Demystifying Linear Regression: Part II

Rishabh Iyer

March 19, 2024

1 Introduction

In the first installment of our exploration into Linear Regression (Part I), we delved into the foundational aspects of this pivotal statistical method, covering its practical applications, the underlying model, the formulation of the loss function, optimization techniques, and evaluation metrics. Building upon that groundwork, this second blog article aims to expand our understanding by exploring several significant extensions to the basic linear regression framework. We will venture into the realms of (i) enhanced feature selection and model complexity, notably through polynomial regression, (ii) a broader spectrum of loss functions to better capture the nuances of various data distributions, and (iii) the introduction of regularization techniques to improve model generalization and prevent overfitting. Join us as we navigate these advanced topics, further enriching our toolkit for tackling real-world predictive modeling challenges with linear regression.

While Linear Regression is a good start, there are several important extensions:

- *Polynomial Regression* to go beyond linear functions particularly when the relationship between the target variable y is not a linear function of the features x.
- *Go beyond the Square Loss* and where does it make sense to use other loss functions?
- *Regularization in Linear Regression* what is regularization, why is it useful, and what are the different kinds of regularization?

2 Polynomial Regression

A critical limitation of linear regression is that the model is a linear function. This means that it can only fit functions if the target variable y is a linear function of the features. However, this can easily fixed via Polynomial Regression.

Polynomial regression is a form of linear regression where the relationship between the independent variable x and the dependent variable y is modeled as an

nth degree polynomial. Despite modeling non-linear relationships, polynomial regression is considered a linear model because it is linear in the coefficients w. This approach significantly enhances the model's flexibility to capture complex relationships by introducing polynomial features.

$$h_{w,b}(x) = w_0 + w_1 x + w_2 x^2 + \ldots + w_n x^n$$

1. **Polynomial Feature Transformation:** Polynomial regression is essentially linear regression applied to polynomial features of the input. The non-linear relationship between x and y is captured by transforming the input x into polynomial terms $(x, x^2, x^3, \ldots, x^n)$, which are then used as inputs in a linear model.

2. **Degree of the Polynomial:** The degree n of the polynomial plays a crucial role in the model's complexity and its ability to fit the data. A higher degree polynomial can fit the training data more closely, capturing more intricate patterns. However, with a high degree n, the model becomes more susceptible to overfitting, where it performs well on the training data but poorly on unseen data due to its excessive complexity.

Balancing Underfitting and Overfitting: Choosing the right degree of the polynomial is critical. A low degree might underfit the data, failing to capture the underlying trend, while a high degree might overfit, capturing noise in the training data as if it were a real pattern. This trade-off necessitates careful selection and validation techniques, such as cross-validation, to determine the optimal degree for the polynomial regression model. Figure 1 illustrates the importance of tuning the degree of the polynomial regression. Degree 1 is underfit, Degree 3 is a good fit, and Degree 9 is overfit to the data.

3 A Richer Class of Loss Functions

The choice of the loss function in linear regression significantly influences the model's training and performance. A generalized form of the loss function can be expressed as a power n loss, where the loss for a single prediction is defined as:

$$L_n(w, b) = \frac{1}{M} \sum_{i=1}^{M} |h_w(x^{(i)}) - y^{(i)}|^n$$

Here, M represents the number of training examples, $h_w(x^{(i)})$ is the model's predicted value for the i-th example, $y^{(i)}$ is the actual value, and n is a positive integer that shapes the loss function.

Special Cases: Two interesting special cases of $L_n(w, b)$ are:

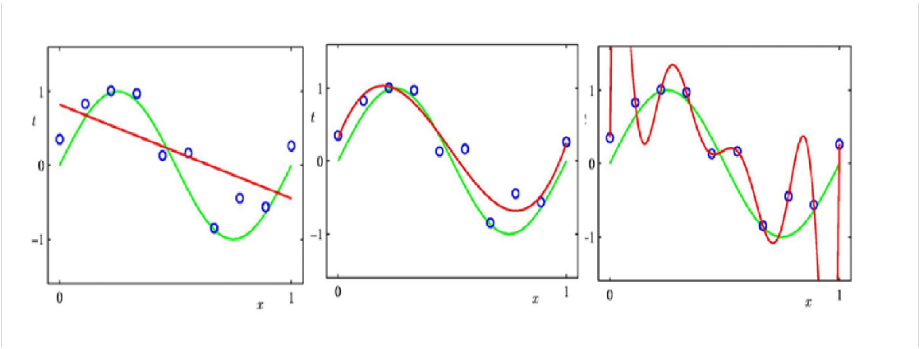

Figure 1: Illustrating the behavior of polynomial regression for different degrees. The green curve is the actual generating function (the blue points are generated based on this) and the model is the red curve. The left figure shows a linear model fit which as we can see, is underfit. The middle figure is a degree 3 polynomial which is a good fit and the right figure is a degree 9 polynomial which has clearly overfit.

