CS 6375
Machine Learning
(Qualifying Exam Section)

Nicholas Ruozzi
University of Texas at Dallas

Slides adapted from David Sontag and Vibhav Gogate
Course Info.

- **Instructor:** Nicholas Ruozzi
  - **Office:** ECSS 3.409
  - **Office hours:** Mon. 1pm-2pm
- **TA:** ?
  - **Office hours and location?**
- **Course website:** [www.utdallas.edu/~nrr150130/cs6375/2018fa/](http://www.utdallas.edu/~nrr150130/cs6375/2018fa/)
- **Piazza (online forum):** sign-up link on eLearning
Prerequisites

- CS 5343 (data structures & algorithms)
- “Mathematical sophistication”
  - Basic probability
  - Linear algebra
    - Eigenvalues, eigenvectors, matrices, vectors, etc.
  - Multivariate calculus
    - Derivatives, gradients, convex functions, etc.
- I’ll review some concepts as we come to them, but you should brush up in areas that you aren’t as comfortable
Grading

• 5-6 problem sets (50%)
  • See collaboration policy on the web
  • Mix of theory and programming (in MATLAB or Python)
  • Available and turned in on eLearning
  • Approximately one assignment every two weeks

• Midterm Exam (20%)

• Final Exam (30%)

-subject to change-
Course Topics

• Dimensionality reduction
  • PCA
  • Matrix Factorizations
• Learning
  • Supervised, unsupervised, active, reinforcement, ...
  • Learning theory: PAC learning, VC dimension
  • SVMs & kernel methods
  • Decision trees, k-NN, logistic regression, ...
  • Parameter estimation: Bayesian methods, MAP estimation, maximum likelihood estimation, expectation maximization, ...
  • Clustering: k-means & spectral clustering
• Probabilistic models
  • Bayesian networks
  • Naïve Bayes
• Neural networks
• Statistical methods
  • Boosting, bagging, bootstrapping
  • Sampling
• Ranking & Collaborative Filtering
What is ML?
What is ML?

“A computer program is said to learn from experience $E$ with respect to some task $T$ and some performance measure $P$, if its performance on $T$, as measured by $P$, improves with experience $E$.”

- Tom Mitchell
Basic Machine Learning Paradigm

• Collect data

• Build a model using “training” data

• Use model to make predictions
Supervised Learning

• **Input:** \( (x^{(1)}, y^{(1)}), \ldots, (x^{(M)}, y^{(M)}) \)
  
  • \( x^{(m)} \) is the \( m^{th} \) data item and \( y^{(m)} \) is the \( m^{th} \) label

• **Goal:** find a function \( f \) such that \( f(x^{(m)}) \) is a “good approximation” to \( y^{(m)} \)
  
  • Can use it to predict \( y \) values for previously unseen \( x \) values
Examples of Supervised Learning

- Spam email detection
- Handwritten digit recognition
- Stock market prediction
- More?
Supervised Learning

• Hypothesis space: set of allowable functions $f: X \rightarrow Y$

• Goal: find the “best” element of the hypothesis space

  • How do we measure the quality of $f$?
Supervised Learning

• Simple linear regression

  • Input: pairs of points \((x^{(1)}, y^{(1)}), \ldots, (x^{(M)}, y^{(M)})\) with \(x^{(m)} \in \mathbb{R}\) and \(y^{(m)} \in \mathbb{R}\)

  • Hypothesis space: set of linear functions \(f(x) = ax + b\) with \(a, b \in \mathbb{R}\)

  • Error metric: squared difference between the predicted value and the actual value
Regression

Hypothesis class: linear functions \( f(x) = ax + b \)

How do we compute the error of a specific hypothesis?
Linear Regression

• For any data point, $x$, the learning algorithm predicts $f(x)$

• In typical regression applications, measure the fit using a squared loss function

$$L(f) = \frac{1}{M} \sum_{m} (f(x^{(m)}) - y^{(m)})^2$$

• Want to minimize the average loss on the training data

• The optimal linear hypothesis is then given by

$$\min_{a,b} \frac{1}{M} \sum_{m} (ax^{(m)} + b - y^{(m)})^2$$
Linear Regression

\[
\min_{a,b} \frac{1}{M} \sum_m (ax^{(m)} + b - y^{(m)})^2
\]

- How do we find the optimal $a$ and $b$?
Linear Regression

\[
\min_{a,b} \frac{1}{M} \sum_m (ax^{(m)} + b - y^{(m)})^2
\]

• How do we find the optimal \(a\) and \(b\)?
  
