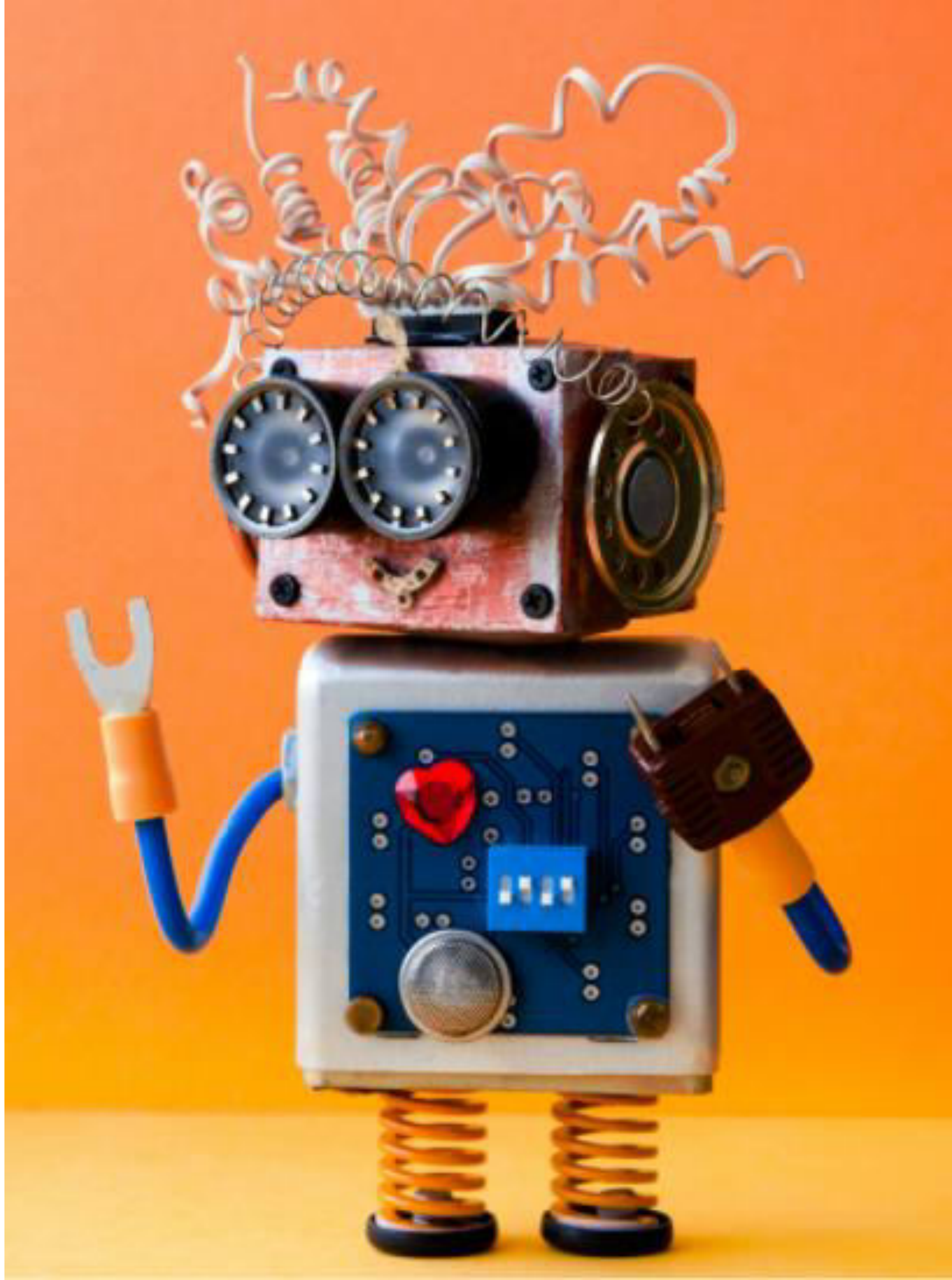
PROJECT INTRODUCTION

EASY ADVANCED



INTRODUCTION AND KEY LEARNING OUTCOMES

- We will also analyze university admission datasets in AWS SageMaker Studio and train an XG-Boost algorithm in Scikit-Learn to solve regression problems.
- We will learn how to:
  1. Perform data visualization using Seaborn and Matplotlib libraries
  2. Understand the theory and intuition behind boosting
  3. Learn the advantages and disadvantages of XG-boost algorithm
  4. Train an XG-boost algorithm in Scikit-Learn to predict university admission
  5. Train an XG-boost algorithm in Scikit-Learn to predict life expectancy (capstone project)
  6. Assess trained models performance

PROJECT CARD

GOAL:
- *Build, train, test and deploy an XG-Boost model to predict chances of university admission into a particular university* given student's profile.

TOOL:
- *AWS SageMaker Studio & Scikit-Learn*

PRACTICAL REAL-WORLD APPLICATION:
- *This project can be effectively used by university admission departments to determine top qualifying students.*

DATA:

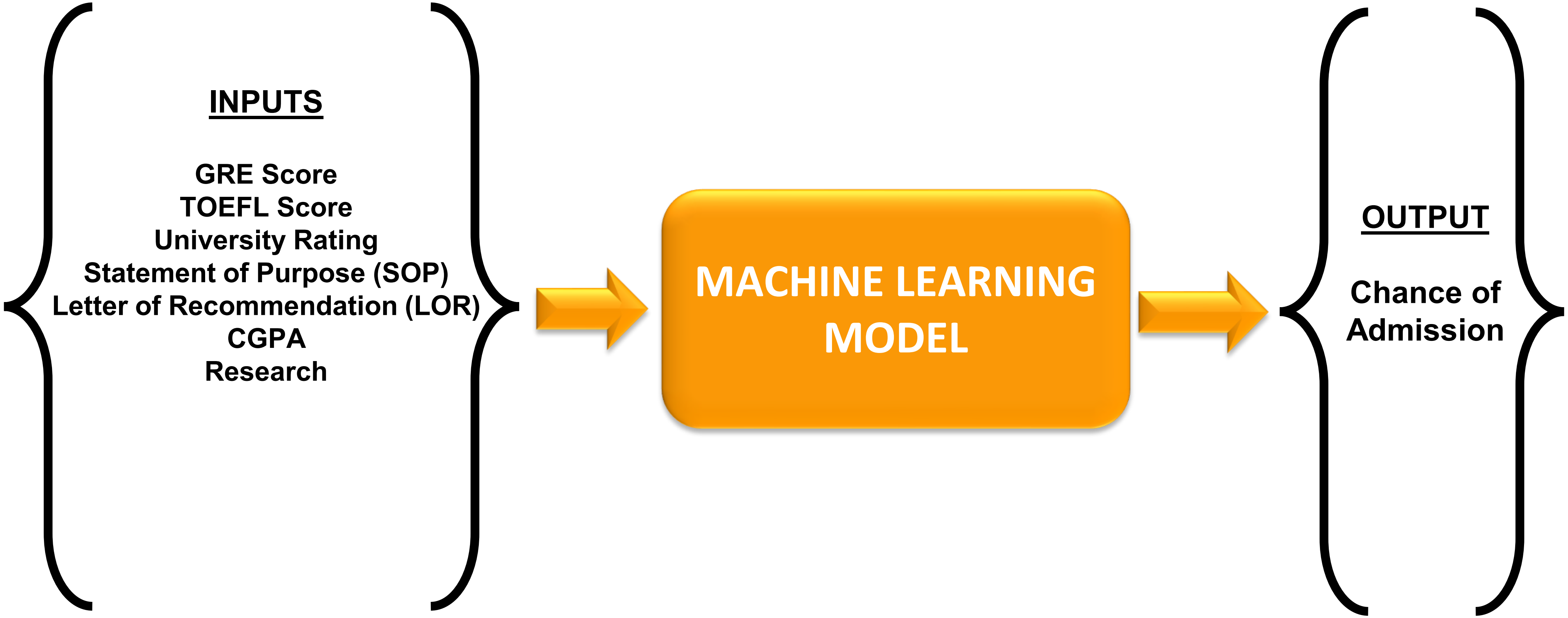
INPUTS (FEATURES):
- *GRE Scores (out of 340)*
- *TOEFL Scores (out of 120)*
- *University Rating (out of 5)*
- *Statement of Purpose (SOP)*
- *Letter of Recommendation (LOR) Strength (out of 5)*
- *Undergraduate GPA (out of 10)*
- *Research Experience (either 0 or 1)*

OUTPUTS:
- *Chance of admission (ranging from 0 to 1)*

- Data Source: https://www.kaggle.com/mohansacharya/graduate-admissions
- Photo Credit: https://www.pexels.com/photo/accomplishment-ceremony-education-graduation-267885/

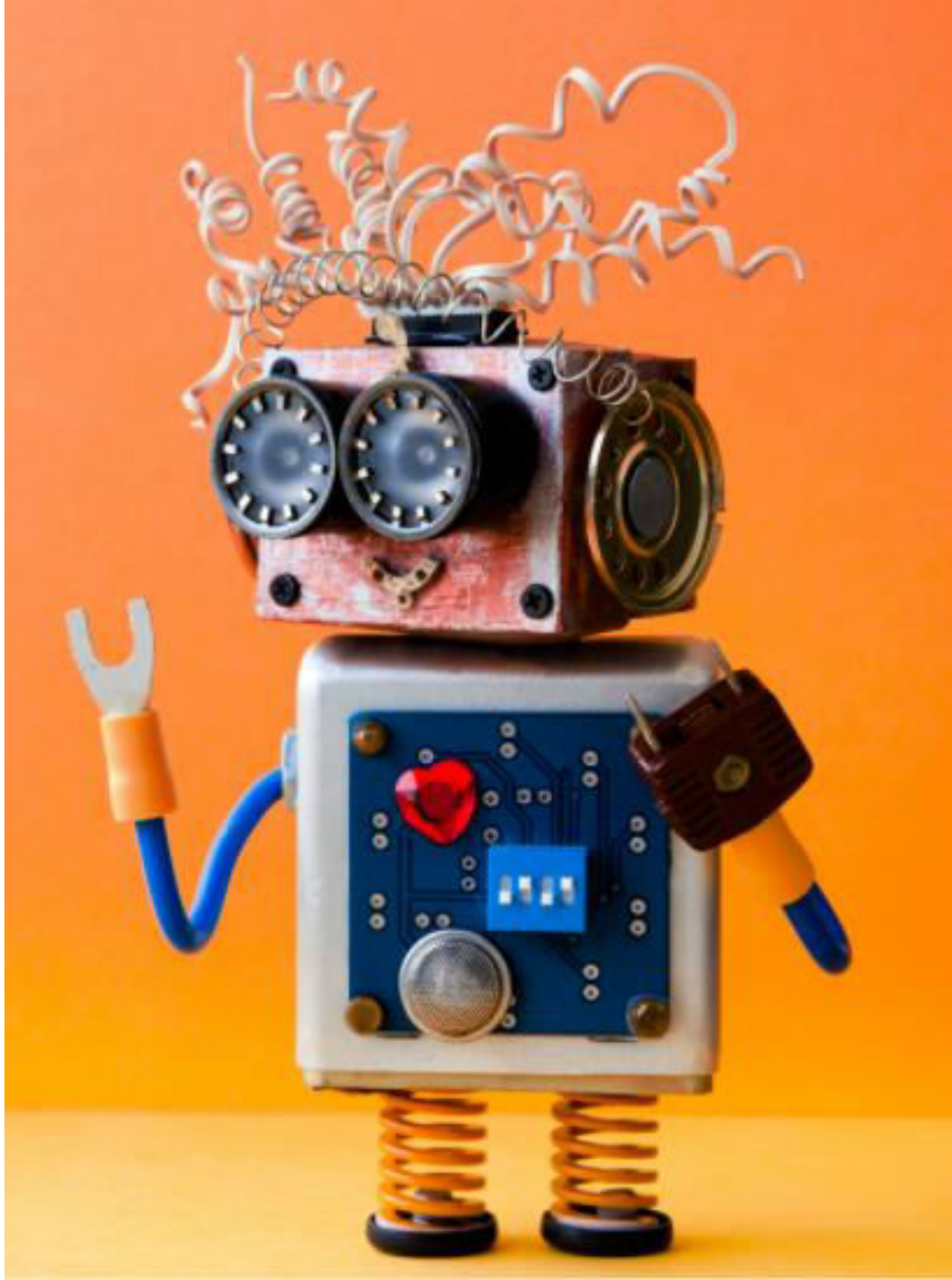
https://www.flickr.com/photos/pasa/6757993805
https://www.kaggle.com/ljanjughazyan/cars1

