A Closer Look at Tree Boosting Methods

Felipe Santos and John Yap

Project Questions

- What is boosting?
- Is boosting used exclusively for trees?
- What are AdaBoost, Gradient Boosting, and XGBoost?
- How are they similar or different?
- Are they used for different purposes?
- Is one better than others?
- How do we use them?

➡ Research-oriented

➡ Focus on binary classification

➡ More details in upcoming blog!

Outline

Part I: Boosting (John)

- What is boosting? (AdaBoost)
- Simulated data example
- AdaBoost, Gradient Boosting, XGBoost
- Model Parameters (Python)

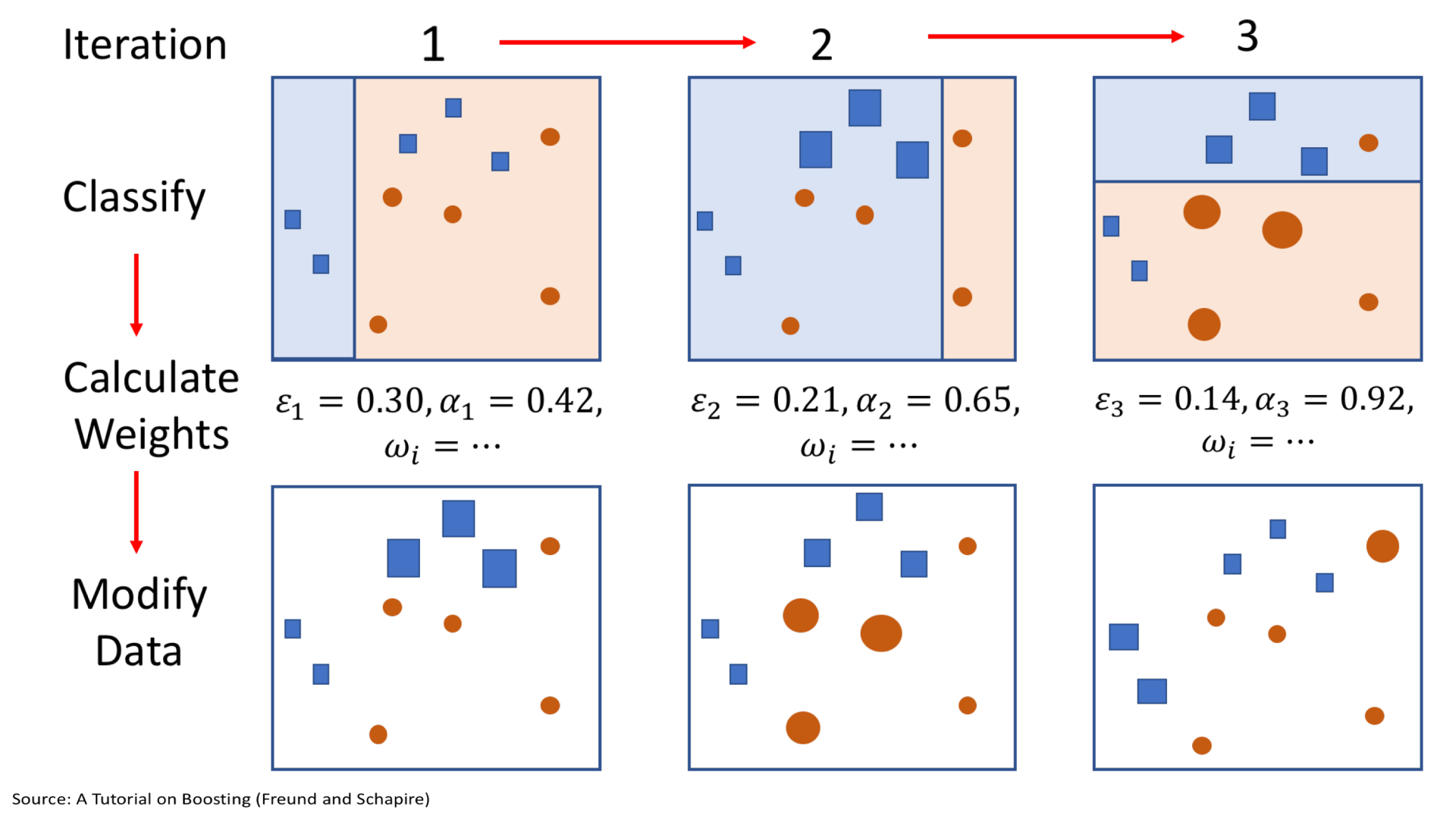
Part II: Toxic Comments Classification (Felipe)

