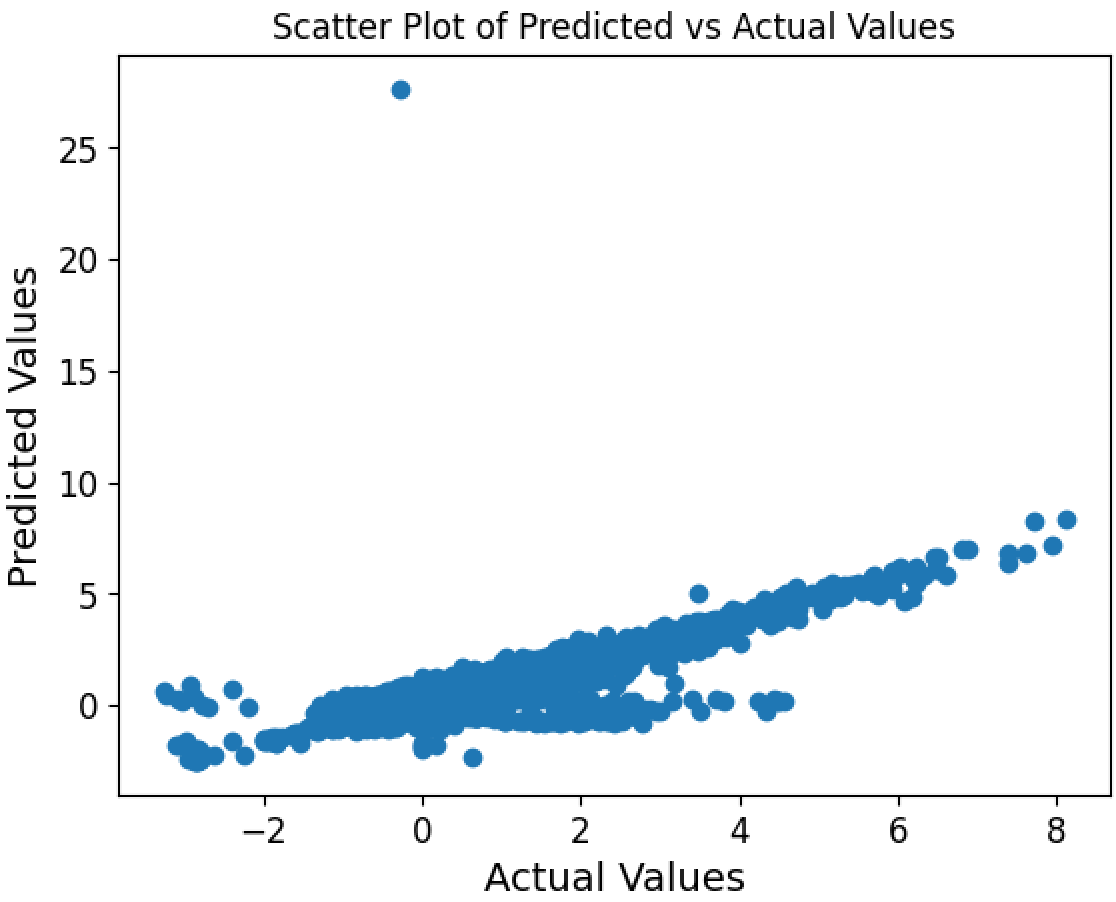
Predicting the Interest Rate Spread in New York State

Executive Report



Lekso Borashvili,Monique Brown, Neloy Kundu

CS 414: Machine Learning

Professor Lopez Bencosme

Problem Definition & Objectives

Current models used by banks to assign credit risk are archaic and not public knowledge. This project aims to create a model of interest rate determination. This project would like to use the personal details of a person and non-conventional attributes to predict the type of interest rate to give. This project could be useful in managing risk as a lender to a borrower. Further the transparency of such a model gives us insight into what goes into interest rates.

Mortgage interests are selected based on a variety of factors of that individual. These factors include:

1. Credit score
2. Home Location
3. Home price and loan amount
4. Down payment
5. Loan term
6. Interest rate type
7. Loan type

This project aims to predict the interest rate spread using data from New York State for 2021. The interest rate spread is the difference between the interest rate that is offered and the national interest rate. The data set used was from Home Mortgage Disclosure Act 2021.

1. Our primary goal is to predict the interest rate spread of a borrower. That is, what is the difference between how much interest is being charged for a person based on race, the loan amount, the value of the property, the national interest rate, and other factors.

Methodology

The models we used were for a regression machine learning problem, specifically using Linear Regression for our winning solution. Linear Regression is a supervised learning technique that involves learning the relationship between the features and the target. The target values are continuous, which means that the values can take any values between an interval.

The problem was modeled using a multivariate linear regression and it was trained using a cost function MSE and evaluated using R^2 . Using MSE as a performance measure of a regression model means to find the value of coefficients on variables that minimize the MSE when training the model. Since loss is the penalty for a bad prediction, if the model's prediction is perfect, the loss is zero; otherwise, the loss is greater. The goal of training a model is to find a set of weights and biases that have low loss, on average, across all examples.

The model was implemented using a polynomial of degree two because we wanted a better performance. This would help reduce some inherent bias. This was one way to make the model more complex. A higher degree would overfit the model. This allows the model to see if there are any non- linear relationships between the target variables.