PROJECT OVERVIEW



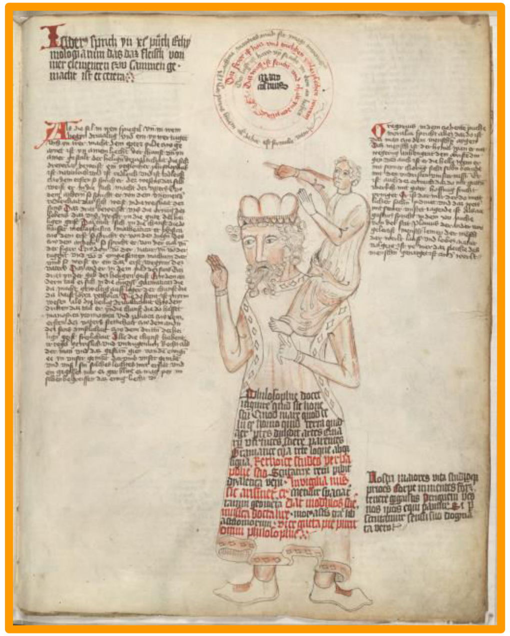
XG-BOOST ALGORITHM

EASY ADVANCED



XGBOOST: INTRODUCTION

- XGBoost or Extreme gradient boosting is the algorithm of choice for many data scientists and could be used for regression and classification tasks.
- XGBoost is a supervised learning algorithm and implements gradient boosted trees algorithm.
- The algorithm work by combining an ensemble of predictions from several weak models.
- It is robust to many data distributions and relationships and offers many hyperparameters to tune model performance.
- Xgboost offers increased speed and enhanced memory utilization.
- Xgboost is analogous to the idea of “discovering truth by building on previous discoveries”.



This picture is derived from Greek mythology: the giant Orion carried his servant Cedalion on his shoulders to act as the giant's eyes.

"If I have seen further it is by standing on the shoulders of Giants”, Isaac Newton

[Source: https://commons.wikimedia.org/wiki/File:Library_of_Congress,_Rosenwald_4,_Bl._5r.jpg](https://commons.wikimedia.org/wiki/File:Library_of_Congress,_Rosenwald_4,_Bl._5r.jpg)

ADVANTAGES AND DISADVANTAGES OF XGBOOST

ADVANTAGES:

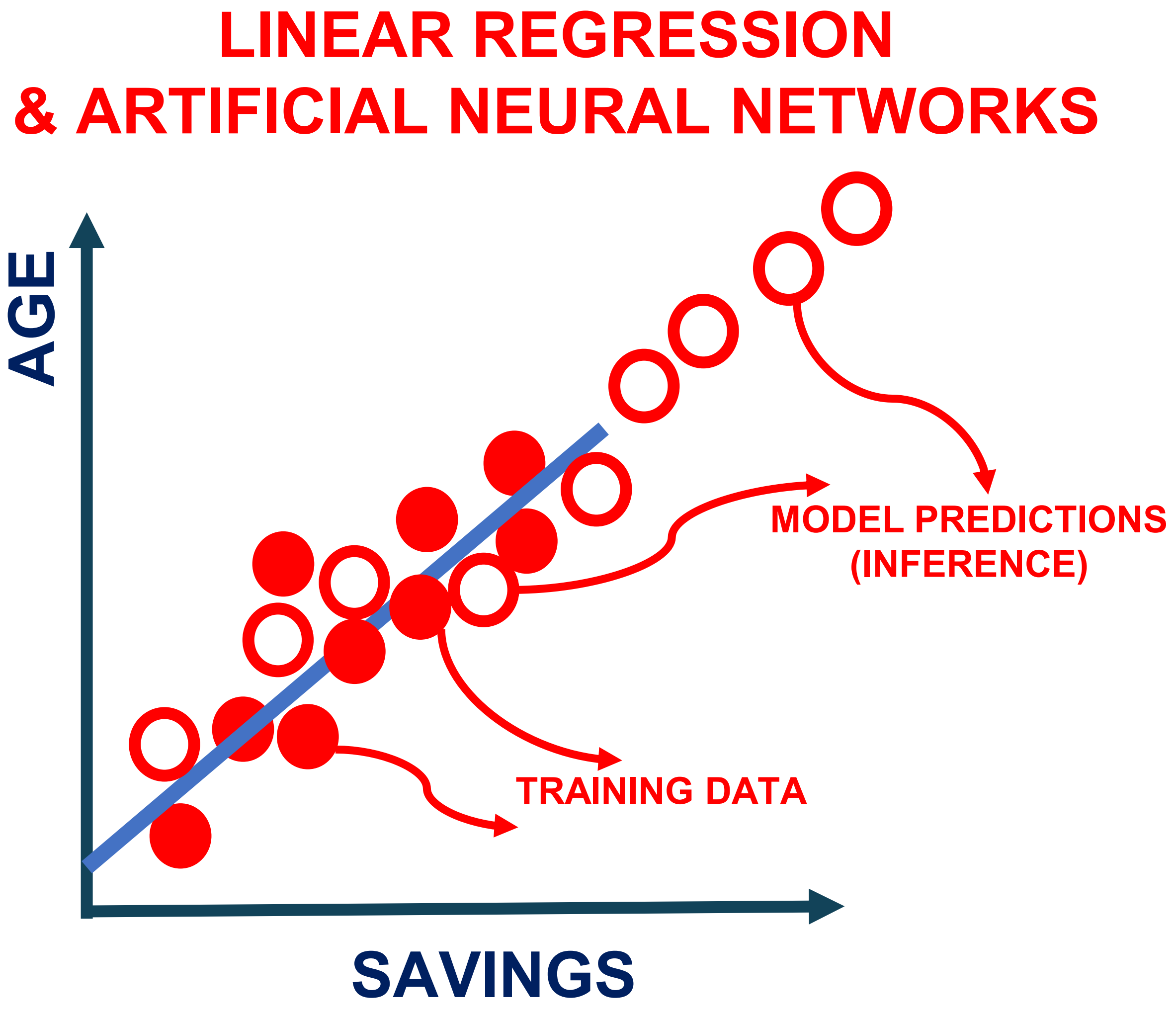
- No need to perform any feature scaling
- Can work well with missing data
- Robust to outliers in the data
- Can work well for both regression and classification
- Computationally efficient and produce fast predictions
- Works with distributed training: AWS can distribute the training process and data on many machines

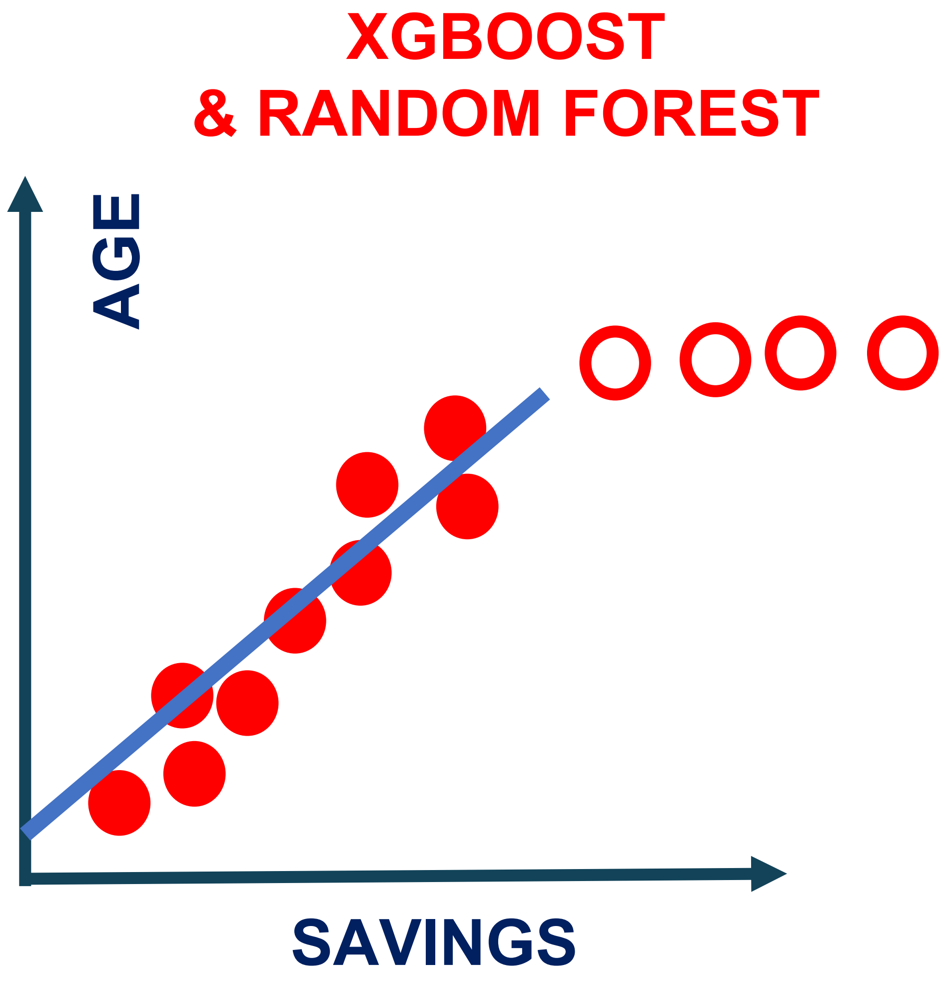
DISADVANTAGES:

