Boosting (afternoon)

Objectives

By the end of this lecture you will be able to answer the following questions:

- What is boosting as a general algorithm?
- What is AdaBoost? How do you implement it?
- What are partial dependency plots? How do you interpret them? How do you build them?

Gradient Boosting - Loss functions

Gradient Boosting - Loss functions

Up to now we only talked about the squared error a loss function (boosted trees) but gradient boosting generalizes easily to other loss functions

Popular loss functions include:

Name	Loss Function
Squared error	$\frac{1}{2}(y_i - \hat{y}_i)^2$
Absolute error	$\lvert y_i - \hat{y}_i \rvert$
Exponential loss	$exp(-y_i \cdot \hat{y}_i)$

Gradient Boosting - Loss functions

The particular loss function you use depends on your problem

- In regression problems, squared or absolute error is used
- In classification problems, exponential loss leads to AdaBoost

There are many more possibilities! E.g., the logistic log likelihood loss function for the gradient boosted logistic regression

Gradient Boosted Logistic Regression

Logistic Regression - Review

Recall **logistic regression**:

$$\hat{\beta} = argmin_{\beta} L(\beta)$$

Where

- $L(\beta) = \sum_{i=1}^{N} \ell(\beta, x_i, y_i)$
- $\ell(\beta, x, y) = y \cdot log(p(\beta, x)) - (1 - y) \cdot log(1 - p(\beta, x))$
- $p(\beta, x) = \frac{1}{1+e^{-u(\beta,x)}}$
- $u(\beta, x) = x\beta = \beta_0 + \beta_1 x_1 + ... + \beta_K x_K$

We interpreted $p(x)$ as *the conditional probability that* $y = 1$, *given* x: $p(x) = P(y = 1 \mid x)$

Gradient Boosted Logistic Regression

Gradient Boosted Logistic Regression:

Replace the linear predictor in logistic regression

$$u(\beta, x) = \beta_0 + \beta_1 x_1 + ... + \beta_K x_K$$

With a sum of small regression trees

$$u(x) = u_M(x) = G_M(x) = \phi_0(x) + \phi_1(x) + ... + \phi_M(x)$$

Gradient Boosted Logistic Regression

To fit a Gradient Boosted Logistic Regression, replace the least squares loss function with the logistic loss

$$L(u) = \sum_{i=1}^{N} \ell(u, x_i, y_i)$$

Where

- $\ell(u, x, y) = y \cdot log(u, p(x)) - (1 - y) \cdot log(1 - p(u, x))$
- $p(u, x) = \frac{1}{1+e^{-u(x)}}$

Then use the same gradient boosting technique

Gradient Boosted Logistic Regression - Derivation

At step m, the gradient of the logistic loss is

$$\nabla_{u_m} L(u_m) = \sum_{i=1}^{N} \nabla_{u_m} \ell(u_m, x_i, y_i)$$

$$\nabla_{u_m} \ell(u_m, x, y) = \frac{\partial \ell}{\partial u_m}(u_m, x, y)$$

Gradient Boosted Logistic Regression - Derivation

$$\frac{\partial \ell}{\partial u} = \frac{\partial \ell}{\partial p}\frac{\partial p}{\partial u}$$

$$\frac{\partial \ell}{\partial p} = y\frac{1}{p} + (1-y)\frac{1}{1-p}(-1) = \frac{y-p}{p(1-p)}$$

$$\frac{\partial p}{\partial u} = (-1)(1+e^{-u})^{-2}e^{-u}(-1) = \frac{e^{-u}}{(1+e^{-u})^2} = p(1-p)$$

$$\frac{\partial \ell}{\partial u} = y - p$$

Gradient Boosted Logistic Regression - Derivation

At step m, the gradient of the logistic loss is

$$\nabla_{u_m} L(u_m) = \sum_{i=1}^{N} y_i - p(u_m(x_i))$$

We can interpret this method as "boosting to the residual probabilities"

