Syed Araib Karim 11479107
Hamza Tahir 10966566
Lydia Karumanch 11466410
Phongsiri Nirachornkul 11125038

Austin, TX House Listings

Abstract:

The purpose of this project is to determine which features of a house have a significant influence on its prices that are located in the Austin region. The data being analyzed has more than 15,000 sale transactions each consisting of 45 unique features of the house being sold. Houses in Austin were positively influenced by its square footage, location to city center, average distance to school and year it was built

Introduction:

In this project, we tried to create a model to determine which house features and locations would impact on house values. In addition, this paper was seeking to report any other additional insight gathered from data analysis with regards to housing prices. There were not many studies found specifically with regards to how house prices in Austin are influenced with the features and its location. This paper seeks to identify these variables and its results will be compared to literature to determine if any insights gathered from this paper is in line with trends identified in other markets.

Background:

Austin is the faster growing major metropolitan city in the United States with the population increasing by approximately 34% since 2010. Over the past 10 years, Austin has gained over half a million residents with 184 new residents averaging each day. Most of these inhabitants have relocated from Silicon Valley following the investment of major technology companies in the Austin region. New residents appreciate lower taxes, cost of living, business regulations which has made it very attractive for technology companies and other digital nomads. As a result of these high profile investments, Austin residential real estate remains extraordinarily demanding by those looking for a new house in this city. Most recent report for Austin Board of realtors have stated that median listed prices have been increasing by 29% year over year and average days for a house to be on the market has decreased by 13 days with the average being 22. Median house price in Austin is $470,000 according to Austin Board of Realtors (Egan, 2022). While home construction is booming in the Austin area, it is not fast enough to keep up with the region's high demand. Inventory is extremely low and continues to fall. The months of supply are just 0.4 months, which would indicate

that at the current pace of sales that are occurring it would take only 2 weeks or less for the supply to come down to zero (Santarelli, 2022).

Experiment Methodology:

Here are the five core tasks in our ML workflow:

1. Get Data.
2. Clean, Prepare & Manipulate Data.
3. Feature selection and Feature engineering.
4. Train a baseline model.
5. Model selection and hyperparameter tuning.

1. Get Data:

The dataset contains information regarding housing prices in Austin, Texas. The information was gathered through a capstone project at Northwestern University. The Austin housing market is expected to be one of the hottest in 2022, and these listings indicate how it has transformed in recent years.

The dataset provides detailed description of the house for example number of bedrooms, bathrooms, has cooling, how far it is from middle school, zip code and other features about the property. The dataset also includes images and descriptions from each home listing on Zillow.

2. Clean Prepare and Manipulate Data:

These are the following steps we took to clean the dataset, find any trends, ,relationships between features.

- Select the target
- Check if the target variable is normalized
- Checking for Data types, null values, missing values and unique values
- Checking for linear relationships between the target and other different numeric features.
- Checking for outliers in different features.
- Check if there is any data that can be considered as categorical data

3. Feature selection and Feature engineering

- See correlation between different features and the target to see feature importance.
- Generate new features based on the existing feature set.

4. Train a baseline model

- Select an evaluation metric. We selected RMSE.
- Train a baseline Linear Regression model.
- Do a permutation of the baseline model to generate feature importance.

5. Model selection and hyperparameter tuning:

- Use different machine learning models such as SVM, Decision Tree, XGBoost, KNN and others.
- Run grid search on all the models to generate the best hyperparameters for the dataset.

Results

Descriptive Summary

To get a preliminary level understanding of the data being analyzed, its value distribution and descriptive statistics were determined. There were 15171 houses listed with a mean price of 5.13 x 10^5. The standard deviation was reported to be 4.532 x 10^5. The minimum price was listed to be 5.50 x 10^3. 25% of the houses were priced lower than 3.09 x 10^3. 50% of the houses were priced lower than 4.05 x 10^5 and 75% of the houses were priced lower than 5.7x10^5. The maximum value was listed to be as 1.35 x 10^7.

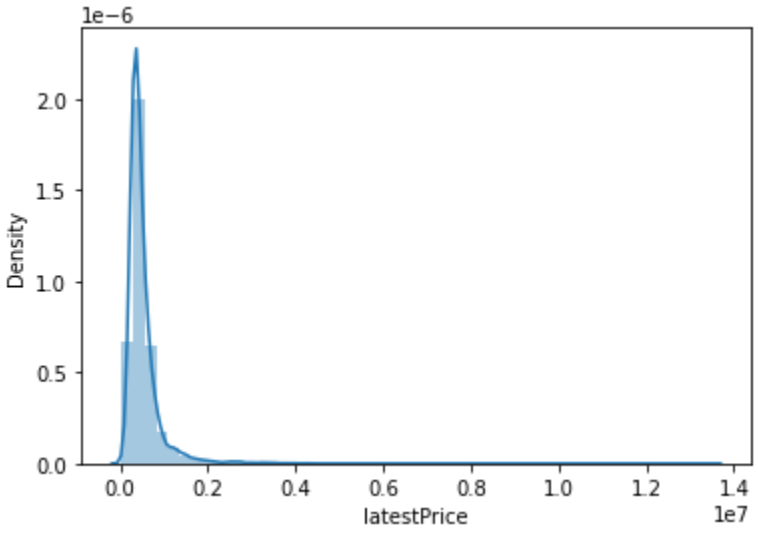


Figure 1: Standard distribution of data was carried out to yield a skewness value of 8.845 and a kurtosis value of 165.31.

According to figure 1,the data distribution is highly skewed in terms of the latest price of the houses reported since the skew value is greater than 1. One can observe that the distribution is slightly skewed to the right. Due to this high degree of skew, the kurtosis value was also quite significant. The kurtosis is greater than 3. This visualization suggests that the distribution is leptokurtic in nature in which it has a long tail as it is heavily skewed to the right. This pattern is caused by a high outlier with a maximum price of 1.35 x 10^7 when the mean price is 5.13 x 10^5.

Syed Araib Karim 11479107
Hamza Tahir 10966566
Lydia Karumanch 11466410
Phongsiri Nirachornkul 11125038

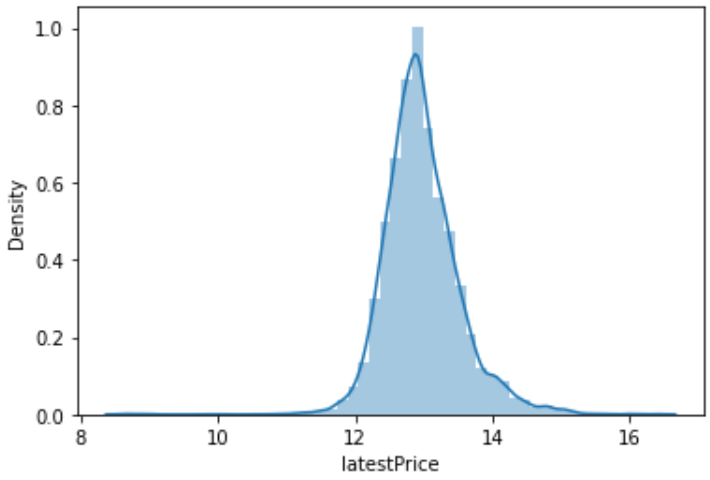


Figure 2: Logarithmic transformation of data distribution

As seen in figure 2, a logarithmic transformation was carried out to account for the skewed data set. One can see that the distribution appears closer to normal.

