Boosting

Lecture 8b

Last time:

1. Ensemble Learning
2. Hard and soft voting classifiers
3. Bagging and Pasting
4. Random Forest
5. Feature Importance

Today: Learning Objectives

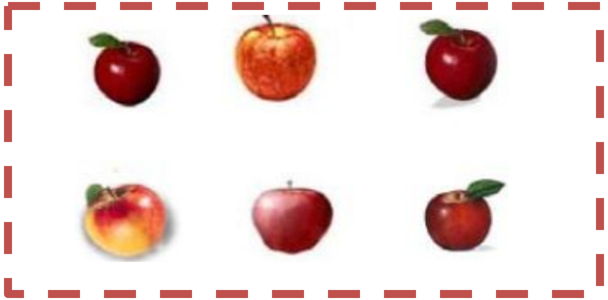
1. Teach machine how to recognize apples
2. Understand how boosting algorithm works
3. Demo and application of boosting
4. Go over Stacking method

1. How to teach machine to recognize apples?

Recognizing apples?

- Collect a set of real apples and plastic apples
- Observe some rules to tell them apart based on their characteristics

Real Apples

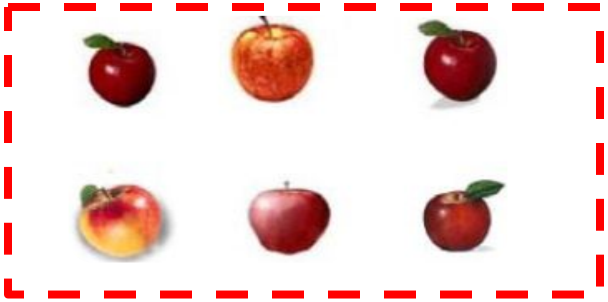


Plastic Apples

A question for you

Can you think of some simple rules to tell the difference between "real apples" and "plastic apples"?

