CPSC429/529: Machine Learning

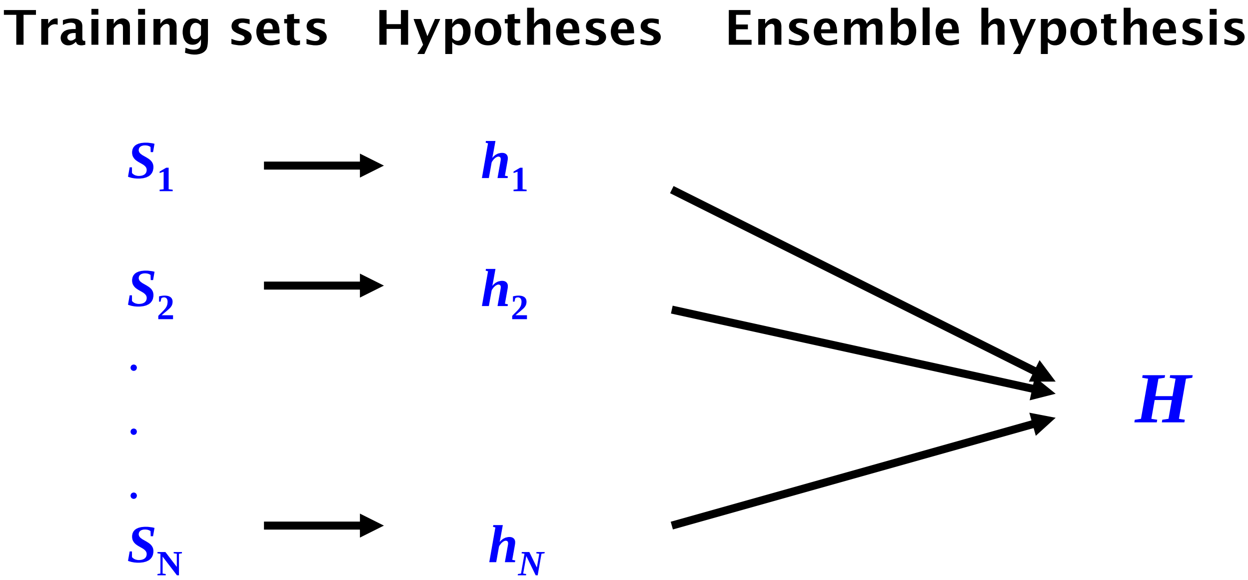
Lecture 06: Ensemble Learning

Dongsheng Che
Computer Science Department
East Stroudsburg University

Ensemble Learning

Introduction

Ensemble learning



Ensemble Learning Algorithms

- Voting
- Bagging
- Random Forest
- Boosting

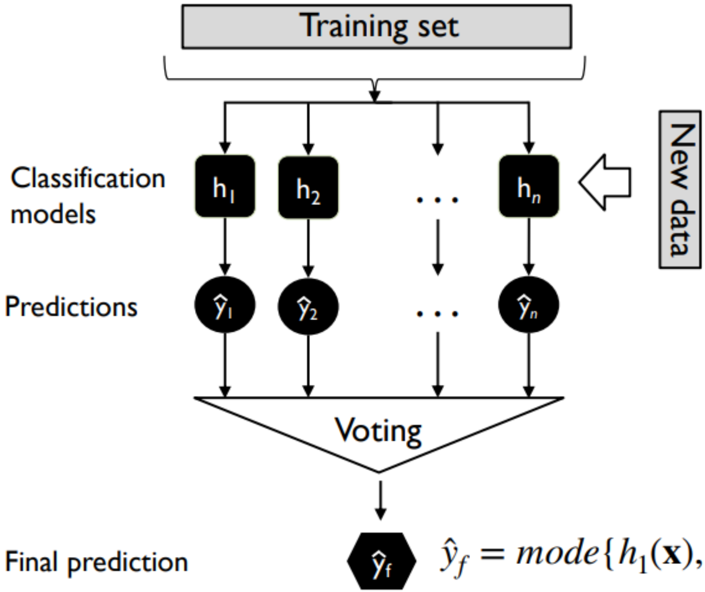
Ensemble Learning

Introduction

Ensemble Learning

Voting

Voting



$$\hat{y}_f = mode\{h_1(\mathbf{x}), h_2(\mathbf{x}), \ldots h_n(\mathbf{x})\}$$

where $h_i(\mathbf{x}) = \hat{y}_i$

- Generally combine **different** classifiers
- Classifiers use **identical** dataset

https://sebastianraschka.com/pdf/lecture-notes/stat451fs20/07-ensembles__slides.pdf

Why voting?