  • Solution 1: take derivatives and solve (there is a closed form solution!)
  
  • Solution 2: use gradient descent
Linear Regression

\[
\min_{a,b} \frac{1}{M} \sum_m \left( ax^{(m)} + b - y^{(m)} \right)^2
\]

• How do we find the optimal \( a \) and \( b \)?

  • Solution 1: take derivatives and solve (there is a closed form solution!)

  • Solution 2: use gradient descent

    • This approach is much more likely to be useful for general loss functions
Gradient Descent

Iterative method to minimize a (convex) differentiable function $f$

- Pick an initial point $x_0$
- Iterate until convergence

$$x_{t+1} = x_t - \gamma_t \nabla f(x_t)$$

where $\gamma_t$ is the $t^{th}$ step size (sometimes called learning rate)
Gradient Descent

Gradient Descent

\[
\min_{a,b} \frac{1}{M} \sum_m (ax^{(m)} + b - y^{(m)})^2
\]

• What is the gradient of this function?

• What does a gradient descent iteration look like for this simple regression problem?

(on board)
Linear Regression

• In higher dimensions, the linear regression problem is essentially the same with \( x^{(m)} \in \mathbb{R}^n \)

\[
\min_{a \in \mathbb{R}^n, b} \frac{1}{M} \sum_{m} (a^T x^{(m)} + b - y^{(m)})^2
\]

• Can still use gradient descent to minimize this

  • Not much more difficult than the \( n = 1 \) case
Gradient Descent

- Gradient descent converges under certain technical conditions on the function $f$ and the step size $\gamma_t$

  - If $f$ is convex, then any fixed point of gradient descent must correspond to a global optimum of $f$

  - In general, convergence is only guaranteed to a local optimum
Regression

- What if we enlarge the hypothesis class?
  - Quadratic functions
  - $k$-degree polynomials
- Can we always learn better with a larger hypothesis class?
Regression

- What if we enlarge the hypothesis class?
  - Quadratic functions
  - $k$-degree polynomials
- Can we always learn better with a larger hypothesis class?
Regression

• Larger hypothesis space always decreases the cost function, but this does **NOT** necessarily mean better predictive performance

  • This phenomenon is known as **overfitting**

  • Ideally, we would select the **simplest** hypothesis consistent with the observed data

• In practice, we cannot simply evaluate our learned hypothesis on the training data, we want it to perform well on unseen data (otherwise, we can just memorize the training data!)

  • Report the loss on some held out **test data** (i.e., data not used as part of the training process)
Binary Classification

- Regression operates over a continuous set of outcomes.
- Suppose that we want to learn a function $f: X \rightarrow \{0,1\}$.
- As an example:

<table>
<thead>
<tr>
<th></th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

How do we pick the hypothesis space?

How do we find the best $f$ in this space?
Binary Classification

- Regression operates over a continuous set of outcomes
- Suppose that we want to learn a function $f: X \rightarrow \{0,1\}$
- As an example:

<table>
<thead>
<tr>
<th></th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

How many functions with three binary inputs and one binary output are there?
Binary Classification

<table>
<thead>
<tr>
<th></th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>?</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>?</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>?</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>?</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

$2^8$ possible functions

$2^4$ are consistent with the observations

How do we choose the best one?

What if the observations are noisy?
Challenges in ML

• How to choose the right hypothesis space?
  • Number of factors influence this decision: difficulty of learning over the chosen space, how expressive the space is, ...

• How to evaluate the quality of our learned hypothesis?
  • Prefer “simpler” hypotheses (to prevent overfitting)
  • Want the outcome of learning to generalize to unseen data
Challenges in ML

• How do we find the best hypothesis?
  • This can be an NP-hard problem!
  • Need fast, scalable algorithms if they are to be applicable to real-world scenarios
Other Types of Learning

- **Unsupervised**
  - The training data does not include the desired output

- **Semi-supervised**
  - Some training data comes with the desired output

- **Active learning**
  - Semi-supervised learning where the algorithm can ask for the correct outputs for specifically chosen data points

- **Reinforcement learning**
  - The learner interacts with the world via allowable actions which change the state of the world and result in rewards
  - The learner attempts to maximize rewards through trial and error