Course Overview

+ Propositional Logic

Matt Gormley
Lecture 1
Oct. 22, 2018
ABOUT THIS COURSE
How to describe 606/607 to a friend

606/607 is...

a **formal** presentation of **mathematics** and **computer science**...

motivated by (carefully chosen) **real-world problems** that arise in **machine learning**...

where the **broader picture** of how those problems arise is treated **somewhat informally**.
How to describe 606/607 to a friend

606/607 is...

The class you should take to prepare for future coursework in machine learning
Machine Learning is NOT a self-driving car

Image from https://www.greencarreports.com/
Reorganization of Today’s Lecture

• 10-606 Students:
  – Don’t worry! We’re going to leave the entire next section (course overview, syllabus until the end)
  – That way, you can leave class early if you don’t want to hear it all over again

• Non-10-606 Students:
  – We’ll save the big introduction until later and jump right into some core content
  – Save your questions about course organization until later
WHAT IS MACHINE LEARNING?
Artificial Intelligence

The basic goal of AI is to develop intelligent machines.

This consists of many sub-goals:
• Perception
• Reasoning
• Control / Motion / Manipulation
• Planning
• Communication
• Creativity
• Learning
What is Machine Learning?

The goal of this course is to provide you with a toolbox:

- Machine Learning
- Statistics
- Probability
- Computer Science
- Optimization
What is ML?

- Computer Science
- Domain of Interest
- Machine Learning
- Optimization
- Statistics
- Probability
- Calculus
- Measure Theory
- Linear Algebra
What is ML?

Machine Learning

- Computer Science
- Optimization
- Statistics
- Probability
- Calculus
- Measure Theory
- Linear Algebra

Domain of Interest
Speech Recognition

1. Learning to recognize spoken words

```
“...the SPHINX system (e.g. Lee 1989) learns speaker-specific strategies for recognizing the primitive sounds (phonemes) and words from the observed speech signal...neural network methods...hidden Markov models...”
```

(Mitchell, 1997)

Source: https://www.stonetemple.com/great-knowledge-box-showdown/#VoiceStudyResults
## Robotics

### 2. Learning to drive an autonomous vehicle

<table>
<thead>
<tr>
<th>THEN</th>
<th>NOW</th>
</tr>
</thead>
<tbody>
<tr>
<td>“…the ALVINN system (Pomerleau 1989) has used its learned strategies to drive unassisted at 70 miles per hour for 90 miles on public highways among other cars…”</td>
<td><img src="waymo.com" alt="Waymo car" /></td>
</tr>
</tbody>
</table>

(Mitchell, 1997)
# Robotics

## 2. Learning to drive an autonomous vehicle

<table>
<thead>
<tr>
<th>THEN</th>
<th>NOW</th>
</tr>
</thead>
<tbody>
<tr>
<td>“…the ALVINN system (Pomerleau 1989) has used its learned strategies to drive unassisted at 70 miles per hour for 90 miles on public highways among other cars…”</td>
<td><img src="https://www.geek.com/wp-content/uploads/2016/03/uber.jpg" alt="Image of an autonomous vehicle" /></td>
</tr>
</tbody>
</table>

(Mitchell, 1997)

Games / Reasoning

3. Learning to beat the masters at board games

**THEN**

“…the world’s top computer program for backgammon, TD-GAMMON (Tesauro, 1992, 1995), learned its strategy by playing over one million practice games against itself…”

**NOW**

(Mitchell, 1997)
4. Learning to recognize images

“…The recognizer is a convolution network that can be spatially replicated. From the network output, a hidden Markov model produces word scores. The entire system is globally trained to minimize word-level errors.…”

(LeCun et al., 1995)
Learning Theory

- 5. In what cases and how well can we learn?

**Sample Complexity Results**

**Definition 0.1.** The sample complexity of a learning algorithm is the number of examples required to achieve arbitrarily small error (with respect to the optimal hypothesis) with high probability (i.e. close to 1).

**Four Cases we care about…**

| Finite $|\mathcal{H}|$ | Realizable | $N \geq \frac{1}{\epsilon^2} [\log|\mathcal{H}| + \log\left(\frac{1}{\delta}\right)]$ labeled examples are sufficient so that with probability $(1 - \delta)$ all $h \in \mathcal{H}$ with $R(h) \geq \epsilon$ have $R(h) > 0$. |
| Infinite $|\mathcal{H}|$ | Realizable | $N = O\left(\frac{1}{\epsilon^2} (\text{VC}(\mathcal{H}) + \log\left(\frac{1}{\delta}\right))\right)$ labeled examples are sufficient so that with probability $(1 - \delta)$ all $h \in \mathcal{H}$ with $R(h) \geq \epsilon$ have $R(h) > 0$. |
| Infinite $|\mathcal{H}|$ | Agnostic | $N \geq \frac{1}{\epsilon^2} \left[ \log|\mathcal{H}| + \log\left(\frac{1}{\delta}\right) \right]$ labeled examples are sufficient so that with probability $(1 - \delta)$ for all $h \in \mathcal{H}$ we have that $|R(h) - R(h)| < \epsilon$. |

**Two Types of Error**

1. **True Error** (aka. expected risk) (aka. Generalization Error)
   
   $R(h) = \mathbb{P}_{x \sim P(x)}(c^*(x) \neq h(x))$

   *Always unknown.*

2. **Test Error** (aka. empirical risk)
   
   $\hat{R}(h) = \mathbb{P}_{x \sim S}(c^*(x) \neq h(x))$

   $S = \{x^{(1)}, \ldots, x^{(N)}\}$ known, computable

1. How many examples do we need to learn?
2. How do we quantify our ability to generalize to unseen data?
3. Which algorithms are better suited to specific learning settings?
What is Machine Learning?

