Machine Learning

Video 81:

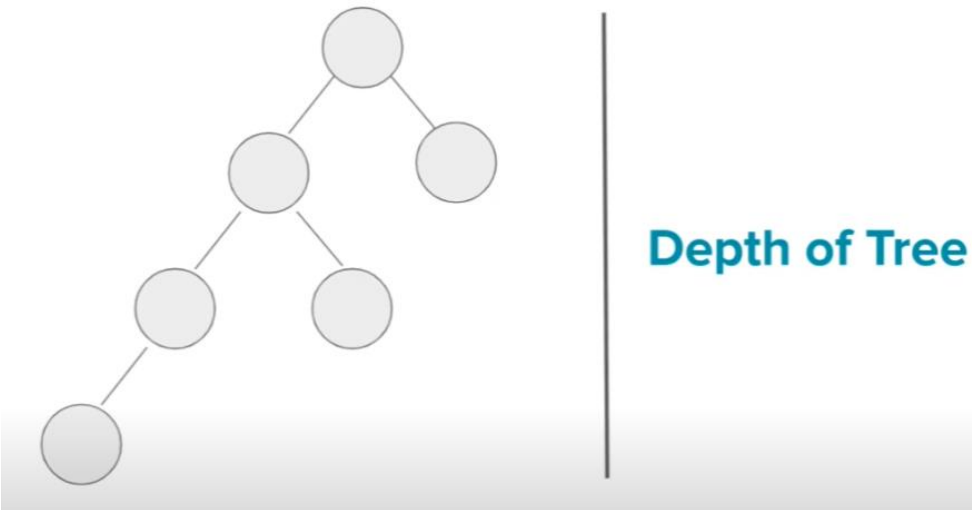
Decision Trees - Hyperparameters | Overfitting and Underfitting in Decision Trees

Hyperparameters in machine learning are the external configurations set before training a model, influencing its performance. These include learning rate, batch size, number of layers, and epochs. Unlike model parameters learned during training, hyperparameters are set manually or through optimization techniques to enhance model accuracy and efficiency.

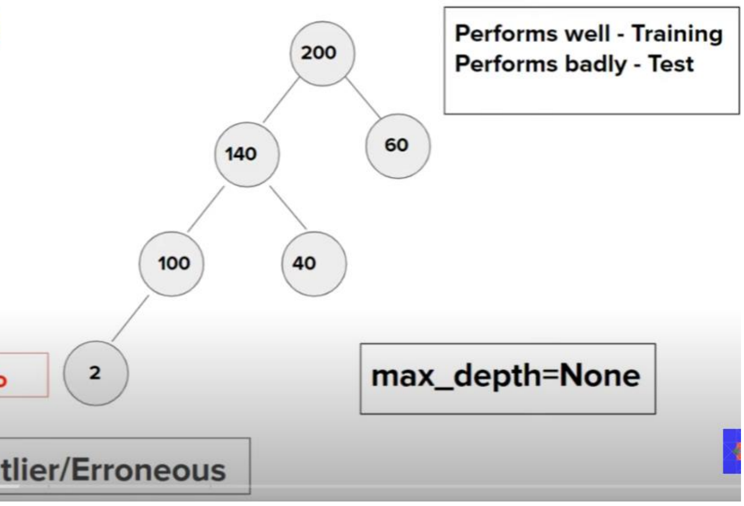
Various hyperparameters in machine learning depend on the type of model being used. Some common ones include:

1. **Learning Rate**: Controls the step size in model optimization.
2. **Batch Size**: Number of samples per update.
3. **Epochs**: Number of complete passes through the training dataset.
4. **Number of Layers**: Layers in neural networks.
5. **Number of Neurons/Units**: Units per layer in neural networks.
6. **Momentum**: Affects optimization speed and stability.
7. **Dropout Rate**: Prevents overfitting by randomly dropping neurons during training.
8. **Regularization (L1, L2)**: Controls model complexity to avoid overfitting.
9. **Activation Function**: Defines how output from neurons is transformed (e.g., ReLU, sigmoid).
10. **Optimizer**: Algorithm used for training, e.g., SGD, Adam, RMSprop.
11. **Learning Rate Scheduler**: Adjusts the learning rate over time.
12. **Kernel Size**: In convolutional layers for CNNs, determines the filter size.
13. **Filter/Number of Filters**: In CNNs, defines how many filters to apply.
14. **Weight Initialization**: Determines how model weights are initialized.
15. **Max Depth (for decision trees)**: Maximum depth of the tree.

