Assignemnt 2: Probabilistic ML: Jupyter Report

Marcus Hansen

November 2024

1 Linnear Regression

In this part of the analysis, we aim to perform linear regression to fit a line through the points generated by the true underlying process. Figure 1 shows the true process that we are trying to estimate using our data points. We performed linear regression using maximum likelihood estimation (MLE) of the parameters, similar to what we discussed in the theoretical sections. The coefficients are calculated using the formula:

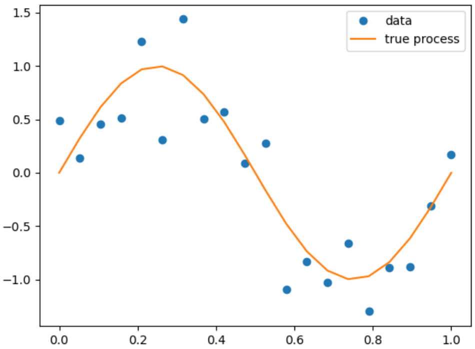

Figure 1: True process

$$\hat{\beta} = (X^T X)^{-1} X^T y \tag{1}$$

where X is the design matrix, with the first column filled with ones to account for the intercept of the line. The remaining columns are populated with the data for different powers of x. The inclusion of the column of ones allows the model to estimate the intercept. The data is stored in a DataFrame for easier access and manipulation.

The beta coefficients were computed in two ways: first, using the formula in equation 1, which yielded the coefficients $[\beta_0, \beta_1, \beta_2] = [0.88516511, -2.23262046, 0.55870444]$. The second method involved using the built-in linear regression function from the `sklearn` library. Both methods produced the same coefficients. The sklearn function automatically adds the column of ones to the design matrix, so we needed to specify `fit_intercept=False` to prevent it from adding another intercept term, as our design matrix already included it.

Next, we calculated not just a point estimate for the coefficients but also their variances. This is done to quantify the uncertainty of our parameter estimates. The variance of the estimated coefficients is computed using the formula:

$$\text{var}[\hat{\beta}] = \hat{\sigma}^2 (X^T X)^{-1}$$

We used this information to calculate the 95% confidence intervals for the coefficients. These intervals provide a range of values within which we are 95% confident the true parameters lie. The results are presented in the table 1.

Coefficient	**Estimate**	**SE**	**Lower 97.5% Bound**	**Upper 97.5% Bound**
Intercept	0.885165	0.379353	0.084801	1.685530
x	-2.232620	1.758354	-5.942423	1.477182
x^2	0.558704	1.697384	-3.022462	4.139871

Table 1: Model Coefficients, Standard Errors, and Confidence Intervals

As shown in Table 1, the confidence intervals for some coefficients are quite wide. This suggests that our data may not provide very precise estimates of the parameters, indicating a high level of uncertainty around the estimated values. For the intercept, the entire confidence interval is above zero, which means we are 95% confident that the intercept is positive. We also compared our results with Ordinary Least Squares (OLS) regression using the `statsmodels` library, and found that both methods produced the same results.

We can also plot our predicted second-order polynomial to see how well it approximates the true process. In Figure 2, it is clear that our estimated model is too simple to accurately capture the true process. Let's now examine what happens when we use a model of order 3.

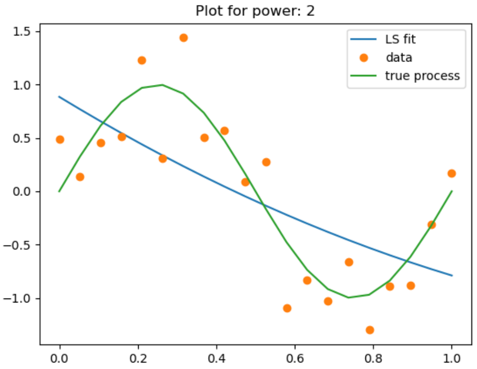

Figure 2: Estimated model: second-order polynomial

Next, we repeated the process with a third-order polynomial, and the results are shown in table 2.

Coefficient	Estimate	SE	Lower 97.5% Bound	Upper 97.5% Bound
Intercept	-0.010460	0.290987	-0.627324	0.606404
x	10.124663	2.586393	4.641755	15.607572
x^2	-31.139397	6.102501	-44.076122	-18.202672
x^3	21.132068	4.006810	12.638011	29.626125

Table 2: Model Coefficients, Standard Errors, and Confidence Intervals

In Table 2, we observe that the confidence intervals are still quite wide, indicating a high degree of uncertainty around the estimated coefficients. However, we are more confident about the signs of the coefficients. Specifically, for x, x^2, and x^3, we are 95% confident that their signs are correctly identified. As with the previous model, we compared the results with OLS regression from `statsmodels` and found that the results were identical.

Here, we can also plot our predicted third-order polynomial to assess how well it approximates the true process. In Figure 3, it is clear that our estimated model successfully captures the true process accurately.