The goal of this course is to provide you with a toolbox:

Machine Learning
Statistics
Probability
Computer Science
Optimization

To solve all the problems above and more
Topics Covered in an ML Course
(e.g. 10-601)

• Foundations
  – Probability
  – MLE, MAP
  – Optimization

• Classifiers
  – KNN
  – Naïve Bayes
  – Logistic Regression
  – Perceptron
  – SVM

• Regression
  – Linear Regression

• Important Concepts
  – Kernels
  – Regularization and Overfitting
  – Experimental Design

• Unsupervised Learning
  – K-means / Lloyd’s method
  – PCA
  – EM / GMMs

• Neural Networks
  – Feedforward Neural Nets
  – Basic architectures
  – Backpropagation
  – CNNs

• Graphical Models
  – Bayesian Networks
  – HMMs
  – Learning and Inference

• Learning Theory
  – Statistical Estimation (covered right before midterm)
  – PAC Learning

• Other Learning Paradigms
  – Matrix Factorization
  – Reinforcement Learning
  – Information Theory
Learning Paradigms:
What data is available and when? What form of prediction?
- supervised learning
- unsupervised learning
- semi-supervised learning
- reinforcement learning
- active learning
- imitation learning
- domain adaptation
- online learning
- density estimation
- recommender systems
- feature learning
- manifold learning
- dimensionality reduction
- ensemble learning
- distant supervision
- hyperparameter optimization

Problem Formulation:
What is the structure of our output prediction?
- boolean
- categorical
- ordinal
- real
- ordering
- multiple discrete
- multiple continuous
- both discrete & cont.

Theoretical Foundations:
What principles guide learning?
- probabilistic
- information theoretic
- evolutionary search
- ML as optimization

Facets of Building ML Systems:
How to build systems that are robust, efficient, adaptive, effective?
1. Data prep
2. Model selection
3. Training (optimization / search)
4. Hyperparameter tuning on validation data
5. (Blind) Assessment on test data

Big Ideas in ML:
Which are the ideas driving development of the field?
- inductive bias
- generalization / overfitting
- bias-variance decomposition
- generative vs. discriminative
- deep nets, graphical models
- PAC learning
- distant rewards

Application Areas:
Key challenges?
NLP, Speech, Computer Vision, Robotics, Medicine, Search
WHY DO WE NEED MATH / CS FOR MACHINE LEARNING?
Why Math for ML?

To best understand ___A___ we need ___B___
Why Math for ML?

To best understand \_A\_ we need \_B\_

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Derivation of Principal Component Analysis (PCA)</td>
<td>Linear Algebra</td>
</tr>
<tr>
<td></td>
<td>• Vector spaces, Functions and Function Spaces</td>
</tr>
<tr>
<td></td>
<td>• Matrices and linear operators</td>
</tr>
<tr>
<td></td>
<td>• Matrix decomposition</td>
</tr>
</tbody>
</table>
Principle Component Analysis (PCA)

Figure from https://medium.com/@luckylwk/visualising-high-dimensional-datasets-in-python-8ef87e7915b

Note: This is just motivation – we’ll cover the math need to understand these topics later!
SVD for PCA

For any arbitrary matrix $\mathbf{A}$, SVD gives a decomposition:

$$\mathbf{A} = \mathbf{U}\Lambda\mathbf{V}^T$$  \hspace{1cm} (1)

where $\Lambda$ is a diagonal matrix, and $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices.

Suppose we obtain an SVD of our data matrix $\mathbf{X}$, so that:

$$\mathbf{X} = \mathbf{U}\Lambda\mathbf{V}^T$$  \hspace{1cm} (1)

Now consider what happens when we rewrite $\Sigma = \frac{1}{N}\mathbf{X}^T\mathbf{X}$ terms of this SVD.

$$\Sigma = \frac{1}{N}\mathbf{X}^T\mathbf{X}$$  \hspace{1cm} (2)

$$= \frac{1}{N}(\mathbf{U}\Lambda\mathbf{V}^T)^T(\mathbf{U}\Lambda\mathbf{V}^T)$$  \hspace{1cm} (3)

$$= \frac{1}{N}(\mathbf{V}\Lambda^T\mathbf{U}^T)(\mathbf{U}\Lambda\mathbf{V}^T)$$  \hspace{1cm} (4)

$$= \frac{1}{N}\mathbf{V}\Lambda^T\Lambda\mathbf{V}^T$$  \hspace{1cm} (5)

$$= \frac{1}{N}\mathbf{V}(\Lambda)^2\mathbf{V}^T$$  \hspace{1cm} (6)

Above we used the fact that $\mathbf{U}^T\mathbf{U} = \mathbf{I}$ since $\mathbf{U}$ is orthogonal by definition.

Note: This is just motivation – we’ll cover the math need to understand these topics later!
SVD for PCA

For any arbitrary matrix $A$, SVD gives a decomposition:

$$A = U\Lambda V^T$$  \hspace{1cm} (1)

where $\Lambda$ is a diagonal matrix.

Suppose we obtain $A = U\Lambda V^T$ in an SVD of $X$.

Now consider what we can do with this SVD of $X$.

We find that $(\Lambda)^2$ is a diagonal matrix whose entries are $\Lambda_{ii} = \lambda_i^2$ the squares of the eigenvalues of the SVD of $X$. Further, both $X$ and $X^TX$ share the same eigenvectors in their SVD.

Thus, we can run SVD on $X$ without ever instantiating the large $X^TX$ to obtain the necessary principal components more efficiently.

$$\Sigma = \frac{1}{N} X^T X$$ \hspace{1cm} (2)

$$= \frac{1}{N} (U\Lambda V^T)^T(U\Lambda V^T)$$ \hspace{1cm} (3)

$$= \frac{1}{N} (V\Lambda^T U^T)(U\Lambda V^T)$$ \hspace{1cm} (4)

$$= \frac{1}{N} V\Lambda^T \Lambda V^T$$ \hspace{1cm} (5)

$$= \frac{1}{N} V(\Lambda)^2 V^T$$ \hspace{1cm} (6)

Above we used the fact that $U^T U = I$ since $U$ is orthogonal by definition.

