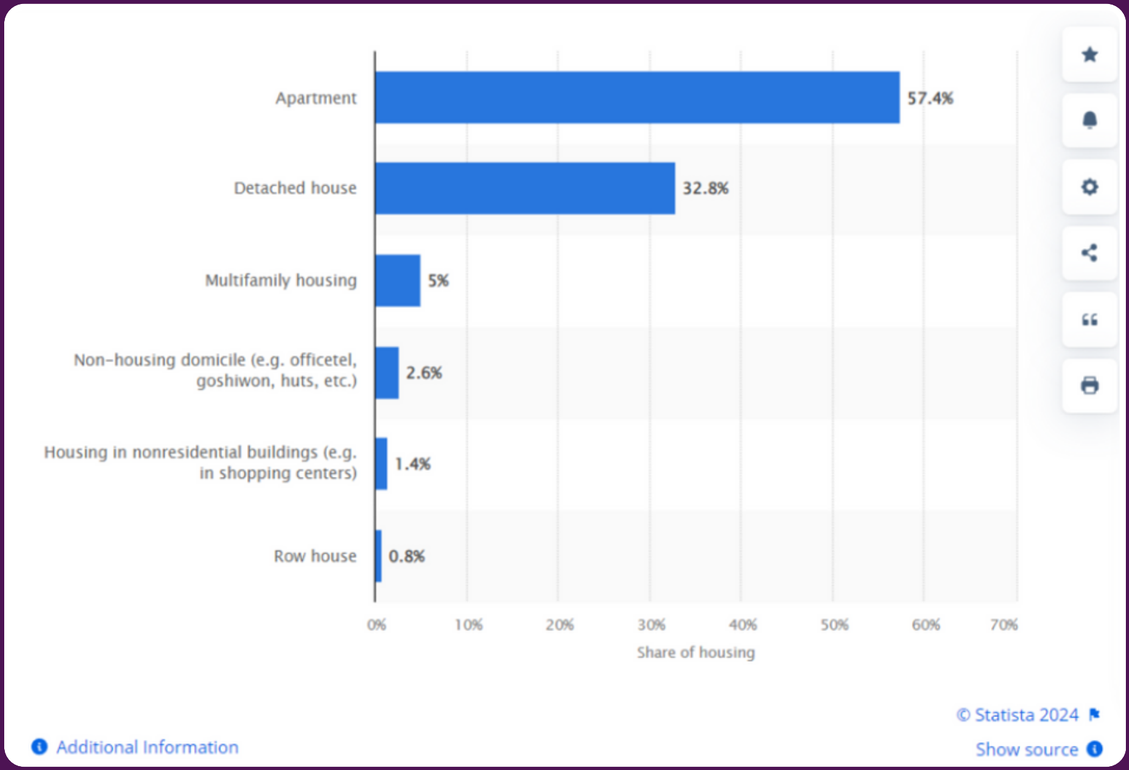
Machine Learning Project

Predicting Apartment Prices in the Daegu Area, South Korea using Machine Learning Analysis

Abyatar Fanuel Lantera

Background



Daegu is the **4th largest** metropolitan area in **South Korea**, and is officially called Daegu Metropolitan City. According to data released by the Korea Ministry of Land, Infrastructure, Transport and Tourism, the volume of housing transactions in regional cities such as **Daegu**, Ulsan and Daejeon doubled last month **(July)** compared to the beginning of the year **(2023)**. In Daegu, apartment transactions reached 1,890 units in the past month, up **111%** from January's **895 units**.

Apart from that, the **share of housing in Daegu**, South Korea in 2021 based on type is **dominated by apartments (57.4%)**. Seeing how profitable it is to invest in housing (especially apartments), **knowing what factors can influence the selling price of apartments in Daegu is very important** for **property owners or real estate agents** in order to increase the attractiveness of potential buyers and get **maximum profits**.

Project Outline

Problem Statement

Some real estate agents and property owners have **difficulty determining the optimal price for the apartment they want to sell**. This is because there are many factors that must be considered in determining the right price, such as investing in areas where development factors are favorable, such as subway stations that pass through popular areas, land value, floor area ratio, etc. **Accuracy** in determining prices is very important to maintain a balance between **profits and market attractiveness**.

