Predicting Bitcoin Price Through ML

Ruairi Delahunt

Contents of this template

You can delete this slide when you're done editing the presentation

Fonts	To view this template in PowerPoint, download and install the fonts we used
Used and alternative resources	An assortment of graphic resources that are suitable for use in this presentation
Thanks slide	You must keep it so that proper credits for our design are given
Colors	All the colors used in this presentation
Icons and infographic resources	These can be used in the template, and their size and color can be edited
Editable presentation theme	You can edit the master slides easily. For more info, click **here**

For more info:
SLIDESGO | **BLOG** | **FAQs**

You can visit our sister projects:
FREEPIK | **FLATICON** | **STORYSET** | **WEPIK** | **VIDEVO**

Table of contents

01

Introduction

Background on the dataset

Bitcoin

The most well known cryptocurrency and most profitable, Bitcoin has become more popular in recent years due to high price peaks. The dataset I used to create my model uses data from 2010 (around the beginning of crypto) until 2024

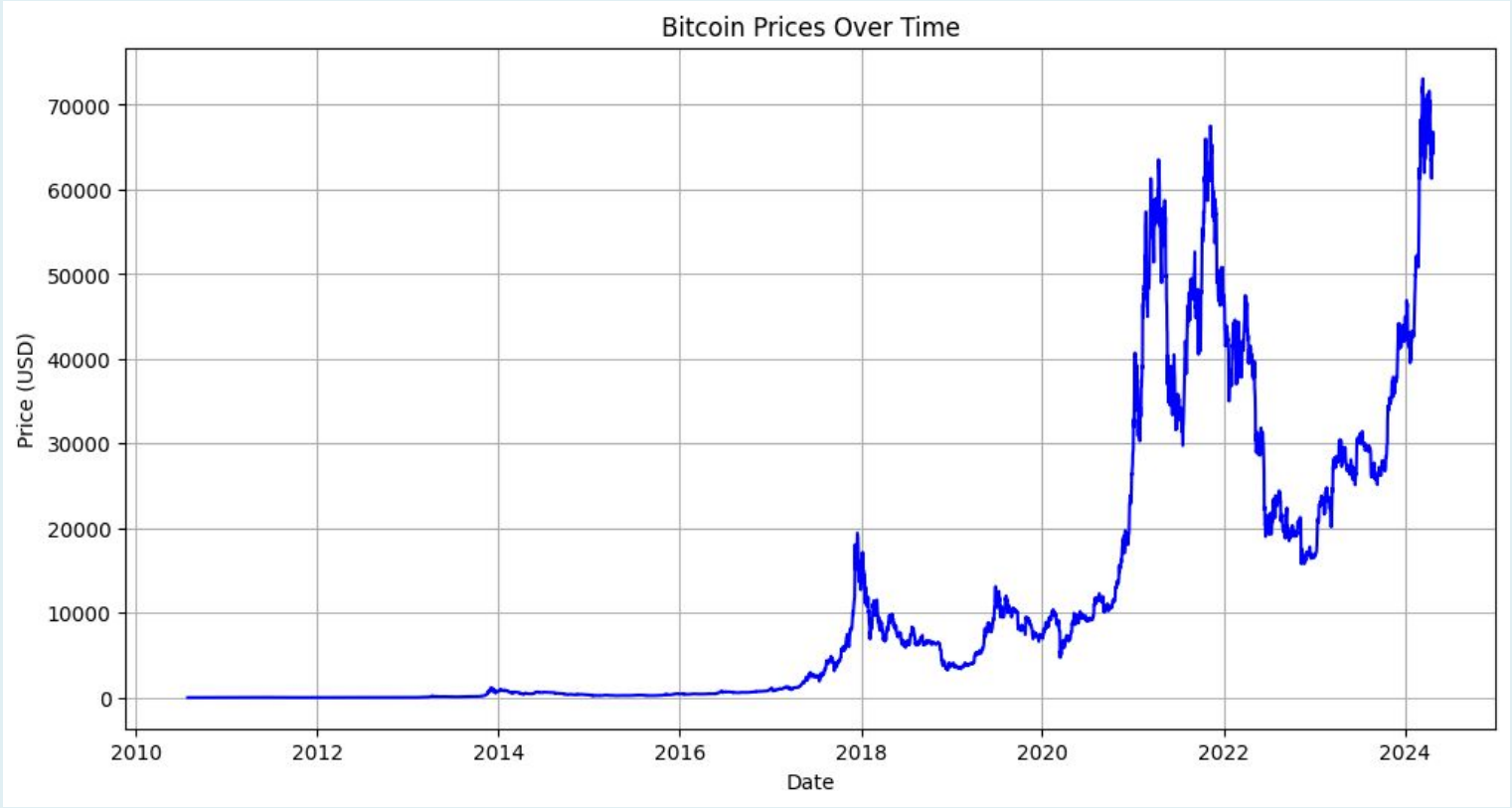
Dataset

Bitcoin Prices 2010 - 2024

https://www.kaggle.com/datasets/priyamchoksi/bitcoin-historical-prices-and-activity-2010-2024

Dataset Summary:
Total entries 5021
Number of attributes: 8

Data Types
Numerical: 6
Categorical: 2


Bitcoin Prices Over Time
Price (USD)
70000
60000
50000
40000
30000
20000
10000
0
2010
2012
2014
2016
2018
2020
2022
2024
Date

The Goal

Create the most accurate model of predicting Bitcoin's future price

Regression Model Problem:

Linear Regression
Polynomial Regression
ElasticNet

02

Data Analysis

Work done on the dataset

Preparing the Data

Cleaning

I identified the non-numeric columns as well as missing values and removed them from the dataset

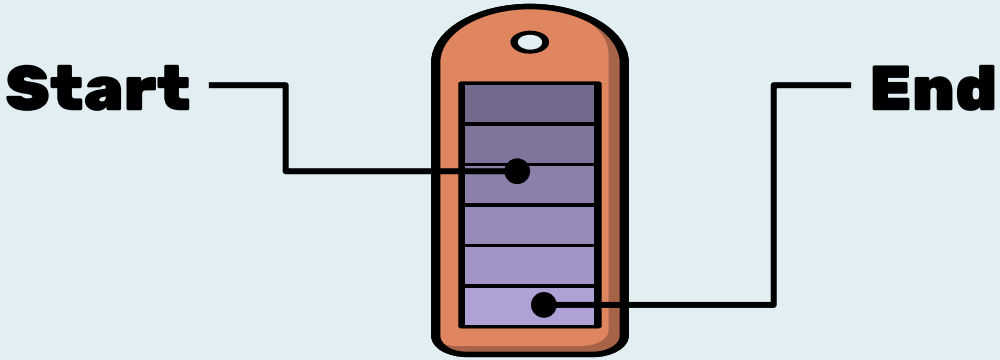
Sorting

The data was sorted based on the ‘End’ dates to make sure the data is organized chronologically

Converting

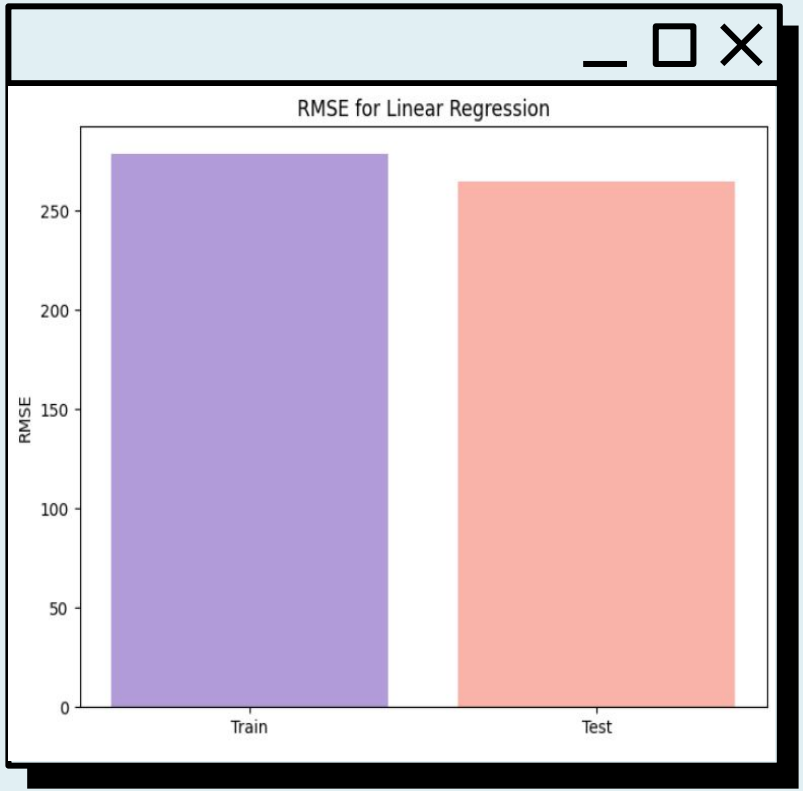
I converted the date values to datetime format using a specific function