Given three hypotheses, h_1, h_2, $h_{3,}$ with $h_i(\mathbf{x}) \in \{-1,1\}$

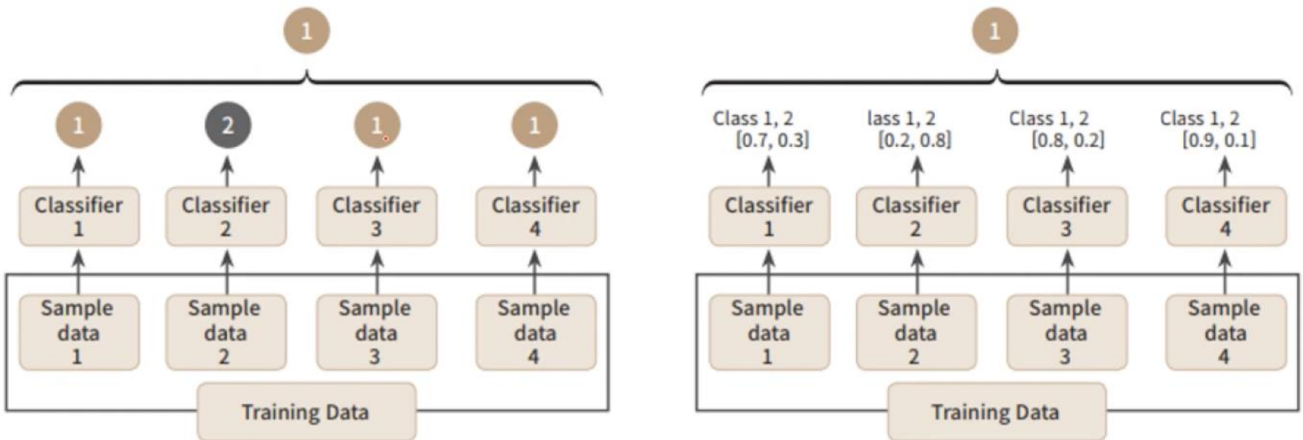
Suppose each h_i has 60% generalization accuracy, and assume errors are independent.

Now suppose $H(\mathbf{x})$ is the majority vote of h_1, h_2, and h_3 . What is probability that H is correct?

h_1	h_2	h_3	***H***	*probability*
correct	correct	correct	**correct**	.6*.6*.6=.216
correct	correct	incorrect	**correct**	.6*.6*.4=.144
correct	incorrect	correct	**correct**	.6*.4*.6=.144
correct	incorrect	incorrect	**incorrect**	.6*.4*.4=.096
Incorrect	correct	correct	**correct**	.4*.6*.6=.144
Incorrect	correct	incorrect	**incorrect**	.4*.6*.4=.096
Incorrect	incorrect	correct	**incorrect**	.4*.4*.6=.096
incorrect	incorrect	incorrect	**incorrect**	.4*.4*.4=.064

.216+3*.144=**0.648** > 60!!!

Hard Voting vs. Soft Voting



- **Hard-voting**: the majority output is chosen to be the final result of the model.
- **Soft-voting**: sums the predicted probabilities for class lables and returns the final classification with the largest sum probability.

https://velog.io/@jiselectric/Ensemble-Learning-Voting-and-Bagging-at6219ae

Voting Example

In [5]:

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import VotingClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC

log_clf = LogisticRegression(solver="lbfgs", random_state=42)
rnd_clf = RandomForestClassifier(n_estimators=100, random_state=42)
svm_clf = SVC(gamma="scale", random_state=42)

voting_clf = VotingClassifier(
    estimators=[('lr', log_clf), ('rf', rnd_clf), ('svc', svm_clf)],
    voting='hard')
```

In [6]:

```
voting_clf.fit(X_train, y_train)
```

Voting Accuracy

```
In [7]: from sklearn.metrics import accuracy_score

        for clf in (log_clf, rnd_clf, svm_clf, voting_clf):
            clf.fit(X_train, y_train)
            y_pred = clf.predict(X_test)
            print(clf.__class__.__name__, accuracy_score(y_test, y_pred))
```

```
LogisticRegression 0.864
RandomForestClassifier 0.896
SVC 0.896
VotingClassifier 0.912
```

Ensemble Learning

Voting

Ensemble Learning

Bagging

(Bootstrap Aggregating)

Bagging

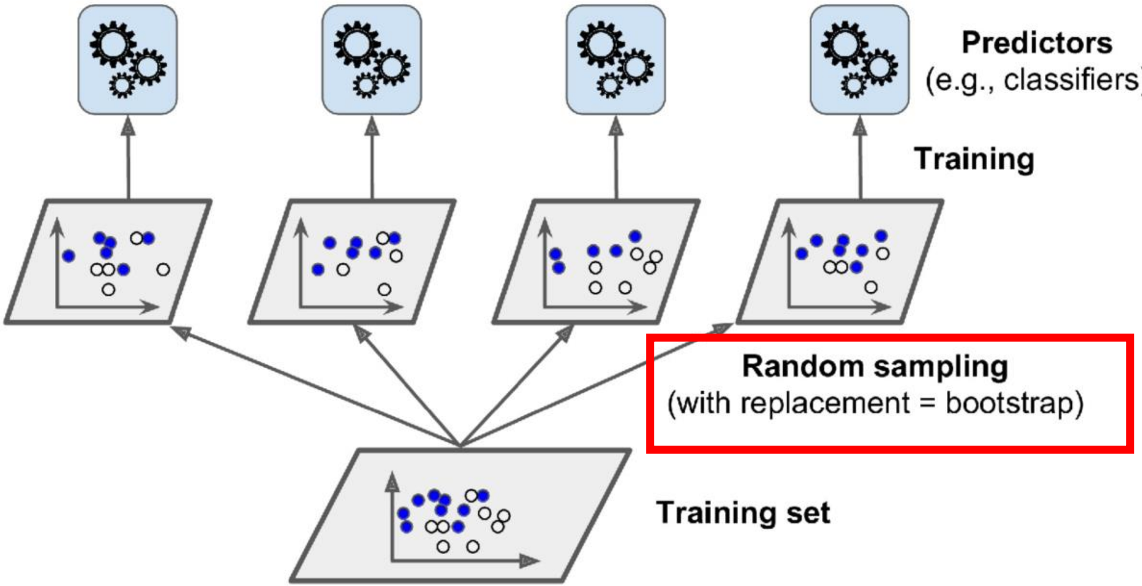


Figure 7-4. Bagging and pasting involves training several predictors on different random samples of the training set

Bootstrap Sampling

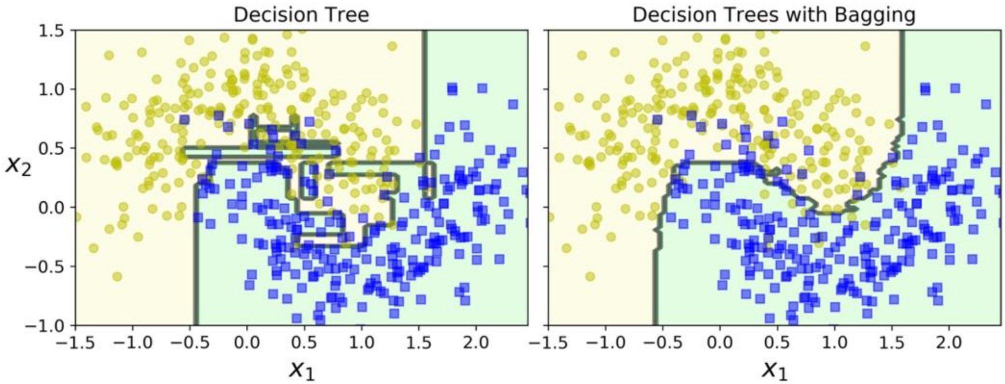
https://sebastianraschka.com/pdf/lecture-notes/stat451fs20/07-ensembles__slides.pdf

Bagging Example

