Price Prediction

Daniel Dunning

***Index Terms*—price prediction, pattern recognition**

I. Introduction

The goal of this project is to predict the future prices of several items sold on Amazon.com. Future prediction has been a common field of study in pattern recognition and data mining. Hedge-fund companies in New York use techniques to try and predict the state of the stock market at future times, even just days ahead. Stock market predictions are often unreliable. However, price predictions for consumer items are different. Prices of products, especially electronics, often follow very clear trends throughout a year. These patterns are often clear to see visually, and they are not as variable as other trends such as the stock market. For the purposes of this project I investigated items sold on Amazon.com with the goal of predicting the prices of those items as far as a year in the future. The remainder of this paper will proceed as follows. Section 2 will cover previous work in this area. Section 3 discusses the methods used for this project. Section 4 will showcase results from this work as well as a discussion of these results. Section 5 will offer a conclusion and future work.

II. Background

There has been a large amount of research put into developing prediction models for future item costs. Pattern recognition as a whole can be said to work in the field of prediction because finding useful patterns in data can always be used to try and predict future patterns and trends in new data sets. One area of interest that is constantly being investigated is stock market prediction. It is often said that predicting the stock market as a whole is impossible because there are too many factors outside of the prices themselves that affect the cost. Even for companies that have a fairly constant trend, future prediction can be rendered entirely useless by a single event or factor. Even so, there is still plenty of research devoted to this area. Kimoto and Asakawa [2] investigated the ability to predict stock market prices with a neural network. The prediction model consisted of a simple modular neural network with three layers (input, hidden, output). Normal neural network procedures such as back propagation for weight correction were performed on the network. Based on their simulations, Kimoto and Asakawa et al. achieved good results with their model. Zhang and Wu [3] also studied prediction within the stock market, specifically targeting the S & P 500. They acknowledge that neural and recurrent networks were the best suited models for the task. However, they note that the normal back propagation used for weight correction within neural networks was not adequate for stock market prediction specifically because of the outside noise occurring in stock markets that back propagation does not take into account. For this reason, they focused less on the model as a whole and instead on making an improvement to the already common neural network process. They had considerable results from simulations as well. It must be noted, however, that simulation results for the stock market are not as promising as they may seem. Unlike other pattern recognition applications, the stock market can change rapidly and without cause, which is something simulations and these proposed models have a hard time representing. Mohsenian-Rad and Leon-Garcia [1] performed work in the area of price prediction outside of the stock market. They looked at price prediction within electric utility price changes. This posed a better baseline for prediction because unlike stock market prices the prices and environment of electric utilities are less dynamic than that of the stock market. Furthermore, this work aimed to provide a framework to have the most effective usage of appliances to create the best value in electric utility prices. This was all achieved using a simple (although more complicated than the model used here) linear programming model.

III. Methods

The process of the project was fairly standard for some sort of future prediction modeling. The first step was to select consumer items sold on Amazon.com that had an adequate price history. Here, we say adequate as having history for at least a year. The five items chosen were: an Amazon Echo (2nd generation), LG Electronics 55 inch TV, Toshiba microwave, Bowflex dumbbell set, and a pair of Bose headphones. These items were chosen because, apart from having the necessary price history, there were already visible trends in past prices. This is a common case for electronics, and that is the reason that the majority of the items fall into that category. After these items were chose, the next step in the process is to obtain more external information about these items in addition to prices. To do this, a custom python script was created to pull data from Amazon.com. The script itself (referred to in the remainder of this paper as a 'scraper') is a modification of an Amazon scraper that allows information such as reviews and product category be scraped from Amazon.com directly. Although these attributes are not directly related to price, they can often play a part in the trends of items. The resource that was used to gather price history from the items was camelcamelcamel.com (for the remainder of the paper camelcamelcamel.com will be referred to as 'Camel'). Here, prices of items can be tracked as far back as there is data for them, often as far back as the item has been for sale. However, the data is sparse in some places, and so external work had to be done in choosing items that had an adequate history to use for prediction. Originally, items were going to be chosen that had specific price fluctuations due

to Black Friday only. However, there was not a large enough historic data set for strictly only Black Friday-reduced items. After external data (mentioned previously) was scraped for the chosen items, analysis on the prices could begin. Due to the drastic price history differences between some of the items extra measures had to be taken to normalize the data. Some items contained history as far back as 5 years in Camel, while others had only a little over a year of history. A simple method to normalize the timestep would be to take a single 'chunk' of a time period, for instance two months, and gather a prediction for that. However, this was decided against because it seemed that for the items that had much more price history this would not take such history into account. Each item was being analyzed individually, so there was no difference in making a timestep a couple months for one item, and up to half a year for another item. For each item an average price over all of the timesteps was gathered. Unfortunately, to the author's knowledge there is no simple way to gather data specified data from Camel directly, so this step had to be done by hand. Each item had a list of its average price for each of the time chunks. The next piece for the project was to find a model for this prediction that was simple enough for the time frame, but that could also give reasonably accurate results. A baseline model was found on Quora.com and this was modified for the project. The predictor is a simple linear predictor, the future price is estimated with a random number as a weight which is created based on the previous prices. To modify the baseline model, the program included the addition of a function to add the predicted number to our list of prices, so that the new price acted as part of historical data. In this way the next price prediction accounts for the already predicted price. This should give the overall prediction a higher accuracy because it is a continual estimation as oppose to a static estimation which would only give an estimation over the same set of data over and over again. The brief algorithm is shown in Algorithm 1. It should be noted that this particular prediction algorithm does not take any other factors into its prediction except for previous prices. In a future section it will be discussed whether additional weights into the predictor would be beneficial and give more accurate results. The last step of the prediction process (predict and add value to the list) is repeated until the amount of data is doubled.

