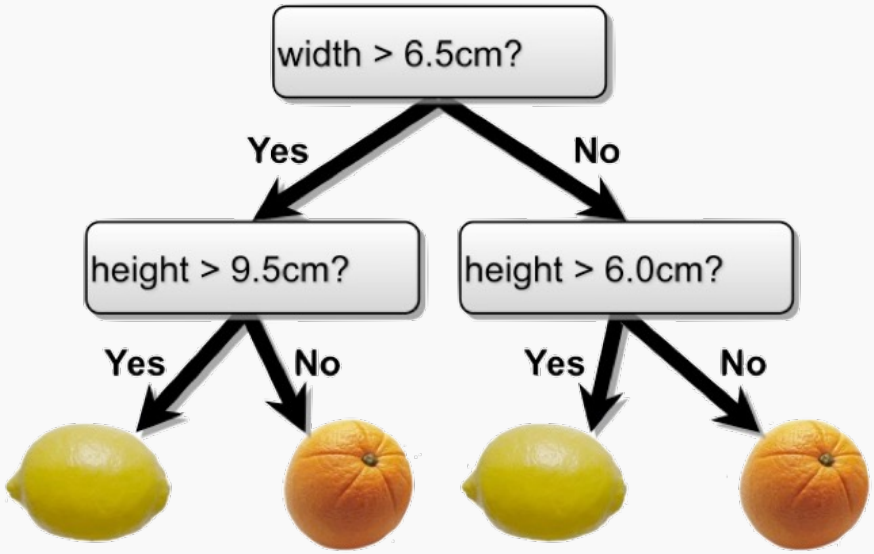
Formally, a ***decision tree model*** is one in which the final outcome of the model is based on a series of comparisons of the values of predictors against threshold values.



The Geometry of Flow Charts

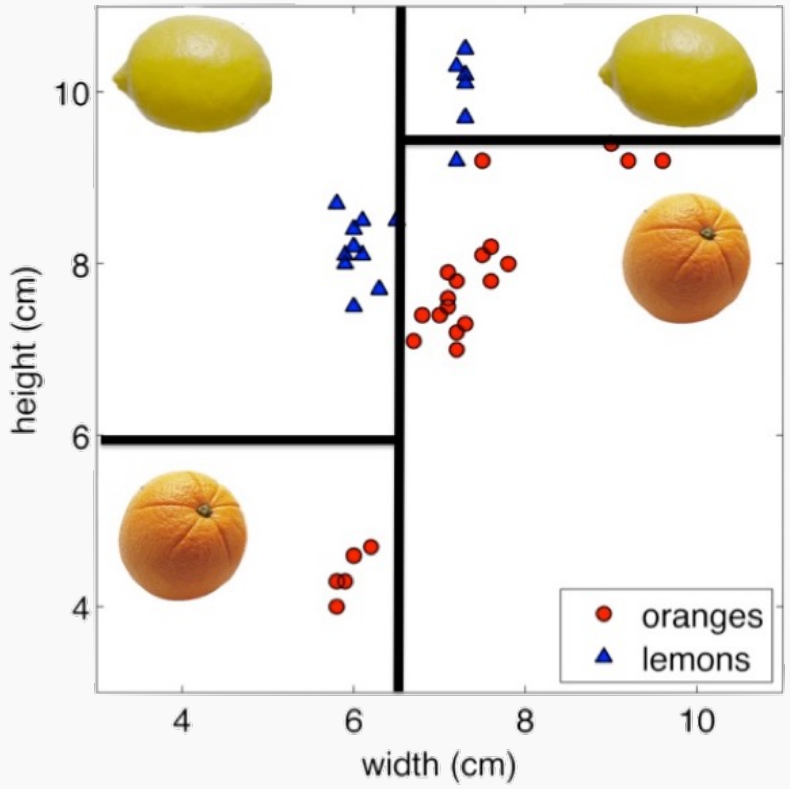
Every flow chart tree corresponds to a partition of the feature space by *axis aligned lines* or (hyper) planes. Conversely, every such partition can be written as a flow chart tree.

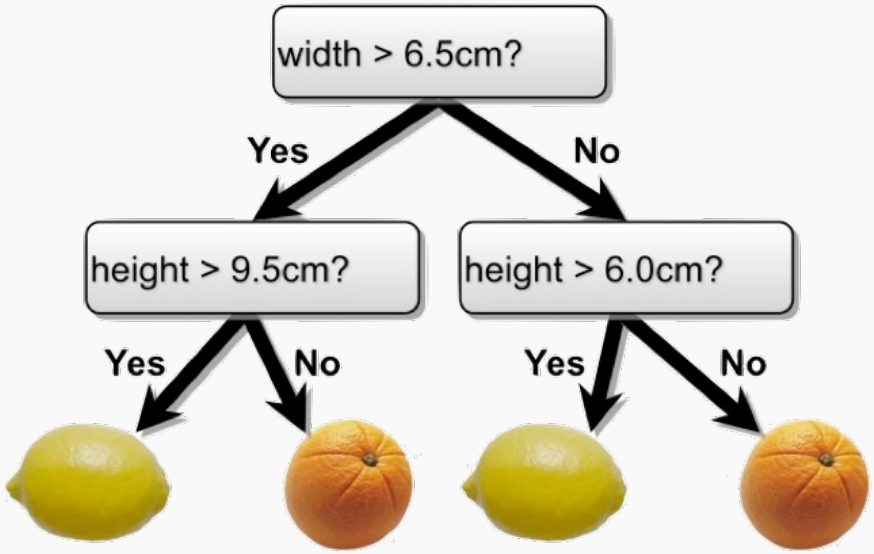




The Geometry of Flow Charts

Each comparison and branching represents splitting a region in the feature space on a single feature. Typically, at each iteration, we split once along one dimension (one predictor).





Learning the Model

Given a training set, learning a decision tree model for binary classification means:

- to produce an ***optimal*** partition of the feature space with axis-aligned linear boundaries,
- wherein each region is given a class label based on the **largest class** of the training points in that region (bayes' classifier) when performing prediction.

Learning the Model

We will seek a reasonable model using a greedy algorithm.

1. Start with an empty decision tree (undivided feature space)
2. Choose the 'optimal' predictor on which to split and choose the 'optimal' threshold value for splitting.
3. Recurse on on each new node until ***stopping condition*** is met

So, we need only define our splitting criterion and stopping condition.

Variance vs Bias

If we don't terminate the decision tree learning algorithm manually, the tree will continue to grow until each region defined by the model possibly contains exactly one training point (and the model attains 100% training accuracy).

To prevent this from happening, we can simply stop the algorithm at a particular depth.

But how do we determine the appropriate depth?

