CS5824: Advanced Machine Learning

Dawei Zhou

CS, Virginia Tech

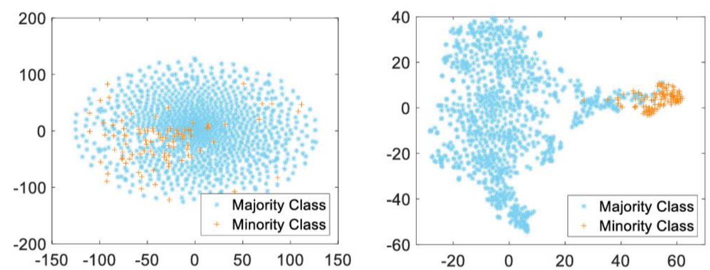
Please keep your face covering on!

COMPUTER SCIENCE
VIRGINIA TECH.

Feature Selection

Feature Selection & Feature Engineering

- **Feature Selection**
  - Given a set of p initial features, how to select a few most effective ones?
  - Why?
    - Irrelevant features (student ID for predicting GPA)
    - Redundant features (monthly income vs. yearly income)
- **Feature Engineering**
  - Given the initial features, how to construct more effective ones?
    - # of daily positive cases, # of daily tests, # of daily hospitalization -> weekly positive rate
  - (traditionally) domain knowledge is the key
  - Deep learning provides an automatic way

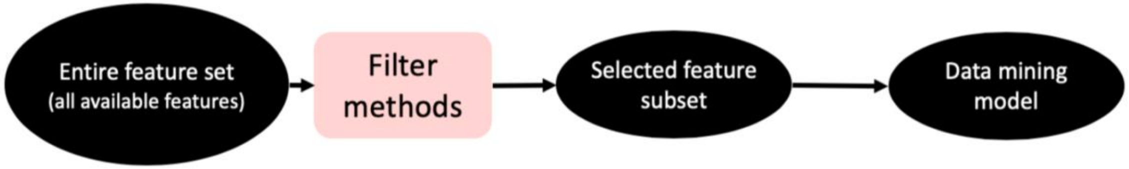


(a) Original Feature Space (b) Embedding Space

Feature Selection Methods

- **Filter methods**
  - Select features based on some goodness measure
  - Independent of the specific classification model
- **Wrapper methods**
  - Combine the feature selection and classifier model construction steps together in a sequential way
  - Iteratively
    - use the currently selected feature subset to construct a classification model
    - Use the current classification model to update the selected feature subset.
- **Embedded methods**
  - Simultaneously constructs the classification model and selects the relevant features
  - Embed the feature selection step during the classification model construction step

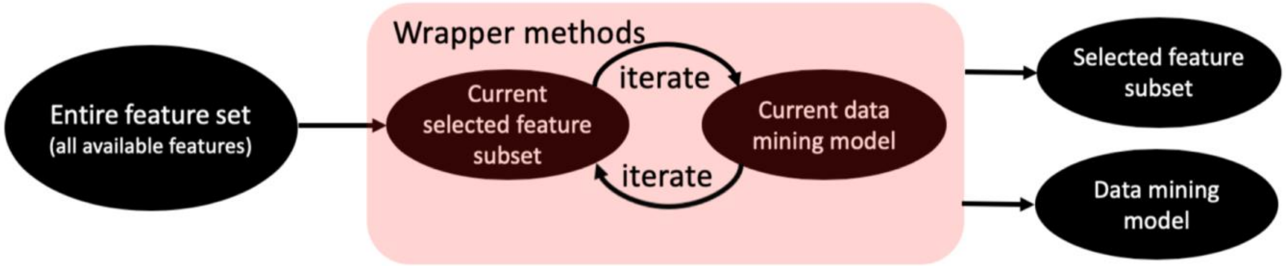
Filter Methods



- **General Procedure**
  - Selects features based on some goodness measure
  - Independent of the specific classification model
- **Fisher Scores**
  - Intuitions: the feature x (e.g., income) is strongly correlated with the class label y (buy computer) if
    - Condition #1: the average income of all customers who buy a computer is significantly different from the average income of all customers who do not buy a computer,
    - Condition #2: all customers who buy a computer share similar income,
    - Condition #3: all customers who do not buy a computer share similar income.
  - Details $$s = \frac{\sum_{j=1}^{c} n_j(\mu_j - \mu)^2}{\sum_{j=1}^{c} n_j \sigma_j^2}$$
- **Other measures:** information gain, mutual information,

Wrapper Methods

- **General Procedure**
  - Combines the feature selection and classifier model construction steps together,
  - Iteratively
    - Use the currently selected feature subset to construct a classification model
    - Use the current classification model to update the selected feature subset.
- **Key: how to search for the best feature subset**
  - Exhaustive search: $2^p - 1$ (exponential)
  - Stepwise forward selection:
    - Start with an empty feature subset.
    - At each iteration, select an additional feature to improve performance most
  - Stepwise backward elimination: start with the full set, eliminate one feature at a time
  - Hybrid method

