# Announcements

- Next Tuesday, May 2
  - Last class
  - Projects due 11:59PM
    - Paper w/ required sections
    - Code
    - Sample runs
    - Instructions on how to run
  - email .zip to dejong@cs.uiuc.edu

# Announcements

- Final Exam
  - Emphasizes material since midterm
  - Length 1hr. 15min.
  - Wed. May 10;
    9:00AM – 10:15AM
  - 1320 DCL for NetIDs starting A - N
  - 151 Loomis for NetIDs starting O - Z

# Dimensionality Reduction

- “Curse of Dimensionality”
- Transform the data
  – from a high-dimensional space
  – to a space of lower dimension
  – keeping as much useful information as possible
- “Feature Extraction”
- (Lossy) Compression

# Principal Component Analysis

- Start with a data set of examples w/ numeric features
- Linear transform of the data
- Replace original features
  - linear combinations of original features
  - new features are orthogonal (uncorrelated)
  - ordered by "importance"
    (importance = account for variance/information in data)
- Ignores any class information
- Spreads out the data in a more natural way
- Best if
  - Observed data = signal (i.e., pattern) + noise
  - noise is independent and Gaussian

# PCA

- Spectral decomposition of the covariance matrix
- Ooooooh!!
- Quick background on covariance

# Recall from earlier

- Assume our data set is an iid random sample
- Mean of a random variable x
  - mean(x) = E[x] (the expectation)
  - estimate from a data sample by averaging
  - expected value need not be very likely (e.g., coin flips)
- Generalizes immediately to multivariate **x**
  - $\mathbf{x}$ = < $x_1$, $x_2$, $x_3$,… $x_n$>
  - mean(**x**) is also a vector:
    $E[\mathbf{x}]$ = < $E[x_1]$, $E[x_2]$, $E[x_3]$,… $E[x_n]$>
  - can translate distribution to the origin by subtracting the mean from each datum