- Kaggle Dataset
- Multi-label Classification
- Analyses and Results

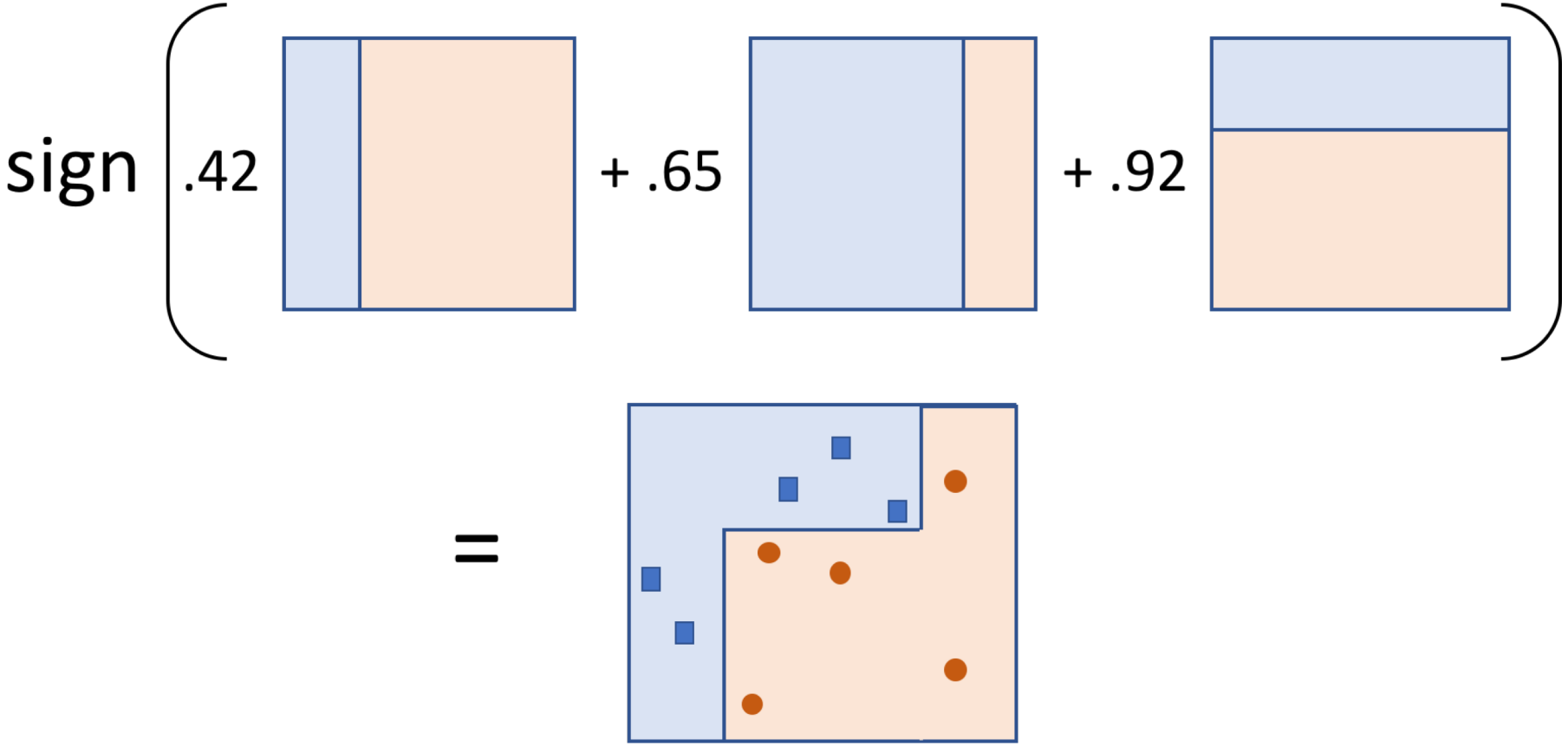
Part I: Boosting

What is boosting?

- AdaBoost (for binary classification)
- Weak classifier
  - slightly better than random guessing
  - e.g. stump
- Basic idea:
  - produce a sequence of weak classifiers
  - data is modified at each iteration where misclassified observations are upweighted while correctly classified observations are down-weighted
  - combine (weighted) predictions from the weak classifiers to obtain a more accurate classifier



Source: A Tutorial on Boosting (Freund and Schapire)



Source: A Tutorial on Boosting (Freund and Schapire)

Simulated Data

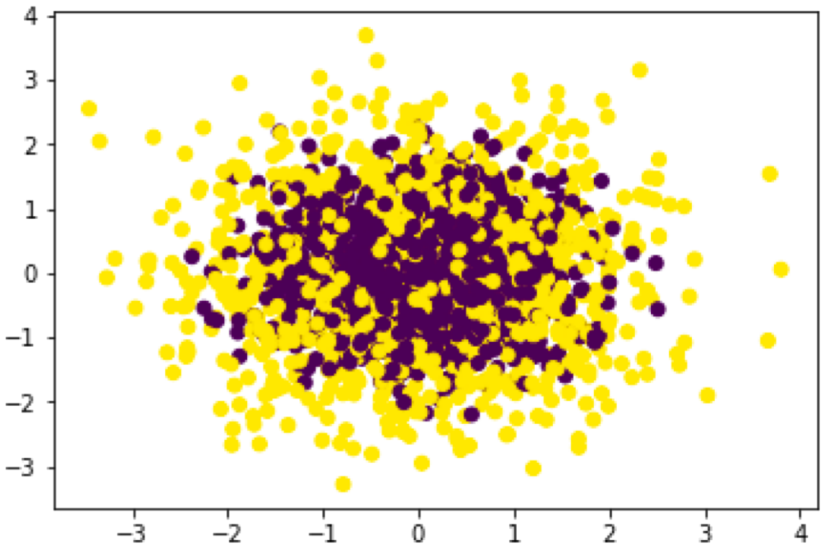
- 10 features, each standard independent Gaussian
- 1 if sum of squares of features > median of chi-square(df=10), -1 ow
- Classifiers:

    - stump

    - decision tree (9 nodes)

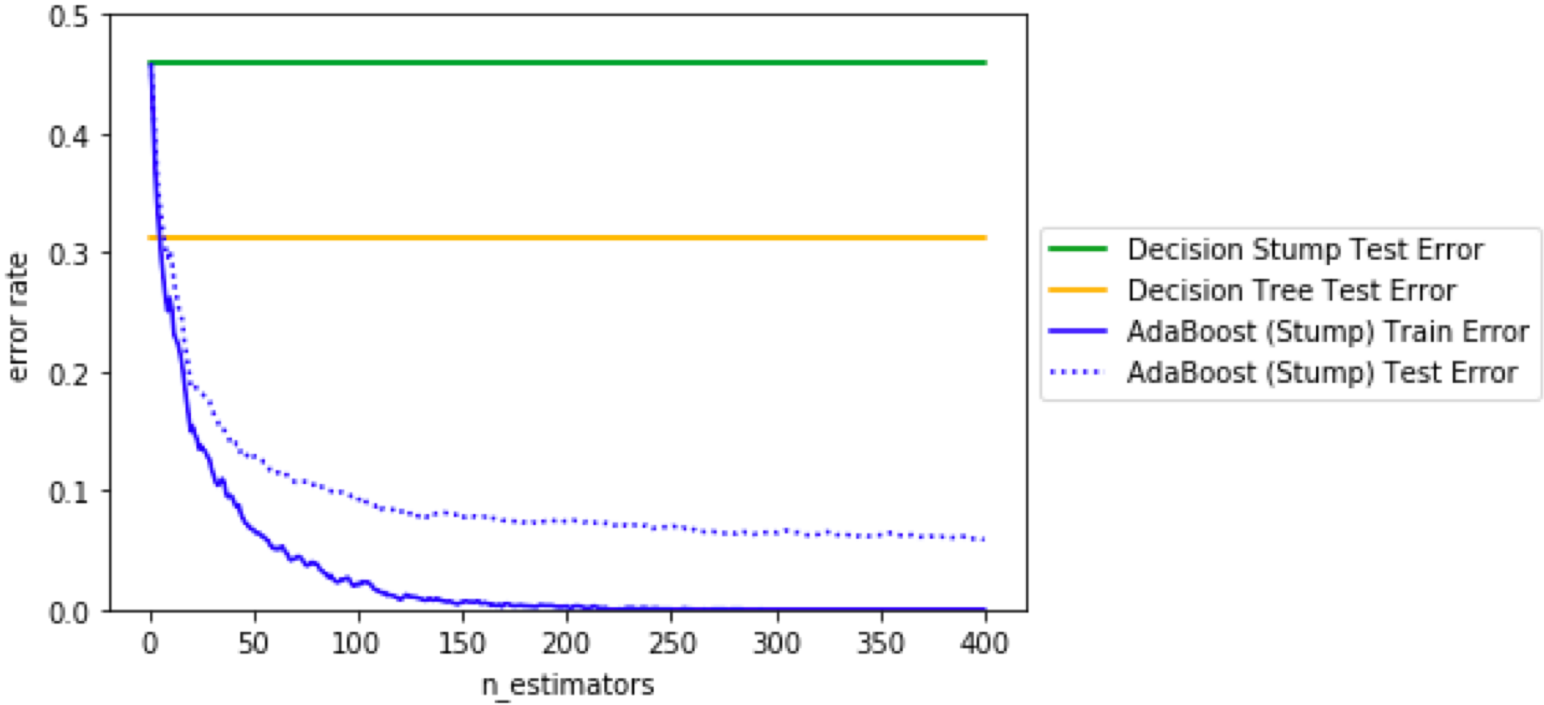
    - boosted stump (400 iter)

- Plot classification error vs iter



References: The Elements of Statistical Learning and Scikit-Learn

Simulated Data...continued



References: The Elements of Statistical Learning and Scikit-Learn

Boosting Methods

- AdaBoost
- Gradient Boosting

  - create a sum of trees

  - estimate gradient with tree

  - for squared-error loss in regression, gradient is residual!

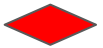
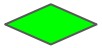
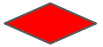
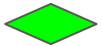
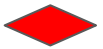
  - for classification, use deviance or exponential loss

- XGBoost

  - same idea behind gradient boosting but uses second order approximations

  - includes more capabilities

Model Parameters

Parameters	AdaBoost	Gradient Boosting	XGBoost
max_depth			
learning_rate			
n_estimators			
subsample			
max_features/colsample_bylevel			
colsample_bytree			
reg_alpha			
reg_lambda			

