Machine Learning: Introduction and Unsupervised Learning

Chapter 18.1, 18.2, 18.8.1
and “Introduction to Statistical Machine Learning”

What is Learning?

- “Learning is making useful changes in our minds” – Marvin Minsky
- “Learning is constructing or modifying representations of what is being experienced” – Ryszard Michalski
- “Learning denotes changes in a system that ... enable a system to do the same task more efficiently the next time” – Herbert Simon

Why do Machine Learning?

- Solve classification problems
- Learn models of data (“data fitting”)
- Understand and improve efficiency of human learning (e.g., Computer-Aided Instruction (CAI))
- Discover new things or structures that are unknown to humans (“data mining”)
- Fill in skeletal or incomplete specifications about a domain

Major Paradigms of Machine Learning

- Rote Learning
- Induction
- Clustering
- Analogy
- Discovery
- Genetic Algorithms
- Reinforcement
Inductive Learning

• Generalize from a given set of (training) examples so that accurate predictions can be made about future examples
• Learn unknown function: \( f(x) = y \)
  – \( x \): an input example (aka instance)
  – \( y \): the desired output
  – \( h \) (hypothesis) function is learned that approximates \( f \)

Representing “Things” in Machine Learning

• An example or instance, \( x \), represents a specific object (“thing”)
• \( x \) often represented by a \( D \)-dimensional feature vector \( x = (x_1, \ldots, x_D) \in \mathbb{R}^D \)
• Each dimension is called a feature or attribute
• Continuous or discrete
• \( x \) is a point in the \( D \)-dimensional feature space
• Abstraction of object. Ignores all other aspects (e.g., two people having the same weight and height may be considered identical)

Feature Vector Representation

• Preprocess raw data
  – extract a feature (attribute) vector, \( x \), that describes all attributes relevant for an object
• Each \( x \) is a list of \( (attribute, value) \) pairs
  \( x = [(Rank, queen), (Suit, hearts), (Size, big)] \)
  – number of attributes is fixed: Rank, Suit, Size
  – number of possible values for each attribute is fixed (if discrete)
  \( \text{Rank:} \) 2, \( \ldots \), 10, jack, queen, king, ace
  \( \text{Suit:} \) diamonds, hearts, clubs, spades
  \( \text{Size:} \) big, small

Feature Vector Representation

Each example can be interpreted as a point in a \( D \)-dimensional feature space, where \( D \) is the number of features/attributes

![Graph showing suit and rank dimensions with example points labeled with attributes and values.]
Feature Vector Representation Example

- Text document
  - Vocabulary of size D (~100,000): aardvark, ..., zulu
- “bag of words”: counts of each vocabulary entry
  - To marry my true love \( \Rightarrow (3531:1 \ 13788:1 \ 19676:1) \)
  - I wish that I find my soulmate this year \( \Rightarrow (3819:1 \ 13448:1 \ 19450:1 \ 20514:1) \)
- Often remove “stopwords”: the, of, at, in, ...
- Special “out-of-vocabulary” (OOV) entry catches all unknown words

More Feature Representations

- Image
  - Color histogram
- Software
  - Execution profile: the number of times each line is executed
- Bank account
  - Credit rating, balance, #deposits in last day, week, month, year, #withdrawals, ...
- Bioinformatics
  - Medical test1, test2, test3, ...

Training Sample

- A training sample is a collection of examples (aka instances), \( x_1, \ldots, x_n \), which is the input to the learning process
- \( x_i = (x_{i1}, \ldots, x_{iD}) \)
- Assume these instances are all sampled independently from the same, unknown (population) distribution, \( P(x) \)
- We denote this by \( x_i \overset{\text{i.i.d.}}{\sim} P(x) \), where i.i.d. stands for independent and identically distributed
- Example: Repeated throws of dice

Training Sample

- A training sample is the “experience” given to a learning algorithm
- What the algorithm can learn from it varies
- Two basic learning paradigms:
  - unsupervised learning
  - supervised learning
Inductive Learning

• **Supervised** vs. **Unsupervised** Learning
  – supervised: “teacher” gives a set of (x, y) pairs
  – unsupervised: only the x’s are given

• In either case, the goal is to estimate f so that it generalizes well to correctly deal with “future examples” in computing f(x) = y

Unsupervised Learning

• Training sample is x_1, ..., x_n, that’s it!
• No “teacher” providing supervision as to how individual examples should be handled
• Common tasks:
  – Clustering: separate the n examples into groups
  – Novelty detection: find examples that are very different from the rest
  – Dimensionality reduction: represent each example with a lower dimensional feature vector while maintaining key characteristics of the training samples

