Proje-Tickers-Sectors

March 11, 2024

1 Veri Yogun Uygulamalar Modul Projesi: Zaman serisi siniflandirma problemi

Dr. Ismail Guzel

Milli Teknoloji Hamlesi altinda Yapay Zeka Uzmanlık Programı kapsamında 18 saatlik Veri Yogun Uygulamar egitimi sonunda bu projenin tamamlanmasi beklenilmektedir.

Bu proje icerisinde amac farkli sektorlerden elde dilmis zaman serileri uzerinden elde edilen faktorler uzerine kurulmus bir classification modeli kurarak benzerlik calismasi yapmaktir.

Mesela ilgilenilen bir hisse senedi X olsun, bunun bulundugu sektor bazli diger hisse senetlerin davranislarindan farkli davrandigini dusunelim. Yani sektor icinde bir artis gozlemlenirken bu hisse senetinde bir hareketlilik olmasin. Dolayisiyla, hangi sektore daha cok benziyor sorusuna cevap verebilirsek, o sektor'un hareketlerine gore bir hipotez kurabiliriz.

Bu proje, asagidaki surecleri kapsayacak:

- Sektorleriin listesine bir web-scraping ile erisilmesi ve verilerin elde edilmesi (`yfinance`, `investpy`, `quandl`)
- 2005-01-01 yilindan itibaren aylik getirelerden olusan serilerin elde edilmesi
- 3 buyuk sektor uzerinden getirilerin faktorleri(momentum gibi) hesaplanmasi
- Bu momentum serileri uzerinden bir tsfresh ile feature engineering yapilmasi (imputing, encoding, transformation, ve daha fazlasi)
- Yeni elde edilmis feature ve sektor siniflari uzerinden bir model kurulmasi (en iyi model secmesi)
- Diger sektorlerden ornekler alip ayni feature engine yontemleri yaptik sonra hangi sektore benzedigine karar vermek.
- **Bonus** Mesela Real-Estate sektorunde bulunan butun sembollerin tahmini edildikten sonra cogunluk hangi sektore(T,F,H) benzedigi bilgisine erismek.

[117]:
```
import yfinance
import pandas as pd
import requests
from bs4 import BeautifulSoup

def fetch_sectors_names():
    url = "https://stockanalysis.com/stocks/industry/sectors/"
    response = requests.get(url)
    if response.status_code == 200:
```

```
        soup = BeautifulSoup(response.content, "html.parser")
        df=pd.read_html(str(soup.find_all("table")))[0]
    else:
        print(f"Error: Failed to fetch data from page {url}")

    return df

def fetch_industry_names():
    url = "https://stockanalysis.com/stocks/industry/all/"
    response = requests.get(url)
    if response.status_code == 200:
        soup = BeautifulSoup(response.content, "html.parser")
        df=pd.read_html(str(soup.find_all("table")))[0]
    else:
        print(f"Error: Failed to fetch data from page {url}")

    return df

def fetch_data(sectors):
    url = f"https://stockanalysis.com/stocks/sector/{sectors}/"
    response = requests.get(url)
    if response.status_code == 200:
        soup = BeautifulSoup(response.content, "html.parser")
        df=pd.read_html(str(soup.find_all("table")))[0]
        df.drop(columns='No.', inplace=True)
    else:
        print(f"Error: Failed to fetch data from page {url}")

    return df
```

[118]:
```
sectors=fetch_sectors_names()
indusrty=fetch_industry_names()
```

[119]:
```
sectors
```

[119]:

	Sector Name	Stocks	Market Cap	Div. Yield	PE Ratio	\
0	Financials	1394	9,654.16B	2.39%	14.68	
1	Healthcare	1219	8,190.44B	0.42%	50.64	
2	Technology	788	17.83T	0.42%	44.79	
3	Industrials	650	5,432.80B	1.10%	26.52	
4	Consumer Discretionary	578	7,315.94B	0.64%	27.68	
5	Materials	263	2,042.11B	1.56%	19.21	
6	Real Estate	260	1,525.70B	4.06%	49.91	
7	Communication Services	260	5,272.98B	1.08%	27.55	
8	Energy	253	3,553.66B	2.94%	6.81	
9	Consumer Staples	242	4,017.29B	1.44%	28.67	
10	Utilities	108	1,291.84B	3.54%	24.71	

