Sneakerhead Gear – Final Report

STAT 385 FA2018 - Team XSWL

Ziwei Liu (ziweil2)
Zepeng Xiao (zepengx2)
Yuquan Zheng (yzheng58)

December 18, 2018

Abstract

This project aims to design a shiny-powered app to address an underlying difficulty of many sneaker shoes lovers & collectors, which is to buy or trade sneakers at reasonable prices. The goal is to use information scrapped from StockX, one of the most popular stock market for sneakers, and produce visualizations as well as give predictions for the prices of popular items by incorporating time series analysis. The motivation behind this project is a shared experience among the group members about previous hardships involved with trading sneaker shoes, and a strong desire to work with web scrapping and to practice statistical methods with R. The expected gain is hence as stated above.

Contents

1 Introduction

1.1 Problem statement(issue that has arisen):

In a world that the prices of sneakers change rapidly, it's always hard for sneaker buyers or sellers to make the deal at the best price. In this case, we intended to design a shiny app that can help users master the price variation of certain brands of shoes and also give predictions about the future prices according to the time trend. Besides, the price information that users collect on the internet by themselves may be hard to interpret so that our intention would also be presenting the information in a direct way.

1.2 Motivation:

The initial purpose of our project is quite realistic—to help sneaker lovers make better shopping decisions or selling decisions. We intended to achieve this purpose by providing sneaker lovers detailed information about the price variation and giving predictions about the future prices for certain shoes. What's more, collecting information online by customer themselves may be time-consuming and searching results may be complicated and hard to interpret. In this case, we believe that by using R's ggplot and shiny packages, we can present the price information in a simpler and more user-friendly way.

1.3 Description of data:

The data we have used in this project are all scrapped from the website StockX using web scrapping techniques, which is an online market with detailed information about the prices of sneakers in different time periods. To be more specific, in the prediction function, we downloaded seven groups of data. Each group of data indicates the price information for a certain shoe and includes 10+ variables and more than 500 observations. The observations are differentiated by different time periods they are in. The construction of data is an important part of the project. Below is a short demo of our data.

Date	Time	Size	Price	Local_Amount	Currency	Name	Brand	Year	Month	Day	Number_Price
2018-12-15	01:18:23+00:00	9.5	276	276	USD	Air Jordan 4 Royalty	Nike	2018	12	15	276
2018-12-11	21:35:22+00:00	9.5	280	280	USD	Air Jordan 4 Royalty	Nike	2018	12	11	280
2018-12-10	00:46:28+00:00	9.5	275	275	USD	Air Jordan 4 Royalty	Nike	2018	12	10	275
2018-12-05	00:29:49+00:00	9.5	291	291	USD	Air Jordan 4 Royalty	Nike	2018	12	5	291
2018-12-02	15:40:30+00:00	9.5	285	285	USD	Air Jordan 4 Royalty	Nike	2018	12	2	285
2018-12-01	02:39:45+00:00	9.5	283	283	USD	Air Jordan 4 Royalty	Nike	2018	12	1	283
2018-11-26	01:49:32+00:00	9.5	350	350	USD	Air Jordan 4 Royalty	Nike	2018	11	26	350
2018-11-26	01:48:35+00:00	9.5	341	341	USD	Air Jordan 4 Royalty	Nike	2018	11	26	341
2018-11-26	01:48:11+00:00	9.5	325	325	USD	Air Jordan 4 Royalty	Nike	2018	11	26	325
2018-11-26	01:47:47+00:00	9.5	298	298	USD	Air Jordan 4 Royalty	Nike	2018	11	26	298
2018-11-25	15:11:03+00:00	9.5	243	243	USD	Air Jordan 4 Royalty	Nike	2018	11	25	243
2018-11-25	14:49:16+00:00	9.5	266	266	USD	Air Jordan 4 Royalty	Nike	2018	11	25	266
2018-11-25	14:46:21+00:00	9.5	269	269	USD	Air Jordan 4 Royalty	Nike	2018	11	25	269
2018-11-25	14:22:23+00:00	9.5	271	271	USD	Air Jordan 4 Royalty	Nike	2018	11	25	271
2018-11-25	14:13:42+00:00	9.5	272	272	USD	Air Jordan 4 Royalty	Nike	2018	11	25	272
2018-11-21	19:08:36+00:00	9.5	325	325	USD	Air Jordan 4 Royalty	Nike	2018	11	21	325

General Source: StockX.com

1.4 Course connection:

In our project, we utilized a wide range of concepts and methods that we learned in the course. Firstly, we scrapped the price data from the StockX website. Secondly, we performed cleaning and merging for the data. Furthermore, we depended on R's forecasting and data manipulation abilities to fit model for the prediction. Finally, we built the `Shiny` application and output visualizations.

