ISYE 6740 Final Project: Modelling and Analysis of Brooklyn Home Prices

Ralph Andraos
Karthik Annigeri
Ojas Majgaonkar
Ani Marellapudi

ISYE 6740, Spring 2024, Team Epsilon
Professor Tuo Zhao

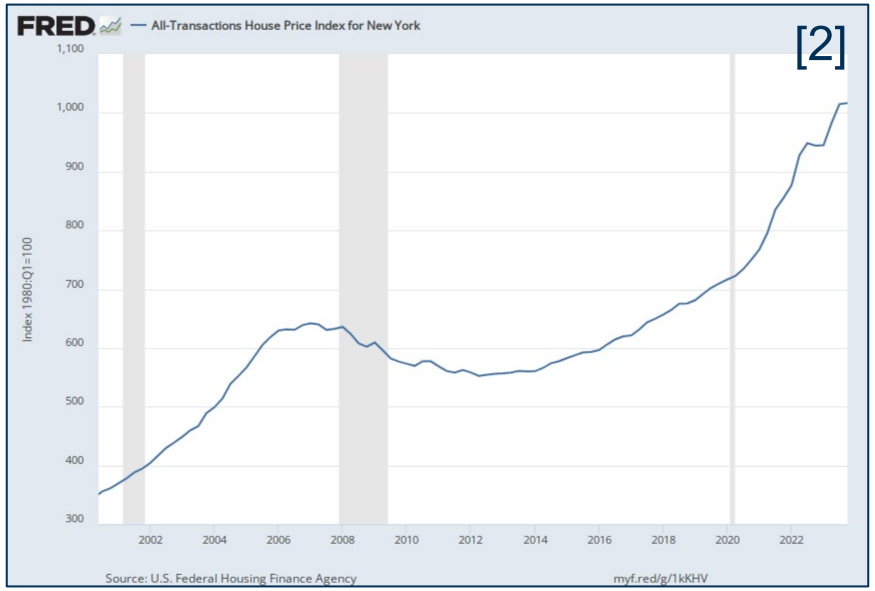
Georgia Tech

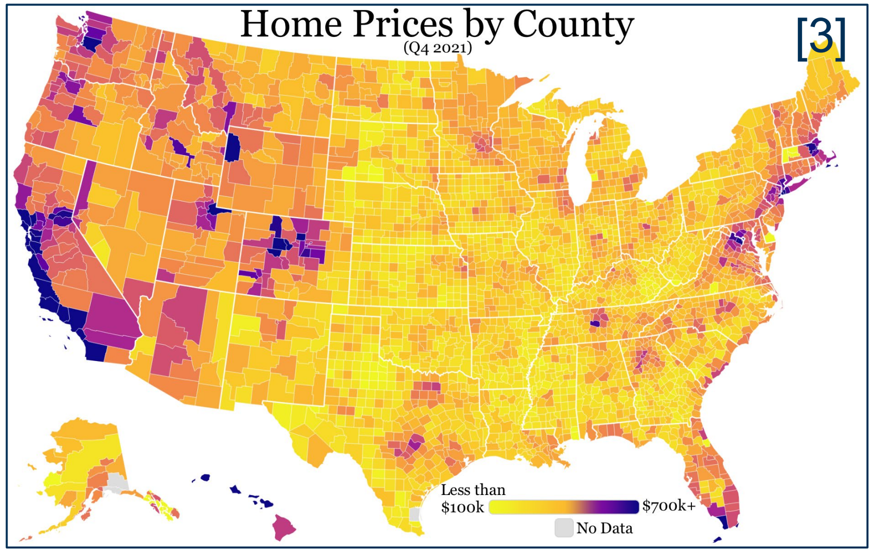
Agenda

1. Introduction

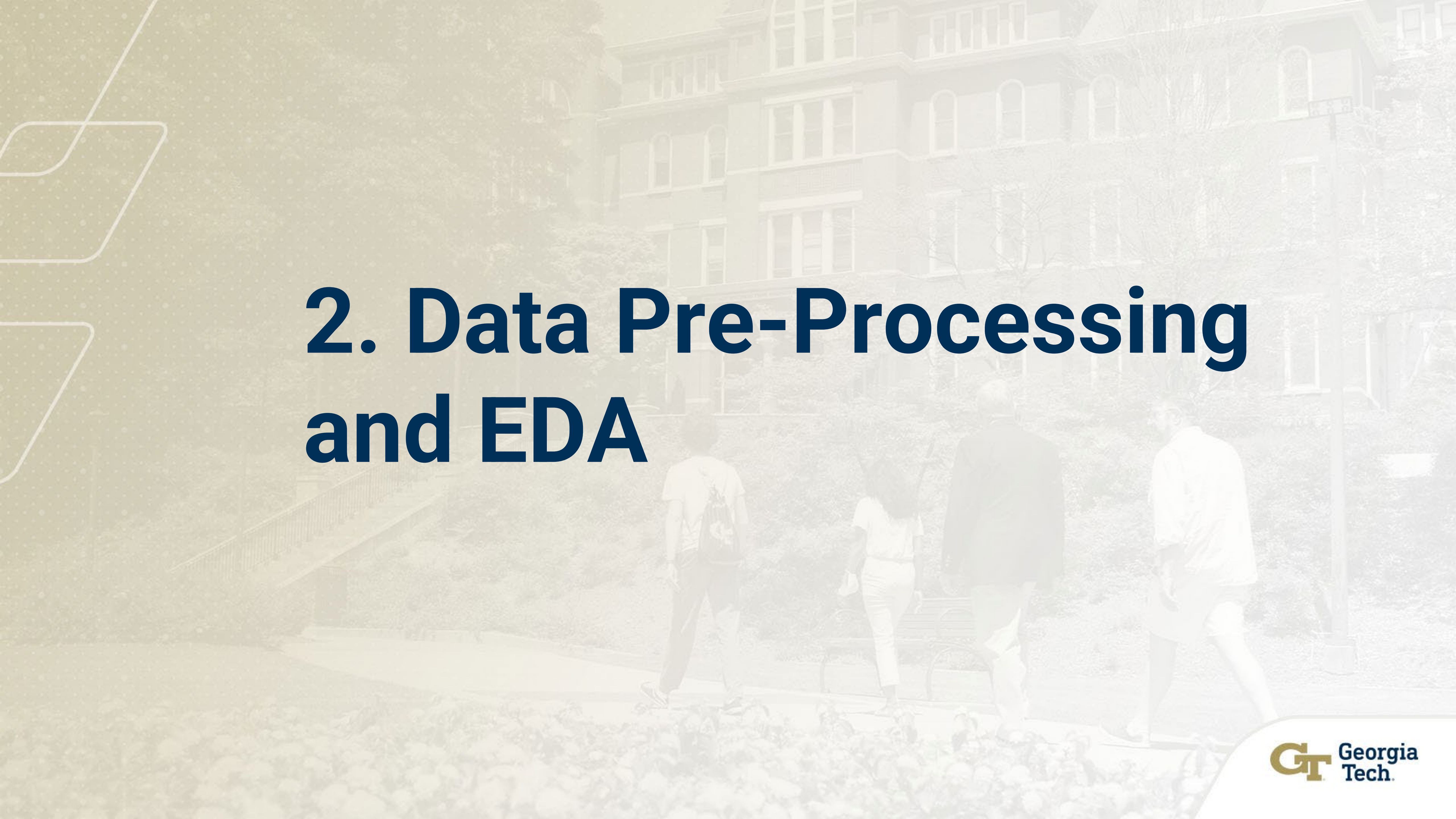
Objectives and Motivation

- Home prices are complex, depending on home characteristics, geography, neighborhood, and more
- Home prices impact individuals and businesses – purchasing a home is the single largest investment many people make[1]
- Accurate prediction of their value can lead to optimized investment decisions
- This project aims to:
  - **Train and compare robust models to predict home prices in Brooklyn, using >40 numerical variables and >15 categorical variables**





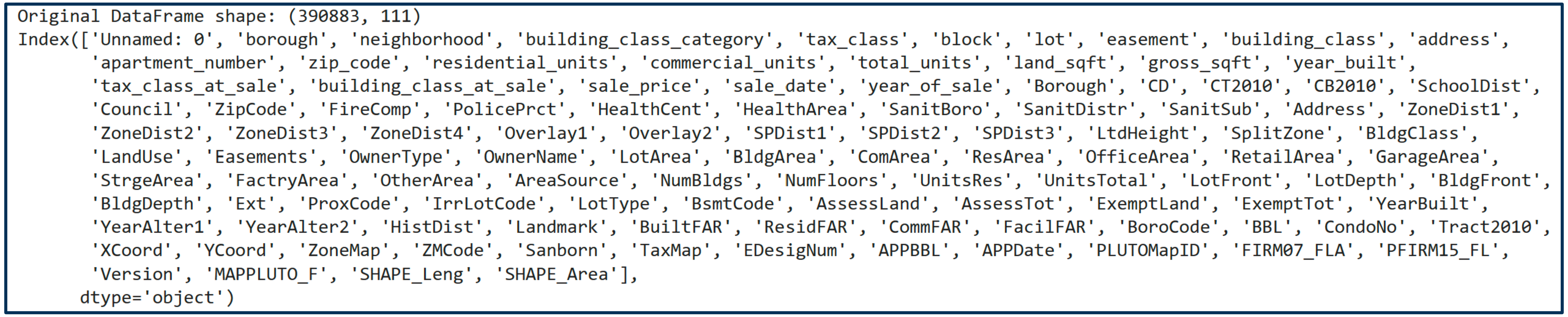
[1] https://www.huduser.gov/publications/pdf/wealthaccumulationandhomeownership.pdf
[2] https://fred.stlouisfed.org/series/NYSTHPI#
[3] https://en.wikipedia.org/wiki/List_of_U.S._states_by_median_home_price#/media/File:Home_prices_by_county.webp



2. Data Pre-Processing and EDA

Dataset Overview and Source

- **Data Set Description:**
  - Dataset includes home sale prices in Brooklyn, NY between 2003 and 2017
  - Kaggle dataset link: (https://www.kaggle.com/datasets/tianhwu/brooklynhomes2003to2017)
  - All original data provided by NYC.gov: (http://www1.nyc.gov/site/finance/taxes/property-rolling-sales-data.page)
- **Data Set Size + Feature Composition:**
  - The data consisted of >390,000 data points across 111 diverse features.
  - Raw dataframe shape: (390883, 111), 197Mb
  - 111 features included a mix of numerical and categorical data, providing comprehensive coverage of factors affecting house prices.

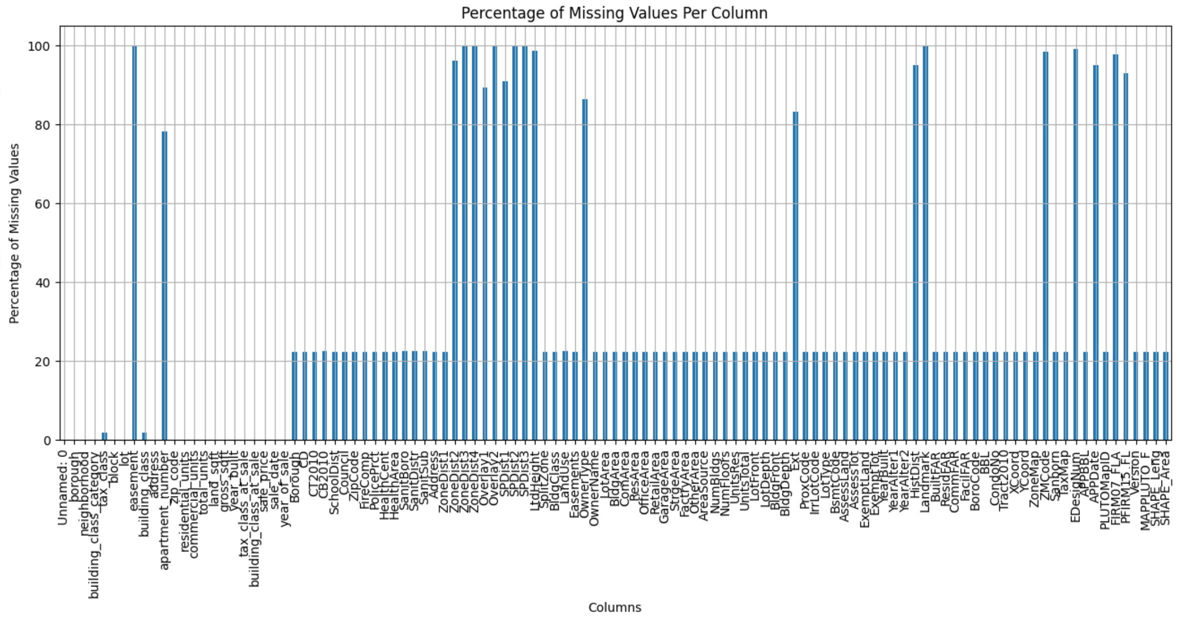


```
Original DataFrame shape: (390883, 111)
Index(['Unnamed: 0', 'borough', 'neighborhood', 'building_class_category', 'tax_class', 'block', 'lot', 'easement', 'building_class', 'address',
       'apartment_number', 'zip_code', 'residential_units', 'commercial_units', 'total_units', 'land_sqft', 'gross_sqft', 'year_built',
       'tax_class_at_sale', 'building_class_at_sale', 'sale_price', 'sale_date', 'year_of_sale', 'Borough', 'CD', 'CT2010', 'CB2010', 'SchoolDist',
       'Council', 'ZipCode', 'FireComp', 'PolicePrct', 'HealthCent', 'HealthArea', 'SanitBoro', 'SanitDistr', 'SanitSub', 'Address', 'ZoneDist1',
       'ZoneDist2', 'ZoneDist3', 'ZoneDist4', 'Overlay1', 'Overlay2', 'SPDist1', 'SPDist2', 'SPDist3', 'LtdHeight', 'SplitZone', 'BldgClass',
       'LandUse', 'Easements', 'OwnerType', 'OwnerName', 'LotArea', 'BldgArea', 'ComArea', 'ResArea', 'OfficeArea', 'RetailArea', 'GarageArea',
       'StrgeArea', 'FactryArea', 'OtherArea', 'AreaSource', 'NumBldgs', 'NumFloors', 'UnitsRes', 'UnitsTotal', 'LotFront', 'LotDepth', 'BldgFront',
       'BldgDepth', 'Ext', 'ProxCode', 'IrrLotCode', 'LotType', 'BsmtCode', 'AssessLand', 'AssessTot', 'ExemptLand', 'ExemptTot', 'YearBuilt',
       'YearAlter1', 'YearAlter2', 'HistDist', 'Landmark', 'BuiltFAR', 'ResidFAR', 'CommFAR', 'FacilFAR', 'BoroCode', 'BBL', 'CondoNo', 'Tract2010',
       'XCoord', 'YCoord', 'ZoneMap', 'ZMCode', 'Sanborn', 'TaxMap', 'EDesigNum', 'APPBBL', 'APPDate', 'PLUTOMapID', 'FIRM07_FLA', 'PFIRM15_FL',
       'Version', 'MAPPLUTO_F', 'SHAPE_Leng', 'SHAPE_Area'],
      dtype='object')
```