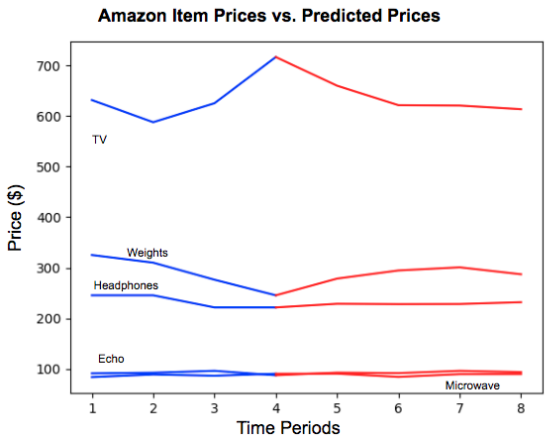
Algorithm 1 Algorithm for prediction

List A
for all Iterations i,from 1 to N **do**
  for all Items in A **do**
    $predict \leftarrow PREDICT(A)$
    $appendpredicttoA$
  end for
end for
return A

IV. Results

The results I encountered did not match my expectations for the predictions. Figure 1 shows the prediction prices for each of the five items. The blue portion of the the graph shows the original price history. The red shows the predicted trend of prices for the next timesteps. The two most interesting predictions were that of the TV and also the dumbbell set. The TV had its average price increase as time passed, however the predictor showed the average price to start to fall immediately at its last time step. This is interesting because in general one would expect the overall price trend to follow the same pattern it started with, in this case steadily increasing. What makes this more interesting is that even though the single price of the TV at the last point in history (the present) was an all time low, its average price for the time period had continued to increase. In contrast, the predictor seem to use an almost "intuitive" prediction and have the average price continue to drop, as I would infer naturally. The dumbbell set actually performed in an opposite manner. It saw the price history have a steadily decreasing trend, and responded with predictions that increased steadily.

Fig. 1. The graph of the items and their respective price predictions. The original price history is shown in blue while the predictions are shown in red



The contrast of these two predictions provoked thoughts about the reasoning as to why the same predictor might go against the expected trend in two opposite cases. For myself, the biggest reasoning to this is the outside sources that affect the price of an item. As is the case for most items (with the exception of antique or rare items), as time goes on price should go down as the value becomes less and less. However, there are factors that change this general trend. First, other items in similar categories affect this price. If there are many competing items in a category, prices tend to drop more or faster because if the items are fairly equal in quality there needs to be something more captivating to the consumer. However, with fewer items in a category, especially in the case of a single item, prices can stay constant or possibly even increase because there is no draw from the seller to decrease the price. In addition, another factor such as reviews can drop the price of the item faster than expected. If the item receives a majority of negative reviews, the seller would have no choice but to drop to price or the item will not sell. On the other hand, if there are lots a reviews and they rate the item highly, the product is a popular commodity and so it does not need to have as drastic price fluctuations. Finally, specific time periods

often affect price more than a simple ”steady decrease with time” would show. For example, times around the holidays often result in a sale for items, especially electronics. This is particularly true of Black Friday where prices may be reduced drastically. When the prices return to their normal state after sales, it creates a higher fluctuation in price history which makes it harder to predict, especially if this is not taken into account by the prediction model. None of these external factors are considered by this simple linear predictor, and so it has to rely solely on the price history. For this reason another more robust predictor may have been a better choice for the project. Possibly a similar neural network to the one that was used in [2].

V. Conclusion

The project presented here had the goal of predicting the price of certain items on Amazon.com. This is not the first study of its kind, so the motivation was not to create the best prediction model for this. Rather, the goal was to use a fairly simple predictor, see its effects, and then make inferences as to what caused possible inaccuracies. This can be useful for future work if one were to use a more robust prediction model, such as a neural network, or if one were to create their own prediction model. For future research, I would like to do what was previously mentioned, and try to create a prediction model, possibly similar to a neural network, but with weights that accurately represent some of the external factors that were discussed. I believe this could give more accurate results, especially if it were coupled with more input instances and more iterations of the prediction model.

Appendix

A. Source Code and Supplementary Materials

https://github.com/dannydthesloth/patternrecognition
amazon.com
camelcamelcamel.com
quora.com

References

[1] Mohsenian-Rad, Amir-Hamed, and Alberto Leon-Garcia. ”Optimal residential load control with price prediction in real-time electricity pricing environments.” *IEEE Trans. Smart Grid* 1.2 (2010): 120-133.

[2] Kimoto, Takashi, et al. ”Stock market prediction system with modular neural networks.” Neural Networks, 1990., 1990 *IJCNN International Joint Conference on. IEEE*, 1990.

[3] Zhang, Yudong, and Lenan Wu. ”Stock market prediction of S&P 500 via combination of improved BCO approach and BP neural network.” *Expert systems with applications* 36.5 (2009): 8849-885