Statistical Machine Learning	Prof. Ulrike von Luxburg + Hein group
Exercise Sheet 5 - 29.5.2024	due: 5.6.2024

Exercise 1 Ridge regression

We consider the regression problem with either L_1 or L_2-loss.

a. **(2 points)** In the following you have to implement least squares and ridge regression (both L_2-loss).

- `w = LeastSquares(Designmatrix,Y):`
  - input: design matrix $\Phi \in \mathbb{R}^{n\times d}$ and the outputs $Y \in \mathbb{R}^n$ (column vector)
  - output: weight vector w of least squares regression as column vector
- `w = RidgeRegression(Designmatrix,Y,Lambda):`
  - input: the design matrix $\Phi \in \mathbb{R}^{n\times d}$, the outputs $Y \in \mathbb{R}^n$ (column vector), and the regularization parameter $\lambda \in \mathbb{R}^+ := \{x \in \mathbb{R} | x \geq 0\}$.
  - output: weight vector w of ridge regression as column vector. Use the non-normalized version: $w = (\Phi^T\Phi + \lambda \mathbb{1}_d)^{-1}\Phi^T Y$.

Note that that the regression with L_1-loss is already provided in `L1LossRegression(Designmatrix,Y,Lambda)`.

b. **(1 Point)** Let us assume $d = 1$. Write a function `Basis`(X, k) to generate the design matrix using the orthogonal Fourier basis functions, with

- input: the input data matrix $X \in \mathbb{R}^{n\times 1}$ and the maximal frequency k of the Fourier basis.
- output: the design matrix $\Phi \in \mathbb{R}^{n\times(2k+1)}$ using the Fourier basis functions:

$$\phi_0(x) = 1, \qquad \phi_{2l-1}(x) = \cos(2\pi l x), \qquad \phi_{2l}(x) = \sin(2\pi l x), \quad l = 1, \ldots, k.$$

c. In the first example we have only one feature ($d = 1$), thus we want to learn a function $f : \mathbb{R} \to \mathbb{R}$. The data is given in `onedim_data.npy`, containing $\texttt{Xtrain}, \texttt{Xtest}, \texttt{Ytrain}, \texttt{Ytest} \in \mathbb{R}^{1000,1}$. First plot the training data, that is the points $(\texttt{Xtrain}_i, \texttt{Ytrain}_i)_{i=1}^{1000}$.

- **(1 Points)** Which loss function (L_1 or L_2) is more appropriate for this kind of data ? Justify this by checking the data plot. Use in the next part only the regression method with your chosen loss (that is either ridge regression or L_1-loss with L_2-regularizer).
- **(1 Points)** Use the basis functions with $k = 1, 2, 3, 5, 10, 15, 20$ from part b. to fit the regularized version of the loss chosen in the previous part. Use regularization parameter $\lambda = 30$. Plot the resulting functions f_k (using as x e.g. 1000 evenly spaced points in $[0, 1]$) for all values of k, together with the training data, with

$$f_k(x) = \langle \phi(x), w^k \rangle = \sum_{i=1}^{2k+1} w_i^k \phi_i(x).$$

  Compute the loss, that is

$$\frac{1}{n}\sum_{i=1}^{n} L(Y_i, f(X_i)),$$

  on the training and test data and plot training and test loss as a function of k.

  Repeat the same for $\lambda = 0$ (unregularized version). How does increasing k affect the estimated functions f_k ? What is the behavior of training and test error for increasing k (explanation on paper)?

d. One observes overfitting when we use a large number k of basis functions. We want to avoid this phenomenon by introducing a normalization of the basis functions according to their complexity. One possible way to do this is to define a measure of complexity $\Omega(f) \in \mathbb{R}^+$ as

$$\Omega(f) = \int_0^1 |f'(x)|^2 \, dx, \tag{1}$$

where $f'(x)$ is the first derivative of f at x and introduce new Fourier basis functions $\{\psi_i(x)\}_{i\in\mathbb{N}^0}$ as

$$\psi_0(x) = \phi_0(x) \qquad \text{and} \qquad \psi_i(x) = \frac{1}{\sqrt{\Omega(\phi_i)}}\phi_i(x), \qquad i \in \mathbb{N}^+,$$

where $\mathbb{N}^0 := \{0, 1, 2, ...\}$, and $\mathbb{N}^+ := \mathbb{N}^0 \backslash \{0\}$.

- **(1 point)** Show that the new Fourier basis functions $\Psi = \{\psi_i\}_{i\in\mathbb{N}^+}$ all have the same complexity $\Omega(\psi_i)$.
- **(1 points)** Derive the explicit form of the new basis functions $\{\psi_i\}_{i\in\mathbb{N}^0}$ and implement a modified function `DesignMatrix = FourierBasisNormalized(X,k):`
  - input: input data matrix $\mathtt{X} \in \mathbb{R}^{n\times 1}$ and maximal frequency k of the Fourier basis
  - output: design matrix $\Phi \in \mathbb{R}^{n\times(2k+1)}$ using the normalized Fourier basis $\{\psi_i\}_{i=0..2k}$
- **(1 points)** Repeat the experiment from part c with both the old (not normalized) basis ϕ_i and the new basis functions ψ_i, using both least squares and ridge regression with regularization parameter $\lambda = 30$ when using ϕ_i and $\lambda = 0.5$ when using ψ_i. How does the new basis functions affect the estimation of the function $f_k = \langle w^k, \Psi(x)\rangle$? What is the difference in terms of the training and test error for the various k (explanation on paper)?