Eliminate Redundant & Non-Impactful Variables

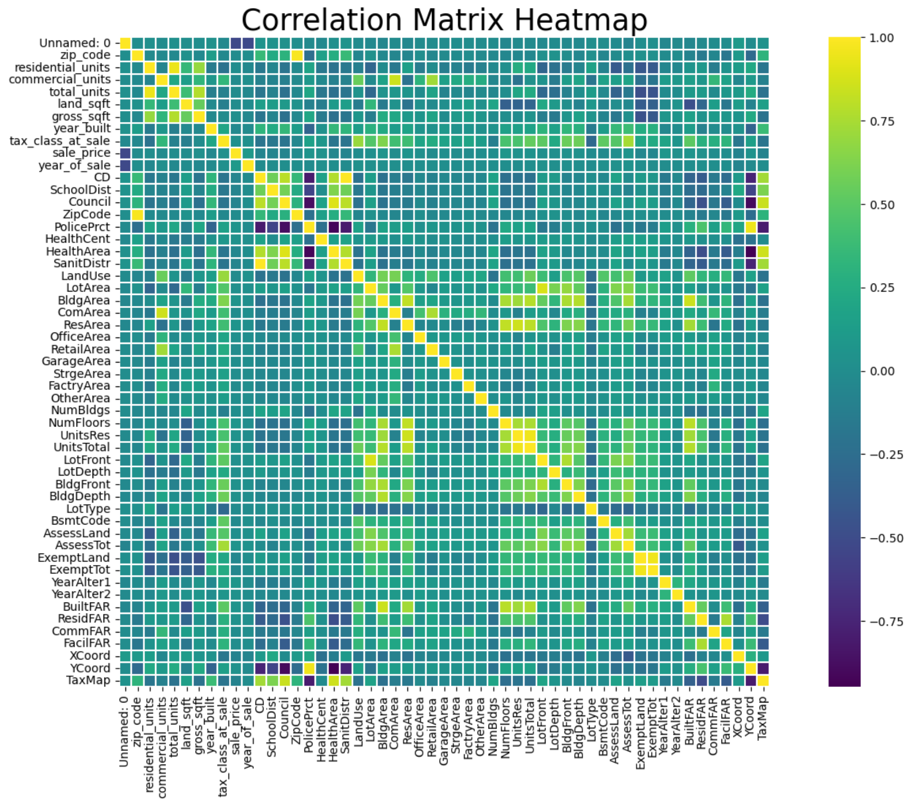
Remove columns with >30% missing values, remove rows with missing values

Manually remove duplicate/unnecessary columns (relating to dataset version, etc)



(390883, 111) -> (301220, 60)

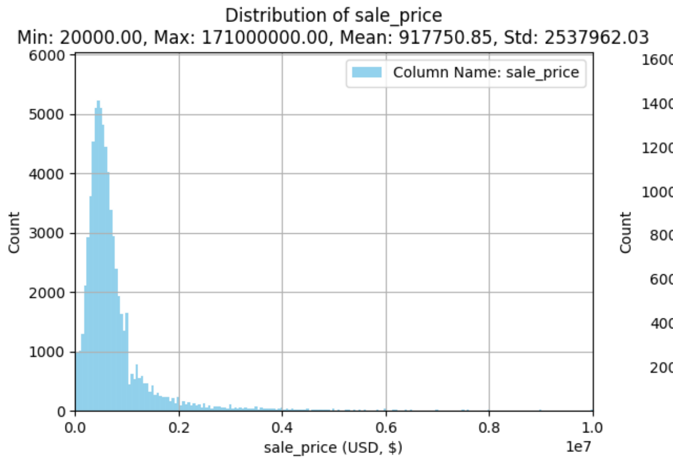
Eliminate variables with >90% correlation to other variables

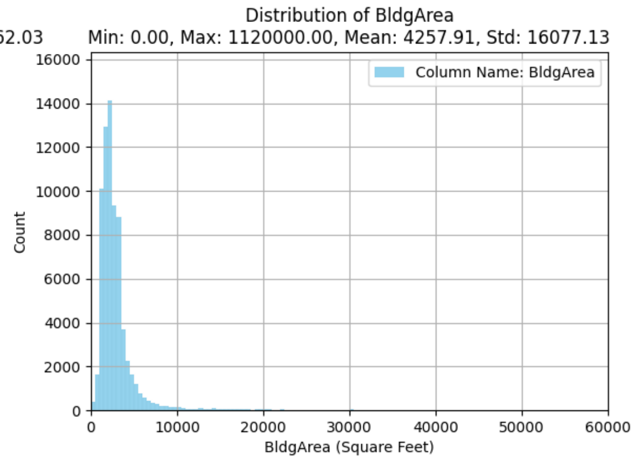


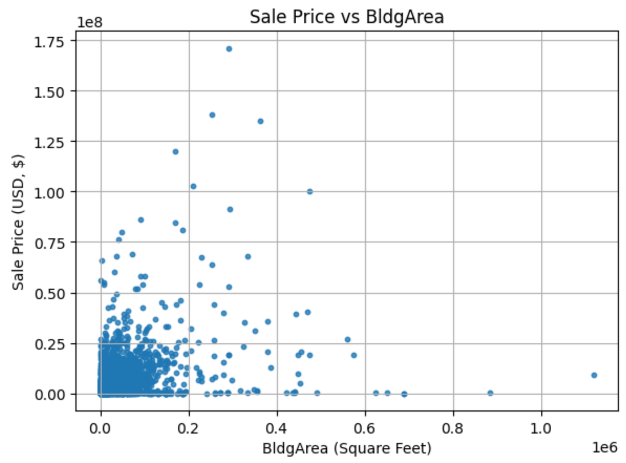
(301220, 60) -> (301220, 54)