We also trained the model using a Neural Network with two (2) and three (3) layers of ten (10) dense nodes for the regression. We wanted to see if we could get a better result than a simple raw linear regression. Artificial Neural Networks for regression over Linear can learn complex non-linear relationships whereas linear regression can only learn the linear relationship between the features and target, unless paired with other tools such as polynomial transformation of variables. Artificial Neural Networks have the ability to learn the complex relationship between the features and target inherently, especially when using multiple layers.

We also used Ridge Regression to regularize our model. Ridge Regression (also called Tikhonov regularization) is a regularized version of Linear Regression: a regularization term equal to $\alpha \sum_{1=1}^{n} \theta_{i}^{2}$ i is added to the cost function. This forces the learning algorithm to not only fit the data but also keep the model weights as small as possible. Note that the regularization term should only be added to the cost function during training. Once the model is trained, you want to use the unregularized performance measure to evaluate the model's performance.

The model was tested using a train-test split. This is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model. We used an 80/20 split instead of something like a k-fold cross-validation procedure because we have sufficient data.

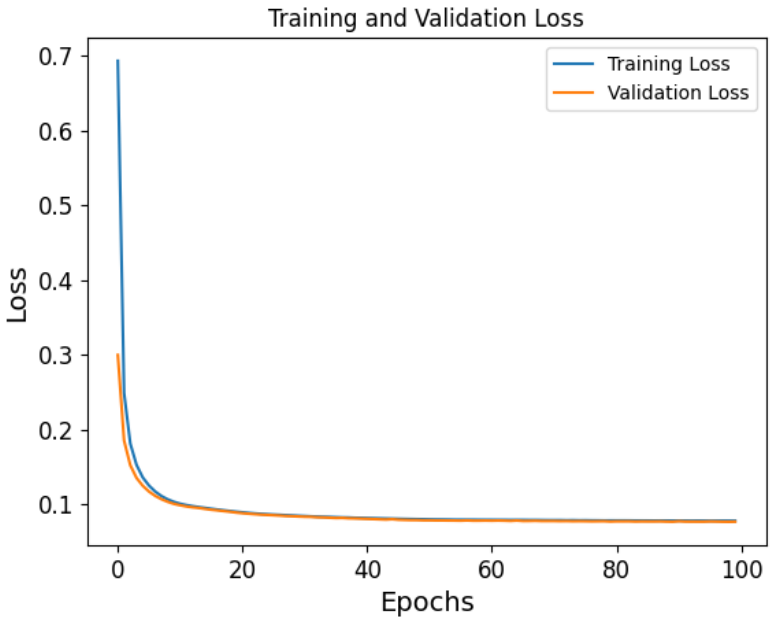
Results

Our initial R-squared is 0.797.

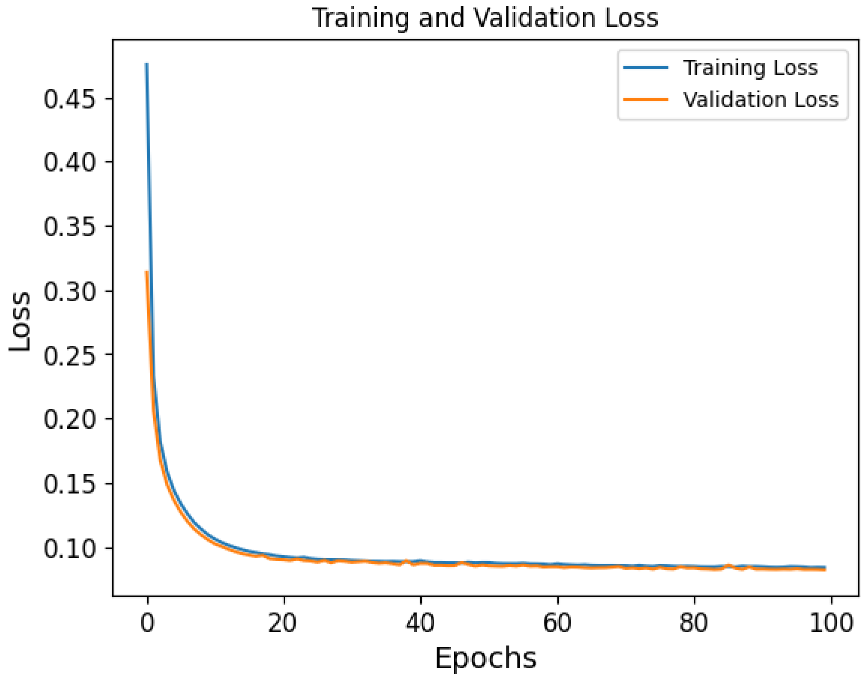
The p-values are all very high. They are significant to the 90% level of significance.

Using sklearn to train a simple linear regression. We had the Mean Squared Error: 509822199.6010236 and R-squared: -1039236392.6255503.

Using a Neural Network with two layers gave us a Mean Squared Error: 0.07680150121450424.



Using a Neural Network with three layers gave us Loss & Mean Squared Error: [0.08194486796855927, 0.07385314255952835].



After regularizing our model the Mean Squared Error was 0.09601065422985178 and the R-squared became 0.8040034880048675.

The Mean Squared Error of our final linear regression model is 0.08630384434852473. Our final R-squared is 0.8238190063408363. This is better than our initial 0.797 model raw linear regression, so we are happy with the results, especially for a model that has not been run before. This was the Linear regression model with the best results from GridSearch in the above cell. Linear Regression and standard edition to the model prove a better performing model overall.