Prediction of Flood Insurance Premium for Houston Area

Cheng Cheng
Springboard Data Science Track Capstone

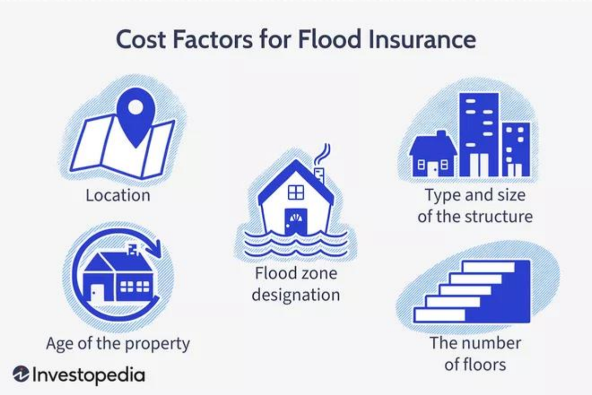
Outline

- Background & Objective
- Data Acquisition & Cleaning
- Data Wrangling & Exploratory Data Analysis
  - Missing values, Categorical features, Numeric features
- Pre-processing & Training
- Machine Learning Modeling
  - Split and scale dataset
  - Models: K-nearest Neighbors regression, Random Forest regression, Extreme Gradient Boosting, Catboost regression
- Summary

Background

- Flood can happen anywhere, and most homeowners' insurance doesn't cover flood damage.
- Flood insurance is a separate policy to cover the building, contents, or both.
- Knowing flood insurance premium is of great interest to the homeowners if their property has flooding risk.
- Who might care:
    - Policy holder
    - Insurance company
    - State and local government agencies
    - Non-profit groups
- **Problem: Can we predict the flood insurance premium based on the characteristics of the building and other factors?**

Data Acquisition



- The National Flood Insurance Program (NFIP), managed by the Federal Emergency Management Agency (FEMA), offers flood insurance to homeowners in participating communities.
- The program has the insurance policy information of more than 23,000 participating NFIP communities across the U.S.
- The dataset is updated every 40 to 60 days. For this study, the dataset was downloaded from NFIP website in May 2022, which includes the policy information from 2009 to 2019.
- Each policy includes the building factors (flood zone, number of floors, location, age, etc.), building and content coverage amount, and insurance premium price.

Data Source: https://www.fema.gov/openfema-data-page/fima-nfip-redacted-policies-v1

Data Cleaning

- The NFIP dataset has the policy information for > 23,000 communities across the U.S., which is not reasonable to use all the data given the computing and modeling resource demands.
- Houston has a long history of extreme rainfall events, including Hurricane Harvey in 2017, so focus is on the insurance premium prediction for Houston area.
- Preliminary exploration of the dataset shows there are more than 100,000 records for Houston area in 2019, which has enough data for machine learning modeling and prediction.

Data Wrangling

OpenFEMA Dataset: FIMA NFIP Redacted Policies - v1

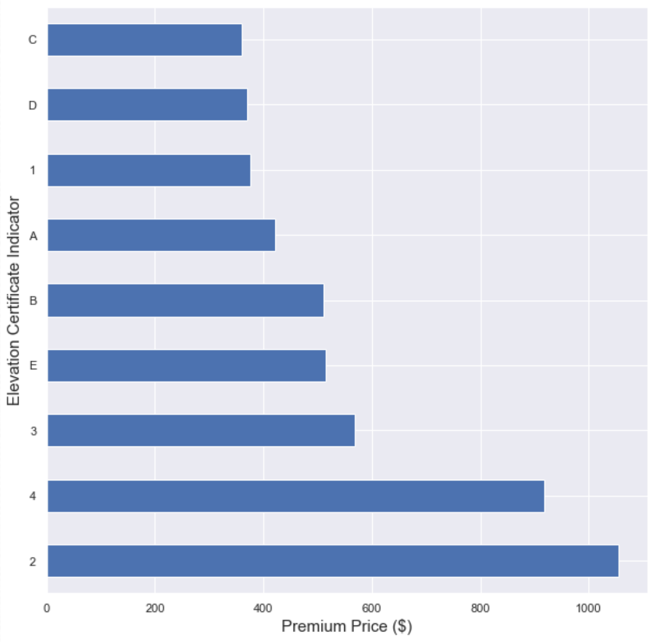
- The dataset has 152,557 unique records with 45 variables, including the target: insurance premium price
  - Data types: 18 object, 5 datetime, 11 float, and 11 integer
- The explanation of each variable is obtained from: https://www.fema.gov/openfema-data-page/fima-nfip-redacted-policies-v1