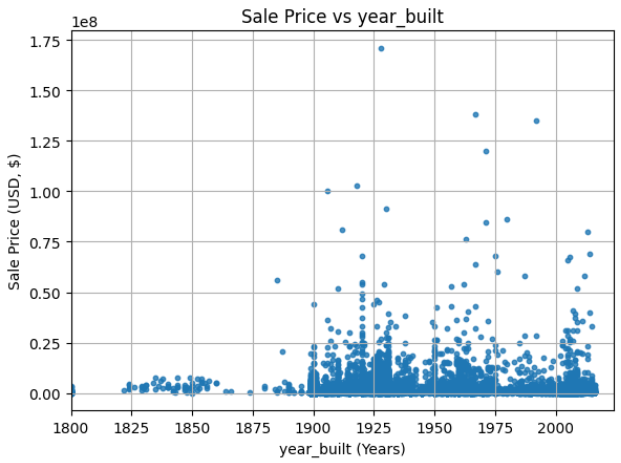
Georgia Tech

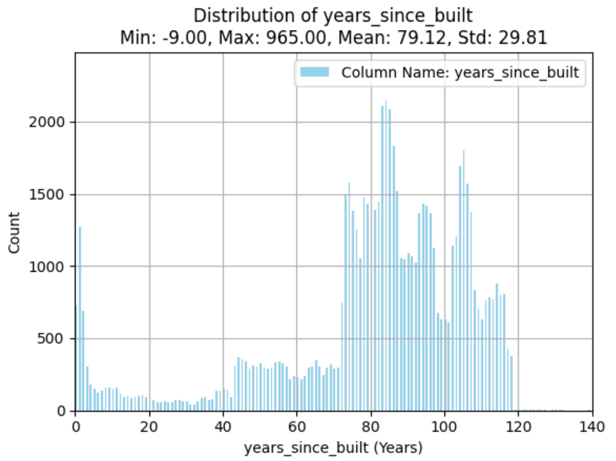
Preliminary Data Visualization

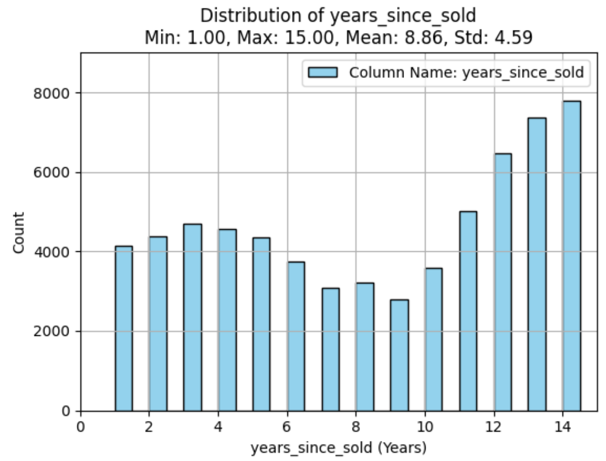


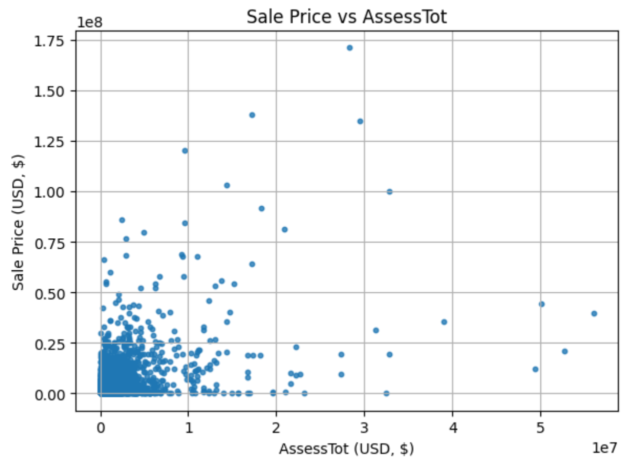


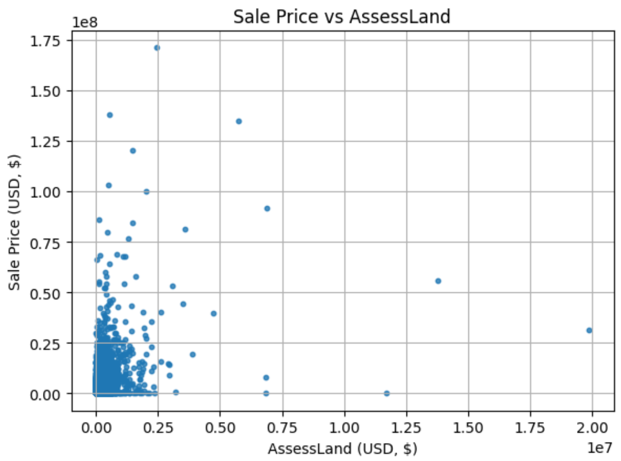






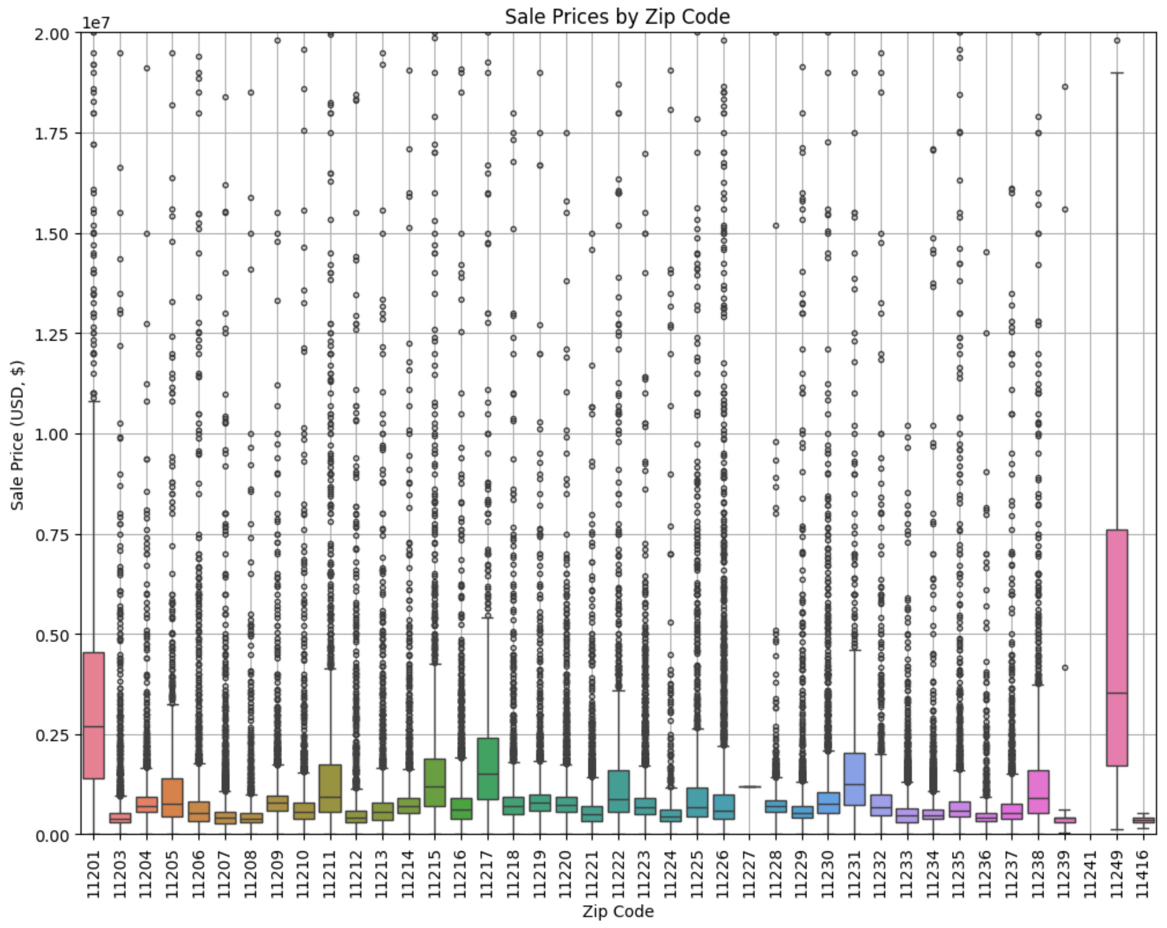


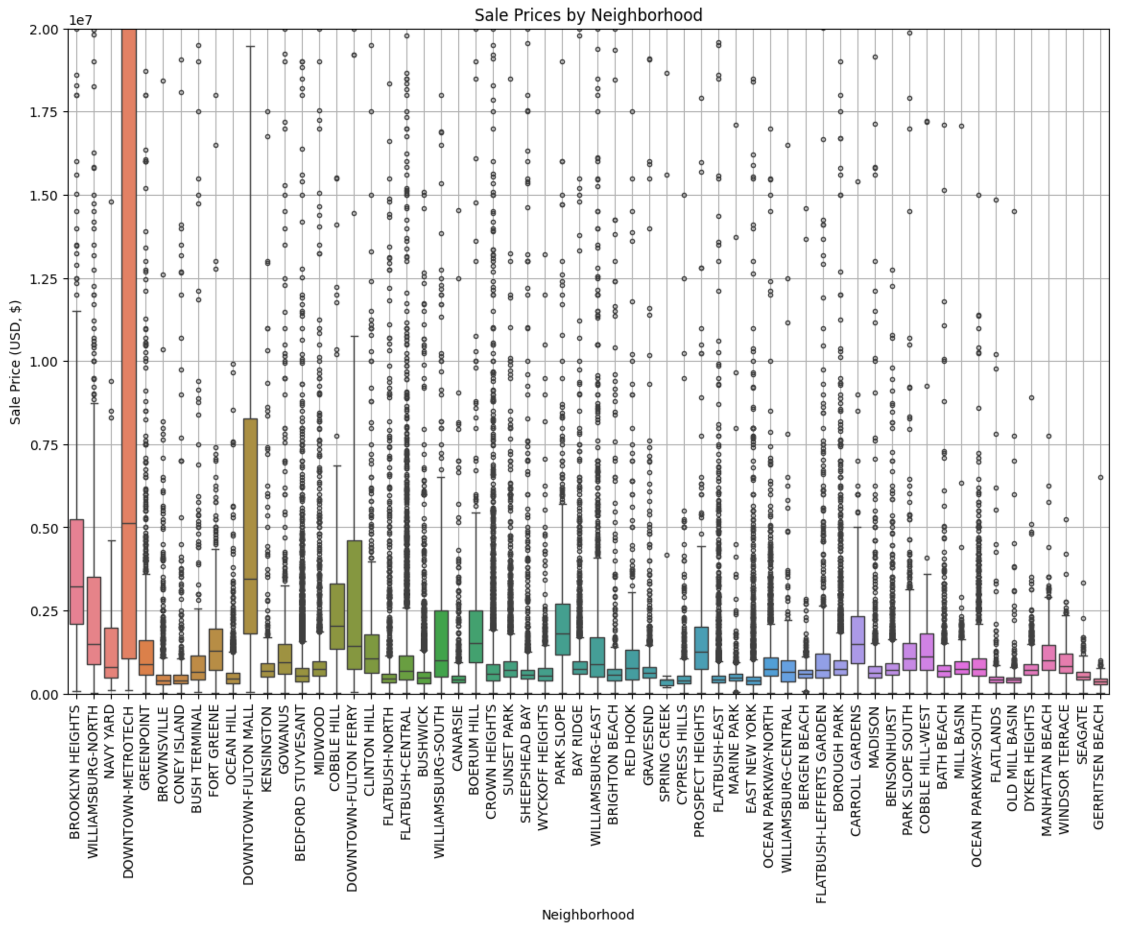




- Sale price has long tail distribution, prices ranging between $20,000 and $171M
- Fewer home sale prices listed during 2008-2010 recession
- No clear/obvious linear or non-linear relation between sale price and building area, year built, assessed total property value, and assessed land value

Preliminary Data Visualization





- As expected, sale price has dependence on categorical variables “zip_code” and “neighborhood”
- Downtown-Metrotech and Downtown-Fulton All seem to have clearly higher median sale prices across all years

Additional Data Pre-Processing & Feature Engineering

- **Feature Selection:**
  - Further removed categorical features that would have overlapping impact on the models
  - For instance, fire department proximity and sanitation district are covered by zip code, neighborhood
  - Removed rows with sale price and critical numerical variables equal to 0
- **Feature Engineering:**
  - Utilized existing data to engineer new features providing refined variables that enhance model accuracy and interpretability.
    - 'zip_code_rank'
    - 'average_price_3m', 'average_price_1y'
- **Data Augmentation:**
  - Added data from S&P500 to address confounding impact of time varying stock market conditions
  - Added data from IYR (iShares U.S. Real Estate ETF) to address confounding impact of time varying real estate sector performance
  - Added data from US annual inflation rate
- **Final Dataframe Shape:**
  - (145027, 60), 41 numerical variables, 17 categorical variables, 1 output (sale price), 1 row index

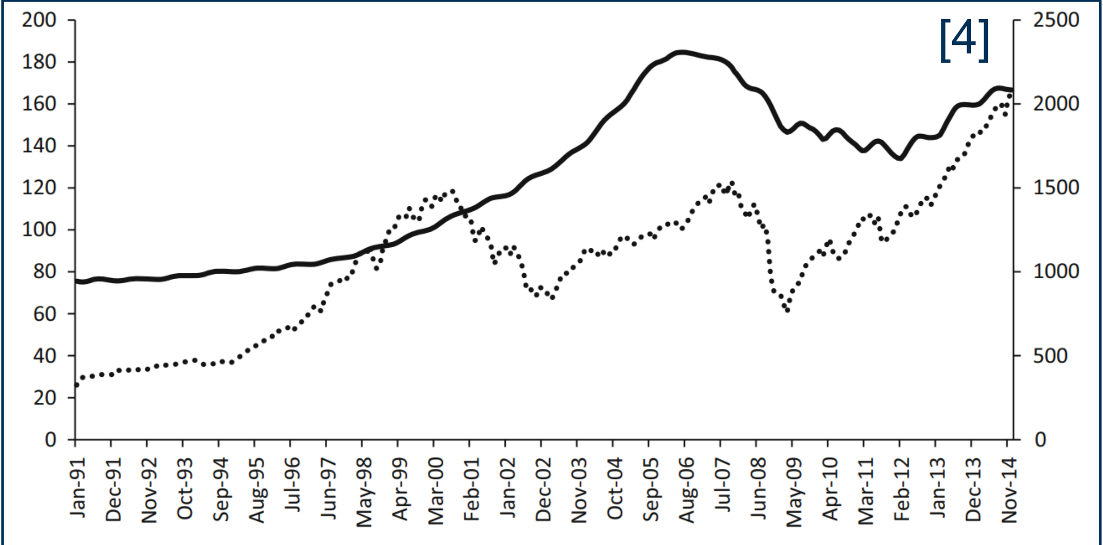
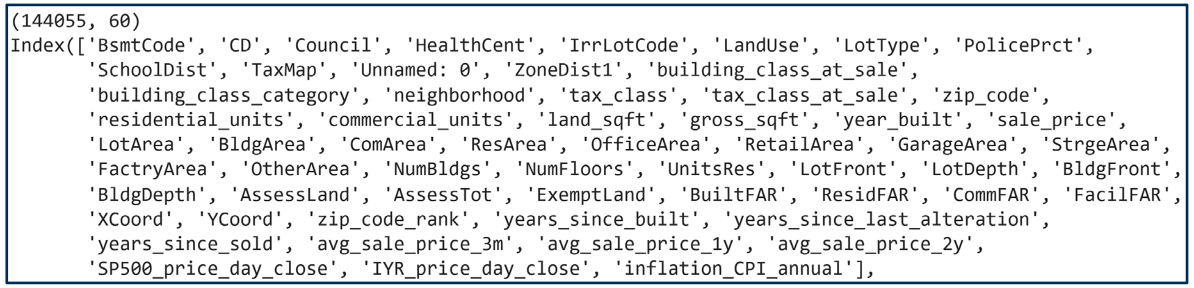


Fig. 1 Evolution of the S&P/Case–Shiller US National Composite Home Price Index and the S&P 500 index (monthly data, Jan 1991–Dec 2014). *Source*: S&P Dow Jones Indices LLC and Prof. R. J. Shiller's Web site



```
(144055, 60)
Index(['BsmtCode', 'CD', 'Council', 'HealthCent', 'IrrLotCode', 'LandUse', 'LotType', 'PolicePrct',
       'SchoolDist', 'TaxMap', 'Unnamed: 0', 'ZoneDist1', 'building_class_at_sale',
       'building_class_category', 'neighborhood', 'tax_class', 'tax_class_at_sale', 'zip_code',
       'residential_units', 'commercial_units', 'land_sqft', 'gross_sqft', 'year_built', 'sale_price',
       'LotArea', 'BldgArea', 'ComArea', 'ResArea', 'OfficeArea', 'RetailArea', 'GarageArea', 'StrgeArea',
       'FactryArea', 'OtherArea', 'NumBldgs', 'NumFloors', 'UnitsRes', 'LotFront', 'LotDepth', 'BldgFront',
       'BldgDepth', 'AssessLand', 'AssessTot', 'ExemptLand', 'BuiltFAR', 'ResidFAR', 'CommFAR', 'FacilFAR',
       'XCoord', 'YCoord', 'zip_code_rank', 'years_since_built', 'years_since_last_alteration',
       'years_since_sold', 'avg_sale_price_3m', 'avg_sale_price_1y', 'avg_sale_price_2y',
       'SP500_price_day_close', 'IYR_price_day_close', 'inflation_CPI_annual'],
```

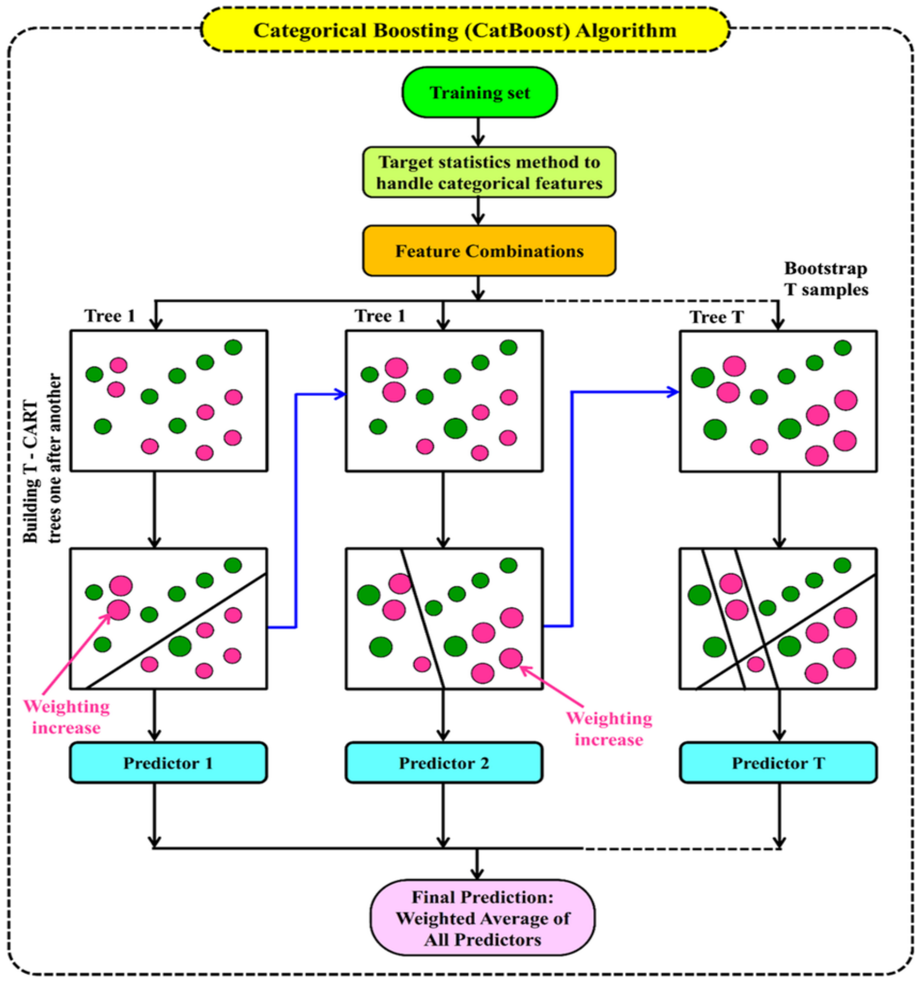
[4] https://link.springer.com/article/10.1007/s00181-015-1037-5

3. Modelling Methodology & Results: Approach 1

Gradient Boosting (CatBoost)

Gradient Boosting - CatBoostRegressor

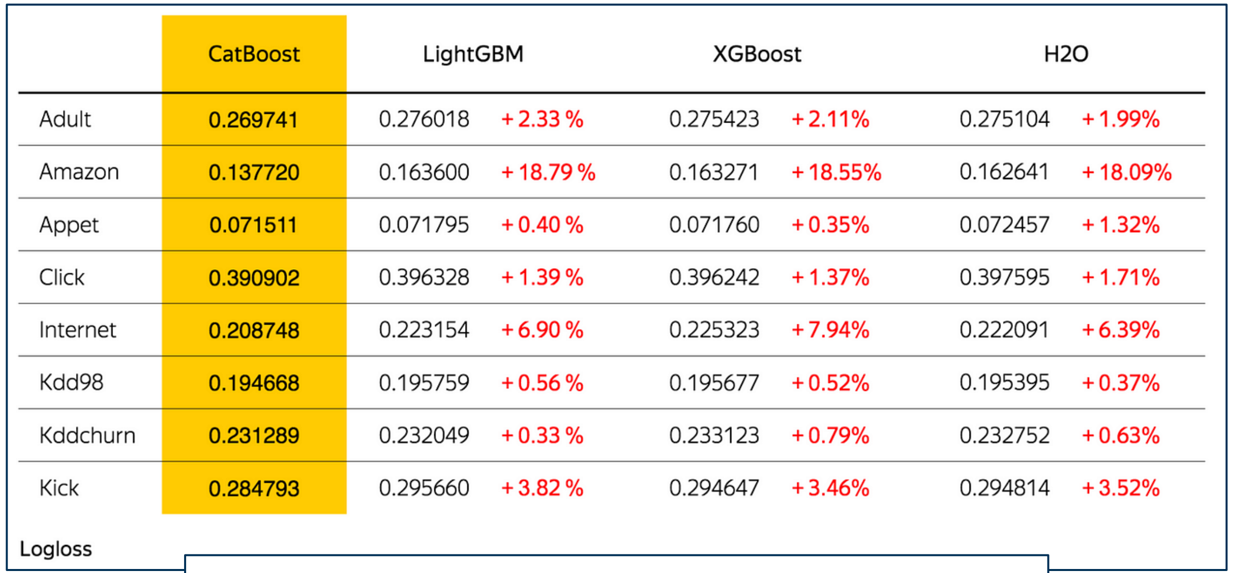
1. **What is CatBoost?**
    1. CatBoost is a variant of Gradient Boosting algorithms which sequentially stacks boosted decision trees. It excels in native support of categorical features.
    2. In general, Gradient Boosting aims to stack improved iterations of learners which are trained based on the residuals (loss function gradients) of the previous learner.
2. **Key Features:**
    1. **Encoding of categorical features:** CatBoost uses Ordered Target Encoding where the categories are encoded using the target. This works well when dealing with high cardinality features.
    2. **Ordered boosting:** CatBoost trains the model on a subset of data while calculating residuals on another subset.
    3. **Balanced architecture:** CatBoost builds symmetric (balanced) trees unlike XGBoost and LightGBM. This helps with overfitting.
3. **Benefits in Predictive Modeling:**
    1. **Accuracy:** Provides state-of-the-art results with minimal hyperparameter tuning. Incorporates L2 regularization to avoid overfitting.
    2. **Computational speed:** Due to its unique encoding algorithm, CatBoost is capable of training large datasets with high cardinality categorical variables.
    3. **Analysis tools:** CatBoost provides in-built tools for analysis and visualizing results.


Categorical Boosting (CatBoost) Algorithm
Training set
Target statistics method to handle categorical features
Feature Combinations
Bootstrap T samples
Tree 1
Tree 1
Tree T
Building T - CART trees one after another
Weighting increase
Weighting increase
Predictor 1
Predictor 2
Predictor T
Final Prediction: Weighted Average of All Predictors

CatBoostRegressor Rationale

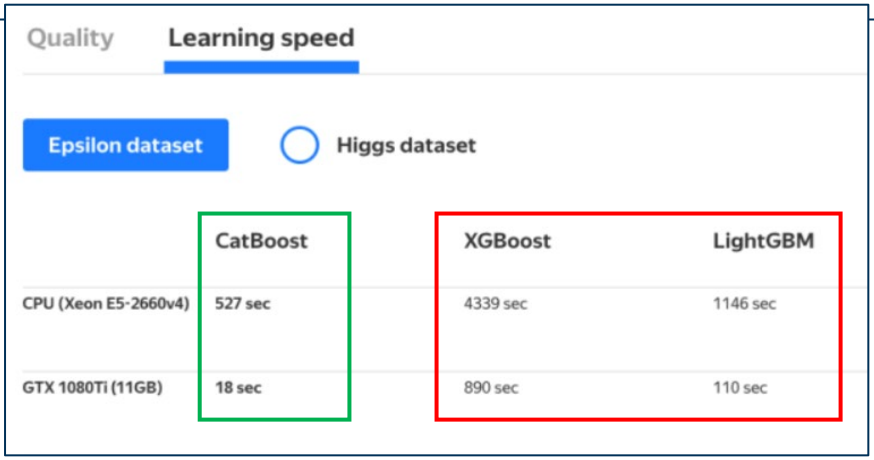
- **High cardinality features:** Our dataset has ~15 categorical features, some of which are high cardinality. CatBoost can efficiently handle these features.
- **High variance:** The dataset has a wide variance in the target values which could bias the training in favor of outliers. CatBoost has some unique characteristics that make it robust to overfitting.
- **Size of dataset:** Our dataset has ~ 150k points and very high dimensionality. CatBoost offers faster computation times compared to XGBoost.

Benchmarks



	CatBoost	LightGBM		XGBoost		H2O	
Adult	0.269741	0.276018	+2.33 %	0.275423	+2.11%	0.275104	+1.99%
Amazon	0.137720	0.163600	+18.79 %	0.163271	+18.55%	0.162641	+18.09%
Appet	0.071511	0.071795	+0.40 %	0.071760	+0.35%	0.072457	+1.32%
Click	0.390902	0.396328	+1.39 %	0.396242	+1.37%	0.397595	+1.71%
Internet	0.208748	0.223154	+6.90 %	0.225323	+7.94%	0.222091	+6.39%
Kdd98	0.194668	0.195759	+0.56 %	0.195677	+0.52%	0.195395	+0.37%
Kddchurn	0.231289	0.232049	+0.33 %	0.233123	+0.79%	0.232752	+0.63%
Kick	0.284793	0.295660	+3.82 %	0.294647	+3.46%	0.294814	+3.52%

Logloss



	CatBoost	XGBoost	LightGBM
CPU (Xeon E5-2660v4)	527 sec	4339 sec	1146 sec
GTX 1080Ti (11GB)	18 sec	890 sec	110 sec

CatBoostRegressor Configuration and Initialization

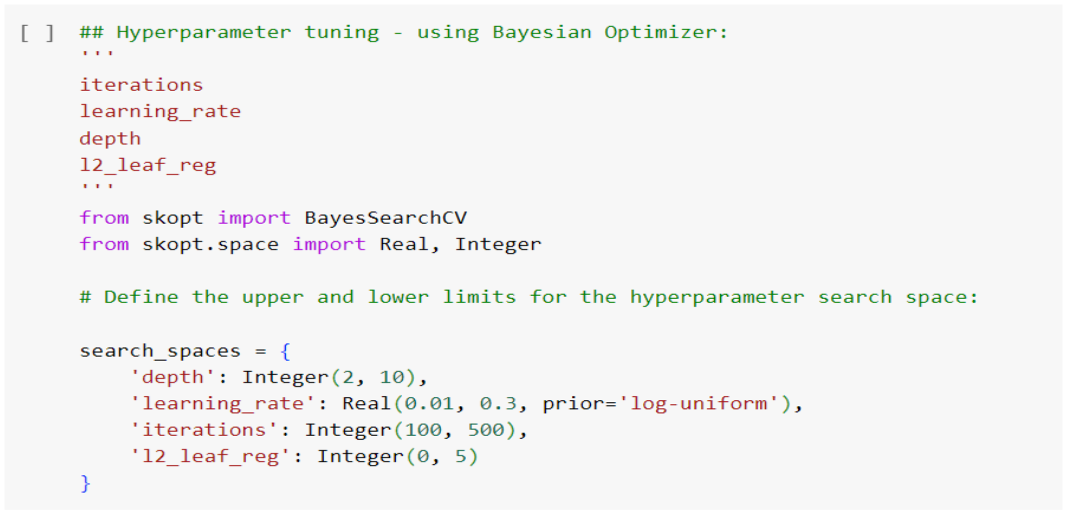
- Standard scaling was performed on the numeric features.
- Training and testing sets were split 80/20. The validation set for tuning was drawn as a subset from the training set.
- Model initialization:

```
gbr = CatBoostRegressor(loss_function='RMSE', random_state=42,
                        one_hot_max_size=2, cat_features=categorical_features,
                        early_stopping_rounds=50)
```

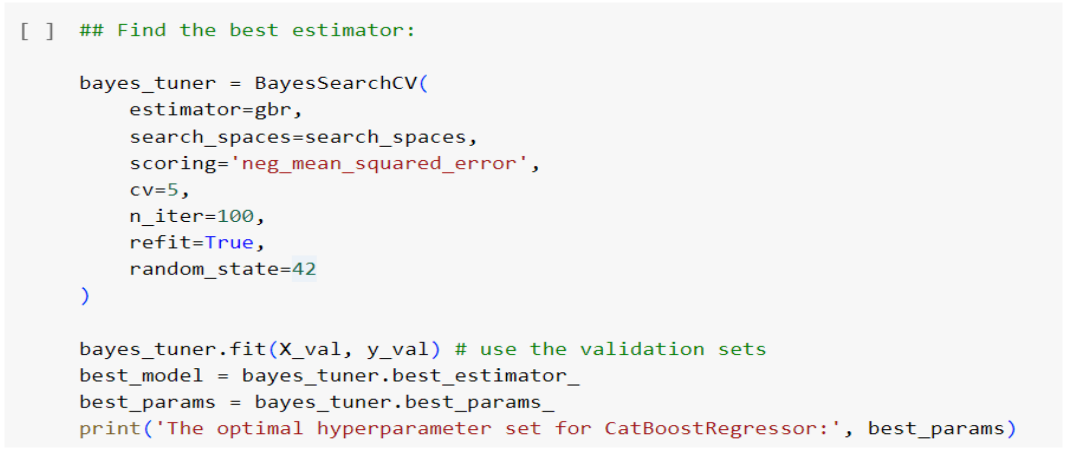
  - Loss function was set as RMSE.
  - “one_hot_max_size” was set to 2 to ensure that categorical features with more than 2 categories are encoded using Ordered Target Encoding.
  - The list of categorical features were passed to the model explicitly.
  - “early_stopping_rounds” was set to 50 to reduce training time. This stops the training for each learner if improvement in the loss function is not observed for more than 50 iterations.

Hyperparameter Tuning - Approach

- A smaller validation set instead of the entire training set was used for hyperparameter tuning for faster convergence.
- Bayesian optimization - `BayesSearchCV` was used for finding the optimal values of the hyperparameters over a specified search space.
- 5-fold cross-validation was used.
- The following hyperparameters were considered:
  - **`iterations`** – equivalent to the no. of estimators – measure of overall model complexity.
  - **`learning_rate`** – equivalent to the step-size in the optimization problem – controls the speed of convergence.
  - **`depth`** – the max. depth of each decision tree – measure of complexity of each estimator.
  - **`l2_leaf_reg`** – equivalent to the regularization strength in LASSO – controls the overfitting.



```
[ ]  ## Hyperparameter tuning - using Bayesian Optimizer:
     '''
     iterations
     learning_rate
     depth
     l2_leaf_reg
     '''
     from skopt import BayesSearchCV
     from skopt.space import Real, Integer

     # Define the upper and lower limits for the hyperparameter search space:

     search_spaces = {
         'depth': Integer(2, 10),
         'learning_rate': Real(0.01, 0.3, prior='log-uniform'),
         'iterations': Integer(100, 500),
         'l2_leaf_reg': Integer(0, 5)
     }
```



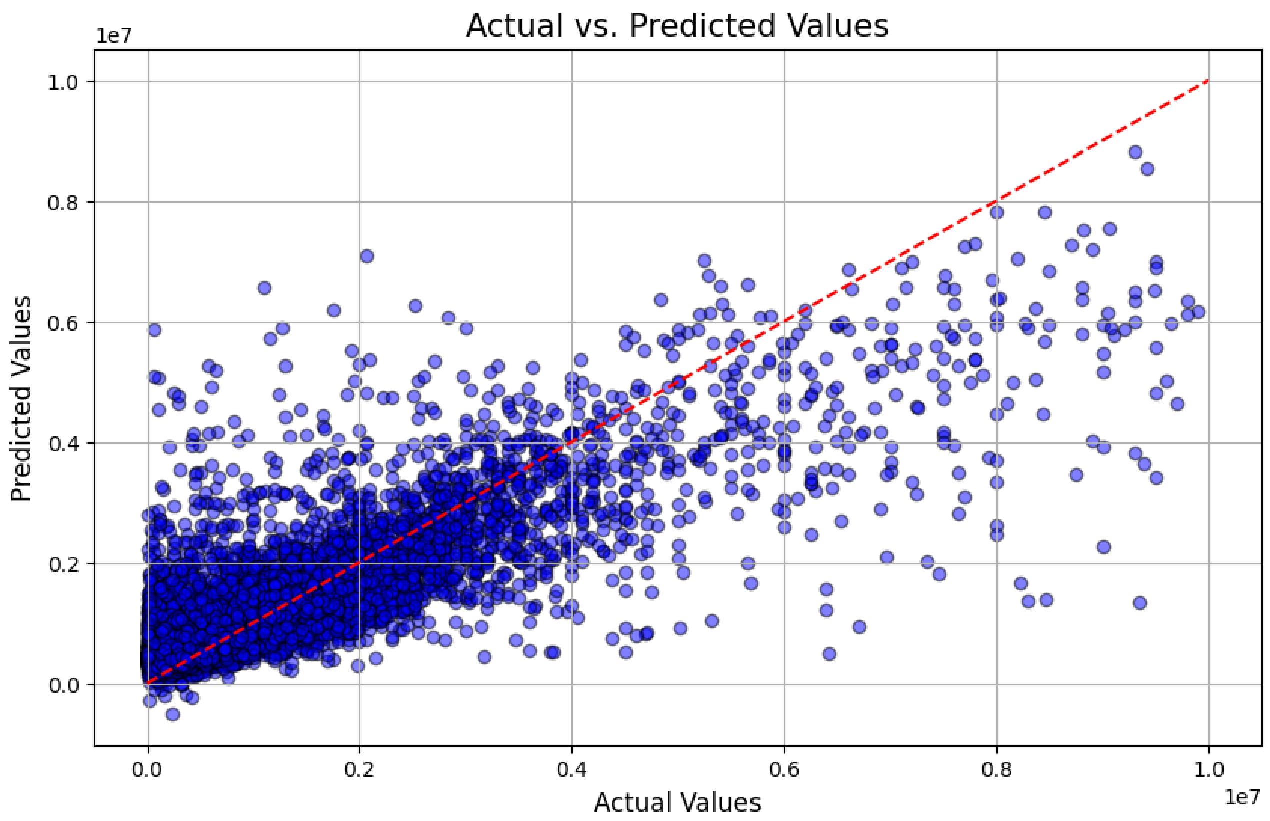
```
[ ]  ## Find the best estimator:

     bayes_tuner = BayesSearchCV(
         estimator=gbr,
         search_spaces=search_spaces,
         scoring='neg_mean_squared_error',
         cv=5,
         n_iter=100,
         refit=True,
         random_state=42
     )

     bayes_tuner.fit(X_val, y_val) # use the validation sets
     best_model = bayes_tuner.best_estimator_
     best_params = bayes_tuner.best_params_
     print('The optimal hyperparameter set for CatBoostRegressor:', best_params)
```

