Georgia Institute of Technology

TEAM 45

AARON JOSEPH, HEMANT KUMAR SHARMA, SHRUTI RAI, MUTHU RESHMAA

PROGRESS REPORT

Mutual Fund Analytics – Selecting the Fund with best performance & NAV Forecasting

November 21, 2022

Contents

List of Figures

List of Tables

1 Introduction & Motivation

1.1 Project Statement

Indian mutual fund market has over 44 fund houses and over 10,000 funds in the market and this makes it the largest in the world, in terms of investment options. Additionally, the Indian market has one of the highest numbers of public stocks, surpassing even that of the United States. Hence, the unique challenge that anyone faces is fund selection. Fund selection is the process of selecting the right mutual funds for your financial goals. Our exercise is to identify the best potential funds to be selected based on historical analysis and use modeling to predict the best funds, from the pool of selected funds.

1.2 Literature Survey

Due to the flux of funds prevailing in the market in each scheme, there is always confusion in the selection of funds to be invested in. Previously researchers had analyzed the performance of a mixture of funds taken from different schemes. But in this project, we have attempted to streamline the fund selection process by first analyzing the funds in each scheme and comparing them with market returns along with a few financial tools like Sharpe and Treynor ratios. Once the best-performing funds are selected in each scheme, we can use them to generate a portfolio providing maximum returns. In addition to that, we took the best-performing funds and analyzed their future returns by applying a series of time-series models ranging from ARIMA to state-of-the-art models like Facebook Prophet which can take daily/weekly seasonality into account and thus can provide more robust results. We also try comparing the performance of these time-series models and try to find out the best model for a given fund. By analyzing the future performance of a fund, we can then decide whether to keep this fund in the mix or can it be replaced by the best-performing fund in that category at that time.

2 Project Overview

2.1 Problem Statement

Hence, the unique challenge that anyone faces is portfolio creation. We're trying to identify the top-performing funds across categories such as equity and debt. These top performers are based on two factors, their performance against the market and risk adjustment.

Once we have identified these funds, we'll forecast NAV for these selected funds. Based on the forecast we'll create a portfolio and optimize the weights that are assigned to each of the selected mutual fund schemes to achieve the highest Sharpe ratio or minimize risk.

2.2 Initial Hypothesis

The NAV forecast is done assuming that the economic conditions remain the same and there are no external factors such as war, recession, or regulation changes at play. Hence the NAV values that are forecasted are only based on the historical performance of the funds.

For the selection of top-performing funds in each category, we have assumed that funds that are very new, and less than two years old are not consistent enough and have been removed from the portfolio.

3 Data Source, Extraction and Cleaning

3.1 Data Sources

There are 2 main sources of data from where most of the analysis was done

1. AMFI Data

2. Sensex Data

Herein, AMFI (Association for Mutual Funds in India) provides all the time series data with respect to 8,000+ Mutual funds. While the Sensex Data is being used for the index data.

3.2 Data Extraction

The data is obtained via web scraping. Post data cleaning and enrichment we stored the final dataset in GCP Big Query. This enabled the team members to have easy access to data, which can be utilized for faster analysis.

AMFI Data Extraction

1. First, we identified an API endpoint on the AMFI website, that would provide data for each of the fund houses
2. Next, we manually identified the Fund house code for over 40+ Fund houses, which was used to automate the scraping process
3. Upon obtaining the data for each fund house, data cleaning logic was implemented, wherein column-wise data is scanned for, and any null values are removed
4. From the AMFI website, it is possible to capture data points that can be obtained from the 1990s. However, it was noticed that some of the old funds get shut down, and therefore we have decided to capture data points from 2017-2022

Sensex Data Extraction

1. Data for Sensex was obtained from the official website, which was needed to measure the performance of the fund relative to the market index, namely the Sensex index, which is a major equity index in India

3.3 Data Architecture

All the data was moved to **GCP BigQuery** into three tables, two for fund details and one for market returns.

