Connor White, David Chalifoux

0

- With no change to the data, the first response is: average price difference for training values: 10.2%
- After adding bathrooms and: average price difference for training values: 10.2%
- This is interesting because it does not help the performance, it actually stays the exact same. This is probably due to the fact that square feet are correlated with bathrooms and bedrooms, which makes sense because the more bedrooms and bathrooms you have the larger the square feet are.

1

After removing the North Bend restriction, the average price difference is 33.1%.

2

- Features: `sqft_living, bedrooms, bathrooms`
  - Result: average price difference for training values: 33.1%
- Features: `sqft_living, bedrooms, bathrooms, sqrft_lot`
  - Result: average price difference for training values: 33.0%
- Features: `sqft_living, bedrooms, bathrooms, sqft_lot, yr_built`
  - Result: average price difference for training values: 31.5%
- Features: `sqft_living, bedrooms, bathrooms, yr_built, waterfront`
  - Result: average price difference for training values: 31.4%
- Features: `sqft_living, bedrooms, bathrooms, sqft_lot, yr_built, waterfront`
  - Result: average price difference for training values: 31.3%
- Features: `sqft_living, sqft_lot, bathrooms, bedrooms, floors, condition, yr_built, yr_renovated`
  - Result: average price difference for training values: 31.1%
- Features: `view, floors, bathrooms, sqft_living, bedrooms, yr_built`
  - Result: average price difference for training values: 30.8%
- According to the features that I chose on my last test it seems like they have the best "predicted training percentage" with 6 features of view, floors, bathrooms, sqft_living, bedrooms, and year built.

3

Testing and training separated using built-in functions to Pandas:

```
housingDF = pd.read_csv('housing.csv')
trainingDF = housingDF.sample(frac=.3).reset_index()
testingDF = housingDF.drop(trainingDF.index).reset_index()
```

4

Using only the 30% training data, the average price difference for training values is 29.0%.

5

- When comparing against the testing dataset, the model performs 0-2% worse on average. For example, `30.4%` when testing against the training dataset vs. `32.6%` when testing against the testing dataset. In my opinion, this is means that the training data predicts the performance of the regression model fairly accurately.
- I am, however, surprised that it performs worse than using a single feature by a little over 20%. My assumption would be that the more features there are, the more accurate the model would be.