Statistical Machine Learning: Exercise 3

Regression, Kernel Theory, Gaussian Processes, Latent Representation
Total Possible Points: 82

Prof. Marcus Rohrbach, Prof. Simone Schaub-Meyer

TECHNISCHE UNIVERSITÄT DARMSTADT

Summer Term 2025

Group 210
Mateo Arenas
Diusne Olavide

Task 1: Regression (34 Points)

Programming Task

Google Colab: https://colab.research.google.com/drive/1kF9azPGS7ZNYaO0_rhi0_j-06X-a27XU?usp=sharing

In this exercise, you will implement various kinds of linear regressors using the data `lin_reg_train.txt` and `lin_reg_test.tx`
The files contain noisy observations from an unknown function $f : \mathbb{R} \mapsto \mathbb{R}$. In both files, the first column represents the input and the second column represents the output. You can load the data using `numpy.loadtxt` and built-in functions for computing the mean.

For all subtasks, assume that the data is identically and independently distributed according to

$$y_i = \mathbf{\Phi}(\mathbf{x_i})^\top \mathbf{w} + \epsilon_i,$$

where

$$\epsilon_i \sim \mathcal{N}(0, \sigma^2),$$

and $\Phi : \mathbb{R} \to \mathbb{R}^n$ is a feature transformation such that

$$\mathbf{y} \sim \mathcal{N}(\mathbf{\Phi}(\mathbf{X})^\top \mathbf{w}, \sigma^2 \mathbf{I}).$$

Additionally, make sure that your implementations support multivariate inputs. The feature transformations are given in each task; if no basis function is stated explicitly, use the data as is, i.e. $\mathbf{\Phi}(x) = x$.

1a) Linear Features (8 Points)

Implement linear ridge regression using linear features, i.e. the data itself by filling in the ToDos of the corresponding task in the provided Colab Template. Include an additional input dimension to represent a bias term and use the ridge coefficient $\lambda = 0.01$.

1. Explain: What is the ridge coefficient and why do we use it?
2. Derive the optimal model parameters by minimizing the squared error loss function.
3. Report the root mean squared error of the train and test data under your linear model with linear features.
4. Include the resulting plot and a short description.

Solution:

1. Explain: What is the ridge coefficient and why do we use it?

In the lecture (Slide 42), the regularized least squares problem is introduced as the Maximum A-Posteriori (MAP) estimate in a Bayesian linear regression setting. The coefficient function is:

$$w = \arg\min_{w} \frac{1}{2}\|\Phi^\top w - y\|^2 + \frac{\lambda}{2}\|w\|^2$$

The term λ controls the complexity of the model and determines the degree of overfitting. A smaller λ allows the model to fit the training data more precisely, while a larger λ reduces the overfitting.

The objective is to avoid unstable "valleys". The ones that appear when using *Least squares*, as seen in the figure 1.

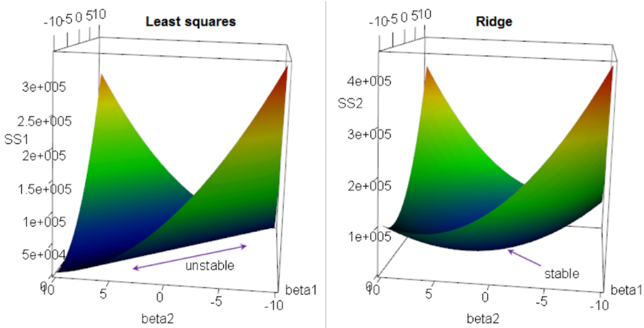

Figure 1: Example. The difference when using Ridge vs. Least squares

2. Derive the optimal model parameters by minimizing the squared error loss function.

As shown in Slide 42 of the lecture, we minimize the regularized least squares loss:

$$L(w) = \frac{1}{2}\|\Phi^\top w - y\|^2 + \frac{\lambda}{2}\|w\|^2$$

Taking the derivative with respect to w and setting it to zero:

$$\frac{\partial L}{\partial w} = \Phi(\Phi^\top w - y) + \lambda w = 0$$

Rearranging:

$$\Phi\Phi^\top w + \lambda w = \Phi y$$

Solving for w:

$$w = (\Phi\Phi^\top + \lambda I)^{-1}\Phi y$$

This is the closed-form solution for regularized linear regression (ridge regression).

3. Report the root mean squared error of the train and test data under your linear model with linear features.

The result of the RMSE of the train and test data for the linear model and features is the following:

`Linear Features`

- `Train RMSE: 0.4122`
- `Test RMSE: 0.3843`

The RMSE values are relatively low and close to each other, indicating that the model generalizes well to unseen data and does not overfit. This is expected for a simple linear model with mild regularization on a smooth dataset.