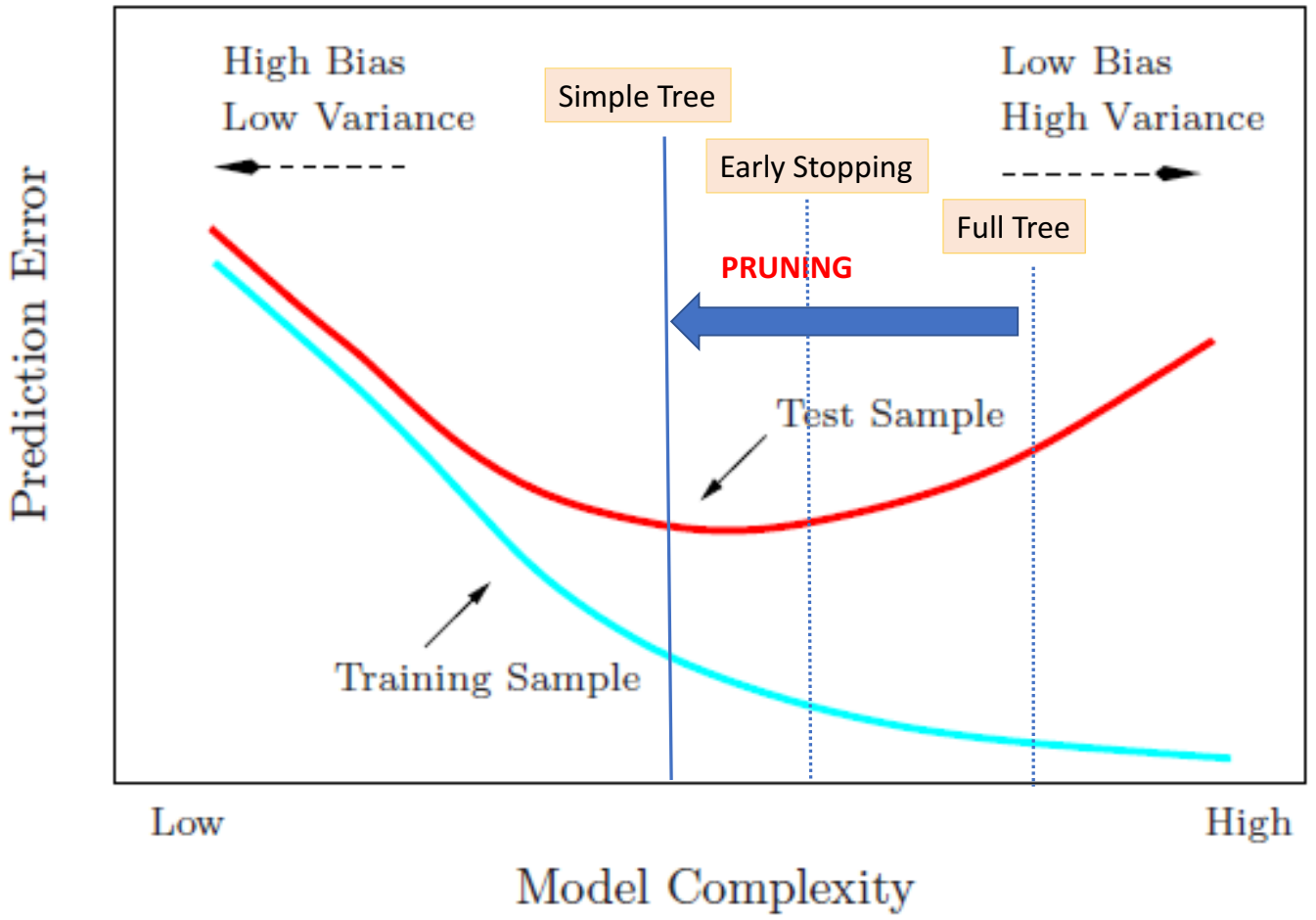
Stopping Conditions

The most common simple stopping condition is to limit the maximum depth of the tree.

The appropriate `max_depth` can be determined by evaluating the model on a validation data set or, better yet, with

cross-validation

Motivation for Pruning



Pruning

Rather than preventing a complex tree from growing, we can obtain a simpler tree by 'pruning' a complex one.

There are many method of pruning, a common one is ***cost complexity pruning***, where by we select from an array of smaller subtrees of the full model that optimizes a balance of performance and efficiency.

Regression Trees

Regression Trees

How can this decision tree approach apply to a ***regression problem*** (quantitative outcome)?

Questions to consider:

- What would be a reasonable loss function?
- How would you determine any splitting criteria?
- How would you perform prediction in each leaf?

Learning Algorithm

To learn a decision tree model, we take a greedy approach:

1. Start with an empty decision tree (undivided feature space)
2. Choose the 'optimal' predictor on which to split and choose the 'optimal' threshold value for splitting by applying a ***splitting criterion***
3. Recurse on on each new node until ***stopping condition*** is met

For classification, we label each region in the model with the label of the class to which the plurality of the points within the region belong.

Adaptations for Regression

With just two modifications, we can use a decision tree model for regression:

1. The three splitting criteria we've examined each promoted splits that were pure - new regions increasingly specialized in a single class.

    A. **For classification**, purity of the regions is a good indicator the performance of the model.
    B. **For regression**, we want to select a splitting criterion that promotes splits that improves the predictive accuracy of the model as measured by, say, the MSE.

2. For regression with output in $\mathbb{R}$, we want to label each region in the model with a real number - typically the average of the output values of the training points contained in the region.

Learning Regression Trees

The learning algorithms for decision trees in regression tasks is:

1. Start with an empty decision tree (undivided features pace)
2. Choose a predictor j on which to split and choose a threshold value t_j for splitting such that the weighted average MSE of the new regions as smallest possible:

$$\underset{j,t_j}{\operatorname{argmin}} \left\{ \frac{N_1}{N} \operatorname{MSE}(R_1) + \frac{N_2}{N} \operatorname{MSE}(R_2) \right\}$$

or equivalently,

$$\underset{j,t_j}{\operatorname{argmin}} \left\{ \frac{N_1}{N} \operatorname{Var}(y|x \in R_1) + \frac{N_2}{N} \operatorname{Var}(y|x \in R_2) \right\}$$

where N_i is the number of training points in R_i and N is the number of points in R.

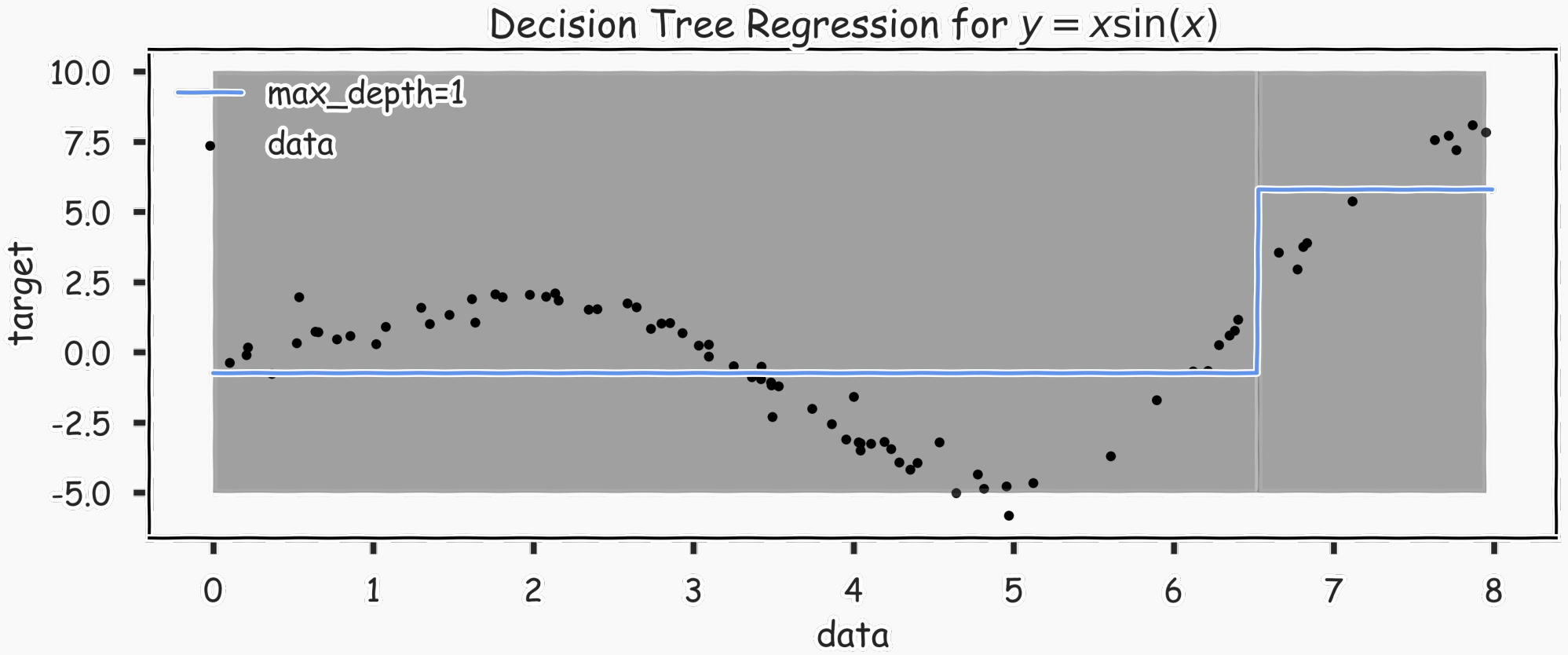
3. Recurse on on each new node until ***stopping condition*** is met

