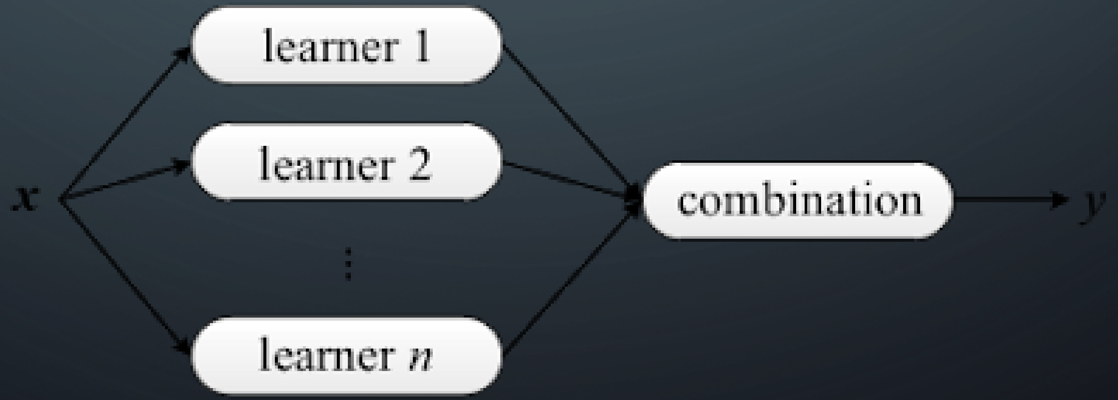
ENSEMBLE LEARNING

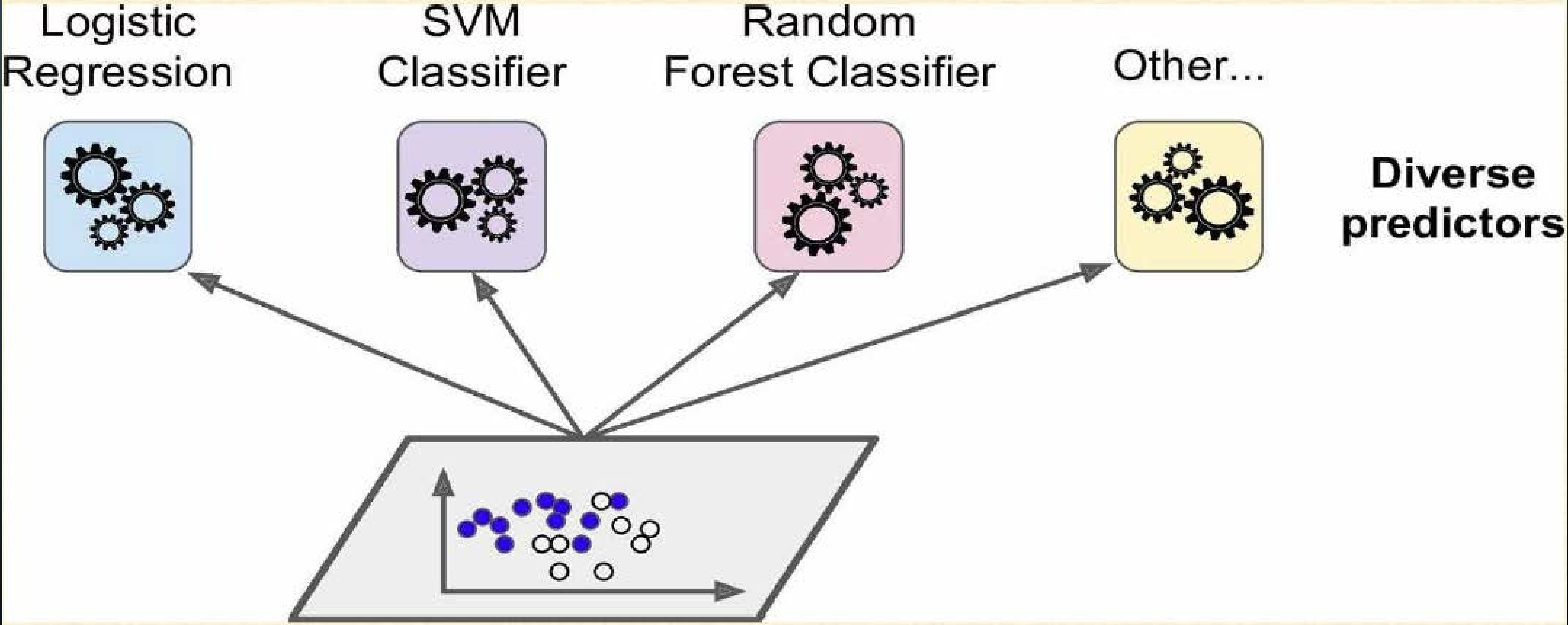
MOHAMMAD GHODDOSI

WHAT IS ENSEMBLE LEARNING

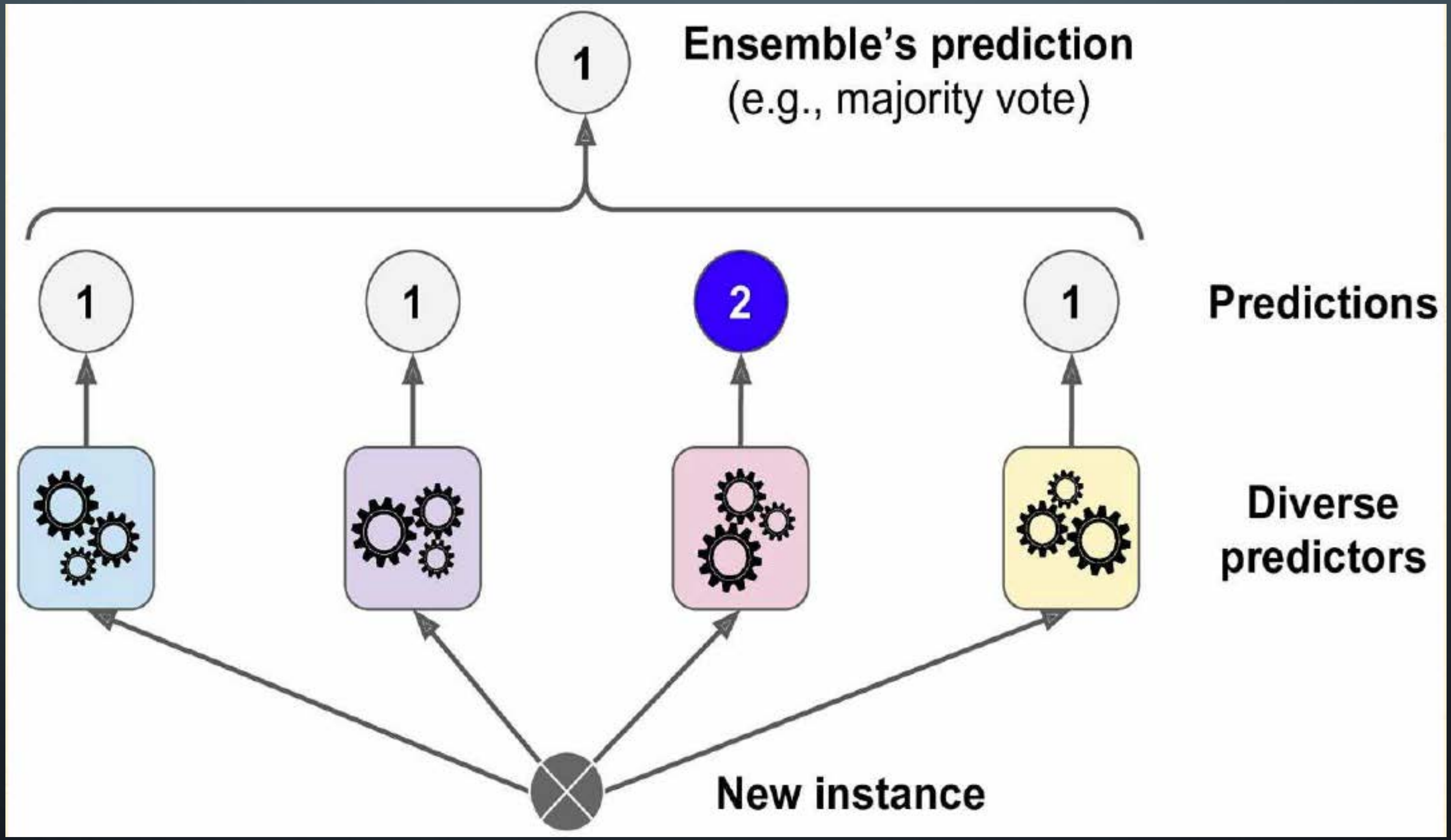
- Grouping multiple predictors is called ensemble learning
- The winning solutions in Machine Learning competitions often involve several Ensemble methods
- Wisdom of crowd



ENSEMBLE LEARNING



VOTING CLASSIFIER



WHY DOSE ENSEMBLE HELP (1)

- We have a biased coin
  - 51% chance of head
  - 49% chance of tail
- If we toss it 10,000 times, we will get more or less
  - 510 heads
  - 490 tails
- So majority of heads.

WHY DOSE ENSEMBLE HELP (2)

- If we flip this coin 3 times:
  - 3 heads and 0 tails => 0.51 * 0.51 * 0.51 * 1 = 0.132651
  - 2 heads and 1 tails => 0.51 * 0.51 * 0.49 * 3 = 0.382347
  - 1 heads and 2 tails => 0.51 * 0.49 * 0.49 * 3 = 0.367353
  - 0 heads and 3 tails => 0.49 * 0.49 * 0.49 * 1 = 0.117649
- P(majority of heads) = 0.514998

WHY DOSE ENSEMBLE HELP (3)

- If we toss a coin **n** times we have: $P(\boldsymbol{r}\ number\ of\ heads) \binom{n}{r} * a^r * b^{n-r}$
  - a is probability of heads
  - b is probability of tails
- For n = 1000, probability of majority of heads is : 72.6%

VOTING

- Hard voting
  - Just final output of each classifier
- Soft voting
  - Use probabilities to compute final output

jupyter

PROBLEMS

- We need a diverse set of classifiers
- Use different training algorithm
- Use different random subsets of training data

BAGGING AND PASTING

- Bagging
  - Sampling with replacement
  - Also called bootstrapping
- Pasting
  - Sampling without replacement

RANDOM PATCHES AND RANDOM SUBSPACES

- Like pasting but for features
- Each predictor is trained on a subset of features
- Useful for high-dimensional inputs
- Random patches:
  - Sampling both training instances and features
- Random subspaces:
  - Just sampling features

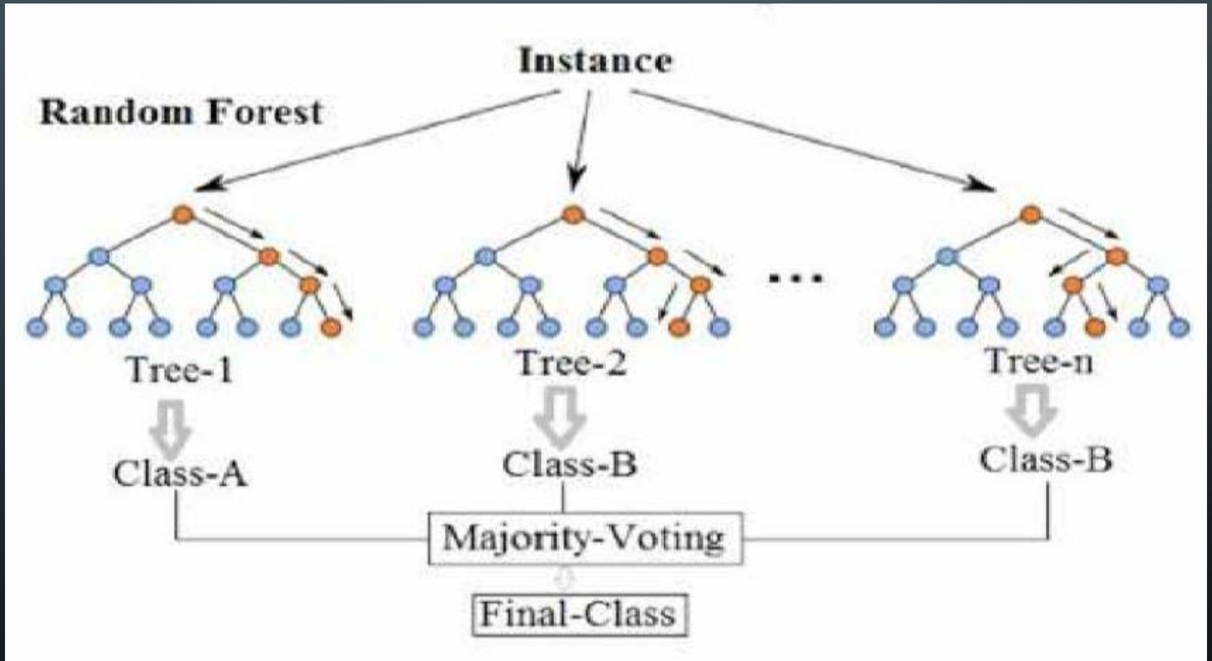
jupyter

RANDOM FORESTS

MOHAMMAD GHODDOSI

RANDOM FORESTS

- An ensemble of Decision Trees
- Generally trained with the bagging method



EXTRA RANDOMNESS

- Searching in a small random subset of features for splitting
  - Greater tree diversity
  - Higher bias
  - Lower variance
- Using a random threshold for each feature
- Extra-Tree

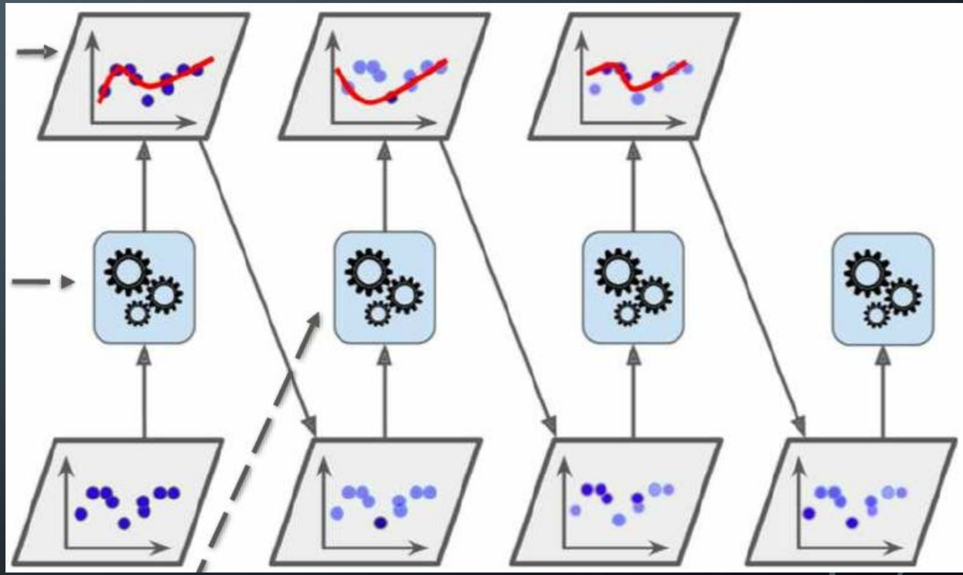
jupyter

BOOSTING

- Training sequentially
- Each predictor trying to correct its predecessor
  - AdaBoost (Adaptive Boosting)
  - Gradient Boosting

ADABOOSTING

- Each predictor pay a bit more attention to the training instances that the predecessor underfitted
- Focusing more and more on the hard cases



class sklearn.ensemble.**AdaBoostClassifier**(*base_estimator=None*, ***, *n_estimators=50*, *learning_rate=1.0*, *algorithm='SAMME.R'*, *random_state=None*) [source]

GRADIENT BOOSTING

- Sequentially adding predictors to an ensemble, each one correcting its predecessor
- Fit new predictor to the residual error
- $h(x) = h_1(x) + h_2(x) + h_3(x) + \ ... + h_k(x)$