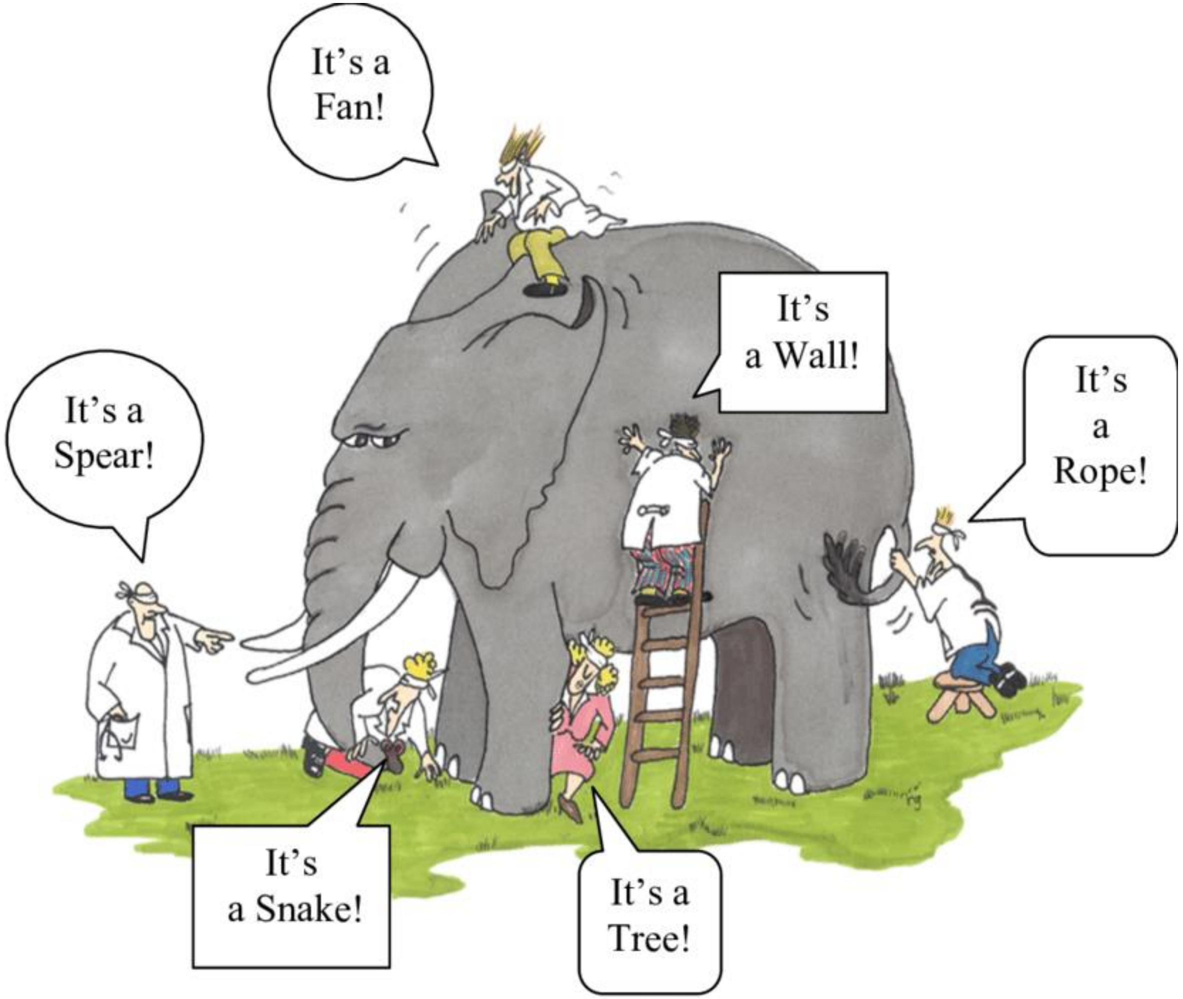
ENSEMBLE LEARNING

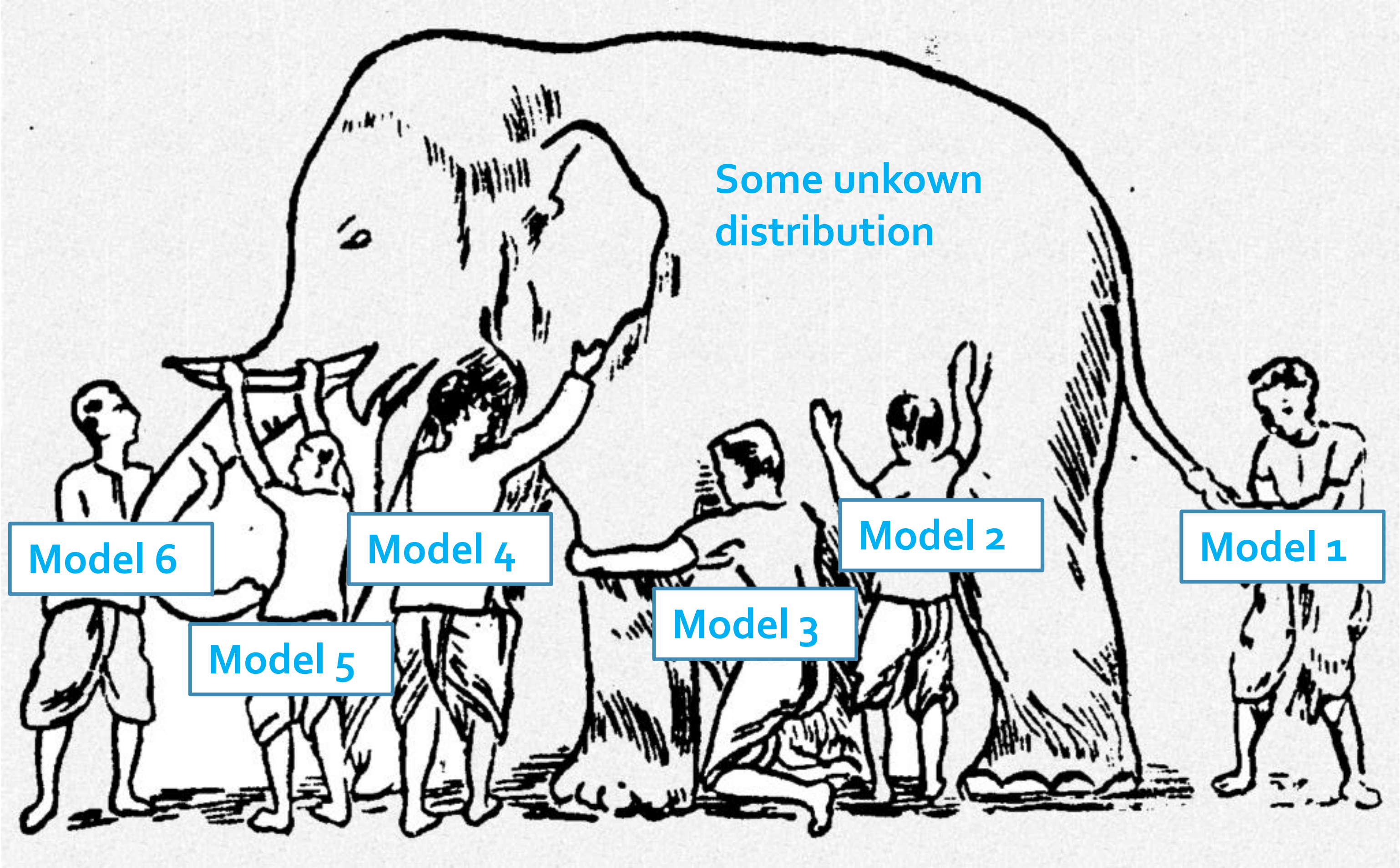
The science of combining models

Wisdom of the Crowd



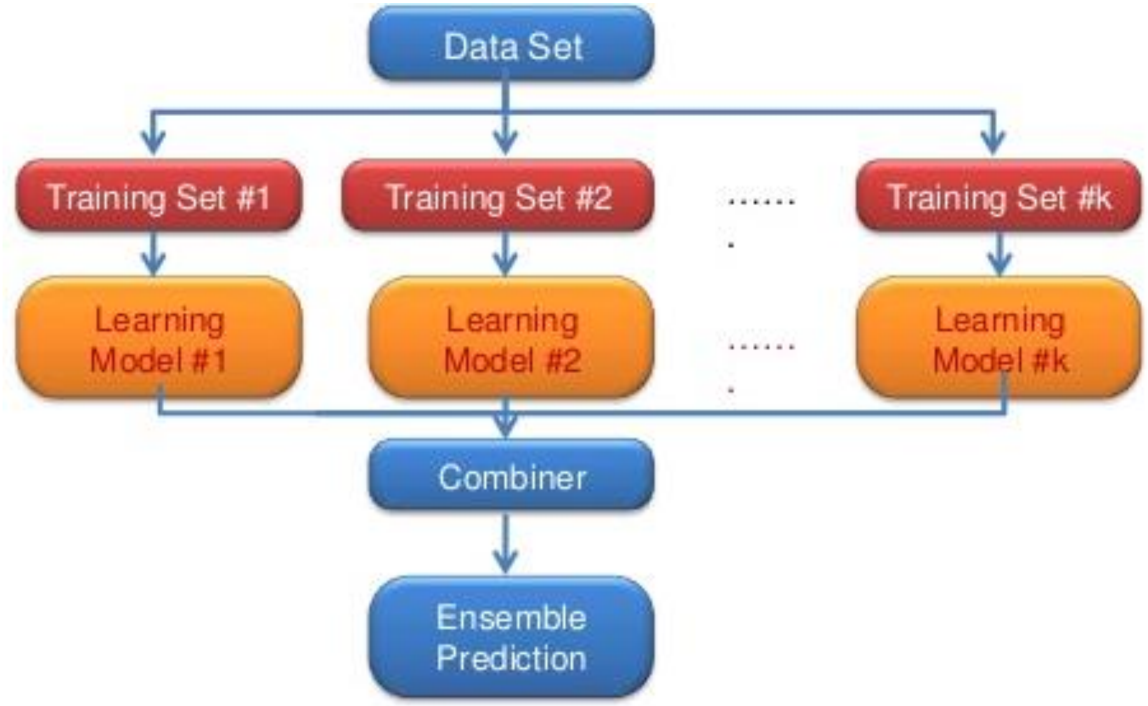
Wisdom of the Crowd (of machines)

- The wisdom of the crowd
  - A diverse set of models are likely to make better decisions as compared to single models
- Combining decisions from multiple models to improve the overall performance.



