Scientific Research Publishing

International Journal of Intelligence Science, 2020, 10, 9-21
https://www.scirp.org/journal/ijis
ISSN Online: 2163-0356
ISSN Print: 2163-0283

Negotiation Assistant Bot of Pricing Prediction Based on Machine Learning

Tingwei Liu[1], Zheng Zheng[2*]

[1]Sichuan University, Chengdu, China
[2]East China University of Science and Technology, Shanghai, China
Email: *zhengzheng191226@outlook.com

How to cite this paper: Liu, T.W. and Zheng, Z. (2020) Negotiation Assistant Bot of Pricing Prediction Based on Machine Learning. *International Journal of Intelligence Science*, **10**, 9-21.
https://doi.org/10.4236/ijis.2020.102002

Received: January 14, 2020
Accepted: March 30, 2020
Published: April 2, 2020

Abstract

Artificial intelligence (AI) has been used to develop and advance numerous fields and industries, including finance, healthcare, education, transportation and more. However, in the business negotiation field, such as bargain, the AI has not yet exerted its power. In order to explore the application of AI into business negotiation, we have built an intelligent robot that can help customers that lack negotiation skills when bargaining in their shopping sceneries. This bot can make decision by itself via price prediction function implemented by machine learning algorithms and the tool of decision tree. As a result, our bot has got a positive performance during a used car trade. Although the algorithm of the project is relatively simple, its main contribution is to show the potential application of AI in the business negotiation. We believe that it can provide ideas and directions for the future development of business negotiation robot.

Keywords

Negotiation, Price Prediction, Machine Learning, Neural Network

1. Literature Review

Before 1980, machine learning algorithms began to develop, such as Nearest Neighbor [1] and Bayesian classifier [2]. After that, machine learning has become an independent discipline, integrating previous machine learning algorithms, and many new algorithms have been proposed, such as Convolutional Neural Network [3] and Support Vector Machine [4]. These machine learning algorithms can be applied to many fields, such as Go game [5], face recognition [6] and medical diagnosis [7]. The applications of them have made a great forward in real-time interaction, e.g. competing with professional player and face check-in systems, and future prediction, e.g. forecasting the probability of intensive disease. However,

due to the complexity and randomness of human activity in real life, there are limited applications that combining machine learning with actual negotiation. In view of this, there is still a lot of exploration space in this field.

2. Introduction

Negotiation is an innate skill of human beings [8]. It plays important roles both in daily communications and trade cooperation. Therefore, it is necessary for everyone to master this kind of skills. However, on one hand, due to existing explicit and implicit rules underlying negotiations, it is not very easy to perform well in every situation, especially for the introverted persons. On the other, AI is very good at dealing with things that have clear rules but complicated processes. For AI, it can be programmed to respond to a set of assumptions and a set of data that have been presented in the past, thus predicting value and calculating these numbers in a more precise way than humans. This predicting data can be used as a standard reference for special features in negotiations. Consequently, it is a natural way to combine AI with negotiation, which can be used to automatically optimize people's decision when negotiating.

Here, due to there are many negotiation processes in the trading and the car price can be a predictable object affected by the car attributes, we choose the real problem of second-hand car transaction as the design basis. It can help the novice to make a decision when buying second-hand car and give the corresponding evaluation to the customer for feedback. The whole program can be divided into four parts: webpages, Price Bot, Advisor Bot and Judge Bot. As a result, our bot has got a positive performance during a used car trade. We believe that our research value is the primary combination of negotiation and machine learning algorithms. The significance of this research lies in: 1) providing an approximately real-time help to negotiators and giving them several negotiation ideas; 2) quickly finding the best alternative and telling them when to say "No" and go away; 3) realizing the preliminary combination of AI and negotiation.

