Ensemble Methods

- An ensemble is a set of classifiers that learn a target function, and their individual predictions are combined to classify new examples.

- Covered so far:
  - Bagging
  - Boosting
  - Random Forests
  - Extra Trees

- Today:
  - Stacking
  - Blending
Stacked Generalization (Stacking)

• **Underlying idea:** *Learn whether training data have been properly learned*

• 2 tiers of classifiers
  – If a particular base-classifier incorrectly learned a certain region of the feature space, the second tier (meta) classifier may be able to detect this undesired behavior
  – Along with the learned behaviors of other classifiers, it can correct such improper training
The Stacking Framework

• There are two approaches for combining models: **voting** and **stacking**

• Difference between the stacking framework and those previously discussed:
  – In contrast to stacking, no learning takes place at the meta-level when combining classifiers by a voting scheme
  – Label that is most often assigned to a particular instance is chosen as the correct prediction when using voting
The Stacking Framework

• Stacking is concerned with combining multiple classifiers generated by different learning algorithms $L_1, \ldots, L_N$ on a single dataset $S$, which is composed by a feature vector $s_i = (x_i, y_i)$.

• The stacking process can be broken into two phases:
  1. Generate a set of base-level classifiers $C_1, \ldots, C_N$
     • Where $C_i = L_i(S)$
  2. Train a meta-level classifier to combine the outputs of the base-level classifiers
The Stacking Framework

Training Set

Hypotheses

Predictions on Training Observations

Meta Learner

Meta Learner’s Hypothesis

Final Prediction
The Stacking Framework

• The training set for the meta-level classifier is generated through a leave-one-out cross validation process.
  \[ \forall i = 1, \ldots, n \text{ and } \forall k = 1, \ldots, N : C^i_k = L_k(S - s_i) \]

• The learned classifiers are then used to generate predictions for \( s_i \):
  \[ \hat{y}^k_i = C^i_k(x_i) \]

• The meta-level dataset consists of examples of the form \( (\hat{y}^1_i, \ldots, \hat{y}^n_i), y_i \), where the features are the predictions of the base-level classifiers and the class is the correct class of the example in hand.
Stacking with Probability Distributions

• Proposed by Ting and Witten (1999)
• The idea:
  – Stack base-level classifiers output probability distributions (PDs) over the set of class values, instead of single class values – *somewhat similar to logistic regression*
  – The meta-level classifiers can then use these for training.
  – PDs represent the confidence of the base-level classifiers, and hence the authors argue that using this information is more appropriate
Stacking with Probability Distributions

• The prediction of a base-level classifier $C$ applied to $x$:  
  \[ p^C(x) = (p^C(c_1|x), p^C(c_2|x), ..., p^C(c_n|x)) \]
  \{c_1, c_2, ..., c_n\} is the set of possible class values
  \[ p^C(c_i|x) \] denotes the probability that example $x$ belongs to class $c_i$ as estimated by classifier $C$

• The class $c_j$ with highest probability is predicted by classifier $C$. 
Stacking with Probability Distributions

- Multi-response linear regression (MLR) is recommended for meta-level learning (Ting & Witten, 1999)
  - For a classification problem with $m$ class values $\{c_1, \ldots, c_m\}$, $m$ regression problems are formulated
    - For each class $c_j$, a linear equation $LR_j$ is constructed to predict a binary value (1: class value is $c_j$, 0: otherwise)
  - Given a new $x$, $LR_j(x)$ is calculated for all $j$ and the class $k$ is predicted with maximum $LR_k(x)$
Mathematical Insight Into Stacking

• If an ensemble has $M$ base models having an error rate $e < 1/2$ and if the errors are independent, then the probability that the ensemble makes an error is the probability that more than $M/2$ base models misclassify that instance.

• In essence, the meta-classifier is trained to learn the error of the base classifiers.

• Adding the estimated errors to the output of the base classifiers can improve prediction.
The Stacking Algorithm

**Input:** Data set $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_m, y_m)\}$;
- First-level learning algorithms $\mathcal{L}_1, \cdots, \mathcal{L}_T$;
- Second-level learning algorithm $\mathcal{L}$.

