Ames Housing Analysis

Presented by: Rob Wygant

Date: June 16th, 2023

Overview

- Scope
- Workflow
- Dataset/EDA
- Modeling
- Selection
- Future work

Scope

- **Problem:** Ames, Iowa housing market needs a resource to better understand local home prices and their associated features
- **Project goal:** explore feature importance in local home sale records and develop a price prediction machine learning model

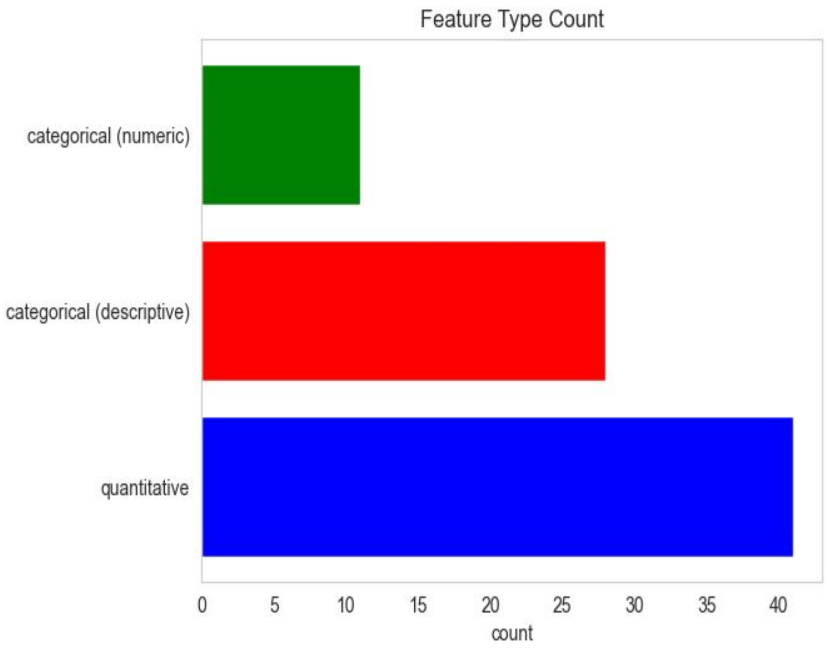
Location: central Iowa
Population: 67,000
Growth: 7% (past 10 years)

Deliverables

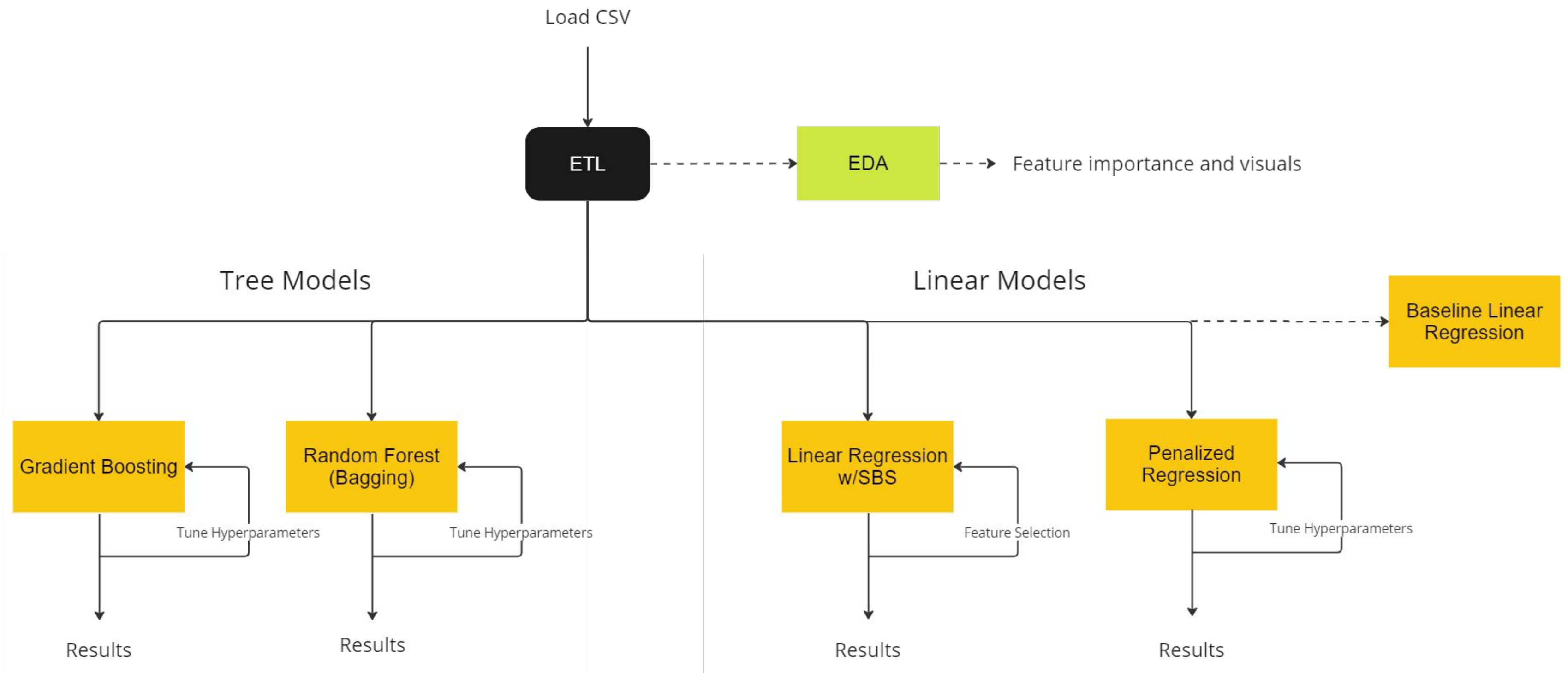
1) Exploratory data insights w/visuals
2) Descriptive: feature importance ranking
3) Predictive: production ready ML model pipeline

