Data Science for Business
Ensembles

Asst. Prof. Teerapong Leelanupab (Ph.D.)
Faculty of Information Technology
King Mongkut's Institute of Technology Ladkrabang (KMITL)

Week 14

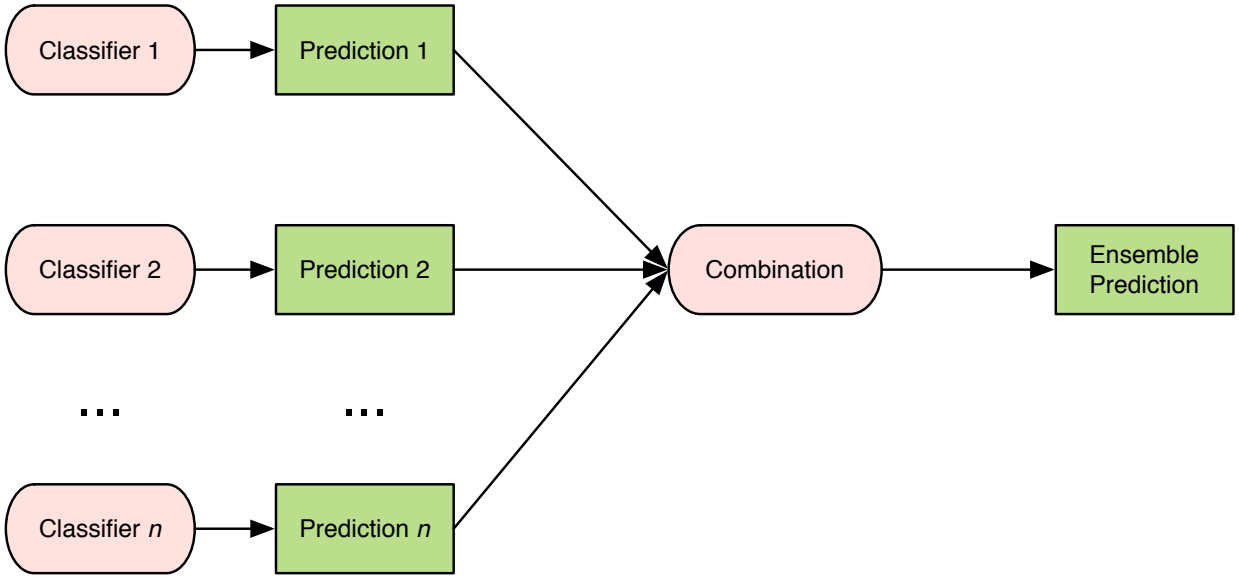
Overview

- Ensemble Classification
- Why do ensembles work?
  - Condorcet Jury Theorem
- Ensemble Generation
  - Bagging v Boosting
- Ensemble Combination
  - Voting v Weighted Voting

Ensemble Idea

Classfier ชนิดเดียวหลายๆตัว ไม่ใช่แปงเป็นหลายๆชนิด

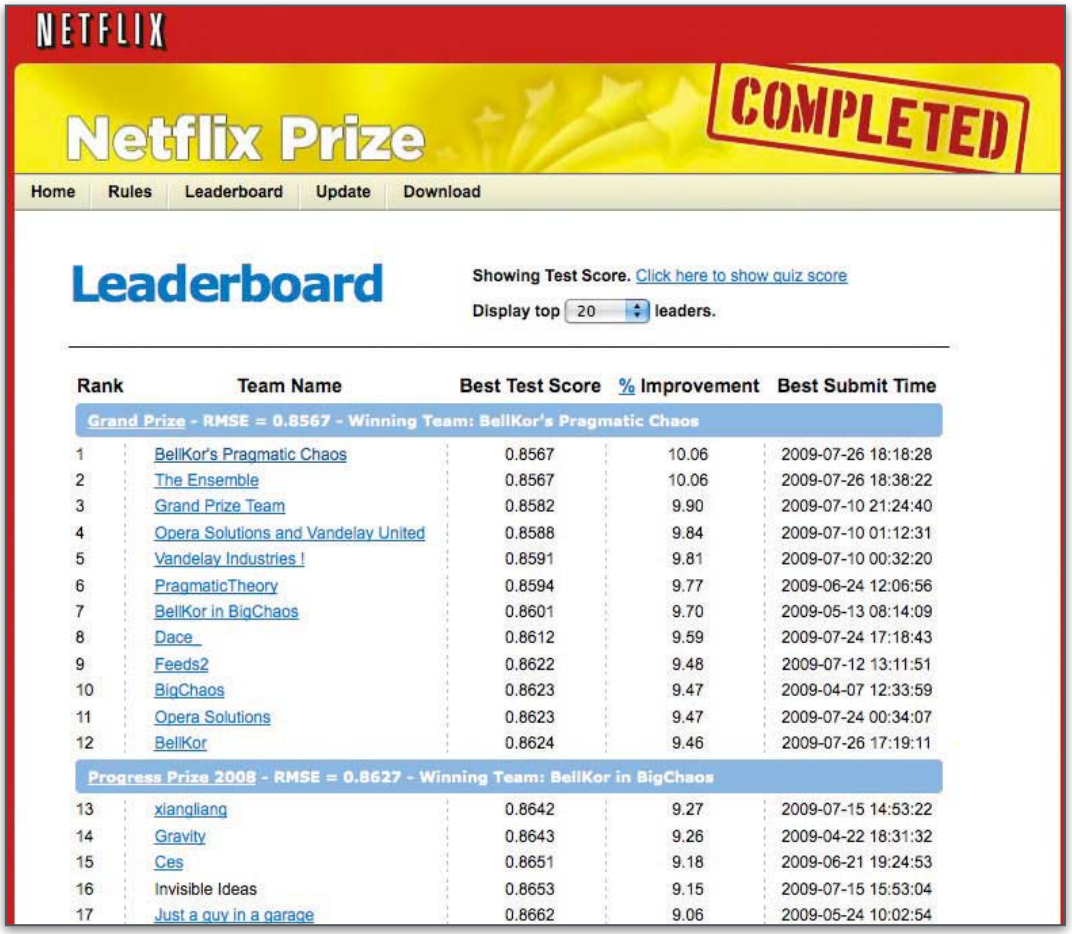
- Ensemble Classification: Aggregation of predictions made by multiple classifiers with the goal of improving accuracy.



- An ensemble of “weak learners” can provide a strong committee.
- Applied using many different types of classifiers - decision trees, *k*-NNs, neural networks, support vector machines…

Application: Netflix Prize

In 2006, Netflix announced a machine learning competition for movie rating prediction. Prize of $1 million to whoever improved the accuracy of existing system by 10%.


NETFLIX

Netflix Prize

COMPLETED

Home | Rules | Leaderboard | Update | Download

Leaderboard

Showing Test Score. Click here to show quiz score

Display top 20 leaders.

Rank	Team Name	Best Test Score	% Improvement	Best Submit Time
Grand Prize - RMSE = 0.8567 - Winning Team: BellKor's Pragmatic Chaos				
1	BellKor's Pragmatic Chaos	0.8567	10.06	2009-07-26 18:18:28
2	The Ensemble	0.8567	10.06	2009-07-26 18:38:22
3	Grand Prize Team	0.8582	9.90	2009-07-10 21:24:40
4	Opera Solutions and Vandelay United	0.8588	9.84	2009-07-10 01:12:31
5	Vandelay Industries !	0.8591	9.81	2009-07-10 00:32:20
6	PragmaticTheory	0.8594	9.77	2009-06-24 12:06:56
7	BellKor in BigChaos	0.8601	9.70	2009-05-13 08:14:09
8	Dace	0.8612	9.59	2009-07-24 17:18:43
9	Feeds2	0.8622	9.48	2009-07-12 13:11:51
10	BigChaos	0.8623	9.47	2009-04-07 12:33:59
11	Opera Solutions	0.8623	9.47	2009-07-24 00:34:07
12	BellKor	0.8624	9.46	2009-07-26 17:19:11
Progress Prize 2008 - RMSE = 0.8627 - Winning Team: BellKor in BigChaos				
13	xiangliang	0.8642	9.27	2009-07-15 14:53:22
14	Gravity	0.8643	9.26	2009-04-22 18:31:32
15	Ces	0.8651	9.18	2009-06-21 19:24:53
16	Invisible Ideas	0.8653	9.15	2009-07-15 15:53:04
17	Just a guy in a garage	0.8662	9.06	2009-05-24 10:02:54

Top submissions all combine several teams and algorithms as an ensemble.

BellKor Team:
"Our final solution consists of blending 107 individual results"

http://techblog.netflix.com/2012/04/netflix-recommendations-beyond-5-stars.html

Ensembles: Motivation

The Condorcet Jury Theorem

- Proposed by the Marquis of Condorcet in 1784, and relates to the relative probability of an ensemble of individuals arriving at a correct decision.
- If each voter has a probability *p* of being correct and the probability of a majority of voters being correct is *M*…
  - Then $p > 0.5$ implies $M > p$
  - Also if *p* always > *0.5*, then *M* approaches 1.0 as the number of voters approaches infinity.

➡ "When the average probability of an individual being correct is > 50%, the chance of the ensemble of them reaching the correct decision increases as more members are added".