Project Goals

The aim of this project is to **create a predictive model using Machine Learning Algorithms to determine the appropriate selling price for an apartment** based on the characteristics of the apartment itself (land area, distance to the station, year built, etc.). **This model is expected to help stakeholders make the right decisions regarding price**s, investment and property development.

Analytic Approach

This research will involve several features in an apartment and will look at their relationship with the price of the apartment itself. Because the target of this research is **numerically continuous apartment prices, a regression analysis approach will be used**. This model will use apartment features as predictors to estimate the selling price.

Project Limitation

- This model can **only predict apartment prices in Daegu city**, and cannot be used as a benchmark for prices in other areas.
- This model only uses data at one time, **it cannot forecast prices in the future**, because there could be changes in trends at times in the future.
- This model only considers the features in the dataset and **cannot capture phenomena outside of the existing features**, especially social, cultural and governmental factors.

Evaluation Metrics

Regression Models can be evaluated using **Mean Absolute Error** (MAE), **Mean Absolute Percentage Error** (MAPE) and **Root Mean Squared Error** (RMSE). This metric **focuses on errors and provides insight into the accuracy of predictions, with lower evaluation values indicating a more accurate model**. Apart from these metrics, the regression model can also be measured using **R-Squared to see the proportion of variance explained by the model**.

Metrics	What	Why	When	Example
MAE	Average absolute difference between estimated and actual values.	Less sensitive to outliers.	With many outliers or non-normal residuals.	in house pricing, if you're off by 20.000 or 40.000, MAE treats these errors linearly (without considering their direction).
MAPE	Percentage error between estimated and actual values.	Easy interpretation as a percentage.	For forecasting and percentage-based error analysis.	For instance, if a house is worth 200.000 and your predict 180.000, the error is 10%.
RMSE	Square root of MSE, in same units as response variable.	Easier interpretation of errors.	When error scale should match target scale.	if RMSE is 20.000 it means the typical prediction error of the price is about 20.000.
R-Squared	Proportion of variance explained by the model.	Indicates model's explanatory power.	To evaluate linear regression models' fit.	in predicting house prices, a high R-Squared would indicate that your model captures most of the variability in house prices.

The Data

Data Dictionary

No.	Column Name	Description
1	Hallway Type	Apartment type
2	TimeToSubway	Time needed to the nearest subway station
3	SubwayStation	The name of the nearest subway station
4	N_FacilitiesNearBy(ETC)	Number of additional facilities near the property (e.g. parks, shopping centers)
5	N_FacilitiesNearBy(PublicOffice)	The number of public office facilities nearby
6	N_SchoolNearBy(University)	The number of universities nearby
7	N_Parkinglot(Basement)	Number of parking spaces in the basement of the property
8	YearBuilt	The year the apartment was built
9	N_FacilitiesInApt	Number of facilities in the apartment
10	Size(sqft)	The apartment size (in square feet)
11	SalePrice	The apartment price (Won)

Dataset Sample

	HallwayType	TimeToSubway	SubwayStation	N_FacilitiesNearBy(ETC)	N_FacilitiesNearBy(PublicOffice)	N_SchoolNearBy(University)	N_Parkinglot(Basement)	YearBuilt	N_FacilitiesInApt	Size(sqf)	SalePrice
0	terraced	0-5min	Kyungbuk_uni_hospital	0.0	3.0	2.0	1270.0	2007	10	1387	346017
1	terraced	10min~15min	Kyungbuk_uni_hospital	1.0	5.0	1.0	0.0	1986	4	914	150442
2	mixed	15min~20min	Chil-sung-market	1.0	7.0	3.0	56.0	1997	5	558	61946
3	mixed	5min~10min	Bangoge	5.0	5.0	4.0	798.0	2005	7	914	165486
4	terraced	0-5min	Sin-nam	0.0	1.0	2.0	536.0	2006	5	1743	311504
...	...	...	...	...	...	...	...	...	...	...	...
4118	terraced	0-5min	Sin-nam	0.0	3.0	2.0	475.0	2008	8	914	323008
4119	mixed	15min~20min	Myung-duk	5.0	6.0	5.0	536.0	1993	4	1451	242477
4120	mixed	15min~20min	Myung-duk	5.0	6.0	5.0	536.0	1993	4	1761	168141
4121	corridor	5min~10min	Daegu	2.0	5.0	0.0	76.0	1985	3	676	73451
4122	terraced	0-5min	Kyungbuk_uni_hospital	0.0	3.0	2.0	1270.0	2007	10	868	250442