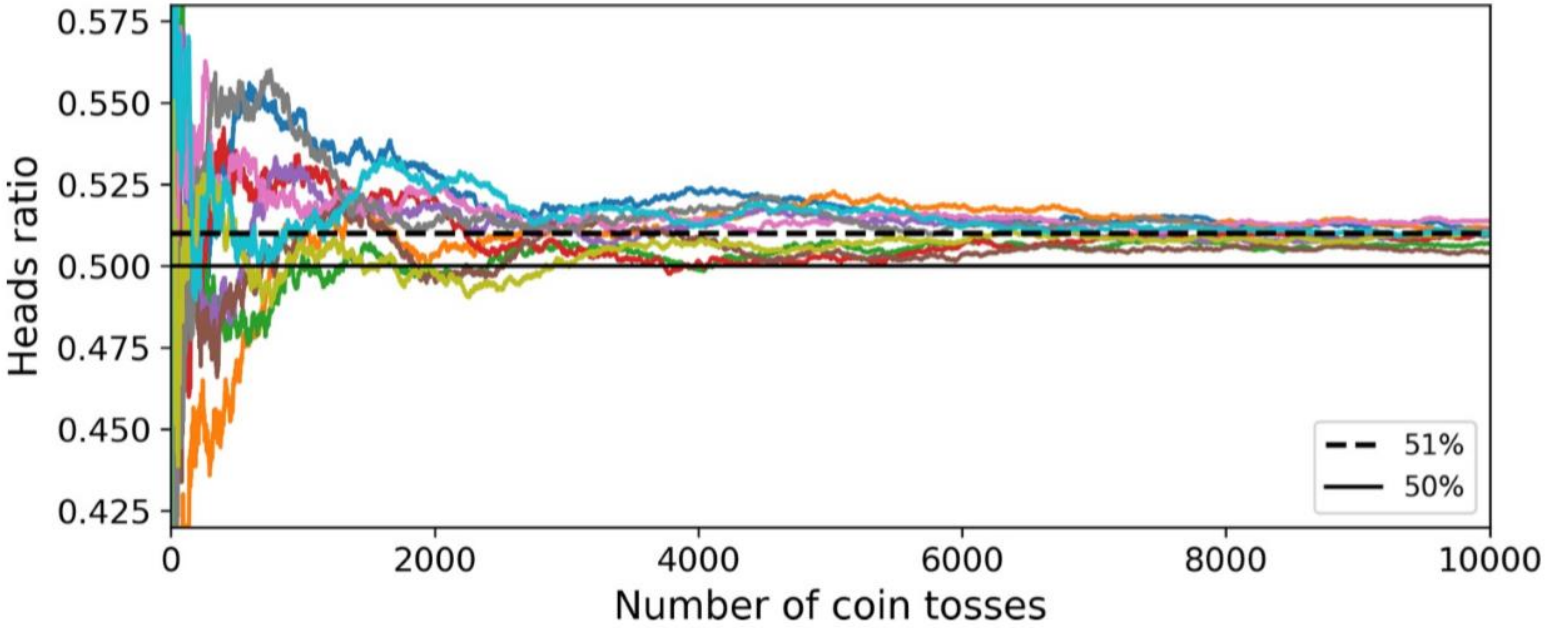
What is ensemble learning?

- An ensemble is a group of predictors
- An ensemble can be a *strong learner* even if each predictor is a weak learner
  - Provided there are a sufficient number of *weak learners* and they are sufficiently diverse



How will combining leads to a strong learner?

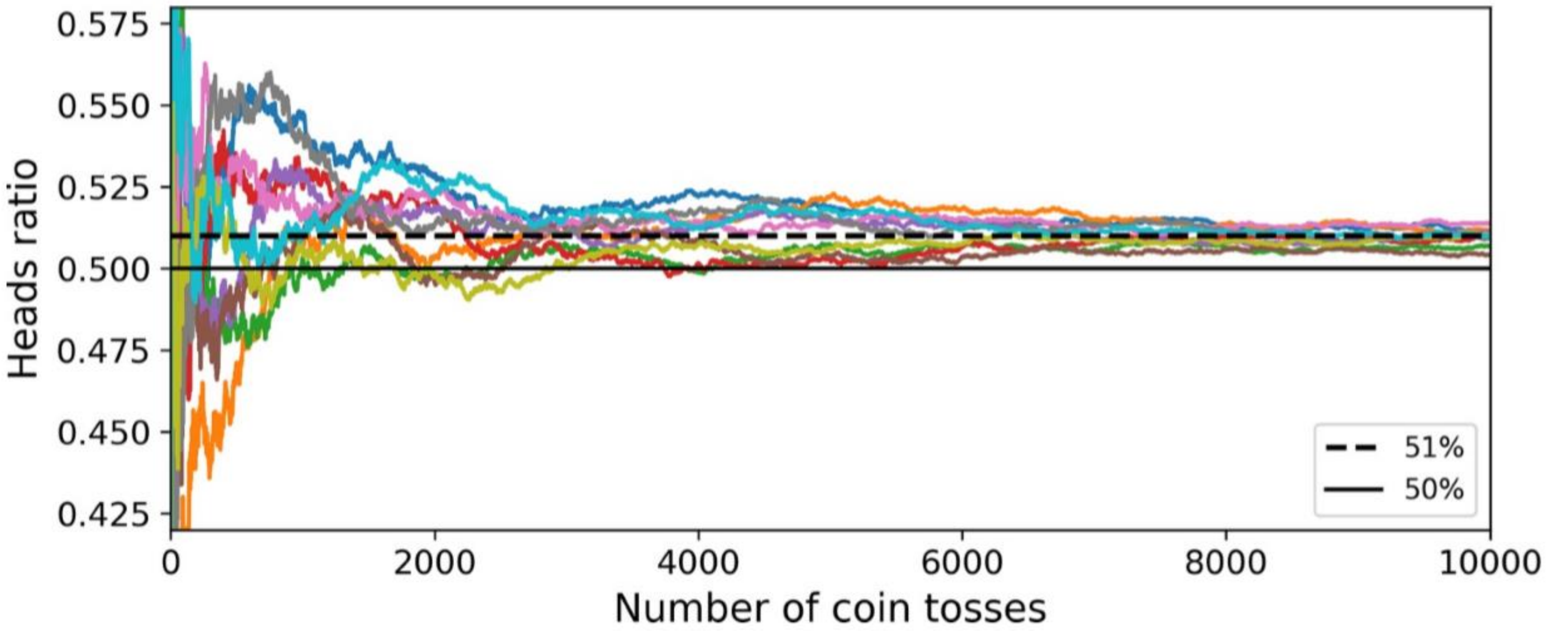
- Think of a slightly biased coin with 51% chance of heads and 49% of tails.
- **Law of large numbers**: as you keep tossing the coin, assuming every toss is independent of others, the ratio of heads gets closer and closer to the probability of heads 51%.



How will combining leads to a strong learner?

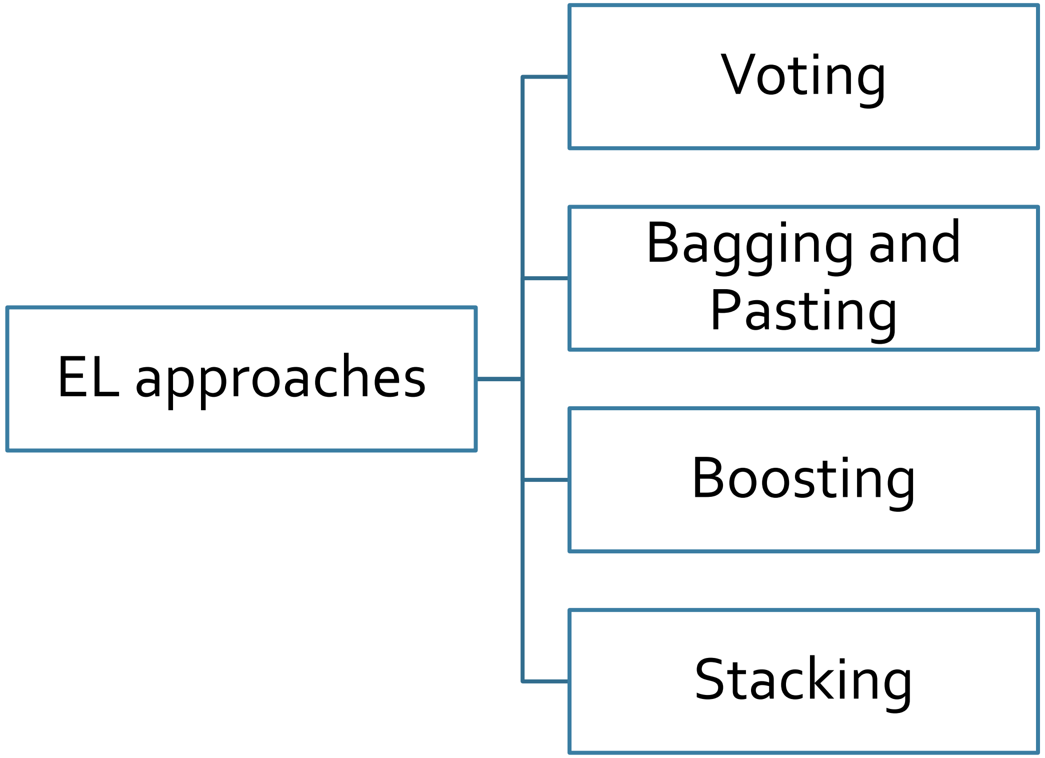
- Tossing the coin 1000 times, we will end with more or less than 510 heads and 490 tails.

- For an ensemble of 1000 classifiers, each correct 51% of the time, the probability of getting the majority of heads is up 75%



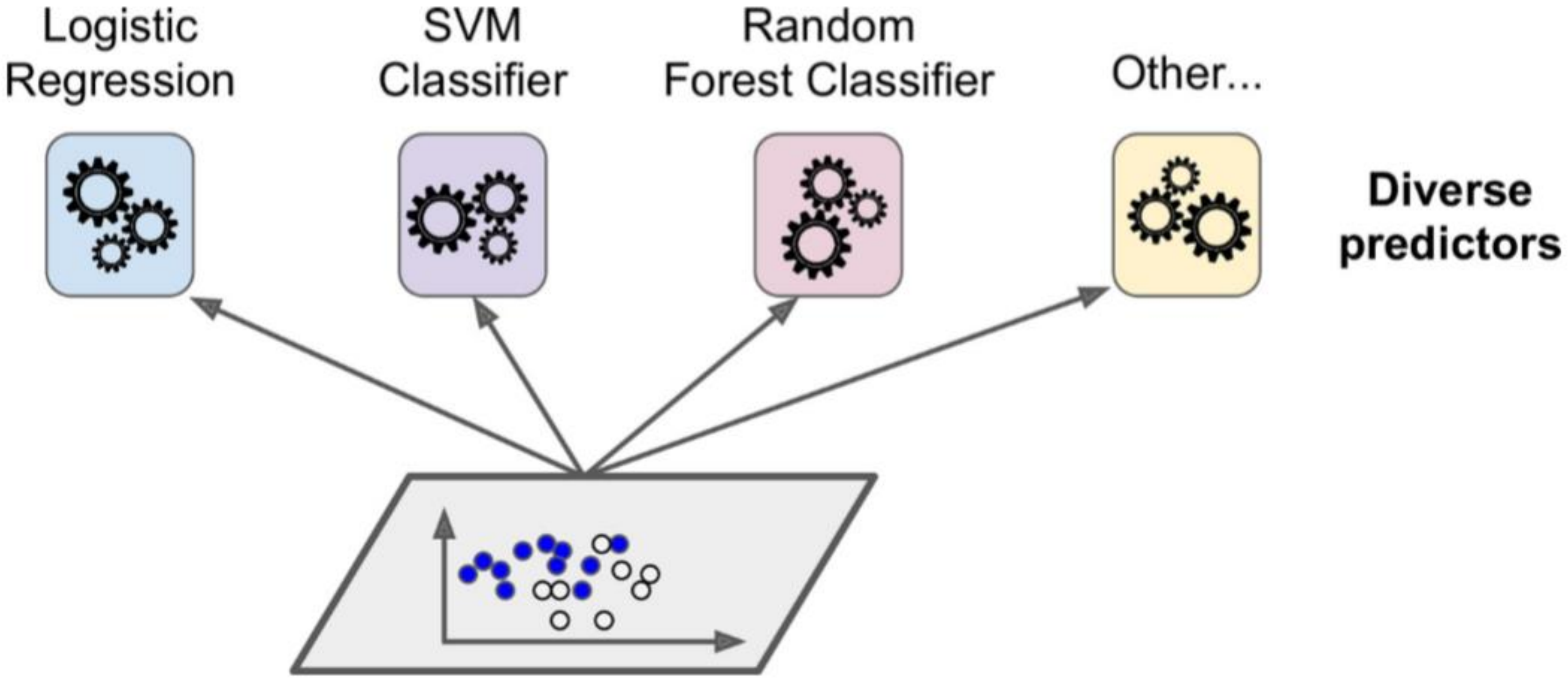
Ensemble Learning

- Different types of ensembles
  - Use the same/different learning algorithms
    - Homogeneous vs heterogeneous ensembles
  - Use the same dataset/ random subsets of data
  - Use the same/different sets of features



