## Abstract and Dataset

## Our aim was to forecast the value of properties in specific zip codes. To accomplish this, we gathered data from Zillow.com ("Data - Zillow Research," n.d.), which offered various real estate metrics at different levels, such as state, metro, city, zip code, and neighborhood. Additionally, we obtained a dataset of monthly crime rates in the Los Angeles area ("Crime Data from 2010 to Present | Los Angeles - Open Data Portal," n.d.) to investigate potential associations and improve the accuracy of our predictions.

## Transformation of Data

### Due to the large size of both datasets, extensive transformations were necessary to make them applicable to our project. Our main objective was to reduce the feature set size and ensure compatibility between the datasets. All the code for data cleaning and transformation can be found in the data collection directory of the repository.

### Zillow Data

The Zillow dataset contained several files for different types of regions, such as cities, counties, and zip codes. Each file included information about the region and metric values for a specific month in columns to the right. To make the data usable for our project, we needed to restructure it vertically by listing each value as a specific instance. To achieve this, we scanned through all the files and grouped the data by region, which allowed for smaller tests with less noise and some initial analysis. However, some metrics did not consistently have data for each zip code due to Zillow's ongoing creation of new metrics for measuring real estate performance. Therefore, we selected Median Value Per Square Foot as our label metric since it was consistently present throughout the dataset and was a reliable, normalized measure of real estate value. All this data processing and restructuring can be found in the data\_collection directory of the repository.

Crime Data

The L.A. crime dataset was extensive and detailed, with each line in the CSV file containing information about a crime, including the official crime code, description, victim age, victim sex, time, geographical coordinates, and other data. As our project focused on zip codes, we needed a way to transform the latitude/longitude coordinates into a specific zip code. Initially, we used the "geopy" Python module, which leverages various services such as Google, Bing, and ArcGIS to offer an API for reverse-geocoding coordinates into an address, from which we could extract the zip code. However, due to the large number of instances (1.6 million) to be transformed, this approach proved inefficient as the services throttled or denied our requests after 1000. To overcome this, we found a dataset of central coordinates for every zip code in the United States (erichurst, n.d.) that allowed us to determine the closest zip code coordinates to the crime's coordinates without relying on third-party services. While we acknowledge that this method is far from perfect as zip codes often have complex borders, it was sufficient for our purposes as we aimed to estimate crimes and real estate value within a smaller geographical area. We gathered the total number of crimes within each zip code, as well as the number of crimes for each crime code, victim age, victim descent, and victim sex. By combining this data with the real estate data, we had 340 pieces of information for each zip code for each month in the date range, spanning from 2008 to 2017. All the code and data processing can be found in the data\_collection directory of the repository.

Categorical Feature Encoding

## At the start of the project, we aimed to apply the classifier to a combined dataset of all zip codes. However, we needed to encode the zip code as a categorical feature since it was not a continuous variable. Initially, we attempted to use one-hot encoding, but we quickly realized that the large number of zip codes in our dataset made this approach unrealistic. As a solution, we turned to feature hashing, which would enable us to generate a hash with a fixed length and a unique identifier for each zip code.

## Unfortunately, the high level of noise resulting from the large number of zip codes in the dataset prompted us to abandon this approach. Instead, we decided to build a separate classifier for each individual zip code.

## Feature Selection

## Due to the large number of features in our dataset, we needed a systematic approach to identify the best features or combination of features for each model. To accomplish this, we trained a model on each feature separately and evaluated its performance using 10-fold cross-validation with root mean squared error as the metric. We added the best feature to the final feature list and repeated the process with the current set of features. We continued this process until there was no significant improvement in performance.

## It was important to ensure that we did not include any features that incorporated the label we were trying to predict, such as Zillow Home Value Index (ZHVI), which calculates home values. Although we were unsure of the exact formula used to calculate ZHVI, the presence of "home value" in the feature name and its strong performance on the dataset raised suspicions. Therefore, we chose not to include any ZHVI features in our model.

## Another challenge we faced was missing data for some features in certain zip codes and months. To ensure that our results were not skewed by a small sample size, we implemented a function that excluded any rows with missing data for the selected features. Additionally, we set constraints to ensure that the data contained at least 10 different zip codes and 20 instances for each zip code. Any data that did not meet these criteria was not considered in our search for the best features. These feature selection results will be discussed further in the evaluation section.

## Implementation of Models

For each model, we followed the scikit-learn standards of having **predict**, **fit** and **set\_params** functions.

The **fit** function took in a two-dimensional array of the features entitled ‘x’ and the one dimensional array of labels entitled ‘y.’ Both inputs were transformed into numpy arrays that allowed us to do quick and easy math to find the weights.