e. **(2 points)** We now consider a modified problem where instead of penalising the weights one directly penalises the gradient of the estimated function $f_w(x) = \langle w, \Psi(x)\rangle$:

$$w^k = \arg\min_{w\in\mathbb{R}^{2k}} \frac{1}{n}\sum_{i=1}^{n} (Y_i - f_w(X_i))^2 + \lambda\,\Omega(f_w)\ ,$$

where $\Omega(f)$ is defined in (1). Show that, when using the normalized Fourier basis $\{\psi_i\}_{i=1..2k}$ without the constant function ψ_0, the above optimization problem is equivalent to ridge regression, that is $\Omega(f_w) = \|w\|^2$.

Zip all plots (.png), scripts (.py), text (.pdf). In addition to the functions mentioned above, there should be scripts to reproduce all the results you submit (plots, losses).

Hints:

- Linear systems, $\mathtt{Ax} = \mathtt{b}$, can be solved with `numpy.linalg.solve(A, b)`.
- In part e you have to use the fact that the functions $(\psi_i)_{i=1..n}$ are mutually orthogonal, i.e.

$$\forall i, j : \int_0^1 \psi_i(x)\psi_j(x)dx = \delta_{ij}c_i, \qquad \text{with } \delta_{ij} = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{else} \end{cases} \text{ and for some } c_i \in \mathbb{R}.$$

- You can load the data with `numpy.load(<file_name>, allow_pickle=True).item()`, which gives a dictionary with the four matrices.

Exercise 2 Lasso

Let $y \in \mathbb{R}^n$ be the n outputs and $\Phi \in \mathbb{R}^{n\times D}$ the design matrix of a regression problem (D basis functions $\phi_1, \ldots, \phi_D$, then $\Phi_{ij} = \phi_j(x_i)$, $i = 1, \ldots, n$, $j = 1, \ldots, D$).

Introducing new variables $w^+, w^- \in \mathbb{R}^D_+$ such that $w = w^+ - w^-$, the Lasso problem

$$\min_{w\in\mathbb{R}^D} \frac{1}{n}\|Y - \Phi w\|_2^2 + \lambda\|w\|_1 \tag{2}$$

can be rewritten into the optimization problem

$$\begin{aligned} \min_{w^+, w^- \in \mathbb{R}^D} \ & \frac{1}{n}\left\|Y - \Phi w^+ + \Phi w^-\right\|_2^2 + \lambda\sum_{i=1}^{D} w_i^+ + \lambda\sum_{i=1}^{D} w_i^- \\ \text{subject to: } \ & w_i^+ \geq 0,\ i = 1, \ldots, D, \\ & w_i^- \geq 0,\ i = 1, \ldots, D. \end{aligned} \tag{3}$$

a. The projection $P_C : \mathbb{R}^d \to \mathbb{R}^d$ onto a closed convex non empty set C is defined for $x \in \mathbb{R}^d$ as

$$P_C(x) := \arg\min_{y \in C} \frac{1}{2} \|x - y\|_2^2 .$$

1. **(3 points)** Show that the projection onto a convex set is uniquely defined (Under which condition on the objective is the global minimum unique ?)
2. **(2 points)** Derive an analytical expression for the projection onto the convex set

$$C = \{x \in \mathbb{R}^d \,|\, x_i \geq 0,\ i = 1, \ldots, d\}, \quad \text{(positive orthant in } \mathbb{R}^d\text{)}.$$

b. We use projected gradient descent which is a simple method for constrained convex optimization. Let C be a convex, closed set and ϕ the differentiable, convex objective function, then the convex optimization problem $\min_{x \in C} \phi(x)$ can be solved via projected gradient descent which is defined as

$$x_{t+1} = P_C(x_t - \alpha_t \nabla \phi(x_t)),$$

where $\alpha_t > 0$ is the stepsize.

1. **(3 points)** Complete the functions needed for `Lasso` which has as arguments Y, Φ, λ and returns the weight vector w of the Lasso problem in Equation (2). Use projected gradient descent for the optimization problem (3).
2. **(2 points)** We use a real dataset, whose data is in `multidim_data_trainval.npy`, with 1300 training points (`Xtrain`, `Ytrain`), 200 test points (`Xval`, `Yval`). The task is to predict the total number of violent crimes per 100K population (output variable $Y_i \in \mathbb{R}$) from a set of features (input variables $X_i \in \mathbb{R}^{100}$) capturing all sorts of properties of the cities and their populations. Before you start do the following preprocessing - for each feature j you compute mean μ_j and standard deviation σ_j from the **training data** and you rescale **training and test data** as $X'_{ij} = \frac{X_{ij} - \mu_j}{\sigma_j}$. The idea behind is that we get rid of different scales of the variables.

   Run your Lasso implementation with linear design with offset, that is the model is $f(x) = \langle w, x \rangle + b$ (add a feature which is 1 for every data point, corresponding to the coefficient of the offset), for the training data with $\lambda = 10$.

   i. Report training and test loss of the obtained model.

   ii. Plot the predicted values by the computed model for the test set vs the true values. What do you observe?

Hints:

- A sum $f + g$ of convex functions f, g is strictly convex if f or g is strictly convex.
- For the computation of the projection, note that each component can be minimized independently of the other ones.