Note: This is just motivation – we’ll cover the math need to understand these topics later!
Why Math for ML?

To best understand __A__ we need __B__

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Derivation of Principal Component Analysis (PCA)</td>
<td>Linear Algebra</td>
</tr>
<tr>
<td></td>
<td>• Vector spaces, Functions and Function Spaces</td>
</tr>
<tr>
<td></td>
<td>• Matrices and linear operators</td>
</tr>
<tr>
<td></td>
<td>• Matrix decomposition</td>
</tr>
</tbody>
</table>
Why Math for ML?

To best understand **A** we need **B**

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Derivation of Principal Component Analysis (PCA)</td>
<td>Linear Algebra</td>
</tr>
<tr>
<td></td>
<td>• Vector spaces, Functions and Function Spaces</td>
</tr>
<tr>
<td></td>
<td>• Matrices and linear operators</td>
</tr>
<tr>
<td></td>
<td>• Matrix decomposition</td>
</tr>
<tr>
<td>Gradient-based Matrix Factorization and Collaborative</td>
<td>Calculus</td>
</tr>
<tr>
<td>Filtering</td>
<td>• Chain-rule; Partial Derivatives;</td>
</tr>
<tr>
<td></td>
<td>• Matrix Differentials; Second and Higher Differentials</td>
</tr>
</tbody>
</table>
Collaborative Filtering

Latent Factor Methods

- Assume that both movies and users live in some low-dimensional space describing their properties
- **Recommend** a movie based on its proximity to the user in the latent space

Note: This is just motivation – we’ll cover the math need to understand these topics later!

Figures from Koren et al. (2009)
Example: Matrix Factorization
for the Netflix Problem

(a) Example of rank-2 matrix factorization

(b) Residual matrix

Note: This is just motivation – we’ll cover the math need to understand these topics later!

Figures from Aggarwal (2016)
Matrix Factorization
(with matrices)

- **User vectors:**
  \[(W_{u*})^T \in \mathbb{R}^r\]

- **Item vectors:**
  \[H_{*i} \in \mathbb{R}^r\]

- **Rating prediction:**
  \[V_{ui} = W_{u*} H_{*i} = [WH]_{ui}\]

Figure sources:
- Figures from Koren et al. (2009)
- Figures from Gemulla et al. (2011)

Note: This is just motivation— we'll cover the math need to understand these topics later!
Matrix Factorization (with matrices)

• Stochastic Gradient Descent

require that the loss can be written as

\[ L = \sum_{(i,j) \in Z} l(V_{ij}, W_{i*}, H_{*j}) \]

Algorithm 1 SGD for Matrix Factorization

Require: A training set \( Z \), initial values \( W_0 \) and \( H_0 \)

while not converged do \{step\}

Select a training point \((i,j) \in Z\) uniformly at random.

\[ W'_{i*} \leftarrow W_{i*} - \epsilon_n N \frac{\partial}{\partial W_{i*}} l(V_{ij}, W_{i*}, H_{*j}) \]

\[ H_{*j} \leftarrow H_{*j} - \epsilon_n N \frac{\partial}{\partial H_{*j}} l(V_{ij}, W_{i*}, H_{*j}) \]

\[ W_{i*} \leftarrow W'_{i*} \]

end while

Note: This is just motivation – we’ll cover the math need to understand these topics later!
Why Math for ML?

To best understand ___A___ we need ___B___

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Derivation of Principal Component Analysis</td>
<td>Linear Algebra</td>
</tr>
<tr>
<td>(PCA)</td>
<td>• Vector spaces, Functions and Function Spaces</td>
</tr>
<tr>
<td></td>
<td>• Matrices and linear operators</td>
</tr>
<tr>
<td></td>
<td>• Matrix decomposition</td>
</tr>
<tr>
<td>Gradient-based Matrix Factorization and</td>
<td>Calculus</td>
</tr>
<tr>
<td>Collaborative Filtering</td>
<td>• Chain-rule; Partial Derivatives;</td>
</tr>
<tr>
<td></td>
<td>• Matrix Differentials; Second and Higher Differentials</td>
</tr>
</tbody>
</table>
Why Math for ML?

To best understand ___A___ we need ___B___

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Derivation of Principal Component Analysis (PCA)</td>
<td>Linear Algebra</td>
</tr>
<tr>
<td></td>
<td>• Vector spaces, Functions and Function Spaces</td>
</tr>
<tr>
<td></td>
<td>• Matrices and linear operators</td>
</tr>
<tr>
<td></td>
<td>• Matrix decomposition</td>
</tr>
<tr>
<td>Gradient-based Matrix Factorization and Collaborative Filtering</td>
<td>Calculus</td>
</tr>
<tr>
<td></td>
<td>• Chain-rule; Partial Derivatives;</td>
</tr>
<tr>
<td></td>
<td>• Matrix Differentials; Second and Higher Differentials</td>
</tr>
<tr>
<td>Probabilistic Study of Ordinal Regression</td>
<td>Probability</td>
</tr>
<tr>
<td></td>
<td>• Events</td>
</tr>
<tr>
<td></td>
<td>• Discrete/Continuous Random Variables</td>
</tr>
<tr>
<td></td>
<td>• Mean &amp; Variance; Factorization</td>
</tr>
<tr>
<td></td>
<td>• Multivariate Distributions</td>
</tr>
</tbody>
</table>
Sentiment Analysis

• **Task:** *Given* a restaurant description, *predict* how many stars the author would give for the “Overall” rating.

