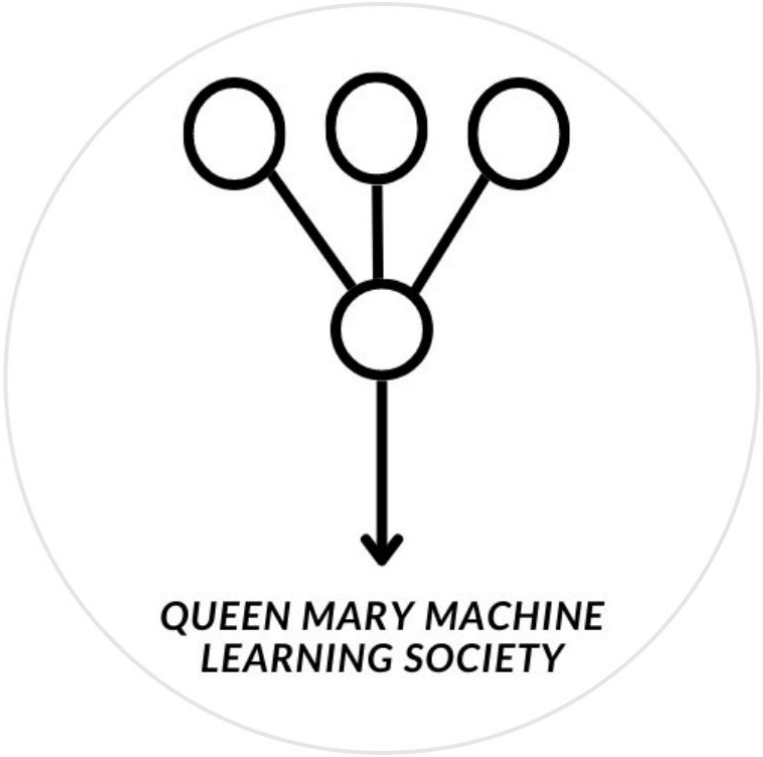
# Kaggle Seasons #07



# Last session recap

- Last session, we introduced gradient boosting
- We saw that gradient boosting was a form of boosting
- We covered why gradient boosting usually works best with decision trees
- We understood why gradient boosting is so important in machine learning

# This session

- We'll go walk through a general boosting algorithm step by step
- We'll work through a simple concrete boosting algorithm
- We'll see how gradient boosting is an improvement on boosting
- We implement a gradient boosting algorithm in Python in a live coding session

# Boosting step by step

1) Initial prediction
2) Compute residuals
3) Train a weak model on residuals
4) Update the model
5) Repeat for multiple rounds

# STEP 1: Initial Prediction

Our initial prediction, $F_0$, can be as simple as an **average** of the target values:

$$F_0(x) = \frac{1}{n}\sum y_i$$

kaggle

# STEP 2: Compute Residuals

Residuals measure how far off our prediction is.

Recall that a residual is the difference between the predicted value and the target:

$$\text{Residual}_i = y_i - \hat{y}_i$$

At step m, the predicted value is $\hat{y}_i = F_m(x_i)$, so the residual is:

$$\text{Residual}_i = y_i - F_m(x_i)$$

This tells us how much our prediction is **off** by at step m.

# STEP 3: Train a Weak Model on Residuals

We now train a small model, $h_{m+1}(x)$, on the residuals:

$$h_{m+1}(x) \approx \text{Residual}_i$$

This tree is learning the "correction" needed to improve our predictions.This tree is learning the "correction" needed to improve our predictions.

# STEP 4: Update the Model

We calibrate our model by adding a learning rate, v, and then add the estimated residuals to our original model:

$$F_{m+1}(x) = F_m(x) + \nu h_{m+1}(x)$$

where:

- v is the learning rate (usually a small number like 0.1)
- $h_{m+1}(x)$ is the residual model's output

kaggle

# STEP 5: Repeat for Multiple Rounds

After iterating steps 1 through 4 m times, our final model is:

$$F_M(x) = F_1(x) + \sum_{m=1}^{M-1} \nu h_{m+1}(x)$$

# Example algorithm (Steps 1 through 3)

**Step 1: Initial Prediction**

$$F_1(x) = \frac{10 + 12 + 14}{3} = 12$$

**Step 2: Compute Residuals**

Since $F_1(x)$ is the first prediction ($\hat{y}_i$):

$$\hat{y}_i = F_1(x_i) = 12$$

Now, compute residuals:

$$\text{Residual}_1 = 10 - 12 = -2$$

$$\text{Residual}_2 = 12 - 12 = 0$$

$$\text{Residual}_3 = 14 - 12 = 2$$

**Step 3: Train a Tree on Residuals**

A small tree learns:

$$h_2(x) = [-2, 0, 2]$$

# Example algorithm (Steps 4 through 5)

**Step 4: Update the Model**

With $\nu = 0.5$, the updated model:

$$F_2(x) = F_1(x) + 0.5h_2(x)$$

For each point:

$$F_2(1) = 12 + 0.5(-2) = 11$$

$$F_2(2) = 12 + 0.5(0) = 12$$

$$F_2(3) = 12 + 0.5(2) = 13$$

**Step 5: Compute New Residuals**

Using the updated predictions:

$$\text{Residual}_1^{(2)} = 10 - 11 = -1$$

$$\text{Residual}_2^{(2)} = 12 - 12 = 0$$

$$\text{Residual}_3^{(2)} = 14 - 13 = 1$$

Now repeat steps **3–5** for the next iteration!

# Example algorithm (Summary)

## Final Summary

1. Start with an initial guess:

$$F_1(x) = \frac{1}{n}\sum y_i$$

2. Compute residuals:

$$\text{Residual}_i = y_i - \hat{y}_i$$

where $\hat{y}_i = F_m(x_i)$.

3. **Train a small tree** on residuals.

4. **Update the model**:

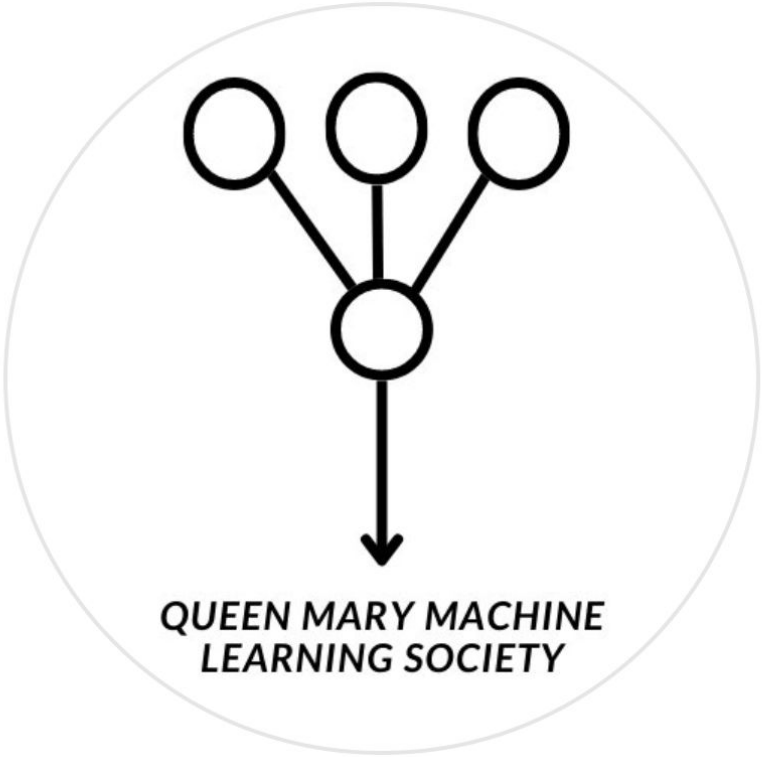
$$F_{m+1}(x) = F_m(x) + \nu h_{m+1}(x)$$

5. **Repeat until the model improves!**

# Gradient boosting

- Gradient boosting is a form of generalised form of boosting which uses a computational trick to reduce overfitting of our residual models (h) due to the noisiness of the residuals
- Let us know if you would like a more in-depth exploration of gradient boosting specifically in our following sessions!
- Prominent examples of gradient boosting include XGBoost, CatBoost, and LightGBM
- We will be implementing XGBoost in our live coding session hosted by Karl

# Live Coding



# Thanks for listening!