CPSC 340: Machine Learning and Data Mining

Course Review/Preview
Fall 2016

Some images from this lecture are taken from Google Image Search.
Admin

• **Assignment 6:**
  – 1 late day to hand in next Monday, 2 for Wednesday, 3 for Friday.

• **Final:**
  – December 12 (8:30am – HEBB 100)
  – Covers Assignments 1-6.
  – List of topics posted.
  – Final from last year will be posted after class.
  – Closed-book, cheat sheet: 4-pages each double-sided.
Last Time: Semi-Supervised Learning

• In **semi-supervised learning** we have:
  – Usual labeled examples \( \{X, y\} \).
  – An additional set of unlabeled examples \( \tilde{X} \).

• Exam analogy for types of supervised/semi-supervised learning:
  – **Regular supervised** learning:
    • You are given the practice final with answers.
    • You want to get the answers right on the real final.
  – **Inductive SSL**:
    • You are given the practice final with answers.
    • You also have the finals from previous years (but no answers).
    • You want to get the answers right on the real final.
  – **Transductive SSL**:
    • You are given the practice final with answers.
    • You want to get the answers right on a **take-home final**.
    • You can study while knowing what questions you need to answer.
Last Time: Graph-Based Semi-Supervised Learning

• **Graph-based** (transductive) SSL uses weighted graph on examples:

![Graph examples with labels and weights]

• Find labels minimizing cost penalizing disagreements on edges.
• Similar to KNN, but labels get ‘propagated’ through unlabeled \( \tilde{x}_i \).
  – Can label cluster or manifold.
• Directly works on labeling: **only need the graph**, not the features.
  – Interpretation as **random walk** in graph or in terms of a **Markov chain**.
Today: Course Review

• The age of “big data” is upon us.
• **Data mining and machine learning** are key tools to analyze big data.
• Very similar to statistics, but more emphasis on:
  1. Computation
  2. Test error.
  4. Models that work across domains.
• Enormous and growing number of applications.
• The field is growing very fast:
  – ~2500 attendees at NIPS 2 years ago, ~5800 next week (Influence of $$$, too).
• Today: **review of topics** we covered, **overview of topics we didn’t**.
Data Representation and Exploration

We first talked about feature representation of data:
  – Each row in a table corresponds to one ‘object’.
  – Each column in that row contains a ‘feature’ of the object.

<table>
<thead>
<tr>
<th></th>
<th>&lt; 20</th>
<th>&gt;= 20, &lt; 25</th>
<th>&gt;= 25</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Discussed numerical/discrete features, feature transformations.
Discussed summary statistics like mean, quantiles, variance.
Discussed data visualizations like boxplots and scatterplots.
Supervised Learning and Decision Trees

• **Supervised learning** builds model to map from features to labels.
  – Most successful machine learning method.

<table>
<thead>
<tr>
<th>Egg</th>
<th>Milk</th>
<th>Fish</th>
<th>Wheat</th>
<th>Shellfish</th>
<th>Peanuts</th>
<th>...</th>
<th>Sick?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.7</td>
<td>0</td>
<td>0.3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0.3</td>
<td>0.7</td>
<td>0</td>
<td>0.6</td>
<td>0</td>
<td>0.01</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

• **Decision trees** consist of a sequence of single-variables ‘rules’:
  – Simple/interpretable but not very accurate.

• Greedily learn from by fitting decision stumps and splitting data.
Training, Validation, and Testing

• In machine learning we are interesting in the **test error**.
  – Performance on new data.
• IID: training and new data drawn independently from same distribution.
• Overfitting: worse performance on new data than training data.
• Fundamental trade-off:
  – How low can make the training error? (Complex models are better here.)
  – How does training error approximate test error? (Simple models are better here.)
• Golden rule: we cannot use test data during training.
• But validation set or cross-validation allow us to approximate test error.
• No free lunch theorem: there is no ‘best’ machine learning model.
Probabilistic Classifiers and Naïve Bayes

- **Probabilistic classifiers** consider probability of correct label.
  - \( p(y_i = \text{"spam"} \mid x_i) \) vs. \( p(y_i = \text{"not spam"} \mid x_i) \).