Ensembles: Motivation

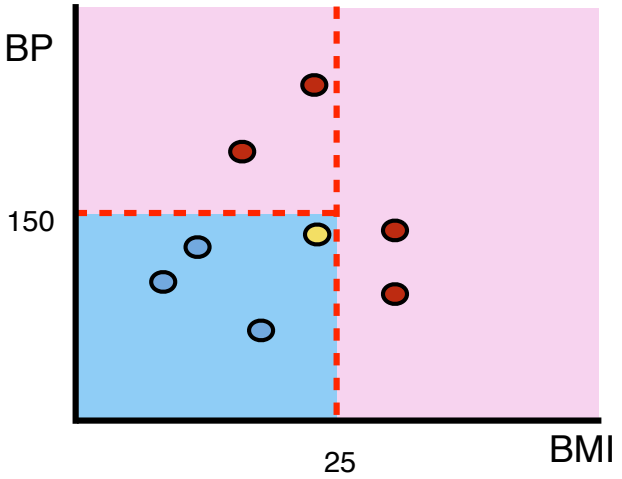
- Condorcet Jury Theorem revisited…
  - We now know that *M* will be greater than *p* only if there is diversity in the pool of voters - i.e. there is some disagreement between their decisions.
  - The probability of a majority of voters being correct will increase as the ensemble grows only if the diversity in the ensemble continues to grow as well.
- Eventually, new ensemble members will have voting patterns collinear with existing members.
- Typically the diversity of the ensemble will plateau as will the accuracy of the ensemble at some size between 10-50 members.

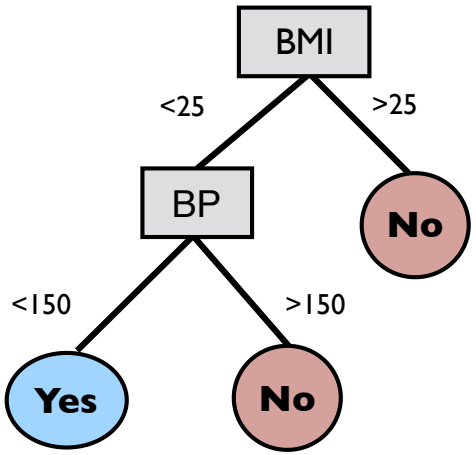
Example: Classification

- **Data:** Heart attack patient admitted. 19 variables measured during first 24 hours. Blood pressure, age, BMI + 16 other variables, considered important indicators of patient's condition.
- **Task:** Identify high risk patients (i.e. will not survive 30 days), based on evidence of initial 24-hour data.

No	Age	BMI	BP	Ok?
1	60	20	140	Yes
2	60	21	145	Yes
3	85	23	130	Yes
4	81	22	160	No
5	70	24	170	No
6	72	26	135	No
7	81	26	145	No
8	66	23	155	No

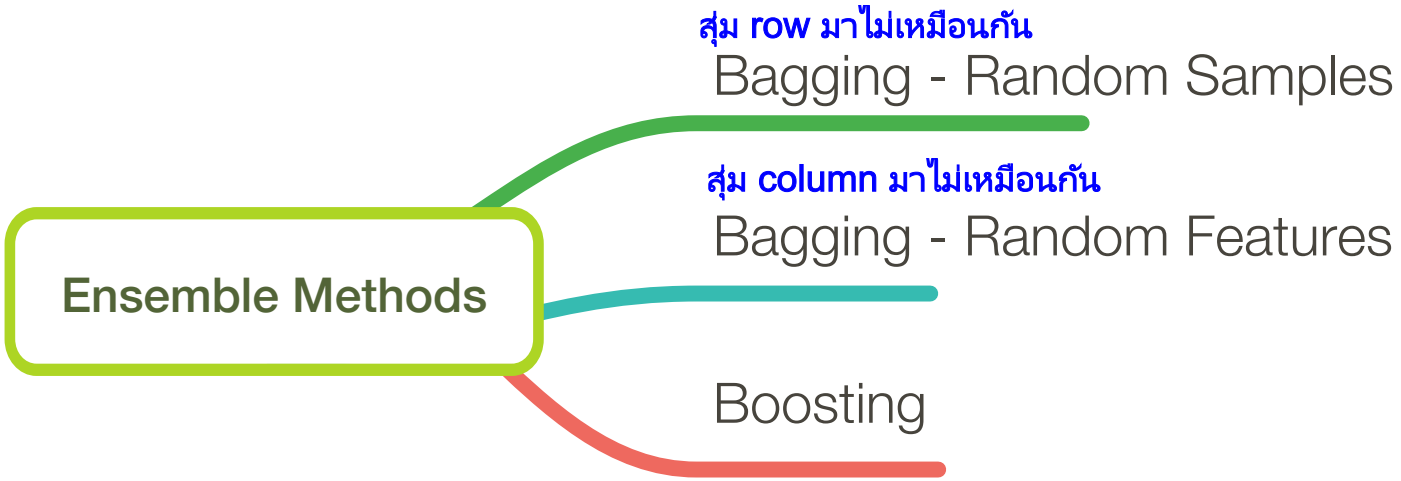
Q	66	24	148	?





Overview

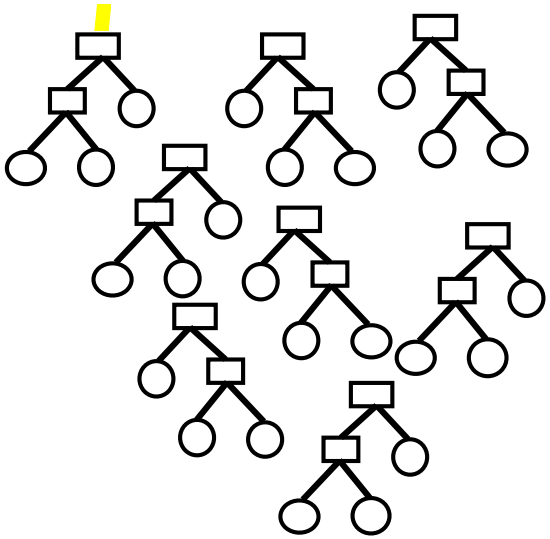
Bagging(decrease variance),
Boosting(Bias), Stacking(Improve accuracy)



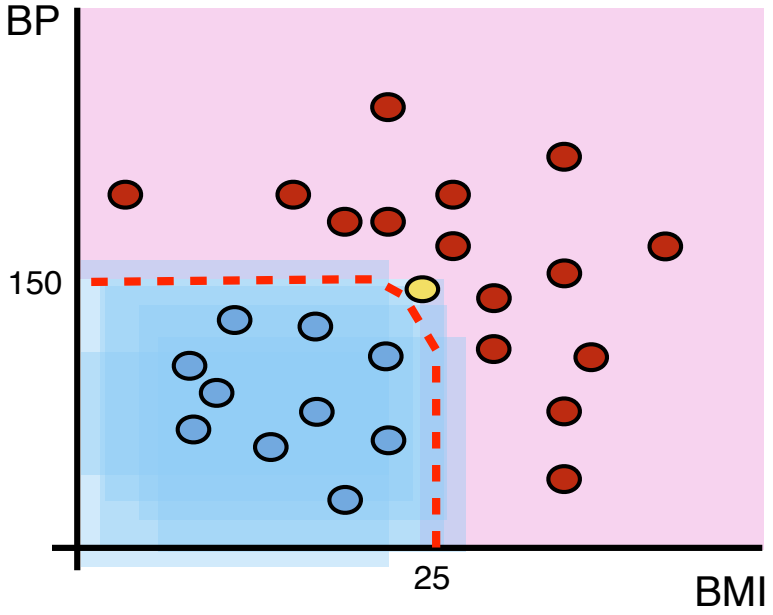
Ensemble Idea

- Build many “base” decision trees, using different subsets of the data. These trees can vote on the class of a new input example.

➡ Accuracy of ensemble should be better than the individual trees.



Ensemble of Decision Trees



Q. How do we generate base classifiers that complement each other?

Q. How do we combine the outputs of base classifiers to maximise accuracy?

Bagging

(Bootstrap Aggregating)

Random With
Replacement

สุ่มแล้วมีโอกาสซ้ำ

Breiman, L. (1996). Bagging predictors. *Machine learning*, *24*(2), 123-140.

Ensemble Generation: Bagging

- **Key Idea:** Train *n* classifiers on different subsets of the training data.
- Bootstrap aggregation / Bagging (Breiman, 1996):
  - Randomly sample from training data with replacement.
  - 100% bootstrap sample will contain ~63% of training examples. Remaining data is “out-of-bag” (OOB).

