Business Analytics

Classification Models

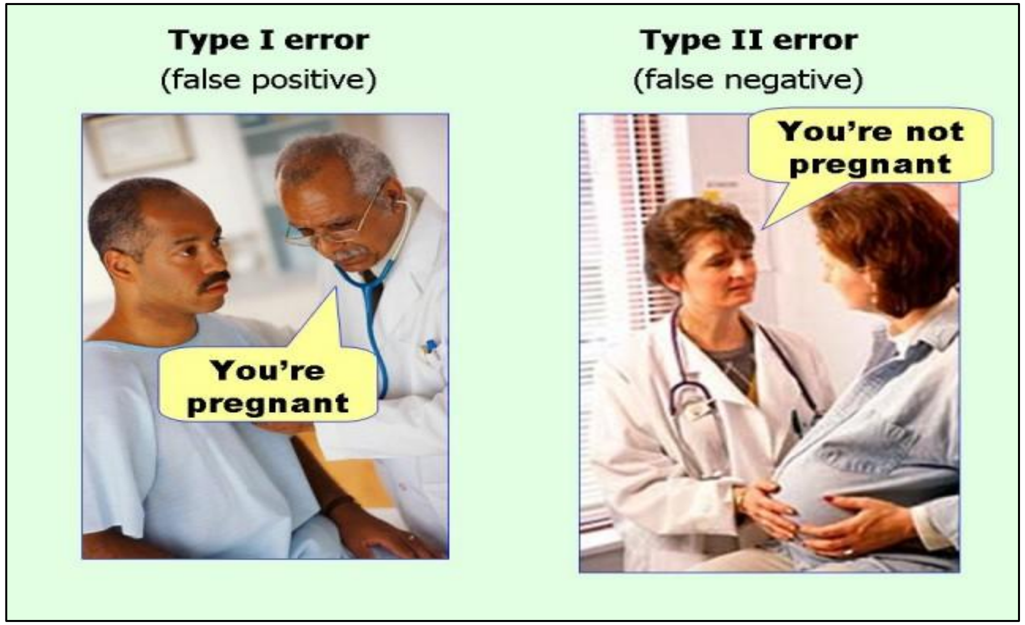
Instructor: Hrant Davtyan
Course: Business Analytics, Fall, MSSM, 2018

Content

1. Recap
    a. Classification Model Evaluation
    b. Classification Model Selection
2. Introduction to Classification and Regression Trees (CART)
    a. Intuition
    b. Advantages
    c. Model Fitting
    ~~d. Model Diagnostics~~
3. Introduction to Logistic Regression (Logit)
    a. Intuition
    b. Advantages
    c. Model Fitting
    d. Model Diagnostics (covered later)

Recap

Prediction Errors and Confusion Matrix



Confusion Matrix		Reality	
		0	1
Predicted	0	TN	FN
	1	FP	TP

Model evaluation metrics

1. Accuracy - % of correct predictions
    - Doesn't concentrate on any of the classes, is only useful for balanced data **only**
2. Recall (Sensitivity, TPR) - % of correctly predicted positives
    - Concentrates on **FN only** (TPR=1 ⇔ FN=0), useful for unbalanced data
3. Specificity (TNR) - % of correctly predicted negatives
    - Concentrates on **FP only** (TNR=1 ⇔ FP=0), useful for unbalanced data
4. **How to concentrate on both?**
    - **Use Harmonic Mean of TPR and TNR (F score)**
    - **Do your calculations for different possible thresholds (ROC AUC score - Area under Receiver Operating Characteristics Curve)**

Model Selection

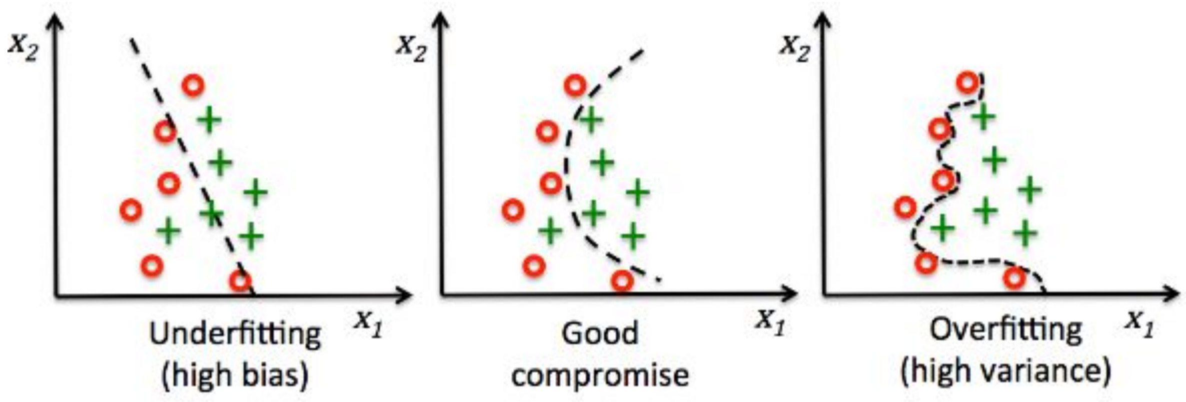
Bias-Variance tradeoff

When making a prediction, we may have 2 objectives:

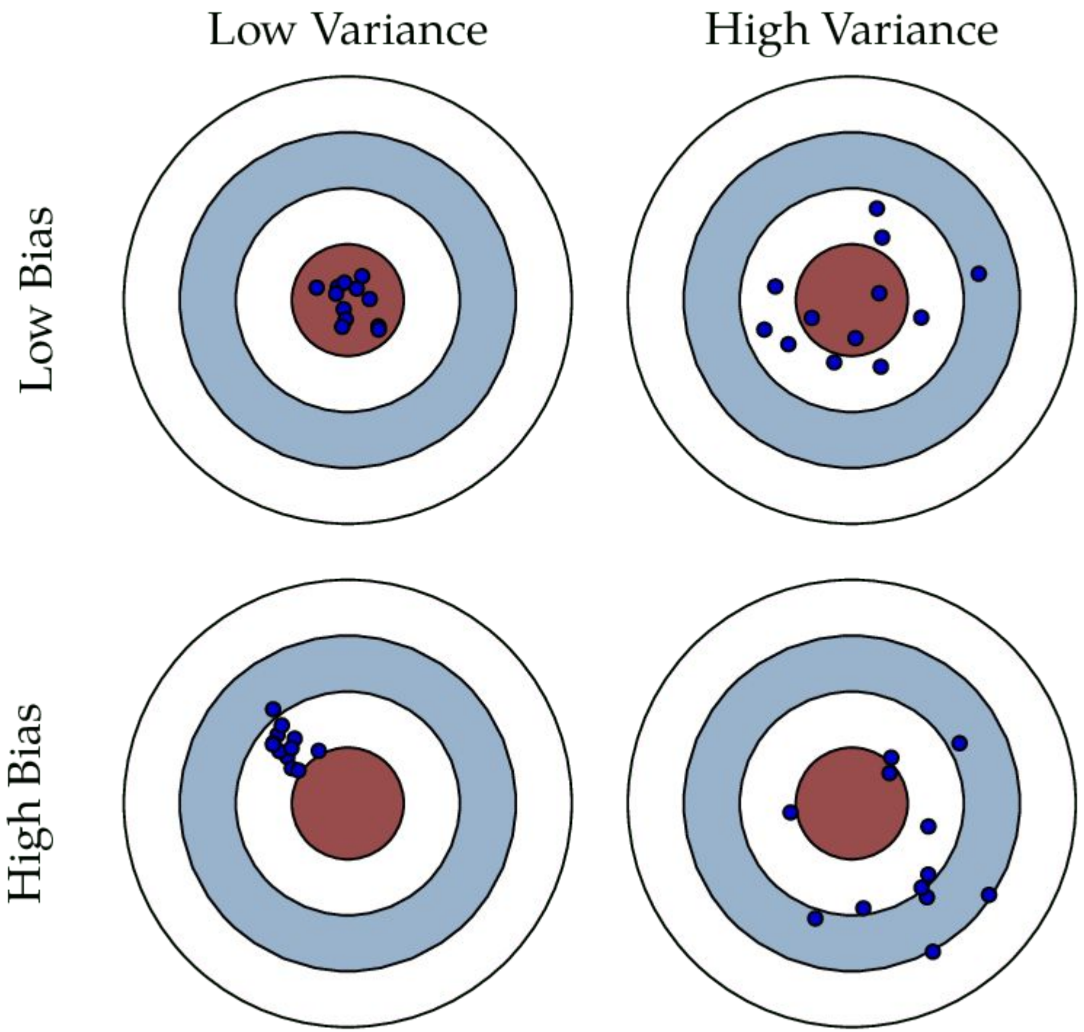
- to get as accurate model as possible (low bias),
- to get as consistent/generalizable model as possible (low variance).

Problem!

- this is not always possible



Bias-Variance tradeoff (cont'd)



Bias-Variance tradeoff (cont’d)

1. The high variance problem is known as overfitting.
2. Overfitting decreases the model generalizability (i.e. model is not useful for external data).
3. To learn about overfitting, the (probably) best solution is train-test split.
4. Develop the model on train, and calculate its accuracy measures on train and test.
5. If they are close, then no overfitting.
6. If 2 measures are different, then you probably have overfitting.

How to fight overfitting?

Develop better models by tuning hyperparameters.

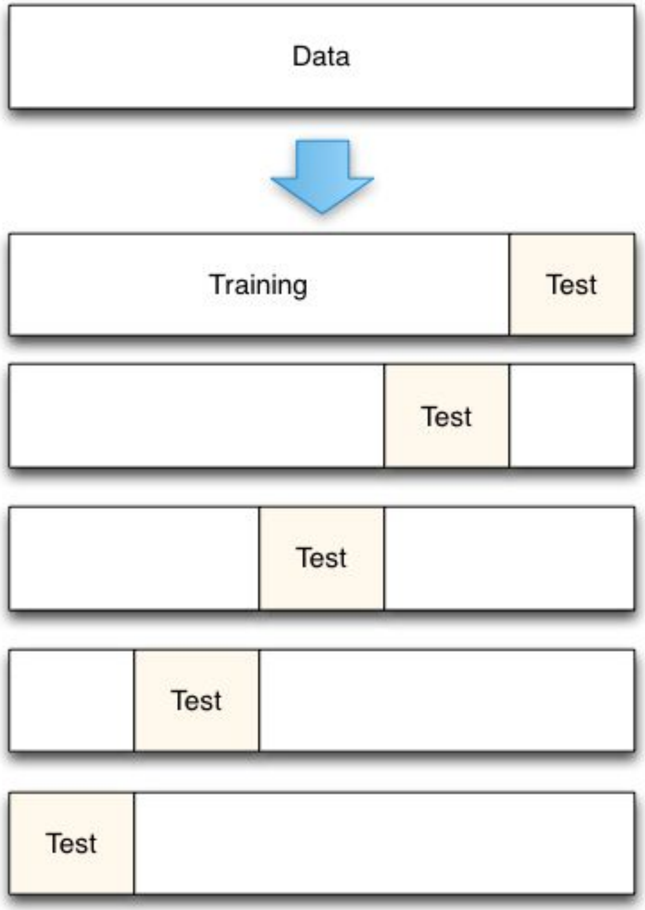
- Either manually put different values for hyperparameters, and choose the best ones.
- Or do the same thing automatically using an algorithm called GridSearch.
- For example: maximum depth of the decision tree to grow.

Problem with Hyperparameter tuning?

- To tune hyperparameters and avoid overfitting the train set, one should calculate accuracy on the test.
- Yet, it is possible, that one starts to overfit train set instead.
- Solution? Evaluate your model on different test sets, not only one.

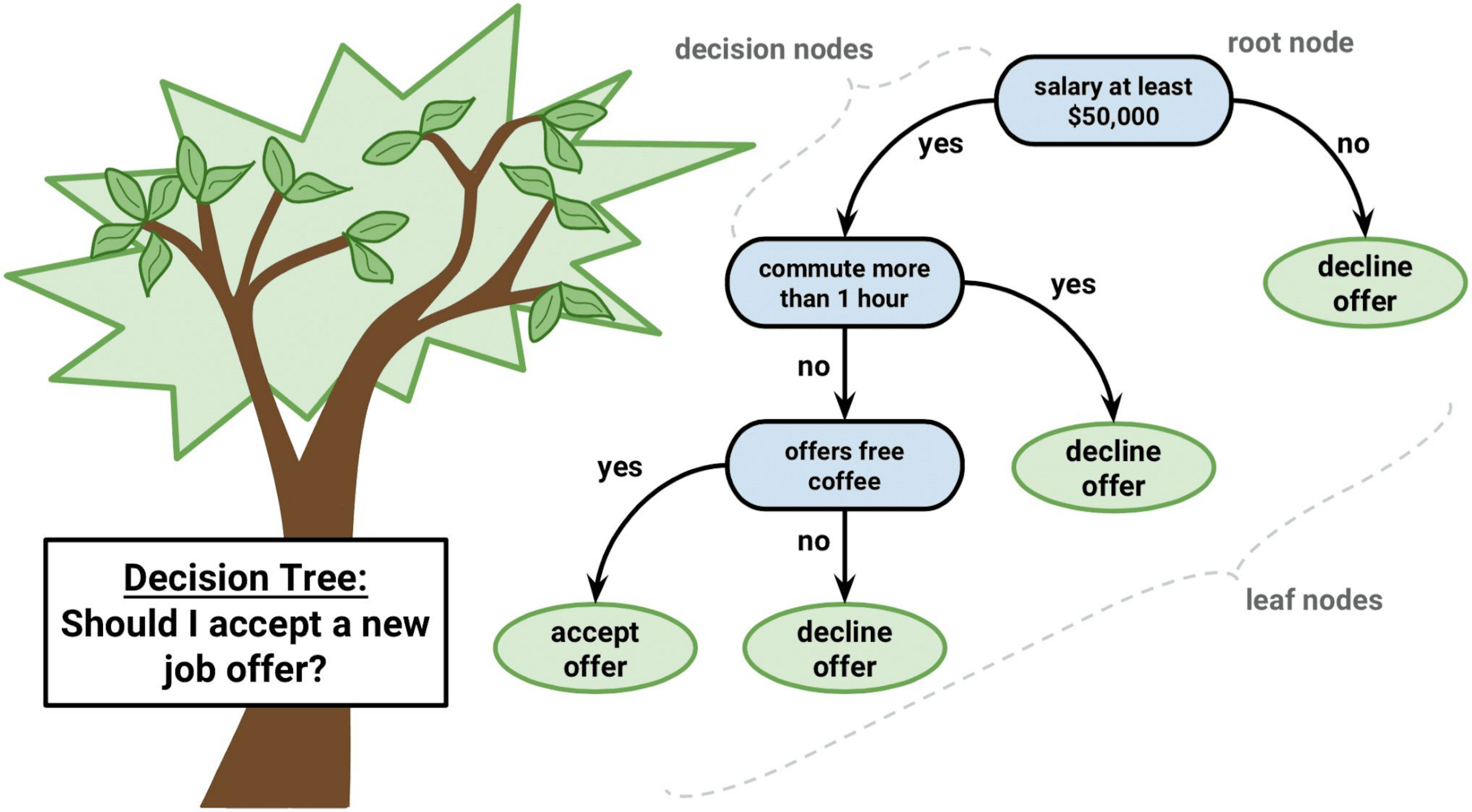
Cross-Validation

1. Train-test split helps to fight overfitting on the Training data.
2. What if we overfit test data?
3. Solution: test on different Test sets (known as Cross-Validation).
4. Example: 5 fold (5 component) cross validation (on the right).



Classification and Regression Tree

CART: intuition



How CART grows?

1. The objective is to make as accurate prediction as possible.
2. For that tree splits in a way to have as pure leafs (nodes) in the end as possible.
3. In other words, it aims to reduce variance after each split.
4. 2 measures of variance are usually aimed to minimize by decision trees:
   - Gini = *2*p*(1-p)*
   - Entropy = *-p*log(p)*
5. Recommendation: almost always use Gini as a Loss function, because:
   - Theory does not suggest any dominance,
   - Published papers and projects report always identical results,
   - Gini is faster, as unlike entropy, there is no logarithm calculated behind.

Gini: derivation for binary case

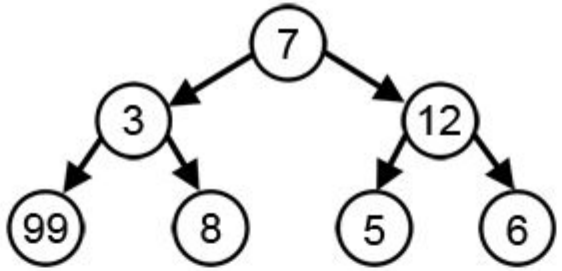
$$\textit{General case (multiple classes): Gini index} = 1 - \sum_{i=1}^{n} {p_i}^2$$

$$\textit{Binary case (2 classes: } n = 2\text{): Gini index} = 1 - \sum_{i=1}^{2} {p_i}^2$$

$$\begin{aligned}
&= 1 - {p_1}^2 - {p_2}^2 \overset{p_1+p_2=1}{\Longleftrightarrow} 1 - {p_1}^2 - (1 - p_1)^2 \\
&\overset{p\equiv p_1}{\Longleftrightarrow} (1 - p^2) - (1 - p)^2 \\
&= (1 - p)(1 + p) - (1 - p)(1 - p) \\
&= (1 - p)(1 + p - 1 + p) \ = (1 - p)2p \\
&= 2p(1 - p) \\
&= \textit{Variance of Binomial distribution } (n = 2)
\end{aligned}$$

Greedy Search algorithm

1. As mentioned, the tree aims to split using a variable (and a value) that minimizes the Gini value at each stage separately.
2. This algorithm is called Greedy Search algorithm, as the search for the optimal path is done by optimizing the split at every single stage.
3. Sometimes, this might not yield the global optimum. Like in chess, sometimes to win one has to sacrifice a figure during the game. But CART never sacrifices, thus, the path it takes may be just locally (and not globally) optimal.
4. The right hand side animation illustrates a similar situation when greedy search algorithm is unable to find the path achieving highest sum (7+3+99).



Logistic Regression

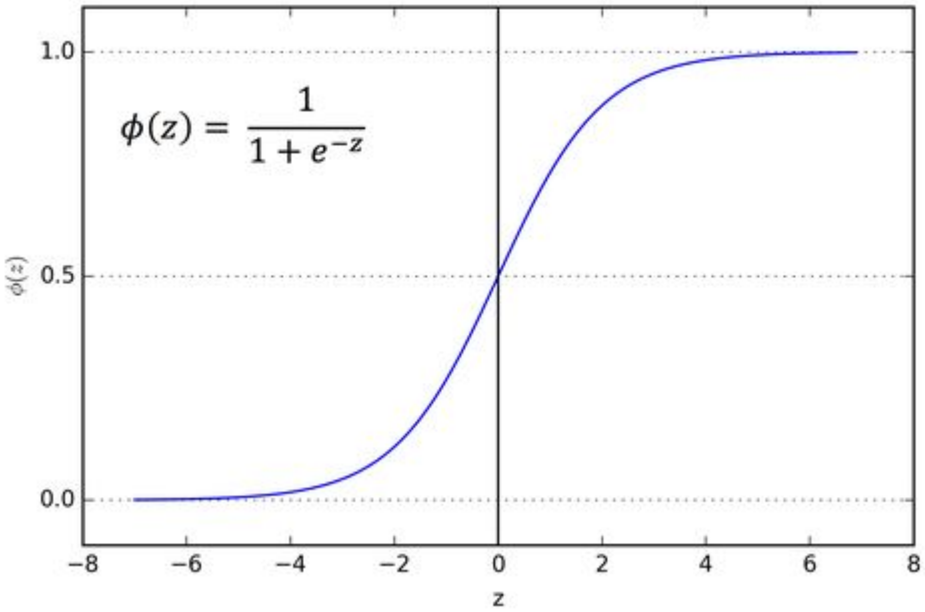
Intuition

1. Classical regression models are not very useful for predicting binary target.
2. As an alternative to Linear regression (for predicting continuous dependent variable), Logistic regression is suggested (for predicting discrete outcomes).
3. It is called Logistic regression as:
    - regression is run behind to predict an intermediate value Z (discussed later),
    - logit (also known as Sigmoid) function is used to transform this Z into probabilities,
    - given the probability threshold (usually 0.5), predicted probabilities as transformed to 1 (if >0.5) and 0 (otherwise).
4. As a result, a linear threshold is drawn between two (or more) classes of observations.
5. Key takeaway: logit is a **linear** model and it provides predicted **probabilities**.

Sigmoid function

Sigmoid function is used to transformed the Z continuous variable resulting from regression into probabilities as:

- Sigmoid assures that whatever value Z attains, it will be transformed into something from range (0,1).
- It is symmetric around 0.5 (outcome classes have equal chances of being predicted).
- It provides the opportunity to interpret Z.



Z: interpretation

1. Due to Sigmoid, Logistic Regression is fully interpretable model.
2. As shown on the right hand side derivation, the intermediate continuous variable Z achieved through regression is nothing else than the natural logarithm of odd ratio.
3. Maximizing log(odd ratio) maximizes odd ratio, which in its turn maximizes predicted probabilities, i.e. increases confidence in predictions.

- $Y = \frac{1}{1+e^{(-z)}}$
- $Y + Y \times e^{(-z)} = 1$
- $Y \times e^{(-z)} = 1 - Y$
- $e^{(-z)} = \frac{1-Y}{Y}$
- $e^{(z)} = \frac{Y}{1-Y}$
- $z = log(\frac{Y}{1-Y})$

Loss function

1. On the one hand, we aim to maximize the probability of predicting one, if an observation has a true value of 1.
2. One the other hand, we aim to maximize the probability of of predicting 0, if an observation is truly 0. Thus, we basically aim to minimize the probability of predicting 1 in this case.
3. Conclusion: we need a Loss function which achieves both of the abovementioned contradicting objectives.

- Probability of $Y_t = 1: Y_p = G(z) \rightarrow max$
- Probability of $Y_t = 0: Y_p = 1 - G(z) \rightarrow max$
- $G(z)^{Y_t} \times [1 - G(z)]^{(1-Y_t)} \rightarrow max$

Cross-entropy: derivation

- $G(z)^{Y_t} \times [1 - G(z)]^{(1-Y_t)} \rightarrow max$
- $log(G(z)^{Y_t} \times [1 - G(z)]^{(1-Y_t)}) \rightarrow max$
- $log(G(z)^{Y_t}) + log([1 - G(z)]^{(1-Y_t)}) \rightarrow max$
- $Y_t \times log(G(z)) + (1 - Y_t) \times log(1 - G(z)) \rightarrow max$

- $-Y_t \times log(G(z)) - (1 - Y_t) \times log(1 - G(z)) \rightarrow min$

Gradient descent

1. Last question: how do we minimize the cross-entropy loss function?
2. Solution steps:
   a. Randomly choose values for beta coefficients,
   b. Make a prediction and calculate Loss,
   c. Calculate the derivative of the Loss given the randomly initialized betas,
   d. Update betas opposite to gradient direction with an amount proportional to the partial derivative.
   e. Repeat steps b-d steps unless Loss is no longer minimized.
3. The only problem:
   a. If Loss is not quadratic, then one may end up in a locally optimal point, not globally (i.e. local and global optimums are the same for a quadratic function).

Regularization

1. What do we do if CART overfits? We tune depth and/or other hyperparameters.
2. Hyperparameters for Logit? Lambda - regularization rate.
3. Logit overfits if beta coefficients get to large. Regularization is the process of minimizing the absolute value or the square of betas directly in the loss function.
   a. Absolute value - L1 regularization
   b. Squares - L2 regularization
4. Lambda parameter controls the strength of regularization: higher lamda indicates stronger regularization and should be used whenever model overfits.
5. Note: in sklearn the parameter to tune is called C = inverse of lambda, i.e. lower values of C indicates higher values of lambda and stronger regularization.

Comparison

1. Logistic Regression
    - Is simple and fast
    - Is linear
    - Provides probabilities
    - **Is fully interpretable**
    - Is parametric
    - Has a few hyperparameters to tune
2. Decision Tree Classification
    - Is simple and fast
    - **Is nonlinear**
    - Does not provide probabilities
    - Is interpretable but not fully (e.g. does not provide p-values)
    - Is nonparametric
    - Has many hyperparameters to tune

Thank you