Solution
Math189R SU17
Homework 5
Monday, June 5, 2017

Feel free to work with other students, but make sure you write up the homework and code on your own (no copying homework *or* code; no pair programming). Feel free to ask students or instructors for help debugging code or whatever else, though.

1 (Murphy 12.5 - Deriving the Residual Error for PCA) It may be helpful to reference section 12.2.2 of Murphy.

(a) Prove that

$$\left\| \mathbf{x}_i - \sum_{j=1}^{k} z_{ij} \mathbf{v}_j \right\|^2 = \mathbf{x}_i^\top \mathbf{x}_i - \sum_{j=1}^{k} \mathbf{v}_j^\top \mathbf{x}_i \mathbf{x}_i^\top \mathbf{v}_j.$$

Hint: first consider the case when $k = 2$. Use the fact that $\mathbf{v}_i^\top \mathbf{v}_j$ is 1 if $i = j$ and 0 otherwise. Recall that $z_{ij} = \mathbf{x}_i^\top \mathbf{v}_j$.

(b) Now show that

$$J_k = \frac{1}{n} \sum_{i=1}^{n} \left(\mathbf{x}_i^\top \mathbf{x}_i - \sum_{j=1}^{k} \mathbf{v}_j^\top \mathbf{x}_i \mathbf{x}_i^\top \mathbf{v}_j \right) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i^\top \mathbf{x}_i - \sum_{j=1}^{k} \lambda_j.$$

Hint: recall that $\mathbf{v}_j^\top \boldsymbol{\Sigma} \mathbf{v}_j = \lambda_j \mathbf{v}_j^\top \mathbf{v}_j = \lambda_j$.

(c) If $k = d$ there is no truncation, so $J_d = 0$. Use this to show that the error from only using $k < d$ terms is given by

$$J_k = \sum_{j=k+1}^{d} \lambda_j.$$

Hint: partition the sum $\sum_{j=1}^{d} \lambda_j$ into $\sum_{j=1}^{k} \lambda_j$ and $\sum_{j=k+1}^{d} \lambda_j$.

(a) We know

$$\begin{aligned}
\left\| \mathbf{x}_i - \sum_{j=1}^{k} z_{ij}\mathbf{v}_j \right\|_2^2 &= \left(\mathbf{x}_i - \sum_{j=1}^{k} z_{ij}\mathbf{v}_j \right)^\top \left(\mathbf{x}_i - \sum_{j=1}^{k} z_{ij}\mathbf{v}_j \right) \\
&= \mathbf{x}_i^\top \mathbf{x}_i - \sum_{j=1}^{k} z_{ij}\mathbf{v}_j^\top \mathbf{x}_i - \mathbf{x}_i^\top \sum_{j=1}^{k} z_{ij}\mathbf{v}_j + \left(\sum_{j=1}^{k} z_{ij}\mathbf{v}_j \right)^\top \left(\sum_{j=1}^{k} z_{ij}\mathbf{v}_j \right) \\
&= \mathbf{x}_i^\top \mathbf{x}_i - 2\sum_{j=1}^{k} z_{ij}\mathbf{v}_j^\top \mathbf{x}_i + \left(\sum_{j=1}^{k} z_{ij}\mathbf{v}_j \right)^\top \left(\sum_{j=1}^{k} z_{ij}\mathbf{v}_j \right) && \text{(bringing } \mathbf{x}_i^\top \text{ into sum)} \\
&= \mathbf{x}_i^\top \mathbf{x}_i - 2\sum_{j=1}^{k} z_{ij}\mathbf{v}_j^\top \mathbf{x}_i + \sum_{j=1}^{k} \mathbf{v}_j^\top z_{ij}^\top z_{ij}\mathbf{v}_j \\
&= \mathbf{x}_i^\top \mathbf{x}_i - 2\sum_{j=1}^{k} z_{ij}\mathbf{v}_j^\top \mathbf{x}_i + \sum_{j=1}^{k} \mathbf{v}_j^\top \mathbf{x}_i \mathbf{x}_i^\top \mathbf{v}_j && \text{(since } \mathbf{v}_i^\top \mathbf{v}_j = 1 \text{ iff } i = j\text{)} \\
&= \mathbf{x}_i^\top \mathbf{x}_i - 2\sum_{j=1}^{k} \mathbf{v}_j^\top \mathbf{x}_i \mathbf{x}_i^\top \mathbf{v}_j + \sum_{j=1}^{k} \mathbf{v}_j^\top \mathbf{x}_i \mathbf{x}_i^\top \mathbf{v}_j && \text{(since } z_{ij} \in \mathbb{R}\text{)} \\
&= \mathbf{x}_i^\top \mathbf{x}_i - \sum_{j=1}^{k} \mathbf{v}_j^\top \mathbf{x}_i \mathbf{x}_i^\top \mathbf{v}_j,
\end{aligned}$$