Regression Trees Prediction

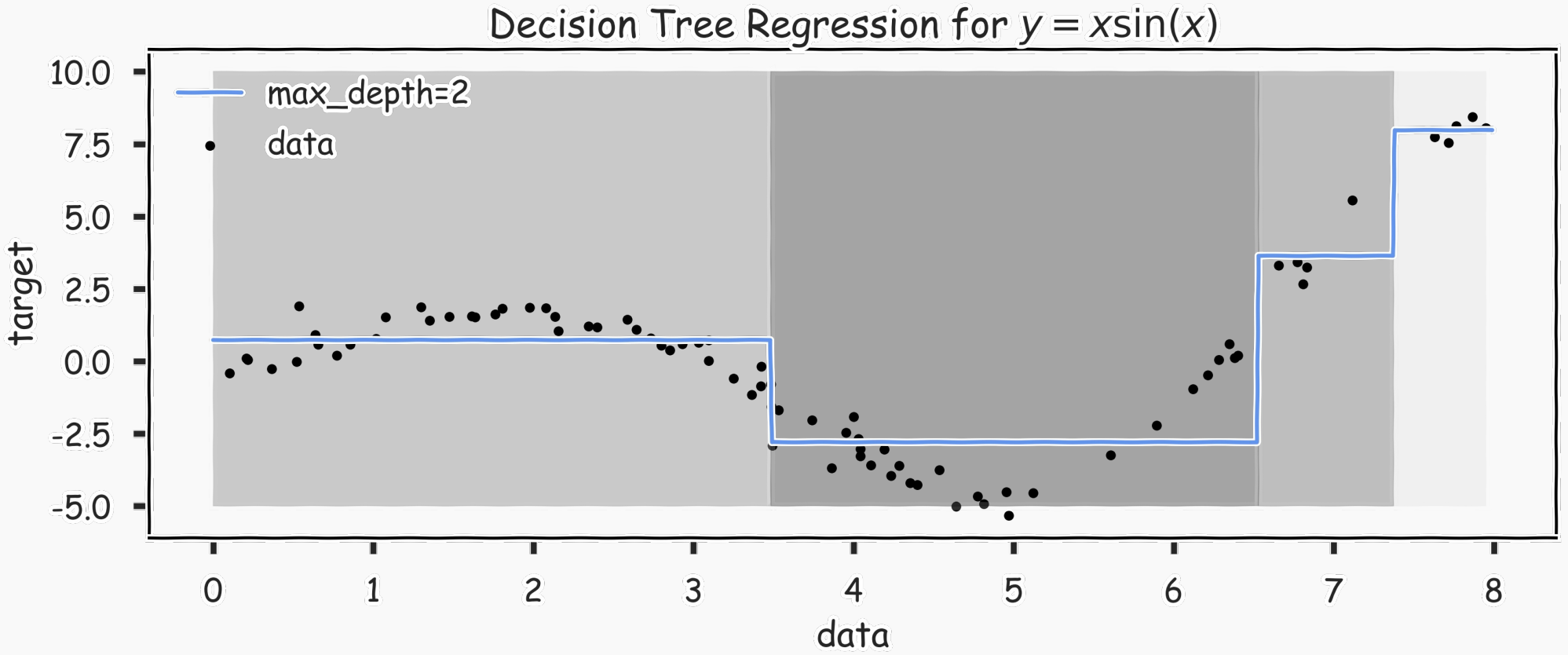
For any data point x_i

1. Traverse the tree until we reach a leaf node.
2. Averaged value of the response variable y's in the leaf (this is from the training set) is the $\hat{y}_i$.

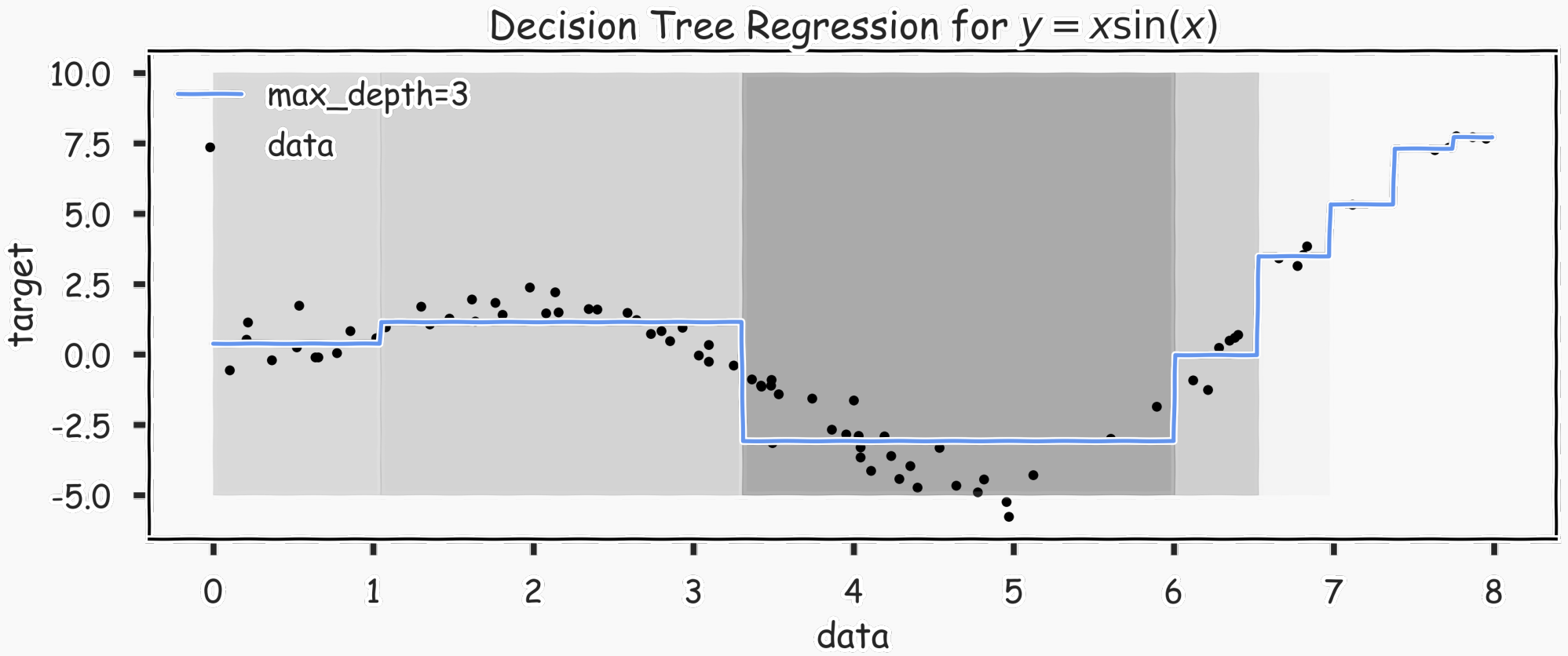
Regression Trees Prediction (grey scale represents MSE)



