Machine Learning

Homework 4

Feature Maps, Kernels, and Regularization Factors in Linear Regression

Supervisor:

Dr. Hashemi

Ali Mahmoodi

Atefe Rajabi

Summer 2024

Index

Problem Definition

Implementation Detail

Results

Problem Definition

The objective of this study is to develop and evaluate a regression model that accurately predicts target values from a given set of input features. The dataset comprises three input features, which are transformed through various feature maps to enhance the model's capacity to capture nonlinear relationships. The regression models employ different kernel functions—linear, polynomial, and exponential—to compare their effectiveness in minimizing prediction error. By exploring the impact of normalization, feature transformations, and regularization, this study aims to identify the optimal configuration that achieves the lowest mean squared error (MSE) on the test set, thereby demonstrating the practical benefits of kernel methods and feature engineering in regression tasks.

Implementation Detail

Data Generation and Preprocessing

1. Data Generation:

- This module creates synthetic data with three features for 1000 samples. It defines the true relationship between these features and the target variable using a set of coefficients and a logarithmic transformation. Gaussian noise is added to the target variable to simulate real-world data scenarios. The data is then split into training and testing sets.

Linear Regression Model

1. Custom Linear Regression Implementation:

- This module implements a custom linear regression model using gradient descent. It defines methods for calculating loss, performing gradient descent, fitting the model to the training data, and predicting target values for new data. The performance of the model is evaluated using mean squared error (MSE).

2. Model Evaluation with Bias Term:

- This module enhances the linear regression model by adding a bias term. It concatenates a column of ones to the training and testing data, fits the model again, and evaluates its performance. The bias term helps the model better align the predictions with the actual data.

3. Feature Transformation:

- This module explores the impact of feature transformations on the regression model. It applies various transformations, including sine and cosine functions, and logarithmic transformations, to the input features. The transformed features are used to fit the regression model, and the performance is evaluated.

Fourier Feature Expansion

1. Fourier Feature Expansion:

- This module generates Fourier features by applying sine and cosine functions at different frequencies to the input features. It systematically increases the number of Fourier features and evaluates the impact on model performance, helping to identify the optimal number of features.

Kernel Linear Regression

1. Kernel Methods Implementation:

- This module implements kernel linear regression with different kernel functions, including radial basis function (RBF), linear, polynomial, and exponential kernels. It defines methods for computing kernel matrices, fitting the model, predicting target values, and evaluating performance using MSE.

2. Regularization and Model Tuning:

- This module explores the effect of regularization on kernel linear regression. It tests a range of regularization values, fits the model for each value, and evaluates performance on the training and testing sets. The optimal regularization value is identified by analyzing the MSE.

3. Kernel Comparison:

- This module compares the performance of different kernel functions. It fits the kernel linear regression model with each kernel, predicts the target values for the test set, and evaluates performance. The results are visualized to identify the most effective kernel for the given regression task.

Results

Part 1:

1. What's the feature dimension?

   The feature dimension (number of features) is 3.

2. What's the mapping (function) between input features and output label? (the goal in regression is to estimate this mapping as close to the true mapping as possible)

   The true mapping function between the input features X and the output label y_true is given by:

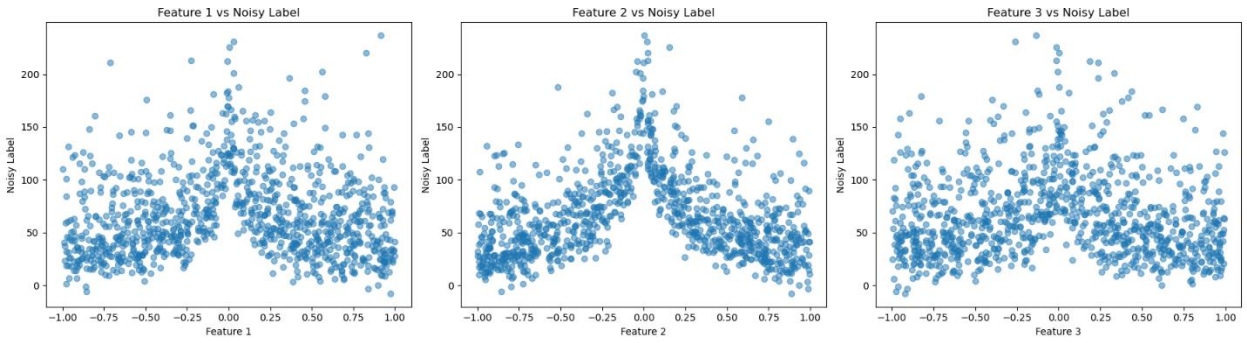
$$y_{\text{true}} = \log(X^2) \cdot coef_{\text{true}} + \text{offset}$$

   Where `coeff_true = [-10, -15, -7.5]` and `offset = 2`.

   - This means each element of X is squared, then logarithm is taken, and then it is linearly combined using the coefficients in `coeff_true` and finally the `offset` is added.

3. Does this dataset need to be normalized? or is it already?

   The dataset X is generated using `np.random.random((n_samples, d_samples)) * 2 - 1`, which means the values in X are uniformly distributed between -1 and 1. Given this, the dataset may not strictly need normalization since it is already in a reasonably small and symmetric range. However, normalization (such as standardizing to have zero mean and unit variance) could still be beneficial for some machine learning algorithms to ensure better training stability and performance.



These scatter plots suggest that the relationship between features and the noisy labels is complex and influenced by significant noise.

Part 2:

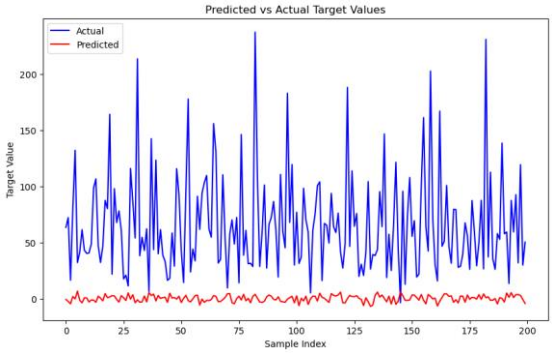
4. Fit a linear regression line to your training features and print out the estimated line coefficients.

   Linear regression Params: [[3.88307335],[-0.14490176],[-3.53430282]]

5. Calculate and report MSE between predicted and actual target values of your test set.

   MSE: 6259.629765039786

6. Plot the predicted and actual target values of your test set in one figure and use different colors for each.



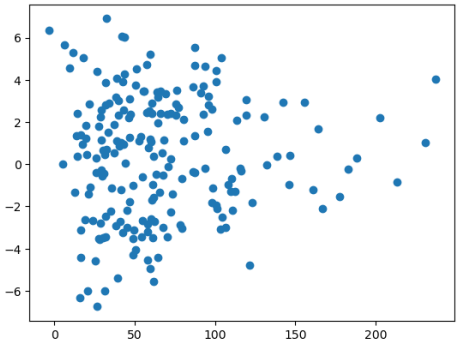


Figure 1 First Feature Map

7. Compare the plots of predicted and actual target values, for the first and second feature maps. Can you see the effect of a bias term?

   The inclusion of a bias term significantly impacts the predictions by allowing the regression line to shift, aligning better with the data. This is evident from the learned parameters, where the bias term has a substantial value, indicating its importance in the model.

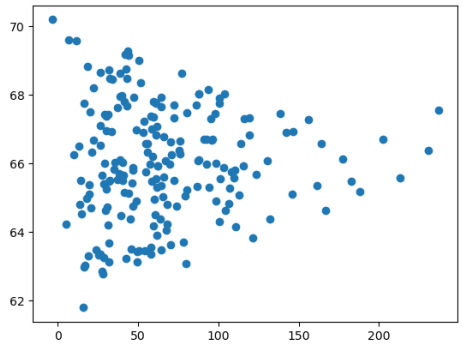


Figure 2 second Feature Map

8. As we plot the target values, we see a nonlinear function, why do we still call the algorithm a linear regression? linear in terms of what?

