Predicting House Prices

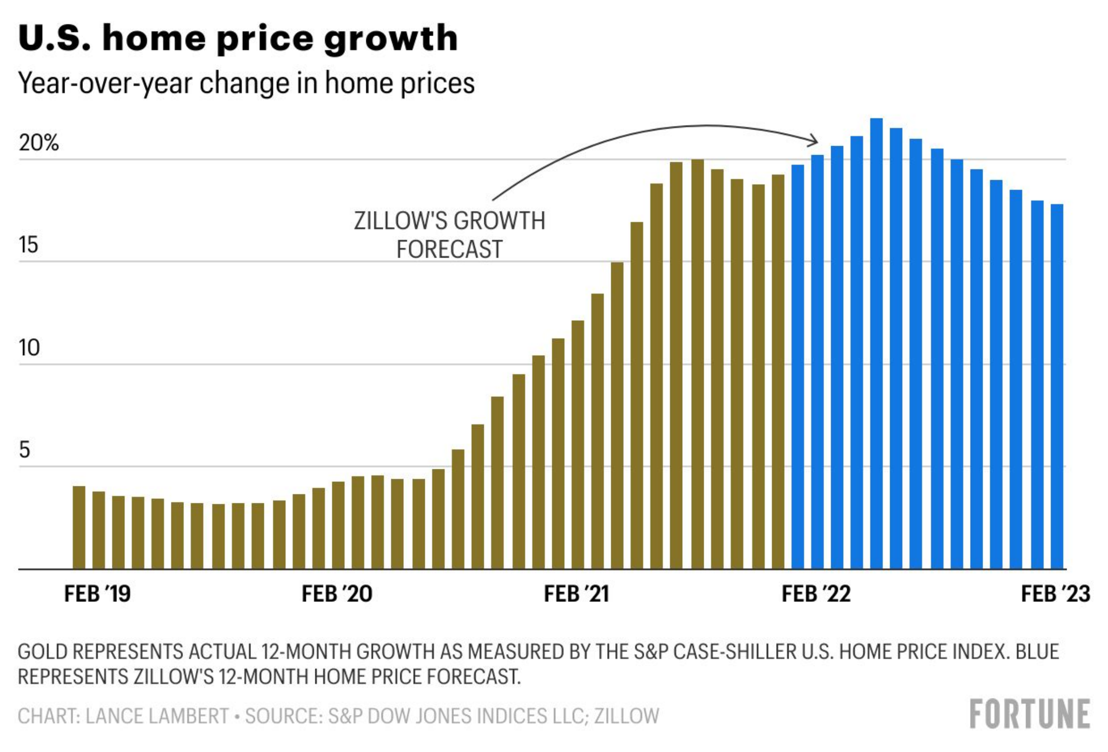
Capstone Two Presentation

Tiffany Kho

House prices

- Always changing
- Reasons
  - Supply and demand, interest rates, recessions, high levels of federal debt, etc
- Stakeholders
  - Homeowner, potential buyer, investor, landlord, real estate agent, broker