Other Attributes



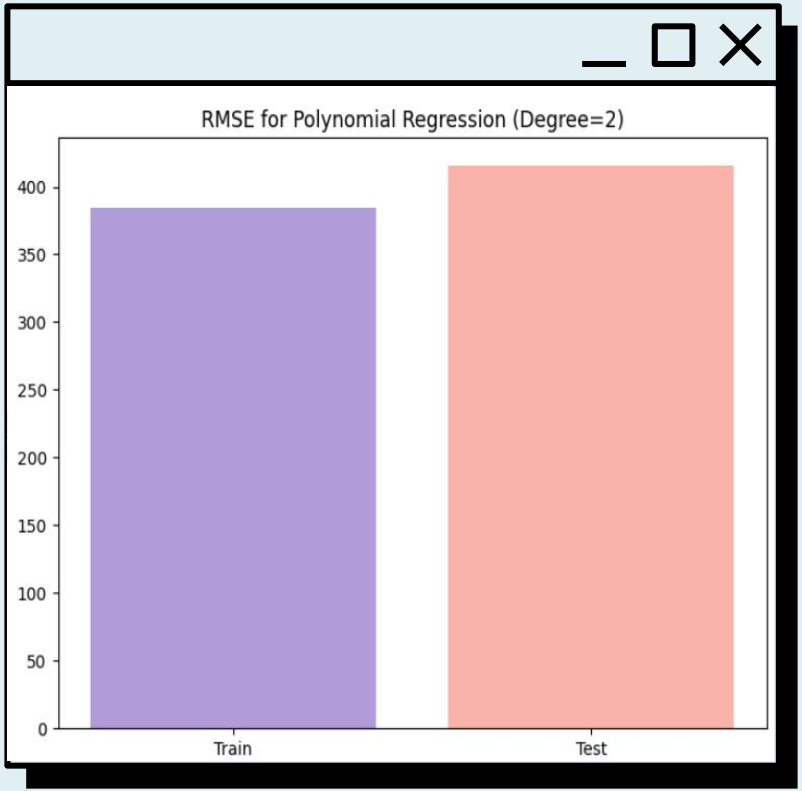
Linear Regression

Train RMSE: 278.53125674371387
Test RMSE: 264.8591679825205



