Machine Learning Exercise Sheet 04

Linear Regression

Exercise sheets consist o k and in-class exercises. You solve the homework exercises on your own or with you d upload it to Moodle for a possible grade bonus. The in-class exercises will be sol ring the tutorial. You do not have to upload any solutions of the in-class exercises.

In-class Exercises

Problem 1: Assume that we are given a dataset, where each sample x_i and regression target y_i is generated according to the following process

$$x_i \sim \text{Uniform}(-10, 10)$$
$$y_i = ax_i^3 + bx_i^2 + cx_i + d + \epsilon_i, \quad \text{where} \quad \epsilon_i \sim \mathcal{N}(0, 1) \quad \text{and} \quad a, b, c, d \in \mathbb{R}.$$

The 3 regression algorithms below are applied to the given data. Your task is to say what the bias and variance of these models are (low or high). Provide a 1-2 sentence explanation to each of your answers.

a) Linear regression

> Bias: high. Variance: low.
>
> A straight line cannot capture a degree 3 polynomial (thus underfitting the data).

b) Polynomial regression with degree 3

> Bias: low. Variance: low.
>
> The model is same as the data generating process. We can achieve a good fit.

c) Polynomial regression with degree 10

> Bias: low. Variance: high.
>
> Since we are using a polynomial regression with a degree much higher compared to the data generating process, the model will overfit the data.

Problem 2: Given is a training set consisting of samples $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N)^{\mathrm{T}}$ with respective regression targets $\boldsymbol{y} = (y_1, y_2, \ldots, y_N)^{\mathrm{T}}$ where $\boldsymbol{x}_i \in \mathbb{R}^D$ and $y_i \in \mathbb{R}$.

Alice fits a linear regression model $f(\boldsymbol{x}_i) = \boldsymbol{w}^T\boldsymbol{x}_i$ to the dataset using the closed-form solution for linear regression (normal equations).

Bob has heard that by transforming the inputs $\boldsymbol{x}_i$ with a vector-valued function $\boldsymbol{\Phi}$, he can fit an alternative function, $g(\boldsymbol{x}_i) = \boldsymbol{v}^T\boldsymbol{\Phi}(\boldsymbol{x}$ [illegible] ocedure (solving the normal equations). He decides to use a linear transformation [illegible] $\boldsymbol{A} \in \mathbb{R}^{D\times D}$ has full rank.

a) Show that Bob's p [illegible] ame function as Alice's original procedure, that is $f(\boldsymbol{x}) = g(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathbb{R}^D$ [illegible] minimize the training set error).

Alice uses the no [illegible] y and obtains

$$\boldsymbol{w}^* = \left(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\right)^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}.$$

Bob fits the model to the transformed data and obtains

$$\begin{aligned}
\boldsymbol{v}^* &= \left((\boldsymbol{X}\boldsymbol{A})^{\mathrm{T}}(\boldsymbol{X}\boldsymbol{A})\right)^{-1}(\boldsymbol{X}\boldsymbol{A})^{\mathrm{T}}\boldsymbol{y} \\
&= \left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\boldsymbol{A}\right)^{-1}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y} && (\dagger) \\
&= \boldsymbol{A}^{-1}\left(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\right)^{-1}\left(\boldsymbol{A}^{\mathrm{T}}\right)^{-1}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y} && (\ddagger) \\
&= \boldsymbol{A}^{-1}\left(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\right)^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y} = \boldsymbol{A}^{-1}\boldsymbol{w}^*.
\end{aligned}$$

(Note that $\boldsymbol{\Phi}$ transforms the column vectors $\boldsymbol{x}_i$ via $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{x}_i$ but $\boldsymbol{X}$ contains the transposed observations $\boldsymbol{x}_i^{\mathrm{T}}$ as rows. Therefore the transformed feature matrix is $\boldsymbol{X}\boldsymbol{A}$.)

Now it is immediate to see that

$$g(\boldsymbol{x}_i) = \boldsymbol{v}^{*\mathrm{T}}\boldsymbol{\Phi}(\boldsymbol{x}_i) = \boldsymbol{w}^{*\mathrm{T}}\boldsymbol{A}^{-\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{x}_i = \boldsymbol{w}^{*\mathrm{T}}\boldsymbol{x}_i = f(\boldsymbol{x}_i).$$

b) Can Bob's procedure lead to a lower training set error than Alice's if the matrix $\boldsymbol{A}$ is not invertible? Explain your answer.