```
In [11]: from sklearn.ensemble import BaggingClassifier
         from sklearn.tree import DecisionTreeClassifier

In [12]: bag_clf = BaggingClassifier(DecisionTreeClassifier(), n_estimators=500,
                                     max_samples=100, bootstrap=True, n_jobs=-1)

In [13]: bag_clf.fit(X_train, y_train)

Out[13]: BaggingClassifier(base_estimator=DecisionTreeClassifier(), max_samples=100,
                           n_estimators=500, n_jobs=-1)

In [14]: y_hat = bag_clf.predict(X_val)
```

Decision Boundary of Bagging

Figure 7-5 compares the decision boundary of a single Decision Tree with the decision boundary of a bagging ensemble of 500 trees (from the preceding code), both trained on the moons dataset. As you can see, the ensemble's predictions will likely generalize much better than the single Decision Tree's predictions: the ensemble has a comparable bias but a smaller variance (it makes roughly the same number of errors on the training set, but the decision boundary is less irregular).



Ensemble Learning

Bagging

(Bootstrap Aggregating)

Ensemble Learning

Random Forest

Random Forest

- Is a variant of bagging algorithm (i.e., bootstrap sampling)
- The base classifiers are decision trees.
- Use a random feature subset at each node split

$$f = \sqrt{F}(total\_features)$$

Random Forest v.s. Bagging Tree

Random forests--

Only m<M features considered for each node for split

Bagging Trees--

All of M features considered for each node for a split

m can be selected via out-of-bag error,
but ***m*** = sqrt(***M***) is a good value to start with

https://stats.stackexchange.com/questions/264129/what-is-the-difference-between-bagging-and-random-forest-if-only-one-explanatory

Random Forest Example

In [21]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [22]:
```
rnd_clf = RandomForestClassifier(n_estimators=500, max_leaf_nodes=16, n_jobs=-1)
```

In [23]:
```
rnd_clf.fit(X_train, y_train)
```

Out[23]:
```
RandomForestClassifier(max_leaf_nodes=16, n_estimators=500, n_jobs=-1)
```

In [24]:
```
y_pred_rf = rnd_clf.predict(X_val)
```

- The following `BaggingClassifier` is roughly equivalent to the previous `RandomForestClassifier`:

In [25]:
```
bag_clf = BaggingClassifier(
    DecisionTreeClassifier(splitter='random', max_leaf_nodes=16),
    n_estimators=500, max_samples=1.0, bootstrap=True, n_jobs=-1
)
```

Extremely Randomized Trees

	Decision Tree	Random Forest	Extra Trees
Number of trees	1	Many	Many
No of features considered for split at each decision node	All Features	Random subset of Features	Random subset of Features