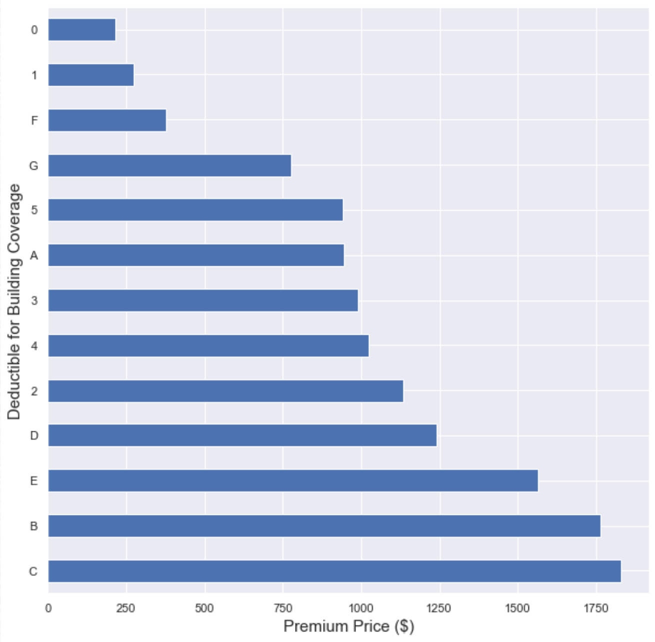
	agriculturestructureindicator	basefloodelevation	basementenclosurecrawlspacetype	cancellationdateoffloodpolicy	reportedcity	smallbusinessindicatorbuilding	totalbuildinginsurancecoverage	totalcontentsinsurancecoverage	totalinsurancepremiumofthepolicy
0	N	NaN	0.0	NaT	HOUSTON	N	250000	100000	376
1	N	NaN	0.0	NaT	HOUSTON	N	200000	80000	353
2	N	NaN	0.0	NaT	HOUSTON	N	250000	100000	374
3	N	NaN	0.0	NaT	HOUSTON	N	250000	100000	376
4	N	NaN	0.0	NaT	HOUSTON	N	250000	100000	376
5	NaN	113.8	0.0	NaT	HOUSTON	N	199700	55100	726
6	N	83.1	0.0	NaT	HOUSTON	Y	500000	275600	2588
7	N	NaN	0.0	NaT	HOUSTON	N	150000	60000	320
8	NaN	NaN	0.0	NaT	HOUSTON	N	250000	100000	376
9	N	NaN	0.0	NaT	HOUSTON	N	91000	0	924

Data Wrangling & Exploratory Data Analysis

- Check missing values
  - 6 variables have more than 70% missing values.
  - These variables may be removed from machine learning model but will be decided later based on their relationships to the target.
  - Some other variables can be deleted certainly, such as city name, state name, zip code, etc.
- Analyze categorical features
  - Check how many unique values for each class.
  - Develop premium price plots (medium price) for each categorical variable to get the idea of whether there is a relationship between each class and premium price value.
- Analyze numeric features
  - Some numeric variables are categorical data types, such as number of floors, etc.
  - Develop histogram plots to get the idea of how they distribute.
  - Create new variables such as building age (using year 2019 – building construction year), and total coverage (using building coverage + contents coverage).

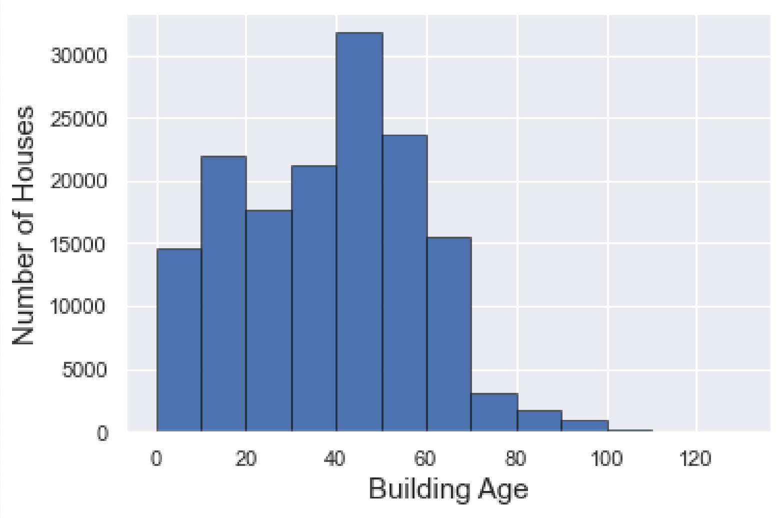
Data Wrangling & Exploratory Data Analysis

