SURVEY OF ADDITIONAL TOPICS

Matrix Factorization

\[ X \approx A \cdot B, \quad X \text{ is the (centered) data matrix} \]

1) PCA

\[
\begin{align*}
\min_{A,B} & \quad \| X - A \cdot B \|_F^2 \\
\text{s.t.} & \quad A \in \mathbb{R}^{d \times k} \\
& \quad B \in \mathbb{R}^{k \times n} \\
& \quad A^T A = I
\end{align*}
\]

2) K-means

\[
\begin{align*}
\min_{A,B} & \quad \| X - A \cdot B \|_F^2 \\
\text{s.t.} & \quad A \in \mathbb{R}^{d \times k} \\
& \quad B \in \mathbb{R}^{k \times n} \\
& \quad \text{columns of } B \text{ are indicator vectors}
\end{align*}
\]

3) Independent component analysis (ICA)

\[ \text{...} \]
\[ X \sim A \cdot S \]

\[ S = [s_i] \] such that

for each \( t \), \( S_t, \ldots, S_T \)

are realizations of independent RVs.

"Cocktail Party Problem"

\[ X = [x_{it}] \]

\( x_{it} = \text{mic } i \) measurement at time \( t \)

\( s_{it} = \text{speaker } i \) speech signal at time \( t \)
4) Nonnegative matrix factorization (NMF)

$$\min \| X - A \cdot B \|_F^2$$

s.t. $A \in \mathbb{R}^{d \times k}$

$B \in \mathbb{R}^{k \times n}$

elements of $A, B$ are nonnegative.
5) Sparse coding / dictionary learning

\[
\min_{D,A} \| X - D \cdot A \|_F^2
\]

\( D \in \mathbb{R}^{d \times s} \quad (s > d) \)
\( A \in \mathbb{R}^{s \times n} \)

columns of D have unit norm
columns of A sparse

Intuitively, find a set of components (dictionary columns) such that every column of X is explained as a superposition of a small number of components.
Sparse coding illustration

\[ [a_1, \ldots, a_{64}] = [0, 0, \ldots, 0, 0.8, 0, \ldots, 0, 0.3, 0, \ldots, 0, 0.5, 0] \]
(feature representation)

Algorithmic strategy: alternating minimization

7) Matrix completion

\[ X = [x_{ij}], \quad \Omega \subseteq \{1, \ldots, d\} x \{1, \ldots, n\} \]

\((d \times n)\)

\(x_{ij}\) is only observed for \((i, j) \in \Omega\)

Basic approach: assume \(X\) has rank \(r \leq \min(d,n)\).

\[
\min_{A,B} \| X - A \cdot B \|_{F, \Omega}^2 \leftarrow \text{sum of squares of entries indexed by } \Omega
\]
8) Sparse PCA (\(\Omega_i\)'s constrained to be sparse)

9) Probabilistic PCA: generative model whose maximum likelihood estimate coincides with PCA. Useful for extending PCA to
   - missing data
   - mixture models

10) Factor analysis: slightly more flexible generative model
relative to PPCA.

Latent semantic indexing: Use PCA/SVD to get low rank approximation of $X$, where columns of $X$ correspond to documents, rows to words in a vocabulary, and entries of $X$ are word counts.

Nuclear Norm Regularization

Let $X \in \mathbb{R}^{d \times n}$ be a data matrix. Suppose we seek a the best rank $r$ approximation to $X$.

Then we know to just apply PCA/SVD. But what if the true $r$ is unknown?

One option is to solve

$$\min_{W \in \mathbb{R}^{d \times n}} \|X - W\|_F^2 + \lambda \cdot \text{rank}(W)$$

However, the rank function is nonconvex. Analogous to how the $l_1$ norm is a convex proxy for the sparsity of a vector, the nuclear norm,
\[ \| W \|_x := \sum \sigma_i \] (sum of singular values)

is the tightest convex relaxation of rank. This leads to

\[
\min_{W \in \mathbb{R}^{d \times n}} \| X - W \|_F^2 + \lambda \| W \|_x
\]

which is now a convex problem. It can be solved using ADMM where the prox operator for the nuclear norm is given by singular value thresholding = soft thresholding applied to the singular values of the argument.

For matrix completion, one solves

\[
\min_W \| X - W \|_F^2 + \lambda \| W \|_x
\]

This approach yields a global minimum, unlike the alternating algorithm mentioned earlier.

As another application, consider robust PCA:

\[
\min \| X - W \|_F^2 + \lambda \| L \|_x + \sigma \| S \|_{1,1}
\]

s.t. \[ W = L + S \] (sum of)
\[ \text{st. } \mathbf{w} = \mathbf{L} + \mathbf{S} \]

\( S \) corresponds to outliers, and

\( L \) gives the low dim. representation.