<table>
<thead>
<tr>
<th>Aspects</th>
<th>Ratings</th>
<th>Reviews</th>
</tr>
</thead>
<tbody>
<tr>
<td>Atmosphere</td>
<td>★★★☆☆☆</td>
<td>Heavy, uninspired food, eaten under appall of cigarette smoke. Very slow service, though not unfriendly. There are many better restaurants in Ashland. Not recommended.</td>
</tr>
<tr>
<td>Food</td>
<td>★★★☆☆☆</td>
<td>I’ll have to disagree with Ms. Kitago’s take on at least one part of the evening, I believe the chicken Tikka Marsala was slightly dry. Decent portion, but not succulent as I am accustomed to. In addition, the Gulub Jaman is served cold, anathema to this diner. I will agree with Ms. K that the mango lassi was delicious, but overall I believe it was slightly inflated.</td>
</tr>
<tr>
<td>Value</td>
<td>★★★★☆☆☆</td>
<td></td>
</tr>
<tr>
<td>Service</td>
<td>★★★☆☆☆☆</td>
<td></td>
</tr>
<tr>
<td>Overall</td>
<td>★★★★☆☆☆</td>
<td></td>
</tr>
</tbody>
</table>

Note: This is just motivation – we’ll cover the math need to understand these topics later!
Ordinal Regression

Ordinal logistic regression

Pamela Warner

Box 1: Glossary of statistical terms used in this article

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary variable</td>
<td>This is a categorical variable with only two possible values (e.g. case or non-case). Also termed dichotomous or binomial.</td>
</tr>
<tr>
<td>Categorical variable</td>
<td>Such a variable has a limited number of distinct values, which might be non-quantitatively descriptive (e.g. hair colour).</td>
</tr>
<tr>
<td>Cumulative distribution</td>
<td>For each value of an ordinal or continuous variable, the cumulative frequency presents the accumulated number of occurrences for that or any 'lower' value (rather than indicating simply the frequency of occurrence of that value alone, as in an ordinary frequency distribution). The cumulative frequency for the last (highest) value must therefore encompass the entire sample. Cumulative frequency is often reported not as counts but as a percentage of the total sample, in which case the last value must have a cumulative frequency of 100%.</td>
</tr>
<tr>
<td>Degrees of freedom</td>
<td>This can be thought of as the modelling capacity (or independent elements of information) in the dataset.</td>
</tr>
<tr>
<td>Logistic regression</td>
<td>This is a method for analysis of the occurrence or not of a particular response value, in relation to potential explanatory variables. What is modelled for each combination of explanatory variables is the logarithm of the odds of that response value (which is termed a logistic transformation). Each association in the model is summarised/estimated in terms of an odds ratio (OR).</td>
</tr>
<tr>
<td>Multinomial variable</td>
<td>This is a categorical variable with more than two possible values. Also termed polychotomous.</td>
</tr>
<tr>
<td>Odds (of a specified response)</td>
<td>This is the number of occurrences of that response value divided by the number without that value (e.g. cases divided by non-cases).</td>
</tr>
<tr>
<td>Odds ratio (OR) for a specified response</td>
<td>For a binary explanatory variable, the OR is calculated as the odds of the specified response in those with the explanatory feature, divided by the odds of that response in those without the explanatory feature. If there is truly no association then the two odds should be approximately equal and the OR approximately 1, which is therefore the value for a ‘null association’.</td>
</tr>
<tr>
<td>Ordinal variable</td>
<td>This is special semi-quantitative type of categorical variable where the values are conceptually ordered, such as degree of pain (e.g. none, mild, moderate, severe) or effectiveness of contraceptive method used (e.g. none, moderate, high).</td>
</tr>
<tr>
<td>Parameter</td>
<td>This is a component in the model, which needs to be estimated from the data, and doing so uses up available degrees of freedom. [The number of parameters needed for a multinomial regression model is a multiple of the number needed for a binary logistic regression model.]</td>
</tr>
<tr>
<td>Power</td>
<td>This term is used here, loosely, as the probability of detecting from the study data what is in fact the real situation.</td>
</tr>
<tr>
<td>Reference category</td>
<td>The category within a categorical explanatory variable that is chosen as the comparator for calculation of ORs (i.e. denominator odds).</td>
</tr>
<tr>
<td>Reference value</td>
<td>This is the value within a response variable that is used as a comparator reference value in multinomial regression because in binary logistic regression there are only two response values, and a reference value can be assumed to be the only other possible response value.</td>
</tr>
<tr>
<td>Response variable</td>
<td>The outcome or dependent variable that is to be modelled/tested.</td>
</tr>
</tbody>
</table>

Note: This is just motivation – we’ll cover the math need to understand these topics later!
Ordinal Regression

**Ordinal logistic regression**

If the response variable $Y$ is ordinal, the categories can be ordered in a natural way such as ‘health status good/moderate/bad’. The polytomous logistic regression model can be applied but does not make use of the information about the ordering. One way to take account of the ordering is the use of cumulative probabilities, cumulative odds and cumulative logits. Considering $k+1$ ordered categories, these quantities are defined by

$$P(Y \leq i) = p_1 + \ldots + p_i$$

$$\text{odds} (Y \leq i) = \frac{P(Y \leq i)}{1 - P(Y \leq i)} = \frac{p_1 + \ldots + p_i}{p_{i+1} + \ldots + p_{k+1}}$$

$$\text{logit} (Y \leq i) = \ln \left( \frac{P(Y \leq i)}{1 - P(Y \leq i)} \right), \quad i=1,\ldots,k$$

The cumulative logistic model for ordinal response data is given by

$$\text{logit}(Y \leq i) = \alpha_i + \beta_1 X_1 + \ldots + \beta_m X_m, \quad i=1,\ldots,k$$

