Project 4: Real Estate Valuation

Tim Spendley, Christian Blomgren, Drew Kirke

Utilizing Machine Learning to predict housing prices in the Phoenix metro region.

Presentation Overview

1. **Intro/Project Summary**
2. **The Data**
3. **Map**
4. **The Modeling Process**
5. **The Results**
6. **Conclusion**
7. **Questions?**

Empowering Zillow's Search Engine, with attention to detail.

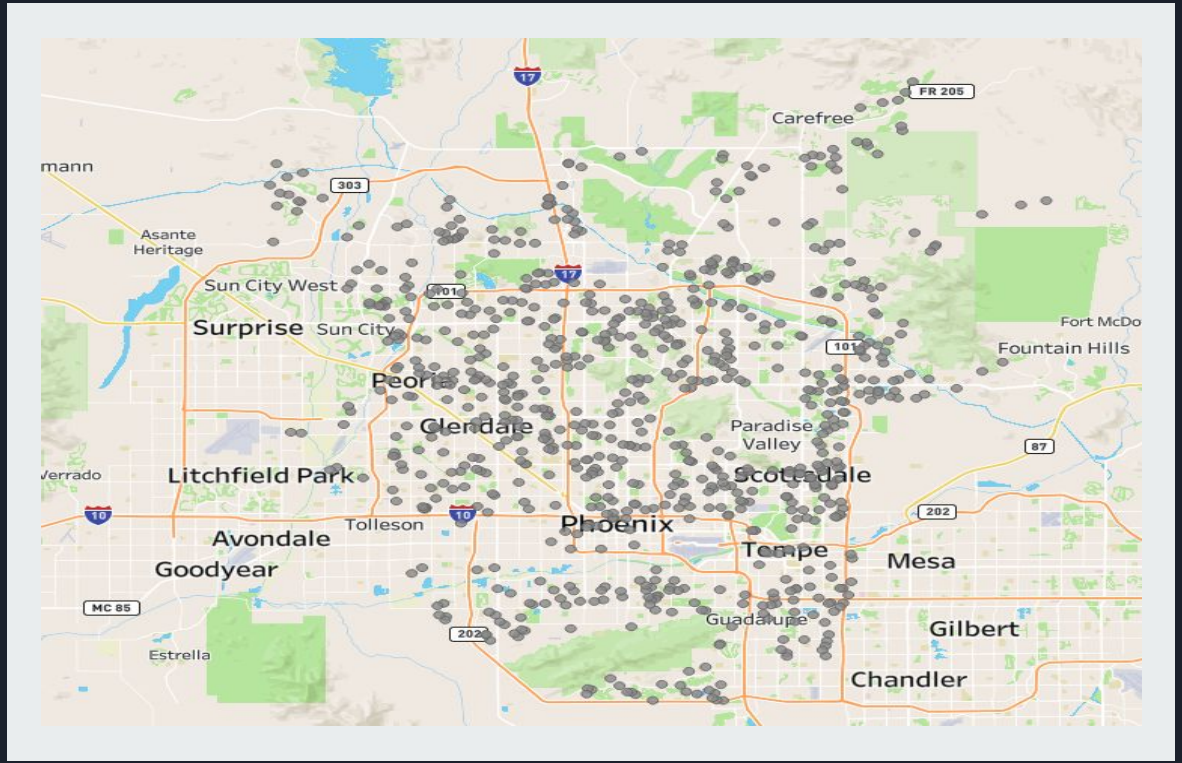
Our data extraction, step-by-step

1. **Utilize** Apify to scrape website data
2. **Transform** data into multiple data frames
3. **Clean** data to reflect model attributes
4. **Load** data to SQLite

Location, Location, Location

Search Cities:

- **Phoenix**
- **Glendale**
- **Peoria**
- **Scottsdale**
- **Tempe**



Behind the Scenes...

Processing, cleaning, formatting to fit our standards.

```
# Create an engine that connects to the SQLite database
engine = create_engine('sqlite:///housing_model.db')
```

```
# Define the SQL query for the training data using text()
query = text('SELECT * FROM train_data')
```

```
# Use Pandas to read the query results into a DataFrame
train_data = pd.read_sql_query(query, engine)
train_data
```

	id	beds	baths	sqft	latitude	longitude	price
0	7463218	3.0	2.0	1682.0	33.534412	-112.221540	385000
1	7464552	4.0	3.0	1868.0	33.522633	-112.231800	400000
2	7464673	3.0	2.0	1819.0	33.509426	-112.245700	435000
3	7468885	3.0	1.0	1006.0	33.492165	-112.212120	330000
4	7470268	3.0	2.0	1390.0	33.486015	-112.252590	370000
...	...	...	...	...	...	...	...
815	2078977316	2.0	3.0	1272.0	33.643810	-112.045210	325000
816	2084045887	1.0	1.0	728.0	33.586388	-112.204860	30000
817	2088234198	4.0	3.0	2303.0	33.484330	-112.000626	910000
818	2098280181	NaN	NaN	NaN	33.469505	-112.024230	260000
819	2104311557	2.0	2.0	1378.0	33.525360	-112.058960	368000

