**Housing Price Prediction**

Zi Wang

CSCI 6397

**Introduction**

This project is an implementation of supervised machine learning algorithms by building some regression models for forecasting house price depends on some house features include average area income, average house age, average area number of rooms, average area number of bedrooms, average population, and address. The reason I am undertaking this project is I want to practice the machine learning knowledge that I learned from this course and try to understand more in machine learning such as different types of regression models, the tools or packages that can be helpful in machine learning and how to complete a machine learning project by using python.

In this project I will implement the process of data description, data preprocess, data standardization, create training and test data, and then apply into machine learning algorithms, training and test with different type of regression models, and finally evaluate the performance of models. By completing this project, I want to know which feature is impact house price most and how regression models perform, which model is most accurate and also the time they take to complete house price prediction.

**Data Introduction**

The data I use for this project is a dataset with 5000 datapoints and 6 attributes, the dataset is downloaded from kraggle. Each row of datasets represents one house information include, average area income, average house age, average area number of rooms, average area number of bedrooms, average population, address, and price. Price is the prediction target, and to predict house price, I analyze the relationship between price and all of other features.

First, I load the dataset into program and get some basic information of my dataset. I use data.info () to get the datatype of all the data, and most features are float64 but the datatype for address is object. From the output, I can see how many non-null data are stored under each feature. The data has 5000 rows with 7 columns, hence, each feature should have 5000 data stores under its column, if any features had less than 5000 counts, that means there is a data missing for that feature, so we need to fill it with some value or remove the row. In my dataset, there is not any data missing because each features have 5000 data. After that, I want to see the correlation between variables. The relationship between variables can help to make assumptions that how price could reflect to features.

A picture containing chart

Description automatically generated

**Figure1.** the correlation within variables

Text

Description automatically generated

**Figure2.** attributes correlation value

From the correlation result, I can see the importance of each feature and which feature is the main factor that affects house price. Before any data analysis, I estimate the house age, and number of rooms are the most important features for price. If the house is older, the price should be cheaper and if the number of rooms are less, and the price also should be cheaper. However, the variable correlation result shows the most important feature that affect price is the area income which is 0.64 and the house’s age get 0.45. I also realize there is a meaningless feature which is address. That is nothing to do with my price prediction because the whole column is just some random house address. I decide to remove Address variable from my dataset.

Before I am working on my data preprocessing, I have to make sure that I got the meaningful variables which means they will be affect the price when happen any value changing. Any variable changes will lead to some positive or negative outcome on the house price’s value. The final features are average. Area income, house age, population, number of rooms, and number of bedrooms. I make a new correlation analyze for the final features. To make it easy to understand, I use the function seaborn. pairplot () which will create a grid of Axes that each numeric variable in my dataset will by shared across the y-axes across a single row and the x-axes across a single column. That function will draw a univariate distribution plot graph to show the marginal distribution of the data in each column.

From the seaborn graph, most of variables have a linear relationship with price because the points in the graph are organized in a circle however the feature number of bedroom seem like there is no linear relationship with price. Because of that, the regression models can be the applicable models for the following machine learning process.

**Data Preprocess**

Prepare the suitable data format for the machine learning algorithms are the main part of my data preprocessing. In my dataset there are not much non-numeric attributes, the only one is the address. I decide to remove it because I think they have no relationship with price. I am not hundred percent sure about that, but for the participant exercise, I do some research about the data encoding. The One-Hot Encoder is the method I participated in my project. According to the research, Sci Kit Lean’s OneHotEncoder are recommend for machine learning models(Yadav, D. 2021) I know there was another way is panda’s get\_dummies but the author claim that get dummies will run into a whole host of problems if the levels of categories in the training and test data set differ. One Hot Encoder can solve this problem with the options that can be set for categories and handling unknows. Which is harder to use compared to get dummies, but it produces a better transformation. One hot encoder will convert each category value into a new column and assigned a 1 or 0 notation for true or false value to the column. This approach eliminates the hierarch or order issues, but the con is adding additional columns for each category level.

**Data Standardization**

Beside of data encoding, I implement data scale to standardize data set. Scaling of features is an essential step in modeling the algorithms with the data sets. The data obtained contains various dimensions and scales altogether. Such as the attributes of average of number of beds just a small integer, but the number of population and average area income are much larger. The different scales of the data attributes highly affect the modeling of a data set. Which will lead it biased outcome of predictions in terms of misclassification error and accuracy rates. Therefore, scale the data is necessary before feed into the machine learning modeling.( Mulani, S. 2022)

The standardizations are a scaling technique where it makes the data scale -free by converting the statistical distribution of the data into approachable format. The formula is z = (x - u) / s. where x is the sample data, u is the mean of the training samples and s stands for standard deviation of the training samples. What I do here is, in python import preprocessing library from sklearn, and then call the function of preprocessing. StandardScaler (). If I want to transform the attributes, I just have to call the pre\_process.fit\_transform(attributes).

