Machine Learning (CSE 446): Unsupervised Learning

Sham M Kakade

© 2018

University of Washington
cse446-staff@cs.washington.edu
Announcements

- HW2 posted. Due Feb 1.
  - It is long. Start this week!
- Today:
  Review: the perceptron algo New: Unsupervised learning
Review
Neuron-Inspired Classifier

\[ f(x) = \text{sign} (w \cdot x + b) \]

remembering that: \( w \cdot x = \sum_{j=1}^{d} w[j] \cdot x[j] \)

Learning requires us to set the weights \( w \) and the bias \( b \).

**Scalings:** Note that assuming \( \|x\| \leq 1 \) doesn’t change anything. Even with this scaling, the scale of \( \|w\| \) is arbitrary.
Perceptron Learning Algorithm

Data: $D = \langle (x_n, y_n) \rangle_{n=1}^{N}$, number of epochs $E$

Result: weights $w$ and bias $b$

initialize: $w = 0$ and $b = 0$;

for $e \in \{1, \ldots, E\}$ do
  for $n \in \{1, \ldots, N\}$, in random order do
    # predict
    $\hat{y} = \text{sign}(w \cdot x_n + b)$;
    if $\hat{y} \neq y_n$ then
      # update
      $w \leftarrow w + y_n \cdot x_n$;
      $b \leftarrow b + y_n$;
  end
end

return $w, b$

Algorithm 1: PerceptronTrain
Linear Decision Boundary

\[ w \cdot x + b = 0 \]

activation = \( w \cdot x + b \)
When does the perceptron not converge?
Linear Separability

A dataset \( D = \{(x_n, y_n)\}_{n=1}^N \) is **linearly separable** if there exists some linear classifier (defined by \( w, b \)) such that, for all \( n \), \( y_n = \text{sign}(w \cdot x_n + b) \).

If data are separable, (without loss of generality) can scale so that:

- “margin at 1”, can assume for all \( (x, y) \)
  \[ y(w \ast \cdot x) \geq 1 \]
  (let \( w \ast \) be smallest norm vector with margin 1).
- CIML: assumes \( ||w \ast|| \) is unit length and scales the ”1” above.
Linear Separability and the Geometric Margin
Theorem: Suppose data are scaled so that $\|x_i\|_2 \leq 1$.
Assume $D$ is linearly separable, and let be $w_*$ be a separator with “margin 1”. Then the perceptron algorithm will converge in at most $\|w_*\|^2$ epochs.

- Let $w_t$ be the param at “iteration” $t$; $w_0 = 0$
- “A Mistake Lemma”: At iteration $t$
  
  If we do not make a mistake, $\|w_{t+1} - w_*\|^2 = \|w_t - w_*\|^2$
  If we do make a mistake, $\|w_{t+1} - w_*\|^2 \leq \|w_t - w_*\|^2 - 1$

- The theorem directly follows from this lemma. Why?
Unsupervised Learning

The training dataset consists only of \( \langle x_n \rangle^{N}_{n=1} \).

There might, or might not, be labels.

Two simple unsupervised learning methods:
- cluster into \( K \) groups.
- project your data into less dimensions
- Today: look at these methods as objective function minimization.
$K$-Means: An Iterative Clustering Algorithm

(Review from last week.)
The stars are **cluster centers**, randomly assigned at first.
Assign each example to its nearest cluster center.
Recalculate cluster centers to reflect their respective examples.
Assign each example to its nearest cluster center.
Recalculate cluster centers to reflect their respective examples.
Assign each example to its nearest cluster center.
Recalculate cluster centers to reflect their respective examples.
At this point, nothing will change; we have converged.
**K-Means Clustering**

**Data:** unlabeled data \( D = \{x_n\}_{n=1}^N \), number of clusters \( K \)

**Result:** cluster assignment \( z_n \) for each \( x_n \)

initialize each \( \mu_k \) to a random location, for \( k \in \{1, \ldots, K\} \);

do

\[
\text{for } n \in \{1, \ldots, N\} \text{ do}
\]

\[
\text{# assign each data point to its nearest cluster-center let}
\]

\[
z_n = \arg\min_k \| \mu_k - x_n \|_2;
\]

\[
\text{end}
\]

\[
\text{for } k \in \{1, \ldots, K\} \text{ do}
\]

\[
\text{# recenter each cluster}
\]

\[
\text{let } X_k = \{x_n \mid z_n = k\};
\]

\[
\text{let } \mu_k = \text{mean}(X_k);
\]

\[
\text{end}
\]

\[
\text{while any } z_n \text{ changes from previous iteration};
\]

\[
\text{return } \{z_n\}_{n=1}^N;
\]

**Algorithm 2: K-Means**
Questions about $K$-Means

1. Does it converge?
   Yes.

2. Does it converge to the right answer?
What would we like to do?

- **Objective function:** find \(k\)-means, \(\mu_1, \ldots \mu_k\), which minimizes the following squared distance cost function:

\[
\sum_{n=1}^{N} \left( \min_{k' \in \{1, \ldots, k-1\}} \|x_n - \mu_{k'}\|^2 \right)
\]

- We can also write this objective function in terms of the assignments \(z_n\)'s. How?

