Classification

Images are 28 x 28 pixels

Represent input image as a vector $x \in \mathbb{R}^{784}$

Learn a classifier $f(x)$ such that,

$$f : x \rightarrow \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$
K-nearest neighbor classification

Algorithm

- For each test point, \( x \), to be classified, find the \( K \) nearest samples in the training data
- Classify the point, \( x \), according to the majority vote of their class labels

\[ \text{e.g. } K = 3 \]

- applicable to multi-class case
Distance functions

Euclidean: \[ \sqrt{\sum_{i=1}^{k} (x_i - y_i)^2} \]

Manhattan: \[ \sum_{i=1}^{k} |x_i - y_i| \]

Minkowski: \[ \left( \sum_{i=1}^{k} (|x_i - y_i|^q)^{1/q} \right) \]
Choice of $k$
Choice of $k$
Binary kNN Classification Training Set

Binary kNN Classification (k=1)

Binary kNN Classification (k=5)

Binary kNN Classification (k=25)
Leave-one-out cross validation

- For a dataset with N examples, perform N experiments
- For each experiment use N-1 examples for training and the remaining example for testing
K-fold cross validation

- For each of K experiments, use K-1 folds for training and the remaining one for testing.

Classification Error = Average classification error on K folds
Linear Classification
Linear separability

- linearly separable
  - Example 1
  - Example 2

- not linearly separable
  - Example 1
  - Example 2
Inseparability

• Real world problems: there may not exist a hyperplane that separates cleanly

• Solution to this “inseparability” problem: map data to higher dimensional space
  • Called the “feature space”, as opposed to the original “input space”
  • Inseparable training set can be made separable with proper choice of feature space
Feature map

\[(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2} x_1 x_2, x_2^2)\]
A linear classifier has the form

\[ f(x) = w^\top x + b \]

- in 2D the discriminant is a line
- \( w \) is the normal to the line, and \( b \) the bias
- \( w \) is known as the weight vector
A linear classifier has the form

\[ f(x) = w^\top x + b \]

- in 3D the discriminant is a plane, and in nD it is a hyperplane

For a K-NN classifier it was necessary to `carry’ the training data
For a linear classifier, the training data is used to learn \( w \) and then discarded
Only \( w \) is needed for classifying new data
Good and bad linear classifiers

- **maximum margin** solution: most stable under perturbations of the inputs
Support Vector Machine

Two popular implementations

SVM\textsuperscript{light}

Support Vector Machine

Author: Thorsten Joachims <thorsten@joachims.org>
Cornell University
Department of Computer Science

LIBSVM -- A Library for Support Vector Machines
Chih-Chung Chang and Chih-Jen Lin
Margin

linearly separable data

Margin = \frac{2}{||w||}

w^T x + b = 1

w^T x + b = 0

w^T x + b = -1

Support Vector

Support Vector
Margin

linearly separable data

Margin = \frac{2}{||w||}

\begin{align*}
    w^T x + b &= 1 \\
    w^T x + b &= 0 \\
    w^T x + b &= -1
\end{align*}

Support Vector

Support Vector
Linear Support Vector Machine

- Learning the SVM can be formulated as an optimization:
  \[
  \max_w \frac{2}{||w||} \quad \text{subject to } \quad w^T x_i + b \begin{cases} 
  \geq 1 & \text{if } y_i = +1 \\
  \leq -1 & \text{if } y_i = -1
  \end{cases} \quad \text{for } i = 1 \ldots N
  \]

- Or equivalently
  \[
  \min_w ||w||^2 \quad \text{subject to } \quad y_i (w^T x_i + b) \geq 1 \quad \text{for } i = 1 \ldots N
  \]

- This is a quadratic optimization problem subject to linear constraints and there is a unique minimum
Inseparable case

\[ \xi_i \geq 0 \]

- for \( 0 < \xi \leq 1 \) point is between margin and correct side of hyperplane. This is a margin violation
- for \( \xi > 1 \) point is misclassified

\[ \frac{\xi_i}{||w||} > \frac{2}{||w||} \]

Margin = \[ \frac{2}{||w||} \]

\[ w^T x + b = 1 \]

\[ w^T x + b = 0 \]

\[ w^T x + b = -1 \]
The optimization problem becomes

$$\min_{\mathbf{w} \in \mathbb{R}^d, \xi_i \in \mathbb{R^+}} \|\mathbf{w}\|^2 + C \sum_{i=1}^{N} \xi_i$$

subject to

$$y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i$$ for $i = 1 \ldots N$

- Every constraint can be satisfied if $\xi_i$ is sufficiently large
- $C$ is a **regularization** parameter:
  - small $C$ allows constraints to be easily ignored $\rightarrow$ large margin
  - large $C$ makes constraints hard to ignore $\rightarrow$ narrow margin
  - $C = \infty$ enforces all constraints: hard margin
- This is still a quadratic optimization problem and there is a unique minimum. Note, there is only one parameter, $C$. 
Classification vs Regression