	Profit Margin	1D Change	1Y Change
0	17.81%	0.20%	107.28%
1	4.11%	0.31%	2.47%
2	13.21%	1.62%	8.65%
3	7.32%	0.64%	8.17%
4	5.92%	0.68%	-3.84%
5	8.76%	1.03%	-5.90%
6	9.22%	0.35%	3.89%
7	10.39%	1.05%	-7.45%
8	13.72%	0.99%	0.25%
9	4.86%	0.04%	4.13%
10	7.82%	0.50%	-6.71%

1.1 Sektor listerine erismek

Yukarida yazilan fonksyionlar ile hangi sembollerin hangi sektorlerde oldugu bilgisine erisim saglanabilir ve asagidaki betikler yardimi ile `.csv` dosyalarinda saklayabiliriz. Sonrasinda, sektor bazli sembollere ait verileri indirilebilir ve siniflandirilabilir.

[120]:
```
fetch_data(sectors='energy').to_csv('../data/stock_sectors/energy.csv')
fetch_data(sectors='financials').to_csv('../data/stock_sectors/financials.csv')
fetch_data(sectors='healthcare').to_csv('../data/stock_sectors/healthcare.csv')
fetch_data(sectors='technology').to_csv('../data/stock_sectors/technology.csv')
fetch_data(sectors='utilities').to_csv('../data/stock_sectors/utilities.csv')
fetch_data(sectors='real-estate').to_csv('../data/stock_sectors/real-estate.
 ↪csv')
fetch_data(sectors='materials').to_csv('../data/stock_sectors/materials.csv')
fetch_data(sectors='technology').to_csv('../data/stock_sectors/technology.csv')
fetch_data(sectors='industrials').to_csv('../data/stock_sectors/industrials.
 ↪csv')
fetch_data(sectors='consumer-staples').to_csv('../data/stock_sectors/
 ↪consumer-staples.csv')
fetch_data(sectors='consumer-discretionary').to_csv('../data/stock_sectors/
 ↪consumer-discretionary.csv')
fetch_data(sectors='communication-services').to_csv('../data/stock_sectors/
 ↪communication-services.csv')
```

[123]:
```
finance = pd.read_csv('../data/stock_sectors/financials.csv')
finance.Symbol
```

[123]:
```
0       BRK.B
1           V
2         JPM
3          MA
4         BAC
        ...
```

```
1002     RELI
1003     NCPL
1004     TIRX
1005      DXF
1006    AIMAU
Name: Symbol, Length: 1007, dtype: object
```

1.2 Veriye erismek

Diyelim ki, finans sektorunden `HSBC` sembolu icin verileri indirmek istiyoruz. Bu adim icin `yfinance` kullanilabilir. Oncelikle `.Ticker` ile bir object olusturup onun uzerinden dogru hissemi olduguna dair bilgileri teyit edebiliriz. Sonrasinda `.get_history_metadata()` ile sembolun metedatasina erisim saglayabiliriz. Sonrasinda, `.history(period='3y')` ile 3 yillik veriyi calisma ortamimiza indirebiliriz.

[124]:
```
import yfinance
ticker_name = yfinance.Ticker("HSBC")
ticker_name.info
```

[124]:
```
{'address1': '8 Canada Square',
 'city': 'London',
 'zip': 'E14 5HQ',
 'country': 'United Kingdom',
 'phone': '44 20 7991 8888',
 'fax': '44 20 7992 4880',
 'website': 'https://www.hsbc.com',
 'industry': 'Banks - Diversified',
 'industryKey': 'banks-diversified',
 'industryDisp': 'Banks - Diversified',
 'sector': 'Financial Services',
 'sectorKey': 'financial-services',
 'sectorDisp': 'Financial Services',
 'longBusinessSummary': 'HSBC Holdings plc provides banking and financial
services worldwide. The company operates through Wealth and Personal Banking,
Commercial Banking, and Global Banking and Markets segments. The Wealth and
Personal Banking segment offers retail banking and wealth products, including
current and savings accounts, mortgages and personal loans, credit and debit
cards, and local and international payment services; and wealth management
services comprising insurance and investment products, global asset management
services, investment management, and private wealth solutions. This segment
serves personal banking and high net worth individuals. The Commercial Banking
segment provides credit and lending, treasury management, payment, cash
management, commercial insurance, and investment services; commercial cards;
international trade and receivables finance services; foreign exchange products;
capital raising services on debt and equity markets; and advisory services. It
serves small and medium sized enterprises, mid-market enterprises, and
corporates. The Global Banking and Markets segment offers financing, advisory,
```

