BM09BAM - Business Analytics Workshop

Henkel Jet Fuel Price Prediction

Group 4

1 Part 1: Set Up

The project utilizes two main programming languages: Python and R. Python is mainly used for data processing and text sentiment analysis while R is exploited for machine learning models and visualizations. Necessary environments and compilers are required to execute the submitted scripts. For Python, it is recommended to use Anaconda for Jupyter Notebook, and for R, it is suggested to use RStudio. Both recommended environments are free and open-source.

The Python script uses two libraries: pandas and numpy, which are automatically included when installing Anaconda. The necessary R libraries can be installed directly from the submitted script and the instructions can be explicitly found within the script.

2 Part 2: External Data Sources

The models require external data sources as additional predictors. The sources and the corresponding predictors are summarized in Table 1. The datasets can be accessed directly by clicking on the sources' names except for Henkel where the datasets were sent to the team through emails. In some cases, the predictors must be selected or entered in the browser to be extracted correctly. Regarding the predictors from the IATA, they are manually extracted from its monthly economic reports due to budget constraints. The regions in the extractions include Africa, Asia Pacific, Europe, Latin America, the Middle East, and North America. The information obtained from the IMF is collected similarly with additions from special economic groups such as G7 or the European Union.

The chosen period for the analysis is from 2015 to 2023, thus the collection of the datasets in the mentioned time is compressed and submitted along with the scripts. If it is decided to extend the timeframe or update the models in the future, new data has to be added.

3 Part 3: Data Processing

After obtaining the datasets from the mentioned sources, the Python script (Jupyter Notebook) can be executed. It is worth noticing that the datasets and the script have to be placed in the same folder; otherwise, the directory of the datasets has to be explicitly adjusted for the script to run. The script is structured as modules with each module used for one dataset. Each module can function and output independently if needed, except for the merging module which requires all of the dataframes. It is assumed that the name of the dataset file is renamed as shown in the script because the original name can be ambiguous such as *data* or *2015-2023dataset*. It is also assumed that the structure of the dataset or the name of the variables does not change significantly compared to the time the script is written as the row of headers or the position of

Table 1: External Data Sources

Source	Predictors
Henkel	EU Crude Oil Price (Brent) US Crude Oil Price (WTI) EU Jet Fuel Price US Jet Fuel Price
Fusion Media Limited	EU Stock Price STOXX600 US Stock Price S&P500 EUR-USE Exchange Rate
International Energy Agency (IEA)	EU Gasoline Price EU Diesel Price NA Gasoline Price NA Diesel Price
Google	Google Search Trend
Eurostat	EU Gas & Diesel Supply EU Gasoline Supply EU Jet Fuel Supply
International Air Transport Association (IATA)	Regions' Revenue Passenger-Kilometers (RPK) Regions' Cargo Tonne-Kilometers (CTK)
World Gold Council (WGC)	Global Gold Price
International Monetary Fund (IMF)	GDP GDP Change Inflation

variables are hard-coded. Further instructions and the purposes of each syntax are specifically mentioned in the Notebook.

4 Part 4: Machine Learning Models

If the data processing script is properly executed, the output is an Excel file named *Processed_Data*. Similar to Part 2, a sample file for the processed data between 2015 and 2023 is submitted along with the script. To use the R script, the processed data file has to be placed in the same folder or the working directory has to be specifically adjusted to the location of the data. Upon submission, the script is structured as follows:

- Part 0: This part is meant for users to quickly change the prediction period and re-run the whole script with ease. The prediction period in the future has to be an integer and has the same frequency as the rest of the data (currently set monthly).
- Part 1: This part loads the necessary libraries and sets the working directories. The R script is mainly based on tidyverse for the machine learning models and the modeltime for the time-series analysis. Installing instructions are provided in the script.
- Part 2: This part processes the data from the previous part and chooses the predictor to feed into the models.
- Part 3: This part splits the data into a training-testing set and specifies the recipes for the models
- Part 4: This part initiates the workflows, trains the model, and fits the trained models into the test set.

- Part 4.5: This part plots the performance of the candidate models and is used to choose the final model.
- Part 5: This part utilizes the Prophet algorithm to project the predictors into the future using a loop.
- Part 6: This part uses the chosen models to forecast the target variable.
- Part 6.5: This part plots the outcomes of the models and extracts variables' importance.

Detailed instructions for adjusting the script and the purposes of each syntax are specifically mentioned in the script.

5 Part 5: Supplement - Text sentiment analysis

This part documents the optional text sentiment analysis which is recommended as a potential improvement for the model. The submission includes the outcomes from 2019 to 2023 and an article scraping tool.

There is no straightforward method to scrape articles from the previously mentioned period thus they are manually scraped from the online database LexisNexis. The articles used for the early analysis are submitted as three CSV files with the prefixes denoting the year of that file. The articles and the Python script should be placed in the same folder, otherwise, the path to the articles has to be specified in the third cell of the script. If the file is executed correctly (in order of appearance), the output will be two Excel files: one for the body sentiment and one for the title sentiment. The corresponding charts are shown in the Jupyter Notebook.

For convenient usage in the future, a scraping tool (*"articles_scraping"*) is provided in the R environment. The tool only serves as a preliminary solution and requires iterations to achieve a desirable outcome. Noticeably, the tool can scrape article titles directly from Google News (within a specified time window) and provide an average sentiment score that can be used as a supplement for the main model. To use the mentioned tool:

- Run part 1 of the code once.
- Run part 2 with each keyword X, where the keyword X is defined as a search term that would normally be typed in Google (for example, "jet fuel" or "global economy"). As an example, in the code where keyword X is mentioned, the keyword **NEVER** has space between words so it has to be written either without spaces (`jetfuel`, `globaleconomy`) or use _ to separate words (jet_fuel, global_economy)
- Run part 3 of the code once.

If the code runs as intended, the result will return a sentiment score within the range of -1 and +1. It is suggested to investigate the possibilities of using this sentiment score as a lagging feature: the sentiment score of one month has significant predictive power when predicting the price of the month three months ahead.