BigQuery was used as a sink for data storage. Big Query is an OLAP database, which offers the benefits of serverless querying, keeping costs down for our analysis work. Furthermore, it can be connected to Data Studio for quick EDA. Data Architecture can be seen in Figure 1.

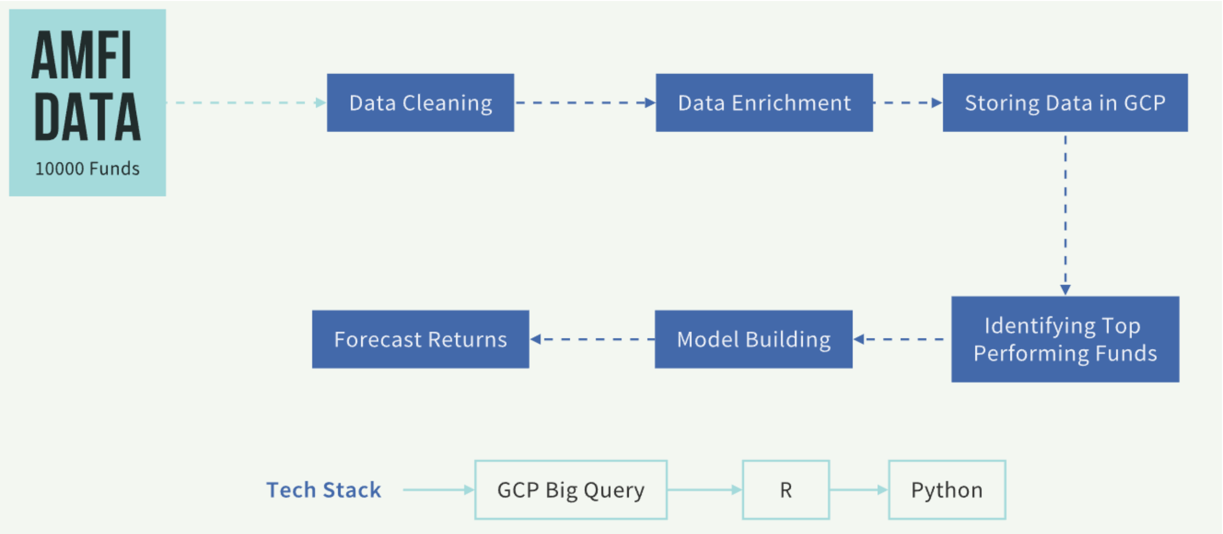


Figure 1: Architecture Diagram

3.4 Data Schematics

Within Bigquery, there are 3 tables wherein data is being stored.

1. **Main NAV Data** - This table contains NAV data of every mutual fund at a day level
2. **Scheme Name** - This table maps the scheme code to mutual fund names and other details like scheme type and the fund house it belongs to.
3. **Market Return** - Contains the day-level market data, inclusive of open and closing values

Table 1: Main Nav Data

Column Name	Significance
Scheme_Code	Scheme Code is the unique identifier for the individual scheme
Net_Asset_Value	Net Asset Value is the value of the fund on the NAV Date, this signifies the value of the fund
Nav_Date	Date on which NAV is applicable

Table 2: Scheme Name

Column Name	Significance
Scheme_Code	Scheme Code is the unique identifier for the individual scheme
Scheme_Name	Name of the scheme
Scheme_Type	Type of scheme
Fund_House_Name	Fund house to which the scheme belongs to

Table 3: Market Return

Column Name	Significance
Date	Date on which Index Details are being logged
Open	Open price of the Index
High	Highest value for the Index on a given date
Low	Lowest value for the Index on a given date
Close	Closing value of the index

4 Methodology

4.1 Data Pre-Processing

For our analysis and modeling, we collected the data for 8138 mutual funds from the AMFI website. For this analysis, we focused on growth and direct funds. We focused on five broad categories that had around 2852 funds. The categories we focused on are:

1. Equity Scheme Small Cap
2. Equity Scheme Mid Cap
3. Equity Scheme Large Cap
4. Debt Schemes (Banking and PSU Fund)

5. Debt Schemes (Corporate Bond Fund)

We removed any funds that had missing values or funds that closed before 2022. We merged the Main_NAV_Data and Scheme_Name to get NAV data by scheme names.