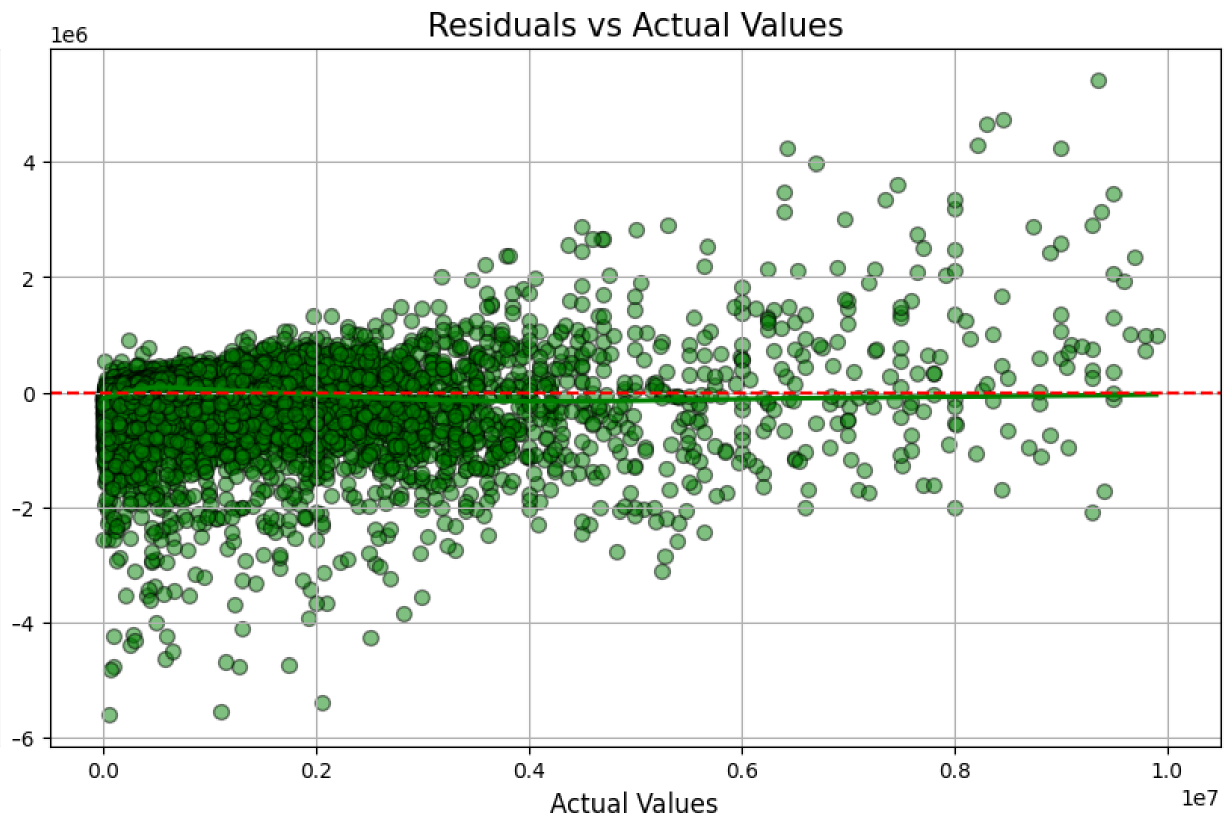
Hyperparameter Tuning - Results

- The optimized values of 3/4 hyperparameters were found to be within the search space specified.
- The overall tuning process took 6 hrs.
- The no. of iterations, however, always converged to the upper limit of the range.
- Further optimization of the no. of iterations was done using `GridSearchCV`.
- It was found that as the no. of iterations increased significantly, the improvement in the MSE on the testing set was marginal.

Hyperparameter	Optimal value	Range specified
`iterations`	2500	(100, 2500)
`learning_rate`	0.0191	(0.01, 0.3)
`depth`	5	(2, 10)
`l2_leaf_reg`	0	(0, 5)

Trained Model - Results

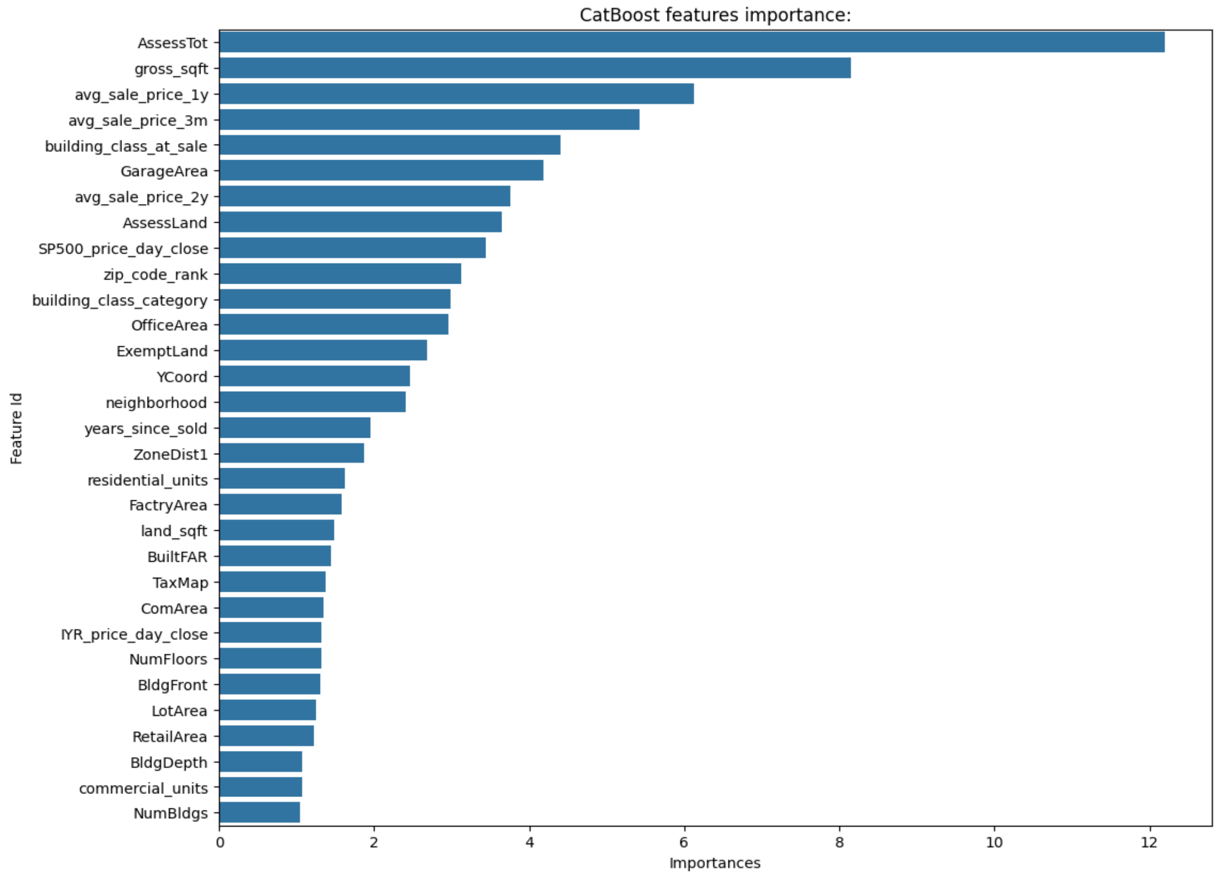




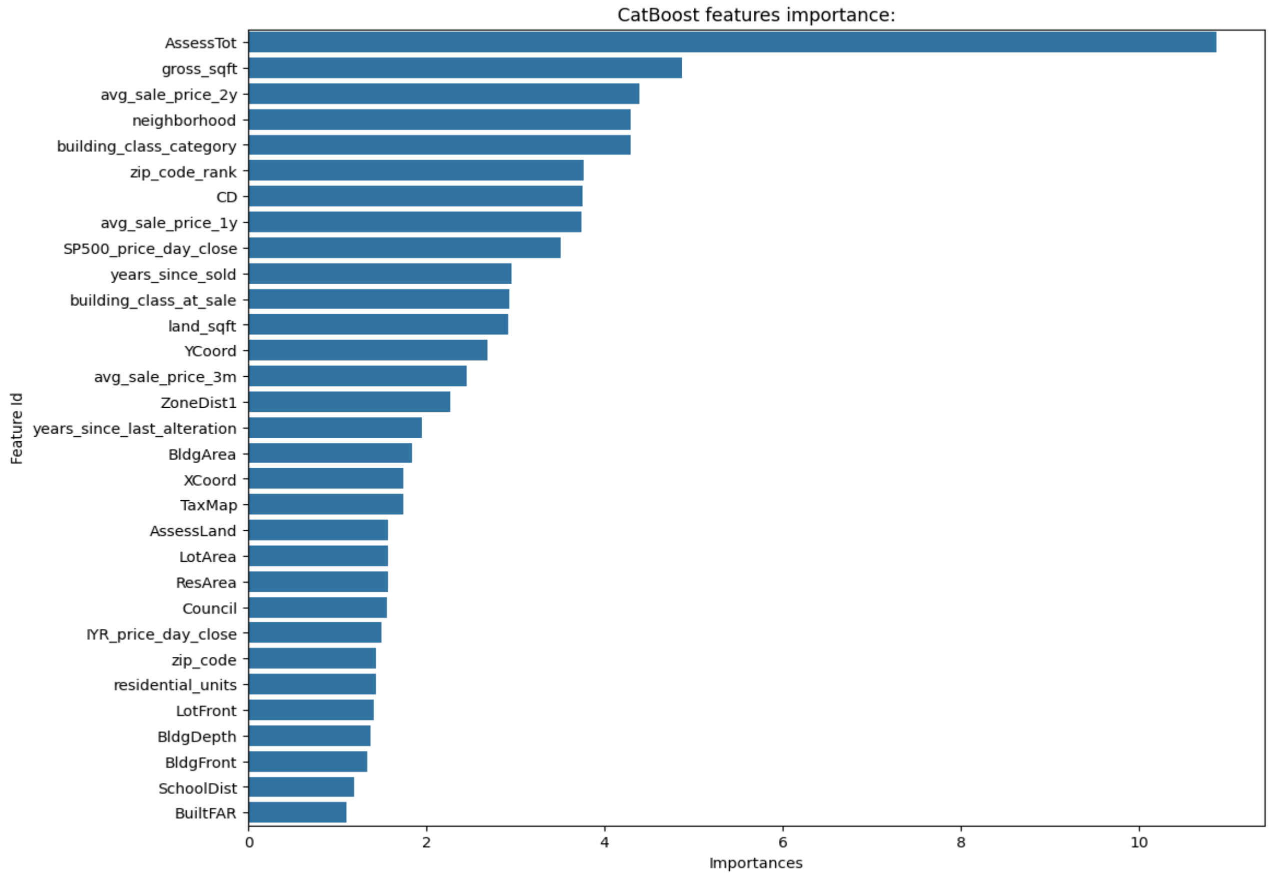
- **Performance Metrics:**
  - **MSE on test set:** 0.3136
  - **RMSE on test set:** 0.5599
  - **R² Score of test set predictions:** 0.7107 - predictors used in the model account for ~ 71% of the total variance.
  - **Data Scope:** Trained on the filtered dataset post-cleaning and EDA (sale price <= $10M)

Feature Importance Plots

Dataset Not Restricted by Sale Price



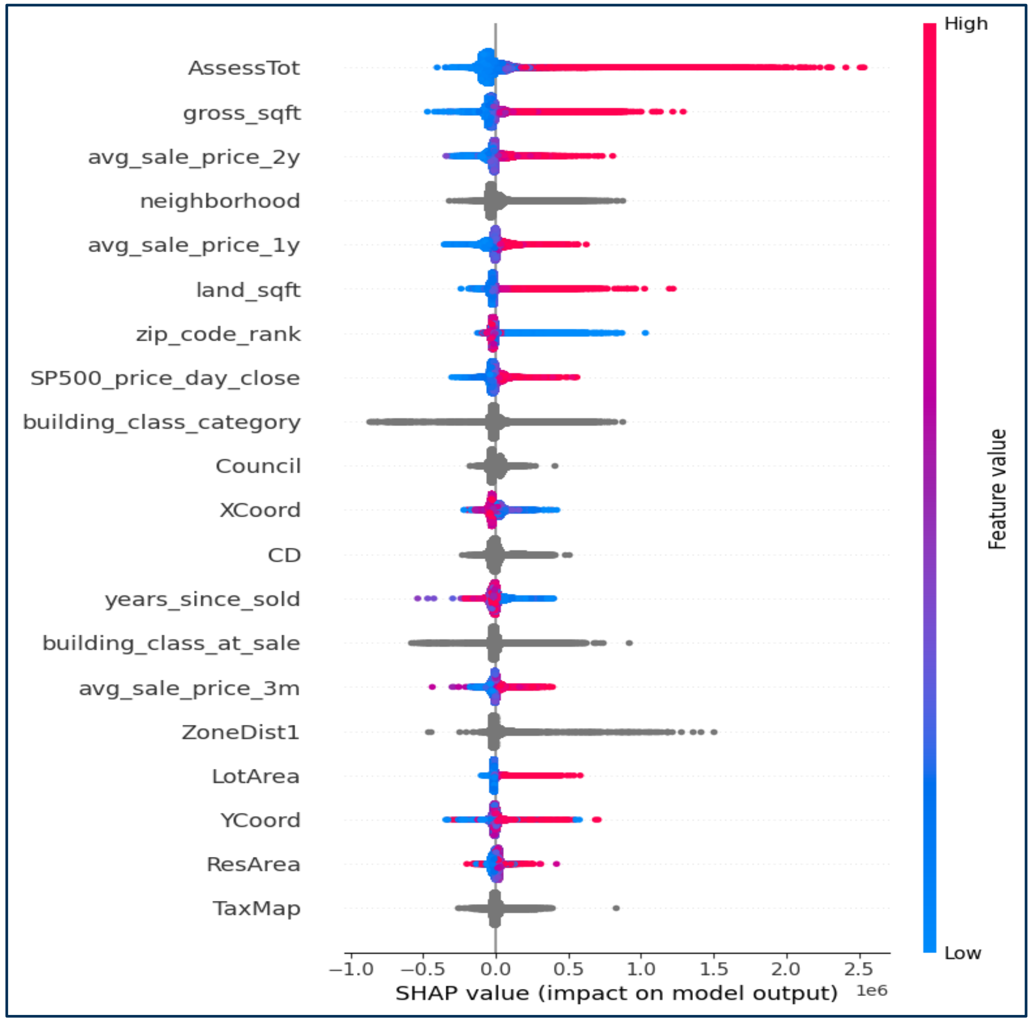
Sale Price Restricted to <= $10M



- The feature importance plot indicates the main features which drive the model prediction.
- The top features are related to the assessed total value, square footage (area), building class, location (ZIP code) and the temporal factor (S&P500, years since sold).