- **Generative classifiers** model probability of the features:

\[
p(y_i = \text{"spam"} \mid x_i) \propto p(x_i \mid y_i = \text{"spam"}) p(y_i = \text{"spam"})
\]

- For tractability, often make strong **independence assumptions**.
  - **Naïve Bayes** assumes independence of features given labels:

\[
p(x_i \mid y_i) = \frac{1}{d} \prod_{j=1}^{d} p(x_{ij} \mid y_i)
\]

- **Decision theory**: predictions when errors have different costs.

  Cost of false negative \( \neq \) cost false positive
Parametric and Non-Parametric Models

- **Parametric model** size does not depend on number of objects ‘n’.
- **Non-parametric model** size depends on ‘n’.
- **K-Nearest Neighbours:**
  - Non-parametric model that uses label of closest x_i in training data.
  - Accurate but slow at test time.

- **Curse of dimensionality:**
  - Problem with distances in high dimensions.
- **Universally consistent methods:**
  - achieve lowest possible test error as ‘n’ goes to infinity.
Ensemble Methods and Random Forests

• **Ensemble methods** are classifiers that have classifiers as input:
  – Boosting: improve training error of simple classifiers.
  – Averaging: reduce overfitting of complex classifiers.

• **Random forests**:
  – Ensemble method that averages random trees fit on bootstrap samples.
  – Fast and accurate, one of the best “out of the box” classifiers.
Clustering and K-Means

- **Unsupervised learning** considers features $X$ without labels.
- **Clustering** is task of grouping similar objects.

- **K-means** is a classic clustering method:
  - Represent each cluster by its mean value.
  - Learning alternates between updating means and assigning to clusters.
  - Sensitive to initialization, but some guarantees with k-means++. 
Density-Based Clustering

• **Density-based clustering** is a non-parametric clustering method:
  – Based on finding dense connected regions.
  – Allows finding non-convex clusters.

• **Grid-based pruning**: finding close points when ‘n’ is huge.
Ensemble and Hierarchical Clustering

- **Ensemble clustering** combines clusterings.
  - But need to account for **label switching** problem.
- **Hierarchical clustering** groups objects at multiple levels.
Outlier Detection

- **Outlier detection** is task of finding “significantly different” objects.
  - Global outliers are different from all other objects.
  - Local outliers fall in normal range, but are different from neighbours.

- **Approaches:**
  - **Model-based:** fit model, check probability under model (z-score).
  - **Graphical approaches:** plot data, use human judgement (scatterplot).
  - **Cluster-based:** cluster data, find points that don’t belong.
  - **Distance-based:** outlierness test of “abnormally far form neighbours”.

Association Rules

• **Association rules** find items that are frequently bought together.
  – (S => T): if you buy ‘S’ then you are likely to buy ‘T’.
  – Rules have **support**, $P(S=1)$, and **confidence**, $P(T=1 \mid S=1)$.

• **A priori algorithm** finds all rules with high support/confidence.
  – Probabilistic inequalities reduce search space.

• **Amazon’s item-to-item recommendation:**
  – Compute similarity of ‘user vectors’ for items.
    \[
    \cos(x_i, x_j) = \frac{x_i^\top x_j}{\|x_i\| \|x_j\|}
    \]
We then returned to supervised learning and **linear regression**:

- Write label as weighted combination of features: \( y_i = w^T x_i \).

**Least squares** is the most common formulation:

\[
\sum_{i=1}^{n} (w x_i - y_i)^2
\]

- Solution is a **linear system**: \((X^T X)w = X^T y\)
- Non-zero y-intercept (**bias**) by adding a feature \( x_{ij} = 1 \).
- Model non-linear effects by **change of basis**:

\[
y_i = w_0 + w_1 x_i + w_2 x_i^2
\]
Regularization, Robust Regression, Gradient Descent

• L2-regularization adds a penalty on the L2-norm of ‘w’:

\[ f(w) = \frac{1}{2} \| Xw - y \|^2 + \frac{\lambda}{2} \| w \|^2 \]

– Several magical properties and usually lower test error.

• Robust regression replaces squared error with absolute error:
  – Less sensitive to outliers.
  – Absolute error has smooth approximations.

