Ensembles and its techniques

① data
↳ Model ! → train → Prediction

② data
→ M1
→ M2
→ M3
→ M4
→ M5
→ Combine → prediction

Analogy
↳ DS jobs
→ Mentor-2 ML
→ Mentor-1
Ajay → data Analytics
Vishwas ML
ML

→ One person might give you wrong advice
→ you will connect to multiple mentors
→ Chances of getting wrong is minimized when you connect to multiple person.

Ensembles : Combine multiple models
: Prediction → more stable and accurate as compared to individual model.

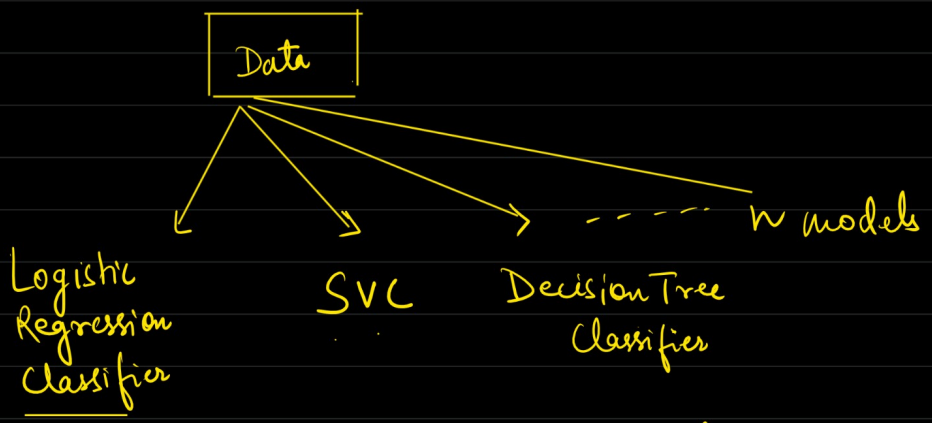
① (of same Algorithm)
- DT_1 – (max depth - 5)
- DT_2 – (max depth - 10)
- DT_3 – (max depth - 12)

② (different Algorithm)
- → Logist Reg
- → SVC
- → DTC

* Ensemble : Not necessarily only one type of Algorithm.

Ensemble techniques
- Parallel technique (Bagging)
- Sequential technique (Boosting)

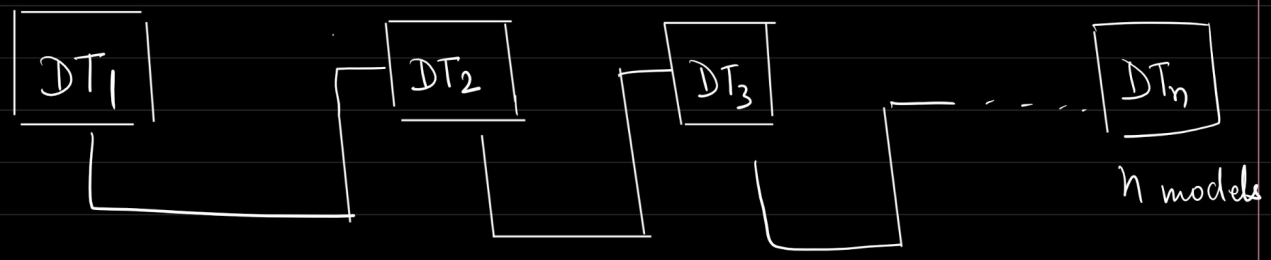
Parallel technique of Ensembles



→ All of the models here are built parallely and independent of each other.

* Sequential technique of Ensemble

→ All the models are built sequentially and dependent on each other.



→ learning from mistake.

