1. Machine Learning Models

For this section, we utilized some Machine Learning models to attempt to learn from the dataset such as patterns within the data, feature importances, and some specific ML problems. There are 2 ML problems that we identified and targeted at in this project: regression of real estates’ prices and classification of property types. We experimented with some well-known and effective learning algorithms, including Linear Regression (LR), Support Vector Machine (SVM), Random Forest (RF), and Gradient Boosting (GB).

1. Methodology

Linear Regression is a supervised learning algorithm used for regression problems, capturing linear relationships of independent variables (also called features) and the dependent variable to be predicted. It models this relationship by fitting a linear hyperplane to the data, learning the coefficients by minimizing differences between predicted and target values using methods like least squares. Regularization methods are also used by adding L1- (Lasso) or L2-norms (Ridge) of coefficients to limit the domain of coefficients to be searched, allowing the model to avoid overfitting. We used the Ordinary Least Squares (OLS) LR, the L2 regularized Ridge and the L1 regularized Lasso for some basic models to see if our dataset follows any linear relationships.

Support Vector Machine is another supervised machine learning algorithm used for classification and regression tasks. It works by finding the hyperplane in higher dimensions or decision boundary that best separates data of different classes, maximizing the margin between them. SVM is effective in high-dimensional spaces and is particularly powerful for datasets that are not linearly separable, as it can use kernel functions to transform the data into a higher-dimensional space where a linear separation is possible. SVM is known for its robustness, especially in situations with complex or non-linear data, so we used the linear version of SVM to compare its performance with LR methods and used SVM with the Gaussian radial basis function as kernel function to attempt to capture more complex patterns.

We also used ensemble learning methods, which are popularly used due to its high performance and robustness. An ensemble learning model works by training multiple weak learners and combine their results to obtain the final predictions. The most common weak learner to be used is the Decision Tree as it provides simplicity, flexibility, and efficiency during training. Moreover, these trees are suited to be used in ensembles since they learn from different subsets of data or focus on correcting errors of previous trees, creating a much stronger overall model. Here, we made use of two ensemble learning models: Random Forest and Gradient Boosting.

Random Forest builds multiple trees during training and merges their results to improve accuracy and prevent overfitting. Each tree is trained on a random subset of the data, and at each split, a random subset of features is considered. This randomness helps make the model more robust and reduces the variance compared to individual decision trees. The final prediction is made by aggregating the outputs of all trees, typically by majority voting for classification or averaging for regression.

Gradient Boosting builds a strong predictive model by sequentially training trees, where each new tree attempts to correct the errors made by the previous ones. The model is built in a way that minimizes the residual errors (the difference between the predicted and actual values) using gradient descent. Gradient Boosting is highly effective for both classification and regression tasks, offering high predictive accuracy, especially when tuned properly. One famous and well-performing distribution of GB is the XGBoost framework which allows the use of GPUs in parallel training of multiple trees, and we will be using this framework in later parts to acquire some well-trained models.

1. Training Process and Evaluation Metrics

Two main ML problems we selected to solve on our real estates dataset are price regression and property type classification. Real estates’ price prediction is a common problem that are explored by many for hands-on experience on Machine Learning, using provided features of estates to predict their prices. In many cases, external features are also included, such as economic indicators, to further improve models’ performance. On the other hand, we found estates’ type classification an appealing problem: identifying type of estates based on their features can allow to learn patterns of different classes of housing properties.

For this project, we used Linear Regression and its regularization variants, SVM Regression, Random Forest Regression and Gradient Boosting Regression to learn the price prediction problem. Tree-based algorithms like RF and GB will use a histogram-based learning technique, separating the real-valued prices into “bins” (discrete ranges of values) instead of learning directly which inputs go to which outputs, allowing more generalization in the model. SVM, Random Forest, Gradient Boosting are also used to learn the property type classification problem, and models provided by the XGBoost framework are also used with the expectation of better performance than the mentioned models.

To evaluate each model, we measured some metrics and used the results for inference and comparison between models. Commonly used metrics for the regression problem include Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Median Absolute Error (MedAE). Beside these metrics, we also used statistics of absolute percentage errors, which is the absolute error divided by the actual value to obtain the percentage by which the predicted value is away from the actual one. The absolute percentage error for one instance of data can be formulated as:

where is the actual result and is the predicted results. The Mean Absolute Percentage Error (MAPE) and Median Absolute Percentage Error (MedAPE), which are the mean and median values of all absolute percentage errors in the dataset, are obtained and included for evaluation. The classification problem is also evaluated on commonly used metrics: Accuracy, Precision, Recall, and F1 score.

The workflow for training a model is as follows. The dataset is preprocessed to encode categorical features (including law document type, address, and city) to numerical values, and the post date feature is extracted to get the post year and quarter. Then, the data is split into a training set and a testing set with 20% of original data used for testing. For each model, we used the Grid Search module from the Scikit-learn library to find the best combination of parameters from a list of provided values. The search will then return the set of parameters that has the best metric result on a 5-fold cross-validation run. The metric used for regression models is MAPE and for classification models is accuracy. The best performing model will run through the 5-fold validation once more to calculate metrics’ results. This process of model selection is run once more on the standardized version of the dataset to compare performance of a model with and without data standardization. Finally, we will test the best performing models of each learning algorithm on the testing set and compare their results.