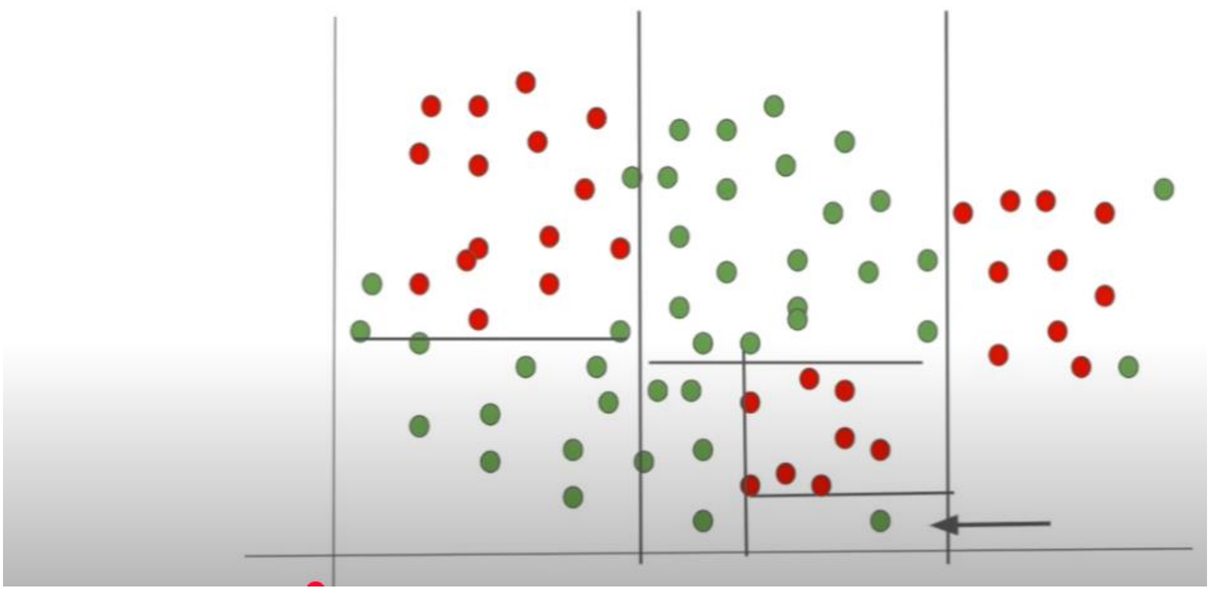
These hyperparameters significantly influence how well the model trains and generalizes to new data.



Overfitting



Geometric Intuition of Overfitting



Underfitting

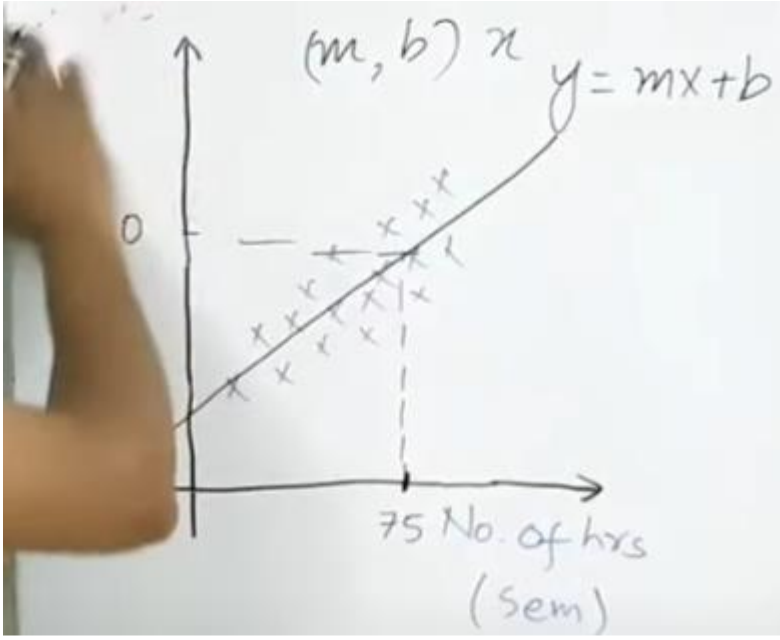
Code video link:

https://www.youtube.com/watch?v=mDEV0Iucwz0&list=PLKnIA16_Rmvbr7zKYQuBfsVkjoLcJgxHH&index=84

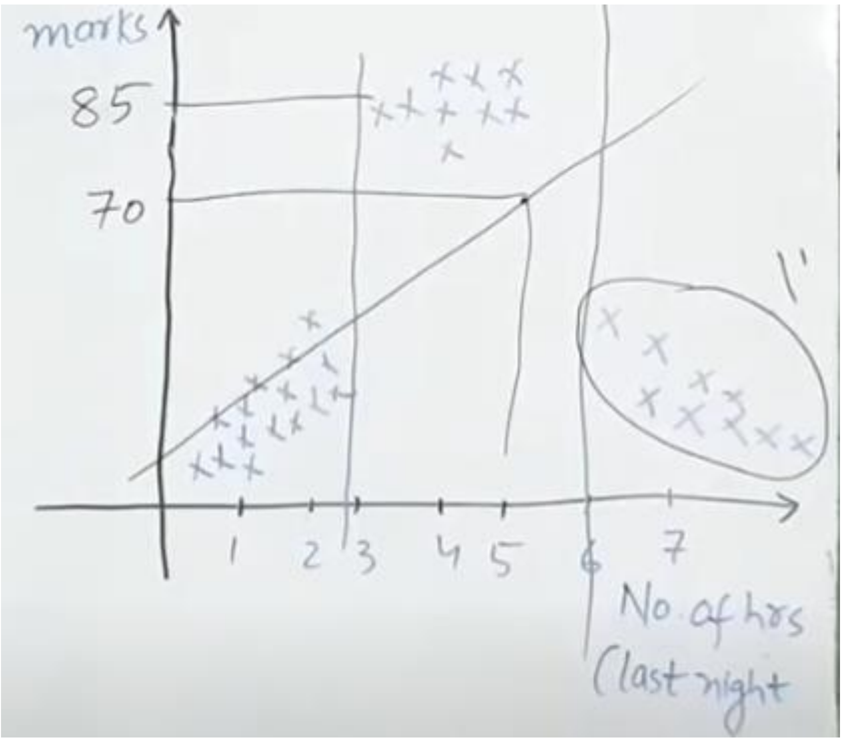
Video 82:

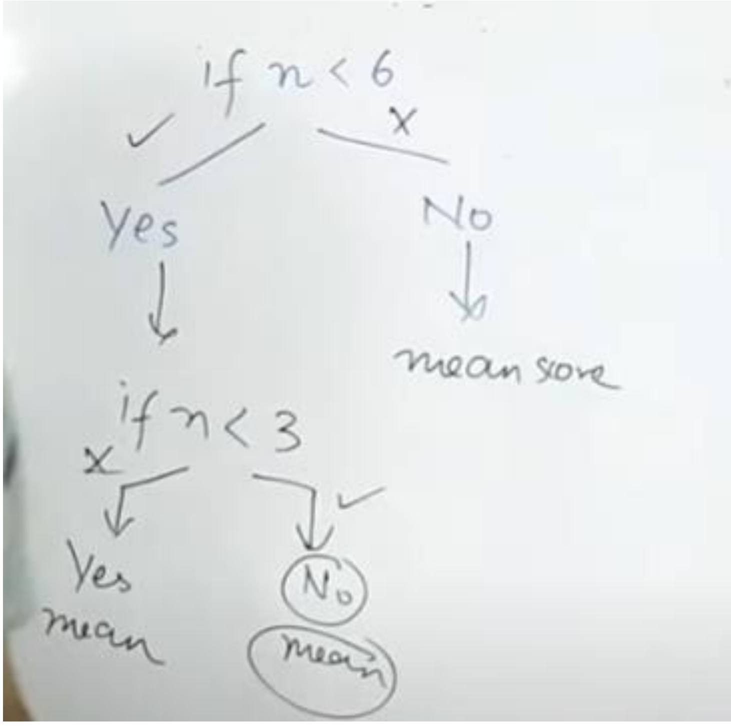
Regression Trees | Decision Trees Part 3