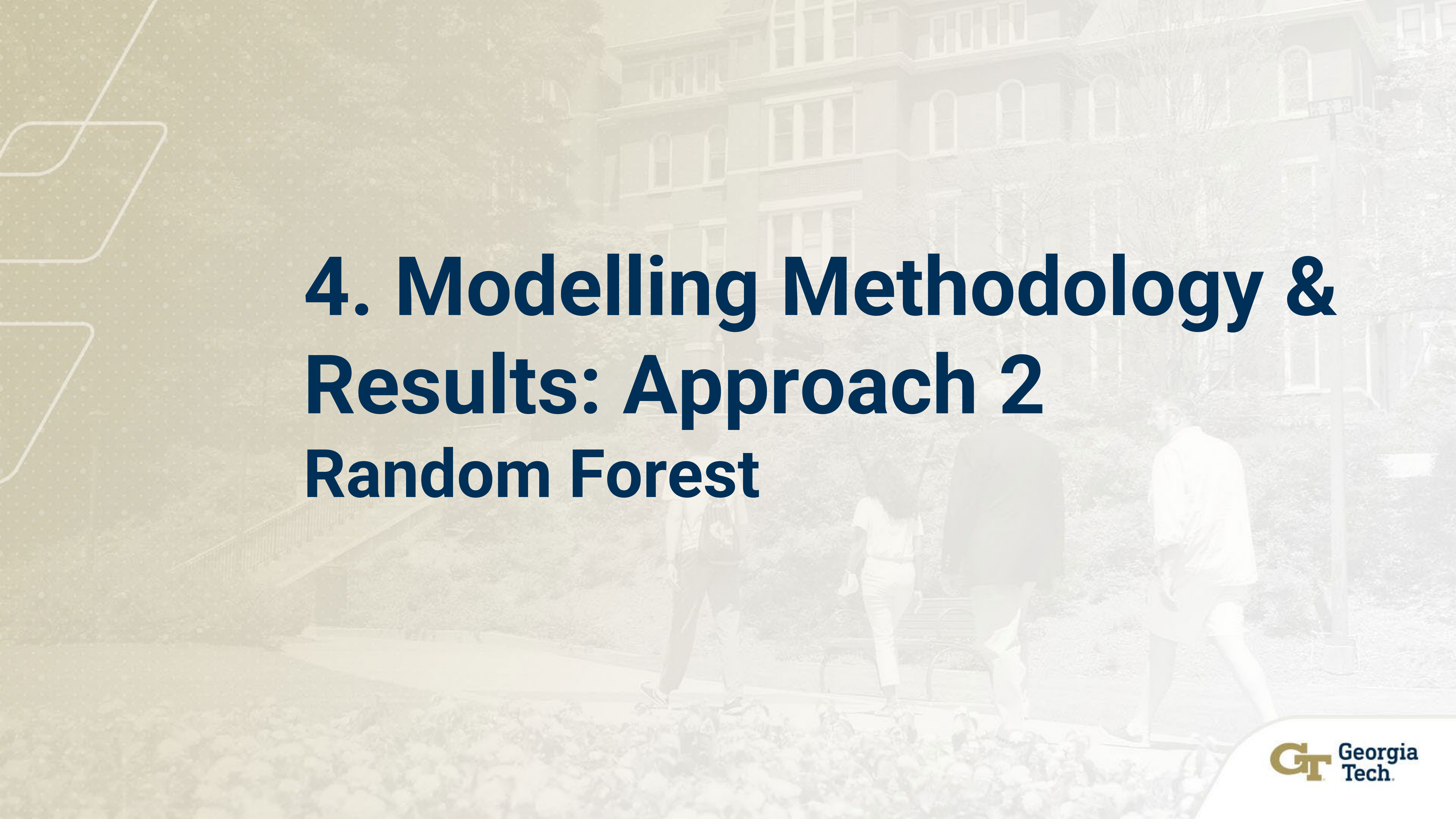
Feature Importance - SHAP Plots

- SHAP (SHapley Additive ExPlanations) value is measure of the contribution of each feature to the overall prediction.
- The location of the dot horizontally shows the impact of that feature on the model's output for that prediction. Features pushing the prediction higher are shown in red and those pushing the prediction lower are in blue.
- A SHAP value of zero means the feature did not change the prediction from the base value (the model's average prediction over the training set). The further away from zero, the more impact the feature has.
- A vertical line of many dots suggests a common impact on the prediction, while scattered dots indicate varying impacts.



CatBoost – Strengths and Limitations

- **Strengths:**
  - CatBoost was able to handle the categorical features passed efficiently.
  - The predictions are accurate, and the feature importance list is consistent with what is expected.
  - Was able to sufficiently account for the temporal dependence.
  - The SHAP plots give insight into which features we could eliminate and still maintain prediction accuracy. This would be useful is scaling up for larger datasets.
- **Limitations:**
  - The predictions are inaccurate at the extreme ends of the dataset.
  - For instance, predictions are undervalued for more expensive houses.

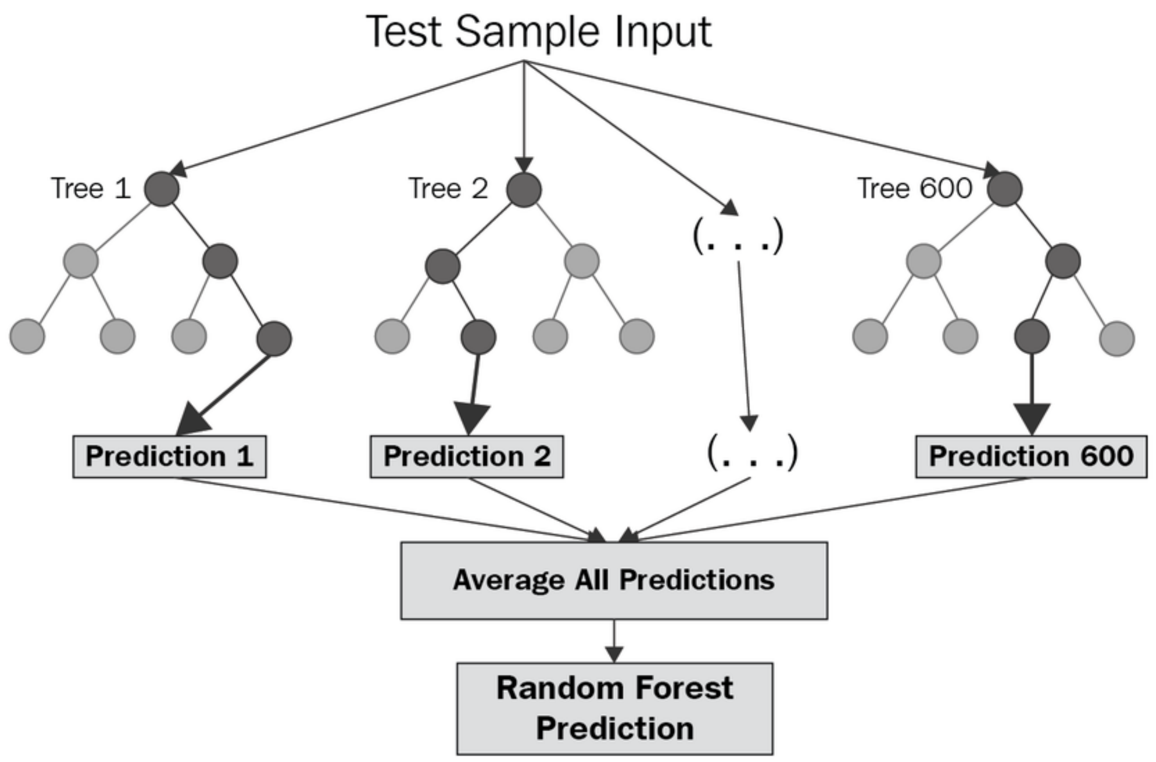


4. Modelling Methodology & Results: Approach 2

Random Forest

Introduction to Random Forest

- **Definition:** A Random Forest Regressor is an ensemble learning method used for regression tasks, which combines multiple decision trees to improve accuracy and control over-fitting.
- **Mechanism:** It builds multiple decision trees during training and outputs the mean prediction of the individual trees, providing a more stable and reliable prediction.
- **Advantages:**
    - **Robust to Overfitting:** Utilizes averaging to reduce the risk of overfitting, making it more reliable than individual decision trees.
    - **Handles Non-linearity:** Effectively manages non-linear relationships between features and the target variable.
    - **Feature Importance:** Offers insights into which features are most influential in predicting the target variable.



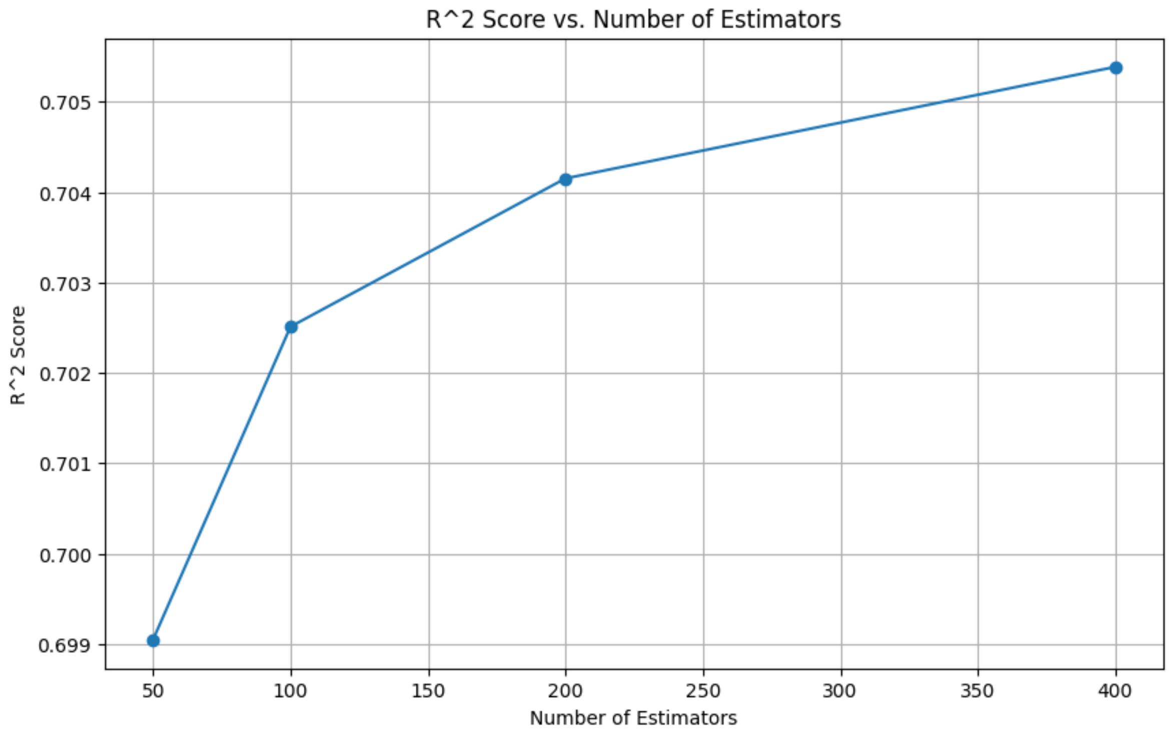
Random Forest Rationale and Implementation

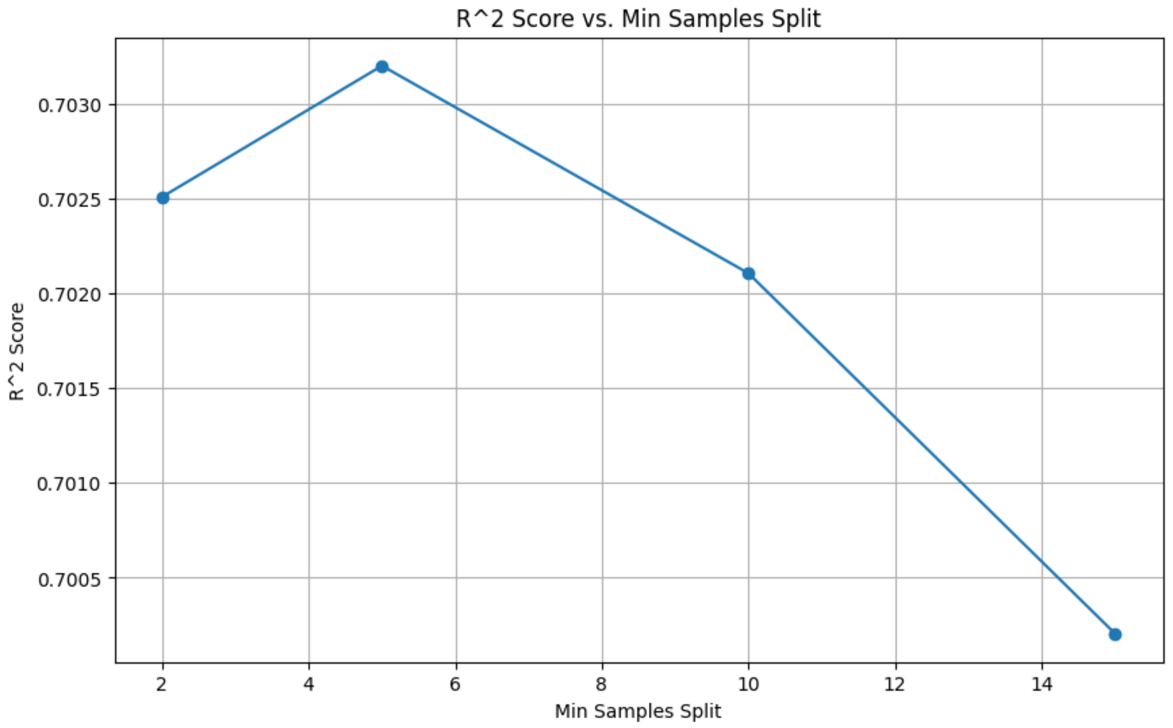
- **Handling Large Datasets:**
  - **Scalable and Efficient:** Manages our dataset with 140,000 data points and 41 numerical features, ensuring efficient processing and robust performance.
  - **Feature Handling:** Optimized for numerical data, the model effectively handles large volumes of quantitative inputs, enhancing predictive accuracy.
  - **Noise robustness:** Its ensemble approach makes it robust against noise in the dataset.
- **Implementation of Random Forest:**
  - **Building the Forest:** Each decision tree is trained on a random subset of data, reducing variance and improving generalization.
  - **Averaging Predictions:** Outputs from various trees are averaged, minimizing errors and stabilizing predictions against noisy data.
- **Hyperparameter Optimization with For Loops:**
  - **Parameters Tuned:**
    - **Tree Count:** Adjusts how many trees are included in the forest.
    - **Tree Depth:** Controls how deep each tree can grow to avoid overly complex models.
    - **Data Points for Splitting:** Sets the minimum number of data points required to consider splitting a node.
    - **Data Points per Leaf:** Specifies the minimum number of data points that must be present in a leaf node.