- Poor extrapolation characteristics
- Need extensive tuning
- Slow training

DISADVANTAGES OF XGBOOST: POOR EXTRAPOLATION CAPABILITY BY XGBOOST

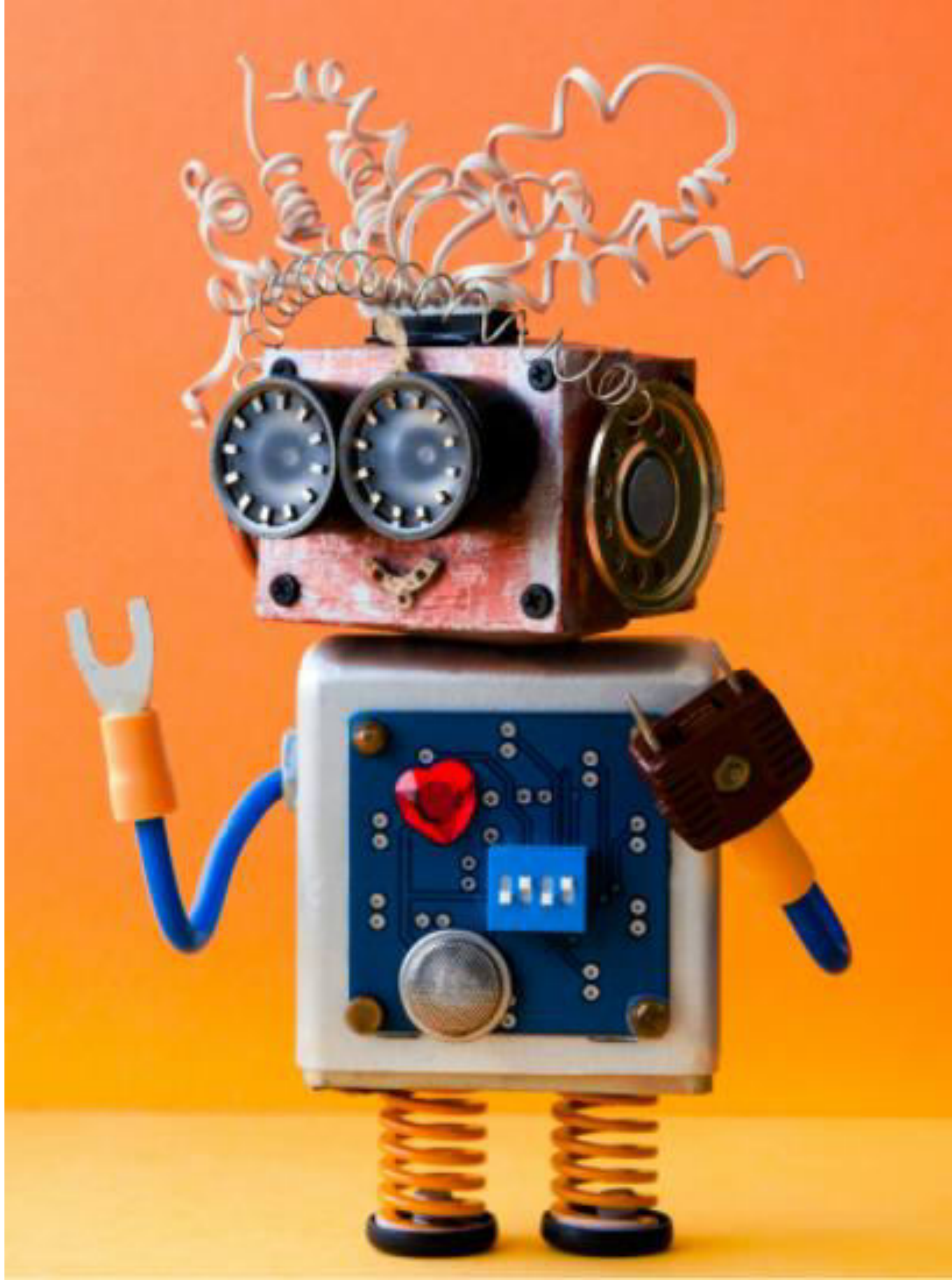
- Out of bound inference with XGBoost will cause issues and result in unreasonable predictions





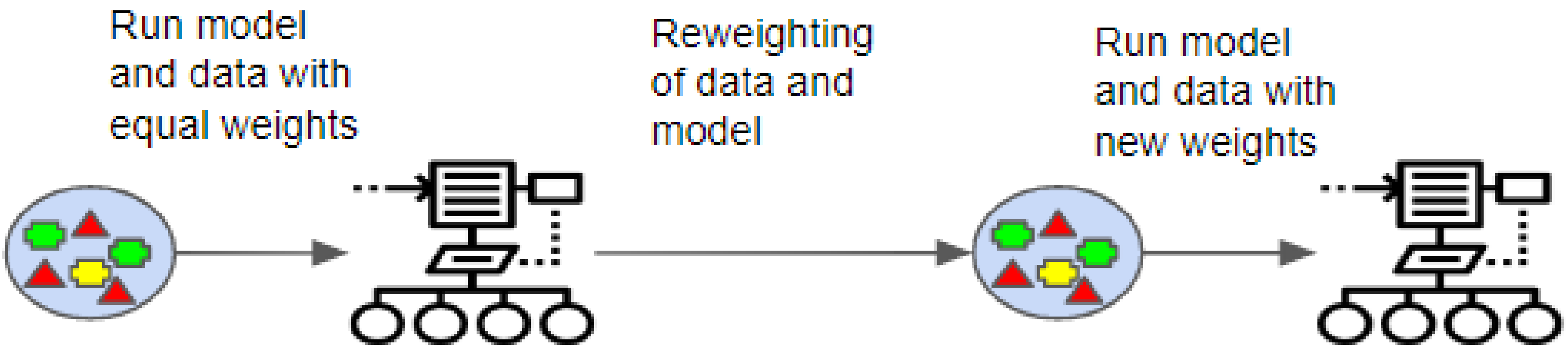
XG-BOOST ALGORITHM: WHAT IS BOOSTING?

EASY ADVANCED



XGBOOST: WHAT IS BOOSTING?

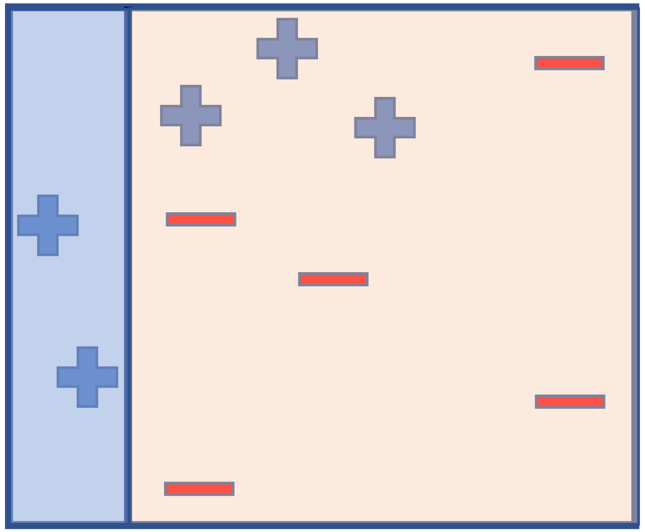
- Boosting works by learning from previous mistakes (errors in model predictions) to come up with better future predictions.
- Boosting is an ensemble machine learning technique that works by training weak models in a sequential fashion.
- Each model is trying to learn from the previous weak model and become better at making predictions.
- Boosting algorithms work by building a model from the training data, then the second model is built based on the mistakes (residuals) of the first model. The algorithm repeats until the maximum number of models have been created or until the model provides good predictions.



Great Resource: https://medium.com/greyatom/a-quick-guide-to-boosting-in-ml-acf7c1585cb5
Photo Credit: https://commons.wikimedia.org/wiki/File:Boosting.png

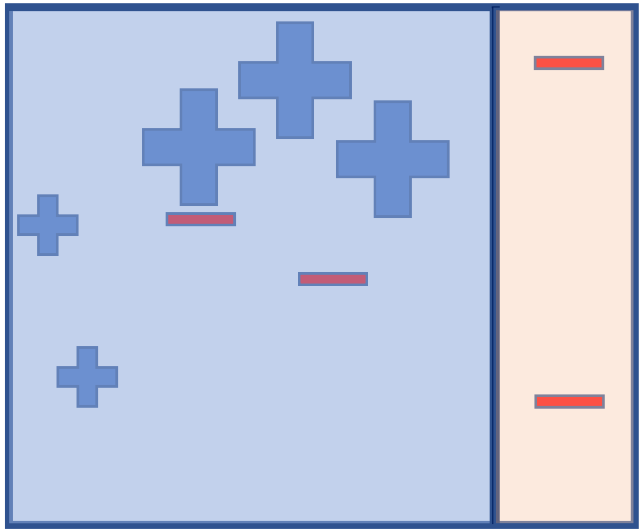
XGBOOST: BOOSTING EXAMPLE

- Model #1 works by attempting to classify the two classes (+) and (-) with the vertical line shown.
- Model #1 has assigned equal weights to all data points since it has no prior knowledge or experience from before.
- Model #1 misclassified 3 (+) samples.



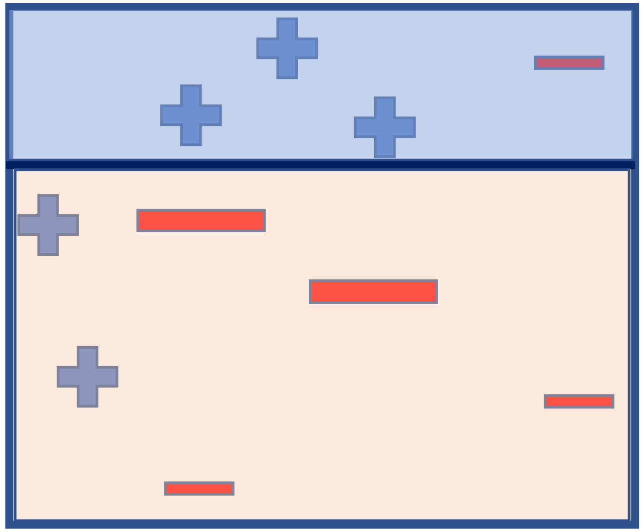
XGBOOST: BOOSTING EXAMPLE

- Model #2 learns from the mistakes of the previous model and assigns more weight to the wrongly classified data points (3 +) as shown in the figure below.
- So model #2 draws a vertical separating line and “made sure” to properly classify these points this time!
- The model did a great job correctly classifying points with higher weights but in the process, it has misclassified two red (-) samples.



