Boosting Machines

EDVANCER
EDUVENTURES

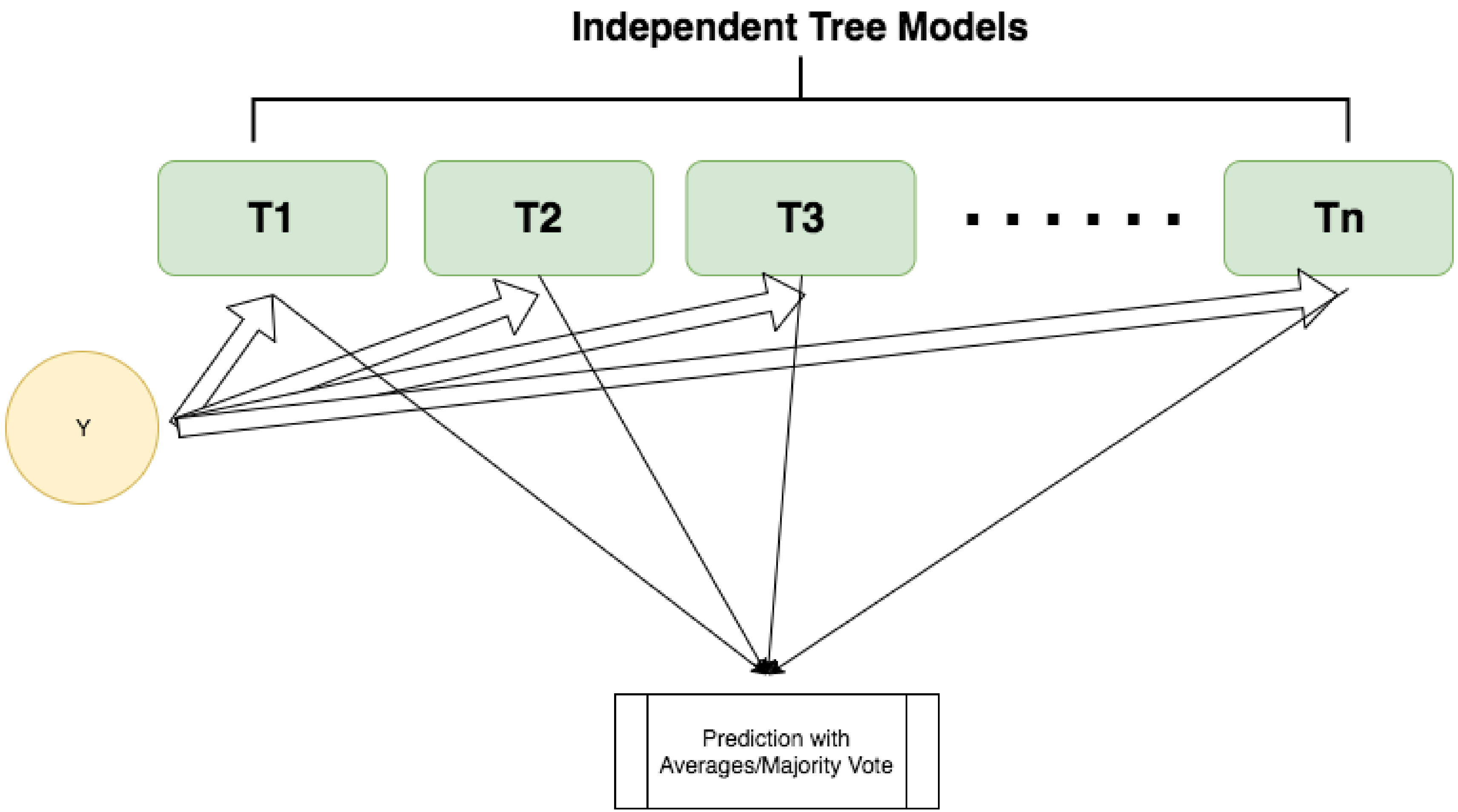
Agenda

Discussion Flow

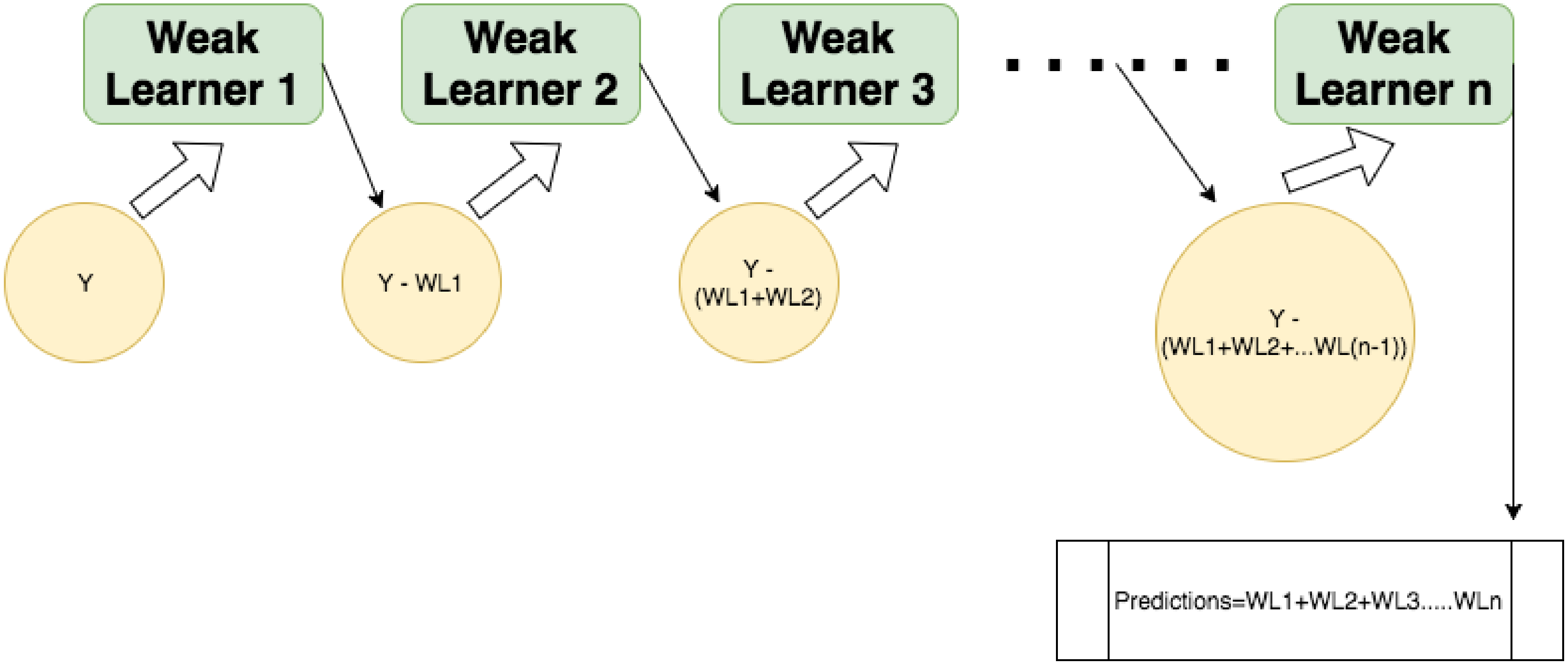
- Bagging Vs Boosting
- Decision tree stumps as weak learners
- Gradient boosting machines
- Boosting machines for Regression
- Boosting machines for classification
- Sklearn Implementation

Bagging Vs Boosting

Bagging


Independent Tree Models
T1
T2
T3
Tn
Y
Prediction with
Averages/Majority Vote

Boosting



Weak Learner 1
Weak Learner 2
Weak Learner 3
Weak Learner n
Y
Y - WL1
Y - (WL1+WL2)
Y - (WL1+WL2+...WL(n-1))
Predictions=WL1+WL2+WL3.....WLn

Weak Learner

What ?

- Simple models which can capture generic patterns
- Examples :
  - Linear models with subset of variables
  - Linear models with heavy penalty
  - Decision Tree Stumps (Shallow tree with low number of splits)

Why ?