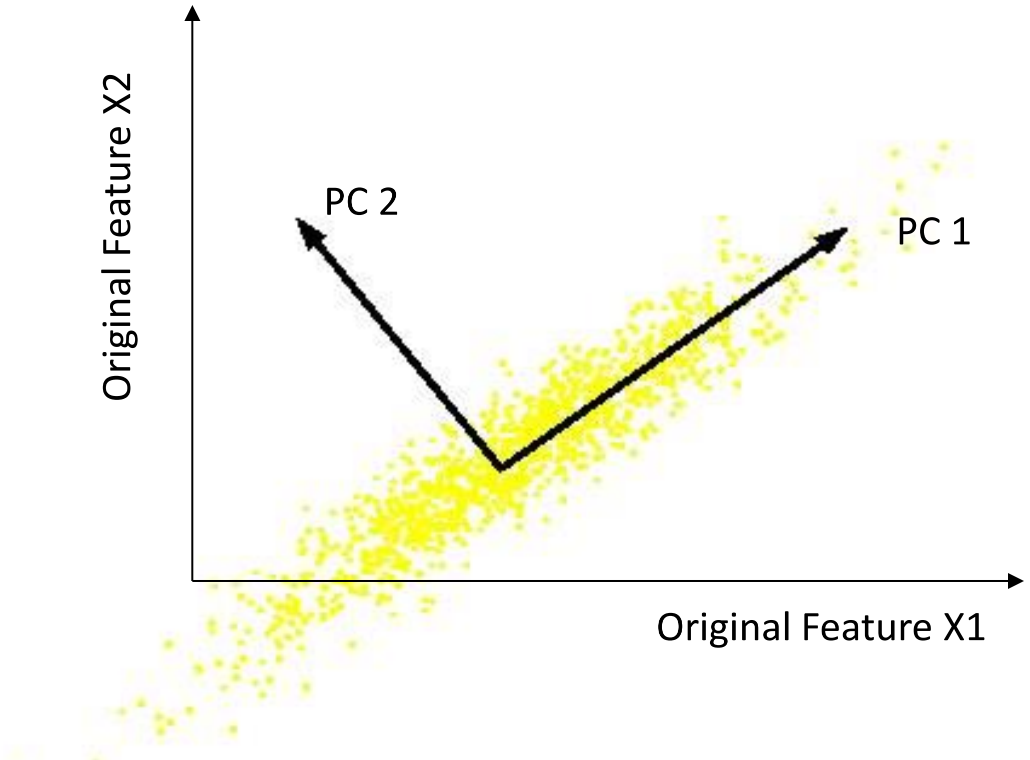
# Recall from earlier

- Variance characterizes the dispersion about the mean
  - $\mathrm{var}(x) = \sigma^2(x) = E[(x - \mathrm{mean}(x))^2]$
  - squared avoids cancelling + / -
  - $\sigma^2(x)$ is non-negative
- Multivariate generalization is the covariance matrix
  - $\mathrm{var}(\mathbf{x})$ is a matrix
  - n x n where **x** has n components
  - $\mathrm{var}(\mathbf{x})_{i,j} = \mathrm{covar}(x_i, x_j) = E[\ (x_i - E[x_i]) \cdot (x_j - E[x_j])\ ]$
  - $\mathrm{var}(\mathbf{x}) = \boldsymbol{\Sigma}(\mathbf{x}) = E[\ (\mathbf{x} - E[\mathbf{x}])\ (\mathbf{x} - E[\mathbf{x}])^T\ ]$
  - $\boldsymbol{\Sigma}(\mathbf{x})$ is symmetric
  - components can be negative
    but $\boldsymbol{\Sigma}(\mathbf{x})$ is positive semi-definite
  - Diagonal components are variances
  - Off diagonal: product of i, j standard deviations times the i, j correlation coefficient

# PCA

- Subtract the mean from the data set
- Find the linear combination of features that accounts for most of the variance
- This is the first principal component
- Project the data onto its subspace
- The projected data have zero variance in this dimension
- Repeat
- For n original features we get n principal components
- If the features are not linearly independent
  - we will run out of variance to account for
  - the covariance matrix is singular
    (non-invertible, has a null space, zero determinant,…)
  - the last principal components will be degenerate

# PCA




- PC1: first principal component, accounting for the most variance
- PC2: second principal component
- Note significant covariance among the original features
- But not in the transformed space; variances are axis-aligned & uncorrelated

# PCA

- Each principal component is a "new" feature
  - Each is a linear combination of old features
  - They are mutually orthogonal
  - uncorrelated, zero covariance
  - the new covariance matrix is diagonal
- With all (non-degenerate) principal components
  - linear transformation of the data
  - preserves all of the information
- Keep only the first k principal components
  - loses information
  - but keeps most of the variance
- Lossy data compression

# PCA

- The principal components are the eigenvectors of the covariance matrix
- The magnitude of their corresponding eigenvalue specifies the amount of variance accounted for
- Eigen decomposition (aka spectral decomposition)

# PCA Procedure

- Find eigenvalues & eigenvectors (e.g., SVD)
- Sort eigenvectors on magnitude of their eigenvalues
- Drop eigenvectors of small eigenvalues
- Assemble remaining (unit) eigenvectors into a transformation matrix
- Centralize the original data (subtract mean)
- Transform into the new lower dim. space (matrix multiply)
- Learn a classifier using the transformed data

# Many Others…
## Random Projection

- PCA projects examples onto principal components
- High dimensional space (BOW for NLP or vision)
- Instead of PCA
  - choose random unit vectors
  - assemble into a transformation matrix
  - significant dimensionality reduction if |RP| << n
- Can work quite well (!)
  - not quite as well…
  - improves w/ number of random vectors
  - much cheaper than PCA, SVD
- As number of random vectors increases
  - interpoint distances between examples are preserved
  - with high probability

# Games & Game Theory

- Tic-Tac-Toe,
  Qubic, Othello, Checkers, Chess
- Monopoly
  Backgammon
- Chutes & Ladders
  Card Game War
  Casino Craps?
- Seven Minutes in Heaven
- Which of these games would you want to play with a computer?

# What is a game?

- Ludwig Wittgenstein, philosopher
  - “Whereof we cannot speak, therefore we must be silent”
  - “Game” cannot be defined except by family resemblance
- Roger Caillois, sociologist
  - fun
  - separate in time and place
  - unforeseeable outcome
  - accomplish nothing useful
  - governed by rules
  - fictitious
- Many others…

# Game Theory

Decision-making According to Rules in a Multi-agent Setting

- Economics, Psychology, Computer Science…
- Multi-agent
  - Do we need to consider other agents?
  - Standard Reinforcement Learning?
  - Agent models (cooperative, competitive)
    - Intelligent, rational
    - bounded rationality
- Decision-making
- Mechanism design

# Important Distinctions

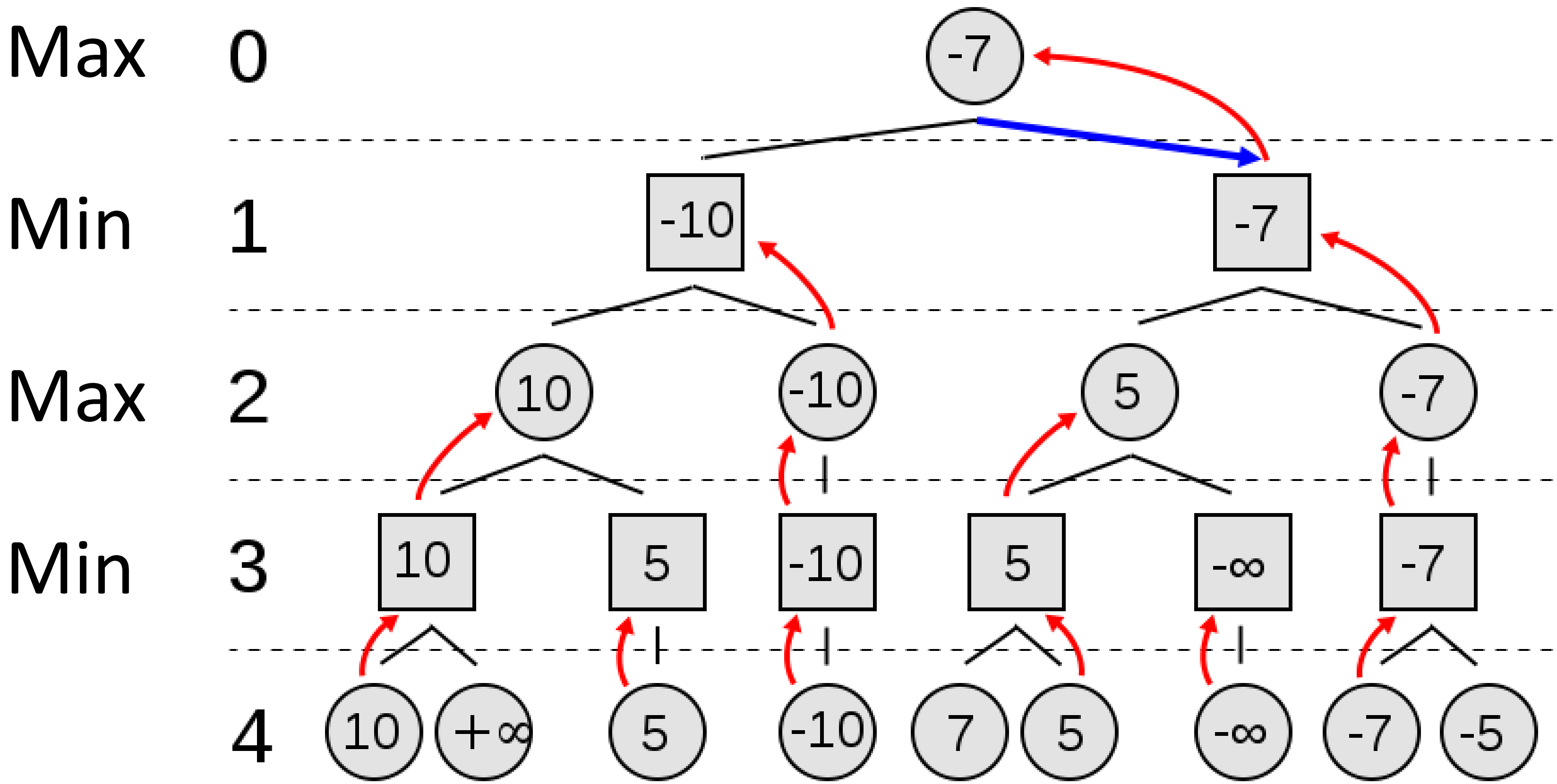
- Games
  - Zero-sum / Non-Zero-sum
  - Simultaneous / Turn-taking / Continuous
  - Perfect information / Imperfect information / Stochastic
  - Extensive and Normal Forms

# Extensive Form

- Best for Sequential or Turn-Taking games
- Generalization of a decision tree
- Game state changes with each action
- Ply: a single action
- Move: two plys
- Evaluator (SBE) computes a Utility for the state
  - For zero-sum, can use the same evaluator
  - High=good for player A; Low=good for player B
- Mini-Max procedure greatly improves over direct Evaluator application
- Alternate levels want to maximize & minimize utility

# 4-Ply Mini-Max Game Tree

## with $\alpha$-$\beta$ Pruning; two possible actions: L & R




Evaluator is applied only at the lowest level; These values a propagated up using the mini-max procedure; Note that some node become dominated and need not be evaluated

From Wikipedia Minimax

# Game Tree Issues

- Horizon effect
  - Good line of play
  - Deferring a loss
- Search until quiescence
  - Unanswered threats
  - Continue the search for additional plys
- Secondary search
  - After an action is chosen
  - Explore the chosen line of play more deeply
- Table of openings and end games
- Training Evaluation function parameters
  - Self play
  - Games w/ expert
  - Expert-Expert games

# Chess playing systems

- 200 million node evalutions per move (3 min)
  - minimax with a decent evaluation function and quiescence search
  - ~ 5 ply; human novice
- Alpha-beta pruning
  - ~ 10 ply; experienced player
- Deep Blue:
  - 30 billion evaluations per move
  - Evaluation function with 8000 features
  - Extensive opening and endgame tables
  - ~ 14 ply; grand master
- Hydra
  - 36 billion evaluations per second
  - ~ 18 ply; unbeatable by humans?  reduces chess to tic-tac-toe?