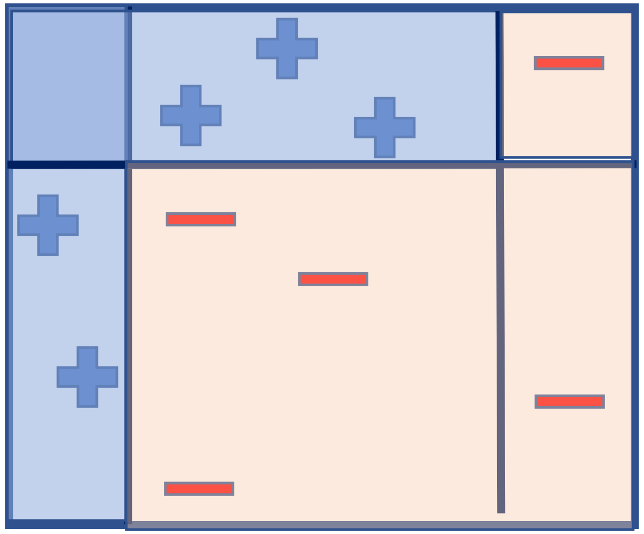
XGBOOST: BOOSTING EXAMPLE

- Model #3 learns from the mistakes of the previous model and assigns more weight to the wrongly classified data points (2 -) as shown in the figure below.
- So model #3 draws a horizontal separating line and "made sure" to properly classify these points this time!
- The model did a great job correctly classifying points with higher weights but in the process, it has misclassified two blue (+) samples.



XGBOOST: BOOSTING EXAMPLE

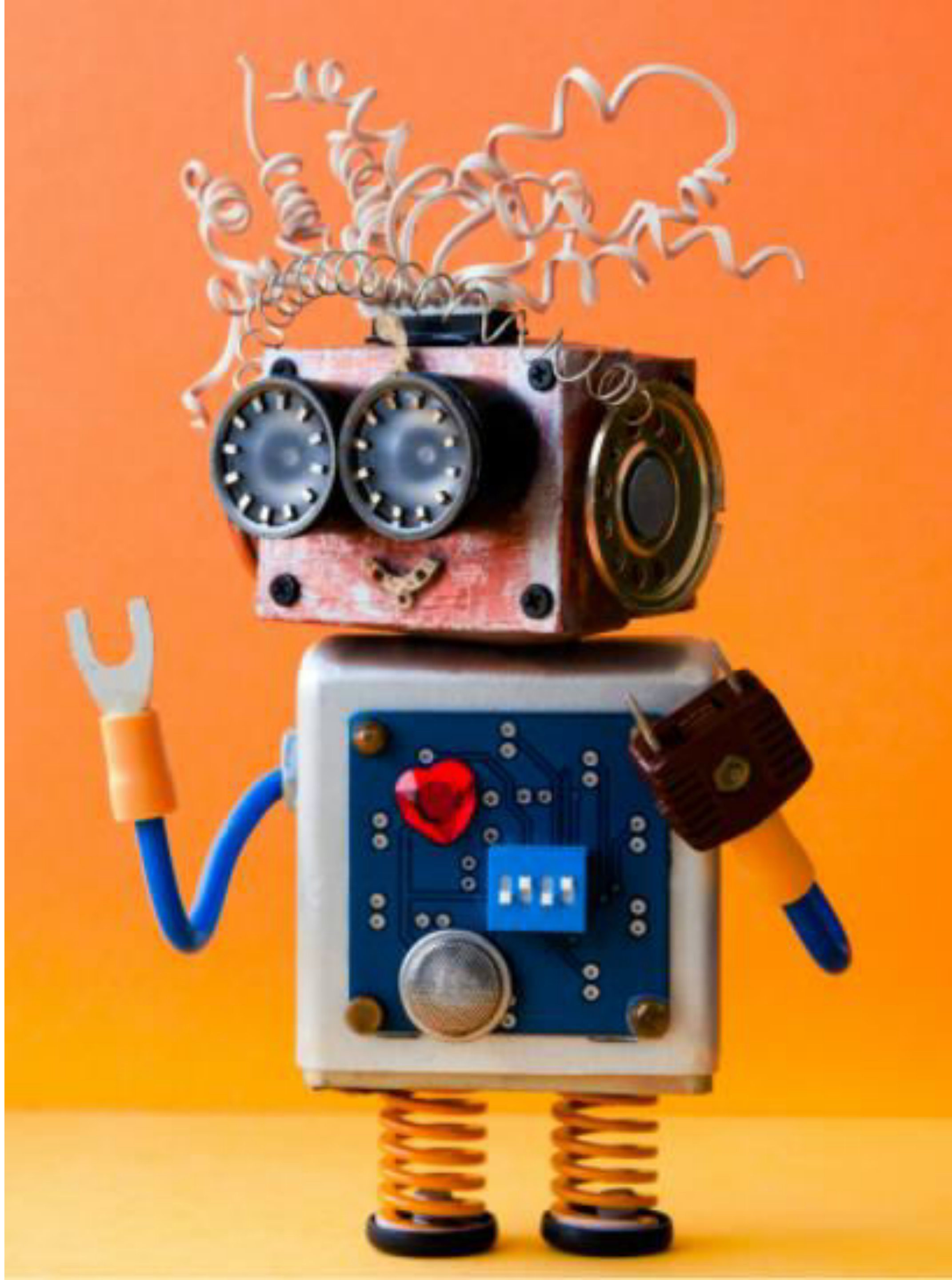
- Model #4 combines all the mistakes from all these weak models to build a much stronger model that correctly classifies all data points.



XG-BOOST ALGORITHM: ENSEMBLE LEARNING

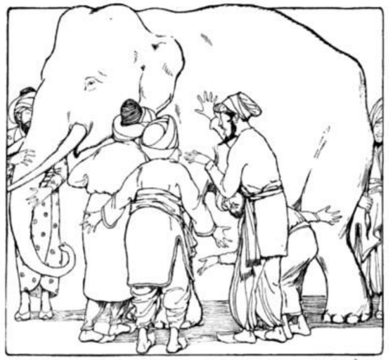
EASY

ADVANCED



XGBOOST: WHAT IS ENSEMBLE LEARNING?

- XGBoost is an example of ensemble learning.
- Ensemble techniques such as bagging and boosting can offer an extremely powerful algorithm by combining a group of relatively weak/average ones.
- For example, you can combine several decision trees to create a powerful random forest algorithm.
- By Combining votes from a pool of experts, each will bring their own experience and background to solve the problem resulting in a better outcome.
- Boosting can reduce variance and overfitting and increase the model robustness.
- Example: Blind men and the elephant



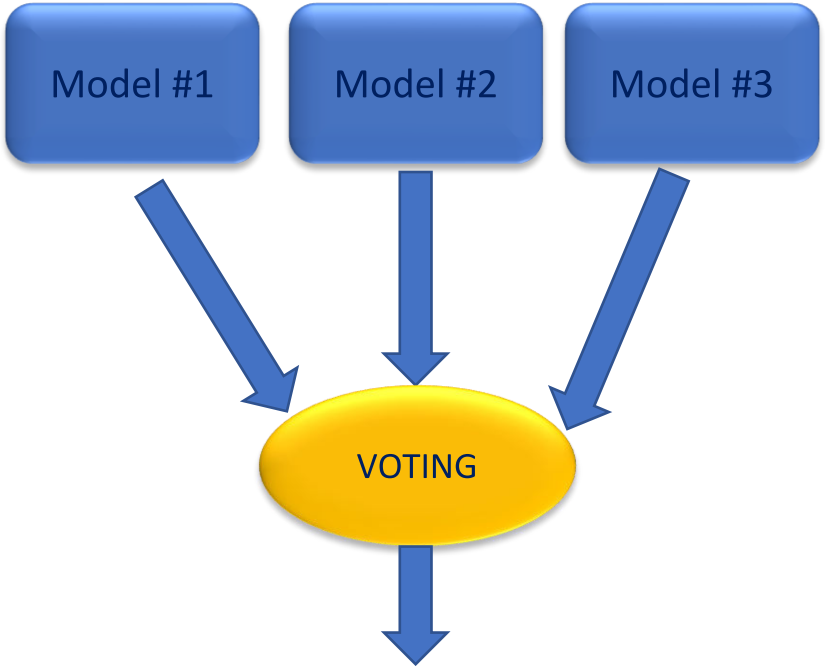
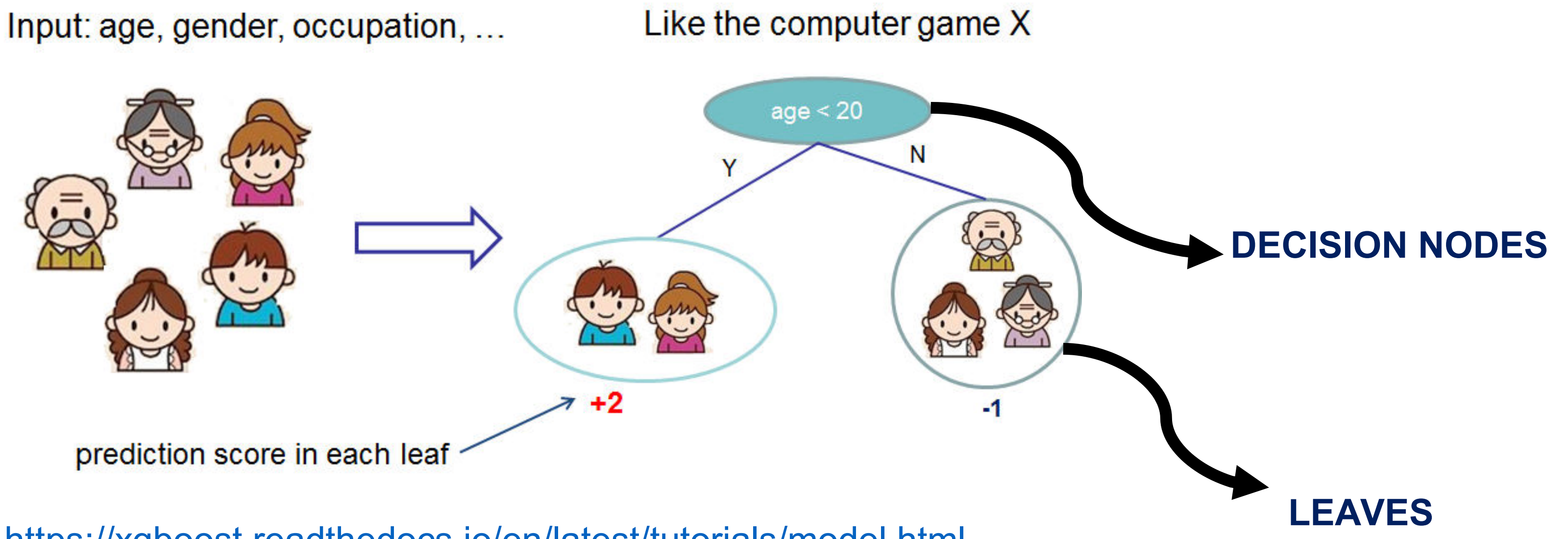


