Consolidation and Review

Applied Machine Learning
Derek Hoiem
Learning a model

\[ \theta^* = \operatorname{argmin}_{\theta} \text{Loss}(f(X; \theta), y) \]

• \( f(X; \theta) \): the model, e.g. \( y = w^T x \)
• \( \theta \): parameters of the model
• \((X, y)\): pairs of training samples
• \( \text{Loss}() \): defines what makes a good model
  – Good predictions, e.g. minimize \(-\sum_n \log P(y_n|x_n)\)
  – Likely parameters, e.g. minimize \(w^T w\)
    • Regularization and priors indicate preference for particular solutions, which tends to improve generalization (for well chosen parameters) and can be necessary to obtain a unique solution
Prediction using a model

\[ y_t = f(x_t; \theta) \]

• Given some new set of input features \( x_t \), model predicts \( y_t \)
  – Regression: output \( y_t \) directly, possibly with some variance
  – Classification
    • Output most likely \( y_t \) directly, as in nearest neighbor or Naïve Bayes
    • Output \( P(y_t|x_t) \), as in logistic regression
Model evaluation process

1. Collect/define training, validation, and test sets
2. Decide on some candidate models and parameters
3. For each candidate:
   a. Learn parameters with training set
   b. Evaluate trained model on the validation set
4. Select best model
5. Evaluate best model’s performance on the test set
   - Cross-validation can be used as an alternative
   - Common measures include error or accuracy, root mean squared error, precision-recall
How to think about ML algorithms

• What is the model?
  – What kinds of functions can it represent?
  – What functions does it prefer? (regularization/prior)

• What is the objective function?
  – What “values” are implied?
  – Note that the objective function often does not match the final evaluation metric
  – Objectives are designed to be optimizable and improve generalization

• How do I optimize the model?
  – How long does it take to train, and how does it depend on the amount of training data or number of features?
  – Can I reach a global optimum?

• How does the prediction work?
  – How fast can I make a prediction for a new sample?
  – Can I find the most likely prediction according to my model?
  – Does my algorithm provide a confidence on its prediction?
## Classification methods

<table>
<thead>
<tr>
<th></th>
<th>Nearest Neighbor</th>
<th>Naïve Bayes</th>
<th>Logistic Regression</th>
<th>Decision Tree</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
<td>Instance-Based</td>
<td>Probabilistic</td>
<td>Probabilistic</td>
<td>Probabilistic</td>
</tr>
<tr>
<td><strong>Decision Boundary</strong></td>
<td>Partition by example distance</td>
<td>Usually linear</td>
<td>Usually linear</td>
<td>Partition by selected boundaries</td>
</tr>
<tr>
<td><strong>Model / Prediction</strong></td>
<td>$i^* = \arg\min_i \operatorname{dist}(X_{trn}[i], x)$</td>
<td>$y^* = \arg\max_y \prod_i P(x_i</td>
<td>y)P(y)$</td>
<td>$\omega_{\infty} \approx \log \frac{P(x</td>
</tr>
<tr>
<td><strong>Strengths</strong></td>
<td>* Low bias</td>
<td>* Estimate from limited data</td>
<td>* Powerful in high dimensions</td>
<td>* Explainable decision function</td>
</tr>
<tr>
<td></td>
<td>* No training time</td>
<td>* Simple</td>
<td>* Widely applicable</td>
<td>* Widely applicable</td>
</tr>
<tr>
<td></td>
<td>* Widely applicable</td>
<td>* Fast training/prediction</td>
<td>* Good confidence estimates</td>
<td>* Does not require feature scaling</td>
</tr>
<tr>
<td></td>
<td>* Simple</td>
<td></td>
<td>* Fast prediction</td>
<td></td>
</tr>
<tr>
<td><strong>Limitations</strong></td>
<td>* Relies on good input features</td>
<td>* Limited modeling power</td>
<td>* Relies on good input features</td>
<td>* One tree tends to either generalize poorly or underfit the data</td>
</tr>
</tbody>
</table>
## Classification methods (extended)