Real Apples

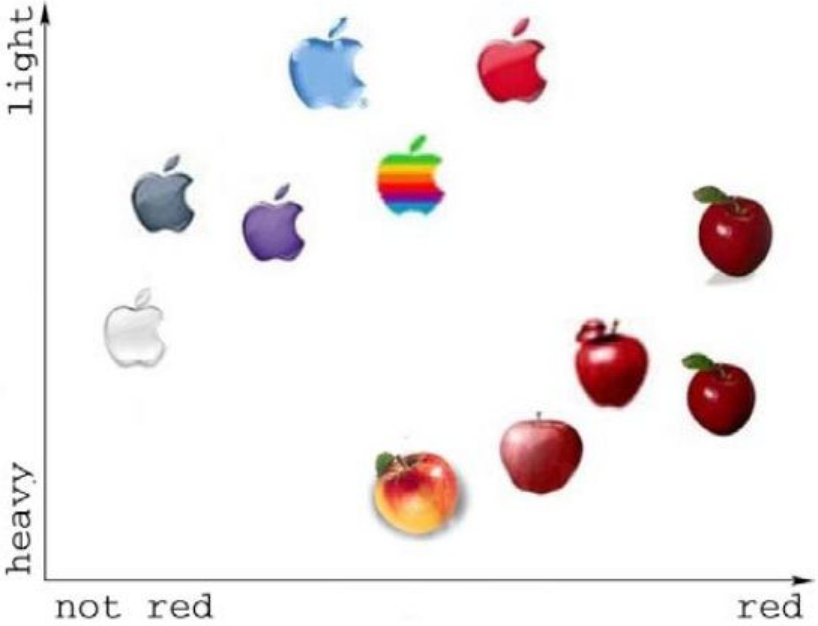


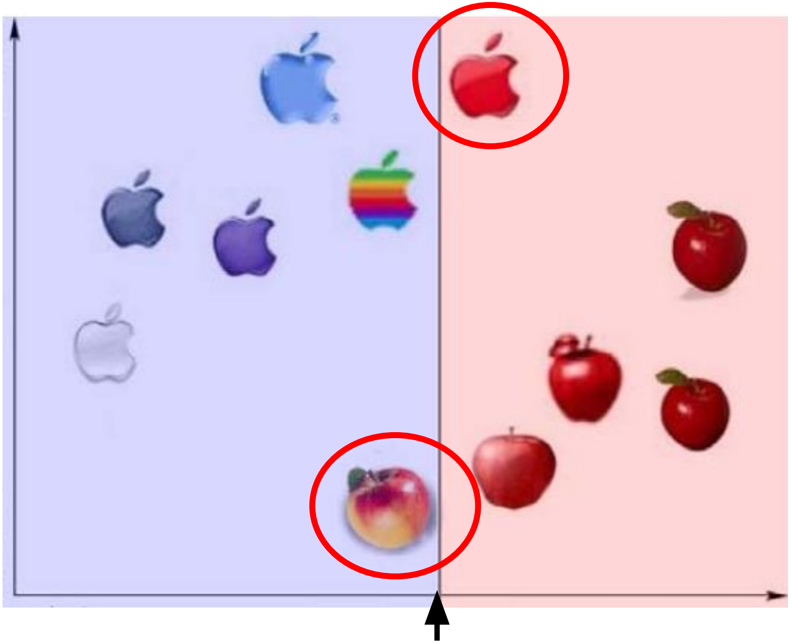
Plastic Apples

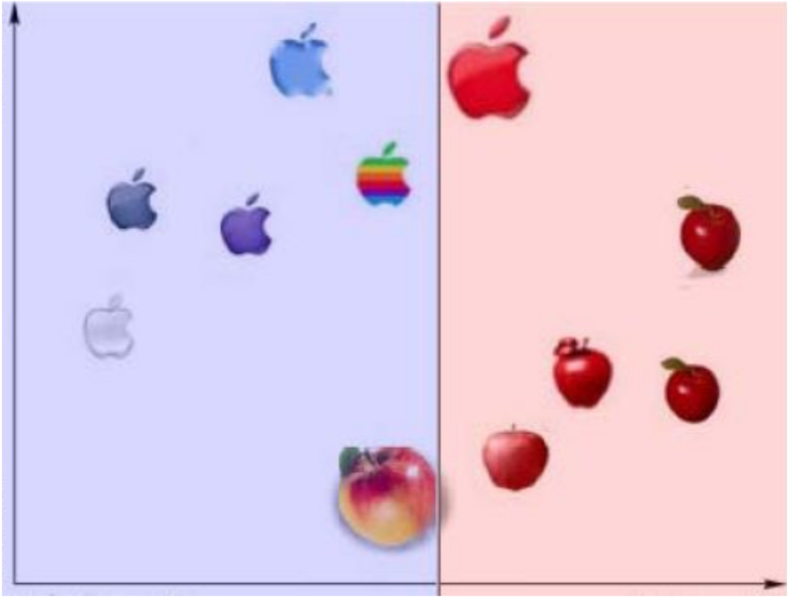
Boosting Strategies

1. Have many rules (base classifiers) to **vote** on the decision
2. Sequentially train rule that **corrects** mistakes of previous rule → focus on **hard** examples
3. Give higher **weight** to better rules