- When $n = 2$, the loss function is the Mean Squared Error (MSE), commonly used in linear regression:

$$L_2(w, b) = \frac{1}{M} \sum_{i=1}^{M} (h_w(x^{(i)}) - y^{(i)})^2$$

- When $n = 1$, the loss function becomes the Mean Absolute Error (MAE), known for its robustness to outliers:

$$L_1(w, b) = \frac{1}{M} \sum_{i=1}^{M} |h_w(x^{(i)}) - y^{(i)}|$$

Behavior as n Increases: Increasing n makes the loss function more sensitive to larger errors, leading to heavier penalties for predictions that significantly deviate from actual values:

- **Sensitivity to Outliers:** Higher n values cause outliers to exert a disproportionately large influence on the loss, potentially skewing the model to fit these anomalies at the expense of general accuracy.
- **Error Penalization:** As n grows, the penalty on larger errors increases more than on smaller ones. This can help minimize large prediction errors but may also make the model less robust to data noise and outliers.

Trade-offs with Loss Functions:

- MSE ($n = 2$) is favored for its mathematical properties, such as differentiability, which facilitates optimization. However, its sensitivity to outliers can be a drawback in some scenarios.
- MAE ($n = 1$) offers greater robustness against outliers, making it suitable for datasets with significant noise. The optimization of MAE can be more challenging due to its piecewise linear nature.
- Higher Powers ($n > 2$) focus the model on reducing large errors but risk overemphasizing outliers, potentially leading to overfitting.

Selecting the appropriate loss function requires balancing these considerations, taking into account the dataset's specific characteristics and the goals of the modeling effort.

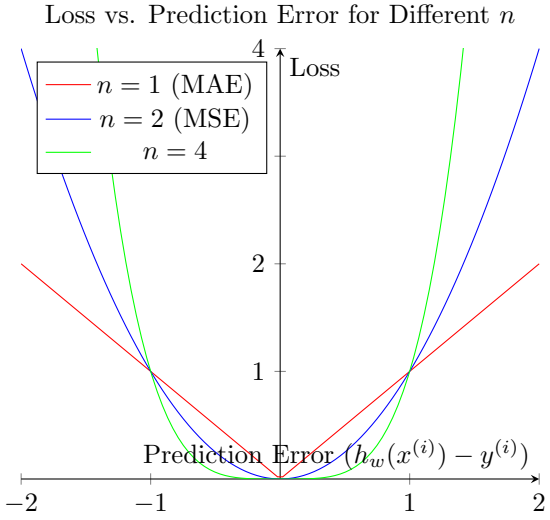

Figure 2: Illustration of how the loss increases with prediction error for different powers of n. This figure demonstrates the sensitivity of the loss function to outliers and the penalization of errors: $n = 1$ shows linear growth (robust to outliers), $n = 2$ shows quadratic growth (sensitive to larger errors), and $n = 4$ emphasizes the penalization of larger errors even more strongly.

4 Importance of Regularization: Ridge and Lasso Regression

In the context of linear regression, overfitting occurs when the model learns the noise in the training data to the extent that it performs poorly on unseen data. Regularization is a technique used to prevent this overfitting by adding

a penalty on the size of the coefficients, which helps to keep them small and reduces the model's complexity.

- **Ridge Regression (L2 Regularization):** Ridge Regression, also known as L2 regularization, adds a penalty equal to the square of the magnitude of coefficients to the loss function. The regularized loss function is:

$$L(w, b) = MSE + \lambda \sum_{j=1}^{n} w_j^2$$

  where λ is the regularization parameter, a hyperparameter that controls the strength of the penalty. The key feature of Ridge regression is that it tends to shrink the coefficients evenly but does not necessarily bring them exactly to zero. This means that all features are retained in the model but their influence is moderated.