as desired.

(b) By definition

$$\begin{aligned}
J_k &= \frac{1}{n}\sum_{i=1}^{n} \left(\mathbf{x}_i^\top \mathbf{x}_i - \sum_{j=1}^{k} \mathbf{v}_j^\top \mathbf{x}_i \mathbf{x}_i^\top \mathbf{v}_j \right) \\
&= \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i^\top \mathbf{x}_i - \sum_{j=1}^{k} \mathbf{v}_j^\top \frac{1}{n} \left(\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top \right) \mathbf{v}_j \\
&= \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i^\top \mathbf{x}_i - \sum_{j=1}^{k} \mathbf{v}_j^\top \boldsymbol{\Sigma} \mathbf{v}_j \\
&= \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i^\top \mathbf{x}_i - \sum_{j=1}^{k} \lambda_j
\end{aligned}$$

as desired.

(c) Since $J_d = 0$ we know $\sum_{j=1}^{d} \lambda_j = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i^\top \mathbf{x}_i$. Then

$$J_k = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i^\top \mathbf{x}_i - \sum_{j=1}^{d} \lambda_j + \sum_{j=k+1}^{d} \lambda_j = \sum_{j=k+1}^{d} \lambda_j.$$

This is an exciting result. This states that the reconstruction error when using a PCA projection of your data is exactly equal to the sum of the eigenvalues you throw out.

■

2 (ℓ_1-Regularization) Consider the ℓ_1 norm of a vector $\mathbf{x} \in \mathbb{R}^n$:

$$\|\mathbf{x}\|_1 = \sum_i |\mathbf{x}_i|.$$

Draw the norm-ball $B_k = \{\mathbf{x} : \|\mathbf{x}\|_1 \leq k\}$ for $k = 1$. On the same graph, draw the Euclidean norm-ball $A_k = \{\mathbf{x} : \|\mathbf{x}\|_2 \leq k\}$ for $k = 1$ behind the first plot. (Do not need to write any code, draw the graph by hand).

Show that the optimization problem

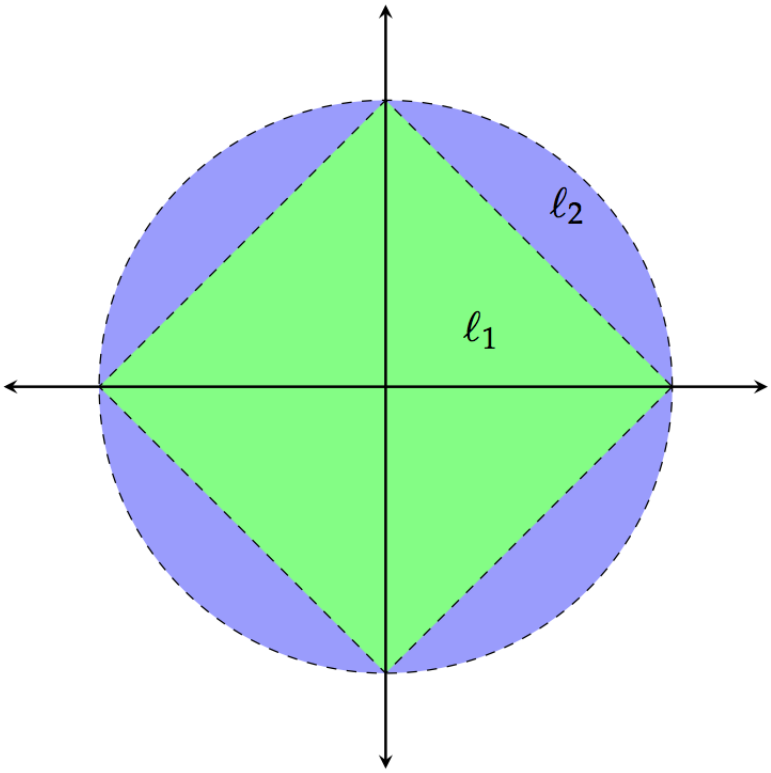
$$\begin{aligned} &\text{minimize: } f(\mathbf{x}) \\ &\quad \text{subj. to: } \|\mathbf{x}\|_p \leq k \end{aligned}$$