and transaction services; and credit, rates, foreign exchange, equities, money markets, and securities services; and engages in principal investment activities. It serves government, corporate and institutional clients, and private investors. HSBC Holdings plc was founded in 1865 and is headquartered in London, the United Kingdom.',
 'fullTimeEmployees': 220861,
 'companyOfficers': [{'maxAge': 1,
   'name': 'Mr. Noel Paul Quinn',
   'age': 61,
   'title': 'Group CEO, Member of the Group Management Board & Executive Director',
   'yearBorn': 1962,
   'fiscalYear': 2023,
   'totalPay': 6921865,
   'exercisedValue': 0,
   'unexercisedValue': 0},
  {'maxAge': 1,
   'name': 'Mr. Georges Bahjat Elhedery',
   'age': 49,
   'title': 'Group CFO, Member of the Group Management Board & Executive Director',
   'yearBorn': 1974,
   'fiscalYear': 2023,
   'totalPay': 4181828,
   'exercisedValue': 0,
   'unexercisedValue': 0},
  {'maxAge': 1,
   'name': 'Ms. Manveen Pam Kaur',
   'age': 59,
   'title': 'Group Chief Risk & Compliance Officer and Member of the Group Management Board',
   'yearBorn': 1964,
   'fiscalYear': 2023,
   'exercisedValue': 0,
   'unexercisedValue': 0},
  {'maxAge': 1,
   'name': 'Mr. Stephen Colin Moss',
   'age': 56,
   'title': 'CEO of Middle East North Africa & Turkey (MENAT) Region and Member of Group Management Board',
   'yearBorn': 1967,
   'fiscalYear': 2023,
   'exercisedValue': 0,
   'unexercisedValue': 0},
  {'maxAge': 1,
   'name': 'Mr. Colin William Bell',
   'age': 55,

```
   'title': 'CEO of HSBC Bank plc & HSBC Europe and Member of the Group
Management Board',
   'yearBorn': 1968,
   'fiscalYear': 2023,
   'exercisedValue': 0,
   'unexercisedValue': 0},
  {'maxAge': 1,
   'name': 'Mr. John David Stuart',
   'age': 60,
   'title': 'CEO of HSBC UK Bank plc & Member of the Group Management Board',
   'yearBorn': 1963,
   'fiscalYear': 2023,
   'exercisedValue': 0,
   'unexercisedValue': 0},
  {'maxAge': 1,
   'name': 'Ms. Elaine  Arden',
   'age': 53,
   'title': 'Group Chief Human Resources Officer & Member of the Group
Management Board',
   'yearBorn': 1970,
   'fiscalYear': 2023,
   'exercisedValue': 0,
   'unexercisedValue': 0},
  {'maxAge': 1,
   'name': 'Mr. Gregory L. Guyett',
   'age': 59,
   'title': 'Chief Executive of Global Banking & Markets and Member of the Group
Management Board',
   'yearBorn': 1964,
   'fiscalYear': 2023,
   'exercisedValue': 0,
   'unexercisedValue': 0},
  {'maxAge': 1,
   'name': "Mr. Barry  O'Byrne",
   'age': 46,
   'title': 'CEO of Global Commercial Banking & Member of the Group Management
Board',
   'yearBorn': 1977,
   'fiscalYear': 2023,
   'exercisedValue': 0,
   'unexercisedValue': 0},
  {'maxAge': 1,
   'name': 'Mr. Jonathan  Calvert-Davies',
   'age': 53,
   'title': 'Group Head of Internal Audit & Member of the Group Management
Board',
   'yearBorn': 1970,
```

```
  'fiscalYear': 2023,