Clustering

• Goal: Group training sample into clusters such that examples in the same cluster are similar, and examples in different clusters are different
• How many clusters do you see?
• Many clustering algorithms

Oranges and Lemons

Google News

Digital Photo Collections

- You have 1000s of digital photos stored in various folders
- Organize them better by grouping into clusters
  - Simplest idea: use image creation time (EXIF tag)
  - More complicated: extract image features

Detecting Events on Twitter

- Use real-time text and images from tweets to discover new social events
- Clusters defined by similar words and word cooccurrences, plus image features

Three Frequently Used Clustering Methods

- Hierarchical Agglomerative Clustering
  - Build a binary tree over the dataset
- K-Means Clustering
  - Specify the desired number of clusters and use an iterative algorithm to find them
- Mean Shift Clustering
Hierarchical Clustering

- Initially every point is in its own cluster

Hierarchical Clustering

- Find the pair of clusters that are the closest

Hierarchical Clustering

- Merge the two into a single cluster

Hierarchical Clustering

- Repeat ...

...
Hierarchical Clustering

- Repeat ...

Hierarchical Clustering

- Repeat ... until the whole dataset is one giant cluster
- You get a binary tree (not shown here)

Hierarchical Clustering Algorithm

Input: a training sample \( \{x_i\}_{i=1}^n \); a distance function \( d() \).

1. Initially, place each instance in its own cluster (called a singleton cluster).
2. while (number of clusters > 1) do:
3.   Find the closest cluster pair \( A, B \), i.e., they minimize \( d(A, B) \).
4.   Merge \( A, B \) to form a new cluster.

Output: a binary tree showing how clusters are gradually merged from singletons to a root cluster, which contains the whole training sample.

Example: Histogram-Based Image Segmentation

- Goal: Segment the image into \( K \) regions
  - Reduce the number of gray levels to \( K \) and map each pixel to the closest gray level
Example: Histogram-Based Image Segmentation

- Goal: Segment the image into $K$ regions
  - Reduce the number of gray levels to $K$ and map each pixel to the closest gray level

Hierarchical Clustering

- How do you measure the closeness between two clusters?
  - At least three ways:
    - **Single-linkage**: the shortest distance from any member of one cluster to any member of the other cluster
    - **Complete-linkage**: the largest distance from any member of one cluster to any member of the other cluster
    - **Average-linkage**: you guessed it

Hierarchical Clustering

- The binary tree you get is often called a dendrogram, or taxonomy, or a hierarchy of data points
  - The tree can be cut at various levels to produce different numbers of clusters: if you want $k$ clusters, just cut the $(k-1)$ longest links
Hierarchical Clustering Example

- 6 Italian cities
- Single-linkage

---

Iteration 1: Merge MI and TO

<table>
<thead>
<tr>
<th></th>
<th>BA</th>
<th>FI</th>
<th>MI/TO</th>
<th>NA</th>
<th>RM</th>
</tr>
</thead>
<tbody>
<tr>
<td>BA</td>
<td>0</td>
<td>662</td>
<td>877</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FI</td>
<td>662</td>
<td>0</td>
<td>295</td>
<td>468</td>
<td>400</td>
</tr>
<tr>
<td>MI/TO</td>
<td>877</td>
<td>295</td>
<td>0</td>
<td>754</td>
<td>564</td>
</tr>
<tr>
<td>NA</td>
<td>255</td>
<td>468</td>
<td>754</td>
<td>0</td>
<td>219</td>
</tr>
<tr>
<td>RM</td>
<td>412</td>
<td>268</td>
<td>564</td>
<td>219</td>
<td>0</td>
</tr>
</tbody>
</table>

Recompute min distance from MI/TO cluster to all other cities

---

Iteration 2: Merge NA and RM

<table>
<thead>
<tr>
<th></th>
<th>BA</th>
<th>FI</th>
<th>MI/TO</th>
<th>NA/RM</th>
</tr>
</thead>
<tbody>
<tr>
<td>BA</td>
<td>0</td>
<td>662</td>
<td>877</td>
<td>255</td>
</tr>
<tr>
<td>FI</td>
<td>662</td>
<td>0</td>
<td>295</td>
<td>268</td>
</tr>
<tr>
<td>MI/TO</td>
<td>877</td>
<td>295</td>
<td>0</td>
<td>564</td>
</tr>
<tr>
<td>NA/RM</td>
<td>255</td>
<td>268</td>
<td>564</td>
<td>0</td>
</tr>
</tbody>
</table>

---

Iteration 3: Merge BA and NA/RM

<table>
<thead>
<tr>
<th></th>
<th>BA/NA/RM</th>
<th>FI</th>
<th>MI/TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>BA/NA/RM</td>
<td>0</td>
<td>268</td>
<td>564</td>
</tr>
<tr>
<td>FI</td>
<td>268</td>
<td>0</td>
<td>295</td>
</tr>
<tr>
<td>MI/TO</td>
<td>564</td>
<td>295</td>
<td>0</td>
</tr>
</tbody>
</table>

---

Example created by Matteo Matteucci
What are factors that affect the outcome of hierarchical clustering?