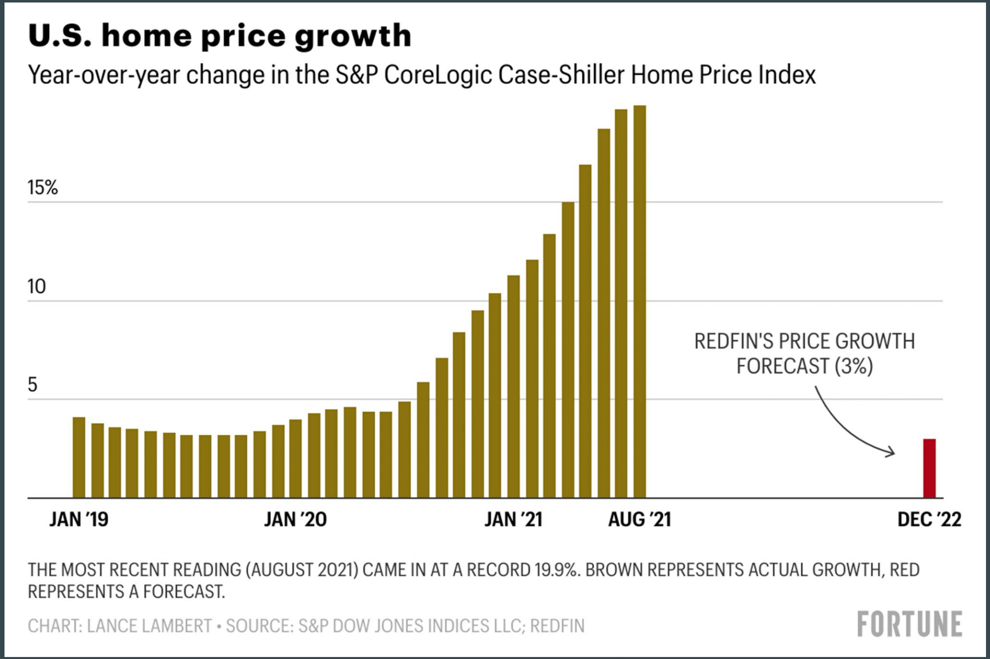
United States housing prices



- Increases over time
- Forecasting
- Fluctuates
  - Economy
  - Pandemic
  - Natural disasters

Project goal

- Goal: Create a model that can accurately predict housing prices



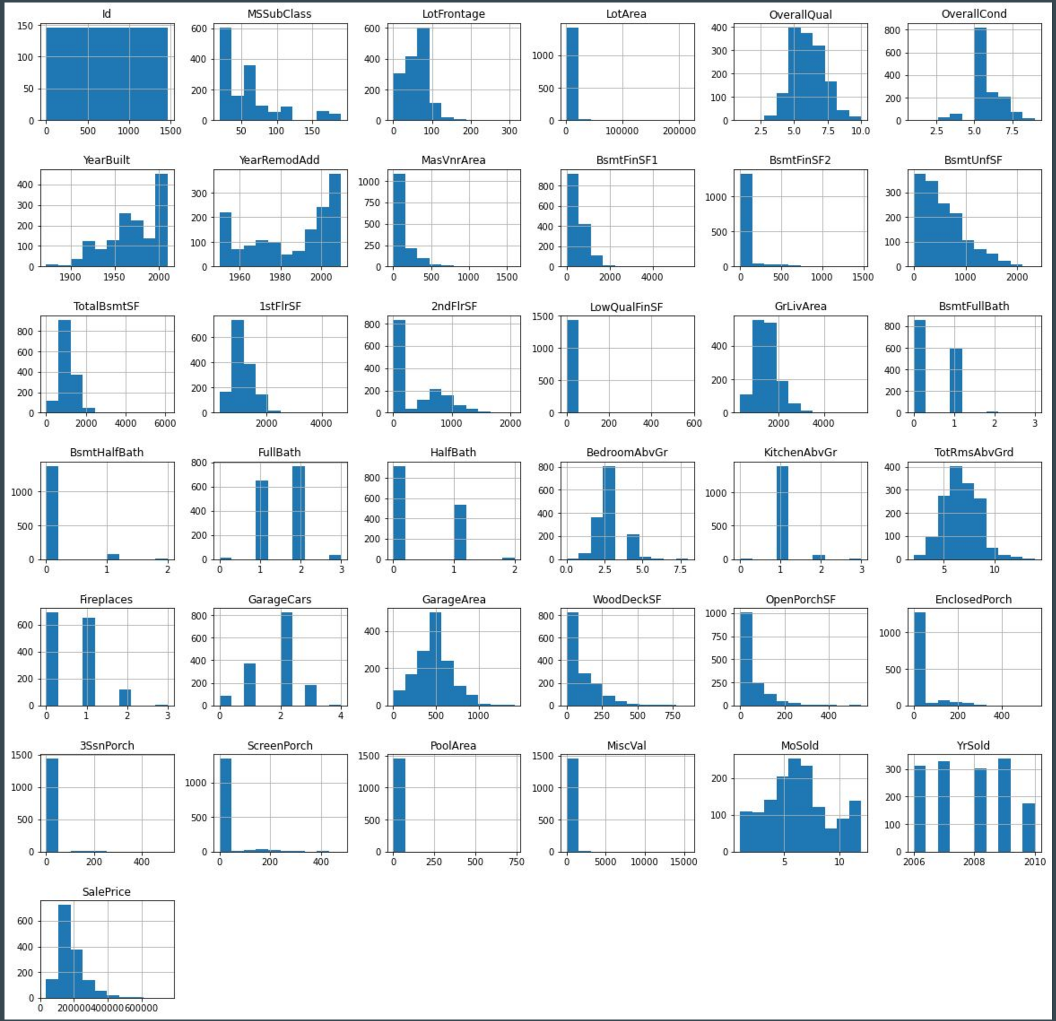
- House amenities
  - Size
  - Upgrades
  - House features
  - Quality of house