4123 rows × 11 columns

The Data

Data Summary

	Features	Data Type	Null Value (%)	Unique Value	Duplicate Data	Negative Value	Outlier(%)	Unique Sample
0	HallwayType	object	0.0	3	1422	0.0	0.000	[terraced, mixed, corridor]
1	TimeToSubway	object	0.0	5	1422	0.0	0.000	[0-5min, 10min~15min, 15min~20min, 5min~10min,...
2	SubwayStation	object	0.0	8	1422	0.0	0.000	[Kyungbuk_uni_hospital, Chil-sung-market, Bang...
3	N_FacilitiesNearBy(ETC)	float64	0.0	4	1422	0.0	0.000	[0.0, 1.0, 5.0, 2.0]
4	N_FacilitiesNearBy(PublicOffice)	float64	0.0	8	1422	0.0	0.000	[3.0, 5.0, 7.0, 1.0, 4.0, 2.0, 6.0, 0.0]
5	N_SchoolNearBy(University)	float64	0.0	6	1422	0.0	0.000	[2.0, 1.0, 3.0, 4.0, 5.0, 0.0]
6	N_Parkinglot(Basement)	float64	0.0	20	1422	0.0	0.000	[1270.0, 0.0, 56.0, 798.0, 536.0, 605.0, 203.0...
7	YearBuilt	int64	0.0	16	1422	0.0	0.000	[2007, 1986, 1997, 2005, 2006, 2009, 2014, 199...
8	N_FacilitiesInApt	int64	0.0	9	1422	0.0	0.000	[10, 4, 5, 7, 2, 9, 8, 1, 3]
9	Size(sqf)	int64	0.0	89	1422	0.0	0.703	[1387, 914, 558, 1743, 1334, 572, 910, 288, 11...
10	SalePrice	int64	0.0	838	1422	0.0	0.194	[346017, 150442, 61946, 165486, 311504, 118584...

This table is a brief summary of the data in each column of the dataset.

- **Data Types**: There are numeric (int64 & float64) and categorical (object) data types.
- **Null Values**: No columns have null values in this dataset
- **Unique Values**: See how many unique values there are in a column
- **Duplicate Data**: There are 1422 rows of data which are duplicates. We will examine this duplicate data further, so that there is no bias in the final results.
- **Negative Values**: There is no negative data in this data set
- **Outliers**: There are outliers in the Size (sqf) and Selling Price columns. Even though the percentage is very small (less than 1%), we will still look at the outlier data further.

In general, this dataset can be said to be quite good because there are not too many data problems that need to be handled.