Complete dataset has 8 examples

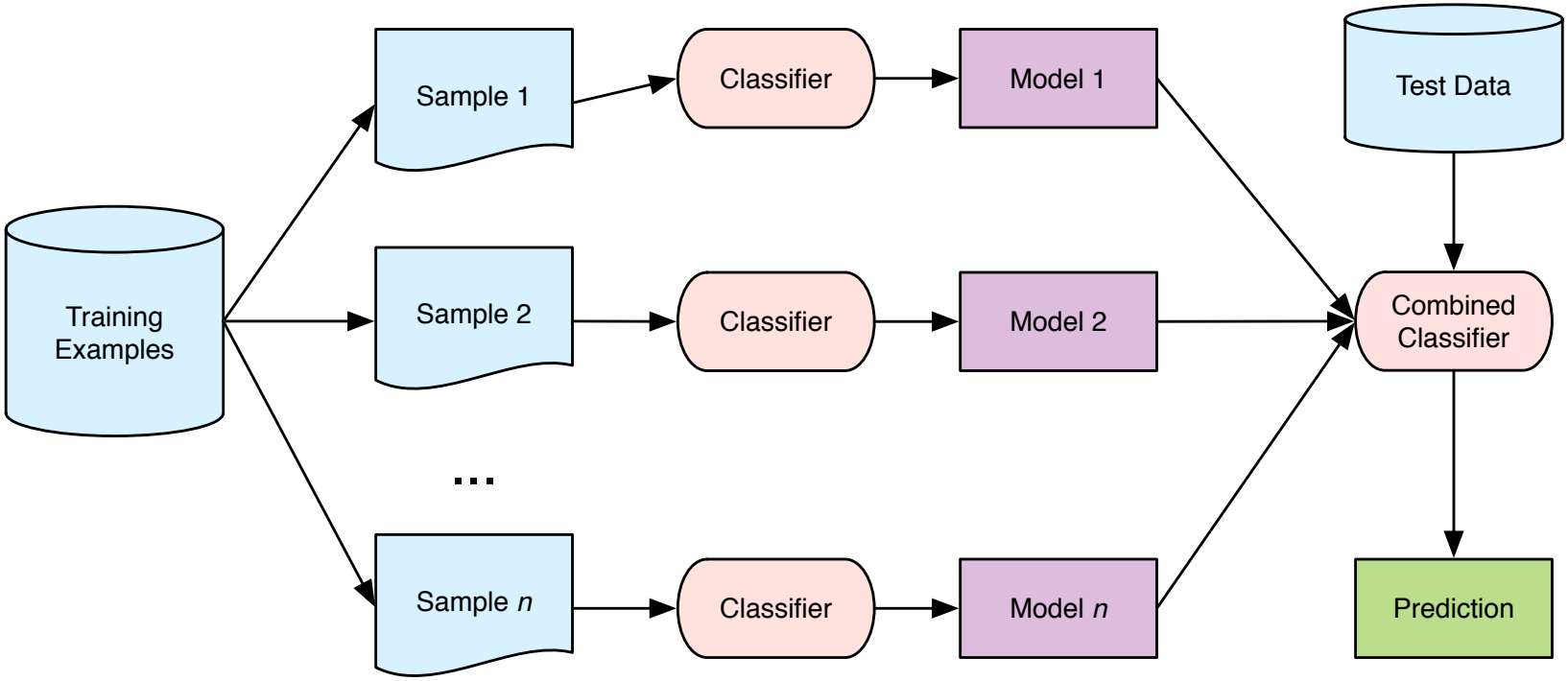
Original	A	B	C	D	E	F	G	H
Set 1	B	G	H	C	G	F	C	A
Set 2	G	H	E	F	D	B	G	A
Set 3	C	F	B	G	E	F	B	B
Set 4	D	E	A	D	E	D	C	H
Set 5	E	F	A	C	E	F	A	H
Set 6	C	H	B	F	D	B	H	F

Each bootstrap subset has 8 examples.

Some examples may be duplicated, others left out.

Ensemble Generation: Bagging

- Bootstrap aggregation: Randomly sample from training data with replacement, apply a classifier to each sample.



➡ Encourages diversity in the ensemble, works better for “unstable” classifiers - e.g. decision trees, neural networks.

Algorithm 1 Bagging

1: Let n be the number of bootstrap samples
2:
3: **for** i=1 to n **do**
4: Draw bootstrap sample of size m, $\mathcal{D}_i$
5: Train base classifier h_i on $\mathcal{D}_i$
6: $\hat{y} = mode\{h_1(\mathbf{x}), ..., h_n(\mathbf{x})\}$

Bootstrap Sampling

Bootstrap Sampling

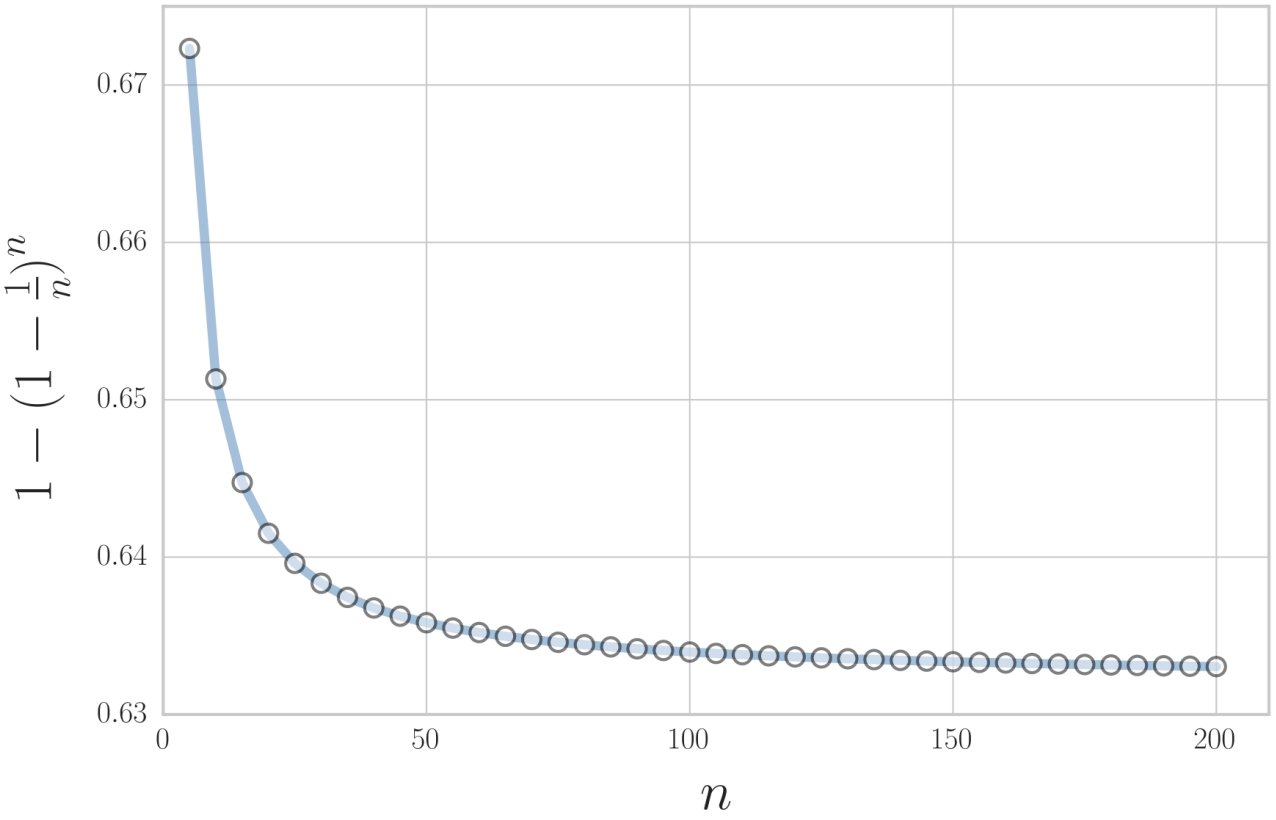
$$P(\textbf{not chosen}) = \left(1 - \frac{1}{n}\right)^n,$$

$$\frac{1}{e} \approx 0.368, \quad n \to \infty\,.$$

$$P(\textbf{not chosen}) = \left(1 - \frac{1}{n}\right)^n,$$

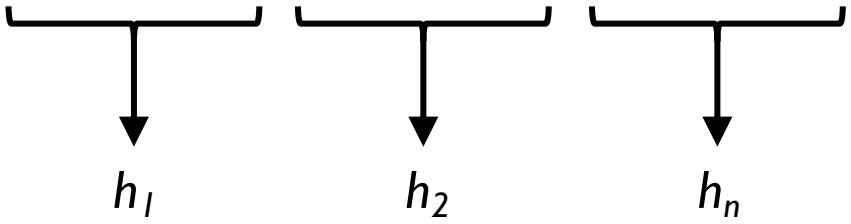
$$\frac{1}{e} \approx 0.368, \quad n \to \infty\,.$$

$$P(\textbf{chosen}) = 1 - \left(1 - \frac{1}{n}\right)^n \approx 0.632$$

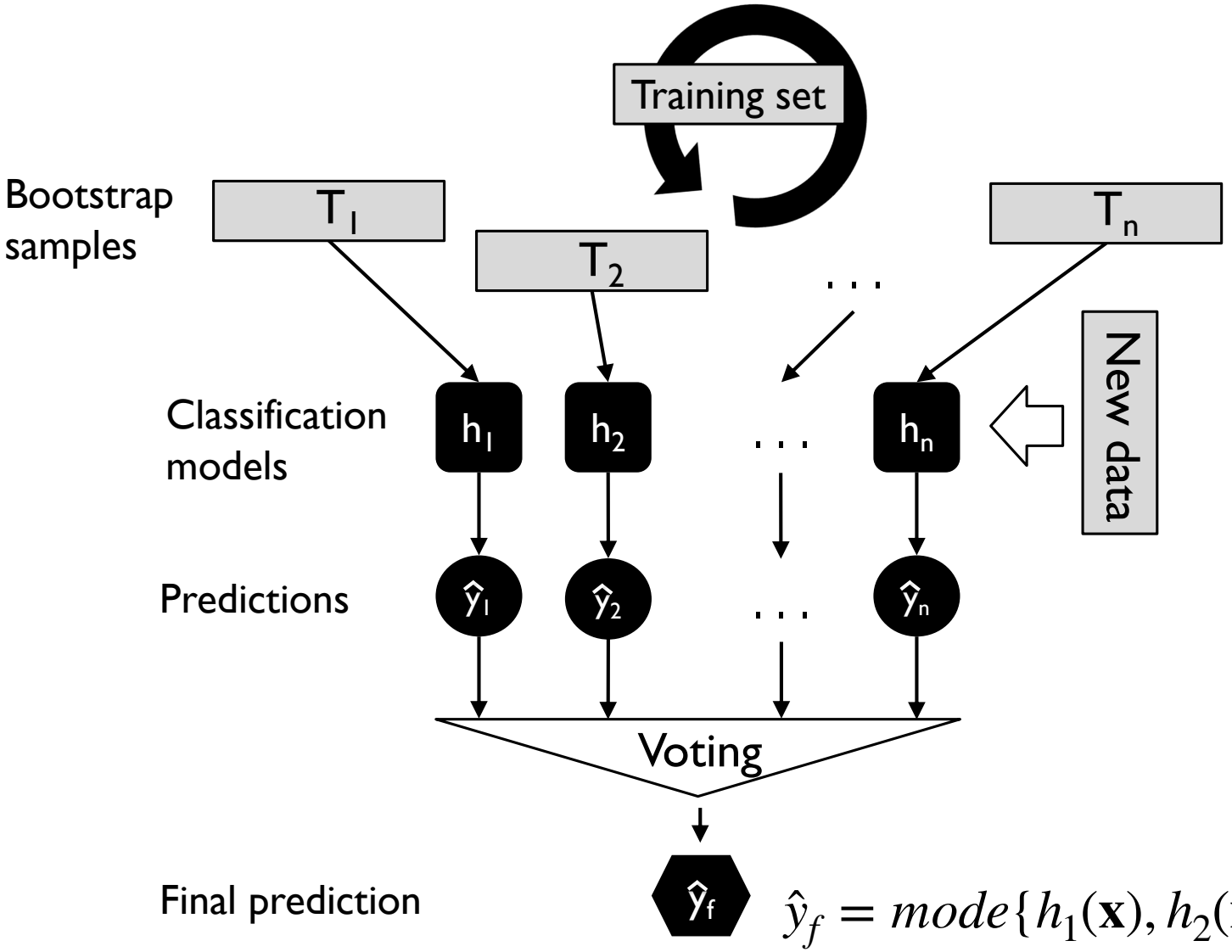


Bootstrap Sampling

Training example indices	Bagging round 1	Bagging round 2	…
1	2	7	…
2	2	3	…
3	1	2	…
4	3	1	…
5	7	1	…
6	2	7	…
7	4	7	…