<table>
<thead>
<tr>
<th>Method</th>
<th>Learning Objective</th>
<th>Training</th>
<th>Inference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naïve Bayes</td>
<td>maximize ( \sum_i \left[ \log P(x_i</td>
<td>y_i; \theta) \right] + \log P(y_i; \theta_0) )</td>
<td>( \theta_{ij} = \frac{\delta(x_i = 1 \land y_i = k) + r}{\sum_j \delta(y_i = k) + Kr} )</td>
</tr>
<tr>
<td>Logistic Regression</td>
<td>( \text{minimize} \sum_i -\log(P(y_i</td>
<td>x, \theta)) + \lambda |\theta| ) ( ) where ( P(y_i</td>
<td>x, \theta) = 1/(1 + \exp(-y_i \theta^T x)) )</td>
</tr>
<tr>
<td>Linear SVM</td>
<td>( \text{minimize} \lambda \sum_i \xi_i + \frac{1}{2} |\theta| ) ( ) such that ( y_i \theta^T x \geq 1 - \xi_i ) ( \forall i ) ( \xi_i \geq 0 )</td>
<td>Quadratic programming or subgradient opt.</td>
<td>( \theta^T x &gt; t )</td>
</tr>
<tr>
<td>Kernelized SVM</td>
<td>complicated to write</td>
<td>Quadratic programming</td>
<td>( \sum_i y_i \alpha_i K(\hat{x}_i, x) &gt; 0 )</td>
</tr>
<tr>
<td>Nearest Neighbor</td>
<td>most similar features ( \rightarrow ) same label</td>
<td>Record data</td>
<td>( y_i ) ( ) where ( i = \text{argmin}_i K(\hat{x}_i, x) )</td>
</tr>
</tbody>
</table>

* Notation may differ from previous slide
## Regression methods

<table>
<thead>
<tr>
<th>Type</th>
<th>Nearest Neighbor</th>
<th>Naïve Bayes</th>
<th>Linear Regression</th>
<th>Decision Tree</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Decision Boundary</strong></td>
<td>Instance-Based</td>
<td>Probabilistic</td>
<td>Data fit</td>
<td>Probabilistic</td>
</tr>
<tr>
<td><strong>Model / Prediction</strong></td>
<td>Partition by example distance</td>
<td>Usually linear</td>
<td>Linear</td>
<td>Partition by selected boundaries</td>
</tr>
<tr>
<td></td>
<td>$i^* = \arg\min_i dist(X_{trn}[i], x)$</td>
<td>$y^* = \max_y \prod_i P(x_i</td>
<td>y)P(y)$</td>
<td>$y^* = w^T x$</td>
</tr>
<tr>
<td><strong>Strengths</strong></td>
<td>* Low bias</td>
<td>* Estimate from limited data</td>
<td>* Powerful in high dimensions</td>
<td>* Explainable decision function</td>
</tr>
<tr>
<td></td>
<td>* No training time</td>
<td>* Simple</td>
<td>* Widely applicable</td>
<td>* Widely applicable</td>
</tr>
<tr>
<td></td>
<td>* Widely applicable</td>
<td>* Fast training/prediction</td>
<td>* Fast prediction</td>
<td>* Does not require feature scaling</td>
</tr>
<tr>
<td></td>
<td>* Simple</td>
<td></td>
<td>* Coefficients may be interpretable</td>
<td></td>
</tr>
<tr>
<td><strong>Limitations</strong></td>
<td>* Relies on good input features</td>
<td>* Limited modeling power</td>
<td>* Relies on good input features</td>
<td>* One tree tends to either generalize poorly or underfit the data</td>
</tr>
<tr>
<td></td>
<td>* Slow prediction (in basic implementation)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Performance vs training size

As we get more training data:
1. The same model has more difficulty fitting the data
2. But the test error becomes closer to training error (reduced generalization error)
3. Overall test performance improves

Due to difference in $P(y|x)$ in training and test (function shift)

Due to limited power of model (model bias) and unavoidable intrinsic error (Bayes optimal error)
Example: Breast Cancer Classification

• Motivation
  – Breast cancer diagnosis from fine needle aspirates (FNA) is reported to be 94%, but results are suspected to be biased
  – Need computer-based tests that are less subjective so that FNA is a more effective diagnostic tool for breast cancer

• Collected data from 569 patients, plus 54 for held-out testing

• A user interface was created to outline borders of suspect cells, and automated measurement of ten characteristics (e.g. radius, area, compactness, ...) was performed and mean of all cells, mean of 3 largest, and std were recorded for each patient

[paper (Wolberg et al. 1995)]
Let’s explore in Python

https://colab.research.google.com/drive/1viVU62gk77THZBFuztWjxgL93xMMpiU0?usp=sharing
Method/Results from Breast Cancer Analysis Paper

• A MSM-Tree was used for classification
  – Fits a linear classifier based on a few features for each split
  – Aimed to minimize the number of splitting planes and number of features used (for simplicity and to improve generalization)
  – Final approach was splitting plane based on mean texture, worst area, and worst smoothness

• 10-fold cross validation
  – Achieved 3% error (± 1.5% for 95% confidence interval)

• Perfect accuracy in held out test set
Next week

• Ensembles: model averaging and forests
• SVM and stochastic gradient descent