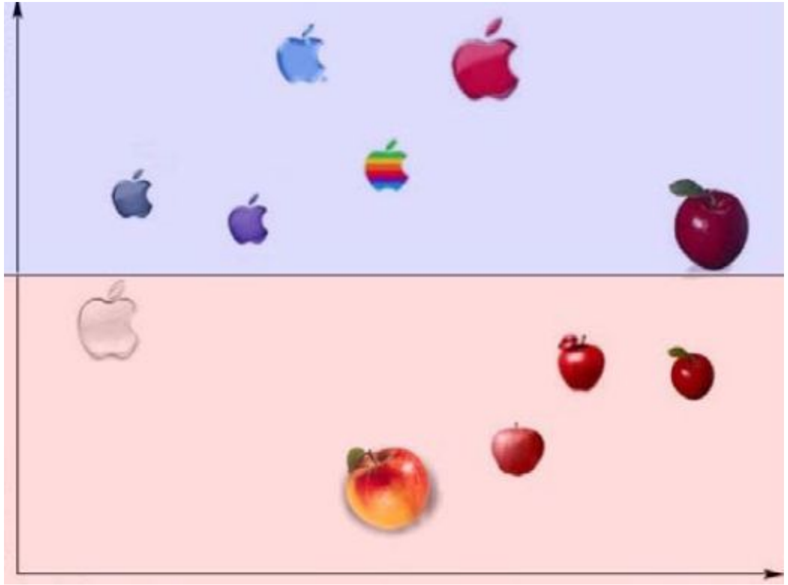
2. How boosting work: an example


light
heavy
not red
red

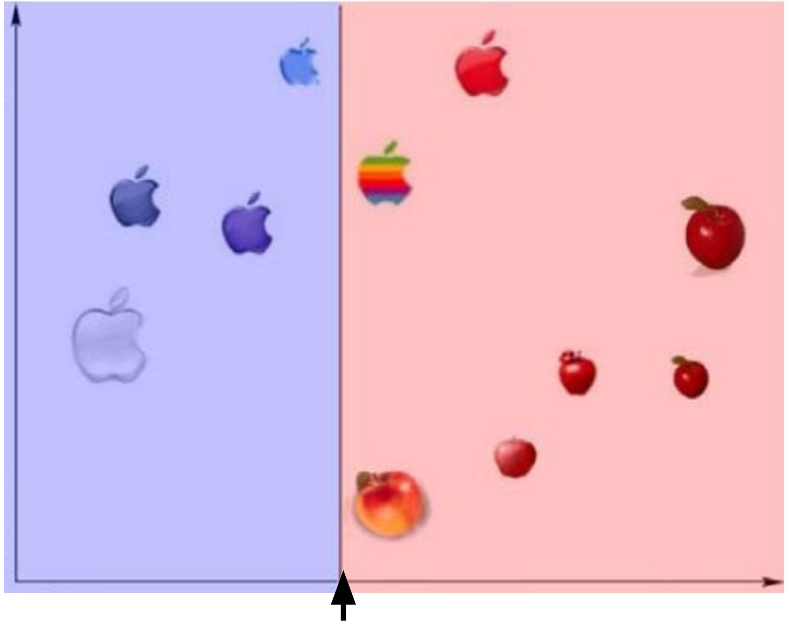

1st Simple Rule



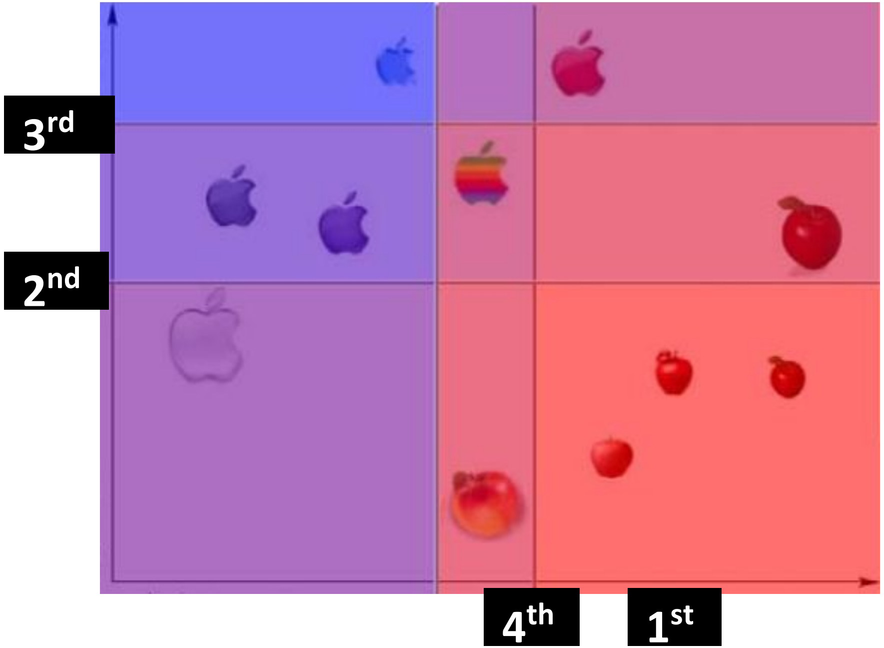
2nd Rule

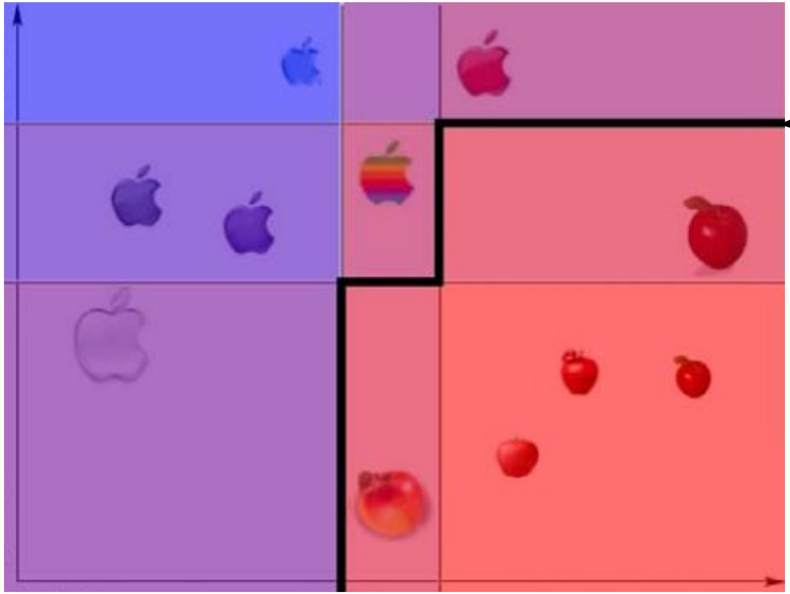


3rd Rule

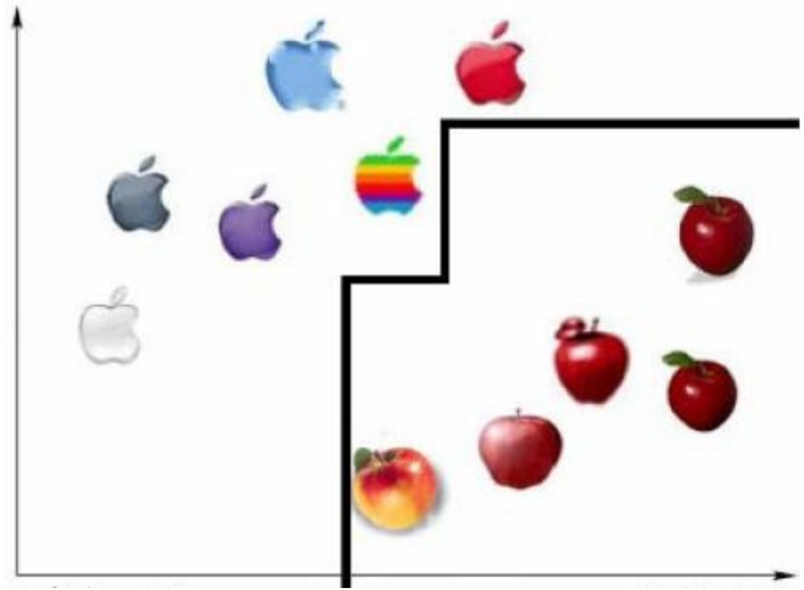


4th Rule


3rd
2nd
4th
1st



A More
Complex
Rule



Boosting vs. Bagging

- Similar to bagging, boosting combines a weighted sum of many classifiers, thus **it reduces variance**.
- One key difference: unlike bagging, boosting fit the tree to the entire training set, and adaptively weight the examples. Boosting tries to do better at each iteration, thus **it reduces bias**.
- In general: **Boosting > Bagging > Decision Tree**

Why do Ensemble Learning Work?

Based on one of 2 basic observations:

- **Variance reduction:** if the training sets are completely independent, it will always help to average an ensemble because this will reduce variance without affecting bias (e.g. bagging)