Any weights $\boldsymbol{v}^*$ Bob finds are also feasible for Alice by letting $\boldsymbol{w} = \boldsymbol{A}\boldsymbol{v}^*$. Therefore Bob can only access a subset of the parameter space and cannot achieve a lower loss value than Alice. (It could still be equal but it cannot be better.)

Note that we are only talking about training error in this example, not test error. Bob might manage to find a model that generalizes better than Alice's, but Alice will always be able to fit the training data at least as well as Bob.

Problem 3: See Jupyter notebook `inclass_04_notebook.ipynb`.

Homework

Least squares regression

Problem 4: Let's assum[illegible] where each datapoint, $(\boldsymbol{x}_i, y_i)$ is weighted by a scalar factor which we will call t_i. We[illegible] > 0 for all i. This makes the sum of squares error function look like the following:

$$[illegible] = \frac{1}{2} \sum_{i=1}^{N} t_i \left[\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i) - y_i\right]^2$$

Find the equation for the value of $\boldsymbol{w}$ that minimizes this error function.

Furthermore, explain how this weighting factor, t_i, can be interpreted in terms of
1) the variance of the noise on the data and
2) data points for which there are exact copies in the dataset.

If we define $\boldsymbol{T} = \operatorname{diag}(t_1, \dots, t_N)$ to be a diagonal matrix with $\boldsymbol{t}$ on the diagonal, we can write the weighted sum-of-squares cost function in the form

$$E_{\text{weighted}}(\boldsymbol{w}) = \frac{1}{2}(\boldsymbol{\Phi w} - \boldsymbol{y})^T \boldsymbol{T} (\boldsymbol{\Phi w} - \boldsymbol{y}).$$

Now we follow along the same steps that we used to derive the optimal solution for ordinary least squares and arrive at

$$\boldsymbol{w}^*_{\text{weighted}} = (\boldsymbol{\Phi}^T \boldsymbol{T} \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^T \boldsymbol{T} \boldsymbol{y}.$$

Can we understand weighted linear regression in a probabilistic context as we did with ordinary least squares? There we had modeled the regression targets as i.i.d. random variables

$$y_i \sim \mathcal{N}\left(\boldsymbol{w}^T \phi(\boldsymbol{x}_i), \beta^{-1}\right)$$

with a common noise precision of β. From this we derived the form of the maximum likelihood error (negative log-likelihood) as

$$E_{\text{ML}}(\boldsymbol{w}, \beta) = \beta \underbrace{\frac{1}{2} \sum_{i=1}^{N} (\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i) - y_i)^2}_{E_{\text{LS}}} - \underbrace{\frac{N}{2} \ln \beta + \frac{N}{2} \ln 2\pi}_{\text{constant w.r.t. } \boldsymbol{w}}.$$

But the least squares error part just stems from the definition of the normal distribution marked with (†) below.

$$\mathcal{N}\left(y_i \mid \boldsymbol{w}^{\mathrm{T}} \phi(\boldsymbol{x}_i), \beta^{-1}\right) \propto \exp\left(-\overbrace{\frac{\beta}{2}(\boldsymbol{w}^{\mathrm{T}} \phi(\boldsymbol{x}_i) - y_i)^2}^{(\dagger)}\right)$$

From this we deduce that weighted least squares is equivalent to probabilistic least squares where we choose $\beta = t_i$, in effect modeling y_i as

$$y_i \sim \mathcal{N}\left(\boldsymbol{w}^T \phi(\boldsymbol{x}_i), t_i^{-1}\right)$$

Upload a single PDF file with your homework solution to Moodle by 17.11.2021, 23:59 CET. We recommend to typeset your solution (using LaTeX or Word), but handwritten solutions are also accepted. If your handwritten solution is illegible, it won't be graded and you waive your right to dispute that.

making the regression targets no longer identically distributed but still independent.

For $t_i \in \mathbb{N}$, t_i can be regarded as an *effective* number of replicated observations of data point $(\boldsymbol{x}_i, y_i)$.

