Machine Learning and Computational Statistics
Trees

Nhung Le

Note: This document consists of concepts and exercises related to trees, arguably one of the most powerful machine learning models.

1 Decision Trees

Under our scope, we will only consider binary decision trees with the following characteristics.

1. Binary Decision Tree: Each node has 2 children or 0 children.
2. Decision at each node involve only a single feature (i.e., input coordinate).
3. Splitting rules:
   - Continuous variables: $x_i \leq t$
   - Discrete variables, partitions values into two groups
4. How to control the complexity of a tree to avoid overfitting
   - Increase the minimum number of instance in each leaf node.
   - Limit max depth of tree.
   - Require a node to have at least a certain number of data points to split.
   - Pre-Pruning (early stopping): stop the algorithm before it becomes a fully-grown tree using stopping conditions
     - Stop if all instances belong to the same class
     - Stop if all the attribute values are the same
     - Stop if expanding the current node does not improve impurity measures (e.g., Gini, Information gain)
   - Post-Pruning:
     - Grow decision tree to its entirety.
     - Trim the nodes of the decision tree in a bottom-up fashion.
     - If generalization error improves after trimming, replace sub-tree by a leaf node.
     - Class label of leaf node is determined from majority class of instances in the sub-tree
5. Decision Tree Class (pseudo code)

- Initialize the decision tree classifier
  - param split-loss-function: method for splitting node
  - param leaf-value-estimator: method for estimating leaf value
  - param depth: depth indicator, default value is 0, representing root node
  - param min-sample: an internal node can be splitted only if it contains points more than min-sample
  - param max-depth: restriction of tree depth.
- Fit the tree
  - If leaf: if depth $\geq$ max-depth or n-samples $\leq$ min-sample
  - If not leaf:
    - Find the optimal splitting point by looping through every points in n-samples and every feature to find the feature and the point (i.e., cut-off value) for the split using the split-loss-function
    - Given the optimal splitting point, get all the points on the left branch and on the right branch
  - Recursively fit trees on left and right branches
- Predict
  Predict label by decision tree (i.e., each point will go through each node level of tree to find its appropriate region and the class/value of that data-point is decided at the leaf node

2 Classification Trees

1. Node Impurity Measurements - aka Splitting Loss Functions
   - Entropy
     - For a node with data $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)$
     - For class $i = 1, 2, .., k$, calculate probability of class i such that $p_i = \frac{\texttt{Number of data in class i}}{\texttt{number of data points in this node (i.e.}}$
     - Entropy of class i : $\texttt{entropy}_i = -p_i * log_2 p_i$
     - Entropy of this node (i.e., branch): $\sum_{i=1}^{k} \texttt{entropy}_i$
     - Weighted entropy of this split: $\frac{n_{left}}{n_{left}+n_{right}} * \texttt{entropy}_{left} + \frac{n_{right}}{n_{left}+n_{right}} * \texttt{entropy}_{right}$
     - **Objective:** minimize entropy
   - Gini
     - For a node with data $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)$
     - For class $i = 1, 2, .., k$, calculate probability of class i such that $p_i = \frac{\texttt{Number of data in class i}}{\texttt{number of data points in this node (i.e.}}$
     - Score of class i : $\texttt{score}_i = p_i * (1 - p_i)$
     - Gini score of this node (i.e., branch): $\sum_{i=1}^{k} \texttt{score}_i$
     - Weighted Gini score of this split: $\frac{n_{left}}{n_{left}+n_{right}} * \texttt{score}_{left} + \frac{n_{right}}{n_{left}+n_{right}} * \texttt{score}_{right}$
     - **Objective:** maximize Gini score
   - Mis-classification error

- For a node (i.e., branch) with data $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)$
- For class $i = 1, 2, .., k$, calculate probability of class i such that $p_i = \frac{\texttt{Number of data in class i}}{\texttt{number of data points in this node (i.e.}}$
- Mis-classification error of this branch $me = 1 - max_{p_i}$
- Weighted mis-classification error of this split $\frac{n_{left}}{n_{left}+n_{right}} * me_{left} + \frac{n_{right}}{n_{left}+n_{right}} * me_{right}$
- **Objective:** Minimize mis-classification error

2. Estimation function used in leaf node: the value of this leaf would be the most popular class
3. Fit and predict using Decision Tree class

3 Regression Trees

1. Node Impurity Measurements - aka Splitting Loss Functions
   - Mean-square-error
   - mean-absolute-deviation-around-median
2. Estimation function used in leaf node: the value of this leaf would be mean (average) or median of the values y of all training data points of this node
3. Fit and predict using Decision Tree class