Dataset

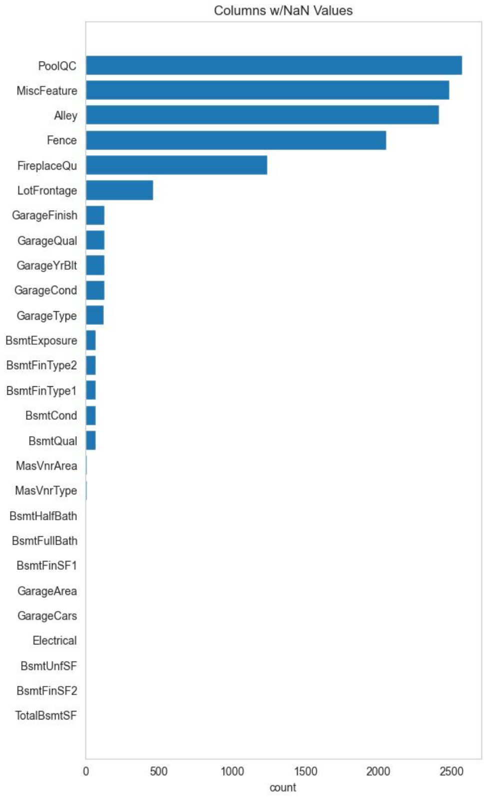
- **Data set:**
  - Sourced from Kaggle competition
  - 2500 home sale price records
  - 80 features associated with a single home price characteristics and condition prior to sale
    - 11 categorical (numeric)
    - 28 categorical (descriptive)
    - 40 quantitative



Workflow


Load CSV
ETL
EDA
Feature importance and visuals
Tree Models
Linear Models
Baseline Linear Regression
Gradient Boosting
Random Forest (Bagging)
Linear Regression w/SBS
Penalized Regression
Tune Hyperparameters
Tune Hyperparameters
Feature Selection
Tune Hyperparameters
Results
Results
Results
Results

ETL

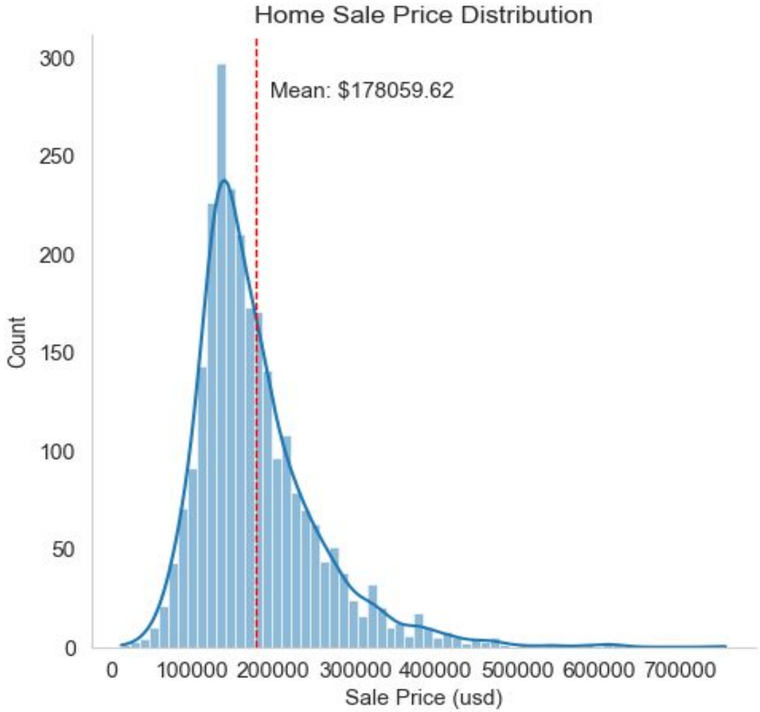


- Load original data: `Ames_Housing_Price_Data.csv`
- 28 columns with missing values
  - Note: reference data dictionary for imputation method
- Encoded categorical nominal features
  - Note: dummy encoded took place in model selection
- Separated features and target variable ‘SalePrice’
- Saved 3 csv files for EDA and modeling
  - Dataframe: `housing_df.csv`
  - Feature set: `housing_X_features.csv`
  - Target variable: `housing_y_target.csv`

EDA (target variable)

- Target variable ‘SalePrice’
  - dtype: int64

	SalePrice
count	2580.000000
mean	178059.623256
std	75031.089374
min	12789.000000
25%	129975.000000
50%	159900.000000
75%	209625.000000
max	755000.000000

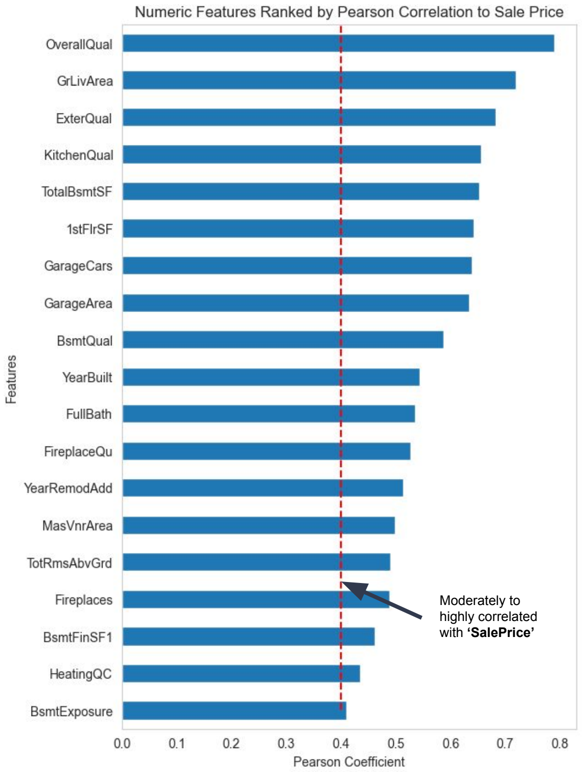


EDA (numeric features)