- Inability to learn niche patterns , difficult to overfit
- Strong learners for the same reason will lead to overfit
- Weak learners emphasise capture of patterns which are generic and yet in combination can make very strong model overall

Decision Tree Stumps as Weak Learner

- Decision Tree Stumps are a popular choice of weak learner
- Easy to implement and shallow trees can learn simple non-linear patterns too

Boosting Machines

Incremental Nature of Boosting Machine Models

$$Prediction\ at\ iteration\ t = F_t(X) = \sum_{i=0}^{t} f_t(X)$$

$$where\ f_t(X)\ is\ t^{th}\ weak\ learner$$

Gradient Descent in Functional Space

$$J = \sum L(y_i, F_t(X_i))$$

$$\frac{\delta J}{\delta F_t(X)} = \sum \frac{\delta L(y_i, F_t(X_i))}{\delta F_t(X)}$$

$$f_{t+1}(X) \rightarrow -\eta \frac{\delta J}{\delta F_t(X)}$$

$$F_{t+1}(X) = F_t(X) + f_{t+1}(X)$$

GBM for regression

$$L(y_i, F_t(x_i)) = (y_i - F_t(x_i))^2$$

$$\frac{\delta L}{\delta F_t(x)} \sim -(y_i - F_t(x_i))$$

$$f_{t+1}(x) \rightarrow -\eta \frac{\delta L}{\delta F_t(x)} \rightarrow \eta(y_i - F_t(x_i))$$

GBM for classification

Model

$$p_i^{(t)} = \frac{1}{1 + e^{-F_t(x_i)}}$$

Loss Function

$$L(y_i, F_t(x_i)) = -(y_i * log(p_i^{(t)}) + (1 - y_i) * log(1 - p_i^{(t)}))$$
$$= log(1 + e^{F_t(x_i)}) - y_i * F_t(x_i)$$

Gradient

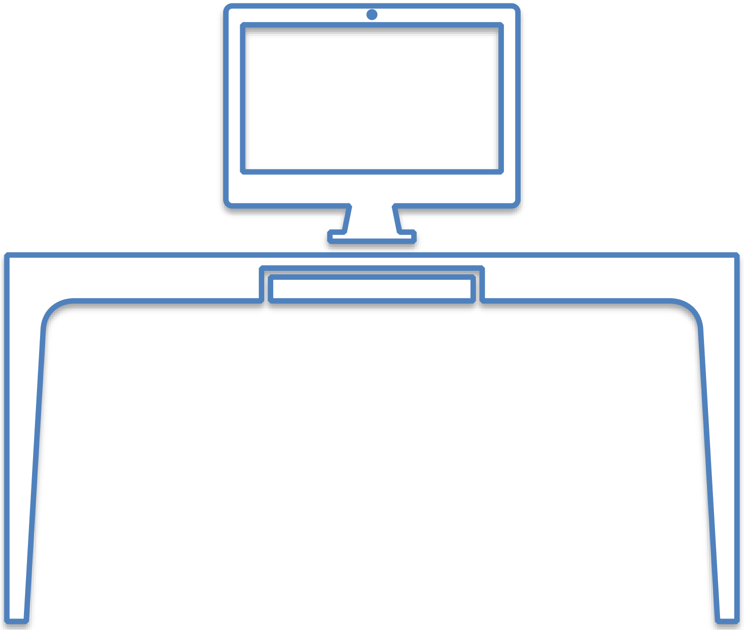
$$\frac{\delta J}{\delta F_t(x_i)} = -(y_i - \frac{1}{1 + e^{-F_t(x_i)}})$$
$$= -(y_i - p_i^{(t)})$$

A Regression Tree

Next Weak Learner

$$f_{t+1}(x) \rightarrow \eta(y_i - p_i^{(t)})$$

Lets see it in action in Python



Issues with usual GBM

- Loss function doesn't consider complexity of the model
- Leads to overfit and not so generalised error performance
- xgboost uses new objective function, which adds model complexity to the traditional loss function