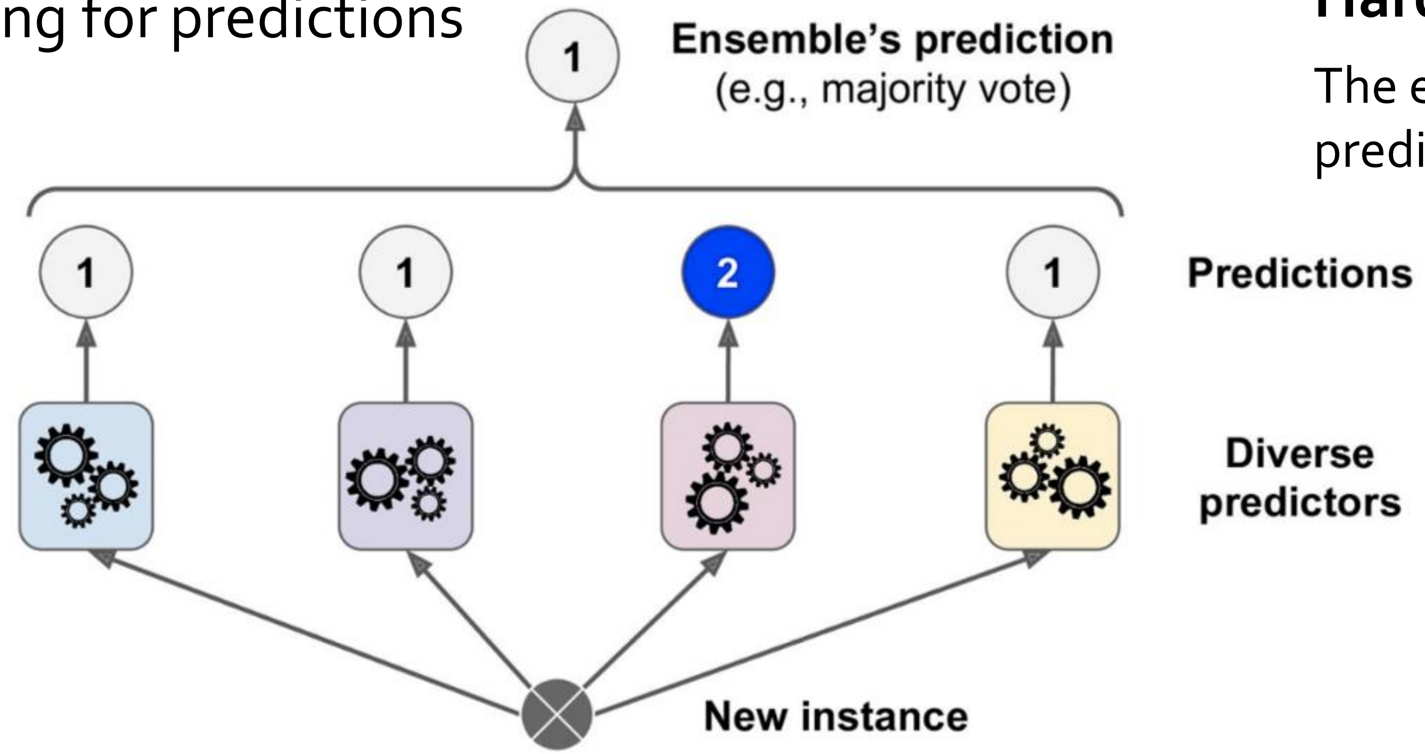
Voting Classifier

- Train diverse predictors on the same data



Voting Classifier

- Use voting for predictions



Hard voting:

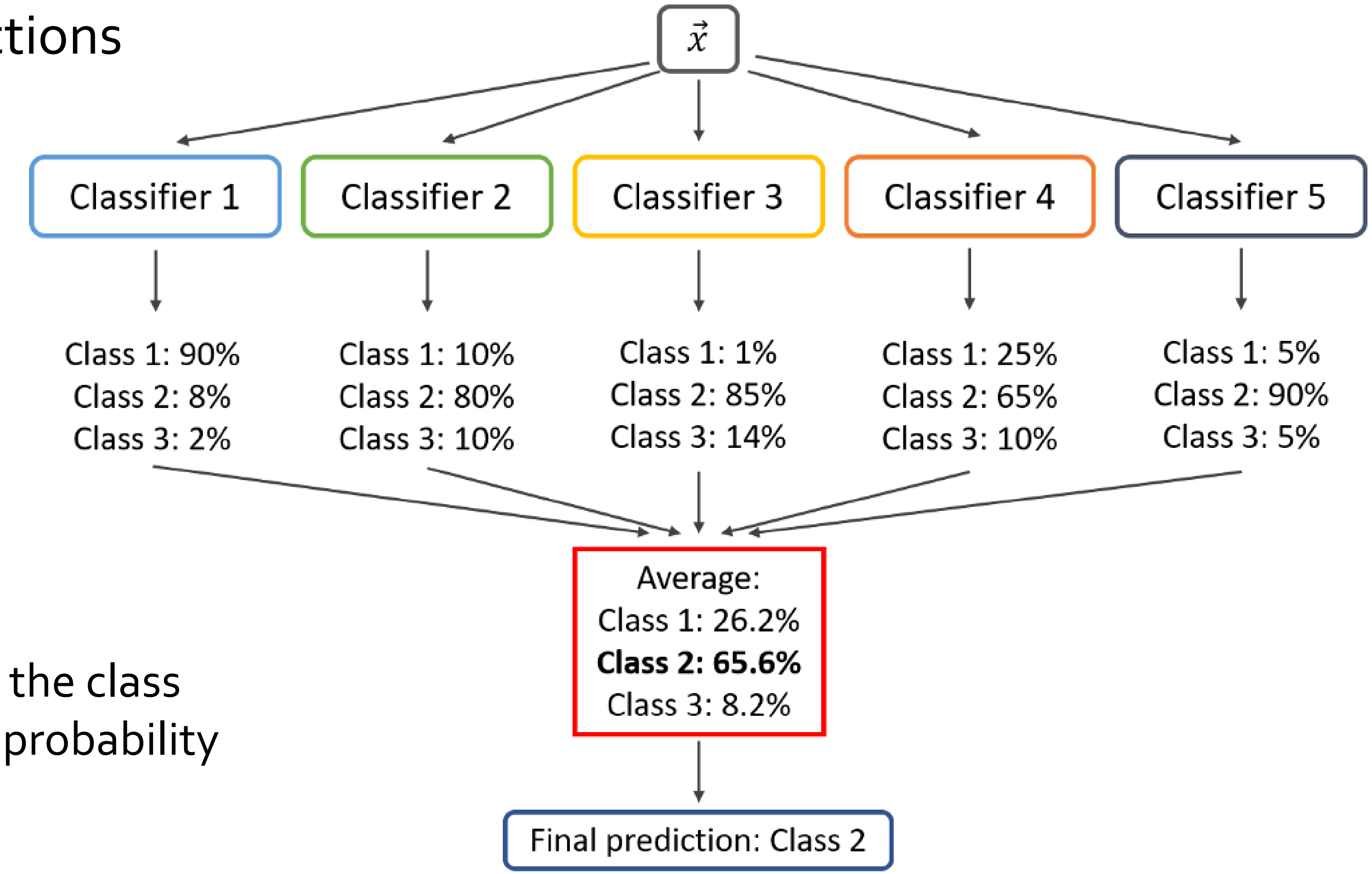
The ensemble prediction is the prediction of the majority

Example

Predictor 1	Predictor 2	Predictor 3	Predictor 4	Predictor 5	Ensemble's prediction
5	4	5	4	4	4

Voting Classifier

- Use voting for predictions

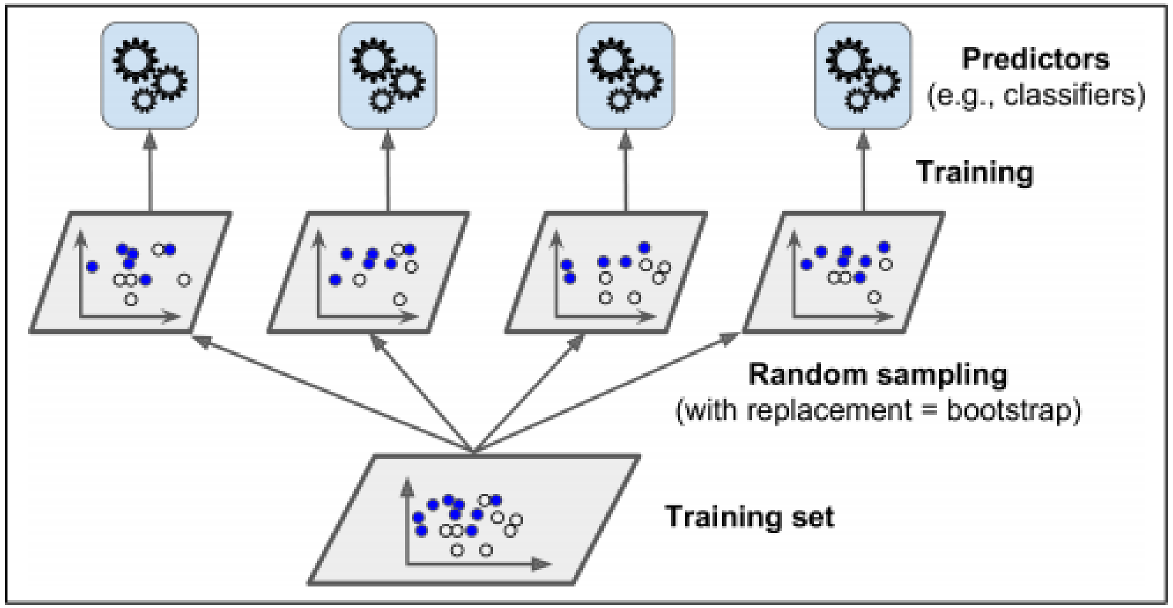


Soft voting:

The ensemble prediction is the class with the highest averaged probability averaged