is equivalent to

$$\text{minimize: } f(\mathbf{x}) + \lambda\|\mathbf{x}\|_p$$

(hint: create the Lagrangian). With this knowledge, and the plots given above, argue why using ℓ_1 regularization (adding a $\lambda\|\mathbf{x}\|_1$ term to the objective) will give sparser solutions than using ℓ_2 regularization for suitably large λ.

We see the norm balls below.



We know the optimization problem

$$\begin{aligned} &\text{minimize: } f(\mathbf{x}) \\ &\quad \text{subj. to: } \|\mathbf{x}\|_p \leq k \end{aligned}$$

is equivalent to

$$\inf_{\mathbf{x}} \sup_{\lambda \geq 0} \mathcal{L}(\mathbf{x}, \lambda) = \inf_{\mathbf{x}} \sup_{\lambda \geq 0} f(\mathbf{x}) + \lambda(\|\mathbf{x}\|_p - k).$$

In its dual we can flip the inf and sup.

$$\sup_{\lambda \geq 0} \inf_{\mathbf{x}} f(\mathbf{x}) + \lambda(\|\mathbf{x}\|_p - k) = \sup_{\lambda \geq 0} g(\lambda)$$

Since the minimizing value of $f(\mathbf{x}) + \lambda(\|\mathbf{x}\|_p - k)$ over $\mathbf{x}$ is equivalent to the minimizing value of $f(\mathbf{x}) + \lambda\|\mathbf{x}\|_p$ ($-\lambda k$ doesn't depend on $\mathbf{x}$), we know the the optimizing $\mathbf{x}$ will solve

$$\text{minimize: } f(\mathbf{x}) + \lambda\|\mathbf{x}\|_p$$

for some suitable value of $\lambda \geq 0$. Looking at the plot and this result, we can consider ℓ_1 regularization as projecting the actual optimal solution of your problem onto some suitably sized ℓ_1 norm ball. Since the ℓ_1 ball has sharper edges, the probability of landing on an edge and not on the face (where both elements of the vector are nonzero) is infinitely larger than the ℓ_2 ball. This is due to the rotation invariance of the ℓ_2 that certainly doesn't hold for the ℓ_1 ball. Generalizing to higher dimensions, we can see that the ℓ_1 penalty will encourage more weights to be zero compared to the ℓ_2 ball, as desired.

■

Extra Credit (Lasso) Show that placing an equal zero-mean Laplace prior on each element of the weights $\boldsymbol{\theta}$ of a model is equivelent to ℓ_1 regularization in the Maximum-a-Posteriori estimate

$$\text{maximize: } \mathbb{P}(\boldsymbol{\theta}|\mathcal{D}) = \frac{\mathbb{P}(\mathcal{D}|\boldsymbol{\theta})\mathbb{P}(\boldsymbol{\theta})}{\mathbb{P}(\mathcal{D})}.$$

Note the form of the Laplace distribution is

$$\text{Lap}(x|\mu, b) = \frac{1}{2b}\exp\left(-\frac{|x-\mu|}{b}\right)$$

where μ is the location parameter and $b > 0$ controls the variance. Draw (by hand) and compare the density $\text{Lap}(x|0,1)$ and the standard normal $\mathcal{N}(x|0,1)$ and suggest why this would lead to sparser solutions than a Gaussian prior on each elements of the weights (which correspond to ℓ_2 regularization).