We know,



Now,





Example:

Code link:

https://github.com/campusx-official/decision-trees/blob/master/regression_tree_example.ipynb

Video 83:

Awesome Decision Tree Visualization using dtreeviz library

Example:

Code link:

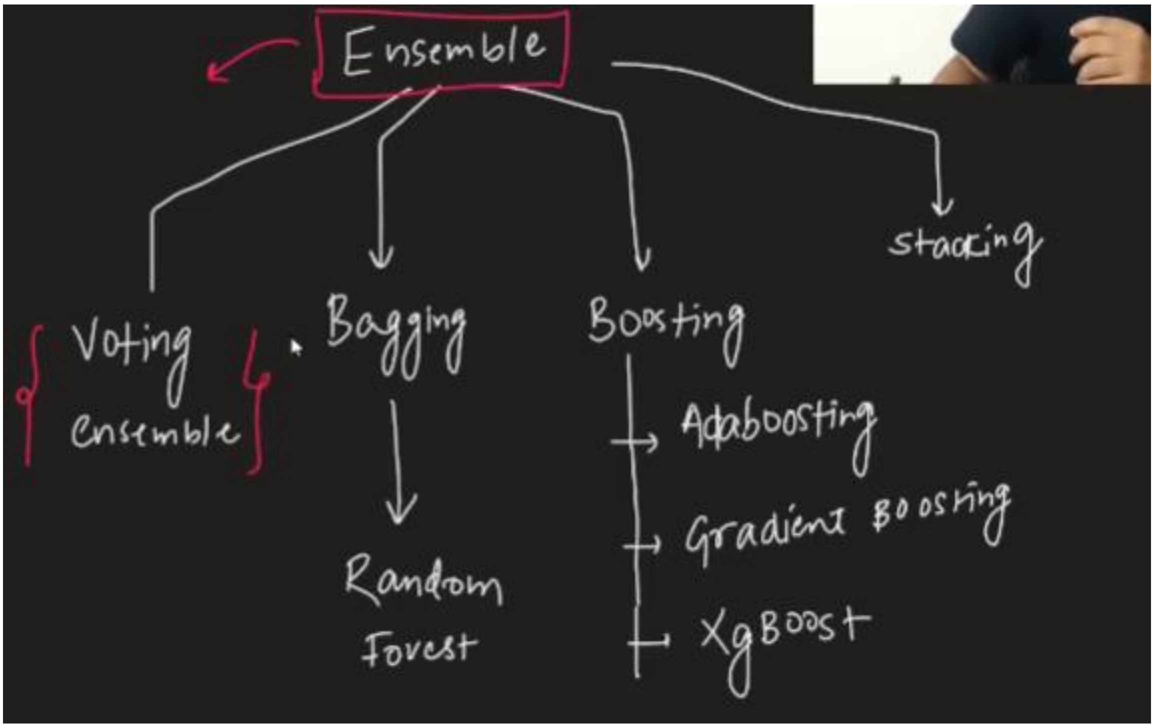
https://colab.research.google.com/drive/1rQOinGqwOnVRQFvSC9Ok9m9LXFkdH9mM?usp=sharing

Video 84:

Introduction to Ensemble Learning | Ensemble Techniques in Machine Learning

Ensemble learning in machine learning combines multiple models to improve accuracy, robustness, and generalization. It reduces overfitting and variance by aggregating diverse predictions through techniques like bagging, boosting, and stacking. Common examples include Random Forest and Gradient Boosting, which leverage multiple weak learners to create a stronger predictive model.