**Training and Test data**

The house price prediction is the competition goal for different type of models. the process of prediction will be completed by training regression models with the training data set, and then test the model with test data. In my project, I use the package of sklearn. model\_selction and import the train\_test\_split to create my training and testing data set by splitting the whole data set into 80% training and 20% testing which means training data consists 4000 house prices with their 5 features each, and 1000 in test data. In detail, the training data will teach models how the regression look like, after the models learned the relationship between house price and their features, the model will be predicted the house prices base on the attributes from test data. And then, the difference between the true house prices from test data and the predicted house price will be the grade of the model. The model with less difference means it has the higher performance and accuracy.

**Machine Learning Algorithms**

The purpose of my project is to use machine learning algorithm to predict house price, which is a numerical outcome , so I plan to select certain regression models with supervised machine learning algorithm. The models that I implement in my project include linear regression, decision tree regression, random forest regression, RANSAC regressor, ridge regression, and gradient boosting regressor.

**Linear regression**

From the data understanding process, I got the graph and tables to show that the house price seems to be in a strong linear relationship with variables of average area income, average aera house age, average number of rooms and area population, and poor relationship with average area number of bedrooms. Due to that estimation, the first model I put in machine learning process is linear regression.

Linear regression will be the baseline model then built on this experience to tune my candidate models. I use the package linear\_model from sklearn, which let me use the linear regression model to predict. What I do here is I fit my training data into linear model and to teach the model by using training data set examples. After this process, I get some numbers that linear model learns from training data set such as the coefficients (2.28, 1.64, 1.22, 1.10, 1.51) and intercept (1233838). The intercept is kind high because the price is a much larger number than other features. I have a solution to make it looks more reasonable which is transform all the price with log which will reduce all price range in to 1 and 15. The mean square error is 9849555082.356997 which looks not make sense from my training data set.

The new result is intercept for training dataset is 12.98, coefficients are [0.20549109 0.15053429 0.11145978 0.00077271 0.1362582]

The coefficients base on real training data without log price are

**Table1**: Linear model’s coefficient with original price value:

|  |  |
| --- | --- |
|  | Values |
| Avg. Area Income | 230822.3486 |
| Avg. Area House Age | 164527.9109 |
| Avg. Area Number of Rooms | 120475.6286 |
| Avg. Area Number of Bedrooms | 1878.633928 |
| Area Population | 151471.2749 |

that looks wired I realize that features are standardized but price is not, so I standardize price by using this formula

XNormed = (X - X. mean())/(X.std())

I got new coefficients and intercept

**Table2**: Linear model’s coefficient with standardized Price value

|  |  |
| --- | --- |
| for training dataset: | Values |
| intercept | 1.438305 |
| Avg. Area Income | 0.644763 |
| Avg. Area House Age | 0.462687 |
| Avg. Area Number of Rooms | 0.341011 |
| Avg. Area Number of Bedrooms | 0.007427 |
| Area Population | 0.423597 |

This tells me that there are 5 outliers that can impact final prediction

The price = 1.438305+ Avg. Area Income\*0.644763+ Avg. Area House Age\*0.462687+Avg. Area Number of Rooms\*0.341011 +Avg. Area Number of Bedrooms\*0.007427+Area Population\*0.423597

After that I done the training, I test the model with test data set. I use this model to predict house price and compare with the true house price from the test data. Calculate the mean square error within them to see how the model is doing.

I got **MSE=0.07755989902508868**

Beside of MSE, I also use R-square to evaluate my model. R-square can dig further into what is happening to models. It does not consider how individual data value performs which is what MSE doing, comparing each prediction number with true number and then get the mean absolute error value. R-square does not tell us about how far or close each predicted value is from the real data, it tells that how much of our target is being captured by our model. According to R-square explanation R-square quantifies how much of the variance of the dependent variable is being explained by the model. The R-square  metric varies from 0% to 100%. The closer to 100%, the better. (Jain, S. 2020) If the R-square value is negative, it means it doesn't explain the target at all.

To calculate R-square, I simply import r2-score from sklearn,metrics. I use linear regression model to get predicted house price and then put predicted price and true price into r2-score () , my model shows R-square result is

**linear\_regression r2\_score: 0.9218478696681118**

which is pretty accurate and satisfied because this model explains 92% of the test data.