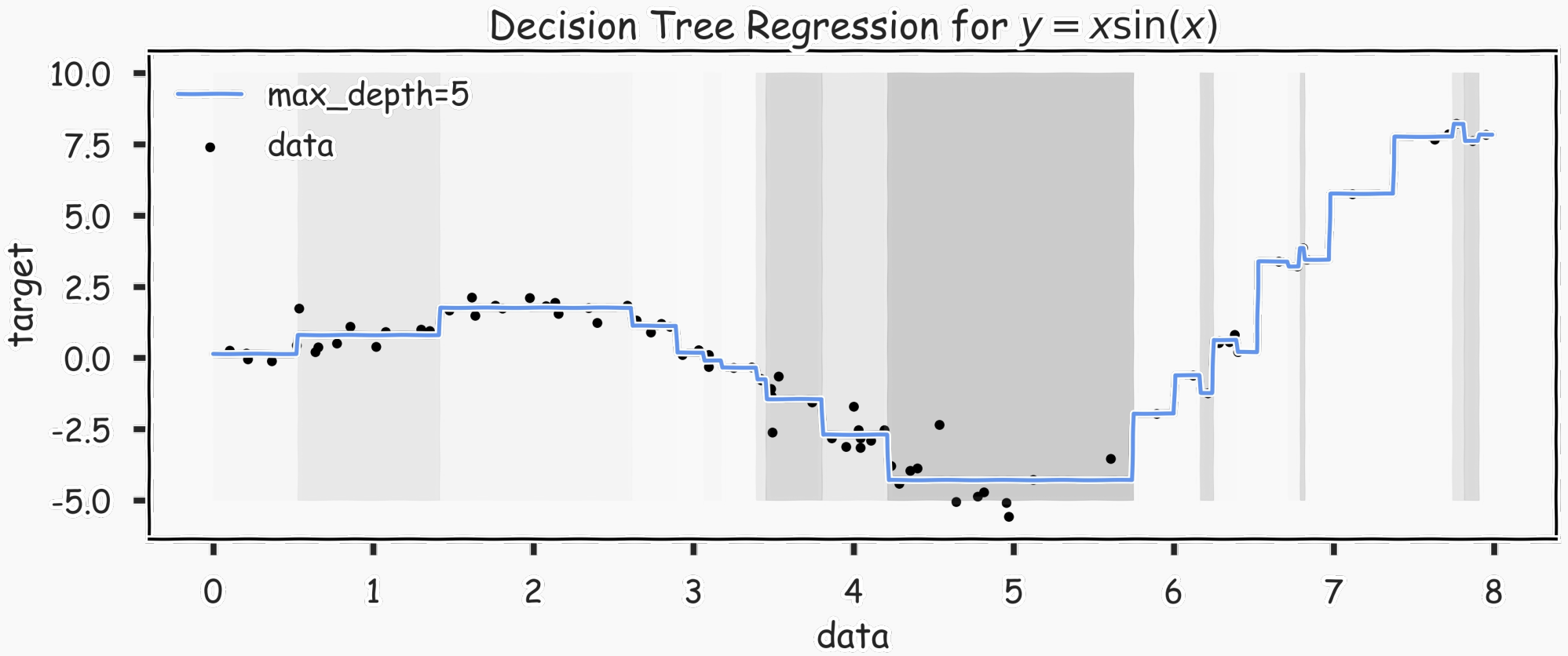
Regression Trees Prediction (grey scale represents MSE)



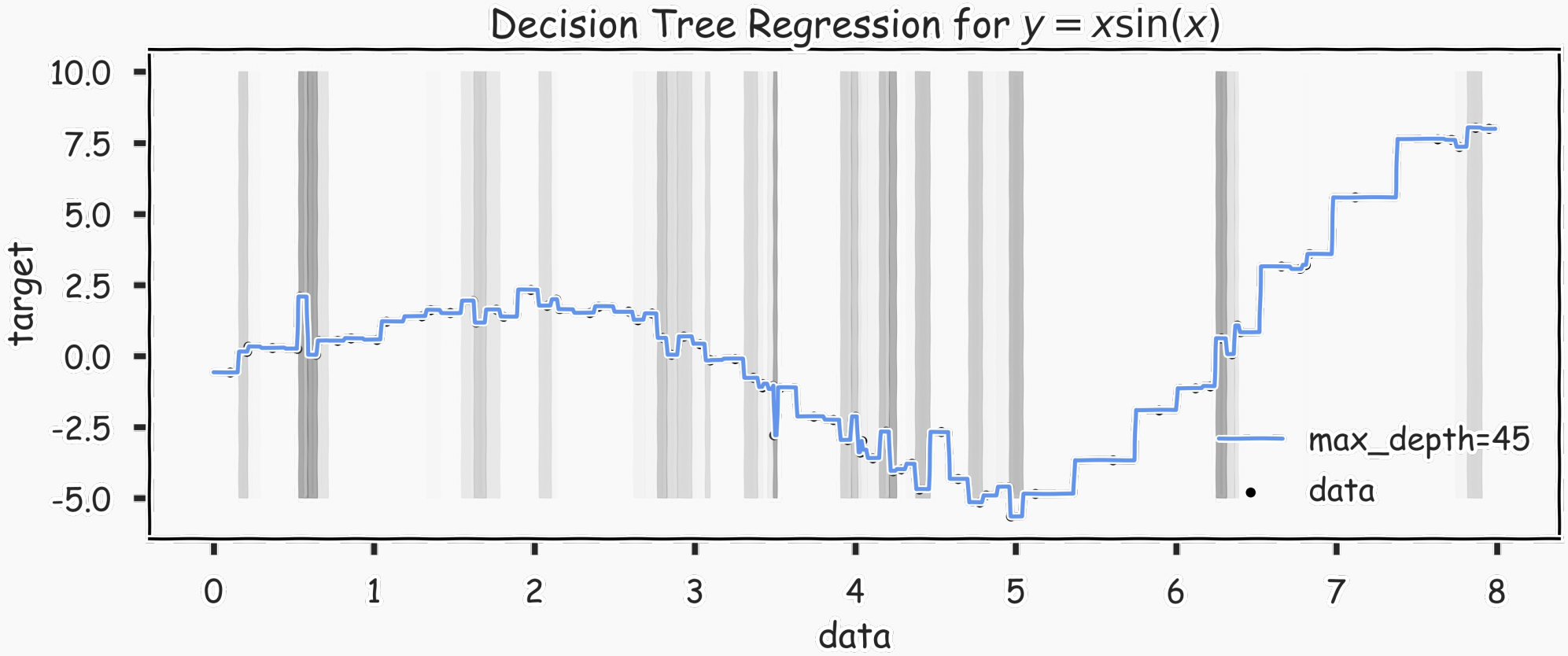
Regression Trees Prediction (grey scale represents MSE)



Regression Trees Prediction (grey scale represents MSE)

