CPSC 340:
Machine Learning and Data Mining

Course Review/Preview

Fall 2015

Admin

- Assignment 6 due now.
- We will have office hours as usual next week.
- Final exam details:
  - December 15: 8:30-11 (WESB 100).
  - 4 pages of cheat sheet allowed.
  - 9 questions.
  - Practice questions and list of topics posted.

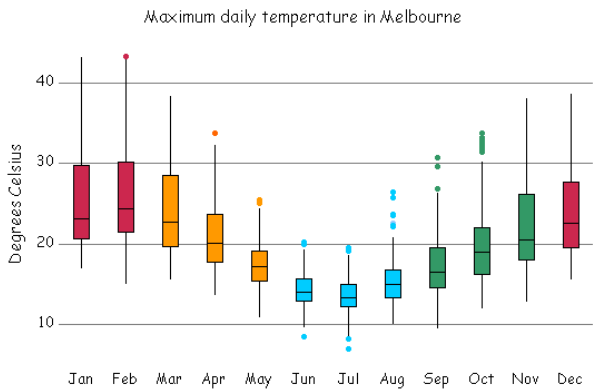
Machine Learning and Data Mining

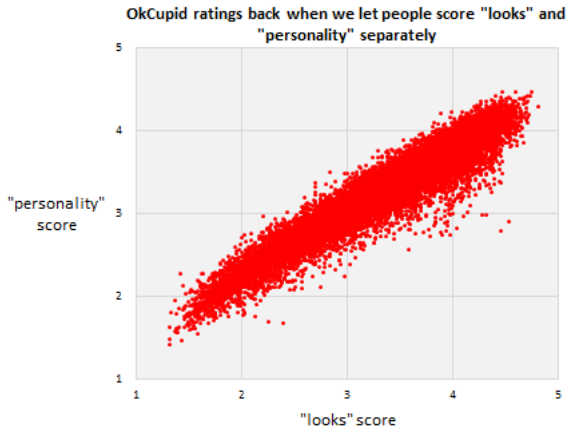
- The age of “big data” is upon us.
- Data mining and machine learning are key tools to analyze big data.
- Very similar to statistics, but more emphasis on:
  1. Computation
  2. Test error.
  3. Non-asymptotic performance.
  4. Models that work across domains.
- Enormous and growing number of applications.
- The field is growing very fast:
  - ~2500 attendees at NIPS last year, ~4000 this year? (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.

< 20	>= 20, < 25	>= 25
0	1	0
0	1	0
0	1	0
0	0	1





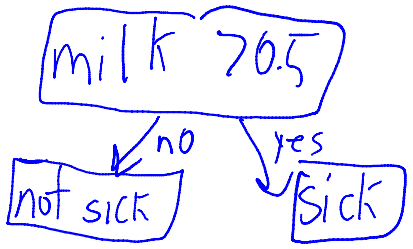
- Discussed continuous/discrete features, feature transformations.
- Discussed summary statistics like mean, quantiles, variance.
- Discussed data visualizations like boxplots and scatterplots.

Supervised Learning and Decision Trees

- Supervised learning builds model to map from features to labels.
  - Most successful machine learning method.

Egg	Milk	Fish	Wheat	Shellfish	Peanuts	…		Sick?
0	0.7	0	0.3	0	0		→	1
0.3	0.7	0	0.6	0	0.01		→	1
0	0	0	0.8	0	0		→	0

- 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:

$$p(y_i = \text{'spam'} \mid x_i) = p(x_i \mid y_i = \text{'spam'})\, p(y_i = \text{'spam'})$$

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

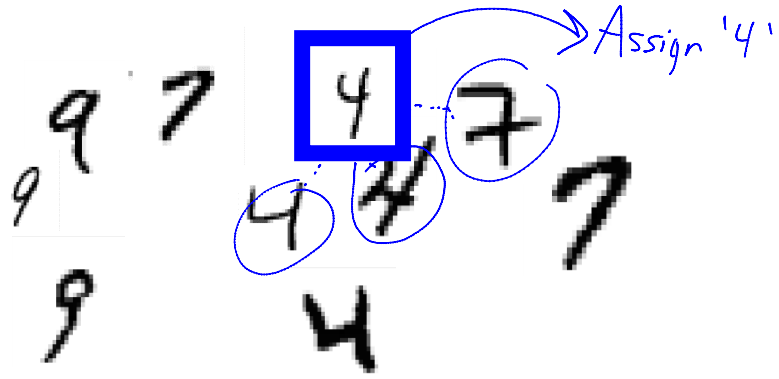
$$p(x_i \mid y_i) = \prod_{j=1}^{d} p(x_{ij} \mid y_i)$$

- Decision theory: predictions when errors have different costs.

Cost of false negative >> cost of false positive.

Parametric and Non-Parametric Models

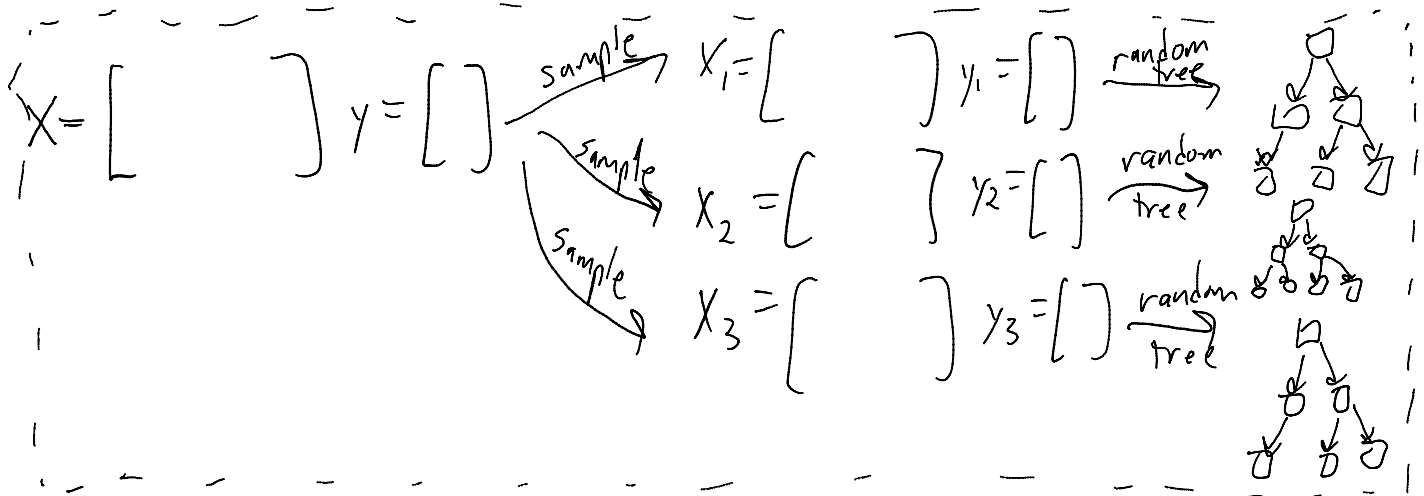
- Parametric model size does not depend on number of objects ‘n’.
- Non-parametric model size depends on ‘n’.
- K-Nearest Neighbours:
  - Non-parametric model that uses label of closest x_i in training data.
  - Accurate but slow at test time.