4.2 Exploratory Data Analysis

Once the data was transferred to Google BigQuery, we were able to quickly analyze the data. There were a few key takeaways from the data. Below are the key findings

- **Spike in NAV Values** - For some of the mutual funds, there were spikes in the NAV values. This is due to drastic changes in asset prices. Upon investigating this in the industry we were able to understand, that this is a common occurrence. Therefore, in our modeling stage, we were able to incorporate change point detection to counter such challenges, when it comes to time-series forecasting

- **Funds of the same type** - A single fund can be of growth, dividend re-invest, direct and regular. Therefore, for any given fund, there are multiple variants. Therefore, to keep things simple, we focused on growth and direct funds. Since in Growth fund, there are no dividend payout so the price reflects the asset accurately, which isn't the case with dividend funds, wherein some of it is lost in terms of dividends, however, that doesn't get tracked in the NAV value

4.3 Identifying Top Performers

We first selected the top performers from each scheme to be used for further analysis. For this purpose, we have calculated cumulative returns for all of the schemes in every 5 categories as well as for the market(SENSEX is being used in our case). In addition to that, we found out Sharpe and Treynor ratios for all schemes. Finally, we used a set of rules to identify top-performing funds in each scheme. Below are the steps for the same -

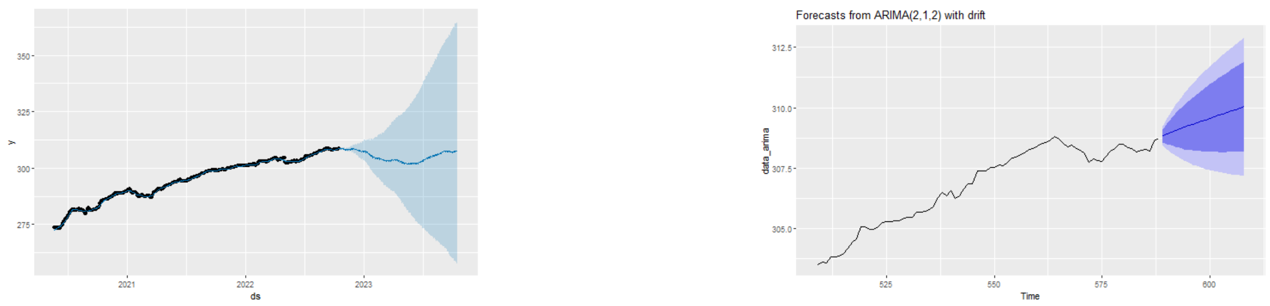
- **For Equity Schemes(Large, Mid and Small)**, we first selected those schemes which are performing better than the market(i.e. having better cumulative returns than the market. Once selected, we sorted them by Treynor Ratio from high to low, to find out the best-performing fund in each scheme. (We preferred the Treynor ratio over Sharpe because it is risk-adjusted)

- **For Debt funds**, as we understand that they rarely perform better than the market, so we directly sorted them by Treynor Ratio from high to low, to find out the best-performing fund in each scheme

4.4 Portfolio Creation: Efficient Frontier Approach

Once the top performers have been identified, we then utilized them to create a portfolio. But as there would be various combinations of different funds in the Portfolio, so to find the optimized portfolio we have used the Simulation modeling technique. In this technique, we have Simulated the random normalized weights for each of the funds 5000 times in the portfolio to calculate Risk, Return, and Sharpe ratio. Finally, we utilized these 5000 simulated points and created a scatter plot with them which is Efficient Frontier. The Efficient Frontier plot can be seen in Figure 2 .

