Exercise 5

Deadline: 10.07.2024, 16:00

This exercise is dedicated to regularized regression. The theoretical tasks represent the level of math proficiency I'd expect from master students in computer science and scientific computing.

Regulations

Please hand in your solution for tasks 1 and 2 as a PDF (either created with LaTeX or another tool of your liking, or scanned from a readable (!) hand-written solution) and for task 3 as a Jupyter notebook `regularized-regression.ipynb`, accompanied with `regularized-regression.html`. Alternatively, you can use Jupiter markdown to write your solutions to tasks 1 and 2 and hand-in everything in a single notebook. **It is important that you stick to these file naming conventions!** Zip all files into a single archive `ex05.zip` and upload this file to MaMPF before the given deadline.
Moreover, please set your **Anzeigename/display name** and **Name in Uebungsgruppen/name in tutorials** in MaMPF to your real name, which should be identical to your name in `muesli` and make sure you **join the submission** of your team via the invitation code before the submission deadline. Check out `https://mampf.blog/handing-in-homework-assignments` for instructions.

1 Bias and variance of ridge regression (8 points)

Ridge regression solves the regularized least squares problem

$$\widehat{\beta}_\tau = \operatorname{argmin}_\beta (y - X\beta)^\top (y - X\beta) + \tau\ \beta^\top \beta$$

with regularization parameter $\tau \geq 0$. Regularization introduces some bias into the solution in order to achieve a potentially large gain in variance. Assume that the true model is $y = X\beta^* + \epsilon$ with zero-mean Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$ and centered features $\frac{1}{N}\sum_i X_i = 0$ (note that these assumptions imply that y is also centered in expectation). Prove (e.g. using the SVD of X) that expectation and covariance matrix of the regularized solution (both taken over all possible training sets of size N) are then given by

$$\mathbb{E}[\widehat{\beta}_\tau] = S_\tau^{-1} S\ \beta^* \qquad \mathrm{Cov}[\widehat{\beta}_\tau] = S_\tau^{-1} S\ S_\tau^{-1} \sigma^2$$

where S and S_τ are the ordinary and regularized scatter matrices:

$$S = X^\top X \qquad S_\tau = X^\top X + \tau\, \mathbb{I}_D$$

Notice that expectation and covariance reduce to the corresponding expressions of ordinary least squares (as derived in the lecture) when $\tau = 0$:

$$\mathbb{E}[\widehat{\beta}_{\tau=0}] = \beta^* \qquad \mathrm{Cov}[\widehat{\beta}_{\tau=0}] = S^{-1}\ \sigma^2$$

Since S_τ is greater than S (in any norm), regularization has a shrinking effect on both expectation and covariance.

2 LDA-Derivation from the Least Squares Error (16 points)

In the lecture, we derived LDA as a generative classifier that fits a Gaussian distribution to the data instances of each class (see exercise 1). Assuming for simplicity that the data are centered (i.e.

we have $\sum_{i=1}^{N} X_i = 0$) and the classes are balanced (i.e. we have $N_1 = N_{-1} = N/2$), this results in the decision rule

$$\widehat{y}_i = \mathrm{sign}(X_i \cdot \widehat{\beta}_{\mathrm{LDA}}) \qquad \text{with} \qquad \widehat{\beta}_{\mathrm{LDA}} = \Sigma^{-1}(\mu_1 - \mu_{-1})^T$$

Here, μ_1 and μ_{-1} are the class means

$$\mu_{-1} = \frac{1}{N_{-1}} \sum_{i:\, y_i^* = -1} X_i \qquad\qquad \mu_1 = \frac{1}{N_1} \sum_{i:\, y_i^* = 1} X_i$$

and Σ is the shared covariance matrix of the two clusters (also known as "within-class covariance")

$$\Sigma = \frac{1}{N}\left[\sum_{i:\, y_i^*=-1} (X_i - \mu_{-1})^T \cdot (X_i - \mu_{-1}) + \sum_{i:\, y_i^*=1} (X_i - \mu_1)^T \cdot (X_i - \mu_1)\right]$$

Thanks to our simplifying assumptions, we don't have to deal with the intercept parameter, because $\widehat{b} = 0$ under these conditions. Recall also that centering and balanced classes imply that $\mu_1 + \mu_{-1} = 0$ (this can be exploited in the derivation below).