Gradient Boosted Logistic Regression

Gradient boosted logistic regression is implemented in sklearn as `GradientBoostingClassifier` whose knobs are essentially the same as those to `GradientBoostingRegressor`:

```
GradientBoostingClassifier(
    loss='deviance',
    n_estimators=100,
    learning_rate=0.1,
    max_depth=3,
    subsample=1.0,
    min_samples_split=2,
    min_samples_leaf=1,
    min_weight_fraction_leaf=0.0,
    ...)
```

So everything we said this morning applies

Adaptive Boosting (AdaBoost)

AdaBoost - General Overview

AdaBoost (short for Adaptive Boosting) is gradient boosting when we plug in an **exponential loss** function

Note that for AdaBoost, our dataset is $\{x_i, y_i\}$, $y_i \in \{+1, -1\}$ (negative elements are denoted by -1, not 0)

AdaBoost - Derivation

Step m:

$$G_m = G_{m-1} + \alpha_m \phi_m$$

Question: Given a weak classifier $\phi_m(x)$ that minimizes $\sum_{i=1}^{N} w_i \cdot I(y_i \neq \phi_m(x_i))$, how should we choose α_m?

Answer: We want to choose α_m that minimizes the loss function $L(G_m)$

$$L(G_m) = \sum_{i=1}^{N} \ell(G_m, x_i, y_i)$$

With

- $\ell(G_m, x, y) = exp(-y \cdot G_m(x))$

AdaBoost - Derivation

$$\frac{\partial L}{\partial \alpha_m}(G_m) = 0$$

...

(check Sean Sall's appendix for his excellent derivation of AdaBoost)

...

$$\alpha_m = log((1 - err_m)/err_m)$$

With

- $err_m = \frac{\sum_{i=1}^{N} w_i \cdot I(y_i \neq \phi_m(x_i))}{\sum_{i=1}^{N} w_i}$

AdaBoost Algorithm

1. Initialize the observation weights $w_i = \frac{1}{N}$ for $i = 1, 2, ..., N$
2. For $m = 1$ to M, do:
   - 2.1 Fit a weak classifier $\phi_m(x)$ to the training data using weights w_i, minimizing $\sum_{i=1}^{N} w_i \cdot I(y_i \neq \phi_m(x_i))$
   - 2.2 Compute $err_m = \frac{\sum_{i=1}^{N} w_i \cdot I(y_i \neq \phi_m(x_i))}{\sum_{i=1}^{N} w_i}$
   - 2.3 Compute $\alpha_m = log((1 - err_m)/err_m)$
   - 2.4 Set $w_i = w_i \cdot exp[\alpha_m \cdot I(y_i \neq \phi_m(x_i))]$, $i = 1, 2, ..., N$
3. Output $G_M(X) = \sum_{m=1}^{M} \alpha_m \phi_m(x)$, and $G(X) = sign[G_M(X)]$

AdaBoost - Interpretation

AdaBoost has a nice interpretation of assigning weights to individual observations, and then iteratively adjusting those weights based on how well we are predicting for each individual observation (hence the name adaptive)

- With Adaboost, 'shortcomings' (of the previous weak learners) are identified by highly weighted samples

On the other end, with gradient boosting, 'shortcomings' are identified by large residuals, i.e., gradients