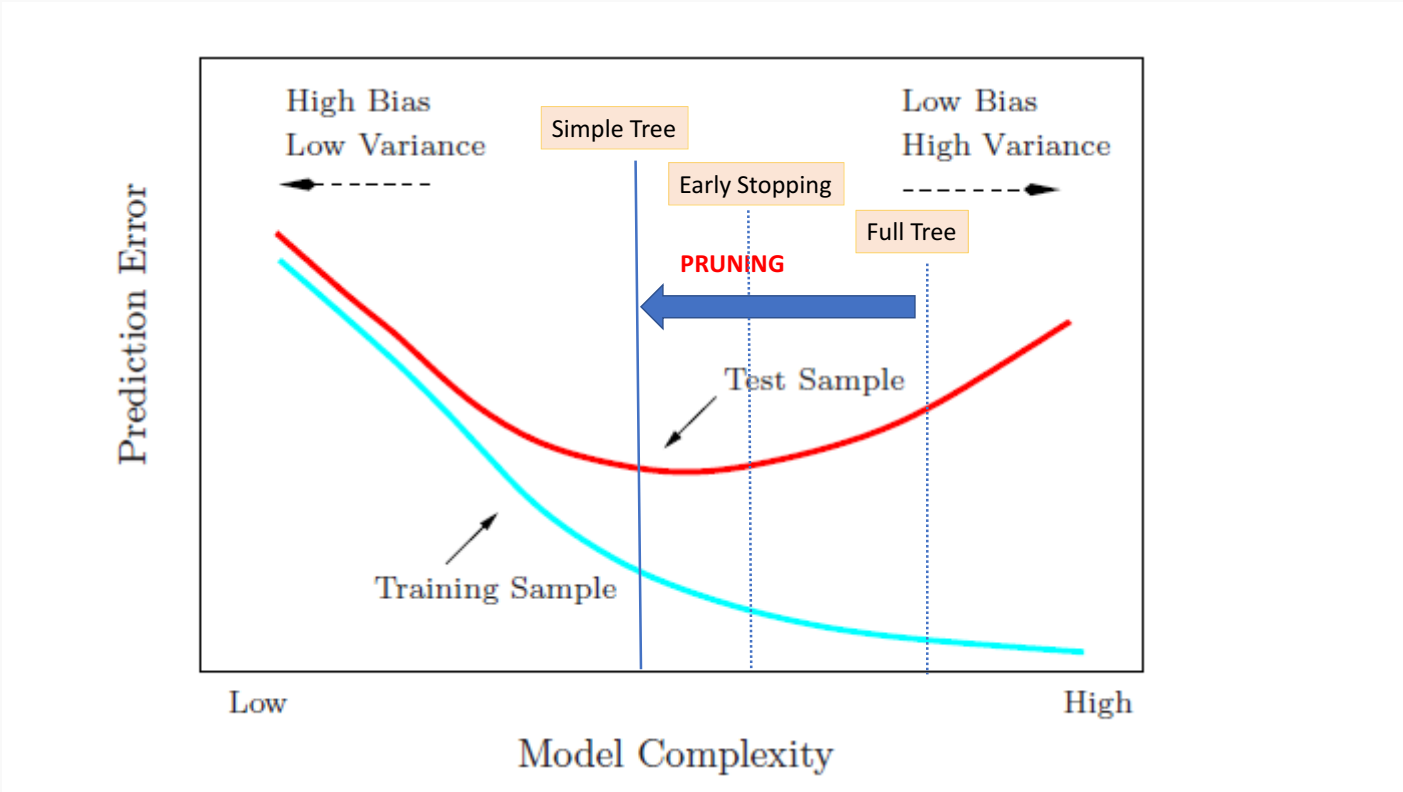


Regression Trees Prediction (grey scale represents MSE)



Overfitting

Same issues as with classification trees. Avoid overfitting by pruning or limiting the depth of the tree and using CV.



Advantages and Disadvantages of Trees

▲ Trees are very easy to explain to people. In fact, they are even easier to explain than linear regression!

▲ Some people believe that decision trees more closely mirror human decision-making than do the regression and classification approaches seen in previous lectures.

▲ Trees can be displayed graphically, and are easily interpreted even by a non-expert (especially if they are small).

▲ Trees can easily handle qualitative predictors without the need to create dummy variables.

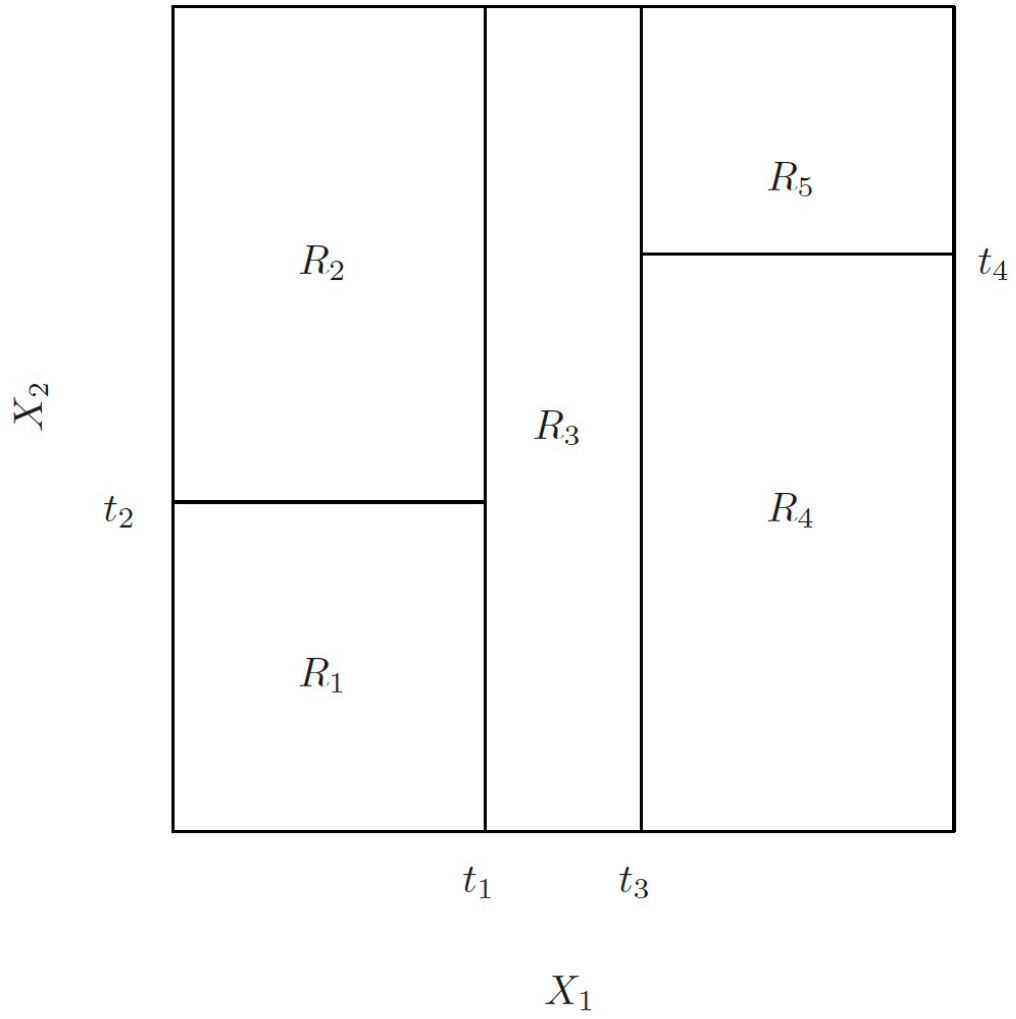
▼ Unfortunately, trees generally do not have the same level of predictive accuracy as some of the other regression and classification approaches seen in previous lectures.

▼ Additionally, trees can be very non-robust. In other words, a small change in the data can cause a large change in the final estimated tree.

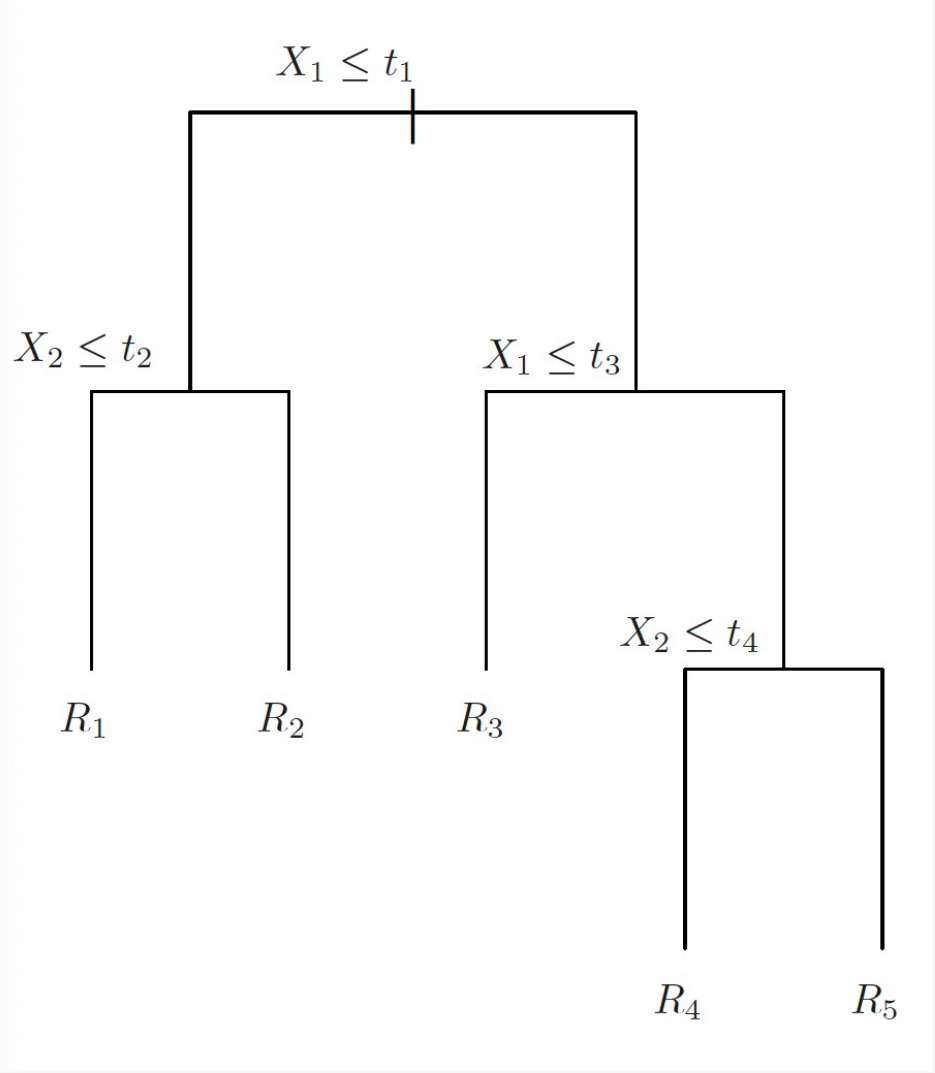
However, by aggregating many decision trees, using methods like bagging, random forests, and boosting, the predictive performance of trees can be substantially improved.