**Ridge Regression**

Ridge regression or Tikhonov regularization is the regularization technique that performs L2 regularization. It modifies the loss function by adding the penalty (shrinkage quantity) equivalent to the square of the magnitude of coefficients. (Great Learning Team. 2022.10)

**Min(||Y – X(theta)||^2 + λ||theta||^2)**

Lambda is the penalty term. λ given here is denoted by an alpha parameter in the ridge function. So, by changing the values of alpha, we are controlling the penalty term. The higher the values of alpha, the bigger is the penalty and therefore the magnitude of coefficients is reduced.

For implementing ridge regression, I import the package linear\_model.Ridge(), then put training data into ridge model to start training. I got the training process’s result:

**Table2.**Ridge model’s coefficient

|  |  |
| --- | --- |
| ridge\_model Values |  |
| Avg. Area Income | 0.656079 |
| Avg. Area House Age | 0.46949 |
| Avg. Area Number of Rooms | 0.343485 |
| Avg. Area Number of Bedrooms | 0.006658 |
| Area Population | 0.427289 |
| intercept | 2.277 |

After that, I test trained ridge model with test data set and calculate the R-square score. The result is ridge regression score: 0.92182610543925. the result is pretty much similar with linear regression. The model explains 92% precent of the test data.

**Decision Tree Model**

Decision tree is another powerful supervised machine learning algorithms that can be used for classification and regression problems. The model is based on decision rules extracted from the training data. In regression problem, the model uses the value instead of class and mean squared error is used to for a decision accuracy(Yadav, D. 2021).

The package I use for decision tree model is DecisionTreeRegressor from sklearn.tree. I fit tree model with normalized training data, and then use the trained model to predict house price from test data. After that I calculate r-squared score which to represent the performance. The result I got for decision tree model is

**decision\_tree\_model score: 0.7702665751856679**

which is lower compared to linear and ridge model, this model only explains 77% of the test data.

**Random Forest Regression**

Random Forest Regression is also a supervised learning algorithm that uses ensemble learning method for regression. Ensemble learning method is a technique that combines predictions from multiple machine learning algorithms to make a more accurate prediction than a single model. How random forest algorithm works(KUMAR, S. 2018):

1. Pick at random k datapoint from training data
2. Build a decision tree associated to these k datapoints
3. Choose the number N of trees you want to build and repeat steps 1 and 2.
4. For a new data point, make each one of your N-tree trees predict the value of y for the data point in question and assign the new data point to the average across all of the predicted y values.

The package I use is RandomForestRegressor import from learn.ensemble. first, fit random forest model with training data, and then the model will implement random forest algorithm by using training data, after that, I use this model to predict the price from test data. Finally, I calculate the R²  score to show the performance. The random forest model’s result is

**random forest model score: 0.8792002474763132**

which means this model is fitted 87.9% of the test data.

**RANSAC Regressor**

The RANSAC stands for **RAN**dom **SA**mple **C**onsensus – a regression algorithm that trains the model by handling the outliers. An outlier is a data point that is noticeably different from the rest.( Alam, B. 2022) They represent measurement errors, bad data collection, or show variables not considered when collecting the data. Most [Machine Learning](https://hands-on.cloud/quick-introduction-to-machine-learning/) algorithms are highly affected by outliers in the dataset because they are influencing algorithms’ predictions. The process steps of RANSAC regression algorithm are(Kumar, A. 2020a):

1. Select a random number of examples to be inliers and train the model.
2. Test all other data points against the trained model
3. Out of all the data points tested in step 2, select the points as inliers which fall within a user-given tolerance. In scikit-learn, median absolute deviation (MAD)is used for selecting the new points as inliers.
4. Retrain the model with all inliers data
5. Estimate the error of the retrained model versus the inliers.
6. Repeat step 1 to step 5
7. Terminate the algorithm execution if the model performance meets a certain user-defined threshold or if a fixed number of iterations were reached

(Cited from <https://vitalflux.com/ransac-regression-explained-with-python-examples/#RANSAC_Regression_Algorithm_Details>)

I use the package of RANSAC regressor from linear\_model. The package allows me to create RANSAC model by fit training data. After I trained this model, I use this model to predict house price in test data and then calculate R²  score:

**RANSAC\_model score: 0.9191515398252306**

The result shows that this model is trained well and explained 91.9% of the test data.

**Gradient Boosting Model**

Same as decision tree model, Gradient Boosting model is also can be used to train models for both classification and regression problems. This algorithm is used to generate an ensemble model by combining the weak learners or weak predictive models. (Kumar, A. 2020b) Gradient Boosting Regression algorithm is used to fit the model which predicts the continuous value.

It builds an additive model by using the multiple decision trees of fixed size as weal learner or weak predictive model. The process of the training will start with some constants such as mean value of the target values. In subsequent stages, the estimators or decision trees are fitted to predict the negative gradients of the samples. For each iteration, the gradients are updated. And then there is a learning rate to shrink the outcome or the contribution from each subsequent trees or estimators.