- **Bias reduction:** for simple models, average of models can reduce bias substantially by increasing capacity, and control variance by fitting one component at a time (e.g. boosting)

Adaboost Algorithm (Proposed by Robert Schapire)

Training Data: $\mathbf{D} = \{(\mathbf{x}_i, y_i) | \mathbf{x}_i \in \mathcal{R}^n, y_i \in \{-1, 1\}, 1 \leq i \leq m\}$

Set uniform example weight $w_i, 1 \leq i \leq m$

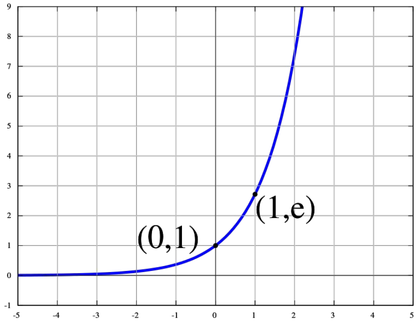
For `t = 1` to `T` iterations:

Select a base classifier: $h_t(\mathbf{x}_i) = \arg\min(\epsilon_t)$

$$\epsilon_t = \sum_{i=1}^{m} w_i [y_i \neq h_t(\mathbf{x}_i)]$$

Set classifier weight: $\alpha_t = \frac{1}{2} \ln \frac{1-\epsilon_t}{\epsilon_t}$

Update example weight: $w_i = w_i e^{-\alpha_t y_i h_t(\mathbf{x}_i)}$

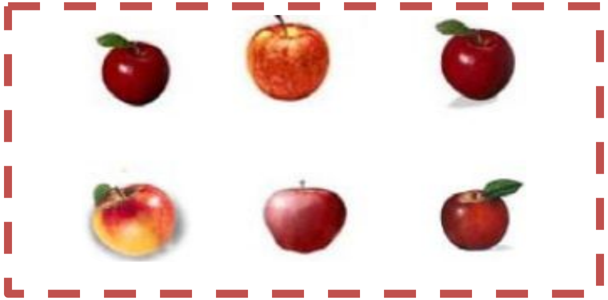


Final Classifier: $\hat{f} = \text{sign}\,(\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x}_i))$

3. Demo and Application

Demo in Python

Real Apple



Plastic Apple

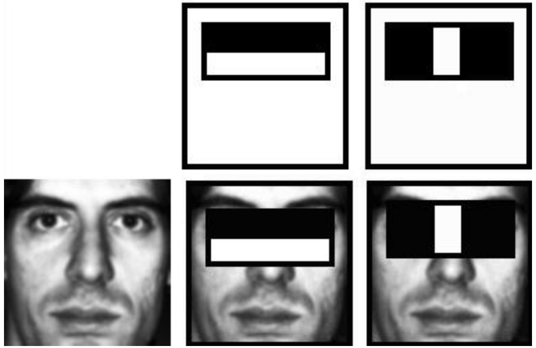
Why use boosting?