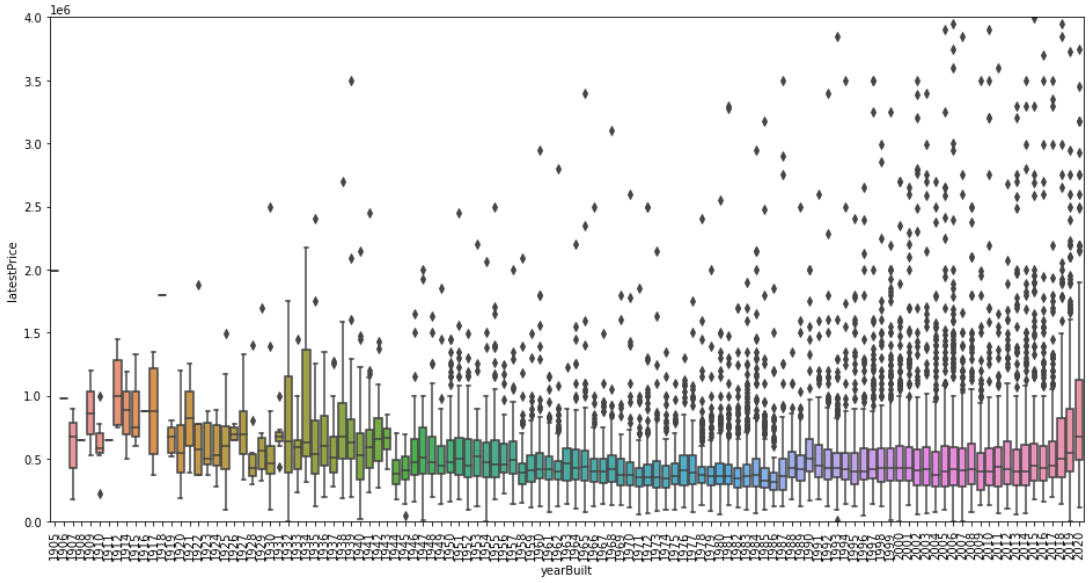


Figure 3: Price of house plotted against year the house was built

According to figure 3, houses built in more recent years were slightly priced higher. This would make sense because newer houses would use better material for construction and are less demanding in terms of maintenance. This would increase the overall value

of the house which would have an impact on the house equity. Older houses would require more maintenance over the years and have features that may be out of date in today's market. As such, they are more likely to hold less value which would negatively impact house equity. Unlike newer houses, older ones typically would have higher utility costs as less improved materials may have been used for insulation to retain heat or air. This would cause older houses to have to pay approximately 17% more on electricity and 38% more on gas per year (Ziraldo, 2022).

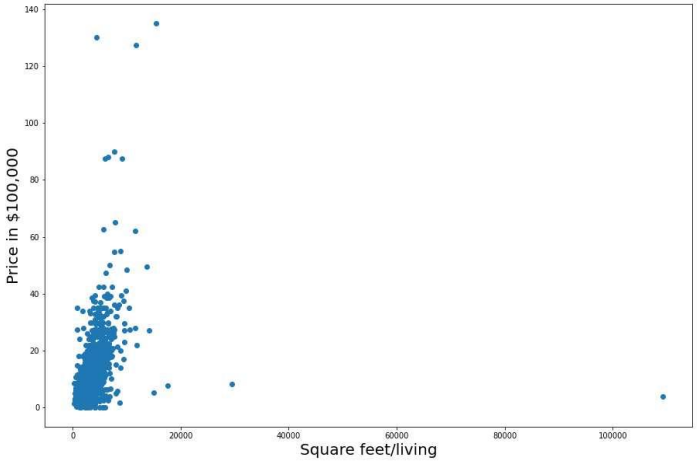


Figure 4: Price of the house plotted against square footage

According to figure 4, higher square footage of the house has a positive impact on price. Square footage is described as the amount of livable space in a house. A house with a higher square footage would demand more material needed in construction for its inhabitants. Higher square footage would require a large foundation, frame, exterior and interior materials needed to build the house. Increases in any of these variables during the house construction phase is often caused by a higher square footage. More square footage could also add to the complexity to house design which also would have a positive impact on its price. In addition, higher labor costs would be associated with constructing a larger house which can include services from roofers and electricians. Furthermore, larger houses would take more time to build which would be proportional to the amount of money needed to finance the construction (Thierault, 2022).

Syed Araib Karim 11479107
Hamza Tahir 10966566
Lydia Karumanch 11466410
Phongsiri Nirachornkul 11125038

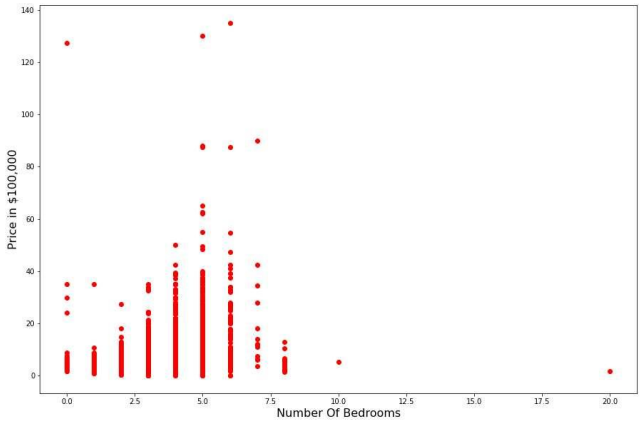


Figure 5: Price of the house plotted against number of bedrooms

According to figure 5, one can see that a higher number of bedrooms in a house slightly corresponds to a higher price. However, this is not always the case as there is part of the distribution that has a higher number of bedrooms than other houses but is priced lower. These houses likely have other features or locations that are considered less demanding in the market and are thus appraised lower. Generally speaking more bedrooms would accommodate more inhabitants and this would have a positive impact on price. The higher number of bedrooms would be associated with a larger square footage of the house which would demand more material in construction which in turn would increase the value of the house (Bartsch, 2021).

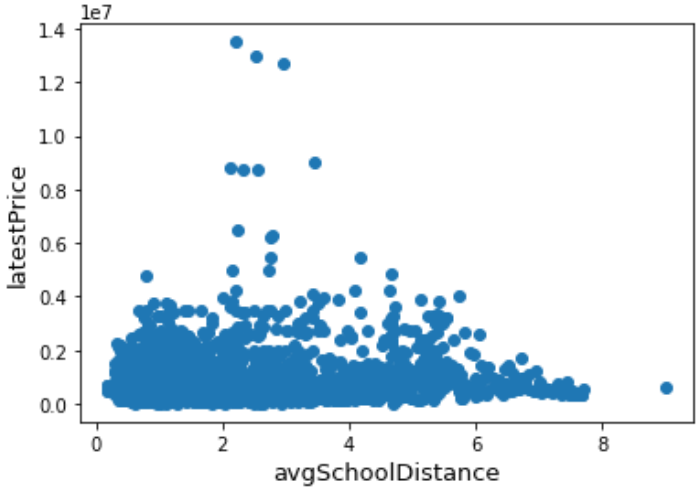


Figure 6: Price of the house plotted against average school distance

According to figure 6, there is a slight inverse relationship between higher house prices and lower average distance from school. In other words, the further away a house is from schools, the less value it would hold. Houses closer to schools were considered more desirable to families with school aged children due to commuting and safety concerns. Furthermore, education provided by schools could enhance economic performance and reduce crime rates in the area nearby. Households who live close to a school are also more likely to have easy access to recreational facilities and other amenities. So the desirability of living closer to schools could well result in an increase in the value of a residential property with a shorter distance to schools (Peng, 2007).

