2016 SDSC Summer Institute
Machine Learning

SDSC SAN DIEGO SUPERCOMPUTER CENTER

UC San Diego

Classification

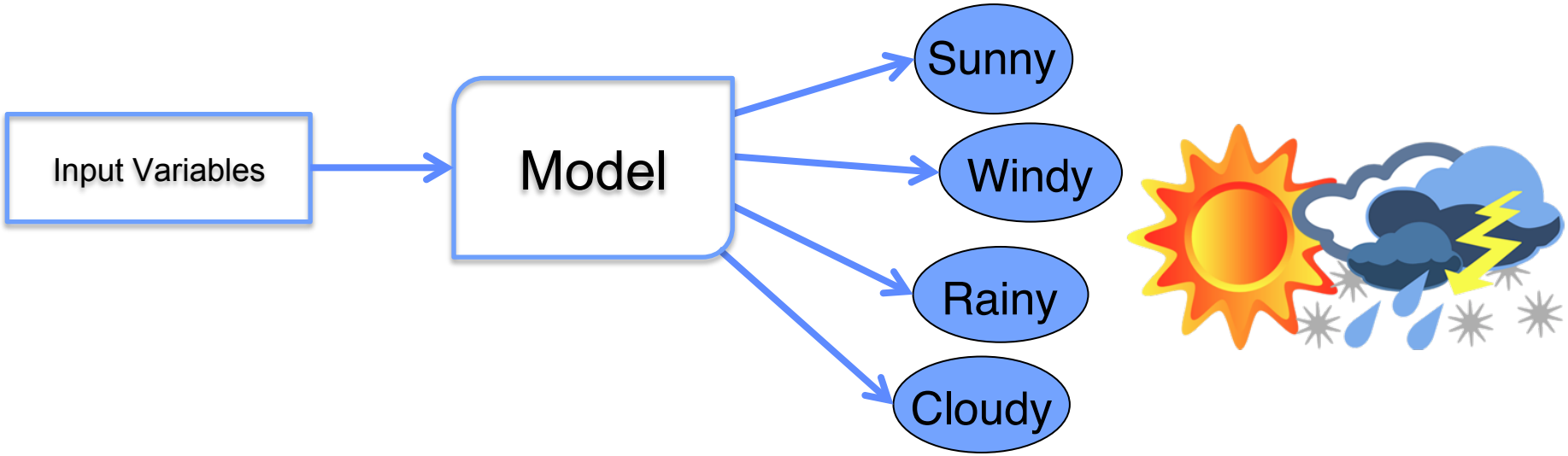
Mai H. Nguyen

Overview

- **What is classification?**
- **Terminology & concepts**
  - Training and test sets
  - Generalization & overfitting
- **Classification Algorithms**
  - K Nearest Neighbors
  - Decision Trees
  - Random Forest
  - Many others:
    - Naive Bayes, Logistic Regression, Neural Network, Support Vector Machines, etc.
- **Hands-on**
  - Build classification models for weather data

What is Classification?

- **Given input variables, predict target variable.**
  - Target variable is *categorical*.
  - Model needs to learn relationship between input data and target.
  - Other names for 'target'
    - label, class, class variable, output



Classification

Target variable

Input variables

RainTomorrow	Cloud9am	Pressure9am	Humidity9am	Temp9am
Yes	7	1019.7	68	14.4
Yes	5	1012.4	80	17.5
Yes	8	1009.5	82	15.4
Yes	2	1005.5	62	13.5
No	7	1018.3	68	11.1
No	7	1023.8	70	10.9

Classification Task: Can we predict the target variable from the input variables?

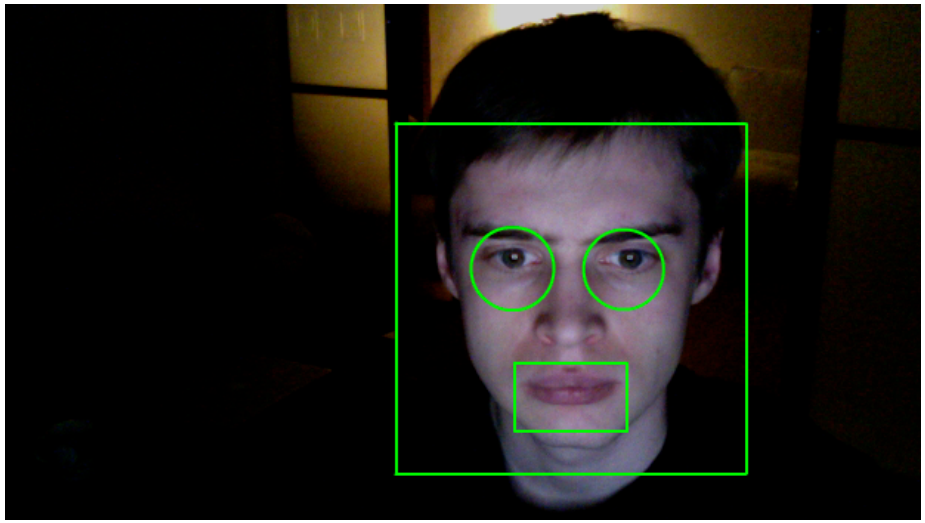
Note that classification is **supervised** learning.

Target Variable in Classification

- **In classification, the target variable is always of categorical type.**
  - Binary Classification:
    - One of two classes for each sample
  - Multinomial or Multiclass Classification:
    - One of many classes for each sample
    - “many” = more than 2

Binary Classification

- **Classify each sample as one of two classes.**
- **Example: Does this image contain a face?**
  - Yes or No



Source: http://blog.inspirit.ru/viola-jones-object-detection/

Other Examples of Binary Classification

- Should this loan application be approved?
- Is this transaction fraudulent or legitimate?
- Will it rain tomorrow or not?
- Is this tumor benign or malignant?

Character Recognition

Example of “multi-class” or “multinomial” classification problem

http://www.r-bloggers.com/a-little-h2o-deeplearning-experiment-on-the-mnist-data-set/

Other examples of multi-class problems

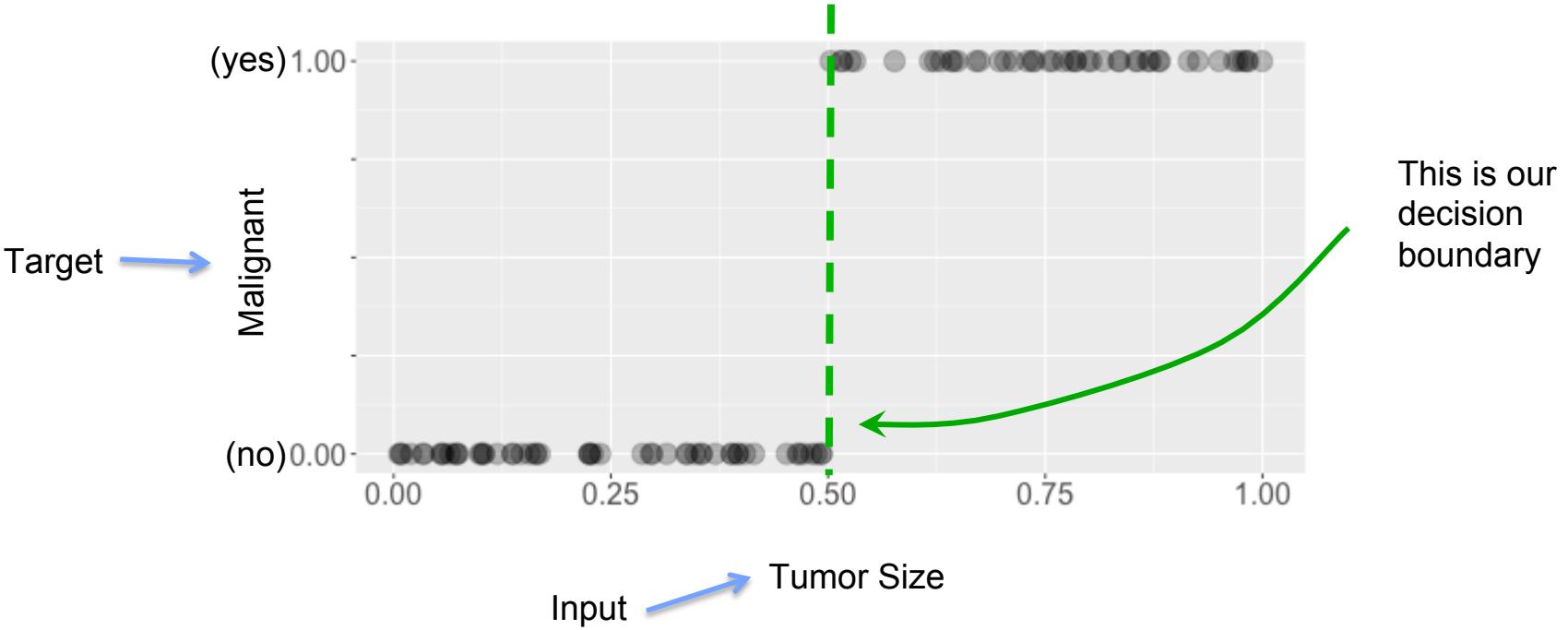
- What product will this customer buy?
- What type of customer buys this product?
- What health grade should this food get?
- Will it be sunny, rainy, or stormy tomorrow?