Photo Credit: https://commons.wikimedia.org/wiki/File:Blind_men_and_elephant.png

XGBOOST: DECISION TREES ENSEMBLES

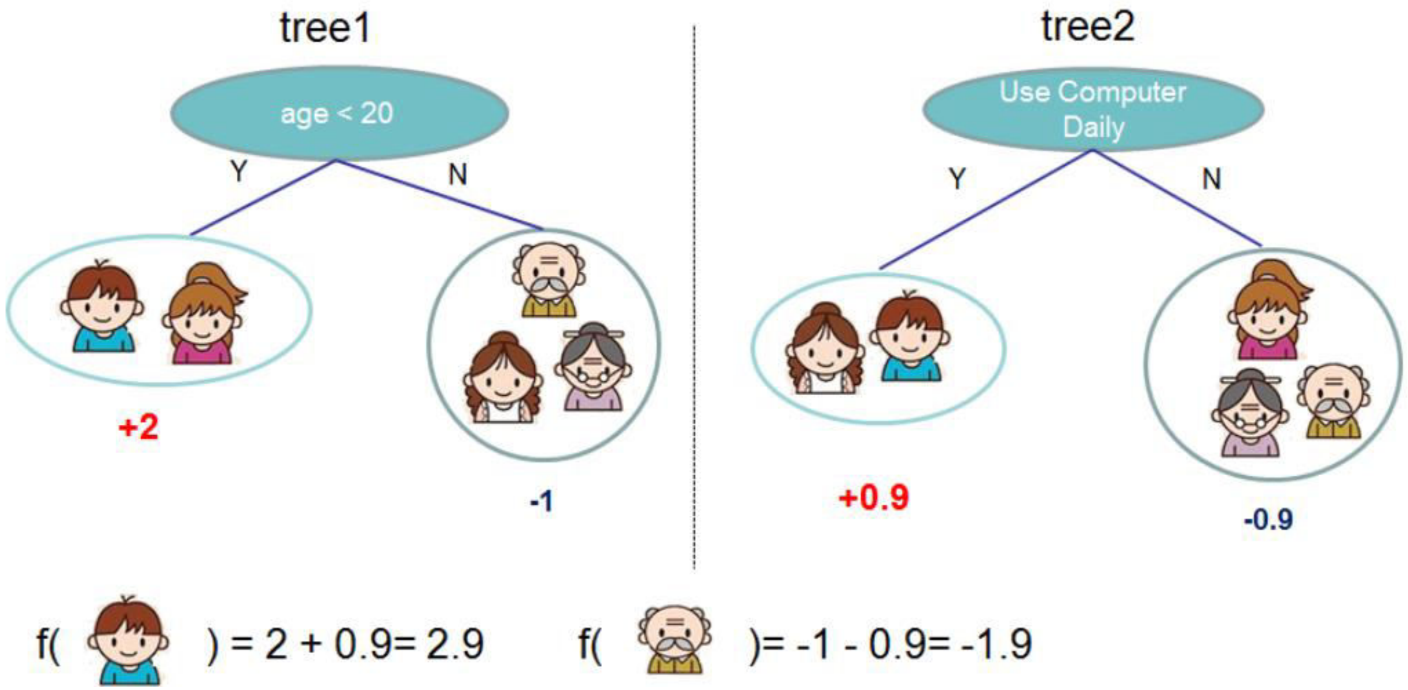
- Decision Trees are supervised Machine Learning technique where data is split according to a certain condition/parameter.
- The tree consists of decision nodes and leaves.
    - Leaves are the decisions or the final outcomes.
    - Decision nodes are where the data is split based on a certain attribute.
- The tree ensemble model consists of a set of classification and regression trees (CART).
- A CART that classifies whether an individual will like a computer game X or not is shown below.
- Members of the family are divided into leaves and a score is assigned to each leaf.



Source: https://xgboost.readthedocs.io/en/latest/tutorials/model.html

XGBOOST: DECISION TREES ENSEMBLES

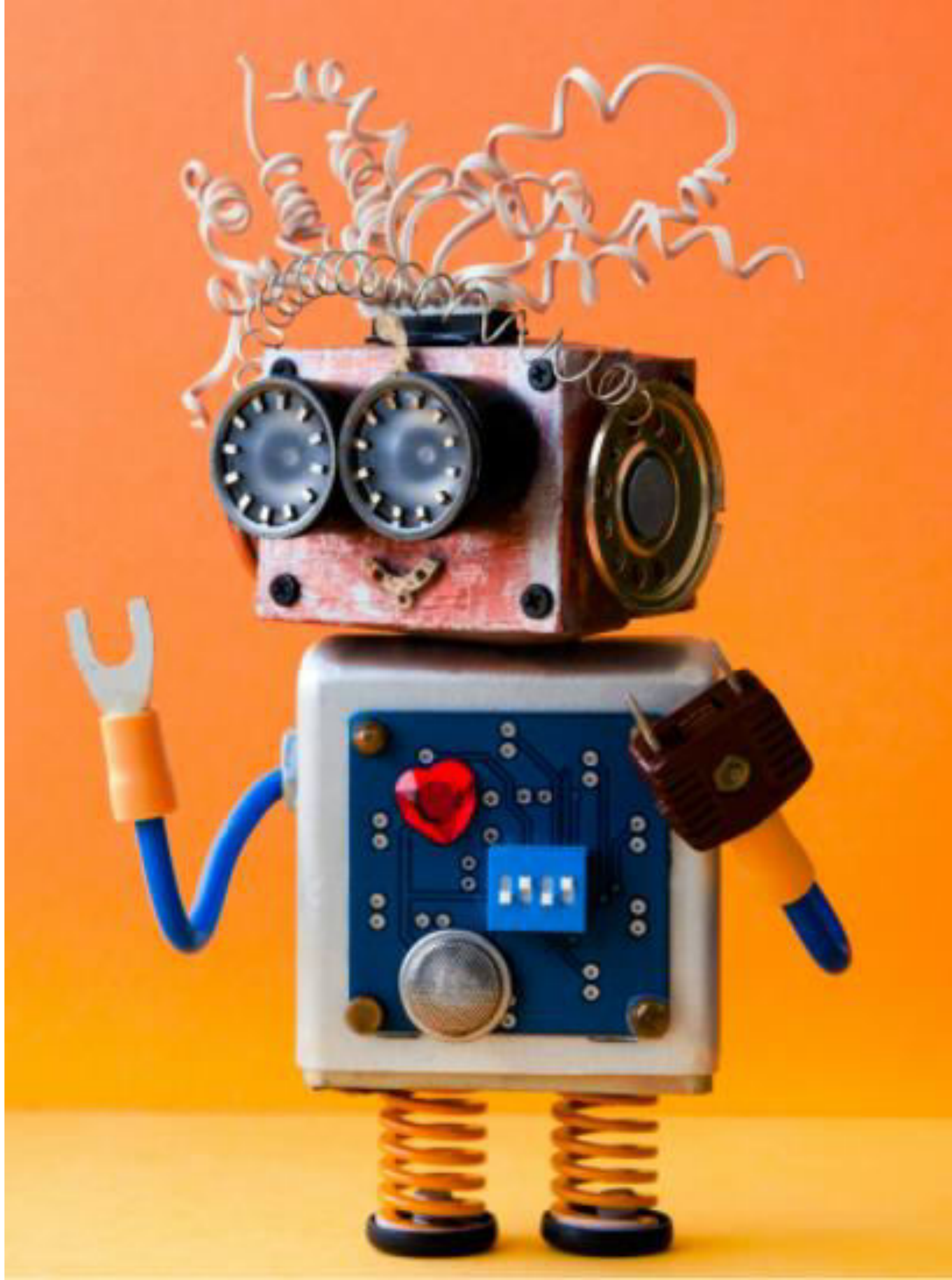
- Ensemble models that combines the predictions from all trees is built as shown below.
- The prediction scores of each individual tree are summed up to get the final score.



