Ensemble Learning

Presented By: **Lina Ben Salem**

Plan

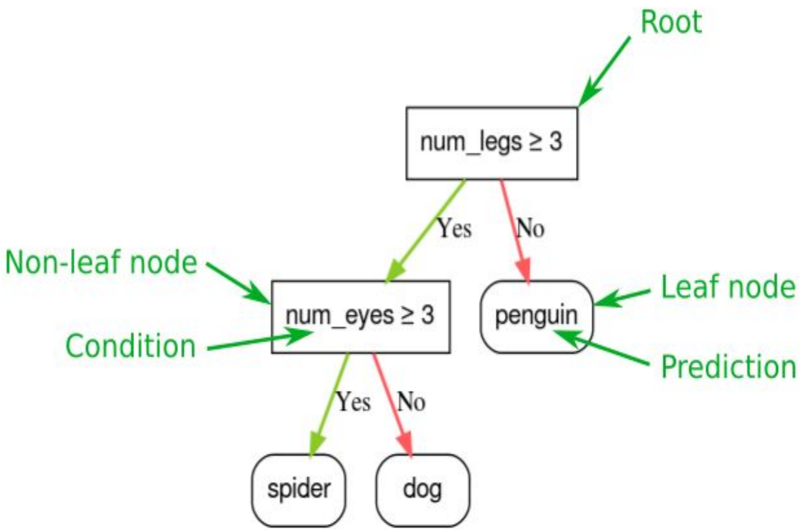
1. Decision Tree
2. Random Forest
3. Ensemble Learning
4. AdaBoost
5. Gradient Boosting
6. XGBoost

Decision Tree

- A model composed of a collection of "questions" organized hierarchically in the shape of a tree. The questions are usually called **conditions**. Each non-leaf node contains a condition, and each **leaf node** contains a prediction.

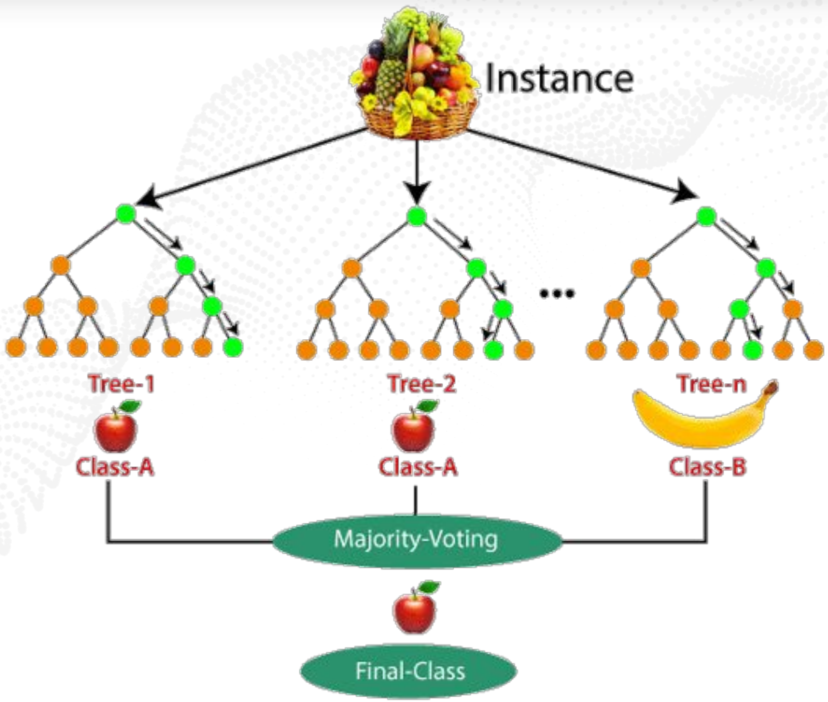
- Common supervised learning algorithm

- Prone to problems, such as bias and overfitting
- => The **random forest** algorithm, which is a multiple decision trees.