Building Classifier

- **In general, classifier is a mathematical model**
  - This model might not necessarily be a closed-form algebraic equation.
- **Classification model has a set of *parameters* that need to be *learned* or *estimated* from data.**
  - This is also referred to as “fitting the model”.
- **Usually, the more data we have, the better we are able to estimate the parameters.**

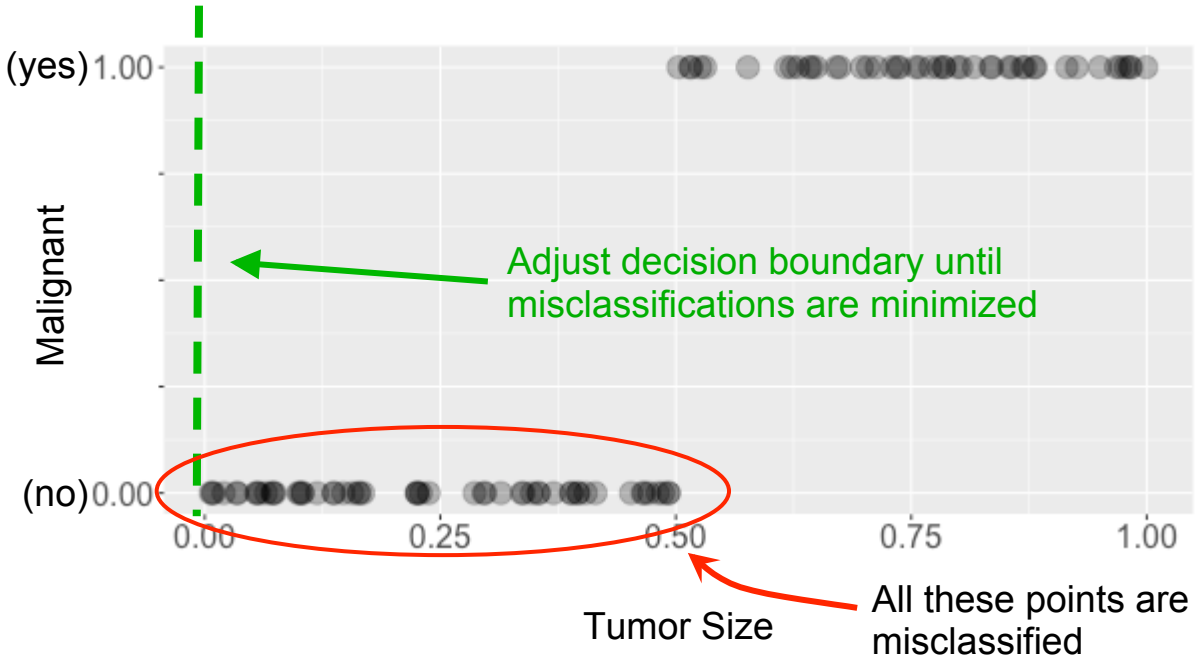
Building Classifier

For discriminative classifiers, to "build" a classifier means finding a decision boundary:



Building Classifier

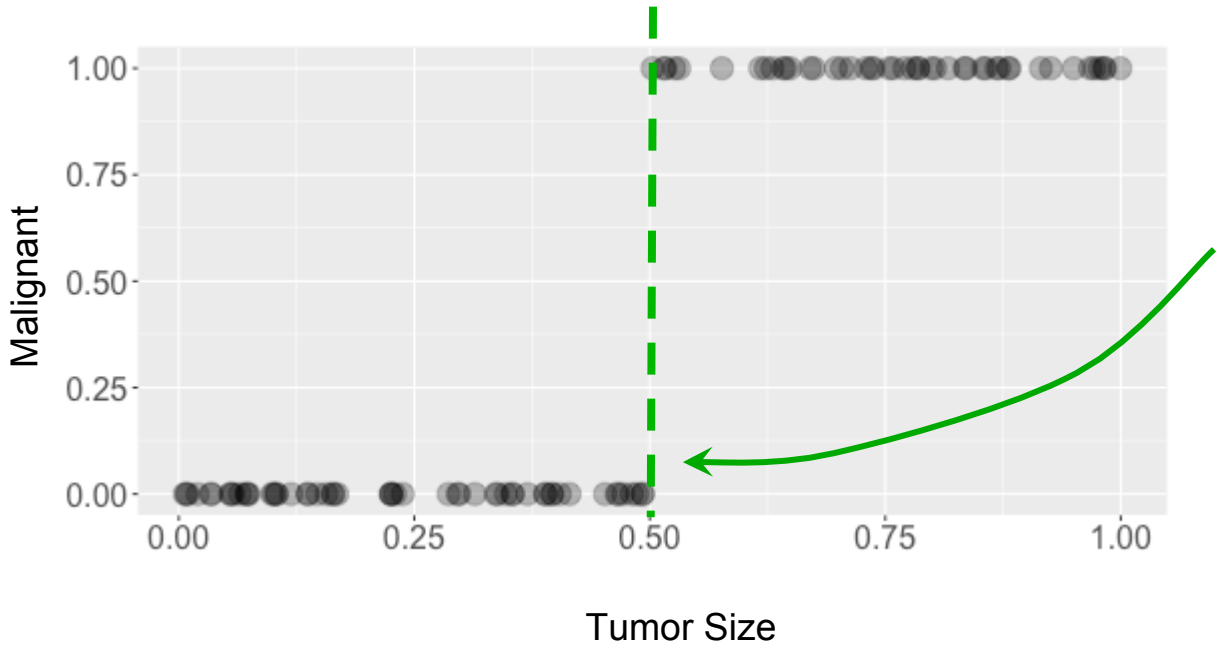
- **High Level**
  - Start with initial model estimate
  - Test model accuracy on data
  - Adjust model and test again until convergence



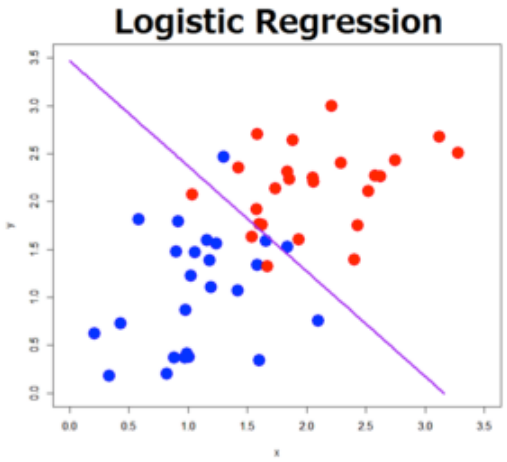
Details:

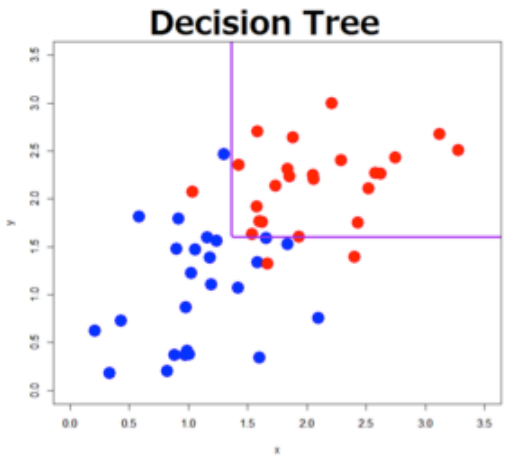
- Need a formal algorithm for adjusting decision boundary.
- Need to specify accuracy and convergence criteria.
- Need to ensure that model can “generalize.”

