DS5230/DS4420 Unsupervised Machine Learning and Data Mining – Fall 2018 – Homework 2

Submission Instructions

- It is recommended that you complete this exercises in **Python 3** and submit your solutions as a **Jupyter notebook**.
- You may use any other language, as long as you **include a README** with simple, clear instructions on how to run (and if necessary compile) your code.
- Please upload all files (code, README, written answers, etc.) to **blackboard** in a single **zip file** named $\{firstname\}\_\{lastname\}\_DS5230/DS4220\_HW2.zip$.

Exercise 1: Bayesian Linear Regression

In this assignment, you will need to derive some identities to express ridge regression as maximum a posteriori (MAP) estimates relative to an appropriate prior. You will also derive confidence intervals on weights in Bayesian linear regression, as well as confidence intervals on predictions. Please complete this part of the exercise as a PDF file (preferably typed up in LaTeX).

a. As discussed in class, ridge regression (i.e. L2-regularized linear regression) minimizes the loss function:

$$\begin{aligned} L(\boldsymbol{w}) &= ||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}||^2 + \lambda||\boldsymbol{w}||^2 \\ &= \sum_{n=1}^{N} (y_n - \boldsymbol{x}_n^\top \boldsymbol{w})^2 + \lambda \boldsymbol{w}^\top \boldsymbol{w}. \end{aligned}$$

The closed form solution for the weights $\boldsymbol{w}^*$ that minimize this loss is

$$\boldsymbol{w}^* = (\boldsymbol{X}^\top \boldsymbol{X} + \lambda \boldsymbol{I})^{-1} \boldsymbol{X}^\top \boldsymbol{y}.$$

To demonstrate that this solution is indeed an extremum of the loss function, show that it satisfies

$$\nabla_{\boldsymbol{w}} L(\boldsymbol{w})\big|_{\boldsymbol{w}=\boldsymbol{w}^*} = 2\boldsymbol{X}^\top(\boldsymbol{X}\boldsymbol{w}^* - \boldsymbol{y}) + 2\lambda \boldsymbol{w} = 0.$$

b. Show that ridge regression (i.e. L2-regularized linear regression) is equivalent to a MAP estimation problem in which a prior is placed on the regression weights

$$\boldsymbol{w}^* = \underset{\boldsymbol{w}}{\operatorname{argmax}}\, p(\boldsymbol{w} \mid \boldsymbol{y})$$
$$p(\boldsymbol{y} \mid \boldsymbol{w}) = \mathcal{N}(\boldsymbol{y}; \boldsymbol{X}\boldsymbol{w}, \sigma^2 \boldsymbol{I})$$
$$p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}; \boldsymbol{0}, s^2 \boldsymbol{I})$$

Express the regularization parameter λ in terms of the constants σ and s, and b. (*note:* this problem should be easy if you have reviewed the slides).

c. We will now consider the more general case where we place a multivariate Gaussian prior with parameters $\boldsymbol{m_0}$ and $\boldsymbol{S_0}$ on the weights

$$\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{m}_0, \boldsymbol{S}_0).$$

Derive the marginal likelihood of $\boldsymbol{y}$, which is defined as

$$p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{m}_0, \boldsymbol{S}_0) = \int d\boldsymbol{w}\; p(\boldsymbol{y}|\boldsymbol{w}, \boldsymbol{X}) p(\boldsymbol{w}|\boldsymbol{m}_0, \boldsymbol{S}_0),$$

by calculating the mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ such that

$$\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{m}_0, \boldsymbol{S}_0 \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

To do so, make use of the fact that for $\boldsymbol{f} = \boldsymbol{X}\boldsymbol{w}$, we can write

$$\mathbb{E}[\boldsymbol{f}] = \mathbb{E}[\boldsymbol{X}\boldsymbol{w}] = \boldsymbol{X}\mathbb{E}[\boldsymbol{w}],$$
$$\mathrm{Cov}[\boldsymbol{f}] = \mathrm{Cov}[\boldsymbol{f}] = \mathrm{Cov}[\boldsymbol{X}\boldsymbol{w}] = \boldsymbol{X}\mathrm{Cov}[\boldsymbol{w}]\boldsymbol{X}^\top.$$

You will additionally need to make use of the fact that the covariance of two uncorrelated variables (which we will here conveniently denote $\boldsymbol{f}$ and $\boldsymbol{\epsilon}$) is

$$\mathrm{Cov}[\boldsymbol{f} + \boldsymbol{\epsilon}] = \mathrm{Cov}[\boldsymbol{f}] + \mathrm{Cov}[\boldsymbol{\epsilon}].$$

d. Using the result from the previous exercise, derive the predictive distribution at a new test point $\boldsymbol{x}_*$, which is defined as

$$p(y_*|\boldsymbol{y}, \boldsymbol{X}, \boldsymbol{x}_*) = \int d\boldsymbol{w}\; p(y_*|\boldsymbol{w}, \boldsymbol{x}_*) p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{X}).$$