4. Include the resulting plot and a short description.

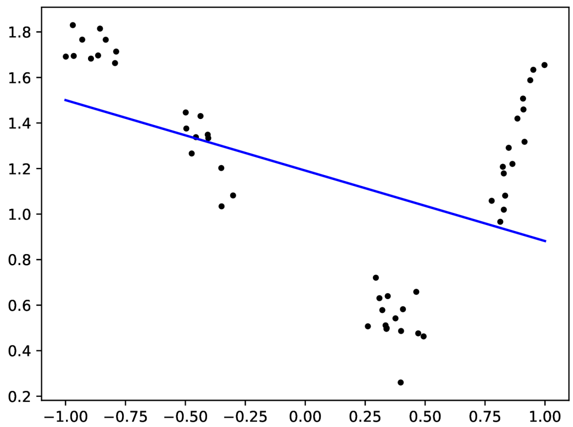

Figure 2: Linear ridge regression fit with $\lambda = 0.01$ using linear features and bias.

The model fits a straight line through the data. Since the data shows a clear nonlinear pattern, the linear model cannot capture it well. Still, the fit shows a general trend and does not overfit the noise, which explains the relatively low RMSE values.

1b) Polynomial Features (8 Points)

Implement linear ridge regression using a polynomial feature projection by filling in the ToDos of the corresponding task in the provided Colab Template. Include an additional input dimension to represent a bias term and use the ridge coefficient $\lambda = 0.01$.

For polynomials of degrees 2, 3 and 4:

1. Report the root mean squared error of the training data and of the testing data under your model with polynomial features.
2. Include the resulting plot and a short description.
3. Why do we call this method *linear* regression despite using polynomials?

Solution:

1. Report the root mean squared error of the training data and of the testing data under your model with polynomial features.

We trained ridge regression models using polynomial features of degrees 2, 3, and 4, with a regularization parameter $\lambda = 0.01$ and a bias term. These are the results:

- **Degree 2:**
  - `Train RMSE: 0.2120`
  - `Test RMSE: 0.2169`
- **Degree 3:**
  - `Train RMSE: 0.0871`
  - `Test RMSE: 0.1084`
- **Degree 4:**
  - `Train RMSE: 0.0870`
  - `Test RMSE: 0.1067`

with a higher degree of the polynomial features, the model's ability to fit the data. This, represented by the lower RMSE values. The model with degree 3 achieves a good balance between training and test error, while the degree 4 model slightly improves the test performance without overfitting.

2. Include the resulting plot and a short description.

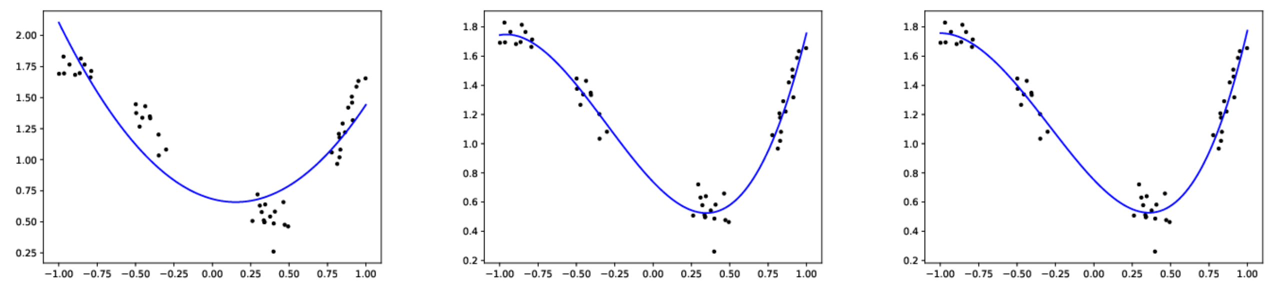

Figure 3: Polynomial ridge regression fits with degrees 2 (left), 3 (center), and 4 (right).

The degree 2 model captures the general U-shape trend but misses the fine details on the ends. The degree 3 model fits the data much more accurately. The degree 4 model performs similarly to degree 3, with no relevant visual differences. The improvement from degree 2 to 3 is significant, while the gain from degree 3 to 4 is minor.

3. Why do we call this method *linear* regression despite using polynomials?

Even though we use polynomial functions of the input, the model is still linear in the parameters w. The prediction is a weighted sum of the features, and we do not multiply or compose the parameters. That's why it's still called linear regression.

$$f(x) = w^\top \phi(x) = w_0 + w_1 x + w_2 x^2 + \cdots + w_d x^d$$

1c) Bayesian Linear Regression (10 Points)

Implement Bayesian linear ridge regression by filling in the ToDos of the corresponding task in the provided Colab Template. Assuming that **w** follows a multivariate Gaussian distribution, such that

$$\mathbf{w} \sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Lambda}_0^{-1}),$$

where ridge regression dictates $\boldsymbol{\mu}_0 = \mathbf{0}$ and $\boldsymbol{\Lambda}_0 = \lambda\mathbf{I}$.