Like the polytomous logistic regression model, we have $k$ model equations and one logistic coefficient $\beta_q$ for each category/covariate combination. Hence, the general cumulative logistic regression model contains a large number of parameters. However, in some cases a more parsimonious model is possible. If the logistic coefficients do not depend on $i$, we have only one common parameter $\beta$ for each covariate. It follows that the cumulative odds are given by

$$\text{odds}(Y \leq i) = \exp(\alpha_i) \exp(\beta_1 X_1 + \ldots + \beta_m X_m), \quad i=1,\ldots,k$$

which means that the $k$ odds for each cut-off category $i$ differ only with regard to the intercepts $\alpha_i$; in other words, the odds are proportional. Hence, McCullagh used the term proportional odds model. The relatively stringent proportional odds assumption may be especially valid in cases where the ordinal response $Y$ is related to an underlying latent continuous variable.

**Polytomous logistic regression**

If the response variable $Y$ is discrete with more than two categories, for example $Y = \text{marital status}$ defined in the 3 categories ‘married, ‘divorced, separated or widowed’ and ‘single’, then the standard binary logistic regression model is not applicable. One possible way to handle such situations is to split the categorical response $Y$ in several ways, for example $Y_1 = \text{married yes/no}$, $Y_2 = \text{single yes/no}$, and to apply binary logistic regression to each dichotomous variable. However, this will result in several different analyses for only one categorical response. A more structured approach is to formulate one model for the categorical response by means of so-called generalised logits. Suppose that $Y$ has $k+1$ categories and the probability for category $i$ is given by $P(Y = i) = p_i$, for $i=1,\ldots,k+1$. Then the $k$ generalised logits are defined by

$$\text{logit}(Y = i) = \ln \left( \frac{p_i}{1 - (p_1 + \ldots + p_i)} \right) = \ln \left( \frac{p_i}{p_{i+1}} \right), \quad i=1,\ldots,k$$

This means that the generalised logits relate the probabilities $p_i$ for the categories $i=1,\ldots,k$ to the reference category $k+1$.

For $m$ covariates the general polytomous logistic regression model becomes

$$\text{logit} (Y = i) = \alpha_i + \beta_1 X_1 + \ldots + \beta_m X_m, \quad i=1,\ldots,k$$

Note that the polytomous logistic model is given by $k$ equations if $Y$ has $k+1$ categories and that we have one logistic coefficient $\beta_q$ for each category/covariate combination. Hence, it is not possible to summarise the effect of a covariate on the response $Y$ by a single measure such as one odds ratio. Although the polytomous model offers the advantage of simplifying testing the effect of each covariate on the response $Y$ by a single measure, polytomous models can be cumbersome and is difficult for physicians and patients to understand. Applications of this class of models, see DeMaris.

Note: This is just motivation – we’ll cover the maths need to understand these topics later!
### Why Math for ML?

To best understand ____A____ we need ____B____

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Derivation of Principal Component Analysis (PCA)</td>
<td>Linear Algebra</td>
</tr>
<tr>
<td></td>
<td>• Vector spaces, Functions and Function Spaces</td>
</tr>
<tr>
<td></td>
<td>• Matrices and linear operators</td>
</tr>
<tr>
<td></td>
<td>• Matrix decomposition</td>
</tr>
<tr>
<td>Gradient-based Matrix Factorization and Collaborative Filtering</td>
<td>Calculus</td>
</tr>
<tr>
<td></td>
<td>• Chain-rule; Partial Derivatives;</td>
</tr>
<tr>
<td></td>
<td>• Matrix Differentials; Second and Higher Differentials</td>
</tr>
<tr>
<td>Probabilistic Study of Ordinal Regression</td>
<td>Probability</td>
</tr>
<tr>
<td></td>
<td>• Events</td>
</tr>
<tr>
<td></td>
<td>• Discrete/Continuous Random Variables</td>
</tr>
<tr>
<td></td>
<td>• Mean &amp; Variance; Factorization</td>
</tr>
<tr>
<td></td>
<td>• Multivariate Distributions</td>
</tr>
</tbody>
</table>

The core content for this course is the **mathematics** (Column B), but you will apply what you learn to **real problems in machine learning** (Column A).
Why Computer Science for ML?

To best understand __A__ we need __B__

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Why Computer Science for ML?

To best understand __A__ we need __B__

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
</table>
| Analysis of Exact Inference in Graphical Models | Computation 
  •  Computational Complexity 
  •  Recursion; Dynamic Programming 
  •  Data Structures for ML Algorithms |
Factor Graph Notation

- **Variables:**
  \[ \mathcal{X} = \{X_1, \ldots, X_i, \ldots, X_n\} \]

- **Factors:**
  \[ \psi_\alpha, \psi_\beta, \psi_\gamma, \ldots \]
  where \( \alpha, \beta, \gamma, \ldots \subseteq \{1, \ldots n\} \)

**Joint Distribution**

\[ p(\mathbf{x}) = \frac{1}{Z} \prod_{\alpha} \psi_\alpha(\mathbf{x}_\alpha) \]

*Note: This is just motivation – we’ll cover the math need to understand these topics later!*
Factors are Tensors

- **Factors:**

  \[ \psi_\alpha, \psi_\beta, \psi_\gamma, \ldots \]
Inference

Given a factor graph, two common tasks ...

– Compute the most likely joint assignment,
  \[ x^* = \arg\max_x p(X=x) \]

  \[
  p(X_i=x_i) = \text{sum of } p(X=x) \text{ over joint assignments with } X_i = x_i
  \]

Both consider all joint assignments.
Both are NP-Hard in general.

So, we turn to approximations.