- Fast and simple to code
- No hyper-parameter to tune (except for **`T`**)
- No prior knowledge needed about base classifier
- Flexible to combine with any learning algorithm

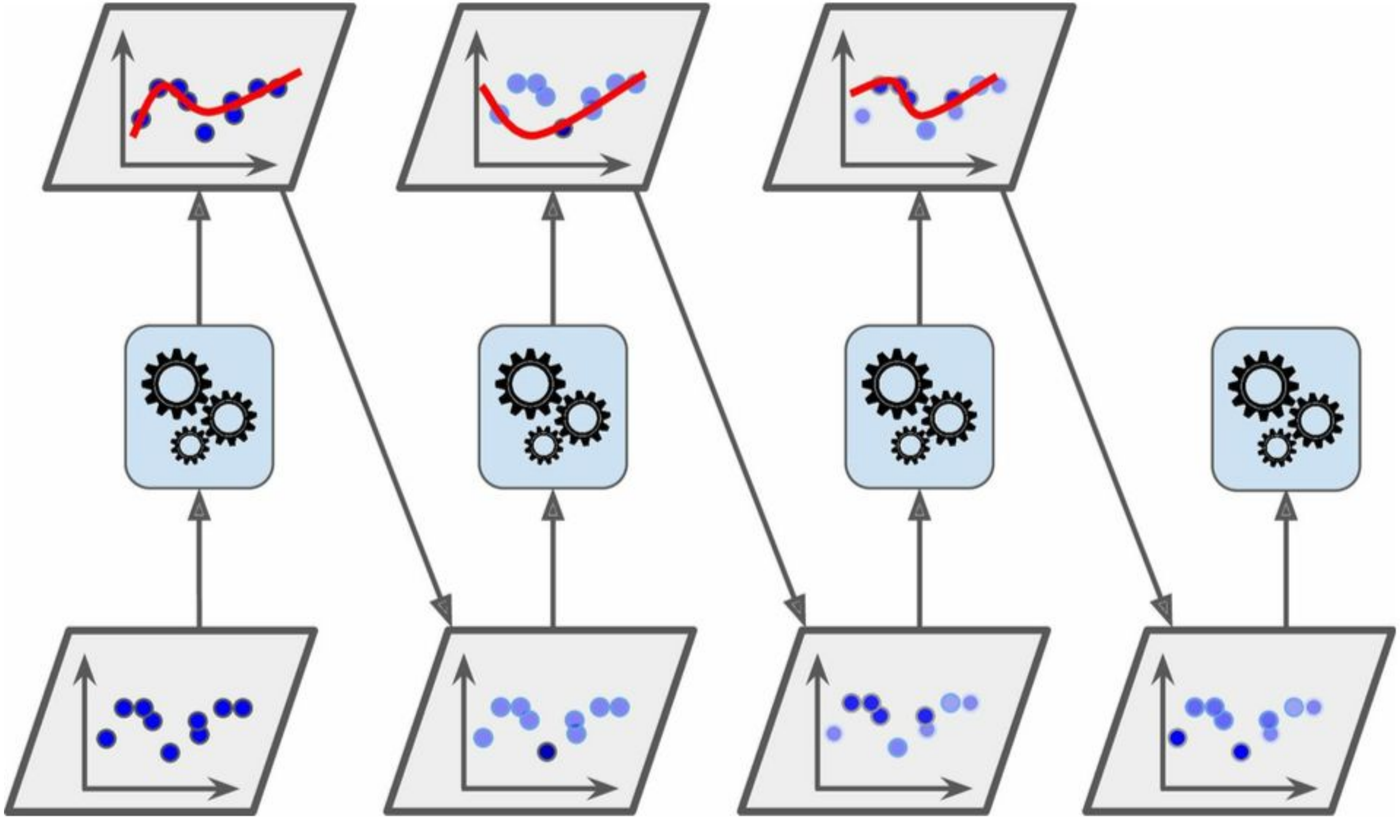
An application in Face Detection

- Viola-Jones Face Detector
- Uses Adaboost algorithm to combine lots of simple rules for face detection