We know the Maximum-a-Posteriori problem

$$\text{maximize: } \mathbb{P}(\boldsymbol{\theta}|\mathcal{D}) = \frac{\mathbb{P}(\mathcal{D}|\boldsymbol{\theta})\mathbb{P}(\boldsymbol{\theta})}{\mathbb{P}(\mathcal{D})}.$$

is equivalent to maximizing $\log \mathbb{P}(\boldsymbol{\theta}|\mathcal{D})$ given the monotonicity of $\log(x)$. This gives

$$\text{maximize: } \log \mathbb{P}(\boldsymbol{\theta}|\mathcal{D}) = \log \mathbb{P}(\mathcal{D}|\boldsymbol{\theta}) + \log \mathbb{P}(\boldsymbol{\theta}) - \log \mathbb{P}(\mathcal{D}).$$

Since $\mathbb{P}(\mathcal{D})$ is a constant not dependent on $\boldsymbol{\theta}$, we can drop that term from the problem and flip into a minimization problem, giving

$$\text{minimize: } -\log \mathbb{P}(\mathcal{D}|\boldsymbol{\theta}) - \log \mathbb{P}(\boldsymbol{\theta}).$$

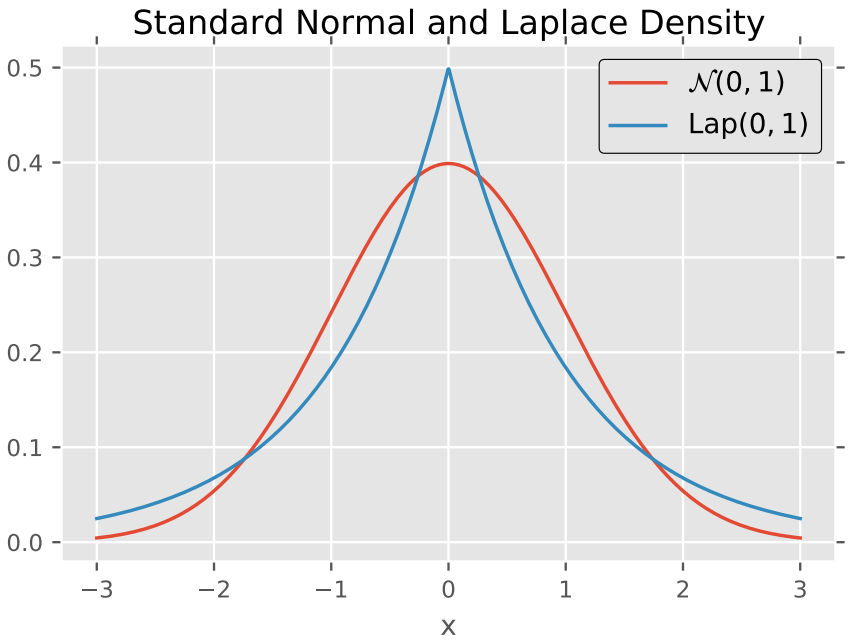
Given a prior $\boldsymbol{\theta}_i \sim \text{Lap}(0, b)$,

$$\begin{aligned}
-\log \mathbb{P}(\boldsymbol{\theta}) &= -\log \prod_i \exp\left(-\frac{|\boldsymbol{\theta}_i|}{b}\right) + Z && \text{(where } Z \text{ is a constant)} \\
&= \frac{1}{b}\sum_i |\boldsymbol{\theta}_i| + Z \\
&= \lambda \|\boldsymbol{\theta}\|_1 + Z. && \text{(where } \lambda = 1/b\text{)}
\end{aligned}$$

It follows that our original problem is equivalent to

$$\text{minimize: } -\log \mathbb{P}(\mathcal{D}|\boldsymbol{\theta}) + \lambda\|\boldsymbol{\theta}\|_1,$$

or a ℓ_1 regularized maximum likelihood estimate, as desired. Note the plots of the Standard Normal and Laplace Densities.



We can see that $\text{Lap}(0,1)$ will place much more mass at $x = 0$. It follows that when we use a Laplace prior instead of a Gaussian prior on our weights, our weights will be more encouraged to be exactly zero, forcing sparsity.

■