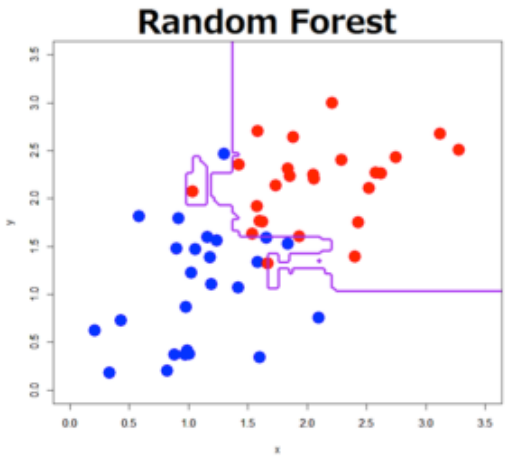
Decision Boundaries – 1 Variable



Decision Boundaries – 2 Variables

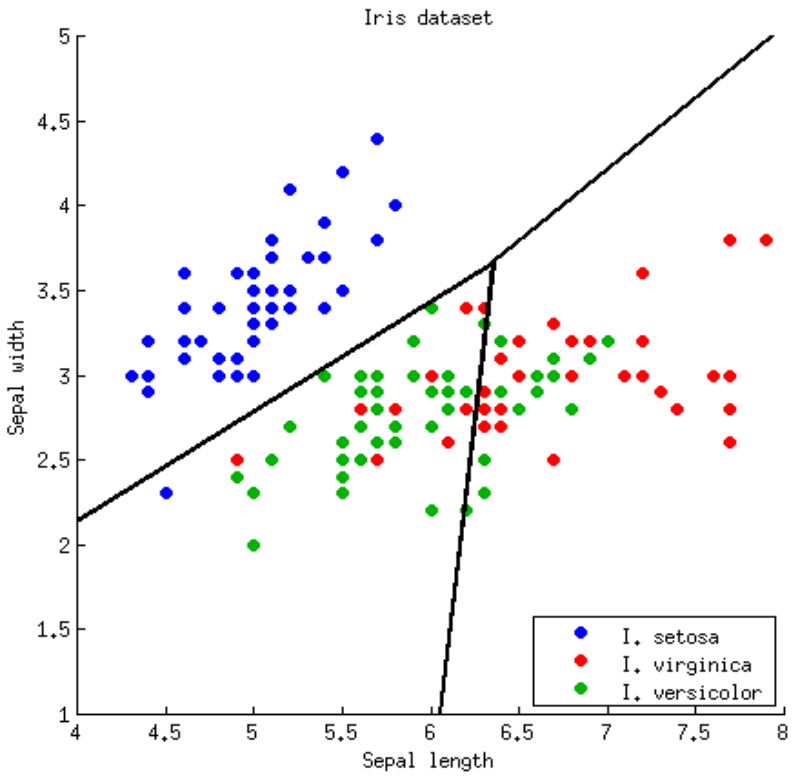






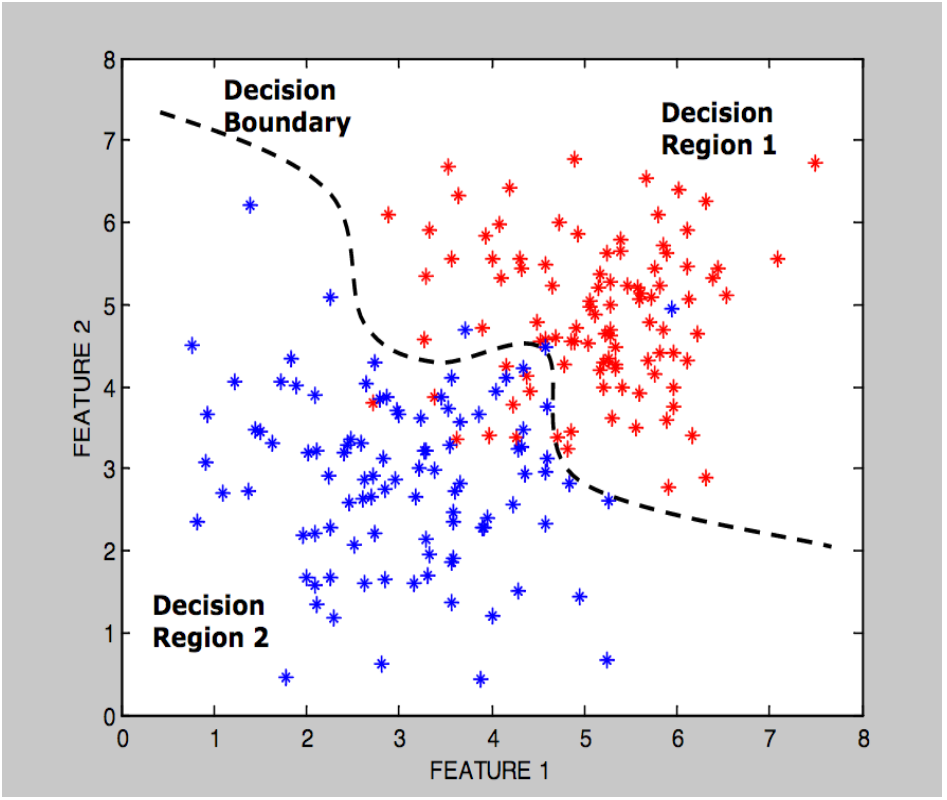
Decision Boundaries – Complex

- Multi-class problem



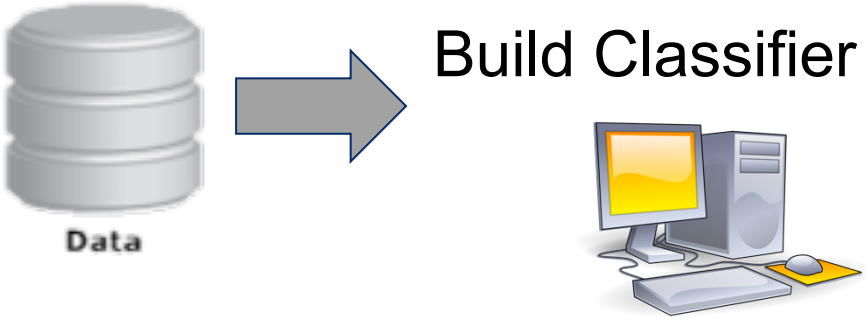
Source: http://stats.stackexchange.com/questions/92157/compute-and-graph-the-lda-decision-boundary

- Complex boundary



Source: https://www.ics.uci.edu/~rickl/courses/cs-171/cs171-lecture-slides/cs-171-19-LearnClassifiers.pdf

Generalization



How do we know the model we built is a good classifier?

Solution: Check to see if the classifier is correctly classifying labels with data we already have.

Generalization

How do we know the model we built is a good classifier?

Solution: Check to see if the classifier is correctly classifying labels with data we already have.

What is wrong with this approach?

Generalization

What is wrong with this approach?

- The error you compute with this data is called *resubstitution error*.
- You have only verified that the classifier is good at rote memorization!

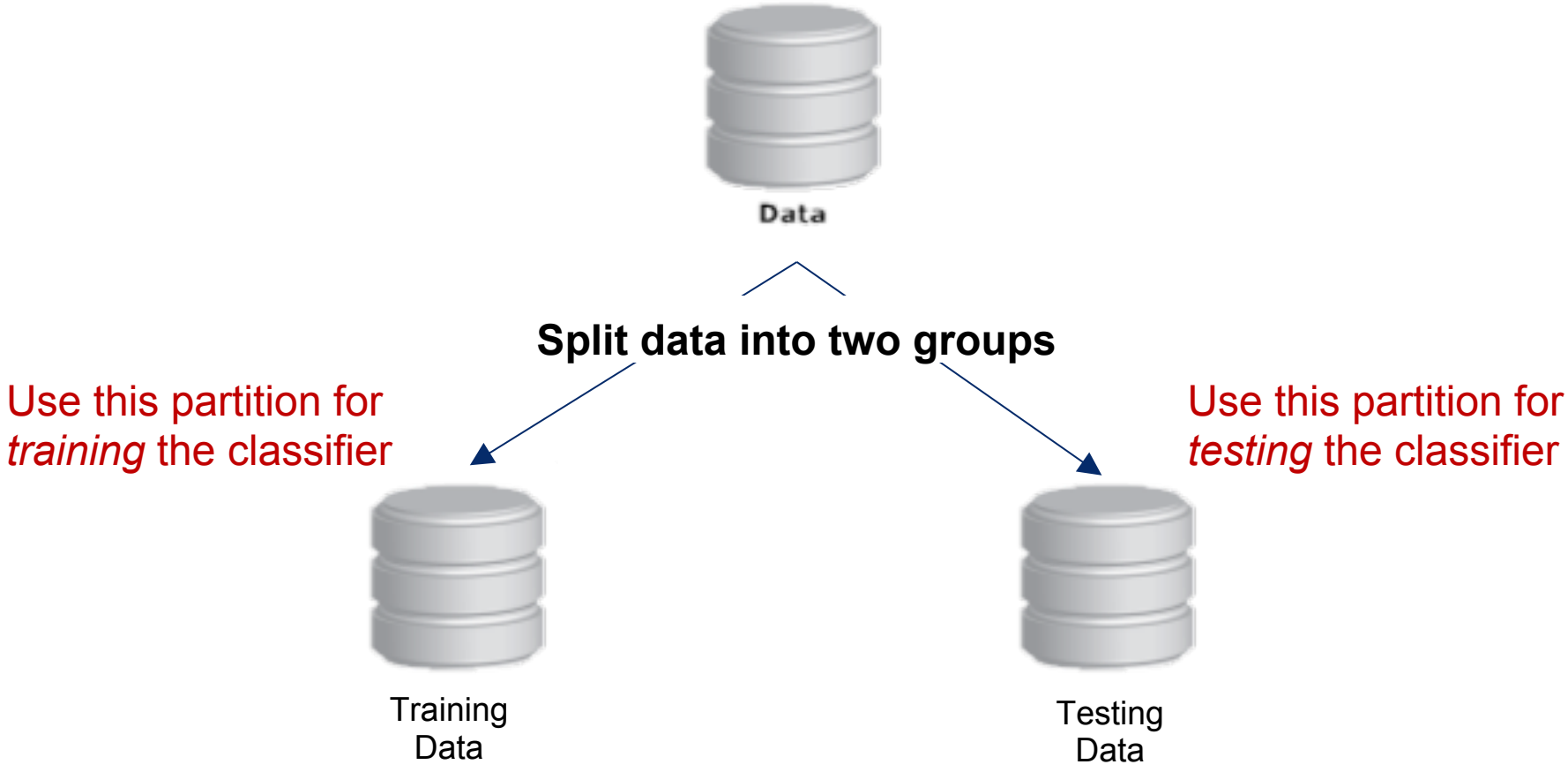
Generalization

What is wrong with this approach?