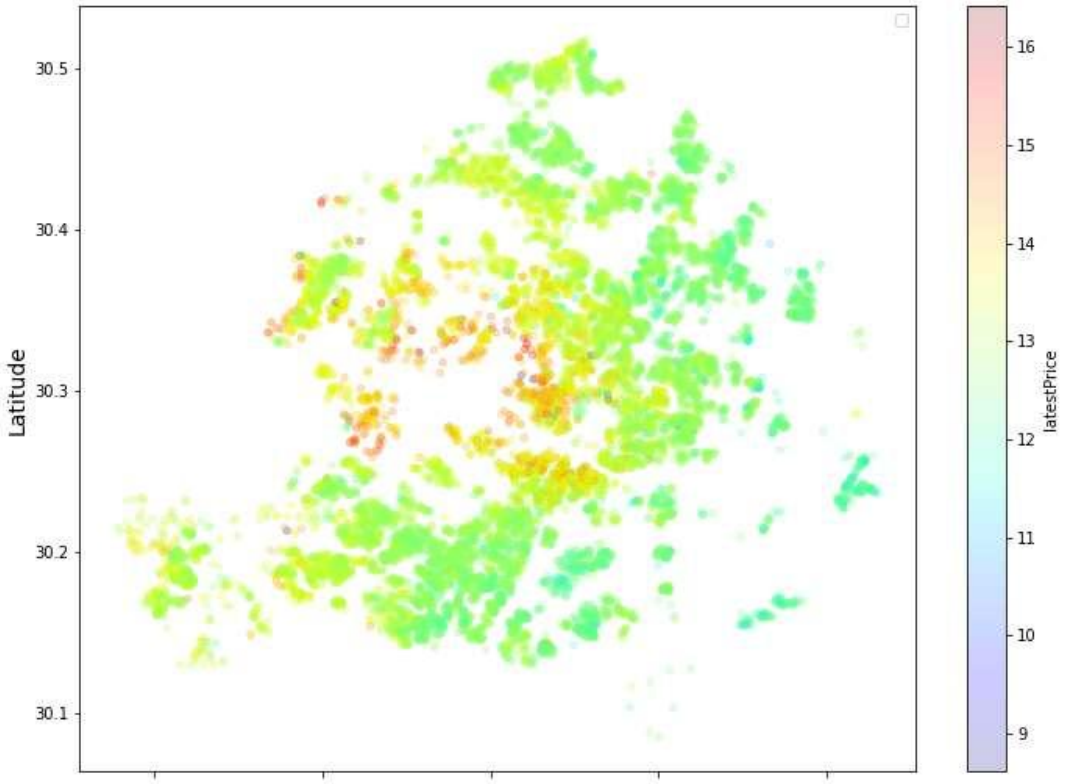


Figure 7: Price of the house plotted against location

According to figure 7, one can see that higher priced houses are located closer to the city center. This is because homebuyers would value accessibility to local amenities and work places. Housing closer to the city is not only scarce in number but also offers

services such as public transportation and trash collection initiatives that future home owners are willing to pay higher for them. City homes are closer to cultural institutions and residents would have more convenient opportunities to enjoy arts and culture than suburban dwellers (DuChene, 2021). A city home would also offer more employment opportunities for white collared professionals who are more likely to move closer to their workplaces and are willing to pay more for the location (Anas, 2017).

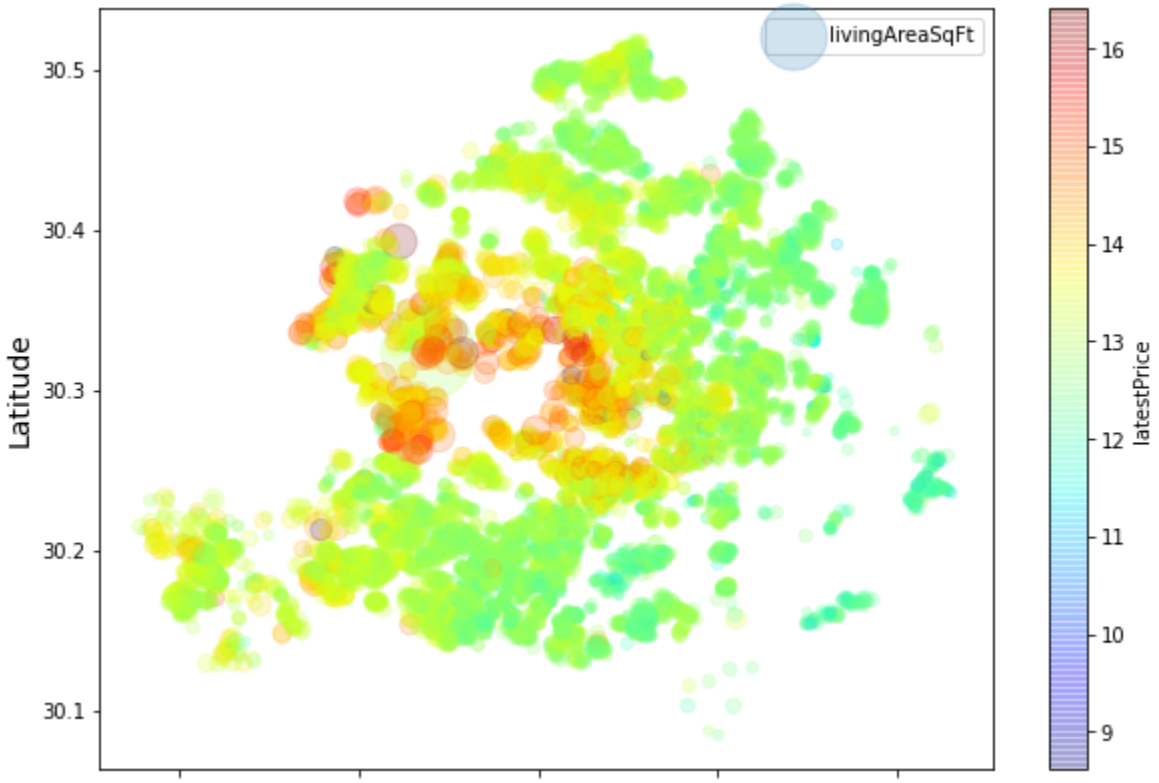


Figure 8: Price of the house plotted against location and square footage

According to figure 8, one can see that high square footage houses that are located closer to the city are priced higher. This is because not only are houses scarce in number in the city but higher square footage houses would also be in more demand. Communities in the city are less likely to expand or grow in size due to lack of real estate present and future home buyers are willing to pay more for higher square footage houses located closer to the city center. Not only do these houses offer more livable space but future homebuyers can also take advantage of local amenities and services

offered within the city. Studies have also shown that residents within the city pay higher per square footage than those who live in the suburbs (DuChene, 2021).