Data

- Ames, Iowa house price data set
- 79 explanatory variables
  - 58 categorical
    - House style, lot type, type of electrical system
  - 21 continuous
    - Number of bedrooms, basement size (sq ft), garage size (sq ft)

Data Wrangling

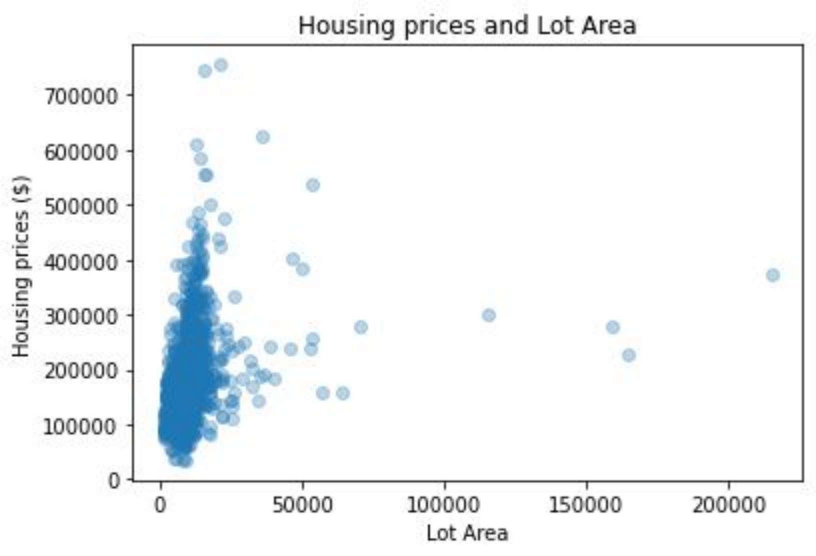
- Categorical: ‘None’
  - Alley, garage
- Continuous: ‘0’
  - Lot area, lot frontage
- Generated histogram



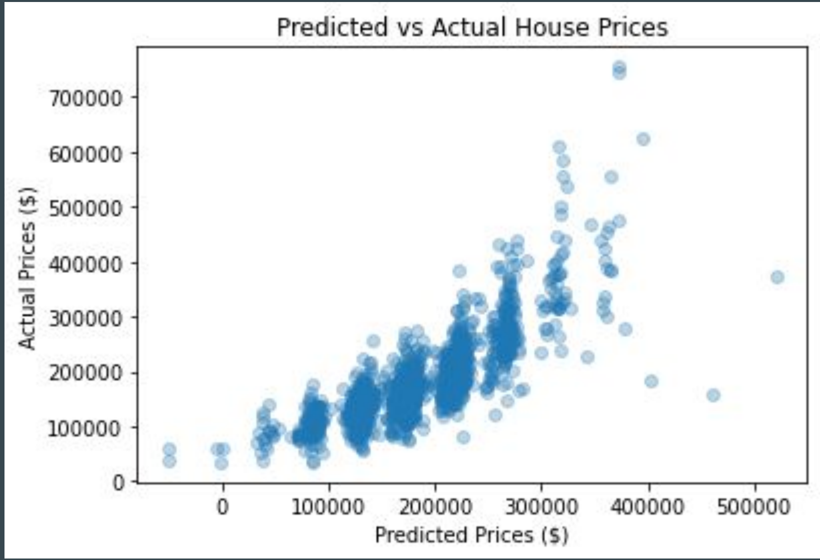
Data Exploration

- Goal
  - Explore relationship between first 4 variables and sale price
  - Linear regression model
- Variables
  - ID
  - Lot area
  - Lot frontage
  - Overall quality