Bagging Classifier



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

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

Bagging

(Random Features)

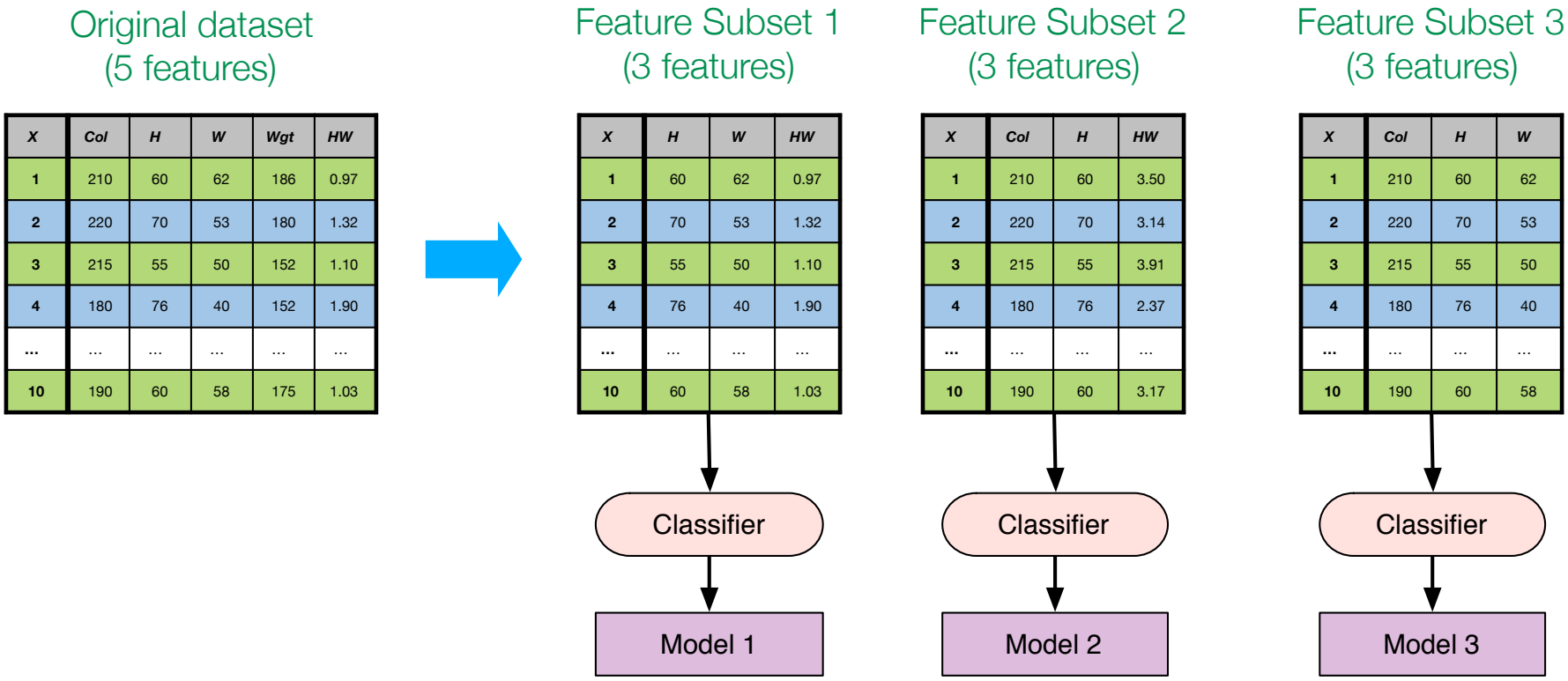
Well-known classifier - Random Forests

Random without replacement สุ่มแบบไม่ใส่คืนเข้าไป(column ไม่มีวันซ้ำ)

Ensemble Generation: Random Features

- **Key Idea:** Train *n* base classifiers, each on a different subset of features.
- Random Subspace Method:
  - A subset of features is randomly selected without replacement.
  - Train a classifier using only selected features to represent the training data.

➡ Encourages diversity in the ensemble, works well for *k*-NNs.



X	Col	H	W	Wgt	HW
1	210	60	62	186	0.97
2	220	70	53	180	1.32
3	215	55	50	152	1.10
4	180	76	40	152	1.90
...	...	...	...	...	...
10	190	60	58	175	1.03

X	H	W	HW
1	60	62	0.97
2	70	53	1.32
3	55	50	1.10
4	76	40	1.90
...	...	...	...
10	60	58	1.03

X	Col	H	HW
1	210	60	3.50
2	220	70	3.14
3	215	55	3.91
4	180	76	2.37
...	...	...	...
10	190	60	3.17

X	Col	H	W
1	210	60	62
2	220	70	53
3	215	55	50
4	180	76	40
...	...	...	...
10	190	60	58

Random Forests

= Bagging w. trees + random feature subsets

Bagging แบบ Random Feature โดยมี decision tree เป็น based

Random Feature Subset for each Tree or Node?

Tin Kam Ho used the "**random subspace method,**" where each tree got a random subset of features.

"Our method relies on an autonomous, pseudo-random procedure to select a small number of dimensions from a given feature space …"

- Ho, Tin Kam. "The random subspace method for constructing decision forests." IEEE transactions on pattern analysis and machine intelligence 20.8 (1998): 832-844.

"Trademark" random forest:

"… random forest with random features is formed by selecting at random, at each node, a small group of input variables to split on."

- **Breiman**, Leo. "Random Forests" Machine learning 45.1 (2001): 5-32.

Random Feature Subset for each Tree or Node?

Tin Kam Ho used the "random subspace method," where each tree got a random subset of features.

"Our method relies on an autonomous, pseudo-random procedure to select a small number of dimensions from a given feature space …"

- Ho, Tin Kam. "The random subspace method for constructing decision forests." IEEE transactions on pattern analysis and machine intelligence 20.8 (1998): 832-844.

"Trademark" random forest:

"… random forest with random feature … andom, at each node, a small group of input variab …

- Breiman, Leo. "Random Forests" Ma … 2.

num features $= \log_2 m + 1$

where *m* is the number of input features

In contrast to the original publication
[Breiman, “Random Forests”, Machine Learning, 45(1), 5-32, 2001]
the scikit-learn implementation combines classifiers by averaging their probabilistic prediction, instead of letting each classifier vote for a single class.

"Soft Voting"

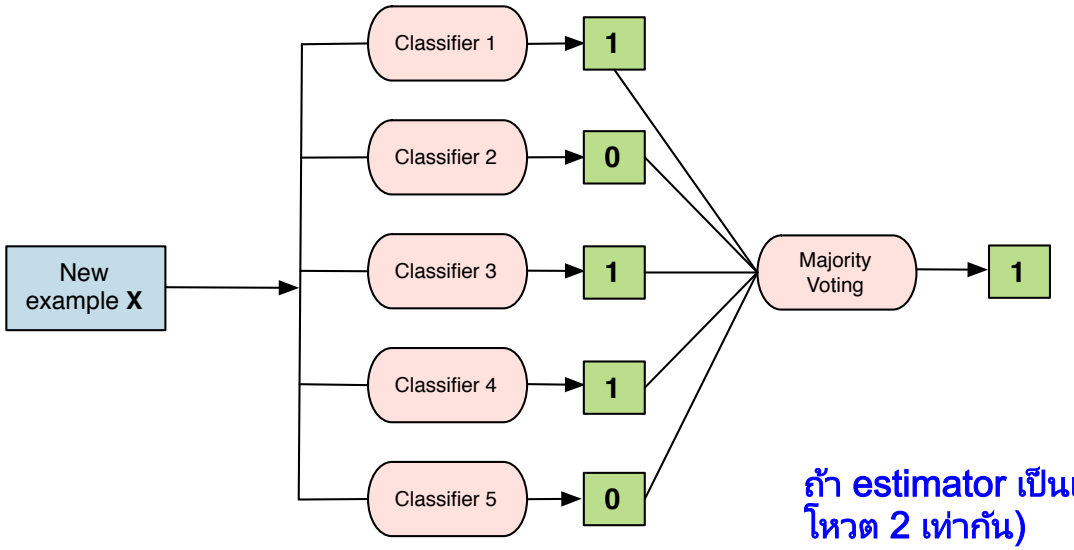
Ensemble Combination

(Majority Voting)