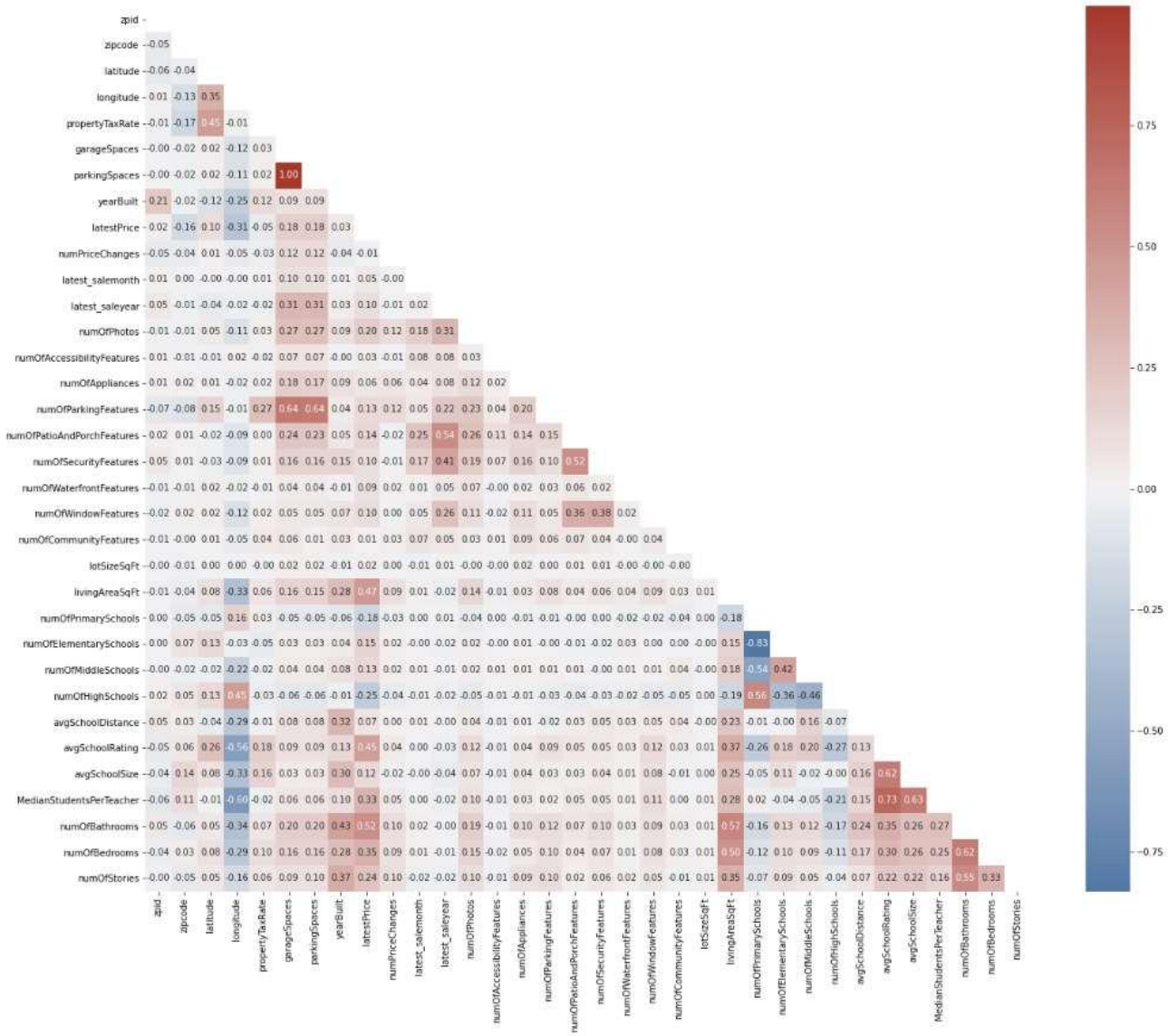


Figure 9: Broad based correlation between house features

According to figure 9, one can see the correlation between several features of the house listings. Not surprisingly, the number of garage spaces are highly correlated with number parking spaces with a score of 0.997. Higher number of garage spaces would

indicate houses that are large enough not only to accommodate for more cars but are located in communities where there are more parking spaces around them. Houses that are close to primary schools are also likely to be close to elementary schools and are correlated with a score of 0.83. Houses that are located in schools with higher school ratings are more likely to accommodate a higher median of students per teacher with a correlation of 0.73.

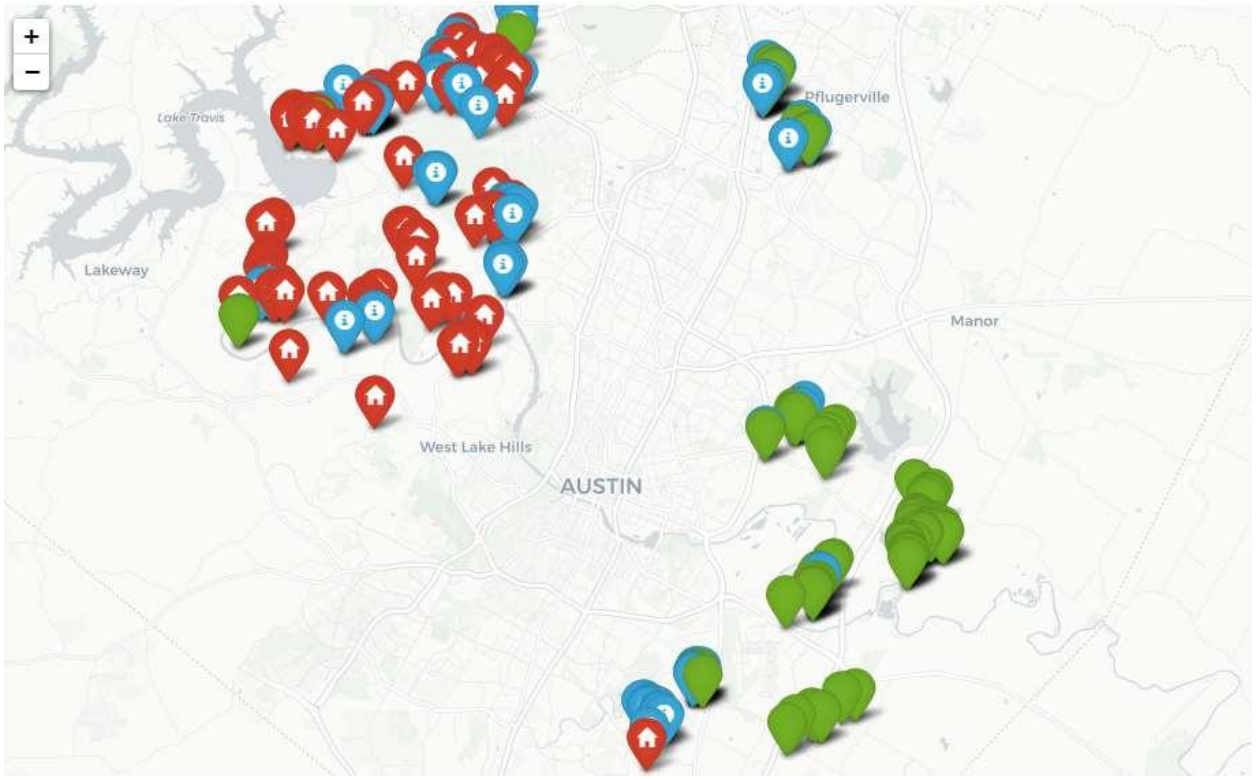


Figure 10: Housing prices and location (Green = the house's price that is less than $300,000, **Blue =** the house's price that is between $300,000 - 500,000, **Red =** the house's price that is more than $500,000)

