

# 2018 SDSC Summer Institute Machine Learning Overview



# Classification

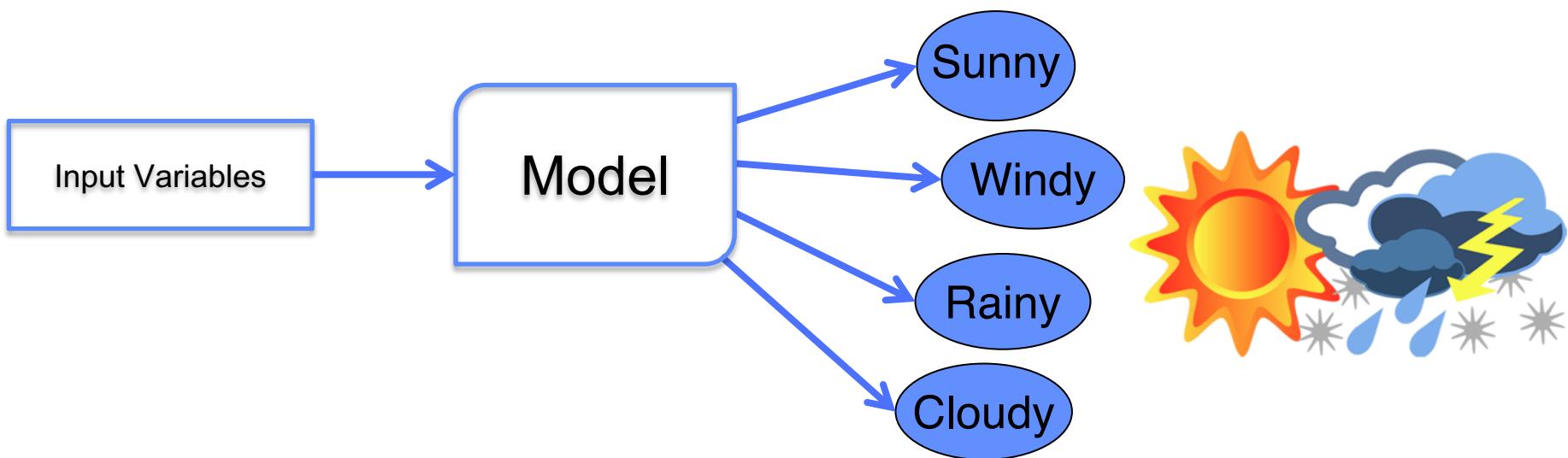
Mai H. Nguyen, Ph.D.

# Overview

- **What is classification?**
- **Terminology & concepts**
  - Training and test sets
  - Generalization & overfitting
- **Classification Algorithms**
  - Decision Trees
  - Random Forest
- **Hands-on**
  - Build classification models for weather data

# What is Classification?

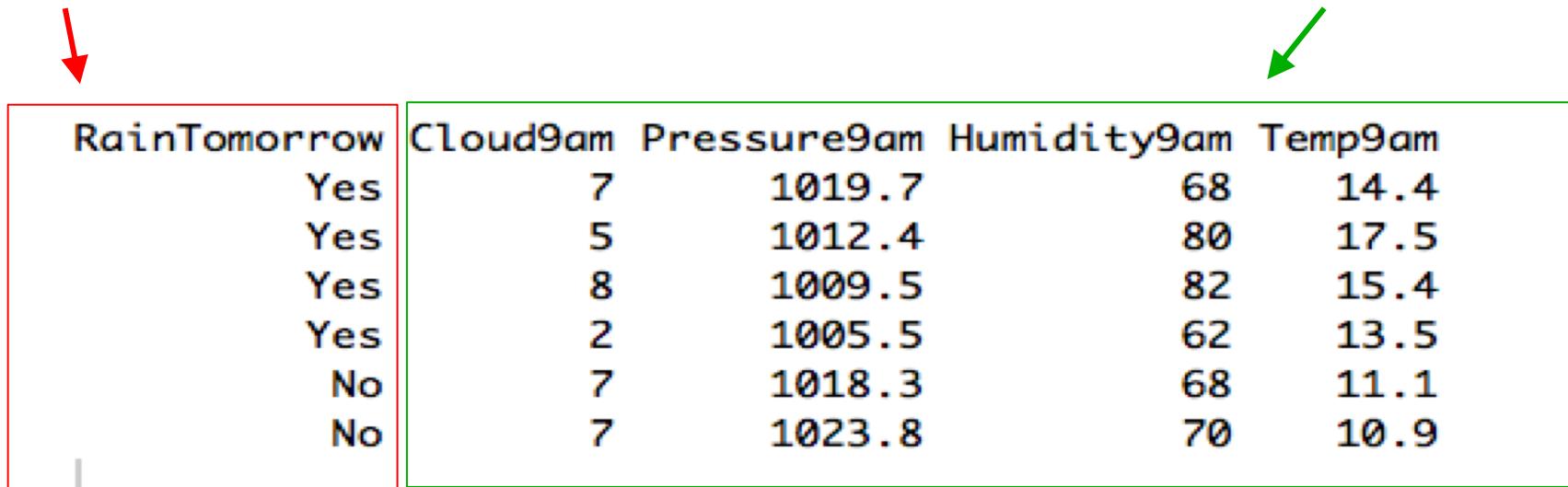
- **Given input variables, predict target variable.**
  - Target variable is *categorical*.
  - Model needs to learn relationship between input data and target.
  - Other names for ‘target’
    - label, class, class variable, output