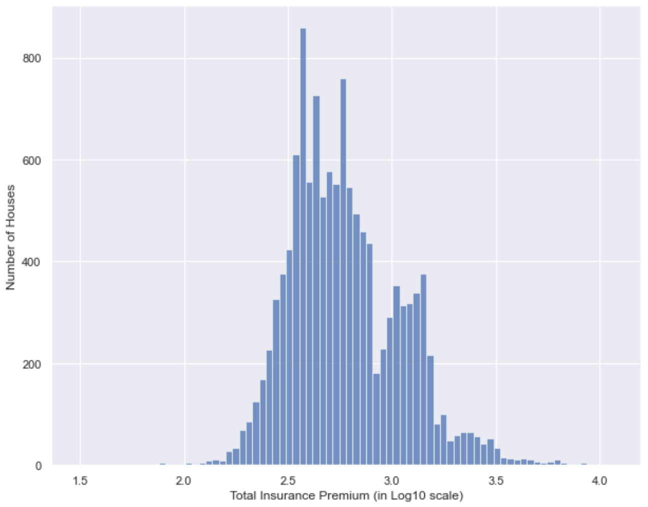




Example Medium Premium Price Plots for Categorical Variables

Data Wrangling & Exploratory Data Analysis

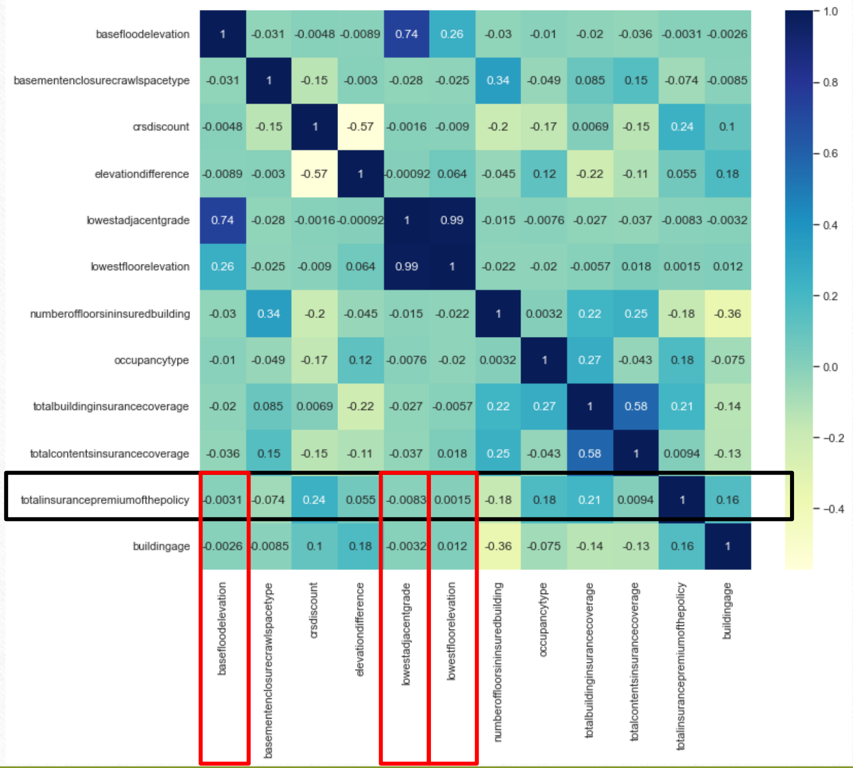




Example Histogram Plots for Numeric Variables

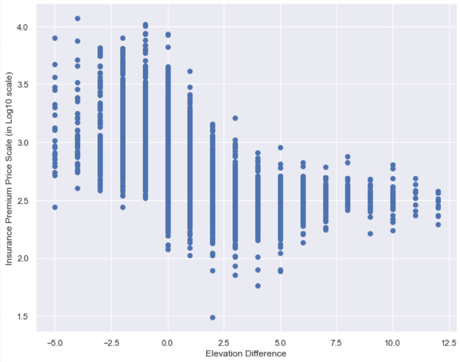
Data Wrangling & Exploratory Data Analysis