3. Current Problems in Negotiation and Our Solutions

Negotiation is an important part of people's daily life. However, generally speaking, people cannot remember all the details in the process of negotiation, thus they will make wrong judgment or hesitate. In addition, many inexperienced people are lack of negotiation skills. In this second-hand car case, negotiators do not remember all the details of the car given by the shopper, including mileage, release, specific car name, etc. Maybe they only remember the price. Our solution is: first use our app to record these data, which will be put into the machine learning model after cleaning to get a fair market price, then we can find the best alternative which will be provided to negotiators through search algorithm. After that, implementing decision tree to help users negotiate. Therefore, our AI blocks many details that need to be memorized for the negotiator, offers a fair market price of the car to the negotiator as reference, provides suggestions for the negotiator and selects the best alternative for him.

4. Method and Data

The main body of the whole program is performed by Python. In the data collection part, Crawler based on Python's request library is used to grab data. The data come from several second-hand car transaction website, including http://www.guazi.com, http://www.kx.cn and http://www.che168.com. Html and JavaScript is used in the webpage and Python in the back end to process the corresponding Get and Post requests from the front end. In the data processing part, we use the Python's pandas library to convert the CSV file into data frame format, and process the discrete data, such as brand, car name, etc., into One-Hot code [9], which is the basis of later price prediction. In the part of price prediction, we use a variety of Machine Learning algorithms [10], including NN (Nearest Neighbor), KNN (K-Nearest Neighbor), SVM (Support Vector Machine), ANN (Artificial Neural Network). In the Judge Bot part, we use the "what-if" clause [11], and the decision tree provides the negotiation logic. In the Price Bot part, we wrote some basic functions to score the negotiation results of customers.

5. Project Simulation

5.1. Car Selection

When the customer comes to the store, the sellers inquire about the customer's demand for the car and retrieve the matching vehicle in the warehouse through the search engine we built. At this time, customers will get the basic attribute information of the car, including the brand, model, service life, mileage, delivery date, insurance status, and after-sales guarantee. The output results will be ranked according to the priority of different attributes.

5.2. Price Prediction

After the customer gets this information, he can input the attributes of the car into the Price Bot. Price Bot will give a more radical recommendation price through ML algorithm. The price is to give the customer a basic price strategy to avoid losing too much in price negotiation.

5.3. Negotiations

After getting the recommended price, Advisor Bot will advise the users step by step according to the pre-made negotiation strategy. When the user fails to make a breakthrough in the price, Advisor Bot will suggest the user to do some tradeoff, such as upgrade the insurance level or get more after-sales support.

5.4. Final Judge

The Judge Bot will score the final deal, and users can decide whether to continue negotiations or direct deal based on their scores. In order to provide users with a reference standard, we will look for alternative options that match the needs of users in the trading market and score them with the same standard, so that users can have an alternative choice.

6. Data Processing

6.1. Data Collection

First, we should collect enough data for training the model, and I used a technique called Crawler, a practical tool based on python's library, to grip useful information from website [12]. Then I get two data sets for future use. One of them is a respectively large dataset containing 10K lines of data. This dataset will be used to train Machine Learning Model. Another is a small dataset including 1K lines of data. It acts as a warehouse of second-hand cars. So, we have two tables. In each table, Columns are split into two parts: Attributes and Label. After training by dataset, Machine Learning Model will use Attributes to predict Label. All the columns except the Label are the related parameters of cars, such as brand, type, mileage, gear and so on, and the price's column becomes label. Each row represents one Sample. The dataset should be split into two parts: train set and test set. Generally, the size of train set is 70 percent of the whole dataset and the rest is test set. To explain these two sets, let's make an analogy, if Machine Learning Model is a student, the train set is like a teacher and the test set like a final exam which will judge the accuracy of training result. The larger dataset is separated into four parameters: x_train (the intersection of train set and Attributes' columns), y_train (the intersection of train set and Label's column), x_test (the intersection of test set and Attributes' set) and y_test (intersection of test set and *Label's* column). The larger dataset's division is shown in **Figure 1**.

6.2. Data Cleaning

Because the datasets are gripped from website, inevitably, there are a lot of dirty data in them. The accuracy will be intensively influenced by them. As a result, cleaning data is our next step. Dirty data have various types. The First is vacancy, and one of the solutions for these types of data is to fill in the blanks by calculating the average value of columns or just cut them out of the dataset. The former method is pragmatic but more time-consuming. The second problem is repetition. This will make the efficiency low. To solve it, people should delete superfluous data manually. The third problem is that outlier exists. For example, normal release is less than 3 liter, but it appears that some abnormal release data are larger than 50 liter which should be eliminated immediately.