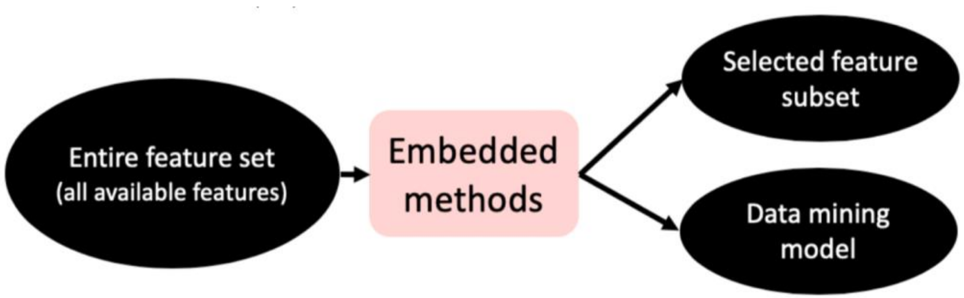


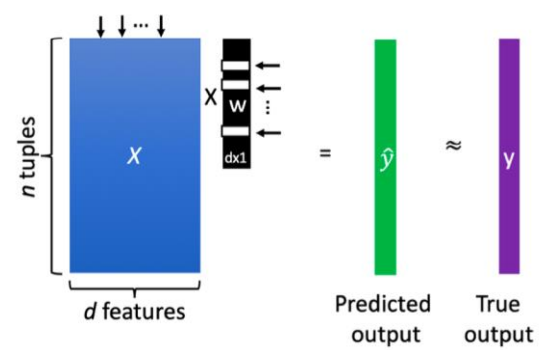
Embedded Methods

- General Procedure
  - Simultaneously constructs the classification model and selects the relevant features
  - Embed the feature selection step during the classification model construction step
- LASSO: Least Absolute Shrinkage and Selection Operator
  - Model

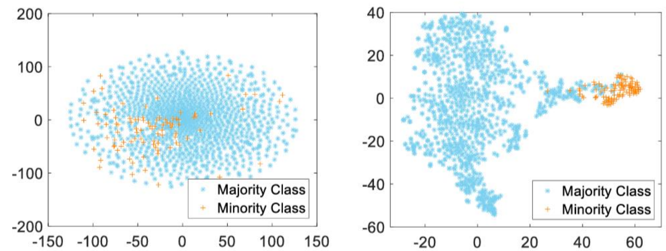
$$\hat{L}(w) = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda\|w\|_1 = \underbrace{\sum_{i=1}^{n}(y_i - w^T x_i)^2}_{\text{Goodness of prediction}} + \underbrace{\lambda\sum_{j=0}^{p}|w_j|}_{\text{(convex) approximation of \# of selected features}}$$