Random Forest

- It is the most popular ensemble of decision trees
- It is made up of multiple decision trees
- It combines the output of multiple decision trees to reach a single result
- It handles both classification and regression problems
- It picks the **most of votes** of all trees and this technique is called training using **bagging**



Ensemble Learning

- “wisdom of the crowd”
- The Ensemble is the art of combining diverse set of learners (individual models) together
- The combination of all the predictions together is termed as **Ensemble Learning**
- It creates multiple models and then combines them to produce improved results
- It usually produces more accurate solutions than a single model would
- It decrease bias and variation while also increasing performance by combining the predictions of many models

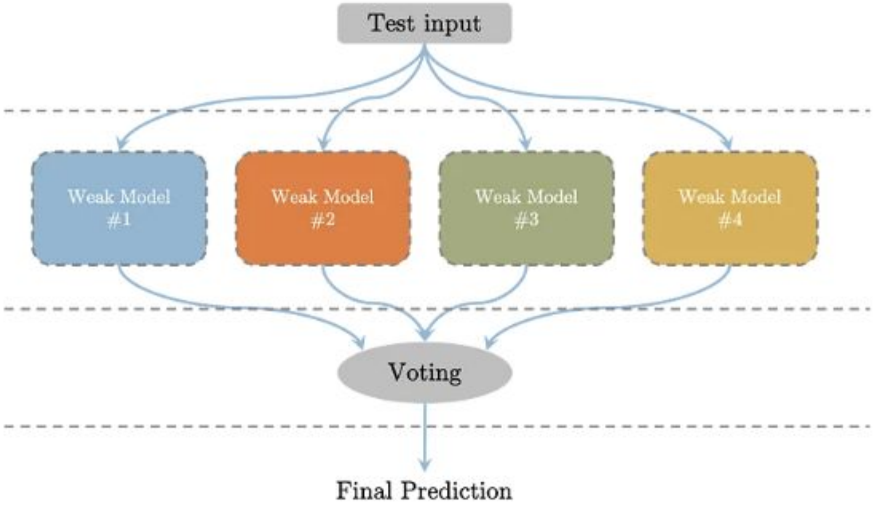
Ensemble Learning

- **Technical Definition:**

  Ensemble learning uses more than one “weak learner” collectively to predict the output. Instead of training one large/complex model for the dataset, you train multiple small/simpler models (weak-learners) and aggregate their output (in various ways) to form your prediction as shown in the figure below

- Types: **bagging**, **boosting**, voting classifier and stacking

- They use the wisdom-of-the-crowd concept but differ in the details of what it focuses on, the **type of weak learners** used, and the type of **aggregation** used to form the final output



Ensemble Learning: Bagging

- Also called **Bootstrap Aggregation**

- It involves creating multiple subsets of the original dataset called **bootstrap sampling**, then **training** a base learning algorithm independently on each subset, and finally **aggregating** their predictions to make a final prediction

- It reduces the impact of outliers and noise in the training data, leading to a more robust model with better generalization performance