Calculate the values f_* and σ_* such that

$$y_*|\boldsymbol{y}, \boldsymbol{X}, \boldsymbol{x}_* \sim N(f_*, \sigma_*).$$

To do so, use the standard identity on Gaussians (which we have already seen in class) that states that

$$\boldsymbol{\alpha}|\boldsymbol{\beta} \sim \mathcal{N}\left(\boldsymbol{a} + \boldsymbol{C}\boldsymbol{B}^{-1}(\boldsymbol{\beta} - \boldsymbol{b}), \boldsymbol{A} - \boldsymbol{C}\boldsymbol{B}^{-1}\boldsymbol{C}^\top\right),$$

when

$$\begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{a} \\ \boldsymbol{b} \end{bmatrix}, \begin{bmatrix} \boldsymbol{A} & \boldsymbol{C} \\ \boldsymbol{C}^\top & \boldsymbol{B} \end{bmatrix}\right).$$

e. To check your work, substitute $\boldsymbol{m}_0 = 0$ and $\boldsymbol{S}_0 = s^2\boldsymbol{I}$ into the result of the previous exercise. Express the solution for $\boldsymbol{f}_* = \mathbb{E}[f(\boldsymbol{x}_*)]$ as a function of $\boldsymbol{x}_*$. What is this result equal to?

Can PCA uncover academic communities?

In this problem, you will try to apply Principal components analysis (PCA) techniques to the real world bibliographic dataset from Aminer (`https://aminer.org/`). The dataset (`publications.txt`) was downloaded and attached with this assignment. We will see if we can use latent semantic analysis (PCA on text) to discover structure in paper titles.

You may implement the following exercise using any language/system you choose. We recommend that you use the Jupyter docker instance with Spark (`http://spark.apache.org/`). The `pyspark.ml.feature` namespace implements many of the prerequisite operations, which are documented here:

`http://spark.apache.org/docs/latest/ml-features.html`

In the first part of the exercise, we will consider the dataset as a whole

a. Transform titles to word count vectors. Truncate your (sparse) vectors to the 1000 most frequent words and perform PCA with 50 components on the counts.

b. Plot the eigenvalues of the principal components. Calculate how many components are needed to explain 50% of the total variance?

c. Identify which words are important in each of the principal components. To do so, take the sum of squares of each of the component vectors to check how they are normalized. For each component, then print out the words for which the absolute value of the component is larger than 0.20 of the norm.

d. Make a scatter plot of some reasonably sized sample (1k-10k titles). Explain the structure (or lack thereof) you see based on your results from item b-c.

e. Run a preprocessing step to remove stop words (a list of stop words is provided which is identical to the list used in Spark). Rerun steps b-d and evaluate whether this has improved your representation.

f. One of the issues with text is that the variance in how often a word appears is often correlated with the base frequency with which a word appears. To account for this, the word counts are often reweighted using the term frequency, inverse document frequency (TF-IDF) scheme:

$$\text{tf-idf}(t, d, D) := \text{tf}(t, d)\text{idf}(t, D)$$

Here $\text{tf}(t, d)$ is a measure of how often a term t (i.e. a vocabulary word) occurs in document d (i.e. a title). Here we will simply use the word counts calculate in part a. of this exercise:

$$\text{tf}(t, d) = |\{t' \in d : t' = t\}|$$

The inverse document frequency $\text{idf}(t, D)$ is used to discount terms t that occur commonly across all documents D. Here we will use:

$$\text{idf}(t, D) := \log \frac{|D| + 1}{|\{d \in D : t \in d\}| + 1}$$

Note that $\text{idf}(t, D) = 0$ for words that occur in all documents.

Calculate TF-IDF features for all titles and rerun the operations in parts b-d of this exercise. How have your results changed?

In the second part of this exercise, we will look at subsets of the data. To construct these subsets, first construct two lists of titles:

- Titles for machine learning papers published at 'NIPS'
- Titles for database papers published at 'VLDB'

i. Merge the two sets of titles. Construct both word count vectors and TF-IDF features. Repeat steps b-d and compare word count results to TF-IDF results.

j. Now make a scatter plot of these two principal components, showing the titles from each subset in different colors. Again compare word counts and TF-IDF. Did PCA succeed in uncovering the differences between the communities?