National College of Ireland

Ensemble Stacking and Optimisation For Annual Revenue Prediction of Individual Airbnb Hosting: Italy

MSc Research Project
Data Analytics

Junghyun Min
Student ID: x20103352

School of Computing
National College of Ireland

Supervisor: Dr. Catherine Mulwa

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	Junghyun Min
Student ID:	x20103352
Programme:	Data Analytics
Year:	2023
Module:	MSc Research Project
Supervisor:	Dr. Catherine Mulwa
Submission Due Date:	14/08/2023
Project Title:	Ensemble Stacking and Optimisation For Annual Revenue Prediction of Individual Airbnb Hosting: Italy
Word Count:	9,837
Page Count:	31

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	17th September 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Ensemble Stacking and Optimisation For Annual Revenue Prediction of Individual Airbnb Hosting: Italy

Junghyun Min
x20103352

Abstract

This project aimed to build ensemble stacking machine learning models with hyperparameter optimisation(HPO) for a profitability prediction in 10 different cities of Italian Airbnb data. The main goal was that successfully performed modelling, prediction and evaluation by comparing HPO methods on ensemble stacking models. Another research point was to identify factors that influenced target data, Airbnb profitability, and explore differences across 10 Italian cities. Various regression machine learning techniques, including random forest, Gradient Boosting, LightGBM, XGBoost and ensemble stacking, were employed to achieve this. The ensemble stacking model was built by the use of XGBoost as a meta-model and the other models as base learners. After model buildings, hyperparameter tuning methods were applied using a combination of random search and Bayesian optimization methods carefully selected for this study. The findings of this research could provide optimum ensemble and hyperparameter tuning methods for B&B data and contribute valuable insights into the key factors of Airbnb's profitability for strategic decision-making.
Keywords: Airbnb, Ensemble Machine Learning, Ensemble Stacking, Bayesian Optimisation, Genetic Algorithm

1 Introduction

B&B business such as hotels and Airbnb has been inseparable from the travel industry. In the aftermath of Covid-19, air travel passenger traffic had stagnated but has now seen an uptrend. According to an article[1] in business travel news Europe, the recovery of airport passenger traffic will be complete by 2025. Especially, Italy is one of the most visited travel destinations in the world. In this project, different prediction models will be used and compared for Airbnb profitability in Italy using relatively recent data.

1.1 Motivation and Background

After being founded in 2009, Airbnb quickly became one of the most popular sharing economy services. In tourist destinations especially, financial benefits are incorporated

[1]https://www.businesstravelnewseurope.com/Air-Travel/Recovery-of-European-air-travel-pushed-back-to-2025

for visitors and hosts through a trustworthy marketplace, enabling the platform to grow and utilize its network to the fullest. The cost of Airbnb lodging is frequently seen by guests as being lower than that of more traditional options, such as a hotel. Additionally, by allowing guests to stay in a listed apartment, house, or private room and experience local authenticity, Airbnb offers visitors the possibility to travel. The opportunity to maximise their idle resources is another benefit that Airbnb offers property owners Hati et al. (2021).

Since Airbnb has emerged as the leader in short-term rentals, it has become one of the most talked-about issues in cities all over the world. For investors, the main benefit of the short-term rental market is that they can make profits by renting their properties to tourists at any time. As a part of the real estate platform, Airbnb is a digital infrastructure that remarkably connects guests Cocola-Gant and Gago (2021). Hosts, guests, and third-party service providers are the three client groups that represent the three sides of the business model, and Airbnb decides how to best serve their requirements. To increase the number of hosts and the value of the investment, finding adequate short-term housing, getting access to the space, reducing the risk of disappointment, and improving the lodging experience are the four problems the value proposition must address. A profile page for each listing provides information including images, the tagline, the main features, a brief description, the facilities, the cost, the cancellation policy, the house rules, the availability, and reviews.

In the interim, numerous machine learning studies have been conducted on Airbnb data, employing various technologies such as linear regression, SVM, random forest, and neural networks for price prediction. Ensemble techniques have notably outperformed other models in regression price prediction and even surpassed neural networks in many cases. Hence, this study explored different ensemble methods, stacking techniques, Bayesian search, and genetic algorithms to optimize hyperparameters. Additionally, a novel approach is introduced, generating a "profitability" column to predict profitability, beyond simple price prediction. The study utilizes a large dataset, categorized into 10 popular tourist cities, including Rome, Milan, Venice, etc. The analysis revealed variations in probabilities across these cities and employed the Anova test to assess factors influencing these probabilities.

1.2 Project Requirement Specifications

Analysed ensemble machine learning techniques' predictive ability for Italian Airbnb profitability. It also determines the most effective optimization method for the best ensemble technique. The research investigates the main factors influencing profitability among 10 Italian regions. The dataset's multi-city nature enables distinctions in profitability patterns among cities. This identification of factors can offer insights for future home purchases and B&B investments, benefiting Airbnb property management practices.

1.2.1 Research Questions

RQ: How well do the ensemble machine learning techniques predict profitability through the Italian Airbnb data and what optimisation method is optimal with the best ensemble technique?

Sub-RQ: How is the relationship explainable between the profitability of 10 regions in Italy and which factors affect the profitability formation mostly?

Because the data was collected from 10 cities, it allows for observing the differences for example how the cities can be identified respectively with profitability patterns. By identifying factors affecting profitability, this project on Airbnb can also contribute to referencing features that should be considered for home purchases and B&B investments in the future.

1.3 Research Project Objectives

Obj.	Objective Description	Techniques	Evaluation
1	**Critical reviews on machine learning techniques related to Regression Analysis**		
2	**Pre-processing**	pandas, numpy, timedelta, math	
3	**Explanatory Data Analysis**	matplotlib, seaborn, missingno	
4	**Analysis of target feature among 10 cities**	Anova Test, Central tendency	p-value, mean, median
5	**Design, Implementation and Implemantation of Machine Learning Ensemble Techniques**		
5a		Random Forest Regression	Adjusted-R^2, RMSE, MAE, Accuracy
5b		Gradient Boosting Regression	
5c		Light Gradient-Boosting Machine	
5d		XGBoost Regression	
6	**Design, Implementation and Implemantation of Advanced Stacking Model with Hyperparameter Parameter Optimisation**		
6a		Ensemble Stacked Generalization	Adjusted-R^2, RMSE, MAE, Accuracy
6b		Bayesian Optimisation	
6c		Genetic Algorithm	
7	**Correlation Rank of Top 10 Important features by 10 Diffent Cities**		
7a		Measures of Kendall's Tau	
7b		Spearman's Rank Correlation	

Table 1: Project Objectives

2 Related Work

2.1 Introduction

This related work investigates the optimal combination of ensemble Machine Learning techniques and optimisation methods. This section consists of many sub-sections such as (i) literature review on machine learning and neural network for housing price and identified gaps, (ii) investigation of ensemble methods for housing price and identified gaps, (iii) investigation of ensemble stacking technique, (iv) hyperparameter optimisation techniques on machine learning regressors and (v) comparison of reviewed techniques used with ensemble machine learning models and hyperparameter optimisation techniques for profitability prediction.

2.2 Machine Learning and Neural Network for Housing Price Prediction

A solution for predicting annual sales can be provided by machine learning (ML) algorithms since they are precise, automated, and flexible. These algorithms employ model algorithms, input variables, and predictions, improving estimation capabilities for complex problems Meharie et al. (2022).

Many factors and features of a particular house have an impact on housing pricing. These factors can be classified into several types, e.g., house factors(room type, room

size, number of rooms or bathrooms), environmental factors(surrounding community), transportation factors(location) and so on. The main technique for predicting house prices was built using multiple linear regression, and it was combined with Spearman's correlation coefficient to identify key variables. The simulation's findings indicated that the suggested model may, to a certain extent, accurately analyze and forecast housing prices. Therefore it was suggested to improve the model's generalizability, for example, exploring new machine learning methodologies and creating more reliable house price prediction algorithms, as a future work due to the constraint of the accuracy at certain points Zhang (2021). In the two reviews below, various Machine Learning techniques were employed for price prediction in addition to linear models. While the first paper showed only the multiple linear regression model, the following two reviews included experiments and comparisons of Ridge and Lasso regression with L1 and L2 regularization that can make feature selection and avoid overfitting more convenient. The experiments of this review used five common algorithms, Linear Regression, Decision Tree, Random Forest, Ridge, and Lasso. These algorithms have experimented on real estate building datasets to examine how certain features can influence property prices, and the findings revealed that the Decision Tree and Radom Forest Regressor performed better with 0.944 and 0.914 of R Squared respectively whereas Ridge and Lasso merely had 0.766 and 0.841 of R Squared respectively. However, it was concluded that these findings were required for the tested dataset and further investigations from diverse perspectives of green buildings in future studies Jamil et al. (2020). More diverse machine learning techniques were used and a comprehensive data-driven framework to analyse and predict real estate house prices based on historical data and a combination of explanatory features was introduced in this related work. A case study of approximately 500 houses in the Boston area was conducted to explore the variations in housing prices attributed to different contributing factors. Fourteen machine learning (ML) regressors were applied to the dataset, enabling a comparative analysis of model accuracy. The ML-based regressors were utilized to forecast real estate home prices based on thirteen influential factors. Additionally, the permutation feature importance technique was employed to identify the most informative features. As the result of a robust and efficient tool for evaluating ML models' performance in predicting housing prices, the findings highlighted Random Forest as the top-performing model, achieving an R-Squared value of 0.88, followed closely by the voting regressor with an R-Squared value of 0.87 Khosravi et al. (2022).

The literature proposed house price predictions and discovered useful models for house buyers and sellers from previous data based on property market data from 2016 to 2018 in Melbourne, Australia by using Machine Learning techniques. The process begins with the initial data being prepared and cleaned, then uses Stepwise and PCA algorithms to reduce and alter the data to identify the best solution. In order to determine which model is the best accurate predictor, a number of models, including Stepwise/PCA and Polynomial Regression, Decision Trees, SVM, and Neural Networks were tested. The Neural Network model displayed the highest error, even than Linear Regression and Decision Tree Regressor. The combination of Stepwise and Support Vector Machine, which is based on mean squared error assessment, is also shown to be a competitive strategy and, the Stepwise and SVM model combination performed favourably, indicating its potential for deployment for further experiments Phan (2018).

The Miami Housing Dataset was used for this study in order to differentiate it from prior ones since it included 13,932 single-family homes sold in 2016, unlike earlier research on housing markets that only focused on Miami. Machine learning and deep learning

models were employed including SVR, Linear Regression, Random Forest, Neural Network, and XGBoost, comparing their performance. The best outcomes are obtained by ensemble learning techniques, particularly Random Forest and XGBoost, with adjusted R-squared values of 0.9234 and 0.9254, respectively. Due to the features of the dataset, linear regression and SVR perform less effectively. Ensemble learning techniques, Random Forest and XGBoost outperformed all other models in evaluation metrics Wu and Yang (2022).

2.3 Ensemble Machine Learning Technologies used for Housing Price

Ensemble learners have emerged as a significant trend, particularly in data science fields. In this research, an optimization model was proposed to design ensembles that minimize bias and variance in price predictions. The main objective was to build models that outperformed individual models in terms of bias and variance. A new optimization framework was introduced to find optimal weights for designing ensembles from multiple base learners, resulting in minimized bias and variance of predictions. As a result, the proposed methodology was validated on the Boston and Ames housing datasets, demonstrating competitive performance in predicting house prices. The designed ensemble outperformed both the benchmark ensemble and individual base learners Shahhosseini et al. (2020).

This research evaluated machine learning algorithms for house price prediction and studied the impact of COVID-19 on how key factors affected house prices, such as location and net household income in a Spanish city using a large dataset between 2019-2021. To improve model fit and reduce prediction errors, k-fold cross-validation was used. For hyperparameter optimisation tuning, random and Bayesian search techniques were employed. Test datasets and performance measures were used in the model evaluation, and graphical methods such as residual plots were used to evaluate prediction errors and overfitting. Permutation significance and partial dependence plots were used to identify relevant features, which led to the model interpretation. The final model, which combined XGBoost with binning, performed exceptionally well across the board. The study also demonstrated the influence of distance and other elements on housing pricing. Ensemble Machine learning algorithms performed better than traditional linear models. Boosting-based algorithms (Gradient Boosting Regressor, XGBoosting, Light GBM) outperformed bagging-based ones (Random Forest, Extra-Trees) Mora-Garcia et al. (2022).