You shall show that an equivalent decision rule arises from minimizing the squared loss:

$$\widehat{\beta}_{\mathrm{OLS}} = \mathrm{argmin}_\beta \sum_{i=1}^{N} (y_i^* - X_i \cdot \beta)^2 \quad \Longrightarrow \quad \widehat{\beta}_{\mathrm{OLS}} = \tau \Sigma^{-1}(\mu_1 - \mu_{-1})^T$$

Here $\tau > 0$ is a constant which doesn't alter the sign of $X_i \cdot \widehat{\beta}_{\mathrm{OLS}}$ and therefore leads to the same predictions $\widehat{y}_i$. The benefit of this formulation is that we can apply the machinery of *sparse regression* to perform automatic feature extraction in task 3 below. To derive the expression for $\widehat{\beta}_{\mathrm{OLS}}$, you must set the derivative of the loss with respect to β to zero:

$$\frac{\partial}{\partial \beta} \sum_{i=1}^{N} (y_i^* - X_i \cdot \beta)^2 \stackrel{!}{=} 0 \tag{1}$$

After some algebraic manipulations (deriving these steps is your task here), you should arrive at the expression

$$\Sigma \cdot \beta + \frac{1}{4}(\mu_1 - \mu_{-1})^T \cdot (\mu_1 - \mu_{-1}) \cdot \beta = \frac{1}{2}(\mu_1 - \mu_{-1})^T \tag{2}$$

By noticing that $(\mu_1 - \mu_{-1}) \cdot \beta = \tau'$ for some scalar τ', we can bring the second term of the left hand side to the right hand side and obtain the desired result

$$\Sigma \cdot \beta = \left(\frac{1}{2} - \frac{\tau'}{4}\right)(\mu_1 - \mu_{-1})^T \quad \Longrightarrow \quad \widehat{\beta}_{\mathrm{OLS}} = \tau \Sigma^{-1}(\mu_1 - \mu_{-1})^T$$

with $\tau = 1/2 - \tau'/4$ (we skip the proof that $\tau > 0$ always holds – providing this proof gives bonus points). Derive the steps to go from equation (1) to equation (2).

3 Automatic feature selection for LDA as regression

Sparse regression regularizes the solution such that unimportant elements of $\widehat{\beta}$ are set to zero. The corresponding columns of X could be dropped from the matrix without any effect on the solution – the remaining columns are obviously more useful as an explanation for the response y. Therefore, sparse regression can be interpreted as a method for *relevant* feature identification.

In exercise 1, you implemented dimension reduction from a full image of a digit to just two features by selecting important pixels manually[1]. In this exercise, we will automatically find relevant pixels by means of sparse regression.

[1] or computing your own features

3.1 Implement Orthogonal Matching Pursuit (8 points)

"Orthogonal Matching Pursuit"[2] is a simple greedy sparse regression algorithm. It approximates the exact algorithms for least squares under L_0 or L_1 regularization (cf. the lecture for details) and is defined as:

Initialization:

- Inputs: $X \in \mathbb{R}^{N\times D}$, $y \in \mathbb{R}^N$, $T > 0$ (the desired number of non-zero elements in the final solution $\widehat{\beta}^{(T)}$)
- Define the initial sets of active resp. inactive columns as
  $A^{(0)} = \emptyset$ and $B^{(0)} = \{j : j = 1 \dots D\}$.
- Define the initial residual $r^{(0)} = y$.

Iteration: for $t = 1 \dots T$ do

1. Find the inactive column that has maximal correlation with the current residual:
   $$j^{(t)} = \text{argmax}_{j\in B} \left| X_j^\top \cdot r^{(t-1)} \right|$$
2. Move $j^{(t)}$ from the inactive set B to the active set A.
3. Form the active matrix $X^{(t)}$ consisting of all currently active columns of X.
4. Solve the least squares problem
   $$\widehat{\beta}^{(t)} = \text{argmin}_\beta \, (y - X^{(t)}\beta)^\top (y - X^{(t)}\beta)$$
5. Update the residual $r^{(t)} = y - X^{(t)}\widehat{\beta}^{(t)}$.

Implement this algorithm as a function

```
solutions = omp_regression(X, y, T)
```

which returns $\widehat{\beta}^{(t)}$ for $t = 1 \dots T$ as a $D \times T$ matrix, i.e. the inactive elements in each solution are not dropped, but explicitly set to zero.

3.2 Classification with sparse LDA (8 points)

We again use the `digits` dataset of scikit-learn and classify '3' vs. '9'. For balanced training sets, LDA can be formulated as a least squares problem, where the rows X_i are the flattened images of each digit, and

$$y_i = \begin{cases} 1 & \text{if instance } i \text{ is a '3'} \\ -1 & \text{if instance } i \text{ is a '9'} \end{cases}$$

are the desired responses. Now execute `omp_regression()` with sufficiently big T to get the sequence of sparse LDA solutions for $t = 1 \dots T$.

Report the error rate on the test set for $t = 1 \dots T$. How many pixels should be used for acceptable error rates? Is it necessary/beneficial to standardize the data before training and testing?

Visualize in which order the pixels are switched to *active* as t increases, and show if a pixel votes in favour or against class '3'. What is a good criterion for this distinction, and how can it be intuitively visualized? Compare these results with your hand-crafted feature selection in exercise 1 – did you select the same pixels?

[2] not to be confused with its simpler cousin "Matching Pursuit"