Ridge regression

Problem 5: Show that ridge regression with design matrix $\boldsymbol{\Phi} \in \mathbb{R}^{N \times M}$ with regularization strength λ is equivalent to ordinary least squares regression with an augmented design matrix and target vector

$$\hat{\boldsymbol{\Phi}} = \begin{pmatrix} \boldsymbol{\Phi} \\ \sqrt{\lambda}\boldsymbol{I}_M \end{pmatrix} \quad \text{and} \quad \hat{\boldsymbol{y}} = \begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{0}_M \end{pmatrix}.$$

Ordinary least squares minimizes $\frac{1}{2}(\boldsymbol{\Phi w} - \boldsymbol{y})^T(\boldsymbol{\Phi w} - \boldsymbol{y})$. Plugging in the augmented design matrix and regression target, we get

$$\frac{1}{2}(\hat{\boldsymbol{\Phi}}\boldsymbol{w} - \hat{\boldsymbol{y}})^T(\hat{\boldsymbol{\Phi}}\boldsymbol{w} - \hat{\boldsymbol{y}}) = \frac{1}{2}(\boldsymbol{\Phi w} - \boldsymbol{y})^T(\boldsymbol{\Phi w} - \boldsymbol{y}) + \frac{1}{2}\left(\sqrt{\lambda}\boldsymbol{I}_M\boldsymbol{w}\right)^{\mathrm{T}}\left(\sqrt{\lambda}\boldsymbol{I}_M\boldsymbol{w}\right)$$
$$= \frac{1}{2}\sum_{i=1}^{N}(\boldsymbol{\Phi}_i\boldsymbol{w} - y_i)^2 + \frac{\lambda}{2}\|\boldsymbol{w}\|_2^2$$

which is equal to the ridge regression loss function.

Implementation

Problem 6: John Doe is a data scientist, and he wants to fit a polynomial regression model to his data. For this, he needs to choose the degree of the polynomial that works best for his problem.

Unfortunately, John hasn't attended IN2064, so he writes the following code for choosing the optimal degree of the polynomial:

```
X, y = load_data()
best_error = -1
best_degree = None

for degree in range(1, 50):
    w = fit_polynomial_regression(X, y, degree)
    y_predicted = predict_polynomial_regression(X, w, degree)
    error = compute_mean_squared_error(y, y_predicted)
    if (error <= best_error) or (best_error == -1):
        best_error = error
        best_degree = degree

print("Best degree is " + str(best_degree))
```

Assume that the functions are implemented correctly and do what their name suggests.

a) Explain briefly why this code doesn't do what it's supposed to do.

> Output: `Best degree is 49`
>
> Error on the trai[illegible]own when we use a higher degree polynomial (unless it's already 0, then i[illegible]

b) Describe a possible [illegible]m with this code. (You don't need to write any code, just describe the appro[illegible]

> Split data into train and validation sets. Choose the degree that achieves the lowest mean squared error on the validation set (not on the training set!).
>
> *Remark: Regularization does not help at all in this case. No matter which λ you choose, higher degree polynomial is still able to fit the training data better.*

Bayesian linear regression

Bishop 3.12

In the lecture we made the assumption that we already knew the precision (inverse variance) for our Gaussian distributions. What about when we don't know the precision and we need to put a prior on that as well as our Gaussian prior that we already have on the weights of the model?

Problem 7: It turns out that the conjugate prior for the situation when we have an unknown mean and unknown precision is a normal-gamma distribution (See section 2.3.6 in Bishop). This means that if our likelihood is as follows:

$$\mathrm{p}(\boldsymbol{y} \mid \boldsymbol{\Phi}, \boldsymbol{w}, \beta) = \prod_{i=1}^{N} \mathcal{N}(y_i \mid \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i), \beta^{-1})$$

Then the conjugate prior for both $\boldsymbol{w}$ and β is

$$\mathrm{p}(\boldsymbol{w}, \beta) = \mathcal{N}(\boldsymbol{w} \mid \boldsymbol{m}_0, \beta^{-1}\boldsymbol{S}_0)\,\mathrm{Gamma}(\beta \mid a_0, b_0)$$

Show that the posterior distribution takes the same form as the prior, i.e.

$$\mathrm{p}(\boldsymbol{w}, \beta \mid \mathcal{D}) = \mathcal{N}(\boldsymbol{w} \mid \boldsymbol{m}_N, \beta^{-1}\boldsymbol{S}_N)\,\mathrm{Gamma}(\beta \mid a_N, b_N)$$

Also be sure to give the expressions for $\boldsymbol{m}_N$, $\boldsymbol{S}_N$, a_N, and b_N.