Question:

Sketch the tree corresponding to the partition of the predictor space



Create partitions of the predictor space using the tree



Tree Ensemble Methods

- Bagging
- Random Forests
- Boosting

Ensemble Philosophy

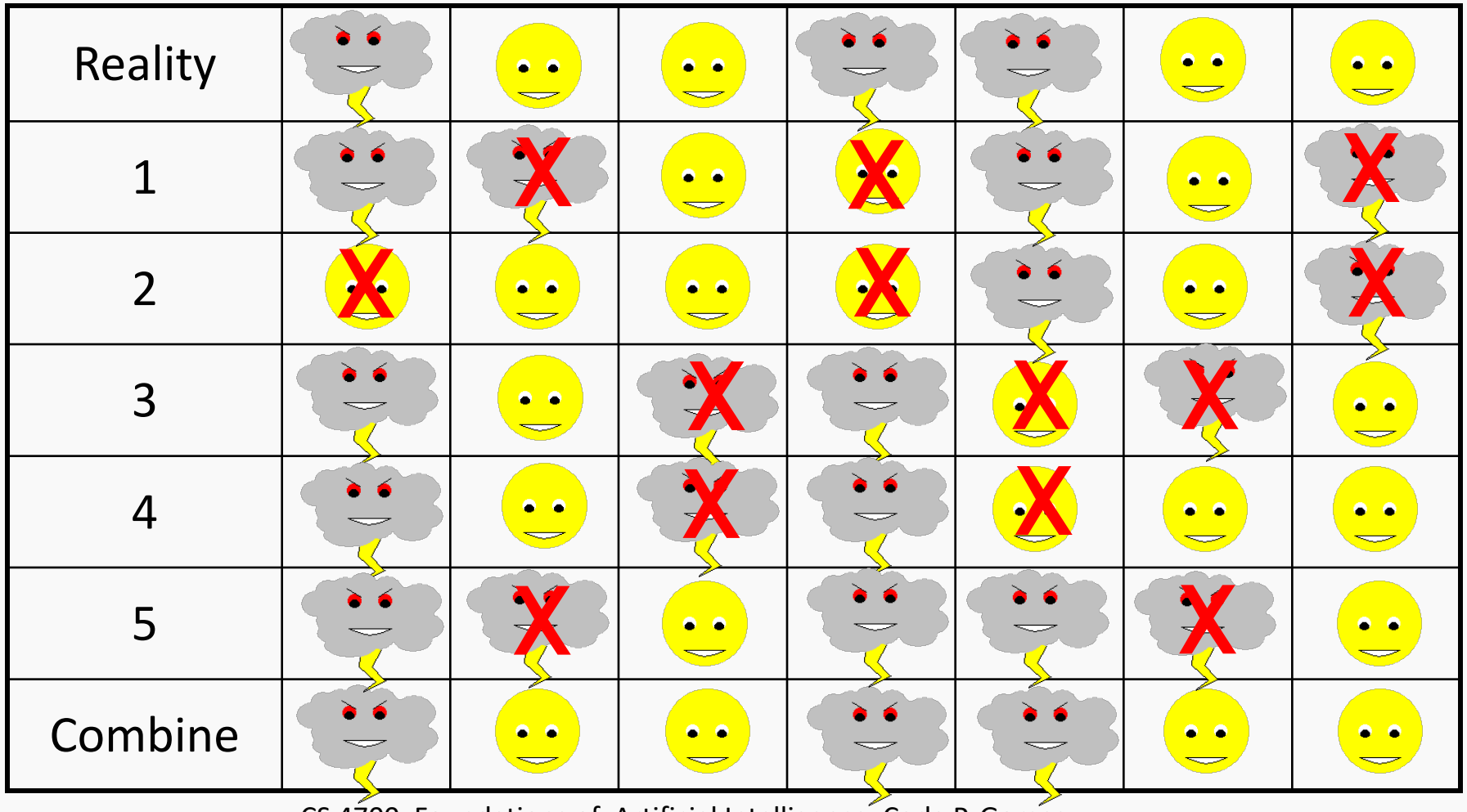
- Build many models and combine them
- Only through averaging do we get at the truth!
- It's too hard (*impossible*?) to build a single model that works best
- Two types of approaches:
  — Models that don't use randomness
  — Models that incorporate randomness

Ensembles of Classifiers

- Idea
  — Combine the classifiers to improve the performance

- Ensembles of Classifiers
  — Combine the classification results from different classifiers to produce the final output
    — Unweighted voting
    — Weighted voting

Example: Weather Forecast



Bagging

Bagging

One way to adjust for the high variance of the output of an experiment is to perform the experiment multiple times and then average the results.

The same idea can be applied to high variance models:

1. **(Bootstrap)** we generate multiple samples of training data, via bootstrapping. We train a full decision tree on each sample of data.
2. **(Aggregate)** for a given input, we output the averaged outputs of all the models for that input.

For classification, we return the class that is outputted by the plurality of the models. For regression we return the average of the outputs for each tree.

This method is called ***Bagging*** (Breiman, 1996), short for, of course, **B**ootstrap **Agg**regat**ing**.