Linear Regression model



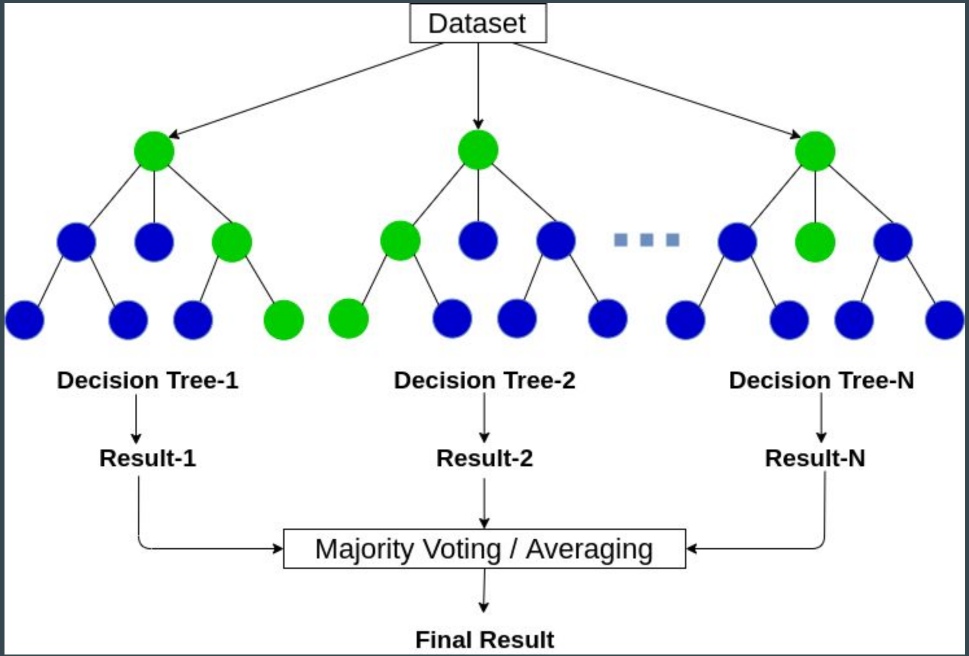
Exploratory data analysis: Actual vs predicted



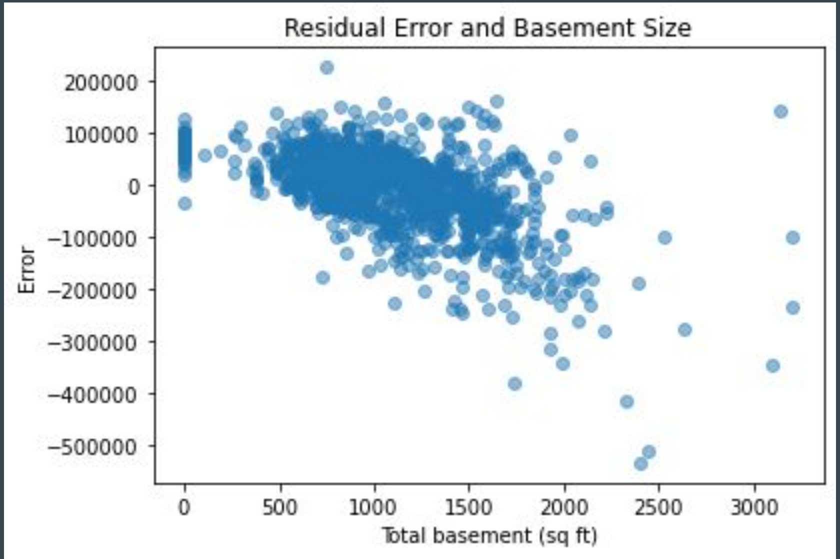
- Hypothesis: Lot area, lot frontage, and overall quality might be good variables at predicting housing prices

Scikit-learn’s models

- Linear Regression
  - Continuous variables
- Random Forest Classifier
  - Continuous and categorical variables
  - Ensemble method, decision trees
- Variables
  - Adjusted thresholds
- Model
  - Tuned hyperparameters