Ensemble Combination: Voting

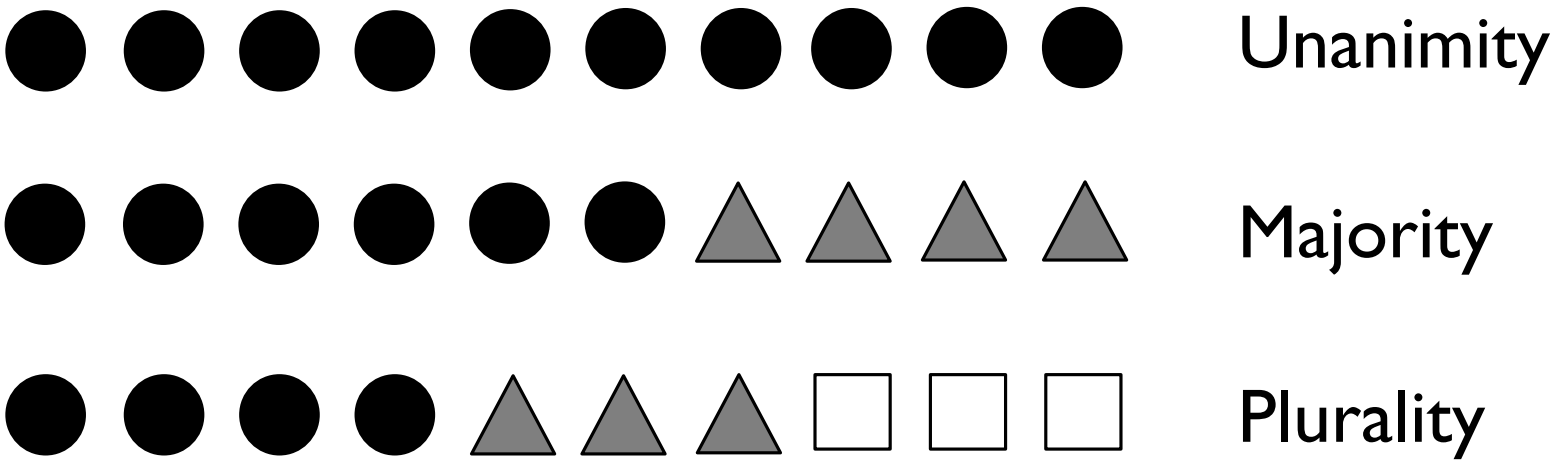
- Simplest way to combine the output of multiple classifiers is to use majority voting.
- All classifiers are run independently in parallel. Results are combined when all runs have completed.
- Each classifier “votes” for a particular class, where all classifiers carry equal weight. The class with the majority vote in the ensemble wins.

ส่วนใหญ่ predict เป็น label อะไรก็คำนวณว่าส่วนใหญ่เลือกอะไร ก็จะเอาอันนั้น

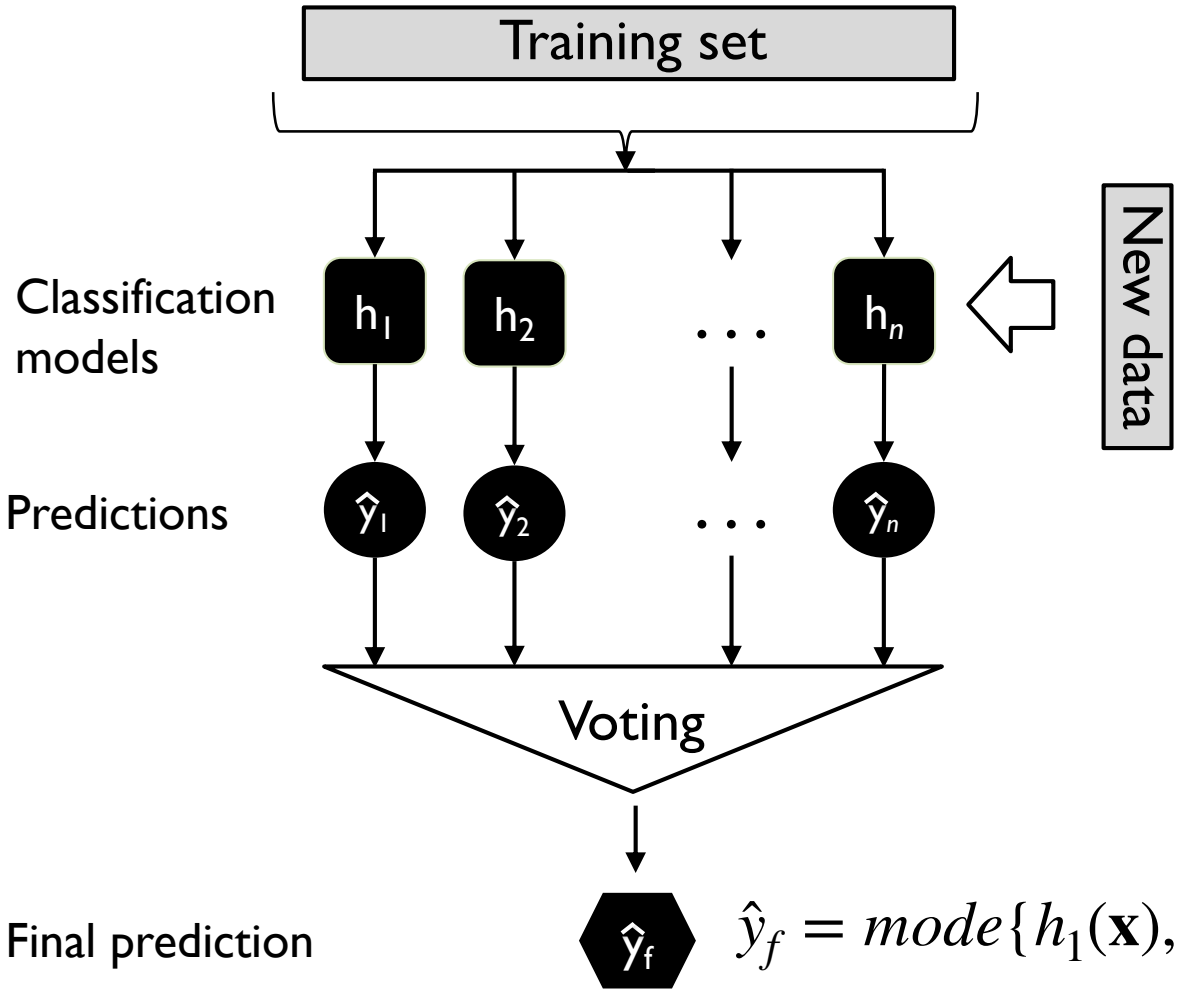


Ensemble votes for label “1” for **X** by 3:2

ถ้า estimator เป็นเลขคู่ อาจจะไม่ดีเท่าไหร่ (มี 4 ตัวโหวต 2 เท่ากัน)


Unanimity
Majority
Plurality

Majority Vote Classifier



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

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

Why Majority Vote?

- assume *n* independent classifiers with a base error rate ϵ
- here, independent means that the errors are uncorrelated
- assume a binary classification task
- assume the error rate is better than random guessing (i.e., lower than 0.5 for binary classification)

$$\forall \epsilon_i \in \{\epsilon_1, \epsilon_2, \ldots, \epsilon_n\}, \epsilon_i < 0.5$$

Why Majority Vote?

- assume *n* independent classifiers with a base error rate ϵ
- here, independent means that the errors are uncorrelated
- assume a binary classification task
- assume the error rate is better than random guessing (i.e., lower than 0.5 for binary classification)

$$\forall \epsilon_i \in \{\epsilon_1, \epsilon_2, \ldots, \epsilon_n\}, \epsilon_i < 0.5$$

The probability that we make a wrong prediction via the ensemble if *k* classifiers predict the same class label

$$P(k) = \binom{n}{k} \epsilon^k (1-\epsilon)^{n-k} \qquad k > \lceil n/2 \rceil$$

Why Majority Vote?

The probability that we make a wrong prediction via the ensemble if *k* classifiers predict the same class label