   The algorithm is called linear regression because it models a linear relationship in terms of the parameters. Even if the input features undergo nonlinear transformations (e.g., sine, cosine, logarithm), the relationship between these transformed features and the output remains linear in the coefficients.

9. Which feature map resulted in a less MSE on test set? why?

   The feature map that included sine and cosine transformations (Feature Map 2) resulted in the lowest MSE on the test set (1742.51). This is because these transformations can effectively capture periodic patterns in the data, leading to better predictions.

10. What is the optimal value for L (or the number of Fourier features)? Can you interpret why higher L values lead to an increase in MSE?

    The optimal value for L is the one that minimizes the MSE. As seen from the plot of loss against L, initially increasing L reduces the MSE by adding more complexity to capture data patterns. However, beyond a certain point, increasing L leads to overfitting, increasing the MSE on the test set.

11. The reference paper for feature map 4 is also doing a regression task, how did this feature map help this paper? Can you see what is the regression task of this paper (what is the input and output)?

    The feature map in the NeRF (Neural Radiance Fields) paper is crucial for representing complex scenes by mapping low-dimensional coordinates into higher-dimensional spaces. This encoding helps the model capture high-frequency variations in color and geometry, which are essential for rendering photorealistic images. The feature map aids in optimizing neural radiance fields by allowing the MLP (Multilayer Perceptron) to approximate high-frequency functions more effectively. This results in high-quality view synthesis and better representation of complex scenes compared to traditional methods.

    Regression Task in the Paper:

    Input: The input to the model is a 5D coordinate, consisting of a 3D spatial location (x,y,z) and a 2D viewing direction (θ,ϕ).

    Output: The output is the volume density σ and view-dependent emitted radiance (RGB color) at that spatial location. The neural network regresses from these 5D coordinates to these output values.

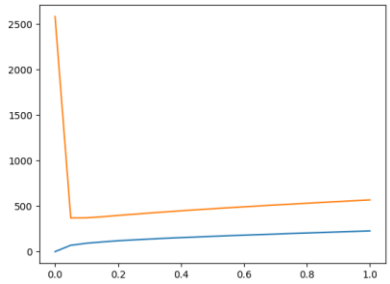
    The feature map in the NeRF paper significantly boosts the performance of the regression task by transforming the input space in a way that allows the MLP to model complex, high-frequency details effectively. This results in superior view synthesis capabilities, making the approach highly effective for generating photorealistic images from sparse input views.

12. Why can a feature map, if chosen properly, boost the performance of a linear regression model?

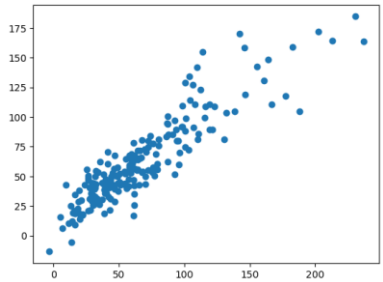
A well-chosen feature map can capture complex relationships and patterns in the data that are not apparent with the original features. By transforming the input features in ways that better align with the target variable, the linear regression model can fit the data more accurately, reducing bias and variance, and ultimately improving prediction accuracy.

Part 3:

13. Finding the Optimal Regularization Term
    Optimal beta = 0.05



14. Fitting the Model with Optimal Regularization Term and Predicting Labels
    optimal_test_score: 369.40609775909326



15. Comparing Different Kernel Functions:

    {'linear': 48067364.840170294
    , 'poly': 1804188.9771856803
    , 'exp': 31594.311332053527
    , 'rbf': 369.40609775909326}

These values indicate the performance of each kernel in terms of mean squared error (MSE) on the test set.