* Ensembles technique
  - Parallel
  - Sequential

* Ensemble models
  - Bagging
  - Boosting
    1. Adaboost
    2. Gradient Boosting
    3. XGboost

① Custom Bagging ← Reg / classifier
② Random forest ← Reg / classifier

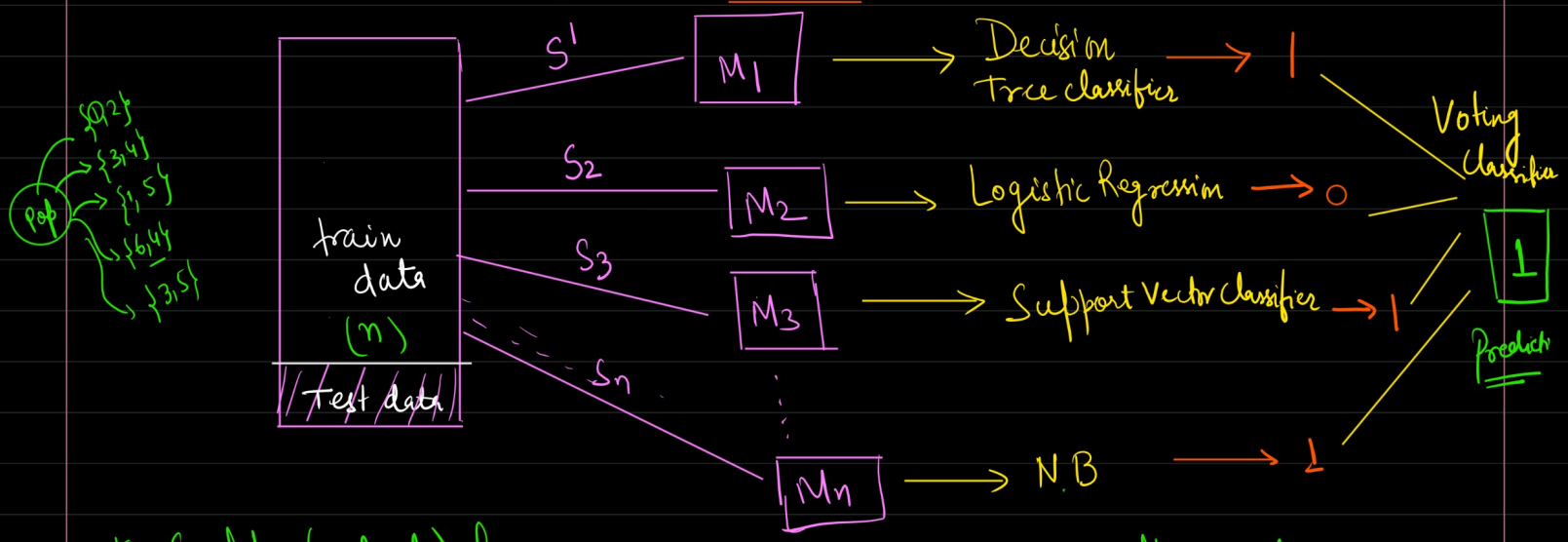
* Bagging Technique

→ Parallel models

Bootstrapping ⇓ $S < n$

(classification)

Aggregating prediction ⇓ from different base learners



* Samples (subset) for each model is taken with replacement.

