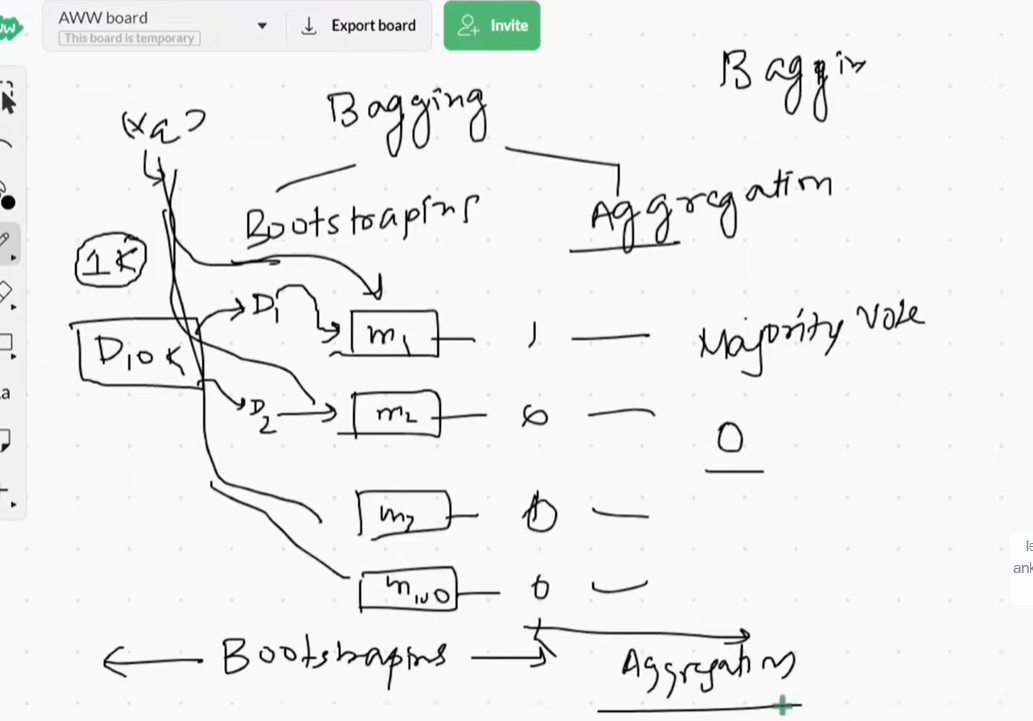
*Bagging*

**It is made up of using 2 Words Bootstrapping & Aggregation**

**Yaha Model jobhi lenge hum 1 Hie Model lenge**

**Har Model alag Random Sampling krke Data pe Train karenge**

**Model 1 , 2,3 sbhpe Random Data Sampling krke Train krenge**

****

There is a Dataset of thousand rows

& there are 3 Models

Then what we do We Takeout 500 Random Data from Main Dataset

& give it to Model 1

Then we do same things with all the remaining 2 Models to

We just take out Random Data & give them both

So because of this there is a Huge Variation of Data for the Model

& then We give them a Query point or unseen Data to Predict or Classify about it

Then we get x result From all the 3 models & now what we do is ,

If it is Classification then we apply Majority Count & jiska Majority is higher lyk whether it is yes or no if there is 2 yes & 1 No then we will take Result as 2

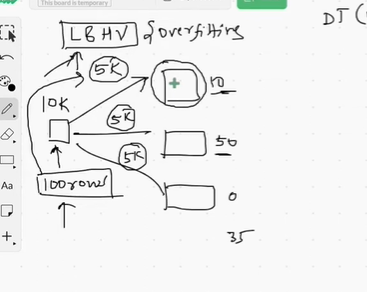
If it is Regression then we take out Mean as a result.

This is How Bagging Works

**Why we use it?**

We use it when ML algo jo hai vo Low Bias & High Variance hoo

**E.g Decision Tree, KNN, SVM 🡪 yeahsbh Low Bias & High Variance hai sbhmai Overfitting ki Problem aati hai.**

****

Scene kya horha hai jo Impact hai Inputs ka Vo Divide hojaraha hai Differently Model mai

Variation aaraha hai & as bhaut saare set of Models hai

So Model ko zyaada farak nahi aarha

That’s why at the end variance bhi Low hojaraha hai

In scikit-learn, a popular machine learning library in Python, the default base estimator used in the BaggingRegressor is indeed a DecisionTreeRegressor. The BaggingRegressor is a type of ensemble learning technique that combines multiple base estimators to create a more robust and accurate predictor.

Bagging stands for "Bootstrap Aggregating." It works by creating multiple subsets of the original training data through random sampling with replacement. Each subset is used to train a separate base estimator (in this case, DecisionTreeRegressor), and then the predictions from all the base estimators are aggregated to make a final prediction.

While DecisionTreeRegressor is the default base estimator, you can customize it by specifying a different base estimator using the **base\_estimator** parameter in the **BaggingRegressor** constructor. This allows you to use other regression algorithms as base estimators if you believe they might work better for your specific problem

**Similar to BaggingRegressor, the BaggingClassifier in scikit-learn also uses a default base estimator, which is the DecisionTreeClassifier. The BaggingClassifier is an ensemble learning method that aims to improve the performance and stability of classification algorithms by combining the predictions of multiple base classifiers.**

**The process of bagging with a classifier is quite similar to that of a regressor:**

**1. Create multiple subsets of the original training data through random sampling with replacement.**

**2. Train a separate base classifier (typically a DecisionTreeClassifier) on each subset.**

**3. Aggregate the predictions from all base classifiers to make a final prediction, often using majority voting for classification tasks.**

**As with the BaggingRegressor, you can customize the base estimator used in the BaggingClassifier by specifying a different classifier using the `base\_estimator` parameter when creating the BaggingClassifier instance. This allows you to use alternative classification algorithms as base estimators, such as random forests or support vector machines, depending on your problem's characteristics and performance requirements.**

**E.g:**

bag = BaggingClassifier(

base\_estimator= DecisionTreeClassifier(),

n\_estimators = 500,

max\_samples = 0.25,

bootstrap = True,

random\_state = 42

)

The code you provided is an example of using the `BaggingClassifier` from the scikit-learn library in Python. This classifier is an ensemble method that combines multiple instances of a base estimator (in this case, `DecisionTreeClassifier`) to improve predictive performance. Here's a breakdown of the code:

1. `bag = BaggingClassifier(`: This initializes an instance of the `BaggingClassifier`.

2. `base\_estimator= DecisionTreeClassifier(),`: Specifies the base estimator to be used for building each individual model within the ensemble. In this case, it's `DecisionTreeClassifier()`.

3. `n\_estimators = 500,`: Sets the number of base estimators to create in the ensemble. The value here is 500, meaning 500 Decision Tree classifiers will be used.

4. `max\_samples = 0.25,`: Determines the maximum number of samples to be used for training each base estimator. Here, `0.25` indicates that each base estimator will be trained on a random 25% of the training data.

5. `bootstrap = True,`: Specifies whether to use bootstrapping when selecting samples for training each base estimator. Bootstrapping involves randomly sampling with replacement from the training data.

6. `random\_state = 42`: Sets the random seed for reproducibility of results. Using the same seed ensures that the same random selections are made each time the code is run.

Overall, this code creates a BaggingClassifier that leverages 500 Decision Tree classifiers. Each Decision Tree is trained on a random 25% of the training data using bootstrapping. This ensemble method can help improve the model's performance by reducing overfitting and variance.

**& We Can use GridSearch CV to make us know which are the best Possible Great Hyperparameters**

**Bagging Tips**

1. Bagging Generally Gives better result than Pasting

Row Sampling krneke as 2 tarike hai Bagging, Pasting, with replacement , without replacement .

1. Good Results come around 0.25 to 0.5 row sampling mark, (this also can vary)
2. Random Patches and subspaces should be used while dealing with High Dimensional data
3. To find the correct Hyperparameter values can we do GridSearchCV/ RandomSearchCV