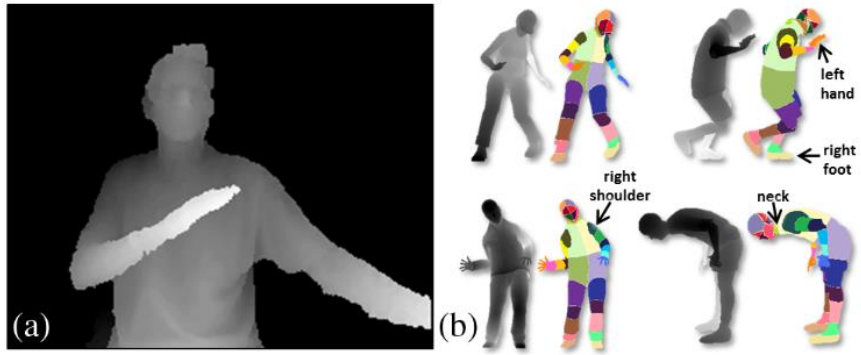


- Curse of dimensionality:
  - Problem with distances in high dimensions.
- Universally consistent methods:
  - achieve lowest possible test error as ‘n’ goes to infinity.

Ensemble Methods and Random Forests

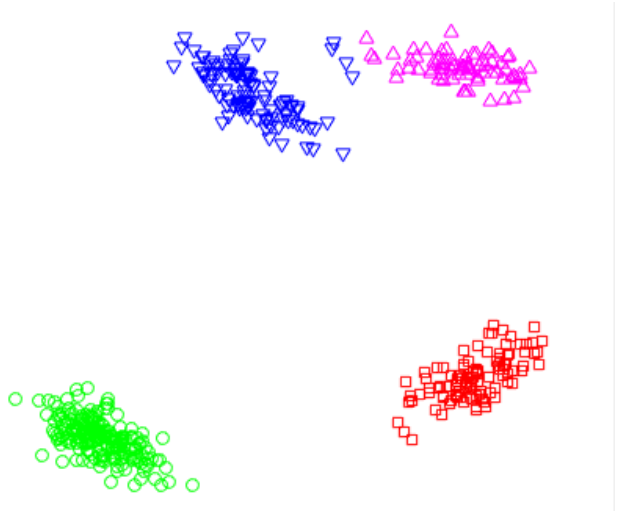
- Ensemble methods are classifiers that have classifiers as input:
  - Boosting: improve training error of simple classifiers.
  - Averaging: improve testing error of complex classifiers.
- Random forests:
  - Ensemble method that averages random trees fit on bootstrap samples.
  - Fast and accurate.





Clustering and K-Means

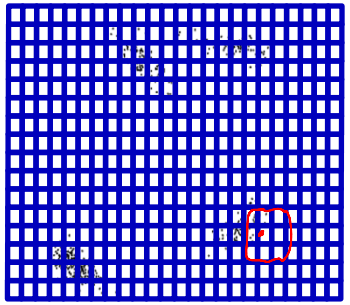
- Unsupervised learning considers features X without labels.
- Clustering is task of grouping similar objects.



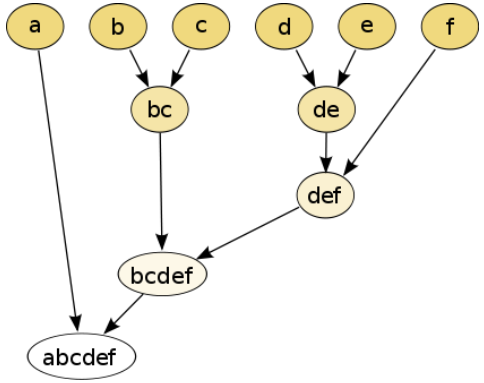
- K-means is classic clustering method:
  - Represent each cluster by its mean value.
  - Learning alternates between updating means and assigning to clusters.
  - Sensitive to initialization, but some guarantees with k-means++.

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.

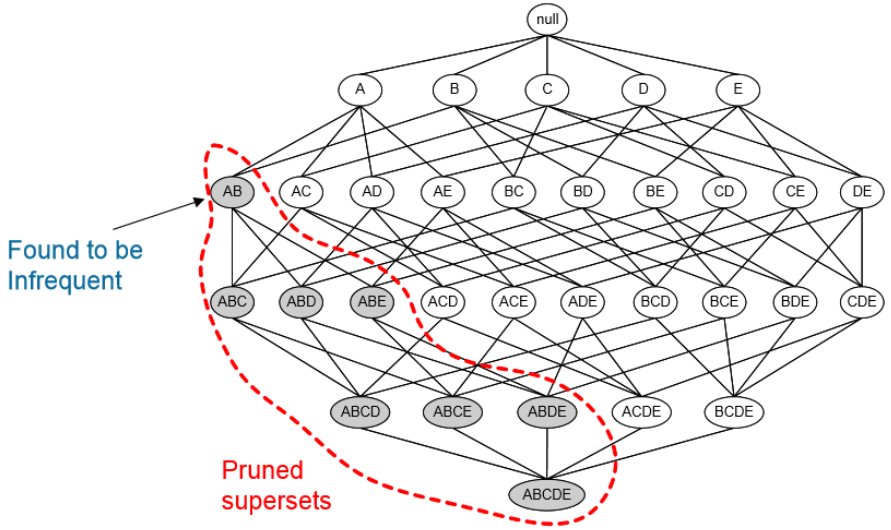


- Ensemble clustering combines clusterings.
  - But need to account for label switching problem.
- Hierarchical clustering groups objects at multiple levels.



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), and confidence, P(T | S).
- A priori algorithm finds all rules with high support/confidence.
  - Probabilistic inequalities reduce search space.

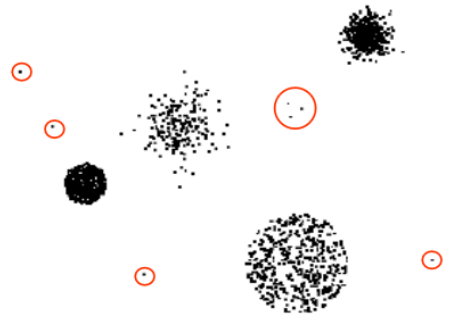


- Amazon’s item-to-item recommendation:
  - Compute similarity of ‘user vectors’ for items.

$$Cos(X_i, X_j) \simeq \frac{X_i^T X_j}{||X_i|| \cdot ||X_j||}$$

Outlier Detection

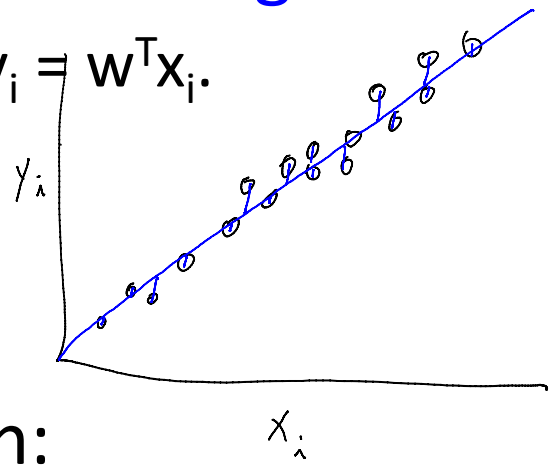
- Outlier detection is task of finding significantly different objects.
  - Global outliers are different from all other objects.
  - Local outliers fall in normal range, but are different from neighbours.



- 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 ratio tests if point is abnormally far form neighbours.