The lower red marked point in the Efficient frontier represents the combination giving the lowest risk while the upper red point(also known as Tangency Point) shows the combination with the Highest Sharpe ratio. As the Sharpe ratio is Market Adjusted return divided by risk, thus we can choose this point as an optimized combination for our portfolio.



(a) FBProphet Plot

(b) ARIMA Plot

Figure 3: Forecast for Aditya Birla Sun Life Fund

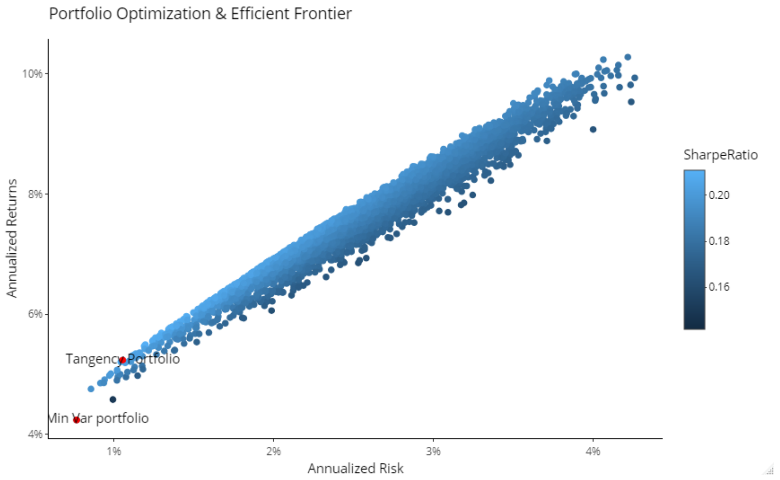


Figure 2: Efficient Frontier Plot

4.5 Time Series Forecasting

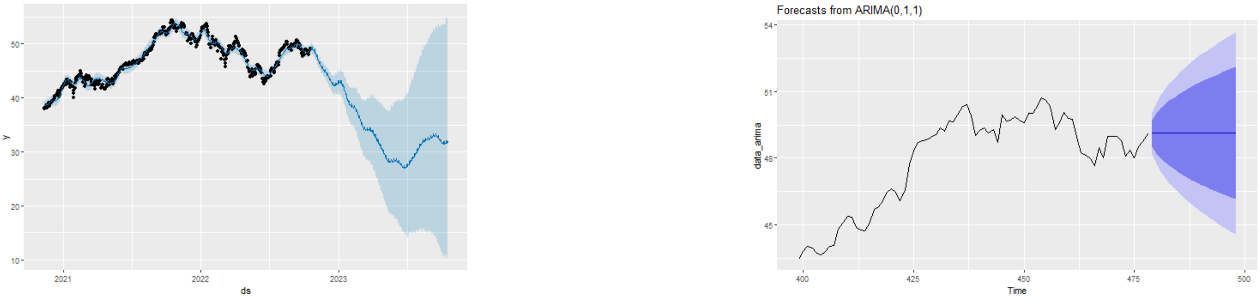
Once the portfolio is created, then our next and final step of analysis is to find out the future Net Asset values of funds in the different categories periodically to predict the performance of each fund currently used in the portfolio(or can be used in the Portfolio in the future). Thus, we can add or remove a fund from the Portfolio by tracking its performance periodically.

To predict the Net Asset value of the fund we have utilized the Time-Series forecasting technique. To avoid any fluctuation in the Net Asset Value of the fund, we also applied the change point detection technique before the Time-Series forecast model to make sure we have used only recent historic trends in the training of the models. We implemented two Time-Series Regression models for Funds forecasting - ARIMA and FBProphet. We compared the accuracy of ARIMA and FBProphet and eventually selected the FBProphet model to forecast Net Asset value since it gave better results as it gives us more flexibility in terms of seasonality(i.e. it considers seasonality at various levels to provide robust results).