• Gradient descent lets us find local minimum of smooth objectives.
  – Find global minimum for convex functions.
Binary Classification and Logistic Regression

• Binary classification using regression by taking the sign:

\[ y_i = \text{sign}(w^T x_i) \]

• But squared error penalizes for being too right (“bad errors”).
  – Ideal 0-1 loss is discontinuous/non-convex.
  – Logistic loss is smooth and convex approximation:

\[ f(w) = \sum_{i=1}^{n} \log(1 + \exp(-y_i w^T x_i)) \]
Support Vector Machines

- SVMs for separable data maximize margin for separable data:

- For non-separable data, hinge loss minimizes penalizes violations:
Kernel Trick

• Non-separable data can be separable in high-dimensional space:

• **Kernel trick**: linear regression using similarities instead of features.

\[
\hat{y} = \left( K(K + nI)^{-1} \right) y
\]

with \( K = ZZ^T \) and \( \hat{K} = \hat{Z}Z^T \)

– If you can compute inner product, you don’t have to store basis \( z_i \).
– Can have exponential/infinite basis.
Stochastic Gradient

- **Stochastic gradient** methods are appropriate when ‘n’ is huge.
  - Take step in negative gradient of random training example.
- Less progress per iteration, but iterations don’t depend on ‘n’.
  - Fast convergence at start.
  - Slow convergence as accuracy improves.
- With infinite data:
  - Optimizes test error directly (cannot overfit).
  - But often difficult to get working.
Feature Selection and L1-Regularization

• **Feature selection** is task of finding “relevant” variables.
  – Can be **hard to precisely define “relevant”**.

• **Hypothesis testing** methods:
  – Do tests trying to make variable ‘j’ conditionally independent of y.
  – Ignores effect size.

• **Search and score** methods:
  – Define score (L0-norm) and search for variables that optimize it.
  – Finding optimal combination is hard, but heuristics exist (forward select).

• **L1-regularization**:
  – Formulate as a convex problem.
  – Very fast but prone to false positives.

\[
\ell(w) = \frac{1}{2} \sum_{i=1}^{n} (w^T x_i - y_i)^2 + \lambda |w|
\]
Maximum Likelihood Estimation

• We discussed maximum likelihood estimation:
  \[ p(y | X, \omega) \]
  "likelihood"

• And how this is equivalent to minimizing negative log-likelihood:
  \[ f(\omega) = - \sum_{i=1}^{n} \log \left( p(y_i | x_i, \omega) \right) \]

• Makes connection between probabilities and loss functions:
  – Gaussian likelihood => squared loss.
  – Laplace likelihood => absolute loss.
  – Sigmoid likelihood => logistic regression.
MAP Estimation

• We discussed **MAP estimation**:

\[
p(w | X, y) \propto p(y | X, w) p(w)
\]

"posterior"  "likelihood"  "prior"

– Prior can take into account that complex models can overfit.

• Makes **connection between probabilities and regularization**:

If

\[
p(y_i | x_i, w) = \frac{1}{1 + \exp(-y_i w^T x_i)} \quad \text{and} \quad p(w_j) \propto \exp\left(-\frac{1}{2} w_j^2\right)
\]

then **MAP estimate is minimum of**

\[
\hat{f}(w) = \sum_{i=1}^n \log(1 + \exp(-y_i w^T x_i)) + \frac{\lambda}{2} ||w||^2
\]
Softmax Loss for Multi-Class Classification

• Sometimes it’s easier to define a likelihood than a loss function.
  – Softmax probability:
    \[ p(y_i = c \mid x_i, W) \propto \exp(w_c^T x_i) \]
    • We have a vector \( w_c \) for each class ‘c’, and classify by choosing largest \( w_c^T x_i \).
  – Leads to softmax loss for multi-class classification:
    \[
    f(w) = -\sum_{i=1}^{n} \log( p(y_i \mid x_i, W))
    = \sum_{i=1}^{n} -w_{y_i}^T x_i + \log(\sum_{c=1}^{k} \exp(w_c^T x_i))
    \]
  – Can define other losses based on other probabilities or probability ratios.
Latent-Factor Models

- **Latent-Factor models** approximate $x_i$ with low-dimensional $z_i$:

\[
X \sim W Z^n \sim \mathcal{N}(\mu, \Sigma)
\]

- Rows are called "latent factors"

- Used for:
  - Dimensionality reduction.
  - Outlier detection.
  - Basis for linear models.
  - Data visualization.
  - Data compression.
  - Interpreting factors.
  - Filling in missing values.

