Bagging, Boosting and RANSAC
Bootstrap Aggregation
Bagging

- The Main Idea
- Some Examples
- Why it works
The Main Idea

Aggregation

• Imagine we have $m$ sets of $n$ independent observations
  
  $S^{(1)} = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}^{(1)}, \ldots, S^{(m)} = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}^{(m)}$
  
  all taken i.i.d. from the same underlying distribution $P$

• Traditional approach: generate some $\varphi(x, S)$ from all the data samples

• Aggregation: learn $\varphi(x, S)$ by averaging $\varphi(x, S^{(k)})$ over many $k$
The Main Idea

Bootstrapping

• Unfortunately, we usually have one single observations set $S$

• Idea: bootstrap $S$ to form the $S^{(k)}$ observation sets
  
  • (canonical) Choose some samples, duplicate them until you fill a new $S^{(i)}$ of the same size of $S$
  
  • (practical) Take a sub-set of samples of $S$, (use a smaller set)

• The samples not used by each set are validation samples
The Main Idea

Bagging

- Generate $S^{(1)},..., S^{(m)}$ from bootstrapping
- Compute the $\varphi(x, S)^{(k)}$ individually
- Compute $\varphi(x, S) = E_k(\varphi(x, S)^{(k)})$ by aggregation
A concrete example

• We select some input samples
• We learn a regression model
• Very sensitive to the input selection
• \( m \) training sets = \( m \) different models

\[
(x_1^{(1)}, y_1^{(1)}), \ldots, (x_n^{(1)}, y_n^{(1)}) \rightarrow \hat{f}^{(1)}(x) = Y^{(1)}
\]

\[
(x_1^{(2)}, y_1^{(2)}), \ldots, (x_n^{(2)}, y_n^{(2)}) \rightarrow \hat{f}^{(2)}(x) = Y^{(2)}
\]

\[
\hat{f}^{(1)}, \ldots, \hat{f}^{(m)} \rightarrow Y^{(1)}, \ldots, Y^{(m)}
\]
Aggregation: combine several models

- Linear combination of simple models
- More examples = Better model
- We can stop when we’re satisfied

\[ Z = \frac{1}{m} \sum_{i=1}^{m} Y^{(i)} \quad m = 60 \]
Proof of convergence

Hypothesis: The average will converge to something meaningful

- Assumptions
  - \( Y^{(1)}, \ldots, Y^{(m)} \) are iid
  - \( E(Y) = y \) (\( E(Y) \) is an unbiased estimator of \( y \))

- Expected Error
  \[ E((Y - y)^2) \]

- With Aggregation
  \[ Z = \frac{1}{m} \sum_{i=1}^{m} Y^{(i)} \]
  \[ E((Z - y)^2) \]
  \[ = \frac{1}{m^2} \sum_{i=1}^{m} \sigma^2(Y^{(i)}) \]

infinite observations = zero error: we have our underlying estimator!
In layman terms

The expected error (variance) of $Y$ is larger than $Z$

The variance of $Z$ shrinks with $m$
Relaxing the assumptions

- We DROP the second assumption
  - \(Y^{(1)}, \ldots, Y^{(m)}\) are iid
  - \(E(Y) = y\) (\(E(Y)\) is an unbiased estimator of \(y\))

\[
E((Y - y)^2) = \sigma^2(Y) \geq 0
\]

we add these

\[
E((Y - E(Y))^2) + E((E(Y) - y)^2) + E(2(Y - E(Y))(E(Y) - y))
\]

we regroup them

\[
E(Y - E(Y))E(2(E(Y) - y)) = 0
\]

\[
E((Y - y)^2) \geq E((E(Y) - y)^2)
\]

\[
E((Y - y)^2) \geq E((Z - y)^2)
\]

using \(Z\) gives us a smaller error

(even if we can’t prove convergence to zero)
Peculiarities

• Instability is good
  • The more variable (unstable) the form of \( \varphi(x, S) \) is, the more improvement can potentially be obtained
  • Low-variability methods (e.g. PCA, LDA) improve less than high-variability ones (e.g. LWR, Decision Trees)