The sample plots for the forecasts(using the ARIMA and FBProphet model) of top performers in chosen five categories can be seen in Figures 3, 4, 5, 6 and 7 :

4.6 Efficient Frontier

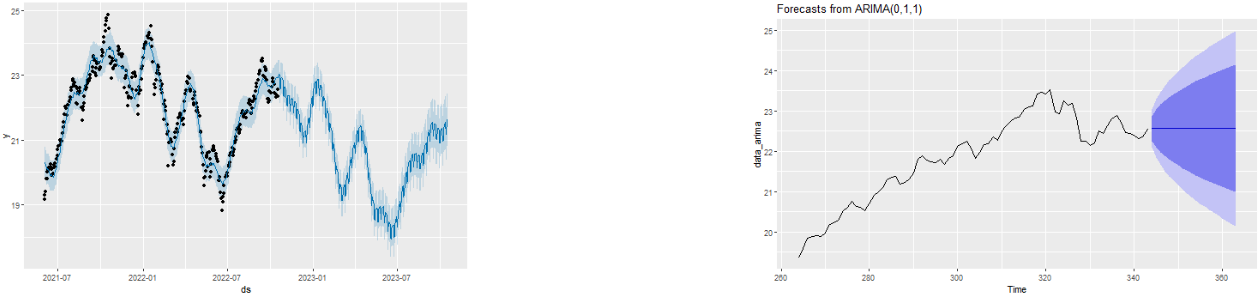
By making use of Efficient Frontier the results obtained are presented in Tables 4 & 5. Table 4 is the optimal combination for minimum risk while table 5 provides the highest return.



