Ensemble Learning Methods

Tushar B. Kute,
http://tusharkute.com

Ensemble Learning

- An ensemble is a composite model, combines a series of low performing classifiers with the aim of creating an improved classifier.
- Here, individual classifier vote and final prediction label returned that performs majority voting.
- Ensembles offer more accuracy than individual or base classifier.
- Ensemble methods can parallelize by allocating each base learner to different-different machines.
- Finally, you can say Ensemble learning methods are meta-algorithms that combine several machine learning methods into a single predictive model to increase performance.
- Ensemble methods can decrease variance using bagging approach, bias using a boosting approach, or improve predictions using stacking approach.

Real life examples

- Let's take a real example to build the intuition.
- Suppose, you want to invest in a company XYZ. You are not sure about its performance though.
- So, you look for advice on whether the stock price will increase by more than 6% per annum or not?
- You decide to approach various experts having diverse domain experience:

The survey prediction

- Employee of Company XYZ:
  - In the past, he has been right 70% times.
- Financial Advisor of Company XYZ:
  - In the past, he has been right 75% times.
- Stock Market Trader:
  - In the past, he has been right 70% times.
- Employee of a competitor:
  - In the past, he has been right 60% times.
- Market Research team in the same segment:
  - In the past, he has been right 75% times.
- Social Media Expert:
  - In the past, he has been right 65% times.

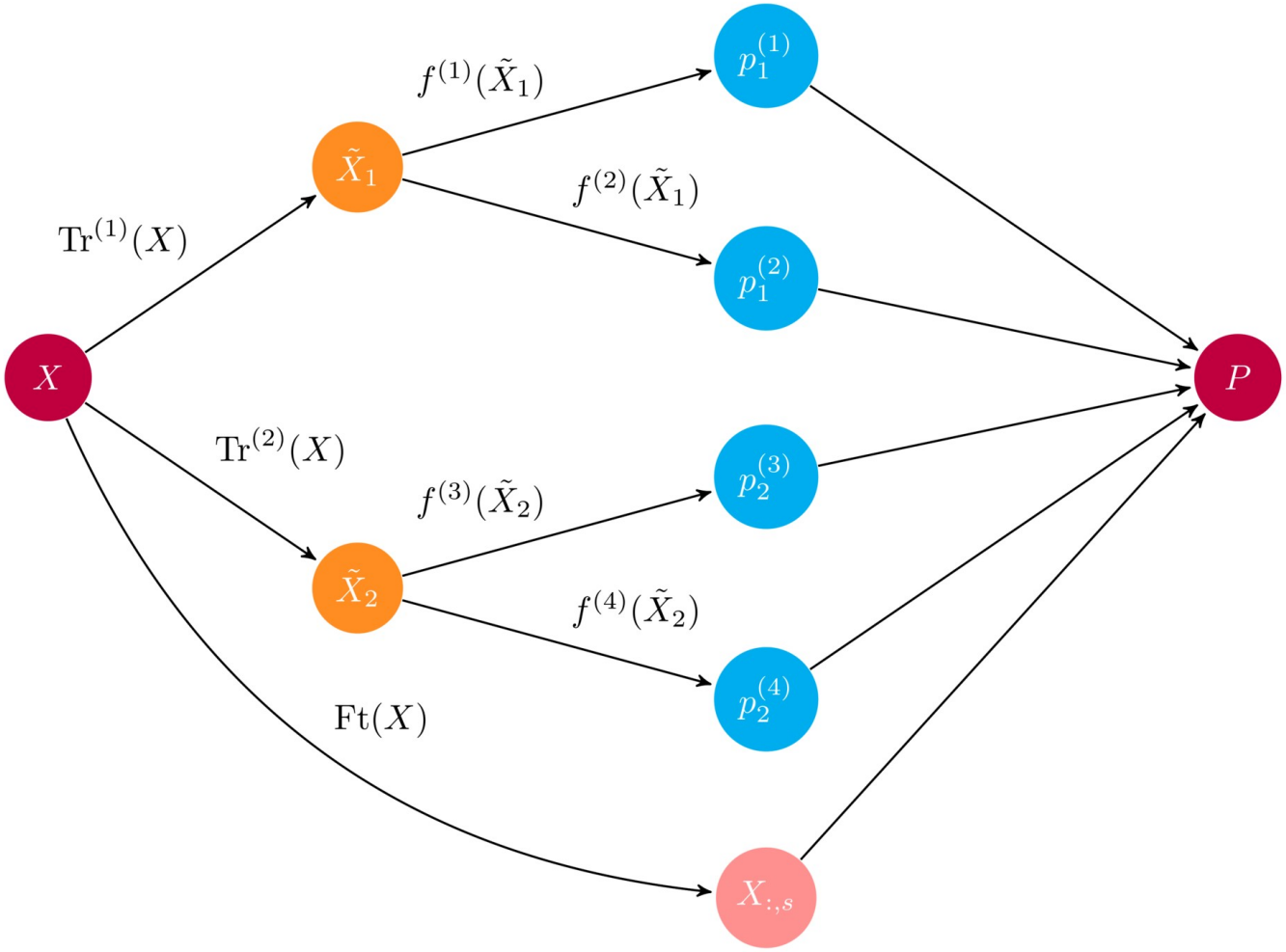
Conclusion

- Given the broad spectrum of access you have, you can probably combine all the information and make an informed decision.
- In a scenario when all the 6 experts/teams verify that it's a good decision(assuming all the predictions are independent of each other), you will get a combined accuracy rate of 1 - (30% . 25% . 30% . 40% . 25% . 35%) = 1 - 0.07875 = 99.92125%
- The assumption used here that all the predictions are completely independent is slightly extreme as they are expected to be correlated. However, you can see how we can be so sure by combining various forecasts together.
- Well, Ensemble learning is **no** different.