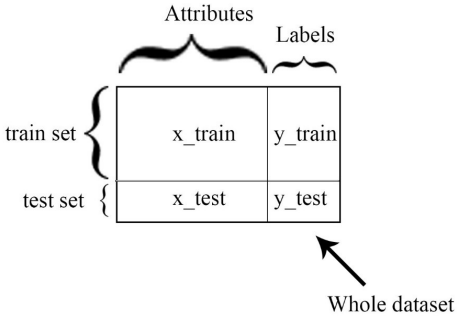

Figure 1. Larger dataset's division.

Another problem is that the data is not available. For example, there is a column of VolkswagenSatana1.5L34.2. This kind of data obviously combines brand, type, release, used years, mileage and other variables, so it's unusable. A simple way to deal with it is to delete this kind of data.

6.3. Data Transformation

The third step is to convert words into numbers, because Machine Learning Models have no way to recognize words. There are two ways to deal with text data. One is to number the words. The number increase by degrees. For second-hand cars, the "brand" data, Volkswagen number is 0, Honda number is 1 and so on. This method has some drawbacks. Also take "brand" as an example, it artificially defines the distance between two brands, e.g. Volkswagen is 0 and Hyundai is 100. Then Machine Learning Model will think that the distance between two brands is 100 (not actual 100, because there may be some algorithms to calculate the distance), but in fact, the two brands are only two symbols. Another method is to use One-Hot Code to process text. The basic idea is to record several values in the same size of columns. Assuming that there are three brands in used cars, Volkswagen, Honda and Hyundai, then the "brand" column is divided into three columns: Volkswagen is 001, Honda is 010 and Hyundai is 100. The advantage of this method is that the distance among all brands is equal, which means the distance between any two brands is equal. The disadvantage is that when there are too many different text attributes, the number of columns will increase rapidly, making the distribution of points extremely sparse, which is not conducive to training Machine Learning Model effectively. In our project, I use One-Hot Code to implement data transformation, because, after searching other experiment results, One-Hot Code performs better. The basic idea of the function that turns words into One-Hot Code is to put a standard line of data, such as Volkswagen, Santana, 1.5, 4, 3, 5.88 (brand, type, release, used years, mileage and price) in the first line of Data frame, a python's data type, to determine whether the first item in each column is a number (all real number). If so, skip that column. If not, turn an entire column into One-Hot Code. The Pandas library in Python has get dummies method that can break an entire column into One-Hot Code.

7. Module Analysis

7.1. Webpages

The main function of webpages is to interact with users. Webpages are implemented in HTML, which is more convenient than python's console. The part of dynamic webpages is accomplished by JavaScript. The static pages use HTML tags and the interface is adjusted by Cascading Style Sheets (CSS). The interaction between the front end and the back end is realized by Flask, a python's library for web, using Form to transfer data. Users enter some parameters and select priorities, or say preference, at the front end, and these priorities are passed to the back end and recorded by our system, which determines the data filtering logic and the assistant strategy of Advisor Bot. The first page is that the seller

will show to the user when the user comes to a used car store. The function of this page is to filter the car in the warehouse according to the user's priorities. The data in the warehouse comes from the smaller dataset. When a car is selected, the user will open his own webpage, input the relevant data of selected car, press the "Predict" button and then get a prediction (fair market) price, then he/she can "turn to Advisor Bot". The price prediction is achieved by Price Bot.

7.2. Price Bot

The function of Price Bot is to predict a fair market price according to the attributes entered by the user, give the corresponding Accuracy, and display the price on the webpage. We consider many algorithms that used in Machine Learning Model. Algorithms include Nearest Neighbor (NN), K-NN, Support Vector Machine (SVM) [13], Artificial Neural Network (ANN). After comparison, it is found that the Accuracy of ANN is the highest, but it takes 30 seconds, so our project adopts 5-NN algorithm (K = 5). Its Accuracy is lower compared to ANN, but it can give a result immediately and display the result on the webpage. After choosing the Machine Learning algorithm, input x_train and y_train as train set and x_test as test set into the Machine Learning Model. It will get the prediction set corresponding to x_test, and compare the prediction set with y_test, and then Accuracy and Square Error can be obtained. In order to predict price, my method is to add a line of Attributes to the last line of x_test, so that the last element of the prediction set is the fair market price. We will present this result to the user.

