Combining predictors

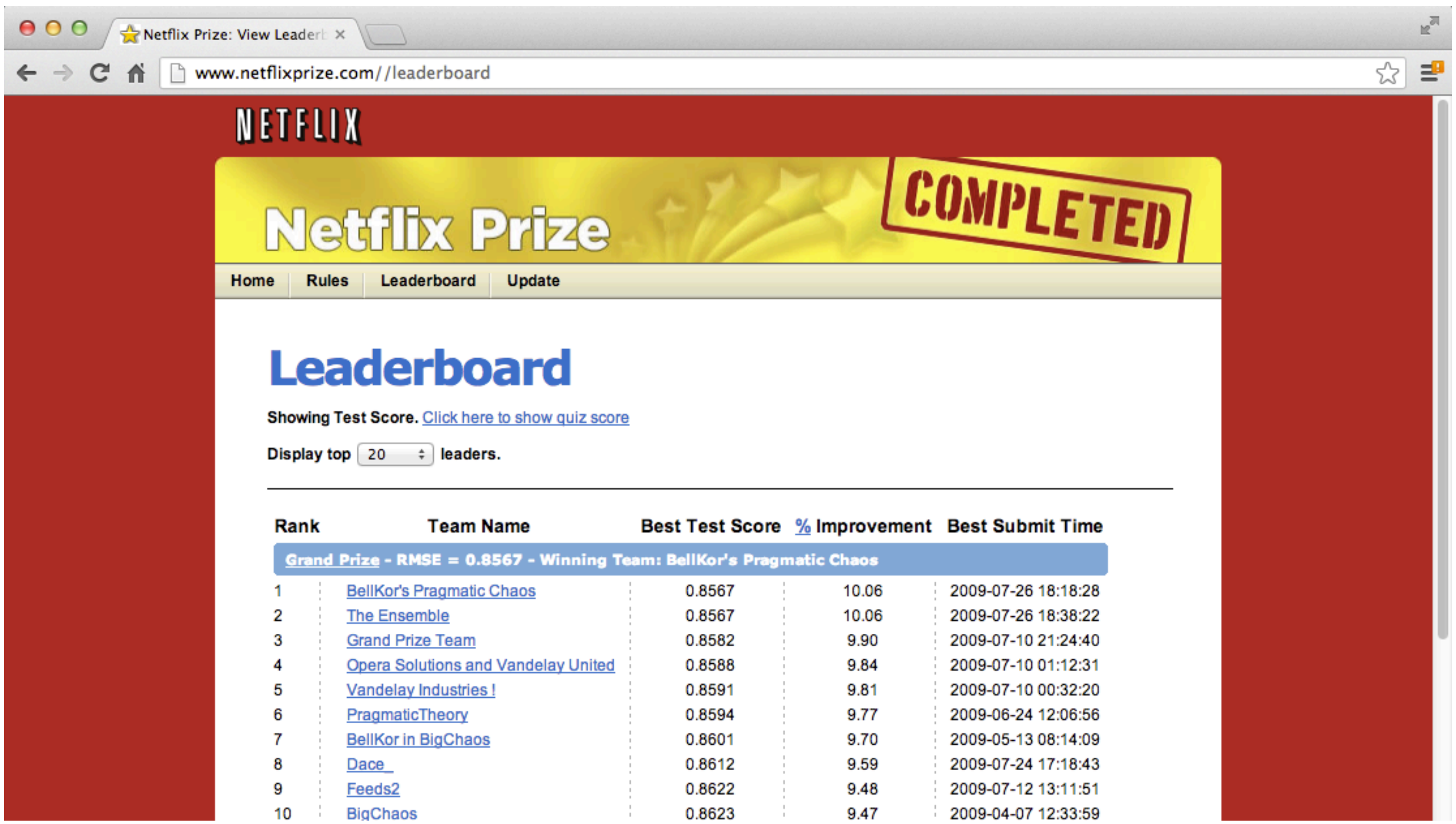
Jeffrey Leek, Assistant Professor of Biostatistics
Johns Hopkins Bloomberg School of Public Health

Key ideas

- You can combine classifiers by averaging/voting
- Combining classifiers improves accuracy
- Combining classifiers reduces interpretability

Netflix prize

BellKor = Combination of 107 predictors



Netflix Prize: View Leaderb

www.netflixprize.com//leaderboard

NETFLIX

Netflix Prize

COMPLETED

Home Rules Leaderboard Update

Leaderboard

Showing Test Score. Click here to show quiz score

Display top 20 **leaders.**

Rank	Team Name	Best Test Score	% Improvement	Best Submit Time
Grand Prize - RMSE = 0.8567 - Winning Team: BellKor's Pragmatic Chaos				
1	BellKor's Pragmatic Chaos	0.8567	10.06	2009-07-26 18:18:28
2	The Ensemble	0.8567	10.06	2009-07-26 18:38:22
3	Grand Prize Team	0.8582	9.90	2009-07-10 21:24:40
4	Opera Solutions and Vandelay United	0.8588	9.84	2009-07-10 01:12:31
5	Vandelay Industries !	0.8591	9.81	2009-07-10 00:32:20
6	PragmaticTheory	0.8594	9.77	2009-06-24 12:06:56
7	BellKor in BigChaos	0.8601	9.70	2009-05-13 08:14:09
8	Dace_	0.8612	9.59	2009-07-24 17:18:43
9	Feeds2	0.8622	9.48	2009-07-12 13:11:51
10	BigChaos	0.8623	9.47	2009-04-07 12:33:59

http://www.netflixprize.com//leaderboard

Heritage health prize - Progress Prize 1

2. *Predictive Modelling*
   Predictive models were built utilising the data sets created in Step 1. Numerous mathematical techniques were used to generate a set of candidate solutions.