1. Results
   1. Real Estates Price Regression

The 5-fold cross-validation results can be seen as below, with the best (lowest) values in bold:

Table 1: Metrics' results of price regression models

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | RMSE | MAE | MedAE | MAPE | MedAPE |
| Linear Regression | 79.908 | 8.972 | 4.545 | 3.238 | 0.743 |
| Ridge | 79.904 | 8.971 | 4.546 | 3.238 | 0.744 |
| Lasso | 79.112 | 8.954 | 4.535 | 3.244 | 0.747 |
| Linear SVM | 86.061 | 7.757 | 2.092 | 1.632 | 0.452 |
| SVM with RBF kernel | 25.376 | 7.582 | 2.607 | 1.739 | 0.555 |
| Random Forest | **19.684** | 5.354 | 1.887 | 2.947 | 0.329 |
| Gradient Boosting | 23.724 | 5.939 | 1.593 | **1.913** | 0.321 |
| XGBoost | 21.925 | **5.186** | **1.279** | 2.497 | **0.239** |

Linear Regression models yielded the worst results among all models, with a RMSE of 79.9 and MAE of 8.97. Percentage errors are as high as 3.23 for MAPE and 0.74 for MedAPE, meaning the model can hardly predict the prices correctly. Using standardized data and regularizations barely improved performance as Ridge and Lasso regressions provided almost similar results, as well as with standardized data. However, the Lasso model, whose learning strategy encourages sparse coefficients, acquired near-zero coefficients for address, postal code, and place importance, signaling that these features may be less important than others.

Linear SVM showed noticeably better results on absolute errors and absolute percentage errors; however, it has higher RMSE which can be interpreted as some larger errors appeared, yet percentage errors are lower meaning the model fitted better to the dataset than LR models. When combined with the Gaussian radial basis function as kernel function, its RMSE value further decreased to just 25.3 in return of just a slight change in other metrics, showing even better performance.

Ensemble learning models all acquired better results than SVM, however MAE and MAPE are still as high as 5.18 and 1.91, respectively. Despite that, median values are significantly lower compared to the corresponding mean values, meaning there are some predictions with significantly skewed predictions. It is also worth noting that while the Gradient Boosting model used Decision Trees with a shallow depth of 3, Random Forest and XGBoost models both used deeper trees with a maximum depth of 12. In terms of feature importances, the most important features inferred from these models are similar, including area, property type, number of bedrooms and bathrooms.

For LR and SVM models, we also noticed that applying standardization to data slightly improved metrics’ results, while doing so for tree-based methods like Random Forest or Gradient Boosting provides either no change or slight reduction in results. This may be because standardizing data converts them into a smaller range of value, making it harder for Decision Trees to discretize numerical values into smaller ranges during learning process.

* 1. Real Estates Type Classification

The 5-fold cross-validation results can be seen as below:

Table 2: Metrics' results of property type classification models

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Accuracy (%) | Precision | Recall | F1 Score |
| Linear SVM | 58.76 | 0.489 | 0.434 | 0.429 |
| SVM with RBF kernel | 49.62 | 0.4 | 0.35 | 0.351 |
| Random Forest | 80.93 | 0.804 | 0.724 | 0.751 |
| Gradient Boosting | 81.59 | 0.745 | 0.724 | 0.733 |
| XGBoost | 83.14 | 0.794 | 0.784 | 0.786 |

SVM models performed not too well with only around 58% accuracy for linear SVM and 49% for non-linear SVM. During training, using standardized data slightly improved the linear SVM model and significantly improved the non-linear one, and a breakdown of linear SVM’s decision boundaries provided some interesting insights to the dataset. Across all classes of property types, number of bedrooms and bathrooms, along with property price, are the most significant features with large coefficient values compared to other coefficients of the same hyperplane. With L1 regularization applied, the studio apartment class is revealed to be independent of area, latitude or longitude when placed next to other classes, and it mostly depends on its price and number of bedrooms, meaning that a studio apartment tends to have possibly lower prices compared to other estate types.

Overall, the ensemble models performed well with above 80% accuracy. However, XGBoost stood out with precision and recall scores closest to each other, signaling that the model made well-generalized predictions. The most important features provided are like other models, including area, number of bedrooms and bathrooms, and price.

We also tried out feature selection with the XGBoost model, excluding some of the least significant features such as address, quarter on which the estate is posted for sale, and location importance; cross-validation results came out not too different with 82.04% accuracy, 0.77 for precision, and 0.76 for recall and F1 score. Although metrics’ results slightly decreased, the difference is small enough to allow removal of some unimportant features, reducing data complexity and provide an easier-to-use model for real-time inference.

* 1. Testing Results