7.3. Advisor Bot

Advisor Bot is designed to help people who do not know any negotiation knowledge to get the best benefit in the transaction with the seller. We use Decision Tree to help users design negotiation ideas as shown in Figure 2 below. Firstly,

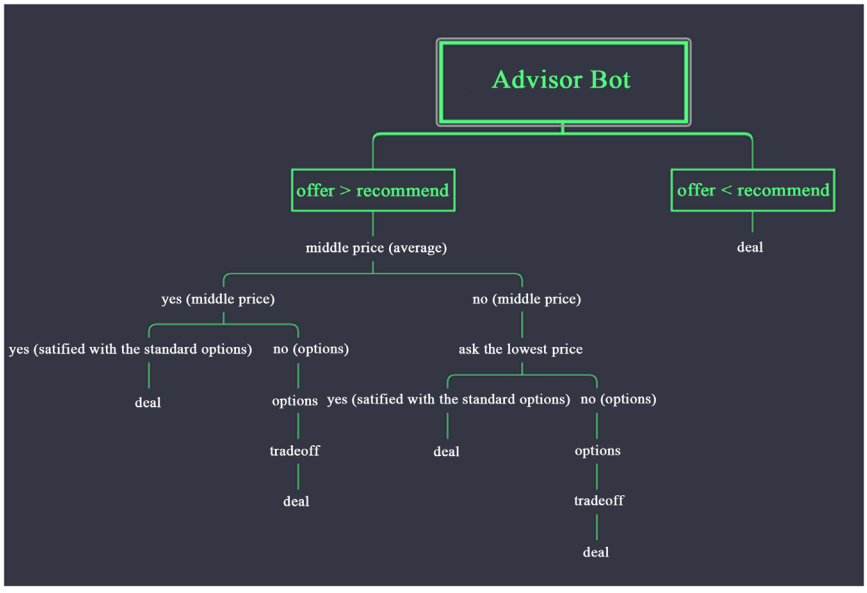

Figure 2. Decision tree model.

the offer in the figure is given by the seller and the recommend is the predicted price, or say fair market price. If offer < recommend, it means that the market average price is higher than the price given by the seller and the seller gives a good offer to user (the buyer). Advisor Bot will inform the user that he should accept the offer. If offer > recommend, it means that the seller gives a price higher than the market price. At this time, we will instruct users step by step to bargain. First, we will use an Aggressive price (in the figure, it is called middle price) to test the bottom line of the seller. The price algorithm is 2 × (recommend − offer). If the seller agrees to the price, user will be asked if he/she is satisfied with the transaction, and if they are satisfied, they will make a deal, but if they are not satisfied, they will make further bargaining. When the user is not satisfied with the transaction, Advisor Bot will give several options customers can negotiate according to their own needs, and Advisor Bot will tell customers two alternatives (Advisor Bot will combine the attributes input by user with the priorities of attributes and search similar commodities in other used car transaction platform). Alternatives serve as a reference helping user to determine whether they should agree to the deal. If the seller does not agree to aggressive price, the Advisor Bot will remind the user of asking the lowest price that the seller can offer. If the seller can give it, he will ask the user if he is satisfied, and the transaction will deal if he/she is satisfied. If he/she is not satisfied, he will be given options. The rest process is the same as the left sub tree in the decision tree as shown in **Figure 2**.

7.4. Judge Bot

Judge Bot's function is to grade user based on the result after negotiating. Judge Bot sets weights on those negotiable attributes and gives an overall score based on the combination of each score. This comprehensive score will be notified to the customer after each negotiation, print out each section's score, give the BATNA (Best Alternative) score and these things will be displayed in the scoring interface (**Figure 3**). If the user is not satisfied, he/she can choose "no" and go back to the negotiation page (**Figure 4**). The negotiation page will show all his/her alternatives. User can negotiate until he/she is satisfied, and the final decision depends on user. As shown in **Figure 5**, the best deal is 100 points meaning that all of the scoring detail is 100 (pay the car below the fair market price, get the best insurance and other attributes are the best).

