BarGain: Crowd Sourced Machine Learning Integrated E-Commerce

Dr A Shanthini, Rohan Anand, Ritesh Bhat, Shivam Saxena

Abstract— Bargain is a venture which aims to explore the domain area of E-Commerce in ways to make the buying experience for customers more organic and interactive by introducing bargaining features, which are otherwise only prevalent in physical brick and mortar stores. Bargain additionally also utilizes machine learning in order to predict future prices for customers to look at and make a more knowledgeable and proficient purchase decision.

Index Terms— E-Commerce, machine learning, data science, Web application

1 Introduction

E-Commerce has been a massive global movement. Over the last decade many major E-Commerce players have emerged such as Amazon, Flipkart which sell almost all types of products as well as many smaller and more specialized players such as Big Basket and Dollar Shave Club which specialize in their own particular areas. All of these businesses provide an online catalog from which customers select and buy. However, in a more traditional setting there is one key component which most E-Commerce platforms do not provide: Customer-Retailer interaction. And one of the reasons that some people still prefer to buy "offline" instead of online is because they can initiate in bargaining. This same interaction is our goal to imitate online in a reasonable and scalable way. We aim to make it reasonable for retailers as well which is why all offers made by retailers need to have a specific goal to be reached to sell.

Another great aspect of ecommerce is that you can see this price of almost any item at that point and time. However for almost all products prices fluctuate. Bargain aims to track these price fluctuations using current E-Commerce platforms API's (Application Programming Interface) such as Amazon's Product API accessible under the Amazon Associates Program. Not only track, but bargain also uses this data to train an ML (Machine Learning) model, to predict future prices of products.

2 Existing System Analysis

In the current existing eCommerce system, offers are initiated from the vendors side. Vendors offer prices on products at rates at either MRP or lower in times of sale. There is no provision for customers to buy in mass, i.e customers are serviced individually rather than in bulk. So vendors sell products usually at slower single unit rates instead of bulk unit rates. Also in current eCommerce sites, there is no dynamic of the customer being able to bargain for a lower rate. This reduces the power of the customer and flexibility of the customer.

3 Problem Definition

With the increase in number of transactions via eCommerce, it is clear that Indians enjoy the freedom and ease of eCommerce. However, some customers still prefer going down to a store to buy their products after bargaining with vendors to get a better price. In current eCommerce sites, this dynamic between the customer and the vendor does not exist and the vendor fixes his prices according to MRP's and providing discounts in sales.

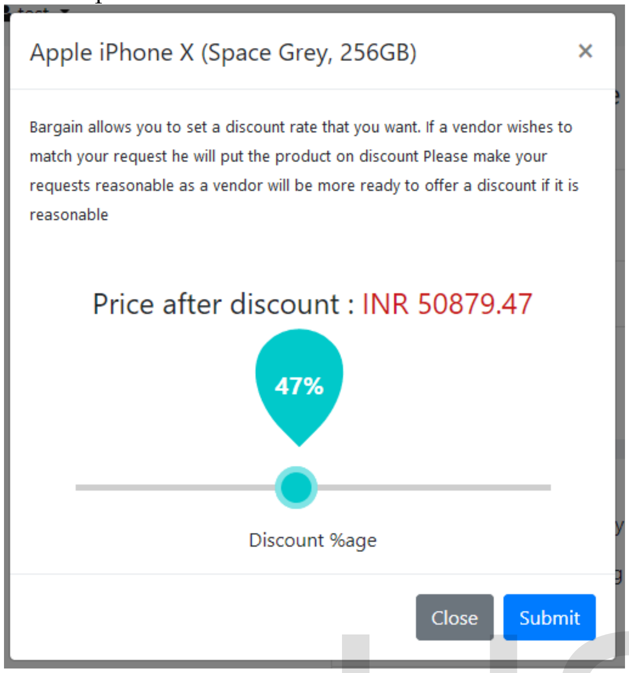
Our model hopes to add this dynamic into eCommerce as well. Also, vendors frequently have old stock that they wish to move out of their storage units to make way for newer and more profitable units. However, the old units are still perfectly sellable and can be sold at mass for a heavy discount by the vendor to quickly move out inventory. This can be achieved via our application.

4 Sample Use Case

A user/customer is looking for a decent arrangement on an item that is fairly costly, yet also highly sought after. Let us assume that in this case the product is a Laptop. The customer will login to Bargain and Search for the product he is looking for. On opening the product page the customer is greeted with 4 modules of information:

1. Price Chart: A price vs time line chart that shows the price of the product over time as well as probable future predictions.
2. Bargain Chart: A bar chart showing the number of bargain requests made at a particular discount range.
3. Bargain Offers: Offers made by retailers based on bargain request. It will display the price after discount and the goal reached (minimum number of units to be sold by the retailer for the deal to go through).
4. General Information: General information of the product.

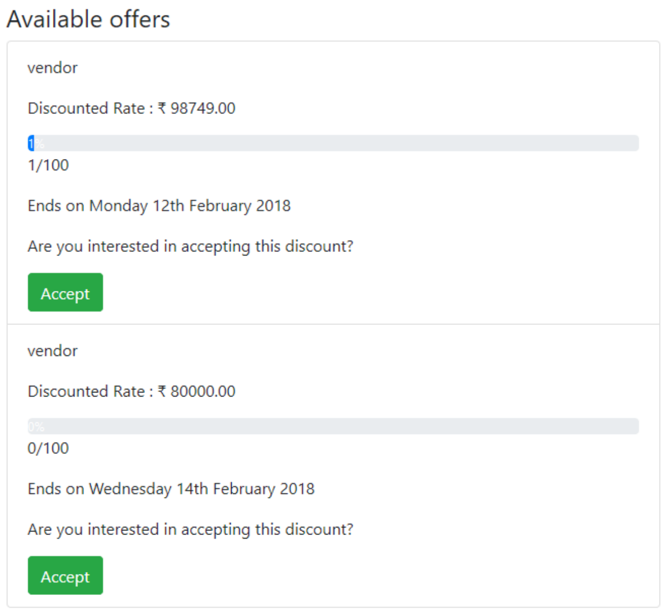
The customer will then check out the price chart and see that the price had once dipped at some point lower than the actual asking price. So the customer now knows that a deal of that price is possible. If he isn't satisfied by the current offers, he can raise another bargain request which will be visible to other retailers. A retailer might see that this product is gaining a lot of attention and that some of the discount rates that customers are asking for is reasonable. The retailer will then create a new offer with a particular discount range and a goal he wishes to reach to be able to sell off his stock at a discount but still profitable.


Apple iPhone X (Space Grey, 256GB)

Bargain allows you to set a discount rate that you want. If a vendor wishes to match your request he will put the product on discount Please make your requests reasonable as a vendor will be more ready to offer a discount if it is reasonable

Price after discount : INR 50879.47

The customer will then receive a notification of the new offer available. He will then evaluate whether or not the offer is satisfactory. If it is then he will accept the offer. On accepting the offer the user will be prompted to submit a small percentage of the price as security to ensure that the user will complete his purchase when the goal is reached


Available offers

vendor

Discounted Rate : ₹ 98749.00

1/100

Ends on Monday 12th February 2018

Are you interested in accepting this discount?

vendor

Discounted Rate : ₹ 80000.00

0/100

Ends on Wednesday 14th February 2018

Are you interested in accepting this discount?

If the goal of the retailer is reached within the end date, the sale will occur and the customer will pay the balance amount. If the sale is not reached, the user gets back his security deposit. This is a general scenario which will be followed on the application.

5 Project Specifications

Bargain is built on a web platform which is designed for the purpose of continuous monitoring of price fluctuations of products marketed by online vendors such as Flipkart, Amazon, Snapdeal etc by providing data visualization.

Bargain also tries to emulate customer-retailer interaction at a basic level by introducing a bargaining feature, which allows customers and vendors to agree to a new price at some rudimentary level.

Bargain uses the Model View Controller (MVC) design pattern in order to make the project more modular, extendable and scalable for future upgrades. The following technologies are used to enable the same.

5.1 Frontend (Client Side and UI)

- Bargains basic frontend is built using HTML (Hypertext Markup Language), SCSS (Sassy Cascading Style Sheet) for preprocessing style sheets, Bootstrap 4, Javascript and Jquery.
- Bargain also uses twig, a powerful templating agent built for the Symfony php framework which is fast, secure, and flexible.
- For data visualization we have used google charts.

5.2 Backend (Server-Side)

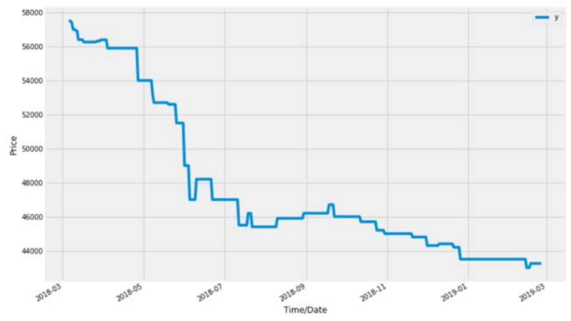
- Bargain is primarily built on top of Symfony 3.4, a PHP (Hypertext Preprocessor) MVC Framework for server side functionality. This handles almost all functional aspects of the Web Application, except the Machine Learning domain.
- For database MySQL was used along with the Doctrine ORM (Object Relational Mapping) to enable further mapping with other databases if needed.
- Composer is used as our package manager
- For automated testing we used PHPUnit, a unit testing framework to enable easy automation of our PHP components.
- All of this is run atop a digitalocean droplet running Apache2 with a bunch of server side CRON jobs running.

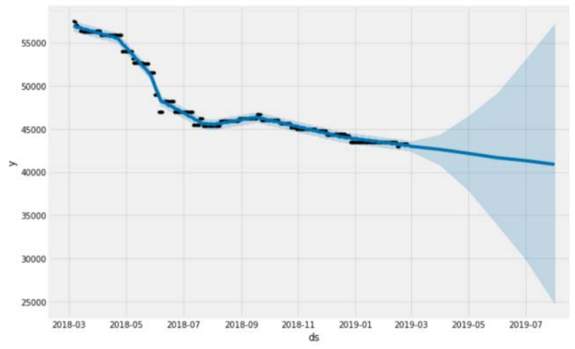
5.3 External API's

- Unfortunately Bargain does not have its own independent product catalog. This implies that we don't have our own data to analyze and use for our Machine Learning Models.
- To compensate for this deficiency, we have utilized Amazon's Product API under the Amazon's Associate program. We utilize this API to search for products and get their daily prices.

5.4 Machine Learning

- Unfortunately Bargain does not have its own independent product catalog. This means that we don't have our own data to analyze and use for our Machine Learning Models.
- To make up for this deficit, we have used Amazon's Product API under the Amazon's Associate program. We use this API to search for products and get their daily prices. These daily prices data we use as the training set to train our machine learning models which after that can be used to get output for our future test data. This dataset has two columns named ds and yhat, here ds column has all the dates/timestamp and yhat has the prices. The reason behind naming these column is that the fbprophet packages takes in column name ds and yhat by default. Where yhat is the values for which the prediction is to be made.
- As already mentioned we are using fbprophet package for price forecasting. At its core, the Prophet procedure is an additive regression model. Also Prophet automatically detects changes in trends by selecting changepoints from the data. And it is much easier to implement and deploy publicly with respect to the other available options.Prophet makes it much more straightforward to create a reasonable, accurate forecast.
- We are getting real time results for the product by creating endpoints of ML models of the product. These endpoints accepts the incoming request and responds back with the json response of the future prices. This system is made with the help of Flask framework. Since Flask is a python based microframework and our ML code is also written in python it makes this very easy to integrate and deploy.
- This is the graphs which we are getting for one of the product .These readings are the price history of the product.
- Following is the graph with future prediction for 3 months(due to lack of real time data there is a bit of uncertainty in the price forecasting):





6 API Schema and Data Consumption

We utilized one external API and two of our own internally facilitated API's.

6.1 Amazon Product API

The Amazon Product API is our external API. This API is consumed by our application to yield search results and get product data.

The response of this API is in the XML format. One important attribute of amazon products are that they each have a unique identification number. We use this ASIN (Amazon Standard Identification Number) to uniquely identify our products.

This API only provides the product price at the current time and not of the past or real time. Hence our data from this API is not real time and instead we choose to run a CRON (Chronos, a software utility that schedules jobs based on time) everyday that will store the current product price and map it to the current date, hence creating our Price chart mentioned earlier.

6.2 Internal Web RESTful API

Our main internal API is built as a RESTful (Representational State Transfer) web service. This means that it is usable on all systems as long as the requester provides the required Auth Headers in their request. However currently this API is not exposed publicly.

This API responds with a JSON (Javascript object Notation) encoded string which can be easily parsed back by any machine that recognizes JSON.

This API provides CRUD (Create Read Update Delete) functionality for almost all our of operations except for processing our ML Models. That is the job of our third API.

6.3 Machine Learning API

Due to high cost of the cloud ML endpoints we have created our own ML API's to anticipate the future values of the product. The backend for these API is created in Flask(a micro-framework of python). The request goes to the <SERVER>/prediction/<PRODUCT_ID> route, this will provide a JSON response for the future price of the product. The future price of the products is created using different python libraries namely pandas, prophet and pickle.

Pandas and prophet is used to created the model whereas we are using the pickle package to serialize and save our ML model so that we don't have to evaluate the whole model every time(this saves a lot of unnecessary computing over our deployed machine).Pickle also helps to deserialize i.e load our model file back which can be used to predict the future values over the new test data. GPU cloud instances are very expensive(almost $800 for a single month, source: Google Cloud Compute with GPU), therefore we need to take care of the cost also. These API's will be used by our main website to get the price prediction. The reason we are using flask is that it is a python based microframework due to which our jupyter notebook code(which is again in python) works seamlessly with it.

7 FUTURE EXTENSIONS

So far, Bargain does not have any payment system in place, which means that right now it is just a demo application. This is actually great news because that means that we get the chance to experiment with a technology that is making headlines worldwide; Cryptocurrency and the Blockchain. While we are not fully convinced with cryptocurrencies, blockchain, which is the underlying technology which enables cryptocurrencies to function seems promising. We would wish to implement a blockchain based transaction system for our platform to allow transactions to be open.

8 CONCLUSION

Technology is taking new shapes and forms to impact our daily lives and change the way the world works. E-Commerce is one of those areas that have forever changed the shopping experience. But there are still some human elements that we miss when we integrate technology. These interactions can be reengineered using technology and emulated in the virtual world. So if we are ever looking for new ideas, the best source of inspirations is observing and analyzing human interaction.

ACKNOWLEDGMENT

The authors would wish to thank Dr G. Vadivu, Head of Department, Information Technology, SRM University for continuous support in our venture.

REFERENCES

[1] Amazon product API Documentation, AWS ECommerce Service Developer Guide (API Version 2013-08-01). http://docs.aws.amazon.com/AWSECommerceService/latest/DG/Welcome.html
[2] Product data integration in B2B e-commerce, D. Fensel ; Ying Ding ; B. Omelayenko ; E. Schulten ; G. Botquin ; M. Brown ; A. Flett
[3] Initializing an ecommerce database framework, Roy Aaron Underwood
[4] Using RESTful web-services and cloud computing to create next generation mobile applications, Jason H. Christensen
[5] Building PHP Applications with Symfony, CakePHP, and Zend Framework, Bartosz Porebski, Karol Przystalski, Leszek Nowak
[6] Gaussian processes for machine learning, CE Rasmussen, CKI Williams
[7] Prophet by Facebook for forecasting. https://facebook.github.io/prophet/
[8] Research Paper on E-Commerce Challenges and Opportunities https://www.slideshare.net/AmithVicky/research-paper-on-ecommerce-challenges-and-opportunities
[9] Asia Pacific Online Retail Forecast, 2011 To 2016 https://www.forrester.com/report/Asia+Pacific+Online+Retail+Forecast+2011+To+2016/-/E-RES72723h