# 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.

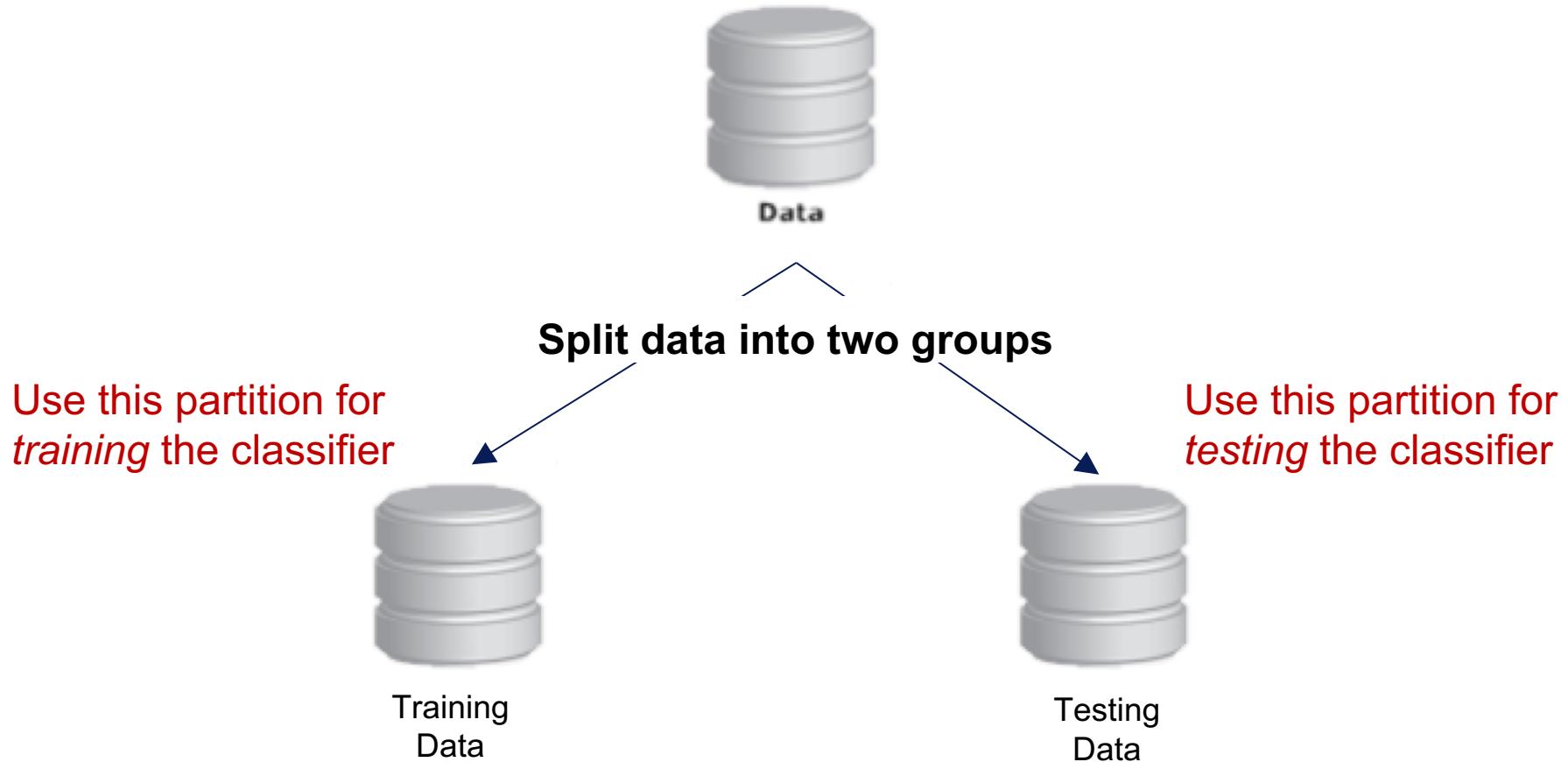
# Target Variable in Classification

- 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

# 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.

# Generalization



# Generalization

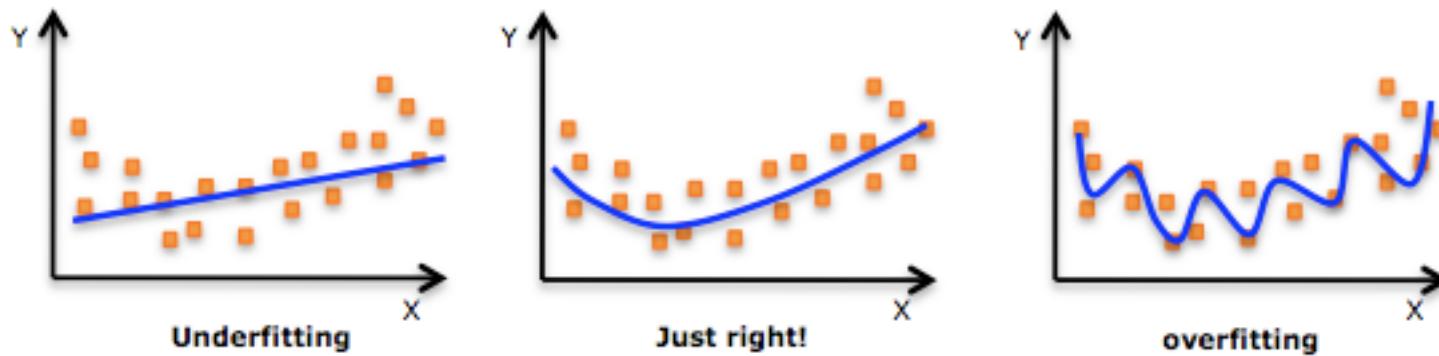


Check to see if classifier built is correctly classifying labels in *new* data

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