According to figure 10, houses located in north Austin were appraised at higher values than the south. One explanation is that the amount of investment from technology companies in north Austin as opposed to the south. This would result in scarcity of houses around the technology companies that would be in high demand. As a result of this demand, prices would be higher. North Austin is also experiencing an increase in real estate and commercial development as opposed to the south which can also invite more inhabitants as a result (Pitcher, 2022). Higher demand in extension would generally cause house prices to be higher.

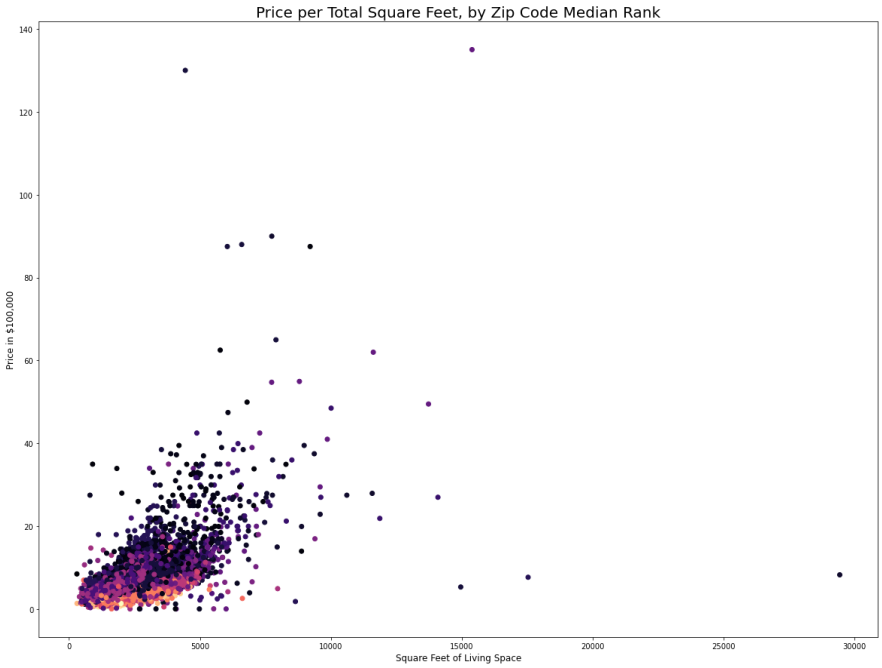
Outlier Removal:

There were multiple inconsistencies in the dataset such as

- House with 10 rooms having 27 bathrooms
- Bunch of some garage spaces are definitely just bogus numbers
- Same with Parking spaces.

Zip Code:

To better understand the relationship between zip code and latestPrice. We decided to take the median of prices of a specific zip code. Our analysis found that only there were only two values for the specific code: 78653



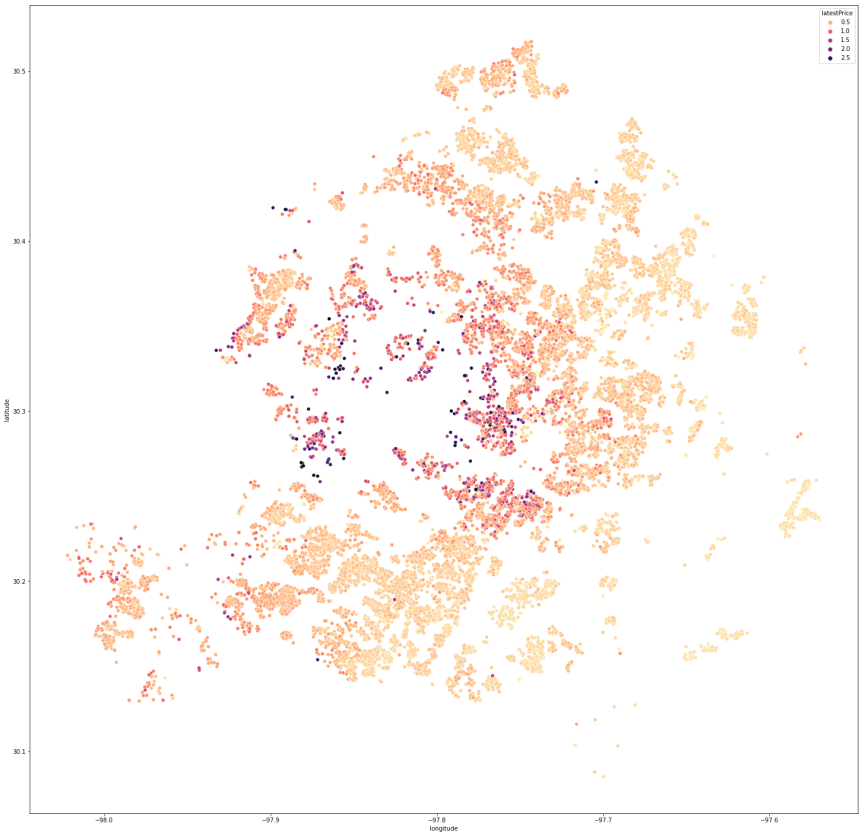
Both of these values seemed as outliers to us, so we decided to remove them from the dataset.

Feature Engineering:

1. Creating new features:

   Based on the median score of the zip code, we decided to create a ziprank feature for the dataset. Plotting ziprank on the map shows us the place near the

downtown are more expensive than the rest of the places.



2. Feature Encoding:

   Some features although numeric were categorical values such as 'zipcode', 'yearBuilt','homeType', 'hasAssociation', 'hasCooling', 'hasHeating', 'hasSpa', 'hasView','latest_salemonth', 'garageSpaces', 'propertyTaxRate'.

   We decided to label them as categories and LabelEncoded the rest of the string category values such as homeTypes.

Model Selection:

We selected four models for the regression problem; to predict the sale price based on our cleaned dataset. The models are:

1. SVR
2. Gradient Boosting
3. XGBoost
4. Random Forest Regressor.

Training and Model Selection:

We split our dataset into 80% training and rest for testing. We randomized the split to avoid biases in the dataset. We performed training on all of the models mentioned before and generated the respective RSME scores. The results showed that Random Forest outperformed the rest of the models by a huge margin with the lowest RMSE score.

Model	**RMSE**
SVR	334855.16
Gradient Boosting	32364.07
XGBoost	25309.37
Random Forest Regressor	1304.15

Our final model utilizes a combination of continuous variables and one-hot-encoded categoricals to produce a random Forest Regressor regression with a root mean squared error of 1304.