Types of ensemble learning:



Stacking (or **stacked generalization**) is an **ensemble learning technique** that combines multiple base models (level-0 models) and a meta-model (level-1 model) to improve predictive performance. The base models generate predictions, which are then used as input features for the meta-model. This final model learns how to optimally combine the base models' outputs for better accuracy.

Bagging is an ensemble learning technique that improves model stability and accuracy by training multiple instances of the same model on different randomly sampled subsets (with replacement) of the training data. The final prediction is obtained by averaging (for regression) or majority voting (for classification). Random Forest is a common example of bagging.

Voting is an ensemble learning technique used for classification, where multiple models (classifiers) make predictions, and the final output is determined based on their collective decision.

There are two types of voting:

1. **Hard Voting** – The class predicted by the majority of models is chosen.
2. **Soft Voting** – The class with the highest average probability (from all models) is selected.

Voting helps improve accuracy by leveraging multiple models' strengths.

Boosting is an ensemble learning technique that sequentially combines multiple weak learners (usually decision trees) to create a strong predictive model. Each new model focuses on correcting the errors of the previous ones by assigning higher weights to misclassified instances. This iterative process reduces bias and variance, improving accuracy.

Common Boosting Algorithms:

- AdaBoost (Adaptive Boosting) – Adjusts instance weights based on errors.
- Gradient Boosting – Minimizes errors using gradient descent.
- XGBoost – Optimized gradient boosting with regularization.
- LightGBM & CatBoost – Faster, efficient variations for large datasets.

Boosting is widely used in machine learning competitions and real-world applications like fraud detection and recommendation systems.

Advantages of Ensemble Learning

1. **Higher Accuracy** – Combines multiple models to improve overall predictive performance.
2. **Reduces Overfitting** – Methods like bagging lower variance and prevent overfitting.
3. **Handles Bias and Variance** – Boosting reduces bias, while bagging reduces variance.
4. **Robustness** – Less sensitive to noise and outliers compared to individual models.
5. **Flexibility** – Can combine different types of models for better generalization.
6. **Better Stability** – More consistent performance across different datasets.

Disadvantages of Ensemble Learning

1. **Higher Complexity** – Requires more computation and memory than single models.
2. **Longer Training Time** – Multiple models increase processing time, especially for boosting.
3. **Difficult Interpretation** – Harder to understand and explain compared to a single model.
4. **Risk of Overfitting** – Boosting can sometimes overfit if not tuned properly.
5. **Requires More Data** – Some ensemble methods perform poorly with small datasets.

Despite its challenges, ensemble learning is widely used in machine learning because of its superior accuracy and robustness.

Video 85:

Voting Ensemble | Introduction and Core Idea | Part 1

Voting ensemble in machine learning combines multiple models to improve prediction accuracy. It aggregates outputs from different classifiers using majority voting (for classification) or averaging (for regression). This approach reduces overfitting and enhances generalization by leveraging diverse model strengths, making it a robust technique for boosting overall performance.

Voting works because it leverages the wisdom of multiple models to improve accuracy and generalization. By combining diverse classifiers, errors from individual models are reduced, and the overall prediction becomes more reliable. Majority voting (for classification) or averaging (for regression) helps mitigate biases and variance, leading to better performance.

A **Voting Classifier** is an ensemble learning technique in machine learning that combines predictions from multiple models to improve accuracy. It aggregates outputs using:

- **Hard Voting** – Chooses the class predicted by the majority of models.
- **Soft Voting** – Averages class probabilities and selects the most probable class.

This approach enhances model robustness and reduces errors.