# Overfitting

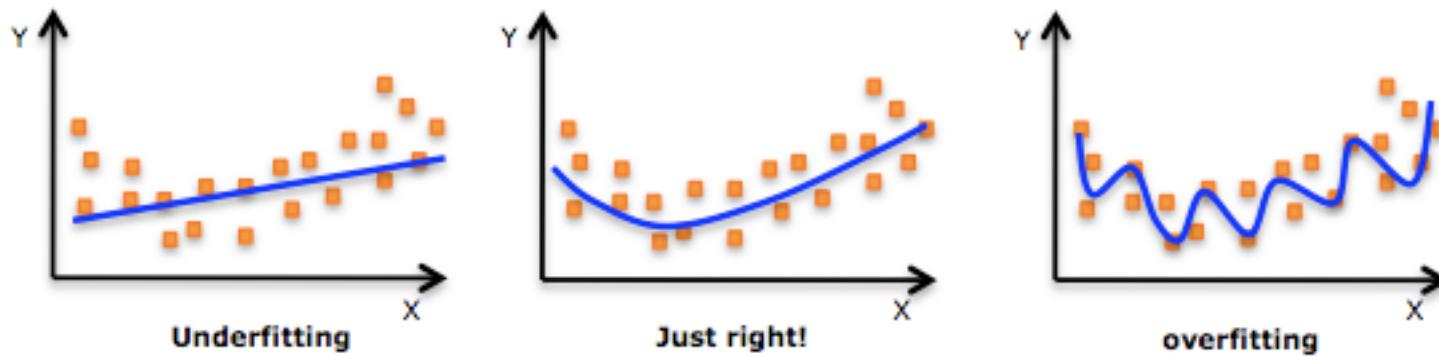
- *Overfitting occurs when model is fitting to noise in training data.*
- *Underfitting occurs when model has not learned structure of data.*



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

# Overfitting & Generalization

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



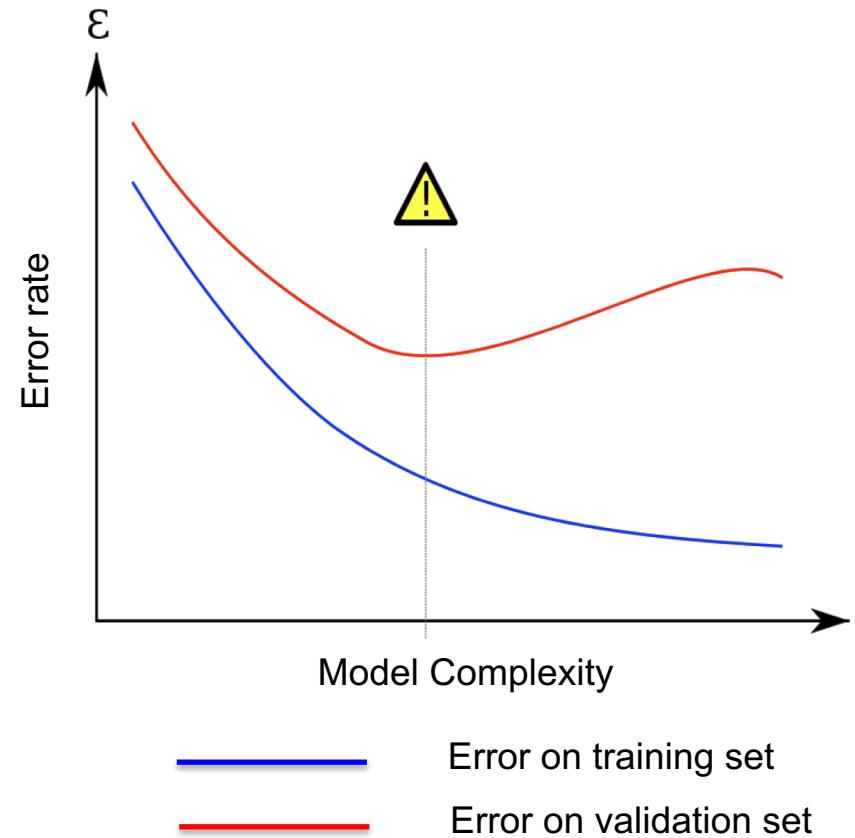
Source: <http://stats.stackexchange.com/questions/192007/what-measures-you-look-at-the-determine-over-fitting-in-linear-regression>