In my implementation, I import GradientBoostingRegressor from sklearn.ensemble, then I start to train the model by fitting with training data, and then calculate the R²  score for the measurement.

**gradient\_boosting\_score : 0.8994319964829833**

Compared to the best possible score 1.0, this model shows a satisfied performance.

**Conclusion**

In this project, I implement supervised machine learn algorithms to predict house price by training regression models include RANSAC regressor,linear regression, ridge regression, Gradient Boosting Regressor, random forest regressor and decision tree regression. The result shows that the linear regression produce the best result, which is 0.92, the lowest score is 0.74 which is decision tree model. The result claim that uses the attributes from my data set to predict house price, the linear regression model will get approximately 92% accurate prediction, but decision tree only gets 74.7% .

**Table4**. The performance of models

|  |  |  |
| --- | --- | --- |
| Models | Scores | Time |
| linear regression | 0.921848 | 0.015625 |
| RANSAC regressor | 0.915967 | 0.203125 |
| ridge regression | 0.909777 | 0.078125 |
| GradientBoostingRegressor | 0.885229 | 0.8125 |
| random forest regressor | 0.856662 | 1.171875 |
| decision tree regression | 0.747033 | 0.015625 |

**Figure3.** Regression models performance

I think the result is satisfied, but I do think that there are some limitations in my project. The data set I use is not very large which just include 5000 data points with 6 features. The lack of features limits the accuracy for house price prediction. A larger data set with various types of features can help to train models better. In my project, the variables of average area income impact the price highly, which is make sense because higher average area income means the location is expensive so the house price will be higher even the house may be smaller and older.

From the project, I implement the theory of machine learning algorithms that I learn from this course, and when I am working on this project, I also learn some different types of regression models. This project helps me to understand how to build a supervised machine learning models and during the implementation process, I know some useful python packages for regression models. I am also getting more familiar with Python codes and the powerful packages or libraries attract me to learn more how to program with python. During this project, I read a lot some open-source samples that teach me what packages I should use, and how the syntax looks like. I do not have any background with python, before this project I prefer to coding with Linux and C++ because I have never built anything with python but right now, this project not just help me to understand more in machine learning but also grow me the interest of coding with python. I think python is a good choice for me to do something with data science in the future.

**Reference**

Ade-Ojo, J. (2021, December 27). *Predicting House Prices with Machine Learning - Towards Data Science*. Medium. https://towardsdatascience.com/predicting-house-prices-with-machine-learning-62d5bcd0d68f

Alam, B. (2022, June 27). *Using the Random Sample Consensus (RANSAC) algorithm in Python*. Hands-On-Cloud. https://hands-on.cloud/using-the-random-sample-consensus-ransac-algorithm-in-python/

Great Learning Team. (2022, November 16). *Ridge Regression Definition & Examples | What is Ridge Regression?* Great Learning Blog: Free Resources What Matters to Shape Your Career! https://www.mygreatlearning.com/blog/what-is-ridge-regression/

Jain, S. (2020, October 18). *A comprehensive beginners guide for Linear, Ridge and Lasso Regression in Python and R*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2017/06/a-comprehensive-guide-for-linear-ridge-and-lasso-regression/

Kumar, A. (2020a, October 2). *RANSAC Regression Explained with Python Examples*. Data Analytics. https://vitalflux.com/ransac-regression-explained-with-python-examples/

Kumar, A. (2020b, December 14). *Gradient Boosting Regression Python Examples*. Data Analytics. https://vitalflux.com/gradient-boosting-regression-python-examples/

KUMAR, S. (2018, September 12). *USA Housing Price Prediction For Beginners*. Kaggle. https://www.kaggle.com/code/suprabhatsk/usa-housing-price-prediction-for-beginners/notebook

Landup, D. (2022, September 5). *Linear Regression in Python with Scikit-Learn*. Stack Abuse. https://stackabuse.com/linear-regression-in-python-with-scikit-learn/

Mulani, S. (2022, August 3). *Using StandardScaler() Function to Standardize Python Data*. DigitalOcean Community. https://www.digitalocean.com/community/tutorials/standardscaler-function-in-python

Singh, S. (2022, February 18). *Understanding the Bias-Variance Tradeoff - Towards Data Science*. Medium. https://towardsdatascience.com/understanding-the-bias-variance-tradeoff-165e6942b229

*sklearn.preprocessing.StandardScaler*. (n.d.). Scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

*USA\_Housing*. (2017, November 19). Kaggle. https://www.kaggle.com/datasets/vedavyasv/usa-housing

Yadav, D. (2021, December 12). *Categorical encoding using Label-Encoding and One-Hot-Encoder*. Medium. https://towardsdatascience.com/categorical-encoding-using-label-encoding-and-one-hot-encoder-911ef77fb5bd