Boostrapping(Drawing Sampling without replacement)	Not applied	Yes	No
How split is made	Best Split	Best Split	Random Split

- Using random threshold for each feature rather than searching for the best possible threshold

Ensemble Learning

Random Forest

Ensemble Learning

Boosting

AdaBoost

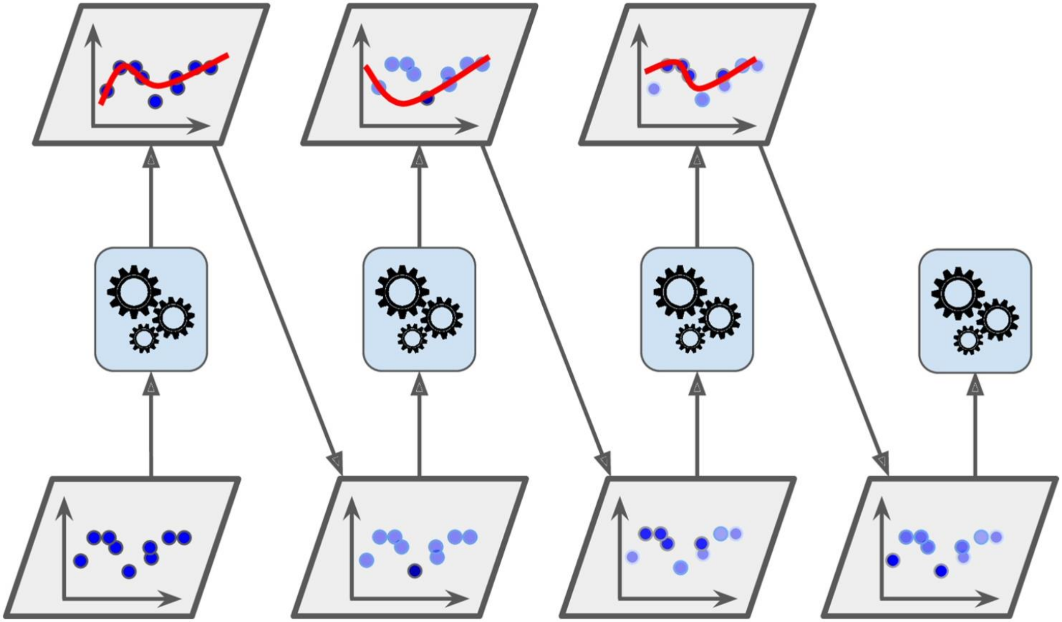


Figure 7-7. AdaBoost sequential training with instance weight updates

AdaBoost General Procedure

Train Base Classifiers:

1. Initialize a weight vector with uniform weights
2. Loop:
   - Apply weak learner to weighted training examples (instead of orig. training set, may draw bootstrap samples with weighted probability)
   - increase weight for misclassified examples

Ensemble Predictor:

Weighted majority voting on trained classifiers

Adaboost algorithm

__**Train Base Classifiers**__:

1. Given $S = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$ where $\mathbf{x} \in \boldsymbol{X}$, $y_i \in \{+1, -1\}$
2. Initialize $\mathbf{w}_1(i) = 1/N$.   (Uniform distribution over data)
3. For $t = 1, \ldots, K$:
   a) Select new training set S_t from S with replacement, according to $\mathbf{w}_t$
   b) Train L on S_t to obtain hypothesis h_t
   c) Compute the training error ε_t of h_t on $\boldsymbol{S}$:

$$\varepsilon_t = \sum_{j=1}^{N} \mathbf{w}_t(j)\, \delta(y_j \neq h_t(\mathbf{x}_j)), \text{ where}$$

$$\delta(y_j \neq h_t(\mathbf{x}_j)) = \begin{cases} 1 \text{ if } y_j \neq h_t(\mathbf{x}_j) \\ 0 \text{ otherwise} \end{cases}$$

   d) Compute coefficient

$$\alpha_t = \frac{1}{2} \ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$$

Error is small, then coefficient is small

Incorrect prediction on its instance leads to large value in the box

e) Compute new weights on data:

For i = 1 to N

$$\mathbf{w}_{t+1}(i) = \frac{\mathbf{w}_t(i)\ \exp\boxed{(-\alpha_t y_i h_t(\mathbf{x}_i))}}{Z_t}$$

where Z_t is a normalization factor chosen so that $\mathbf{w}_{t+1}$ will be a probability distribution:

$$Z_t = \sum_{i=1}^{N} \mathbf{w}_t(i)\ \exp(-\alpha_t y_i h_t(\mathbf{x}_i))$$

Ensemble Prediction:

$$H(\mathbf{x}) = \mathrm{sgn} \sum_{t=1}^{K} \alpha_t h_t(\mathbf{x})$$

A Hypothetical Example

$$S = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6, \mathbf{x}_7, \mathbf{x}_8,\}$$

where **{** $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4$ } are class +1

{$\mathbf{x}_5, \mathbf{x}_6, \mathbf{x}_7, \mathbf{x}_8$ } are class −1

***t* = 1**:

$\mathbf{w}_1 = \{1/8, 1/8, 1/8, 1/8, 1/8, 1/8, 1/8, 1/8\}$

