NORT-P_parser

June 9, 2025

[1]:
```python
#Import modules
import xml.etree.ElementTree as ET
import pandas as pd
```

[7]:
```python
#CONSTANTS
#This dictionary lists the ticker names, and the series ID numbers of␣
↪investment instruments
fundHash = {'VEMIX': 'S000005786', 'VIIIX': 'S000002853', 'VTIVX': 'S000002574',
            'VMCPX': 'S000002844', 'VSCPX': 'S000002845', 'FSMDX': 'S000033637',
            'FSSNX': 'S000033638', 'VTSPX': 'S000038501', 'FXAIX': 'S000006027'}

#This dictionary lists the number of shares owned of each investment instrument
sharesHash = {'VEMIX': 62.01, 'VIIIX': 7.065, 'VTIVX': 0.045,
            'VMCPX': 4.66, 'VSCPX': 5.041, 'FSMDX': 91.872,
            'FSSNX': 112.97, 'VTSPX': 1197.552, 'FXAIX': 33.225}

#This is a prefix that seems to be built into all of the xml tag names
pT = "{http://www.sec.gov/edgar/nport}"

#This is an empty dictionary that describes what data elements should be␣
↪extracted from the xml file
recordFeatures={'name': [], 'lei': [], 'title':[], 'cusip': [],
                'balance':[], 'units':[], 'currencyConditional':␣
↪['curCd','exchangeRt'],
                'valUSD': [], 'pctVal': [], 'payoffProfile': [], 'assetCat':␣
↪[], 'issuerCat': [],
                'invCountry': [], 'isRestrictedSec': [], 'fairValLevel': []}
```