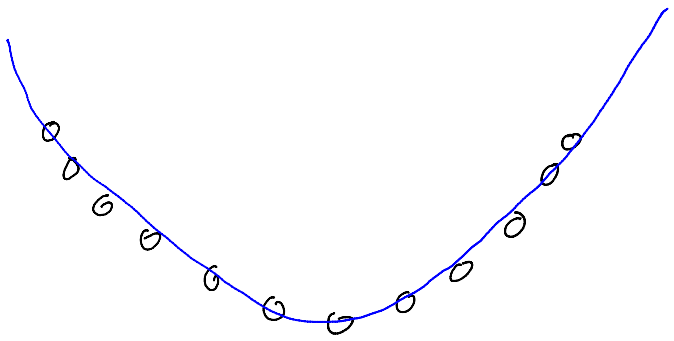
Linear Regression and Least Squares

- 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:

$$\underset{w \in \mathbb{R}^d}{\operatorname{argmin}} \frac{1}{2} \sum_{i=1}^{n} (y_i - w^T x_i)^2$$



  - Has a closed-form solution.
  - Non-zero y-intercept (bias) by adding a feature $x_{ij} = 1$.
  - Model non-linear effects by change of basis:

$$y_i = w_0(1) + w_1 x_i + w_2 x_i^2 + w_3 x_i^3$$

Regularization, Robust Regression, Gradient Descent

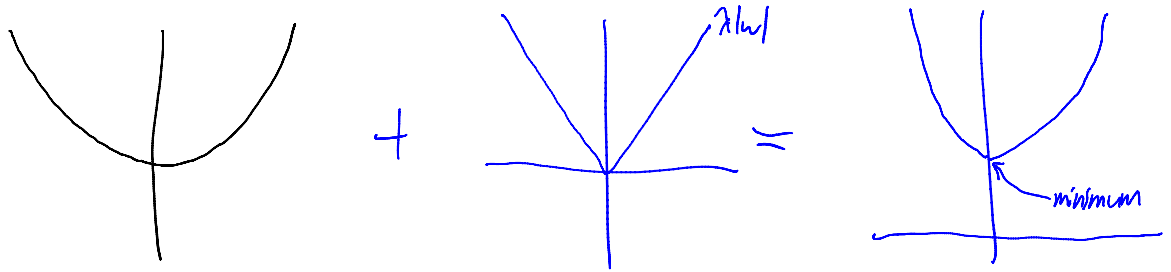
- L2-regularization adds a penalty on the L2-norm of 'w':

$$\underset{w \in \mathbb{R}^d}{\operatorname{argmin}} \frac{1}{2} \sum_{i=1}^{n} (y_i - w^T x_i)^2 + \frac{\lambda}{2} \sum_{j=1}^{d} w_j^2$$

- 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.

Feature Selection and L1-Regularization

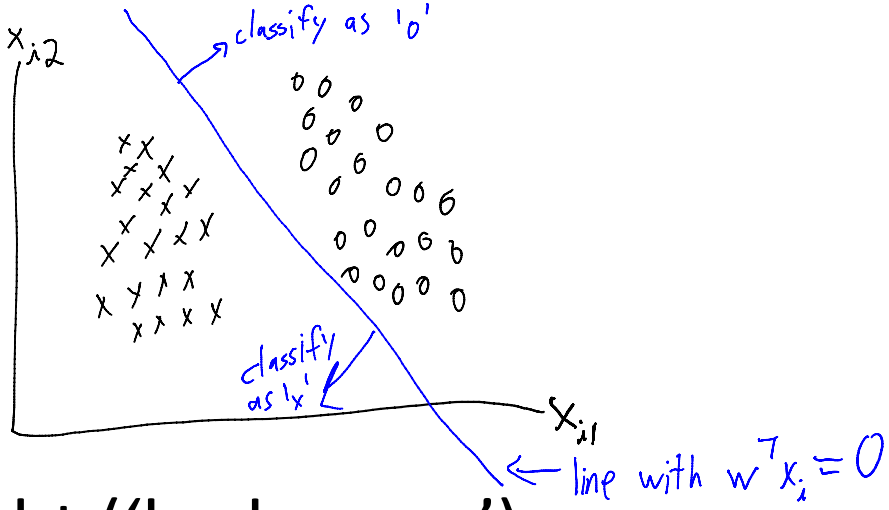
- Feature selection is task of finding ‘relevant’ variables.
  - Can be hard to precisely define ‘relevant’.
- Hypothesis testing methods:
  - Do tests trying to make variable ‘j’ conditionally independent of y.
  - Ignores effect size.
- Search and score methods:
  - Define score and search for variables 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.



Binary Classification and Logistic Regression

- Binary classification using regression by taking the sign:

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

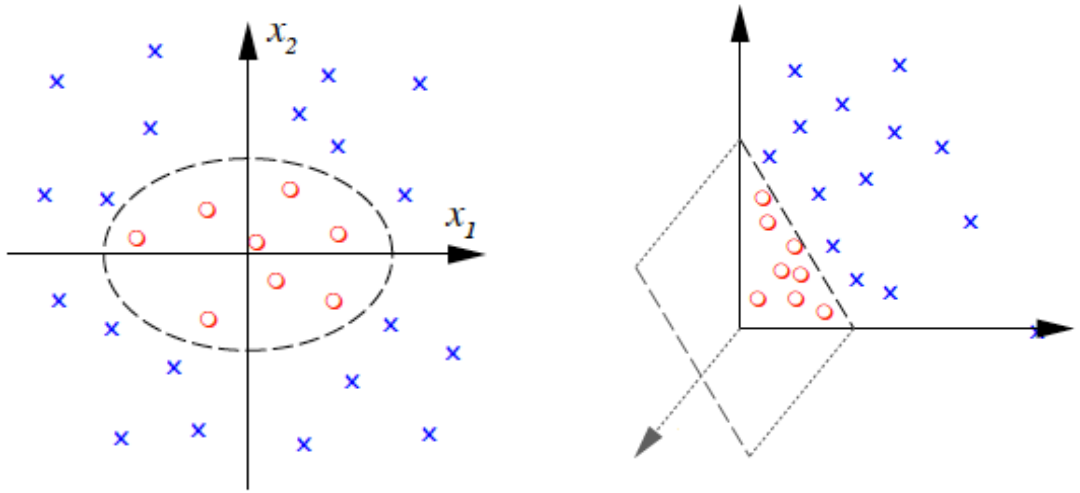


- But squared error penalizes for being too right ('bad errors').
- Ideal 0-1 loss is discontinuous/non-convex.
- Logistic loss is smooth and convex approximation:

$$\underset{w \in \mathbb{R}^d}{\text{argmin}} \sum_{i=1} \log(1 + \exp(-y_i w^T x_i))$$

