The Future of Home Values - Machine Learning Predictions

Francisco Lara Loza - Levi McLeod

Steve Read - Veronica Ostapowich

April 17, 2023

FORECASTING FORTRESS

A Project-Group-3 Machine-Learning Production

INTRO TO HOME VALUE PREDICTIONS

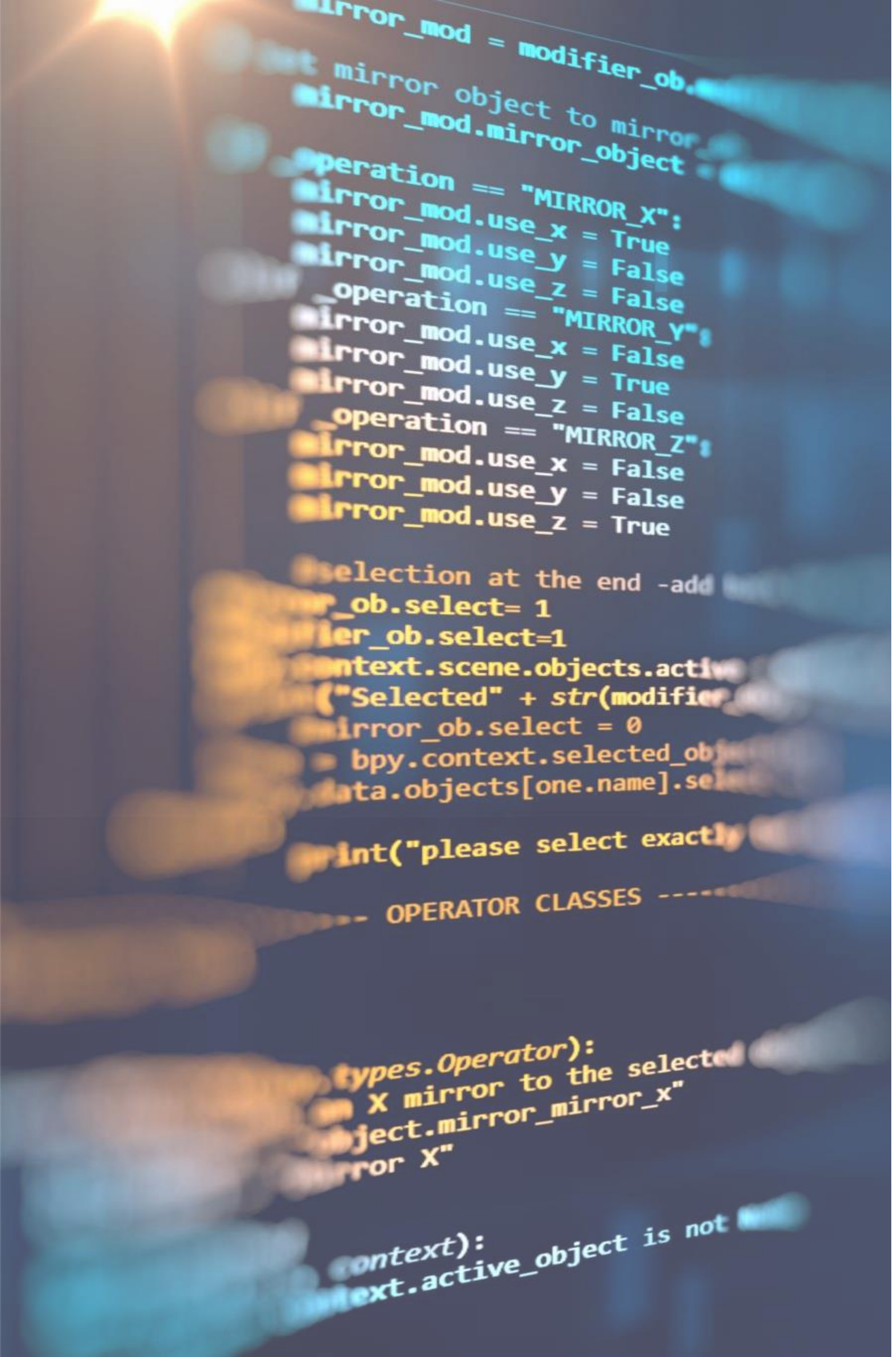
DATA COLLECTION AND PREPARATION

MODEL SELECTION AND TRAINING

MODEL EVALUATION AND REFINEMENT

REAL WORLD APPLICATIONS OF HOME VALUE PREDICTIONS

CONCLUSION



Tools Utilized

- HTML/CSS/JS
- Jupyter notebook
- Matplotlib
- Pandas
- RapidAPI
- Scikit-learn
- Spark/SQL
- SQLite DB

Intro to Home Value Predictions

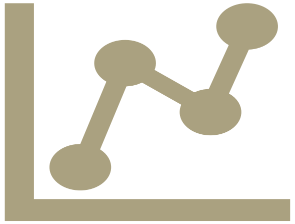
- **Home value predictions using machine learning** is a rapidly growing field that has the potential to **revolutionize the real estate industry**.
- By leveraging advanced algorithms and statistical models, machine learning can help **predict** home values with **greater accuracy** than ever before.
- One of the key **benefits** of using machine learning for home value predictions is that it can consider a **wide range of factors** that may impact a home's value, such as location, neighborhood demographics, and local market trends.

