Прогнозирование рынка: ИИ в работе с временными рядами

Автор:Васюрина Варвара

Куратор: Матвеев Иван Алексеевич

Постановка проблемы

Анализ портфеля: Новым в исследовании является максимизация коэффициента Шарпа с помощью построения генеративной сети

$$Sharpe\ Ratio = \frac{R_p - R_f}{\sigma_p}$$

Sharpe Ratio – коэффициент Шарпа

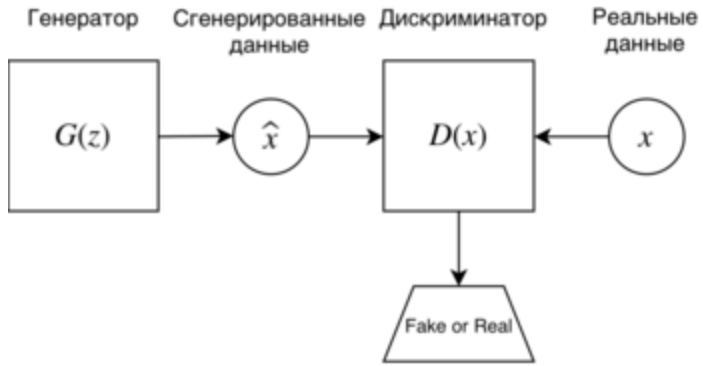
R_p – доходность портфеля

R_f – безрисковая ставка доходности

σ_p – стандартное отклонение доходности портфеля (волатильность)

Постановка проблемы

Модель использует набор данных, определяющих цены на акции за определенное количество предыдущих дней. Затем модель выведет прогнозируемые данные о запасах на следующий период времени.
Мы обучаем его, используя дискриминатор, цель которого - определить, являются ли данные корректными(погрешности из сезонности, погрешностей реальной жизни)

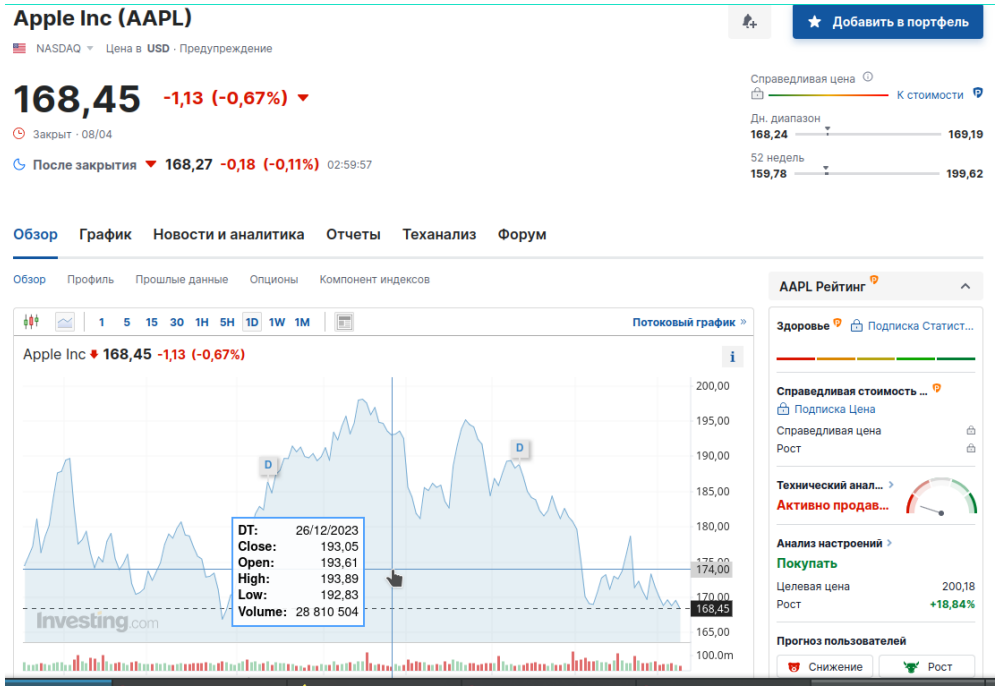


Теоретические результаты

- обсуждение с куратором (план и показ промежуточных результатов)
- Анализ временных рядов, применение нейросетей(https://habr.com/ru/articles/693562/)- то, как обрабатывала датасет
- Событийно-ориентированный бэктестинг на Python шаг за шагом(https://habr.com/ru/companies/iticapital/articles/268929/)

Результаты

- Задача датасета: взяла котировки акций apple-> считываю в формате csv




```
2024-01-10,167.51750183105469,166.46499633789062,167.1500015258789,167.12000274658203,90420400,166.4460220336914
2024-01-11,167.7750015258789,167.125,167.20249938964844,167.69249725341797,78756800,167.01276397705078
2024-01-12,168.13999938964844,166.83000183105469,166.94499969482422,167.86499786376953,137310400,167.18354797363281
2024-01-13,168.82499694824219,167.73249816894531,167.86499786376953,168.7874984741211,133587600,168.09677124023438
2024-01-16,170.19750213623047,169.24500274658203,169.25,69.96499633789062,128186000,169.26245880126953
2024-01-17,170.44249725341797,169.69999694824219,169.89250183105469,170.10250091552734,114158400,169.39857482910156
2024-01-18,170.4749984741211,169.77999877929688,169.94999694824219,169.93499755859375,116028400,169.23274993896484
2024-01-19,170.29499816894531,169.73750305175781,169.875,170.00499725341797,98369200,169.30204772949219
2024-01-20,170.6624984741211,169.63999938964844,170.55750274658203,169.86000061035156,275978000,169.15850830078125
2024-01-23,171.0625,170.09249877929688,170.13249969482422,171.0,98572000,170.28706359863281
2024-01-24,171.22250366210938,170.73000335693336,171.17250061035156,171.06749725341797,48478800,170.3538818359375
2024-01-26,172.49500274658203,171.17500305175781,171.20500183105469,172.47750091552734,93121200,171.7497329711914
2024-01-27,173.49250030517578,172.02999877929688,172.77999877929688,172.44999694824219,146266000,171.72248840332031
2024-01-30,173.17250061035156,171.30500030517578,172.36499786376953,172.87999725341797,144114400,172.14817810058594
2024-01-31,173.41999816894531,172.37999725341797,172.48249816894531,173.4124984741211,100805600,172.67533874511719
```

```
import numpy as np
import array
import matplotlib.pyplot as plotter
import pandas
import csv

def readData():
    file = "data/datasset.csv"
    # reader = pandas.read_csv("data/Downloads/datasset.csv",index_col='Date',parse_dates = True )
    df = pandas.read_csv(file, parse_dates=True, index_col='Date',
                         usecols=['Date', 'Close Price'])
    df = df.fillna(method='ffill')
    dfAll = pandas.read_csv(file)
    dfAll = dfAll.fillna(method='ffill')
    stock = np.array(dfAll)
    return df, stock
```

Результаты

```
class Generator():
    def __init__(self):
        pass

    def SMA(self, dataset, windows):
        result = dataset.rolling(window = windows).mean()
        return result

    def EMA(self, dataset, windows):
        res = dataset.ewm(span = windows).mean()
        return res

    def MACD(self, dataset, long, short, windows):
        l_ = dataset.ewm(span = long).mean()
        sh_ = dataset.ewm(span = short).mean()
        result = (short_ - long_).ewm(span = windows).mean()
        return result

    def RSI(self, data, windows):
        delta = data.diff(1)
        up = delta.copy()
        down = delta.copy()
        up[up < 0] = 0
        down[down > 0] = 0
        avg_up = up.rolling(window = windows).mean()
        avg_down = down.rolling(window = windows).mean()
        rs = avg_up/ avg_down
        rsi = 100. -(100./ (1. + rs))
        return rsi


    def boll_band(self, dataset, windows):
        std = dataset.rolling(window = windows).std()
        sma = dataset.rolling(window = windows).mean()
        up = sma + 2 * std
        low = sma - 2 * std
        return up, low
```

- Задание генератора прогнозов

(pandas.DataFrame.rolling)

- Черновик дискриминатор и работа с временным рядом(Series из pandas)

```
def create_sharpe_ratio(returns, periods=30):
    """
    Создает коэффициент Шарпа для стратегии, основанной на бенчмарке ноль (нет информации о рисках ).

    Параметры:
    returns -  Series из Pandas представляет процент прибыли за период - Дневной (252), Часовой (30*6.5), Минутный (30*6.5*60) и т.п..
    """
    return np.sqrt(periods) * (np.mean(returns)) / np.std(returns)
```

Соответствие с задачами по первому докладу

- Пригодились github, статьи на хабр, поднятие вопроса важности актуарной математики в исследовании
- Сужение цели работы
- Часть инструментов встроено в библиотеку python, нейронная сеть нужна только для прогноза

Дальнейшая работа

- оценка волатильности, безрисковой ставки и подсчёт коэффициент Шарпа
- коррекция кода(иерархию файлов, соотнести функции подсчёта с сетью прогноза)
- рассмотреть датасет по золоту
- сравнить прогнозированные коэффициенты, посчитанные с помощью сети и посчитанные коэффициенты через pandas