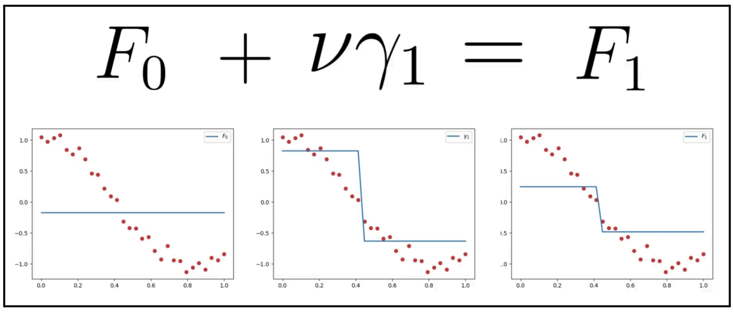
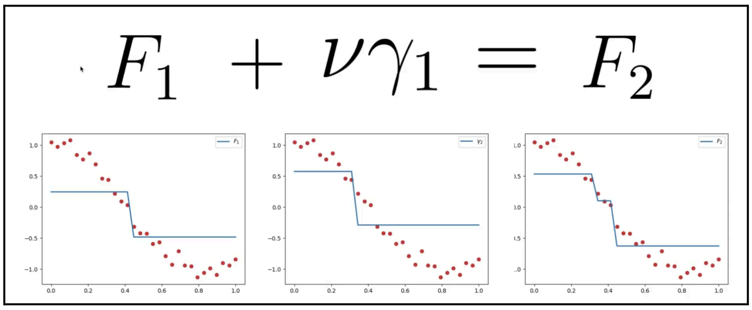
<u>Gradient Boosting</u>

- Supervised learning
  - Finds the best function f to minimize the cost
- Contains one base value (constant) and to this it adds weak learners to better the prediction
  - The constant value is F_0 (our initial best guess)
- Basically just adding a new trained model to our best guess to get a better guess

$$\hat{F}(x) = \sum_{m=1}^{M} \gamma_m h_m(x) + \text{const}$$





https://www.youtube.com/watch?v=lOwsMpdjxog

<u>XGBoost (Extreme Gradient Boosting)</u>

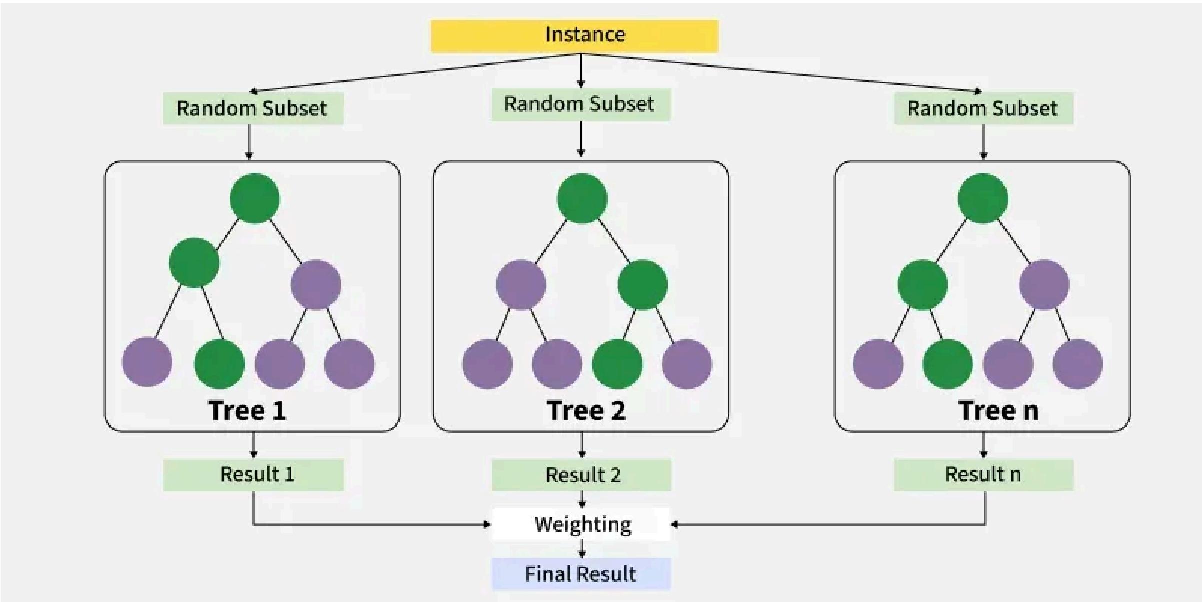
- Combines decision trees
  - Each new tree is trained to correct the previous one → boosting
- Parallel processing for high speed
- Allows for adjusting parameters to optimize performance based on the specific problem (would work well for us since we can change what it takes)

***** Gradient boost + Optimization techniques *****

- L1 and L2 Regularization
  - Prevents overfitting and increases generalization
- Model regularization
  - Built in feature for missing values
- Cross validation
- Parallelization
- Pruning
  - Depth first strategy
  - Deeper but more optimised trees
- Versatility
- Easy interface

The algorithm

1. _Base model_
    - The first decision tree is trained on the data
    - Regression tasks → predicts the average of the target variable
2. _Calculate the errors_
    - Errors between the predicted and actual values are calculated
3. _Train the next tree_
    - The next tree is trained on the errors of the previous tree (corrects the previous errors)
4. _Repeat the process_
    - Process continues until a stopping criterion is met
5. _Combine the predictions_
    - The final prediction is the sum of the predictions from all the trees



Advantages

- Good for large datasets
- Supports parallel processing and GPU acceleration → faster training
- Offers customizable parameters and regularization for fine-tuning
- Includes feature importance analysis for interpretability and explainability

Disadvantages

- Computationally intensive
- Sensitive to noise or outliers, requiring careful data preprocessing
- Prone to overfitting, especially on small datasets or with too many trees.
- Model interpretability is limited compared to simpler methods
    - But does show feature importance

Usage

When should we use it?

- Classification problems, especially those related to real-world business problems.
- Problems in which the range or distribution of target values present in the training set can be expected to be similar to that of real-world testing data.
- Situations in which there are many categorical variables.
- large number of observations in training data.
- The number of features is smaller than the number of observations in training data.

When should we not use it?

- The number of observations in training data is significantly smaller than the number of features.
- Computer vision
- Natural language processing
- Regression tasks that involve predicting a continuous output.
- Predicting increases in targets beyond the range present in the training data.
- Tasks involving extrapolation.

Installing links

- https://xgboost.readthedocs.io/en/stable/
- https://xgboost.readthedocs.io/en/stable/python/python_intro.html

References

DataMListic. (2023, October 17). *XGBoost Explained in Under 3 Minutes*. YouTube. https://www.youtube.com/watch?v=33fGfuleXw0

GeeksforGeeks. (2021, September 18). *XGBoost*. GeeksforGeeks. https://www.geeksforgeeks.org/xgboost/

Nvidia. (2024). *What is XGBoost?* NVIDIA Data Science Glossary. https://www.nvidia.com/en-us/glossary/xgboost/

Adebisi, O. (2021). *When to NOT use XGBoost? | Kaggle*. Www.kaggle.com. https://www.kaggle.com/discussions/general/196542