(a) FBProphet Plot

(b) ARIMA Plot

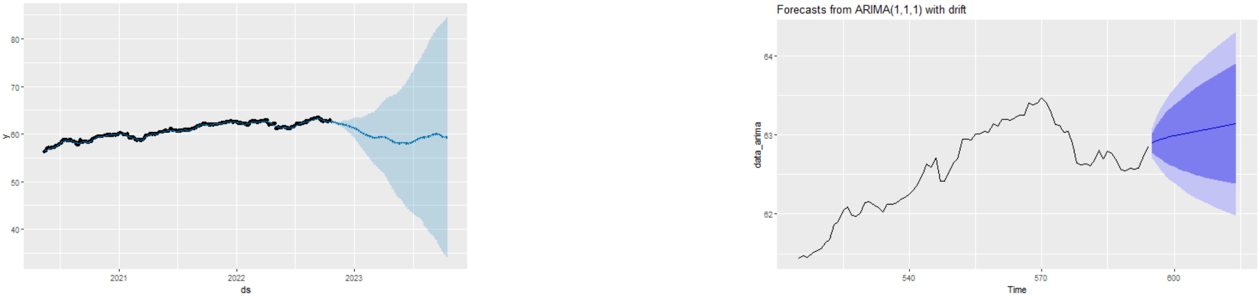
Figure 4: Forecast for Axis Fund



(a) FBProphet Plot

(b) ARIMA Plot

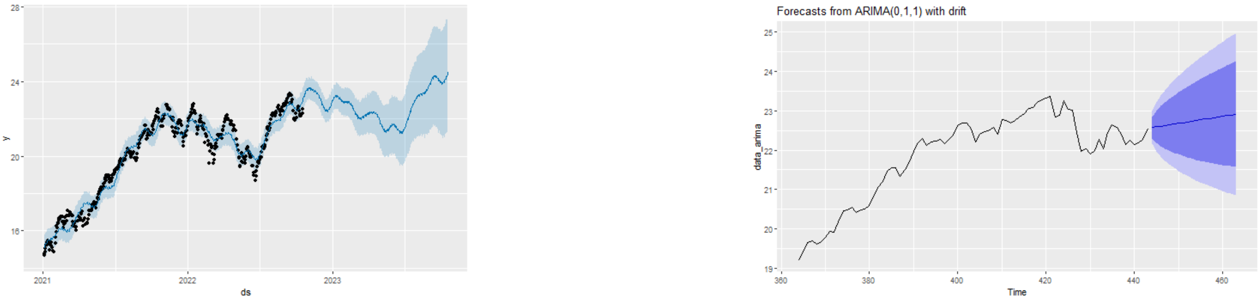
Figure 5: Forecast for IDFC



(a) FBProphet Plot

(b) ARIMA Plot

Figure 6: Forecast for L&T



(a) FBProphet Plot

(b) ARIMA Plot

Figure 7: Forecast for Mirae

Table 4: Portfolio with Minimum Risk

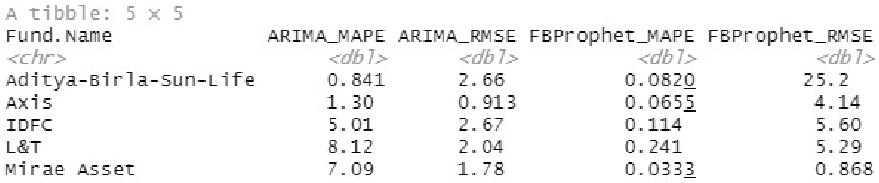
AB	Axis	IDFC	L&T	Mirae	Return	Risk	SharpeRatio
0.44	0.02	0.04	0.45	0.05	0.05	0.0064	0.18

Table 5: Portfolio with Maximum Sharpe Ratio

AB	Axis	IDFC	L&T	Mirae	Return	Risk	SharpeRatio
0.56	0.02	0.21	0.19	0.02	0.05	0.010	0.21

5 Model Results & Comparisons

For the purpose of model comparison, we have divided time-series data into train(80%) and test(20%) and calculated RMSE(root mean square error) and MAPE(Mean Average Percentage Error) from both of the models. From Figure 8, we can observe the errors coming from ARIMA and FBProphet models. We can observe that FBProphet has significantly lower MAPE values for almost all of the funds while the RMSE values are comparable for both of them. Thus, we chose FBProphet over ARIMA even on statistical grounds.



```
A tibble: 5 x 5
Fund.Name                ARIMA_MAPE ARIMA_RMSE FBProphet_MAPE FBProphet_RMSE
<chr>                         <dbl>      <dbl>          <dbl>          <dbl>
Aditya-Birla-Sun-Life         0.841      2.66          0.0820          25.2
Axis                          1.30       0.913         0.0655           4.14
IDFC                          5.01       2.67          0.114            5.60
L&T                           8.12       2.04          0.241            5.29
Mirae Asset                   7.09       1.78          0.0333           0.868
```

Figure 8: Models Error Comparison Table

6 Future Scope

There are certain methodologies that can improve the results further. For portfolio creation, we can build an optimization model which could give us better returns. For NAV forecasting while we have used state-of-the-art machine learning models, we can improve our predictions even more by using neural networks such as LSTM. This can result in an even better portfolio creation.

7 Conclusion and Discussion

From the work done above, we have identified ways of obtaining an optimal portfolio, either by risk or Sharpe ratio. Additionally, forecasting models have been developed using accuracy metrics such as MAPE and RSME to determine the best model for predicting the same. Our work looked into specific types of Debt and Equity funds. Comprehensive work can be done to benchmark all the funds in the market and determine the optimal combination. This will in a way reduce risk by diversifying the portfolio. Furthermore, it would be worth understanding if adding more funds to the Efficient Frontier analysis will make the portfolio basket safer in terms of investment.