  - Training: path-wise coordinate decent

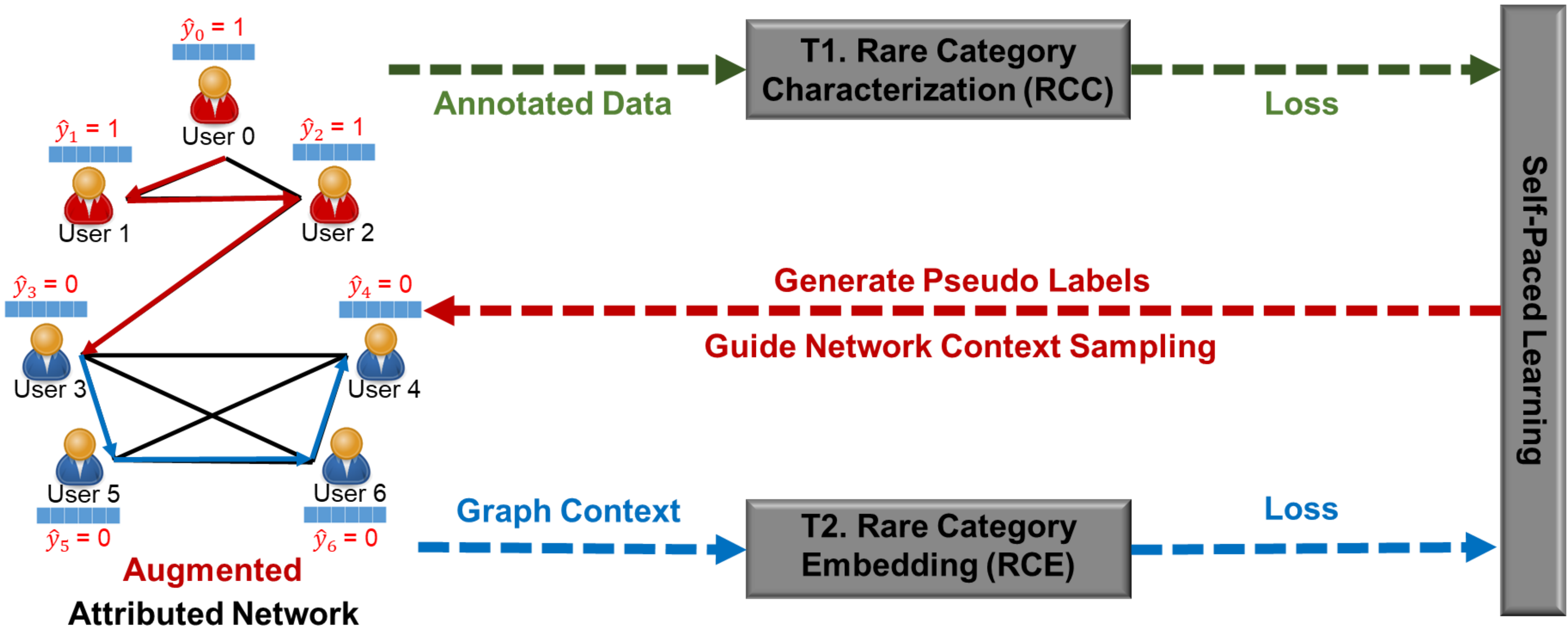




Ex. Embedded Methods



(a) Original Feature Space (b) Embedding Space

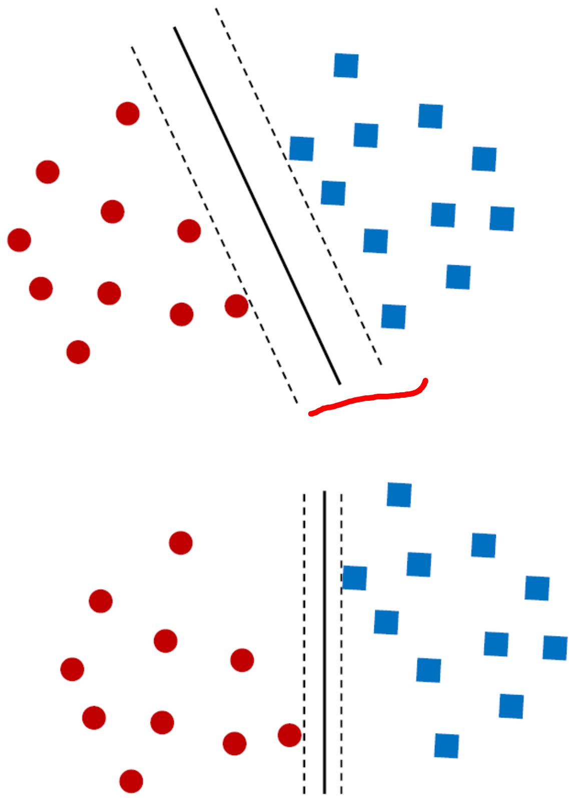


Support Vector Machines

Classification: A Mathematical Mapping

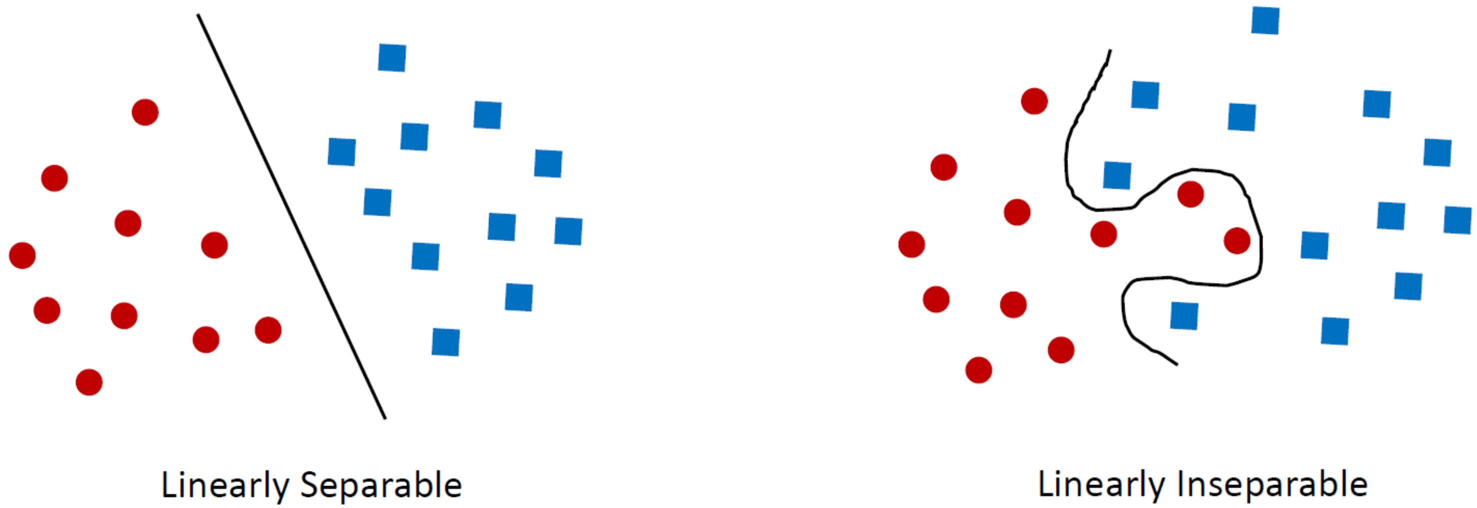
- **The binary classification problem:**
  - E.g., Movie review classification
    - $\boldsymbol{x} = (x_1, x_2, x_3, \dots), y_i = +1 \; or -1$ (positive, negative)
    - x_1 : # of word “awesome”
    - x_2 : # of word “disappointing”
  - Mathematically, $\boldsymbol{x} \in \boldsymbol{X} = \Re^n, y \in Y = \{+1, -1\}$
    - We want to derive a function f: $X \rightarrow Y$
    - which maps input examples to their correct labels

SVM—General Philosophy



- Learning a max-margin classifier
  - From the infinite set of lines (hyperplanes) separating two classes
  - Find the one which separates two classes with the largest margin
  - i.e. a maximum marginal hyperplane (MMH)

SVM—When Data Is Linearly Separable



- The simplest case: When data is linearly separable
  - Data sets whose classes can be separated exactly by linear decision surfaces are said to be linearly separable

Linear SVM for Linearly Separable Data

- A separating hyperplane can be written as

$$\boldsymbol{w}^T\boldsymbol{x} + b = 0$$

Model parameters to learn

  - where $\boldsymbol{w} = (w_1, w_2, \ldots, w_n)^T$ is a weight vector and b is a scalar (bias)
  - For 2-D, it can be written as: $w_1x_1 + w_2x_2 + b = 0$
- The hyperplane defining the sides of the margin:

H_1: $w_0 + w_1 x_1 + w_2 x_2 \geq 1$ for $y_i = +1$, and

H_2: $w_0 + w_1 x_1 + w_2 x_2 \leq -1$ for $y_i = -1$

- Any training tuples that fall on hyperplanes H1 or H2 (i.e., the sides defining the margin) are support vectors

Linear SVM for Linearly Separable Data

- The distance from any data point $\boldsymbol{x}$ to the separating hyperplane is

$$\text{distance}(ax + by + c = 0, (x_0, y_0)) = \frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}} \qquad r = \frac{|f(\boldsymbol{x})|}{||\boldsymbol{w}||} = \frac{y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)}{||\boldsymbol{w}||}$$

- Our objective is to maximize the distance of the closest data point to the hyperplane

$$\arg\max_{w,b}\left\{\frac{1}{||\boldsymbol{w}||}\min[y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)]\right\}$$

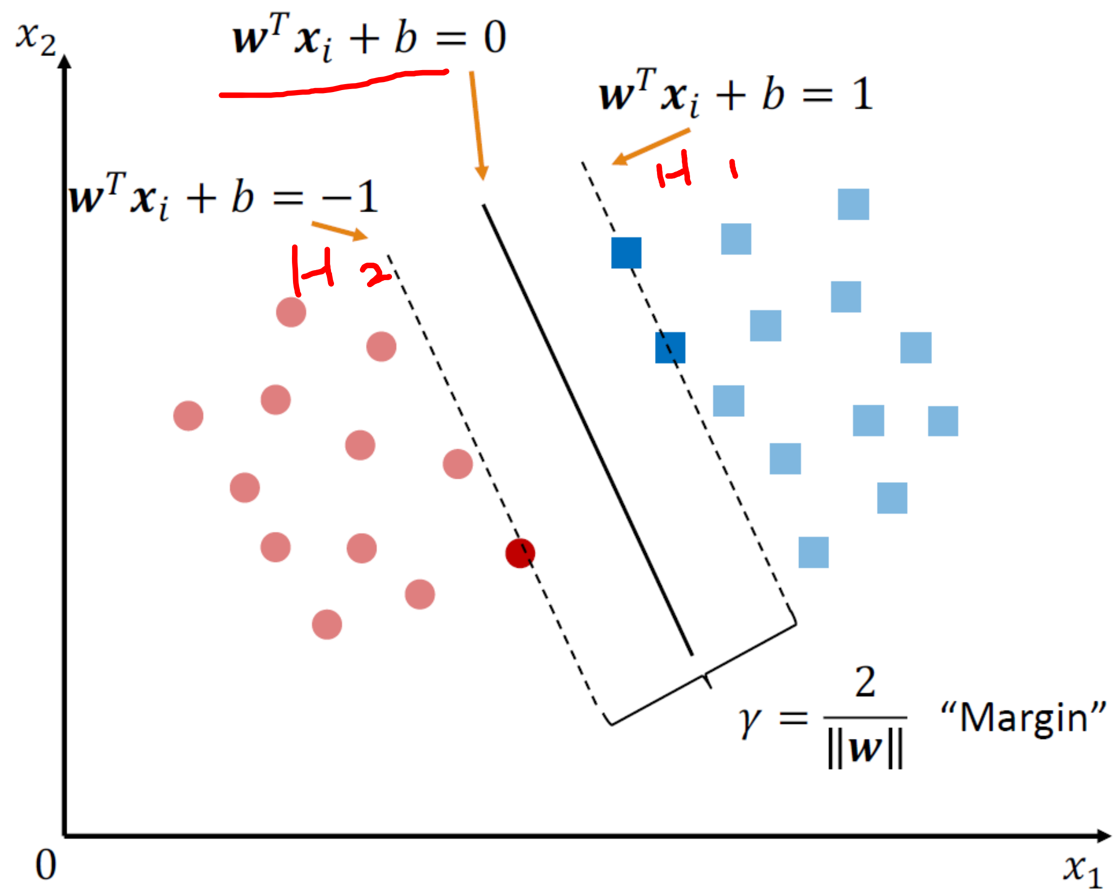
- This is hard to solve, we shall convert it to an easier problem

$$\boxed{\begin{aligned} &\arg\min_{w,b} \quad ||\boldsymbol{w}||^2 \\ &\quad\text{s. t.} \quad \underline{y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) \geq 1,} \qquad i = 1,2, \dots, n \end{aligned}}$$

- This is the basic form of SVM, and it can be solved by using quadratic programming

Linear SVM for Linearly Separable Data

- The data points closest to the separating hyperplane are called support vectors