Basement adjusted threshold



- Threshold: < 2,000 sq ft

Hyperparameter tuning: Random Forest Classifier

```
param_dist = {'n_estimators': randint(50,500),
              'max_depth': randint(1,20)}

# Create a random forest classifier
rf = RandomForestClassifier()

# Use random search to find the best hyperparameters
rand_search = RandomizedSearchCV(rf,
                                 param_distributions = param_dist,
                                 n_iter=5,
                                 cv=5)

# Fit the random search object to the data
rand_search.fit(X_train, y_train)

# Create a variable for the best model
best_rf = rand_search.best_estimator_

# Print the best hyperparameters
print('Best hyperparameters:',  rand_search.best_params_)
```

- **n_estimator** (# of trees)
- **max_depth** (# of splits/tree)

Models

	RMSE	RMSEtest	featurenames	model	params
0	76546.713815	79919.058600	LotArea	LinearRegression	NaN
1	73569.131484	83162.654656	LotArea,LotArea_squared	LinearRegression	NaN
2	73139.716444	84373.200915	LotArea,LotArea_squared,LotArea_lt_50000	LinearRegression	NaN
3	72169.128354	85650.352243	LotArea,LotArea_squared,LotArea_lt_50000,LotFrontage	LinearRegression	NaN
4	71470.438467	86435.185894	LotArea,LotArea_squared,LotArea_lt_50000,LotFrontage,LotFrontage_gt_0	LinearRegression	NaN
5	71469.968406	86432.358822	LotArea,LotArea_squared,LotArea_lt_50000,LotFrontage,LotFrontage_gt_0,LotFrontage_lt_150	LinearRegression	NaN
6	20834.005665	111315.356683	LotArea,LotArea_squared,LotArea_lt_50000,LotFrontage,LotFrontage_gt_0,LotFrontage_lt_150	RandomForestClassifier	NaN
7	58099.385054	95527.341760	LotArea,LotArea_squared,LotArea_lt_50000,LotFrontage,LotFrontage_gt_0,LotFrontage_lt_150,TotalBsmtSF	LinearRegression	NaN
8	57641.957486	95437.131745	LotArea,LotArea_squared,LotArea_lt_50000,LotFrontage,LotFrontage_gt_0,LotFrontage_lt_150,TotalBsmtSF,TotalBsmtSF_lt_2000	LinearRegression	NaN
9	51832.535131	98242.764980	LotArea,LotArea_squared,LotArea_lt_50000,LotFrontage,LotFrontage_gt_0,LotFrontage_lt_150,TotalBsmtSF,TotalBsmtSF_lt_2000,YearBuilt	LinearRegression	NaN
10	48741.063518	100165.073309	LotArea,LotArea_squared,LotArea_lt_50000,LotFrontage,LotFrontage_gt_0,LotFrontage_lt_150,TotalBsmtSF,TotalBsmtSF_lt_2000,YearBuilt,YearBuilt_lt_1990	LinearRegression	NaN
11	39450.203717	105058.330597	LotArea,LotArea_squared,LotArea_lt_50000,LotFrontage,LotFrontage_gt_0,LotFrontage_lt_150,TotalBsmtSF,TotalBsmtSF_lt_2000,YearBuilt,YearBuilt_lt_1990,OverallQual	LinearRegression	NaN
12	37801.397783	106568.816990	LotArea,LotArea_squared,LotArea_lt_50000,LotFrontage,LotFrontage_gt_0,LotFrontage_lt_150,TotalBsmtSF,TotalBsmtSF_lt_2000,YearBuilt,YearBuilt_lt_1990,OverallQual,OverallQual_lt_9	LinearRegression	NaN
13	1364.061580	109548.059839	LotArea,LotArea_squared,LotArea_lt_50000,LotFrontage,LotFrontage_gt_0,LotFrontage_lt_150,TotalBsmtSF,TotalBsmtSF_lt_2000,YearBuilt,YearBuilt_lt_1990,OverallQual,OverallQual_lt_9	RandomForestClassifier	(n_estimators=276, max_depth=14)

Best model: Random Forest Classifier (n_estimators = 276, max_depth = 14)

Conclusions and Future work

- Train error ~$1,600
- Test error > $100,000
- Risk of undervaluing houses
- Large potential loss
- Future work:
    - Reduce bias variance problem
    - Add thresholds
    - More/different data
    - Different models