<table>
<thead>
<tr>
<th>Trait</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Openness</td>
<td>Being curious, original, intellectual, creative, and open to new ideas.</td>
</tr>
<tr>
<td>Conscientiousness</td>
<td>Being prepared, systematic, punctual, achievement-oriented, and dependable.</td>
</tr>
<tr>
<td>Extraversion</td>
<td>Being outgoing, talkative, sociable, and enjoying social situations.</td>
</tr>
<tr>
<td>Agreeableness</td>
<td>Being affable, tolerant, sensitive, trusting, kind, and warm.</td>
</tr>
<tr>
<td>Neurotic</td>
<td>Being anxious, irritable, temperament, and moody.</td>
</tr>
</tbody>
</table>
Principal Component Analysis

• Principal component analysis (PCA): LFM based on squared error.

\[ \sum_{i=1}^{n} \sum_{j=1}^{d} (w_{ij}z_{i} - x_{ij})^2 = \|ZW - X\|_F^2 \]

• With 1 factor, minimizes ‘orthogonal’ distance:

• To give unique solution:
  – Constrain factors to have norm of 1.
  – Constrain factors to have inner product of 0.
  – Fit factors sequentially.

• Found by SVD or alternating minimization.
Beyond PCA

• Non-negative matrix factorization:
  – Latent-factor model with non-negative constraints.
  – Sparsity due to non-negativity means we learn ‘parts’.

• Could use different loss functions or regularizers:
  – Robust PCA.
  – Sparse PCA.

• Collaborative filtering:
  – Use LFM to “fill in” missing values in matrix.
  – SVDfeature combines this with linear models.
Multi-Dimensional Scaling

• Multi-dimensional scaling:
  – Non-parametric visualization.
  – Find low-dimensional ‘$z_i$’ that preserve distances.

• Classic MDS and Sammon mapping are similar to PCA.

• ISOMAP uses graph to approximate geodesic distance on manifold.

• T-SNE encourages ‘repulsion’ of close points.
Neural Networks and Deep Learning

• **Neural networks** combine latent-factor and linear models.
  – Linear-linear model is degenerate, so introduce non-linearity:
    • Sigmoid or ReLU function.
  – **Backpropagation** uses chain rule to compute gradient.
• **Deep learning** considers many layers of latent factors.

  Neural network with 3 hidden layers

  \[
  \gamma_i = w^T h(W^{(3)} h(W^{(2)} h(W^{(1)} x_i)))
  \]

• A lot of tricks are needed to make deep learning work:
  – Parameter initialization
  – Setting stochastic gradient step sizes.
  – L2-regularization, early stopping, dropout.
Convolutional Neural Networks

• **Convolutional neural networks:**
  – Incorporate *convolutional* and *max-pooling* layers.

• Unprecedented performance on vision tasks.
• Lots of neat new applications:
Semi-Supervised Learning

• Semi-supervised learning considers labeled and unlabeled data.
  – Sometimes helps but in some settings it cannot.

  – Inductive SSL: use unlabeled to help supervised learning.
  – Transductive SSL: only interested in these particular unlabeled examples.

• Self-training methods alternate between labeling and fitting model.
Random Walks and Markov Chains

• We often have data organized according to a graph:
  – Could construct graph based on features and KNNs.
  – Or if you have a graph, you don’t need features.

• Models based on random walks on graphs:
  – PageRank: how often does infinitely-long random walk visit page?
  – Graph-based SSL: which label does random walk reach most often?

• Markov chains are probabilistic models of sequences:
  1. Sampling using random walk.
  2. Inference using matrix multiplication.

• Most common model of sequential data.
CPSC 340: Overview

1. **Intro to supervised learning** (using counting and distances).
   – Training vs. testing, parametric vs. non-parametric, ensemble methods.
   – Fundamental trade-off, no free lunch, universal consistency.
2. **Intro to unsupervised learning** (using counting and distances).
   – Clustering, outlier detection, association rules.
3. **Linear models and gradient descent** (for supervised learning)
   – Loss functions, change of basis, regularization, feature selection.
   – Gradient descent and stochastic gradient.
4. **Latent-factor models** (for unsupervised learning)
   – Typically using linear models and gradient descent.
5. **Neural networks** (for supervised and multi-layer latent-factor models).
6. **Markov chains**
   – Random walk models for sequences and data living on graphs.
CPSC 340 vs. CPSC 540

• **Goals of CPSC 340** this term: *practical machine learning*.
  – Make accessible by avoiding some technical details/topics/models.
  – Present most of the fundamental ideas, sometimes in simplified ways.
  – Choose models that are widely-used in practice.