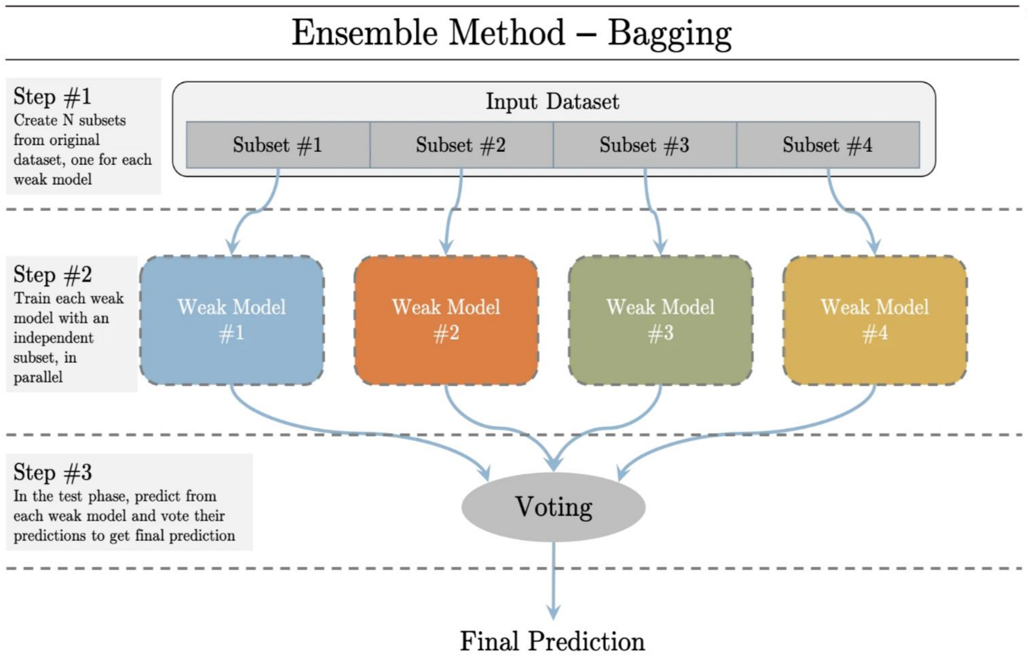
- Example of bagging: **Random Forest**

- 3 simple steps = Bootstrap sampling + model training + Aggregation

Ensemble Learning: Bagging

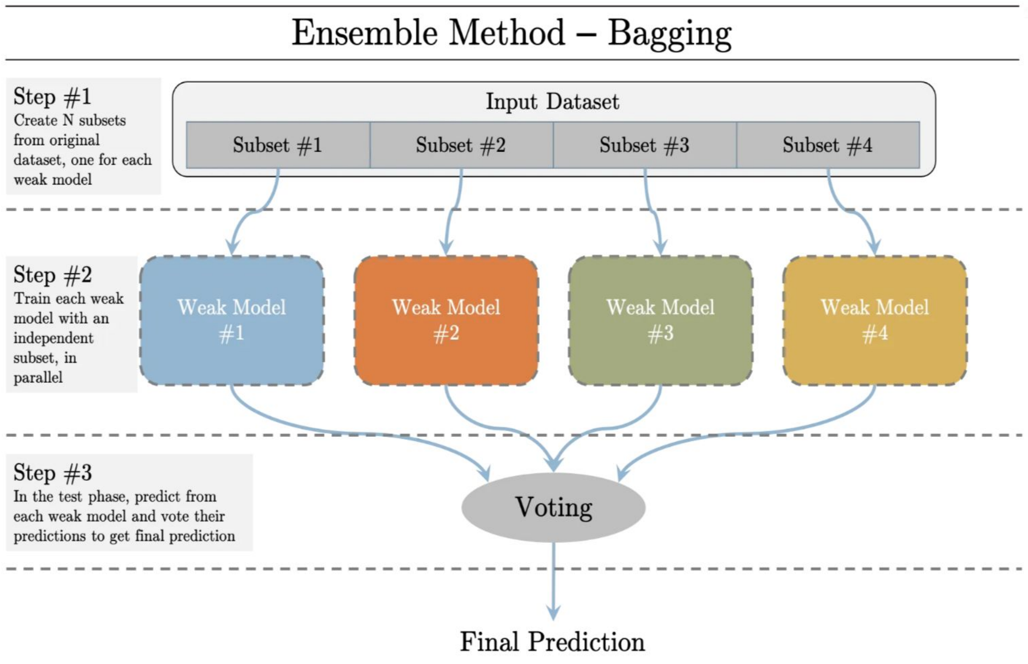
1. Bootstrap sampling:
   Sampling with replacement, meaning randomly sampling data and allowing the risk of duplicates

2. Model Training:
   A base learning algorithm is trained on each bootstrap sample independently. Each model is trained on a slightly different version of the original data due to the random sampling, introducing diversity among the models.