Hint: Expand $\log \mathrm{p}(\boldsymbol{w}, \beta \mid \mathcal{D})$ once with the prior and likelihood and once with the presumed posterior form. The resulting expressions have to be equal, so you should be able to match all terms in the two expansions against each other and then read off the parameters of the posterior distribution.

It is easiest to work in log space. The log of the posterior distribution is given by

$$
\begin{aligned}
\log \mathrm{p}(\boldsymbol{w}, \beta \mid \mathcal{D}) &= \log p(\boldsymbol{w}, \beta) + \sum_{i=1}^{N} \log p(y_i \mid \boldsymbol{w}^T \phi(\boldsymbol{x}_i), \beta^{-1}) + \text{const.} \\
&= [illegible] (\boldsymbol{w} - \boldsymbol{m}_0)^T \boldsymbol{S}_0^{-1} (\boldsymbol{w} - \boldsymbol{m}_0) - b_0 \beta + (a_0 - 1) \log \beta \\
&\quad [illegible] \sum_{i=1}^{N} \{\boldsymbol{w}^T \phi(\boldsymbol{x}_i) - y_i\}^2 + \text{const.}
\end{aligned}
$$

Now we use the prod[illegible] posterior distribution as $\log \mathrm{p}(\boldsymbol{w}, \beta \mid \mathcal{D}) = \log \mathrm{p}(\boldsymbol{w} \mid \beta, \mathcal{D}) + \log \mathrm{p}(\beta \mid \mathcal{D})$ and begin some elaborate pattern matching. Consider first the posterior of $\boldsymbol{w}$. The general form of $\log \mathrm{p}(\boldsymbol{w} \mid \beta, \mathcal{D})$ is given by

$$
-\frac{\beta}{2}(\boldsymbol{w} - \boldsymbol{m}_N)^{\mathrm{T}} \boldsymbol{S}_N^{-1} (\boldsymbol{w} - \boldsymbol{m}_N) + \frac{M}{2} \log \beta + \text{const.}
$$

which we expand to

$$
-\frac{\beta}{2} \boldsymbol{w}^{\mathrm{T}} \boldsymbol{S}_N^{-1} \boldsymbol{w} + \beta \boldsymbol{m}_N^{\mathrm{T}} \boldsymbol{S}_N^{-1} \boldsymbol{w} - \frac{\beta}{2} \boldsymbol{m}_N^{\mathrm{T}} \boldsymbol{S}_N^{-1} \boldsymbol{m}_N + \frac{M}{2} \log \beta + \text{const.}
$$

If we can bring some part of the combined likelihood into this form, we can read off the parameters $\boldsymbol{m}_N$ and $\boldsymbol{S}_N$. We begin by rewriting

$$
\frac{\beta}{2} \sum_{i=1}^{N} \{\boldsymbol{w}^T \phi(\boldsymbol{x}_i) - y_i\}^2 = \frac{\beta}{2} (\boldsymbol{\Phi} \boldsymbol{w} - \boldsymbol{y})^{\mathrm{T}} (\boldsymbol{\Phi} \boldsymbol{w} - \boldsymbol{y}) = \frac{\beta}{2} \boldsymbol{w}^{\mathrm{T}} \boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi} \boldsymbol{w} - \beta \boldsymbol{y}^{\mathrm{T}} \boldsymbol{\Phi} \boldsymbol{w} + \frac{\beta}{2} \boldsymbol{y}^{\mathrm{T}} \boldsymbol{y}
$$

and

$$
\frac{\beta}{2} (\boldsymbol{w} - \boldsymbol{m}_0)^{\mathrm{T}} \boldsymbol{S}_0^{-1} (\boldsymbol{w} - \boldsymbol{m}_0) = \frac{\beta}{2} \boldsymbol{w}^{\mathrm{T}} \boldsymbol{S}_0^{-1} \boldsymbol{w} - \beta \boldsymbol{m}_0^{\mathrm{T}} \boldsymbol{S}_0^{-1} \boldsymbol{w} + \frac{\beta}{2} \boldsymbol{m}_0^{\mathrm{T}} \boldsymbol{S}_0^{-1} \boldsymbol{m}_0
$$

