Machine Learning I: Fractal 2

Rajendra Nagar

Assistant Professor
Department of Electircal Engineering
Indian Institute of Technology Jodhpur
http://home.iitj.ac.in/∼rn/

These slides are prepared from the following book:
Shalev-Shwartz, Shai, and Shai Ben-David. Understanding machine learning: From theory to algorithms. Cambridge university press, 2014.

Gaussian Mixture Models

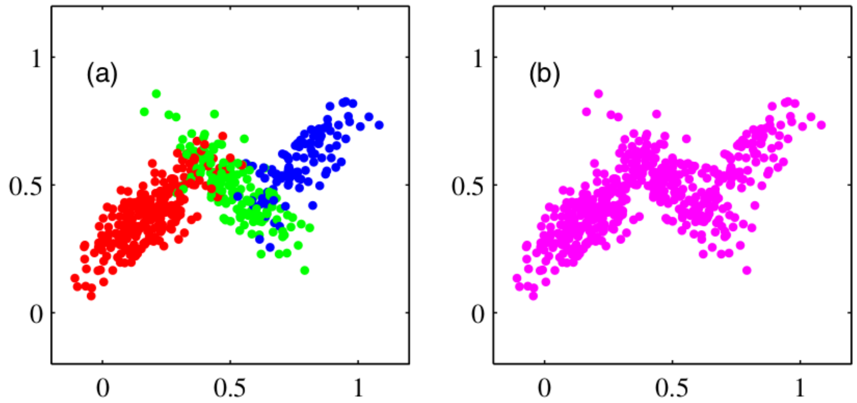


Image Source: Bishop, C. M. (2006). Pattern recognition and machine learning. springer.

Algorithm 1 Expectation Maximization

```
1: Input: $\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 & \cdots & \mathbf{x}_n \end{bmatrix} \in \mathbb{R}^{d \times n}$, where $p(\mathbf{x}) = \sum_{k \in [K]} \pi_k \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$.
2: Maximize log-likelihood: $\max_{\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}} \sum_{n \in [N]} \log \Big( \sum_{k \in [K]} \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \Big)$
3: Initialize: $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$, and $\pi_k$, $\forall k \in [K]$.
4: E step. Evaluate the responsibilities using the current parameter values.
       $\gamma(z_{nk}) \leftarrow \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}$.
5: M step. Re-estimate the parameters using the current responsibilities.
       $\boldsymbol{\mu}_k^{\text{new}} \leftarrow \frac{1}{N_k} \sum_{n \in [N]} \gamma(z_{nk}) \mathbf{x}_n$
       $\boldsymbol{\Sigma}_k^{\text{new}} \leftarrow \frac{1}{N_k} \sum_{n \in [N]} \gamma(z_{nk}) (\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}})(\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}})^\top$
       $\pi_k^{\text{new}} \leftarrow \frac{N_k}{N}$. Here, $N_k = \sum_{n \in [N]} \gamma(z_{nk})$.
```

Gaussian Mixture Models

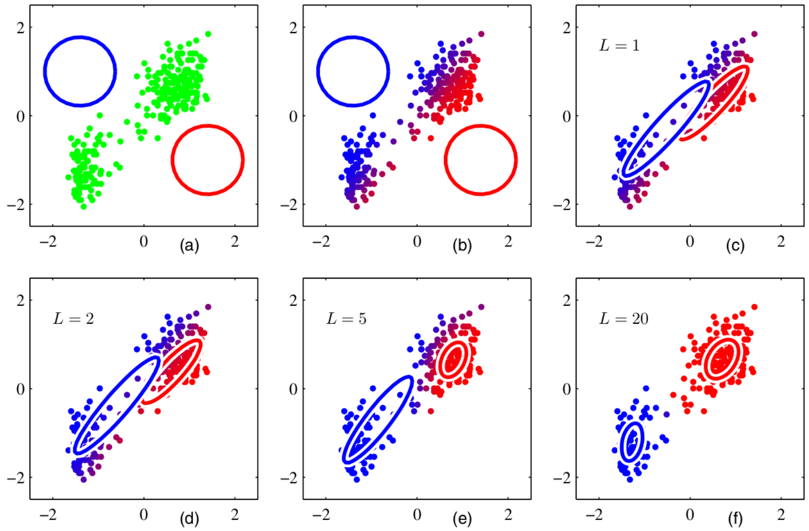
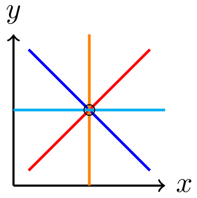
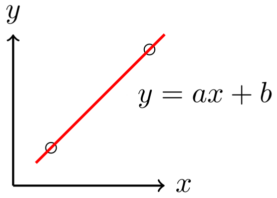
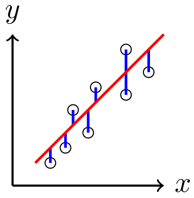


Image Source: Bishop, C. M. (2006). Pattern recognition and machine learning. springer.

