



Boosting

Jeffrey Leek
Johns Hopkins Bloomberg School of Public Health

Basic idea

1. Take lots of (possibly) weak predictors
2. Weight them and add them up
3. Get a stronger predictor

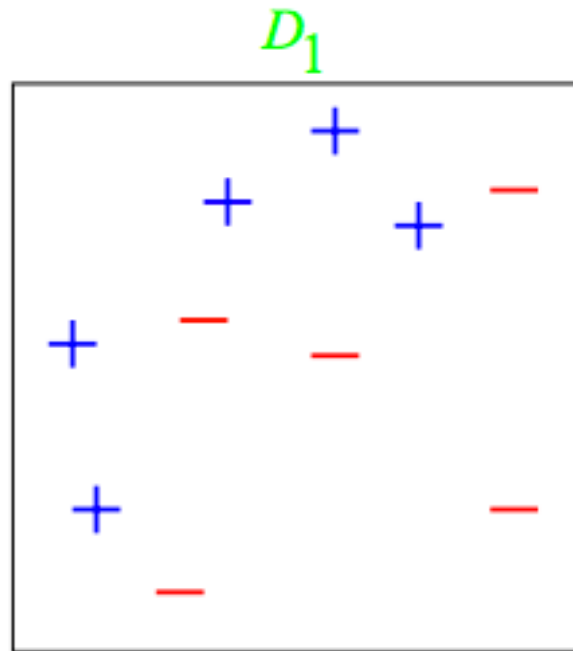
Basic idea behind boosting

1. Start with a set of classifiers h_1, \dots, h_k
 - Examples: All possible trees, all possible regression models, all possible cutoffs.
2. Create a classifier that combines classification functions: $f(x) = \text{sgn}\left(\sum_{t=1}^T \alpha_t h_t(x)\right)$.
 - Goal is to minimize error (on training set)
 - Iterative, select one h at each step
 - Calculate weights based on errors
 - Upweight missed classifications and select next h

[Adaboost on Wikipedia](#)

<http://webee.technion.ac.il/people/rmeir/BoostingTutorial.pdf>

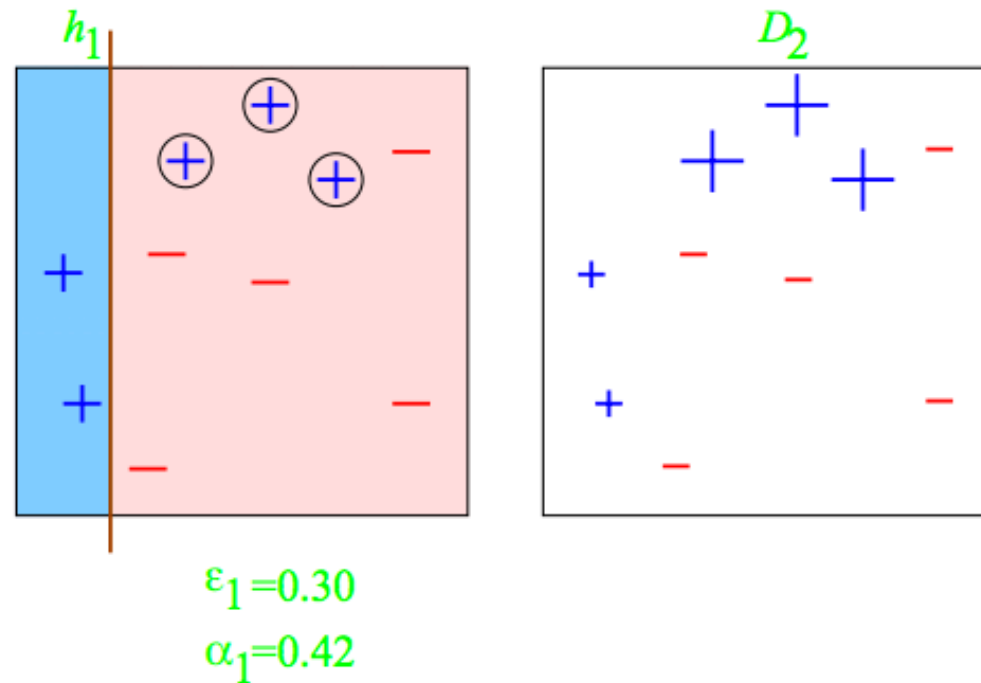
Simple example



<http://webee.technion.ac.il/people/rmeir/BoostingTutorial.pdf>

Round 1: adaboost

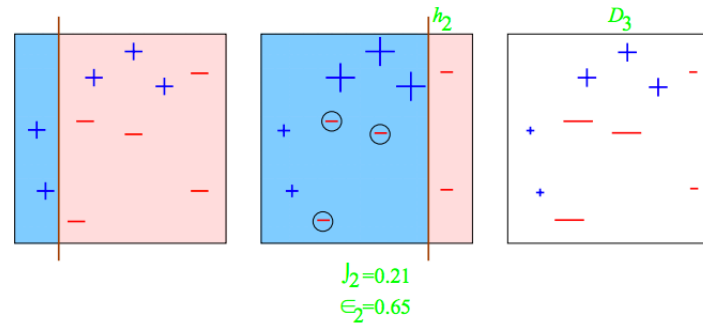
Round 1



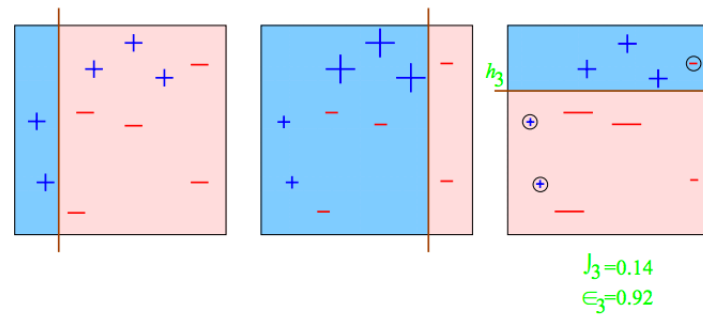
<http://webee.technion.ac.il/people/rmeir/BoostingTutorial.pdf>