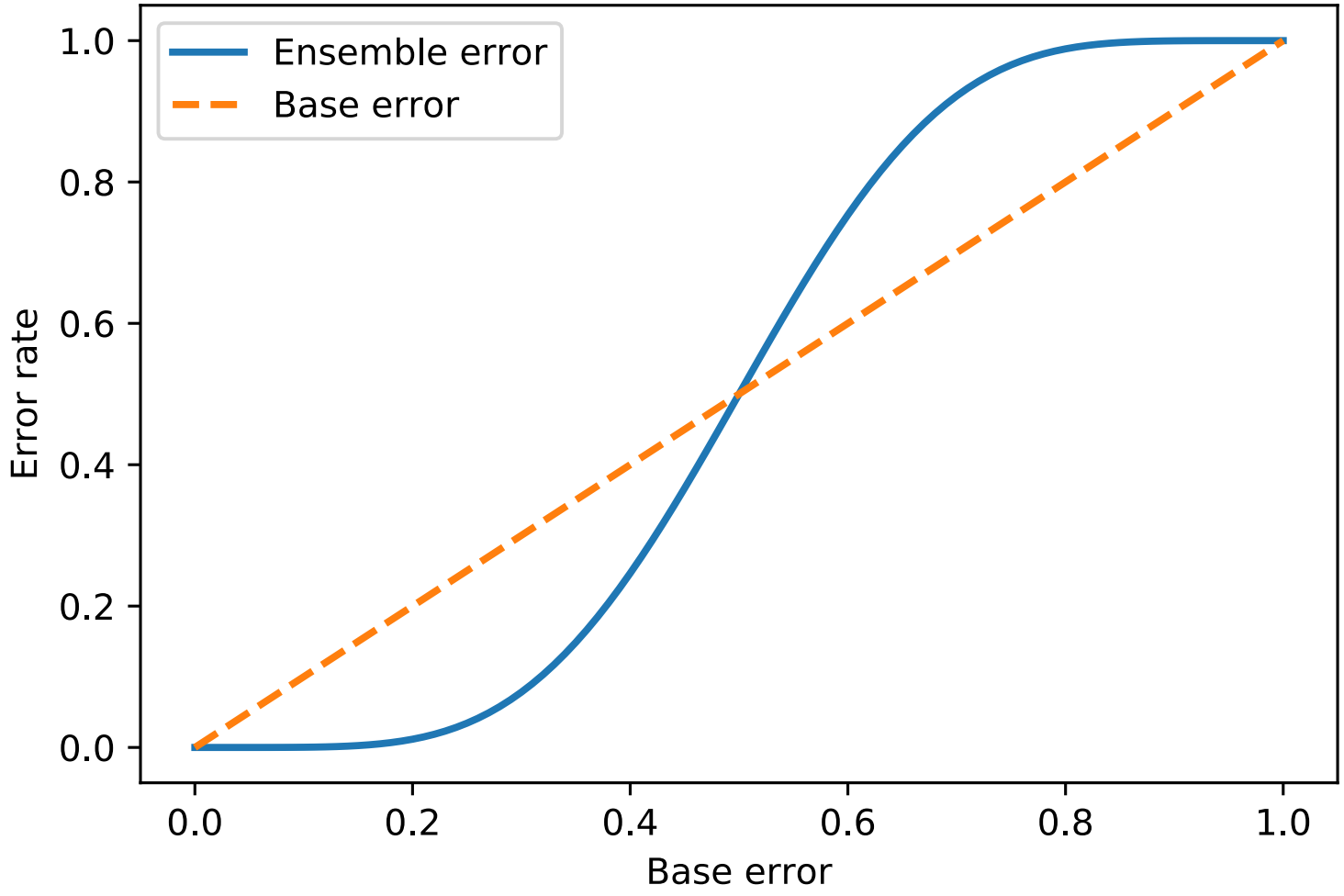
$$P(k) = \binom{n}{k} \epsilon^k (1-\epsilon)^{n-k} \qquad k > \lceil n/2 \rceil$$

Ensemble error:

$$\epsilon_{ens} = \sum_{k}^{n} \binom{n}{k} \epsilon^k (1-\epsilon)^{n-k}$$

$$\epsilon_{ens} = \sum_{k=6}^{11} \binom{11}{k} 0.25^k (1-0.25)^{11-k} = 0.034$$

$$\epsilon_{ens} = \sum_{k}^{n} \binom{n}{k} \epsilon^k (1-\epsilon)^{n-k}$$



Ensemble Combination

(Weighted/Soft Voting)

Ensemble Combination: Weighted Voting

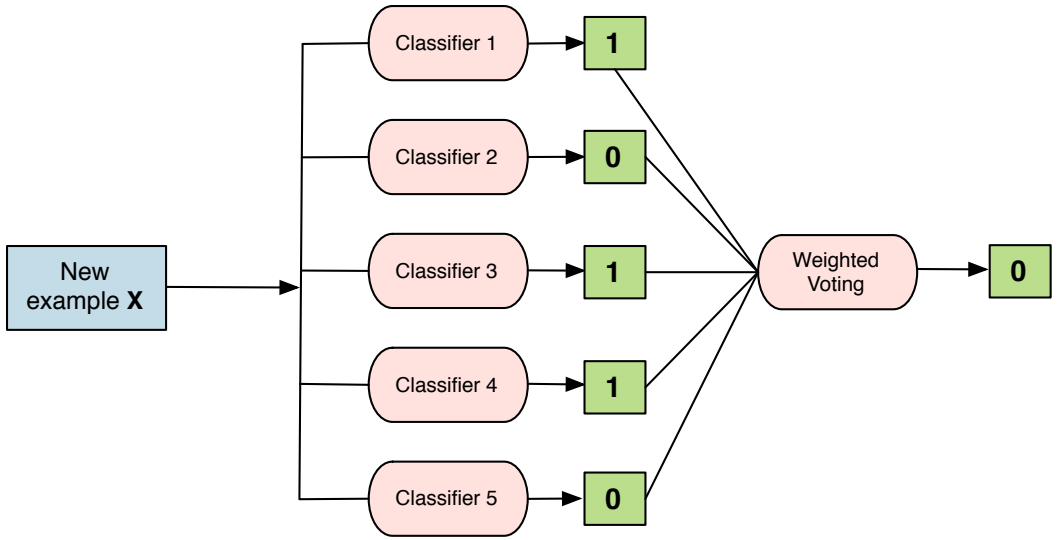
- **Intuition:** If individual classifiers do not give equal performance, we should give more power to better classifiers.
- Weighted Voting Combination:
  - Rather than treating every classifier's vote equally, we weight each classifier's vote based on its accuracy/error.

  ➡More accurate classifiers contribute more to the ensemble.

accuracy เยอะ ก็ weight เยอะ (มาจากการ normalize)

Classifier	Accuracy	Weight
1	0.52	0.14
2	1.00	0.28
3	0.57	0.16
4	0.55	0.15
5	0.95	0.26
TOTAL	**3.59**	**1.00**

e.g. C1: 0.52/3.59 = 0.14



Vote "0": 0.28 + 0.26 ≅ 0.54

Vote "1": 0.14 + 0.16 + 0.15 ≅ 0.46

"Soft" Voting

Pij คือผลการทำนาย

$$\hat{y} = \arg\max_{j} \sum_{i=1}^{n} w_i p_{i,j}$$

$p_{i,j}$: predicted class membership probability of the *i*th classifier for class label *j*

w_j: optional weighting parameter, default $w_i = 1/n, \forall w_i \in \{w_1, \ldots, w_n\}$

"Soft" Voting

Use only for well-calibrated classifiers!

$$\hat{y} = \arg\max_{j} \sum_{i=1}^{n} w_i p_{i,j}$$

$p_{i,j}$: predicted class membership probability of the *i*th classifier for class label *j*

w_j: optional weighting parameter, default $w_i = 1/n, \forall w_i \in \{w_1, \ldots, w_n\}$

"Soft" Voting

$$\hat{y} = \arg\max_{j} \sum_{i=1}^{n} w_i p_{i,j}$$

Binary classification example

$j \in \{0,1\}$ $h_i(i \in \{1,2,3\})$

$$h_1(\mathbf{x}) \rightarrow [0.9,0.1]$$
$$h_2(\mathbf{x}) \rightarrow [0.8,0.2]$$
$$h_3(\mathbf{x}) \rightarrow [0.4,0.6]$$

"Soft" Voting

$$\hat{y} = \arg\max_{j} \sum_{i=1}^{n} w_i p_{i,j}$$

Binary classification example

$$j \in \{0,1\} \qquad h_i(i \in \{1,2,3\})$$

$$h_1(\mathbf{x}) \rightarrow [0.9,0.1]$$
$$h_2(\mathbf{x}) \rightarrow [0.8,0.2]$$
$$h_3(\mathbf{x}) \rightarrow [0.4,0.6]$$

$$p(j = 0 \,|\, \mathbf{x}) = 0.2 \cdot 0.9 + 0.2 \cdot 0.8 + 0.6 \cdot 0.4 = 0.58$$
$$p(j = 1 \,|\, \mathbf{x}) = 0.2 \cdot 0.1 + 0.2 \cdot 0.2 + 0.6 \cdot 0.6 = 0.42$$

$$\hat{y} = \arg\max_{j} \left\{ p(j = 0 \,|\, \mathbf{x}), p(j = 1 \,|\, \mathbf{x}) \right\}$$

Committees of Experts

- **Consider:** *" … a medical school that has the objective that all students, given a problem, come up with an identical solution"*.
- No value in a committee of experts from such a group - the committee will not improve on the judgement of an individual.
- There needs to be disagreement for the committee to have the potential to be better than an individual.
- Fundamental work by Krogh & Vedelsby (1995) for regression:
  - Increasing "ambiguity" (disagreement) decreases overall combined error, provided it does not result in an increase of average error.