Classification

Regression

Discrete

Continuous
\[
f(x_1, x_2) = \begin{cases} 
\text{blue} & \text{or} \\
\text{red} & \end{cases}
\]

\[
y = f(x) = 10.5
\]
Protein structure prediction as regression

Regression task: given sequence predict 3D structure

3D coordinates and angles
Stock price prediction

- Task is to predict stock price at future date
- This is a regression task, as the output is continuous
Linear regression
Nonlinear regression
When does linear regression work?
K nearest neighbor regression

Algorithm

- For each test point, $x$, find the $K$ nearest samples $x_i$ in the training data and their values $y_i$
- Output is mean of their values \( f(x) = \frac{1}{K} \sum_{i=1}^{K} y_i \)
- Again, need to choose (learn) $K$
Nearest neighbor regression

\[ Y(X) \]

\[ \hat{Y}(X_0) \]

\[ \hat{Y}(X) \]

\[ m=16 \]
Nearest neighbor regression
Filling patches in images
Linear regression

- Linear regression is a simple approach to supervised learning. It assumes that the dependence of $Y$ on $X_1, X_2, \ldots, X_p$ is linear.
- True regression functions are never linear!

- although it may seem overly simplistic, linear regression is extremely useful both conceptually and practically.
How to measure the accuracy of linear regression models

\[ \hat{y} = \frac{1}{2} x - 1 \]
Linear Regression

\[ y_i = x_i^T \beta + \epsilon_i = \sum_j \beta_j X_{i,j} + \epsilon_i \]

Fitting error: \( \epsilon_i \)
Assumption: errors are Gaussian noises

\[ y = X\beta + \epsilon \]

\[ \beta^* = \arg\min_\beta \sum_{i} (y_i - \sum_j \beta_j X_{i,j})^2 \]
Linear Regression

\[ \beta^* = \arg \min_{\beta} \sum_i (y_i - \sum_j \beta_j X_{i,j})^2 \]

\[ = \arg \min_{\beta} (y - X\beta)^T (y - X\beta) \]

\[ = (X^T X)^{-1} X^T y \]

Question: How to derive the closed-form solution?
Clustering
Finding hidden structure in data
Expression analysis

Blood

Brain

Liver
Single-cell expression analysis
Clustering: examples

Image segmentation
Goal: Break up the image into meaningful or perceptually similar regions
Network clustering
Clustering

- **Basic idea:** group together similar instances
- **Example:** 2D point patterns
Clustering

- **Basic idea:** group together similar instances
- **Example:** 2D point patterns

What could “similar” mean?
- One option: small Euclidean distance (squared)
  \[
  \text{dist}(\vec{x}, \vec{y}) = ||\vec{x} - \vec{y}||^2_2
  \]
- Clustering results are crucially dependent on the measure of similarity (or distance) between “points” to be clustered
Given: $N$ unlabeled examples $\{x_1, \ldots, x_N\}$; the number of partitions $K$

Goal: Group the examples into $K$ partitions

The only information clustering uses is the similarity between examples

Clustering groups examples based on their mutual similarities
Clustering algorithms

1. **Flat or Partitional clustering** (K-means, Gaussian mixture models, etc.)
   - Partitions are independent of each other

2. **Hierarchical clustering** (e.g., agglomerative clustering, divisive clustering)
   - Partitions can be visualized using a tree structure (a dendrogram)
   - Does not need the number of clusters as input
   - Possible to view partitions at different levels of granularities (i.e., can refine/coarsen clusters) using different $K$
K-means

- **Input:** $N$ examples $\{x_1, \ldots, x_N\}$ ($x_n \in \mathbb{R}^D$); the number of partitions $K$
- **Initialize:** $K$ cluster centers $\mu_1, \ldots, \mu_K$. Several initialization options:
  - Randomly initialized anywhere in $\mathbb{R}^D$
  - Choose any $K$ examples as the cluster centers
- **Iterate:**
  - Assign each of example $x_n$ to its closest cluster center
    \[
    C_k = \{ n : \ k = \text{arg min}_k \| x_n - \mu_k \|^2 \}
    \]
    ($C_k$ is the set of examples closest to $\mu_k$)
  - Recompute the new cluster centers $\mu_k$ (mean/centroid of the set $C_k$)
    \[
    \mu_k = \frac{1}{|C_k|} \sum_{n \in C_k} x_n
    \]
  - Repeat while not converged
K-means for segmentation
When will K-means fail?

Non-convex/non-round-shaped clusters: Standard $K$-means fails!

Clusters with different densities
Hierarchical clustering

A hierarchical approach can be useful when considering versatile cluster shapes:

10-means

By first detecting many small clusters, and then merging them, we can uncover patterns that are challenging for partitional methods.
Agglomerative clustering

- **Agglomerative clustering:**
  - First merge very similar instances
  - Incrementally build larger clusters out of smaller clusters

- **Algorithm:**
  - Maintain a set of clusters
  - Initially, each instance in its own cluster
  - Repeat:
    - Pick the two closest clusters
    - Merge them into a new cluster
    - Stop when there’s only one cluster left

- Produces not one clustering, but a family of clusterings represented by a **dendrogram**
We need a notion of similarity between clusters.
Single linkage uses the minimum distance.
Example
Complete linkage uses the maximum distance.
Group average linkage uses the average distance between groups.
Example
Mouse tumor data from [Hastie et al.]
Application to breast cancer expression data