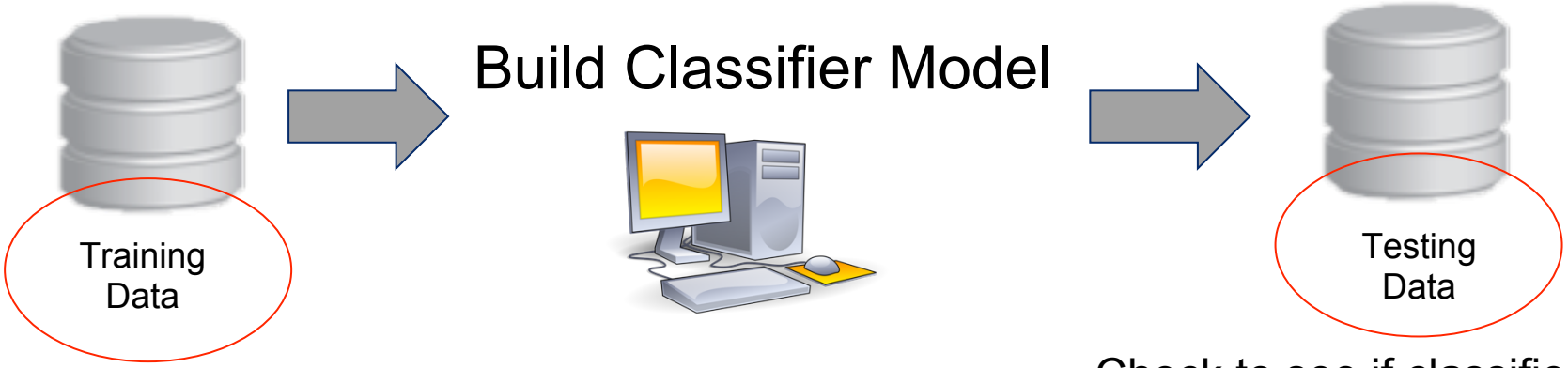
Check to see if classifier built is correctly classifying labels in our data

Another way of putting it: The error that we compute from the data used in training does not tell us anything by generalization capability of model; how does it do on *unseen* data.

Generalization



Generalization



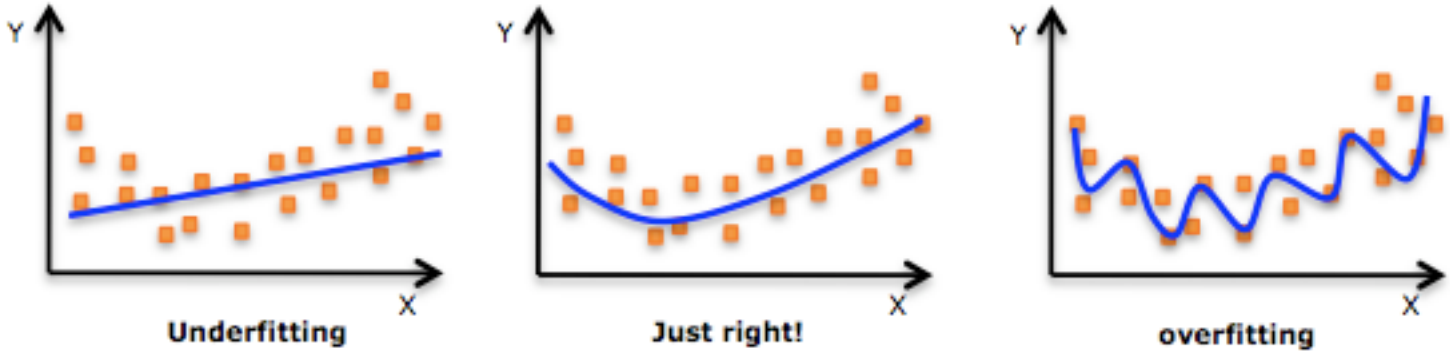
Check to see if classifier built is correctly classifying labels in *new* data

Generalization – Main Points

- **Build model on *training* dataset**
- **Test model on *test* dataset**

Overfitting

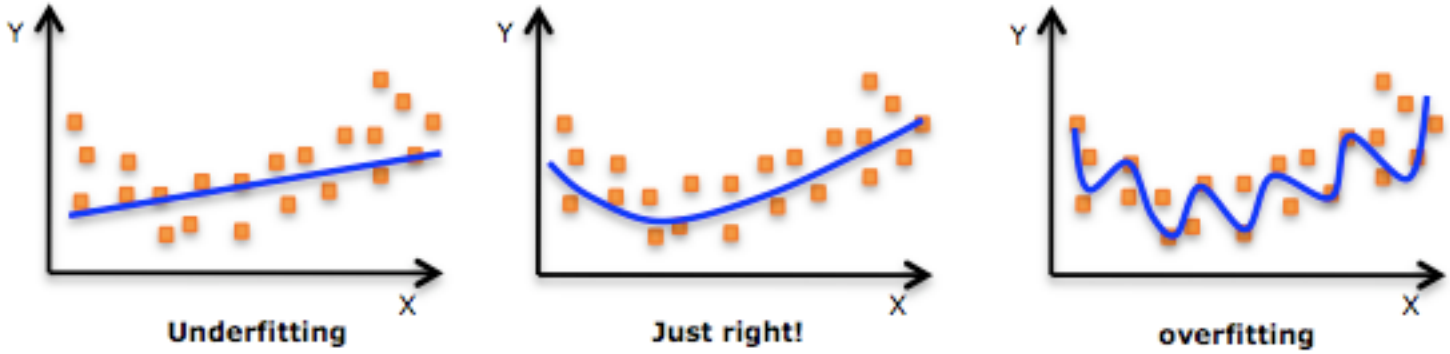
- *Overfitting* **occurs when model is fitting to noise in training data.**
- *Underfitting* **occurs when model has not learned structure of data.**



Source: http://stats.stackexchange.com/questions/192007/what-measures-you-look-at-the-determine-over-fitting-in-linear-regression

Overfitting & Generalization

- *Overfitting* **results in high training error & high generalization error.**
- *Underfitting* **results in low training error & high generalization error.**



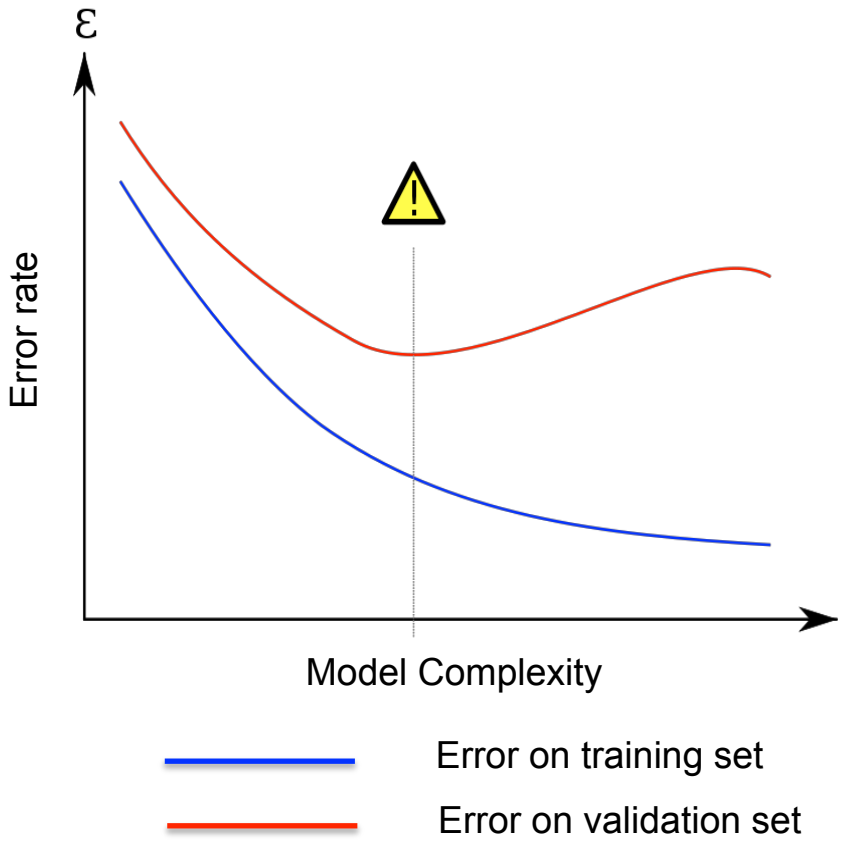
Source: http://stats.stackexchange.com/questions/192007/what-measures-you-look-at-the-determine-over-fitting-in-linear-regression

Model Validation

- **To address overfitting and estimate generalization performance.**
- **Idea:**
  - Divide training set into multiple datasets
    - Training set: Used to fit model parameters to data
    - Validation set: Used to validate model performance
  - Monitor performance on training and validation sets to determine when to stop training.

Validation with Holdout Set

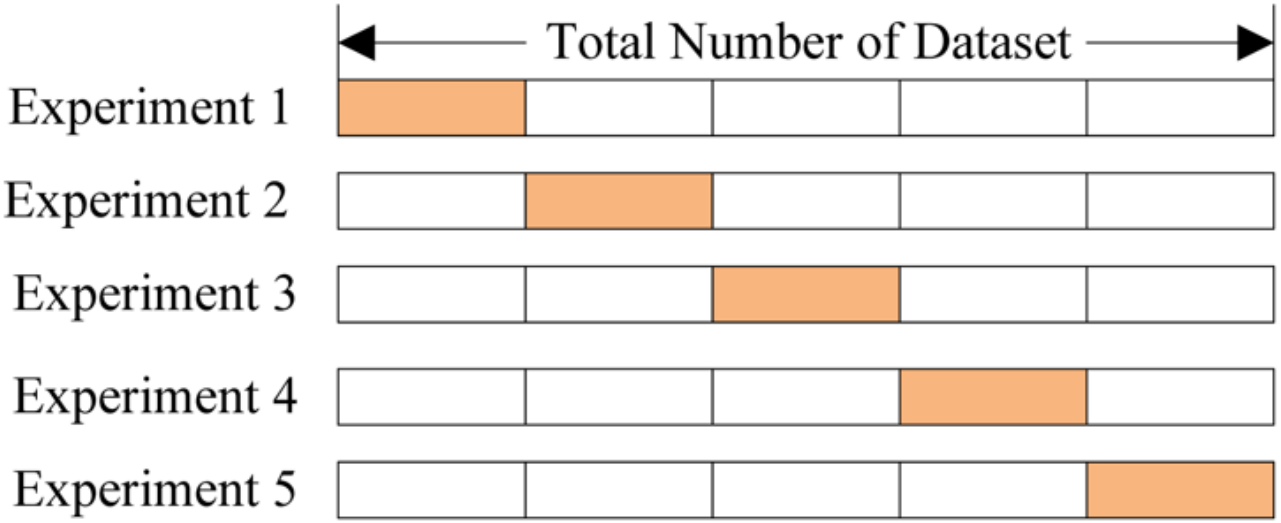
- Overfitting is occurring if training error decreases while validation error increases.
- Model with best generalization performance is one with lowest validation error.



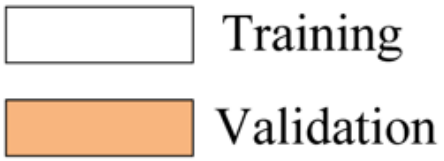
Source: https://en.wikipedia.org/wiki/Overfitting

Cross-Validation

- **K-fold cross-validation**
  - Partition data into k disjoint datasets
  - For each iteration, use 1 partition for testing and the rest for training.
  - Repeat process k times. Each partition is used for testing exactly once.
  - Overall error estimate (generalization performance) is average of error rates for k iterations.



Source: http://stackoverflow.com/questions/31947183/how-to-implement-walk-forward-testing-in-sklearn

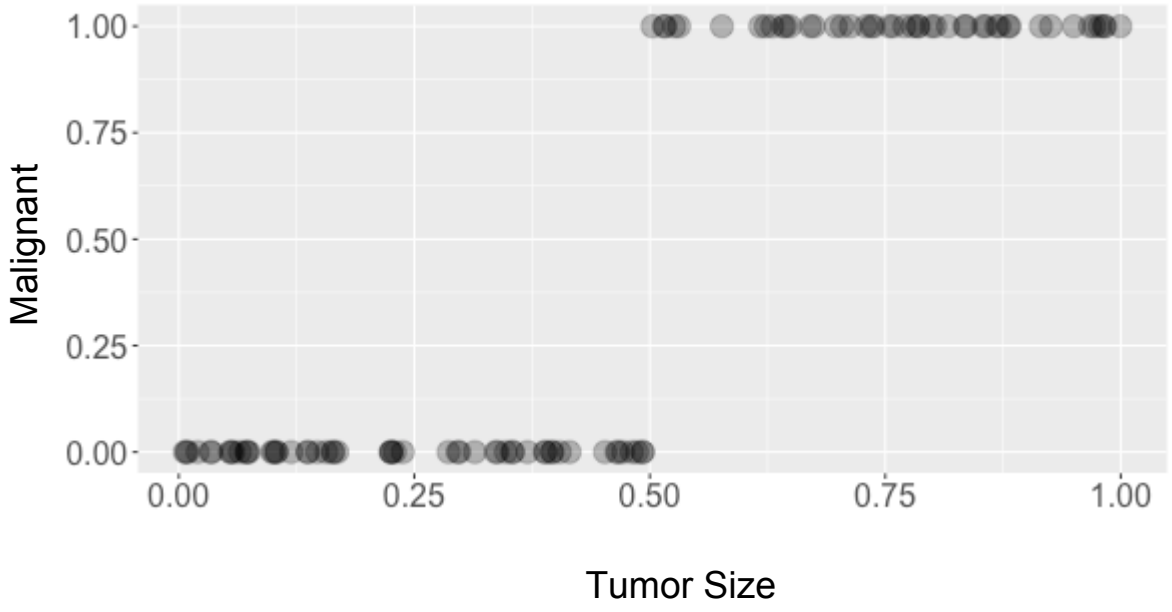


Classification Algorithms

- **k-Nearest Neighbor**
- **Decision Tree**
- **Random Forest**

k Nearest Neighbor Classifier (kNN)

- **Given a new example input, look for the "closest" or "nearest" example in the data**

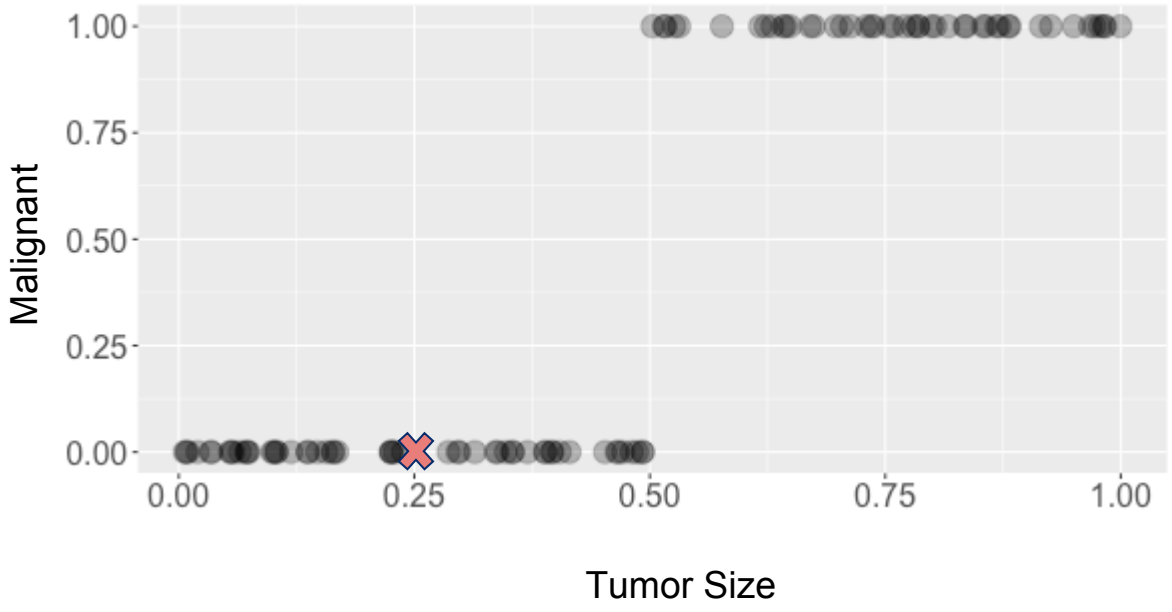


We have measured a new tumor to be of size 0.25.

Given our data in the left, what is the closest tumor size from what we have already recorded?

k Nearest Neighbor Classifier

- **Given a new example input, look for the "closest" or "nearest" example in the data**



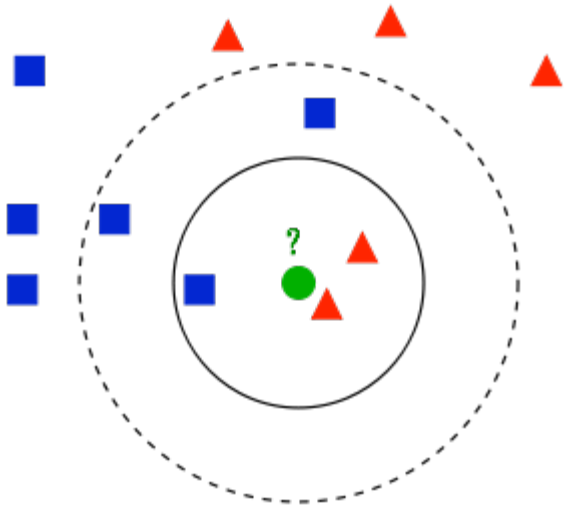
We have measured a new tumor to be of size 0.25.

Nearest neighbor: looks like a lot of tumors that have a size close to 0.25 are malignant

k Nearest Neighbor Classifier

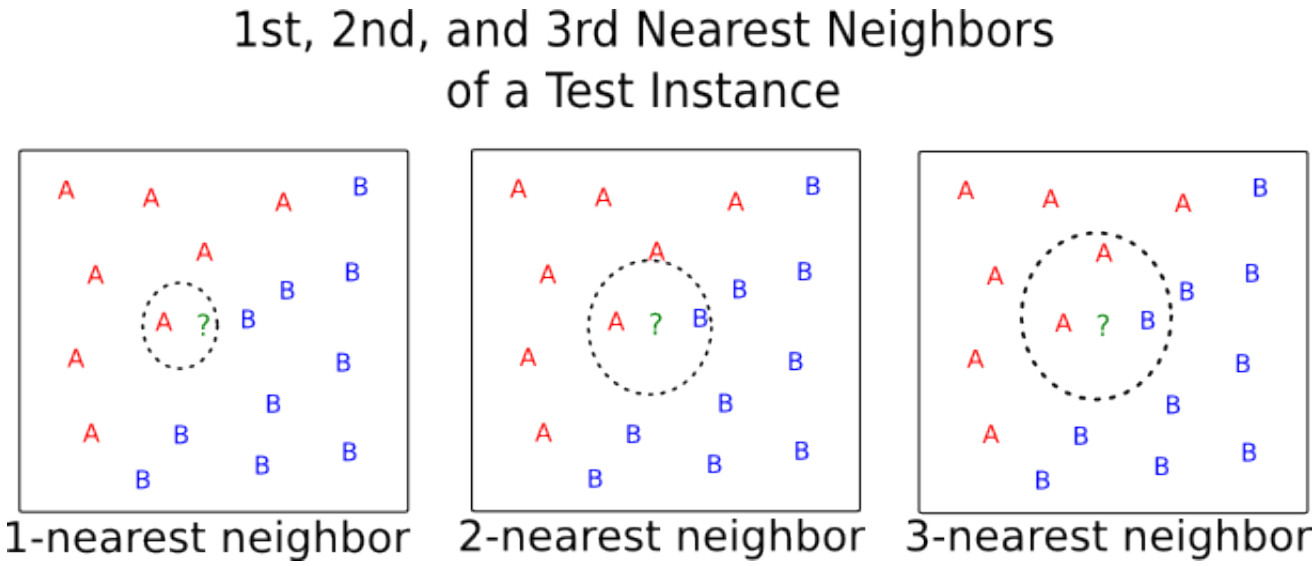
- When we select the closest example in our data, this is referred to as the 1-Nearest Neighbor classifier

- Often looking at multiple nearest neighbors (and voting) yields more robust results



k Nearest Neighbor Classifier

- kNN with different values of k



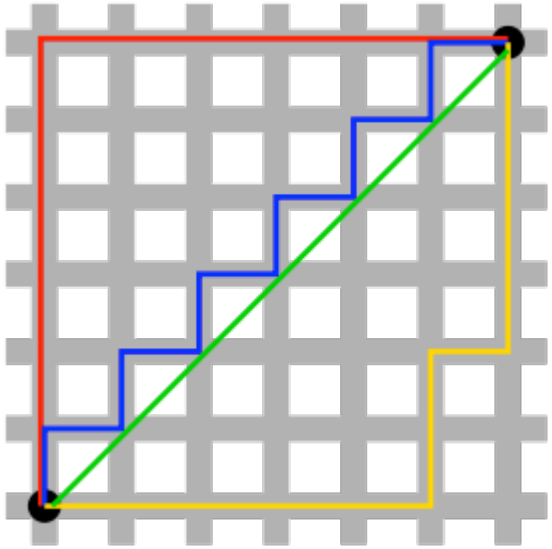
Source: http://trevorwhitney.com/data_mining/classification

k Nearest Neighbor Classifier

- **Note that in more than 1 dimension we need a more formal definition of distance**

- **Common distances used:**
  - Euclidean (green line)
  - Manhattan (red, blue, yellow lines)
  - Others…

$$\left(\sum_{i=1}^{N} |x_i - y_i|^p \right)^{\frac{1}{p}}$$



Source: https://en.wikipedia.org/wiki/Taxicab_geometry

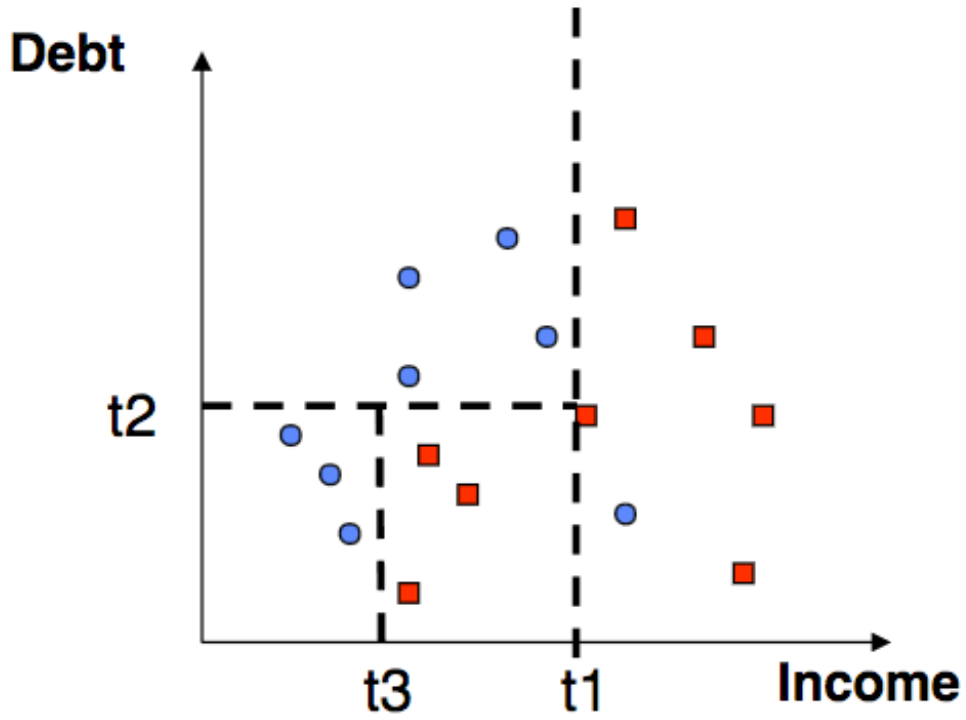
Classification Algorithms

- k-Nearest Neighbor
- **Decision Tree**
- Random Forest

Decision Tree Overview

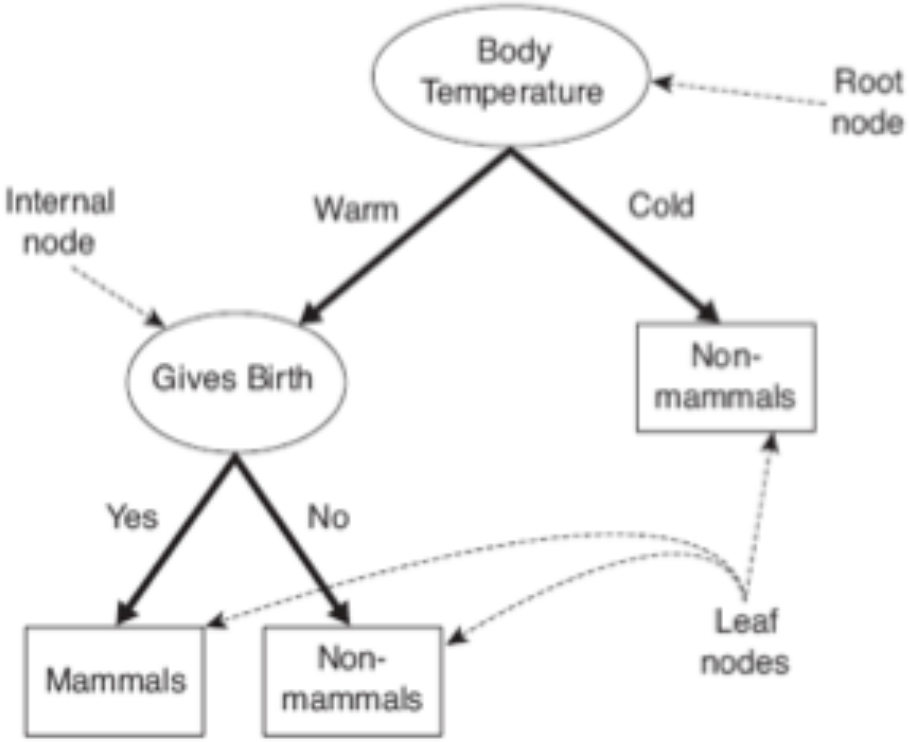
Can we split the data into cohesive groups?

- Split data into “pure” regions
- Classification decision based on these regions



Decision Tree Structure

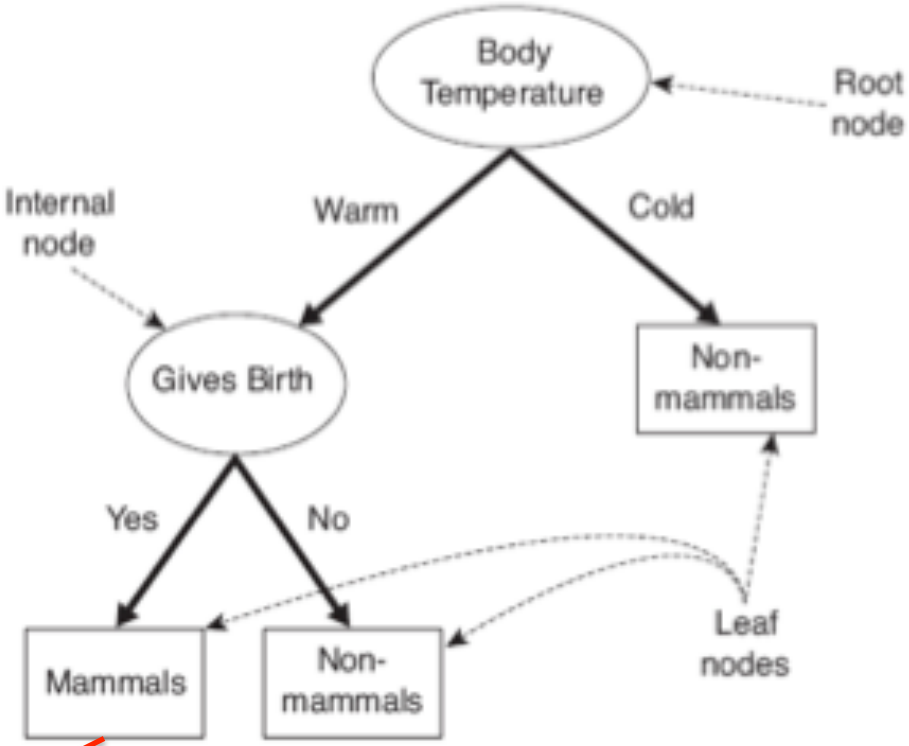
- Hierarchical structure with nodes and directed edges
- Root and internal nodes have test conditions.
- Leaf nodes have class labels.
- Paths from root to leaf represents classification rules.



Source: http://www-users.cs.umn.edu/~kumar/dmbook/dmslides/chap4_basic_classification.pdf

Classification Using Decision Tree

- Traverse tree from root to leaf.
- At each node, answer to test condition determines child node to move to.
- Category at leaf node determines label for sample.