With the dataset over the five years, from January 2015 to November 2019, this study employed machine learning techniques, specifically XGBoost, CatBoost, Random Forest, Lasso, and Voting Regressor algorithms. Among these, the Binning-incorporated XGBoost algorithm demonstrated superior performance across all metrics being investigated, including the coefficient of determination R^2 scores, average error, and computational time Jha et al. (2020).

2.4 Ensemble Stacking Methods used for Regression Models

Typically, a stacking ensemble has utilized to create a robust model that considers the results of various different and carefully chosen modelling methods. Every model contributes significantly, and the strength of other algorithms balances out any algorithm's bias or weakness, increasing the forecast's total accuracy. Regression and classification

issues have been extensively addressed by ensemble learning methods, which combine several model algorithms Meharie et al. (2022). Ensemble-based ML techniques, like stacking, are emerging in price estimation systems. To achieve greater predicted accuracy and stability, a stacking ensemble model was proposed, combining three learning algorithms(base-learners) of RF, GB, LGB and XGBoost as meta-learner. Although the different single models are capable of dealing with categorical and numerical variables in real-world classification or regression problems, these were combined to reduce individuals' limitations through stacking ensemble machine learning algorithms. In Meharie et al. (2022), a stacking methodology was employed with a collection of base learners, Linear Regression(LR), Support Vector Machine(SVM) and Neural Network(NN), and a meta-learner of Gradient Boosting(GB). Although the Neural Network (NN) demonstrated superior performance among the base learners, a comparison between NN and the individual Gradient Boosting (GB) results was unfeasible due to the absence of standalone GB outcomes. However, the stacking ensemble demonstrated specifically outperforming in terms of performance than the NN model. For instance, in both evaluation tools, R-Square increased from 0.936 to 0.978 and RMSE decreased from 0.228 to 0.215 on testing datasets. This observation underscored the efficacy of the stacking approach in enhancing model performance. A meta-learning process is a systematic procedure addressing model errors and a key mechanism in achieving outperformed predictive accuracy and model stability.

Graczyk et al. (2010) focused on the application of six distinct Machine Learning algorithms to three ensemble methods in WEKA. Additive Regression Analysis, Bagging, and Stacking have provided several intriguing observations. Results in terms of Mean Absolute Percentage Error (MAPE) as a measure of prediction accuracy have revealed significant variations between individual algorithms and the methods. In most cases, models utilizing the stacking technique exhibited the lowest prediction errors. Notably, a substantial enhancement in accuracy was achieved by stacking most LRM, M5R, and RBF multi-model ensembles. These findings emphasised the value of seeking a hybrid multi-model solution through various datasets. Employed the Stacking technique within the realm of Ensemble Machine Learning, utilizing random forest, Gradient Boosting Decision Trees (GBDT), and XGBoost (XGB) in this research Cao et al. (2018). A California housing price prediction model was constructed based on ET, RF, GBDT, and XGB through the application of the Stacking ensemble learning methodology. Extracting pertinent features from data, conducting dimensionality reduction, and training the models for each component technique were all steps in the ensemble model-building process. The strength of the model lay in its ability to enhance prediction accuracy and effectively mitigated overfitting when confronted with noisy or feature-laden datasets. Liu et al. (2021) introduced a stacking model to improve the accuracy of predicting fluctuating house prices using emerging machine learning algorithms. It combined strong base models like Bagging regression, Extra-Trees regression, XGBoost, and LightGBM. Features impacting house prices were analysed, and a complex data preprocessing method with creative feature engineering was presented. The stacking model outperformed individual base models and the meta-model, especially in predicting extreme values. And the model was stable, useful, and did not require significant parameter adjustments.

2.5 Hyperparameter Optimisation Techniques

By utilizing different machine learning models such as KNN, MLR, LASSO regression, Ridge regression, Random Forest, Gradient Boosting, and XGBoost, this sought to construct an accurate pricing prediction model. As prior studies on Airbnb pricing ran into issues with model robustness and inadequate training, this study carefully performed explanatory data analysis, used robust models spanning from regularized regression to ensemble approaches, and makes use of cross-validation and random search for parameter tuning to fill in these gaps. XGBoost model showed the greatest performance with an amazing R^2 score of 0.6321, closely followed by Gradient Boosting came in second place with an R^2 score of 0.6292 Liu (2021). Hyperparameter optimization aims to enhance model fit and minimize prediction errors. Mora-Garcia et al. (2022) utilized various hyperparameter search strategies, evaluates algorithm performance, examines overfitting, and interprets models. The search strategies involved a random search and a Bayesian search. The best optimization approach for Linear Regression and Random Forest Regressor was the use of initial hyperparameters. On the other hand, for Gradient Boosting Regressor, Extreme Gradient Boosting, and Light Gradient Boosting, the best optimization was Bayesian optimization. Notably, the Random Search was somewhat less effective compared to other methods.

Bayesian Optimisation(BO): BO was used to find the best values for key parameters in the boosting ensemble regression trees, support vector regression, and Gaussian process regression in order to achieve optimal configuration Lahmiri et al. (2023). The Bayesian optimization process was incorporated within a 10 fold cross-validation framework to ascertain the optimal parameter and kernel values for each model. As a result, among the employed machine learning models, boosting ensemble regression trees had superior performance, even surpassing neural network models. This also had rapid performance because it used the posterior distribution that was modelled to determine the best possible points. BOHB stood as the preferred selection for optimizing machine learning models, especially when the randomly chosen subsets closely mirrored the characteristics of the provided dataset. This was due to its adeptness in effectively optimizing various hyperparameters. When dealing with a smaller hyperparameter configuration space, Bayesian optimization (BO) models were advisable. On the other hand, Particle Swarm Optimization (PSO) typically emerged as the optimal choice for scenarios involving larger configuration spaces Yang and Shami (2020).

Genetic Algorithm(GA): To increase accuracy, an ideal set of eXtreme Gradient Boosting(XGB)'s hyperparameters was found using GA. A benchmark model, GA-ANN, an artificial neural network(ANN) model optimised by the same meta-heuristic method served as a comparison for the objective prediction model (GA-XGB). The GA-XGB outperformed the GA-ANN model and other well-known mathematical techniques, according to the simulation results in terms of error range and statistical indices. To identify the most relevant input variable further explored done on the relative importance of the various attributes. The outcomes demonstrated that GA-XGB was a precise and dependable technique for column design and behaviour prediction Luat et al. (2021). Hyperparameter optimization (HPO) was used in this study to increase the precision of Length of Stay(LOS) prediction in Iranian hospitals. K-nearest neighbours (KNN), multivariate regression, decision trees (DT), random forests (RF), artificial neural networks (ANN), and XGBoost were all experimented with in this study. In specifically, the effect of combining one of the most accurate machine learning models, XGBoost, with GA, has been evaluated

and enhanced in performance. Additionally, to improve prediction accuracy, categorical features were encoded using the One-Hot encoding technique. When compared to other modelling methodologies, the suggested strategy performed better Mansoori et al. (2023).

2.6 Summary of Findings, Identified Gaps and Conclusion

In the past, linear regression models were mainly used in price prediction models, while advanced LR (Lasso and Ridge) models were used in Zhang (2021). In research Jamil et al. (2020) and Khosravi et al. (2022), where more diverse machine learning regression methods were used for housing data, the Random Forest technique was commonly emphasized as the top-performing model. An interesting finding is that in many cases, Neural Networks performed lower than traditional machine learning techniques on tabular-format data Phan (2018) and Shahhosseini et al. (2020). In previous works on Ensemble ML, Ensemble technology, especially Random Forest and Boosting techniques, has shown better results. Comparing based on the error measures, the optimal ensemble model had lower MSE, lower MAPE, and higher R^2 value in Shahhosseini et al. (2020) and Wu and Yang (2022). In Mora-Garcia et al. (2022), Ensemble on boosting and bagging comparisons for housing data was performed, although data in the form of time series were used. Bagging showed that boosting is better than boosting because overfitting can occur. Same as in Jha et al. (2020), with the dataset over the five years, among these of XGBoost, CatBoost, Random Forest, Lasso, and Voting Regressor algorithms, specifically, the XGBoost algorithm demonstrated superior performance across all metrics being investigated. A compilation of research on stacking techniques had been conducted with the aim of enhancing the performance of individual models. In the case of Stacked modelling, it performed well in all cases Meharie et al. (2022), Graczyk et al. (2010) and Cao et al. (2018).

The last part of the reviews is about HPO. The use of a combination of XGBoost and Random search has evolved a single modelling of XGBoost. However, Random search is one of the optimization methods that has been used too often and easily Liu (2021). The search strategies involved a random search and a Bayesian search in Mora-Garcia et al. (2022). Multiple experiments showed that the use of initial hyperparameters and Bayesian optimization were better optimization approaches than Random Search. In Lahmiri et al. (2023), as well as, the Bayesian optimization process on boosting ensemble regression trees had superior performance, even surpassing neural network models, followed by Gaussian process regression. Compared to other studies, more machine learning techniques have been studied in combination with various optimization methods in Yang and Shami (2020). When dealing with a smaller hyperparameter configuration space, Bayesian optimization (BO) models were advisable. On the other hand, Particle Swarm Optimization (PSO) typically emerged as the optimal choice for scenarios involving larger configuration spaces. The Bayesian Optimization method has proven to perform better than Random Search, since this method was also frequently used, additional reviews of GA were made. As GA is one of the heuristic methods which was mainly used for deep learning rather than machine learning, there have been not many discoveries of a large amount of research. According to this Luat et al. (2021) in which GA was used as HPO, GA-XGB was a precise and dependable technique for column design and behaviour prediction. Likewise in Mansoori et al. (2023), XGBoost, with GA, has been evaluated and enhanced in performance.

In conclusion, ensemble methods are versatile, powerful, and fairly simple to use.

Random forests, AdaBoost, and GBRT are among the first models you should test for most machine learning tasks, and they particularly shine with heterogeneous tabular data. Moreover, as they require very little preprocessing, they're great for getting a prototype up and running quickly. Lastly, ensemble methods like voting classifiers and stacking classifiers can help push your system's performance to its limits Géron (2022). To summarise, there had been diverse and different research on housing data with diverse machine learning and HPO methodologies though, studies on the B&B market by applying price prediction modelling with Ensemble stacking in Italy were rare, or elusive to find. Especially, a comparison of Bayesian optimisation and GA on Ensemble stacking methods.

3 Scientific Methodology Used

3.1 Introduction

A project process plan is a essential part of the project's initiation phase. Knowledge Discovery in Database (KDD) has a structure that can be applied in data mining, machine learning, artificial intelligence, pattern recognition, and data visualization. In this project, KDD methodology has been adopted that consists of dataset selection, data preprocessing, data transformation, data mining, and evaluation. The following Figure 1 describes the KDD methodology for modeling and evaluating Italian Airbnb profitability prediction with Ensemble stacked models and Hyperparameter optimisations.

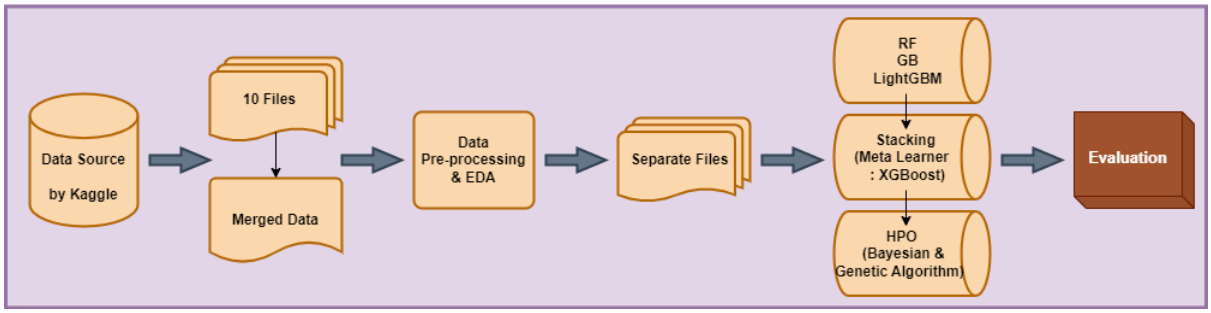


Figure 1: Modified Scientific Methodology used

3.2 Methodology Approach

Data Selection: The process of data selection involves choosing the necessary data from databases or raw sources for analysis. It requires a comprehensive understanding of the business domain and clear objectives for the analysis task. In this project, the goal was to develop pricing or profitability prediction models using B&B data obtained from Kaggle.com. The selected data was substantial in volume, relatively recent, and originated from regions with limited research and development focus. The number of features was abundant, and and the feature columns to be required for a creation of profitability were observed with minimal to no missing values in the initial dataset. The data was accessible through local software and hardware, and further supported by Colab environment and Google Drive for software and hardware.