**Process:**

for $t = 1, \cdots, T$:
  $h_t = \mathcal{L}(\mathcal{D})$ % Train a first-level individual learner $h_t$ by applying the first-level learning algorithm $\mathcal{L}_t$ to the original data set $\mathcal{D}$
end;

$\mathcal{D}' = \emptyset$; % Generate a new data set
for $i = 1, \cdots, m$:
  for $t = 1, \cdots, T$:
    $z_{it} = h_t(x_i)$ % Use $h_t$ to classify the training example $x_i$
  end;
  $\mathcal{D}' = \mathcal{D}' \cup \{(z_{i1}, z_{i2}, \cdots, z_{iT}, y_i)\}$
end;
$h' = \mathcal{L}(\mathcal{D}')$. % Train the second-level learner $h'$ by applying the second-level learning algorithm $\mathcal{L}$ to the new data set $\mathcal{D}'$

**Output:** $H(x) = h'(h_1(x), \cdots, h_T(x))$
A few more thoughts on ensembles

• There are two basic ways of aggregating classification methods:
  – **After the fact:** Combines existing solutions
    • Ex.: Netflix challenge teams merging / “blending”
  – **Before the fact:** Creates solutions to be combined
    • Ex.: Bagging

• Each of these methods can be more/less appropriate to particular problems
Before-the-fact aggregation

• Suppose we want to create a robust method to determine if a given image contains a face

• **Problem:** It may be very difficult and computationally intensive to create a classifier for that task

• **One idea:** detect eyes instead of faces
  – **Advantage:** A lot more efficient
  – **Problem #2:** this may produce models with low accuracy
  – **Solution:** Combine it with other similar classifiers
Before-the-fact aggregation – Detecting Faces

- Eyes detector
- Mouth detector
- Ears detector
- Skin detector

Meta Classifier

High accuracy!

Low individual accuracy
Computationally efficient
After-the-fact aggregation

- Netflix challenge teams merging
- Different models are already built. Find an intelligent way to combine them
- Given each hypothesis $h_i, \ldots, h_N$ and a new instance $x$, we can use the previously discussed regression to compute the prediction $g(x)$

$$h_1, h_2, \ldots, h_N \rightarrow g(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$$

- Choose $\alpha_t$'s as to minimize the error on aggregation set
  - Can be done using min-squared-error
After-the-fact aggregation

• One highly ranked team proposed
  *Merge with us by giving us your solution. We will split the winnings according to your contribution.*
  
  – To determine the value of each new team’s hypothesis, they re-evaluated the aggregation set while stacking each new model sequentially
  – Each increase in performance denoted the contribution of that new model
  – Doing something weird (different than what other teams may have done), yielded higher contribution values.
  • Models with 3% accuracy often contributed more than their counterparts that achieved much higher accuracies alone
  – This team ended up ranking second overall and their model was ultimately adopted by Netflix
More on the Netflix Challenge

- **BellKor’s Pragmatic Chaos** (1st place) was a hybrid team: KorBell (AT&T Research), which won the first Progress Prize milestone in the contest, combined with the Austrian team Big Chaos to improve their score.
- **The Ensemble** (2nd place) was also a composite team.
- BellKor’s Pragmatic Chaos submitted their solution 10 minutes earlier than the second-place team, The Ensemble, while the two teams’ algorithms were a perfect tie, score-wise.
- The dataset is still available (if you want to beat them).
More on the Netflix Challenge

Leaderboard

Showing Test Score. Click here to show quiz score

Display top 20 leaders.

<table>
<thead>
<tr>
<th>Rank</th>
<th>Team Name</th>
<th>Best Test Score</th>
<th>% Improvement</th>
<th>Best Submit Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BellKor's Pragmatic Chaos</td>
<td>0.8567</td>
<td>10.06</td>
<td>2009-07-25 18:18:28</td>
</tr>
<tr>
<td>2</td>
<td>The Ensemble</td>
<td>0.8567</td>
<td>10.05</td>
<td>2009-07-25 18:38:22</td>
</tr>
<tr>
<td>3</td>
<td>Grand Prize Team</td>
<td>0.8582</td>
<td>9.90</td>
<td>2009-07-10 21:24:40</td>
</tr>
<tr>
<td>4</td>
<td>Opera Solutions and Vandalay United</td>
<td>0.8588</td>
<td>9.84</td>
<td>2009-07-10 01:12:31</td>
</tr>
<tr>
<td>5</td>
<td>Vandelay Industries</td>
<td>0.8591</td>
<td>9.81</td>
<td>2009-07-10 00:32:20</td>
</tr>
<tr>
<td>6</td>
<td>PragmaticTheory</td>
<td>0.8594</td>
<td>9.77</td>
<td>2009-06-24 12:06:56</td>
</tr>
<tr>
<td>7</td>
<td>BellKor in PingChaos</td>
<td>0.8601</td>
<td>9.70</td>
<td>2009-05-13 08:14:09</td>
</tr>
<tr>
<td>8</td>
<td>face</td>
<td>0.8612</td>
<td>9.59</td>
<td>2009-07-24 17:18:43</td>
</tr>
</tbody>
</table>