Bagging

Given data: $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$

For $m = 1 : M$

Obtain bootstrap sample D_m from the training data D

Build a model $G_m(\mathbf{x})$ from bootstrap data D_m

Bagging

- Regression

$$\hat{y} = \frac{1}{M}\sum_{m=1}^{M} G_m(\mathbf{x})$$

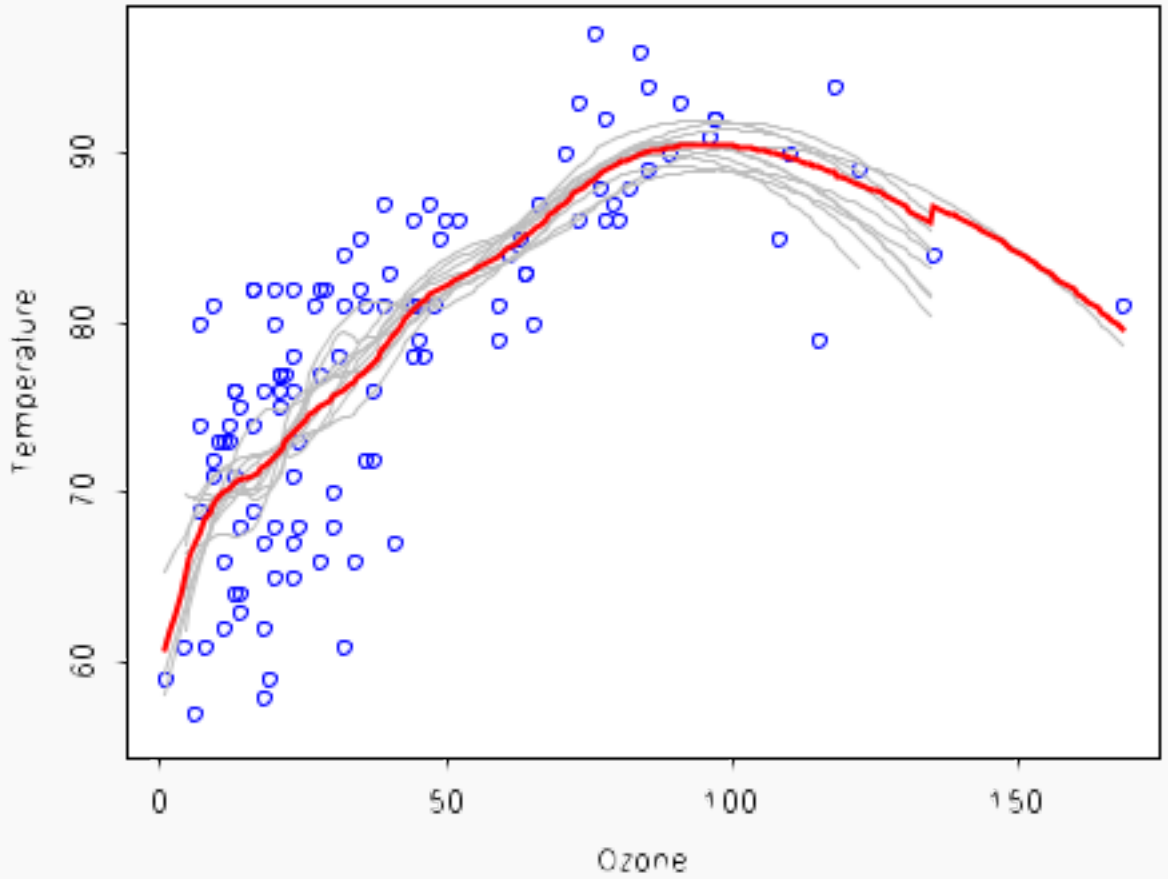
- Classification:
  - Vote over classifier outputs $G_1(\mathbf{x}),\ldots,G_M(\mathbf{x})$

Bagging

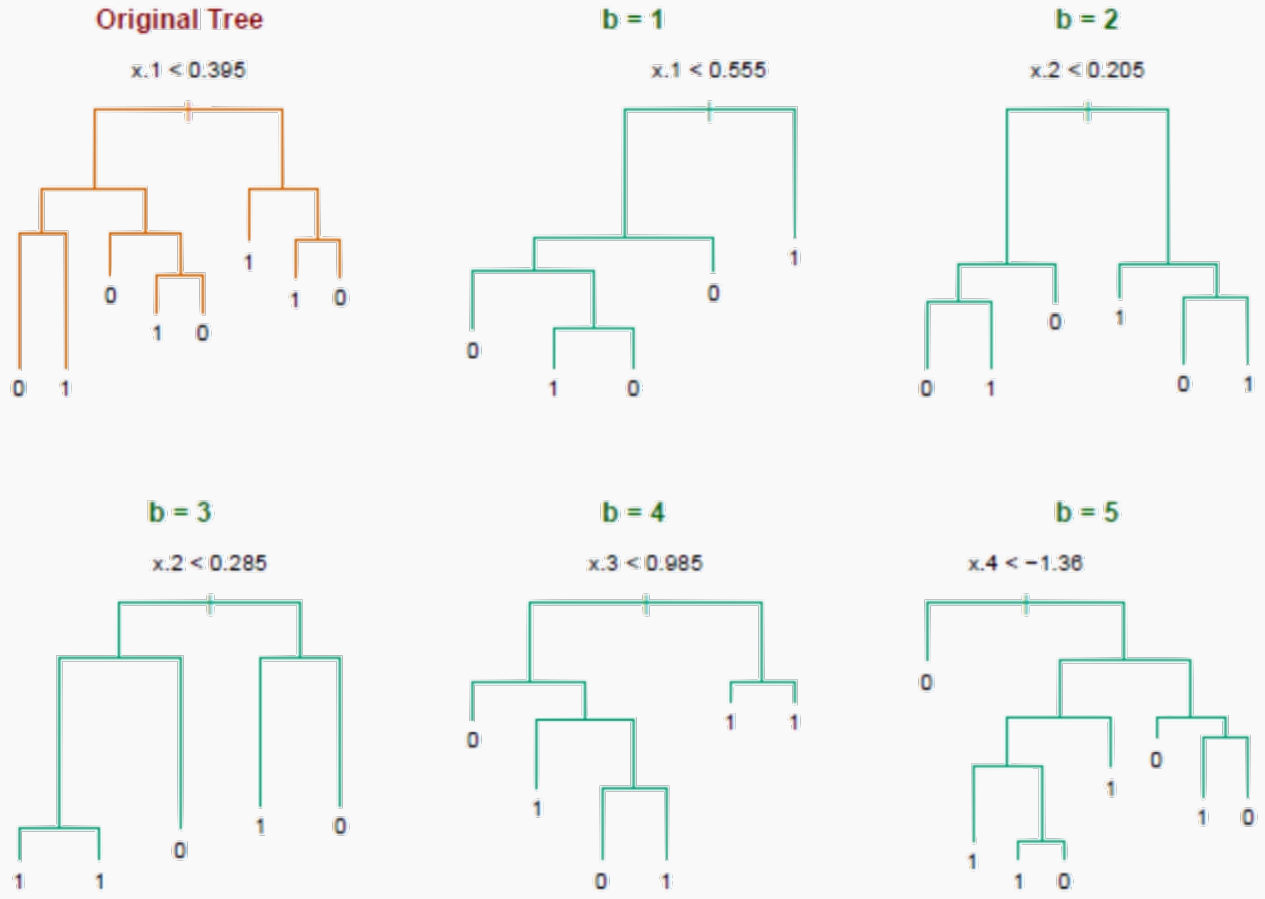
Note that bagging enjoys the benefits of:

1. High expressiveness - by using full trees each model is able to approximate complex functions and decision boundaries.
2. Low variance - averaging the prediction of all the models reduces the variance in the final prediction, assuming that we choose a sufficiently large number of trees.

Bagging



Bagging


Original Tree
x.1 < 0.395
b = 1
x.1 < 0.555
b = 2
x.2 < 0.205
b = 3
x.2 < 0.285
b = 4
x.3 < 0.985
b = 5
x.4 < −1.36

Bagging

Question: Do you see any problems?

- Still some overfitting if the trees are too large
- If trees are too shallow it can still underfit.

Cross Validations

- Interpretability — The **major drawback** of bagging (and other ***ensemble methods*** that we will study) is that the averaged model is no longer easily interpretable - i.e. one can no longer trace the 'logic' of an output through a series of decisions based on predictor values!

Out-of-Bag Error

Out-of-Bag Error

Bagging is an example of an ***ensemble method***, a method of building a single model by training and aggregating multiple models.

With ensemble methods, we get a new metric for assessing the predictive performance of the model, the ***out-of-bag error***.

Given a training set and an ensemble of models each trained on a bootstrap sample, we compute the ***out-of-bag error*** of the averaged model by

1. For each point in the training set, we average the predicted output for this point over the models whose bootstrap training set excludes this point. We compute the error or squared error of this averaged prediction. Call this the point-wise out-of-bag error.
2. We average the point-wise out-of-bag error over the full training set.