# Overfitting & Generalization

- To address overfitting and estimate generalization performance, use validation set.
- 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.

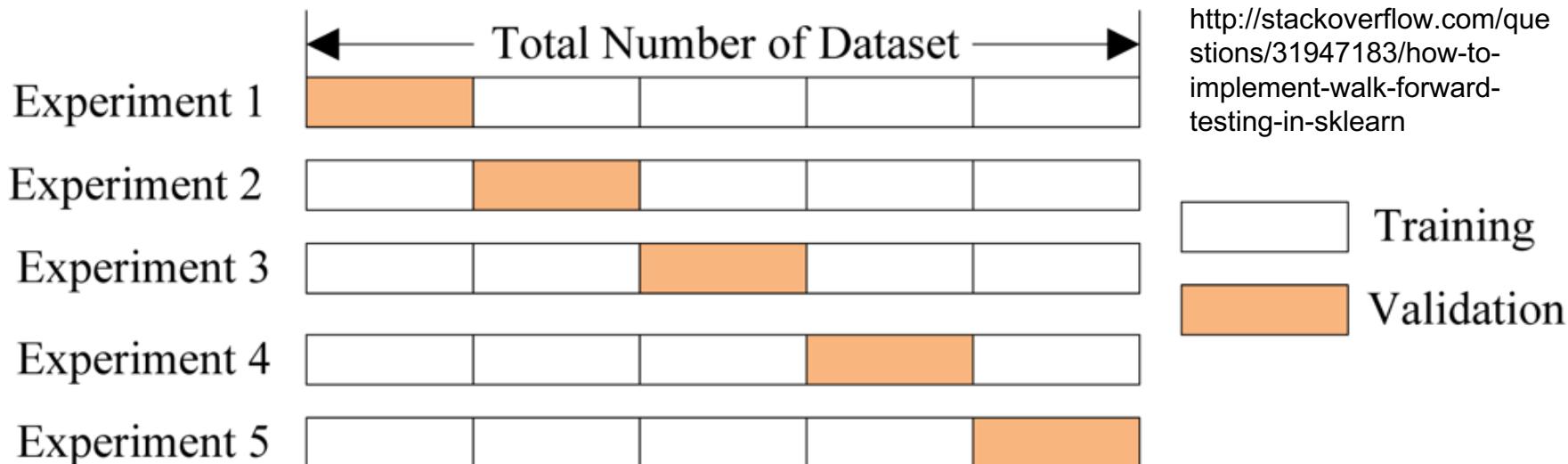


Source: <https://en.wikipedia.org/wiki/Overfitting>

# Cross-Validation

- **K-fold cross-validation**

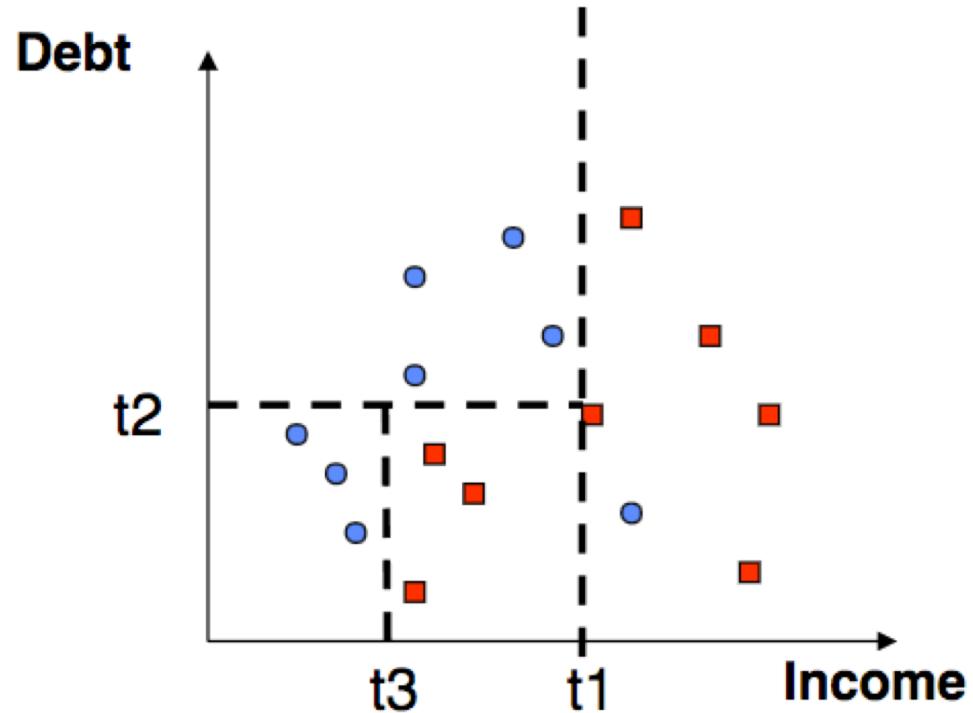
- Partition data into k disjoint datasets
- For each iteration, use 1 partition for validation and the rest for training.
- Repeat process k times. Each partition is used for validation exactly once.
- Overall error estimate (generalization performance) is average of error rates for k iterations.