Data Preprocessing: Data preprocessing involves identifying noise, outliers, and missing values that might be present in the selected dataset, and refining such as deletion and transformation. The data selection process can be revisited if necessary through this step. Since the data comprise 10 distinct city datasets, it was merged into a single dataset to address problems such as outliers, data type transformation, and normalization simultaneously. Subsequently, unnecessary elements were removed after the dataset was split into 10 separate datasets once again, for not only city-specific preprocessing but also the next steps, modelling and evaluation. Visualizations like distribution plots and statistical tests such as ANOVA were employed to ascertain the experiment's validity.

Data Transformation: A step to convert refined data to create and select the feature of the data according to the purpose of analysis, reduce the dimension of the data, and proceed with data mining. Through the correlations and multicollinearities identified in the previous step, removed unspecified columns and columns that affected high VIF scores. In addition, features that were required to flatten the values applied one-hot-encoding or label encoding to flatten the data. In this step, the dataset should be split into training and test datasets.

Data Mining: This is a step of executing data mining by selecting a data mining algorithm according to the purpose of analysis using the learning dataset. Data preprocessing and data conversion procedures may be repeated as necessary. Random Forest, Gradient Boosting, LightGBM, XGBoost, Ensemble Stacking, Bayesian Optimization, and Genetic Algorithm technologies were applied, and AdaBoost which was initially considered was excluded at this stage.

Evaluation: Interpretation and evaluation of the results of data mining are conducted. The model evaluation methods used in this project were Adjusted R-squared, root mean square error (RMSE), mean absolute error (MAE), and Accuracy scores. To evaluate the factors that affected models the most, measures of Kendall's Tau and Spearman's Rank Correlation were used.

3.3 Description Of dataset

The original dataset is made up of ten distinct CSV files, each one representing a different city in Italy. To start with this project, these ten files were merged into a single dataset to facilitate explanatory data analysis (EDA). The combined dataset has 76 columns and 181,956 rows. There are 34 columns with no missing values. On the other hand, 42 columns of the dataset had missing values, which required either to be replaced with new values or dropped. Additionally, 37 out of 76 columns must be converted to numerical type from object type in order for a regression model to be executed. This is a description of the Italian Airbnb data in Table 2

Merged initial datasets encompass a total of 181,956 rows, but the target for modelling is Profitability. In the pursuit of this objective, a new column can be newly created through the pre-processing and feature selection steps and the features essential for the new feature named 'profitability_by_numOfYears' include 'price', 'host_since', and 'host_total_listings_count'. Since the data size should be tailored to these three features, the data would be 181870 rows from 181,956 rows. Therefore, columns with missing values can be viewed as columns in the red box in Figure 2

Description of Data	
price	
id, scrape_id, host_id	IDs
accommodates, reviews_per_month	Integer
listing_url, picture_url, host_url, host_thumbnail_url, host_picture_url	URLs
last_scraped, calendar_last_scraped, host_since, calendar_updated, first_review, last_review	Date
host_verifications, has_availability, instant_bookable, host_is_superhost, host_has_profile_pic, host_identity_verified, license	Boolean
host_location, host_neighbourhood, neighbourhood, neighbourhood_cleansed, neighborhood_overview, neighbourhood_group_cleansed	Location
latitude, longitude	Distance
property_type, room_type	type of property
city, name, source, amenities, host_name, host_about, description	Name or Description
availability_30, availability_60, availability_90, availability_365	Possible dates
number_of_reviews, number_of_reviews_ltm, number_of_reviews_l30d	Number of Reviews
host_listings_count, host_total_listings_count, calculated_host_listings_count, calculated_host_listings_count_entire_homes, calculated_host_listings_count_private_rooms, calculated_host_listings_count_shared_rooms	Counting
host_response_time, host_response_rate, host_acceptance_rate	Host's Response
bathrooms, bathrooms_text, bedrooms, beds	Rooms, Beds and Bathrooms
minimum_nights, maximum_nights, minimum_minimum_nights, maximum_minimum_nights, minimum_maximum_nights, maximum_maximum_nights, minimum_nights_avg_ntm, maximum_nights_avg_ntm	Requirement of nights
review_scores_rating, review_scores_accuracy, review_scores_cleanliness, review_scores_checkin, review_scores_communication, review_scores_location, review_scores_value	Review Scores

Table 2: Data Features

Figure 2: Null Values

3.4 Evaluation Tool

3.4.1 Adjusted R-sqaure

Adjusted R-squared, by accounting for degrees of freedom, penalises the inclusion of additional predictors(features) in a model to mitigate overfitting Bruce et al. (2020).

$$\text{Adjusted } R^2 = 1 - \frac{(1 - R^2) \cdot (n - 1)}{n - p - 1}$$

where R^2 is the R-squared value. n is the sample size, and p is the number of independent variables in the regression model.

3.4.2 Mean Absolute Error (MAE)

MAE computes the absolute value of the error instead of squaring it. Additionally, MAE treats each observation equally during the average calculation, making it more robust to outliers compared to squared metrics that disproportionately penalize outliers Lewinson (2020).

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

n represents the number of samples, yi represents the actual observation, and ŷi represents the predicted value. $|x|$ represents the absolute value of x.

3.4.3 Accuracy

Mean Absolute Percentage Error (MAPE) is akin to MAE but presented as a percentage, making it more comprehensible for business stakeholders Lewinson (2020).

$$\text{Accuracy} = 1 - \text{MAPE} = 1 - \frac{1}{n} \sum_{i=1}^{n} \frac{|y_i - \hat{y}_i|}{y_i} \times 100\%$$

3.4.4 Root Mean Square Error(RMSE)

RMSE is the most important performance metric from a data science perspective is root mean squared error or RMSE. This evaluates the overall accuracy of the model and serves as a benchmark for comparison with other models Bruce et al. (2020).

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2}$$

4 Design Specification

Since this project did not require the creation of a database to generate new data, the architecture design of this project used a 2-tier architecture and consists of a presentation layer for Tier 1 and a business logic and data Layer for Tier 2. Tier 1 is a client layer that consists of the visualisation of the user interface and prediction results. Tier 2 is a layer of overall modelling preparation and model fitting, prediction and evaluation. The design of this study is shown in Figure 3.

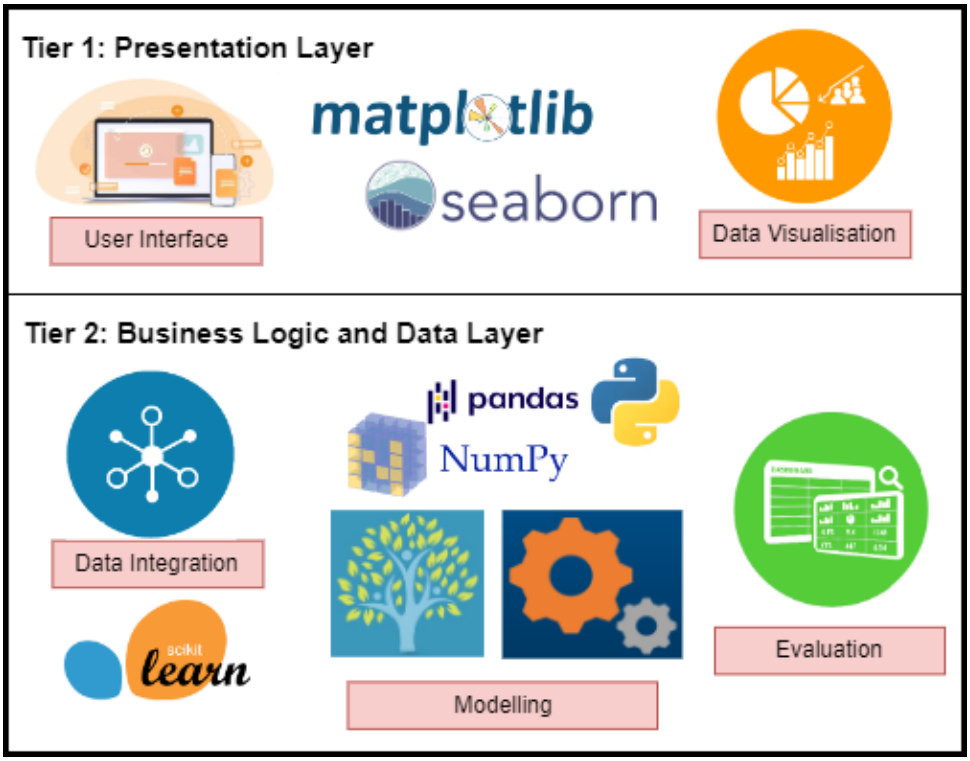


Figure 3: Architecture Design

5 Implementation, Evaluation and Result of Models

5.1 Data Pre-Processing and Feature Engieneering

One column with a date type(yyyy-mm-dd) was 'host_since', which serves as a pivotal column for establishing data specifications because the measurement of hosting tenure and the profitability calculation became challenging without them. Consequently, the data is cut to conform with this column as dropping rows if they have NA in the 'host_since' column. And then, remove the $ and % symbols by using regular expressions. When the same host has posted multiple hostings, the values of 'host_listings_count' and 'host_total_listings_count' are duplicated. Therefore, rectification of this issue is necessary. Because, for instance in a dormitory room, if a host posts a listing for each bed in order to accommodate guests to full capacity, the hosting count for each bed is equivalently assigned as a product of the bed count. A new column called 'profitability_by_numOfYears' can now be calculated and generated. The formula is as follows:

$$\text{profitability\_by\_numOfYears} = \frac{\text{price} \times \text{total\_listings\_coount}}{\frac{\text{calender\_last\_scraped} - \text{host\_since}}{365}}$$

For the purpose of normalization, the calculated 'profitability_by_numOfYears' column (referred to as "profitability" henceforth) can be subjected to a logarithmic transformation. Subsequently, the values can be mapped into percentiles, aligning with the inherent characteristics of the "profitability" feature.

In preparation for the regression model, it is necessary to convert all data types to numeric. Utilizing regular expressions, the values within the 'host location' column are transformed. If a host is situated at a property location, it is assigned a value of 1; if

not, it is given 0; and in cases of missing values, -1 is used as a replacement. Similarly, for instances of "Yes/No," or "True/False," a similar strategy is applied: "Yes/True" is encoded as 1, "No/False" as 0, and missing values as -1. Next, data flattening for 'host_verifications'. Data presented in list format undergoes One-Hot Encoding to be transformed into a flattened structure. Additionally, the 'bathrooms_text' column, a combination of numbers and text, is separated to create a column containing the count of bathrooms and another indicating the type of bathroom. The 'amenities' column contains a considerable number of diverse unique values. In order to manage this complexity, a strategy is implemented whereby the occurrence count of each amenity is meticulously calculated, focusing on the top 100 amenities. The counted numbers are then assigned as new values within the 'amenities' column. For columns 'host_response_time', and 'room_type', replace values with -1 to 3 because there are not many unique values. In the case of the 'property_type' column, which exhibits a wide array of values, a label encoder is employed. This approach is effective in converting categorical variables into a numerical format.

Lastly, drop columns. Even without performing correlation and multicollinearity checks, it was evident from the data set analysis stage that there was substantial column overlap. Therefore, the drop of columns can be carried out within the data pre-processing stage once in advance. And then, replace all missing values with -1 by fillna function in the case of values left with missing values.

5.2 Explanatory Data Analysis

5.2.1 Correlation and Multicollinearity

There are still columns that haven't dropped in the previous stage even if there are overlapping attributes. For example, 'review_scores_cleanness' is an assessment of cleanliness, and 'review_scores_communication' is a score for communication with the host. Although these columns share similarities in characteristics and values, correlations need to be checked as they might pertain to different aspects. When observing the Heatmap displaying correlations >0.7 in Figure 4, notably high correlations can be observed, suggesting that retaining only the 'review_scores_rating' column could be a favourable choice. It can be made sure that the remaining score-related columns are better to be removed except only 'review_scores_rating' because the VIF scores were too high. However, in the context of availability-related columns, eliminating only 'availability_90' resulted in a reduction of all remaining column's VIF scores to values under five, Figure 5.