• Loads of redundancy
  • Most predictors do roughly “the same thing”
From Bagging to Boosting

• Bagging: each model is trained independently
• Boosting: each model is built on top of the previous ones
Adaptive Boosting

AdaBoost

• The Main Idea
• The Thousand Flavours of Boost
• Weak Learners and Cascades
The Main Idea

Iterative Approach

• Combine several simple models (weak learners)
• Avoid redundancy
  • Each learner complements the previous ones
• Keep track of the errors of the previous learners
Weak Learners

• A “simple” classifier that can be generated easily
  • As long as it is better than random, we can use it

• Better when tailored to the problem at hand
  • E.g. very fast at retrieval (for images)
AdaBoost

Initialization

• We choose a weak learner model $\varphi(x)$
  
  (e.g. $f(x, v) = x \cdot v > \theta$)

• Initialization
  
  • Generate $\varphi_1(x), \ldots, \varphi_N(x)$ weak learners
    
    • $N$ can be in the hundreds of thousands
  
  • Assign a weight $w_i$ to each training sample
AdaBoost

Iterations

• Compute the error $e_j$ for each classifier $\varphi_j(x)$

$$e_j : \sum_{i=1}^{n} \left( w_i \cdot 1_{\varphi_j(x_i) \neq y_i} \right)$$

• Select the $\varphi_j$ with the smallest classification error

$$\text{argmin}_j \left( \sum_{i=1}^{n} \left( w_i \cdot 1_{\varphi_j(x_i) \neq y_i} \right) \right)$$

• Update the weights $w_i$ depending on how they are classified by $\varphi_j$.

Here comes the important part
Updating the weights

Evaluate how “well” $\varphi_j(x)$ is performing

$$\alpha = \frac{1}{2} \ln \left( \frac{1 - e_j}{e_j} \right)$$

Update the weights for each sample

$$w_i^{(t+1)} = \begin{cases} 
  w_i^{(t)} \exp(\alpha^{(t)}) & \text{if } \varphi_j(x_i) \neq y_i, \\
  w_i^{(t)} \exp(-\alpha^{(t)}) & \text{if } \varphi_j(x_i) = y_i.
\end{cases}$$
AdaBoost
Rinse and Repeat

- Recompute the error $e_j$ for each classifier $\varphi_j(x)$ using the updated weights

$$e_j : \sum_{i=1}^{n} (w_i \cdot 1_{\varphi_j(x_i) \neq y_i})$$

- Select the new $\varphi_j$ with the smallest classification error

$$\arg\min_j \left( \sum_{i=1}^{n} (w_i \cdot 1_{\varphi_j(x_i) \neq y_i}) \right)$$

- Update the weights $w_i$. 
Boosting In Action

The Checkerboard Problem
Boosting In Action

Initialization

• We choose a simple weak learner

\[ f(x, v) = x \cdot v > \theta \]

• We generate a thousand random vectors \( v_1, \ldots, v_{1000} \) and corresponding learners \( f_j(x, v_j) \)

• For each \( f_j(x, v_j) \) we compute a good threshold \( \theta_j \)
Boosting In Action

- We look for the best weak learner
- We adjust the importance (weight) of the errors
- Rinse and repeat

and we keep going...
Boosting In Action
Drawbacks of Boosting

• Overfitting!
  • Boost will always overfit with many weak learners

• Training Time
  • Training of a face detector takes up to 2 weeks on modern computers
A thousand different flavors

- A couple of new boost variants every year
  - Reduce overfitting
  - Increase robustness to noise
  - Tailored to specific problems
- Mainly change two things
  - How the error is represented
  - How the weights are updated
An example

• Instead of counting the errors, we compute the probability of correct classification

Discrete AdaBoost

\[
e_j = \sum_{i=1}^{n} \left( w_i \cdot 1_{\varphi_j(x_i) \neq y_i} \right)
\]

\[
\alpha = \frac{1}{2} \ln \left( \frac{1 - e_j}{e_j} \right)
\]

\[
w_{i}^{(t+1)} = \begin{cases} 
    w_{i}^{(t)} \exp(\alpha^{(t)}) & \text{if } \varphi_j(x_i) \neq y_i, \\
    w_{i}^{(t)} \exp(-\alpha^{(t)}) & \text{if } \varphi_j(x_i) = y_i.
\end{cases}
\]