Square footage is, unsurprisingly, a key player in house pricing. And as they say, location is everything, and it is the primary influencing factor for a home price in the Austin metro area. Number of bathrooms, school rating, and lot size all contributed a large amount as well. These five features alone explain 71% of the price variance.

Related Work

There were not many studies done looking at analyzing the Austin housing market specifically. While the data was present and analyzed, the results were compared with other studies looking at which features of the house would have an impact on its price. The references cited in this paper include broad based generalizations regarding which factors would affect housing prices and compared with the results obtained from the analysis. Numerous studies have been done looking at the impact of prices on house square footage, number of bedrooms, location from city

center, and average school distance. These references were cited in the paper and also appear in the reference section.

Conclusion:

This paper determined which factors of house specifically in the Austin market would have a significant impact on its listed price. Numerous scatterplots and correlation analysis were conducted between the housing prices and the features representing these houses. Over 15,000 house listings in Austin along with their key features were analyzed to see its effect on the price. It was determined that house square footage, its location to city center, average school distance, number of bedrooms and year it was built would have some impact on its overall value. There were some instances where trends generated in the study did not explain the impact on house prices. For example, some houses, while they had a higher number of bedrooms, were still priced to be lower than houses with fewer bedrooms. Future studies should aim to look at significant features of these with higher numbers of bedrooms to determine why they were still priced to be lower. Future studies should also seek to determine the value of these listing prior to investment by technological companies. It would be beneficial to see if the same house value has gone up as a result of these investments when controlling for all other variables.

References:

Ziraldo, Katie. “Old House vs. New House: What to Consider.” Old House Vs. New House: What To Consider | Rocket Mortgage, 23 Feb. 2022, https://www.rocketmortgage.com/learn/old-house-vs-new-house.

Theriault, Melanie. “How Much Does It Cost to Build a House in 2022?” NewHomeSource, Builders Digital Experience, LLC, 7 Feb. 2022, https://www.newhomesource.com/learn/cost-to-build-house-per-square-foot/.

Bartsch, Christine. “Need More Sleeping Space? Here's How Much Value Another Bedroom Adds.” HomeLight Blog, HomeLight, Inc, 26 Feb. 2021, https://www.homelight.com/blog/how-much-value-does-a-bedroom-add/.

Huang, Peng, Impact of Distance to School on Housing Price: Evidence from a Quantile Regression (January 3, 2017). The Empirical Economics Letters, 2018, 17 (2), 149-156, Available at SSRN: https://ssrn.com/abstract=3096145

Anas, Brittany. “Is It Cheaper to Live in the City or the Suburbs?” Apartment Therapy, Apartment Therapy, LLC., 3 May 2019, https://www.apartmenttherapy.com/suburbs-vs-city-cost-of-living-265646.

DuChene, Courtney. “City Living vs Suburbs: Which Is the Right Place for You to Call Home?” HomeLight Blog, 29 Aug. 2021, https://www.homelight.com/blog/buyer-city-living-vs-suburbs/.

Pitcher, Michelle. “North Austin's Hot Housing Market Attracts New Developers.” Bizjournals.com, 11 Apr. 2022, https://www.bizjournals.com/austin/news/2022/04/11/north-austin-housing-pipeline-going-strong.html.

Egan, John. “Austin Named the No. 1 Housing Market for Growth and Stability.” CultureMap Austin, 25 Mar. 2022, https://austin.culturemap.com/news/real-estate/03-25-22-austin-best-us-metro-housing-growth-and-stability/.

Santarelli, Marco. “Austin Real Estate Market: Prices: Trends: Forecasts 2022.” Norada Real Estate Investments, 21 Mar. 2022, https://www.noradarealestate.com/blog/austin-real-estate-market/.