In most cases, ensemble machine learning does not need to take the feature's multicollinearity into account. This is due to the fact that ensemble models do not take into account interactions or linear correlations between individual model operations and predictions. Each model makes predictions about various aspects of the data and then combines these predictions to improve its predictive ability. As a result, the multicollinearity of the feature usually has little impact on how well the ensemble model performs. The study, Chowdhury et al. (2022), stated a strong correlation between predictors in a dataset resulted in multicollinearity. Since tree-based models which were non-parametric perform better, were able to handle intricated interactions between variables, and were unaffected by multicollinearity. Taking this into account, a few columns if they are not far from 5 were retained as they were considered essential feature columns. This decision was based on the fact that, aside from these few columns, most others were found to have a VIF of less than 5 when considering multicollinearity.

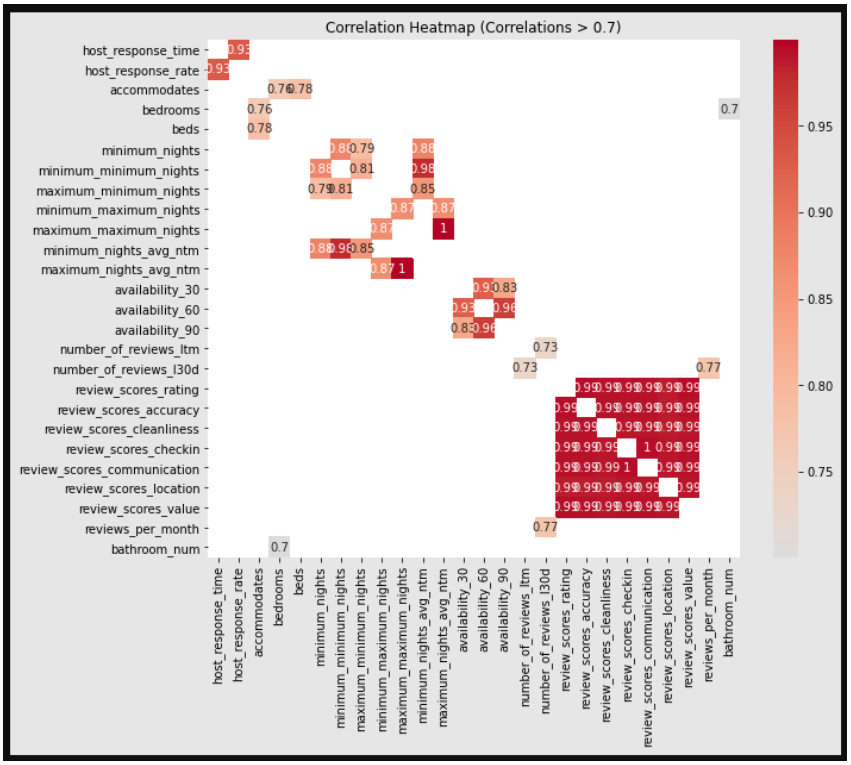


Figure 4: Correlation HeatMap >0.7

	Variable	VIF
4	review_scores_communication	138.410
3	review_scores_checkin	137.117
0	review_scores_rating	129.128
1	review_scores_accuracy	123.837
6	review_scores_value	105.388
2	review_scores_cleanliness	87.855
5	review_scores_location	60.760

	Variable	VIF
1	availability_60	51.732
2	availability_90	26.124
0	availability_30	11.824
3	availability_365	1.924

	Variable	VIF
1	availability_90	4.462
0	availability_30	3.234
2	availability_365	1.870

Figure 5: Multicollinearity with VIF scores

5.2.2 Distribution and ANOVA Test

Using histograms, the relative distributions of the target data are compared, Figure 6. However, the differences in distributions observed through histograms are not prominently evident. Therefore, variance analysis for the ten different cities should be conducted by the ANOVA test.

ANOVA Test: The hypothesis is related to whether the means of the profitability values are equal across different cities or not.

1) Null Hypothesis(H0): The means of profitability values are equal across all cities.

2) Alternative Hypothesis(H1): At least one mean is different from the others.

3) Result of One-way ANOVA p-value: 0.0

There is a significant difference in the means of profitability values among the cities. In other words, the observed difference among the city groups is extremely unlikely to have occurred due to random chance because the p-value is nearly zero.

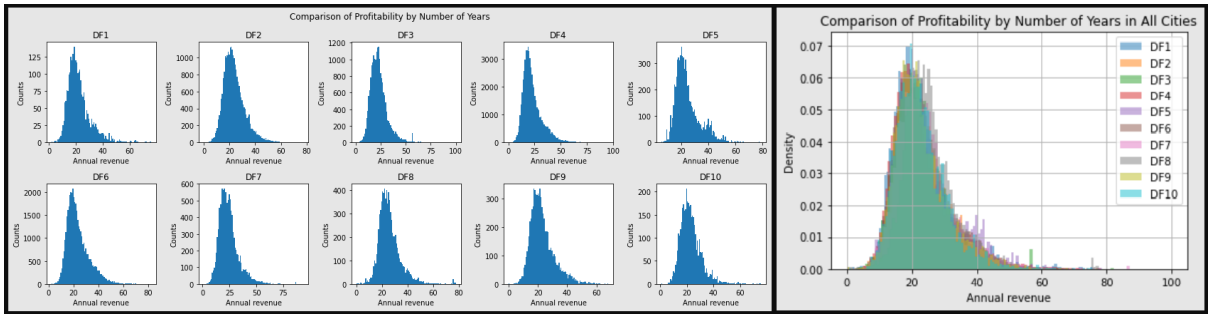


Figure 6: Distributions of profitability in All cities(10 cities: DF1- -DF10)

5.3 Implementation, Evaluation and Results of Italian Airbnb Profitability Prediction using Ensemble Models and HPO

5.3.1 Random Forest

The ensemble method known as the random forest algorithm mixes various decision trees. Due to randomization, a random forest typically performs better in generalization than a single decision tree, which lowers the variance of the model. Random forests also have the benefit of requiring little parameter adjustment and being less susceptible to outliers in the dataset. Often only need to experiment with the number of ensemble trees in random forests as a parameter. The predicted target variable is calculated as the average forecast of all the decision trees in the random forest, which grows each individual decision tree according to the MSE criterion Raschka and Mirjalili (2017).

$$\text{Prediction} = \frac{1}{N}\sum_{i=1}^{N}\text{Prediction}_i$$

Where N is the number of trees in the ensemble, and Prediction represents the predicted value of each tree. The average of these predictions becomes the final prediction of the random forest.

Implementation: By iterating over a range from 10 to 100 with increments of 10, from 10 to 200 with increments of 20 and from 15 to 300 with increments of 15, the best scores were achieved with n_estimators=90 for 0.95, n_estimators=190 for 0.952 and n_estimators=275 for 0.953 respectively on the training dataset and the learning state graphs are followed by Figure 7. The range of 200 to 300 showed similar optimal scores. Therefore, using a loop function with 200 estimators could be a cost-effective choice.

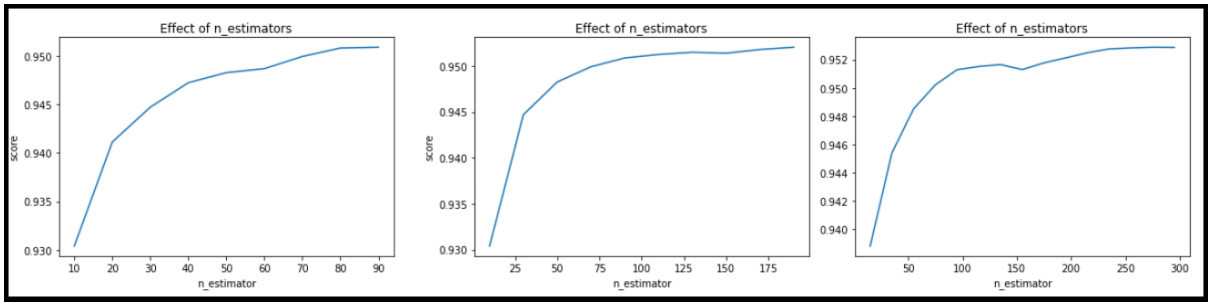
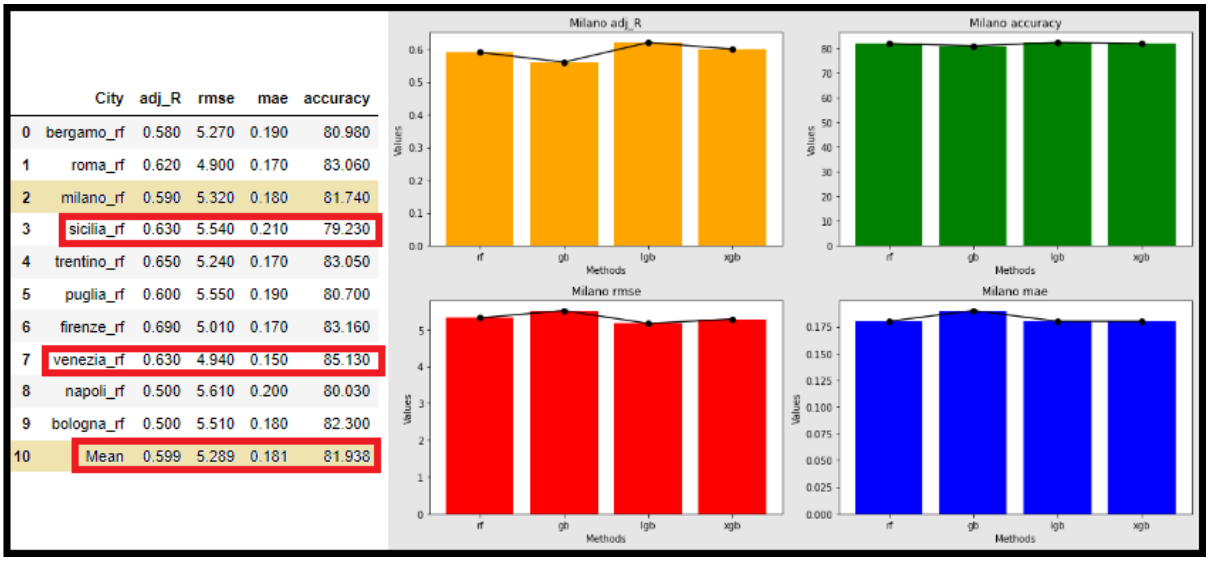


Figure 7: Train Random Forest Model with Iteration range of 100, 200 and 300

Evaluation and Results: The lowest values recorded are 0.63 for Adjusted-R^2, 5.54 for RMSE, 0.21 for MAE, and 79.23 for Accuracy. Conversely, the highest values achieved are 0.63 for Adjusted-R^2, 4.94 for RMSE, 0.15 for MAE, and 85.13 for Accuracy. On average, the values stand at 0.59 for Adjusted-R^2, 5.28 for RMSE, 0.18 for MAE, and 81.93 for Accuracy. Random Forest showed better performance than the basic Gradient Boosting model on all 10 data. The results for models on the Milano data in all four measurements are depicted in Figure 8, closely resembling the average values.