- **Lasso Regression (L1 Regularization):** Lasso Regression, or L1 regularization, adds a penalty equal to the absolute value of the magnitude of coefficients. This is represented as:

$$L(w, b) = MSE + \lambda \sum_{j=1}^{n} |w_j|$$

  Unlike Ridge, Lasso can reduce some coefficients to zero, effectively performing feature selection. This property makes Lasso particularly useful when we believe many features are irrelevant or if we desire a sparse model.

4.0.1 Why Regularization is Useful

Regularization is a technique used in machine learning to prevent models from becoming too complex and closely fitting the training data, a problem commonly known as overfitting. Without regularization, models might capture the noise along with the underlying pattern in the training data, making them perform poorly on new, unseen data. By applying regularization, we gently guide the model towards simpler explanations for the observed data, which, although might make it slightly less perfect on the training data, significantly improves its performance on new data by ensuring it captures the true underlying patterns rather than the noise.

4.0.2 Differences Between Ridge and Lasso Regression

- **Coefficient Shrinkage:** Ridge regression shrinks the coefficients towards zero but does not set any to zero. Lasso regression can shrink some coefficients to zero, providing a method of feature selection.

- **Feature Selection:** Lasso regression inherently performs feature selection by eliminating some features entirely, which can be beneficial in models with a large number of features, many of which might be irrelevant.

- **Model Interpretability:** Because Lasso can eliminate some features, it can lead to more interpretable models, especially when the number of observations is much larger than the number of features.

Choosing between Ridge and Lasso regression depends on the specific dataset and the importance of feature selection. Ridge is generally preferred when the relationship between features and the target is approximately linear, and all features are expected to be relevant, and we aim to reduce the overfitting of the model. Lasso is useful when it is suspected that only a subset of features are relevant, or when the goal is to reduce the complexity of the model by eliminating some features.

5 Summarizing Linear Regression

The process of building and evaluating a linear regression model involves several key steps, from data collection to model evaluation. Here is a summary of these steps:

1. **Collect the Dataset:** Gather a dataset that includes the input features (x) and the target variable (y). The dataset should be representative of the problem you are trying to solve.
2. **Decide the Model:** Choose the type of model based on the relationship between the input features and the target variable. Options include:
   - Linear model: $h_{w,b}(x) = w^T x + b$
   - Polynomial model: $h_{w,b}(x) = w_0 + w_1 x + w_2 x^2 + \ldots + w_n x^n$
3. **Select the Loss Function:** Choose a loss function that will measure the difference between the predicted values and the actual values. Common choices are:
   - Mean Absolute Error (MAE) Loss: $L(w,b) = \frac{1}{M} \sum_{i=1}^{M} |h_{w,b}(x^{(i)}) - y^{(i)}|$
   - Mean Squared Error (MSE) Loss: $L(w,b) = \frac{1}{2M} \sum_{i=1}^{M} (h_{w,b}(x^{(i)}) - y^{(i)})^2$
   - Higher Power Loss: $L_n(w,b) = \frac{1}{M} \sum_{i=1}^{M} |h_{w,b}(x^{(i)}) - y^{(i)}|^n$
4. **Regularization:** Decide if you would like to regularize the linear regression or not (either L1 regularization or Ridge Regression or L2 Regularization or Lasso Regression). Also, decide the λ hyper-parameter for regularization.
5. **Optimize Using Gradient Descent:** Apply gradient descent (or another optimization algorithm) to minimize the chosen loss function and (if applicable) regularization. This step adjusts the model parameters (w and b) to find the best fit to the training data.

6. **Training Procedure Output:** The output of the training procedure is the optimized model characterized by the parameters (w, b) that define the relationship between the input features and the target variable.

7. **Predict on Test Dataset:** Use the trained model to predict the target variable for a new, unseen test dataset. This involves applying the model $h_{w,b}(x)$ to each input feature set in the test data to obtain the predicted labels.

8. **Evaluate the Model:** Assess the model's performance on the test dataset using evaluation metrics such as RMSE, R^2, or MAE. This step provides insight into how well the model generalizes to new data.

This procedure outlines the comprehensive steps involved in building a linear regression model, from initial data collection to final evaluation, ensuring a systematic approach to model development and assessment.

6 Conclusion

Linear regression is a fundamental technique in machine learning, offering a simple yet powerful tool for predicting continuous variables. Understanding its underlying principles, from hypothesis functions and loss functions to optimization algorithms and evaluation metrics, is essential for effectively applying this method in practice.