Source: https://xgboost.readthedocs.io/en/latest/tutorials/model.html

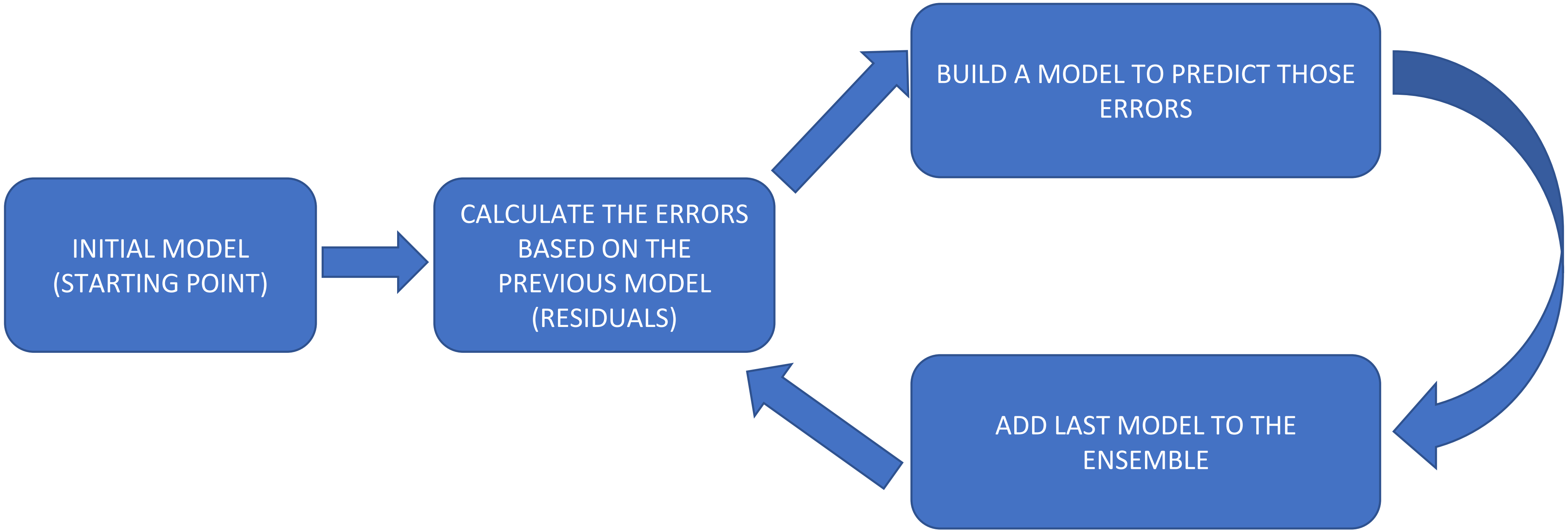
XG-BOOST ALGORITHM: DEEP DIVE

EASY ADVANCED

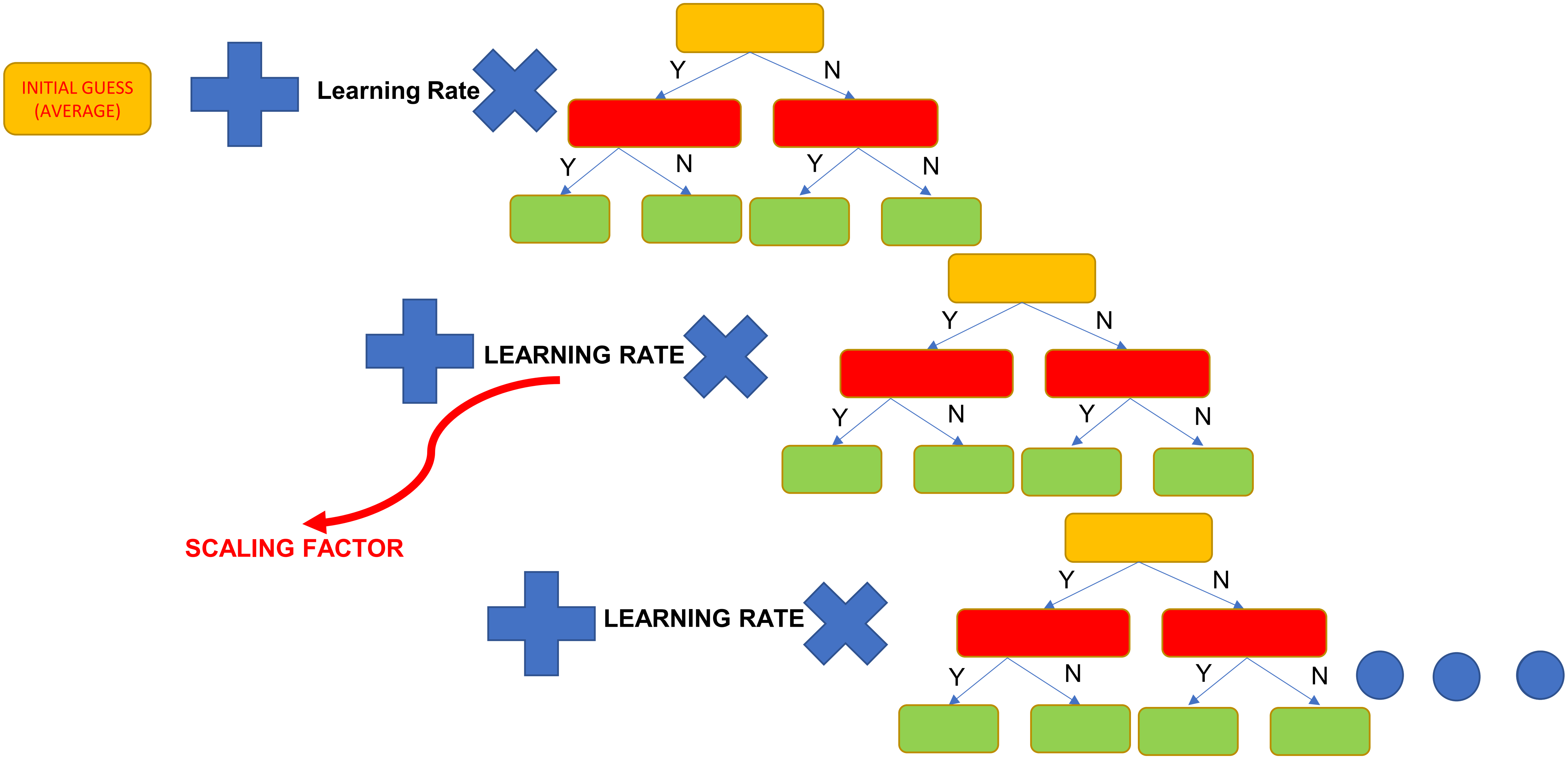


XGBOOST: STEPS

- XGBoost repeatedly builds new models and combine them into an ensemble model
- Initially build the first model and calculate the error for each observation in the dataset
- Then you build a new model to predict those residuals (errors)
- Then you add prediction from this model to the ensemble of models
- XGboost is superior compared to gradient boosting algorithm since it offers a good balance between bias and variance (Gradient boosting only optimized for the variance so tend to overfit training data while XGboost offers regularization terms that can improve model generalization).



XGBOOST: GRADIENT BOOSTING ALGORITHM



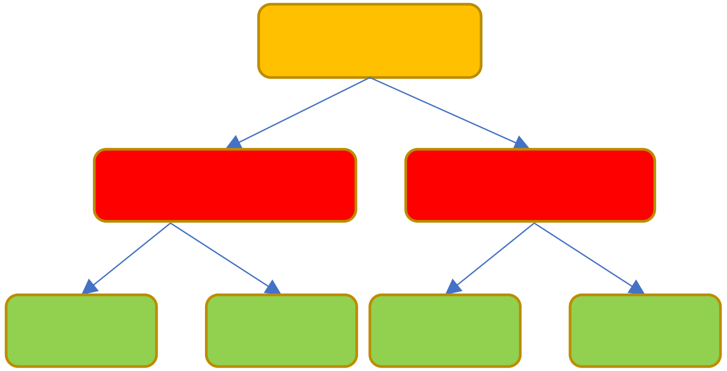
XGBOOST: GRADIENT BOOSTING ALGORITHM

- Gradient boost works by building a tree based on the error (residuals) from the previous tree.
- Gradient boost scales trees and then adds the predictions from the new tree to the predictions from previous trees
- [Example adopted from the awesome StatQuest (by Josh Starmer): https://www.youtube.com/watch?v=3CC4N4z3GJc&t=87s](https://www.youtube.com/watch?v=3CC4N4z3GJc&t=87s)

Height	Color	Gender	Weight (Kg)
1.6	Blue	Male	88
1.6	Green	Female	76
1.5	Blue	Female	56
1.8	Red	Male	73
1.5	Green	Male	77
1.4	Blue	Female	57

INPUT FEATURES

VARIABLE TO BE PREDICTED



XGBOOST: GRADIENT BOOSTING ALGORITHM

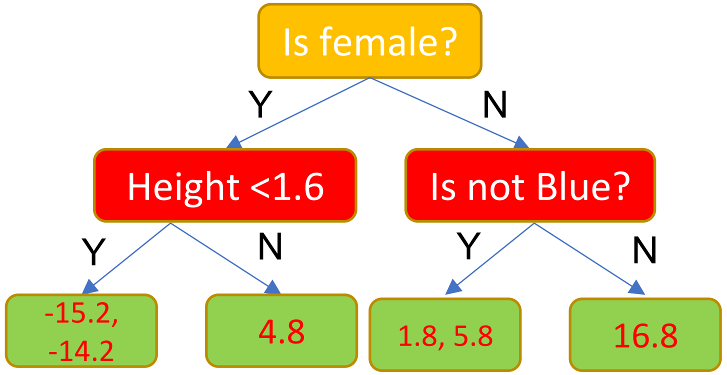
- Let’s assume that the initial model predictions (starting point) is the average weight is 71.2
- Gradient boost builds a tree based on the error from the first tree.
- The tree is built by assuming that the features (heights, color, and gender) predicts the residuals (new column that we have just created).

71.2

INITIAL STARTING POINT (PREDICTIONS)

Height	Color	Gender	Weight (Kg)
1.6	Blue	Male	88
1.6	Green	Female	76
1.5	Blue	Female	56
1.8	Red	Male	73
1.5	Green	Male	77
1.4	Blue	Female	57

Error = True – predicted
88-71.2=16.8
76-71.2=4.8
56-71.2=-15.2
73-71.2=1.8
77-71.2=5.8
57-71.2=-14.2



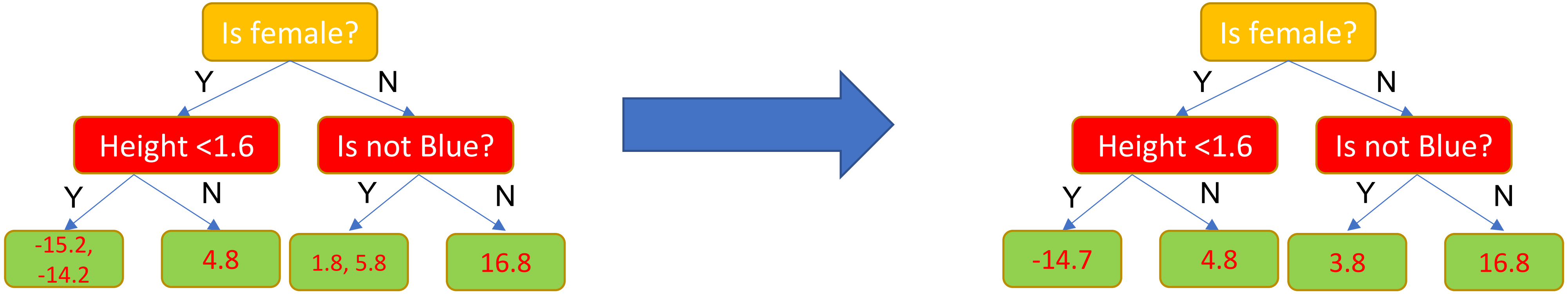
INPUT FEATURES

ERRORS (RESIDUALS)

Example adopted from the awesome StatQuest (by Josh Starmer): https://www.youtube.com/watch?v=3CC4N4z3GJc&t=87s

XGBOOST: GRADIENT BOOSTING ALGORITHM

- Note that the number of leaves is restricted to 4 in this example for the sake of simplicity.
- Let's replace the values with the average a shown below.



$$Average_1 = (-15.2 - 14.2)/2 = -14.7$$
$$Average_2 = (1.8 + 5.8)/2 = 3.8$$

[Example adopted from the awesome StatQuest (by Josh Starmer): https://www.youtube.com/watch?v=3CC4N4z3GJc&t=87s](https://www.youtube.com/watch?v=3CC4N4z3GJc&t=87s)

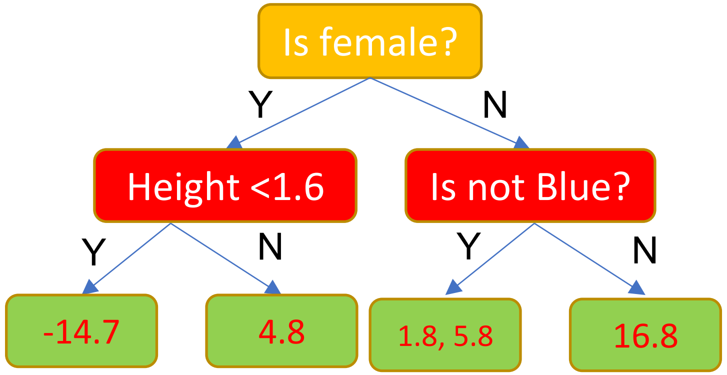
XGBOOST: GRADIENT BOOSTING ALGORITHM

- Now that we have built a tree, let's combine the previous predictions with the new tree to generate new predictions!