**This is the general approach of loss function minimization:** find parameters which make our objection function “small” (and which also “generalizes”)
Convergence Proof Sketch

- The cluster assignments, the $z_n$'s take only finitely many values. So the cluster centers, the $\mu_k$'s, also must only take a finite number of values. Each time we update any of them, we will never increase this function:

$$L(z_1, \ldots, z_N, \mu_1, \ldots, \mu_K) = \sum_{n=1}^{N} \|x_n - \mu_{z_n}\|_2^2 \geq 0$$

$L$ is the **objective function** of $K$-Means clustering.

- Convergence must occur in a **finite number** of steps, due to: $L$ decreases at every step; $L$ can only take on finitely many values. See CIML, Chapter 15 for more details.

- Does the solution depend on the random initialization of the means $\mu_*$?
Does $K$-means converge to the minimal cost solution?

- No! The objective is an NP-Hard problem, so we can’t expect any algorithm to minimize the cost without essentially checking (near to) all assignments.
- Bad example for $K$-means:
Aside: Is NP-hardness a relevant concept for ML problems?

- Maybe the set of 'hard' problems may not be interesting.
A Heuristic for Initializing $K$-Means

**Data:** unlabeled data $D = \langle x_n \rangle_{n=1}^N$, number of clusters $K$

**Result:** initial points $\langle \mu_1, \ldots, \mu_K \rangle$

pick $n$ uniformly at random from $\{1, \ldots, N\}$ and let $\mu_1 = x_n$;

for $k \in \{2, \ldots, K\}$ do

# find the example that is furthest from all previously selected means

let $n = \text{argmax}_{n \in \{1, \ldots, N\}} \left( \min_{k' \in \{1, \ldots, k-1\}} \| x_n - \mu_{k'} \|_2^2 \right)$;

let $\mu_k = x_n$;

end

return $\langle \mu_1, \ldots, \mu_K \rangle$;

**Algorithm 3:** FurthestFirst ($K$-means++)
FurthestFirst in action
FurthestFirst in action
Some Comments

- $K$-means usually converges very quickly in practice.
- $K$-means++ still not guaranteed to find the global optima,
  - in practice, we can get stuck.
  - often try multiple initializations (use a little randomness in $K$-means++ and run the algorithm multiple times).
  - it does have ("multiplicative") approximation guarantees.
- How to choose $K$?
  - Information theory criterion (see CIML).
  - Based on 'good' function value decrease on 'holdout' set.

See CIML.
Recap: Unsupervised Learning

The training dataset consists only of $\langle x_n \rangle_{n=1}^N$.

There might, or might not, be labels.

Simplest kind of unsupervised learning: cluster into $K$ groups.

Second kind of unsupervised learning: dimensionality reduction.
- Useful for visualization.
- Also fight the curse of dimensionality.
Linear Dimensionality Reduction
Linear Dimensionality Reduction

As before, you only have a training dataset consisting of \( \langle x_n \rangle_{n=1}^{N} \).

Is there a way to represent each \( x_n \in \mathbb{R}^d \) as a lower-dimensional vector?

Why would we want to do this?
Assume that the data are \textit{centered}, i.e., that \( \text{mean} \left( \langle x_n \rangle_{n=1}^N \right) = 0 \).
Assume that the data are *centered*, i.e., that 
mean \((\langle x_n \rangle_{n=1}^N) = 0\).
Projection into One Dimension

Let \( \mathbf{u} \) be the dimension of greatest variance, and (without loss of generality) let \( \|\mathbf{u}\|_2^2 = 1 \).

\( p_n = \mathbf{x}_n \cdot \mathbf{u} \) is the projection of the \( n \)th example onto \( \mathbf{u} \).

Since the mean of the data is 0, the mean of \( \langle p_1, \ldots, p_N \rangle \) is also 0.

This implies that the variance of \( \langle p_1, \ldots, p_N \rangle \) is \( \frac{1}{N} \sum_{n=1}^{N} p_n^2 \).

The \( \mathbf{u} \) that gives the greatest variance, then, is:

\[
\arg\max_{\mathbf{u}} \sum_{n=1}^{N} (\mathbf{x}_n \cdot \mathbf{u})^2
\]
Projecting $x$ onto a vector $u$
Projecting $x$ onto an 'orthonormal' basis $u$