Appendices

A. Abstract **1**
B. Introduction **1**
C. Background **1**
D. Experiment Methodology **2**
a. Get Data 2

b. Clean, Prepare & Manipulate Data 2
c. Feature selection and Feature engineering 3
d. Train a baseline model 3
e. Model selection and hyperparameter tuning 3
E. **Results** **3**
F. Related Work **12**
G. Conclusion **12**
H. References **13**
I. Coding **14**

```
# -*- coding: utf-8 -*-

"""Datamining_Project.ipynb


Automatically generated by Colaboratory.


Original file is located at

    https://colab.research.google.com/drive/1X_xd4DnXi78urXiQO45ltb2aEjUmk7uC

"""


!unzip austinHousingData.csv.zip


import pandas as pd


data = pd.read_csv("austinHousingData.csv")


# Plots

import seaborn as sns

import matplotlib.pyplot as plt

import numpy as np


df = data.copy()


"""# EDA
```

```
## Checking for Data types, null values, missing values and unique values
"""

print("Rows     : ", data.shape[0])
print("Columns  : ", data.shape[1])
print("\nFeatures : \n", data.columns.tolist())
print("\nMissing values :  ", data.isnull().sum().values.sum())
print("\nUnique values :  \n", data.nunique())

data.dtypes

data.isnull().sum()

data.head()

"""## Checking Target Value's distribution"""

data['latestPrice'].describe()

#Deviate from the normal distribution.
#Have appreciable positive skewness.
#Shows shape peakedness.
```

```
sns.distplot(data['latestPrice']);


#skewness and kurtosis
print("Skewness: %f" % data['latestPrice'].skew())
print("Kurtosis: %f" % data['latestPrice'].kurt())


"""## checking if YearBuilt plays a role; slightly does"""


var = 'yearBuilt'
data = pd.concat([data['latestPrice'], data[var]], axis=1)
f, ax = plt.subplots(figsize=(16, 8))
fig = sns.boxplot(x=var, y="latestPrice", data=data)
fig.axis(ymin=0, ymax=4000000);
plt.xticks(rotation=90);


"""## logically 'livingAreaSqFt' should affect the price more."""


fig, ax = plt.subplots()
ax.scatter(x = data['livingAreaSqFt'], y = data['latestPrice'])
plt.ylabel('latestPrice', fontsize=13)
plt.xlabel('livingAreaSqFt', fontsize=13)
plt.show()
```

```
"""## Can see on outlier there"""


fig, ax = plt.subplots()

ax.scatter(x = data['numOfBedrooms'], y = data['latestPrice'])

plt.ylabel('latestPrice', fontsize=13)

plt.xlabel('numOfBedrooms', fontsize=13)

plt.show()


df.describe()


"""## Taking log transform of the latestPrice as its positivitly skewed"""


data['latestPrice'] = np.log(data['latestPrice'])

sns.distplot(data['latestPrice']);


"""## Plotting longitude and latitude areas to see if how all 3 variables varies with
prices"""


house_map_plot = data[['longitude','latitude', 'latestPrice','livingAreaSqFt']]


house_map_plot.plot(kind='scatter', x='longitude', y='latitude', alpha=0.2,
                    figsize=(10,7),
                    c='latestPrice', cmap=plt.get_cmap('jet'), colorbar=True)
```

```
plt.ylabel("Latitude", fontsize=14)

plt.xlabel("Longitude", fontsize=14)

plt.legend()

plt.show()

## proves that locations close to city center are more expensive


house_map_plot.plot(kind='scatter', x='longitude', y='latitude', alpha=0.2,

        s=house_map_plot['livingAreaSqFt']/50, label='livingAreaSqFt', figsize=(10,7),

        c='latestPrice', cmap=plt.get_cmap('jet'), colorbar=True)



plt.ylabel("Latitude", fontsize=14)

plt.xlabel("Longitude", fontsize=14)

plt.legend()

plt.show()

## proves that locations close to city center and larger livingAreaSqFt are more
expensive



"""## Removing all categorically data to see correlation """


#working with only numbers right now
```

```
df = data.select_dtypes(include='number')

df.head()


# Finding numeric features

numeric_dtypes = ['int16', 'int32', 'int64', 'float16', 'float32', 'float64']

numeric = []

for i in df.columns:

    if df[i].dtype in numeric_dtypes:

      numeric.append(i)

# visualising some more outliers in the data values

fig, axs = plt.subplots(ncols=2, nrows=0, figsize=(12, 120))

plt.subplots_adjust(right=2)

plt.subplots_adjust(top=2)

sns.color_palette("husl", 8)

for i, feature in enumerate(list(df[numeric]), 1):

    plt.subplot(len(list(numeric)), 3, i)

    sns.scatterplot(x=feature, y='latestPrice', hue='latestPrice', palette='Blues', data=df)


    plt.xlabel('{}'.format(feature), size=15,labelpad=12.5)

    plt.ylabel('latestPrice', size=15, labelpad=12.5)


    for j in range(2):

        plt.tick_params(axis='x', labelsize=12)
```

```
        plt.tick_params(axis='y', labelsize=12)


    plt.legend(loc='best', prop={'size': 10})


plt.show()


df.dtypes


df.corr()


import numpy as np


def top_entries(df):
    mat = df.corr().abs()


    # Remove duplicate and identity entries
    mat.loc[:,:] = np.tril(mat.values, k=-1)
    mat = mat[mat>0]


    # Unstack, sort ascending, and reset the index, so features are in columns
    # instead of indexes (allowing e.g. a pretty print in Jupyter).
    # Also rename these it for good measure.
    return (mat.unstack()
```

```
        .sort_values(ascending=False)
        .reset_index()
        .rename(columns={
            "level_0": "feature_a",
            "level_1": "feature_b",
            0: "correlation"
        }))

corr_df = top_entries(df)

#highest corr
corr_df.head()

import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib import cm
matrix = df.corr()

# Create a mask
plt.figure(figsize=(16,12))
cmap = sns.diverging_palette(250, 15, s=75, l=40,
                             n=9, center="light", as_cmap=True)
mask = np.triu(np.ones_like(matrix, dtype=bool))
```

```
sns.heatmap(matrix, mask=mask, center=0, annot=True,
            fmt='.2f', square=True, cmap=cmap)


"""## Latest prices affect by 2 features positivitly (anything above 0.3)"""


corr_df[corr_df["feature_a"] == "latestPrice"]


#Next Steps:

#Outlier removal.

#Feature Engineering; setting some categorical data.

#I'll run two model on situations; one without NLP and Images and one with both (need to search here)


data.sort_values('livingAreaSqFt', ascending=False).head(5)


data.iloc[705]


data.drop(index=[705], inplace=True)


data.zipcode.unique


zipcode_data = data.groupby('zipcode').aggregate(np.mean)

zipcode_data.reset_index(inplace=True)
```

```
data['count'] = 1

count_houses_zipcode = data.groupby('zipcode').sum()

count_houses_zipcode.reset_index(inplace=True)

count_houses_zipcode = count_houses_zipcode[['zipcode','count']]

data.drop(['count'], axis = 1, inplace = True)


zipcode_data = pd.merge(zipcode_data, count_houses_zipcode, how='left',
on=['zipcode'])

zipcode_data.head(10)


import folium as folium


from scipy import stats

fig, ax = plt.subplots(figsize=(12,4))

sns.boxplot(y = 'numOfBathrooms', x = 'latestPrice', data = data,width = 0.8,orient = 'h',
showmeans = True, fliersize = 3, ax = ax)

plt.show()


# Calculate the correlation coefficient

r, p = stats.pointbiserialr(data['numOfBathrooms'], data['latestPrice'])

print ('point biserial correlation r is %s with p = %s' %(r,p))


data.loc[(data['numOfBathrooms']==0) & (data['numOfBedrooms']>0) & (data['yearBuilt']
> 1989), 'numOfBathrooms'] = 2
```

```
data.loc[(data['numOfBathrooms']==0) & (data['numOfBedrooms']>0) & (data['yearBuilt'] <= 1989), 'numOfBathrooms'] = 1

data.loc[(data['numOfBathrooms']==0) & (data['numOfBedrooms']>=3) & (data['yearBuilt'] > 1989), 'numOfBathrooms'] = 2.5

data.loc[(data['numOfBathrooms']==0) & (data['numOfBedrooms']>=3) & (data['yearBuilt'] <= 1989), 'numOfBathrooms'] = 2


data.drop(data[data['numOfBathrooms']==0].index, inplace=True)

data.drop(data[data['numOfBedrooms']==0].index, inplace=True)


data.loc[data.index==8597, 'numOfBedrooms'] = 2


data.loc[(data['garageSpaces'] > 3) & (data['latestPrice'] < 1000000) & (data['homeType'] == 'Single Family'), 'garageSpaces'] = 3

data.loc[(data['garageSpaces'] > 5) & (data['latestPrice'] > 1000000)& (data['homeType'] == 'Single Family'), 'garageSpaces'] = 4

data.loc[data.index==6885, 'garageSpaces'] = 2


data.loc[(data['parkingSpaces'] > 3) & (data['latestPrice'] < 1000000 & (data['homeType'] == 'Single Family')), 'parkingSpaces'] = 3

data.loc[(data['parkingSpaces'] > 5) & (data['latestPrice'] > 1000000& (data['homeType'] == 'Single Family')), 'parkingSpaces'] = 5

data.loc[data.index==6885, 'parkingSpaces'] = 2


