Gradient Boosting for Regression and Classification

--Summary and Understanding from StatQuest

	Regression	**Classification**	
	Solution, Action	Solution, Action	Understanding
Input data $\{(x_i, y_i)\}_{i=1}^n$, and a differentiable **Loss Function** $L\big(y_i, F(x_i)\big)$	$F(x)$ is a weak model, **Loss function** uses scaled RRS $\frac{1}{2}\sum_i [y_i - F(x_i)]^2$	**Loss function** uses cross entropy $-\sum_i [p_i ln(\hat{p}_i) + (1-p_i) ln(1-\hat{p}_i)]$ **log(odds)**:$ln(odds) = -ln\left(\frac{1-p}{p}\right)$ **Probability**: $p = \frac{1}{1+e^{-log(odds)}}$	Using those forms of loss functions are for later convenience; For classification, $F(x_i)$ is log(odds), log(odds) calculated by different models can add up together, but probabilities cannot.
Step1: the 0th model is set to be a constant, with $F_0(x) = arg\min_\gamma \sum_i^n L(y_i, \gamma)$	Average over all y_i's $\frac{\sum_i^n y_i}{n}$	Log(odds) of positive samples $ln\left(\frac{\#ofpositives}{\#ofnegatives}\right)$	Minimize the loss function over all data points, using a constant estimate
Step2: for m=1 to M Where m denotes mth model	Iterate through weak models one by one, until a max number of models is reached or additional models don't improve the fit.		
Step2.1: compute pseudo residual $\gamma_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x_i)=F_{m-1}(x_i)}$	$\gamma_{im} = y_i - F_{m-1}(x_i)$	$\gamma_{im} \approx \mathrm{p_i} - \hat{\mathrm{p}}_{m-1,i}$, when γ is small. $\hat{\mathrm{p}}_{m-1,i}$ is the predicted probability that ith data point is positive, by model $F_{m-1}(x_i)$.	Calculate the pseudo residual for each data point, indicating where the previous model creates errors, preparing for building up the next model.

Step2.2: Fit a **regression tree** to γ_{im} values and denotes each leaf by R_{jm}, for all leaves in m[th] tree.	Use mean squared error (across the leaves under the same node) as the metric to choose the best feature to split on.		Gather larger pseudo residuals together in a leaf, smaller pseudo residuals together in a leaf.
Step 2.3: For all j in m[th] tree, compute $\gamma_{jm} =$ $arg\min_{\gamma} \sum_{x_i \in R_{jm}} L(y_i, F_{m-1}(x_i) + \gamma)$	Average of pseudo residuals that end up in the same leaf $\gamma_{jm} = ave(\gamma_{im} \in R_{jm})$	$\gamma_{jm} = \frac{\sum(\gamma_{im} \in R_{jm})}{\sum_{x_i \in R_{jm}}[\hat{p}_{m-1,i} \times (1 - \hat{p}_{m-1,i})]}$ $\hat{p}_{m-1,i}$ is the predicted prob by model $F_{m-1}(x_i)$.	To approximate the values in the same leaf by a constant; iterate over all leaves.
Step 2.4: Update model $F_m(x) = F_{m-1}(x) + \eta \cdot \gamma_{jm} \cdot I(x \in R_{jm})$, Where η is a scaling factor for each model, similar to step size	Add up all scaled results from a series of weak models.	Add up all scaled log(odds) from a series of weak models. Still need to convert to probability.	This is why it's called an ensemble method.
Step 3: Output $F_M(x)$	For new input x, use model $F_M(x)$ to predict.	Convert the result form the last step to probability by logistic function $p = \frac{1}{1 + e^{-log(odds)}}$	May need to tune η to find the best converged result.