8. Model Evaluation

8.1. Error Comparison

In order to evaluate our model, we use Quadratic Loss Function [14] to calculate the error of the algorithm in Price Bot. The data is shown below (**Table 1**). To better view the result, we also make a histogram (**Figure 6**).

We run the algorithm ten times and get the average, variance, range and median. We can make several conclusions from **Figure 6**: 1) The performance of the

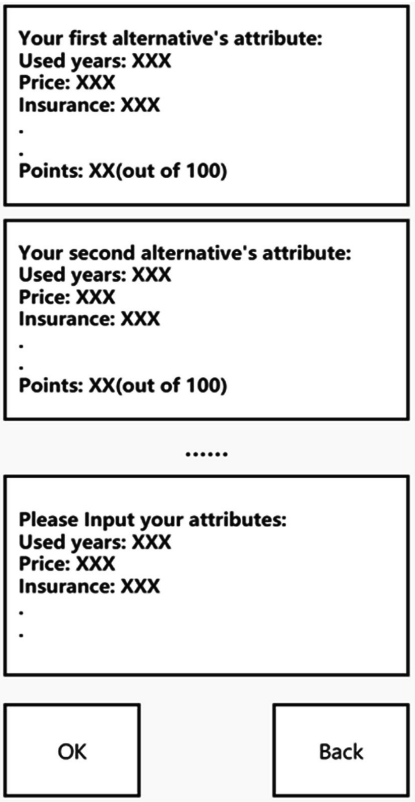

Figure 3. Negotiation page.

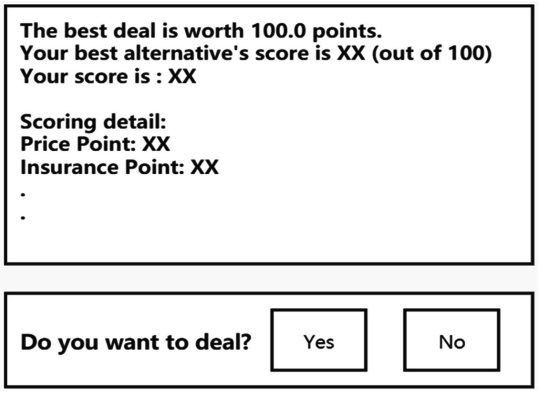

Figure 4. Scoring page.

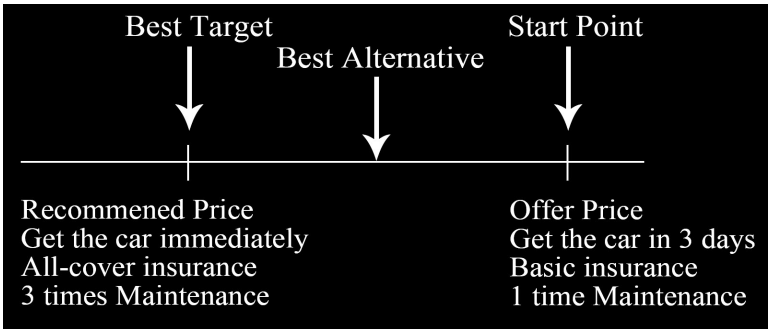

Figure 5. Scoring standard.

Table 1. Quadratic loss for each machine learning algorithm.