The functions used for fitting the weight vector are defined as follows:

|  |  |
| --- | --- |
| Linear Regression: |  |
| Ridge Regression |  |

For all models, the **predict** function took in a list of instances entitled ‘x’, converted them to a numpy array and simply returned a dot product of x and the weight vector. We also implemented a baseline classifier that simply returned the mean of the training labels.

## Evaluation of Models

To evaluate each model, we first ran through the process of selecting the most valuable features, as discussed above. Here were the features we found for each model:

|  |  |
| --- | --- |
| Linear | Ridge |
| MonthId  MedianListingPricePerSqft\_4Bedroom  MedianListingPrice\_3Bedroom | MonthId  MedianListingPricePerSqft\_4Bedroom  MedianListingPrice\_3Bedroom  PriceToRentRatio\_AllHomes  ZriPerSqft\_AllHomes |

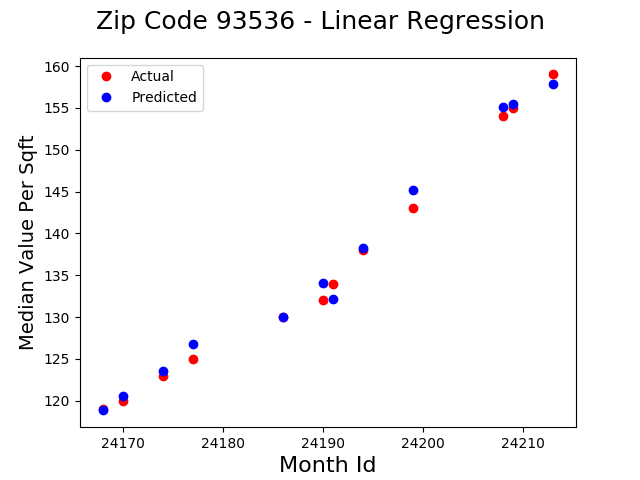
We weren’t at all surprised to see that Ridge did better with a higher dimensionality on this dataset, but we were a little saddened to see that none of the features from our crime dataset made the cut on either model.

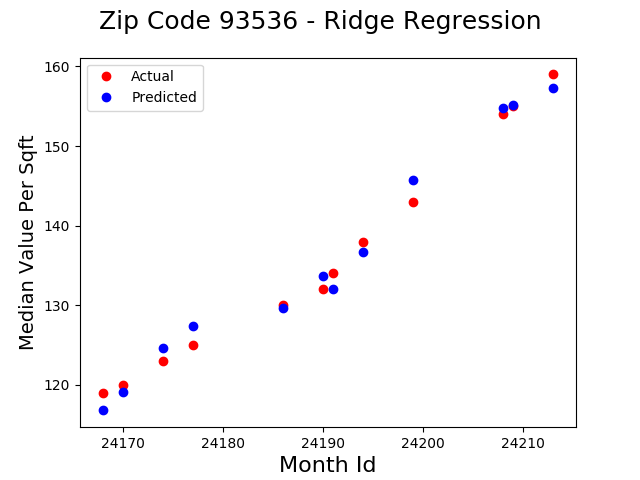
After selecting the features to split the data, using 75% as training data and 25% as test data. We ran this on each zip code finding the mean and standard deviation of the root mean squared error on the test and training sets for each model. Here were our results:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Test Mean | Test Std Dev. | Train Mean | Train Std Dev. |
| Linear | 8.518 | 6.178 | 8.241 | 6.089 |
| Ridge | 5.719 | 3.300 | 4.932 | 2.980 |
| Baseline | 22.887 | 12.432 | 20.847 | 11.441 |

As shown, Ridge regression performed consistently better throughout the dataset, scoring better on both the training and test sets, as well as having a smaller standard deviation of predictions for both. We were also happy to see that each performed significantly better than the baseline classifier.

For each evaluation on the test dataset, we generated a graph that shows the comparison of the actual value with the predicted value. Those graphs can be viewed in the ["graphs/" GitHub repository directory](https://github.com/tbarlow12/Real-Estate-Prediction/tree/master/graphs). We have included one example that displays the results generated from both Linear and Ridge regression, which happen to be very similar in this zip code.





## 

## References

Title: Crime Data from 2010 to Present | Los Angeles

Source: Open Data Portal

URL: <https://data.lacity.org/A-Safe-City/Crime-Data-from-2010-to-Present/y8tr-7khq>

Title: Zillow Research Data

Source: Zillow Research

URL: <https://www.zillow.com/research/data/>

Title: All US zip codes with their corresponding latitude and longitude coordinates

Source: erichurst

URL: <https://gist.github.com/erichurst/7882666>

Title: Feature hashing

Source: Wikipedia

URL:<https://en.wikipedia.org/wiki/Feature_hashing#Feature_vectorization_using_the_hashing_trick>