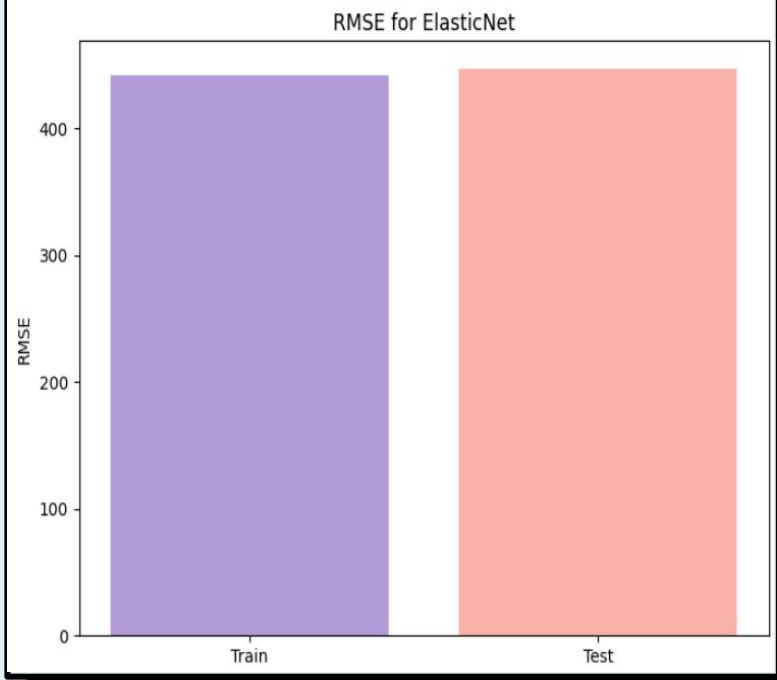
Polynomial Regression

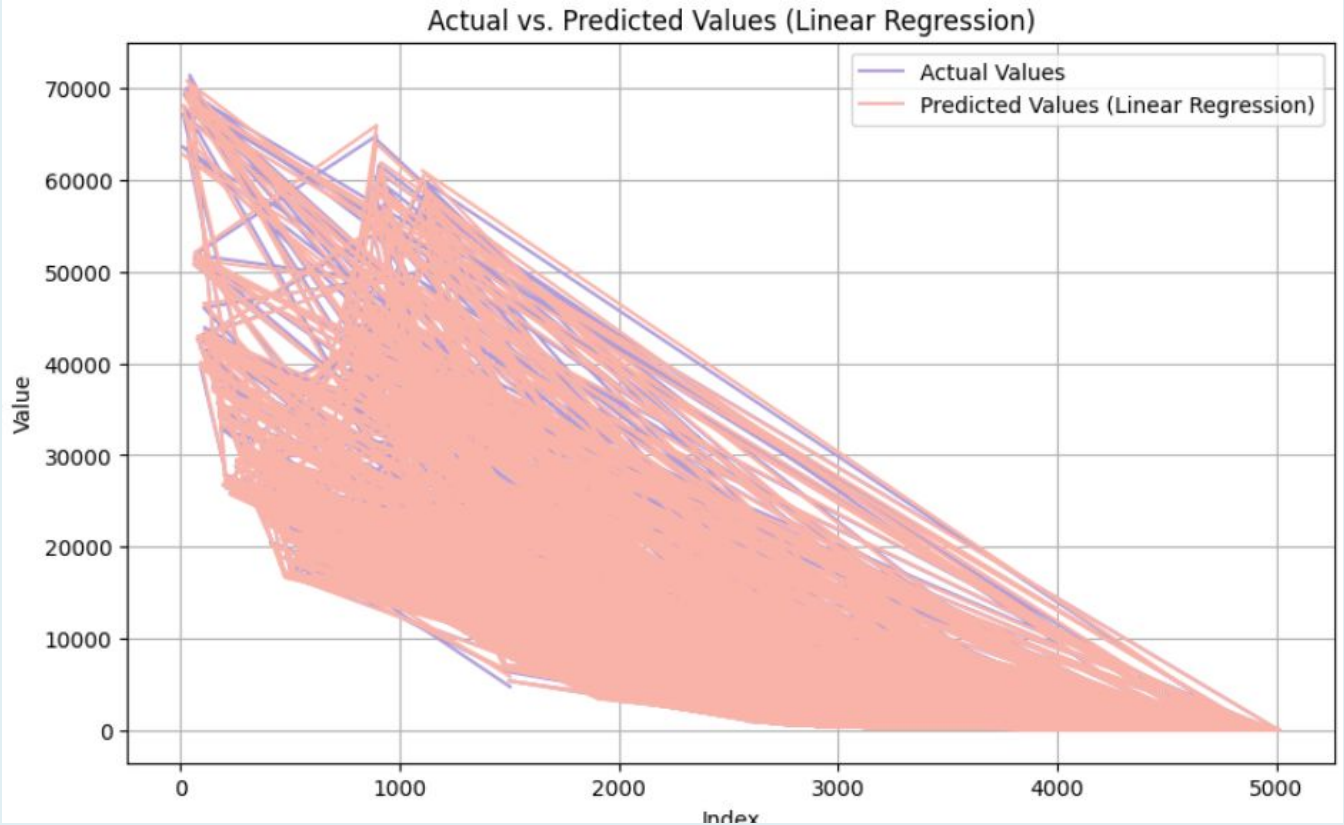
Train RMSE: 384.08758850204214
Test RMSE: 415.32191154327455