- Pearson correlation to 'SalePrice'
- 19 numeric features w/ correlation greater than 0.4 to 'SalePrice'

Top 5 Numeric Features:

1) **Overall Quality**
2) **Living Area Size**
3) **Exterior Quality**
4) **Kitchen Quality**
5) **Basement Size**

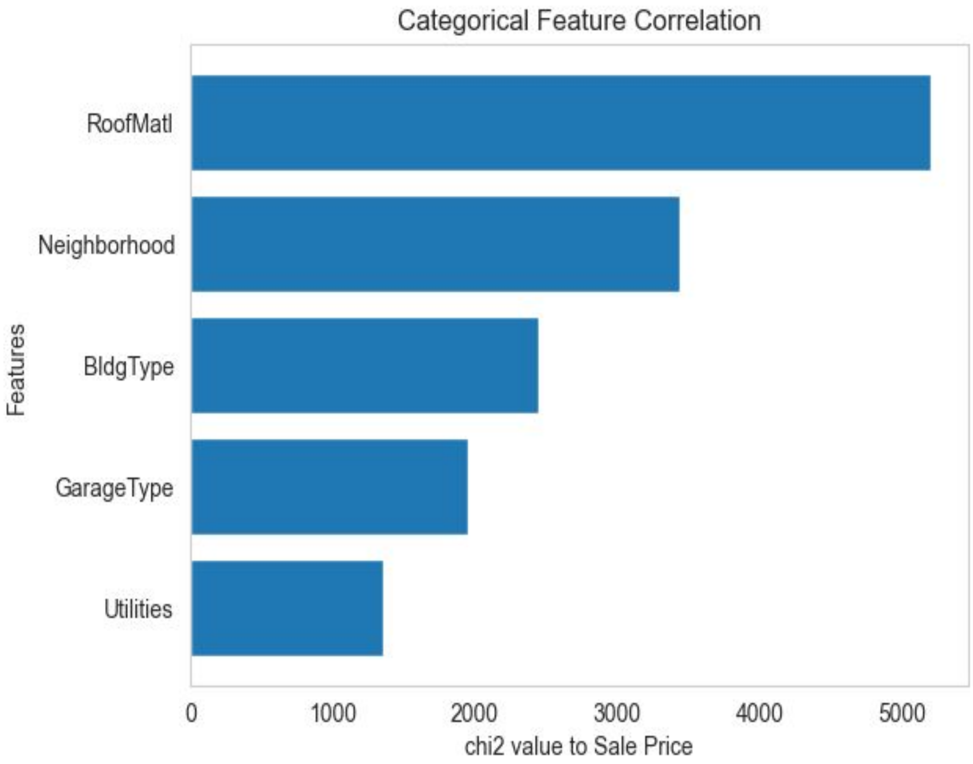


EDA (categorical features)

- Top categorical features related to Sale Price
- Ranked by chi squared value

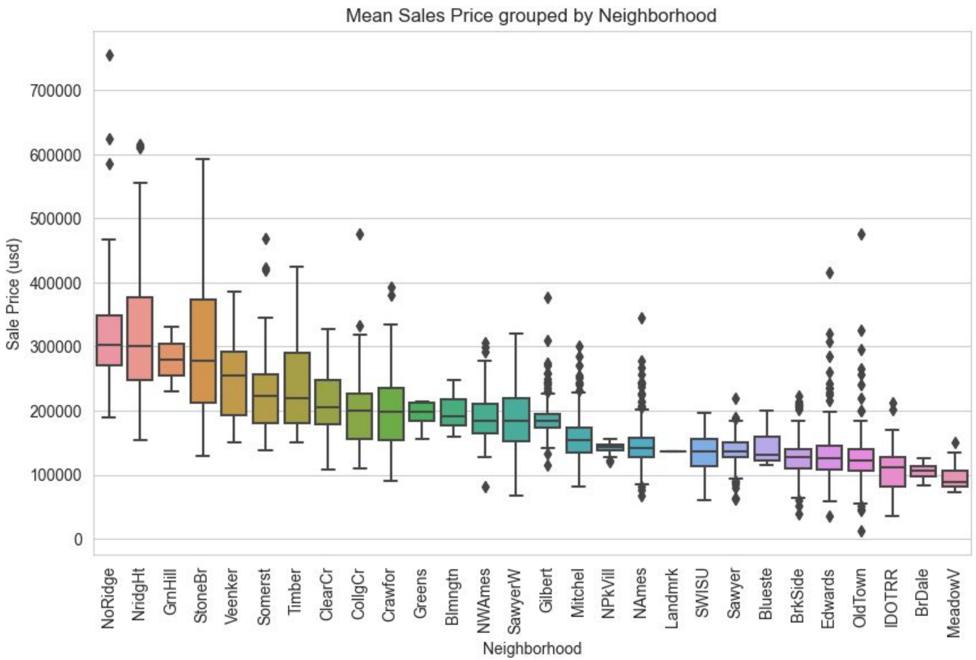
Top 5 Categorical Features:

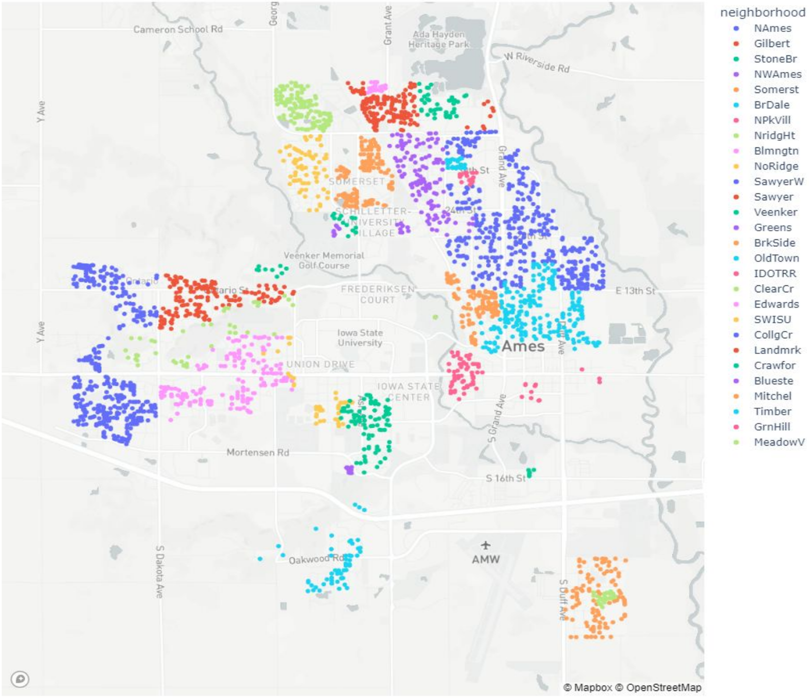
1) **Roof Material**
2) **Neighborhood**
3) **Building Type**
4) **Garage Type**
5) **Utilities**



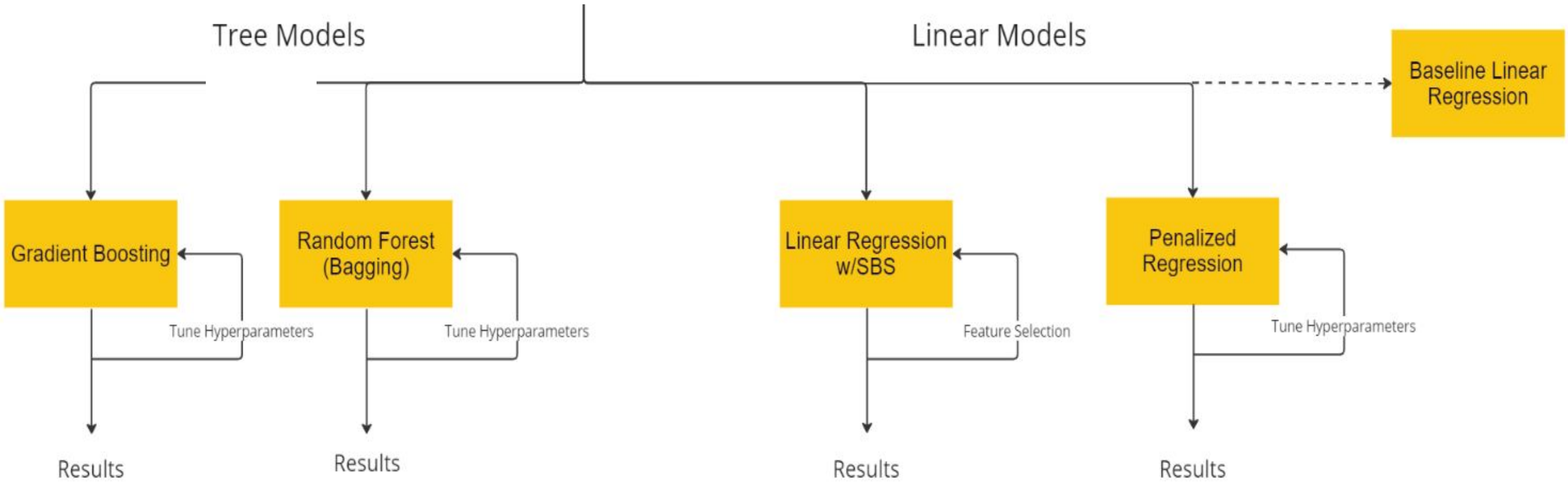
EDA (neighborhood analysis)

- Neighborhood type ranked by mean sale price





ML Modeling

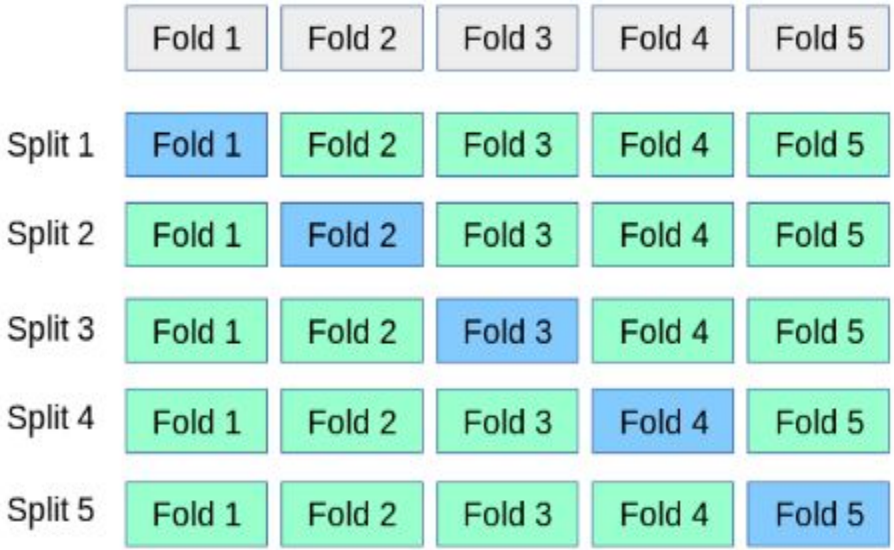


Tree Models
Linear Models
Baseline Linear Regression
Gradient Boosting
Random Forest (Bagging)
Linear Regression w/SBS
Penalized Regression
Tune Hyperparameters
Tune Hyperparameters
Feature Selection
Tune Hyperparameters
Results
Results
Results
Results