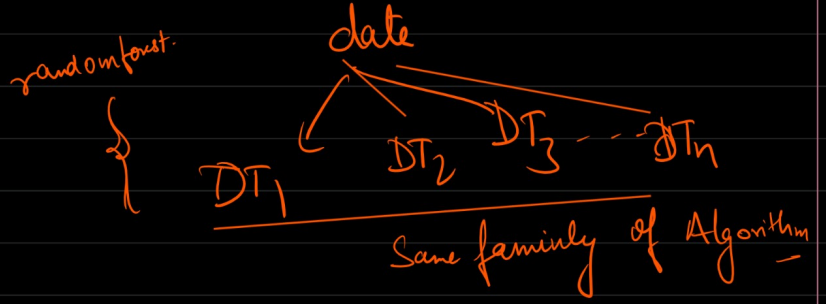
Bagging

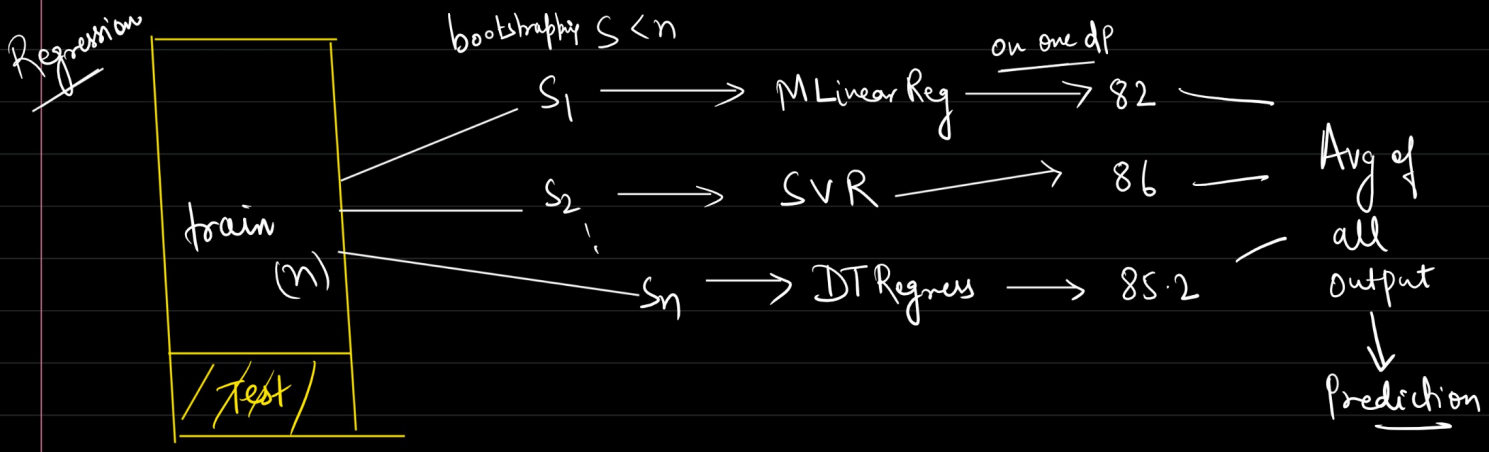
Bootstrap — Aggregating

⇓ different Samples with replacement ⇓ Bootstrap sample

⇓ Aggregating the prediction

* Custom bagging technique





→ Custom Bagging Regressor | classifier.

→ Pipeline | Column Transformer (missing value, Encoding, Scaling)