Data Collection and Preparation

The **first** step in building a machine learning model for home value predictions is to **collect and prepare the data**.

This involves **gathering** information on a **large number of homes in a given area**, including details such as square footage, number of bedrooms and bathrooms, and recent sales prices.

Once the data has been collected, it must be **cleaned and prepared for analysis**.



This typically involves **removing any outliers or errors**, and standardizing the data so that it can be used effectively by the machine learning algorithms.

Model Selection and Training

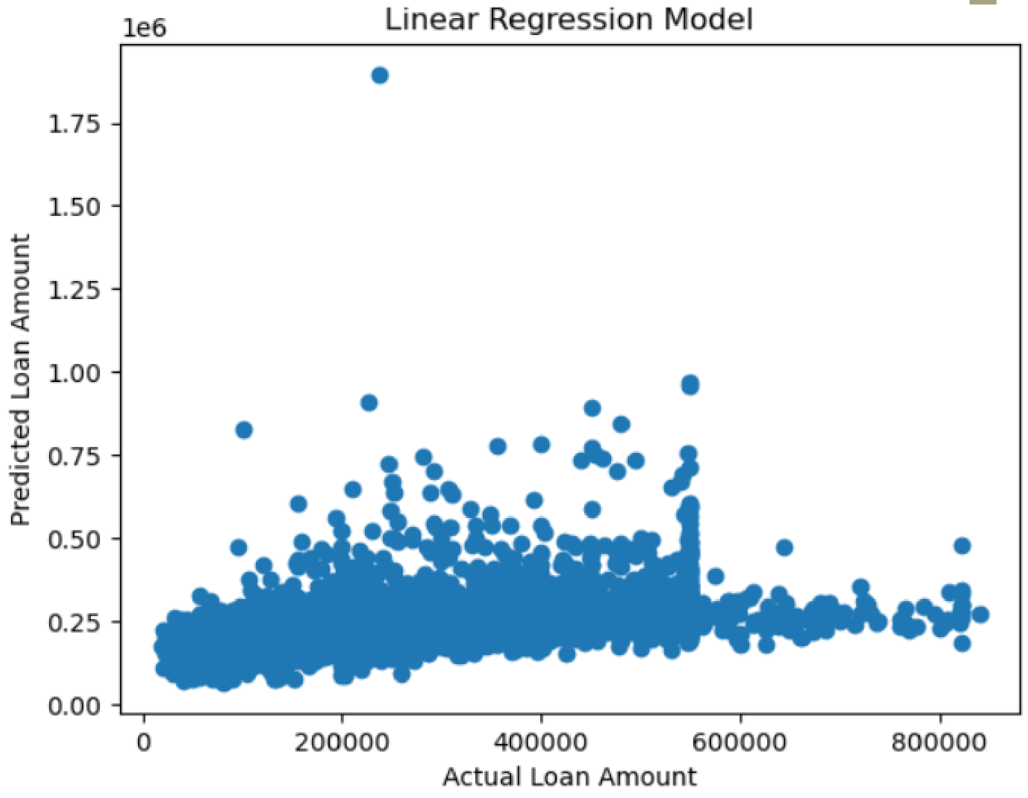
After the data has been collected and prepared, the next step is to **select an appropriate machine learning model and train** it using the data.

There are many different types of models that can be used for home value predictions, **including linear regression, decision trees, and neural networks**.

During the training phase, the machine learning algorithm will learn to **identify patterns and relationships between different features** of the homes and their corresponding values.

This process may involve **adjusting various parameters and hyperparameters** to optimize the performance of the model.

Our First Machine Learning Model yielded low accuracy over a large, unfiltered dataset



- Dataset: **fhfagov-2021.csv**
- It is a linear regression model attempting to predict home values from a fhfa.gov public dataset.
- The model uses a target variable of 'NoteAmount' (the amount the bank has loaned to an individual in our dataset)
- The independent variables are 'TotalyMonthlyIncomeAmount' and 'LTVRatioPercentage' (loan-to-value ratio -- think equity)

Our second Linear Regression Machine Learning example improved, but only slightly

```
# Make predictions on the test set
y_pred = model.predict(X_test)
```

```
# Calculate the evaluation metrics
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = mean_squared_error(y_test, y_pred, squared=False)
r2 = r2_score(y_test, y_pred)

print(f"Mean Absolute Error: {mae:.2f}")
print(f"Mean Squared Error: {mse:.2f}")
print(f"Root Mean Squared Error: {rmse:.2f}")
print(f"R-squared: {r2:.2f}")
```

```
Mean Absolute Error: 42456.52
Mean Squared Error: 2546348534.07
Root Mean Squared Error: 50461.36
R-squared: 0.10
```

- Dataset: **Tulsa, OK based on data from the years 2000 to 2023**
- This model had a low accuracy but taught us how to finetune our parameters.
- The current model is only 10% accurate with an **R-squared value of 0.10**

3rd Attempt - This time we used the Random Forest Regressor

```python
# Fit the model to the training data
model.fit(train_data[ind_vars], train_data[target_var])
```

```
RandomForestRegressor(max_depth=20, min_samples_split=15, n_estimators=500,
                      random_state=42)
```

```python
# Make predictions on the test data
test_data['predictions'] = model.predict(test_data[ind_vars])
```

```python
# Calculate the mean absolute error
mae = (test_data[target_var] - test_data['predictions']).abs().mean()
```

```python
print(f"Mean Absolute Error: {mae:.2f}")
```

```
Mean Absolute Error: 77396.07
```

- For this model, we achieved a **Mean Absolute Error** (MAE) of 80015.92 which means, on average, the predictions made by the **Random Forest Regressor** are off by about $80,016 in terms of the approved loan amount by a traditional bank.
- According to this model, if the actual value of a loan is **$200K**, then the prediction made by this model for that amount might be around **$280K**
- We decided to continue improving the model.

Optimized Model

```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

```
# Scale the data using StandardScaler
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

```
# Create logistic regression object
lr = LogisticRegression()

# Fit the model on training data
lr.fit(X_train_scaled, y_train)

# Make predictions on test data
y_pred = lr.predict(X_test_scaled)

# Calculate accuracy score
accuracy = accuracy_score(y_test, y_pred)
print('Accuracy:', accuracy)

# Calculate other evaluation metrics
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)

print("Precision:", precision)
print("Recall:", recall)
print("F1 Score:", f1)
```

```
Accuracy: 0.9200913242009132
Precision: 0.897196261682243
Recall: 0.9365853658536586
F1 Score: 0.9164677804295943
```

- Data: **housing.csv**
- The code uses **logistic regression** to predict whether a house's price is above the median based on several features. The data is retrieved from a **SQLite database**, analyzed for insights, split into training and testing sets, scaled, and trained.
- In model optimization, we took a similar approach but only used features with a high correlation to the target variable.
- Our model achieved 92% predictive accuracy.

Optimized Model

```python
# Preprocess the data
df = df.dropna()
X = df.drop("price", axis=1)
y = df["price"]
scaler = StandardScaler()
X = scaler.fit_transform(X)
```

```python
# Add quadratic features
poly = PolynomialFeatures(degree=3)
X_poly = poly.fit_transform(X)
```

```python
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X_poly, y, test_size=0.2, random_state=42)
```

```python
# Train a linear regression model on the training data
model = LinearRegression()
model.fit(X_train, y_train)
```

```
LinearRegression()
```

```python
# Evaluate the model on the testing data
y_pred = model.predict(X_test)
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print("MAE:", mae)
print("R^2:", r2)
```

```
MAE: 0.08277684853366725
R^2: 0.9814852880289772
```

- Our final model uses a polynomial features for our linear regression model.
- This model gives us 98% accuracy when predicting home values in Tulsa, OK based on: 1) Bedrooms, 2) Bathrooms, 3) Square Footage of house & 4) Year Built
- It is using a RapidAPI function to acquire the dataset.

Model Evaluation and Refinement

Once the machine learning model has been **trained**, it must be **evaluated** to determine its **accuracy** and **effectiveness**.

This typically involves **testing** the model on a **separate set of data** that was not used during the training phase, in order to assess its ability to generalize to new situations.

Based on the results of the evaluation, the model may need to be **refined or adjusted** in order to **improve its performance**.

This may involve **tweaking** the **parameters** or hyperparameters of the model or **collecting additional data** to improve its accuracy.

Real World Applications of Home Value Predictions

- **Home value predictions using machine learning** have a wide range of real-world applications, beyond simply helping buyers and sellers make more informed decisions.
- For example, **lenders** and **insurers** may use these predictions to **assess risk** and make more **accurate pricing decisions**.
- Similarly, **city planners** and **policymakers** may use these predictions to **identify areas of high demand** and allocate resources accordingly, or to track **changes in property values** over time in response to economic or demographic shifts.

Conclusion

Home value predictions using machine learning represent a **powerful tool** for both individuals and organizations looking to make more informed **decisions** about real estate.

By leveraging advanced algorithms and statistical models, these predictions can provide greater **accuracy** and **insight** than traditional methods.

While there are still **challenges** and **limitations** to be addressed in this field, the potential benefits are significant, and we can expect to see **continued grow**th and **innovation** in the years ahead.

References

Project Requirements - https://courses.bootcampspot.com/courses/2982/assignments/42956?module_item_id=804131

How to build your first neural network to predict house prices with Keras

Retrieved from https://www.freecodecamp.org/news/how-to-build-your-first-neural-network-to-predict-house-prices-with-keras-f8db83049159/

Public Use Database - Fannie Mae and Freddie Mac

https://www.fhfa.gov/DataTools/Downloads/Pages/Public-Use-Databases.aspx

RapidAPI

https://rapidapi.com/hub

Zillow. (2017). Zillow prize: Zillow's home value prediction (Zestimate).

Retrieved from https://www.kaggle.com/c/zillow-prize-1/data Lee Wei En, J. (2019)