	City	adj_R	rmse	mae	accuracy
0	bergamo_rf	0.580	5.270	0.190	80.980
1	roma_rf	0.620	4.900	0.170	83.060
2	milano_rf	0.590	5.320	0.180	81.740
3	sicilia_rf	0.630	5.540	0.210	79.230
4	trentino_rf	0.650	5.240	0.170	83.050
5	puglia_rf	0.600	5.550	0.190	80.700
6	firenze_rf	0.690	5.010	0.170	83.160
7	venezia_rf	0.630	4.940	0.150	85.130
8	napoli_rf	0.500	5.610	0.200	80.030
9	bologna_rf	0.500	5.510	0.180	82.300
10	Mean	0.599	5.289	0.181	81.938

Figure 8: Random Forest on Milano Dataset

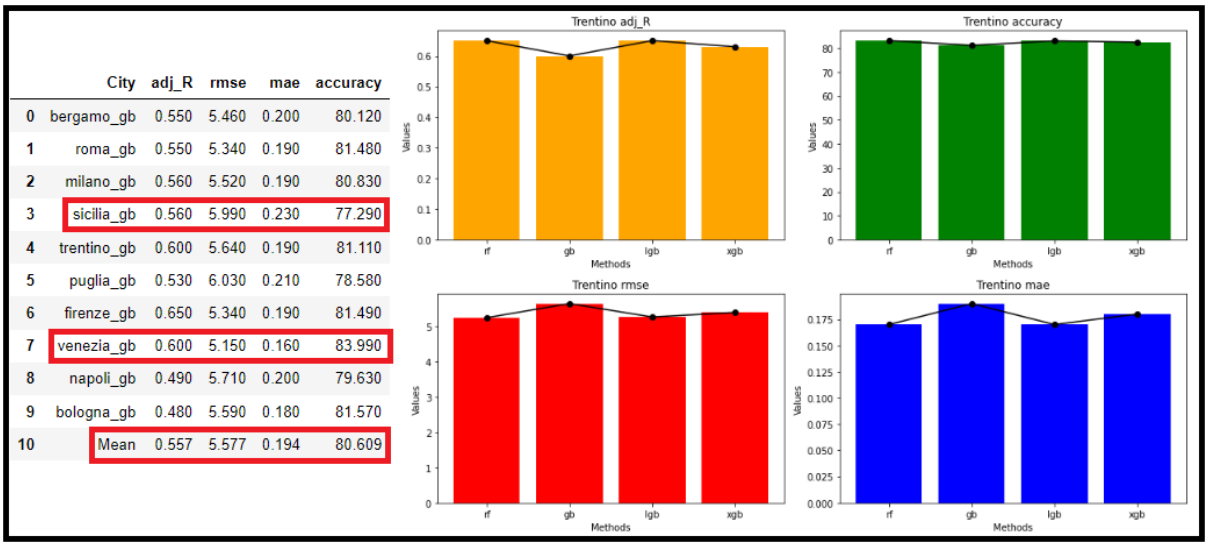
5.3.2 Gradient boosting

Gradient Boosted Trees (GBTs) leverage the concept of "boosting" in statistical modelling, combining multiple weak models into a more robust aggregated model. Boosting is an ensemble learning algorithm aimed at enhancing the predictive performance of regression or classification procedures, like decision trees Meharie et al. (2022). Gradient boosting regression tree is another ensemble technique that combines different decision trees to produce a more potent model. Gradient boosting builds trees in a serial fashion, trying to fix the mistakes of the preceding tree, as opposed to the random forest approach. Gradient boosting by default employs strong pre-pruning rather than randomisation. In order to reduce memory requirements and speed up prediction, Gradient boosting frequently utilises shallow trees (depths one to five). The primary principle of gradient boosting is to integrate numerous weak learners, or simple models, such as shallow trees. More and more trees are added to incrementally increase performance since each tree can only provide accurate predictions for a portion of the data Muller and Guido (2017).

Implementation: The implementation condition of Gradient Boosting is the same as Random Forest.

Evaluation and Results: The lowest performances recorded are 0.56 for Adjusted-R^2, 5.99 for RMSE, 0.23 for MAE, and 77.29 for Accuracy. Conversely, the highest performances achieved are 0.6 for Adjusted-R^2, 5.15 for RMSE, 0.16 for MAE, and 83.99 for Accuracy. On average, the performances stand at 0.557 for Adjusted-R^2, 5.57 for RMSE,

0.194 for MAE, and 80.609 for Accuracy. Gradient Boosting showed the worst performances among the single basic models on all datasets except for Bergamo at Adjusted-R^2 and RMSE. The modelling results for the dataset most closely resembling the mean values once again correspond to Milan. Therefore, a graph in Figure 9 depicting the evaluations of the GB modelling for Trentino which is the second most similar.



	City	adj_R	rmse	mae	accuracy
0	bergamo_gb	0.550	5.460	0.200	80.120
1	roma_gb	0.550	5.340	0.190	81.480
2	milano_gb	0.560	5.520	0.190	80.830
3	sicilia_gb	0.560	5.990	0.230	77.290
4	trentino_gb	0.600	5.640	0.190	81.110
5	puglia_gb	0.530	6.030	0.210	78.580
6	firenze_gb	0.650	5.340	0.190	81.490
7	venezia_gb	0.600	5.150	0.160	83.990
8	napoli_gb	0.490	5.710	0.200	79.630
9	bologna_gb	0.480	5.590	0.180	81.570
10	Mean	0.557	5.577	0.194	80.609

Figure 9: Gradient Boosting on Trentino Dataset

5.3.3 LightGBM

LightGBM is a newer algorithm that includes some improvements compared with XGBoost, although it does not always outperform XGBoost in practice. It creates the decision trees in the ensemble differently using novel techniques, which allows it to run faster and use less memory than XGBoost. It also can handle missing values and categorical data natively. It was created by Microsoft and is what Azure's ML GUI uses when a boosted decision tree ML algorithm is chosen George (2021).

Implementation: The implementation condition of LightGBM is the same as Random Forest and Gradient Boosting.

Evaluation and Results: In the case of LightGBM, the 4 types of evaluation values did not consistently indicate a common trend on different datasets. Therefore, a city of comparison graph was selected based on Accuracy. The lowest performances recorded are 0.63 for Adjusted-R^2, 5.54 for RMSE, 0.21 for MAE, and 79.43 for Accuracy. Conversely, the highest performances achieved are 0.65 for Adjusted-R^2, 4.8 for RMSE, 0.15 for MAE, and 85.49 for Accuracy. On average, the performances stand at 0.608 for Adjusted-R^2, 5.25 for RMSE, 0.179 for MAE, and 82.054 for Accuracy. Comparing LightGBM and GB, all the results of LightGBM on all datasets of 10 cities showed better performances. Figure 10 shows a table comparing the results with the evaluation indicators of LGBM modelling and comparison graphs of basic models (RF, GB, LGBM, XGBoost) on the Bologna dataset is composed of four indicators.

	City	adj_R	rmse	mae	accuracy
0	bergamo_lgb	0.570	5.370	0.190	80.570
1	roma_lgb	0.620	4.950	0.170	82.900
2	milano_lgb	0.620	5.170	0.180	82.170
3	sicilia_lgb	0.630	5.540	0.210	79.430
4	trentino_lgb	0.650	5.260	0.170	82.920
5	puglia_lgb	0.600	5.570	0.190	80.530
6	firenze_lgb	0.690	4.980	0.170	83.070
7	venezia_lgb	0.650	4.800	0.150	85.490
8	napoli_lgb	0.530	5.470	0.190	80.700
9	bologna_lgb	0.520	5.400	0.170	82.760
10	Mean	0.608	5.251	0.179	82.054

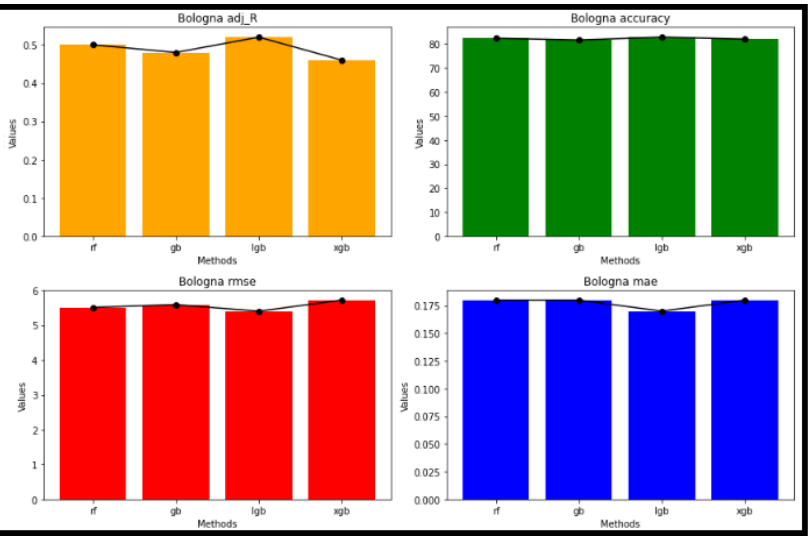


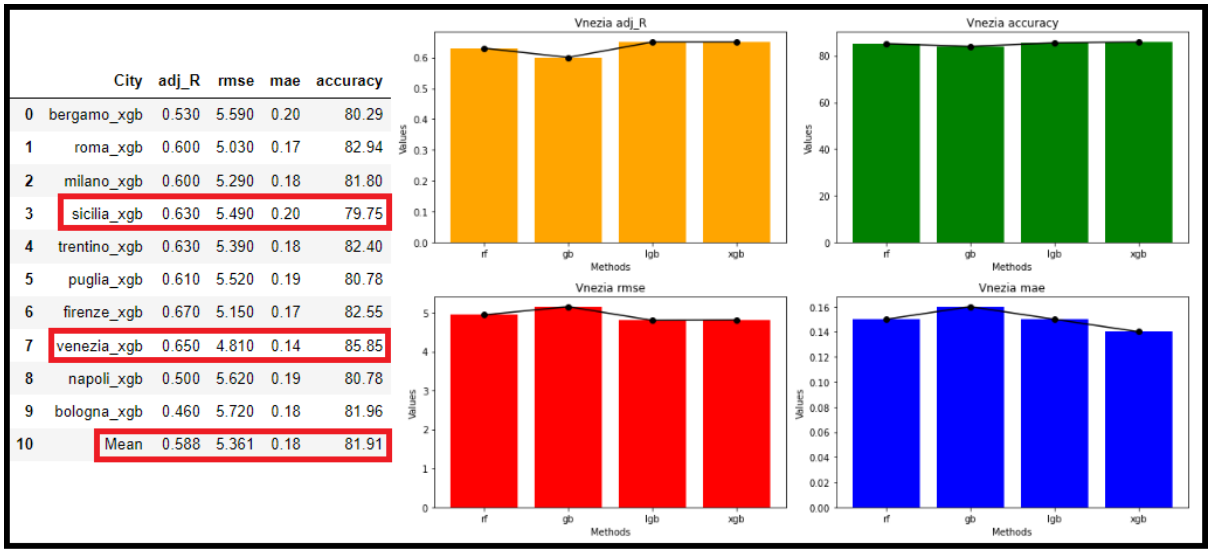
Figure 10: LightGBM on Bologna Dataset

5.3.4 XGBoost

XGBoost stands for "extreme gradient boosting." It makes several improvements to plain gradient boosting, such as using Newton boosting. Instead of finding the ideal multiplier to scale each weak learner by (which is like a step length in our gradient descent), XGBoost solves the direction and step length in one equation. By contrast, gradient boosting uses something called a line search to find the optimum multiplier (step length) for each weak learner. This means XGBoost can be faster than plain gradient boosting George (2021). The general structure of XGBoost is the same as that of gradient boosting, which means that it strengthens weak learners into strong learners by averaging the residuals of trees. Since XGBoost regularly produces superior results and is faster than gradient boosting, it is favoured over gradient boosting in general Wade and Glynn (2020).

Implementation: In the case of XGBoost, owing to the inherent functionality of XGBRegressor, more parameter adjustments were executed automatically. Among the parameters used for the best model which is the result of repeated execution, max depth, learning_rate, gamma, and a number of estimators are as follows: max_depth=6, learning_rate=0.300000012, gamma=0, n_estimators=190.

Evaluation and Results: For XGBoost as well, the 4 types of evaluation values did not consistently indicate a common trend on different datasets. Therefore, a city of comparison graph was selected based on Accuracy. The lowest performances recorded are 0.63 for Adjusted-R^2, 5.49 for RMSE, 0.2 for MAE, and 79.75 for Accuracy. Conversely, the highest performances achieved are 0.65 for Adjusted-R^2, 4.81 for RMSE, 0.14 for MAE, and 85.85 for Accuracy. On average, the performances stand at 0.58 for Adjusted-R^2, 5.36 for RMSE, 0.18 for MAE, and 81.91 for Accuracy. XGBoost has performed better than other boosting regression tree algorithms in some datasets while other approaches have performed better than XGBoost in other datasets. Interpreting this, it can be hard to assert that XGBoost always outperforms other boosting regression techniques, contrary to the findings from related works. Figure 11 shows a table comparing the results with the evaluation indicators of XGBoost modelling and comparison graphs of basic models (RF, GB, LGBM, XGBoost) on the Venezia dataset is composed of four indicators.