# Decision Tree Overview

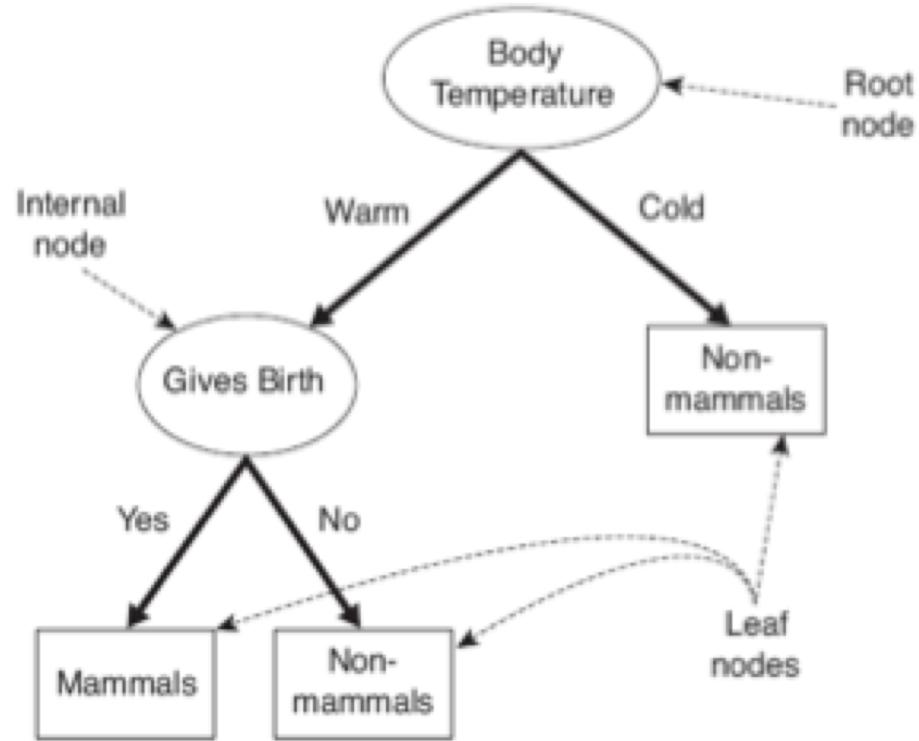
Can we split the data into cohesive groups?

- 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 represent classification rules.

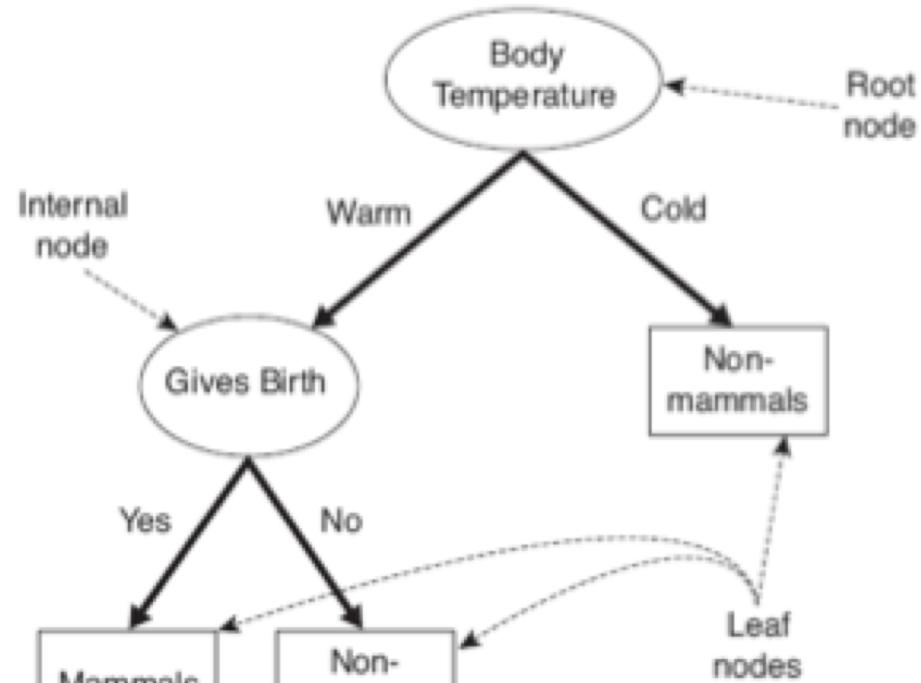


Source: [http://www-users.cs.umn.edu/~kumar/dmbook/dmslides/chap4\\_basic\\_classification.pdf](http://www-users.cs.umn.edu/~kumar/dmbook/dmslides/chap4_basic_classification.pdf)