Simple Rules



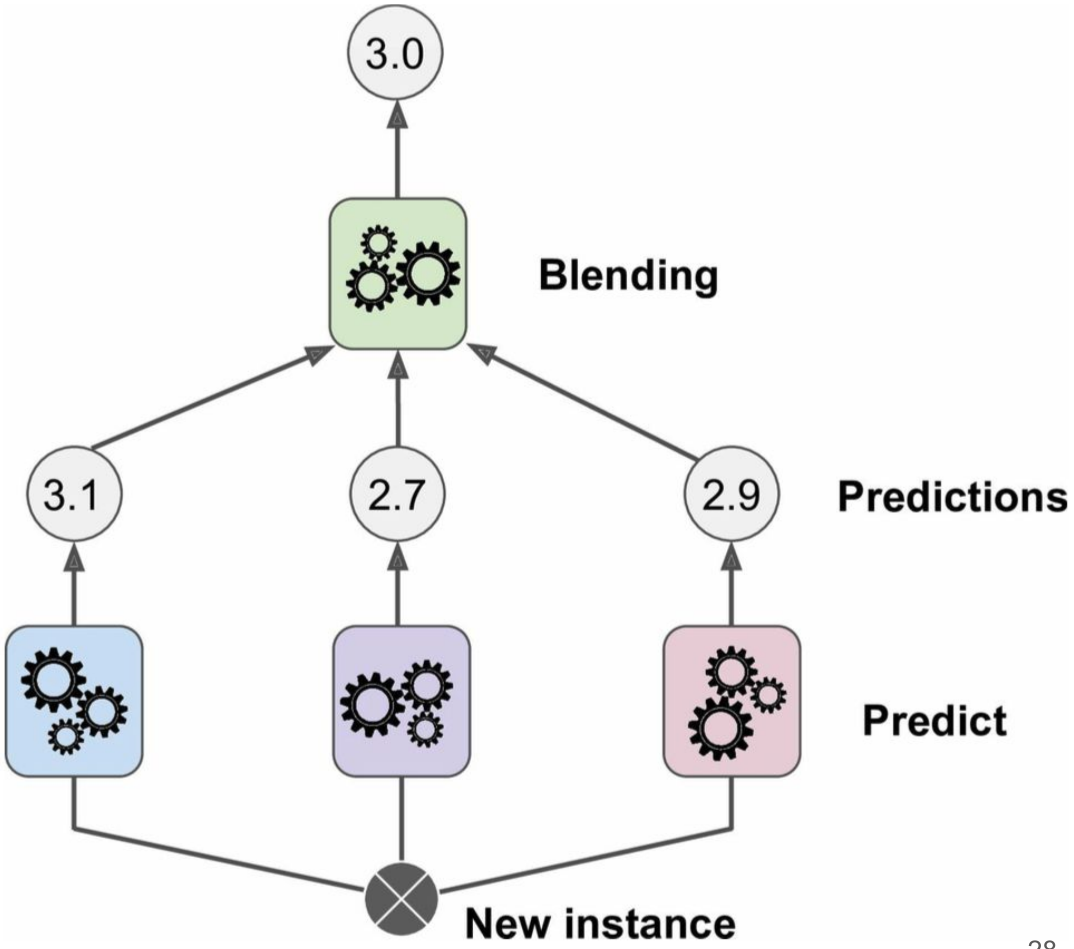
Boosting (visual perspective)



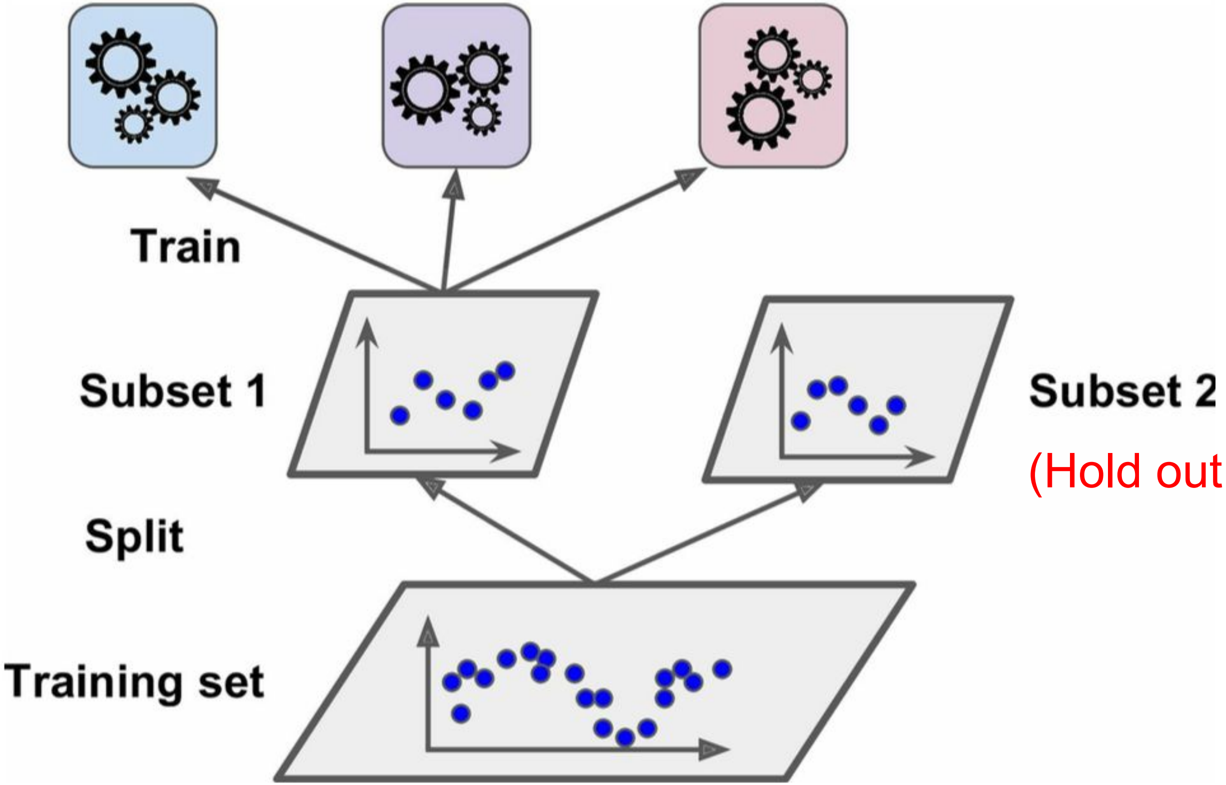
4. Stacking

Stacking

Instead of using a trivial function (such as hard voting) to aggregate the predictions of the ensemble, we train a model to perform the aggregation.