* Random Forest Classifer and Regressor

↓

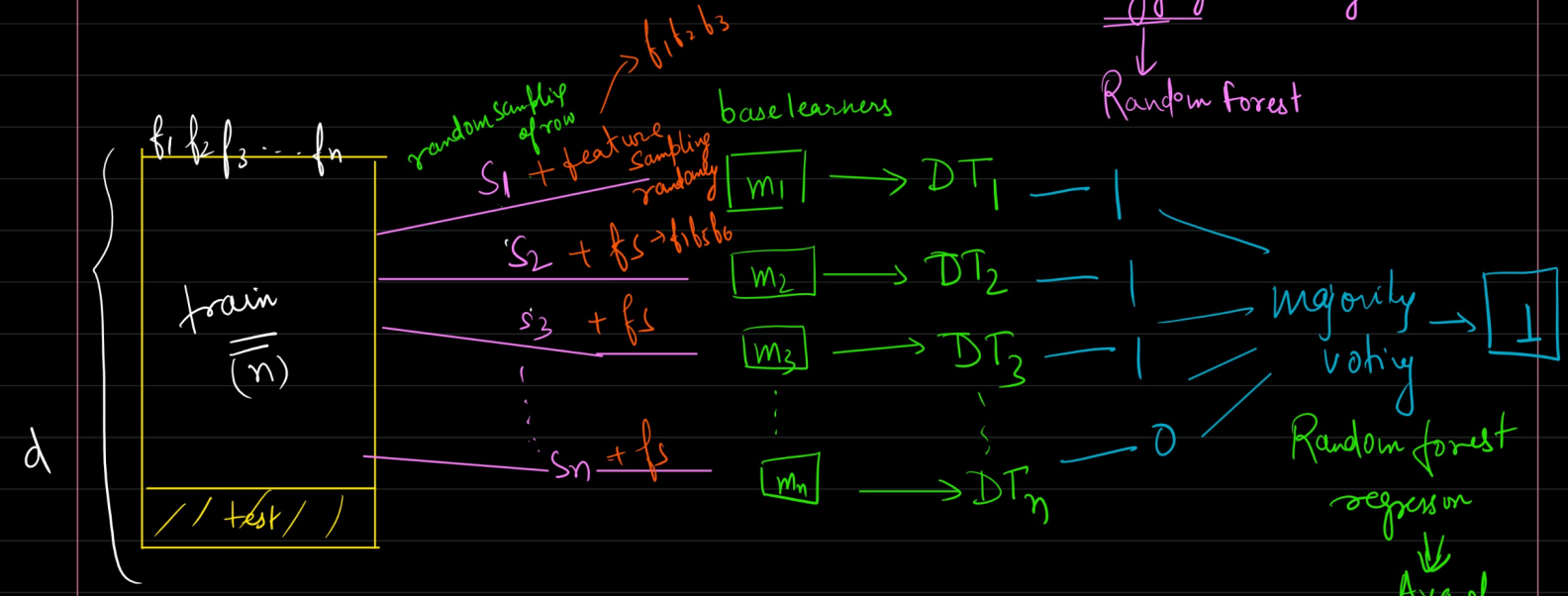
Forest → multiple trees

Ensembles

Bagging boosting

↓

Random forest



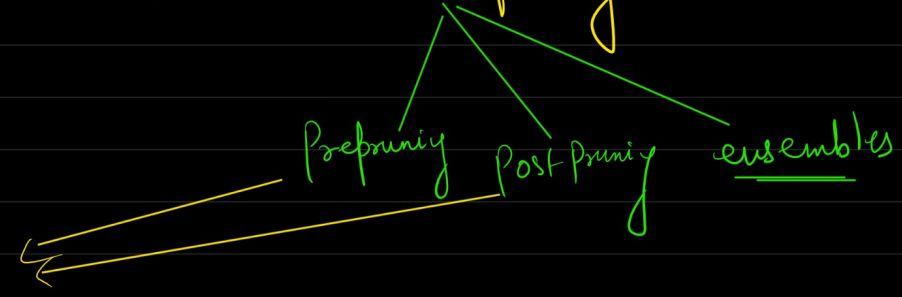
Random forest → multiple decision tree model in Parallel

→ The rows and features will be randomly sampled.

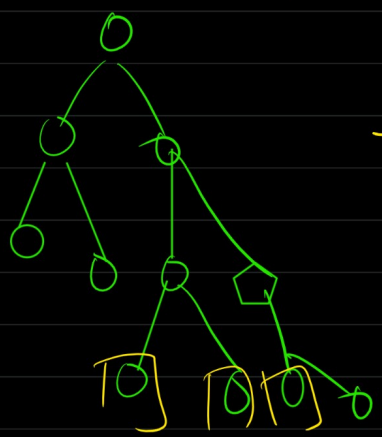
* Random forest classifier → Majority voting as the predicted result
* Random forest Regressor → Average o/p of all the models.

* If Decision tree, then why Random forest?

→ DT is a greedy Algorithm. It will keep splitting untill all the d/p's are memorized. and this leads to overfitting.



DT



Overfitting
→ Training Acc ↑↑ ⇒ low bias → Low bias
→ Test Acc ↓↓ ⇒ High variance → Low Variance

Generalised model

Random forest

* Random forest Reduces the variance (reduces Overfitting)
↳ give more robust/generalized model

→ in RF, random sampling of rows and features are done.

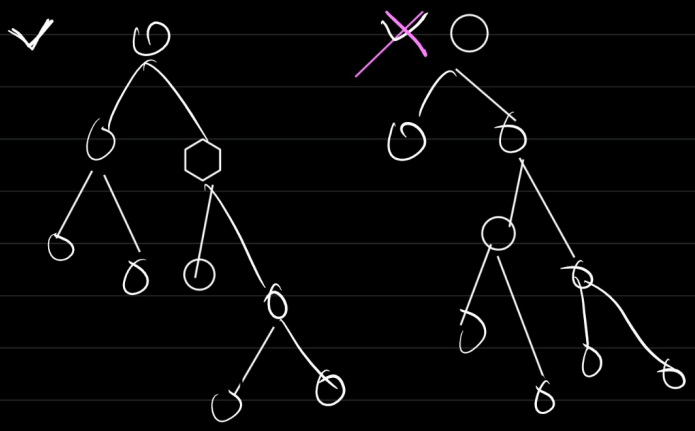
⇓

data is splitted into small chunks (randomly)

⇓

due to random subset of rows and features, each of the subset is different representation of itself.