Here, $\boldsymbol{\mu}_0$ is the prior weight mean and $\boldsymbol{\Lambda}_0$ is the prior weight precision matrix, i.e. the inverse of the covariance matrix. The corresponding posterior parameters can be denoted as $\boldsymbol{\mu}_n$ and $\boldsymbol{\Lambda}_n$.

Assume $\sigma = 0.1$, use $\lambda = 0.01$, and include an additional input dimension to represent a bias term. Use all of the provided training data for a single Bayesian update.

1. State the posterior distribution of the model parameters $p(\mathbf{w} \mid \mathbf{X}, \mathbf{y})$ (no derivation required).
2. State the predictive distribution $p(\mathbf{y}_* \mid \mathbf{X}_*, \mathbf{X}, \mathbf{y})$ (no derivation required).
3. Report the RMSE of the train and test data under your Bayesian model (use the predictive mean).
4. Report the average log-likelihood of the train and test data under your Bayesian model.
5. Include the resulting plot and a short description.
6. Explain the differences between linear regression and Bayesian linear regression.

Solution:

1. State the posterior distribution of the model parameters $p(w|X, y)$ (no derivation required).

We place a Gaussian prior on the weight vector:

$$w \sim \mathcal{N}(0, \lambda^{-1} I)$$

Given the likelihood

$$p(y \mid X, w) = \mathcal{N}(y \mid \Phi w, \sigma^2 I)$$

and the prior, we compute the posterior using Bayes' rule:

$$p(w \mid X, y) \propto p(y \mid X, w)\, p(w)$$

Because both the likelihood and prior are Gaussian, the posterior is also Gaussian:

$$p(w \mid X, y) = \mathcal{N}(w \mid \mu, \Lambda^{-1})$$

where the parameters are:

$$\Lambda = \lambda I + \frac{1}{\sigma^2}\Phi^\top \Phi, \qquad \mu = \frac{1}{\sigma^2}\Lambda^{-1}\Phi^\top \mathbf{y}$$

2. State the predictive distribution $p(y_*|X_*, X, y)$ (no derivation required).

For a new input x_*, the output y_* is also Gaussian-distributed, even after marginalizing over the uncertainty in w:

$$p(y_* \mid x_*, X, y) = \int p(y_* \mid x_*, w)\, p(w \mid X, y)\, dw$$

Both terms are Gaussian, so the result is Gaussian as well:

$$p(y_* \mid x_*, X, y) = \mathcal{N}(y_* \mid \mu^\top \phi(x_*),\, \sigma^2 + \phi(x_*)^\top \Lambda^{-1} \phi(x_*))$$

The mean and variance of the predictive distribution is:

$$\mathbb{E}[y^*] = \mu^\top \phi(x^*) \qquad \text{Var}(y^*) = \sigma^2 + \phi(x^*)^\top \Lambda^{-1} \phi(x^*)$$

3. Report the RMSE of the train and test data under your Bayesian model (use the predictive mean).

The result of the RMSE of the train and test data for the Bayesian Linear Regression is:

Bayesian Linear Regression

- `Train RMSE: 0.4122`
- `Test RMSE: 0.3843`

These RMSE values are the same as those obtained from the linear ridge regression model, since the predictive mean of the Bayesian model corresponds exactly to the ridge solution in this case. This confirms the theoretical equivalence between MAP estimation and Bayesian inference using Gaussian priors and likelihoods.

4. Report the average log-likelihood of the train and test data under your Bayesian model.

- `Average Train Log-LLH: -27568.08`
- `Average Test Log-LLH: -25265.41`

These values indicate the model's overall likelihood fit. While the absolute values are negative due to the log of probabilities, closer values between train and test indicate reasonable generalization.

5. Include the resulting plot and a short description.

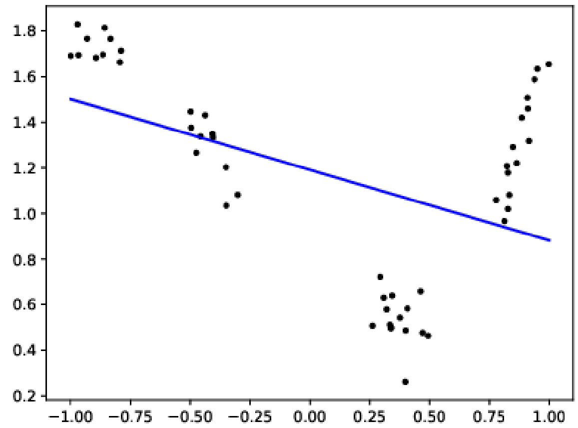

Figure 4: Bayesian linear regression fit with predictive mean and linear features.