Now we have to collect all quadratic terms in $\boldsymbol{w}$ and get

$$
\begin{aligned}
\log \mathrm{p}(\boldsymbol{w}, \beta \mid \mathcal{D}) = \frac{M}{2} \log \beta - \frac{\beta}{2} \boldsymbol{w}^{\mathrm{T}} (\boldsymbol{S}_0^{-1} + \boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi}) \boldsymbol{w} + \beta (\boldsymbol{m}_0^{\mathrm{T}} \boldsymbol{S}_0^{-1} + \boldsymbol{y}^{\mathrm{T}} \boldsymbol{\Phi}) \overbrace{\boldsymbol{S}_N \boldsymbol{S}_N^{-1}}^{!} \boldsymbol{w} \\
+ \left(\frac{N}{2} + a_0 - 1\right) \log \beta - \left(b_0 + \frac{1}{2} \boldsymbol{y}^{\mathrm{T}} \boldsymbol{y} + \frac{1}{2} \boldsymbol{m}_0^{\mathrm{T}} \boldsymbol{S}_0^{-1} \boldsymbol{m}_0\right) \beta + \text{const.}
\end{aligned}
$$

This is already quite close and in this form we can already match term by term with the general form of $\log \mathrm{p}(\boldsymbol{w} \mid \beta, \mathcal{D})$ and see that

$$
\boldsymbol{m}_N = \left((\boldsymbol{m}_0^{\mathrm{T}} \boldsymbol{S}_0^{-1} + \boldsymbol{y}^{\mathrm{T}} \boldsymbol{\Phi}) \boldsymbol{S}_N \right)^{\mathrm{T}} \quad \text{and} \quad \boldsymbol{S}_N = \left(\boldsymbol{S}_0^{-1} + \boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi} \right)^{-1}.
$$

However, to actually bring the likelihood into the required form, we need to complete the square $\log \mathrm{p}(\boldsymbol{w}, \beta \mid \mathcal{D})$.

$$
\overbrace{\frac{M}{2} \log \beta - \frac{\beta}{2} \boldsymbol{w}^{\mathrm{T}} \left(\boldsymbol{S}_0^{-1} + \boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi} \right) \boldsymbol{w} + \beta \left(\boldsymbol{m}_0^{\mathrm{T}} \boldsymbol{S}_0^{-1} + \boldsymbol{y}^{\mathrm{T}} \boldsymbol{\Phi} \right) \boldsymbol{S}_N \boldsymbol{S}_N^{-1} \boldsymbol{w} - \frac{\beta}{2} \boldsymbol{m}_N^{\mathrm{T}} \boldsymbol{S}_N^{-1} \boldsymbol{m}_N}^{\log \mathrm{p}(\boldsymbol{w} \mid \beta, \mathcal{D})}
$$

$$
\underbrace{+ \frac{\beta}{2} \boldsymbol{m}_N^{\mathrm{T}} \boldsymbol{S}_N^{-1} \boldsymbol{m}_N + \left(\frac{N}{2} + a_0 - 1 \right) \log \beta - \left(b_0 + \frac{1}{2} \boldsymbol{y}^{\mathrm{T}} \boldsymbol{y} + \frac{1}{2} \boldsymbol{m}_0^{\mathrm{T}} \boldsymbol{S}_0^{-1} \boldsymbol{m}_0 \right) \beta}_{\log \mathrm{p}(\beta \mid \mathcal{D})} + \text{const.}
$$

Next we repeat the same pattern matching with β. The general form of $\log \mathrm{p}(\beta \mid \mathcal{D})$ is

$$\log \mathrm{p}(\beta \mid \mathcal{D}) = (a_N - 1)\log\beta - b_N\beta + \text{const.}$$

Luckily, all the hard w[illegible]nd we read off immediately

$$a_N = \frac{N}{2} \; [illegible] \;) + \frac{1}{2}\left(\boldsymbol{m}_0^T \boldsymbol{S}_0^{-1}\boldsymbol{m}_0 - \boldsymbol{m}_N^T \boldsymbol{S}_N^{-1}\boldsymbol{m}_N + \boldsymbol{y}^{\mathrm{T}}\boldsymbol{y}\right).$$

Problem 8: Derive the closed form solution for ridge regression error function

$$E_{\text{ridge}}(\boldsymbol{w}) = \frac{1}{2}\sum_{i=1}^{N}(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_i) - y_i)^2 + \frac{\lambda}{2}\boldsymbol{w}^T\boldsymbol{w}$$

Additionally, discuss the scenario when the number of training samples N is smaller than the number of basis functions M. What computational issues arise in this case? How does regularization address them?

$$\begin{aligned} E_{\text{ridge}}(\boldsymbol{w}) &= \frac{1}{2}\sum_{i=1}^{N}(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_i) - y_i)^2 + \frac{\lambda}{2}\boldsymbol{w}^T\boldsymbol{w} \\ &= \frac{1}{2}(\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{y})^T(\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{y}) + \frac{\lambda}{2}\boldsymbol{w}^T\boldsymbol{w} \end{aligned}$$

Taking the gradient

$$\begin{aligned} \nabla_{\boldsymbol{w}} E_{\text{ridge}}(\boldsymbol{w}) &= \boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{\Phi}^T\boldsymbol{y} + \lambda\boldsymbol{w} \\ &= (\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \lambda\boldsymbol{I})\boldsymbol{w} - \boldsymbol{\Phi}^T\boldsymbol{y} \end{aligned}$$

Set it to zero

$$\begin{aligned} (\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \lambda\boldsymbol{I})\boldsymbol{w} &= \boldsymbol{\Phi}^T\boldsymbol{y} \\ \boldsymbol{w} &= (\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \lambda\boldsymbol{I})^{-1}\boldsymbol{\Phi}^T\boldsymbol{y} \end{aligned}$$

If $N < M$ the covariance matrix $\boldsymbol{\Phi}^T\boldsymbol{\Phi} \in \mathbb{R}^{M\times M}$ will be singular, therefore not invertible. (this may happen even if $N \geq M$, e.g. when some features are correlated).

When regularization is used, $\lambda\boldsymbol{I}$ is added to the covariance matrix, thus fixing the potential degeneracy issue and making the problem tractable.

Comparison of Linear Regression Models

Problem 9: We want to perform regression on a dataset consisting of N samples $\boldsymbol{x}_i \in \mathbb{R}^D$ with corresponding targets $y_i \in \mathbb{R}$ (represented compactly as $\boldsymbol{X} \in \mathbb{R}^{N\times D}$ and $\boldsymbol{y} \in \mathbb{R}^N$).

Assume that we have fitted an L_2-regularized linear regression model and obtained the optimal weight vector $\boldsymbol{w}^* \in \mathbb{R}^D$ as

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \frac{1}{2} \sum_{i=1}^{N} (\boldsymbol{w}^T \boldsymbol{x}_i - y_i)^2 + \frac{\lambda}{2} \boldsymbol{w}^T \boldsymbol{w}$$

Note that there is no bia[illegible]

Now, assume that we obt[illegible]rix $\boldsymbol{X}_{new}$ by scaling all samples by the same positive factor $a \in (0, \infty)$. That is, $\boldsymbol{X}_{ne}$[illegible]ively $\boldsymbol{x}_i^{new} = a\boldsymbol{x}_i$).

a) Find the weight ve[illegible]produce the same predictions on $\boldsymbol{X}_{new}$ as $\boldsymbol{w}^*$ produces on $\boldsymbol{X}$.

Predictions of a linear regression model are generated as $\hat{y} = \boldsymbol{w}^T \boldsymbol{x}$.

This means that we need to ensure that $\boldsymbol{w}^{*T} \boldsymbol{x} = \boldsymbol{w}_{new}^T \boldsymbol{x}_i^{new}$ or equivalently $\boldsymbol{w}^{*T} \boldsymbol{x}_i = \boldsymbol{w}_{new}^T a \boldsymbol{x}_i$. Solving for $\boldsymbol{w}_{new}$ we get $\boldsymbol{w}_{new} = \frac{\boldsymbol{w}^*}{a}$

b) Find the regularization factor $\lambda_{new} \in \mathbb{R}$, such that the solution $\boldsymbol{w}^*_{new}$ of the new L_2-regularized linear regression problem

$$\boldsymbol{w}^*_{new} = \arg\min_{\boldsymbol{w}} \frac{1}{2} \sum_{i=1}^{N} (\boldsymbol{w}^T \boldsymbol{X}_i^{new} - y_i)^2 + \frac{\lambda_{new}}{2} \boldsymbol{w}^T \boldsymbol{w}$$

will produce the same predictions on $\boldsymbol{X}_{new}$ as $\boldsymbol{w}^*$ produces on $\boldsymbol{X}$.