  'exercisedValue': 0,
  'unexercisedValue': 0}],
'auditRisk': 1,
'boardRisk': 3,
'compensationRisk': 8,
'shareHolderRightsRisk': 1,
'overallRisk': 2,
'governanceEpochDate': 1709251200,
'compensationAsOfEpochDate': 1703980800,
'maxAge': 86400,
'priceHint': 2,
'previousClose': 37.68,
'open': 37.83,
'dayLow': 37.715,
'dayHigh': 38.03,
'regularMarketPreviousClose': 37.68,
'regularMarketOpen': 37.83,
'regularMarketDayLow': 37.715,
'regularMarketDayHigh': 38.03,
'dividendRate': 3.05,
'dividendYield': 0.0776,
'exDividendDate': 1709769600,
'payoutRatio': 0.46490002,
'fiveYearAvgDividendYield': 6.08,
'beta': 0.593,
'trailingPE': 6.642983,
'forwardPE': 11.136765,
'volume': 978259,
'regularMarketVolume': 978259,
'averageVolume': 2236426,
'averageVolume10days': 2994290,
'averageDailyVolume10Day': 2994290,
'bid': 37.89,
'ask': 37.91,
'bidSize': 1300,
'askSize': 1400,
'marketCap': 144247095296,
'fiftyTwoWeekLow': 32.41,
'fiftyTwoWeekHigh': 42.47,
'priceToSalesTrailing12Months': 2.5597057,
'fiftyDayAverage': 39.341,
'twoHundredDayAverage': 38.99255,
'trailingAnnualDividendRate': 0.61,
'trailingAnnualDividendYield': 0.01618896,
'currency': 'USD',
'enterpriseValue': 312753586176,
```

```
'profitMargins': 0.41759998,
'floatShares': 18744149907,
'sharesOutstanding': 3809509888,
'sharesShort': 10200971,
'sharesShortPriorMonth': 10900292,
'sharesShortPreviousMonthDate': 1705017600,
'dateShortInterest': 1707955200,
'sharesPercentSharesOut': 0.0027,
'heldPercentInstitutions': 0.014839999,
'shortRatio': 5.07,
'impliedSharesOutstanding': 4159399936,
'bookValue': 8.819,
'priceToBook': 4.293571,
'lastFiscalYearEnd': 1703980800,
'nextFiscalYearEnd': 1735603200,
'mostRecentQuarter': 1703980800,
'netIncomeToCommon': 22432000000,
'trailingEps': 5.7,
'forwardEps': 3.4,
'enterpriseToRevenue': 5.55,
'52WeekChange': 0.06835508,
'SandP52WeekChange': 0.30279303,
'lastDividendValue': 0.5,
'lastDividendDate': 1699488000,
'exchange': 'NYQ',
'quoteType': 'EQUITY',
'symbol': 'HSBC',
'underlyingSymbol': 'HSBC',
'shortName': 'HSBC Holdings, plc.',
'longName': 'HSBC Holdings plc',
'firstTradeDateEpochUtc': 932131800,
'timeZoneFullName': 'America/New_York',
'timeZoneShortName': 'EST',
'uuid': 'f8f8f7fc-ceff-3d4c-a6e2-e4f9670b27aa',
'messageBoardId': 'finmb_382645',
'gmtOffSetMilliseconds': -18000000,
'currentPrice': 37.865,
'targetHighPrice': 48.52,
'targetLowPrice': 48.52,
'targetMeanPrice': 48.52,
'targetMedianPrice': 48.52,
'recommendationMean': 1.0,
'recommendationKey': 'strong_buy',
'numberOfAnalystOpinions': 1,
'totalCash': 1083986018304,
'totalCashPerShare': 57.033,
'totalDebt': 643836018688,
```

```
 'totalRevenue': 56353001472,
 'revenuePerShare': 3.61625,
 'returnOnAssets': 0.0082,
 'returnOnEquity': 0.13001001,
 'operatingCashflow': -5910000128,
 'revenueGrowth': -0.54,
 'operatingMargins': -0.124560006,
 'financialCurrency': 'USD',
 'trailingPegRatio': 0.6227}
```