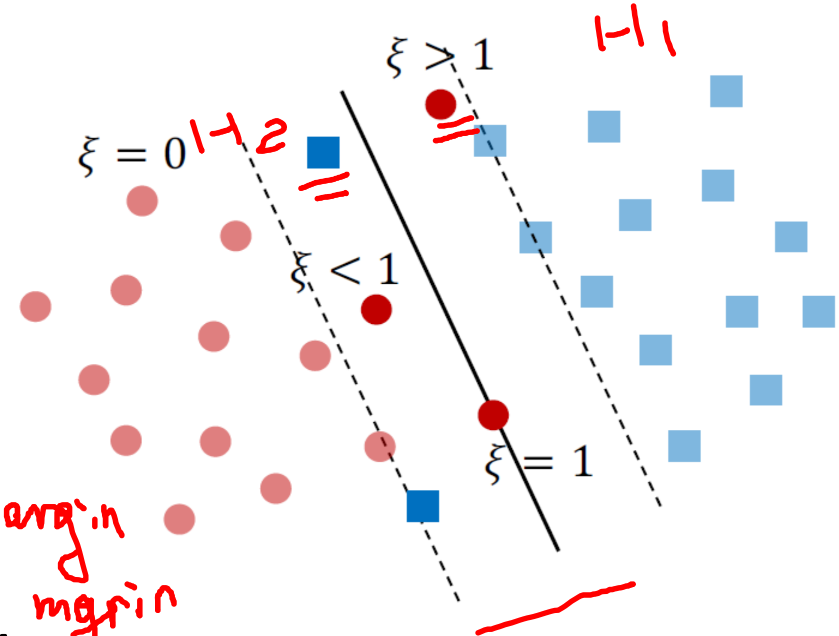


SVM for Linearly Inseparable Data

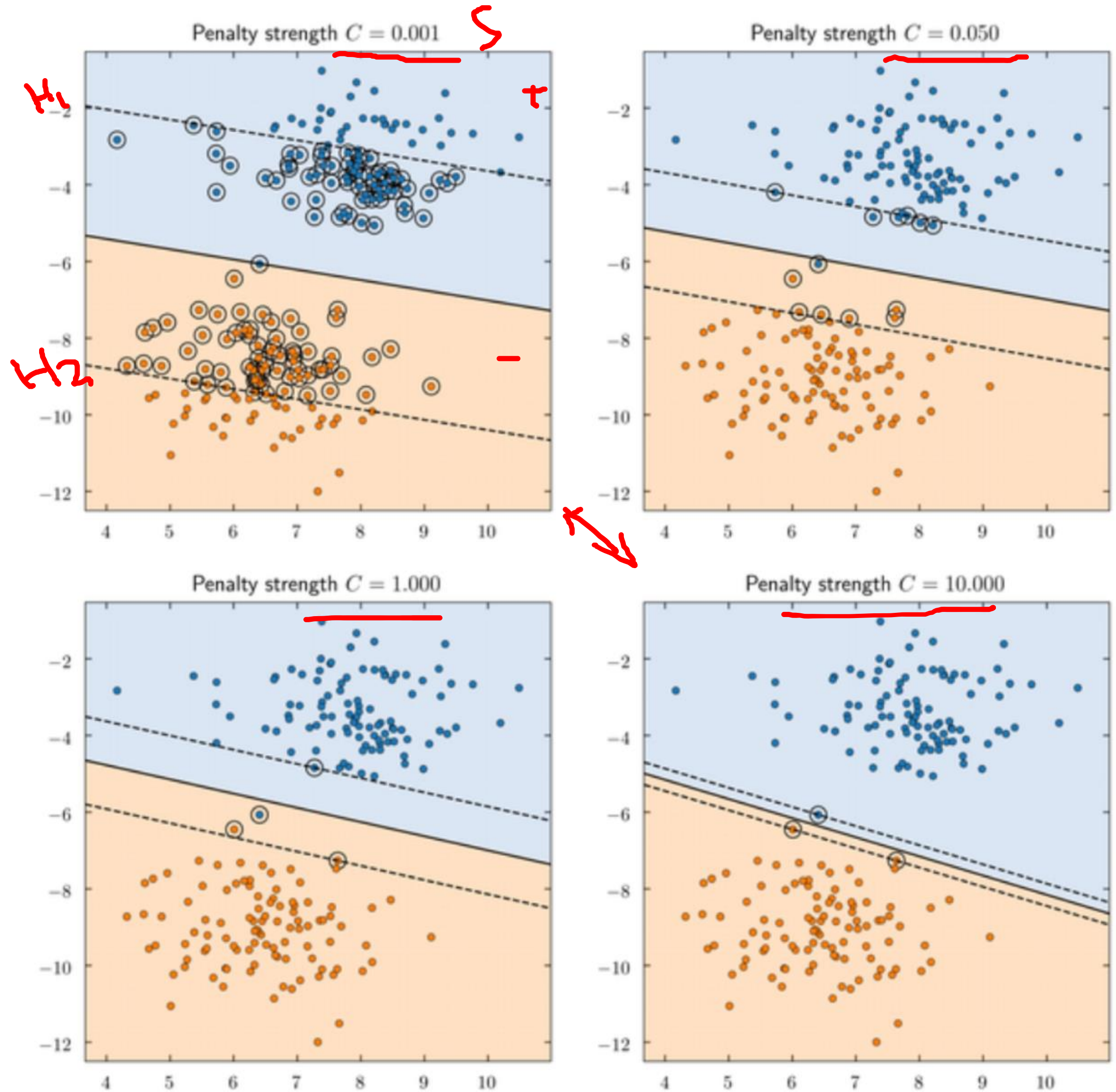
- We allow data points to be on the "wrong side" of the margin boundary
- Penalize points on the wrong side according to its distance to the margin boundary
- ξ: slack variable
- C (> 0): Controls the trade-off between the penalty and the margin
  - Smaller C: allow more mistake
  - Larger C: allow less mistake
- This is the widely used soft-margin SVM

larger margin

smaller margin

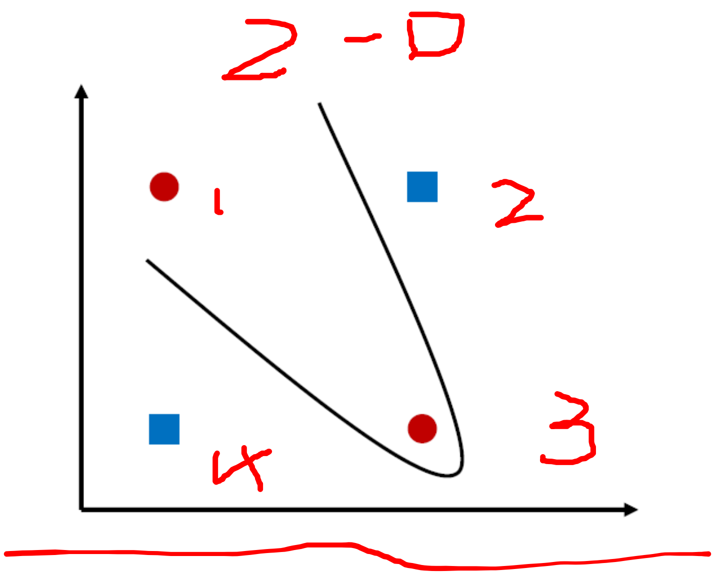


Effect of slack variable

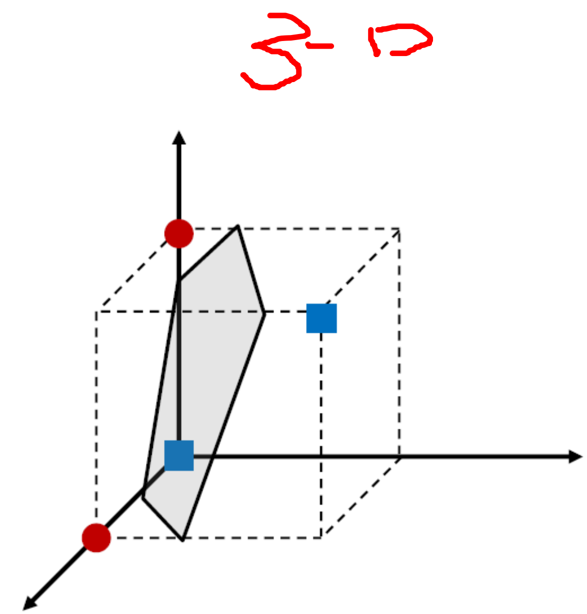


SVM for Linearly Inseparable Data

- Alternatively, for linearly inseparable data, we can map them to a higher dimensional space
- We search for a linear separating hyperplane in the new space
  - Example: The XOR problem



$$\boldsymbol{x} \longmapsto \phi(\boldsymbol{x})$$



Kernel Functions for Nonlinear Classification

L→H

- Instead of computing the dot product on the transformed data, it is mathematically equivalent to applying a kernel function $K(x_i, x_j)$ to the original data, i.e.,

$$K(x_i, x_j) = \phi(x_i)\phi(x_j)$$

- Typical Kernel Functions

Polynomial kernel of degree h : $K(X_i, X_j) = (X_i \cdot X_j + 1)^h$

Gaussian radial basis function kernel : $K(X_i, X_j) = e^{-\|X_i - X_j\|^2 / 2\sigma^2}$

Sigmoid kernel : $K(X_i, X_j) = \tanh(\kappa X_i \cdot X_j - \delta)$

- SVMs can efficiently perform a non-linear classification using kernel functions, implicitly mapping their inputs into high-dimensional feature spaces
  - https://www.youtube.com/watch?time_continue=42&v=3liCbRZPrZA
  - http://crsouza.com/2010/03/17/kernel-functions-for-machine-learning-applications/

SVM Summary

- Pros
  - Elegant mathematical formulation, guaranteed global optimal with optimization
  - Trains well on small or medium data sets
  - Flexibility through kernel functions
  - Conformity with semi-supervised training
- Cons
  - Not naturally scalable to large data sets

SVM Related Links

- SVM Website: http://www.kernel-machines.org/
- Representative implementations
  - LIBSVM: an efficient implementation of SVM, multi-class classifications, nu-SVM, one-class SVM, including also various interfaces with java, python, etc.
  - SVM-light: simpler but performance is not better than LIBSVM, support only binary classification and only in C
  - SVM-torch: another recent implementation also written in C

Lazy Learner

Lazy vs. Eager Learning

- Lazy vs. eager learning
  - Lazy learning (e.g., instance-based learning): Simply stores training data (or only minor processing) and waits until it is given a test tuple
  - Eager learning (the above discussed methods): Given a set of training tuples, constructs a classification model before receiving new (e.g., test) data to classify
- Lazy: less time in training but more time in predicting
- Accuracy
  - Lazy method effectively uses a richer hypothesis space since it uses many local linear functions to form an implicit global approximation to the target function
  - Eager: must commit to a single hypothesis that covers the entire instance space

Lazy Learner: Instance-Based Methods

- Instance-based learning:
  - Store training examples and delay the processing ("lazy evaluation") until a new instance must be classified
- Typical approaches
  - k-nearest neighbor approach (KNN)
    - Instances represented as points in a Euclidean space.
  - Locally weighted regression (KNN for regression)
    - Constructs local approximation
  - Case-based reasoning
    - Uses symbolic representations and knowledge-based inference

The k-Nearest Neighbor Algorithm

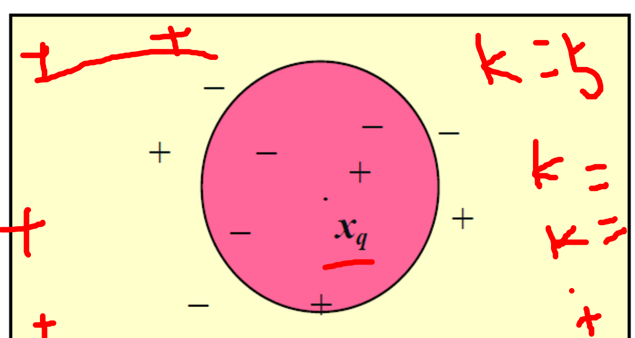
inputs

- All instances correspond to points in the n-D space
- The nearest neighbor are defined in terms of Euclidean distance, dist(X_1, X_2)
- Target function could be discrete- or real- valued
- For discrete-valued, k-NN returns the most common value among the k training examples nearest to x_q
- Vonoroi diagram: the decision surface induced by 1-NN for a typical set of training examples

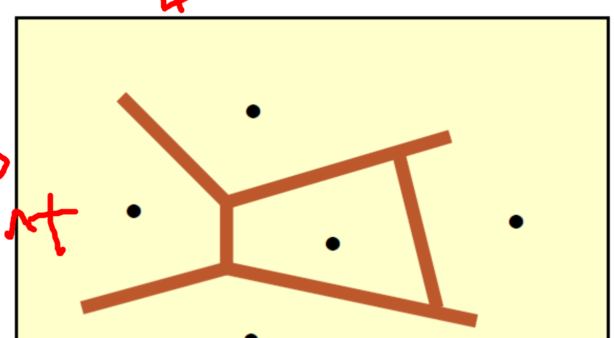
x_q

6−
4+

3−
2+



k=5
k=10
k=int



Discussion on the k-NN Algorithm

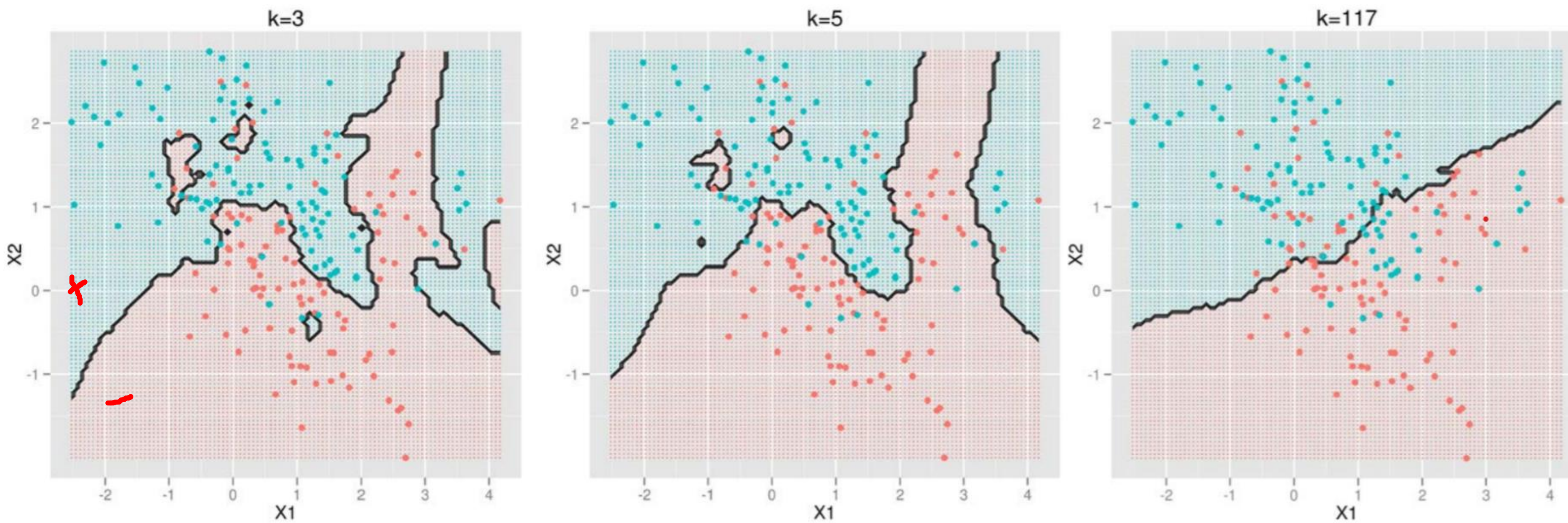
- k-NN for real-valued prediction for a given unknown tuple
  - Returns the mean values of the k nearest neighbors
- Distance-weighted nearest neighbor algorithm
  - Weight the contribution of each of the k neighbors according to their distance to the query x_q
  - Give greater weight to closer neighbors

$$w = \frac{1}{d(x_q, x_i)^2}$$

- Robust to noisy data by averaging k-nearest neighbors
- Curse of dimensionality: distance between neighbors could be dominated by irrelevant attributes
  - To overcome it, axes stretch or elimination of the least relevant attributes

Selection of k for kNN

- The number of neighbors k
  - Small k: overfitting (high var., low bias)
  - Big k: bringing too many irrelevant points (high bias, low var)



http://scott.fortmann-roe.com/docs/BiasVariance.html

Case-Based Reasoning (CBR)

- **CBR:** Uses a database of problem solutions to solve new problems
- Store symbolic description (tuples or cases)—not points in a Euclidean space
- **Applications:** Customer-service (product-related diagnosis), legal ruling
- **Methodology**
  - Instances represented by rich symbolic descriptions (e.g., function graphs)
  - Search for similar cases, multiple retrieved cases may be combined
  - Tight coupling between case retrieval, knowledge-based reasoning, and problem solving
- **Challenges**
  - Find a good similarity metric
  - Indexing based on syntactic similarity measure, and when failure, backtracking, and adapting to additional cases

Weakly Supervised Learning

Weakly Supervised Learning

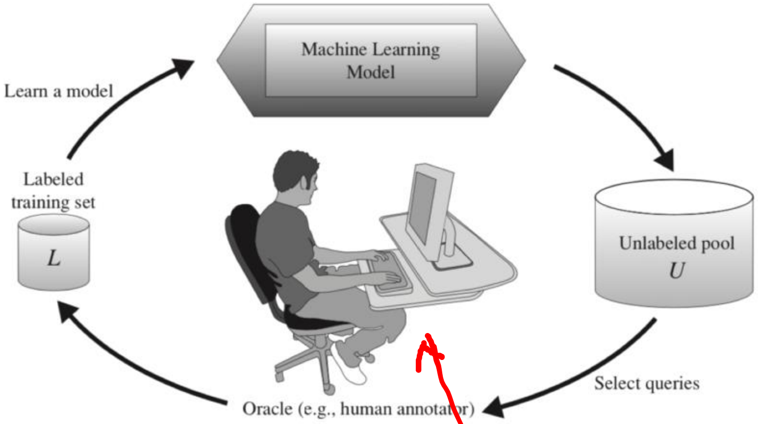
- Semi-supervised learning
- Active learning
- Transfer learning
- Distant supervision
- Zero-shot learning

Semi-supervised Learning

- Goal: uses labeled data and unlabeled data to build a classifier
- Self-training
  1. Select a learning method such as, say, Bayesian classification. Build the classifier using the labeled data, X_l.
  2. Use the classifier to label the unlabeled data, X_u.
  3. Select the tuple $x \in X_u$ having the highest confidence (most confident prediction). Add it and its predicted label to X_l.
  4. Repeat (i.e., retrain the classifier using the augmented set of labeled data).
- Co-training
  1. Define two separate nonoverlapping feature sets for the labeled data, X_l.
  2. Train two classifiers, f_1 and f_2, on the labeled data, where f_1 is trained using one of the feature sets and f_2 is trained using the other.
  3. Classify X_u with f_1 and f_2 separately.
  4. Add the most confident $(x, f_1(x))$ to the set of labeled data used by f_2, where $x \in X_u$. Similarly, add the most confident $(x, f_2(x))$ to the set of labeled data used by f_1.
  5. Repeat.
- When does SSL work? Clustering assumption vs. manifold assumption

Active Learning

- **Goal**: find 'the best' unlabeled data samples to ask the oracle for their labels, so as to maximally improve the classification performance.
- **Pool-based active learning**



- Key: how to choose the data tuples to be queried
  - Uncertainty sampling
  - Query-by-committee
  - Version space
  - Decision-theoretic appr

Transfer Learning

- **Goal**: aims to extract the knowledge from one or more source tasks and apply the knowledge to a target task
- **An example**
  - Source task: review sentiment classification on electronics
  - Target task: review sentiment classification on movies

rich L

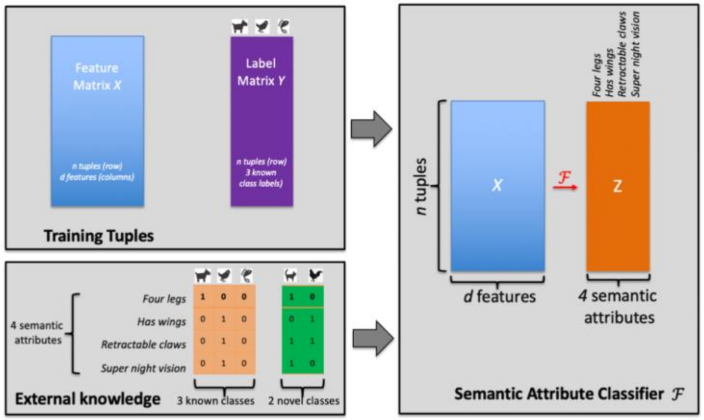
limited L

- **Instance-based transfer learning**: reweight some of the data from the source task and uses it to learn the target task.
  - Intuition: `transfer' most relevant/similar data from source to target
- **Key challenge**: negative transfer
- **Related problems**
  - Multi-task learning; pre-training+fine-tuning

Distant Supervision

- **Goal**: automatically generate a large number of labelled tuples.
- **Characteristic of the generated labels**
  - Noisy, but large in volume
- **Example #1**: Twitter sentiment classification (positive vs. negative)
  - if a tweet contains :-), treat it as a positive tuple
  - If a tween contains :-(, treat it as a negative tuple
- **Example #2**: Twitter category classification (e.g., news, health, science, games, etc.)
  - If a tweet contains a URL, use the ODP (open directory project) category of the URL as the label of the tweet
  - If a tween contains a Youtube video link, treat the label of Youtube video as the label of the tweet

Zero-shot Learning



- **A motivating example:**
  - A trained classifier to classify an animal image into owl vs. dog vs. fish.
  - But, the test image is actually about cat. novel class
- **Goal:** predict a test tuple whose class label was never observed during the training stage
- **General strategies:** leverage external knowledge or side information
- **Semantic attribute classifier**
  - Use the training tuples to build semantic attribute classifier
  - Use the semantic attribute classifier to infer semantic attributes
  - Use the semantic attributes to predict the novel class

Summary

model independent

- Feature Selection
  - Filter methods, wrapper methods and embedded methods
- Support Vector Machines
  - Large margin approach, Kernel tricks
- Lazy Learner
  - KNN, case-based reasoning
- Weakly Supervised Learning
  - Semi-supervised learning, Active learning, Transfer learning, Distant supervision, Zero-shot learning