Separability and Kernel Trick

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



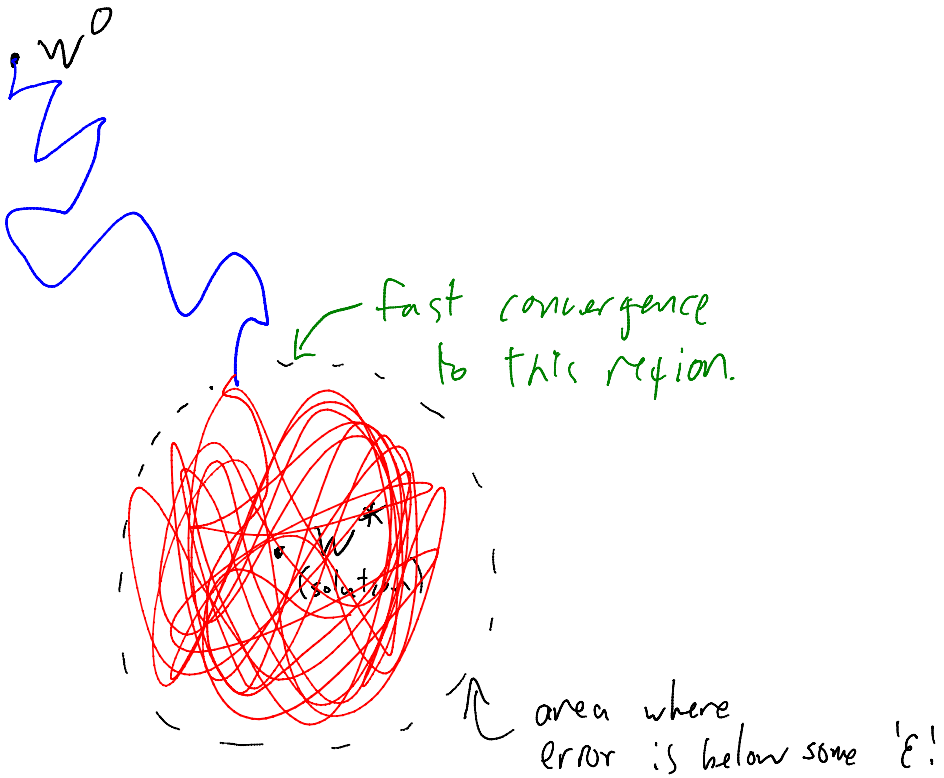
- Kernel trick: linear regression using similarities instead of features.

$$\hat{y} = \hat{X}w = \hat{K}(K + \lambda I)^{-1} y$$

where $K = XX^T$ and $\hat{K} = \hat{X}X^T$

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.

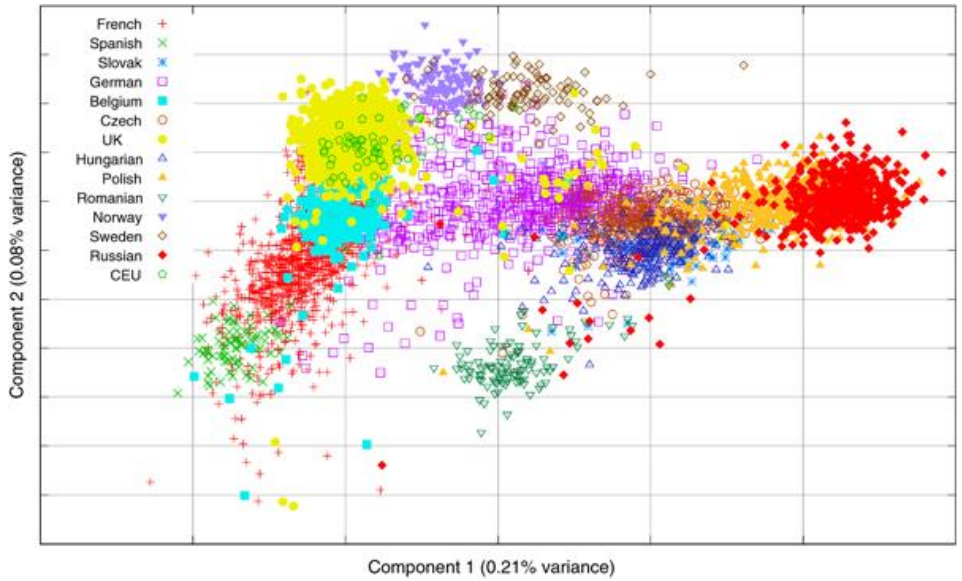


Latent-Factor Models

- Latent-factor models are unsupervised models that
  - Learn to predict features 'x_{ij}' based on weights 'w_j' and new features 'z_i'.

$$x_{ij} = w_j^T z_i$$

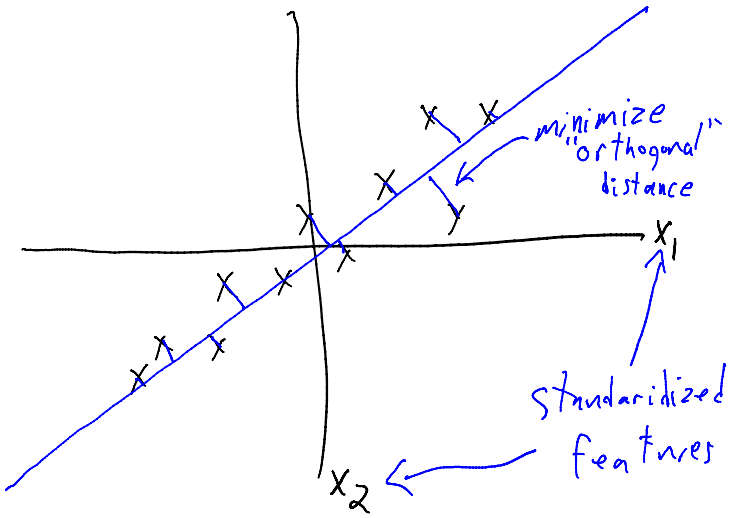
- Used for:
  - Dimensionality reduction.
  - Outlier detection.
  - Basis for linear models.
  - Data visualization.
  - Data compression.
  - Interpreting factors.



Principal Component Analysis

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

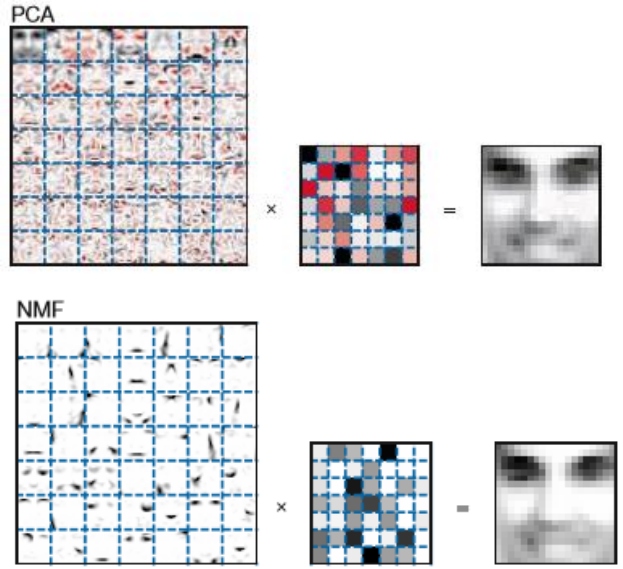
$$\sum_{i=1}^{n}\sum_{j=1}^{d}\left(x_{ij} - w_j^T z_i\right)^2$$

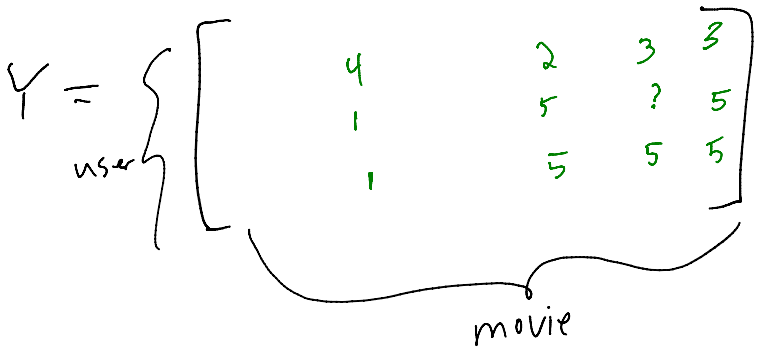


- With 1 factor, minimizes 'orthogonal' distance:
- To reduce non-uniqueness:
  - Constrain factors to have norm of 1.
  - Constrain factors to have inner product of 0.
  - Fit factors sequentially.
- Found by SVD or alternating minimization.