	City	adj_R	rmse	mae	accuracy
0	bergamo_xgb	0.530	5.590	0.20	80.29
1	roma_xgb	0.600	5.030	0.17	82.94
2	milano_xgb	0.600	5.290	0.18	81.80
3	sicilia_xgb	0.630	5.490	0.20	79.75
4	trentino_xgb	0.630	5.390	0.18	82.40
5	puglia_xgb	0.610	5.520	0.19	80.78
6	firenze_xgb	0.670	5.150	0.17	82.55
7	venezia_xgb	0.650	4.810	0.14	85.85
8	napoli_xgb	0.500	5.620	0.19	80.78
9	bologna_xgb	0.460	5.720	0.18	81.96
10	Mean	0.588	5.361	0.18	81.91

Figure 11: LightGBM on Bologna Dataset

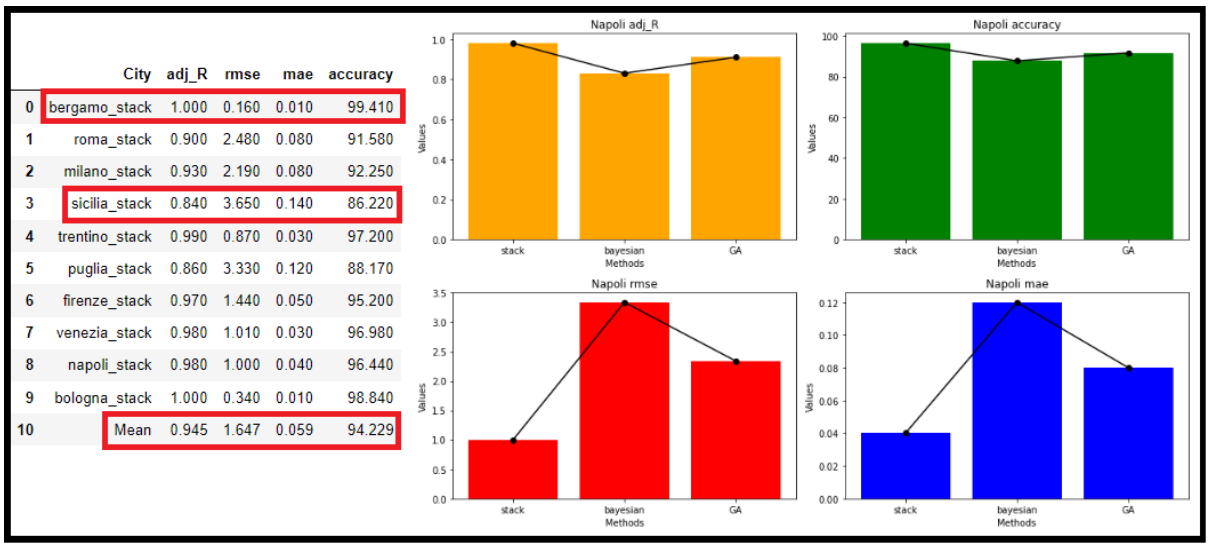
5.3.5 Proposed Stacking Ensemble Model

Ensemble stacking or stacked generalization is an advanced ensemble learning technique that goes beyond traditional methods of combining predictions from individual models. Instead of using simple aggregation methods like hard voting, stacking involves training an additional model called a "blender" or "meta learner" to learn how to combine the predictions from multiple base models. This blender takes the predictions made by individual base models as inputs and generates the final prediction. Ensemble stacking is a technique that introduces an extra layer of learning to optimally blend the predictions of base models, enhancing the overall predictive power of the ensemble Géron (2022). For increased performance, multiple layers of blenders can be used. In this case, additional blenders are trained on the outputs of previous blenders. However, this complexity comes at the cost of longer training times and system complexity. Ensemble stacking leverages the strengths of different models by allowing them to focus on specific areas of the problem. It enables the ensemble to learn how to best combine the predictions from individual models, potentially resulting in better predictive accuracy compared to using simple aggregation methods Géron (2022).

Implementation: The implementation setup for ensemble stacking is akin to that of XGBoost, and the hyperparameter condition is the same as XGBoost, given that XGBoost was adopted as the meta-learner. Random Forest, Gradient Boosting (GB), and LightGBM were employed as base learners, and then their predicted values were stacked comprised of new features for the meta-learner, XGBoost.

Evaluation and Results: When the stacked generalization ensemble technique was formed through the combination of base learners and meta-learners, it demonstrated remarkable outcomes. Across all datasets, any single-model approach didn't achieve better performance than those obtained even with the lowest performance from the ensemble stacking model. The lowest values recorded are 1.0 for Adjusted-R^2, 0.16 for RMSE, 0.01 for MAE, and 99.41 for Accuracy. Conversely, the highest values achieved are 0.84 for Adjusted-R^2, 3.65 for RMSE, 0.14 for MAE, and 86.22 for Accuracy. On average, the performances stand at 0.945 for Adjusted-R^2, 1.647 for RMSE, 0.059 for MAE, and

94.229 for Accuracy. Figure 12 shows a table comparing the results with the evaluation indicators of ensemble staking modelling and comparison graphs of advanced models (Staking, Staking+Beysian HPO, Staking+GA HPO) on the Napoli dataset composed of four indicators.



	City	adj_R	rmse	mae	accuracy
0	bergamo_stack	1.000	0.160	0.010	99.410
1	roma_stack	0.900	2.480	0.080	91.580
2	milano_stack	0.930	2.190	0.080	92.250
3	sicilia_stack	0.840	3.650	0.140	86.220
4	trentino_stack	0.990	0.870	0.030	97.200
5	puglia_stack	0.860	3.330	0.120	88.170
6	firenze_stack	0.970	1.440	0.050	95.200
7	venezia_stack	0.980	1.010	0.030	96.980
8	napoli_stack	0.980	1.000	0.040	96.440
9	bologna_stack	1.000	0.340	0.010	98.840
10	Mean	0.945	1.647	0.059	94.229

Figure 12: Ensemble Stacking on Napoli Dataset

5.3.6 Bayesian Optimisation on Stacking Ensemble Model

The Bayesian methods optimize Hyperparameter Optimization by prioritizing the selection of hyperparameters over evaluating the objective function, leading to computational efficiency. This approach uses Sequential Model-Based Optimization (SMBO), utilizing a surrogate model and an acquisition function to iteratively choose the most promising hyperparameters. The surrogate model approximates the expensive true objective function, often cross-validation error, by mapping hyperparameters to probability scores. In each iteration, the surrogate model is updated with past evaluations, and the acquisition function guides the selection of hyperparameters based on their expected utility. This balances the action of new areas and exploitation of known successful regions in the hyperparameter space. The simplified steps of Bayesian optimization involve creating a surrogate model for the true objective function, finding optimal hyperparameters on the surrogate, using those hyperparameters to evaluate the true objective, updating the surrogate with results, and repeating the process until a stop criterion is met. As the algorithm runs longer, the surrogate approximates the true objective more accurately. Bayesian Hyperparameter Optimization (HPO) reduces the search time for optimal parameters, especially for complex cases Lewinson (2020).

Implementation: Bayesian optimisation on the Stacked model was applied by using BayesSearchCV() which is a hyperparameter tuning tool. The scope of Bayesian search defined and the manual parameter setting of BayesSearchCV() were as follows:

```
bayes_search = {
    'n_estimators': (100, 1000),
```

```
    'learning_rate': (0.01, 1.0),
    'max_depth': (1, 16),
    'gamma': (0.01, 1.0)}

BayesSearchCV(meta_learner, bayes_search, n_iter=10, cv=10,
              scoring="neg_mean_squared_error", verbose=4, random_state=123)
```

The best parameters obtained were 'gamma' with a value of 0.7703399241440627, 'learning_rate' of 0.19204784774815406, 'max_depth' set to 2, and 'n_estimators' at 764.Under the conditions of n_iter=10 and cv=10, a total of 100 iterations were executed, and the results of 10-fold cross-validations are presented in Figure 13. A common observation across the graphs is that towards the end of each iteration, there is a notable increase in errors(the y-label for each graph is negative, so it should be interpreted in reverse).

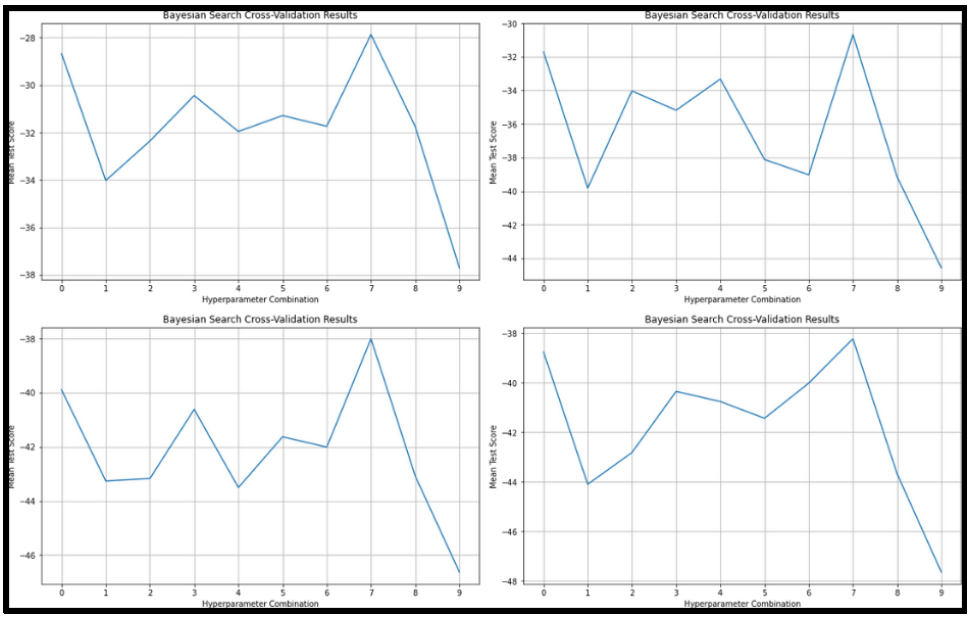


Figure 13: Bayesian Cross Validations

Evaluation and Results: The combination of the stacking model and Bayesian optimization across all 10 datasets demonstrated that the error scores for all models were higher and the Adjusted-R^2 values for all models were lower compared to those of the individual Stacking model. The lowest values recorded are 0.71 for Adjusted-R^2, 4.88 for RMSE, 0.18 for MAE, and 81.8 for Accuracy. Conversely, the highest values achieved are 0.94 for Adjusted-R^2, 2.09 for RMSE, 0.08 for MAE, and 91.77 for Accuracy. On average, the performances stand at 0.825 for Adjusted-R^2, 3.46 for RMSE, 0.12 for MAE, and 87.709 for Accuracy. Figure 14 shows a table comparing the results with the evaluation indicators of ensemble staking modelling. And comparison graphs that show the results closest to the mean scores can be found in Figure 12.

5.3.7 Genetic Algorithm with DEAP Framework on Stacking Ensemble Model

DEAP (Distributed Evolutionary Algorithm in Python) open-source library was highly recommended to both beginners and experts. The genetic algorithm, multi-objective evolutionary algorithms like NSGA-II and SPEA2, and strongly and loosely typed genetic

	City	adj_R	rmse	mae	accuracy
0	bergamo_bayes	0.940	2.09	0.080	91.770
1	roma_bayes	0.750	3.97	0.140	86.180
2	milano_bayes	0.770	3.98	0.140	85.740
3	sicilia_bayes	0.710	4.88	0.180	81.800
4	trentino_bayes	0.890	2.97	0.100	89.730
5	puglia_bayes	0.710	4.74	0.170	83.350
6	firenze_bayes	0.860	3.36	0.120	88.340
7	venezia_bayes	0.870	3.00	0.090	90.660
8	napoli_bayes	0.830	3.34	0.120	87.790
9	bologna_bayes	0.920	2.27	0.080	91.730
10	Mean	0.825	3.46	0.122	87.709

Figure 14: Bayesian Optimisation on 10 cities

programming are all supported by DEAP. The majority of the fundamental functions needed for evolutionary computation are included in it, allowing users to create a variety of single- and multi-objective evolutionary algorithms with ease and run them over numerous processors. It may be used with many other Python modules for data processing and machine-learning methods, making it ideal for quick prototyping Kim and Yoo (2019).

Implementation: The synthesis of ensemble stacking and the DEAP framework for genetic algorithms is accomplished through the integration of custom-built functions and the DEAP framework. It involves defining and registering essential components such as initial population creation, crossover and mutation operations, selection, and evaluation functions. The code establishes "FitnessMin" and "idx" classes for creating and initializing individuals. It registers functions for random number generation and operations. Among the parameters used for the best model which is the result of repeated execution, max depth, learning rate, gamma, and several estimators are as follows max_depth=10, learning_rate=0.3661267728814921, gamma=0.07173330604877207 and n_estimators=190. As depicted in Figure 15, unlike Bayesian search, Genetic Algorithms (GAs) demonstrate a trend across 20 generations of iterations where the stability of trends increases towards the later stages.