Ensemble Learning: Bagging

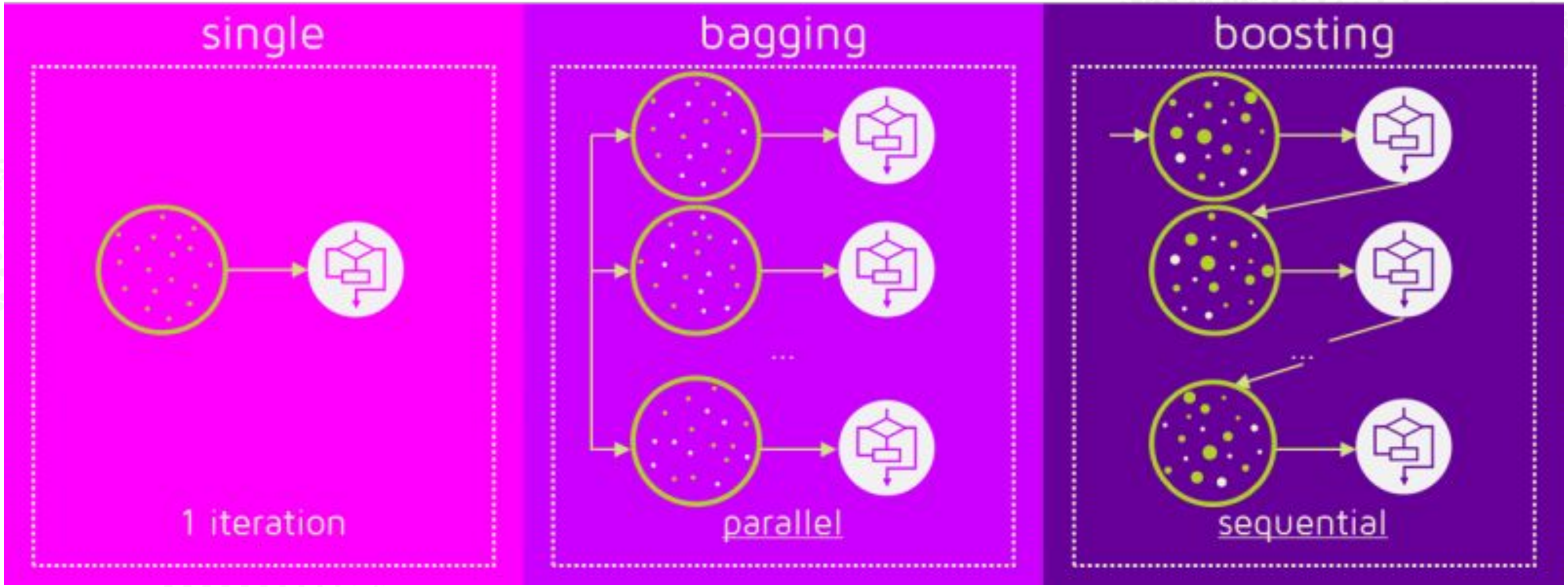
3. Aggregation:
   Once all models are trained, predictions from each model are combined to make a final prediction. This aggregation can be done through techniques like majority voting for classification tasks or averaging for regression tasks.



Ensemble Learning: Boosting

- In boosting, multiple weak-learners are learned sequentially
- Each subsequent model is trained by giving more importance to the data points that were **misclassified** by the previous weak-learner(low biased). In this way, the weak-learners can focus on specific data points and can collectively reduce the bias of the prediction to correct its predecessor
- If an observation was classified incorrectly, it tries to increase the weight of this observation and vice versa
- => It learns from **past mistakes**
- Boosting in general decreases the bias error and builds strong predictive models
- Examples of boosting: **adaboost**, **gradient boosting**, **xgboost**
- Side effects: it may cause overfitting on the training data

Ensemble Learning: Bagging vs Boosting



Ensemble Learning: Bagging vs Boosting

Bagging	Boosting
– Bootstrapped samples	– Fit entire data set
– Base trees created independently	– Base trees created successively
– Only data points considered	– Use residuals from previous models
– No weighting used	– Up-weight misclassified points

AdaBoost

- AdaBoost short for **Adaptive Boosting**

- The main idea is to iteratively train **the weak classifier** on the training dataset with each successive classifier giving **more weightage** to the data points that are **misclassified**.