Model Selection

- Cross validation for model selection bias
  - 5 fold
  - Random state = 12
- Mean R^2 value for model selection
- Mean Absolute Error (MAE) on complete data set for final evaluation

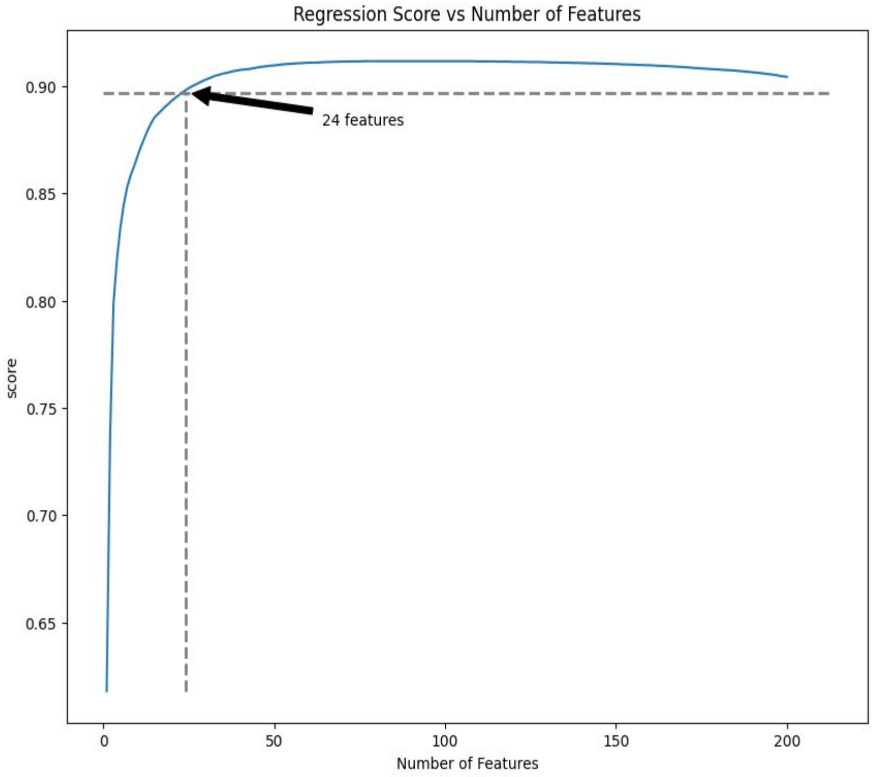
```python
# cross validation on train set
lr_base = LinearRegression()
k=5
cv = KFold(n_splits=k, shuffle=True, random_state=12)
cv_results = cross_validate(lr_base, X, y, cv=cv, return_train_score=True)
cv_pred = cross_val_predict(lr_base, X, y, cv=cv)

for test_score in cv_results['test_score']:
    print(test_score)

print('Mean Score (r^2)=' + str(cv_results['test_score'].mean()))
print('Mean Absolute Error (cross validation):' + str(mean_absolute_error(y, cv_pred)))
```



Linear Regression w/SBS

- Sequential backwards selection: sequentially removing features in reverse from original set by keeping best scoring set at each step
- Selection with cross validated r^2 value on linear regression model
- **24 features selected**

Metrics	Linear Regression w/SBS
MAE	15905
Score (r^2)	0.89
Number of Features	24
CPU time	< 1s



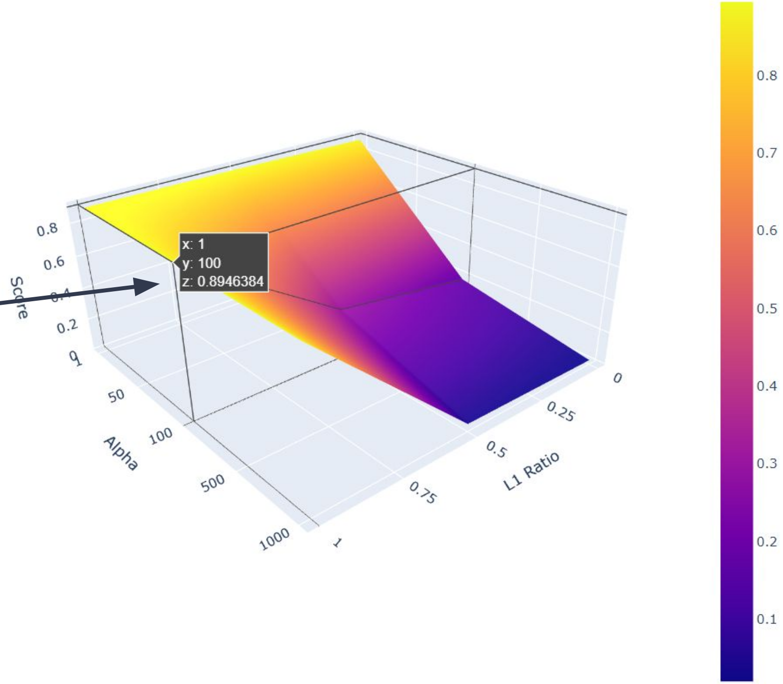
Penalized (Elastic Net)

- **Coarse grid search w/ElasticNet**
- Pipeline utilizing StandardScaler()

Alpha

1	0.890	0.888	0.875
100	0.894	0.304	0.179
1000	0.893	0.039	0.019
	1	**0.5**	**0**