Part II:
Toxic Comments Classification

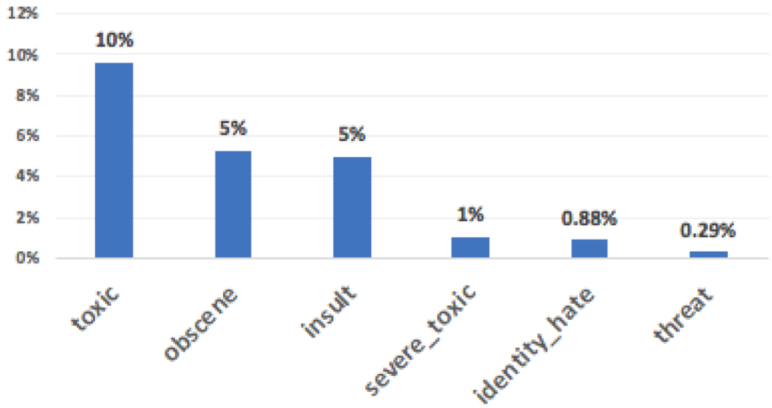
Model Accuracies

Dataset	n_train; #feat	AdaBoost	Gradient Boosting	XGBoost
Simulation	2000; 10	0.948	0.947	0.940
Spam Email*	2300; 57	0.950	0.948	0.951
Evergreen	2389; 22	0.684	0.703	0.720

Toxic Comment Classification

- **Identify and classify toxic online comments**: from Wikipedia's talk page

Unbalanced Distribution:
159,571 comments



Multi-label problem: each comment can be categorized into more than one label.

toxic	insult	obscene	severe toxic	threat	identity hate	Count
0	0	0	0	0	0	143.346
1	0	0	0	0	0	5.666
1	1	1	0	0	0	3.800
1	0	1	0	0	0	1.758
1	1	0	0	0	0	1.215
1	1	1	1	0	0	989
1	1	1	0	0	1	618
0	0	1	0	0	0	317
0	1	0	0	0	0	301
1	1	1	1	0	1	265

Toxic Comment Classification - **Steps**

Comments

Multilabel *(some options)*:

- "One vs. All" (our method of choice)
- Chain Classifier
- Power Set

NLP:

- Stemmer: *Closed, Closing, Closing = Close*
- TF-IDF: term frequency–inverse document frequency
  - Bigram = ***1***, 2, 3

Boosting Methods (test_size= 33%):

- AdaBoost
- Gradient Boosting
- XGBoost

Results:

- Accuracy
- Confusion Matrix
- Time required for calculation

Toxic Comment Classification - **Results**

Exploring Label = Toxic Comments

accuracy_score

n_estimators	AddaBoost max_depth 3	AddaBoost max_depth 5	AddaBoost max_depth 7	GradientBoosting max_depth 3	GradientBoosting max_depth 5	GradientBoosting max_depth 7	XGBClassifier max_depth 3	XGBClassifier max_depth 5	XGBClassifier max_depth 7
500	94,86%	95,09%	95,25%	95,44%	95,33%	95,19%	95,70%	95,77%	95,84%
800	95,03%	95,25%	95,41%	95,33%	95,20%	95,18%	95,79%	95,81%	95,87%
1000	95,10%	95,42%	95,55%	95,38%	95,26%	95,10%	95,84%	95,82%	95,82%

Method	Best Score	
XGBClassifier	95,87%	n_estimators = 1000, max_deph = 7
AddaBoost	95,55%	n_estimators = 1000, max_deph = 5
GradientBoosting	95,44%	n_estimators = 500, max_deph = 3

time required for calculation (average)

n_estimators	AdaBoost	GradientBoosting	XGBClassifier
500	1h 20min	1h 35min	31min
800	2h 8min	2h 34min	49min
1000	3h	3h 10min	1h

Further Work

- Scikit-learn pipeline issue:
  - TfidfVectorizer does not work outside pipeline
  - TfidfVectorizer in pipeline seems inefficient
- Find datasets where XGBoost will perform much better than Gradient Boosting
- Explore NLP techniques that might help improve predictions (e.g. lemmatization)
- Explore techniques other than OneVsAll for multi-label classification (e.g. chain classifier and power set)

Obrigado!