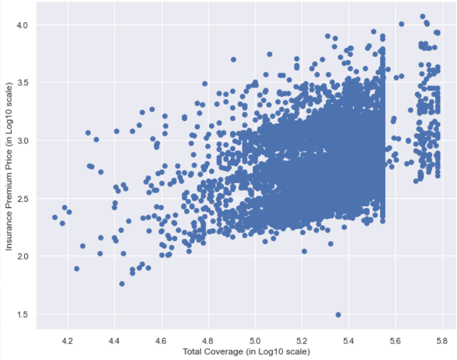
- Check correlations between numeric features using heatmap
  - The correlation coefficients between the target and a few numeric features are very small.
  - These numeric features have a lot of missing values (>70%).
  - Therefore, these numeric features are removed from machine learning model given that they won't provide many benefits to modeling, and they have many missing values.



Data Wrangling & Exploratory Data Analysis

- Categorical features
  - Use dummy encoding for the nominal categorical variables.
  - Use ranking order approach for the ordinal categorical variables
  - Important categorical variables include:
    - Flood zone, Location of contents, Number of floors, etc.
- Numeric features
  - Keep the numeric data within 95% confidence interval (2.5% to 97.5) to remove the extreme outliers.
  - Use log scale for premium price and total coverage because their values have a range of several orders of magnitude.
- Finalize dataset for modeling
  - Check any missing values left, remove the variables which are not used in modeling, and delete duplicate data.
  - The final dataset has 12,332 records with 30 variables, including the target feature.





Preprocessing and Training

- Train/test dataset split.

```
train, test = train_test_split(df, test_size=0.3, random_state=42, shuffle=True)
```

- Use Pycaret to determine the best models.

```
from pycaret.regression import *
s = setup(data=train, target='premium_scale', fold_shuffle=True, session_id=123)
```

- Catboost regression model gives the best R2, MAE, MSE, and RMSE.

- Other good models include extreme gradient boosting, extra trees regression, light gradient boosting, random forest, decision tree regression, etc.

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
catboost	CatBoost Regressor	0.0211	0.0020	0.0445	0.9719	0.0116	0.0076	1.0500
xgboost	Extreme Gradient Boosting	0.0241	0.0026	0.0504	0.9642	0.0132	0.0087	0.2190
et	Extra Trees Regressor	0.0252	0.0031	0.0556	0.9565	0.0145	0.0091	0.3980
lightgbm	Light Gradient Boosting Machine	0.0263	0.0033	0.0568	0.9547	0.0147	0.0094	0.1800
rf	Random Forest Regressor	0.0271	0.0036	0.0594	0.9502	0.0154	0.0097	0.3940
gbr	Gradient Boosting Regressor	0.0336	0.0039	0.0618	0.9466	0.0159	0.0119	0.1340
dt	Decision Tree Regressor	0.0325	0.0053	0.0725	0.9260	0.0191	0.0117	0.0580
knn	K Neighbors Regressor	0.0569	0.0111	0.1049	0.8478	0.0271	0.0203	0.0900
ada	AdaBoost Regressor	0.0993	0.0175	0.1320	0.7591	0.0348	0.0360	0.1200
br	Bayesian Ridge	0.1105	0.0218	0.1476	0.6995	0.0401	0.0398	0.0900
lr	Linear Regression	0.1105	0.0218	0.1476	0.6995	0.0401	0.0398	1.2530
ridge	Ridge Regression	0.1106	0.0218	0.1476	0.6994	0.0401	0.0399	0.0170
lar	Least Angle Regression	0.1106	0.0219	0.1479	0.6980	0.0402	0.0399	0.0570
par	Passive Aggressive Regressor	0.1343	0.0335	0.1812	0.5335	0.0492	0.0484	0.0290
omp	Orthogonal Matching Pursuit	0.1437	0.0360	0.1896	0.5051	0.0499	0.0515	0.0070
huber	Huber Regressor	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1270
lasso	Lasso Regression	0.2152	0.0729	0.2698	-0.0016	0.0705	0.0773	0.0200
en	Elastic Net	0.2152	0.0729	0.2698	-0.0016	0.0705	0.0773	0.0090
llar	Lasso Least Angle Regression	0.2152	0.0729	0.2698	-0.0016	0.0705	0.0773	0.0070