Provide a mathematical justification for your answer.

The closed form solution for $\boldsymbol{w}^*$ on the original data $\boldsymbol{X}$ is

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

The closed form solution for $\boldsymbol{w}^*_{new}$ on the new data $\boldsymbol{X}_{new}$ is

$$\begin{aligned} \boldsymbol{w}^*_{new} &= (\boldsymbol{X}_{new}^T \boldsymbol{X}_{new} + \lambda_{new} \boldsymbol{I})^{-1} \boldsymbol{X}_{new}^T \boldsymbol{y} \\ &= a(a^2 \boldsymbol{X}^T \boldsymbol{X} + \lambda_{new} \boldsymbol{I})^{-1} \boldsymbol{X}^T \boldsymbol{y} \end{aligned}$$

by setting $\lambda_{new} = a^2 \lambda$, we get

$$\begin{aligned} &= a(a^2 \boldsymbol{X}^T \boldsymbol{X} + a^2 \lambda \boldsymbol{I})^{-1} \boldsymbol{X}^T \boldsymbol{y} \\ &= \frac{1}{a} (\boldsymbol{X}^T \boldsymbol{X} + \lambda \boldsymbol{I})^{-1} \boldsymbol{X}^T \boldsymbol{y} \\ &= \frac{1}{a} \boldsymbol{w}^* \end{aligned}$$

Which (according to our answer in part (a) of this problem) will produce the same predictions on $\boldsymbol{X}_{new}$ as $\boldsymbol{w}^*$ does on $\boldsymbol{X}$, as desired.

Upload a single PDF file with your homework solution to Moodle by 17.11.2021, 23:59 CET. We recommend to typeset your solution (using LaTeX or Word), but handwritten solutions are also accepted. If your handwritten solution is illegible, it won't be graded and you waive your right to dispute that.

Equivalent solution

$$\boldsymbol{w}^*_{new} \overset{!}{=} \boldsymbol{w}^* \ldots 1 \ldots \arg\min \ldots \frac{1}{\ldots} \sum_{i=1}^{N} (\boldsymbol{w}^T \boldsymbol{x}_i - y_i)^2 + \frac{\lambda}{2} \boldsymbol{w}^T \boldsymbol{w}$$

[illegible]

$$\ldots \sum_{i=1}^{N} (\frac{\boldsymbol{w}^T}{a} a \boldsymbol{x}_i - y_i)^2 + \frac{a^2 \lambda}{2} \frac{\boldsymbol{w}^T}{a} \frac{\boldsymbol{w}}{a}$$

[illegible]

$$\ldots \sum_{i=1}^{N} (\boldsymbol{w}_{new}^T \boldsymbol{x}_i^{new} - y_i)^2 + \frac{a^2 \lambda}{2} \boldsymbol{w}_{new}^T \boldsymbol{w}_{new}$$

$$\overset{!}{=} \boldsymbol{w}^*_{new} = \arg\min_{\boldsymbol{w}_{new}} \frac{1}{2} \sum_{i=1}^{N} (\boldsymbol{w}_{new}^T \boldsymbol{x}_i^{new} - y_i)^2 + \frac{\lambda_{new}}{2} \boldsymbol{w}_{new}^T \boldsymbol{w}_{new}$$

For this equality to hold we need to match the regularization term by setting $\lambda_{new} = a^2 \lambda$.

Programming Task

Problem 10: Download the notebook `exercise_04_linear_regression.ipynb` from Moodle. Fill in the missing code and follow the instructions in the notebook to append the solution to your PDF submission.

Note: We suggest that you use Anaconda for installing Python and Jupyter, as well as for managing packages. We recommend that you use Python 3.

For more information on Jupyter notebooks and how to convert them to other formats, consult the Jupyter documentation and nbconvert documentation.