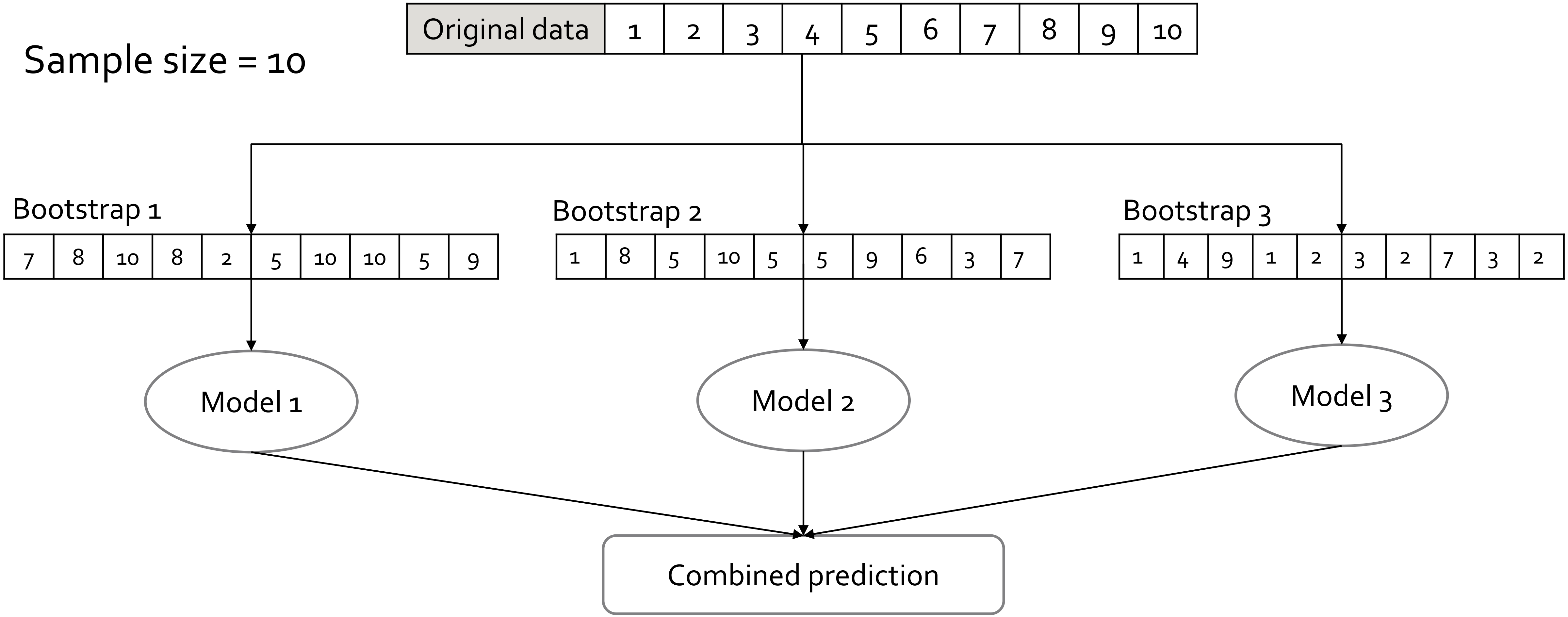
Bagging and Pasting

- Use different random subset of samples
  - Usually predictors of the same type are used
- **Bagging**: sampling with replacement
  - For a given predictor, a training instance may be sampled several times
- **Pasting**: sampling without replacement
  - Training instances may be sampled several times across predictors
- Training and predictions can be performed in parallel



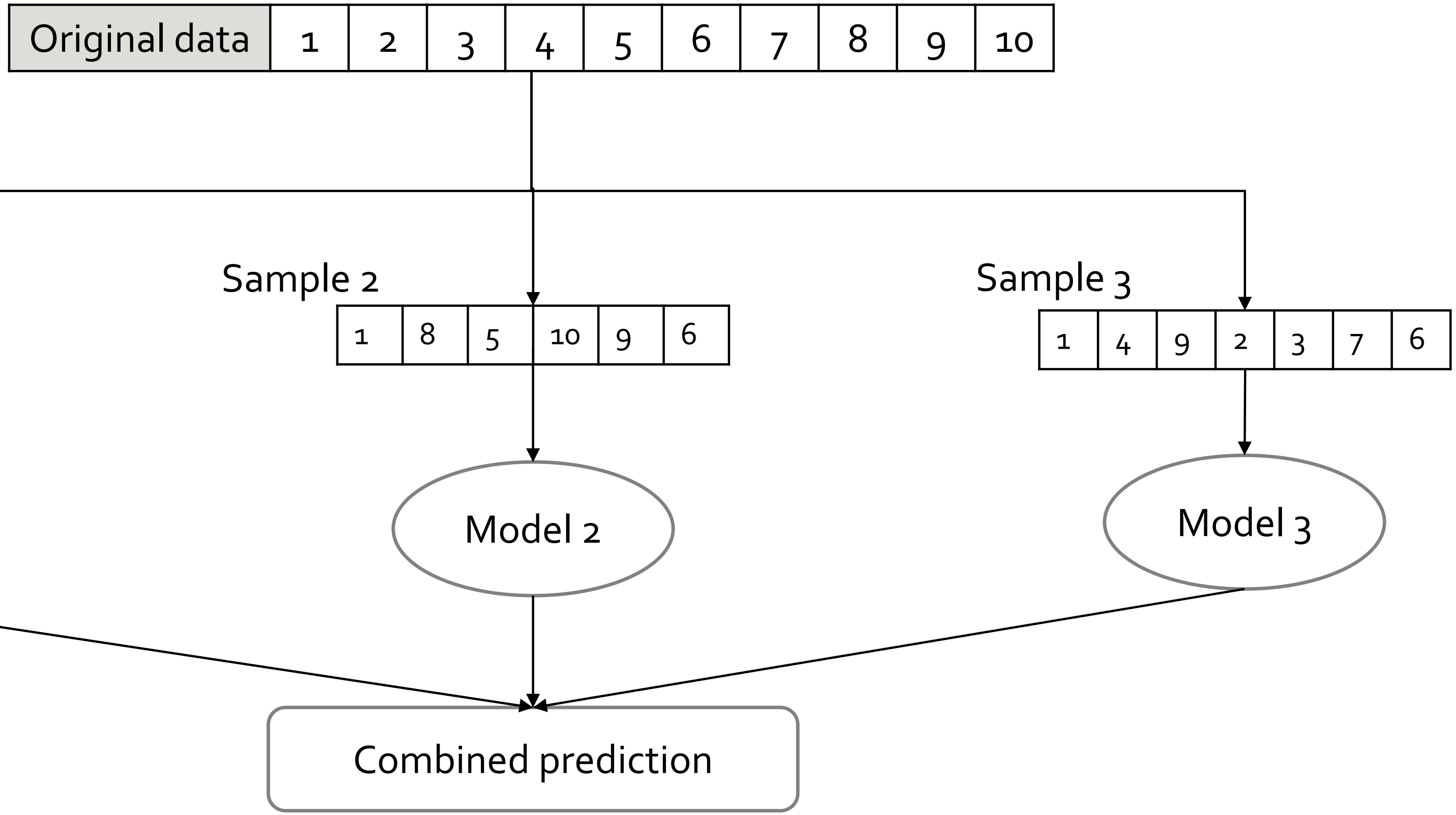
Example of Bagging

Sample size = 10



Example of Pasting

Sample size = 6



Random Subspaces and Random patches

	X_1	X_2	X_3	X_4	X_5
1	10	21	30	44	15
2	12	20	35	40	20
3	10	24	34	43	14
4	15	22	31	41	12
5	19	25	35	42	19
6	12	29	30	45	11

Random Subspaces

1	21	44	15
2	20	40	20
3	24	43	14
4	22	41	12
5	25	42	19
6	29	45	11

1	10	30	44
2	12	35	40
3	10	34	43
4	15	31	41
5	19	35	42
6	12	30	45

1	10	21	30
2	12	20	35
3	10	24	34
4	15	22	31
5	19	25	35
6	12	29	30

Keep all instances and sample Features

Random Patches

1	10	21	30	44	15
3	10	24	34	43	14
4	15	22	31	41	12
5	19	25	35	42	19

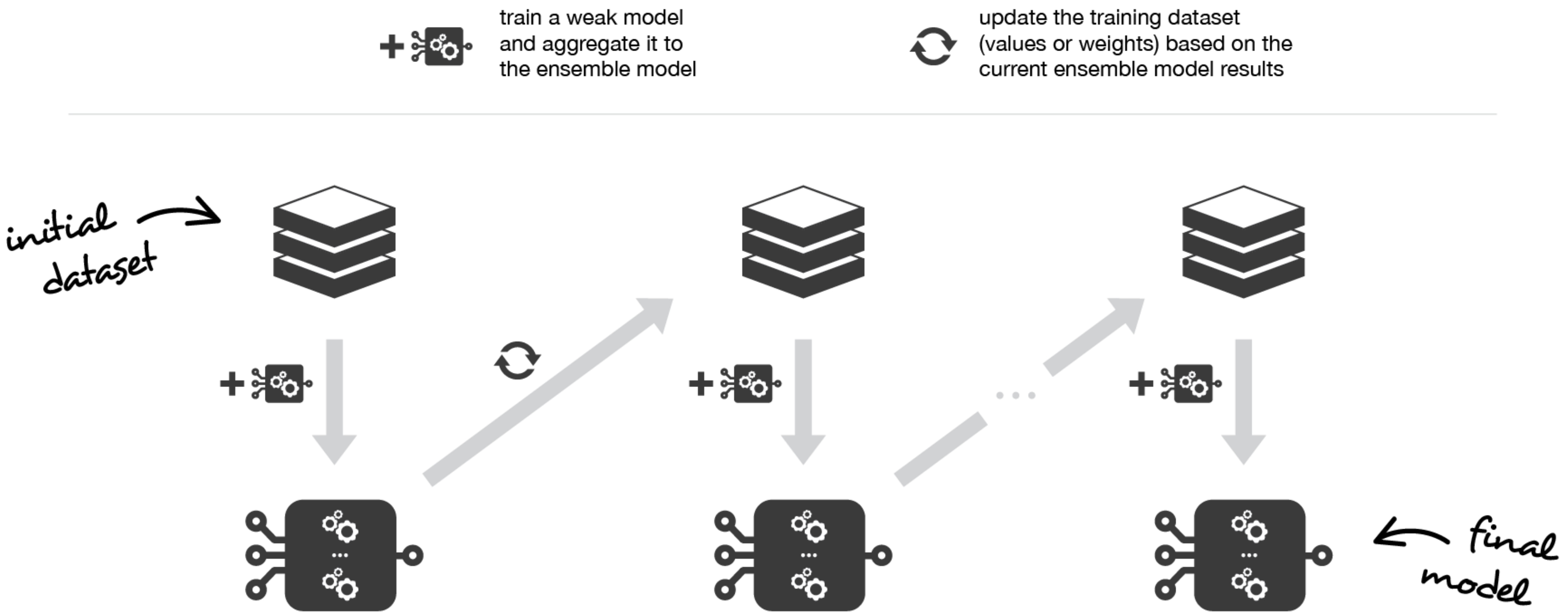
2	12	20	35	40	20
3	10	24	34	43	14
4	15	22	31	41	12
6	12	29	30	45	11