Note: This is just motivation – we’ll cover the math need to understand these topics later!
Suppose we took many samples from the distribution over taggings:

\[ p(x) = \frac{1}{Z} \prod_{\alpha} \psi_\alpha(x_{\alpha}) \]
The marginal $p(X_i = x_i)$ gives the probability that variable $X_i$ takes value $x_i$ in a random sample.
Marginals by Sampling on Factor Graph

Estimate the marginals as:

Sample 1: \( \frac{n}{v} \frac{4}{6} \frac{2}{6} \)
Sample 2: \( \frac{n}{v} \frac{3}{6} \frac{3}{6} \)
Sample 3: \( \frac{p}{v} \frac{4}{6} \frac{2}{6} \)
Sample 4: \( d \frac{6}{6} \)
Sample 5: \( n \frac{6}{6} \)
Sample 6: \( n \frac{6}{6} \)

Note: This is just motivation – we’ll cover the math need to understand these topics later!
Why Computer Science for ML?

To best understand **A** we need **B**

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis of Exact Inference in Graphical Models</td>
<td>Computation</td>
</tr>
<tr>
<td></td>
<td>• Computational Complexity</td>
</tr>
<tr>
<td></td>
<td>• Recursion; Dynamic Programming</td>
</tr>
<tr>
<td></td>
<td>• Data Structures for ML Algorithms</td>
</tr>
<tr>
<td>Implementation Design of a Deep Learning Library</td>
<td>Programming &amp; Efficiency</td>
</tr>
<tr>
<td></td>
<td>• Debugging for Machine Learning</td>
</tr>
<tr>
<td></td>
<td>• Efficient Implementation / Profiling ML Algorithms</td>
</tr>
</tbody>
</table>
Finite Difference Method

The centered finite difference approximation is:

$$\frac{\partial}{\partial \theta_i} J(\theta) \approx \frac{(J(\theta + \epsilon \cdot d_i) - J(\theta - \epsilon \cdot d_i))}{2\epsilon}$$

(1)

where $d_i$ is a 1-hot vector consisting of all zeros except for the $i$th entry of $d_i$, which has value 1.

Notes:

- Suffers from issues of floating point precision, in practice
- Typically only appropriate to use on small examples with an appropriately chosen epsilon

Note: This is just motivation – we’ll cover the math need to understand these topics later!
Chain Rule Quiz #1:
Suppose x = 2 and z = 3, what are dy/dx and dy/dz for the function below?

\[
y = \exp(xz) + \frac{xz}{\log(x)} + \frac{\sin(\log(x))}{\exp(xz)}
\]

Finite Difference Solution:

Note: This is just motivation – we’ll cover the math need to understand these topics later!
Automatic Differentiation – Reverse Mode (aka. Backpropagation)

Forward Computation
1. Write an **algorithm** for evaluating the function \( y = f(x) \). The algorithm defines a **directed acyclic graph**, where each variable is a node (i.e. the “computation graph”)
2. Visit each node in **topological order**.
   For variable \( u_i \) with inputs \( v_1, \ldots, v_N \)
   a. Compute \( u_i = g_i(v_1, \ldots, v_N) \)
   b. Store the result at the node

Backward Computation
1. **Initialize** all partial derivatives \( \frac{dy}{du_j} \) to 0 and \( \frac{dy}{dy} = 1 \).
2. Visit each node in **reverse topological order**.
   For variable \( u_i = g_i(v_1, \ldots, v_N) \)
   a. We already know \( \frac{dy}{du_i} \)
   b. Increment \( \frac{dy}{dv_j} \) by \( (\frac{dy}{du_i})(\frac{du_i}{dv_j}) \)
      (Choice of algorithm ensures computing \( \frac{du_i}{dv_j} \) is easy)

Return partial derivatives \( \frac{dy}{du_i} \) for all variables

Note: This is just motivation – we’ll cover the math need to understand these topics later!
## Why Computer Science for ML?

To best understand **A** we need **B**

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis of Exact Inference in Graphical Models</td>
<td>Computation</td>
</tr>
<tr>
<td></td>
<td>• Computational Complexity</td>
</tr>
<tr>
<td></td>
<td>• Recursion; Dynamic Programming</td>
</tr>
<tr>
<td></td>
<td>• Data Structures for ML Algorithms</td>
</tr>
<tr>
<td>Implementation Design of a Deep Learning Library</td>
<td>Programming &amp; Efficiency</td>
</tr>
<tr>
<td></td>
<td>• Debugging for Machine Learning</td>
</tr>
<tr>
<td></td>
<td>• Efficient Implementation / Profiling ML Algorithms</td>
</tr>
<tr>
<td>Optimization for Support Vector Machines (SVMs)</td>
<td>Optimization</td>
</tr>
<tr>
<td></td>
<td>• Unconstrained Optimization</td>
</tr>
<tr>
<td></td>
<td>• Preconditioning</td>
</tr>
<tr>
<td></td>
<td>• Constrained Optimization</td>
</tr>
</tbody>
</table>
Support Vector Machines (SVMs)

Hard-margin SVM (Primal)

\[
\min_{w, b} \frac{1}{2} \|w\|^2 \\
\text{s.t. } y^{(i)}(w^T x^{(i)} + b) \geq 1, \quad \forall i = 1, \ldots, N
\]

Hard-margin SVM (Lagrangian Dual)

\[
\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y^{(i)} y^{(j)} (x^{(i)} \cdot x^{(j)}) \\
\text{s.t. } \alpha_i \geq 0, \quad \forall i = 1, \ldots, N \\
\sum_{i=1}^{N} \alpha_i y^{(i)} = 0
\]

• Instead of minimizing the primal, we can maximize the dual problem

• For the SVM, these two problems give the same answer (i.e. the minimum of one is the maximum of the other)

• **Definition:** **support vectors** are those points \( x^{(i)} \) for which \( \alpha^{(i)} \neq 0 \)

Note: This is just motivation – we’ll cover the math need to understand these topics later!
SVM QP

Note: This is just motivation – we’ll cover the math need to understand these topics later!
Note: This is just motivation—we’ll cover the math need to understand these topics later!
SVM QP

Note: This is just motivation – we'll cover the math need to understand these topics later!
SVM QP

Note: This is just motivation – we’ll cover the math need to understand these topics later!
SVM QP

Note: This is just motivation – we’ll cover the math need to understand these topics later!
SVM QP

Note: This is just motivation – we’ll cover the math need to understand these topics later!
Why Computer Science for ML?