References

[1] https://www.iosrjournals.org/iosr-jbm/papers/Vol19-issue3/Version-4/C1903041924.pdf

[2] https://www.investopedia.com/investing/measure-mutual-fund-risk/

[3] https://www.researchgate.net/publication/350312468_Performance_Analysis_of_Mutual_funds_-A_comparative_study_on_equity_diversified_mutual_fund

[4] https://towardsdatascience.com/asset-allocation-model-in-linear-programmin

[5] https://www.codingfinance.com/post/2018-05-31-portfolio-opt-in-r/

[6] https://nextjournal.com/eric-brown/forecasting-with-prophet-part-4

[7] https://search.r-project.org/CRAN/refmans/TSPred/html/fittestArima.html

8 Appendix

8.1 Repository Link

- All our code was stored in GitHub enterprise. The link to the repo can be found here Link
- Youtube Video Link Youtube Link

8.2 Python Logic - Data Scraping

```
# Main Python Logic to obtain Data from AMFI
import pandas as pd
import datetime
import logging
import traceback
import pandas_gbq
from google.oauth2 import service_account
from google.cloud import bigquery

credentials =
    service_account.Credentials.from_service_account_file(’Credentials\master-314712-
project_id = ’master-314712’
client = bigquery.Client(credentials=
    credentials,project=project_id)
```

```
Fund_Dictionary = { "Axis": 53, "Baroda Pioneer": 4,
           "BNP Paribas": 59, "BOI AXA": 46, "Canara Robeco": 32,
              "Daiwa": 60,
           "DBS Chola": 31, "Deutsche": 38, "DHFL": 58,
              "Edelweiss": 47, "Escorts": 13, "Essel": 54,
           "Fidelity": 40, "Fortis": 51,
           "Franklin Templeton": 27, "GIC": 8, "Goldman Sachs":
              49, "HSBC": 37, "IDBI": 57,
           "IDFC": 48, "IIFCL": 68, "IIFL": 62, "IL&F S": 11,
              "IL&FS": 65, "Indiabulls": 63, "ING": 14,
           "Invesco": 42,
           "JM": 16, "JPMorgan": 43, "Kotak Mahindra": 17, "L&T":
              56, "LIC": 18, "Mahindra": 69,
           "Mirae Asset": 45,
           "Morgan Stanley": 19, "Motilal Oswal": 55,
              "PineBridge": 44, "PNB": 34, "PPFAS": 64,
           "PRINCIPAL": 10,
           "Quantum": 41, "Nippon": 21, "Sahara": 35, "Shinsei":
              52, "Shriram": 67,
           "SREI": 66, "Standard Chartered": 2, "Sun F&C": 24,
              "Sundaram": 33, "Tata": 25, "Taurus": 26,
           "Union": 61, "UTI": 28, "SBI": 22, "ICICI": 20, "HDFC":
              9, "DSP-BlackRock": 6,
           "Aditya-Birla-Sun-Life": 3, "Zurich": 29}

# Month Dictionary for URL
Dict_Month = {"Jan": 1, "Feb": 2, "Mar": 3, "Apr": 4, "May": 5,
   "Jun": 6, "Jul": 7, "Aug": 8, "Sep": 9,
         "Oct": 10, "Nov": 11, "Dec": 12}

today_date = datetime.date.today()
end_day, end_month, end_year = today_date.day, today_date.month,
   today_date.year
start_time = datetime.datetime.now() - datetime.timedelta(days=2000)
start_day = start_time.day
start_month = start_time.month
start_year = start_time.year

# Modifying the Month Value to adhere to the URL requirements
start_month_list = [key for key,value in Dict_Month.items() if
   value == start_month]
start_month = ' '.join(map(str,start_month_list))
end_month_list = [key for key,value in Dict_Month.items() if value
   == end_month]
end_month = ' '.join(map(str,end_month_list))

for key,value in Fund_Dictionary.items():
    try:
       Fund_Name = key
```

```
Fund_Name_Value = value
logger.info("Start of {} Fund-House".format(Fund_Name))
url =