- The final AdaBoost model is decided by combining all the weak classifier that has been used for training with the weightage given to the models according to their accuracies.

- The weak model which has the highest accuracy is given the highest weightage while the model which has the lowest accuracy is given a lower weightage.

Gradient Boosting

- Gradient Boosting works by sequentially adding predictors to an ensemble, each one correcting its predecessor. However, instead of tweaking the instance weights at every iteration like AdaBoost does, this method tries to **fit the new predictor to the residual errors** made by the previous predictor.

- It uses **gradient descent algorithm** in order to minimize the loss function, that's where the term gradient comes from

XGBoost

What is XGBoost Algorithm?

- XGBoost or the **Extreme Gradient boost** is a machine learning algorithm that uses the gradient boosting decision tree algorithm
- Developed by the **university of washington** in 2016
- Credits with numerous competitions
- It uses many tricks to optimize **accuracy** and **speed**

XGBoost

XGBoost

The Features of XGboost:

1. It computes **2nd order gradients** to figure out the direction of gradients (first order gives the slope)
2. It uses advance **L1** and **L2 regularization** to prevent overfitting
3. It **parallelizes** for fast computing
4. It automatically takes care of **early stopping**
5. It handles sparse data: Missing values or data processing steps like one-hot encoding make data sparse

XGBoost

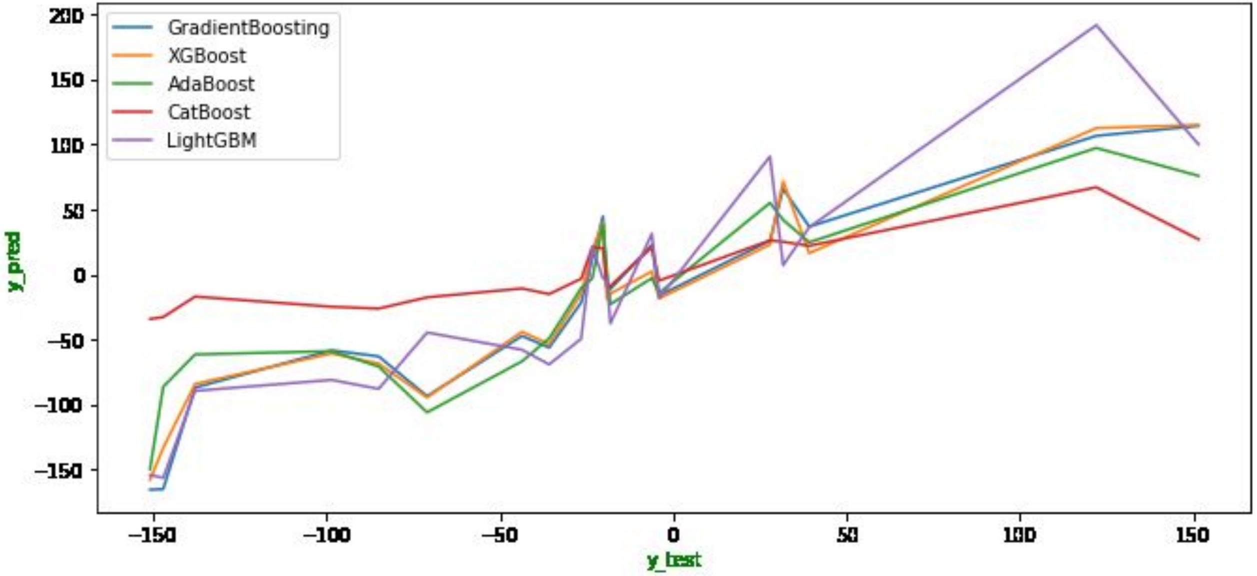
6. It is having built-in **cross-validation** features that are being implemented at each iteration in the model creation

7. **Hardware optimization** was also considered during the design of the XGBoost algorithm. Internal buffers are allocated for each of the threads to store the gradient statistics.

=> We tend to use it if we want to import the whole xgboost library but in the case of classification we use **xgbclassifier** which is an interface for classification problems while **xgbRegressor** is for regression problems

XGBoost

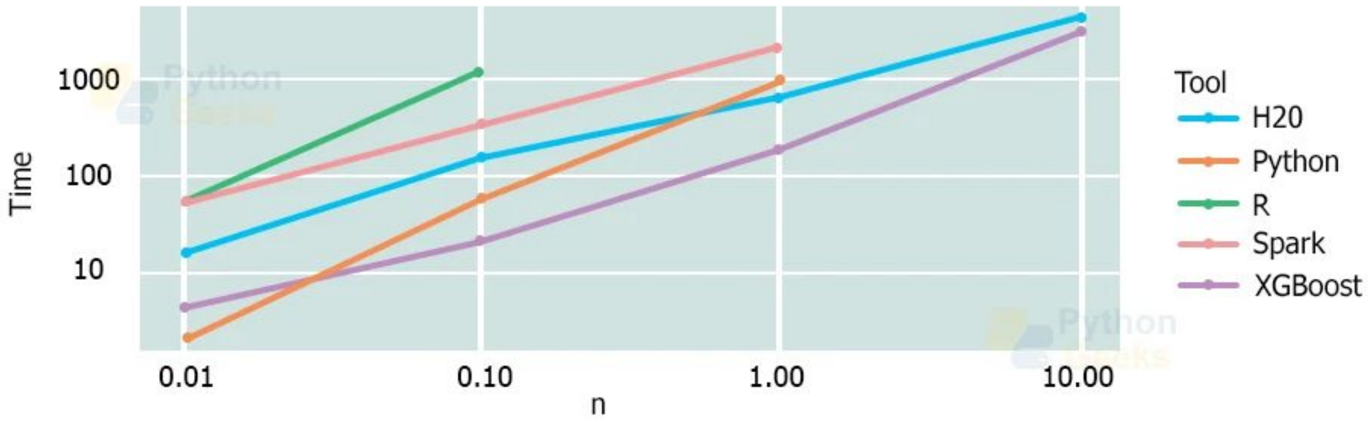
XGBoost performance compared to the rest of decision tree algorithms