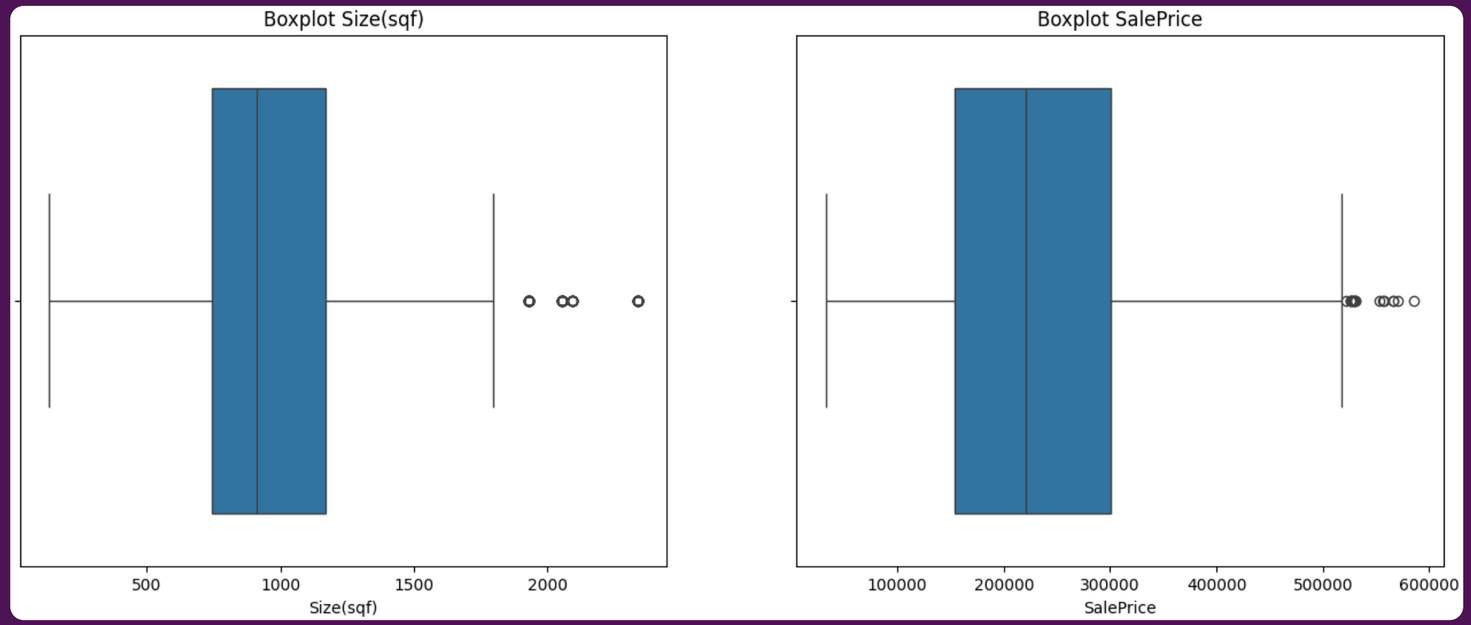
Handling Data Anomalies

Duplicated Data

- If you use the **.duplicated()** function **in pandas without specifying a subset**, it considers all columns. **A row is marked as duplicated only if all its column values are identical to those of another row**.
- Duplicate data can have **detrimental effects** on **machine learning models** and outcomes, such as **reducing data diversity and representativeness**, which can lead to **overfitting** or **biased models**.
- Therefore, it is best to **eliminate data that has duplicates.**

Handling Data Anomalies

Outliers Data



- It is important to find outliers and remove them from the dataset as part of the feature engineering before training machine learning algorithms for predictive modeling.
- Outliers present in a classification or regression dataset can lead to lower predictive modeling performance.
- Since the percentage of outliers is not very large (below 5%), we will not include the outlier data when training the model.

Data Preprocessing & Feature Engineering

	Size(sqf)	N_FacilitiesNearBy(ETC)	N_FacilitiesNearBy(PublicOffice)	N_SchoolNearBy(University)	N_Parkinglot(Basement)	N_FacilitiesInApt	HallwayType_corridor	HallwayType_mixed	HallwayType_terraced	SubwayStation_Bangoge
0	1.174877	-0.2	-0.5	0.0	1.189627	1.666667	0.0	0.0	1.0	0.0
1	0.009852	0.0	0.5	-0.5	-0.868720	-0.333333	0.0	0.0	1.0	0.0
2	-0.866995	0.0	1.5	0.5	-0.777958	0.000000	0.0	1.0	0.0	0.0
3	0.009852	0.8	0.5	1.0	0.424635	0.666667	0.0	1.0	0.0	1.0
4	2.051724	-0.2	-1.5	0.0	0.000000	0.000000	0.0	0.0	1.0	0.0
...	...	...	...	...	...	...	...	...	...	...
2606	-0.226601	-0.2	-1.5	0.0	0.000000	0.000000	0.0	0.0	1.0	0.0
2607	2.096059	0.8	1.0	1.5	0.000000	-0.333333	0.0	1.0	0.0	0.0
2608	0.009852	-0.2	-0.5	0.0	-0.098865	1.000000	0.0	0.0	1.0	0.0
2609	1.332512	0.8	1.0	1.5	0.000000	-0.333333	0.0	1.0	0.0	0.0
2610	-0.576355	0.2	0.5	-1.0	-0.745543	-0.666667	1.0	0.0	0.0	0.0

2611 rows × 20 columns

In this dataset, we will divide the features into several categories based on the characteristics of the data. The purpose of doing this is to make it easier when you want to do feature engineering.

- **Numeric features**: ('Size(sqf)', 'N_FacilitiesNearBy(ETC)', 'N_FacilitiesNearBy(PublicOffice)', 'N_SchoolNearBy(University)', 'N_Parkinglot(Basement)', 'N_FacilitiesInApt')
- **Categorical features**: ('HallwayType', 'SubwayStation')
- **Ordinal features**: ('TimeToSubway')
- **YearBuilt** will be left alone (passthrough) because this data already has its own meaning and order based on year & more than that the data is discrete, only having 16 unique count values.