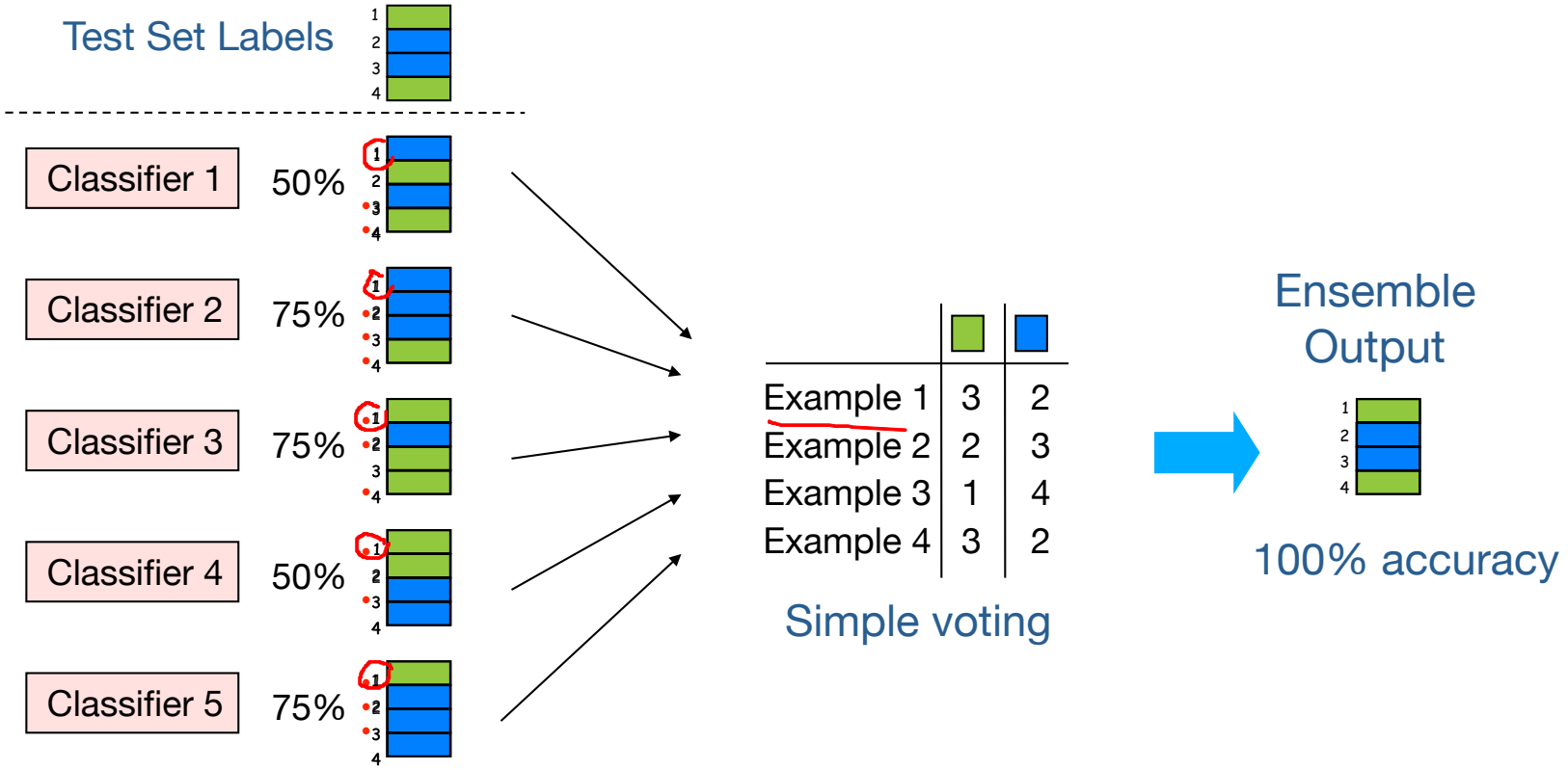
$$\overline{E} - \overline{A} = E$$

Average error of classifiers — Ensemble ambiguity — Ensemble error

We need accuracy + diversity in classifier ensembles!

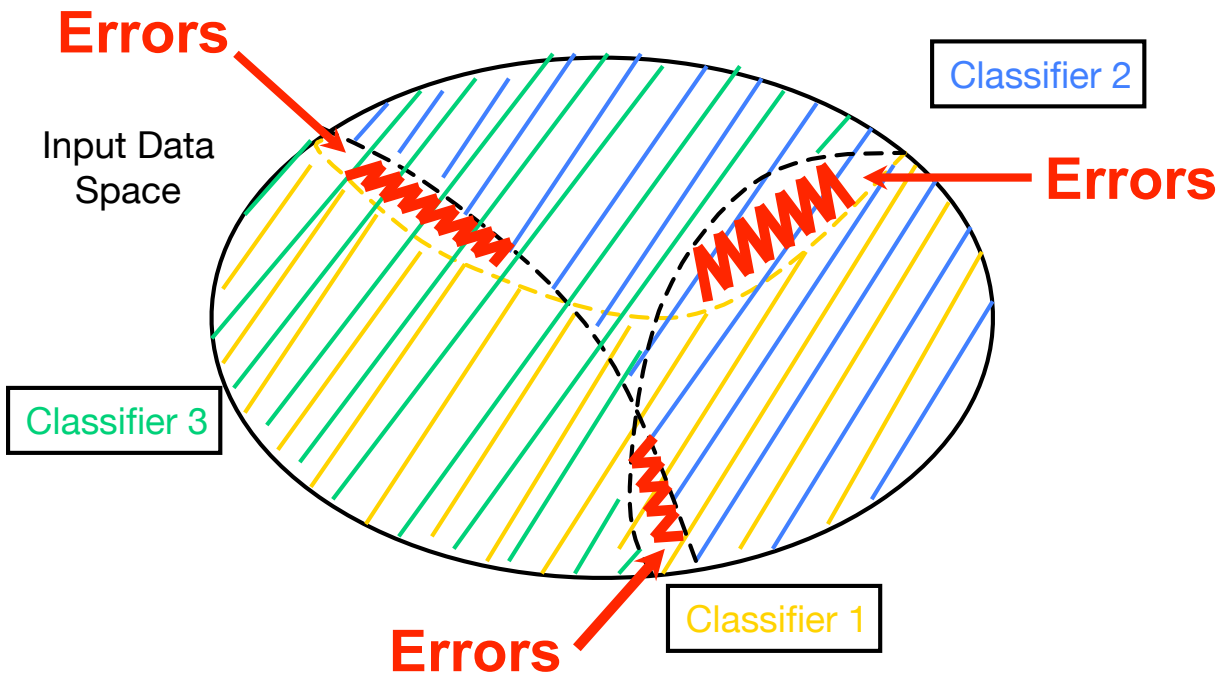
Ensemble Diversity

- **Local Learning in Ensembles**
  - Every single classifier performs well on a subset of the test set.
  - The mistakes that one classifier makes are “corrected” by the other classifiers.



Ensemble Specialisation

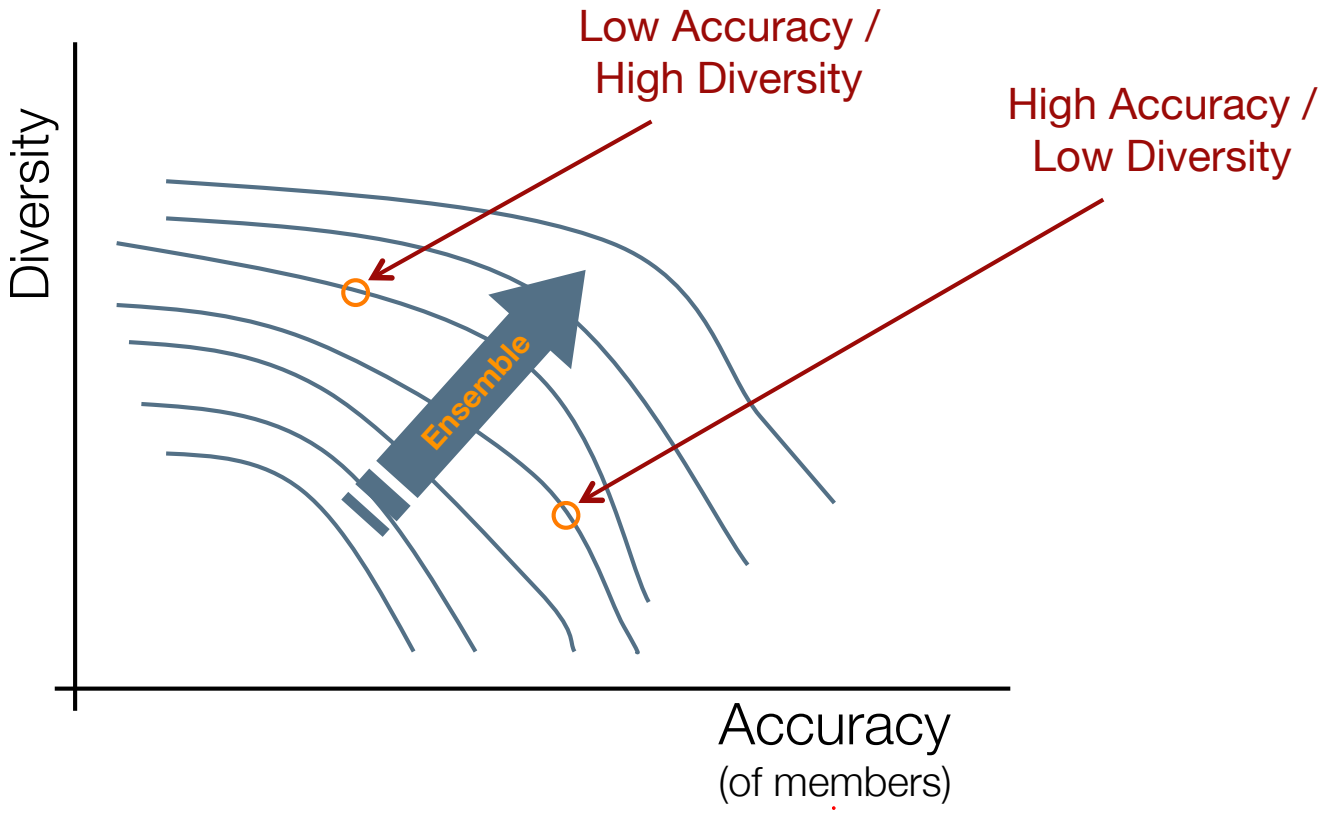
- Classifiers can specialise on accurately classifying only related examples from certain regions of the input data space.
- **Example:** Visualisation of specialisation in classifier ensembles…



➡ Want ensemble members to make mistakes in different areas.

Ensemble Diversity

- **Recall:** Krogh & Vedelsby said an ideal ensemble is one that consists of highly accurate members which at the same time disagree.
- Often face a trade-off between diversity and accuracy when constructing an ensemble of classifiers.



Boosting

Adaptive Boosting

e.g., AdaBoost (here!)

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, *55*(1), 119-139.

Gradient Boosting

e.g., XGBoost

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189-1232.

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International conference on knowledge discovery and data mining* (pp. 785-794). ACM.

Adaptive Boosting

e.g., AdaBoost (here!)

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, *55*(1), 119-139.