Beyond PCA

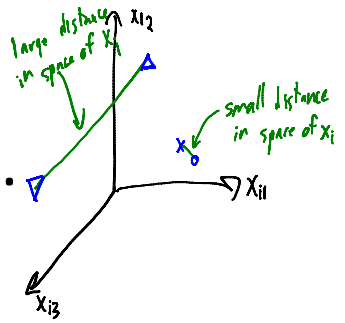
- Like L1-regularization, non-negative constraints lead to sparsity.
  - Although no parameter λ that controls level of sparsity.
- Non-negative matrix factorization:
  - Latent-factor model with non-negative constraints.
  - Learns additive 'parts' of objects.
- Could also use L1-regularization directly:
  - Sparse PCA and sparse coding.
- Regularized SVD and SVDfeature:
  - Filling in missing values in matrix.

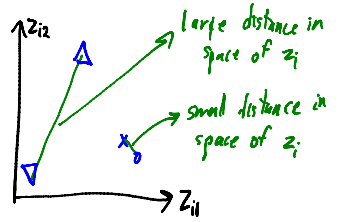




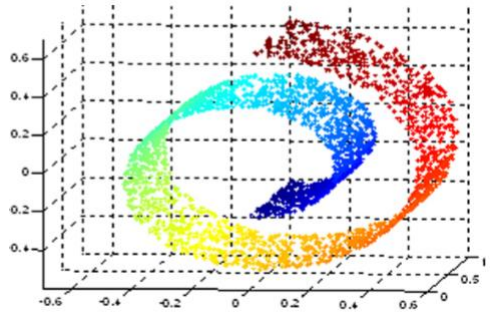
Multi-Dimensional Scaling

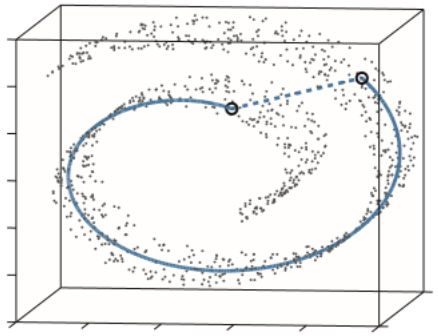
- Multi-dimensional scaling:
  - Non-parametric dimensionality reduction visualization.
  - Find low-dimensional 'z_i' that preserve distances.

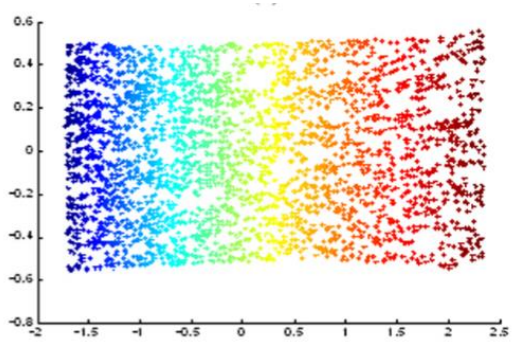




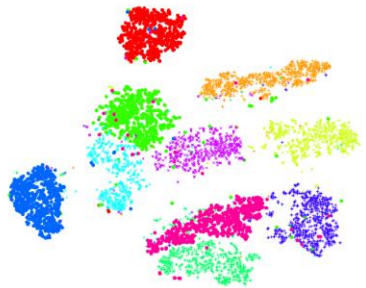
- Classic MDS and Sammon mapping are similar to PCA.
- ISOMAP uses graph to approximate geodesic distance on manifold.



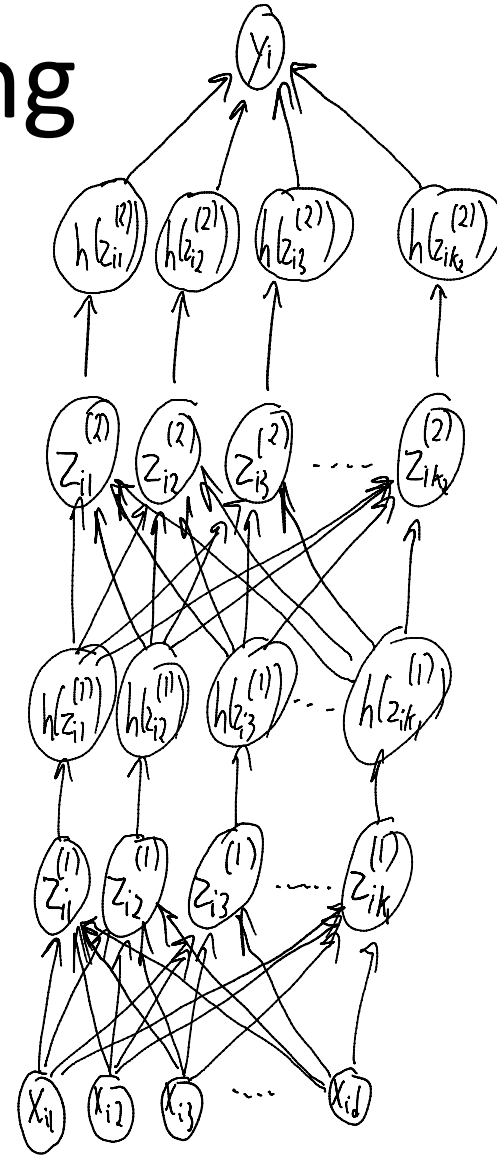




- T-SNE encourages 'repulsion' of close points.



Neural Networks and Deep Learning



- Neural networks combine latent-factor and linear models.
  - Linear-linear model is degenerate, so introduce non-linearity:
    - Sigmoid or hinge function.
  - Backpropagation uses chain rule to compute gradient.
  - Autoencoder is variant for unsupervised learning.
- Deep learning considers many layers of latent factors.
- Various forms of regularization:
  - Explicit L2- or L1-regularization.
  - Early stopping.
  - Dropout.
  - Convolutional and pooling layers.
- Unprecedented results on speech and object recognition.

Maximizing Probability and Discrete Label

- We can interpret many losses as maximizing probability:

$$\underset{\hat{y}_i}{\operatorname{argmax}} \prod_{i=1}^{\hat{n}} p(y_i \mid \hat{y}_i) \iff \underset{\hat{y}_i}{\operatorname{argmax}} - \sum_{i=1}^{\hat{n}} \log\left(p(y_i \mid \hat{y}_i)\right)$$

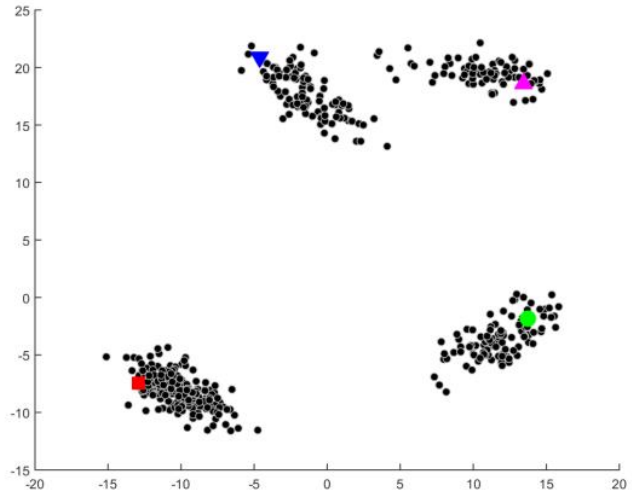
  - Sigmoid probability leads to logistic regression.
  - Gaussian probability leads to least squares.
- Allows us to define losses for with non-binary discrete 'y_i'.
- Softmax loss for categorical 'y_i':

$$p(y_i = c \mid \hat{y}_{i1}, \hat{y}_{i2}, \hat{y}_{i3}, \dots, \hat{y}_{ik}) = \frac{\exp(\hat{y}_{ic})}{\exp(\hat{y}_{i1}) + \exp(\hat{y}_{i2}) + \exp(\hat{y}_{i3}) + \cdots + \exp(\hat{y}_{ik})}$$

- Other losses for unbalanced, ordinal, and count labels.
- We can also define losses in terms of probability ratios:
  - Ranking based on pairwise preferences.

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.

Sequence Data

- Our data is often organized according to sequences:
  - Collecting data over time.
  - Biological sequences.
- Dynamic programming allows approximate sequence comparison:
  - Longest common subsequence, edit distance, local alignment.
- Markov chains define probability of sequences occurring.
  1. Sampling using random walk.
  2. Learning by counting.
  3. Inference using matrix multiplication.
  4. Stationary distribution using principal eigenvector.
  5. Decoding using dynamic programming.

Graph Data

- 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:
  - Graph-based SSL: which label does random walk reach most often?
  - PageRank: how often does infinitely-long random walk visit page?
  - Spectral clustering: which groups tend to contain random walks?
- Belief networks:
  - Generalization of Markov chains.
  - Allow us to define probabilities on general graphs.
  - Certain operations remain efficient.

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.
2. Intro to unsupervised learning (using counting and distances).
   - Clustering, association rules, outlier detection.
3. Linear models and gradient descent (for supervised learning)
   - Loss functions, change of basis, regularization, features 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. Sequence- and graph-structured data.
   - Specialized methods for these important special cases.

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 will we cover in CPSC 540?

1. Linear Models: Notation Upgrade

- We'll revisit core ideas behind linear models:
  - As we've seen, these are fundamental to more complicated models.
  - Loss functions, basis/kernels, robustness, regularization, large datasets.
- This time using matrix notation and matrix calculus:

Instead of $f(w) = \frac{1}{2}\sum_{i=1}^{n}(y_i - w^T x_i)^2 + \frac{\lambda}{2}\sum_{j=1}^{d} w_j^2$ and $\frac{\partial}{\partial w_j} f(w) = -x_{ij}(y_i - w^T x_i) + \lambda w_j$

We'll use $f(w) = \frac{1}{2}\|y - Xw\|^2 + \frac{\lambda}{2}\|w\|^2$ and $\nabla f(w) = -X^T(y - Xw) + \lambda w$

- Everything in terms of probabilities:

$\nabla^2 f(w) = X^T X + \lambda I$

  - Needed if you want solve more complex problems.

$f(w) = -\log(p(y|Xw)) - \log(p(w))$

1. Linear Model: Filling in Details

- We'll also fill in details of topics we've ignored:
  - How can we write the fundamental trade-off mathematically?

$$E[(y_i - w^T x_i)^2]$$

  - How do we show functions are convex?

$$X^T D X \succeq 0$$

  - How many iterations of gradient descent do we need?

$$f(w^t) - f(w^*) \leq \left(1 - \frac{\mu}{L}\right)^t [f(w^0) - f(w^*)] \qquad \varepsilon = \varepsilon_{apr} + \varepsilon_{est} + \varepsilon_{opt}$$

  - How do we solve non-smooth optimization problems?

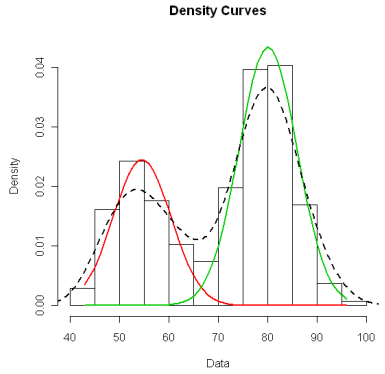
$$\min c^T w \text{ s.t. } Aw \leq b \qquad \text{prox}_g(x) = \underset{y}{\text{argmin}} \left\{ \frac{1}{2}\|y - x\|^2 + g(y) \right\}$$

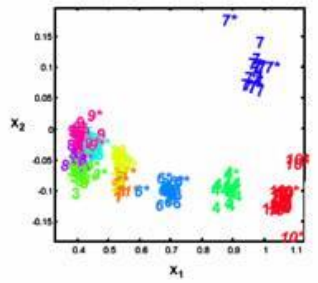
  - How can get sparsity in terms of 'groups' or 'patterns' of variables?

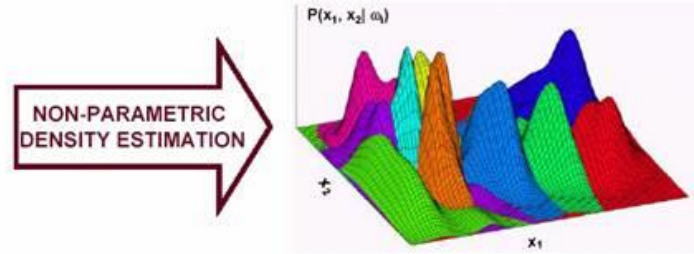
$$\frac{1}{2}\|y - Xw\|^2 + \sum_{g \in G} \|w_g\|_2$$

2. Density Estimation

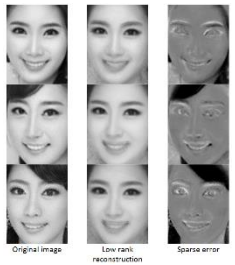
- Methods for estimating multivariate distributions p(x) or p(y|x).
  - Abstract problem, includes most of ML as a special case.
  - But going beyond simple Gaussian and independent models.
- Classic models:
  - Mixture models.
  - Non-parametric models.
- Latent-factor models:
  - Factor analysis, robust PCA, ICA, topic models.

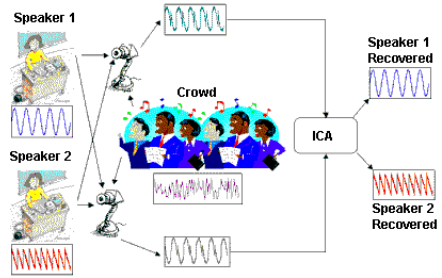


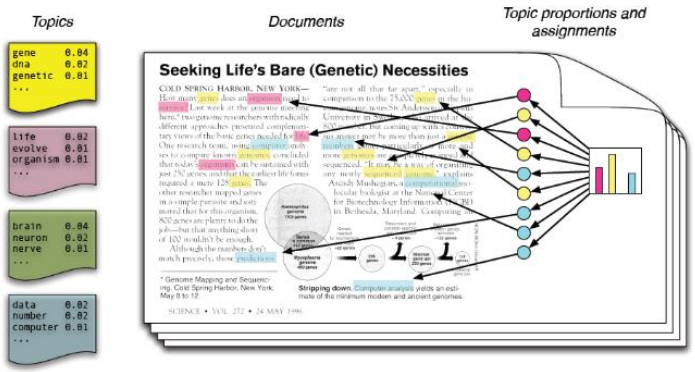




Trait	Description
Openness	Being curious, original, intellectual, creative, and open to new ideas.
Conscientiousness	Being organized, systematic, punctual, achievement-oriented, and dependable.
Extraversion	Being outgoing, talkative, sociable, and enjoying social situations.
Agreeableness	Being affable, tolerant, sensitive, trusting, kind, and warm.
Neuroticism	Being anxious, irritable, temperamental, and moody.

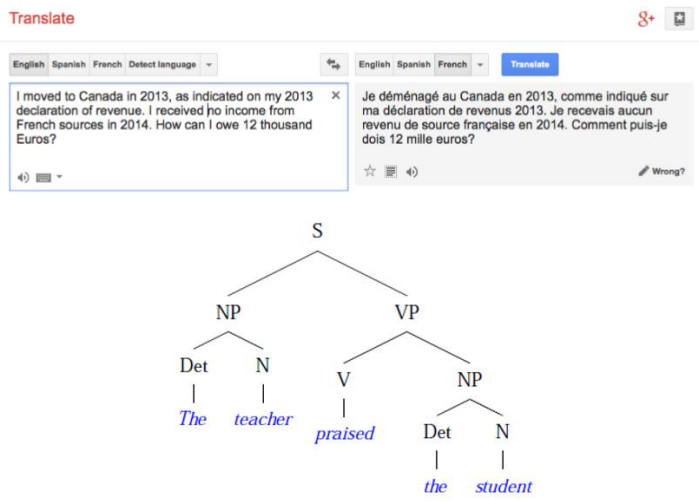


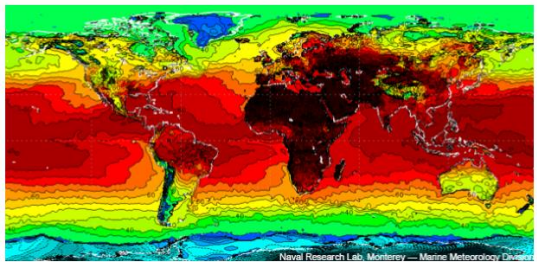


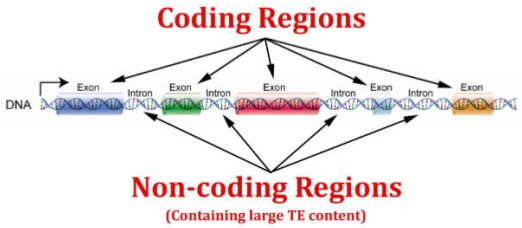


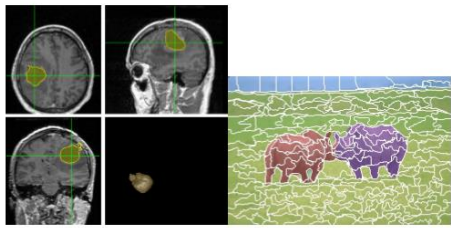
3. Structured Prediction and Graphical Models

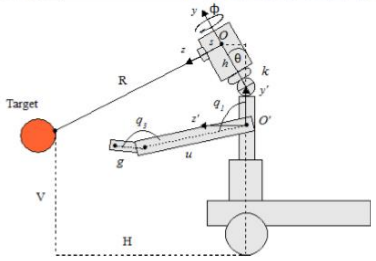
- Structured prediction:
  - Instead of class label 'y_i', our output is a general object.

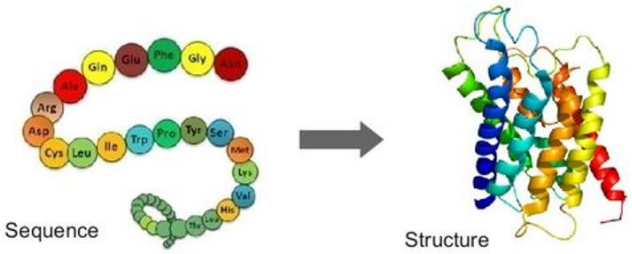


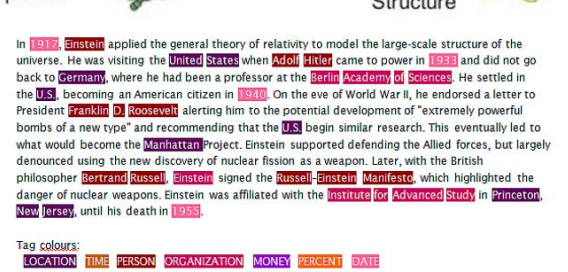








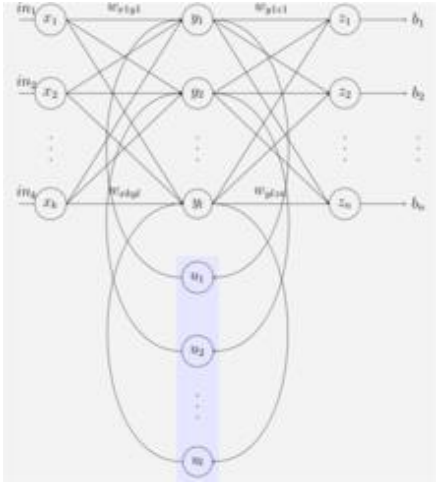




- Conditional random fields and structured support vector machines.
- Relationship of graph to dynamic programming (treewidth).
- Variational and Markov chain Monte Carlo for inference/decoding.

4. Deep Learning

- Deep learning with matrix calculus:
  - Backpropagation and convolutional neural networks in detail.
- Unsupervised deep learning:
  - Deep belief networks and deep restricted Boltzmann machines.
- How can we add memory to deep learning?
  - Recurrent neural networks, long short-term memory, memory vectors.



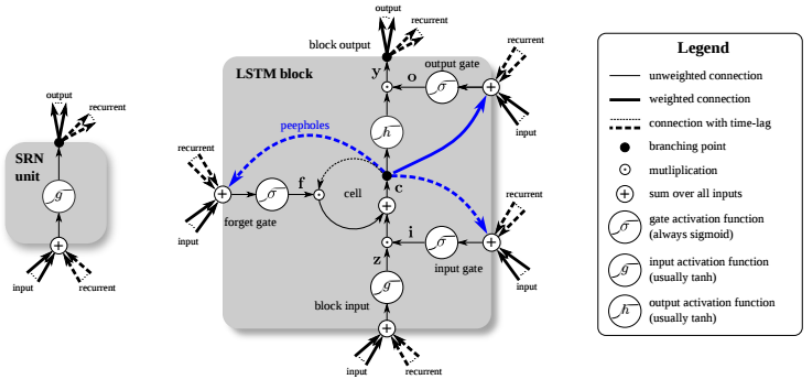
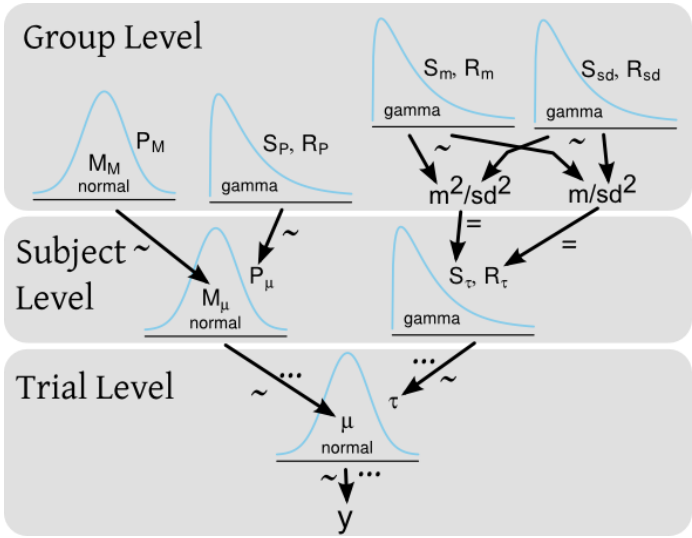


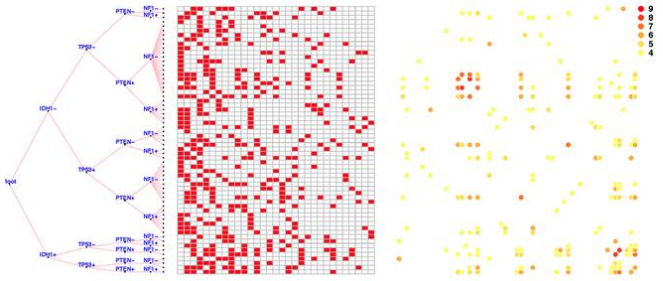
Figure 1. Detailed schematic of the Simple Recurrent Network (SRN) unit (left) and a Long Short-Term Memory block (right) as used in the hidden layers of a recurrent neural network.

Bilbo travelled to the cave.
Gollum dropped the ring there.
Bilbo took the ring.
Bilbo went back to the Shire.
Bilbo left the ring there.
Frodo got the ring.
Frodo journeyed to Mount-Doom.
Frodo dropped the ring there.
Sauron died.
Frodo went back to the Shire.
Bilbo travelled to the Grey-havens.
The End.
Where is the ring? A: Mount-Doom
Where is Bilbo now? A: Grey-havens
Where is Frodo now? A: Shire

5. 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.





6. Online, Active, and Causal Learning

- 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.

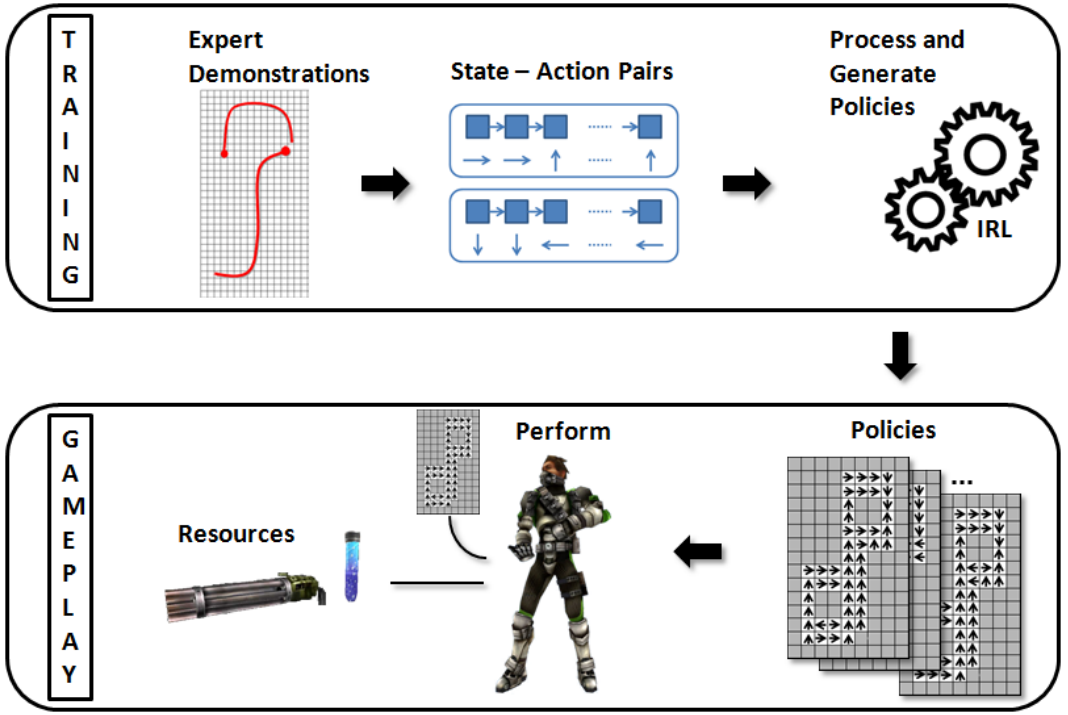
6. Online, Active, and Causal Learning

- 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

- 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=SH3bADiB7uQ
https://www.youtube.com/watch?v=nUQsRPJ1dYw



8. Learning Theory

- Other forms of fundamental trade-off.

$$\mathrm{E}\Big[\big(y-\hat{f}(x)\big)^2\Big] = \mathrm{Bias}\big[\hat{f}(x)\big]^2 + \mathrm{Var}\big[\hat{f}(x)\big] + \sigma^2$$

$$P\left(\text{test error} \le \text{training error} + \sqrt{\frac{h(\log(2N/h)+1) - \log(\eta/4)}{N}}\right) = 1-\eta$$

$$\mathbb{P}(\overline{X} - \mathbb{E}[\overline{X}] \geq t) \leq e^{-2nt^2}$$

notion of sample complexity answers this question. The sample complexity of a learning algorithm is a function $n(\rho, \epsilon, \delta)$ such that for all $n \geq n(\rho, \epsilon, \delta)$,

$$\mathbb{P}_{S_n}(\mathcal{R}(f_n) - \mathcal{R}^*_{\mathcal{H}} > \epsilon) < \delta.$$

In words, the sample complexity $n(\rho, \epsilon, \delta)$ defines the rate of consistency of the algorithm: given a desired accuracy ϵ and confidence δ, one needs to sample at most $n(\rho, \epsilon, \delta)$ data points to guarantee that the risk of the output function is within ϵ of the best possible, with probability at least $1 - \delta$.[2]

Say that there is an algorithm A that given access to $EX(c, D)$ and inputs ϵ and δ that, with probability of at least $1 - \delta$, A outputs a hypothesis $h \in C$ that has error less than or equal to ϵ with examples drawn from X with the distribution D. If there is such an algorithm for every concept $c \in C$, for every distribution D over X, and for all $0 < \epsilon < 1/2$ and $0 < \delta < 1/2$ then C is **PAC learnable** (or *distribution-free PAC learnable*). We can also say that A is a **PAC learning algorithm** for C.

Many authors (e.g., Bousquet, 2002; Bartlett and Mendelson, 2006; Bartlett et al., 2006) obtain fast statistical estimation rates in more general conditions. These bounds have the general form

$$\mathcal{E}_{\text{app}} + \mathcal{E}_{\text{est}} \le c\left(\mathcal{E}_{\text{app}} + \left(\frac{d}{n}\log\frac{n}{d}\right)^{\alpha}\right) \quad \text{for } \frac{1}{2} \le \alpha \le 1\,. \tag{1.4}$$

This result holds when one can establish the following variance condition:

$$\forall f \in \mathcal{F} \quad \mathbb{E}\left[\left(\ell(f(X),Y) - \ell(f^*_{\mathcal{F}}(X),Y)\right)^2\right] \le c\left(E(f) - E(f^*_{\mathcal{F}})\right)^{2-\frac{1}{\alpha}}. \tag{1.5}$$