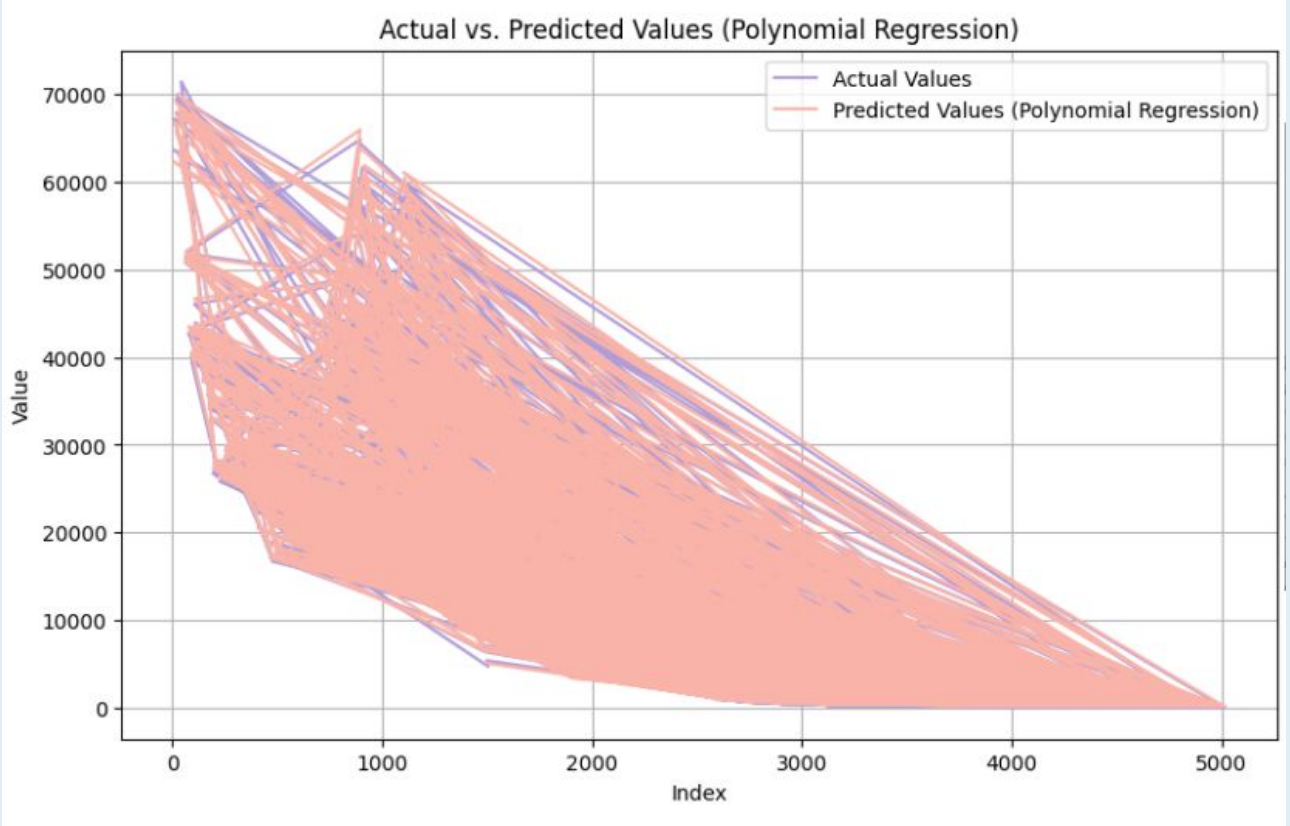


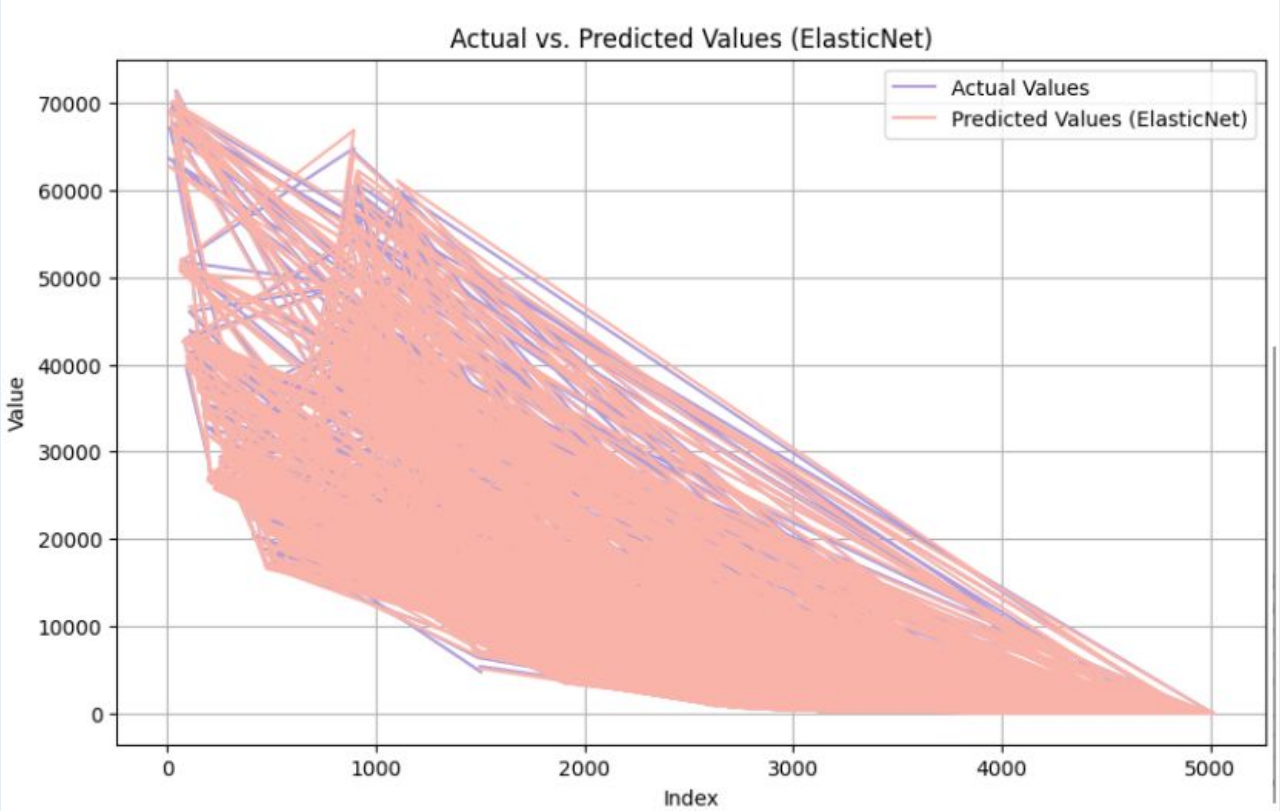
ElasticNet Regression

Train RMSE: 441.365814562673
Test RMSE: 446.4429922434832




Actual vs. Predicted Values (Linear Regression)
Actual Values
Predicted Values (Linear Regression)
Value
Index
70000
60000
50000
40000
30000
20000
10000
0
0
1000
2000
3000
4000
5000

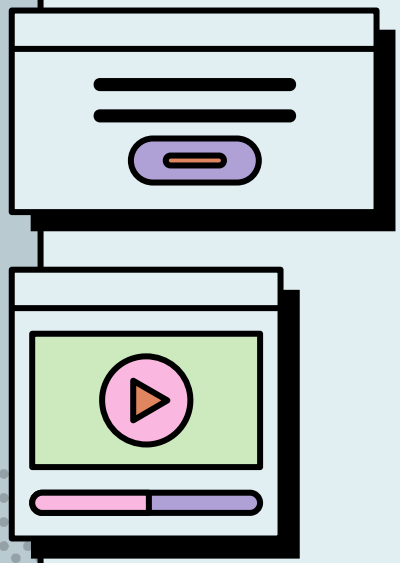

Actual vs. Predicted Values (Polynomial Regression)
Actual Values
Predicted Values (Polynomial Regression)
Value
Index
70000
60000
50000
40000
30000
20000
10000
0
0
1000
2000
3000
4000
5000


Actual vs. Predicted Values (ElasticNet)
Actual Values
Predicted Values (ElasticNet)
Value
Index
70000
60000
50000
40000
30000
20000
10000
0
0
1000
2000
3000
4000
5000

03

Takeaways

Which Model Worked Best



The Linear regression model turned out to be the most accurate model as the RMSE values were:

278.53 Train, 264.86 Test

The Polynomial Regression Model was the second best with an RMSE of:

384.09 Train, 415.32 Test

ElasticNet ended with the worst performing results with RMSE values of:

441.37 Train, 446.44 Test

21º

Thank You

Ruairi Delahunt