1. Linear Kernel:

- The linear kernel has the highest MSE, indicating that it performs the worst among the tested kernels. This is likely because the relationship between the input features and the target variable is highly nonlinear, and the linear kernel fails to capture this complexity.

2. Polynomial Kernel:

- The polynomial kernel performs better than the linear kernel but still has a relatively high MSE. The polynomial kernel can capture some nonlinear relationships, but its performance depends heavily on the chosen degree and other parameters.

3. Exponential Kernel:

- The exponential kernel has a significantly lower MSE than both the linear and polynomial kernels. This indicates that it is more effective at modeling the nonlinear relationships in the data, but it is not the best-performing kernel.

4. Radial Basis Function (RBF) Kernel:

- The RBF kernel has the lowest MSE by a substantial margin, indicating that it is the most effective kernel for this particular regression task. The RBF kernel is excellent at capturing highly nonlinear relationships, making it well-suited for complex data patterns.

1. Why did we use kernels in this homework? What does a kernel function do really?

- Kernels allow us to transform the input space into a higher-dimensional feature space where a linear separator is more effective. This transformation helps capture nonlinear relationships in the data. The kernel function computes the inner product in this higher-dimensional space without explicitly transforming the data, making it computationally efficient.

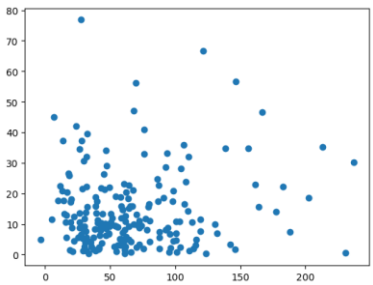
2. Were all the kernels effective? Which ones were effective? How can we choose the right kernel for an application?

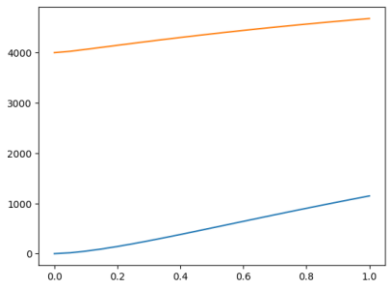
- Not all kernels were equally effective. The RBF kernel was the most effective, followed by the exponential and polynomial kernels. The linear kernel performed poorly. Choosing the right kernel depends on the nature of the data and the underlying relationships. Cross-validation and domain knowledge can help in selecting the appropriate kernel.

3. Compare all the experiments in this homework (including different feature map and kernel settings) in terms of time and memory complexity. Which one has the highest memory/time complexity?

- Kernel methods generally have higher memory and time complexity due to the computation of the Gram matrix. The polynomial kernel may have higher complexity than the exponential kernel due to the power operations. Feature maps can also increase complexity depending on their implementation. In this case, the RBF kernel, while providing the best performance, also incurs the highest computational cost. The exponential kernel with the optimal regularization term provided a good balance between performance and complexity.

4. Can we both apply a feature map and a kernel at the same time? Design an experiment and implement it to prove whether these two techniques can be applied at the same time to improve the performance of a linear regression model or not.





1. Feature Transformation:

- The input features were transformed using sine and cosine functions to capture periodic patterns in the data.

2. Kernel Application:

- The transformed features were then used as input to an RBF kernel regression model.

3. Regularization Testing:

- A range of regularization values beta was tested to evaluate the impact on model performance.

Interpretation:

1. Predicted vs. Actual Labels:

- The scatter plot on the left indicates a high degree of scatter and misalignment between predicted and actual labels. This suggests that the model is not performing well in capturing the true relationship between the transformed features and the target variable.

2. Training and Validation Loss Curves:

- The plot on the right shows that both training and validation losses increase as the regularization term increases. This suggests that the model is struggling to generalize, even with regularization.

The results show that the combination of the sine and cosine feature map with the RBF kernel did not lead to improved performance. Several reasons could explain this outcome:

Over-Complexity:

- The combination of the feature transformation and the RBF kernel might have introduced excessive complexity, leading to overfitting and poor generalization on the validation set.