1	10	21	30	44	15
4	15	22	31	41	12
5	19	25	35	42	19
6	12	29	30	45	11

Sampling both instances and Features

Boosting

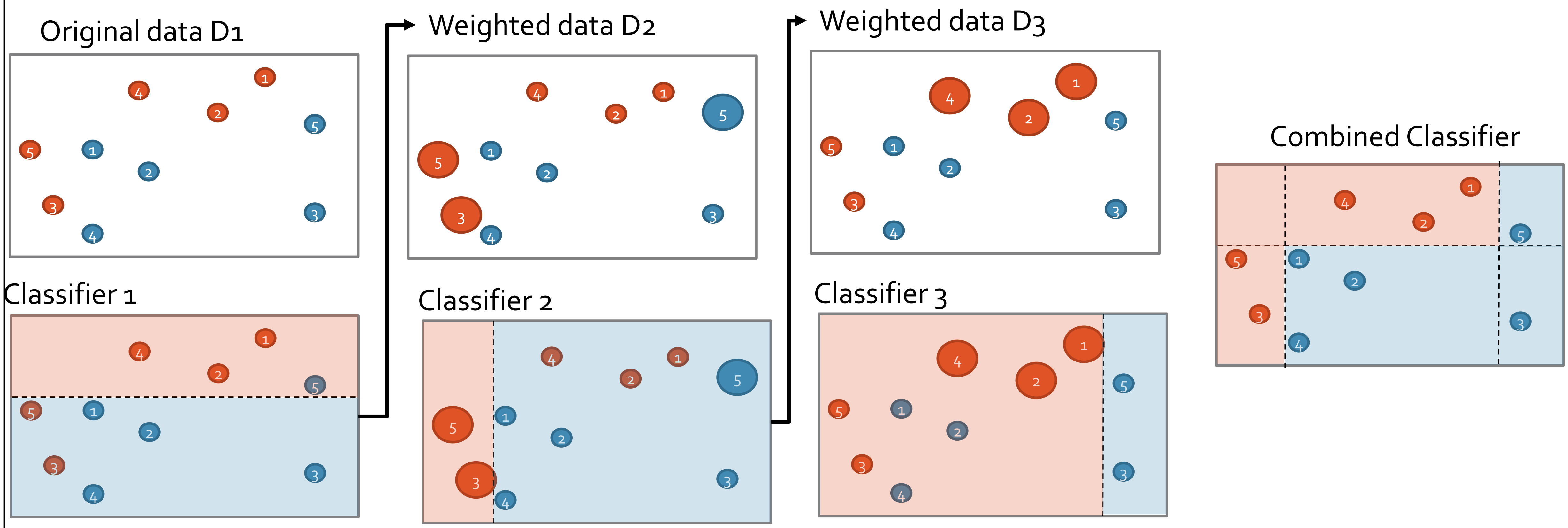
- Unlike bagging, individual predictors are trained sequentially, each trying to correct its predecessor.



Adaboost (Adaptive boosting)

- Instead of sampling, re-weigh samples
  - Samples are given weights.
  - Start with uniform weighting
- At each iteration, a model is learned and the samples are re-weighted so the next classifiers focus on samples that were wrongly predicted by previous classifier
  - Weights of correctly predicted samples are decreased
  - Weights of incorrectly predicted samples are increased
- Final prediction is a combination of model predictions weighted by their respective error measures

Adaboost


Original data D1
Weighted data D2
Weighted data D3
Classifier 1
Classifier 2
Classifier 3
Combined Classifier

Adaboost algorithm

- Each sample weight w_i in the training set is initialized to $\frac{1}{m}$, where m is the number of samples in training set.

- For each trained predictor j, compute its weighted error r_i and its weight α_j

$$r_j = \frac{\sum\limits_{i=1,\, \hat{y}_j^{(i)} \neq y^{(i)}}^{m} w^{(i)}}{\sum\limits^{m} w^{(i)}}, \ \alpha_j = \eta \cdot \log \frac{1-r_j}{r_j}$$

- Update sample weights

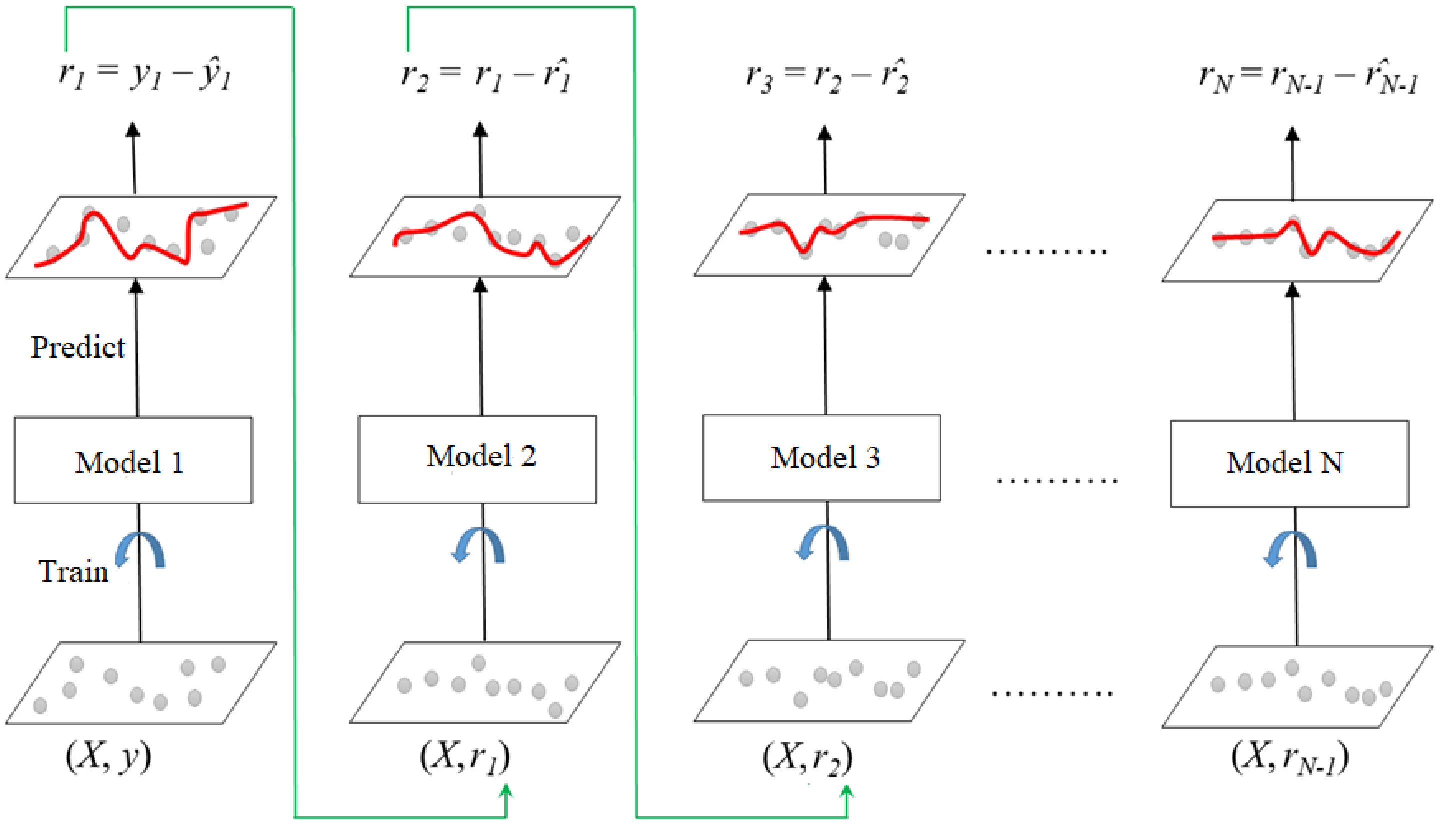
$$w^{(i)} \leftarrow \begin{cases} w^{(i)} & \text{if } \hat{y}_j^{(i)} = y^{(i)} \\ w^{(i)} \cdot exp(\alpha_j) & \text{if } \hat{y}_j^{(i)} \neq y^{(i)} \end{cases}$$

- For final predictions

$$\hat{y}(\mathbf{x}) = \arg\max_k \sum_{j=1,\, \hat{y}_j(\mathbf{x})=k}^{N} \alpha_j$$

Gradient Boosting

- Unlike AdaBoost, Gradient Boosting tries to fit the new predictor to the *residual errors* made by the previous predictor.