2 Related Work

2.1 Familiarity with prior work:

As for the detailed contents, we decided to do visualizations for historical prices, and then we discovered a related study Scraping StockX: Adidas Yeezy Resell Analysis. which offered a lot of ideas pertaining to scrapping and data analysis for a certain type of sneaker shoes.

2.2 Originality:

To ensure originality, we incorporated time-series analysis in order to give prediction about shoes prices. There have been similar analysis project focused on forecasting stock prices, and we think our originality lies in bringing this idea to the market of sneaker shoes.

3 Method

3.1 Workflow:

The main body of this project can be broken into three parts: construction of data by scrapping data from StockX, design and implementation of user interface and app server with shiny, and barplot/line graph visualizations of price information and prediction results with ggplot2. Our intended action is to perform time-series analysis (Hyndman and Athanasopoulos (n.d.)) through fitting ARIMA models (with built-in function arima) that is available in base R.

3.2 Packages:

Packages that have been used include:

1. `ggplot2` Wickham (2016a) for visualizations
2. `rvest` Wickham (2016b) for web scrapping
3. `tidyr` Wickham and Henry (2018) for data cleaning
4. `stringr` Wickham (2018) for string manipulation
5. `shiny` Chang et al. (2018) for creating shiny app
6. `RSelenium` Harrison (2018) for doing search live
7. `XML` Lang and CRAN Team (2018) for doing assistant work of search live
8. `sqldf` Grothendieck (2017) for selecting desired data
9. `forecast` Hyndman et al. (2018) for predicting prices using arima model

3.3 Code:

The group wrote codes to first complete web scrapping, then potentially split or combined some of the variables for data cleaning. In order to fulfill the requirement for the number of attributes, there was also the need to merge data using SQL after we imported them. Next, to provide prediction of prices, we explored `auto.arima`, and the plotting of line graphs demonstrating price trends required using `ggplot2`. Finally, codes are written to build a shiny application.

3.4 Logic (very important regarding usage):

To properly drive this application, the user should first make sure connection to the internet, **then all R Scripts must be run before opening the `app.R`**. This is to support all function calls that will happen in the server of the shiny app while operating. A further improvement arises here as we want to simplify this process and write all functions into a package. In doing this, the user can directly run the `app.R` and have all the functionalities work appropriately.

3.5 Important note:

The `Search Live` functionality in our application seems not to work well on RStudio Cloud, in that the `RSelenium` would fail to evoke another Chrome session, but it would work perfectly well on our local RStudio, which we have utilized to implement this functionality. To use it locally, follow the below steps (On Windows 10):

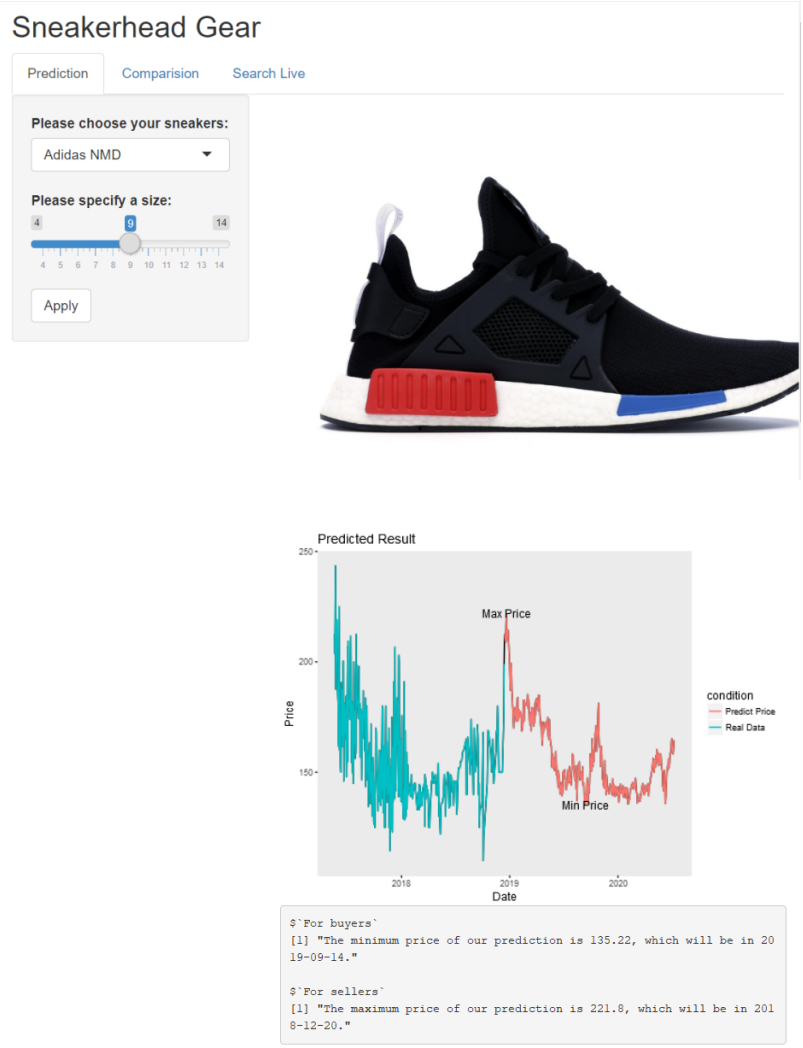
1. Copy the url of the project Repository.
2. Open terminal, direct to the preferable path for placing the project.
3. `git clone "the url here"`
4. Type user ID and password
5. Open up project.

4 Discussion About Results

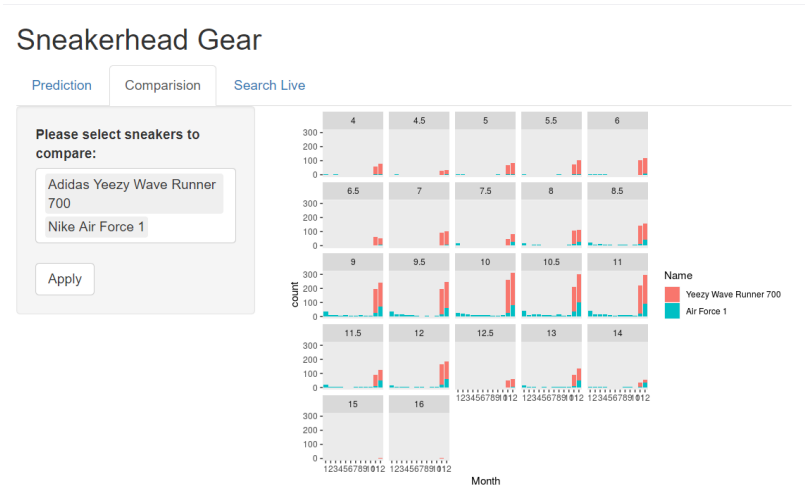
We have successfully designed the shiny app that have three different functions: predict,comparison and search live. In the prediction function panel, we are able to predict the future prices of a certain sneaker among the default options and also predict the time when the max or minimum price will be reached. To make it clearer for our users, we have also provided the picture for the shoe, the prediction graph with maximum and minimum prices marked and the text description about the prediction. In the comparison panel, we are able to compare the sales of two different kinds of shoes using graphs to better understand the popularity of certain sneakers. Finally in the search live function we are able to get the latest information about prices and available sizes for the shoes we type in the searching bar.

- Shiny UI:

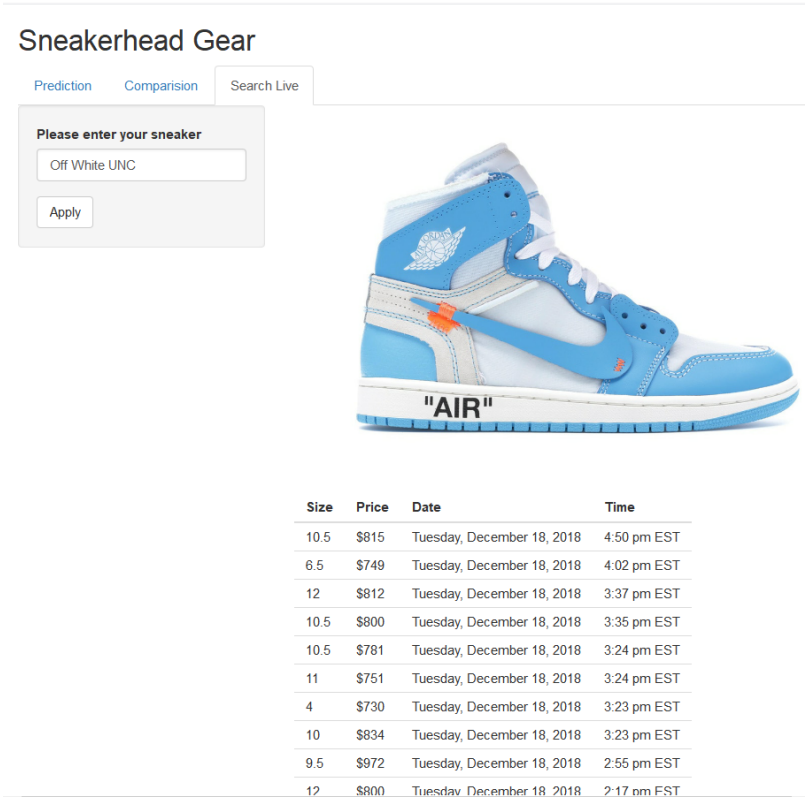
Prediction:



Comparison:



Search Live:



Size	Price	Date	Time
10.5	$815	Tuesday, December 18, 2018	4:50 pm EST
6.5	$749	Tuesday, December 18, 2018	4:02 pm EST
12	$812	Tuesday, December 18, 2018	3:37 pm EST
10.5	$800	Tuesday, December 18, 2018	3:35 pm EST
10.5	$781	Tuesday, December 18, 2018	3:24 pm EST
11	$751	Tuesday, December 18, 2018	3:24 pm EST
4	$730	Tuesday, December 18, 2018	3:23 pm EST
10	$834	Tuesday, December 18, 2018	3:23 pm EST
9.5	$972	Tuesday, December 18, 2018	2:55 pm EST
12	$800	Tuesday, December 18, 2018	2:17 pm EST

4.1 Project timeline:

Just want to check if my editing works

4.2 Breakdown of work:

We originally wanted to enable users to select any type of sneakers from any brands and give price predictions and visualizations, but then we realized to enable searching and then scrapping the corresponding data from different webpages requires a higher level of web scrapping technique, which takes time to master. We then decided that for this project, we would allow 5 to 10 pairs of shoes to choose from, and the current web scrapping techniques would still ensure that the information is real-time. Therefore we finished before the end of semester, since we have been able to send one week constructing data and building & testing UI and server, and another one to two week implementing time-series analysis and to compile reports and video demo. Our group has also maintained a schedule to meet at least twice a week and taken advantages of the office hours.

4.3 Contributors:

As for tasks split, Ziwei Liu has undertaken the part of data construction through web scrapping techniques and data cleaning, and he is the main contributor to the `Search Live` functionality.

Zepeng Xiao has fitted ARIMA models on getting time-series analysis work and output visualizations. He is the main contributor to the `Prediction` functionality.

Yuquan Zheng has taken charge of visualizations in `Comparison` as well as design of the entire shiny app UI.

Finally, all team members have been working closely with each other on getting each part work properly.

5 Conclusion

In summary, this project has successfully implemented a gear for sneaker lovers to buy or sell their sneakers at desired prices, eased by visualization and powered by data scrapped from StockX. It resolves to a certain degree the uncertainty in market prices faced by sneaker traders when using StockX. The novelty arose from the attempt to apply time-series analysis to predict shoes prices.

References

Chang, Winston, Joe Cheng, JJ Allaire, Yihui Xie, and Jonathan McPherson. 2018. *Shiny: Web Application Framework for R*. https://CRAN.R-project.org/package=shiny.

Grothendieck, G. 2017. *Sqldf: Manipulate R Data Frames Using Sql*. https://CRAN.R-project.org/package=sqldf.

Harrison, John. 2018. *RSelenium: R Bindings for 'Selenium Webdriver'*. https://CRAN.R-project.org/package=RSelenium.

Hyndman, Rob, George Athanasopoulos, Christoph Bergmeir, Gabriel Caceres, Leanne Chhay, Mitchell O'Hara-Wild, Fotios Petropoulos, Slava Razbash, Earo Wang, and Farah Yasmeen. 2018. *forecast: Forecasting Functions for Time Series and Linear Models*. http://pkg.robjhyndman.com/forecast.

Lang, Duncan Temple, and the CRAN Team. 2018. *XML: Tools for Parsing and Generating Xml Within R and S-Plus*. https://CRAN.R-project.org/package=XML.

Wickham, Hadley. 2016a. *Ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. http://ggplot2.org.

———. 2016b. *Rvest: Easily Harvest (Scrape) Web Pages*. https://CRAN.R-project.org/package=rvest.

———. 2018. *Stringr: Simple, Consistent Wrappers for Common String Operations*.

Wickham, Hadley, and Lionel Henry. 2018. *Tidyr: Easily Tidy Data with 'Spread()' and 'Gather()' Functions*.