AdaBoost - Visual

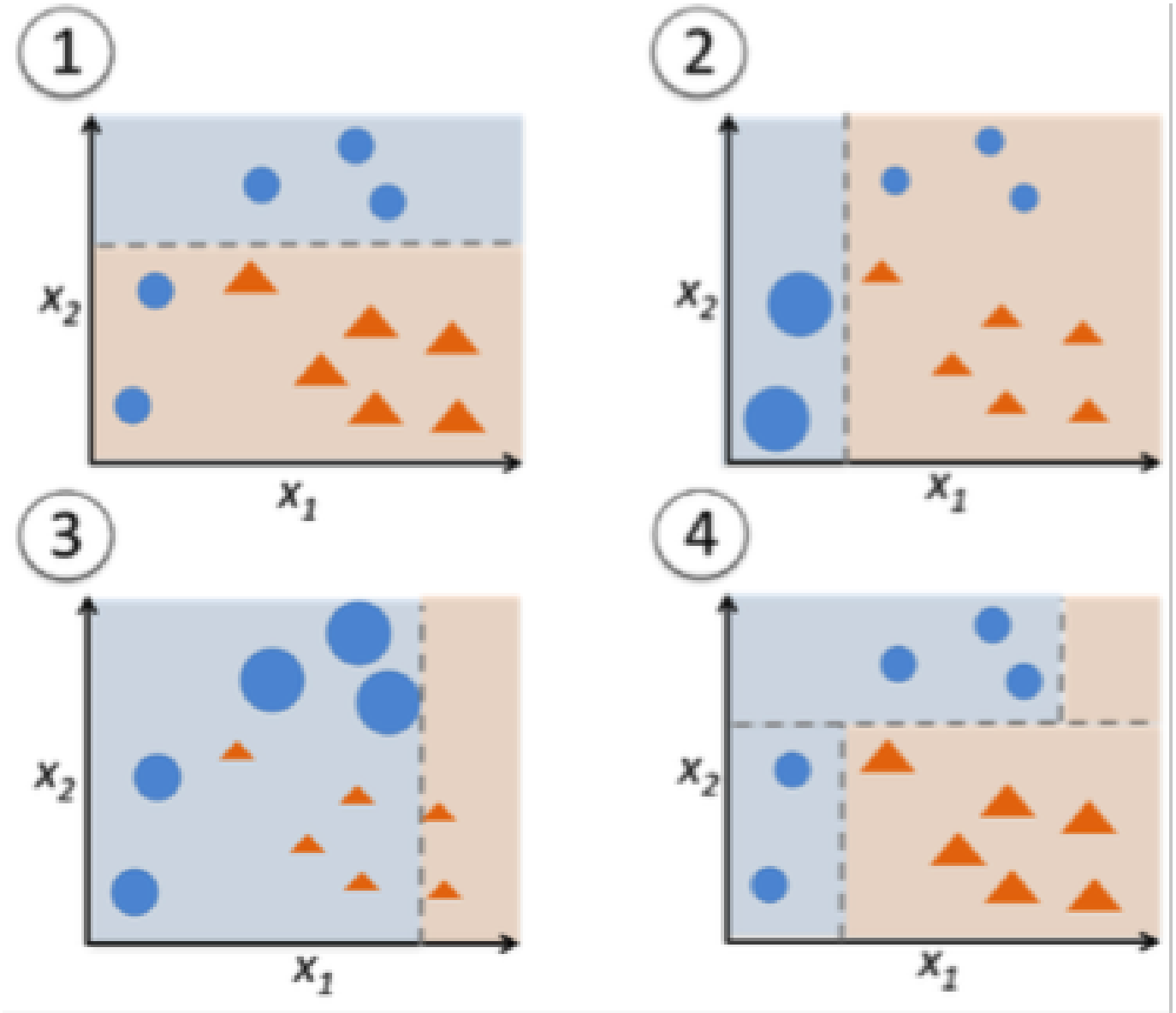

Figure 1:AdaBoost

Interpreting Gradient Boosting

It's time for an example

Let's try to predict median housing value in California for neighborhoods (pictures from the Elements of Statistical Learning)...

Figure 2:Real estate?

Interpreting Gradient Boosting

Gradient boosting models, while offering massive predictive power, are very complex and hard to interpret

There are two popular high-level summarization techniques that can help you understand the high-level content of the model and diagnose issues

- **Relative Variable Importance**: Measures the amount a predictor "participates" in the model fitting procedure
- **Partial Dependence Plots**: Are analogous to parameter estimates in linear regressions, they summarize the effect of a single predictor while controlling for the effect of all others

Relative Variable Importance - Overview

Remember that when using random forests, we can get a look at which variables are most important by looking at feature importance