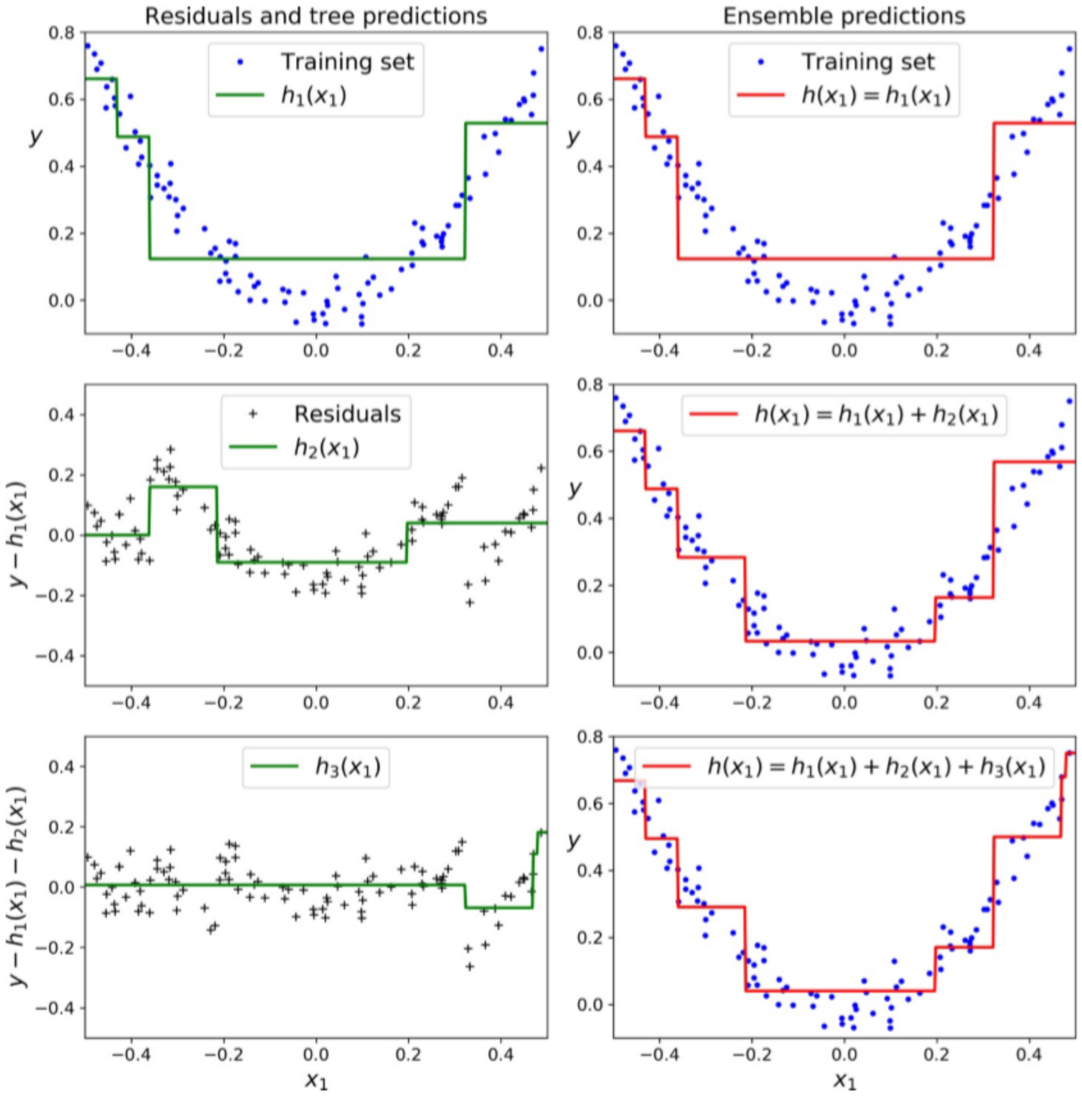
Gradient Boosting

$$Dataset\ D = \{x_i, y_i\}_1^m$$

$$D_1 = \{x_i, y_i\}_1^m$$

$$D_2 = \{x_i, y_i - h_1(x_i)\}_1^m$$

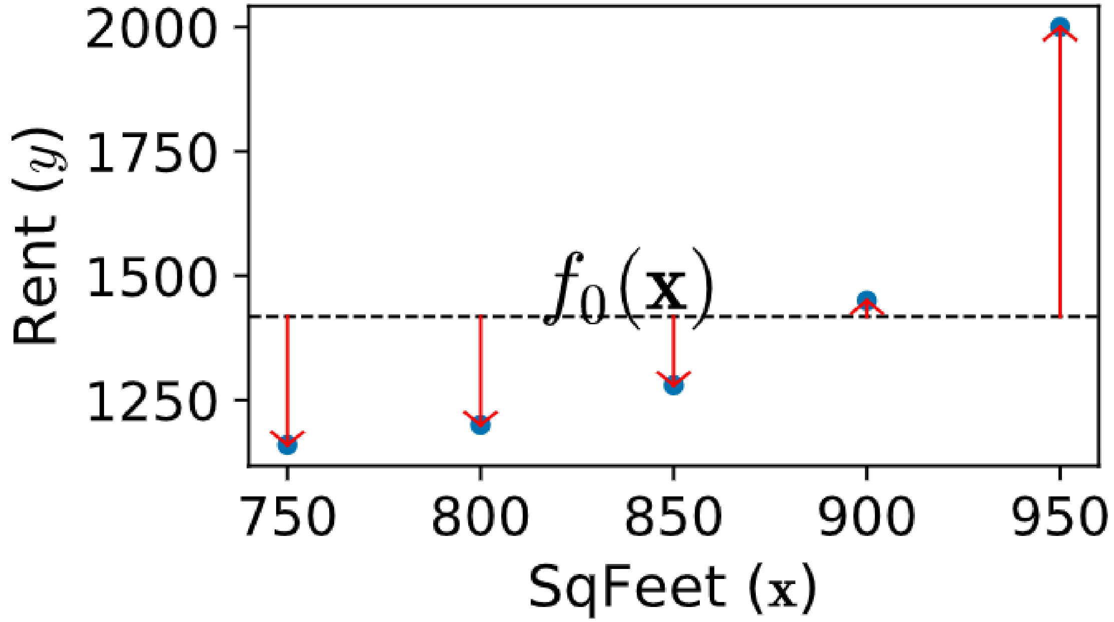
$$D_3 = \{x_i, y_i - h_1(x_i) - h_2(x_i)\}_1^m$$



Gradiant boosting: example

- Training data : square footage data on five apartments and their rent prices in dollars per month
- We use the mean (average) of the rent prices as our initial model Fo

sqfeet	rent	F_0	$\mathbf{y} - F_0$
750	1160	1418	-258
800	1200	1418	-218
850	1280	1418	-138
900	1450	1418	32
950	2000	1418	582



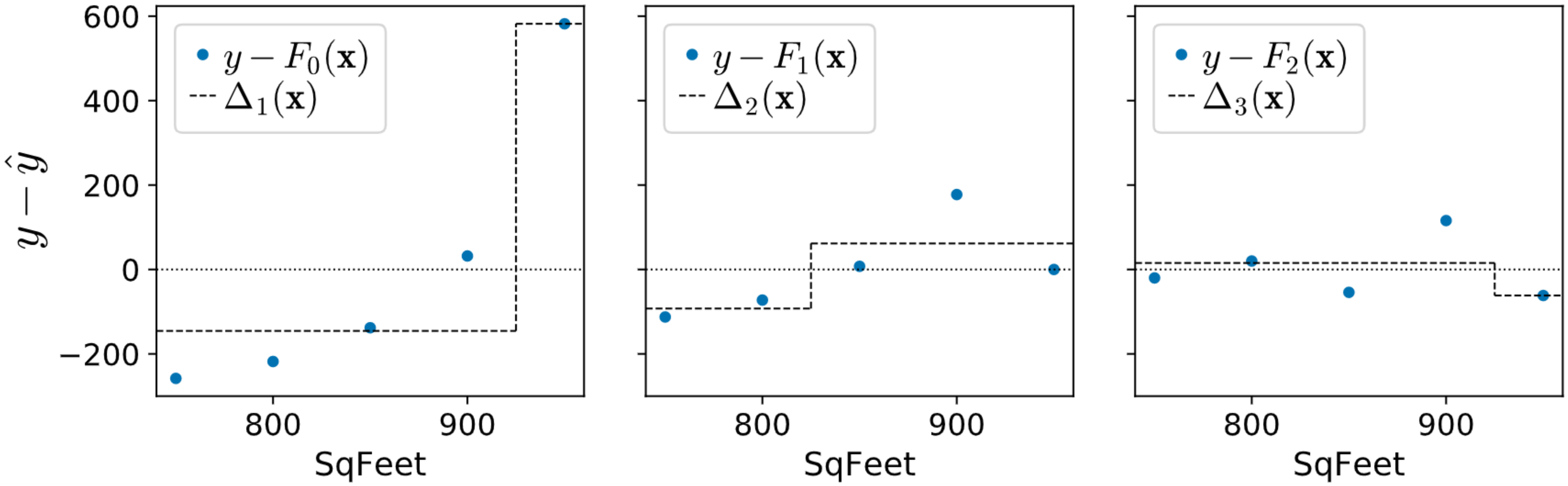
Gradiant boosting: example

- Next, we train weak models Δ_i to predict residuals for all i observations

Δ_1	F_1	$\mathbf{y}$-F_1	Δ_2	F_2	$\mathbf{y}$ - F_2	Δ_3	F_3
-145.5	1272.5	-112.5	-92.5	1180	-20	15.4	1195.4
-145.5	1272.5	-72.5	-92.5	1180	20	15.4	1195.4
-145.5	1272.5	7.5	61.7	1334.2	-54.2	15.4	1349.6
-145.5	1272.5	177.5	61.7	1334.2	115.8	15.4	1349.6
582	2000	0	61.7	2061.7	-61.7	-61.7	2000

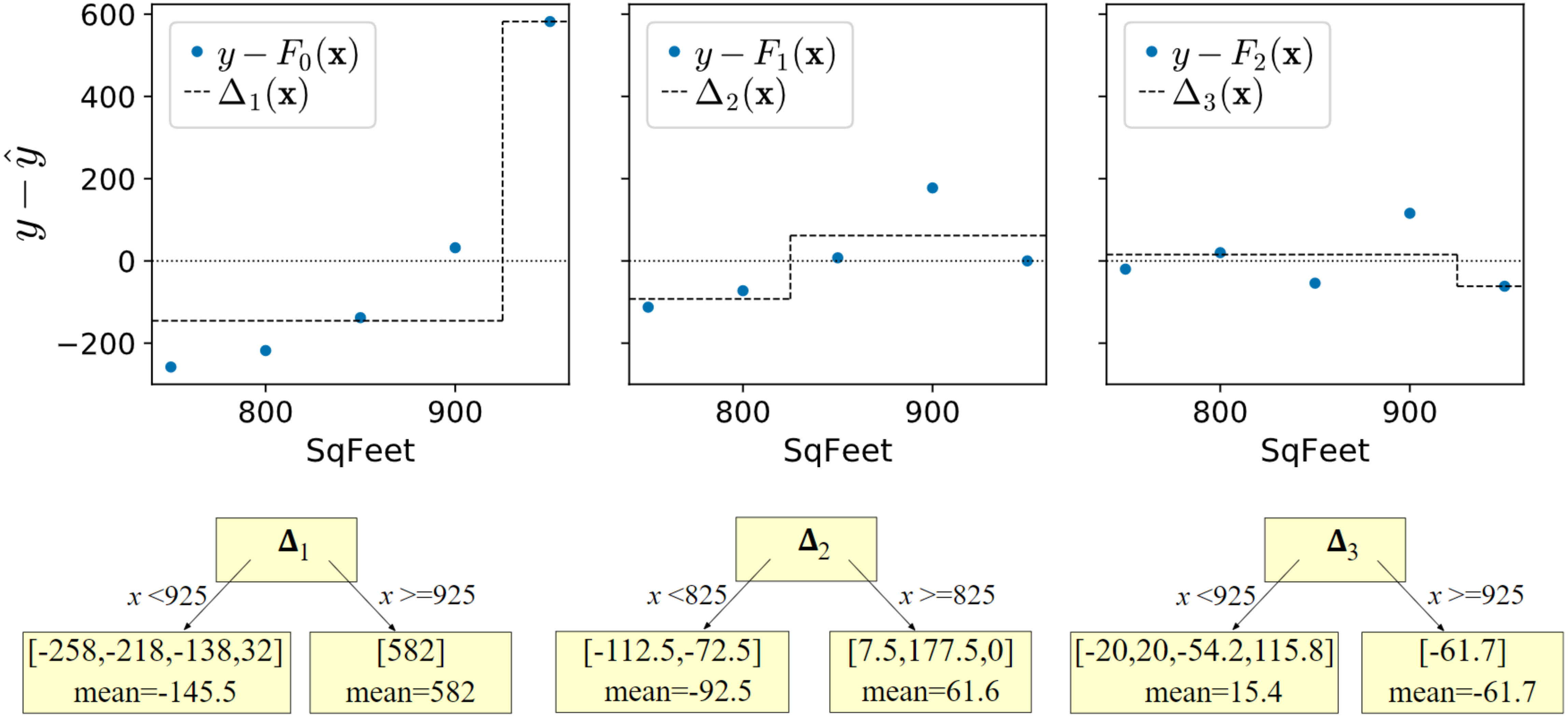
Gradiant boosting: example

- Next, we train weak models Δ_i to predict residuals for all i observations



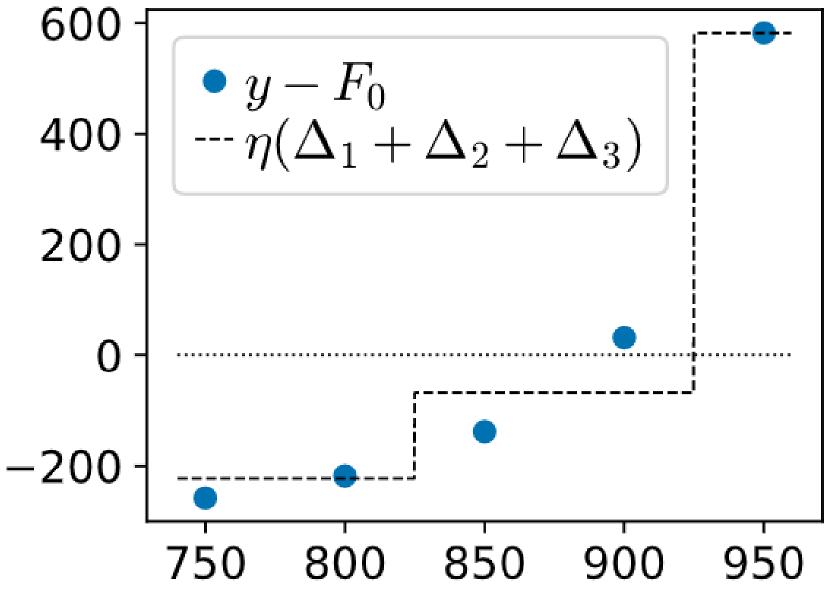
- The residuals (blue dots) get smaller as we add more learners