Machine Learning Model – Split and Scale Dataset

- Train/test dataset split.

```
X = df.drop(columns='premium_scale')
y = df.premium_scale
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=42)
```

- Use standard scaler to scale the train and test datasets.

```
scaler = StandardScaler()
X_train[['crsdiscount', 'deductibleamountinbuildingcoverage', 'deductibleamountincontentscoverage',
        'elevationdifference', 'numberoffloorsininsuredbuilding', 'buildingage',
         'coverage_scale']] = scaler.fit_transform(X_train[['crsdiscount',
         'deductibleamountinbuildingcoverage', 'deductibleamountincontentscoverage',
          'elevationdifference', 'numberoffloorsininsuredbuilding', 'buildingage', 'coverage_scale']])

X_test[['crsdiscount', 'deductibleamountinbuildingcoverage', 'deductibleamountincontentscoverage',
        'elevationdifference', 'numberoffloorsininsuredbuilding', 'buildingage',
        'coverage_scale']] = scaler.transform(X_test[['crsdiscount',
        'deductibleamountinbuildingcoverage', 'deductibleamountincontentscoverage',
        'elevationdifference', 'numberoffloorsininsuredbuilding', 'buildingage', 'coverage_scale']])
```

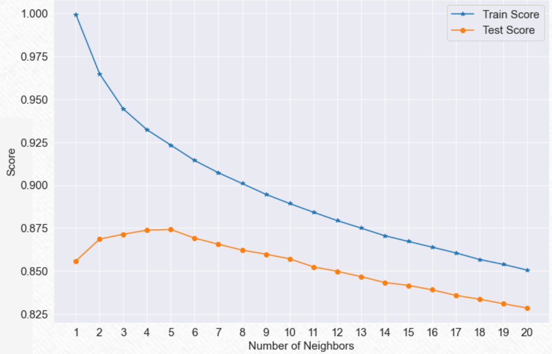
Machine Learning Model – K-nearest Neighbors

- Use GridSearchCV to tune the hyperparameters.

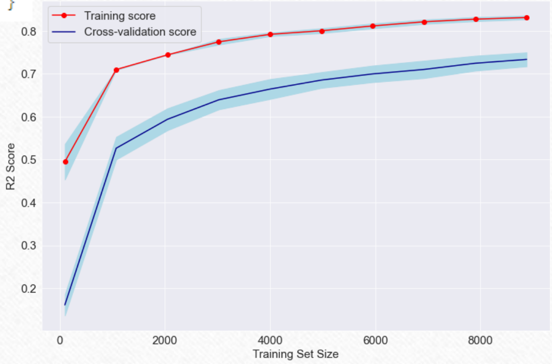
```
knn_reg = KNeighborsRegressor()
param_grid = {
    'weights': ['uniform', 'distance'],
    'n_neighbors': [5, 8, 10, 12],
    'algorithm': ['auto', 'ball_tree', 'kd_tree'],
    'p': [1, 2],
    'leaf_size': [20, 30, 40]
}
GSCV_knn = GridSearchCV(estimator=knn_reg, param_grid=param_grid, cv=5)
GSCV_knn.fit(X_train, y_train)
print("Best parameters:", GSCV_knn.best_params_)
```

```
Best parameters: {'algorithm': 'ball_tree', 'leaf_size': 30, 'n_neighbors': 5, 'p': 1, 'weights': 'distance'}
```

- A good fit between predicted and actual premium price – R2 is 0.9 and MSE (mean squared error) is 0.008.
- Both train and test datasets have low bias (i.e. high accuracy rates) and low variance.