The model fits a straight line with the predictive mean. Since the underlying function is nonlinear, the model underfits, but still captures a global trend.

6. Explain the differences between linear regression and Bayesian linear regression.

The key difference between these two regressions lies in how the model treats the weights:

- **Linear regression** finds a single point estimate for the weight vector w by minimizing the squared loss (possibly with regularization).
- **Bayesian linear regression** treats w as a random variable and infers a posterior distribution over it using Bayes' rule.

As a result:

- Linear regression produces a single prediction.
- Bayesian linear regression provides both a mean prediction and uncertainty (variance) around it.

Treating Bayesian as a probability allows models to quantify confidence in their predictions and to update beliefs as new data is observed.

1d) Squared Exponential Features (8 Points)

Implement Bayesian linear ridge regression using squared exponential (SE) features by filling in the ToDos of the corresponding task in the provided Colab Template. In other words, replace your observed data matrix $\mathbf{X} \in \mathbb{R}^{n \times 1}$ by a feature matrix $\mathbf{\Phi} \in \mathbb{R}^{n \times k}$, where

$$\mathbf{\Phi}_{ij} = \exp\left(-\frac{1}{2}\beta(\mathbf{X}_i - \alpha_j)^2\right).$$

Set $k = 20$, $\alpha_j = j * 0.1 - 1$ and $\beta = 10$. Use the ridge coefficient $\lambda = 0.01$ and assume known Gaussian noise with $\sigma = 0.1$. Include an additional input dimension to represent a bias term.

1. Report the RMSE of the train and test data under your Bayesian model with SE features.
2. Report the average log-likelihood of the train and test data under your Bayesian model with SE features.

3. Include the resulting plot and a short description.
4. How can SE features be interpreted from a statisticians point of view? What are α and β in that context?

Solution:

1. Report the RMSE of the train and test data under your Bayesian model with SE features.

- `Train RMSE: 0.0816`
- `Test RMSE: 0.1434`

The RMSE values indicate a very accurate fit to the training data, and a adequate generalization performance on the test data. The improvement over the linear and polynomial features suggests that the SE features capture the local structure of the function well.

2. Report the average log-likelihood of the train and test data under your Bayesian model with SE features.

- `Average Train Log-LLH: -244.12`
- `Average Test Log-LLH: -225.88`

These values indicate a high likelihood fit on both training and testing data, significantly better than the linear model. The SE features allow the model to express nonlinear patterns, leading to improved log-likelihood scores and better uncertainty estimates.

3. Include the resulting plot and a short description.

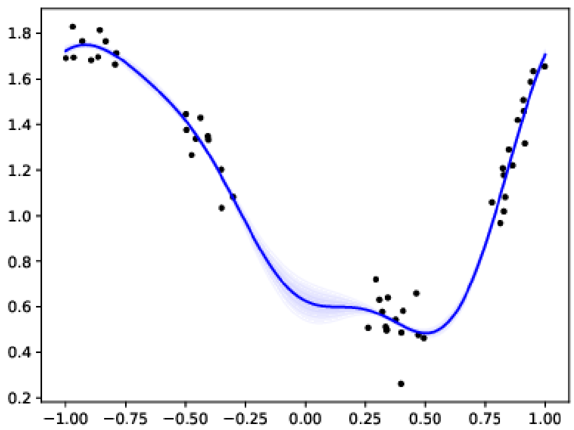

Figure 5: Bayesian regression with squared exponential features ($k = 20$, $\beta = 10$).

The blue curve represents the predictive mean, while the shaded area indicates the uncertainty. The fit is smooth and accurate across the entire input space, capturing both the global trend and local variations.

4. How can SE features be interpreted from a statisticians point of view? What are α and β in that context?

From a statistician's perspective, they serve as a form of kernel feature transformation, mapping scalar inputs into a higher-dimensional feature space where each dimension captures similarity to a fixed center.

Each feature is of the form:

$$\phi_j(x) = \exp\left(-\frac{1}{2}\beta(x-\alpha_j)^2\right)$$

Here:

- α_j defines the center of the j-th Gaussian basis function.
- β controls the width of the basis function; it is the inverse of the variance and determines how local or global the feature response is.

By combining multiple SE features, the model can approximate complex nonlinear functions while maintaining interpretability and smoothness.