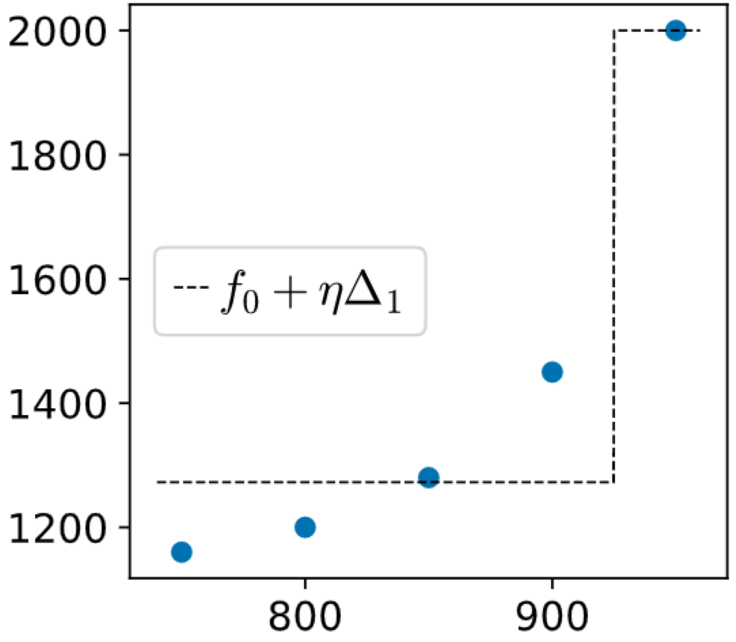
Gradiant boosting: example

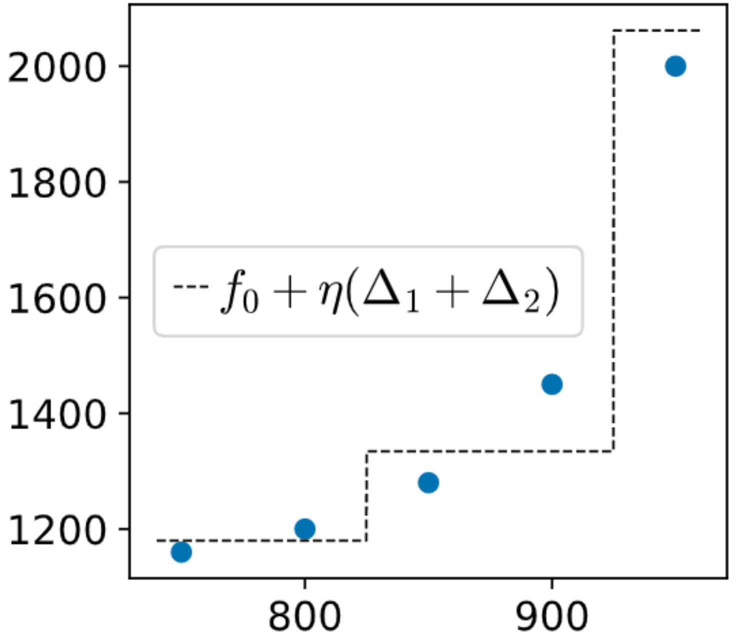


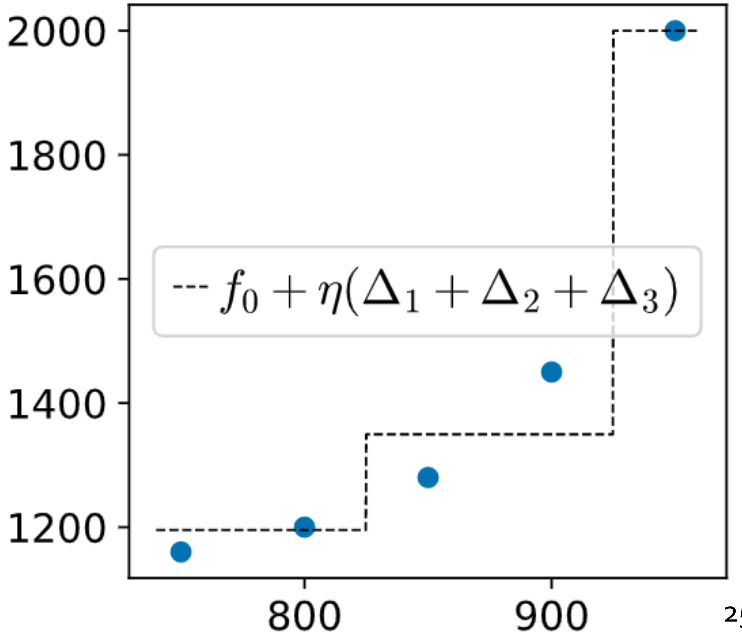
Gradiant boosting: example

- Summing the three learners



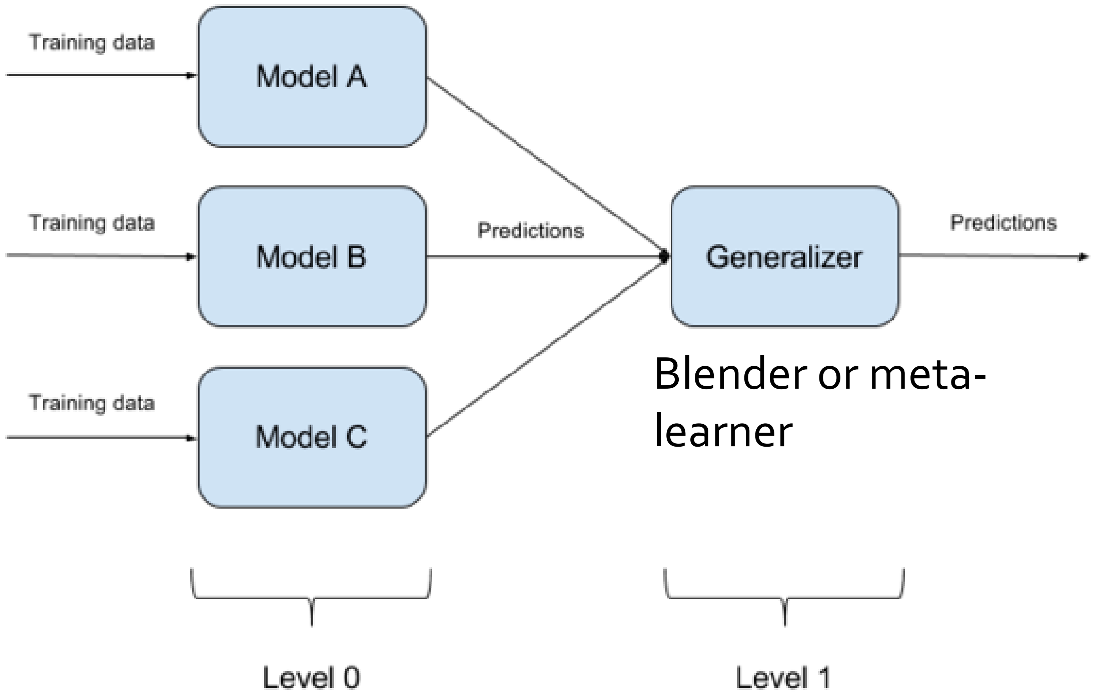






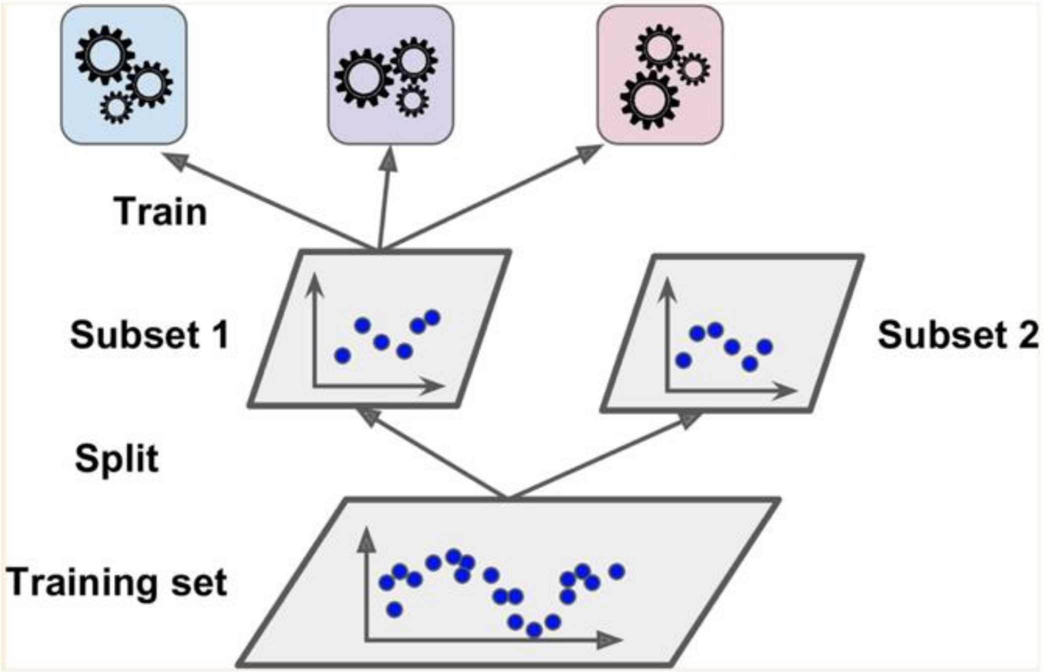
Stacking (Stacked generalization)

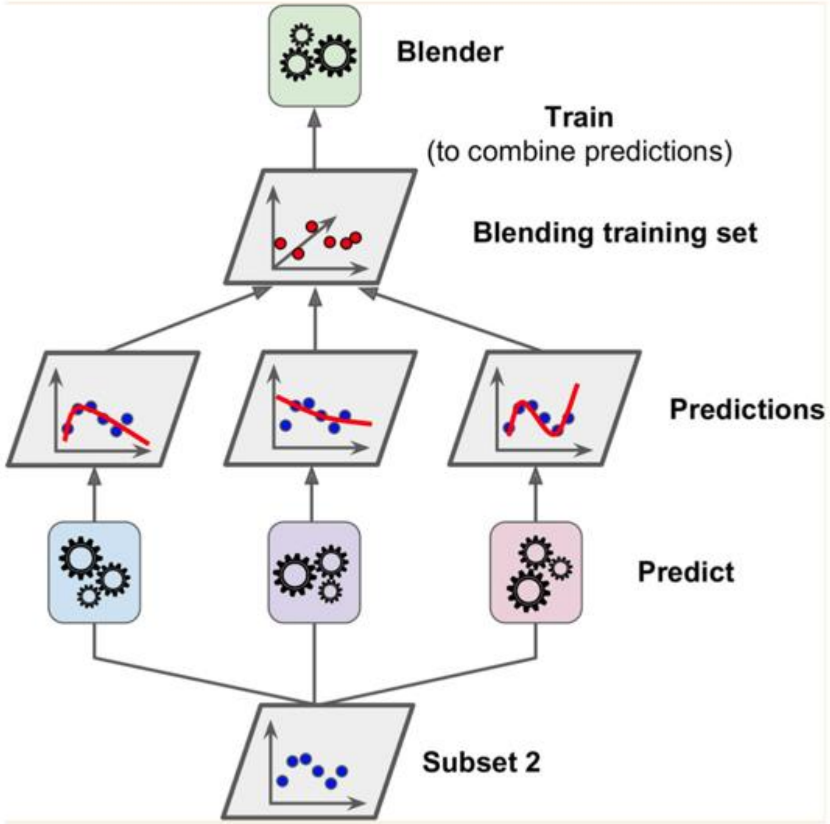
- Basic idea:
  - Train a separate model to perform aggregation of the predictions of the individual classifiers
  - Predictions from the train set are used as features for level 1 model.
  - Level 1 model is used to make a prediction on the test set