$S_1 = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_2, \mathbf{x}_5, \mathbf{x}_5, \mathbf{x}_6, \mathbf{x}_7, \mathbf{x}_8\}$ (notice some repeats)

Train classifier on S_1 to get h_1

Run h_1 on **S**. Suppose classifications are: {**1**, **−1**, **−1**, **−1**, **−1**, **−1**, **−1**, **−1**}

Calculate error:

$$\epsilon_1 = \sum_{j=1}^{N} \mathbf{w}_t(j)\,\delta(y_j \neq h_t(\mathbf{x}_j)) = \frac{1}{8}(3) = .375$$

Calculate α's:

$$a_1 = \frac{1}{2}\ln\left(\frac{1 - e_t}{e_t}\right) = .255$$

Calculate new **w**'s:

$$\mathbf{w}_{t+1}(i) = \frac{\boxed{\mathbf{w}_t(i)\ \exp(-a_t y_i h_t(\mathbf{x}_i))}}{Z_t}$$

$$\hat{\mathbf{w}}_2(1) = (.125)\exp(-.255(1)(1)) = 0.1$$

$$\boxed{\begin{aligned}\hat{\mathbf{w}}_2(2) &= (.125)\exp(-.255(1)(-1)) = 0.16\\ \hat{\mathbf{w}}_2(3) &= (.125)\exp(-.255(1)(-1)) = 0.16\\ \hat{\mathbf{w}}_2(4) &= (.125)\exp(-.255(1)(-1)) = 0.16\end{aligned}}$$

$$\hat{\mathbf{w}}_2(5) = (.125)\exp(-.255(-1)(-1)) = 0.1$$

$$\hat{\mathbf{w}}_2(6) = (.125)\exp(-.255(-1)(-1)) = 0.1$$

$$\hat{\mathbf{w}}_2(7) = (.125)\exp(-.255(-1)(-1)) = 0.1$$

$$\hat{\mathbf{w}}_2(8) = (.125)\exp(-.255(-1)(-1)) = 0.1$$

$$\mathbf{w}_2(1) = 0.1 / .98 = 0.102$$

$$\mathbf{w}_2(2) = 0.163$$

$$\mathbf{w}_2(3) = 0.163$$

$$\mathbf{w}_2(4) = 0.163$$

$$\mathbf{w}_2(5) = 0.102$$

$$\mathbf{w}_2(6) = 0.102$$

$$\mathbf{w}_2(7) = 0.102$$

$$\mathbf{w}_2(8) = 0.102$$

$$Z_1 = \sum_i \hat{\mathbf{w}}_2(i) = .98$$

$t = 2$:

$w_2 = \{0.102, 0.163, 0.163, 0.163, 0.102, 0.102, 0.102, 0.102\}$

$S_2 = \{x_1, x_2, x_2, x_3, x_4, x_4, x_7, x_8\}$

Run classifier on S_2 to get h_2

Run h_2 on S. Suppose classifications are: {1, 1, 1, 1, 1, 1, 1, 1}

Calculate error:

$$e_2 = \sum_{j=1}^{N} \mathbf{w}_t(j)\,\delta(y_j \neq h_t(\mathbf{x}_j))$$

$$= (.102) \times 4 = 0.408$$

Calculate α's:

$$\alpha_2 = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right) = .186$$

Calculate **w**'s:

$$\mathbf{w}_{t+1}(i) = \frac{\mathbf{w}_t(i)\ \exp(-\alpha_t y_i h_t(\mathbf{x}_i))}{Z_t}$$

$$\hat{\mathbf{w}}_3(1) = (.102)\exp(-.186(1)(1)) = 0.08$$
$$\hat{\mathbf{w}}_3(2) = (.163)\exp(-.186(1)(1)) = 0.135$$
$$\hat{\mathbf{w}}_3(3) = (.163)\exp(-.186(1)(1)) = 0.135$$
$$\hat{\mathbf{w}}_3(4) = (.163)\exp(-.186(1)(1)) = 0.135$$
$$\hat{\mathbf{w}}_3(5) = (.102)\exp(-.186(-1)(1)) = 0.122$$
$$\hat{\mathbf{w}}_3(6) = (.102)\exp(-.186(-1)(1)) = 0.122$$
$$\hat{\mathbf{w}}_3(7) = (.102)\exp(-.186(-1)(1)) = 0.122$$
$$\hat{\mathbf{w}}_3(8) = (.102)\exp(-.186(-1)(1)) = 0.122$$

$$\mathbf{w}_3(1) = 0.08/.973 = 0.082$$
$$\mathbf{w}_3(2) = 0.139$$