→ Each of random subset trains a different model.



⇒ The feature/row which was causing an overfitted model might not be contributing to overfitting in other DT models of a random forest.

Ensemble techniques in machine learning combine multiple models to improve overall performance. The idea is that a group of models working together can outperform a single model. Here are some common ensemble methods:

Bagging (Bootstrap Aggregating):

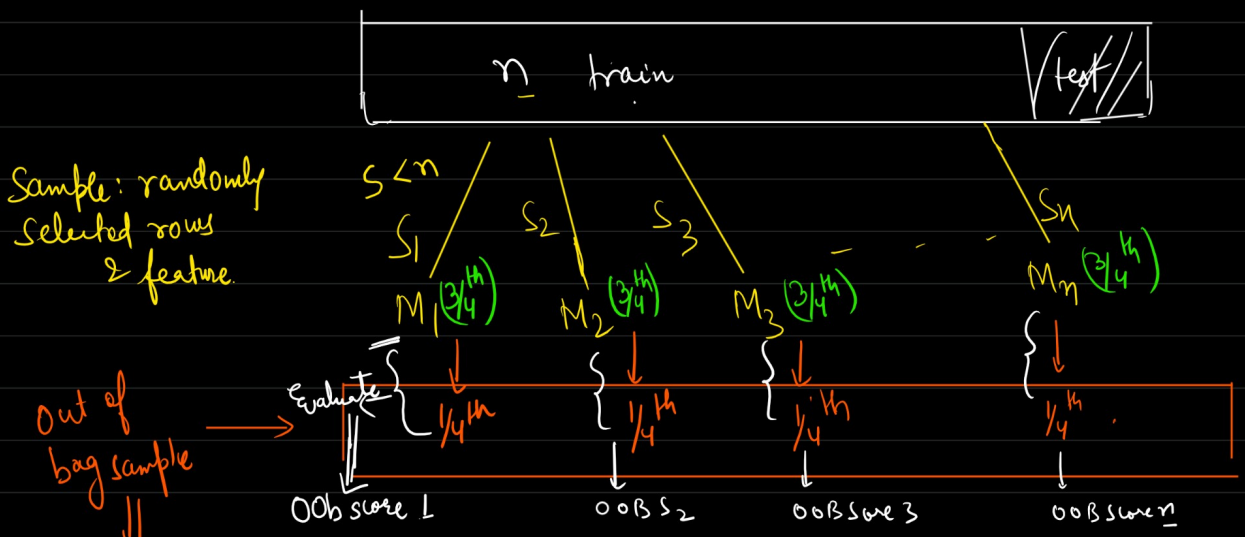
Trains multiple models on different subsets of the data (with replacement) and averages their predictions.
Example: Random Forest.
Boosting:

Builds models sequentially, with each model focusing on the errors of the previous ones.
Example: AdaBoost, Gradient Boosting, XGBoost.

Random Forest is an ensemble learning technique that builds multiple decision trees using different subsets of data and features. Each tree independently predicts the output, and the final result is determined by majority voting (classification) or averaging (regression). By introducing randomness in data sampling and feature selection, Random Forest reduces overfitting and increases model robustness. It is highly accurate, handles large datasets well, and provides feature importance insights. However, it can be computationally intensive and less interpretable compared to single decision trees. It's widely used in various applications, including finance, healthcare, and image recognition

The Out-of-Bag (OOB) score is an internal validation method used in Random Forest to estimate the model's performance. It works by using the data samples that were not included (out-of-bag) in the training subset for each decision tree. After the forest is built, the OOB samples are used to test the model, providing an unbiased estimate of its accuracy without needing a separate validation set. This is particularly useful when the dataset is small, as it maximizes data usage for training while still evaluating performance.

OOB Score (Out of Bag Score)



Part of train data, not used in model training for individual decision tree.

→ 1/4th acts as validation data for each specific Individual DT

→ [OOb score 1, 2, 3, . . n] → Avg of all individual OOB Score

Acts as validation score

OOB score

In training itself OOB score is low, model is not performing well on train data.