Stacking (Stacked generalization)

- Stacking with a hold-out set → Blending





Stacking

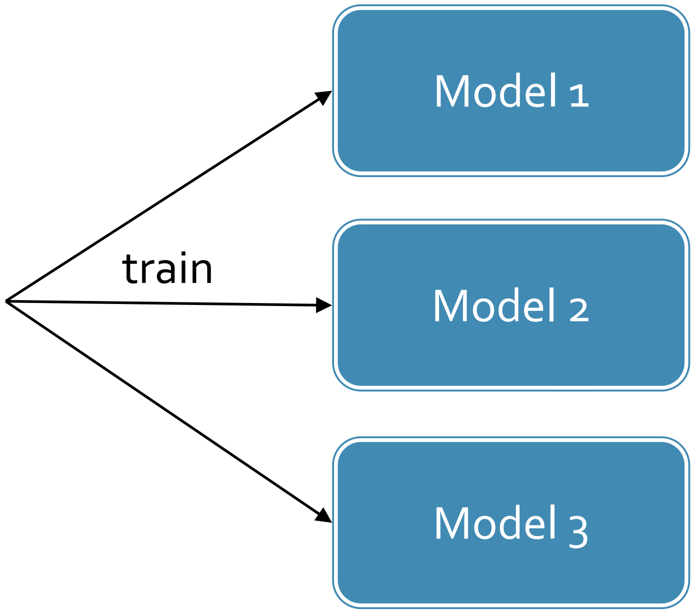
- Training level 0 models

Training set

X1	X2	X3	X4	C
$x_{1,1}$	$x_{2,1}$	$x_{3,1}$	$x_{4,1}$	True
$x_{1,2}$	$x_{2,2}$	$x_{3,2}$	$x_{4,2}$	False
…	…	…	…	…
$x_{1,n}$	$x_{2,n}$	$x_{3,n}$	$x_{4,n}$	False

Training Subset 1

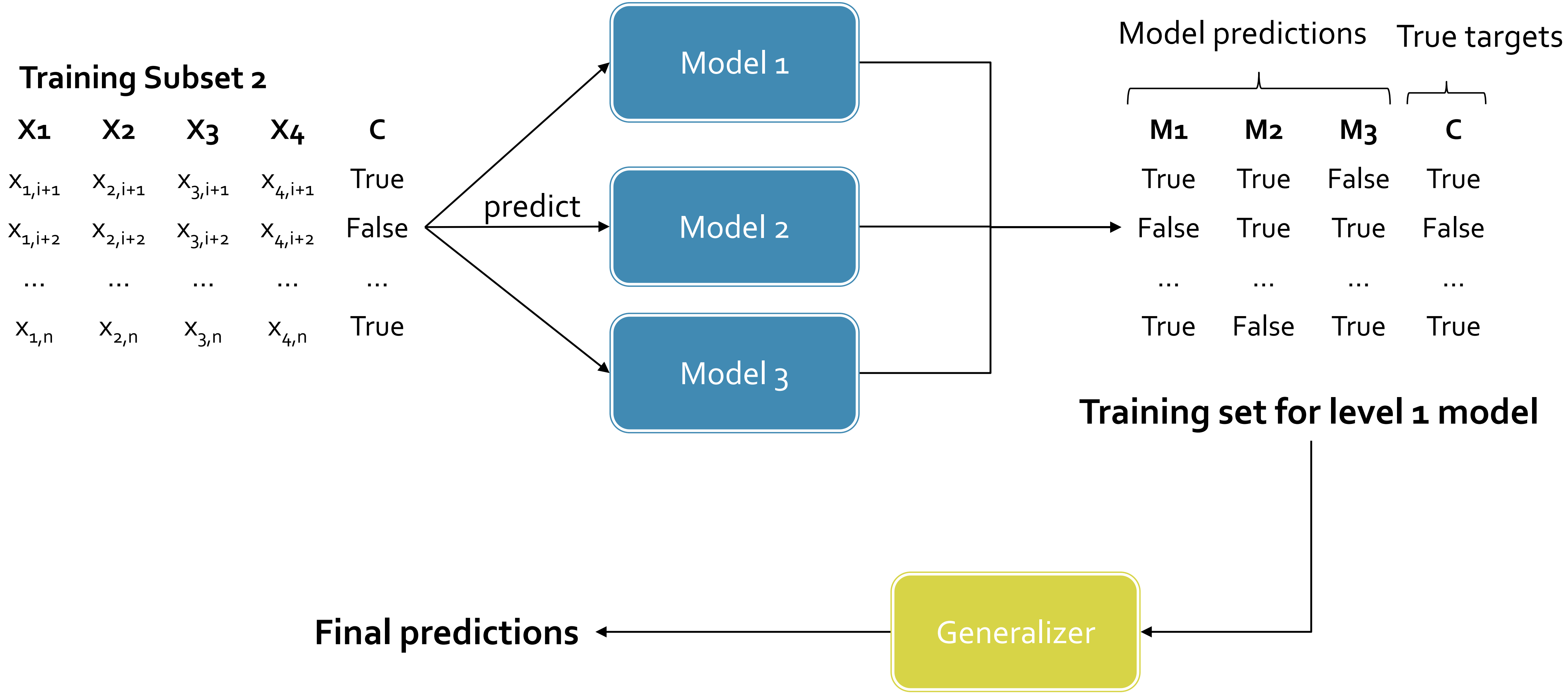
X1	X2	X3	X4	C
$x_{1,1}$	$x_{2,1}$	$x_{3,1}$	$x_{4,1}$	True
$x_{1,2}$	$x_{2,2}$	$x_{3,2}$	$x_{4,2}$	False
…	…	…	…	…
$x_{1,i}$	$x_{2,i}$	$x_{3,i}$	$x_{4,i}$	True



Training Subset 2

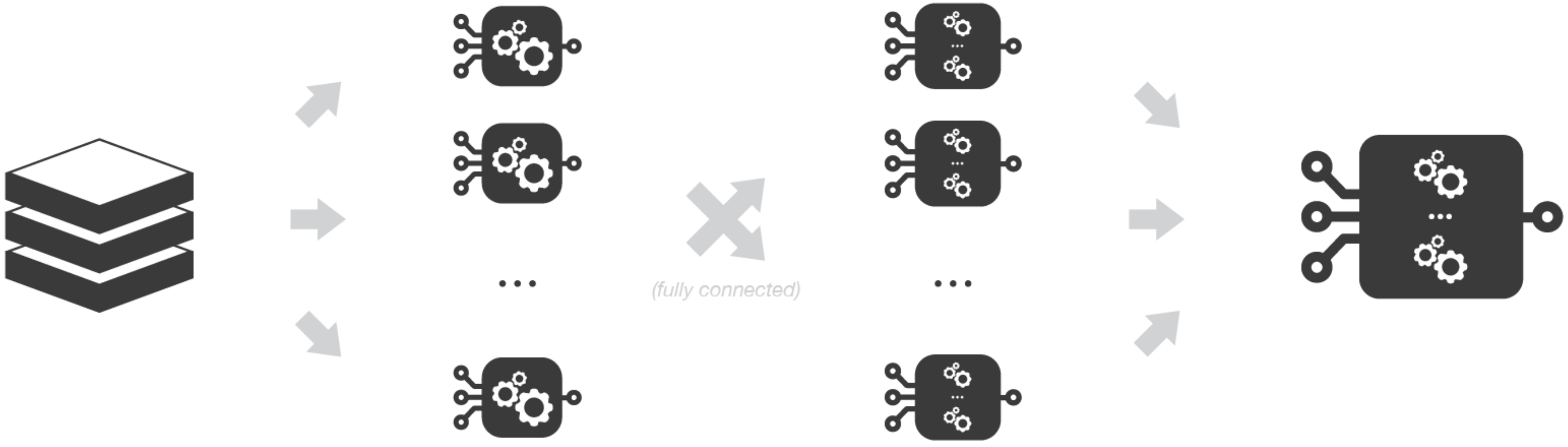
X1	X2	X3	X4	C
$x_{1,i+1}$	$x_{2,i+1}$	$x_{3,i+1}$	$x_{4,i+1}$	True
$x_{1,i+2}$	$x_{2,i+2}$	$x_{3,i+2}$	$x_{4,i+2}$	False
…	…	…	…	…
$x_{1,n}$	$x_{2,n}$	$x_{3,n}$	$x_{4,n}$	True

Stacking



Training Subset 2
X1 X2 X3 X4 C
$x_{1,i+1}$ $x_{2,i+1}$ $x_{3,i+1}$ $x_{4,i+1}$ True
$x_{1,i+2}$ $x_{2,i+2}$ $x_{3,i+2}$ $x_{4,i+2}$ False
… … … … …
$x_{1,n}$ $x_{2,n}$ $x_{3,n}$ $x_{4,n}$ True
predict
Model 1
Model 2
Model 3
Model predictions
True targets
M1 M2 M3 C
True True False True
False True True False
… … … …
True False True True
Training set for level 1 model
Generalizer
Final predictions

Multilayer stacking



initial dataset

L weak learners
(that can be non-homogeneous)

M meta-models
(**trained** to output predictions based on previous layer predictions)

final meta-model
(**trained** to output predictions based on previous layer predictions)

Conclusions

- Ensemble learning is about training multiple base models and combined them to obtain a strong model with better performance
  - Ideally low bias, low variance
- In ***bagging ensembles***, instances of the same base model are trained in parallel on random subsets of data and then aggregated
  - Using random sampling reduce variance
- In ***boosting ensembles***, instances of the same base model are trained iteratively, such that, each model attempts to correct the predictions of the previous model.
- ***Stacking ensembles*** use multi-stage training. Different types of base models are trained at the very first stage on top of which a meta-model is trained to make predictions based on based model predictions

Ensemble learning on diabetes data

- **Load the diabetes data and split it into training set, a validation set, and a test set**
  - 30% of data for test, 30% of training for validation
- **Train various classifiers individually: Decision tree, KNN and SVM**
- **Voting ensemble: Combine the classifiers into an ensemble using hard or soft voting.**
  - Use the validation set to find the best ensemble (it must outperforms the individual classifiers)
  - Evaluate the best model found on the test set and compare the results
- **Stacking ensemble: using the previous classifiers**
  - Create a new training set (for the meta learner) using the predictions on the validation set
  - Train a classifier (e.g, random forest) on the new training set
  - Evaluate the model on the test set and compare the results