Random Forest Evaluation and Performance Metrics

- **Setup:** Training and validation on a dataset with 140,000 points, focusing on numerical features.
- **Key Metrics:**
  - **MSE & RMSE:** Indicate prediction error magnitude.
  - **R-squared (R^2):** Reflects the variance explained by the model.
- **Significance:**
  - Metrics chosen for accuracy and interpretability.
  - High R^2 and lower error rates signify a robust model.

Random Forest Hyperparameter Optimization

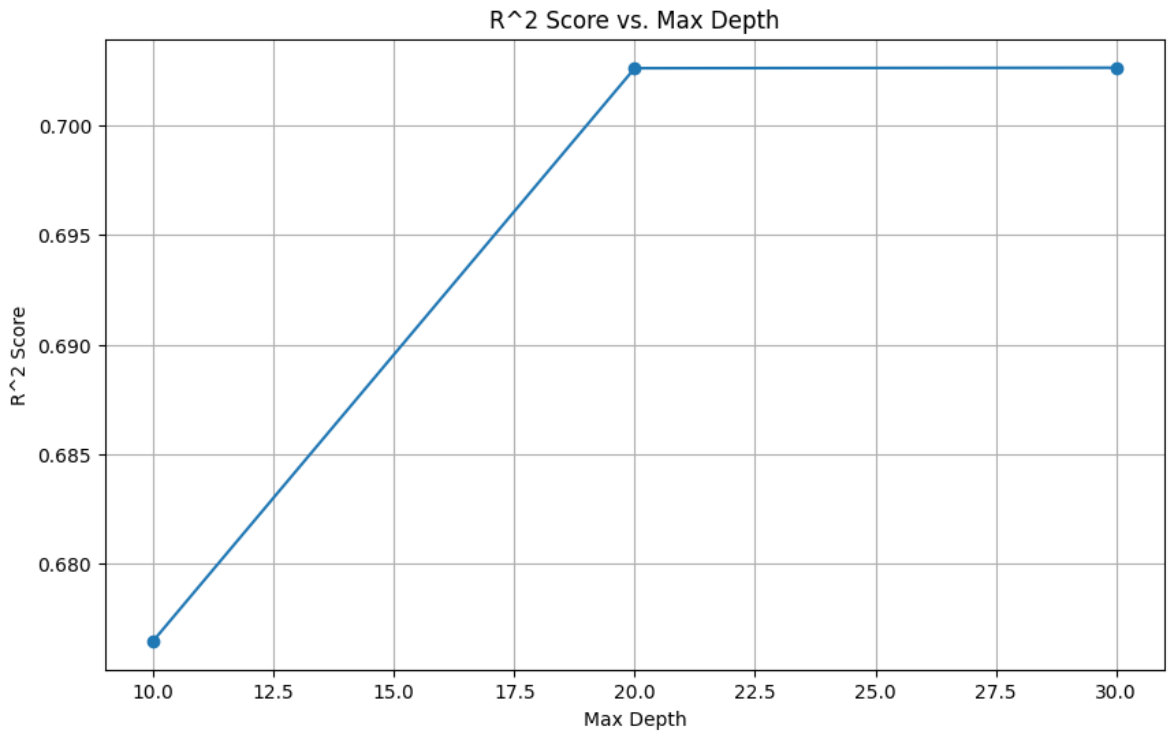




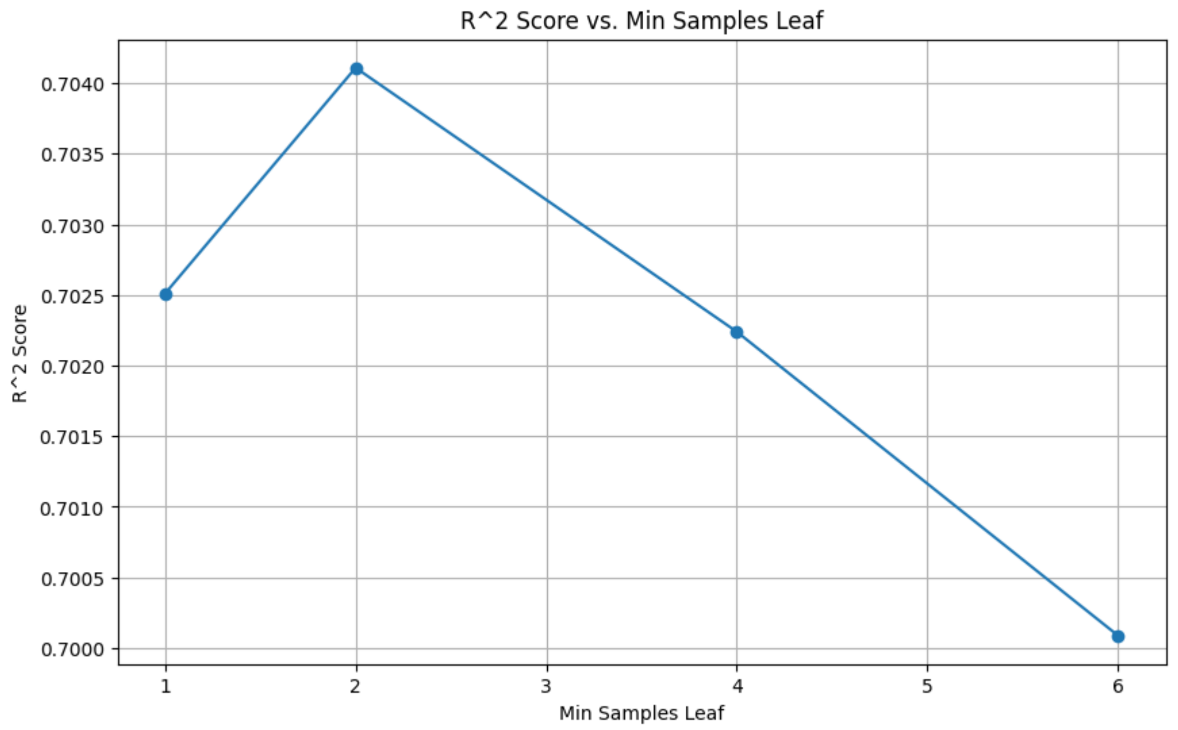
Increasing the number of trees in the forest leads to a slight improvement in the model's R² score, indicating better performance up to a certain point.

The model's R² score peaks at a low number of minimum samples required to split a node, then declines as the criterion becomes less stringent.

Random Forest Hyperparameter Optimization



The model's explanatory power improves significantly with increased tree depth, plateauing as maximum depth reaches 20.

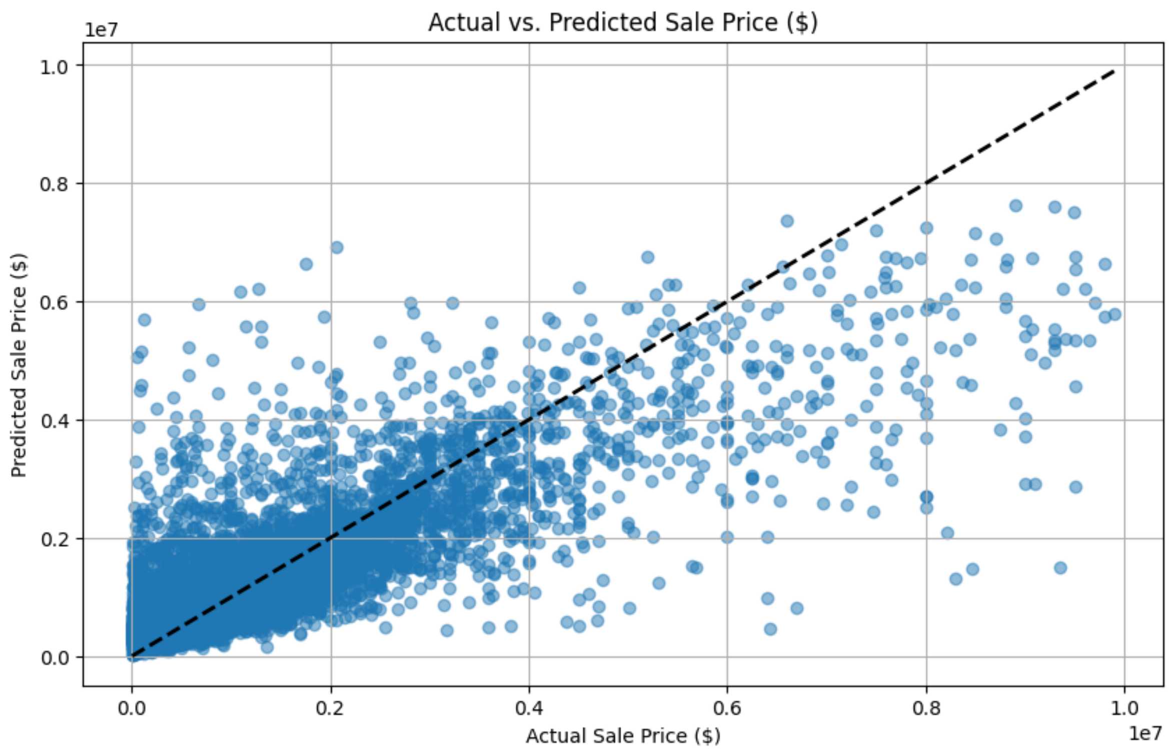


The optimal balance for leaf samples is found with a single data point, as more samples per leaf lead to a steady decrease in R² score.

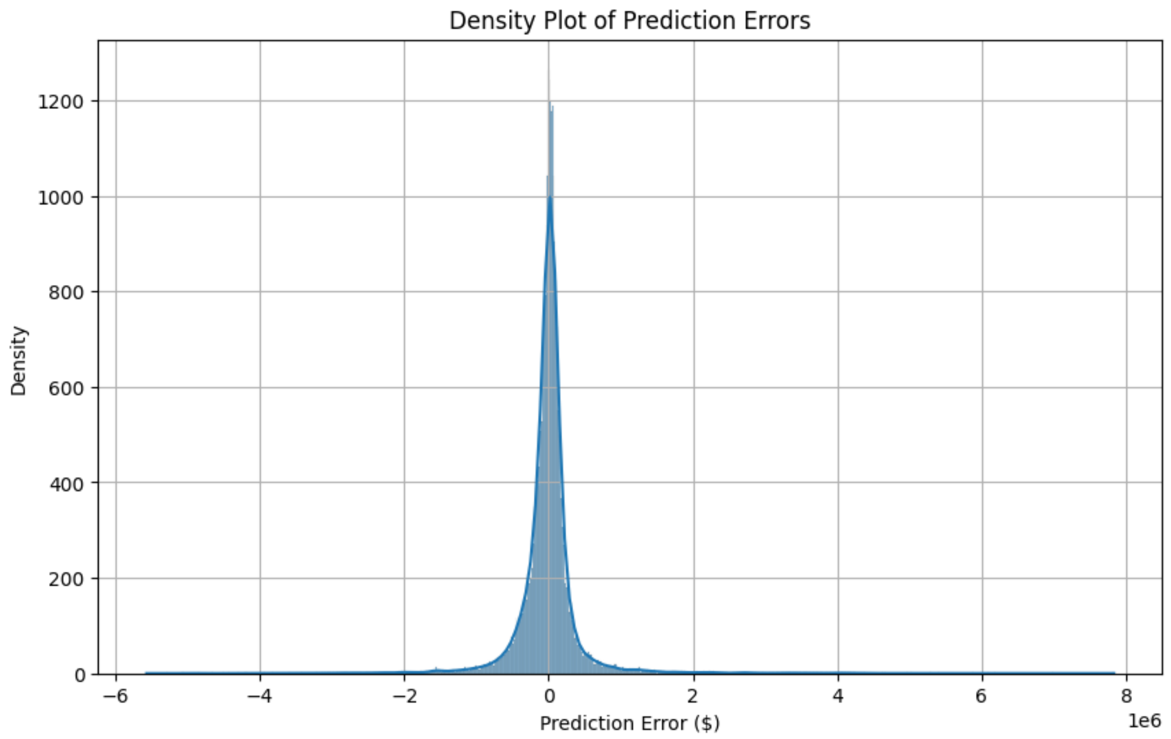
Random Forest Results

- **Optimized Parameters:**
  - **Tree Ensemble Size:** Set to 100 trees for robust predictions.
  - **Tree Depth:** Unrestricted, allowing the model to learn complex patterns.
  - **Node Split Requirements:** Minimum of 2 data points to split, ensuring fine-grained learning.
  - **Leaf Node Requirements:** At least 1 data point per leaf for detailed predictions.
- **Performance Outcome:**
  - **MSE:** Noted at 0.3 (after normalizing) , reflecting average prediction error.
  - **RMSE:** Approximately 0.55 (after normalizing) , indicating typical error magnitude.
  - **R^2 Score:** 0.70, the model accounts for 70% of variance in the target.
- **Data Scope:** Trained on properties with prices up to $10M to maintain focus.