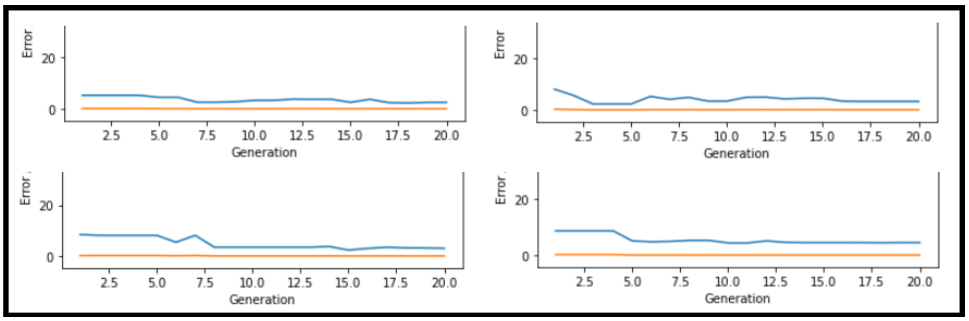
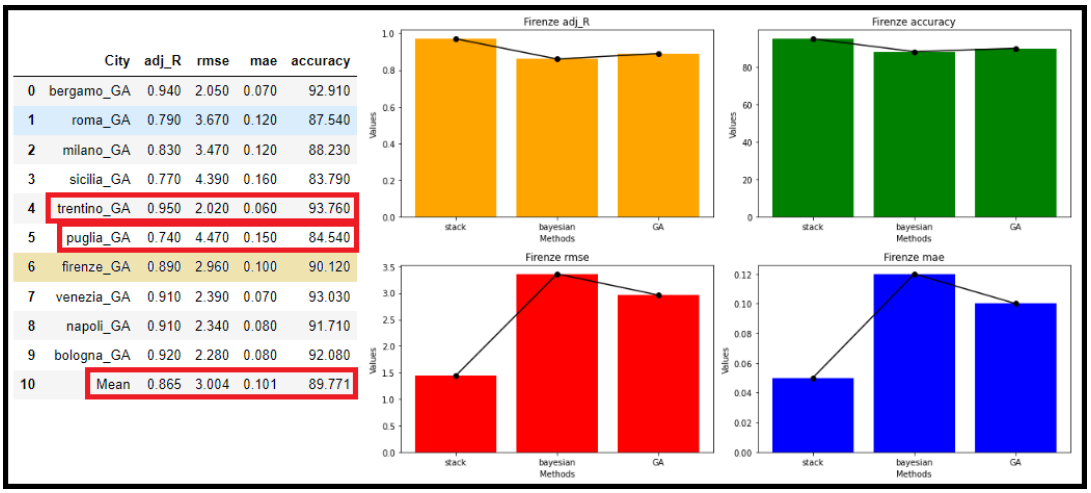


Figure 15: Bayesian Cross Validations

Evaluation and Results: In the case of the lowest values were selected based on Adjusted-R^2 and RMSE metrics. The lowest performances recorded are 0.74 for Adjusted-

R^2, 4.47 for RMSE, 0.15 for MAE, and 84.54 for Accuracy. Conversely, the highest performances achieved are 0.95 for Adjusted-R^2, 2.02 for RMSE, 0.06 for MAE, and 93.76 for Accuracy. On average, the performances stand at 0.865 for Adjusted-R^2, 3.004 for RMSE, 0.101 for MAE, and 89.77 for Accuracy. GA has performed better than Bayesian optimisation in all different datasets. Figure 16 shows a table comparing the results with the evaluation indicators of GA modelling and comparison graphs of advanced models (Staking, Staking+Beysian HPO, Staking+GA HPO) on the Firenze dataset composed of four indicators.



	City	adj_R	rmse	mae	accuracy
0	bergamo_GA	0.940	2.050	0.070	92.910
1	roma_GA	0.790	3.670	0.120	87.540
2	milano_GA	0.830	3.470	0.120	88.230
3	sicilia_GA	0.770	4.390	0.160	83.790
4	trentino_GA	0.950	2.020	0.060	93.760
5	puglia_GA	0.740	4.470	0.150	84.540
6	firenze_GA	0.890	2.960	0.100	90.120
7	venezia_GA	0.910	2.390	0.070	93.030
8	napoli_GA	0.910	2.340	0.080	91.710
9	bologna_GA	0.920	2.280	0.080	92.080
10	Mean	0.865	3.004	0.101	89.771

Figure 16: HPO Genetic Algorithm with on 10 cities

5.4 Rank Correlation on Different 10 Cities

5.4.1 Introduction

The Spearman and Kendall methods assess correlations based on rank order instead of raw values. Data points are ranked in ascending order for two data columns. When both column ranks increase simultaneously, a strong correlation is indicated. These methods excel at detecting associations between columns of non-linear data George (2021). Because they operate on ranks instead of actual values, these estimations are resilient against outliers and can manage specific forms of nonlinearity Bruce et al. (2020). Both methods are sensitive to ranking information, and both indices are useful if order information is important because the exact order of the data affects the results of the correlation. The interpretation of the two indices is context-sensitive, and you must select the appropriate indices for your research purposes. The difference between the two methods lies in the calculation method and the definition of correlation coefficients. Kendall's Tau calculates the correlation based on the matching of the ordered pairs, and Spearman's Rank Correlation calculates the correlation based on the difference in rank. Kendall's Tau also measures the monotonous (monotonic) correlation between two variables, while Spearman's Rank Correlation can also be used to measure the nominal relationship between two variables.

5.4.2 Kendall's Tau and Spearman's Rank Correlation

Kendall's Tau measures correlation based on the matching of ordered pairs between the two variables. Values range from -1 to 1, with -1 indicating a complete inverse relationship, 1 indicating a complete equal order relationship, and 0 indicating no correlation. Positive values indicate a tendency to match in order, and negative values indicate a tendency to match in reverse order. The larger the absolute value of the value, the stronger the order relationship between the two variables is interpreted. Spearman's Rank Correlation measures correlation based on the difference in rank between two variables. Values also range from -1 to 1, with -1 indicating a complete inverse relationship, 1 indicating a complete equal ranking relationship, and 0 indicating no correlation. Like Kendall's Tau, positive values tend to match in order and negative values tend to match in reverse order. The larger the absolute value of the value, the stronger the ranking relationship between the two variables is interpreted. Both indices are nonparametric, so they are useful when both variables are ordinal or contain only ranking information. This is a nice little introduction with some in Figure 17.

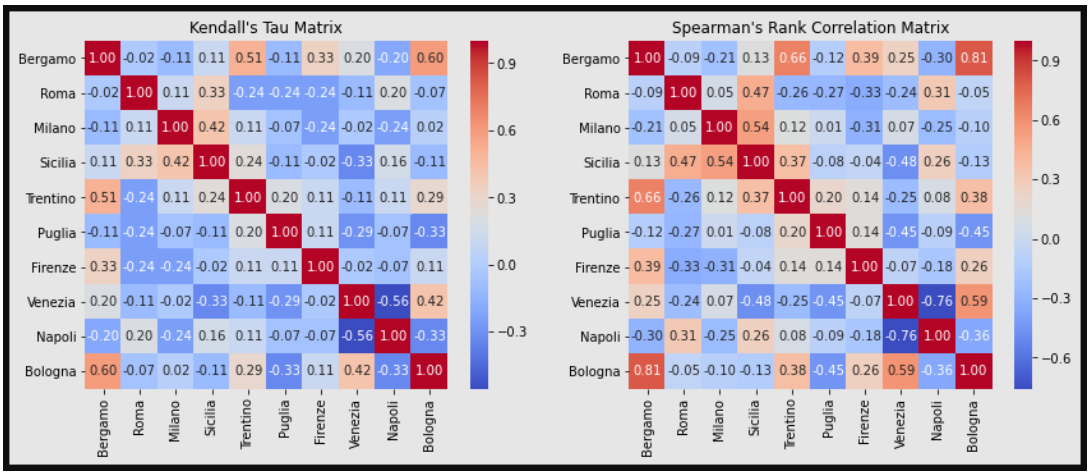


Figure 17: Rank Correlation

5.4.3 Frequency of Important Features

The most influential factors for the models were determined by the top 10 factors for each of the 10 datasets. The frequencies of these selected factors were then presented in a histogram, Figure 18. As a result, the most impactful factors, in descending order, were found to be the host's location, the number of bathrooms, the total number of hostings, the number of reviews, property types, the number of bedrooms, and the bookable.

6 Discussion and Comparison

6.1 Discussion

Challenges This project exhibited novelty through several distinctive features. It focused on implementing an Ensemble Stacking technique composed exclusively of ensemble

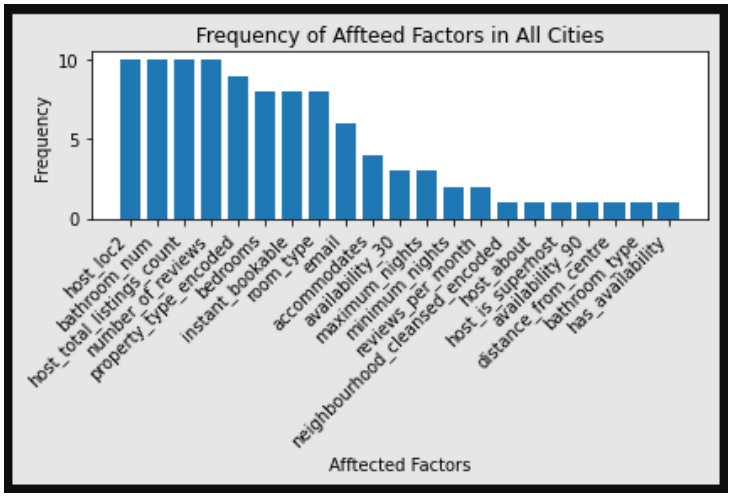


Figure 18: Frequency of top affected factors in all cities

machine learning models. The utilization of Airbnb data from Italy, a less frequently employed source for predictive models, is noteworthy. An aspect of particular significance lay in the inclusion of genetic algorithms, which have not often been employed in optimizing prediction regression models, and in the comparison with Bayesian Optimisation.

Results The achieved average performances for the three targeted models, namely Ensemble Stacking, Stacking with Bayesian Optimisation, and Stacking with Genetic Algorithm (GA), were 94.229, 87.709, and 89.771, respectively. These outcomes indicate the future possibility of deploying the implemented predictive models through this project. Notably, the Ensemble Stacking model, particularly prior to the incorporation of additional Hyperparameter Optimisation, demonstrated exceptional performance.

Issues and Limitations The utilization of the BayesSearchCV tool was straightforward and in the case of DEAP framework was not excessively intricate. However, Genetic algorithm is one of the heuristics techniques which are commonly utilized in the realm of deep learning. Moreover, since their algorithms were different, it was challenging to establish completely identical conditions between Genetic algorithm and Bayesian Optimisation. A limitation of the project was the hardware specifications, only 8GB of RAM. This constraint resulted in a modelling process that was comparably slower than a Colab environment. Consequently, it is advisable to undertake the modelling process in an environment equipped with 16GB or more RAM, or alternatively, in a Colab environment for efficient Jupyter notebook file running.

6.2 Comparison of Developed Models

The three main models were executed ten times each using data from across ten cities. The results of these models are illustrated in the graphs in Figure 19. The colours of the bars represent different cities, and within each city, there are three bars representing Stacking, Stacking+Bayesian, and Stacking+GA, respectively. For Adjusted-R^2 and Accuracy, higher bars indicate better performance, while for RMSE and MAE, lower bars signify better performance. Upon examining the results of modelling all ten datasets, a consistent trend is observed across all four evaluation metric graphs. The Stacking model demonstrates the highest performance, followed by the Genetic Algorithm and Bayesian

Optimisation.