Hierarchical Clustering Applet

http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/AppletH.html
Three Frequently Used Clustering Methods

- **Hierarchical Agglomerative Clustering**
  - Build a binary tree over the dataset

- **K-Means Clustering**
  - Specify the desired number of clusters and use an iterative algorithm to find them

- **Mean Shift Clustering**

K-Means Clustering

- Suppose I tell you the cluster centers, \( c_i \)

  - Q: How to determine which points to associate with each \( c_i \)?
  - A: For each point \( x \), choose closest \( c_i \)

- Suppose I tell you the points in each cluster

  - Q: How to determine the cluster centers?
  - A: Choose \( c_i \) to be the mean / centroid of all points in the cluster

K-Means Clustering

- Randomly pick 5 positions as initial cluster centers (not necessarily data points)
K-Means Clustering

• Each point finds which cluster center it is closest to; the point belongs to that cluster

K-Means Clustering

• Each cluster computes its new centroid based on which points belong to it

• Repeat until convergence (i.e., no cluster center moves)

K-Means: Initial Cluster Centers
K-Means in Action
K-Means Stops

K-Means Demo

- [http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/AppletKM.html](http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/AppletKM.html)

K-Means Algorithm

- Input: \( x_1, \ldots, x_n, k \)
- **Step 1**: select \( k \) cluster centers \( c_1, \ldots, c_k \)
- **Step 2**: for each point \( x_n \), determine its cluster: find the closest center in Euclidean space
- **Step 3**: update all cluster centers as the centroids

\[
  c_i = \frac{1}{\text{SizeOf(cluster } i)} \sum_{x \in \text{cluster } i} x
\]

- Repeat steps 2 and 3 until cluster centers no longer change

Example: Image Segmentation

- Input image
- Clusters on intensity
- Clusters on color
K-Means Questions

• Will it always terminate?
  – Yes (finite number of ways of partitioning a finite number of points into \( k \) groups)
• Is it guaranteed to find an “optimal” clustering?
  – No, but each iteration will reduce the error/distortion of the clustering

Non-Optimal Clustering

Say \( k=3 \) and you are given the following points:

Non-Optimal Clustering

Given a poor choice of the initial cluster centers, the following result is possible:

Picking Starting Cluster Centers

Which local optimum \( k \)-Means goes to is determined solely by the starting cluster centers

– Idea 1: Run \( k \)-Means multiple times with different starting, random cluster centers (hill climbing with random restarts)
– Idea 2: Pick a random point \( x_1 \) from the dataset
  1. Find the point \( x_2 \) farthest from \( x_1 \) in the dataset
  2. Find \( x_3 \) farthest from the closer of \( x_1, x_2 \)
  3. ... Pick \( k \) points like this, and use them as the starting cluster centers for the \( k \) clusters
Picking the Number of Clusters

- Difficult problem
- Heuristic approaches depend on number of points and number of dimensions

Uses of $K$-Means

- Often used as an exploratory data analysis tool
- In one-dimension, a good way to quantize real-valued variables into $k$ non-uniform buckets
- Used on acoustic data in speech recognition to convert waveforms into one of $k$ categories (known as Vector Quantization)
- Also used for choosing color palettes on graphical display devices

Three Frequently Used Clustering Methods

- Hierarchical Agglomerative Clustering
  - Build a binary tree over the dataset

- $K$-Means Clustering
  - Specify the desired number of clusters and use an iterative algorithm to find them

- Mean Shift Clustering

  1. Choose a search window size
  2. Choose the initial location of the search window
  3. Compute the mean location (centroid of the data) in the search window
  4. Center the search window at the mean location computed in Step 3
  5. Repeat Steps 3 and 4 until convergence

The mean shift algorithm seeks the mode, i.e., point of highest density of a data distribution:
**Intuitive Description**

**Objective**: Find the densest region

Distribution of identical billiards balls

- **Region of interest**
- **Center of mass**
- **Mean Shift vector**
**Objective**: Find the densest region
distribution of identical billiard balls