Linear Regression







Given a set of m points $\{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$, our goal is to find the optimal line parameters a and b , such that $(\hat{y}_i - y_i)^2$ is as small as possible for all the training points. Here $\hat{y}_i = ax_i + b$ is the predicted target value. Therefore, we minimize the below error with respect to a and b.

$$\sum_{i \in [m]} (\hat{y}_i - y_i)^2 = \sum_{i \in [m]} (ax_i + b - y_i)^2$$

Linear Regression

$$\min_{a,b} \sum_{i=1}^{m} (ax_i + b - y_i)^2 \Rightarrow \min_{\mathbf{x}} \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2.$$

$$\mathbf{A} = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_m & 1 \end{bmatrix}, \mathbf{x} = \begin{bmatrix} a \\ b \end{bmatrix}, \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

Solution

$$\nabla_{\mathbf{x}} f = 2\mathbf{A}^\top \mathbf{A}\mathbf{x} - 2\mathbf{A}^\top \mathbf{y} = \mathbf{0} \Rightarrow \mathbf{x}^\star = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{y}.$$

Feature Selection

- Let $\mathcal{X} = \mathbb{R}^d$ be the instance space. That is, each point is represented as a vector of d features.
- Our goal is to learn a predictor that only relies on $k << d$ features.
- Predictors that use only a small subset of features require a smaller memory footprint and can be applied faster.
- A naive approach would be to try all subsets of k out of d features and choose the subset which leads to the best performing predictor.
- However, such an exhaustive search is usually computationally intractable.

Filters

Assess individual features, independently of other features, according to some quality measure. We can then select the k features that achieve the highest score.

Pearson's correlation coefficient

Consider a linear regression problem. Let $\mathbf{X} = \begin{bmatrix}\mathbf{x}_1 & \cdots & \mathbf{x}_m\end{bmatrix}^\top \in \mathbb{R}^{d\times m}$ be the training points. Let $\mathbf{v} = \begin{bmatrix}x_{1,j} & \cdots & x_{m,j}\end{bmatrix}^\top \in \mathbb{R}^m$ be a vector denoting the j^{th} mean centered feature and let $\mathbf{y} = \begin{bmatrix}y_1 & \cdots & y_m\end{bmatrix}^\top \in \mathbb{R}^m$ be the mean centered values of the target. The occurred loss that uses only the j^{th} feature would be

$$\min_{a,b\in\mathbb{R}} \|a\mathbf{v} + b\mathbf{1} - \mathbf{y}\|_2^2$$

Pearson's correlation coefficient

The solution to this optimization problem is $b = 0$ and $a = \frac{(\mathbf{v}^\top \mathbf{y})}{\|\mathbf{v}\|_2^2}$. Plugging this value back into the objective we obtain the value

$$\|\mathbf{y}\|_2^2 - \frac{(\mathbf{v}^\top \mathbf{y})^2}{\|\mathbf{v}\|_2^2} = \|\mathbf{y}\|_2^2 \left(1 - \frac{(\mathbf{v}^\top \mathbf{y})^2}{\|\mathbf{v}\|_2^2 \times \|\mathbf{y}\|_2^2}\right)$$

Ranking the features according to the minimal loss they achieve is equivalent to ranking them according to the absolute value of the following score (where now a higher score yields a better feature):

$$\frac{(\mathbf{v}^\top \mathbf{y})}{\|\mathbf{v}\|_2 \times \|\mathbf{y}\|_2} = \frac{\frac{1}{m}(\mathbf{v}^\top \mathbf{y})}{\sqrt{\frac{1}{m}\|\mathbf{v}\|_2^2}\sqrt{\frac{1}{m}\|\mathbf{y}\|_2^2}}$$

- The numerator is the empirical estimate of the covariance of the j-th feature and the target value, $\mathbb{E}[(\mathbf{v}-\mathbb{E}(\mathbf{v}))(\mathbf{y}-\mathbb{E}(\mathbf{y}))]$, while the denominator is the squared root of the empirical estimate for the variance of the j-th feature, $\mathbb{E}[(\mathbf{v}-\mathbb{E}(\mathbf{v}))^2]$, times the variance of the target.
- Pearson's coefficient ranges from -1 to $+1$, where if the Pearson's coefficient is either $+1$ or -1, there is a linear mapping from $\mathbf{v}$ to $\mathbf{y}$ with zero empirical risk.
- If Pearson's coefficient equals zero it means that the optimal linear function from $\mathbf{v}$ to $\mathbf{y}$ is the all-zeros function, which means that $\mathbf{v}$ alone is useless for predicting $\mathbf{y}$.
- However, this does not mean that $\mathbf{v}$ is a bad feature, as it might be the case that together with other features $\mathbf{v}$ can perfectly predict $\mathbf{y}$.