820 rows × 7 columns

```
train_data_df = train_data.dropna()
train_data_df
```

	beds	baths	sqft	latitude	longitude	price
0	3.0	2.0	1682.0	33.534412	-112.221540	385000
1	4.0	3.0	1868.0	33.522633	-112.231800	400000
2	3.0	2.0	1819.0	33.509426	-112.245700	435000
3	3.0	1.0	1006.0	33.492165	-112.212120	330000
4	3.0	2.0	1390.0	33.486015	-112.252590	370000
...	...	...	...	...	...	...
814	5.0	3.0	2900.0	33.506016	-112.090930	720000
815	2.0	3.0	1272.0	33.643810	-112.045210	325000
816	1.0	1.0	728.0	33.586388	-112.204860	30000
817	4.0	3.0	2303.0	33.484330	-112.000626	910000
819	2.0	2.0	1378.0	33.525360	-112.058960	368000

806 rows × 6 columns

Behind the Scenes...

```
# Creating a model for housing price prediction
housing_price_model = Sequential([
    Dense(32, activation='linear', input_dim=len(X_train[0])),
    Dense(32, activation='linear'),
    Dense(1)
])
```

```
# Compiling the model for regression
housing_price_model.compile(optimizer='adam', loss='mean_absolute_error')
```

```
housing_price_model.fit(X_train, y_train, batch_size=64, epochs=100)
```

Results

How successful is the model at predicting home prices?

Our metrics for prediction power:

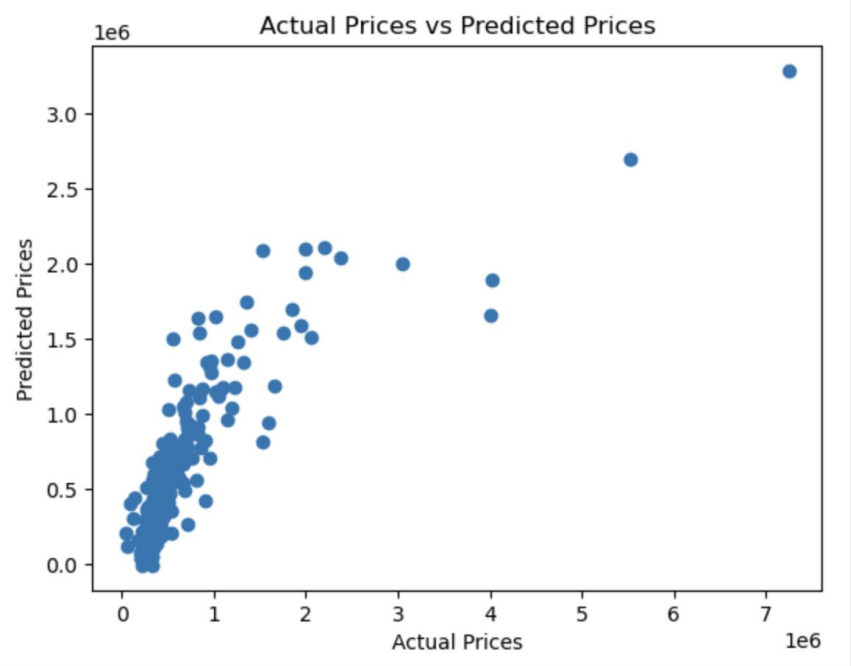
1. **Mean Absolute Error**: Average magnitude of the errors without considering their direction.
2. **Mean Standard Error**: Average squared difference between actual and predicted values, emphasizing larger errors.
3. **Root Mean Squared Error**: Square root of MSE, giving error magnitude in the same units as the target variable.
4. **R-squared (R^2)**: Proportion of the variance in the dependent variable that is predictable from the independent variables.

Results:

MAE 227,021,944,224

MSE 231,140

R^2 64%



Conclusion

So what did we learn?

Results were not ideal, so next time we might:

1. **Identify** additional attributes that could impact price variance and determine data sources.
2. **Execute** other model types (Regression Tree, KNN) to determine if a better model fit can be found.
3. **Expand** our dataset by including additional locations in the metro area to increase the sample size.
4. **Research** similar projects to understand whether others investigating a similar outcome of new ideas that might aid in the process.

Questions?