# 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/cha\\_p4\\_basic\\_classification.pdf](http://www-users.cs.umn.edu/~kumar/dmbook/dmslides/cha_p4_basic_classification.pdf)

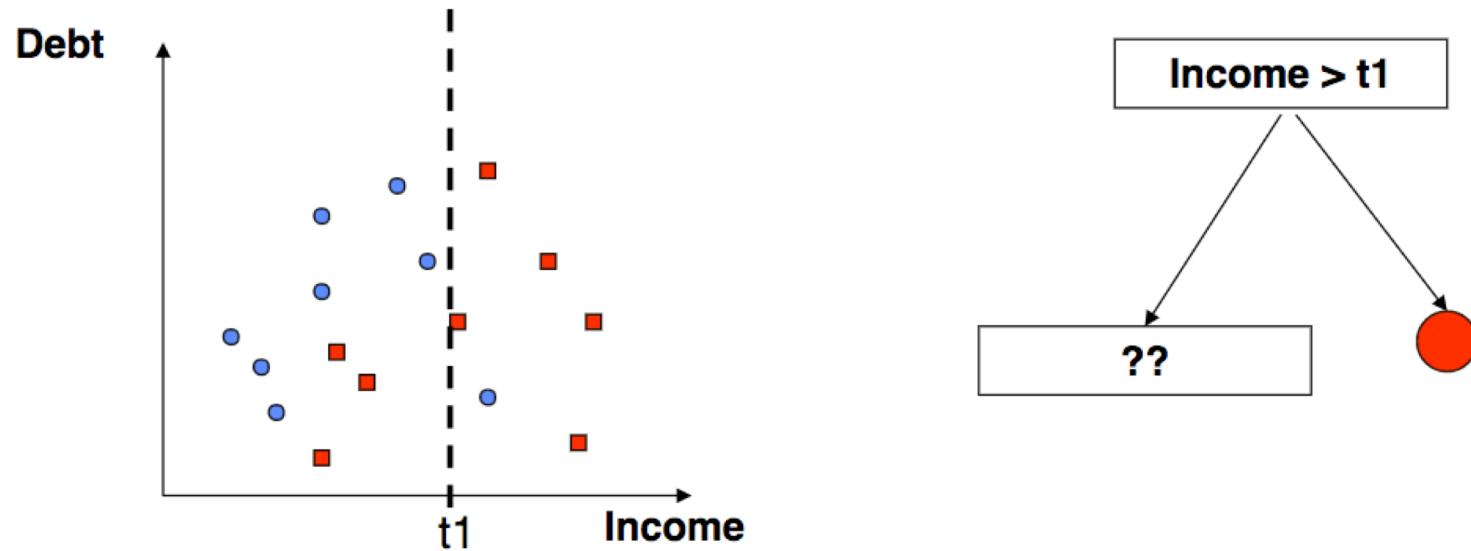
# Constructing Decision Tree

- **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.