71.2

AVERAGE WEIGHT
(INITIAL GUESS FROM
PREVIOUS STEP

+



Height	Color	Gender	Weight (Kg)
1.6	Blue	Male	88
1.6	Green	Female	76
1.5	Blue	Female	56
1.8	Red	Male	73
1.5	Green	Male	77
1.4	Blue	Female	57

New predictions
88

The model predictions match the true weight. This indicates that the model is overfitting the training data

Predictions = 71.2+16.8=88

[Example adopted from the awesome StatQuest (by Josh Starmer): https://www.youtube.com/watch?v=3CC4N4z3GJc&t=87s](https://www.youtube.com/watch?v=3CC4N4z3GJc&t=87s)

XGBOOST: GRADIENT BOOSTING ALGORITHM

- We add a learning rate (range from 0 to 1) to overcome this issue.
- This parameter is used for scaling purposes by adjusting the newly added information from the new tree.
- Adding this tree and scaling it with the learning rate helps us get a little closer to the true values.
- By taking smaller steps, the model results in better predictions on the testing dataset (low variance).

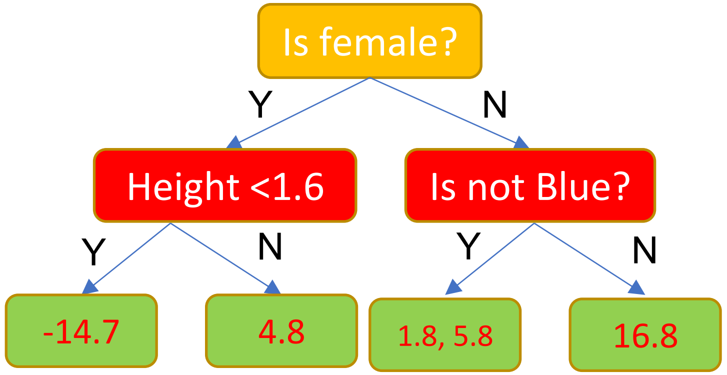
71.2

AVERAGE WEIGHT (INITIAL GUESS FROM PREVIOUS STEP

+

LEARNING RATE 0.1

×



Height	Color	Gender	Weight (Kg)
1.6	Blue	Male	88
1.6	Green	Female	76
1.5	Blue	Female	56
1.8	Red	Male	73
1.5	Green	Male	77
1.4	Blue	Female	57

New predictions
72.9

Predictions = 71.2+ 0.1 * 16.8=72.9

XGBOOST: GRADIENT BOOSTING ALGORITHM

- Now let's build another tree with the new residuals from the new predictions.

RECALL THAT THESE ARE THE INITIAL RESIDUALS

Initial Residuals
16.8
4.8
-15.2
1.8
5.8
-14.2

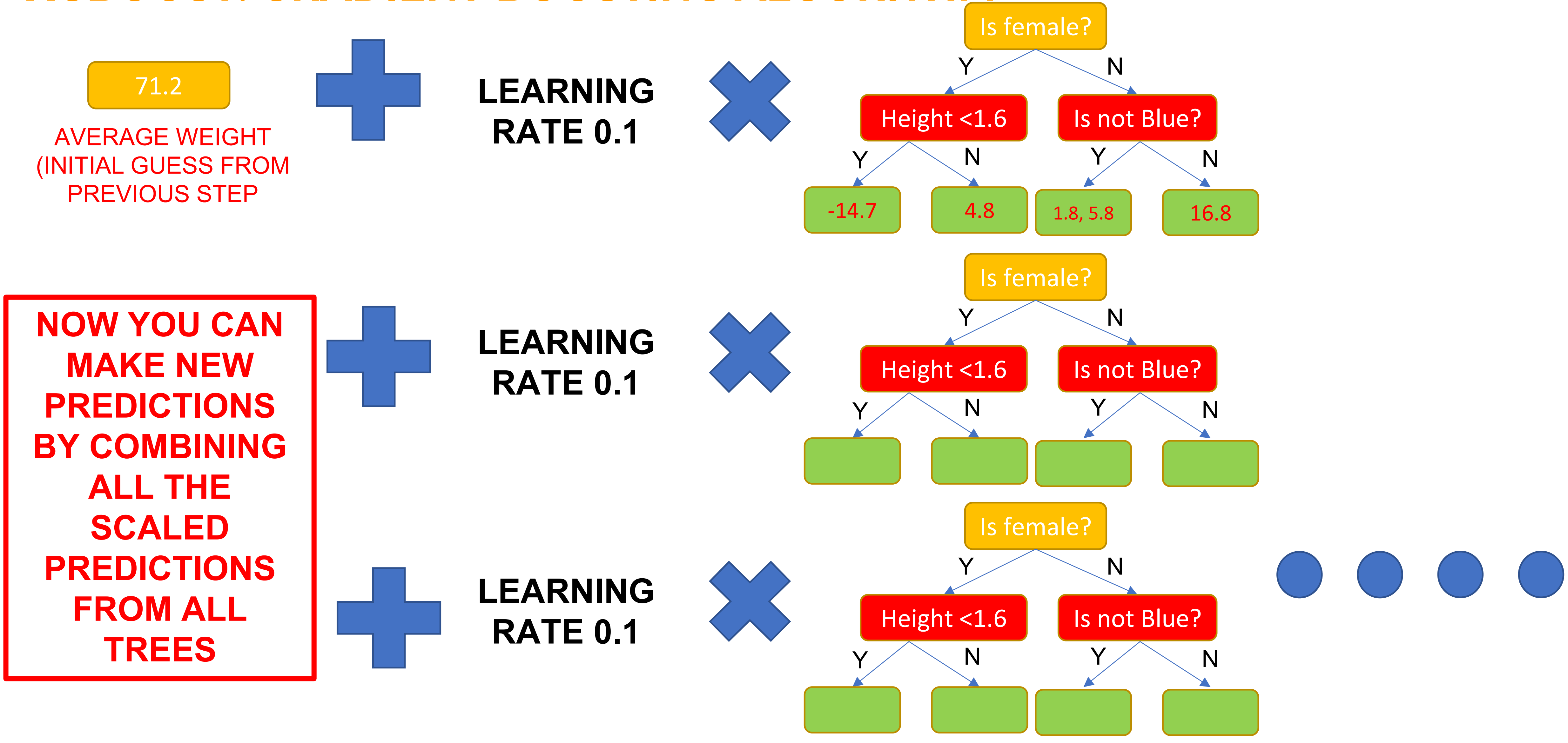
RESIDUALS HAVE GONE DOWN!

Predictions = 71.2+ 0.1 * 16.8=72.9

Height	Color	Gender	Weight (Kg)
1.6	Blue	Male	88
1.6	Green	Female	76
1.5	Blue	Female	56
1.8	Red	Male	73
1.5	Green	Male	77
1.4	Blue	Female	57

New predictions	New Residuals
72.9	88-72.9 = 15.1
	4.3
	-13.7
	1.4
	5.4
	-12.7

XGBOOST: GRADIENT BOOSTING ALGORITHM



XGBOOST: WHAT IS THE EXTREME GRADIENT BOOSTING THEN? GREAT RESOURCES

Paper: https://arxiv.org/pdf/1603.02754.pdf
https://xgboost.readthedocs.io/en/latest/tutorials/model.html

XGBoost: A Scalable Tree Boosting System

Tianqi Chen
University of Washington
tqchen@cs.washington.edu

Carlos Guestrin
University of Washington
guestrin@cs.washington.edu

ABSTRACT

Tree boosting is a highly effective and widely used machine learning method. In this paper, we describe a scalable end-to-end tree boosting system called XGBoost, which is used widely by data scientists to achieve state-of-the-art results on many machine learning challenges. We propose a novel sparsity-aware algorithm for sparse data and weighted quantile sketch for approximate tree learning. More importantly, we provide insights on cache access patterns, data compression and sharding to build a scalable tree boosting system. By combining these insights, XGBoost scales beyond billions of examples using far fewer resources than existing systems.

Keywords

Large-scale Machine Learning

problems. Besides being used as a stand-alone predictor, it is also incorporated into real-world production pipelines for ad click through rate prediction [15]. Finally, it is the de-facto choice of ensemble method and is used in challenges such as the Netflix prize [3].

In this paper, we describe XGBoost, a scalable machine learning system for tree boosting. The system is available as an open source package[2]. The impact of the system has been widely recognized in a number of machine learning and data mining challenges. Take the challenges hosted by the machine learning competition site Kaggle for example. Among the 29 challenge winning solutions [3] published at Kaggle's blog during 2015, 17 solutions used XGBoost. Among these solutions, eight solely used XGBoost to train the model, while most others combined XGBoost with neural nets in ensembles. For comparison, the second most popular method, deep neural nets, was used in 11 solutions. The success