	Average	Variance	Range	Median
ANN	10.899	7.25	7.25	10.08
SVR	25.894	94.76	25.51	23.74
NN	13.689	10.98	9.22	12.92
5-NN	12.625	7.36	8.99	11.7

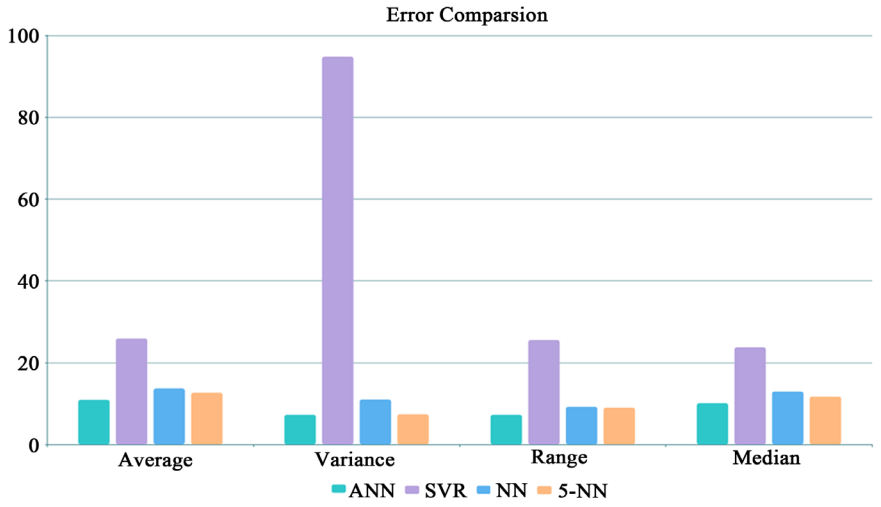

Figure 6. Error comparison.

Artificial Neural Network algorithm is the best compared with other algorithms and the 5-NN algorithm is the second; 2) The average, variance, range and median of SVR are very large, which means that the error rate fluctuates greatly and is very high when using this method, so its prediction of price is not accurate, probably may misleading the users.

8.2. Accuracy Comparison

For practical use, we defined a variable called Accuracy and a fluctuation region. For example, when the fluctuation 10% deviated from real price, if 0.9 × real price < predict < 1.1 × real price, then Accuracy count will plus 1, Accuracy = (Accuracy Count)/(Size of test set) × 100. The data (Table 2) are shown below, as well as the histogram (Figure 7).

From Figure 7, the conclusion is almost the same as which is shown in the error comparison graph where the stability and precision of ANN is the best and SVR is the worst, but, in this case, NN is a little better than 5-NN.

8.3. Time Cost

The Time Cost Table (Table 3) and the histogram (Figure 8) are shown below.

From Figure 8, conclusion can be made: 1) the average time cost of the NN and 5-NN algorithms are almost 0, they are far better compared with other algorithms; 2) the ANN algorithm is the most time-consuming algorithm.

In summary, we chose NN algorithm as Machine Learning algorithm in our Price Bot. The reason why we did not choose the ANN is that although the neural network has higher accuracy and lower error, we need to consider the real-time interaction of users. Thus, the computing time required by the ANN is unacceptable.

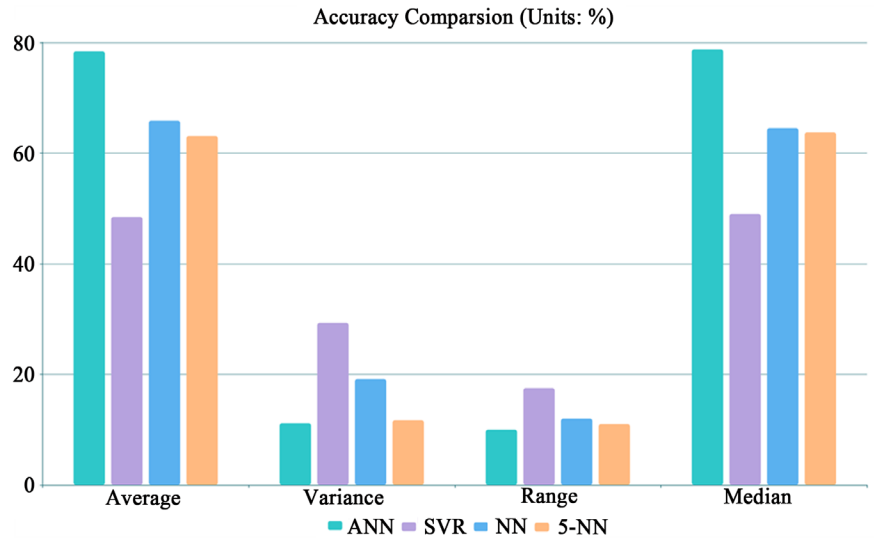

Figure 7. Accuracy comparison.

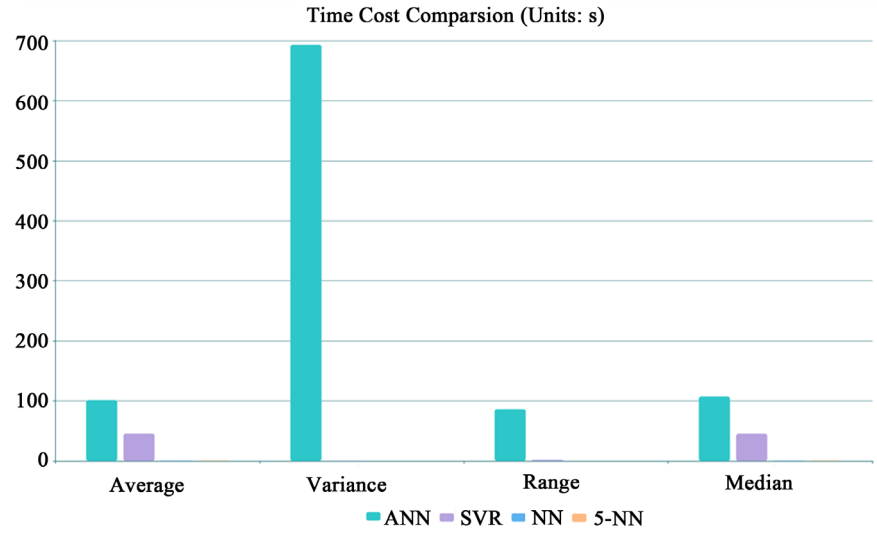

Figure 8. Time cost.

Table 2. Accuracy rate for each machine learning algorithm.

	Average	Variance	Range	Median
ANN	78.4	11.15556	10	78.75
SVR	48.45	29.30278	17.5	49
NN	65.855	19.14358	12	64.5
5-NN	63.1	11.71111	11	63.75

Table 3. Time cost for each machine learning algorithm.

	Average	Variance	Range	Median
ANN	101.311	693.1001	86.28	107.325
SVR	45.547	0.284712	1.88	45.43
NN	0.515	0.000339	0.07	0.515
5-NN	0.596	0.000538	0.08	0.595

8.4. Limitations of the App

The shortcomings of the project are: 1) when there are a large number of discrete data, the prediction results will be inaccurate. For example, the brand of the car, we process it into one hot data. As many brands as there are, there will be as many 0 - 1 characters. It can be imagined that when the number of brands is increasing, that is, the distribution is becoming sparser, and the predicted results will be greatly biased. 2) The car's attributes are pre-designed, so extra attributes cannot be accepted. For example, the business suddenly said that the wheelbase of this car is 2 m, which cannot be added to the model calculation, and will not affect the market price. 3) No matter how deep the decision tree is designed, it can't meet all the needs of users. Maybe an artificial intelligence similar to Siri can help us further improve the Advisor Bot.

9. Conclusion

We have implemented a primary intelligence robot that combines negotiation and deep learning algorithms and gave its application scenario. In this scenario, this robot can help buyers negotiate. The data in the negotiation robot comes from the data of a used car trading platform gripped by Crawler. These data are processed to predict the fair market price of used cars and are used as part of the negotiation basis to assist buyers in making decisions. Our negotiation bot gives a fair market price based on the used car data provided by the customer, and provides relative suggestions to the negotiation novice according to the market price, and then scores the performance, successfully simulating the application of a negotiation assistant robot in the negotiation process. This project is a preliminary attempt to combine artificial intelligence with negotiation technology, and there are many areas for revision and improvement. In the future, if Siri-like AI can be chosen as Advisor Bot, the robot will be more humane and likely to get more user feedback, thus improving the quality and satisfaction of negotiations. Negotiations today are almost based on experience. Although there are courses available for people to learn negotiation skills, people may not be able to immediately find the breakthrough point when negotiating. As a result, we try to quantify the negotiation elements, analyzing and refining them as the basis of creating negotiation assistant robots. The second-hand car trade is just a scenario that we realize. This negotiation assistant can also be used on other occasions where there are negotiations, such as bargaining when going to market or applying for jobs, just collecting enough data to train the machine learning model.