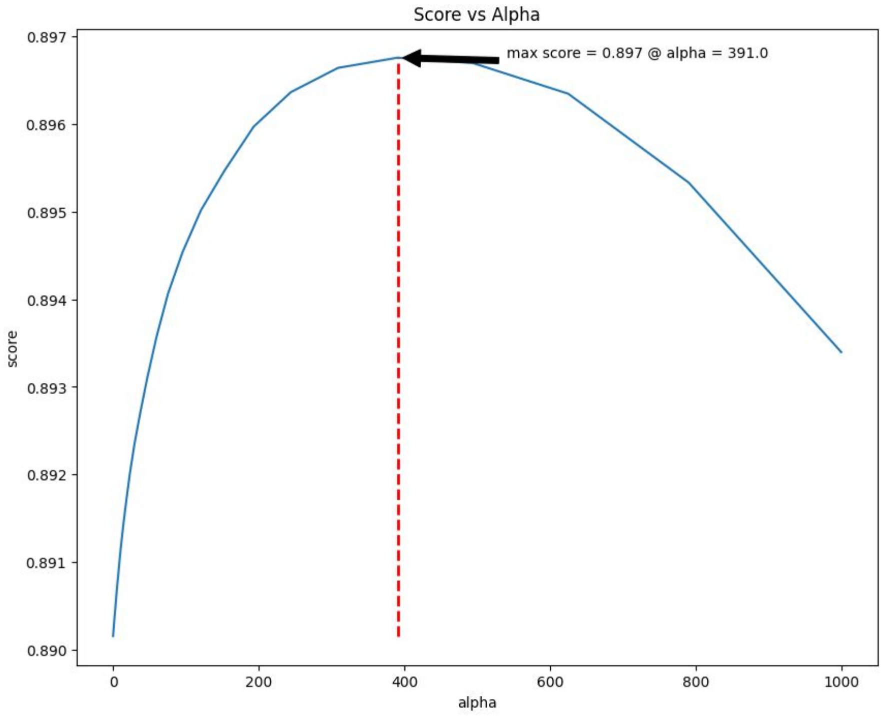
L1 Ratio



Penalized (Lasso)

- **Fine search w/Lasso**
- Pipeline utilizing StandardScaler()
- 50 values of Alphas (0.001 to 1000)
- Optimal alpha=391

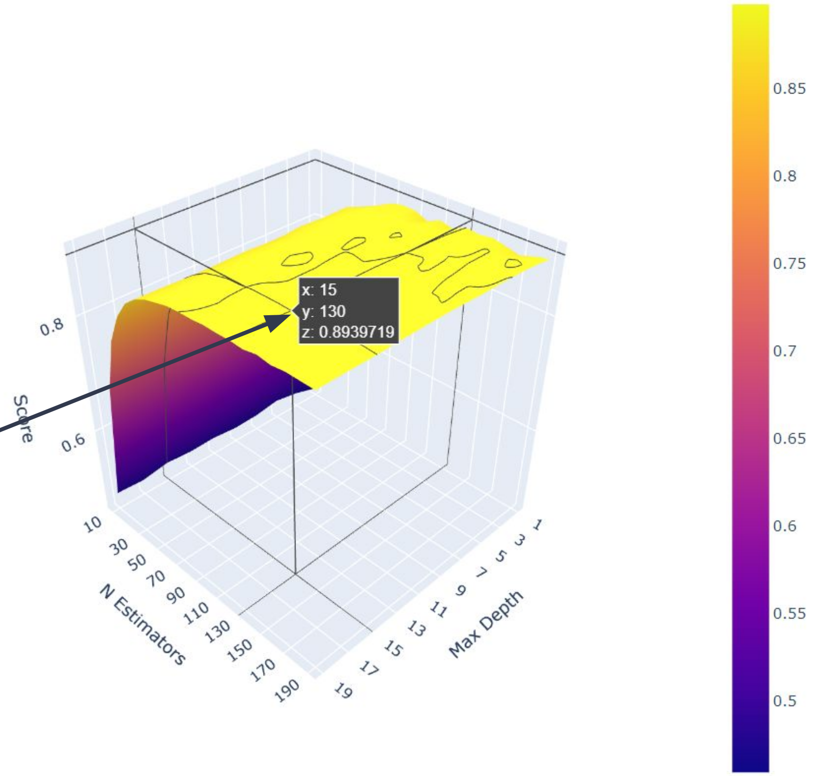
	Lasso
MAE	14102
Score (r^2)	0.9
Number of Features	213
CPU time	1min 3s



Random Forest (bagging)

- Random Forest Regressor w/bagging
- Grid search for max depth and number of trees
- Optimal parameters
  - Depth: 15
  - Number of trees: 130

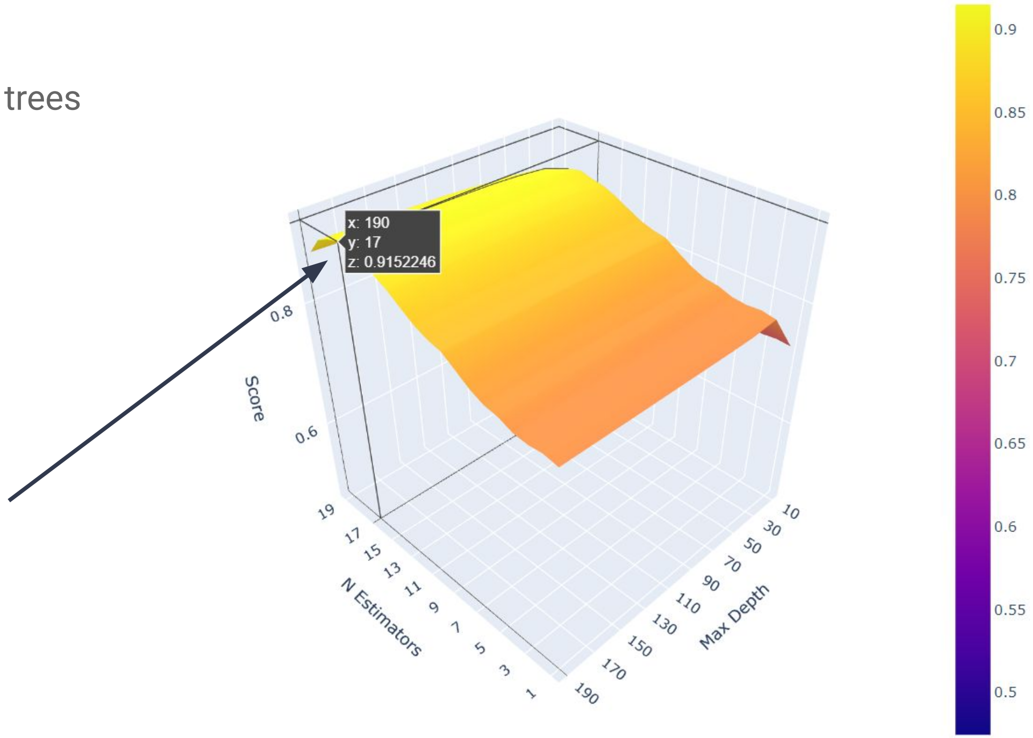
	Random Forest
MAE	5662
Score (r^2)	0.9
Number of Features	213
CPU time	39min 26s