Data Preprocessing & Feature Engineering

Feature Engineering is the process of creating new features or changing existing features with the aim of improving model performance. The Feature Engineering process in this project uses the Pipeline library and the description is as follows:

- **RobustScaler**: **Robust Scaler** is a feature scaling technique based on statistics that is less sensitive to outliers (median and interquartile range) and is **commonly used for numeric features**. This scaling **helps the model from being affected by extreme values that might bias/misinterpret the model**.
- **OneHotEncoder**: **OneHotEncoder** is used to **convert categorical variables into a series of binary variables (numerical 1 or 0)** or can also be called dummy variables. The goal is to **facilitate the processing of categorical information without assuming order between (nominal) categories**.
- **OrdinalEncoder**: **OrdinalEncoder** translates values such as **'TimeToSubway'** into an **ordinal numerical scale according to a predefined mapping**. Ordinal features are **category features that have a sequence or level**.
- **Passthrough**: **Passthorugh** is used to **ignore features that you do not want to change (in pre-processing) because these features are already considered suitable for the model to be run**. In this model we leave the `YearBuilt` column as is.

Model Selection

The Regression Algorithm that will be used to compare models is:

- Linear regression
- Lasso Regression
- Ridge Regression
- Decision Tree Regressor
- KNN Regressor
- Random Forest
- Support Vector Regressor

Among the 7 algorithms, we will look for **2 algorithm that produces the best evaluation metrics, then we will tune the model**.

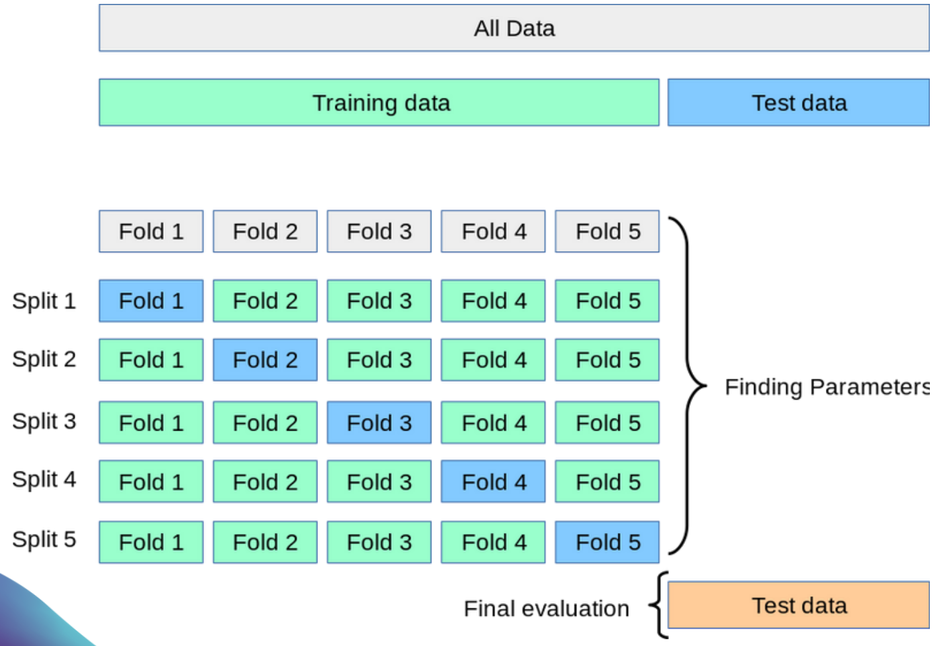
But before training the model, we will divide the dataset and divide it into **2 parts**, that is the **training set and the test set**. This is done so that we can distinguish which data the model uses to learn and which data the model uses to make predictions. Apart from that, **splitting data will be very useful because it aims to avoid information leakage**.