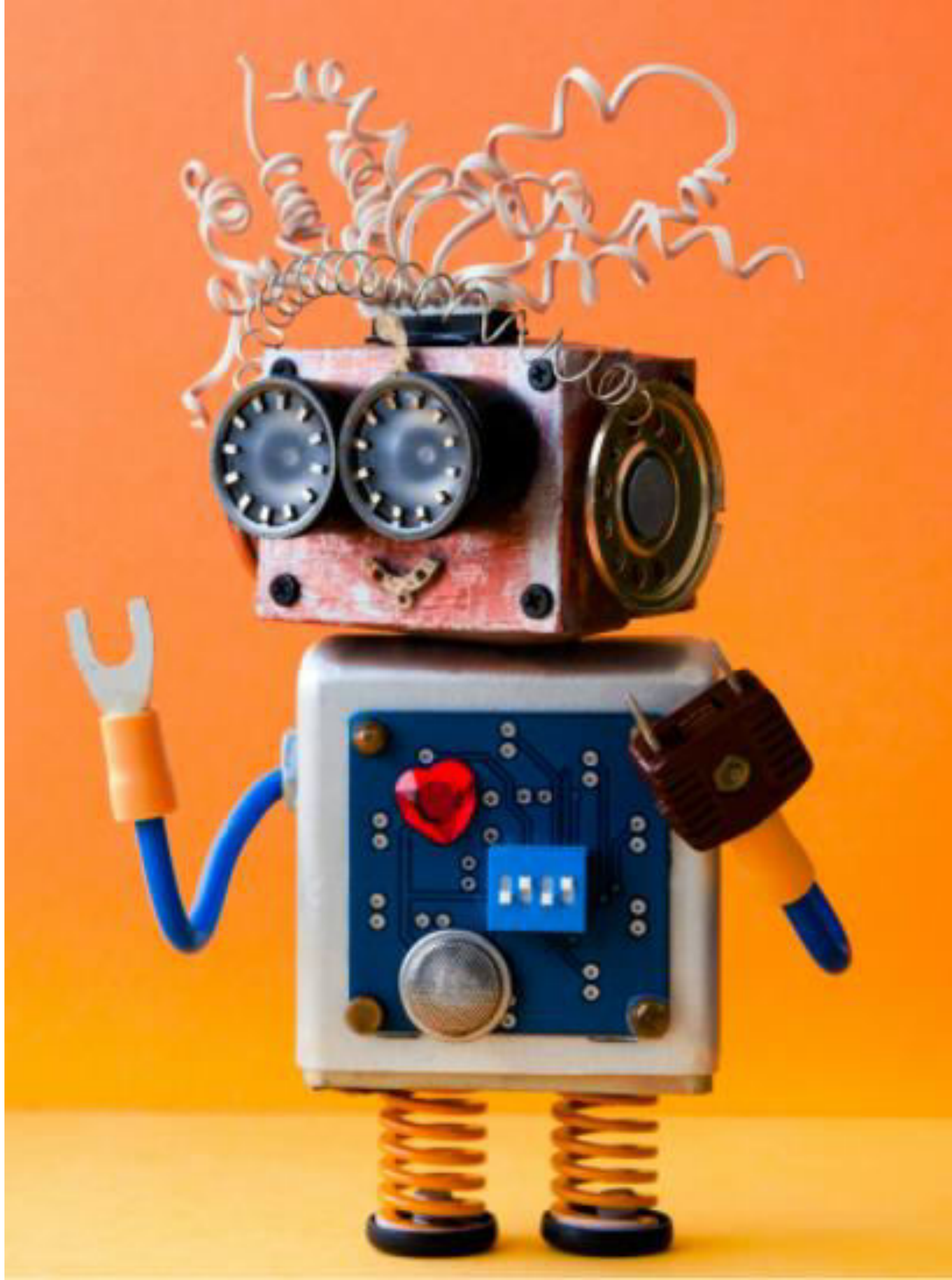
XGBOOST: PAPER HIGHLIGHTS

- *"The most important factor behind the success of XGBoost is its scalability in all scenarios. The system runs more than **ten times faster than existing popular solutions** on a single machine and **scales to billions of examples** in **distributed or memory-limited** settings."*
- *"The scalability of XGBoost is due to several important systems and **algorithmic optimizations**. These innovations include: **a novel tree learning algorithm** is for handling **sparse data**; a theoretically justified **weighted quantile sketch procedure** enables handling instance weights in approximate tree learning. **Parallel and distributed computing** makes learning faster which enables quicker model exploration".*
- *"More importantly, XGBoost exploits **out-of-core computation** and enables data scientists to process hundred millions of examples on a desktop".*
- *"Finally, it is even more exciting to combine these techniques to make an end-to-end system that scales to even larger data with the least amount of cluster resources".*

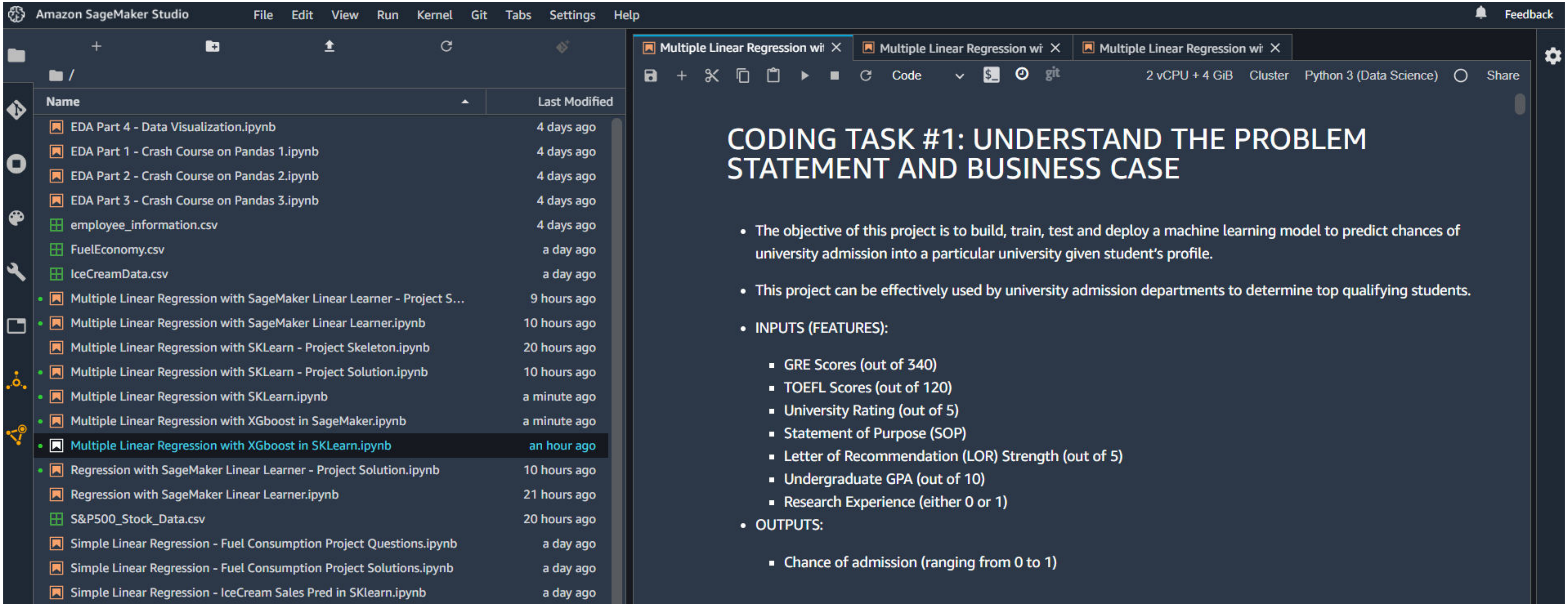
Paper: https://arxiv.org/pdf/1603.02754.pdf

CODE DEMO

EASY ADVANCED

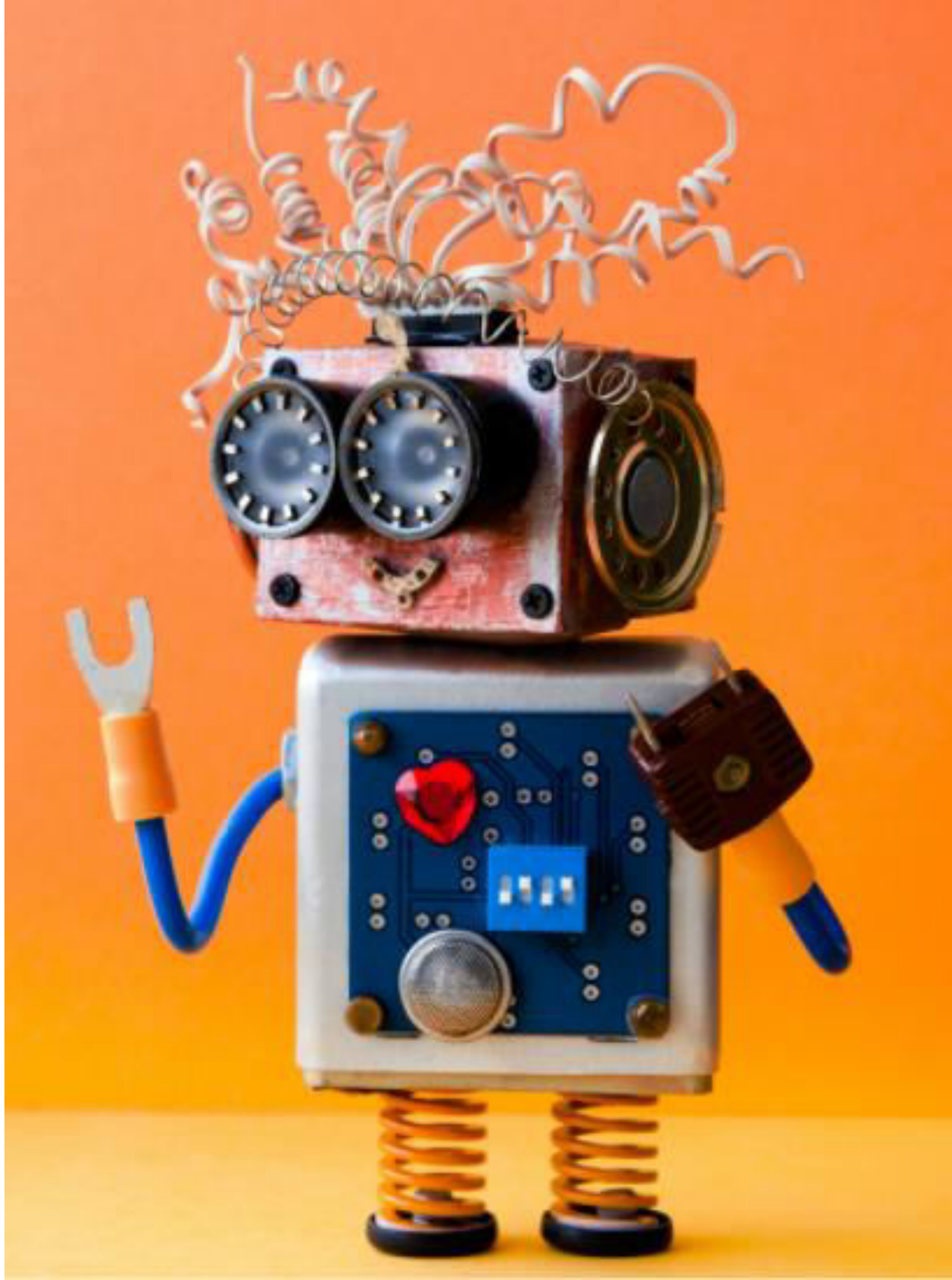


CODE DEMO: XG-BOOST IN SAGEMAKER



FINAL END-OF-DAY CAPSTONE PROJECT

EASY ADVANCED



PROJECT OVERVIEW: LIFE EXPECTANCY PREDICTION

- In this hands-on project, we will train an XG-Boost model to predict life expectancy.
- This data was initially obtained from World Health Organization (WHO) and United Nations Website. Data contains features like year, status, life expectancy, adult mortality, infant deaths, percentage of expenditure, alcohol etc.
- **Tasks:**
  1. Perform exploratory data analysis
  2. Train an XG-boost algorithm in Scikit-Learn
  3. Assess trained model performance
  4. Plot trained model predictions vs. ground truth output

INPUT FEATURES

YEAR

ADULT MORTALITY

STATUS

INFANT DEATHS

ALCOHOL

HEPATITIS B

PREDICTION

LIFE EXPECTANCY

Source: https://www.kaggle.com/jkumarajarshi/life-expectancy-who