The concept here is the same

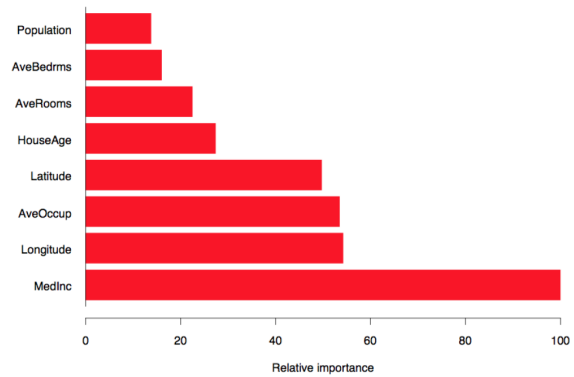

Figure 3:Feature Importances on California Housing

Relative Variable Importance

For each tree we grow, we keep track of how much the error metric decreases at each split and allocate that decrease to a predictor

The importance of a predictor in a **tree** is the **total** amount the error metric decreased over all splits on that predictor

The importance of a predictor in the **boosted model** is the **normalized average** importance of the predictor over all the trees

Relative Variable Importance - Comments

- The name "feature importances" is misleading
  - A highly-ranked ("important") feature might not be statistically significant
- If your model contains both numeric and binary predictors, the importance metric is biased to assign higher values to the numeric predictors
  - Try not to compare feature importances across these two classes
- Feature importance rankings can have very high variance
  - Make sure any important conclusions are robust to different random seeds and training sets
- Make sure your model only includes *up to the optimal point*, otherwise you'll allocate importance to overfitting
- Dominant features should be treated with suspicion
  - They can often be a sign of data leakage

Partial Dependency Plots - Overview

A **partial dependency plot** is a visualization of the effect of a single predictor, averaging out the effects of all the rest

- After fitting the model, we cycle over some pre-determined values of the individual variable of interest, predicting on those values and observing how our responses changes
- How the response changes across different values of our variable of interest is the **partial dependency** of the response on that variable

$$pd_j(x) = \overbrace{\frac{1}{N}\sum_{i=1}^{N}}^{\text{average accross all training samples}} f\left(x_{i,1}, \ldots, x_{i,j-1}, x, x_{i,j+1}, \ldots, x_{i,M}\right)$$

Partial Dependency Plots - Visual

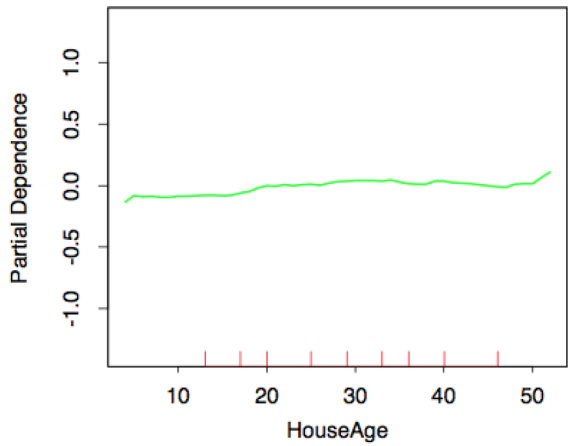

Figure 4:Partial dependence of median house value on median age of houses in the neighborhood

Here, we can see that once we control for the average effects of all other variables, median house value has a small partial dependence on median age of the house

Partial Dependency Plots - Visual

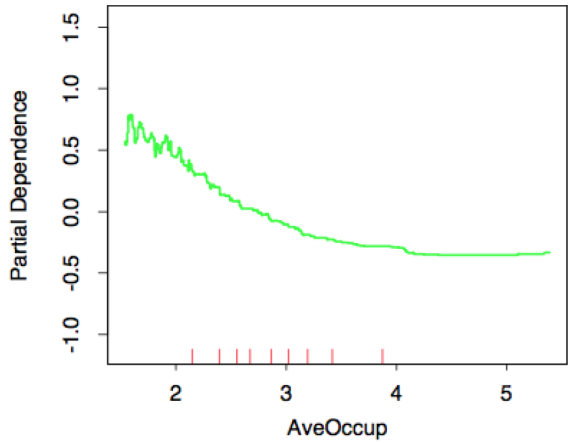

Figure 5:Partial dependence of median house value on average occupancy of houses in the neighborhood