To best understand **A** we need **B**

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis of Exact Inference in Graphical Models</td>
<td>Computation</td>
</tr>
<tr>
<td></td>
<td>• Computational Complexity</td>
</tr>
<tr>
<td></td>
<td>• Recursion; Dynamic Programming</td>
</tr>
<tr>
<td></td>
<td>• Data Structures for ML Algorithms</td>
</tr>
<tr>
<td>Implementation</td>
<td>Programming &amp; Efficiency</td>
</tr>
<tr>
<td>Design of a Deep Learning Library</td>
<td>• Debugging for Machine Learning</td>
</tr>
<tr>
<td></td>
<td>• Efficient Implementation / Profiling ML Algorithms</td>
</tr>
<tr>
<td>Optimization for Support Vector Machines (SVMs)</td>
<td>Optimization</td>
</tr>
<tr>
<td></td>
<td>• Unconstrained Optimization</td>
</tr>
<tr>
<td></td>
<td>• Preconditioning</td>
</tr>
<tr>
<td></td>
<td>• Constrained Optimization</td>
</tr>
</tbody>
</table>

The core content for this course is the **computer science** (Column B), but you will apply what you learn to **real problems in machine learning** (Column A)
# Why Computer Science for ML?

To best understand ___A___ we need ___B___

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis of Exact Inference in Graphical Models</td>
<td>Computation</td>
</tr>
<tr>
<td></td>
<td>• Recursion</td>
</tr>
<tr>
<td></td>
<td>• Dynamic Programming</td>
</tr>
<tr>
<td>Implementation Design of a Deep Learning Library</td>
<td>Programming</td>
</tr>
<tr>
<td></td>
<td>• Debugging</td>
</tr>
<tr>
<td></td>
<td>• Efficiency</td>
</tr>
<tr>
<td>Optimization for Support Vector Machines (SVMs)</td>
<td>Optimization</td>
</tr>
<tr>
<td></td>
<td>• Unconstrained</td>
</tr>
<tr>
<td></td>
<td>• Constrained</td>
</tr>
<tr>
<td></td>
<td>• Preconditioning</td>
</tr>
</tbody>
</table>

**Note:**
You may want to take the 10-607 schedule with a grain of salt given Matt’s propensity in 10-606 for swapping in new applications as the semester progressed.

The core content for this course is the **computer science** (Column B), but you will apply what you learn to **real problems in machine learning** (Column A)
SYLLABUS HIGHLIGHTS
Syllabus Highlights

The syllabus is located on the course webpage:

http://www.cs.cmu.edu/~mgormley/courses/606-607-f18

The course policies are required reading.
606/607 Syllabus Highlights

- **Grading:** 55% homework, 10% in-class quizzes, 30% final exam, 5% participation
- **Final Exam:**
  - 606: Mini-I final exam week, date TBD
  - 607: Mini-II final exam week, date TBD
- **In-Class Quizzes:** always announced ahead of time
- **Homework:** 4 assignments with written / programming portions
  - 2 grace days for the unexpected
  - Late submissions: 80% day 1, 60% day 2, 40% day 3, 20% day 4
  - No submissions accepted after 4 days w/o extension
  - Extension requests: see syllabus
- **Recitations:** Fridays, same time/place as lecture (optional, interactive sessions)
- **Readings:** required, online, recommended for after lecture
- **Technologies:** Piazza (discussion), Gradescope (homework), Canvas (gradebook only)
- **Academic Integrity:**
  - Collaboration encouraged, but must be documented
  - Solutions must always be written independently
  - No re-use of found code / past assignments
  - Severe penalties (i.e. failure)
- **Office Hours:** posted on Google Calendar on “People” page
606/607 Syllabus Highlights

• **Grading**: 55% homework, 10% in-class quizzes, 30% final exam, 5% participation

• **Final Exam**:
  – 606: Mini-I final exam week, date TBD
  – 607: Mini-II final exam week, date TBD

• **In-Class Quizzes**: always announced ahead of time

• **Homework**: 4 assignments with written / programming portions
  – 2 grace days for the unexpected
  – Late submissions: 80% day 1, 60% day 2, 40% day 3, 20% day 4
  – No submissions accepted after 4 days w/o extension
  – Extension requests: see syllabus

• **Recitations**: Fridays, same time/place as lecture (optional, interactive sessions)

• **Readings**: required, online, recommended for after lecture

• **Technologies**: Piazza (discussion), Gradescope (homework), Canvas (gradebook only)

• **Academic Integrity**:
  – Collaboration encouraged, but must be documented
  – Solutions must always be written independently
  – No re-use of found code / past assignments
  – Severe penalties (i.e. failure)

• **Office Hours**: posted on Google Calendar on “People” page
Lectures

• You should ask lots of questions
  – Interrupting (by raising a hand) to ask your question is strongly encouraged
  – Asking questions later on Piazza is also great

• When I ask a question...
  – I want you to answer
  – Even if you don’t answer, think it through as though I’m about to call on you

• Interaction improves learning (both in-class and at my office hours)
Expected Background

10-606 (Math Background 4 ML)
You should be familiar with some of the following...