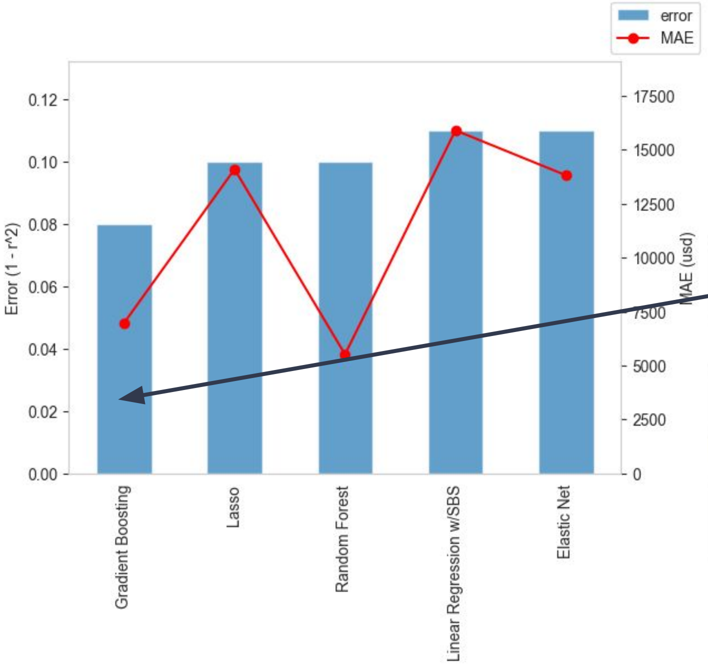
Gradient Boosting

- Gradient Boosting
- Grid search for max depth and number of trees
- Optimal parameters
  - Depth: 4
  - Number of trees: 190

	Gradient Boosting
MAE	6983
Score (r^2)	0.92
Number of Features	213
CPU time	58min 11s

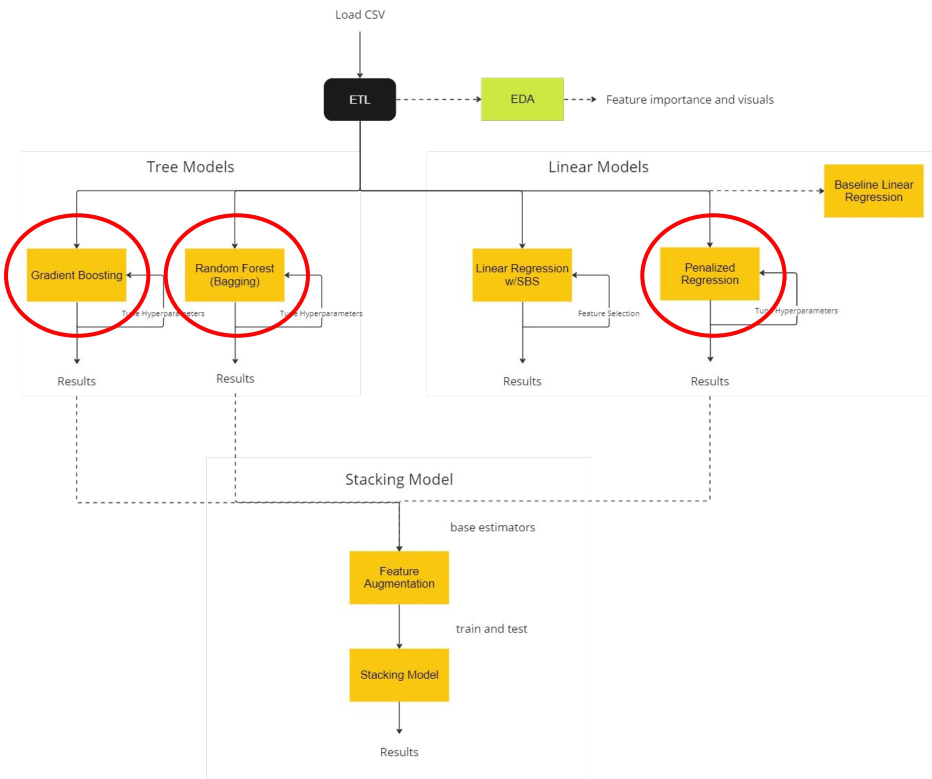


Modeling Summary



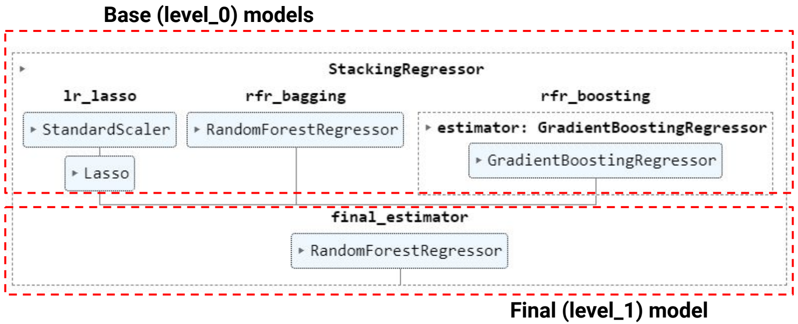
Model	Score (r^2)	MAE	Number of Features	CPU time
Gradient Boosting	0.92	6983	79	1h 28min 54s
Lasso	0.90	14102	213	1min 3s
Random Forest	0.90	5522	79	1h 9min 52s
Linear Regression w/SBS	0.89	15905	24	< 1s
Elastic Net	0.89	13824	213	12s