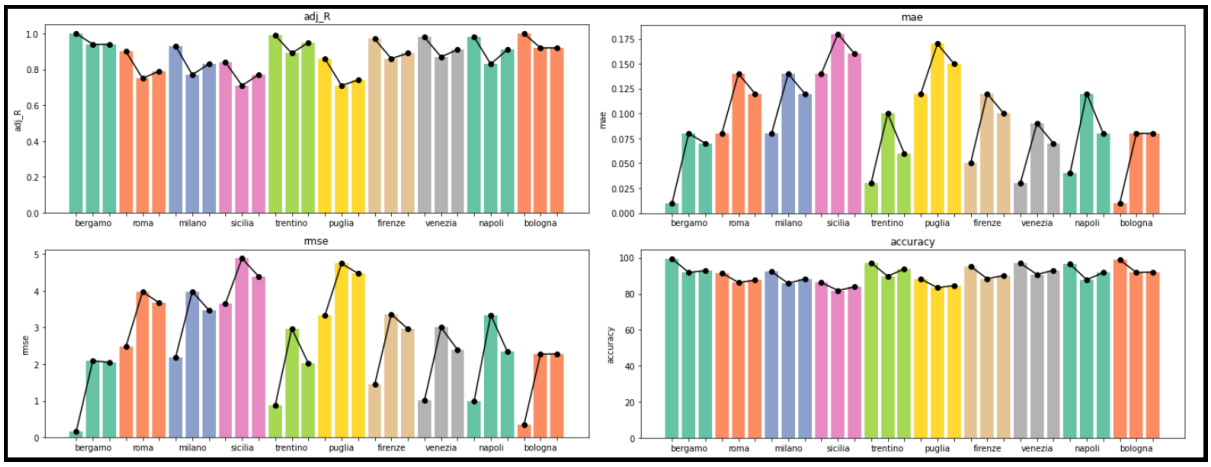


Figure 19: Stacking, Bayesian and GA Across 10 cities

Quantitative comparisons can be demonstrated in Table 3, which provides numerical values. The comparison of hyperparameter values used for modelling in each method is available in the pink-shaded section of the table.

	Ensemble Stacking	Bayesian Search	Genetic Algorithm
max_depth	6	2	10
learning_rate	0.300000012	0.1920478477	0.3661267729
gamma	0	0.7703399241	0.07173330605
n_estimators	190	764	190
Adjusted R^2	0.945	0.825	0.865
RMSE	1.647	3.46	3.004
MAE	0.059	0.122	0.101
Accuracy	94.229	87.709	89.771

Table 3: Result comparison of Hyperparameters and Evaluation values

6.3 Comparison of Developed Models and Existing work

In Table 4, comparisons between the results of the models conducted in this project and the models from reviewed studies in the Related Work section can be observed. For Ensemble Stacking, the performance of the reviewed study's models are slightly better. However, considering the differences in data and the composition of stacked models with various single models, it is challenging to conclude that this project's predictive model performance is inferior. On the other hand, for Bayesian Optimisation and Genetic Algorithms involving the Stacking model, the performance gap is more noticeable, emphasizing the relative deficiency in this project's prediction models. Nevertheless, it should be considered that the application of optimisation techniques to different data and combinations of stacking models remains an aspect to be taken into account. Regarding the utilized data in the experiment of this project, since the dataset is simple tabular data without a time series in the form of a CSV file although the size is sufficiently large, overfitting could be led potentially with the addition of more complex hyperparameters and cross-validation.

	Project			Literature Review		
	Ensemble Stacking	Bayesian Search	Genetic Algorithm	Ensemble Stacking	Bayesian Search	Genetic Algorithm
Adjusted R^2	0.945	0.825	0.865	0.978	-	0.995
RMSE	1.647	3.46	3.004	0.215	5.424	0.113
MAE	0.059	0.122	0.101	0.131	3.8042	0.0443
Resource	-	-	-	Meharie et al. (2022)	Lahmiri et al. (2023)	Luat et al. (2021)

Table 4: Result comparison of project models & Literature Review models

7 Conclusion and Future Work

Across all 10 datasets, the modelling results indicated that Random Forest, Gradient Boosting, LightGBM, and XGBoost exhibited a relatively similar trend with only minor differences in performance. However, with the application of Ensemble Stacking techniques, there was a notable surge in accuracy and a decline in error scores(RMSE, MAE). Subsequently, the model incorporating Bayesian Optimization witnessed a decrease from the previous method in performance, while the model incorporating Genetic Algorithm displayed a slightly elevated trend compared to the Bayesian approach as shown in Figure 20. In conclusion, to state RQ and sub-RQ how have been answered,

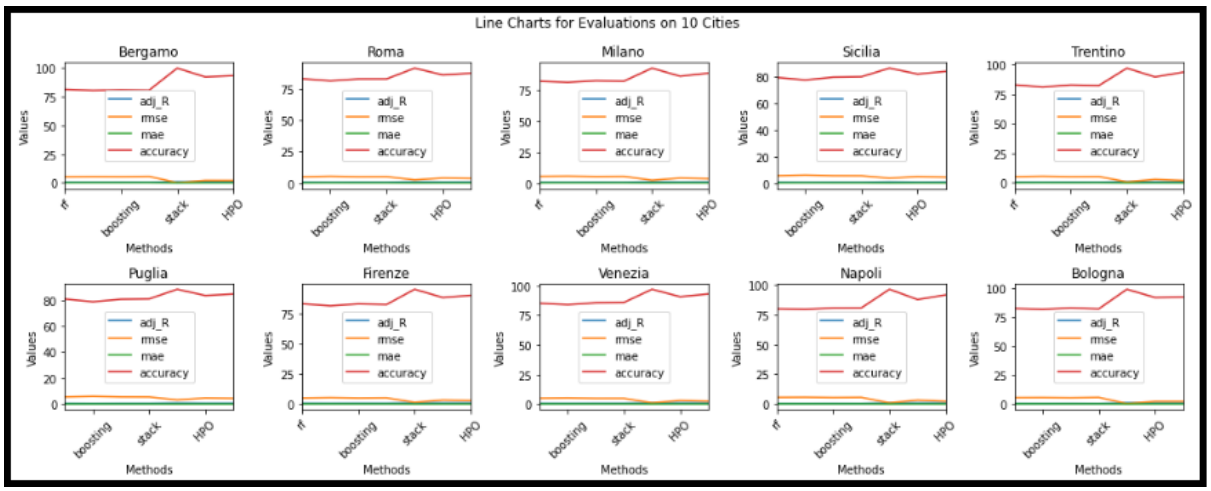


Figure 20: All methods by 10 different city datasets

Ensemble Stacking Technique did a good job of predicting profitability in all datasets. This method was much better than other single ensemble models and showed better results when the Bayesian optimization or Genetic algorithm was not combined. For sub-RQ, the median value and distribution of profitability in 10 cities were somewhat similar. However, the difference could be seen as significant through the Anova test, and the most affected factors conducted after the modelling was completed were the location of the host, number of bathrooms, a total host listing, property type, number of beds, and the possibility of instant booking and so on. Among the cities, Bergamo and Bologna showed the most similarities to each other in their top 10 influential factors affecting the ensemble prediction models.

Future Work: As future work, explore more hyperparameter Optimisation approaches can be explored. New feature selection also could be attempted from a different perspective, considering that another approach can sometimes lead to substantial changes in the model's predictive power. Different data sources will help a wider perspective of

the model's performance and robustness in various contexts provided by expanding the project's scope by incorporating data from various sources, such as different countries or regions. Moreover, real-world deployment can validate the usefulness of the prediction model in real-world situations and its performance. This effort is critical to confirming the model's applicability and dependability in actual decision-making scenarios.

References

Bruce, P., Bruce, A. and Gedeck, P. (2020). *Practical statistics for data scientists: 50+ essential concepts using R and Python*, O'Reilly Media.

Cao, B., Yang, B. et al. (2018). Research on ensemble learning-based housing price prediction model, *Big Geospatial Data and Data Science* **1**(1): 1–8.

Chowdhury, S., Lin, Y., Liaw, B. and Kerby, L. (2022). Evaluation of tree based regression over multiple linear regression for non-normally distributed data in battery performance, *2022 International Conference on Intelligent Data Science Technologies and Applications (IDSTA)*, IEEE, pp. 17–25.

Cocola-Gant, A. and Gago, A. (2021). Airbnb, buy-to-let investment and tourism-driven displacement: A case study in lisbon, *Environment and Planning A: Economy and Space* **53**(7): 1671–1688.

George, N. (2021). Practical data science with python: learn tools and techniques from hands-on examples to extract insights from data, *(No Title)* .

Géron, A. (2022). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*, " O'Reilly Media, Inc.".

Graczyk, M., Lasota, T., Trawiński, B. and Trawiński, K. (2010). Comparison of bagging, boosting and stacking ensembles applied to real estate appraisal, *Intelligent Information and Database Systems: Second International Conference, ACIIDS, Hue City, Vietnam, March 24-26, 2010. Proceedings, Part II 2*, Springer, pp. 340–350.

Hati, S. R. H., Balqiah, T. E., Hananto, A. and Yuliati, E. (2021). A decade of systematic literature review on airbnb: the sharing economy from a multiple stakeholder perspective, *Heliyon* **7**(10).

Jamil, S., Mohd, T., Masrom, S. and Ab Rahim, N. (2020). Machine learning price prediction on green building prices, *2020 IEEE symposium on industrial electronics & applications (ISIEA)*, IEEE, pp. 1–6.

Jha, S. B., Babiceanu, R. F., Pandey, V. and Jha, R. K. (2020). Housing market prediction problem using different machine learning algorithms: A case study, *arXiv preprint arXiv:2006.10092* .

Khosravi, M., Arif, S. B., Ghaseminejad, A., Tohidi, H. and Shabanian, H. (2022). Performance evaluation of machine learning regressors for estimating real estate house prices.

Kim, J. and Yoo, S. (2019). Software review: Deap (distributed evolutionary algorithm in python) library, *Genetic Programming and Evolvable Machines* **20**: 139–142.

Lahmiri, S., Bekiros, S. and Avdoulas, C. (2023). A comparative assessment of machine learning methods for predicting housing prices using bayesian optimization, *Decision Analytics Journal* **6**: 100166.

Lewinson, E. (2020). *Python for Finance Cookbook: Over 50 recipes for applying modern Python libraries to financial data analysis*, Packt Publishing Ltd.

Liu, Y. (2021). Airbnb pricing based on statistical machine learning models, *2021 International Conference on Signal Processing and Machine Learning (CONF-SPML)*, IEEE, pp. 175–185.

Liu, Y., Wu, Y., Su, L., Li, W. and Lei, J. (2021). Stacking-based ensemble learning method for house price prediction, *Proceedings of the Computational Methods in Systems and Software*, Springer, pp. 224–237.

Luat, N.-V., Han, S. W. and Lee, K. (2021). Genetic algorithm hybridized with extreme gradient boosting to predict axial compressive capacity of ccfst columns, *Composite Structures* **278**: 114733.

Mansoori, A., Zeinalnezhad, M., Nazarimanesh, L. et al. (2023). Optimization of tree-based machine learning models to predict the length of hospital stay using genetic algorithm, *Journal of Healthcare Engineering* **2023**.

Meharie, M. G., Mengesha, W. J., Gariy, Z. A. and Mutuku, R. N. (2022). Application of stacking ensemble machine learning algorithm in predicting the cost of highway construction projects, *Engineering, Construction and Architectural Management* **29**(7): 2836–2853.

Mora-Garcia, R.-T., Cespedes-Lopez, M.-F. and Perez-Sanchez, V. R. (2022). Housing price prediction using machine learning algorithms in covid-19 times, *Land* **11**(11): 2100.

Muller, A. C. and Guido, S. (2017). *Introduction to machine learning with Python*, O'Reilly.

Phan, T. D. (2018). Housing price prediction using machine learning algorithms: The case of melbourne city, australia, *2018 International conference on machine learning and data engineering (iCMLDE)*, IEEE, pp. 35–42.

Raschka, S. and Mirjalili, V. (2017). Python machine learning second edition.

Shahhosseini, M., Hu, G. and Pham, H. (2020). Optimizing ensemble weights for machine learning models: A case study for housing price prediction, *Smart Service Systems, Operations Management, and Analytics: Proceedings of the 2019 INFORMS International Conference on Service Science*, Springer, pp. 87–97.

Wade, C. and Glynn, K. (2020). *Hands-On Gradient Boosting with XGBoost and scikit-learn: Perform accessible machine learning and extreme gradient boosting with Python*, Packt Publishing Ltd.

Wu, X. and Yang, B. (2022). Ensemble learning based models for house price prediction, case study: Miami, us, *2022 5th International Conference on Advanced Electronic Materials, Computers and Software Engineering (AEMCSE)*, IEEE, pp. 449–458.

Yang, L. and Shami, A. (2020). On hyperparameter optimization of machine learning algorithms: Theory and practice, *Neurocomputing* **415**: 295–316.

Zhang, Q. (2021). Housing price prediction based on multiple linear regression, *Scientific Programming* **2021**: 1–9.