# Decision Tree Example

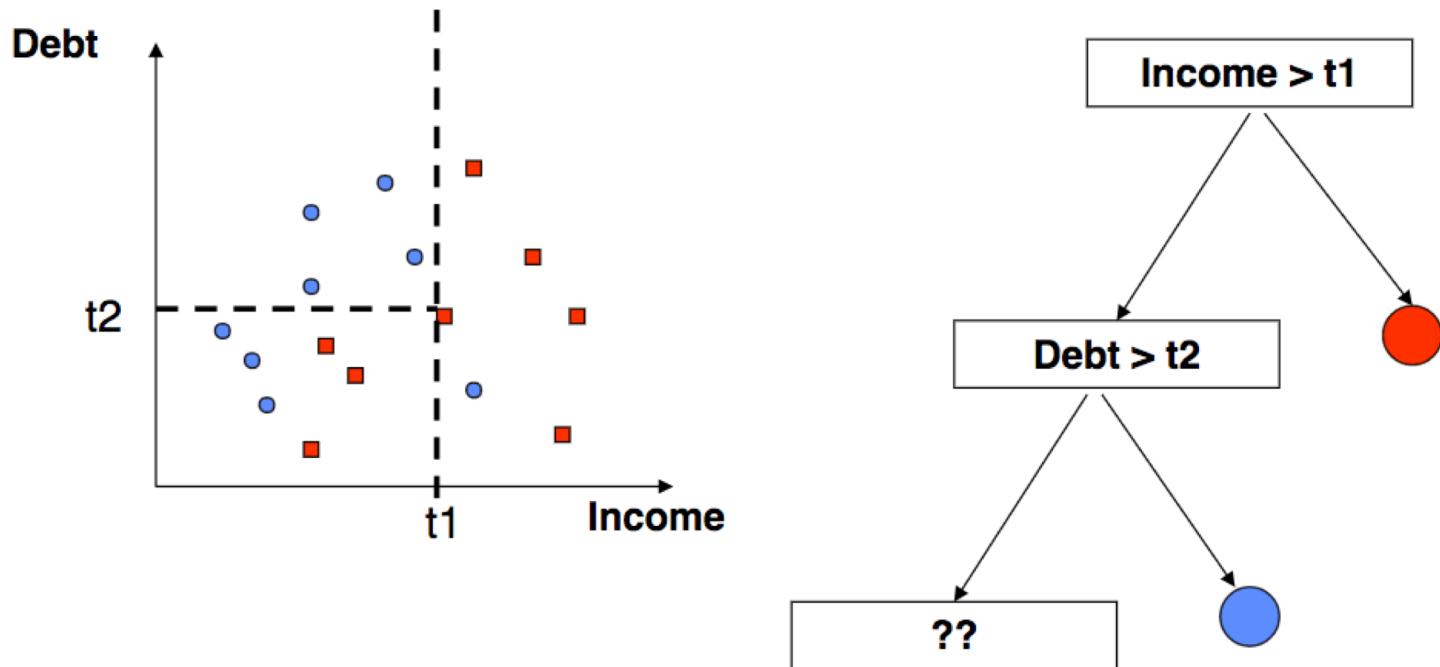
- Split 1



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

# Decision Tree Example

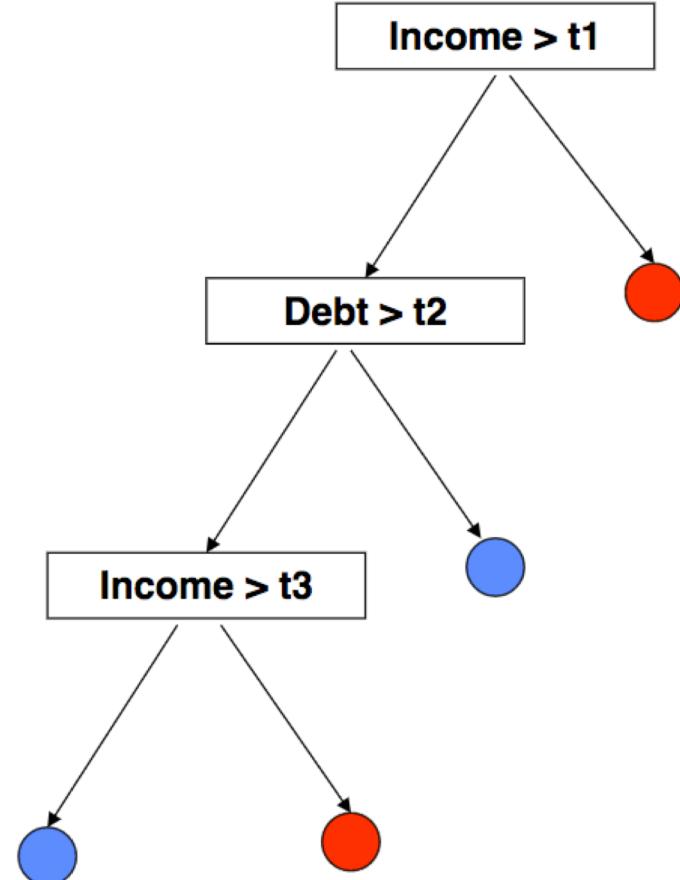
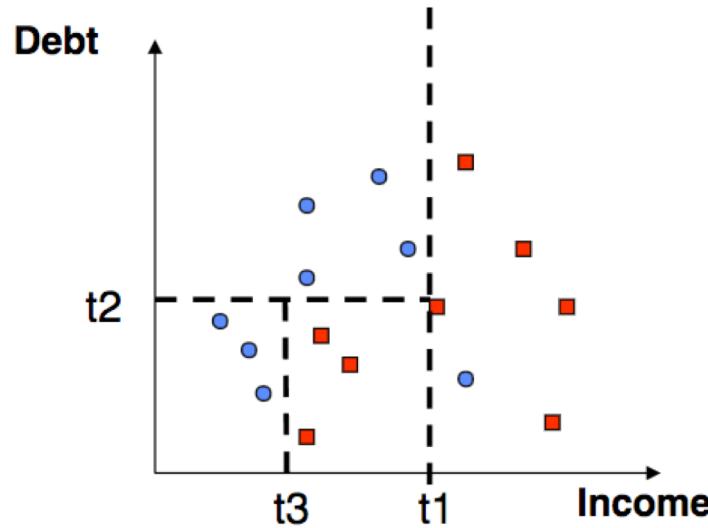
- Split 2



Source: <https://www.ics.uci.edu/~rckl/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/~rckl/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.