Real AdaBoost

\[
p_j = \prod_{i=1}^{n} w_i P(y_i = 1 | x_i)
\]

\[
\alpha = \frac{1}{2} \ln \left( \frac{1 - p_j}{p_j} \right)
\]

\[
w_{i}^{(t+1)} = w_{i}^{(t)} \exp(-y_i \alpha^{(t)})
\]
A celebrated example
Viola-Jones Haar-Like wavelets

\[ I(x) : \text{pixel of image } I \text{ at position } x \]

\[ f(x) = \sum_{x \in A} I(x) - \sum_{x \in B} I(x) \]

\[ \varphi(x) = \begin{cases} 1 & \text{if } f(x) > 0, \\ -1 & \text{otherwise}. \end{cases} \]

- Millions of possible classifiers

2 rectangles of pixels
1 positive, 1 negative

\[ \varphi_1(x) \quad \varphi_2(x) \]
Real-Time on HD video
Some simpler examples

Feature: the distance from a point $c$

$$f(x, c) = (x - c)^T(x - c) > \theta$$
Some simpler examples

Feature: being inside a rectangle $R$

$$f(x, R) = 1_{x \in R}$$
Some simpler examples

Feature: full-covariance gaussian

\[ f(x, \mu, \Sigma) = P(x | \mu, \Sigma) \]
Weak Learners don’t need to be weak!

20 boosted SVMs with 5 SVs and the RBF kernel
Cascades of Weak Classifiers

- Split classification task in Stages
- Each stage has increasing numbers of weak classifiers
- Each stage only needs to ‘learn’ to classify what the previous ones let through
Cascades of Weak Classifiers

1000 samples → 10’000 tests
100 samples

1000 weak learners
Stage 1

10’000 tests + 10’000 tests = 50’000 tests
100 samples + 50 samples = 70 tests per sample

90% classification
Stage 2

95% classification
Stage 3

99.9% classification

70 tests per sample instead of 1110

yes
Cascades of Weak Classifiers

• Advantages

  • Stage splits can be chosen manually
  • Trade-off performance and accuracy of the first stages

• Disadvantages

  • Later stages become difficult to train (few samples)
  • Very large amount of samples for training
  • Even slower to train than standard boosting
The curious case of RANSAC

RANdom SAmple Consensus

• Created to withstand hordes of outliers

• Hasn’t really been proven to converge to optimal solution in any reasonable time

• Extremely effective in practice
  • Easy to implement
  • Widely used in Computer Vision
RANSAC in practice

- Select a random subset of samples
- Estimate the model on these samples
- Sift through all other samples
  - If they are close to the model, add to the Consensus
- If the consensus is big enough, keep it
- Repeat from top

Keep the best consensus
(e.g. most samples, least error, etc.)
Some examples of RANSAC

Reference Panoramic Image

Input Video

Some examples of RANSAC

Scaramuzza et al. (ETHZ)

Input image: 360° Camera

Compute visual features (KLT)

Exclude Outliers (bad features)

Compute Visual Odometry

# RANSAC vs Bagging

<table>
<thead>
<tr>
<th>RANSAC</th>
<th>Bagging</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select a random subset of samples</td>
<td></td>
</tr>
<tr>
<td>Train many individual models</td>
<td></td>
</tr>
<tr>
<td>Keep the best</td>
<td>Keep them all</td>
</tr>
<tr>
<td>Needs to be lucky</td>
<td>Proven to converge</td>
</tr>
<tr>
<td>Very light to compute</td>
<td>Heavy to compute</td>
</tr>
</tbody>
</table>
Summing up

- **Bagging**: linear combination of multiple learners
  - + Very robust to noise
  - - A lot of redundant effort

- **Boosting**: weighed combination of arbitrary learners
  - + Very strong learner from very simple ones
  - - Sensitive to noise (at least Discrete Adaboost)

- **RANSAC**: iterative evaluation of random learners
  - + Very robust against outliers, simple to implement
  - - Not ensured to converge (although in practice it does)