Feature Transformation

We denote by $\mathbf{f} = \begin{bmatrix} f_1 & f_2 & \cdots & f_m \end{bmatrix}^\top \in \mathbb{R}^m$ the value of the feature f over the m training examples. We denote by $\bar{f} = \frac{1}{m} \sum_{i=1}^{m} f_i$ the empirical mean of the feature over all examples.

Centering

This transformation makes the feature have zero mean, by setting $f_i \leftarrow f_i - \bar{f}$.

Unit Range

This transformation makes the range of each feature be $[0, 1]$. Formally, let $f_{\text{max}} = \max\{f_1, f_2, \ldots, f_m\}$ and $f_{\text{min}} = \min\{f_1, f_2, \ldots, f_m\}$. Then, we set

$$f_i \leftarrow \frac{f_i - f_{\text{min}}}{f_{\text{max}} - f_{\text{min}}}$$

Feature Transformation

Standardization

This transformation makes all features have a zero mean and unit variance. Formally, let $\sigma_{\mathbf{v}}^2 = \frac{1}{m}\sum_{i=1}^{m}(f_i - \bar{f})^2$ be the empirical variance of the feature. Then, we set:

$$f_i \leftarrow \frac{f_i - \bar{f}}{\sigma_{\mathbf{v}}}.$$

Sigmoid

This transformation applies a sigmoid function on the feature. For example,

$$f_i \leftarrow \frac{1}{1 + e^{bf_i}}.$$

Here, where b is a user-specified parameter.

Feature Learning

We start with some instance space, $\mathcal{X}$, and would like to learn a function, $\phi : \mathcal{X} \to \mathbb{R}^d$, which maps instances in $\mathcal{X}$ into a representation as d-dimensional feature vectors.

Auto Encoders

We learn a pair of functions: an "encoder" function $\psi : \mathbb{R}^d \to \mathbb{R}^k$, and a "decoder" function $\phi : \mathbb{R}^k \to \mathbb{R}^d$. The goal of the learning process is to find a pair (ψ, ϕ) of functions such that the reconstruction error, defined as below, is as small as possible.

$$\sum_{i=1}^{m} \|\mathbf{x}_i - \phi(\psi(\mathbf{x}_i))\|_2^2.$$

Feature Learning

PCA

We constrain $k < d$ and restrict ψ to a matrix $\mathbf{W} \in \mathbb{R}^{k \times d}$ and ϕ to a matrix $\mathbf{U} \in \mathbb{R}^{d \times k}$ and minimize the reconstruction error $\sum_{i=1}^{m} \|\mathbf{x}_i - \mathbf{U}\mathbf{W}\mathbf{x}_i\|_2^2$

k-Means

In k-means, k is not restricted to be smaller than d, but now ψ and ϕ rely on k centroids, $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_k$, and $\psi(\mathbf{x})$ returns an indicator vector in $\{0,1\}^k$ that indicates the closest centroid to $\mathbf{x}$, while ϕ takes as input an indicator vector and returns the centroid representing this vector.

Feature Learning

Sparse Representation

- An important property of the k-means construction, which is key in allowing k to be larger than d, is that ψ maps instances into sparse vectors.
- In fact, in k-means only a single coordinate of $\psi(x)$ is nonzero.
- An immediate extension of the k-means construction is therefore to restrict the range of ψ to be vectors with at most s nonzero elements, where s is a small integer.

Feature Learning

- In particular, let ψ and ϕ be functions that depend on $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_k$.
- The function ψ maps an instance vector $\mathbf{x}$ to a vector $\psi(\mathbf{x}) \in \mathbb{R}^k$, where $\psi(\mathbf{x})$ should have at most s nonzero elements.
- The function $\phi(\mathbf{v})$ is defined to be $\sum_{i=1}^{k} v_i \boldsymbol{\mu}_i$.
- As before, our goal is to have a small reconstruction error, and therefore we can define

$$\psi(\mathbf{x}) = \arg\min_{\mathbf{v}} \|\mathbf{x} - \phi(\mathbf{v})\|_2^2 \text{ subject to } \|\mathbf{v}\|_0 \leq s.$$