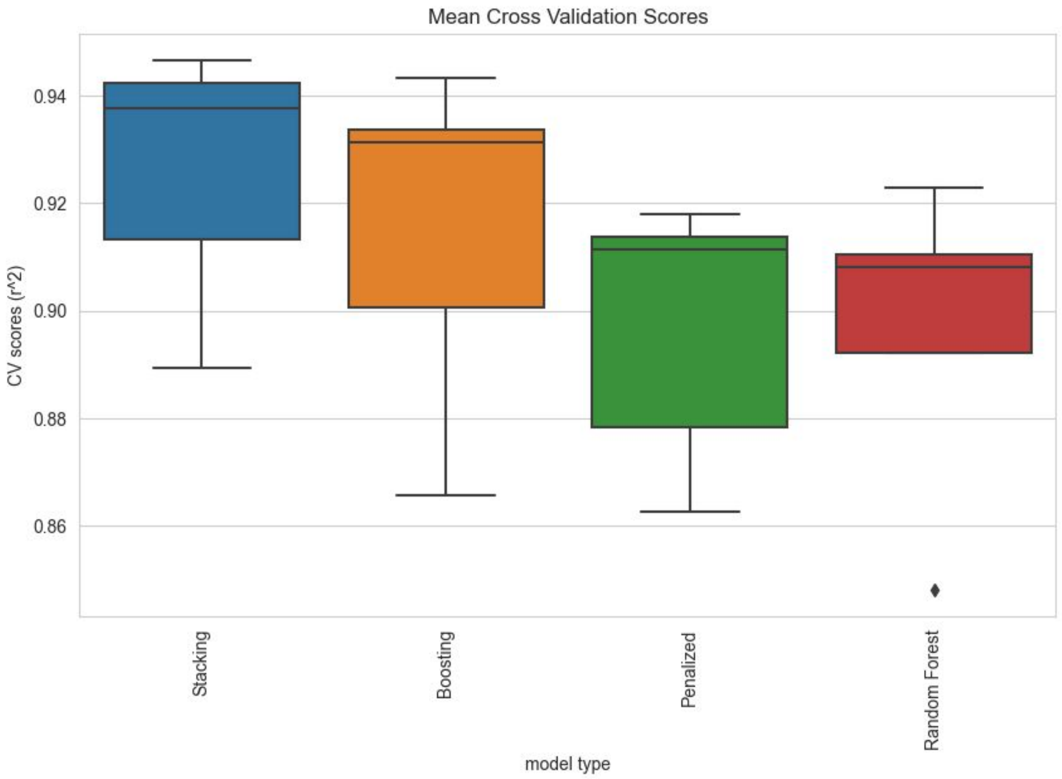
Stacking Model



Stacking Model

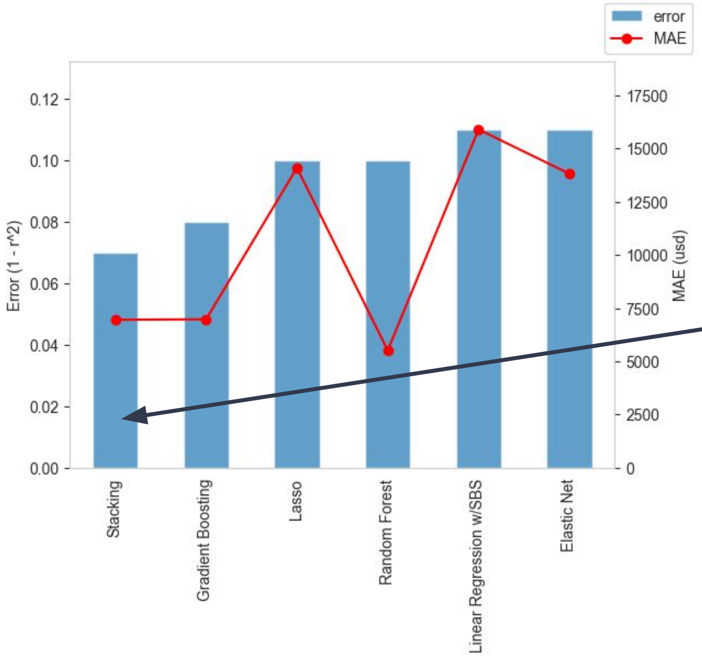
- Gradient Boosting
- Grid search for max depth and number of
- Optimal parameters
    - Depth: 4
    - Number of trees: 190





Selection Summary w/stacking comparison

- Stacking model out performs all models



Model	Score (r^2)	MAE	Number of Features	CPU time
Stacking	0.93	6958	79	2min 28s
Gradient Boosting	0.92	6983	79	55min 14s
Lasso	0.90	14102	213	1min 3s
Random Forest	0.90	5522	79	36min 53s
Linear Regression w/SBS	0.89	15905	24	< 1s
Elastic Net	0.89	13824	213	12s

Future Work

- XGBoost and other modeling
- Flask app for client side home predictions
- Explore classification modeling

Questions?