3. *Ensembling*
   The individual solutions produced in Step 2 were combined to create a single solution that was more accurate than any of its components.

[Market Makers](#)

1 Introduction

My milestone 1 solution to the Heritage Health Prize with a RMSLE score of 0.457239 on the leaderboard consists of a linear blend of 21 result. These are mostly generated by relatively simple models which are all trained using stochastic gradient descent. First in section 2 I provide a description of the way the data is organized and the features that were used. Then in section 3 the training method and the post-processing steps are described. In section 4 each individual model is briefly described, all the relevant meta-parameter settings can be found in appendix Parameter settings. Finally the weights in the final blend are given in section 5.

4/13

[Mestrom](#)

Basic intuition - majority vote

Suppose we have 5 completely independent classifiers

If accuracy is 70% for each:

- $10 \times (0.7)^3(0.3)^2 + 5 \times (0.7)^4(0.3)^2 + (0.7)^5$
- 83.7% majority vote accuracy

With 101 independent classifiers

- 99.9% majority vote accuracy

Approaches for combining classifiers

1. Bagging (see previous lecture)
2. Boosting
3. Combining different classifiers

Example

```
#library(devtools)
#install_github("medley","mewo2")
library(medley)
set.seed(453234)
y <- rnorm(1000)
x1 <- (y > 0); x2 <- y*rnorm(1000)
x3 <- rnorm(1000,mean=y,sd=1); x4 <- (y > 0) & (y < 3)
x5 <- rbinom(1000,size=4,prob=exp(y)/(1+exp(y)))
x6 <- (y < -2) | (y > 2)
data <- data.frame(y=y,x1=x1,x2=x2,x3=x3,x4=x4,x5=x5,x6=x6)
train <- sample(1:1000,size=500)
trainData <- data[train,]; testData <- data[-train,]
```

Basic models

```
library(tree)
lm1 <- lm(y ~.,data=trainData)
rmse(predict(lm1,data=testData),testData$y)
```

```
[1] 1.294
```

```
tree1 <- tree(y ~.,data=trainData)
rmse(predict(tree1,data=testData),testData$y)
```

```
[1] 1.299
```

```
tree2 <- tree(y~.,data=trainData[sample(1:dim(trainData)[1]),])
```

Combining models

```
combine1 <- predict(lm1,data=testData)/2 + predict(tree1,data=testData)/2
rmse(combine1,testData$y)
```

```
[1] 1.281
```

```
combine2 <- (predict(lm1,data=testData)/3 + predict(tree1,data=testData)/3
             + predict(tree2,data=testData)/3)
rmse(combine2,testData$y)
```

```
[1] 1.175
```

Medley package

```
#library(devtools)
#install_github("medley","mewo2")
library(medley)
library(e1071)
library(randomForests)
x <- trainData[,-1]
y <- trainData$y
newx <- testData[,-1]
```

http://www.kaggle.com/users/10748/martin-o-leary

Blending models (part 1)

```
m <- create.medley(x, y, errfunc=rmse);
for (g in 1:10) {
  m <- add.medley(m, svm, list(gamma=1e-3 * g));
}
```

```
CV model 1 svm (gamma = 0.001) time: 0.362 error: 0.5557
CV model 2 svm (gamma = 0.002) time: 0.373 error: 0.5367
CV model 3 svm (gamma = 0.003) time: 0.38 error: 0.5345
CV model 4 svm (gamma = 0.004) time: 0.376 error: 0.5333
CV model 5 svm (gamma = 0.005) time: 0.364 error: 0.5301
CV model 6 svm (gamma = 0.006) time: 0.355 error: 0.5265
CV model 7 svm (gamma = 0.007) time: 0.365 error: 0.5197
CV model 8 svm (gamma = 0.008) time: 0.359 error: 0.5115
CV model 9 svm (gamma = 0.009) time: 0.369 error: 0.5026
CV model 10 svm (gamma = 0.01) time: 0.355 error: 0.4946
```

Blending models (part 2)

```
for (mt in 1:2) {
  m <- add.medley(m, randomForest, list(mtry=mt));
}
```

```
 CV model 11 randomForest (mtry = 1) time: 2.015 error: 0.4668
 CV model 12 randomForest (mtry = 2) time: 3.532 error: 0.4135
```

```
m <- prune.medley(m, 0.8);
rmse(predict(m,newx),testData$y)
```

```
Sampled... 96.00 %:  3 svm (gamma = 0.01)
1.00 %:  4 svm (gamma = 0.009)
1.00 %:  5 svm (gamma = 0.008)
1.00 %:  6 svm (gamma = 0.007)
1.00 %:  7 svm (gamma = 0.006)
CV error: 0.4953
```

Notes and further resources

Notes:

- Even simple blending can be useful
- Majority vote is typical model for binary/multiclass data
- Makes models hard to interpret

Further resources:

- Bayesian model averaging
- Heritage health prize
- Netflix model blending