   ’https://portal.amfiindia.com/DownloadNAVHistoryReport_Po.aspx?mf={}&tp=1&f
   start_year, end_day, end_month, end_year)
# Loading the data-set from AMFI
df_iter = pd.read_csv(url, delimiter=";", low_memory=False)
df_iter = df_iter.drop([’ISIN Div Payout/ISIN Growth’, ’ISIN
   Div Reinvestment’, ’Repurchase Price’,’Sale
   Price’,’Scheme Name’], axis=1)

# Renaming Columns
df_iter = df_iter.rename(columns={"Scheme Code":
   "Scheme_Code", "Net Asset Value":
   "Net_Asset_Value","Date":"NAV_Date"})

# Converting the dataframe to list of dictionary
list_iter = df_iter.to_dict(’records’)
del(df_iter)

# NAV Data Cleaning
list_iter = [i for i in list_iter if
   str(i[’Net_Asset_Value’]) != ’nan’]
list_iter = [i for i in list_iter if
   str(i[’Net_Asset_Value’]) != ’N.A.’]
list_iter = [i for i in list_iter if
   str(i[’Net_Asset_Value’]) != ’B. C.’]
list_iter = [i for i in list_iter if
   str(i[’Net_Asset_Value’]) != ’B.C.’]

# Converting NAV to str
for i in list_iter:
  i[’Net_Asset_Value’] = str(i[’Net_Asset_Value’])

# List Iter filteration for other string values
for i in list_iter:
  if ’,’ in i[’Net_Asset_Value’]:
    i[’Net_Asset_Value’] =
       i[’Net_Asset_Value’].replace(’,’, ’’)
  elif ’`’ in i[’Net_Asset_Value’]:
    i[’Net_Asset_Value’] =
       i[’Net_Asset_Value’].replace(’`’, ’’)
  elif ’-’ in i[’Net_Asset_Value’]:
    i[’Net_Asset_Value’] =
       i[’Net_Asset_Value’].replace(’-’, ’’)
  elif ’#DIV/0!’ in i[’Net_Asset_Value’]:
    i[’Net_Asset_Value’] =
       i[’Net_Asset_Value’].replace(’#DIV/0!’, ’’)
  # Removing Empty spaces within NAV Value
```

```
        i[’Net_Asset_Value’] = str(i[’Net_Asset_Value’]).replace(’
           ’,’’)

      # Deleting Empty records
      logger.info("No of Null NAV Value :{}".format(len([i for i
         in list_iter if str(i[’Net_Asset_Value’]) == ’’])))
      list_iter = [i for i in list_iter if
         str(i[’Net_Asset_Value’]) != ’’]

      # Storing only Numeric Values from NAV
      Non_Numeric_NAV = len([i for i in list_iter if
         not(str(i[’Net_Asset_Value’]).strip(’-’).replace(’.’,
         ’’).isdigit())])
      logger.info("No of Non-Numeric Values
         :{}".format(Non_Numeric_NAV))
      list_iter = [i for i in list_iter if
         str(i[’Net_Asset_Value’]).strip(’-’).replace(’.’,
         ’’).isdigit()]

      # Converting NAV to float
      for i in list_iter:
        i[’Net_Asset_Value’] = float(i[’Net_Asset_Value’])

      # Code to Modify Data time column
      for i in list_iter:
        i[’NAV_Date’] = datetime.datetime.strptime(i[’NAV_Date’],
           ’%d-%b-%Y’)
        i[’NAV_Date’] = datetime.datetime.strftime(i[’NAV_Date’],
           format="%Y-%m-%d %H:%M:%S")

      # Converting List of Dictionary to Pandas Dataframe
      df_iter = pd.DataFrame(list_iter)
      # Since Postgres columns are lowercase
      df_iter.columns = map(str.lower, df_iter.columns)
      logger.info("Fund Successful - {}".format(Fund_Name))
      # df_iter.rename(columns={’scheme name’:’scheme_name’},
         inplace=True)
      # Logic to Add to BQ Table
      table_id = ’mgt_6203.Main_NAV_Data’
      pandas_gbq.to_gbq(df_iter, table_id, project_id=project_id,
         if_exists=’append’)

    except :
      logger.info(’Failed Main-Update {} Block :
         {}’.format(Fund_Name,traceback.format_exc()))
      logger.info("Failed for Fund House - {}".format(Fund_Name))
```