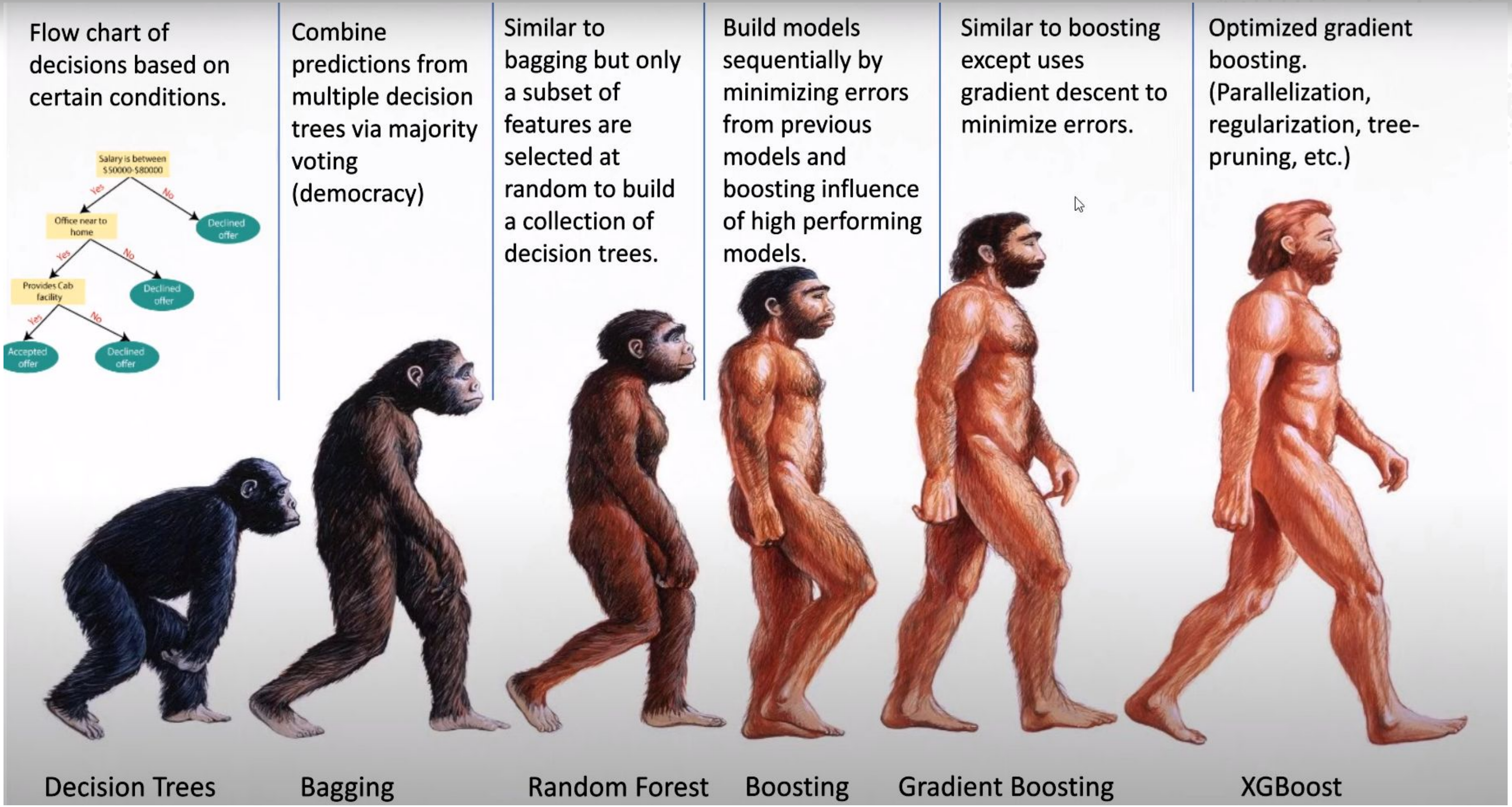
XGBoost

XGBoost Speed

- XGBoost exhibits really fast performance. When we compare the computational speed of XGBoost to other algorithms, it shows high variance in the speed of all other algorithms.
- **Szilard Palka** conducted an experiment to compare the performance of all the implementations of gradient boosting and the results are quite significant.
- The results in the above graph demonstrate that XGBoost always shows a better performance rate as compared to other implementations from R, Python, Spark, and H2O.



Summary


Flow chart of decisions based on certain conditions.
Salary is between $50000-$80000
Yes
No
Office near to home
Declined offer
Yes
No
Provides Cab facility
Declined offer
Yes
No
Accepted offer
Declined offer
Combine predictions from multiple decision trees via majority voting (democracy)
Similar to bagging but only a subset of features are selected at random to build a collection of decision trees.
Build models sequentially by minimizing errors from previous models and boosting influence of high performing models.
Similar to boosting except uses gradient descent to minimize errors.
Optimized gradient boosting. (Parallelization, regularization, tree-pruning, etc.)
Decision Trees
Bagging
Random Forest
Boosting
Gradient Boosting
XGBoost

XGBoost

- **Catboost**

- **Light Gradient Boosting**

- **Voting classifier**

Recommended link:
https://www.datacamp.com/tutorial/xgboost-in-python

Thank you!