Model Selection

	Training R^2	Test R^2	Training MAE	Test MAE	Training MAPE	Test MAPE	Training RMSE	Test RMSE
Random Forest	0.811973	0.816057	35500.124031	35100.191186	0.184307	0.194029	43825.113066	43370.406742
Decision Tree	0.812751	0.812067	35384.255846	35164.103715	0.182647	0.194253	43734.389388	43838.302998
KNN	0.780951	0.791367	37257.525434	36957.529250	0.191021	0.202540	47302.412283	46189.587548
Ridge	0.756360	0.759163	40916.081923	40669.539049	0.214201	0.225990	49886.989618	49626.485507
Lasso	0.756411	0.759071	40900.878489	40682.325990	0.214385	0.226122	49881.727530	49636.057680
Linear Regression	0.756412	0.759052	40900.380308	40684.335617	0.214406	0.226138	49881.697588	49637.954142
SVR	-0.005486	-0.000007	82434.297109	83378.051781	0.540651	0.595025	101344.661649	101123.927351

Based on benchmarking models, **Random Forest** and **Decision Tree** are the 2 models that have the **best evaluation metrics values. Next, we will tune these 2 models**.

Hyperparameter Tuning



- **RandomizedSearchCV** randomly goes through a set of hyperparameters and calculates a score and gives the best set of hyperparameters that gives the best score as output. So here is the recipe How we can find optimal parameters using RandomizedSearchCV for Regression.
- Notice that **RandomizedSearchCV()** requires the extra **n_iter** argument, which determines how many random cells must be selected. **This determines how many times k-fold cross-validation will be performed**.
- **Cross-validation** is a resampling procedure used to evaluate machine learning models on a limited data sample. If you have a machine learning model and some data, you want to tell if your model can fit. You can split your data into training and test set.

Hyperparameter Tuning

Random Forest

Here are some hyperparameters in Random Forest that will be adjusted

```
# Number of trees in random forest
n_estimators = [100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 1100, 1200]
# Number of features to consider at every split
max_features = ['log2', 'sqrt', None]
# Maximum number of levels in tree
max_depth = [2, 3, 4, 5, 6, 7, 8, 9, 10, None]
# Minimum number of samples required to split a node
min_samples_split = [2, 5, 10]
# Minimum number of samples required at each leaf node
min_samples_leaf = [1, 2, 4]
# Max samples for bootstrapping
max_samples = [0.5, 0.7, 0.8, 1.0]
```

and here are the best parameter results

- **'n_estimators'**: 1000,
- **'min_samples_split'**: 5,
- **'min_samples_leaf'**: 1,
- **'max_samples'**: 1.0,
- **'max_features'**: 'log2',
- **'max_depth'**: 9

Decision Tree

Here are some hyperparameters in Decision Tree that will be adjusted

```
# Number of features to consider at every split
max_features = ['log2', 'sqrt', None]
# Maximum number of levels in tree
max_depth = [2, 3, 4, 5, 6, 7, 8, 9, 10, None]
# Minimum number of samples required to split a node
min_samples_split = [2, 5, 10]
# Minimum number of samples required at each leaf node
min_samples_leaf = [1, 2, 4]
```

and here are the best parameter results

- **'min_samples_split':** 2**,**
- **'min_samples_leaf':** 1**,**
- **'max_features':** 'sqrt'**,**
- **'max_depth':** 10

Compare Tuning Result

Random Forest

	Random Forest	Random Forest RandomizedSearchCV
R^2	0.816057	0.819379
MAE	35100.191186	34966.132851
MAPE	0.194029	0.193297
RMSE	43370.406742	42977.042072

Decision Tree

	Decision Tree	Decision Tree RandomizedSearchCV
R^2	0.812067	0.819319
MAE	35164.103715	34879.458423
MAPE	0.194253	0.191225
RMSE	43838.302998	42984.162292

- After performing hyper parameter tuning, the model using the **Random Forest algorithm was selected because it had a higher R-Squared value** (**0.819379**) than the Decision Tree (**0.819319**).
- **Random forests are generally more accurate** than individual decision trees **because they combine multiple trees and reduce overfitting**, **providing better predictive performance and robustness**.
- However, both Random Forest and Decision Tree experienced increases in all aspects of Evaluation Metrics after hyper parameter tuning was carried out.