(just apply standard PCA to \( L \)).

\[ \text{sum of absolute values of all entries} \]

\[ \text{Group Lasso} \]

Recall that the \( l_1 \) or "lasso" penalty promotes sparsity and is useful for feature selection. The "group lasso" penalty is useful for group feature selection.

Consider a prediction problem (classification or regression) where the features can be naturally grouped.

**Example**

In classification of brain images, groups of pixels correspond to anatomical units (e.g., hippocampus, visual cortex).

Let \( G_1, \ldots, G_m \) be a partition of \( \{1, \ldots, d\} \), so that

- \( G_r \cap G_s = \emptyset \) if \( r \neq s \)
- \( \bigcup_{r=1}^{m} G_r = \{1, \ldots, d\} \).
Let $w_G$ denote the vector $w$ restricted to features in $G$, e.g.,

\[
\begin{bmatrix}
11 \\
-4 \\
-1 \\
17 \\
8
\end{bmatrix}, \quad G = \{2, 5\} \Rightarrow w_G = \begin{bmatrix}
-4 \\
8
\end{bmatrix}.
\]

The group lasso penalty is $\sum_{r=1}^{M} \|w_r\|_2$. Therefore, to perform linear regression with group feature selection, we would solve

\[
\min_w \frac{1}{n} \sum_{i=1}^{n} (y_i - w^T x_i - b)^2 + \sum_{r} \|w_r\|_2.
\]

The intuition is that $\sum \|w_r\|$ can be viewed as the $\ell_1$ norm of $(\|w_1\|_2, \ldots, \|w_r\|_2)$, which encourages most values of $\|w_r\|_2$ to be zero, i.e., $w_r = \text{zero vector}$.

**Multiclass SVM**
One way to define a linear SVM in the multiclass case is
\[
    f(x) = \arg \max_{k=1,\ldots,K} \langle w_k, x \rangle
\]
where \( w_k \) is associated with class \( k \), and solves
\[
    \min_{w_1, \ldots, w_K} \frac{1}{2} \sum_{k=1}^{K} \|w_k\|^2 + \frac{C}{n} \sum_{i=1}^{n} \xi_i
\]
\( \text{s.t.} \quad \langle w_{y_i} - w_k, x_i \rangle \geq 1 - \xi_i, \quad \forall i, \forall k \neq y_i \)
\[\xi_i \geq 0 \quad \forall i\]

The above formulation can be kernelized using the dual optimization problem.

**Q:** How could we incorporate embedded feature selection into the linear multiclass SVM?

**A:** Group lasso penalty where groups correspond to features

---

Multitask Learning

Suppose there are \( N \) different (but possibly related)
classification problems, referred to as tasks, and let
\[
\{ (x_j^{(i)}, y_j^{(i)}) \mid j = 1, \ldots, n_i \}
\]
be training data for the \( i \)th task.

In multi-task learning, the goal is to learn the \( N \) classifiers simultaneously, in hopes that if some tasks are sufficiently similar, training data can be pooled, thus leading to a larger effective sample size for some or all tasks.

Let's consider the linear case. Let \( W^{(i)} \in \mathbb{R}^d \) be the parameter associated with task \( i \), and write
\[
W = \begin{bmatrix}
W^{(1)} & \cdots & W^{(N)}
\end{bmatrix} = \begin{bmatrix}
W_1^T \\
\vdots \\
W_d^T
\end{bmatrix}
\] (d \times n)

A basic approach is to solve
\[
\min_{W} \frac{1}{N} \sum_{i=1}^{N} \frac{1}{n_i} \sum_{j=1}^{n_i} l(y_{ij}, \langle w_j^{(i)}, x_i \rangle) + \lambda R(W)
\]
\[ \min \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} \mathcal{L}(y_i, w_j, x_i, y_i), \quad R(w) \]

where \( R \) is a regularizer that encourages \( w^{(1)}, \ldots, w^{(N)} \) to be similar. Can you suggest a good \( R? \)

Here are some possibilities:

- **shared mean**: 
  \[
  R(w) = \sum_{i=1}^{N} \left\| w^{(i)} - \frac{1}{N} \sum_{k=1}^{N} w^{(k)} \right\|_2^2
  \]

- **nuclear norm**: 
  \[
  R(w) = \| W \|_*
  \]

- **group lasso**: 
  \[
  R(w) = \sum_{k=1}^{d} \| w_k \|_2
  \]

For which of the above regularizers can the method be kernelized?