• **Goals of CPSC 540** next term: *research-level machine learning*.
  – Covers complicated details/topics/models that we avoided.
  – Targeted at people with algorithms/math/stats/sciComp background.
  – Goal is to be able to understand ICML/NIPS papers at the end of course.

• Rest of this lecture:
  – What did we not cover? ⇔ What I’m planning to cover in CPSC 540.
1. Large-Scale Machine Learning

• We’ll also fill in details of topics we’ve ignored:
  – How do we convexity of general multivariate functions?
    \[ X^T P X \geq 0 \]
  – How many iterations of gradient descent do we need?
    \[ f(w^t) - f(w^*) \leq (1 - \frac{\mu}{L})^t [ f(w^0) - f(w^*) ] \]
  – How do we solve non-smooth optimization problems?
    \[ f(w) = c^T w \quad \text{with} \quad Aw \leq b \]
  – How can get sparsity in terms of ‘groups’ or ‘patterns’ of variables?
    \[ f(w) = \frac{1}{2} \| Xw - y \|^2 + \sum_{l \in B} \| w_l \|_2 \]
  – How can we apply kernels to general linear models?
    \[ f^*(z) = \sup_{w \in b} \sum_{l \in B} w_l z_l - f(w) \]
2. Density Estimation

• Methods for estimating multivariate distributions $p(x)$.
  – Abstract problem, includes most of ML as a special case.
  – But going beyond simple Gaussian and independent models.

• Classic models:
  – Mixture models.
  – Non-parametric models.
  – Markov chains.

• More latent-factor models: factor analysis, ICA, topic models.
3. Structured Prediction and Graphical Models

- Structured prediction:
  - Instead of class label ‘$y_i$’, our output is a general object.

- Conditional random fields and structured support vector machines.
- Relationship of graph to dynamic programming (treewidth).
- Variational and Markov chain Monte Carlo for inference/decoding.
- Unsupervised deep learning: Boltzmann machines and GANs.
4. Bayesian Statistics

- Key idea: treat the **model as a random variable**.
  - Now use the rules of probability to make inferences.
  - **Learning with integration** rather than differentiation.

- Can do things with Bayesian statistics that can’t otherwise be done.
  - **Bayesian model averaging**.
  - **Hierarchical** models.
  - Optimize regularization parameters and things like ‘k’.
  - Allow **infinite number of latent factors**.
  - Non-IID data.
5. Recurrent Neural Networks

• How can we add memory to deep learning?

https://www.youtube.com/watch?v=mLxsbWAYIpw
6. Online and Active Learning (Time Permitting)

- **Online** learning:
  - Training examples are streaming in over time.
  - Want to predict well in the present.
  - Not necessarily IID.

- **Active** learning:
  - Generalization of semi-supervised learning.
  - Model can choose which example to label next.
6. Causal Learning (Time Permitting)

- **Causal learning:**
  - **Observational** prediction (CPSC 340):
    - Do people who take Cold-FX have shorter colds?
  - **Causal** prediction:
    - Does taking Cold-FX cause you to have shorter colds?
  - **Counter-factual** prediction:
    - You didn’t take Cold-FX and had long cold, would taking it have made it shorter?

- Modeling the **effects of actions**.
- Predicting the **direction of causality**.
7. Reinforcement Learning (Time Permitting)

- **Reinforcement** learning puts everything together:
  - Use **observations** to build a model of the world (learning).
  - We care about performance in the present (online).
  - We have to make decisions (active).
  - Our decisions affect the world (causal).

https://www.youtube.com/watch?v=Ih8EfvOzBOY
https://www.youtube.com/watch?v=SH3bADiB7uQ
https://www.youtube.com/watch?v=nUQsRPJ1dYw
Final Slide: Data Science Job Board

• Data Science Job Board: [http://makedatasense.ca/jobs](http://makedatasense.ca/jobs)
  – Set up by students to connect employers/employees.
  – More companies looking for people than people looking for jobs.
  – Make a profile if you are looking for a job in this area.