Evaluation Metrics Interpretation

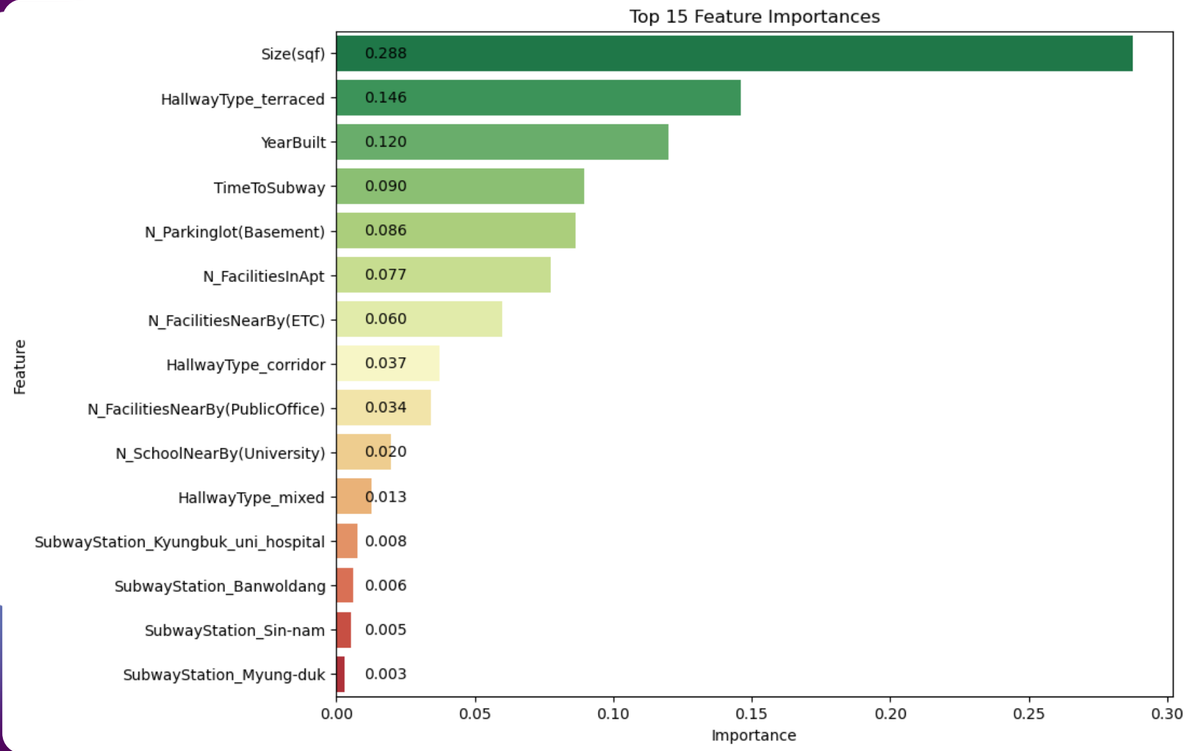
These are metrics used to evaluate the performance of Random Forest Regressor model in predicting apartment prices in Korean Won (₩). Here's how to interpret them:

1. **R^2 (Coefficient of Determination)**: This is 0.819379, which means that approximately **81.94%** of the variability in the apartment prices can be explained by the features in the model. **This is a relatively high R^2, suggesting that the model explains a large portion of the variation in apartment prices** & the model provides a reasonably good fit to the data.
2. **MAE (Mean Absolute Error)**: This is **34,966.132851**, which means that on average, your model's **predictions are about ₩34,966.13 off from the actual apartment prices**. This gives you an idea of the magnitude of the errors the model is making.
3. **MAPE (Mean Absolute Percentage Error)**: This is 0.193297, or **19.33%**. This means that on average, **the model's predictions are about 19.33% off from the actual apartment prices**. This gives you a relative measure of the errors the model is making.
4. **RMSE (Root Mean Square Error)**: This is **42,977.042072**, which is a measure of **the differences between the values predicted by the model and the values actually observed**. It's more sensitive to large errors (Outliers) than MAE because it squares the differences before averaging them. The RMSE being larger than the MAE suggests that your model is occasionally making large errors.

Notes:

- The MAE and the RMSE can be used together to diagnose the variation in the errors in a set of forecasts. The RMSE will always be larger or equal to the MAE; the greater difference between them, the greater the variance in the individual errors in the sample. If the RMSE=MAE, then all the errors are of the same magnitude **(₩42,977.04 - ₩34,966.13 = ₩8,010.91 => The difference between MAE and RMSE is not too big, meaning that it can be said that the variance error in the prediction results is not too big.).**
- Overall, these metrics suggest that the model is doing a reasonably good job of predicting apartment prices in Korean Won, but there is still room for improvement, particularly in reducing the size of the errors.