Round 2 & 3

Round 2



Round 3



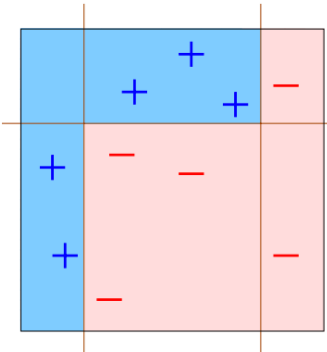
<http://webee.technion.ac.il/people/rmeir/BoostingTutorial.pdf>

Completed classifier

Final Hypothesis

$$H_{\text{final}} = \text{sign} \left(0.42 \begin{array}{|c|} \hline \text{blue} \\ \hline \text{red} \end{array} + 0.65 \begin{array}{|c|} \hline \text{blue} \\ \hline \text{red} \end{array} + 0.92 \begin{array}{|c|} \hline \text{blue} \\ \hline \text{red} \end{array} \right)$$

=



<http://webee.technion.ac.il/people/rmeir/BoostingTutorial.pdf>

Boosting in R

- Boosting can be used with any subset of classifiers
- One large subclass is [gradient boosting](#)
- R has multiple boosting libraries. Differences include the choice of basic classification functions and combination rules.
 - [gbm](#) - boosting with trees.
 - [mboost](#) - model based boosting
 - [ada](#) - statistical boosting based on [additive logistic regression](#)
 - [gamBoost](#) for boosting generalized additive models
- Most of these are available in the caret package

Wage example

```
library(ISLR); data(Wage); library(ggplot2); library(caret);  
Wage <- subset(Wage, select=-c(logwage))  
inTrain <- createDataPartition(y=Wage$wage,  
                                p=0.7, list=FALSE)  
training <- Wage[inTrain,]; testing <- Wage[-inTrain,]
```

Fit the model

```
modFit <- train(wage ~ ., method="gbm", data=training, verbose=FALSE)
print(modFit)
```

2102 samples

10 predictors

No pre-processing

Resampling: Bootstrap (25 reps)

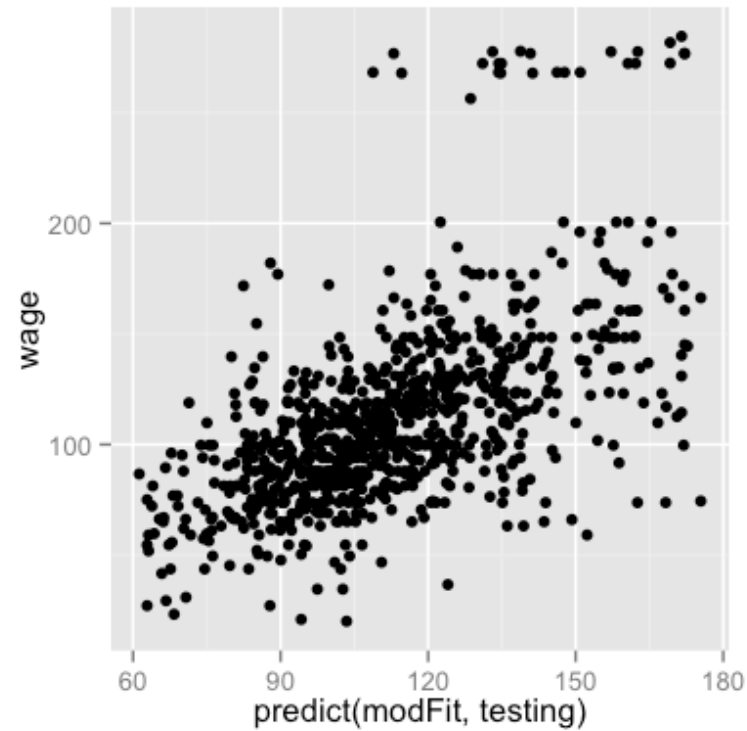
Summary of sample sizes: 2102, 2102, 2102, 2102, 2102, 2102, ...

Resampling results across tuning parameters:

interaction.depth	n.trees	RMSE	Rsquared	RMSE SD	Rsquared SD
1	50	30	0.3	1	0.02
1	100	30	0.3	1	0.02
1	200	30	0.3	1	0.02
2	50	30	0.3	1	0.02
2	100	30	0.3	1	0.02

Plot the results

```
qplot(predict(modFit,testing),wage,data=testing)
```



Notes and further reading

- A couple of nice tutorials for boosting
 - Freund and Shapire - <http://www.cc.gatech.edu/~thad/6601-gradAI-fall2013/boosting.pdf>
 - Ron Meir- <http://webee.technion.ac.il/people/rmeir/BoostingTutorial.pdf>
- Boosting, random forests, and model ensembling are the most common tools that win Kaggle and other prediction contests.
 - http://www.netflixprize.com/assets/GrandPrize2009_BPC_BigChaos.pdf
 - <https://kaggle2.blob.core.windows.net/wiki-files/327/09ccf652-8c1c-4a3d-b979-ce2369c985e4/Willem%20Mestrom%20-%20Milestone%201%20Description%20V2%202.pdf>