The Chinese University of Hong Kong, Shenzhen

MAT 2040

Linear Algebra

# Price Prediction

*Author:*
Shi Wenlan

*Student Number:*
119010265

December 29, 2020

# Contents

# 1 Introduction:

This is a report aimed at realizing price prediction by fitting three models. In the process, Python code used numpy package to fit the three models by least square method, and calculated the most suitable values of undetermined coefficients and the error under these coefficients. Finally, the best model was selected, the coefficient was obtained by training, and the prices of five products were predicted.

# 2 Theory:

The three models used to fit the data are as follows:

First, autoregressive model.In this model, the product price at the current time slot is a linear function of the prices at the past time slots:

$$x(t) = \sum_{n=1}^{N} a_n x(t-n) \quad (1)$$

where $x(t)$ denotes the product price at the t-th time slot, both $a_n$ and $N$ denotes the coefficients that need to be calculated.

Second, fourier series. This model uses the following truncated version of the Fourier series to express the change of product price over time:

$$x(t) = \sum_{n=1}^{N} a_n \cos nt + b_n \sin nt \quad (2)$$

where $a_n$, $b_n$ and $N$ denotes the coefficients that need to be calculated.

Third, taylor formula. This model uses the truncated Taylor expansion to approximate the price function over time:

$$x(t) = \sum_{n=1}^{N} a_n t^{n-1} \quad (3)$$

where $a_n$ and $N$ denotes the coefficients that need to be calculated.

# 3 Method:

Firstly, the Python code *part*1 in the folder was used to fit the training data with different N, and the coefficient was calculated by the least square method. Then, this set of coefficients was used to build a model to predict the test data. The square root of the average of the square of the difference between each predicted result and the actual result was recorded as the *error*, and the coefficients that minimizes the error is taken. Finally, the minimum error of each model is compared and the model is sorted.

**Code thinking of part1** All data is read and stored in the list: train data is in *data_list* and test data is in *data_list_test*. Function $least_square$ solves the equation $Ax = b$ by least square method: $x = (A^T A)^{-1} A^T b$. Function *Autoregressive* generates array $A$ and array $b$ , so that each row of $Ax$ is calculating formula 1 for each set of data. Function $Fourier$ and $Taylor$ work similar to *Autoregressive*. In calculation part, the code tries all possible $N$ value, and output the best coefficients with the smallest error in each model. In the end, the code would output $N$, *error* and $x$ of the coefficients with the smallest error of each model. And when the model is Autoregressive model or Fourier series, elements in array $x$ are $a_1, a_2, ..., a_N$ in turn. When the model is Taylor formula, elements in array $x$ are $a_1, b_1, a_2, b_2, ..., a_N, b_N$ in turn.

Secondly, the Python code *part*2 in the folder was written based on the best model selected in the previous step. This code took into account both the prices in the past and the prices of other goods at the current time in the price prediction.

**Code thinking of part2** Data storage and function *least_square* is the same as *part*1. In function *Autoregressive*, the price of the other four product are added to the rightmost four columns of array $A$. So the output array $x$ also added the coefficients of other four product prices to the last four lines.

# 4 Results:

The part of output of the code *part*1 was as follows:

| Model | Best N | Best error |
|---|---|---|
| Autoregressive model | 52 | 558.65557739 |
| Fourier series | 96 | 1556.84257584 |
| Taylor formula | 3 | 1271.07290324 |

**Table 1:** Part1: model selector

The part of output of the code *part*2 was as follows:

| Product | Best N | Best error |
|---|---|---|
| 1 | 198 | 1754.07939199 |
| 2 | 198 | 950.74375019 |
| 3 | 175 | 1579.27346518 |
| 4 | 123 | 1100.35820301 |
| 5 | 93 | 947.58922339 |

**Table 2:** Part2: price prediction for five products

# 5 Discussion:

According to Table 1, the best model is Autoregressive model. Because of the function limitation of list reverse slicing in Python, a set of data was discarded in the fitting process, but I think the amount of the rest of the data is large enough. Besides, with the increase of n, the model goes from under-fitting to optimal fitting and finally over-fitting, and the error changes from large to small, and then from small to large. Therefor, I thought the result was generally reliable.

While writing the code for *part*2, I encountered two problems with understanding the problem: (1) Why there is a negative number in the data. The price of goods should not be negative. (2) How the price of a product related to the price of other products. It is related to the current price of other products or the previous price of other products?

In the end, the standards used when writing code were as follows: (1) Took all negative numbers as inverse numbers. (2) The factors that affect the price of a product are the previous prices of the product and the price of other products at last time slot.

According to Table 2, each N was quite large, which meant that the number of the coefficients was too large to show them all in the report. So it was recommended to run the code *part*2 and look at the output *best_x*. The array *best_x* consisted of $a_1$ to $a_N$ and the coefficients of the other 4 products from product 5 to product 1, excluding the product predicted.

# 6 Conclusion

Using Python code to build models and traverse all parameters, the best model selected after comparing errors is autoregressive model. After processing the data in the second part, the model is applied, and the price prediction model of each product is obtained. Because there are many coefficients, it is recommended to run Python code to browse the results of model coefficients.