[4]:
```python
#parseRecord(aNode, rF = recordFeatures)
#aNode: XML node that represents an individual investment instrument (XML tag␣
↪invstOrSec)
#rF: Empty dictionary describing what data elements to extract from the XML␣
↪records
#Returns a dictionary of data values for the individual investment record
def parseRecord(aNode, rF = recordFeatures):
    #parseValue(k, v, rH, partStr = "")
    #k: Key value that designates either the tag name or the next-level node
    #v: Empty list (if it's the tag name) or list of 2nd-level tags to extract
    #rH: Dictionary to return, will populate with data values
    #partStr: partial string - not currently implemented, but would be needed␣
↪for deeper nodes
    #No return value
    def parseValue(k, v, rH, partStr = ""):
        #Empty list means the key is the XML tag name
        if len(v) == 0:
            try:
                #Extract the node text
                rH[k] = aNode.find(partStr+pT+k).text
            except AttributeError: #This item is missing
                try:
                    if k == 'issuerCat': #The issuer category had a backup field
                        rH[k] = aNode.find(pT+'issuerConditional').
↪get('issuerCat')
                except KeyError: #Otherwise it's not found
                    #print(f"Attribute not found {rH[k]}: {partStr+pT+k}")
                    rH[k] = ""
        #If the list is not empty, we need to go down a level and extract the␣
↪items
        else:
            #Each item in the list is a sub-value
            for sV in v:
                try:
                    #Get the value from the sub-node
                    rH[sV] = aNode.find(partStr+pT+k).get(sV)
                except AttributeError: #Otherwise it's not found
                    #print(f"Attribute not found {rH['name']}: {partStr+pT+k}")
                    rH[sV] = ""
    #Initialize an empty dictionary
    returnHash = {}

    #The ID record is unique in that it has several different potential tag␣
↪types
    idRecord = aNode.find(pT+'identifiers')[0]
    returnHash['IDtype'] = idRecord.tag.split("}")[1]
    returnHash['ID'] = idRecord.attrib['value']

    #Call parseValue for each value in the record features dictionary
    for k, v in rF.items():
        parseValue(k, v, returnHash)

    return returnHash
```

```
[27]: fundDFhash = {}
      #For each fund in the list
      for aFund, sID in fundHash.items():
          xmlFN = f"dataFiles/{sID}.xml" #Load XML file
          xmlTree = ET.parse(xmlFN)
          rootNode = xmlTree.getroot()
          #Get a list of all the investment instruments in the XML file
          allRecs = rootNode.findall("./"+pT+"formData/"+pT+"invstOrSecs/")
          #Call parseRecord function for each record
          parsedRecs = [parseRecord(aRec) for aRec in allRecs]
          #Transpose the dictionaries to call DataFrame constructor
          df = pd.DataFrame({k: [rec[k] for rec in parsedRecs] for k in parsedRecs[0].
       ↪keys()})

          #Convert missing data
          df = df.replace("N/A", None)

          #Convert to numeric data types
          df['valUSD'] = df['valUSD'].astype(float)
          df['balance'] = df['balance'].astype(float)
          df['pctVal'] = df['pctVal'].astype(float)

          #Calculating average price per share f
          df['avgPricePerShare'] = df['valUSD']/df['balance']
          df['amtInvested'] =␣
       ↪df['avgPricePerShare']*abs(df['pctVal'])*sharesHash[aFund]
          fundDFhash[aFund] = df
          print(f"{aFund}: contains {df.shape[0]} investment instruments")
```

```
VEMIX: contains 5931 investment instruments
VIIIX: contains 506 investment instruments
VTIVX: contains 7 investment instruments
VMCPX: contains 318 investment instruments
VSCPX: contains 1364 investment instruments
FSMDX: contains 813 investment instruments
FSSNX: contains 1973 investment instruments
VTSPX: contains 27 investment instruments
FXAIX: contains 507 investment instruments
```

```
[50]: #For example:
      fundDFhash['VEMIX']
```

```
[50]:      IDtype           ID                                  name  \
      0      isin  CNE000000M72               Wingtech Technology Co Ltd
      1      isin  CNE000001L07  LianChuang Electronic Technology Co Ltd
      2      isin  INE133A01011                     Akzo Nobel India Ltd
      3      isin  CNE100000JH1               Gaona Aero Material Co Ltd
```

```
4       isin  INE647A01010                               SRF Ltd
…      …           …                                  …
5926  ticker           INR                                   N/A
5927    isin  CNE000001CN3        Shinva Medical Instrument Co Ltd
5928    isin  CNE100000767             China Shenhua Energy Co Ltd
5929    isin  CNE100002GQ4                 Bank of Hangzhou Co Ltd
5930    isin  CNE000000GJ4       Sichuan Changhong Electric Co Ltd

                      lei                 title cusip    balance units curCd  \
0                     N/A       WINGTECH TECH-A   N/A  1159831.0    NS   CNY
1                     N/A      LIANCHUANG ELE-A   N/A   747661.0    NS   CNY
2     335800Z6FCJYII12VJ88      AKZO NOBEL INDIA   N/A   152844.0    NS   INR
3                     N/A          GAONA AERO-A   N/A   530560.0    NS   CNY
4     335800436F28GT8ZW506               SRF LTD   N/A  1784858.0    NS   INR
…                      …                    …   …        …    …   …
5926                  N/A  INR/USD FWD 20250319   N/A        1.0    NC   INR
5927  300300517GYTH3UJ9T68      SHINVA MEDICAL-A   N/A   239522.0    NS   CNY
5928  529900N9JOX4C108MA40       CHINA SHENHUA-A   N/A  2429648.0    NS   CNY
5929  300300C1092033000075      BANK OF HANGZH-A   N/A  2629388.0    NS   CNY
5930  300300WM1QVA4ET9HJ12       SICHUAN CHANG-A   N/A  3659100.0    NS   CNY

      exchangeRt       valUSD    pctVal payoffProfile assetCat issuerCat  \
0     0.13765500   5389130.74  0.004920          Long       EC      CORP
1     0.13765500    914577.74  0.000835          Long       EC      CORP
2     0.01154500   6676856.33  0.006096          Long       EC      CORP
3     0.13765500   1090847.53  0.000996          Long       EC      CORP
4     0.01154500  57719140.64  0.052694          Long       EC      CORP
…          …           …        …             …       …        …
5926         N/A  -1620723.05 -0.001480           N/A      DFE     OTHER
5927  0.13765500    530766.74  0.000485          Long       EC      CORP
5928  0.13765500  13388321.53  0.012223          Long       EC      CORP
5929  0.13765500   5355798.58  0.004889          Long       EC      CORP
5930  0.13765500   4347208.51  0.003969          Long       EC      CORP

     invCountry isRestrictedSec fairValLevel  avgPricePerShare    amtInvested
0            CN               N            2      4.646479e+00       1.417565
1            CN               N            2      1.223252e+00       0.063334
2            IN               N            2      4.368412e+01      16.511861
3            CN               N            2      2.056030e+00       0.126968
4            IN               N            2      3.233823e+01     105.666276
…          …              …            …               …            …
5926        N/A               N            2     -1.620723e+06 -148702.399671
5927         CN               N            2      2.215942e+00       0.066583
5928         CN               N            2      5.510396e+00       4.176469
5929         CN               N            2      2.036899e+00       0.617581
5930         CN               N            2      1.188054e+00       0.292380
```

```
[5931 rows x 20 columns]
```

```
[48]: #Try to join them up into a common sheet
      #Starter set of columns
      summarySheet = list(fundDFhash.values())[0][['ID', 'name']]
      #I'm not totally sure what to join on here, because it doesn't seem like there␣
       ↪is a reliable unique ID
      for name, df in fundDFhash.items():
          df = df[['ID', 'name', 'balance', 'valUSD', 'pctVal', 'avgPricePerShare',␣
       ↪'amtInvested']]
          df.columns = ['ID', 'name'] + [c + "_" + name for c in df.columns if not c␣
       ↪in ["ID", "name"]]
          summarySheet = pd.merge(summarySheet, df, how = 'outer', on = ['ID',␣
       ↪'name'])
```

```
[49]: #Dump output to Excel
      with pd.ExcelWriter('output.xlsx') as writer:
          summarySheet.to_excel(writer, sheet_name='Summary')
          for name, df in fundDFhash.items():
              df.to_excel(writer, sheet_name = name)
```