Bagging

Original Data

X	Y
X_1	y_1
X_2	y_2
X_3	y_3
X_4	y_4
X_5	y_5
⋮	⋮
X_n	y_n

Bootstrap Sample 1

X	Y
X_4	y_4
X_{14}	y_{14}
X_1	y_1
X_2	y_2
X_{35}	y_{35}
⋮	⋮
X_k	y_k

Decision Tree 1

Used and unused data

X	Y
X_1	y_1
X_2	y_2
X_3	y_3
X_4	y_4
X_5	y_5
⋮	⋮
X_n	y_n

Bagging

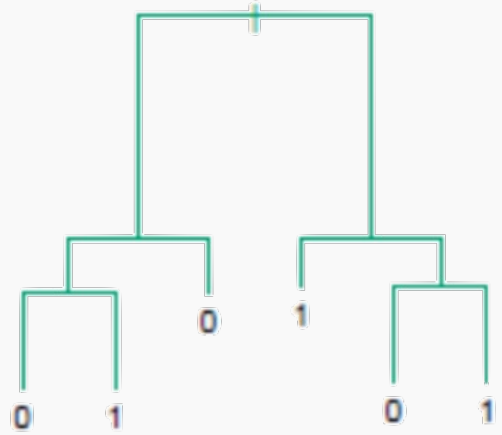
Original Data

X	Y
X_1	y_1
X_2	y_2
X_3	y_3
X_4	y_4
X_5	y_5
⋮	⋮
X_n	y_n

Bootstrap Sample 2

X	Y
X_5	y_5
X_3	y_3
X_{12}	y_{12}
X_{43}	y_{43}
X_1	y_1
⋮	⋮
X_k	y_k

Decision Tree 2



Used and unused data

X	Y
X_1	y_1
X_2	y_2
X_3	y_3
X_4	y_4
X_5	y_5
⋮	⋮
X_n	y_n

Bagging

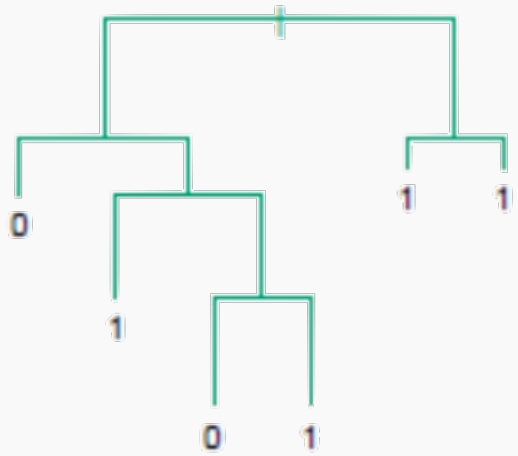
Original Data

X	Y
X_1	y_1
X_2	y_2
X_3	y_3
X_4	y_4
X_5	y_5
⋮	⋮
X_n	y_n

Bootstrap Sample 3

X	Y
X_9	y_9
X_4	y_4
X_1	y_1
X_1	y_1
X_{65}	y_{65}
⋮	⋮
X_k	y_k

Decision Tree 3



Used and unused data

X	Y
X_1	y_1
X_2	y_2
X_3	y_3
X_4	y_4
X_5	y_5
⋮	⋮
X_n	y_n

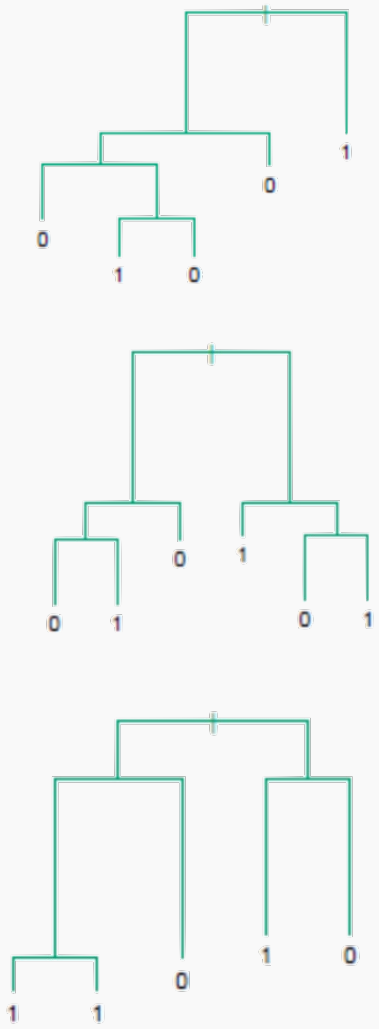
Point-wise out-of-bag error

X	Y
X_1	y_1
X_2	y_2
X_3	y_3
⋮	⋮
X_i	y_i
⋮	⋮
X_n	y_n

Point-wise out-of-bag error

B Trees that did not see $\{X_i, y_i\}$

X	Y
X_1	y_1
X_2	y_2
X_3	y_3
⋮	⋮
X_i	y_i
⋮	⋮
X_n	y_n



Classification

$$\hat{y}_{i,pw} = \text{majority}(\hat{y}_i)$$

$$e_i = \mathbb{I}(\hat{y}_{i,pw} = y_i)$$

Regression

$$\hat{y}_{i,pw} = \sum_{j \in B} \hat{y}_{i,j}$$

$$e_i = \left(y_i - \hat{y}_{i.pw}\right)^2$$

OOB Error

We average the point-wise out-of-bag error over the full training set.

Classification

$$Error_{OOB} = \sum_{i}^{n} e_i = \sum_{i}^{n} \mathbb{I}(\hat{y}_{i,pw} = \mathrm{y_i})$$

Regression

$$Error_{OOB} = \sum_{i}^{n} e_i = \sum_{i}^{n} \left(y_i - \hat{y}_{i,pw}\right)^2$$

Improving on Bagging

In practice, the ensembles of trees in Bagging tend to be highly correlated.

Suppose we have an extremely strong predictor, x_j , in the training set amongst moderate predictors. Then the greedy learning algorithm ensures that most of the models in the ensemble will choose to split on x_j in early iterations.

That is, each tree in the ensemble is identically distributed, with the expected output of the averaged model the same as the expected output of any one of the trees.

Random Forests

Random Forests

Random Forest is a modified form of bagging that creates ensembles of independent decision trees.

To de-correlate the trees, we:

1. train each tree on a separate bootstrap sample of the full training set (same as in bagging)
2. for each tree, at each split, we ***randomly*** select a set of J' predictors from the full set of predictors.

From amongst the J' predictors, we select the optimal predictor and the optimal corresponding threshold for the split.

Tuning Random Forests

Random forest models have multiple hyper-parameters to tune:

1. the number of predictors to randomly select at each split
2. the total number of trees in the ensemble
3. the minimum leaf node size

In theory, each tree in the random forest is full, but in practice this can be computationally expensive (and added redundancies in the model), thus, imposing a minimum node size is not unusual.

Tuning Random Forests

There are standard (default) values for each of random forest hyper-parameters recommended by long time practitioners, but generally these parameters should be tuned through **OOB** (making them data and problem dependent).

e.g. number of predictors to randomly select at each split:

$\sqrt{N_j}$ for classification

$\frac{N}{3}$ for regression

Using out-of-bag errors, training and cross validation can be done in a single sequence - we cease training once the out-of-bag error stabilizes

Final Thoughts on Random Forests

When the number of predictors is large, but the number of relevant predictors is small, random forests can perform poorly.

Why?
In each split, the chances of selected a relevant predictor will be low and hence most trees in the ensemble will be weak models.

Final Thoughts on Random Forests (cont.)

Increasing the number of trees in the ensemble generally does not increase the risk of overfitting.

Again, by decomposing the generalization error in terms of bias and variance, we see that increasing the number of trees produces a model that is at least as robust as a single tree.

However, if the number of trees is too large, then the trees in the ensemble may become more correlated, increase the variance.

THANK YOU

VISIT US

WWW.XJTLU.EDU.CN

FOLLOW US

@XJTLU

Xi'an Jiaotong-Liverpool University
西交利物浦大學