Feature Importances



These are the feature importances from the Random Forest model, which indicate how much each feature contributes to the predictions of the model.

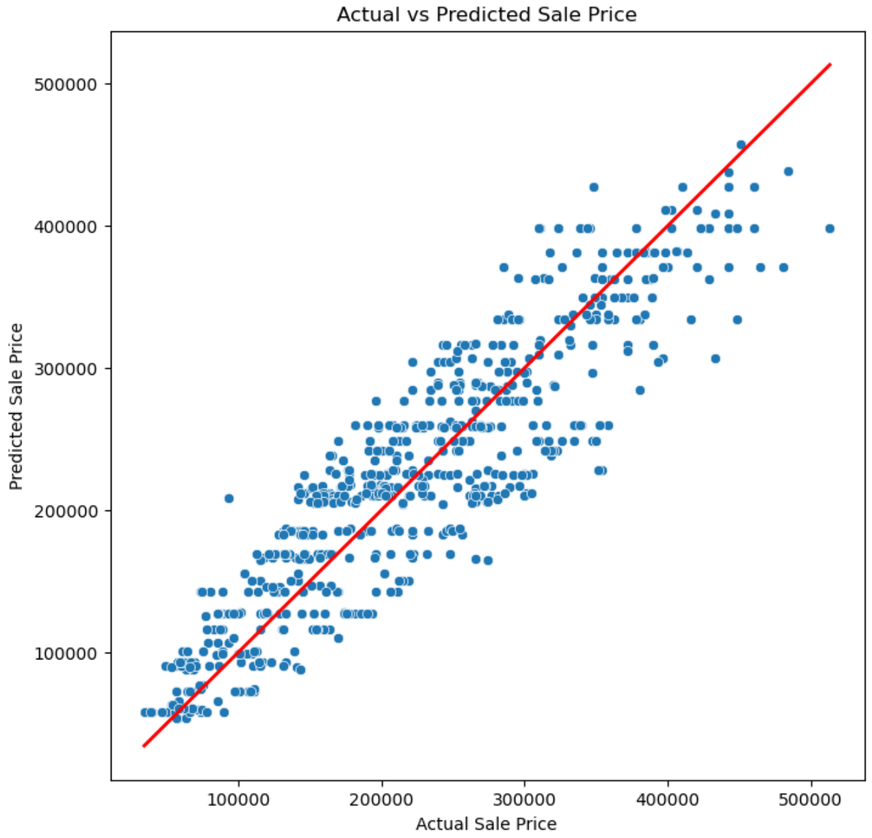
- **Size(sqf)**: This is the most important feature in your model, with an importance score of **0.287796**. This means that the **size of the property (in square feet) is the most significant factor in predicting the target variable**, which is the price of the apartment.
- **HallwayType_terraced**: This feature has the second highest importance score **(0.146190)**. This suggests that whether **the hallway type is terraced or not is the second most important factor in this model**. It seems that the type of hallway has a significant impact on the price of the apartment.
- **YearBuilt**: This feature has the third highest importance score **(0.119892)**. This suggests that **the year the property was built is the third most important factor in this model**. Newer apartments might be priced higher than older ones.
- **TimeToSubway**: This feature has an importance score of **0.089587**, making it the fourth most important feature. This suggests that the **time it takes to get to the subway from the property is a significant factor in this model**. Properties closer to the subway might be priced higher.
- and so on.

Notes:
Feature importance doesn't imply causality, it just tells us which features are most influential in the model's predictions. Also, the importance of a feature depends on the specific model and could change if you use a different model or change the model parameters.

Actual Price vs Predicted Price

and below are the results of the prices prediction when compared to the actual prices

	Actual Price	Predicted Price
1290	130088.0	116398.318680
1324	320353.0	287914.272698
799	210619.0	248251.249558
797	207079.0	259582.177386
1263	234513.0	284637.481943
...	...	...
1166	266814.0	225167.884180
1106	120796.0	168845.515456
1713	144690.0	182364.829318
914	75221.0	100676.986028
833	156637.0	166564.810151



For the code and other analysis, you can check out the following GitHub repositories:
https://github.com/AbyatarFL/Capstone-Machine-Learning-Apartment-Price-Prediction

Thank You

www.github.com/AbyatarFL

abyatarfl@gmail.com

www.linkedin.com/in/abyatarfanuel