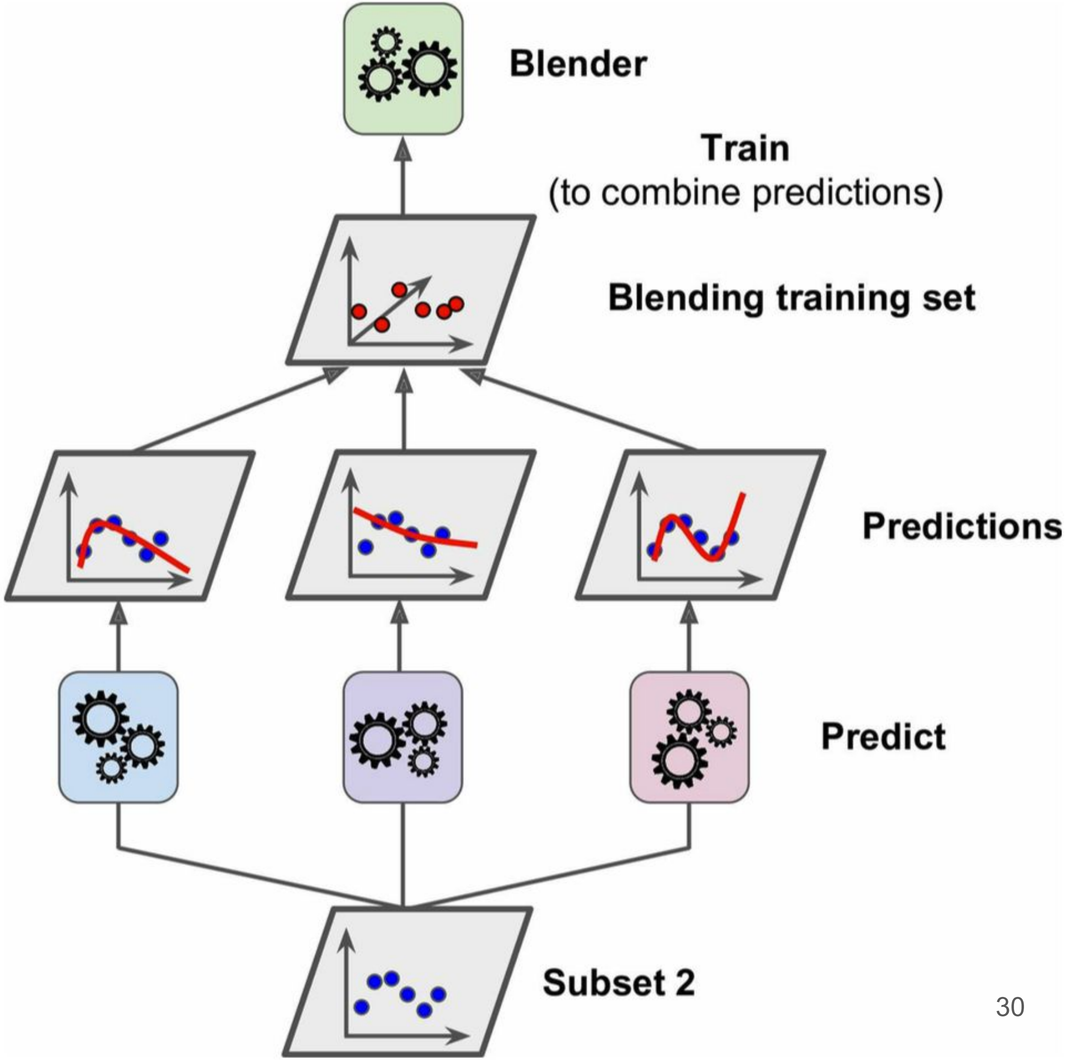


Training the predictors

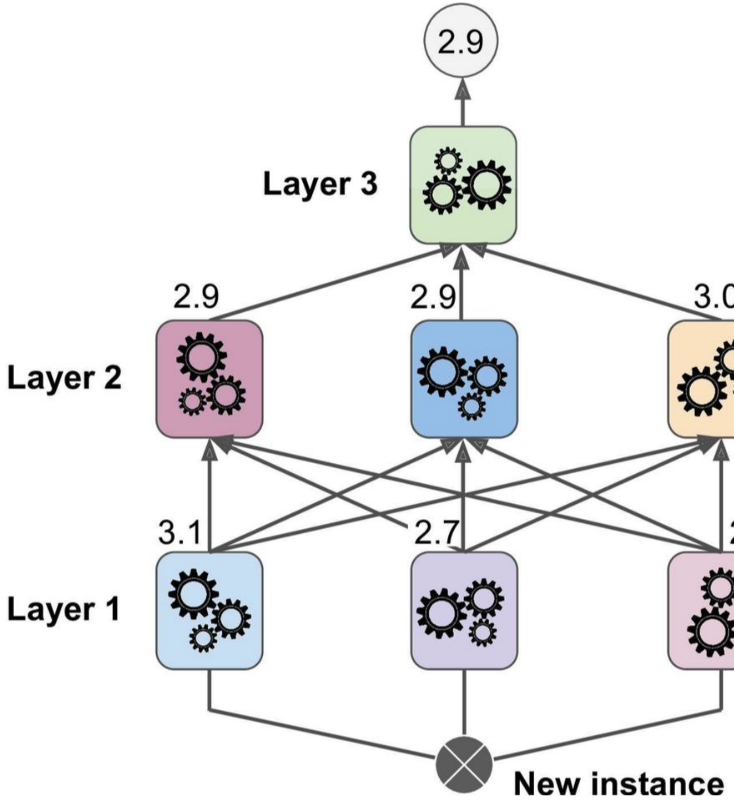


(Hold out for later)

Training the blender



Prediction in a multilayer stacking ensemble



This resembles a neural network… (more on that later in this course)

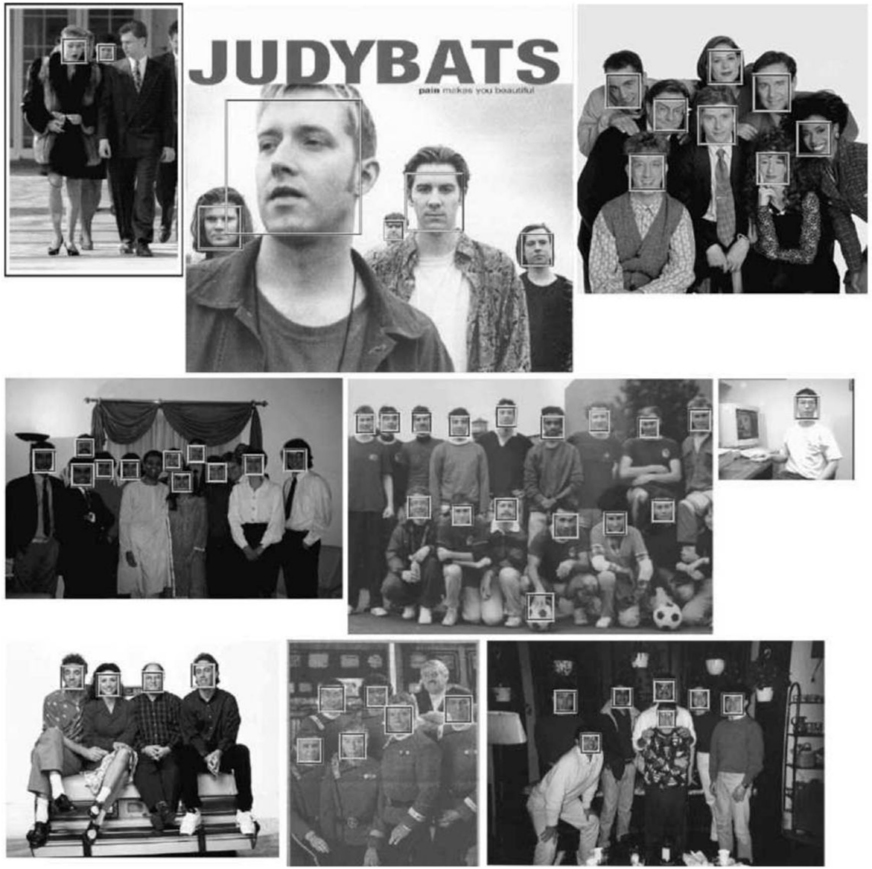
Today: Learning Objectives

1. Teach machine how to recognize apples
2. Understand how boosting works
3. Demonstrate boosting on a toy dataset and application
4. Go over Stacking

Next is Unsupervised Learning:
Clustering and Dimensionality Reduction

Bonus Slides

Adaboost Face Detection Results



Summary: Variance-Bias Tradeoff

We always try to minimize two sets of errors:

Variance: error from sensitivity to small fluctuations in the training set

Bias: error from the assumptions in the model

Variance-bias decomposition is a way of analyzing the generalization error as a sum of 3 terms: variance, bias and irreducible error (resulting from the problem itself)

Various Loss Functions