# 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.

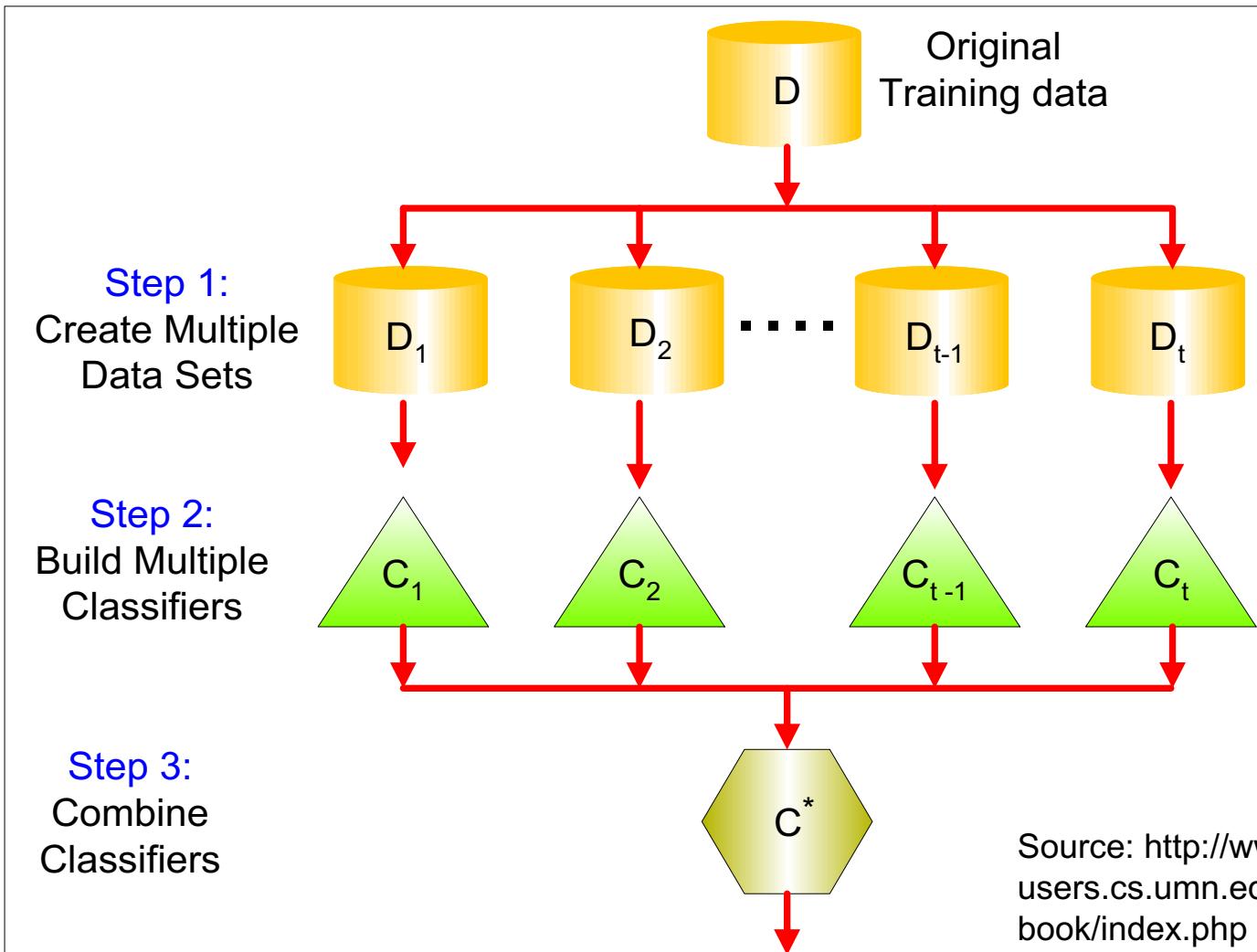
- **“ensemble”:**

- 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 be incorporated in several 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.

# **Random Forest Classification**

- In practice, often results in improved performance due to lower variance.
- Training takes longer.
- Ensembles more difficult to understand than single classifiers.

# Classification – Key Points

- **Classification task**
  - Predict categorical variable from input variables
- **Overfitting & Generalization**
  - Avoid overfitting to have model generalize to new data
  - Techniques: validation with holdout set, cross-validation
- **Algorithms**
  - Decision Trees, Random Forest
  - Others: k-nearest-neighbor, naïve Bayes, logistic regression, neural networks, ...

# 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>

# Questions?



# Classification Hands-On

- Predict whether it will rain tomorrow
  - Decision tree
  - Random forest
- Using same Australian weather dataset from EDA hands-on