**Intuitive Description**

Region of interest
Center of mass
Mean Shift vector

**Results**

Feature space is only gray level
http://www.caip.rutgers.edu/~comanici/MSPAMI/msPamiResults.html
Supervised Learning

- A labeled training sample is a collection of examples: \((x_1, y_1), \ldots, (x_n, y_n)\)

- Assume \((x, y) \sim P(x, y)\) and \(P(x, y)\) is unknown

- **Supervised learning** learns a function \(h: x \rightarrow y\) in some function family, \(H\), such that \(h(x)\) predicts the true label \(y\) on future data, \(x\), where \((x, y) \sim P(x, y)\)
  - **Classification**: if \(y\) discrete
  - **Regression**: if \(y\) continuous
Labels

• Examples
  – Predict gender (M, F) from weight, height
  – Predict adult, juvenile (A, J) from weight, height
• A label $y$ is the desired prediction on an instance $x$
• Discrete label: classes
  – M, F; A, J: often encode as 0, 1 or -1, 1
  – Multiple classes: 1, 2, 3, ..., C. No class order implied.
• Continuous label: e.g., blood pressure

Concept Learning

• Determine from a given a set of examples if a given example is or is not an instance of the concept/class/category
  – If it is, call it a positive example
  – If not, called it a negative example

Example: Mushroom Classification

Mushroom Features/Attributes

1. cap-shape: bell=b, conical=c, convex=x, flat=f, knobbed=k, sunken=s
2. cap-surface: fibrous=f, grooves=g, scaly=y, smooth=s
3. cap-color: brown=n, buff=b, cinnamon=c, gray=g, green=r, pink=p, purple=u, red=e, white=w, yellow=y
4. bruises?: bruises=t, no=f
5. odor: almond=a, anise=l, creosote=c, fishy=y, foul=f, musty=m, none=n, pungent=p, spicy=s
6. gill-attachment: attached=a, descending=d, free=f, notched=n
7. ...

**Classes:** edible=e, poisonous=p

http://www.usask.ca/biology/fungi/
Supervised Concept Learning by Induction

- Given a **training set** of positive and negative examples of a concept:
  \[- \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}\]
  where each \(y_i\) is either + or -
- Construct a description that accurately classifies whether **future examples** are positive or negative:
  \[- h(x_{n+1}) = y_{n+1}\]
  where \(y_{n+1}\) is the + or - prediction

Inductive Learning by Nearest-Neighbor Classification

- A simple approach:
  - save each training example as a point in Feature Space
  - classify a new example by giving it the same classification as its **nearest neighbor** in Feature Space

**k-Nearest-Neighbor (k-NN)**

*Input:* Training data \((x_1, y_1), \ldots, (x_n, y_n)\); distance function \(d()\); number of neighbors \(k\); test instance \(x^*\)

1. Find the \(k\) training instances \(x_1, \ldots, x_k\) closest to \(x^*\) under distance \(d()\).
2. Output \(y^*\) as the majority class of \(y_1, \ldots, y_k\). Break ties randomly.

- **1-NN:**

  ![Decision boundary](image)

  (a) classification by gender  
  (b) classification by age

**k-NN**

- What if we want regression?
  - Instead of majority vote, take **average** of neighbors’ \(y\)

- How to pick \(k\)?
  - Split data into training and tuning sets
  - Classify tuning set with different \(k\) values
  - Pick \(k\) that produces least tuning-set error
**k-NN**

Doesn't generalize well if the examples in each class are not well "clustered"

---

**Inductive Bias**

- Inductive learning is an inherently conjectural process. Why?
  - any knowledge created by generalization from specific facts cannot be proven true
  - it can only be proven false

- Hence, inductive inference is “falsity preserving,” not “truth preserving”

---

**Inductive Bias**

- Learning can be viewed as searching the Hypothesis Space $H$ of possible $h$ functions
- Inductive Bias
  - is used when one $h$ is chosen over another
  - is needed to generalize beyond the specific training examples
- Completely unbiased inductive algorithm
  - only memorizes training examples
  - can't predict anything about unseen examples

---

**k-NN Demo**

- [http://www.cs.cmu.edu/~zhuxj/courseproject/knndemo/KNN.html](http://www.cs.cmu.edu/~zhuxj/courseproject/knndemo/KNN.html)
Inductive Bias

Biases commonly used in machine learning:

- **Restricted Hypothesis Space Bias:**
  allow only certain types of $h$’s, not arbitrary ones

- **Preference Bias:**
  define a metric for comparing $h$’s so as to determine whether one is better than another