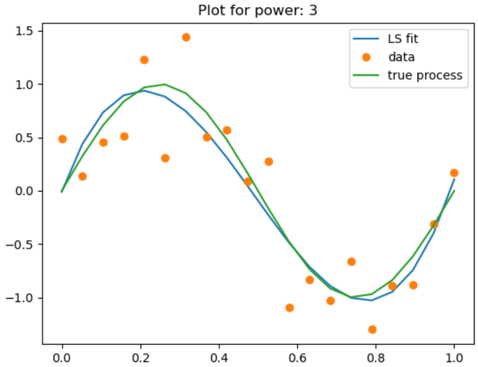

Figure 3: Estimated model: second-order polynomial

Finally we created a function:

```
def linear_regression(X: np.array, y: np.array, predictors: list) -> pd.DataFrame:
```

This function takes in the design matrix X, the vector y of observed outputs, and a list of predictors. It outputs a DataFrame containing the coefficients, standard errors, and the 95% confidence intervals.

2 Bayesian Linear Regression

In contrast to the previous section, we now implement a Bayesian linear regression model. This Bayesian linear regression model incorporates prior beliefs about the parameters: $\beta \sim N(\mu_0, \sigma^2\Omega_0)$ and $\sigma^2 \sim IGamma(a_0, b_0)$, where:

- μ_0: The prior mean of β.
- Ω_0: The prior covariance (scaled by σ^2).
- σ^2: Represents prior uncertainty in the variance.
- a_0: Reflects the prior belief about how concentrated σ^2 is around a specific value.
- b_0: Represents the magnitude of the prior belief about σ^2.

Using these priors, the posterior distribution of β combines the prior with the likelihood of the observed data:

$$\pi(\beta|X, y) \propto \mathcal{L}(y|X, \beta) \cdot \pi(\beta),$$

which provides an updated belief about β after observing the data.

The posterior parameters are calculated using the following update rules:

$$\mu_n = (X^T X + \Omega_0^{-1})^{-1}(\Omega_0^{-1}\mu_0 + X^T y),$$
$$\Omega_n = (X^T X + \Omega_0^{-1})^{-1}.$$

To assign the unit information prior, we set $\Omega_0 = n(X^T X)^{-1}$ or equivalently $\Omega_0^{-1} = \frac{X^T X}{n}$.

For σ^2, if we set the **prior** $IGamma(a_0, b_0)$, the **posterior** distribution is also $IGamma(a_n, b_n)$, where:

$$a_n = a_0 + \frac{n}{2},$$
$$b_n = b_0 + \frac{1}{2}\left(y^T y + \mu_0^T \Omega_0^{-1}\mu_0 - \mu_n^T \Omega_n^{-1}\mu_n\right).$$

Which resulted in the following results

Predictor	**Mean**	**Lower 97.5% CI**	**Upper 97.5%**
x_0	0.4504	-0.0395	0.9404
x_1	-0.6416	-1.6961	0.4129
x_2	-0.5618	-1.6149	0.4913

Table 3: Posterior Means and 95% Credible Intervals for Each Predictor using a t distribution

2.1 Monte Carlo Credible Intervals

To obtain credible intervals for β that apply to general models, such as logistic regression, we use the Monte Carlo method. This approach generates random samples from the posterior distribution and computes the desired intervals directly from these samples, instead of relying on analytical formulas.

The result of calculating the Monte Carlo credible intervals is:

Coefficient	**Posterior Mean (μ_β)**	**Lower 97.5%**	**Upper 97%**
β_1	0.4504	-0.0636	0.9508
β_2	-0.6447	-1.7096	0.4258
β_3	-0.5598	-1.6194	0.4808

Table 4: Posterior Mean and 95% Credible Intervals for β Based on Monte Carlo Sampling

When comparing the results from the Monte Carlo approach (Table **??**) and the analytical approach (Table 3), we observe that the Monte Carlo method provides an excellent estimate of the analytical solution for the posterior, which is based on a t-distribution in Table 3. Both the mean and credible intervals align closely, demonstrating that the Monte Carlo method is a robust estimator for these parameters.

2.2 Model Evidence

To calculate the model evidence, we use the following equation:

$$\pi(\beta, \sigma^2 | y, X) = \frac{\pi(y|\beta, \sigma^2, X)\pi(\beta, \sigma^2)}{\pi(y|X)},$$

where $\pi(y|X)$ represents the marginal likelihood of the data.

By testing different order polynomials, we find that the optimal n-th order polynomial is 3, which aligns with the order of the polynomial used to generate the data. This result is as expected.

When applying this method to the automobile insurance case, we test all different combinations of variables to identify the model that maximizes the model evidence. The best combination of variables is **('ATTORNEY', 'CLMAGE')**, with a model log evidence of -4454. While this value is relatively low, it should be interpreted in the context of the number of data points in the dataset.