Optimizing Sales of Ducks and Fish

PATRICK LOWE, University of Passau, Germany

The reproducibility of sales optimization, and how to future proof the tools used. The Excel Solver used in the Head First Data Analysis's Optimization chapter may not always be accessible, the challenge is to create an open-source alternative. While Google Sheets offers a similar solution, this project investigates the more widely accessible Python programming language and the package pulp. This tool was able to replicate results as Excels Solver while being more versatile.

Additional Key Words and Phrases: datasets, optimization, python, excel solver

ACM Reference Format:
Patrick Lowe. 2023. Optimizing Sales of Ducks and Fish. 1, 1 (January 2023), 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Excel is becoming outdated for big data analysis, python is one tool which can replace it. The book Head First Data Analysis's (HFDA) chapter 3 on optimization utilizes Excels Solver functionality to find the optimal number of rubber ducks and fish needed to maximise profits with limitations on resources. It then looks at a pitfall, not recognising seasonal changes in sales. However, Excel requires a subscription and is less accessible than open-source programming languages. This project looks at replicating the results from the book using Python and its available packages. There are 2 input files (1) for the data constraints and values of profit and (2) for the historical sales of both products. The python script utilises the Pulp package to analyse both files to replicate the results of the Excel Solver.

2 METHODOLOGY

Docker is used to create a replicable environment of HFDAs Chapter 3 on optimization since it has the ability to create an environment which can replicate the results in 1 step. Other alternatives such as Podman and VirtualBox however have more set-up and space requirements than Docker. It is also a widely used, open-source, virtual container which increases its chance of being available long term for future replications. The main challenge is getting docker to produce output from LaTeX. Since Excel requires a license to be run, we have opted to use Python since it is open source, requires minimal set-up, and is more human-readable. The docker container includes Ubuntu packages are used to create a LaTeX compiler, the version of Python to use, and a requirements file to specify the version to use for each python package.

Author's address: Patrick Lowe, lowe03@ads.uni-passau.de, University of Passau, Innstr. 41, Passau, Germany, 94032.

XXXX-XXXX/2023/1-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Initially, we created the script using Jupyter Notebooks for easy readability and corrections, however this is not used as part of the package. Instead, an export of the python script is used. This contains identical code but allows the script to be easily ran in docker. The Python package pandas is used to read the XLS data file, math is used to for calculations, matplotlib is used for the graph, OS is used to create the PDF report through pdflatex, and the main package we use is PuLP. The PuLP package is a linear problem solver package, and has a BSD license allowing for redistribution. Creating a function to run this means that any updates to the resources or profits could easily be adjusted in the excel file.

The challenges in replication were firstly understanding what the book had performed. I choose to reproduce their report by following the Excel Solver steps, but Solver no longer comes pre-installed and has to be manually added through the Addon manager. In selecting the constraints, the first 2 analysis are replicable but the 3rd is open to the reader, therefore I opted to get an average change in sales of both products between the Decembers and Januaries. When the Solver is ran it produced 2 empty sheets, which is seen in the XLS artefacts. This meant providing a named sheet when opening in Python.

The only potential changes to the solver were the limits on productions, so a function was created to be called when needed, passing in 2 arguments; the upper limits on Fish and Ducks. The limitation in reproducibility is the final analysis which uses the readers prediction of values. For consistency, we opted to analyse the historical data sales. This section of code, input block 7, modifies 2 columns into 1 more readable column. The month column uses the first letter of the month, 'J' could be January or June or July. Since the files begins on January, and uses all months of the 3 years, we have changed the month to be the index + 1. So, January at index 0 now changes from 'J' to 1. This is normally not ideal when coding as the XLS file could easily change and skew the month numbers but this felt the most appropriate for a small dataset.

Next, the a dictionary containing the months of December and January for the years 2006 to 2008 was iterated over. It collects the sale of both products for that respective month, calculating the average change in sales from December to January to 2 decimal places. This average is used to predict the sales for the next month, January 2009. Finally, the predicted sales are added to the dataframe to be visualised using the matplotlib package. This graph contains 1 image with 3 lines representing the sale of fish, ducks and total sales, as in the book. We have highlighted the 3 categories more clearly, providing dates instead of letter, a clear legend, markers for our final sales, and an extra month of our predicted sales. The last line in this script uses the operating system package OS to call pdflatex, installed in Docker, producing a PDF LaTeX report.

The data itself could have been stored better. The file has decorations, subsections, and inconsistent layout which created a problem in reading the file to Python. This was resolved with slicing the pandas dataframe and storing the values into variables. The website

for artefacts did not contain the instructions i.e. Chapter 3 of the book, and were sourced elsewhere.

3 RESULTS

Overall, the project was replicable, but not reproducible. The artefacts site did not contain the instructions, Chapter 3 of the book, and had to be sourced elsewhere. Excel was also not provided and when locally installed the Solver add-on had to be manually added, which is not covered in the chapter. The data itself was fully available, and while having no impact on achieving results the structure could be improved. The first analysis uses 400 ducks and 300 fish as a limit. Our replication produces a profit of $ 2320 which matches the profit in the book while also calculating identical limits, 80 fish and 400 ducks to be sold.

The 2nd analysis replicates 150 ducks and 50 fish with a profit of $ 950, again identical to the book. Although there are 2 analyses, we felt the book left it open to interpretation, that 50 fish and 150 ducks were suggestions. We analysed the average change in sales and predicted 98 fish and 133 ducks could be sold, for a profit of $ 1057; slightly higher than what is estimated by HFDS. From the graph, we can see the sales seem to be climbing in comparison to the year previous.

4 FIGURES

HERE

5 CONCLUSIONS

The project is not reproducible as access to Excel is limited, and support for Solver appears to be reducing as it is no longer a default add-on. Instead, it is replicable since we can use a new tool, Python, to produce identical results. On larger scale data, this new tool can be more efficient than Excel and easily updated or customised to better suit the users needs. The artefacts provided by HFDS were reliable but could be improved. Namely, there were 2 blank sheets in the XLS files and also the website does not stick to a naming structure as it does with other chapters. A PDF of the chapter should also be provided as part of the artefacts.

6 REFERENCES

* Github for continuing work * Archive in Zenodo * Conda for environment dependencies * Docker for environment container