data.sort_values('parkingSpaces', ascending=False).head(5)
```

```
zipsorted =
pd.DataFrame(data.groupby('zipcode')['latestPrice'].median().sort_values(ascending=Tr
ue))


# divide our dataframe into groups with entries per group as specified above,

# and assign this number to a new column

zipsorted['rank'] = np.divmod(np.arange(len(zipsorted)), 1)[0]+1


# function that looks up a segment that a data entry belongs to

def make_group(x, frame, column):

    y = frame.loc[(frame.index == x)][column]

    z = np.array(y)

    z[0]

    return z[0]


# make a new column on our dataframe. Look up each zip entry's group, and append to
the column.

data['zip_rank'] = data['zipcode'].apply(lambda x: make_group(x, zipsorted, 'rank'))


# apply the median home price per zip code to the data frame

data['median_zip'] = data['zipcode'].apply(lambda x:
round(data.loc[data['zipcode']==x]['latestPrice'].median(), 0))


# visualize zip code as a color function, on a plot of price per square footage
```

```
fig, ax = plt.subplots(figsize=(20, 15))


ax.scatter(data['livingAreaSqFt'], data['latestPrice'] /100000, c=data['zip_rank'],
cmap='magma_r')


ax.set_xlabel('Square Feet of Living Space', fontsize=12)

ax.set_ylabel('Price in $100,000', fontsize=12)

ax.set_title('Price per Total Square Feet, by Zip Code Median Rank', fontsize=20)


# we're dropping the values above 3 million, and the 3 entries from zipcode 78734

data.drop(data[data['latestPrice']>3000000].index, inplace=True)

data.drop(data[data['zipcode']==78734].index, inplace=True)


plt.figure(figsize=(25,25))


sns.scatterplot(data=data, x="longitude", y="latitude", hue="latestPrice",
palette="magma_r");


plt.figure(figsize=(25,25))


sns.scatterplot(data=data, x="longitude", y="latitude", hue="zip_rank",
palette="magma_r");


"""# Feature Engineering."""
```

```
# we're going to convert some of our ordinal features to binary 0/1

convert_to_bool = ['numOfAccessibilityFeatures', 'numOfAppliances',
'numOfParkingFeatures', 'numOfPatioAndPorchFeatures', 'numOfSecurityFeatures',
'numOfWaterfrontFeatures', 'numOfWindowFeatures', 'numOfCommunityFeatures']


df_convert_to_bool = data[convert_to_bool]

df_convert_to_bool.describe()


data.columns




del data["description"]

del data["homeImage"]

del data["numOfPhotos"]


#city, streetAddress, homeType, latest_saledate, latestPriceSource

del data["city"]

del data["streetAddress"]

del data["latestPriceSource"]

del data["latest_saledate"]


categories = ['zipcode', 'yearBuilt','homeType', 'hasAssociation', 'hasCooling',
'hasHeating', 'hasSpa', 'hasView','latest_salemonth', 'garageSpaces', 'propertyTaxRate',
]
```

```
for item in categories:
    data[item] = data[item].astype('category')


from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
data.homeType= le.fit_transform(data.homeType.values)


check = data.copy()
from sklearn import preprocessing
lbl = preprocessing.LabelEncoder()
check['homeType'] = lbl.fit_transform(check['homeType'].astype(str))


from sklearn.model_selection import cross_val_score, RepeatedKFold, train_test_split,
GridSearchCV
randomstate = 45


y = pd.DataFrame(data['latestPrice'])
x = data.drop('latestPrice', axis=1,)


# creating our train/validation sets and our test sets
train_labels, test_labels, y_train, y_test = train_test_split(data, y, test_size=0.2,
random_state=randomstate)


from sklearn.model_selection import GridSearchCV
```

```
from sklearn.model_selection import KFold, cross_val_score

from sklearn.metrics import mean_squared_error

from sklearn.preprocessing import OneHotEncoder

from sklearn.preprocessing import LabelEncoder

from sklearn.pipeline import make_pipeline

from sklearn.preprocessing import scale

from sklearn.preprocessing import StandardScaler

from sklearn.preprocessing import RobustScaler

from sklearn.decomposition import PCA

from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor,
AdaBoostRegressor, BaggingRegressor

from sklearn.kernel_ridge import KernelRidge

from sklearn.linear_model import Ridge, RidgeCV

from sklearn.linear_model import ElasticNet, ElasticNetCV

from sklearn.svm import SVR

from mlxtend.regressor import StackingCVRegressor

import lightgbm as lgb

from lightgbm import LGBMRegressor

from xgboost import XGBRegressor

kf = KFold(n_splits=12, random_state=42, shuffle=True)

# Define error metrics

def rmsle(y, y_pred):

    return np.sqrt(mean_squared_error(y, y_pred))
```

```
# Light Gradient Boosting Regressor
lightgbm = LGBMRegressor(objective='regression',
                         num_leaves=6,
                         learning_rate=0.01,
                         n_estimators=7000,
                         max_bin=200,
                         bagging_fraction=0.8,
                         bagging_freq=4,
                         bagging_seed=8,
                         feature_fraction=0.2,
                         feature_fraction_seed=8,
                         min_sum_hessian_in_leaf = 11,
                         verbose=-1,
                         random_state=42)


# XGBoost Regressor
xgboost = XGBRegressor(learning_rate=0.01,
                       n_estimators=6000,
                       max_depth=4,
                       min_child_weight=0,
                       gamma=0.6,
                       subsample=0.7,
                       colsample_bytree=0.7,
```

```
                    objective='reg:linear',
                    nthread=-1,
                    scale_pos_weight=1,
                    seed=27,
                    reg_alpha=0.00006,
                    random_state=42)

# Ridge Regressor
ridge_alphas = [1e-15, 1e-10, 1e-8, 9e-4, 7e-4, 5e-4, 3e-4, 1e-4, 1e-3, 5e-2, 1e-2, 0.1,
0.3, 1, 3, 5, 10, 15, 18, 20, 30, 50, 75, 100]
ridge = make_pipeline(RobustScaler(), RidgeCV(alphas=ridge_alphas, cv=kf))

# Support Vector Regressor
svr = make_pipeline(RobustScaler(), SVR(C= 20, epsilon= 0.008, gamma=0.0003))

# Gradient Boosting Regressor
gbr = GradientBoostingRegressor(n_estimators=6000,
                                learning_rate=0.01,
                                max_depth=4,
                                max_features='sqrt',
                                min_samples_leaf=15,
                                min_samples_split=10,
                                loss='huber',
                                random_state=42)
```

```
# Random Forest Regressor
rf = RandomForestRegressor(n_estimators=1200,
                           max_depth=15,
                           min_samples_split=5,
                           min_samples_leaf=5,
                           max_features=None,
                           oob_score=True,
                           random_state=42)

# Stack up all the models above, optimized using xgboost
stack_gen = StackingCVRegressor(regressors=(xgboost, lightgbm, svr, ridge, gbr, rf),
                                meta_regressor=xgboost,
                                use_features_in_secondary=True)

lgb_model_full_data = lightgbm.fit(train_labels, y_train)

svr_model_full_data = svr.fit(train_labels, y_train)

print('Ridge')
ridge_model_full_data = ridge.fit(train_labels, y_train)

print('RandomForest')
```

```
rf_model_full_data = rf.fit(train_labels, y_train)

print('GradientBoosting')

gbr_model_full_data = gbr.fit(train_labels, y_train)


# Blend models in order to make the final predictions more robust to overfitting

def blended_predictions(X):

    return ((0.1 * ridge_model_full_data.predict(X)) + \

            (0.2 * svr_model_full_data.predict(X)) + \

            (0.1 * gbr_model_full_data.predict(X)) + \

            (0.1 * lgb_model_full_data.predict(X)) + \

            (0.05 * rf_model_full_data.predict(X)))


# Get final precitions from the blended model

def testAndGetRMSEscore(model):

  test_predictions = model.predict(test_labels)

  rmse = round(np.sqrt(mean_squared_error(y_test, test_predictions)), 2)

  print("RMSE",rmse)


testAndGetRMSEscore(svr_model_full_data)

testAndGetRMSEscore(gbr_model_full_data)

testAndGetRMSEscore(lgb_model_full_data)

testAndGetRMSEscore(rf_model_full_data)
```

Syed Araib Karim 11479107
Hamza Tahir 10966566
Lydia Karumanch 11466410
Phongsiri Nirachornkul 11125038