Differ mainly in terms of how
- weights are updated
- classifiers are combined

Gradient Boosting

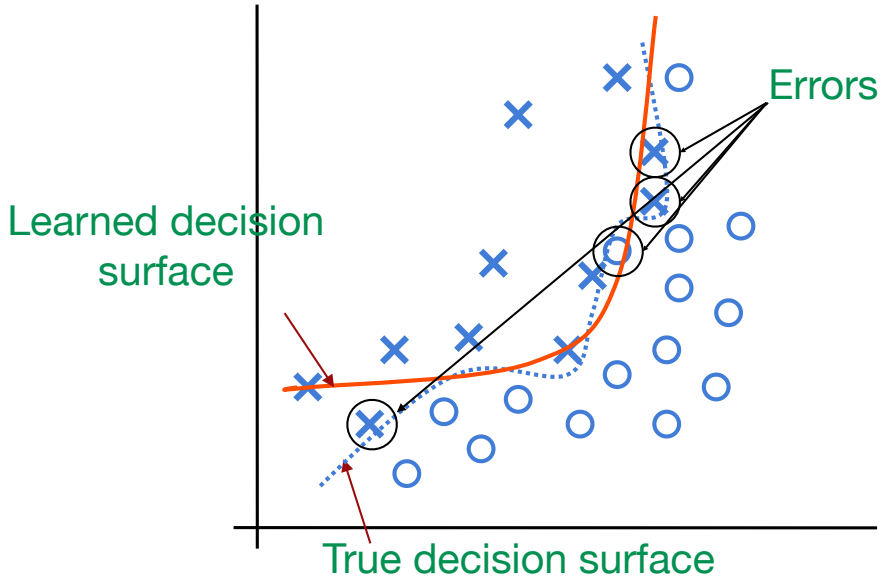
e.g., XGBoost

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189-1232.

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International conference on knowledge discovery and data mining* (pp. 785-794). ACM.

Ensemble Generation: Boosting

- **Key idea:** Train a series of classifiers such that later classifiers are trained to better predict on examples that earlier ones perform poorly on.
- Focus on previous errors when building next ensemble member.



Boosting Approach:

1. Weight all training examples equally.
2. FOR *i = 1 to T*
   (a) Train classifier using current weights.
   (b) Compute errors.
   (c) Increase weights for misclassified examples, decrease weights for those classified correctly.
3. Output final model.

ดูว่าตัวไหนที่มักจะทำนายผิด ก็จะเอาตัวนั้นไปใส่ training set ให้ทำนายแม่นขึ้น (เรียนรู้จากความผิดพลาด)

Example: Boosting

- **Problem:** Apply a classifier to a training set with 8 examples *{A, B, C, D, E, F, G, H}*, where example *A* is an outlier and difficult to classify.

- Selected training sets for 4 runs of bagging - i.e. simple random sampling with replacement.

➡ All examples equally weighted.

Original	A	B	C	D	E	F	G	H
Set 1	B	G	H	C	G	F	C	A
Set 2	G	H	E	F	D	B	G	A
Set 3	C	F	B	G	E	F	B	B
Set 4	D	E	A	D	E	D	C	H

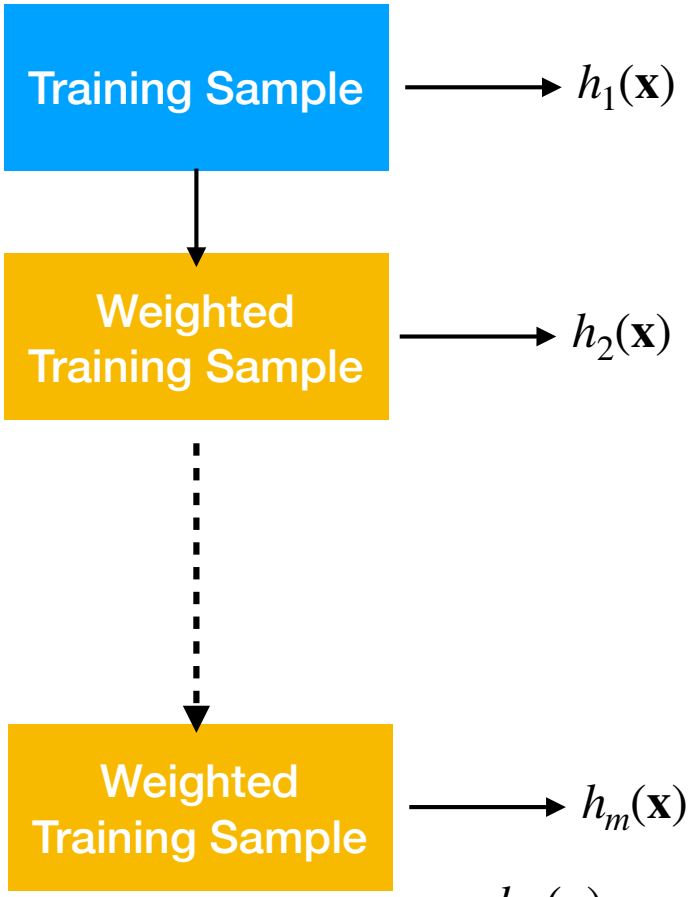
- Selected training sets for 4 runs of boosting - i.e. increase weights for misclassified examples.

➡ The "hard" example *A* appears more frequently in later sets.

Original	A	B	C	D	E	F	G	H
Set 1	B	G	H	C	G	F	C	A
Set 2	A	D	E	D	A	E	F	D
Set 3	G	A	E	H	A	H	A	D
Set 4	A	A	F	A	A	C	A	E

D. Opitz & R. Maclin. "Popular Ensemble Methods: An Empirical Study" (1999)

General Boosting



$$h_m(\mathbf{x}) = sign\left(\sum_{j=1}^{m} w_j\, h_j(\mathbf{x})\right) \quad \text{for} \quad h(\mathbf{x}) \in \{-1,1\}$$

$$\text{or} \quad h_m(\mathbf{x}) = \arg\max_i \left(\sum_{j=1}^{m} w_j\, \mathbf{1}[h_j(\mathbf{x}) = i]\right) \quad \text{for} \quad h(\mathbf{x}) = i, \qquad i \in \{1,\ldots,n\}$$

General Boosting

- Initialize a weight vector with uniform weights
- 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
- (Weighted) majority voting on trained classifiers

* a learner slightly better than random guessing

AdaBoost

Algorithm 1 AdaBoost

Initialize k: the number of AdaBoost rounds
Initialize $\mathcal{D}$: the training dataset, $\mathcal{D} = \{\langle \mathbf{x}^{[1]}, y^{[1]} \rangle, ..., \mathbf{x}^{[n]}, y^{[n]} \rangle\}$
Initialize $w_1(i) = 1/n, \quad i = 1, ..., n, \ \mathbf{w}_1 \in \mathbb{R}^n$

for r=1 to k **do**
    For all $i : \mathbf{w}_r(i) := w_r(i) / \sum_i w_r(i)$ [normalize weights]
    $h_r := FitWeakLearner(\mathcal{D}, \mathbf{w}_r)$
    $\epsilon_r := \sum_i w_r(i) \ \mathbf{1}(h_r(i) \neq y_i)$ [compute error]
    if $\epsilon_r > 1/2$ then stop
    $\alpha_r := \frac{1}{2} \log[(1 - \epsilon_r)/\epsilon_r]$ [small if error is large and vice versa]
    $w_{r+1}(i) := w_r(i) \times \begin{cases} \mathrm{e}^{-\alpha_r} & \text{if } h_r(\mathbf{x}^{[i]}) = y^{[i]} \\ \mathrm{e}^{\alpha_r} & \text{if } h_r(\mathbf{x}^{[i]}) \neq y^{[i]} \end{cases}$

Predict: $h_k(\mathbf{x}) = \arg\max_j \sum_r \alpha_r \mathbf{1}[h_r(\mathbf{x}) = j]$

Modify AdaBoost w. Bootstrap

Algorithm 1 AdaBoost

Initialize k: the number of AdaBoost rounds
Initialize $\mathcal{D}$: the training dataset, $\mathcal{D} = \{\langle \mathbf{x}^{[1]}, y^{[1]} \rangle, ..., \mathbf{x}^{[n]}, y^{[n]} \rangle\}$
Initialize $w_1(i) = 1/n, \quad i = 1, ..., n, \ \mathbf{w}_1 \in \mathbb{R}^n$

for r=1 to k **do**
    For all $i : \mathbf{w}_r(i) := w_r(i) / \sum_i w_r(i)$ [normalize weights]
    $h_r := FitWeakLearner(\mathcal{D}, \mathbf{w}_r)$
    $\epsilon_r := \sum_i w_r(i) \, \mathbf{1}(h_r(i) \neq y_i)$ [compute error]
    if $\epsilon_r > 1/2$ then stop
    $\alpha_r := \frac{1}{2} \log[(1 - \epsilon_r)/\epsilon_r]$ [small if error is large and vice versa]
    $w_{r+1}(i) := w_r(i) \times \begin{cases} e^{-\alpha_r} & \text{if } h_r(\mathbf{x}^{[i]}) = y^{[i]} \\ e^{\alpha_r} & \text{if } h_r(\mathbf{x}^{[i]}) \neq y^{[i]} \end{cases}$
Predict: $h_k(\mathbf{x}) = \arg\max_j \sum_r \alpha_r \mathbf{1}[h_r(\mathbf{x}) = j]$

Summary

- Ensemble Classification
- Why do ensembles work?
  - Condorcet Jury Theorem
- Ensemble Generation
  - Bagging v Boosting
- Ensemble Combination
  - Voting v Weighted Voting

References

- D. Opitz and R. Maclin. "Popular Ensemble Methods: An Empirical Study" (1999). Journal of Artificial Intelligence Research.
- Breiman, L., (1996) “Bagging predictors”. Machine Learning, 24:123-140.
- Krogh, A., Vedelsby, J., (1995) “Neural Network Ensembles, Cross Validation and Active Learning”, in Advances in Neural Information Processing Systems 7
- Baur, E., and Kohavi, R. (1999) “An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants”, Machine Learning.