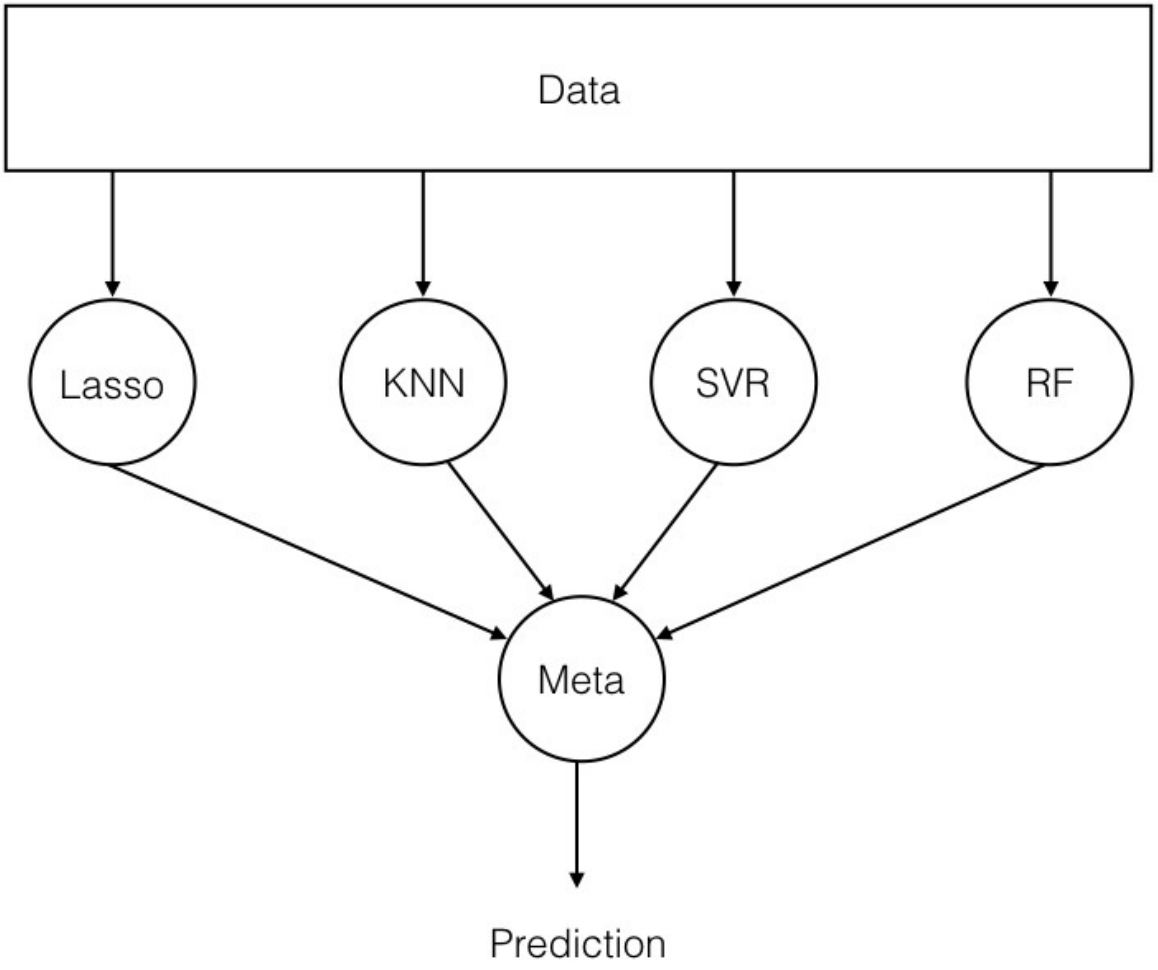
Ensembling

- An ensemble is the art of combining a diverse set of learners (individual models) together to improvise on the stability and predictive power of the model.
- In our example, the way we combine all the predictions collectively will be termed as Ensemble learning.
- Moreover, Ensemble-based models can be incorporated in both of the two scenarios, i.e., when data is of large volume and when data is too little.

Ensemble Schematic



Basic Ensemble Structure



Summary

- Use multiple learning algorithms (classifiers)
- Combine the decisions
- Can be more accurate than the individual classifiers
- Generate a group of base-learners
- Different learners use different
  - Algorithms
  - Hyperparameters
  - Representations (Modalities)
  - Training sets

How models are different?

- Difference in population
- Difference in hypothesis
- Difference in modeling technique
- Difference in initial seed

Why ensembles ?

- There are two main reasons to use an ensemble over a single model, and they are related; they are:
  - Performance: An ensemble can make better predictions and achieve better performance than any single contributing model.
  - Robustness: An ensemble reduces the spread or dispersion of the predictions and model performance.

Model Error

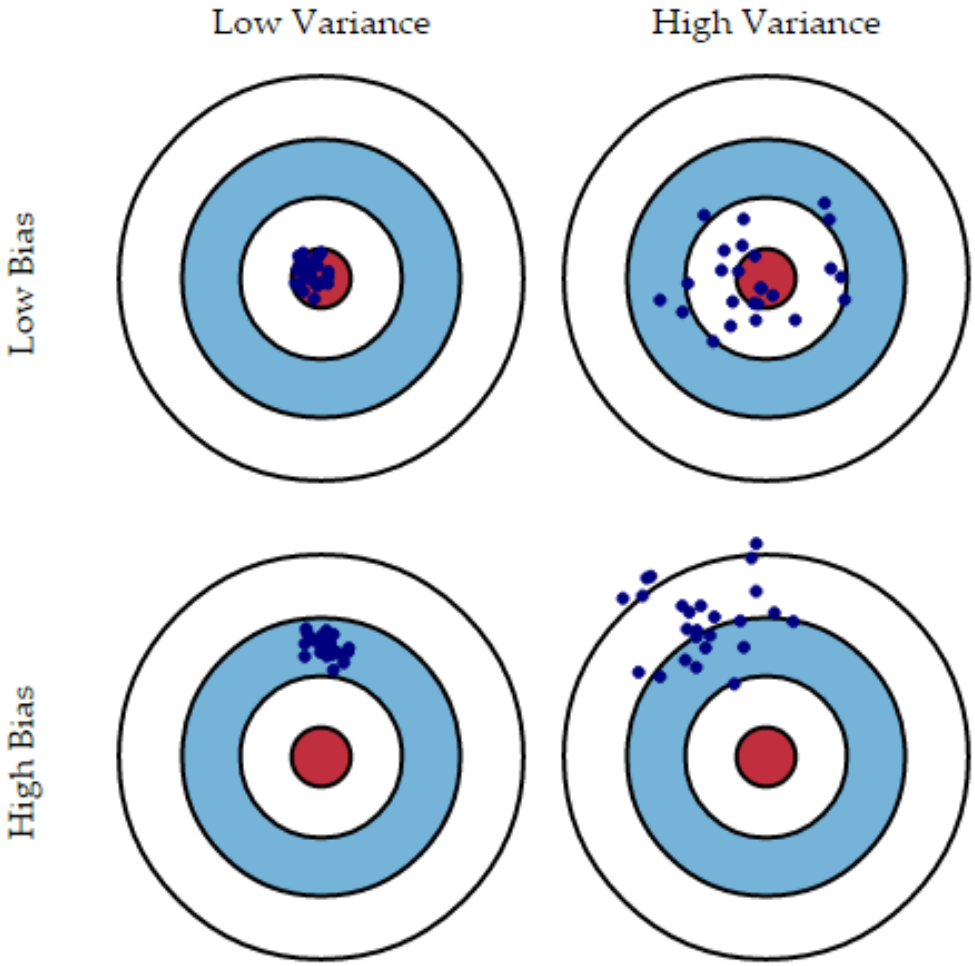
- The error emerging from any machine model can be broken down into three components mathematically. Following are these component:

$$Err(x) = \Big(E[\hat{f}\,(x)] - f(x)\Big)^2 + E\Big[\hat{f}\,(x) - E[\hat{f}\,(x)]\Big]^2 + \sigma_e^2$$

$$Err(x) = \text{Bias}^2 + \text{Variance} + \text{Irreducible Error}$$

- **Bias error** is useful to quantify how much on an average are the predicted values different from the actual value. A high bias error means we have an under-performing model which keeps on missing essential trends.
- **Variance** on the other side quantifies how are the prediction made on the same observation different from each other. A high variance model will over-fit on your training population and perform poorly on any observation beyond training.

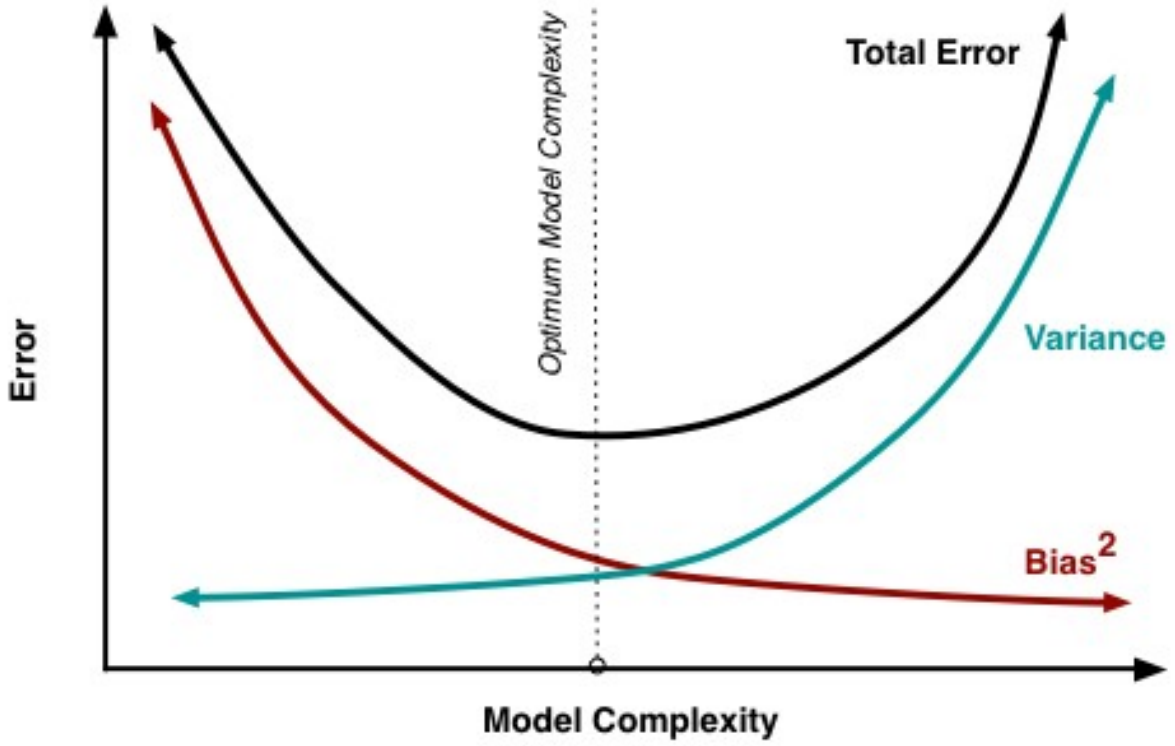
Bias and Variance



Bias – Variance Tradeoff

- Normally, as you increase the complexity of your model, you will see a reduction in error due to lower bias in the model. However, this only happens till a particular point.
- As you continue to make your model more complex, you end up over-fitting your model and hence your model will start suffering from high variance.
- A champion model should maintain a balance between these two types of errors. This is known as the trade-off management of bias-variance errors. Ensemble learning is one way to execute this trade off analysis.

Bias – Variance Tradeoff



Ensemble Creation Approaches

- Unweighted Voting (e.g. Bagging)
- Weighted voting – based on accuracy (e.g. Boosting), Expertise, etc.
- Stacking - Learn the combination function

Ensemble Learning Methods

- Bagging
- Boosting
- Stacking

Bootstrap

- The bootstrap is a powerful statistical method for estimating a quantity from a data sample. This is easiest to understand if the quantity is a descriptive statistic such as a mean or a standard deviation.
- Let’s assume we have a sample of 100 values (x) and we’d like to get an estimate of the mean of the sample.
- We can calculate the mean directly from the sample as:

$$mean(x) = 1/100 * sum(x)$$

Bootstrap

- We know that our sample is small and that our mean has error in it. We can improve the estimate of our mean using the bootstrap procedure:
- Create many (e.g. 1000) random sub-samples of our dataset with replacement (meaning we can select the same value multiple times).
- Calculate the mean of each sub-sample.
- Calculate the average of all of our collected means and use that as our estimated mean for the data.