$$\mathbf{w}_3(3) = 0.139$$
$$\mathbf{w}_3(4) = 0.139$$
$$\mathbf{w}_3(5) = 0.125$$
$$\mathbf{w}_3(6) = 0.125$$
$$\mathbf{w}_3(7) = 0.125$$
$$\mathbf{w}_3(8) = 0.125$$

$$Z_2 = \sum_i \hat{\mathbf{w}}_2(i) = .973$$

***t* =3:**

w_3 = {0.082, 0.139, 0.139, 0.139, 0.125, 0.125, 0.125, 0.125}

S_3 = {x_2, x_3, x_3, x_3, x_5, x_6, x_7, x_8}

Run classifier on S_3 to get h_3

Run h_3 on S. Suppose classifications are: {1, 1 , −1, 1, -1, -1, 1, -1 }

Calculate error:

$$\varepsilon_3 = \sum_{j=1}^{N} \mathbf{w}_t(i)\,\delta(y_j \neq h_t(\mathbf{x}_j))$$

$$= (.139) + (.125) = 0.264$$

- Calculate α's:

$$\alpha_3 = \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) = .512$$

- Ensemble classifier:

$$H(\mathbf{x}) = \operatorname{sgn}\sum_{t=1}^{K}\alpha_t h_t(\mathbf{x})$$
$$= \operatorname{sgn}\left(.255\times h_1(\mathbf{x}) + .186\times h_2(\mathbf{x}) + .512\times h_3(\mathbf{x})\right)$$

Example	Actual class	h_1	h_2	h_3
$\mathbf{x}_1$	1	1	1	1
$\mathbf{x}_2$	1	−1	1	1
$\mathbf{x}_3$	1	−1	1	−1
$\mathbf{x}_4$	1	1	1	1
$\mathbf{x}_5$	−1	−1	1	−1
$\mathbf{x}_6$	−1	−1	1	−1
$\mathbf{x}_7$	−1	1	1	1
$\mathbf{x}_8$	−1	−1	1	−1

Recall the training set:

$$S = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6, \mathbf{x}_7, \mathbf{x}_8,\}$$

where $\{ x_1, x_2, x_3, x_4 \}$ are class +1

$\{x_5, x_6, x_7, x_8\}$ are class −1

What is the accuracy of *H* on the training data?

$$H(\mathbf{x}) = \operatorname{sgn} \sum_{t=1}^{T} \alpha_t h_t(\mathbf{x})$$

$$= \operatorname{sgn}\left(.255 \times h_1(\mathbf{x}) + .186 \times h_2(\mathbf{x}) + .512 \times h_3(\mathbf{x})\right)$$

Boosting Example

```
In [36]: from sklearn.ensemble import AdaBoostClassifier

In [37]: ada_clf = AdaBoostClassifier(base_estimator=DecisionTreeClassifier(max_depth=1),
                                      n_estimators=200, algorithm='SAMME.R',
                                      learning_rate=0.5)

In [38]: ada_clf.fit(X_train, y_train)

Out[38]: AdaBoostClassifier(base_estimator=DecisionTreeClassifier(max_depth=1),
                            learning_rate=0.5, n_estimators=200)

In [39]: from sklearn.metrics import accuracy_score

In [40]: accuracy_score(ada_clf.predict(X_val), y_val)

Out[40]: 0.8196969696969697
```

Summary of ensemble learning methods:

Methods	Features	Data	Weights
Voting	All features	All data	same
Bagging	All features	Bootstrap sampling	same
Random Forest	Subspace sampling	Bootstrap sampling	same
Boosting	All features	Bootstrap sampling	Miss-classified instances have high weights

Ensemble Learning

Boosting