In the future, we can also improve in the Advisor Bot part for adding artificial intelligence similar to Siri to talk to human so that our app can meet more diverse needs of users.

Acknowledgements

Finally, we would like to express our most sincere thanks to Prof. Seth Freeman and our mentor Tanujay for their tremendous support in our project. They inspired us on how to combine negotiation with artificial intelligence, and we achieve the final program that we not expected before. We also want to express our appreciation to TA Ethan-Zhou F. for his guidance in programming technology and data processing.

It's an unforgettable experience for us to make this program this summer and we believe this research direction will make a huge success in a not so distant future.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Cover, T.M. and Hart, P.E. (1967) Nearest Neighbor Pattern Classification. *IEEE Transactions on Information Theory*, **13**, 21-27. https://doi.org/10.1109/TIT.1967.1053964

[2] Fu, K.S. (1970) Learning Control Systems—Review and Outlook. *IEEE Transactions on Automatic Control*, **15**, 210-221. https://doi.org/10.1109/TAC.1970.1099405

[3] LeCun, Y., Bottou, L., Bengio, Y. and Haffner, P. (1998) Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, **86**, 2278-2323. https://doi.org/10.1109/5.726791

[4] Guyon, I., Weston, J., Barnhill, S. and Vapnik, V. (2002) Gene Selection for Cancer Classification Using Support Vector Machines. *Machine Learning*, **46**, 389-422. https://doi.org/10.1023/A:1012487302797

[5] Silver, D., *et al.* (2016) Mastering the Game of Go with Deep Neural Networks and Tree Search. *Nature*, **529**, 484-489. https://doi.org/10.1038/nature16961

[6] Shan, C., Gong, S.G. and McOwan, P.W. (2009) Facial Expression Recognition Based on Local Binary Patterns: A Comprehensive Study. *Image and Vision Computing*, **27**, 803-816. https://doi.org/10.1016/j.imavis.2008.08.005

[7] Litjens, G., *et al.* (2017) A Survey on Deep Learning in Medical Image Analysis. *Medical Image Analysis*, **42**, 60-88. https://doi.org/10.1016/j.media.2017.07.005

[8] Martin-Raugh, M.P., *et al.* (2019) Negotiation as an Interpersonal Skill: Generalizability of Negotiation Outcomes and Tactics across Contexts at the Individual and Collective Levels. *Computers in Human Behavior*, **104**, Article ID: 105966. https://doi.org/10.1016/j.chb.2019.03.030

[9] He, X. and Chua, T.S. (2017) Neural Factorization Machines for Sparse Predictive Analytics. *Proceedings of the* 40*th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Tokyo, 7-11 August 2017, 335-364. https://doi.org/10.1145/3077136.3080777

[10] Pedregosa, F., *et al*. (2011) Scikit-Learn: Machine Learning, in Python. *Journal of Machine Learning*, *Research*, **12**, 2825-2830.

[11] Carbonneau, R., *et al*. (2008) Predicting Opponent's Moves in Electronic Negotiations Using Neural Networks. *Expert Systems with Applications*, **34**, 1266-1273. https://doi.org/10.1016/j.eswa.2006.12.027

[12] Mahto, D.K. and Singh, L. (2016) A Dive into Web Scraper World. 3*rd International Conference on Computing for Sustainable Global Development*, New Delhi, 16-18 March 2016, 689-693.

[13] Chang, C.C. and Lin, C.J. (2011) LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology*, **2**, 27:1-27:27. https://doi.org/10.1145/1961189.1961199

[14] Diebold, F.X. and Mariano, R.S. (1995) Comparing Predictive Accuracy. *Journal of Business and Economic Statistics*, **13**, 253-263. https://doi.org/10.1080/07350015.1995.10524599