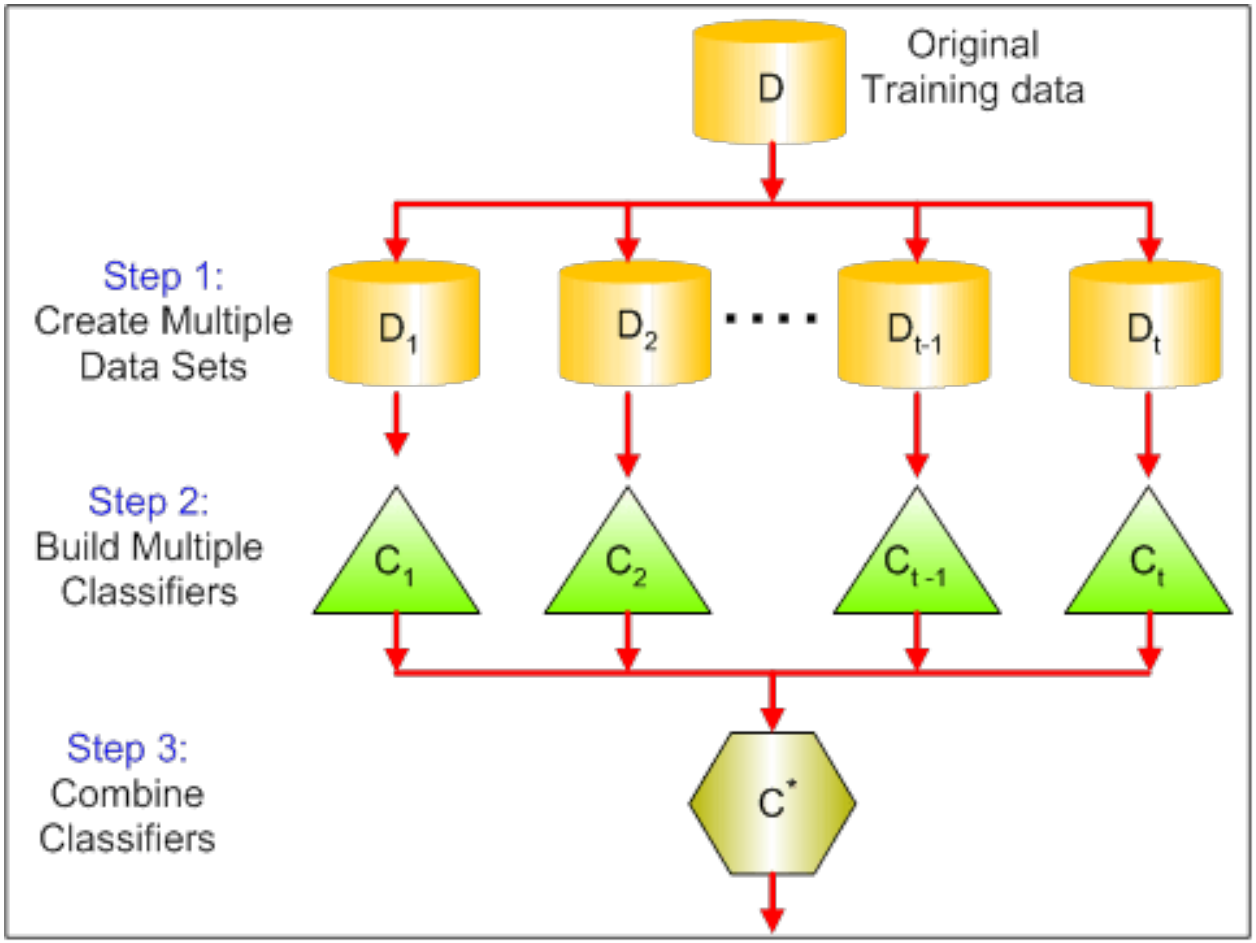
Bootstrap

- For example, let’s say we used 3 resamples and got the mean values 2.3, 4.5 and 3.3.
- Taking the average of these we could take the estimated mean of the data to be 3.367.
- This process can be used to estimate other quantities like the standard deviation and even quantities used in machine learning algorithms, like learned coefficients.

Bagging

- Bagging stands for bootstrap aggregation.
- It combines multiple learners in a way to reduce the variance of estimates.
- For example, random forest trains M Decision Tree, you can train M different trees on different random subsets of the data and perform voting for final prediction.
- Example:
  - Random Forest
  - Extra Trees.

Bagging


D
Original
Training data
Step 1:
Create Multiple
Data Sets
D1
D2
Dt-1
Dt
Step 2:
Build Multiple
Classifiers
C1
C2
Ct-1
Ct
Step 3:
Combine
Classifiers
C*

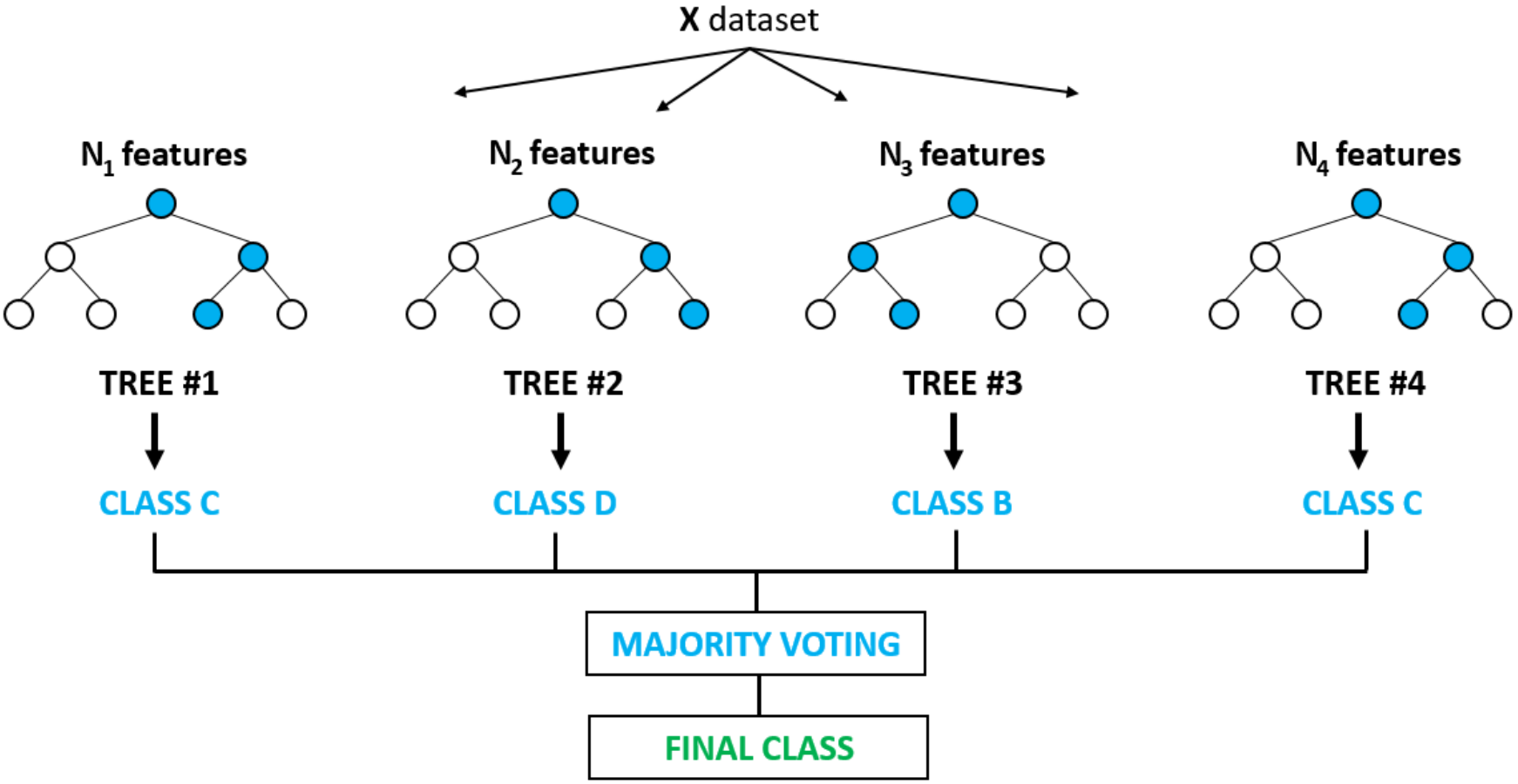
Random Forest

- Random forest is a type of supervised machine learning algorithm based on ensemble learning.
- Ensemble learning is a type of learning where you join different types of algorithms or same algorithm multiple times to form a more powerful prediction model.
- The random forest algorithm combines multiple algorithm of the same type i.e. multiple decision trees, resulting in a forest of trees, hence the name "Random Forest".
- The random forest algorithm can be used for both regression and classification tasks.

How it works ?

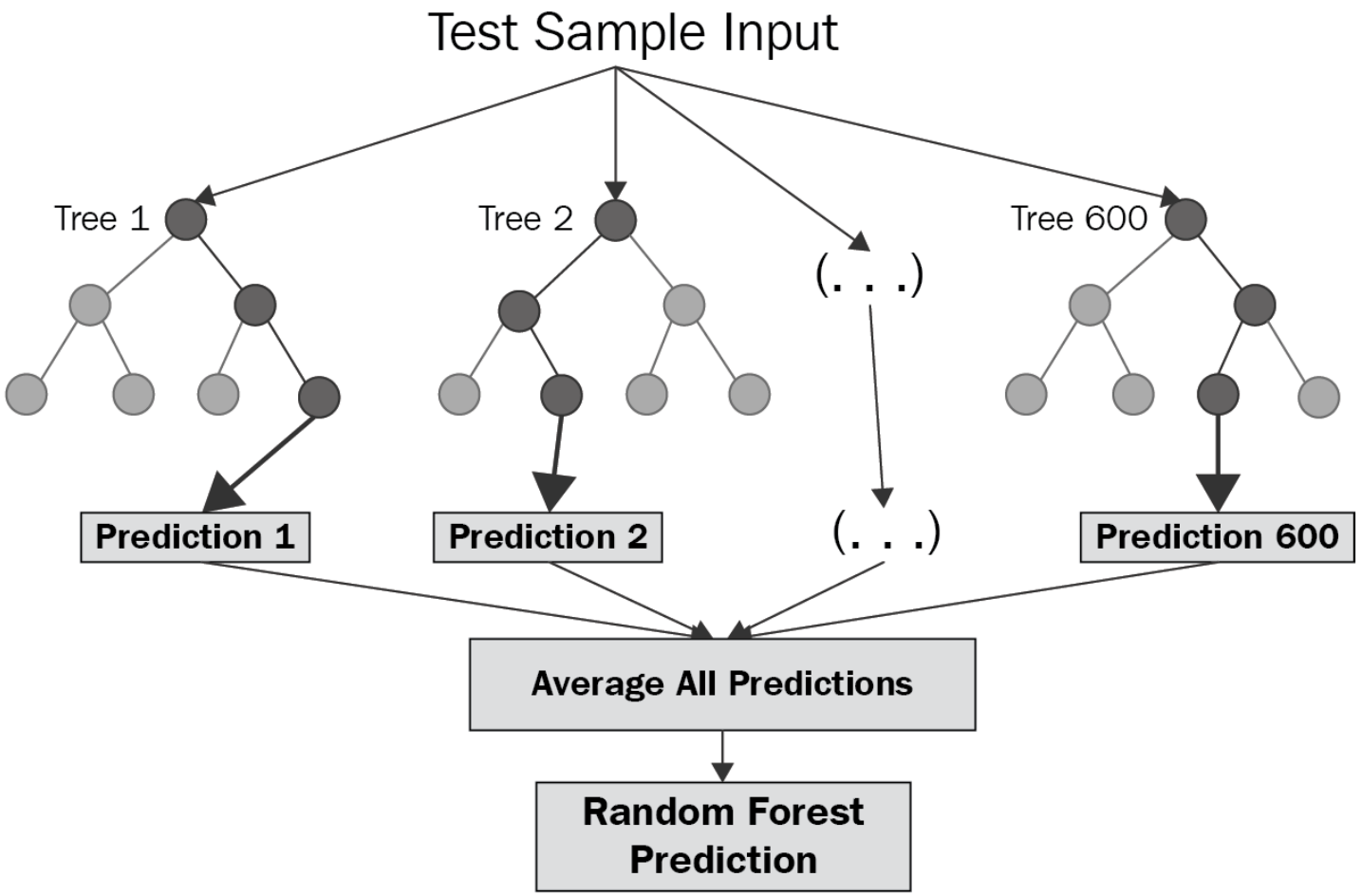
- Pick N random records from the dataset.
- Build a decision tree based on these N records.
- Choose the number of trees you want in your algorithm and repeat steps 1 and 2.
- In case of a regression problem, for a new record, each tree in the forest predicts a value for Y (output). The final value can be calculated by taking the average of all the values predicted by all the trees in forest. Or, in case of a classification problem, each tree in the forest predicts the category to which the new record belongs. Finally, the new record is assigned to the category that wins the majority vote.

Majority Voting



Source: medium.com

Regressor Output



Source: medium.com

Boosting

- Boosting algorithms are a set of the low accurate classifier to create a highly accurate classifier.
- Low accuracy classifier (or weak classifier) offers the accuracy better than the flipping of a coin.
- This is done by building a model from the training data, then creating a second model that attempts to correct the errors from the first model. Models are added until the training set is predicted perfectly or a maximum number of models are added.
- Highly accurate classifier(or strong classifier) offer error rate close to 0. Boosting algorithm can track the model who failed the accurate prediction.
- Boosting algorithms are less affected by the overfitting problem.

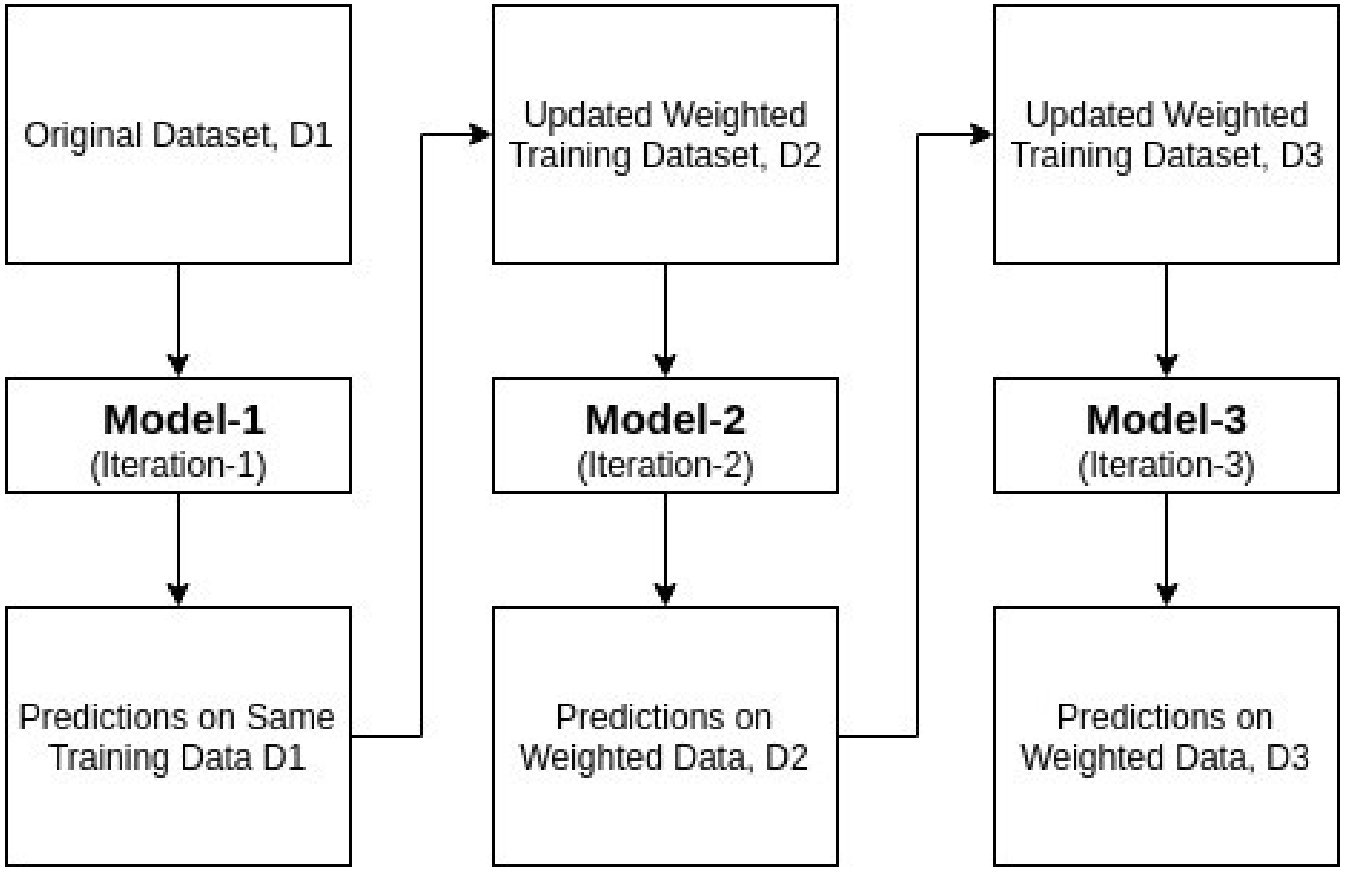
Boosting Models

- Models that are typically used in Boosting technique are:
  - XGBoost (Extreme Gradient Boosting)
  - GBM (Gradient Boosting Machine)
  - ADABoost (Adaptive Boosting)

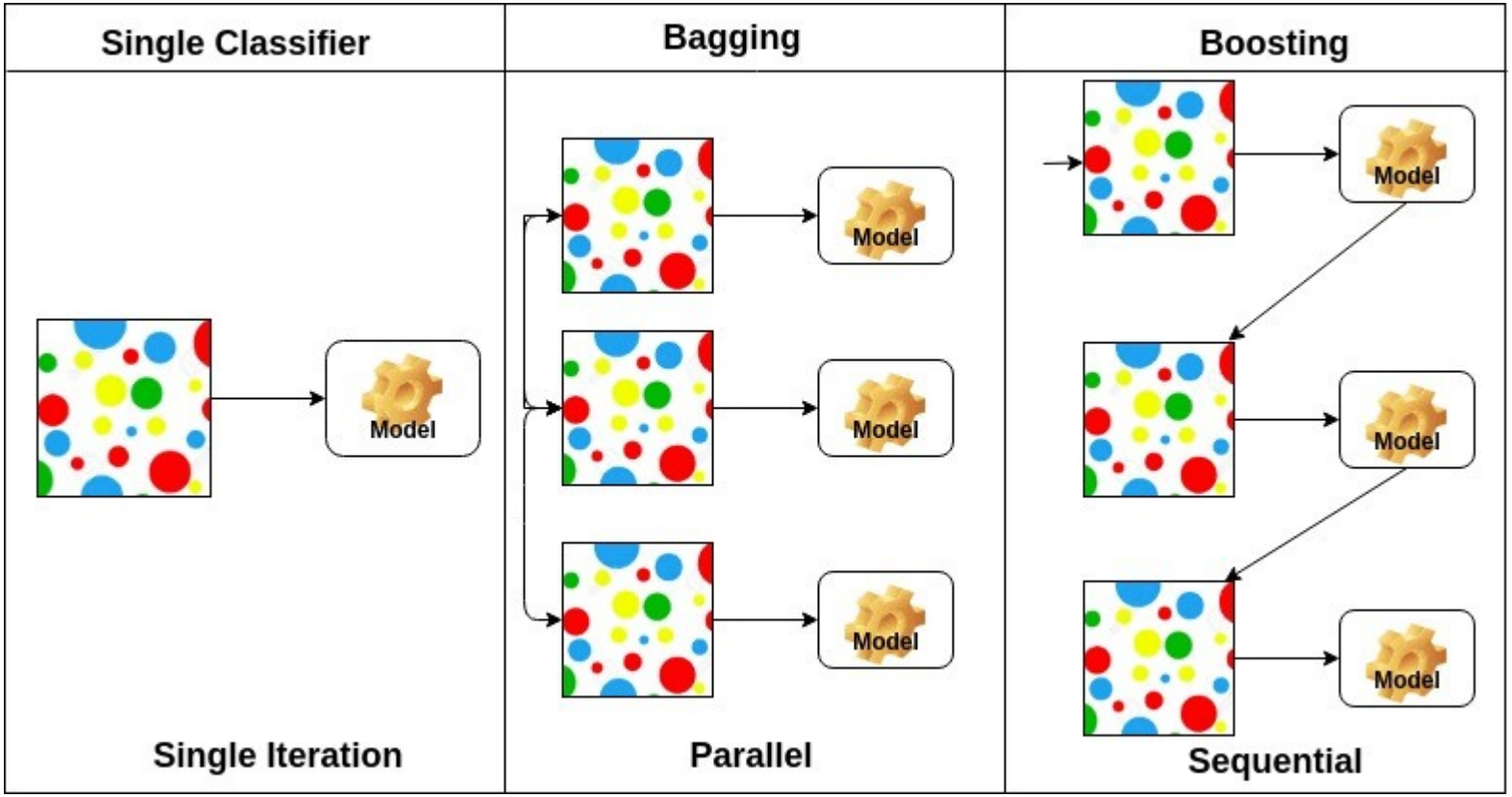
Adaboost Summary

- Initially, Adaboost selects a training subset randomly.
- It iteratively trains the AdaBoost machine learning model by selecting the training set based on the accurate prediction of the last training.
- It assigns the higher weight to wrong classified observations so that in the next iteration these observations will get the high probability for classification.
- Also, It assigns the weight to the trained classifier in each iteration according to the accuracy of the classifier. The more accurate classifier will get high weight.
- This process iterate until the complete training data fits without any error or until reached to the specified maximum number of estimators.
- To classify, perform a "vote" across all of the learning algorithms you built.

How Adaboost Works?



Comparison


Single Classifier
Bagging
Boosting
Model
Model
Model
Model
Model
Model
Model
Model
Single Iteration
Parallel
Sequential

Stacking

- Stacked Generalization or “Stacking” for short is an ensemble machine learning algorithm.
- It involves combining the predictions from multiple machine learning models on the same dataset, like bagging and boosting.
- Stacking addresses the question:
  - Given multiple machine learning models that are skillful on a problem, but in different ways, how do you choose which model to use (trust)?

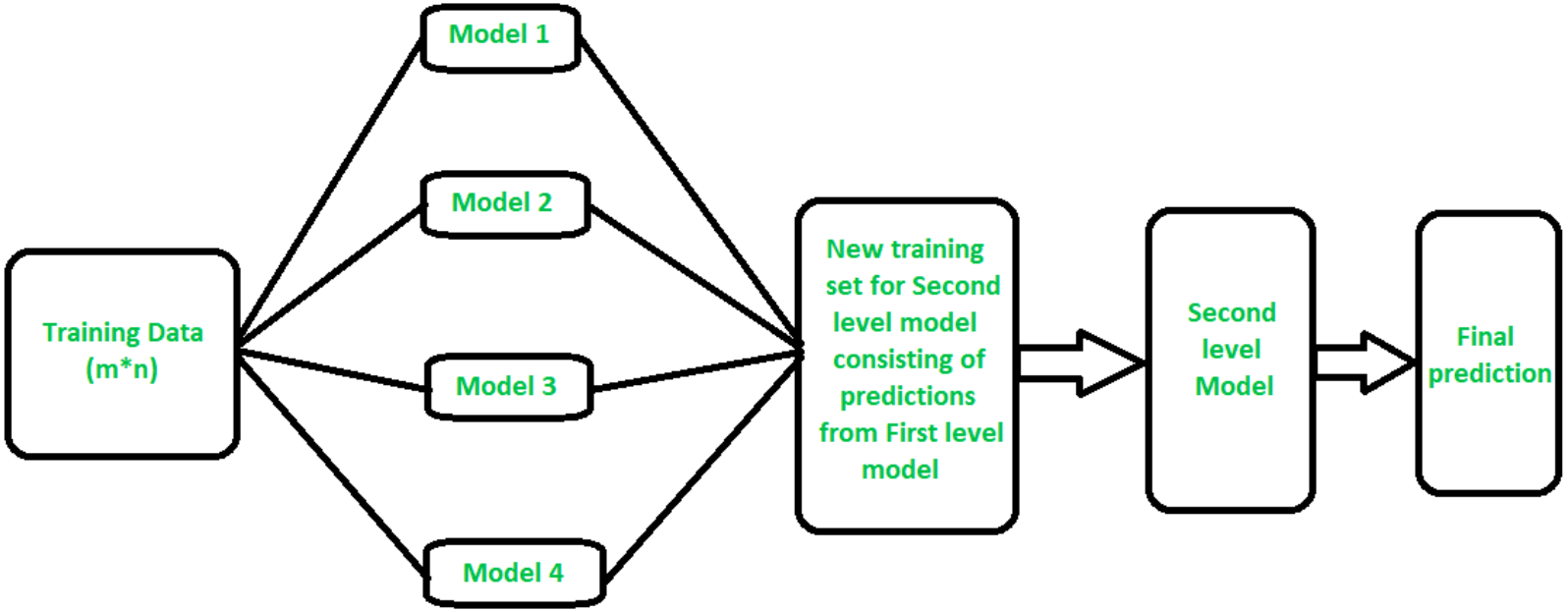
Stacking

- The approach to this question is to use another machine learning model that learns when to use or trust each model in the ensemble.
  - Unlike bagging, in stacking, the models are typically different (e.g. not all decision trees) and fit on the same dataset (e.g. instead of samples of the training dataset).
  - Unlike boosting, in stacking, a single model is used to learn how to best combine the predictions from the contributing models (e.g. instead of a sequence of models that correct the predictions of prior models).

Stacking

- The architecture of a stacking model involves two or more base models, often referred to as level-0 models, and a meta-model that combines the predictions of the base models, referred to as a level-1 model.
  - Level-0 Models (Base-Models): Models fit on the training data and whose predictions are compiled.
  - Level-1 Model (Meta-Model): Model that learns how to best combine the predictions of the base models.

Stacking


Model 1
Model 2
Model 3
Model 4
Training Data (m*n)
New training set for Second level model consisting of predictions from First level model
Second level Model
Final prediction

Stacking

- The meta-model is trained on the predictions made by base models on out-of-sample data.
- That is, data not used to train the base models is fed to the base models, predictions are made, and these predictions, along with the expected outputs, provide the input and output pairs of the training dataset used to fit the meta-model.
- The outputs from the base models used as input to the meta-model may be real value in the case of regression, and probability values, probability like values, or class labels in the case of classification.

Voting

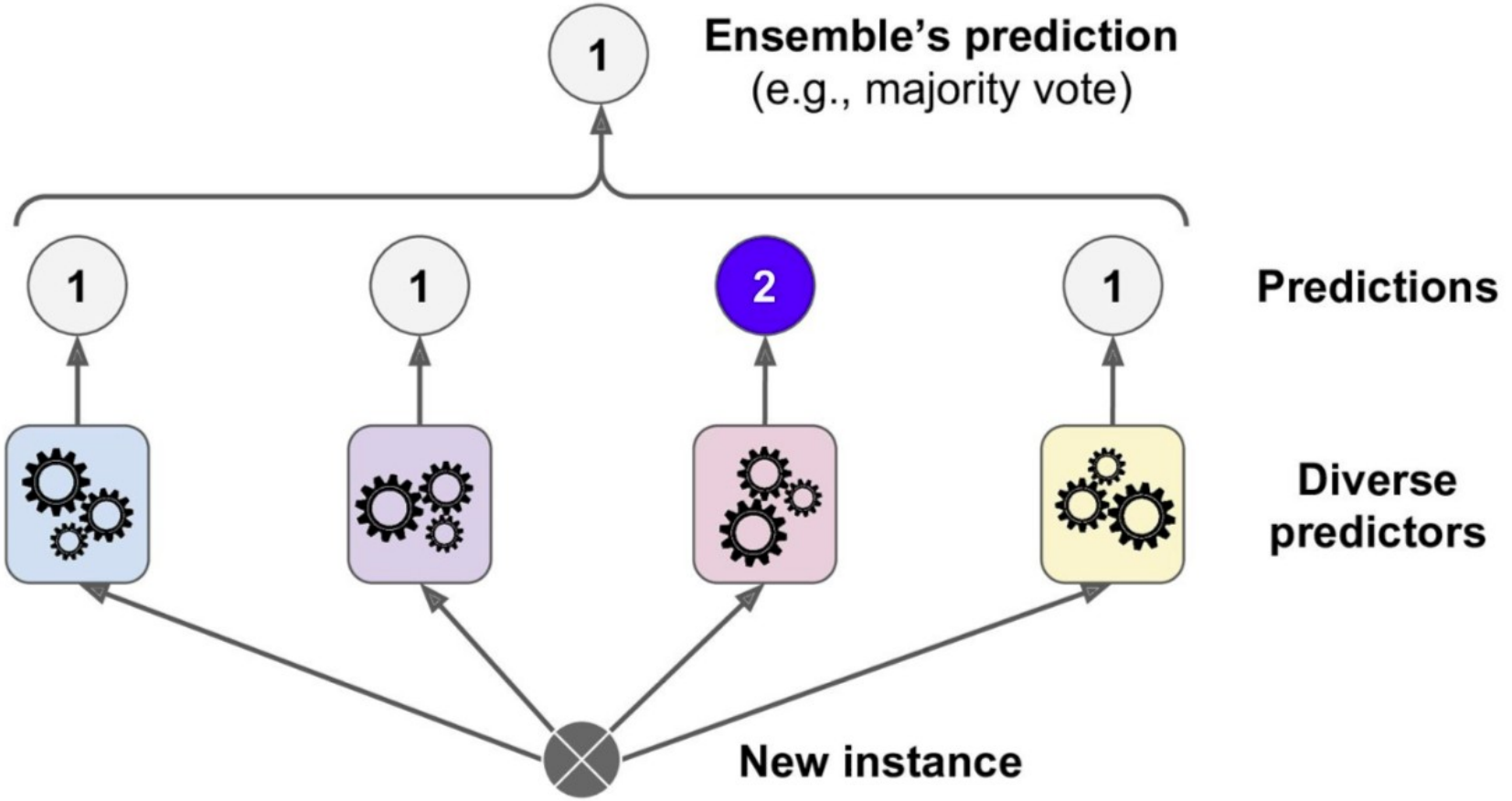
- Voting is an ensemble machine learning algorithm.
- For regression, a voting ensemble involves making a prediction that is the average of multiple other regression models.
- In classification, a hard voting ensemble involves summing the votes for crisp class labels from other models and predicting the class with the most votes.
- A soft voting ensemble involves summing the predicted probabilities for class labels and predicting the class label with the largest sum probability.

Voting

- A voting ensemble (or a “majority voting ensemble“) is an ensemble machine learning model that combines the predictions from multiple other models.
- It is a technique that may be used to improve model performance, ideally achieving better performance than any single model used in the ensemble.

  A voting ensemble works by combining the predictions from multiple models.
- It can be used for classification or regression.

Voting


1
Ensemble's prediction
(e.g., majority vote)
1
1
2
1
Predictions
Diverse predictors
New instance

Voting

- In the case of regression, this involves calculating the average of the predictions from the models.
- In the case of classification, the predictions for each label are summed and the label with the majority vote is predicted.
  - Regression Voting Ensemble: Predictions are the average of contributing models.
  - Classification Voting Ensemble: Predictions are the majority vote of contributing models.

Voting

- There are two approaches to the majority vote prediction for classification; they are hard voting and soft voting.
- Hard voting involves summing the predictions for each class label and predicting the class label with the most votes. Soft voting involves summing the predicted probabilities (or probability-like scores) for each class label and predicting the class label with the largest probability.
  - Hard Voting. Predict the class with the largest sum of votes from models
  - Soft Voting. Predict the class with the largest summed probability from models.

Voting

- Use voting ensembles when:
  - All models in the ensemble have generally the same good performance.
  - All models in the ensemble mostly already agree.
- Hard voting is appropriate when the models used in the voting ensemble predict crisp class labels. Soft voting is appropriate when the models used in the voting ensemble predict the probability of class membership.

Voting

- Soft voting can be used for models that do not natively predict a class membership probability, although may require calibration of their probability-like scores prior to being used in the ensemble (e.g. support vector machine, k-nearest neighbors, and decision trees).
  - Hard voting is for models that predict class labels.
  - Soft voting is for models that predict class membership probabilities.
- The voting ensemble is not guaranteed to provide better performance than any single model used in the ensemble.

Voting

- Use a voting ensemble if:
  - It results in better performance than any model used in the ensemble.
  - It results in a lower variance than any model used in the ensemble.
- A voting ensemble is particularly useful for machine learning models that use a stochastic learning algorithm and result in a different final model each time it is trained on the same dataset.
- One example is neural networks that are fit using stochastic gradient descent.

Useful resources

- www.mitu.co.in
- www.pythonprogramminglanguage.com
- www.scikit-learn.org
- www.towardsdatascience.com
- www.medium.com
- www.analyticsvidhya.com
- www.kaggle.com
- www.stephacking.com
- www.github.com

Thank you

This presentation is created using LibreOffice Impress 5.1.6.2, can be used freely as per GNU General Public License

@mitu_skillologies

/mITuSkillologies

@mitu_group

/company/mitu-skillologies

c/MITUSkillologies

Web Resources
https://mitu.co.in
http://tusharkute.com

contact@mitu.co.in

tushar@tusharkute.com