Body Temp	Gives Birth	Target Label
Warm	Yes	Mammal

Source: http://www-users.cs.umn.edu/~kumar/dmbook/dmslides/chap4_basic_classification.pdf

Constructing Decision Tree

- **Overview of tree induction:**
  - Start with all records at a node.
  - Partition records based on input variables.
    - Goal is to create subsets of records that are *purest* (i.e., most homogeneous) with respect to class distributions.
    - All possible splits on all input variables are considered and best split is determined.
  - Repeatedly partition data into successively purer subsets until stopping criteria are satisfied.

How to Determine Best Split?

- **Nodes with purest (most homogeneous) subsets of data are preferred.**

C0: 5 C1: 5	C0: 9 C1: 1
non-homogeneous high degree of impurity	more homogeneous low degree of impurity

- **Need measure of node impurity**

Impurity Measures

- **Gini Index**

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

where *p(j|t)* is the relative frequency of class j at node t

  - The lower the Gini index, the lower the purity of the split.
  - Minimum of 0 is when all samples in node belong to single category.
  - Want to find split that minimizes the Gini index.

- **CART: Classification and Regression Tree**
  - Popular decision tree algorithm that uses Gini index

Impurity Measures

- **Measures of impurity for tree induction:**
  - Gini index
  - Entropy (information gain)
  - Chi-square
  - Minimum Description Length
  - Others…

Stopping Criteria for Tree Induction

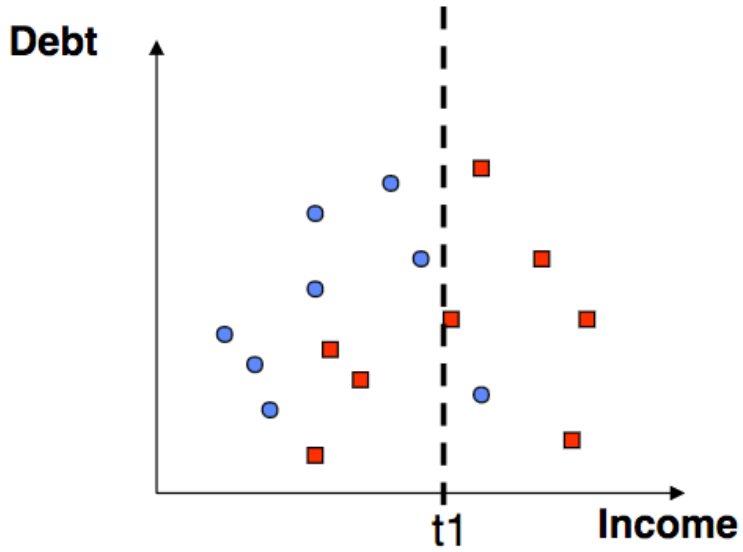
- Stop expanding a node when all samples belong to same class (i.e., all samples have same class label).
- Stop expanding a node when all samples have similar attribute values.
- Stop expanding a node when improvement in impurity falls below certain threshold.
- Stop expanding a node when number of samples in node is below minimum threshold.
- Stop expanding a node when tree depth is reached.

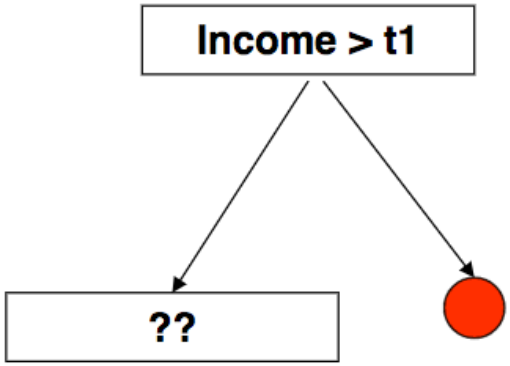
Tree Pruning

- **Reduces model complexity to avoid overfitting**
- **Process:**
  - Grow tree to maximum size or until stopping criterion is reached.
  - Prune tree using bottom-up approach to reduce tree size
    - Idea is to remove nodes that do not add to generalization performance.
  - Use error on validation set or cross-validation to determine when to stop pruning.

Decision Tree Example

- Split 1

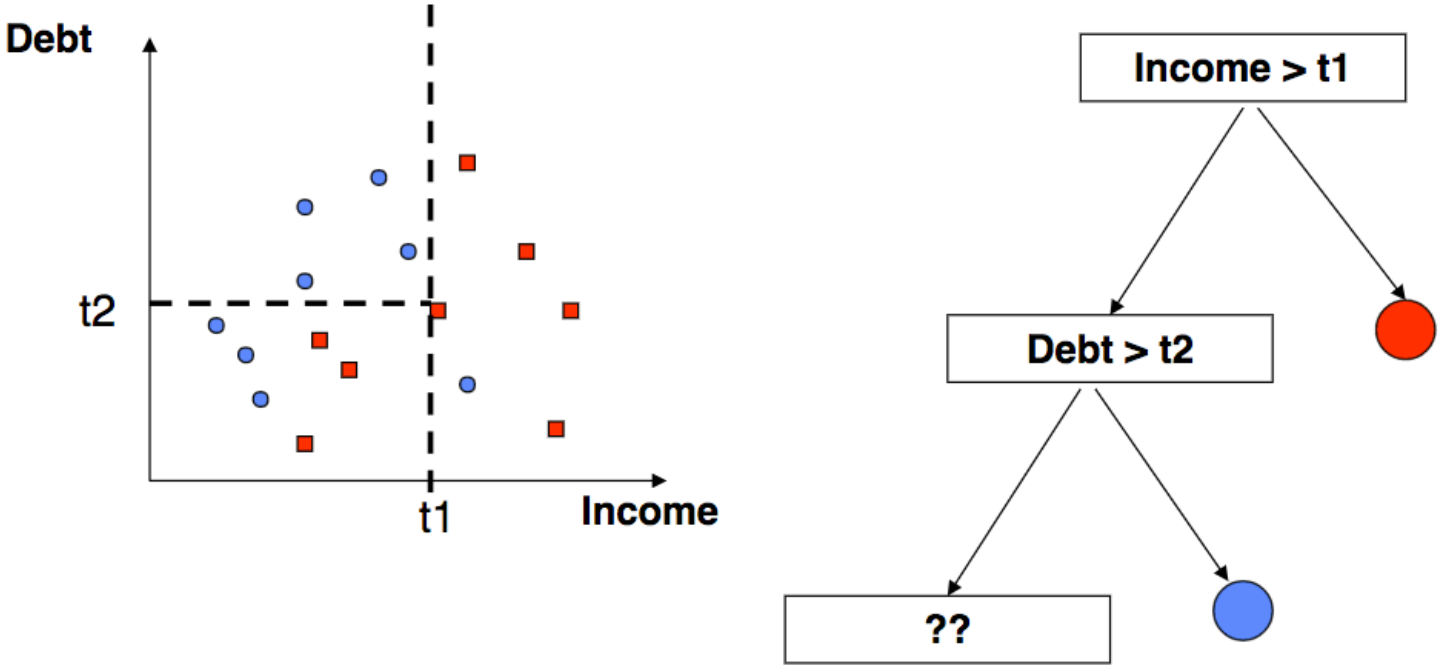




Source: https://www.ics.uci.edu/~rickl/courses/cs-171/cs171-lecture-slides/cs-171-19-LearnClassifiers.pdf

Decision Tree Example

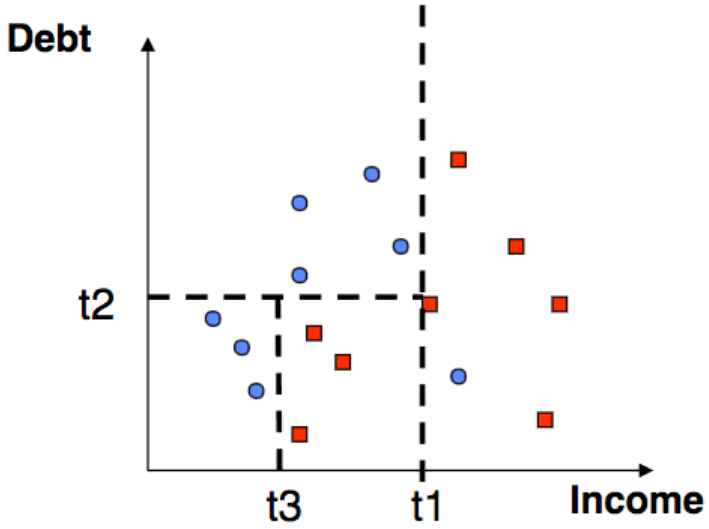
- Split 2

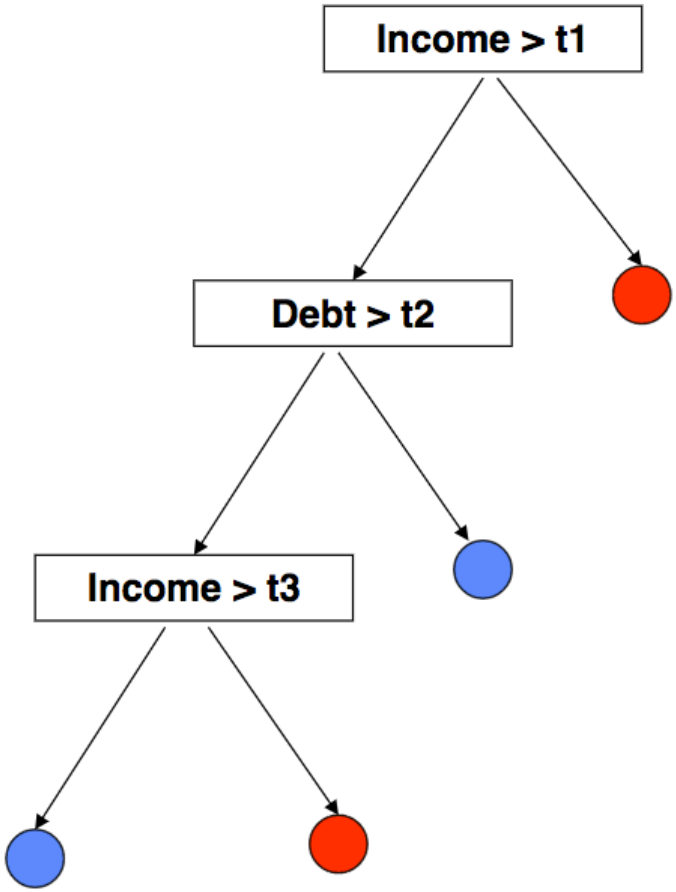


Source: https://www.ics.uci.edu/~rickl/courses/cs-171/cs171-lecture-slides/cs-171-19-LearnClassifiers.pdf