4 Bootstrap

- Parameters: parameters are characteristics of a probability distribution P. Parameters include expected value, variance, kurtosis, median. Parameters are NOT random.
- Statistics: Suppose we have $D_n = (x_1, x_2, \cdots, x_n)$ is an i.i.d. sample from P. A statistic $s = s(D_n)$ is any function of the data. Statistics are random, so they have probability distributions, each is called a **sampling distribution**
  Examples of statistics:
  - mean: $\bar{x}(D_n) = \frac{1}{n}\sum_{i=1}^{n} x_i$
  - median: $m(D_n) = median(x_1, x_2, \cdots, x_n)$
  - sample variance $\sigma^2(D_n) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x}(D_n))^2$
  - a data histogram
  - empirical distribution function
  - a confidence interval
- Point estimator: a statistic (e.g., $\hat{\mu}$) is a point estimator of μ if $\hat{\mu} \approx \mu$
- Bias and Variance
  - Let $\mu = \mu(P)$ be a real-valued parameter of distribution P
  - Let $\hat{\mu} = \hat{\mu}(D_n)$ be a point estimator of μ

  - Bias($\hat{mu}$) = $E(\hat{\mu}) - \mu$
  - Variance($\hat{mu}$) = $E(\hat{\mu}^2) - (E(\hat{\mu}))^2$

- The Bootstrap Method: stimulate having B independent samples from P by taking B bootstrap samples from the sample D_n.

5 Bagging and Random Forest

1. Parallel ensembles: each model is built independently
2. Combine many high complexity, low bias models to reduce variance – the power of averaging
3. Bagging and bootstrap:
   - Draw B bootstraps samples $D_1, D_2, \cdots, D_B$ from original data D
   - Let $\hat{f}_1, \hat{f}_2, \cdots, \hat{f_B}$ be the prediction functions from training on $D_1, D_2, \cdots, D_B$ respectively
   - **Bagged prediction function**:

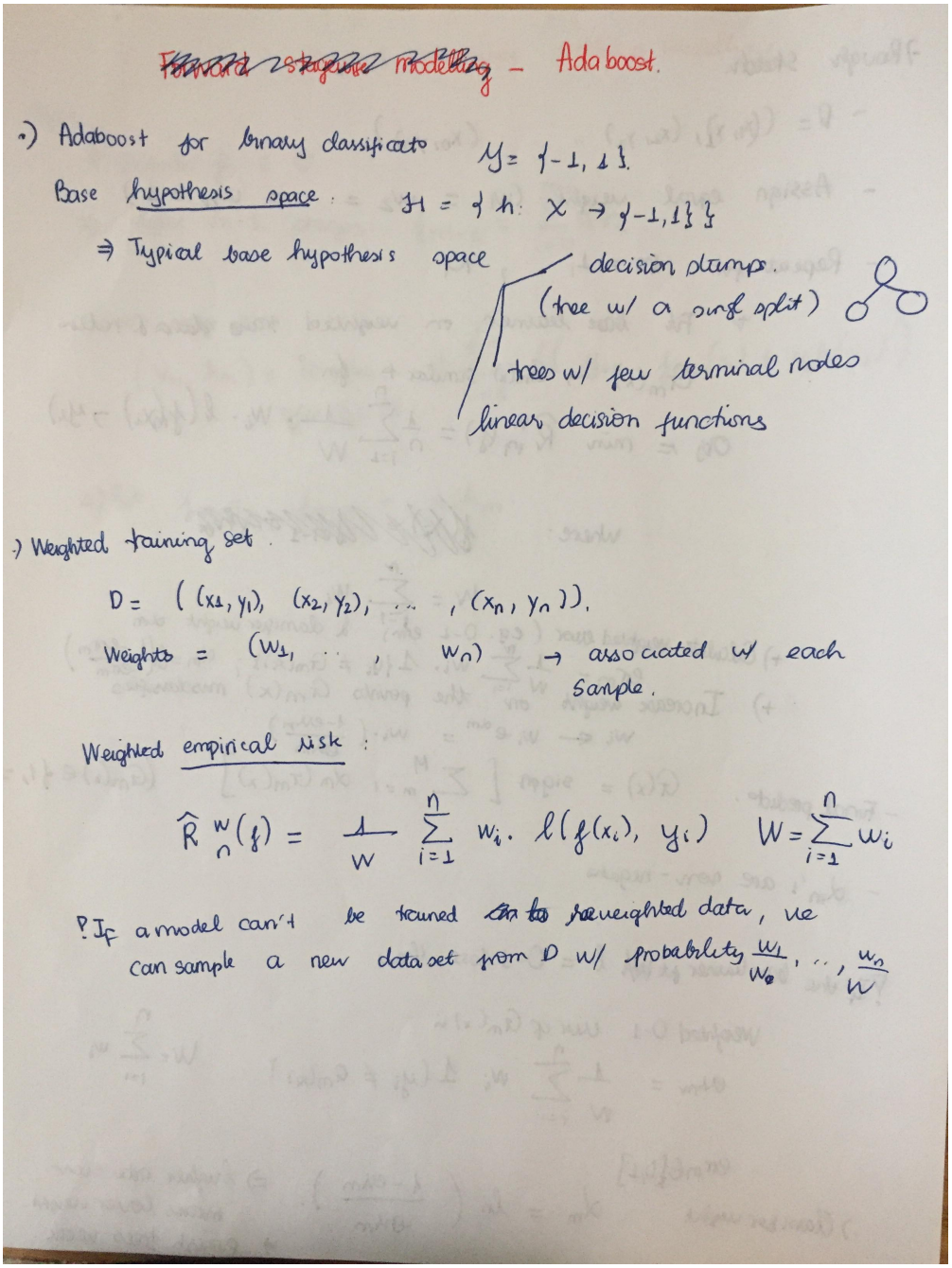
$$\hat{f}_{avg}(x) = \hat{f}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}_b(x)$$

   - Empirically, $\hat{f}_{bag}$ often performs similarly to what we would get from training on B independent samples. $\hat{f}_{bag}(x)$ has the same expectation as $\hat{f}_1(x)$ but smaller variance than $\hat{f}_1(x)$
4. Random Forest
   - Use bagged decision trees, but modify the tree-growing procedure to reduce the dependence between trees
   - When constructing each tree node, restrict choice of splitting variable to a randomly chosen subset of features of size m, which is typically $m = \sqrt{p}$, where p is the number of features
5. Bootstrap, Bagging, RF

6 Boosting

1. Sequential ensembles: models are generated sequentially
2. Try to add new models that do well where previous models lack
3. Adaboost and Gradient Boosting
4. Adaboost and Gradient Boosting - 2

6.1 Adaboost


Forward stagewise modelling – Ada boost.
•) Adaboost for binary classificato $y = \{-1, 1\}$.
Base hypothesis space: $\mathcal{H} = \{h : X \to \{-1, 1\}\}$
⇒ Typical base hypothesis space
decision stumps.
(tree w/ a single split)
trees w/ few terminal nodes
linear decision functions
•) Weighted training set.
$D = ((x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n))$.
Weights = $(w_1, \ldots, w_n)$ → associated w/ each sample.
Weighted empirical risk:
$\hat{R}^w_n(f) = \frac{1}{W} \sum_{i=1}^{n} w_i \cdot \ell(f(x_i), y_i)$ $W = \sum_{i=1}^{n} w_i$
? If a model can't be trained reweighted data, we can sample a new data set from D w/ probability $\frac{w_1}{W}, \ldots, \frac{w_n}{W}$

-) Rough Sketch.

- $D = ((x_1, y_1), (x_2, y_2) \ldots (x_n, y_n))$
- Assign equal weight $(w_1 = w_2 = \ldots = w_n = 1)$
- Repeat for $m = 1, \ldots, M$

+) Fit base learner on weighted train data & relearn $G_m(x) \rightarrow G_m(x)$ similar to f_m ?

Obj $\approx \min \hat{R}^{w}_{m}(f) = \frac{1}{n} \sum_{j=1}^{n} \frac{1}{W} \cdot w_i \cdot l(f(x_i), y_i)$

where: [illegible]

$$W = \sum_{i=1}^{n} w_i$$

+) Calculate weighted error (e.g. 0-1 err_m) & classifier weight α_m

+) Increase weight on the points $G_m(x)$ misclassifies

$err_m = \frac{1}{W} \sum_{i=1}^{n} w_i \cdot 1\{y_i \neq G_m(x_i)\}$; $\alpha_m = \ln\left(\frac{1 - err_m}{err_m}\right)$

$w_i \leftarrow w_i e^{\alpha_m} = w_i \cdot \left(\frac{1 - err_m}{err_m}\right)$

- Final predict, $G(x) = \text{sign}\left[\sum_{m=1}^{M} \alpha_m G_m(x)\right]$ $(G_m(x) \in \{1, -1\}$

- α_m's are non-negative

! If the base learner optimizes l = 0-1 loss, then

weighted 0-1 error of $G_m(x)$ is

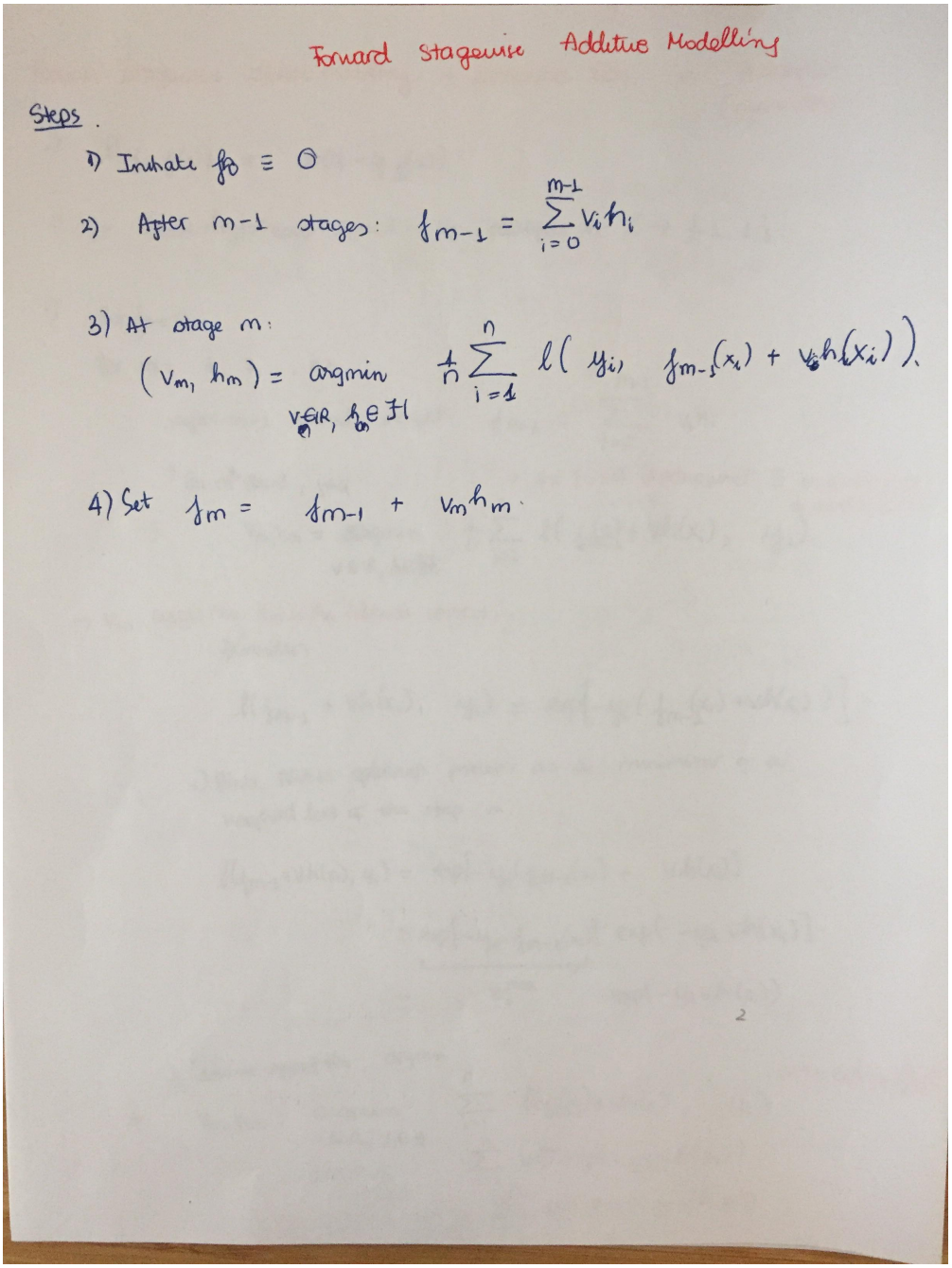
$$err_m = \frac{1}{W} \sum_{i=1}^{n} w_i \, 1(y_i \neq G_m(x_i)) \qquad W = \sum_{i=1}^{n} w_i$$

$err_m \in [0, 1]$

-) Classifier weight $\alpha_m = \ln\left(\frac{1 - err_m}{err_m}\right)$. $\Rightarrow$ higher rate error means lower weight.

$\Rightarrow$ punish trees weak

6.2 Forward Stage-wise Addictive Modeling



Forward stagewise additive modeling + exponential loss = Adaboost (more robust).

- $l(y, f(x)) = \exp(-y f(x))$.
- $\mathcal{H}$: base hypothesis space of classifier $h: X \to \{-1, 1\}$.
- Set $f_0 = 0$

For $m = 1, 2, \dots, M$

after $m-1$ round, we get $f_{m-1} = \sum_{i=1}^{m-1} v_i h_i$

On m^{th} round, find → the FSAM objective func^n ≡ minimizer of a weighted loss

$$v_m, h_m = \underset{v \in \mathbb{R},\, h \in \mathcal{H}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} l(f_{m-1}(x_i) + v h(x_i), y_i).$$

→ v_m equals to α_m in the Adaboost context!

Consider:

$$l(f_{m-1} + v h(x_i), y_i) = \exp\left[-y_i (f_{m-1}(x_i) + v h(x_i))\right]$$

+) Write FSAM optimizer problem as a minimizer of a weighted loss of the step m.

$$l(f_{m-1} + v h(x_i), y_i) = \exp[-y_i (f_{m-1}(x_i) + v h(x_i)]$$

$$= \underbrace{\exp[-y_i \cdot f_{m-1}(x_i)]}_{w_i^m} \exp(-y_i v h(x_i))$$

$$= w_i^m \exp(-y_i v h(x_i))$$

$w_i^m = \exp(-y_i \cdot f_{m-1}(x_i))$

$\frac{1}{n}$ doesn't affect the argmin

$$\Rightarrow v_m, h_m = \underset{v \in \mathbb{R},\, h \in \mathcal{H}}{\operatorname{argmin}} \sum_{i=1}^{n} l(f_{m-1}(x_i) + v h(x_i), y_i)$$

$$= \underset{v \in \mathbb{R},\, h \in \mathcal{H}}{\operatorname{argmin}} \sum_{i=1}^{n} w_i^m \exp(-y_i v h(x_i))$$

$$= \underset{v \in \mathbb{R},\, h \in \mathcal{H}}{\operatorname{argmin}} \frac{1}{\sum w_i^m} \sum_{i=1}^{n} w_i^m \exp(-y_i v h(x_i))$$

• Update:

$$f_m = f_{m-1} + v_m h_m$$

• Calculate weighted error.

$$err_m(h_m) = \frac{1}{\sum_{i=1}^{n} w_i^m} \sum_{i=1}^{n} w_i^m \mathbb{1}(y_i \neq h(x_i))$$

Note: We will prove that

$$h_m = \underset{h \in H}{\operatorname{argmin}} \sum_{i=1}^{n} w_i^m \mathbb{1}(y_i \neq h(x_i)).$$

① Show:

$$\frac{1}{\sum_{i=1}^{n} w_i^m} \cdot \sum_{i=1}^{n} w_i^m \exp(-v y_i h(x_i)) = e^{-v} + (e^{v} - e^{-v}) \cdot err_m(h)$$

~~where etc~~

Correct: $e^{-v} + (e^{v} - e^{-v})\, err_m(h)$

$$= e^{-v} + (e^{v} - e^{-v}) \cdot \frac{1}{\sum_{i=1}^{n} w_i^m} \cdot \sum_{i=1}^{n} w_i^m \mathbb{1}\{y_i \neq h(x_i)\}.$$

$$= e^{-v} + \frac{1}{\sum_{i=1}^{n} w_i^m} \cdot \sum_{i=1}^{n} w_i^m \cdot (e^{v} - e^{-v}) \mathbb{1}\{y_i \neq h(x_i)\}$$

• LHS $= \frac{1}{\sum_{i=1}^{n} w_i^m} \cdot \sum_{i=1}^{n} w_i^m \cdot \exp(-v \cdot \mathbb{1}\{y_i = h(x_i)\} + v\, \mathbb{1}\{y_i \neq h(x_i)\})$

($y_i \in \{1, -1\}$; if $y_i = h(x_i)$ then $y_i h(x_i) = 1$; if $y_i \neq h(x_i)$ then $y_i h(x_i) = -1$).

$$= \frac{1}{\sum_{i=1}^{n} w_i^m} \sum_{i=1}^{n} w_i^m \exp(-v(1 - \mathbb{1}\{y_i \neq h(x_i)\}) + v\, \mathbb{1}\{y_i \neq h(x_i)\}).$$

$$= \frac{1}{\sum_{i=1}^{n} w_i^m} \sum_{i=1}^{n} w_i^m \exp\left(-v\left(1 - 1_{\{y_i \neq h(x_i)\}}\right)\right) \exp^{v 1_{\{y_i \neq h(x_i)\}}}$$

$$= \frac{1}{\sum_{i=1}^{n} w_i^m} \sum_{i=1}^{n} w_i^m \cdot \left(e^{-v} - e^{-v} 1_{\{y_i \neq h(x_i)\}} + e^{v} 1_{\{y_i \neq h(x_i)\}} \right)$$

$$= e^{-v} \cdot \left(e^{+v} - e^{-v}\right) \cdot \frac{1}{\sum_{i=1}^{n} w_i^m} \sum_{i=1}^{n} w_i^m \cdot 1_{\{y_i \neq h(x_i)\}}$$

$$= e^{-v} \cdot (e^{+v} - e^{-v}) \cdot \quad err_m(h) .$$

) Minimize $e^{-v}(e^{+v} - e^{-v}) \cdot err_m(h) \equiv \begin{cases} \min err_m(h) & \text{if } v \geq 0 \\ \max err_m(h) & \text{if } v < 0 \end{cases}$

thus $\begin{cases} \min err_m(h) & \text{if } v \geq 0 \\ \min err_m(-h) & \text{if } v < 0 \end{cases}$

) Since $\mathcal{H}$ is symmetric $\Rightarrow$ always have $h \in \mathcal{H}$ implies $-h \in \mathcal{H}$

$\Rightarrow$ always an optimal FSAM step (v_m, h_m) w/ $v_m \geq 0$

$\Rightarrow$ $h_m = \underset{h \in \mathcal{H}}{\operatorname{argmin}}\ err_m(h)$

-) After finding h_m, find V_m.

At $h_m = \text{argmin } err_m(h)$ or $\text{argmin } e^{-V} + (e^V - e^{-V})\, err_m(h)$

$$\Rightarrow \frac{\partial J(h)}{\partial h} = 0, \quad J(h) = \frac{1}{\sum_{i=1}^{n} w_i^m} \sum_{i=1}^{n} w_i^m \exp(-V y_i h(x_i))$$

$$\frac{\partial J(h)}{\partial h} = \frac{1}{\sum_{i=1}^{n} w_i^m} \cdot \sum_{i=1}^{n} w_i^m \cdot \exp(-V y_i h(x_i)) \cdot (-V y_i) = 0$$

$$\Rightarrow V y_i$$

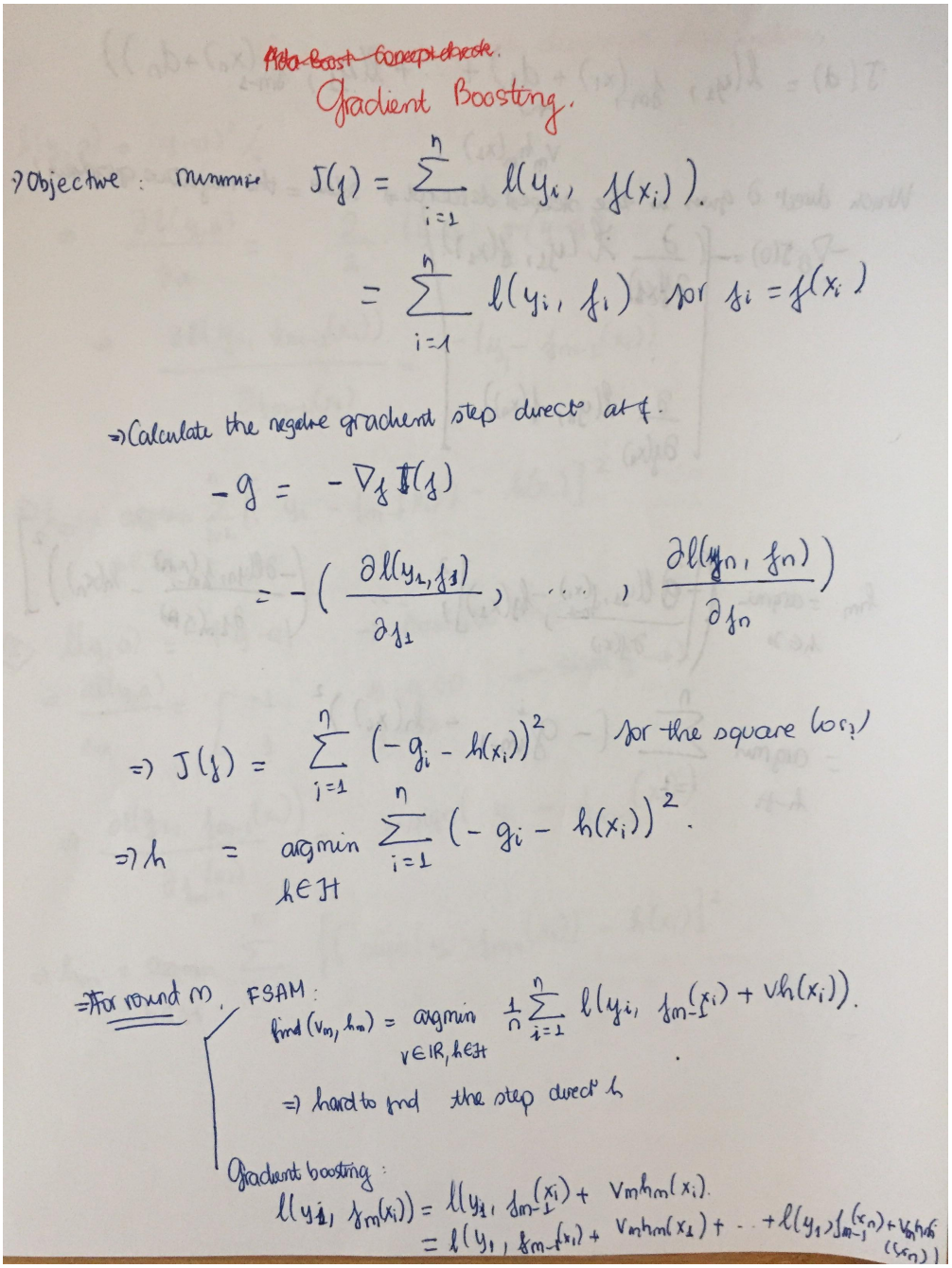
$\Rightarrow$ show $V = \frac{1}{2} \log\left(\frac{1 - err_m}{err_m}\right)$

-) Update $w_i^{m+1} = w_i^m \exp(-V_m y_i h_m(x_i))$.

$$= \begin{cases} w_i^m \exp(-V_m) \, \mathbb{1}\{y_i = h(x_i)\} \\ w_i^m \exp(V_m) \, \mathbb{1}\{y_i \neq h(x_i)\}. \end{cases}$$

$$= \begin{cases} w_i^m e^{-V_m} & \text{if } y_i = h(x_i). \\ w_i^m e^{-V_m} e^{2V_m} & \text{o.w.} \end{cases}$$

6.3 Gradient Bossting



$$J(d) = l(y_1, f_{m-1}(x_1) + d_1) + \dots + l(y_1, f_{m-1}(x_n) + d_n))$$

$\sim$

$v_m h_m(x_1)$

Which direct ∂ gives us the steepest descent $\Rightarrow$ solut = the negative gradient!

$$-\nabla_\partial J(0) = -\begin{bmatrix} \frac{\partial}{\partial f(x_1)} l(y_1, f(x_1)) \\ \vdots \\ \frac{\partial}{\partial f(x_n)} l(y_n, f(x_n)) \end{bmatrix}$$

$$h_m := \underset{h \in \mathcal{H}}{\operatorname{argmin}} \left(\left[\frac{\partial l(y_1, f_{m-1}(x_1))}{\partial f(x_1)} - h(x_1) \right]^2, \dots \left(\frac{-\partial l(y_n, f_{m-1}(x_n))}{\partial f_{m-1}(x_n)} - h(x_n) \right)^2 \right]$$

$$= \underset{h \in \mathcal{H}}{\operatorname{argmin}} \sum_{i=1}^{n} \left(-g^i_{m-1} - h(x_i) \right)^2$$

Gradient Boosting with different loss functions

① $l(y, a) = (y-a)^2/2$

$$\Rightarrow \frac{\partial l(y,a)}{\partial a} = \frac{2}{2}\cdot(y-a) = -(y-a).$$

$$\Rightarrow \frac{\partial l(y_i, f_{m-1}(x_i))}{\partial f_{m-1}(x_i)} = -(y_i - f_{m-1}(x_i)).$$

$$\Rightarrow h_m = \underset{h \in H}{\operatorname{argmin}} \sum_{i=1}^{n} [(y_i - f_{m-1}(x_i)) - h(x_i)]^2.$$

② $l(y, a) = |y-a|$ → Absolute loss

$$\Rightarrow \frac{\partial l(y,a)}{\partial a} = \begin{cases} -1 & y-a < 0 \\ 1 & y-a \geq 0. \end{cases} = -\operatorname{sign}(y-a)$$

$$\Rightarrow \frac{\partial l(y_i, f_{m-1}(x_i))}{\partial f_{m-1}(x_i)} = -\operatorname{sign}(y_i - f_{m-1}(x_i))$$

$$\Rightarrow h_m = \underset{h \in H}{\operatorname{argmin}} \sum_{i=1}^{n} [(\operatorname{sign}(y_i - f_{m-1}(x_i)) - h(x_i)]^2$$

③ $\ell(y,a) = e^{-ya}$.

$\frac{\partial \ell(y,a)}{\partial a} = e^{-ya}(-y) \Rightarrow \frac{\partial \ell(y_i, f_{m-1}(x_i))}{\partial f_{m-1}(x_i)} = e^{-y_i f_{m-1}(x_i)}(-y_i)$

$h_m := \underset{h \in H}{\operatorname{argmin}} \sum_{i=1}^{n} \left[\left(e^{-y_i f_{m-1}(x_i)} (+y_i) \right) - h(x_i) \right]^2$

④ Gradient boosting & Adaboost.

GBM: $h_m := \underset{h \in H}{\operatorname{argmin}} \sum_{i=1}^{n} \left((y_i e^{-y_i f_{m-1}(x_i)} - h(x_i)) \right)^2$.

$= \underset{h \in H}{\operatorname{argmin}} \sum_{i=1}^{n} \left(r_i - h(x_i) \right)^2$ Convert to vectors to remove $\sum$.

$= \underset{h \in H}{\operatorname{argmin}} \|\vec{r} - \vec{h}\|_2^2$

$= \underset{h \in H}{\operatorname{argmin}} \|\vec{r}\|_2^2 + \|\vec{h}\|_2^2 - 2\langle \vec{r}, \vec{h} \rangle$.

where $\vec{r} = \left[y_i e^{-y_i f_{m-1}(x_i)} \right]_{i=1}^{n}$

$\vec{h} = (h(x_i))_{i=1}^{n}$

$\because \vec{h} \in \{1,-1\} \Rightarrow \|h\|_2^2 = n$.

$\|\vec{r}\|_2^2 \geq 0$.

$\Rightarrow \underset{h \in H}{\operatorname{argmin}} \|\vec{r}\|_2^2 + \|\vec{h}\|_2^2 - 2\langle \vec{r}, \vec{h} \rangle \Leftrightarrow \underset{h \in H}{\operatorname{argmax}} 2\langle \vec{r}, \vec{h} \rangle$.

convert from inner product to sum Σ

$\because \langle \vec{r}, \vec{h} \rangle = \sum_{i=1}^{n} h(x_i) r(x_i) = \sum_{i=1}^{n} h(x_i) y_i e^{-y_i f_{m-1}(x_i)}$

$= \sum_{i=1}^{n} \left(1 - 2 \cdot 1_{\{h(x_i) \neq y_i\}} \right) \cdot e^{-y_i f_{m-1}(x_i)}$

$(h(x_i) y_i = \begin{cases} 1 & h(x_i) = y_i \\ -1 & \text{o.t.w} \end{cases}$

$= \sum_{i=1}^{n} e^{-y_i f_{m-1}(x_i)} - 2 \sum_{i=1}^{n} e^{-y_i f_{m-1}(x_i)} 1_{\{h(x_i) \neq y_i\}}$

$$\text{argmax } 2\langle \vec{r}, \vec{h} \rangle$$

$$\equiv \text{argmax } \langle \vec{r}, \vec{h} \rangle$$

$$\equiv \text{argmax } \underbrace{\sum_{i=1}^{n} e^{-y_i f_{m-1}(x_i)}}_{\text{irrelevant to } h!} - 2\sum_{i=1}^{n} e^{-y_i f_{m-1}(x_i)} \mathbb{1}\{h(x_i) \neq y_i\}$$

$$\equiv \text{argmin } \sum_{i=1}^{n} e^{-y_i f_{m-1}(x_i)} \mathbb{1}\{h(x_i) \neq y_i\}.$$

$\Rightarrow h_m$ minimizes a weighted 0-1 loss. The weights are.

$$e^{-y_i f_{m-1}(x_i)} = e^{-y_i \left(\sum_{i=1}^{m-1} v_i h_i\right)} = \prod_{i=1}^{m-1} e^{-y_i v_i h_i}$$

$$= \prod_{i=1}^{m-1} \underbrace{e^{-v_i (y_i h_i)}}.$$

$$= \prod_{i=1}^{m-1} e^{-v_i (1 - 2 \cdot \mathbb{1}_{y_i \neq h(x_i)})}.$$