Random Forest Results (Continued)



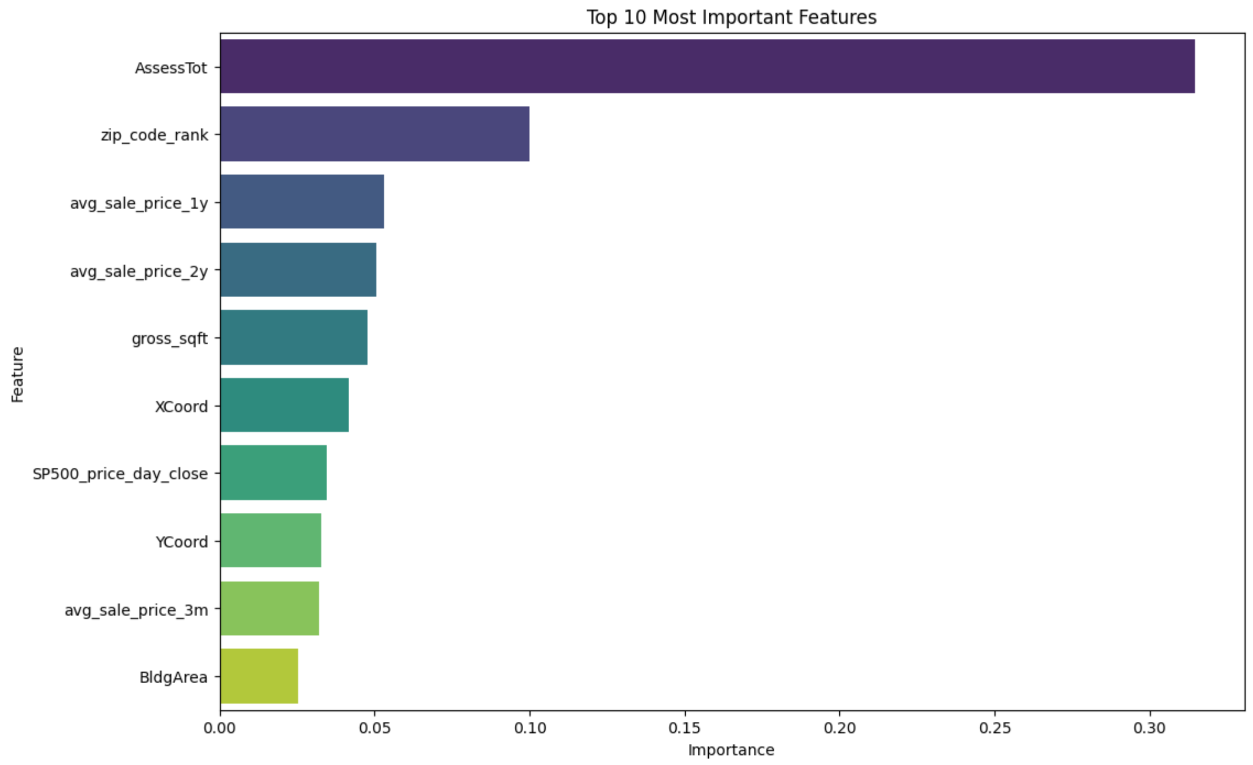
The scatter plot reveals a strong alignment along the dashed line representing perfect predictions, with some spread indicating variability in accuracy for higher values.



The density plot shows a sharp peak at zero, indicating that most predictions are close to the actual values, with very few large errors.

Random Forest Feature Importance

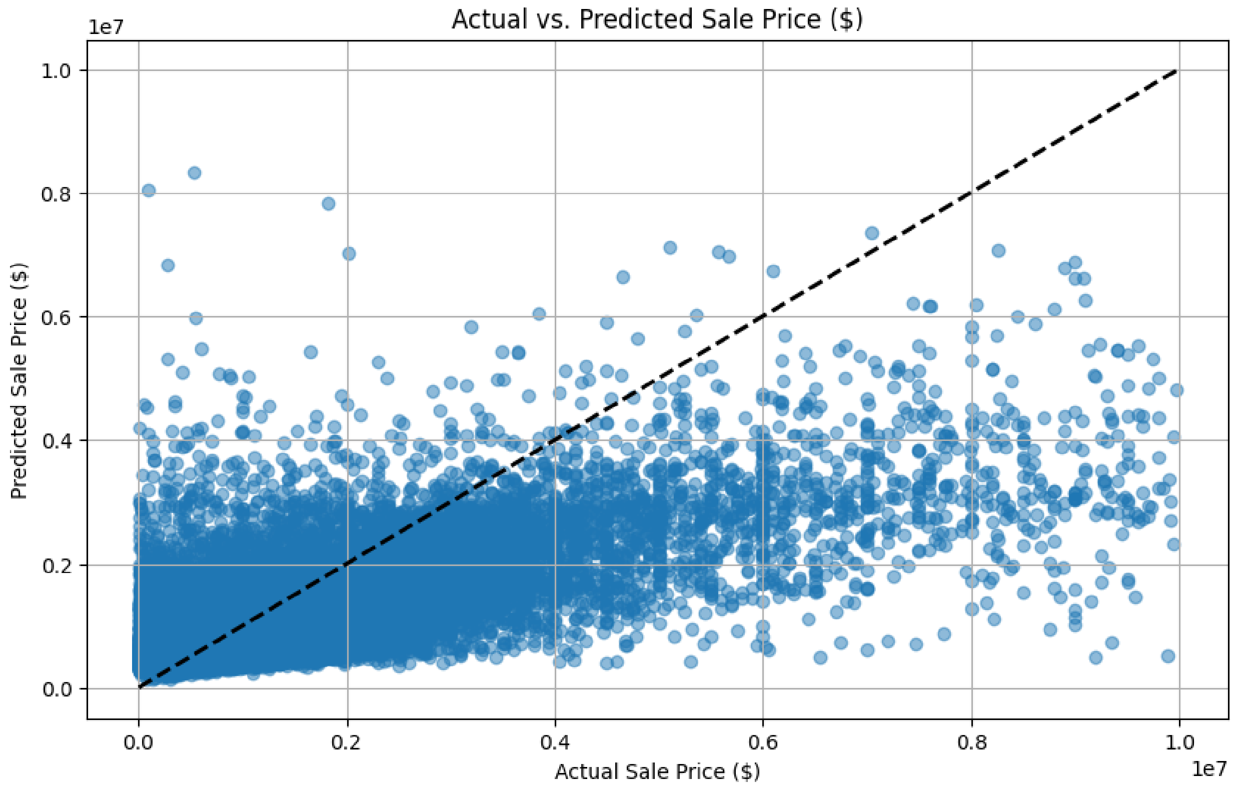
- **Primary Driver:** Total assessed value ('AssessTot') is the most influential feature, significantly impacting predictions.
- **Sale Price History:** Recent average sale prices ('avg_sale_price_1y', 'avg_sale_price_2y', 'avg_sale_price_3m') show strong predictive power.
- **Property Size Metrics:** Square footage ('gross_sqft') and building area ('BldgArea') are key size indicators affecting property valuation.
- **Location Influence:** Zip code ranking ('zip_code_rank') and coordinates ('YCoord') underscore the value of location in real estate pricing.
- **Market Context**: The presence of the S&P 500 closing price (SP500_price_day_close) as a feature suggests that broader economic performance may have a nuanced influence on real estate pricing, albeit less pronounced than the direct property-related factors.



Understanding Time Dependence on Model Performance

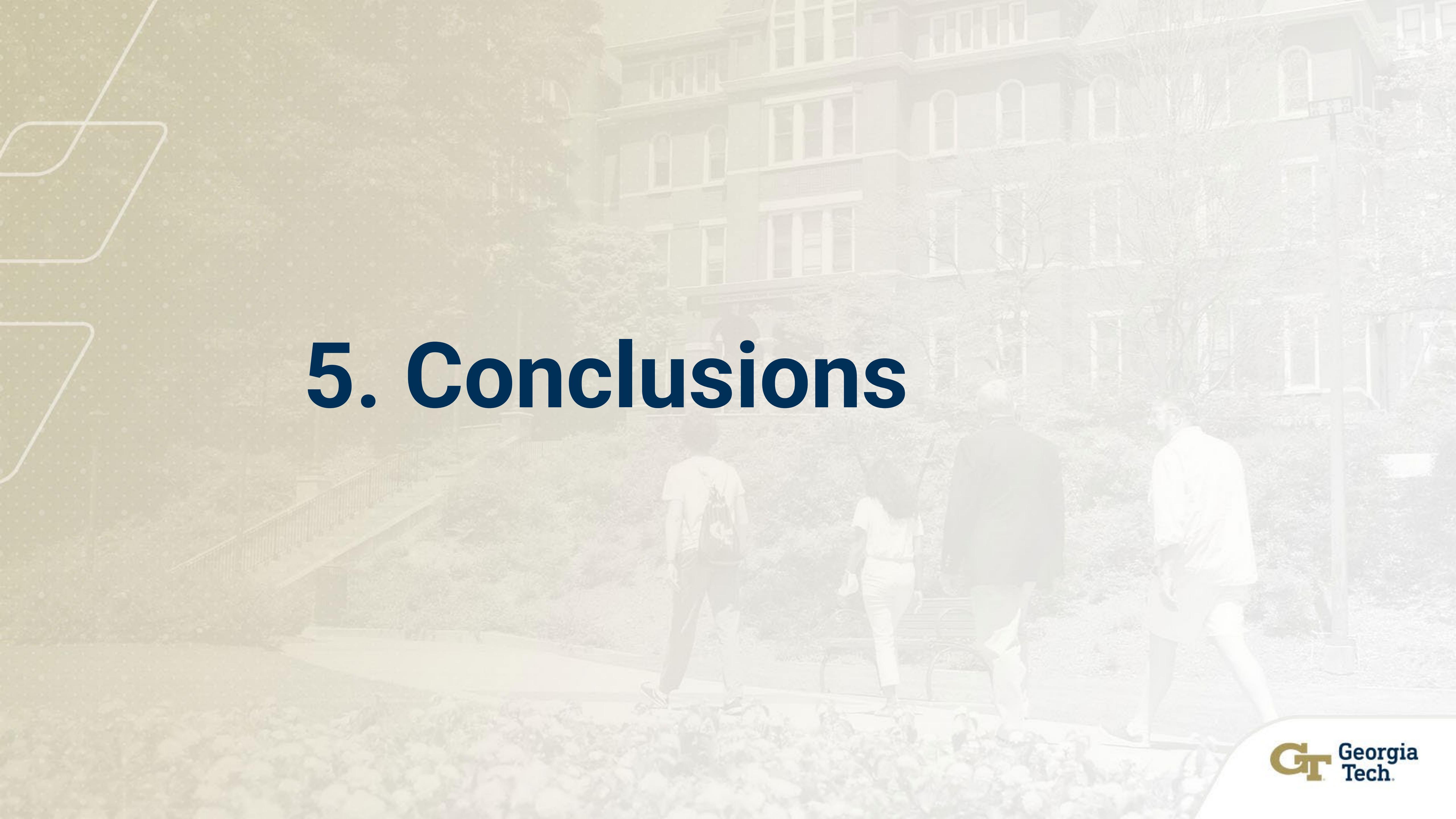
- **Training on Historical Data**: The model was trained on older property sales data where years_since_sold is greater than 8, capturing older market dynamics and trends.
- **Testing on Recent Data**: Evaluation was conducted on more recent property sales with years_since_sold less than or equal to 8, representing a shift to newer market conditions.
- **Moderate Performance**: The model achieved an r-squared (R^2) score of 0.505, indicating that approximately 50.5% of the variance in the sale prices can be explained by the model — a moderate level of predictive accuracy.
- **Interpretation**: The moderate R^2 score suggest that while the model can capture some patterns from historical data, its predictions on recent data lack precision. This could imply that market factors and property values have evolved over time.

Model Trained on Sale Prices 2003 – 2009
Model Tested on Sale Prices 2010 – 2017



Random Forest: Strengths & Limitations

- **Strengths:**
  - **High Accuracy:** Demonstrated strong predictive power, particularly with numerical data.
  - **Robust to Overfitting:** Due to ensemble approach, the model is less prone to overfitting compared to single decision trees.
  - **Feature Interpretability:** Provides clear insights into which features are most influential in predicting outcomes.
- **Limitations:**
  - **Excludes Categorical Data:** While performance remained high, the exclusion of categorical data may overlook potential insights.
  - **Complexity and Resource Use:** With a large number of estimators, the model requires considerable computational resources.
- **Surprises:**
  - **Numerical Data Sufficiency:** Surprisingly, the removal of all categorical variables did not significantly impact the model's performance.
- **Learnings Beyond Prediction:**
  - **Feature Importance as a Guide:** The model's feature importance can inform strategic business decisions and policy-making.
  - **Data Preparation Insight:** Highlights the impact of feature selection on model efficiency and outcome.
  - **Scope for Model Simplification:** Suggests potential for a simpler model that could still capture most of the predictive variance.



5. Conclusions

Conclusions and Future Directions

- Main Takeaways:
  - Gradient Boosting (CatBoost):
    - R^2 of 0.71
  - Random Forest:
    - R^2 of 0.70
    - Performance is similar to CatBoost only considering zip_code_rank
  - Feature importance analysis showed many variables common to both models- AssesTot, gross_sqft, zip_code_rank, YCoord, Rolling avgs (3m, 1y), S&P
- Real-World Applications:
  - High performance home price estimating tools can be used by Zillow, Opendoor, and Trulia to provide better data to consumers
- Future Directions:
  - Different models for different price ranges may be appropriate
  - Applicability of model to different geographies and future time periods is limited
  - Future work may consider inflation-adjusted prices for previous years
  - Future work may also consider tracking a set of homes through time (time series analysis)
  - Future work may consider Named Entity Recognition (NER) in the neighborhoods and addresses to identify landmarks in addresses and use them as predictors