Here, we can see that once we control for the average effects of all other variables, median house value has a noticeable partial dependence on the average occupancy of houses in the neighborhood, up to around 4 and levels off after that

Partial Dependency Plots - Visual

We can even plot the partial dependency of two variables relative to the response (more than two gets tough):

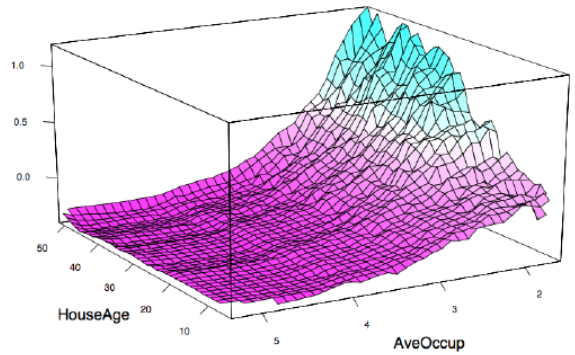

Figure 6:Partial dependence of median house value on median house age and average occupancy

Here, we see that there is a strong interaction between *HouseAge* and *AveOccup*, which we weren't able to see in looking at either the feature importances or partial dependency plots of a single variable

Partial Dependency Plots - Visual

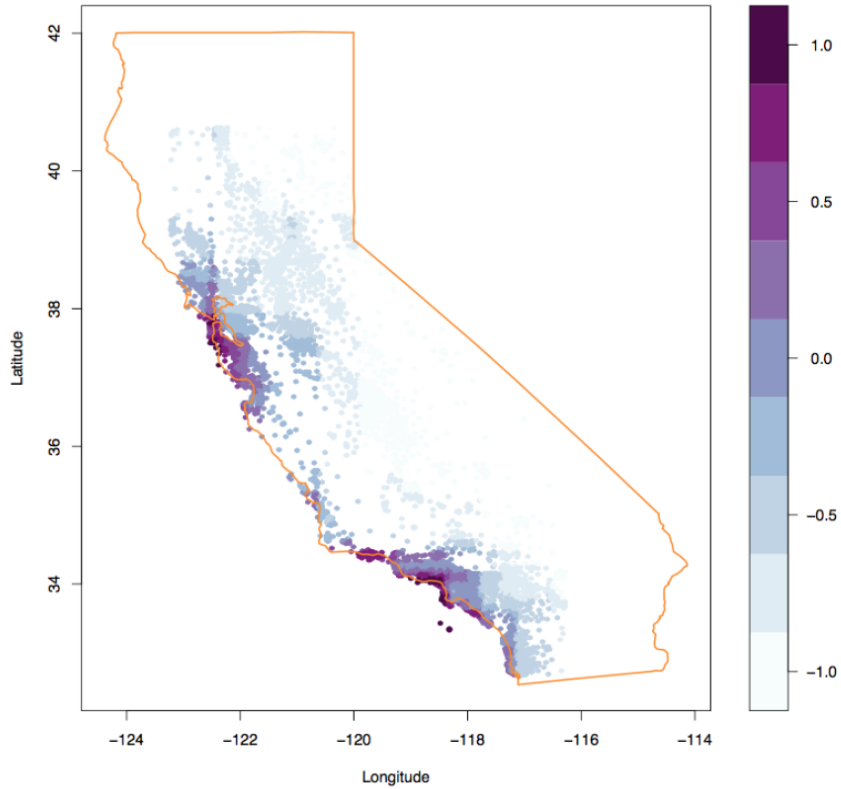

Figure 7:Partial dependence of median house value on latitude and longitude

Gradient Boosting - Wrap-up

Gradient Boosting - (More) Disadvantages

Gradient Boosting is one of the best learning algorithm available today. Nonetheless, it has drawbacks:

Boosting creates very complex models that are therefore difficult to explain

- Extracting intuitive, conceptual, or inferential information from them is a challenging
- It can be difficult to convince business leader to accept such a black-box model