• Calculus:
  – can take scalar derivatives
  – can solve scalar integrals
• Linear Algebra:
  – know basic vector operations
  – seen matrix multiplication
• Probability:
  – seen the basics: conditioning, Bayes Rule, etc.
• Programming:
  – know some Python
    OR
  have sufficient programming background to pick up the basics of Python

But we’ll offer practice to make sure you can catch up on your weaker areas

10-607 (CS Background 4 ML)
You should...

• be comfortable with all the topics listed for 10-606
• ideally, have the mathematical maturity of someone who completed 10-606 because it will aide in understanding the motivating examples from machine learning

That said, the content of 10-607 is designed stand alone
LOGIC
Propositional Logic

Chalkboard

– Form of arguments
– Components of propositional logic
– Two-column proofs
– *modus ponens*
– Inference rules
– Lemmas
Exercise: Inference Rules

- **modus ponens:** from premises \( \phi \) and \( \phi \rightarrow \psi \), conclude \( \psi \).
- \( \land \) introduction: if we separately prove \( \phi \) and \( \psi \), then that constitutes a proof of \( \phi \land \psi \).
- \( \land \) elimination: from \( \phi \land \psi \) we can conclude either of \( \phi \) and \( \psi \) separately.
- \( \lor \) introduction: from \( \phi \) we can conclude \( \phi \lor \psi \) for any \( \psi \).
- \( \lor \) elimination (also called proof by cases): if we know \( \phi \lor \psi \) (the cases) and we have both \( \phi \rightarrow \chi \) and \( \psi \rightarrow \chi \) (the case-specific proofs), then we can conclude \( \chi \).
- \( \top \) introduction: we can conclude \( \top \) from no assumptions.
- \( \bot \) elimination: from \( \bot \) we can conclude an arbitrary formula \( \phi \). (This rule is sometimes called ex falso or ex falso quodlibet, from the Latin for "from falsehood, anything.") This rule can be counterintuitive, but one way to think about it is this: we should never be able to prove \( \bot \), so there’s no danger in letting ourselves prove an arbitrary formula given \( \bot \).
- Associativity: both \( \land \) and \( \lor \) are associative; it doesn’t matter how we parenthesize an expression like \( a \land (b \land c) \). (So in fact we often just leave the parentheses out.)
- Distributivity: \( \land \) and \( \lor \) distribute over one another; for example, \( a \land (b \lor c) \) is equivalent to \( (a \land b) \lor (a \land c) \).
- Commutativity: both \( \land \) and \( \lor \) are commutative (symmetric in the order of their arguments), so we can re-order their arguments however we please. For example, \( b \lor c \land a \) is equivalent to \( a \lor b \land c \).

Use the above inference rules to prove

\[ (a \land b) \rightarrow (b \land a) \]

Write your proof in two-column format: i.e., give an explicit justification for each statement based on previous statements.

Reminder: use only the above rules, even if you’ve learned other useful rules in previous courses.
Exercise: Inference Rules

- **modus ponens**: from premises \( \phi \) and \( \phi \rightarrow \psi \), conclude \( \psi \).
- \( \wedge \) introduction: if we separately prove \( \phi \) and \( \psi \), then that constitutes a proof of \( \phi \wedge \psi \).
- \( \wedge \) elimination: from \( \phi \wedge \psi \) we can conclude either of \( \phi \) and \( \psi \) separately.
- \( \vee \) introduction: from \( \phi \) we can conclude \( \phi \lor \psi \) for any \( \psi \).
- \( \vee \) elimination (also called proof by cases): if we know \( \phi \lor \psi \) (the cases) and we have both \( \phi \rightarrow \chi \) and \( \psi \rightarrow \chi \) (the case-specific proofs), then we can conclude \( \chi \).
- \( T \) introduction: we can conclude \( T \) from no assumptions.
- \( F \) elimination: from \( F \) we can conclude an arbitrary formula \( \phi \). (This rule is sometimes called \text{ex falso or ex falso quodlibet}, from the Latin for "from falsehood, anything.") This rule can be counterintuitive, but one way to think about it is this: we should never be able to prove \( F \), so there's no danger in letting ourselves prove an arbitrary formula given \( F \).

Use the above inference rules to prove \[ (a \wedge b) \rightarrow (b \wedge a) \] .

Write your proof in two-column format: i.e., give an explicit justification for each statement based on previous statements.

Reminder: use only the above rules, even if you've learned other useful rules in previous courses.

Exercise, version 2: prove the same statement without using the inference rule for commutativity.
Classical Logic

Chalkboard

– Negation and constructive logic
– Law of the extended middle
– DeMorgan’s laws
– Double negation elimination
– Contraposition
– Resolution
– Scoping rules
Exercise: Mini-Sudoku

In mini sudoku, the digits 1..4 must appear exactly once in each row, column, and bold-edged 2*2 box of the grid. In the grid below, we've been given five fixed digits (e.g., the 3 in the upper right corner). The squares labeled a, b, c, d are currently blank, and we'd like to figure out how to fill them in:

```
  1   2
  3   a
  4   b
```

For example, we know that square d can't contain the digit 2, because there's already a 2 directly above it in the same column.

Fill in the squares a, b, c, d. (Note: no guessing is required.)

Use the rules of propositional logic to write down the constraints that squares a, b, c, d must satisfy. For example, you should write that the digit 1 must appear exactly once in the squares a, b, c, d. (It may take several logical formulas to implement this constraint.) For another example, you should write that the digit 2 can't appear in squares b or d (because of the 2 above them in the same column).

Prove that the solution you gave above is correct, using your formulation of the constraints together with the rules of propositional logic.
Proof Techniques

Chalkboard

– Proof by Construction
– Proof by Cases
– Proof by Contradiction
– Proof by Induction
– Proof by Contraposition