Decision Tree Example

- **Split 3**





More splits increases "purity" at the leaf nodes

Source: https://www.ics.uci.edu/~rickl/courses/cs-171/cs171-lecture-slides/cs-171-19-LearnClassifiers.pdf

Decision Tree Classification

- **Pros**
  - Resulting tree is often simple to understand and interpret.
  - Features can be any type (numerical or categorical), and can include missing values.
  - Tree induction algorithms are relatively computationally inexpensive.
- **Cons**
  - Induction algorithms use greedy approach where locally optimal decisions are made. No guarantee of globally optimal model.
  - Decision boundaries are rectilinear, limiting expressiveness for modeling complex relationships.

Classification Algorithms

- K-nearest neighbors
- Decision Tree
- **Random Forest**

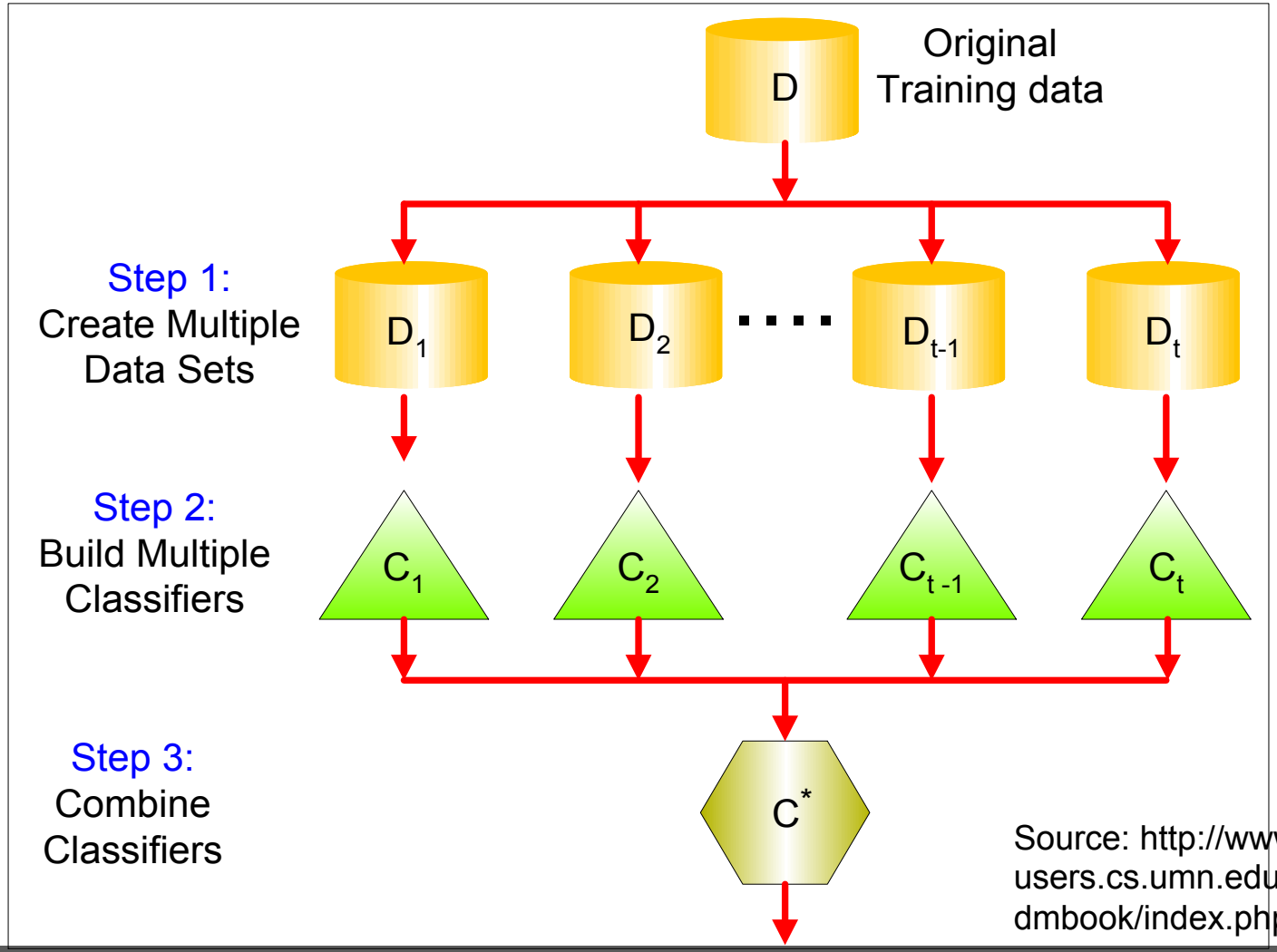
Random Forest

- **“forest”**
  - Ensemble method
    - Model is composed of set of decision trees => forest!
- **“random”**
  - For each tree, only subset of variables used to determine best split.
  - Subset of attributes is determined randomly.
- **Idea:**
  - To improve generalization of model

Ensemble Methods

- **Idea:**
  - Improve model performance by combining predictions of several models.
- **"ensemble":**
  - a group producing a single effect (from Merriam-Webster)
- **Approach:**
  - Construct a set of classifiers from training data.
  - Prediction is made by combining outputs of the multiple classifiers
    - Classification: Combine votes of classifiers

Ensemble Methods – Illustration



Ensemble Methods

- **Rationale for ensemble methods**
  - Consider ensemble that performs majority voting.
  - Base classifiers may make mistakes, but ensemble will misclassify a pattern only if over half of base classifiers are incorrect.
  - Intuitively, combining decisions from multiple “experts” may be more reliable than relying on a single “expert”.

Bagging

- **Bagging stands for "bootstrap aggregation".**
- **Approach:**
  - Sample training data set with replacement to construct bootstrap samples.
  - Build separate classifier on each bootstrap sample.
  - Each classifier predicts class label for unknown record.
  - Bagged classifier takes majority vote.
- **Generalization can be improved since variance of individual base classifiers is reduced.**

Random Forests – Revisited

- **Ensemble of decision tree classifiers.**
  - Base decision tree built by using only subset of attributes to determine split at each node.
    - Subset of attributes is determined randomly.
  - Bagging is used to generate bootstrap samples for base decision trees.
  - Final classification is based on the majority vote.
- **Randomization reduces correlation among base trees.**
  - Generalization of ensemble can be improved.

Randomness in Random Forest

- **Randomness can also be incorporated in other ways:**
  - Forest-RI
    - Random attribute selection: At each node, randomly select subset of input attributes to consider in splitting node.
  - Forest-RC
    - Random combinations of attributes: At each node, randomly select subset of input attributes to be linearly combined. These new attributes are considered in splitting node.
  - Randomly select best split:
    - At each node, randomly select one of F best splits.

Out-of-Bag Error

- **OOB Data**
  - Each tree is trained using bootstrap sample, which leaves out some data points (“OOB” data points).
  - These data points not used to create tree can then be used for testing.
- **OOB Error**
  - Error averaged over all OOB data points and over all trees is OOB error.
  - OOB error can be used to estimate generalization error for random forest.

Variable Importance

- **Mean Decrease in Accuracy**
  - For each feature, permute values, and measure decrease in model accuracy.
  - Then average this over all trees to measure impact of feature on model accuracy.
  - Higher decrease -> more important feature
- **Mean Decrease in Impurity**
  - During tree creation, compute decrease in impurity measure for each feature for each node split.
  - Then average this over all splits and over all trees for feature.
  - Higher decrease -> more important feature

Random Forest Classification

- In practice, often results in improved performance due to lower variance.
- OOB error can be used to estimate generalization error
- Provides methods for determining variable importance.
- Training takes longer.
- More parameters to set.
- Ensembles more difficult to understand than single classifiers.

Classification – Key Points

- **Classification task**
  - Make decisions on data
- **Examples**
  - Determining if transaction is legitimate, detecting cancer, recognizing handwritten digits, etc.
- **Algorithms**
  - KNN, Decision Trees, Random Forest

References

- A. Liaw et al. Package 'randomForest'. Retrieved from https://cran.r-project.org/web/packages/randomForest/randomForest.pdf
- P. Tan, M. Steinbach, & V. Kumar. *Introduction to Data Mining*. Pearson: 2005.
- T. Therneau, B. Atkinson, & B. Ripley. Package 'rpart'. Retrieved from https://cran.r-project.org/web/packages/rpart/index.html