Learning Curve for K-nearest Neighbors Regression

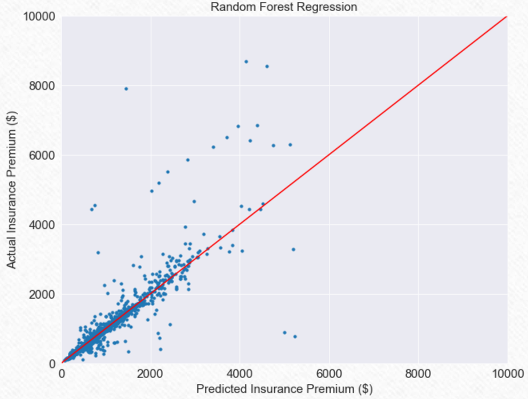


Machine Learning Model – Random Forest

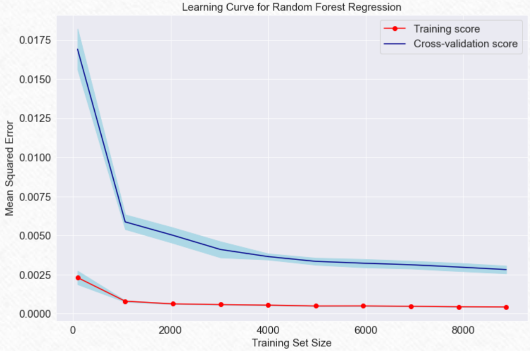
- Use GridSearchCV to tune the hyperparameters.

```
rf_reg = RandomForestRegressor(random_state=123)
param_grid = {
    'bootstrap': [True, False],
    'n_estimators': [100, 200, 300],
    'max_depth': [None, 4, 6, 8],
    'max_features': ['auto', 'sqrt', 'log2'],
}
GSCV_rf = GridSearchCV(estimator=rf_reg, param_grid=param_grid, cv=5)
GSCV_rf.fit(X_train, y_train)
print("Best parameters:", GSCV_rf.best_params_)
```

```
Best parameters: {'bootstrap': True, 'max_depth': None, 'max_features': 'auto', 'n_estimators': 300}
```

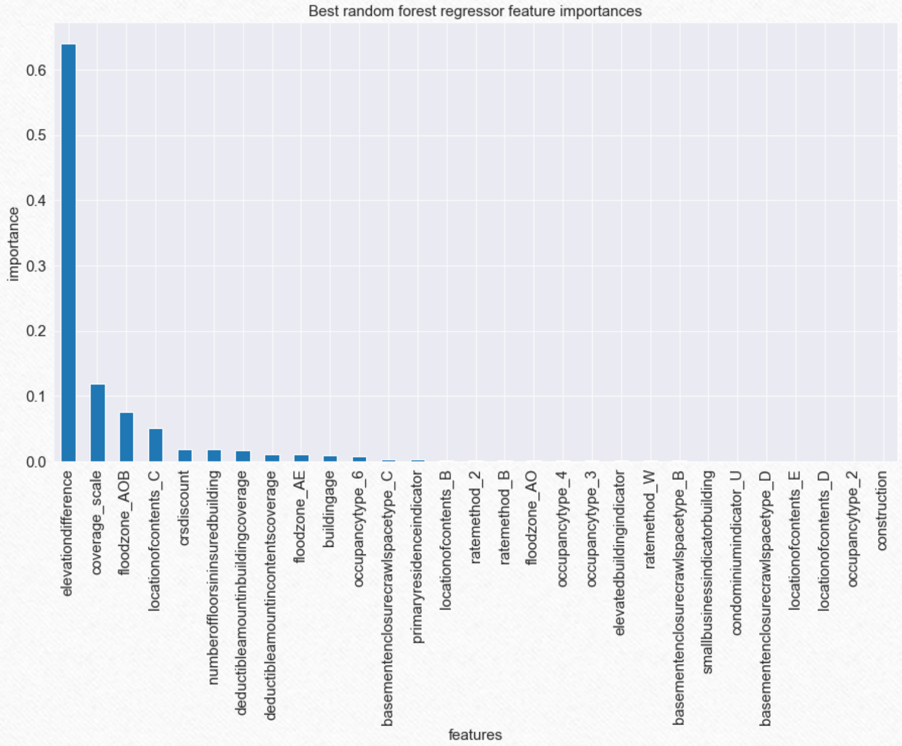


- Predicted and actual premium prices match very well.
    - R2 is 0.95 and MSE (mean squared error) is 0.004.
- MSE values decrease with the number of training set size for both training and test datasets, and they are converging to a very small value for both datasets.



Machine Learning Model – Random Forest

- Use feature importance function to determine the top five factors affecting the premium price:
    - **Elevation difference**: Difference between the elevation of the lowest floor and the base flood elevation
    - **Coverage**: Total insurance coverage for both building and contents
    - **Flood zone**: NFIP specified flood zones used to rate the property
    - **Location of contents**: The location where the contents are located within the structure
    - **CRS discount**: The Community Rating System (CRS) flood insurance policy premium discount

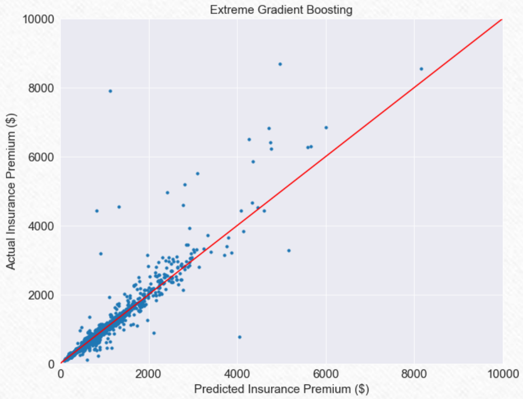


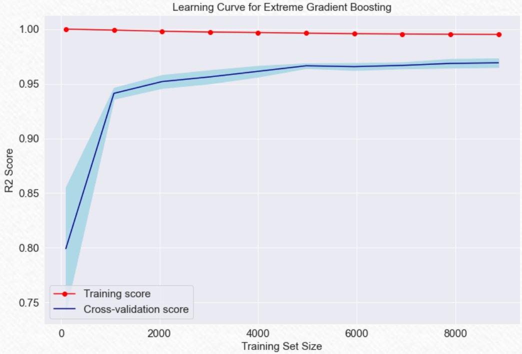
Machine Learning Model – Extreme Gradient Boosting

- Use RandomizedSearchCV to tune the hyperparameters.

```
xgbr = xgb.XGBRegressor()

param_grid = {'max_depth': [3, 5, 6, 10, 15, 20],
           'learning_rate': [0.01, 0.1, 0.2, 0.3],
           'subsample': np.arange(0.5, 1.0, 0.1),
           'colsample_bytree': np.arange(0.4, 1.0, 0.1),
           'colsample_bylevel': np.arange(0.4, 1.0, 0.1),
           'n_estimators': [100, 200, 300]
            }

clf = RandomizedSearchCV(estimator=xgbr,
                         param_distributions=param_grid,
                         scoring='neg_mean_squared_error',
                         n_iter=25)

clf.fit(X_train, y_train)
```

- A very good fit between predicted and actual premium price – R2 is 0.97 and MSE is 0.002.
- The learning curve shows that the R2 scores remain very high (close to 1) for the training dataset and increase with the train set size for the test dataset.





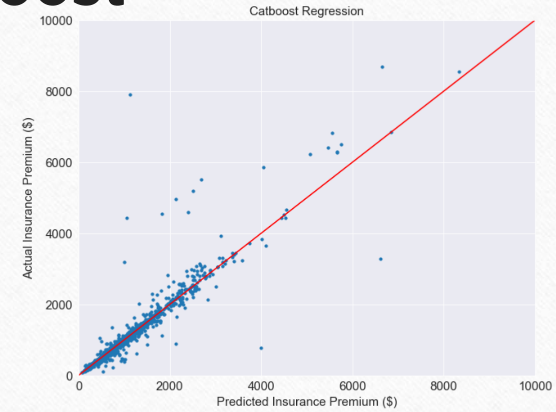
Machine Learning Model – Catboost

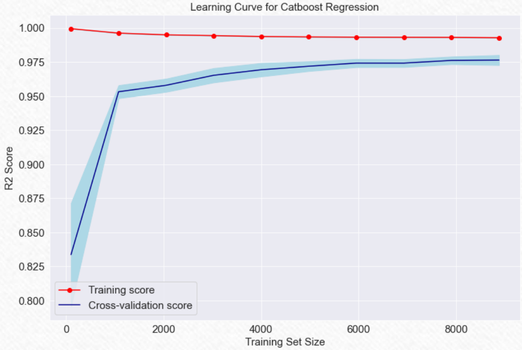
- Use GridSearchCV to tune the hyperparameters.

```
cat_reg = cat.CatBoostRegressor()
param_grid = {
    'depth': [6, 8, 10],
    'learning_rate': [0.01, 0.05, 0.1],
    'iterations': [100,500,1000],
}
GSCV_cat = GridSearchCV(estimator=cat_reg, param_grid=param_grid, cv=5, n_jobs=-1)
GSCV_cat.fit(X_train, y_train, verbose=False)
print("Best parameters:", GSCV_cat.best_params_)
```

```
Best parameters: {'depth': 6, 'iterations': 1000, 'learning_rate': 0.1}
```

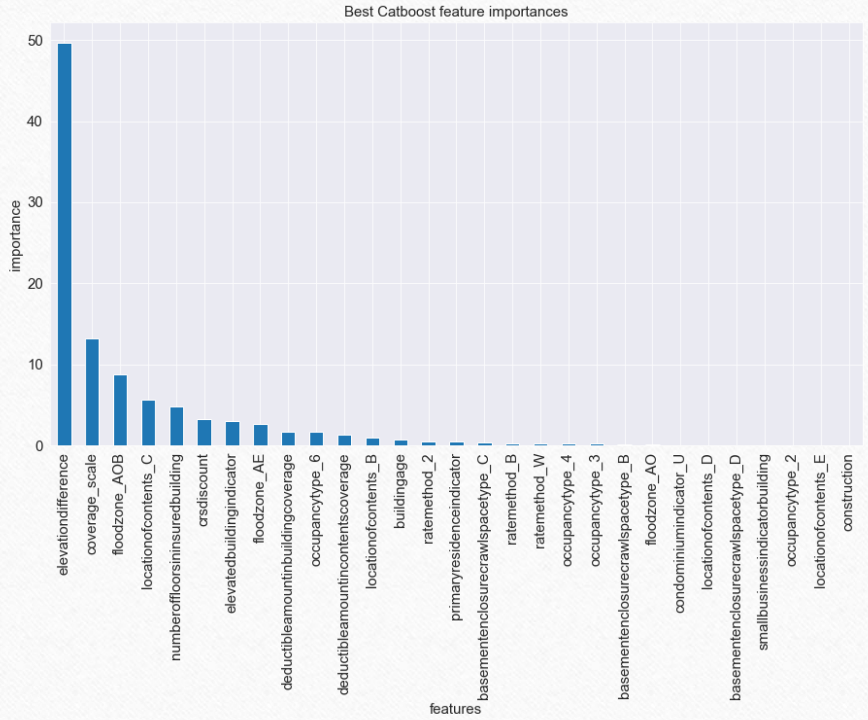
- A very good fit between predicted and actual premium price – R2 is 0.97 and MSE is 0.002.
- The learning curve shows that the R2 scores remain very high (close to 1) for the training dataset and increase with the train set size for the test dataset.





Machine Learning Model – Catboost

- Use feature importance function to determine the top five factors affecting the premium price:
    - **Elevation difference**: Difference between the elevation of the lowest floor and the base flood elevation
    - **Coverage**: Total insurance coverage for both building and contents
    - **Flood zone**: NFIP specified flood zones used to rate the property
    - **Location of contents**: The location where the contents are located within the structure
    - **Number of floors**: The number of floors for the property



Conclusion

- Four different machine learning models were used to evaluate the dataset.
    - Models: K-nearest Neighbors, Random Forest, Extreme Gradient Boosting, Catboost
    - Catboost, Extreme Gradient Boosting, and Random Forest perform slightly better than K-nearest neighbors based on lower mean squared error and higher R2 values.
    - The learning curves shows that there is neither overfitting nor underfitting issue for the train and test datasets - both datasets have low bias and low variance.
- Top factors affecting the flood insurance premium price are:
    - Elevation difference
    - Insurance coverage
    - Flood zone
    - Location of contents
    - Number of floors
    - CRS discount

Summary

- NFIP, managed by FEMA, provides flood insurance policy information for cities across the U.S..
- The 2019 dataset for Houston area obtained from FEMA open dataset is used to predict the flood insurance premium price.
- Data wrangling and exploratory data analysis are conducted to clean the original dataset, analyze relationships among features, handle missing values, remove the features which are not used, and finalize the dataset used in the machine learning modeling.
- Four machine learning models are used to evaluate the data.
    - Catboost, Extreme Gradient Boosting, and Random Forest give very good prediction whereas K-nearest neighbors performs slightly worse.
    - Elevation difference, insurance coverage, and flood zone are determined to be the top factors.
- Future Steps: ensemble model.