UNIT-5-INTRODUCTION TO ENSEMBLE LEARNING

MADE BY: Saanvi Dhakane

- **What is Ensemble Learning?**

1. Ensemble Learning is a technique in machine learning that involves combining multiple models (also known as **weak learners or base learners**) are combined to form a strong model (also called a **strong learner**) to solve a problem more accurately.

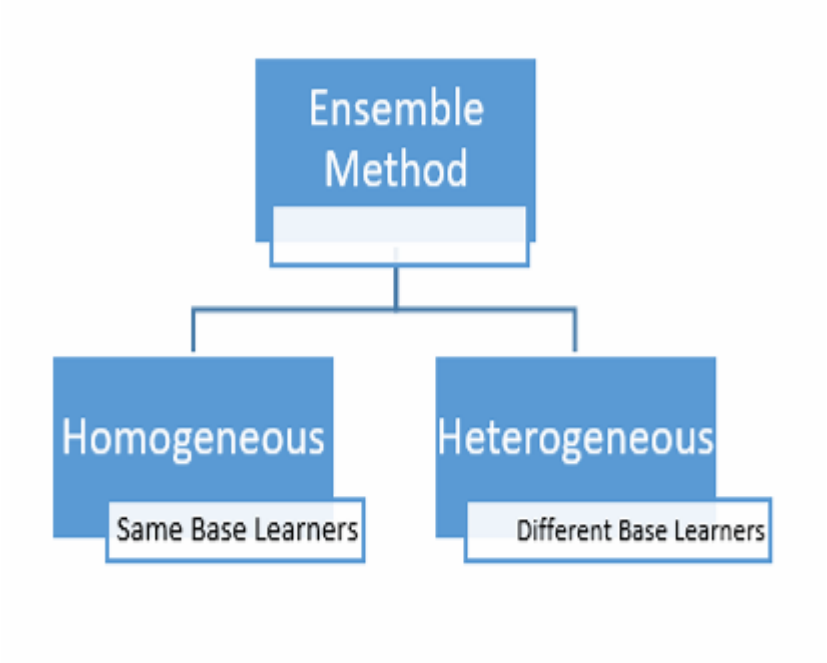
   Think of it like asking multiple friends for advice and making a decision based on what most of them say – this is more reliable than asking just one.

2. In this concept, several models are trained using machine learning algorithms. The prediction of each weak model is combined to get the final prediction.

- **Types of Ensemble Learning**:

Based on the type of base learners, we can classify ensemble learning methods into 2 parts:

- **Homogenous Ensemble Method**- If the base learners are same then its homogenous.
- **Heterogenous Ensemble Method**- If the base learners are different then its heterogenous.



- **Why is Ensemble Learning Used?**

1) **To Reduce Bias and Variance –** High bias is the difference between the predicted value and actual value by the model. Bias is introduced when the model doesn't consider the variation of data and creates a simple model. The simple model doesn't follow the patterns of data, and hence the model gives errors in predicting training as well as testing data i.e. the model with high bias and high variance.

2) **To Improve Accuracy –** High bias causes underfitting which results model not remembering patterns whereas high variance results in model not being able to generalize. So, when ensemble methods are used the bias-variance issue is solved and overall accuracy of the model is improved.

- **ADVANTAGES –**

    1) **Improves Accuracy –** By solving the bias and variance problem , accuracy of model is improved.

    2) **Makes the model more robust and generalized -** Ensemble learning combines multiple models to perform reliably across different types of data, even unseen ones.

3) **Handles complex data efficiently -** It captures intricate patterns in the data that a single model might miss by combining different learning strategies.

4) **Reduces the risk of both underfitting and overfitting**: By balancing bias and variance, ensemble methods avoid overly simple or overly complex models.

- **DISADVANTAGES –**

1) **Expensive Computational Cost –** Requires more time, memory and processing power.

2) **Hard to interpret –** As we combine several models, Ensemble learning is complex to implement.

3) **Slower Predictions –** Several models are involved so its time-consuming, especially for real time applications.

- **Ensemble Learning Methods -**

- **Ensemble techniques are classified into three types:**

1. Bagging

2. Boosting

3. Stacking

1] BAGGING

What is Bagging?

1) Bagging is also known as bootstrap aggregation. It is basically one of the ensemble techniques that improve the accuracy of model by reducing variance.

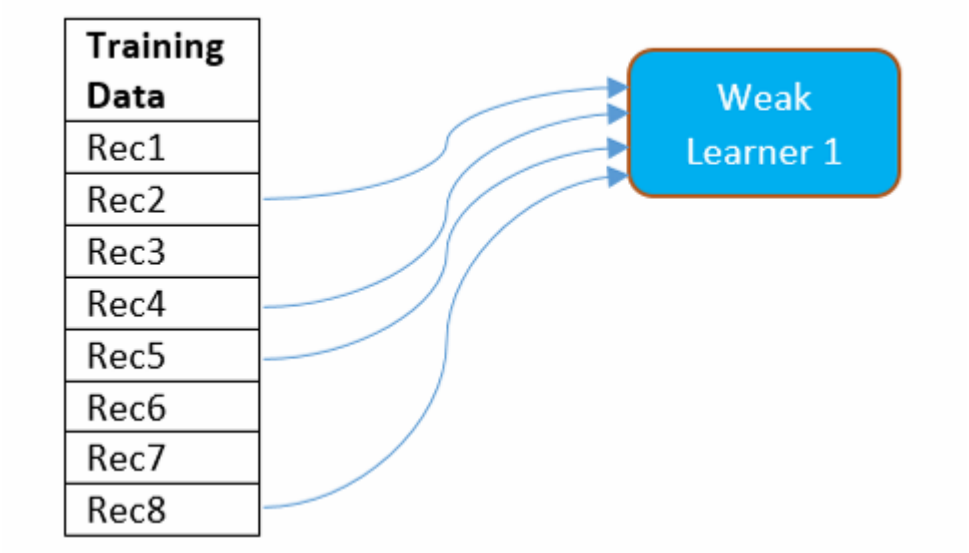
2) In simple words, bagging is a parallel method in which the final output is the average of all the output produced by each weak learner. Each weak learner has equal say and gives us accurate and reliable output with reduced variance.

- **STEPWISE WORKING –**

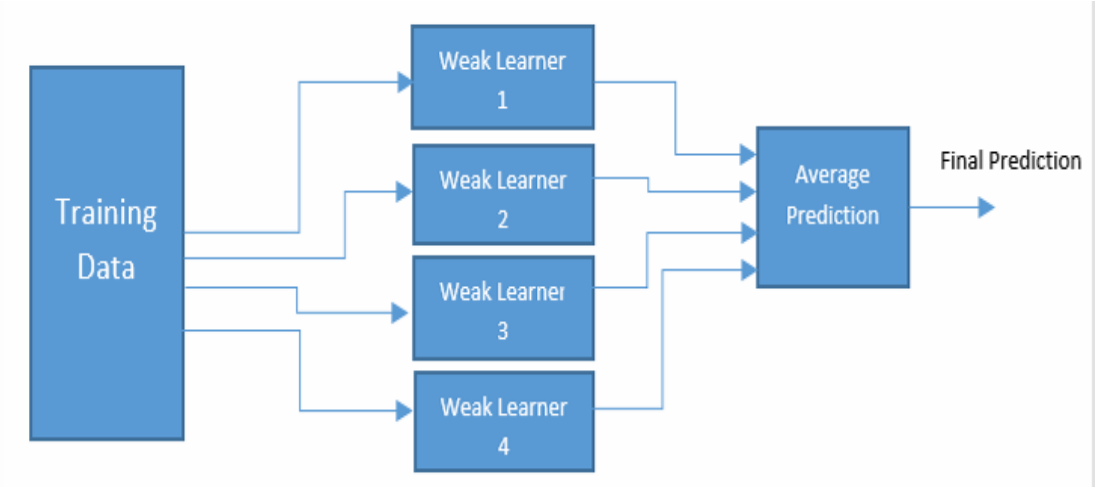
  1) Bootstrapping the sample – We take the original training dataset. We randomly select samples by **replacement method** and create new dataset. Each new dataset is called a **bootstrap sample**.

  (**Replacement method** means when samples are selected from the original dataset to create a new one then some samples might get repeated while some might not appear at all in the new dataset.)

  Example: If you have 8 samples, each weak learner might get 5 samples — some of them could be repeated.



2) **Training the Weak learners –** We train all the models that is the weak learners on each bootstrap sample. All the models are trained independently and parallelly.



Example: Each decision tree in the Random Forest is trained on a different bootstrap sample of the original dataset.
During training, each tree also considers only a random subset of features at each split to increase diversity.

3) **Aggregate the Results –**

- **For Classification:** We take majority vote from all the models.

- **For Regression:** We take average of all model outputs.

4) **Final Prediction –** The result that we get is the final output of the bagging model. The averaged output reduces variability but does not reduce bias though. Since final prediction is an average of output of each weak learner, it means that each weak learner has equal say or weight in the final output.

Real-World Example:

Random Forest is the most popular algorithm that uses bagging. It builds multiple decision trees on different bootstrap samples and takes the majority vote.

WHY BAGGING?

Bagging (Bootstrap Aggregation) helps to reduce variance, meaning the model becomes less sensitive to noise and more robust, especially when using high-variance algorithms like decision trees.

- **ADVANTAGES –**

1) Time Saving due to Parallel running of Models.
2) Prevents overfitting.
3) Robust to noise and outliers**.**
4) Improves accuracy by reducing variance**.**
5) Ideal for models that are sensitive to data changes.

- **DISADVANTAGES –**

1) Increased Computational Cost.
2) Hard to Interpret and Explain.
3) If the base learners are too complex and the data is noisy, bagging can still be overfit to that noise despite its variance reduction capability.
4) Multiple models increase memory requirements.

5) Bagging mainly reduces variance. If the base learner already has low variance (e.g., linear models), bagging offers little improvement.

2] BOOSTING

We saw that in bagging every model is given equal preference, but if one model predicts data more correctly than the other, then higher weightage should be given to this model over the other. Also, the model should attempt to reduce bias. These concepts are applied in the second ensemble method that we are going to learn, that is Boosting.

1) Basically, Boosting is machine learning method that combines multiple weak models in sequence to form a strong learner. Each new model corrects the errors of the previous models to achieve peak accuracy.

- **STEPWISE WORKING –**

1) In the start, boosting assigns equal weights to all the samples as all of them are equally important.
For Example- If there are N number of samples then, it assigns weight 1/N to each sample.

2) All the models are run sequentially. The weak learner classifies data. It classifies some samples correctly while some incorrectly.

3) After each classification, the sample weights are changed. The weight of each correctly classified sample is reduced while the weight of incorrectly classified sample is increased. Then, the next weak model is run. This step focuses on fixing the errors made by the previous models.

4) This process is continued till all the model give strong and accurate predictions.

5) The outputs of all the weak learners are combined into a strong model via weighted voting.

Real World Example –
In disease prediction (e.g predicting diabetes), a single weak model might focus only on blood sugar levels. Boosting combines multiple models that each focus on different risk factors—like age, BMI, and family history—to build a strong model with high accuracy.

- **WHY BOOSTING?**
  Boosting is used to improve accuracy of the model even better as it learns from the mistakes of its previous models and corrects them. It also reduces bias along with variance. It also handles complex patterns in data better than standalone models.

- **ADVANTAGES –**

  1) Higher accuracy as variance and bias both are reduced.
  2) Focuses on errors to rectify it.
  3) No need for Feature-scaling.
  4) Can handle categorical, numerical and mixed data types.
  5) Flexible and versatile.

- **DISADVANTAGES –**

  1) Slower learning as training happens sequentially.
  2) Sensitive to noise – can easily overfit due to outliers' presence.
  3) High memory usage.
  4) Complex to implement from scratch.
  5) Slower predictions and less effective on smaller datasets.

(Adaboost algorithm is an ensemble learning method used in binary classification only)

- **HOW DOES ADABOOST ALGORITHM WORK?**

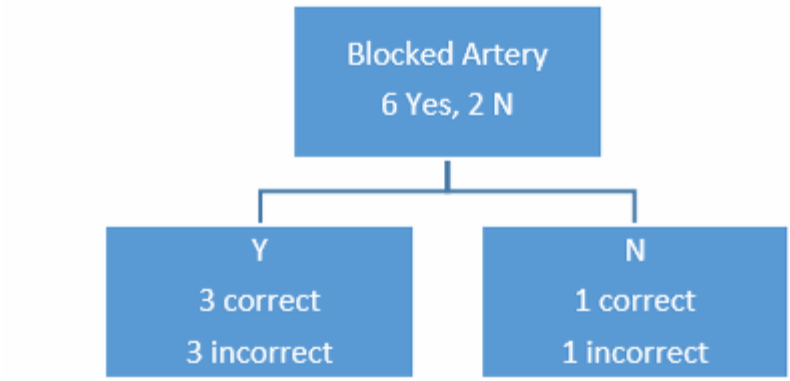
Blocked Arteries	Chest Pain	Weight	Heart Disease
Y	Y	200	Y
Y	N	185	Y
N	Y	200	Y
Y	Y	160	Y
Y	N	140	N
Y	N	130	N
N	Y	170	N
Y	Y	170	N

1) **Initialise Weights -** Let's take a heart disease training dataset to train the model. We will initialise equal weights to all samples as all samples are equally important in the start**. Sum of the weights of all the samples is always 1.** Here, there are total 8 samples. So, weight assigned to each sample is 1/8 = 0.125.

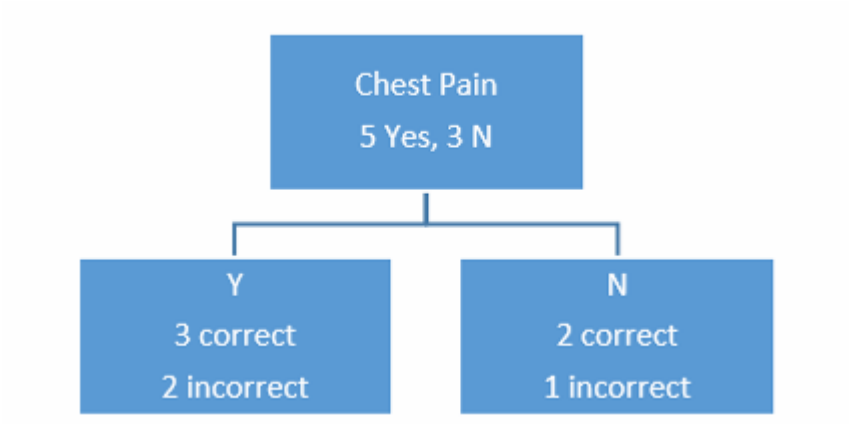
2) **Create Stumps –** Stump is a decision tree with one node and two leaves. Adaboost creates forest of decision stumps. To create stump, only one attribute should be chosen. But it is not randomly selected. The attribute that does the best job of classifying the sample is selected first.

   So let's see how each attribute classifies the samples:

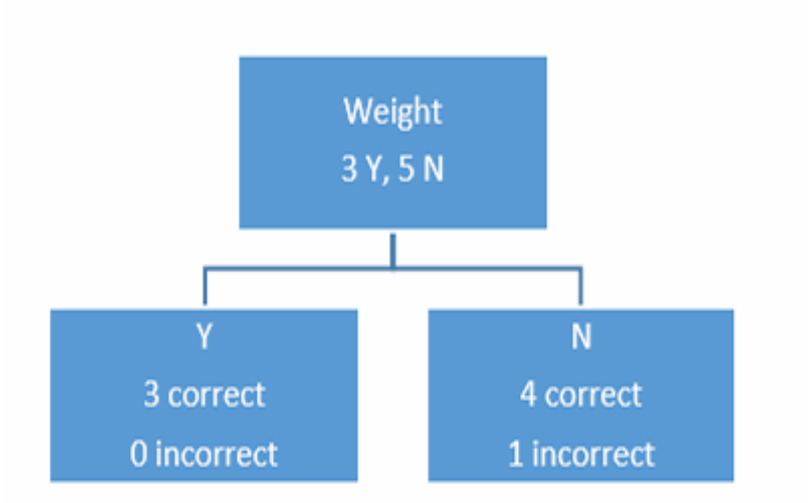
- **Blocked Artery:**



- **Chest Pain:**



- **Weight:**



From above classification, it can be seen that:

- 'Blocked Arteries' as a stump made **4** errors.
- 'Chest Pain' as a stump made **3** errors.
- 'Weight' as a stump made **1** error.

Since, the weight attribute has the least number of errors, it is the best stump to start with.

3) Calculate Total Error And Voting Power (Amount Of Say) Of The Stump-

After choosing best attribute for stump, next step is to calculate the voting power (amount of say) of that stump in the final classification.

Formula to calculate amount of say is:

$$\text{Amount of say} = \tfrac{1}{2} * ln((1-\text{Total Error}) / \text{Total Error})$$

Where ln is Natural Log Total Error = **Number of misclassified samples * Weight of sample.**
Since the stump misclassified only 1 sample,
Total Error made by stump **= weight of 1 sample = 1/8 = 0.125** Substituting Total Error in the equation,
Amount of say = ½ * ln ((1-0.125)/0.125) = 0.9729

4) **Start to build a Final Classifier Function –** The Adaboost classifier then starts building a function that will assign this 'amount of say' as voting power of the stump.
   It will assign 0.9729 as voting power to the stump created for 'Weight' attribute. **The function h(x) = 0.9729* 'Weight' as stump**.

5) **Updating the Weights of the samples-** After calculating the amount of say, next step is to create a new stump. But before that we assign new weights to each sample. Samples that are correctly classified are given less importance while samples that are incorrectly classified are given more importance.

The formula to assign new weight for **correctly classified** samples is:

$$W_{new} = W_{old} / 2(1\text{-Total Weight})$$

The formula to assign new weight for **incorrectly classified** samples is:

$$W_{new} = W_{old} / (2*\text{Total Weight})$$

New weights of correctly classified samples =

Wnew = (0.128) / 2(1-0.125) = 0.07

New weights of incorrectly classified samples =

Wnew = (0.125) / (2*0.125) = 0.5

Substitute new weights:

Blocked Arteries	Chest Pain	Weight	Heart Disease	Weights Assigned
Y	Y	200	Y	0.07
Y	N	185	Y	0.07
N	Y	200	Y	0.07
Y	Y	160	Y	0.5
Y	N	140	N	0.07
Y	N	130	N	0.07
N	Y	170	N	0.07
Y	Y	170	N	0.07

From above classification, it can be seen that –

- 'Blocked Arteries' as a stump made 4 errors. Therefore, Total Error for 'Blocked Arteries' = 0.07*4 = **0.28**
- 'Chest Pain' as a stump made 3 errors. Therefore, Total Error for 'Chest Pain' = 0.07*3 = **0.21**
- 'Weight' as a stump made 1 error. Therefore, Total error for weight = 1*0.5 = **0.5.**

6) **Calculate Voting Power and add it to the Final Classifier Function-**

As the attribute Chest Pain has the least total error, we will choose it as our second stump.
The Amount of Say = ½ ln(1-0.21)/0.21 =0.66
Add this classifier with its weight to the function.
The function h(x) = 0.9729* 'Weight' as stump + 0.66*'Chest Pain' as stump

7) **Assign New Weights and Repeat** - Again, assign new weights to the correctly classified samples with formula:
Wnew = Wold / 2(1-total weight)
And incorrectly classified samples with the formula:
Wnew = Wold / (2*total weight)

We repeat this process until:
1. Enough rounds of this process are done to create strong classifier.
2. No good classifiers left for making predictions.

When it comes to classifying data point, even if one classifier misclassifies a point, the voting powers of other classifiers override it making strong predictions. There is another method to train Adaboost algorithm. In this method, new sample dataset is created in every iteration. The sample dataset is of same size as original dataset, and it contains some samples that are repeatedly selected. Samples with higher weights (i.e. samples that are incorrectly classified in previous round), will be selected repeatedly. The weight vector is normalized.

To summarize:
1. Adaboost is a sequential method where one weak learner runs after other.
2. Weak learners are stumps i.e. decision tree with 2 nodes.

3. Voting power is assigned to stumps based on how correctly it classifies the data.
4. Boosting reduces bias.

- **OTHER BOOSTING ALGORITHMS:**

  Apart from Adaboost, there are other boosting algorithms too like- **Gradient Boosting Machine (GBM).**

  1.The main idea behind this algorithm is that it creates decision trees with smaller leaves and scales it with a learning rate.

  2.Decision trees lead to overfitting and often produce high variance.
  Hence, the models' prediction is multiplied by a small value called the
  Learning rate.
  This value helps the model to learn slowly and steadily also while making it prone to overfitting.

  3. This method slows the training period. Also, reduces the training accuracy but provides excellent test accuracy in the long run.

  4. GBM is used for both classification and regression problems.

  3] STACKING
  What is Stacking?

- **Definition-** It is an ensemble learning technique that combines predictions from multiple different models (like logistic regression, SVMs, decision trees). The outputs of these models are combined and given as input to the final model (also known as **meta-learner**). The final model makes the overall prediction.

This method improves accuracy very much as it leverages that is combines the strengths of each model individually.

- **WHY STACKING USED?**

1) This method combines smart guesses from various models and makes a better final guess.
2) Stacking also balances strength which means if a model is weak in an area, then the other model can cover for it.
3) It works well where the data is tricky, and no model is perfect.
4) The final model learns which model is good in which area.
5) The model learns from its mistakes and rectifies it.

- **STEPWISE WORKING –**

1) **Training models –** Firstly, we train the base models (like decision tree, SVMs) on training data.

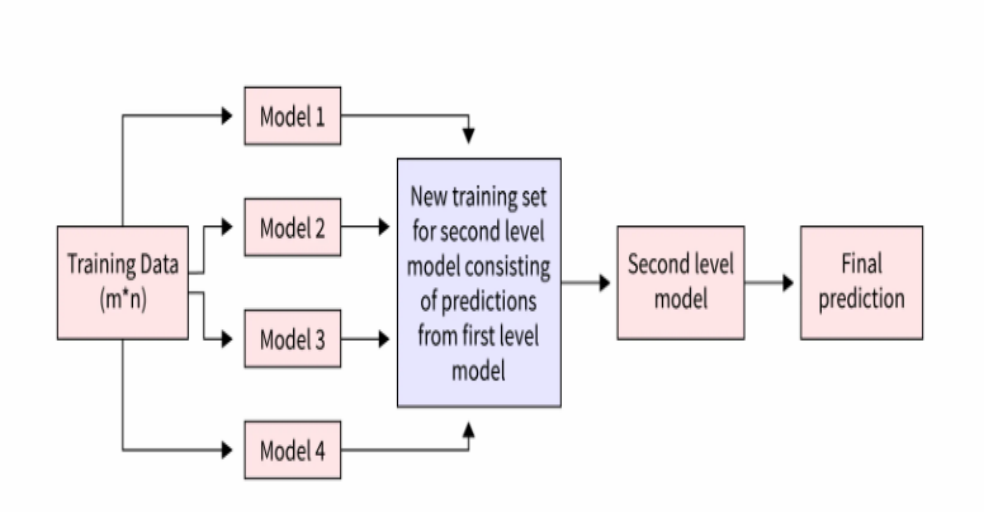
2) **Make predictions on Validation data –** Then, the models make predictions on that part of the data that has not been used for training yet (it is called validation data). This data is used for tuning parameters and is very different from test data.

3) **Collection of Predictions as Input data –** Instead of using original features, we use predictions from the base models as out input data.

4) **Training of the Final Model –** A new model (like a linear regression model) is used. It is trained on these predictions.

   This model learns how to best combine the outputs of the base models to get a better final prediction.

5) **Make Final Prediction -** In the end, when you give new data, each base model predicts, and the meta-model combines them to make the final decision.



- **Real-life example:**
  Student exam—
  1) **Training of student:** In this phase, the student learns the concept and practices problems here.
  2) **Validation Data:** In this phase, the student gives mock tests and exams to see how well they are prepared.
     They may adjust their study patterns or style (**tuning**) based in the result.
  3) **Test Data:** Here the student gives his or her unseen final exam. That decides its final performance result.

- **ADVANTAGES –**

  1) Flexible- allows use of any kind of ML models.
  2) Works well for complex datasets.
  3) Improves generalisation to unseen data.
  4) Supports diversity of learning algorithms.
  5) Popular in competitions (eg Kaggle).

- **DISADVANTAGES –**

  1) Needs careful choice of meta-model.
  2) Difficult to debug and troubleshoot errors.

3) More data preprocessing may be required.
4) Performance gain not guaranteed always.
5) Slower prediction time.

- **VOTING CLASSIFIERS-**

1) **Definition-** It is an ensemble learning method that combines the outputs of all base models (called base estimators) to give us the final prediction. It's like a "majority wins" in a group decision.

2) Basically, each model gives its prediction (like a vote), and a voting classifier combines them to form a final output, like a majority wins situation.

- **WHY ARE THEY USED?**

No single model is perfect. Different models perform well on different parts of data so, when a voting classifier combines the outputs of each base model together, the chances of getting a more accurate and precise prediction are high. It improves accuracy by learning from multiple models, not just one.

- **TYPES OF VOTING CLASSIFIERS:**
  - **i.** Hard classifiers.
  - **ii.** Soft classifiers.

1] HARD VOTING:

Hard voting is a type of voting classifier in which each model predicts a class label. The final prediction is the class having most votes. Hard

voting does not consider how confident a model is- only the predicted label matters.

Hard voting should only be used when all the models are reliable.

For Example- Let's say classifiers predicted the output classes as (Cat, Dog, Dog). As the classifiers predicted class "dog" a maximum number of times, we will proceed with Dog as our final prediction.

2] SOFT VOTING:

Soft voting is a type of voting classifier in which each model predicts the **probability** of each class instead of just the class label. The final prediction is made by **averaging** the predicted probabilities and selecting the class with the **highest average probability**.

Soft voting considers how confident each model is in its prediction. It works best when models can give probability scores and when they have different strengths.

For Example- Let's say classifiers predicted the probabilities for classes (Cat, Dog) as:

- Classifier 1: (Cat: 0.4, Dog: 0.6)
- Classifier 2: (Cat: 0.3, Dog: 0.7)
- Classifier 3: (Cat: 0.2, Dog: 0.8)

Average probabilities:

- Cat: (0.4 + 0.3 + 0.2) / 3 **= 0.3**
- Dog: (0.6 + 0.7 + 0.8) / 3 **= 0.7**

Final prediction: Dog, because it has the highest average probability.

BAGGING BOOSTING STACKING SIMILARITIES AND DIFFERENCES

- Similarities:

1) **Ensemble Learning** – Combine multiple models.
2) **Supervised** – Require labelled data.
3) **Base Learners** – Use multiple weak/strong models.
4) **Classification & Regression** – Support both tasks.
5) **Improves Accuracy** – Boosts model performance.
6) **Generalization** – Reduce error on unseen data.
7) **Model Combination** – Merge outputs (vote/average/meta).
8) **Overfitting Control** – Enhance robustness.
9) **Data-Driven** – Learn from training data.
10) **Widely Used** – Popular in real-world ML.

- Differences:

Feature	Bagging	Boosting	Stacking
Model Training	Parallel	Sequential	Parallel (base), then meta
Focus	Reduces variance	Reduces bias	Combines diverse models
Data Sampling	Bootstrapped subsets	Weighted samples	Often different (mix)
Model Type	Usually same (e.g., trees)	Learns from errors	Learns how to combine models
Error Handling	Averages predictions	Learns prone	Depends on meta-model
Complexity	Less prone	Weighted Voting	Meta-model
Example	Voting / Averaging	Moderate	Blending, Super Learner
Training Time	Random Forest	AdaBoost, XGBoost	Slowest (2-stage process)