[126]:
```
ticker_name.get_history_metadata()
```

[126]:
```
{'currency': 'USD',
 'symbol': 'HSBC',
 'exchangeName': 'NYQ',
 'instrumentType': 'EQUITY',
 'firstTradeDate': 932131800,
 'regularMarketTime': 1709833492,
 'hasPrePostMarketData': True,
 'gmtoffset': -18000,
 'timezone': 'EST',
 'exchangeTimezoneName': 'America/New_York',
 'regularMarketPrice': 37.87,
 'chartPreviousClose': 30.64,
 'priceHint': 2,
 'currentTradingPeriod': {'pre': {'timezone': 'EST',
   'start': 1709802000,
   'end': 1709821800,
   'gmtoffset': -18000},
  'regular': {'timezone': 'EST',
   'start': 1709821800,
   'end': 1709845200,
   'gmtoffset': -18000},
  'post': {'timezone': 'EST',
   'start': 1709845200,
   'end': 1709859600,
   'gmtoffset': -18000}},
 'dataGranularity': '1d',
 'range': '3y',
 'validRanges': ['1d',
  '5d',
  '1mo',
  '3mo',
  '6mo',
  '1y',
  '2y',
  '5y',
```

```
'10y',
'ytd',
'max']}
```

[128]:
```
data=ticker_name.history(period='3y')
data.tail()
```

[128]:

Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
2024-03-01 00:00:00-05:00	39.060001	39.160000	38.790001	38.980000	2206000	0.0	0.0
2024-03-04 00:00:00-05:00	38.689999	38.910000	38.680000	38.779999	2124700	0.0	0.0
2024-03-05 00:00:00-05:00	38.730000	39.119999	38.730000	38.930000	2212500	0.0	0.0
2024-03-06 00:00:00-05:00	39.150002	39.340000	38.990002	39.230000	3372300	0.0	0.0
2024-03-07 00:00:00-05:00	37.830002	38.029999	37.715000	37.869999	986622	0.0	0.0

Simdi ise, belirlenen sembollerin verilerini belli bir tarih sonrasinda cekebiliriz. Sonrasinda aylik getirileri hesaplayabiliriz. Proje kapsaminda bu yontem kullanilacak.

[135]:
```
ticker_list=['AAPL', 'NVDA', 'INTC', 'ORCL', 'SONY']
data = yfinance.download(ticker_list, start='2005-01-01')
data_close=data['Adj Close'].resample('M').last().pct_change()+1
```

```
[**********100%%**********]  5 of 5 completed
```

[136]:
```
data_close
```

[136]:

Date	AAPL	INTC	NVDA	ORCL	SONY
2005-01-31	NaN	NaN	NaN	NaN	NaN
2005-02-28	1.166710	1.072397	1.264834	0.940450	1.020800
2005-03-31	0.928890	0.968320	0.819593	0.963707	1.059010
2005-04-30	0.865371	1.012484	0.922980	0.926282	0.917291
2005-05-31	1.102607	1.150120	1.235750	1.107266	1.015255
…	…	…	…	…	…
2023-11-30	1.113780	1.228684	1.146886	1.123888	1.034919
2023-12-31	1.013583	1.124161	1.058934	0.907237	1.101687
2024-01-31	0.957773	0.857314	1.242418	1.063576	1.032316
2024-02-29	0.981457	1.002233	1.285809	0.999821	0.877852
2024-03-31	0.935657	1.074564	1.161446	1.018893	1.012703

[231 rows x 5 columns]

[137]:
```
# 3 aylik momentum faktorleri
def get_rolling_ret(data, n):
    return data.rolling(n).apply(np.prod)
calisilacak_veri = get_rolling_ret(data_close, 3).dropna()
calisilacak_veri
```

[137]:

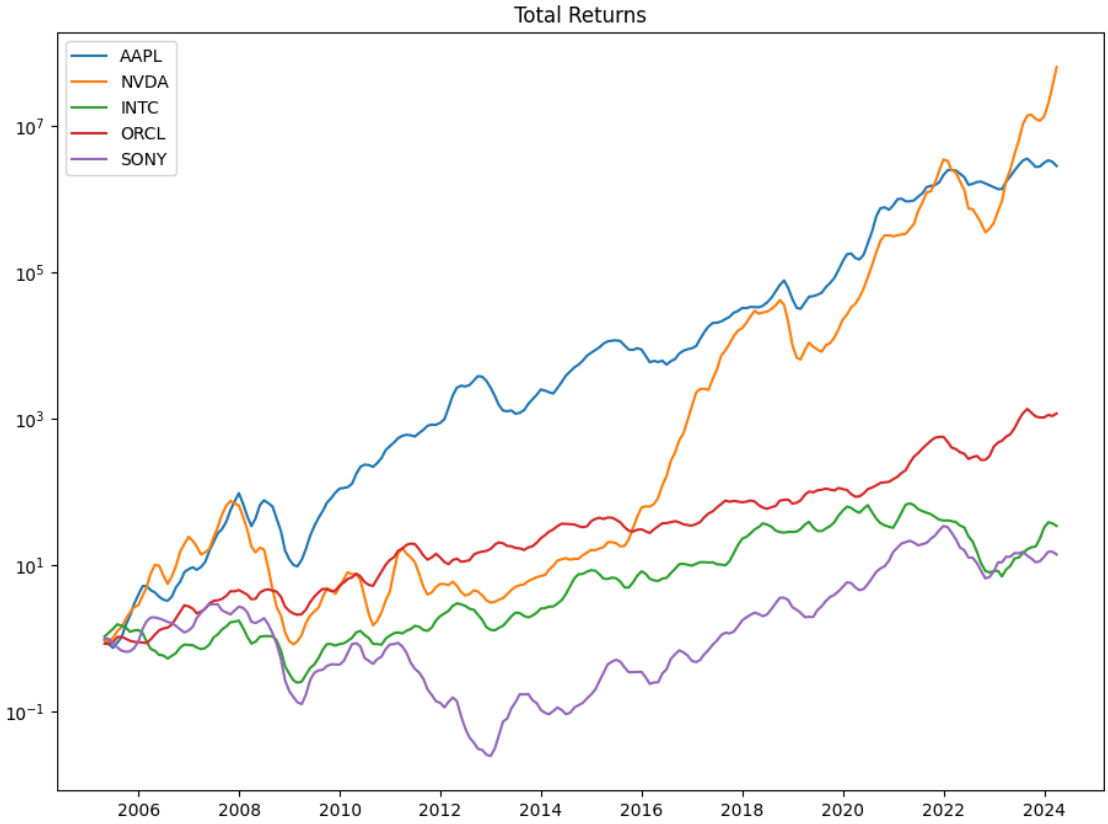
Ticker	AAPL	INTC	NVDA	ORCL	SONY
Date					
2005-04-30	0.937842	1.051387	0.956806	0.839506	0.991626
2005-05-31	0.886313	1.127587	0.934805	0.988417	0.986240
2005-06-30	0.883370	1.123876	1.124579	1.057692	0.860570
2005-07-31	1.182751	1.157798	1.233926	1.173876	0.885590
2005-08-31	1.179326	0.956783	1.132103	1.014844	0.902066
…	…	…	…	…	…
2023-11-30	1.012403	1.276238	0.947703	0.968812	1.033177
2023-12-31	1.126007	1.418150	1.138563	0.999016	1.149011
2024-01-31	1.081238	1.184155	1.508887	1.084457	1.177002
2024-02-29	0.952780	0.965910	1.691660	0.964743	0.998371
2024-03-31	0.879529	0.923295	1.855424	1.083476	0.917732

[228 rows x 5 columns]

[138]:
```
import matplotlib.pyplot as plt
plt.figure(figsize=[11,8])

for ticker in ticker_list:
    plt.plot(calisilacak_veri[ticker].cumprod(), label = ticker)

plt.yscale('log')
plt.title('Total Returns')
plt.legend()
plt.show()
```


Total Returns
AAPL
NVDA
INTC
ORCL
SONY
10^7
10^5
10^3
10^1
10^{-1}
2006
2008
2010
2012
2014
2016
2018
2020
2022
2024

[ ]: