МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования

«Гомельский государственный технический университет

имени П.О.Сухого»

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

**ЛАБОРАТОРНАЯ РАБОТА №1**

по дисциплине: «Введение в кластерный анализ и классификацию»

на тему: **«**Создание реляционной базы данных и ее таблиц посредством специализированного программного обеспечения**»**

Выполнил: студент гр. ИТИ-22

Василенко Р.В.

Принял: ассистент

Карась О. В.

Гомель 2024

**Цель работы:** изучить основы ПО для изучения, анализа и работы с данными. Выполнить задание согласно варианту.

Ход выполнения работы:

1. Загрузить исходные данные согласно варианта (таблица 1) из csv файлов

2. На основании считанных данных сформировать DataFrame

3. Провести анализ на полноту данных

4. Построить гистограммы и графики плотности распределения по каждому признаку в исходных данных, приэтом уметь изменять цвета, тип

графика и строить графики на одном поле.

5. Согласно заданию, выбрать несколько параметров и построить для них

распределения на одном графике

6. Вычислить математическое ожидание и дисперсию

7. Основываясь на решаемой задаче, визуализировать совместное распределение отдельных признаков нескольких переменных

8. Выполнить агрегирование данных по каждому из признаков (или по значению поля, например, по месяцу или году при анализе акций) и визуализировать с использованием различных функций (barplot, countplot, boxplot, violinplot, stripplot, swarmplot и др.), при этом уметь пояснить,

что изображено на графиках

9. Построить матрицу диаграмм рассеивания, тепловую карту и матрицу

корреляции (scatterplotmatrix, heatmap, plotcorr)

10.Провести анализ на основе построенных графиков

11.Согласно индивидуального задания, построить уравнение регрессии

(линейное, полиномиальное, логистическая, пробит-модель, «эластичная сеть»)

12.Решить задачу, согласно варианту задания

Вариант 2 - Найти месяц, в котором наблюдался наибольший рост курса акций на конец месяца по сравнению с началом месяца. Восстановить значения в пропущенные дни за январь месяц 2017 года

**Ход работы и результаты выполнения**

Чтение данных из .csv файла происходит с помощью библиотеки pandas, команды .read\_csv(). На рисунке 1 изображен результат чтения csv файла.

**

Рисунок 1 – Результат выполнения задания.

Проверка на полноту данных происходит с помощью метода .info(), в котором мы можем проверить, есть ли в столбцах нулевые ячейки.

С помощью .mean() и .var() можно найти среднее значение и дисперсию выбранных данных. На рисунке 2 продемонстрированно выполнение описанных выше команд.

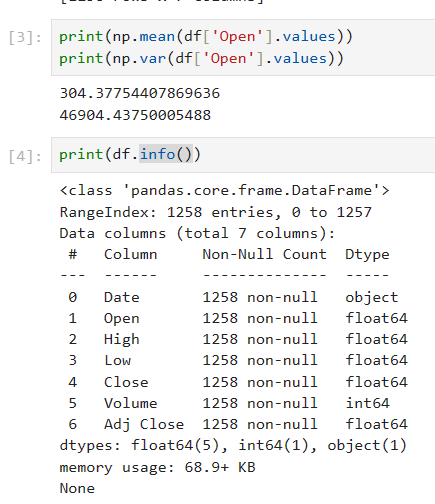


Рисунок 2 – Результат выполнения команд.

Для визуального изображения собранных данных используется библиотека matplotlib. На рисунке 3 изображен график зависимости цены начала дня от даты.

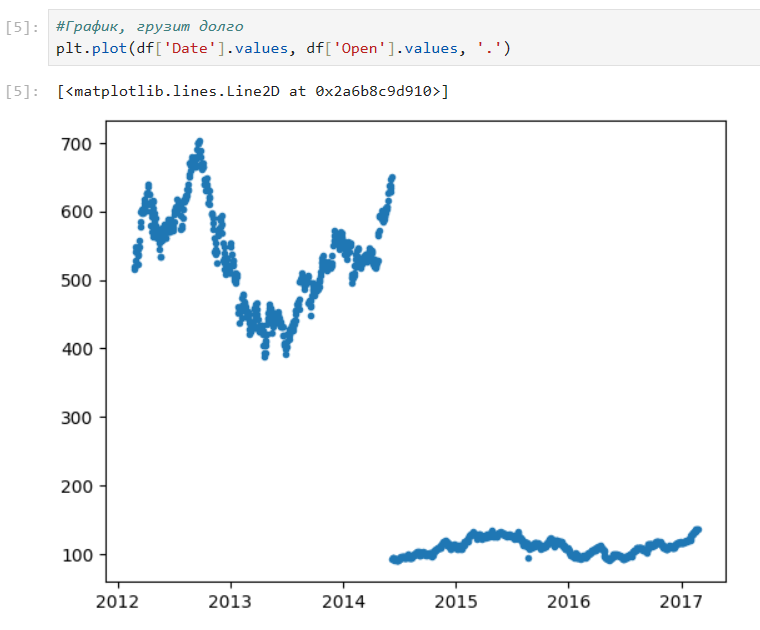


Рисунок 3 – Нарисованный благодаря библиотеке matplotlib график

Матрица диаграмм рассеивания изображается благодаря библиотеке seaborn и matplotlib.pyplot. На рисунке 4 изображен результат выполнения команд.

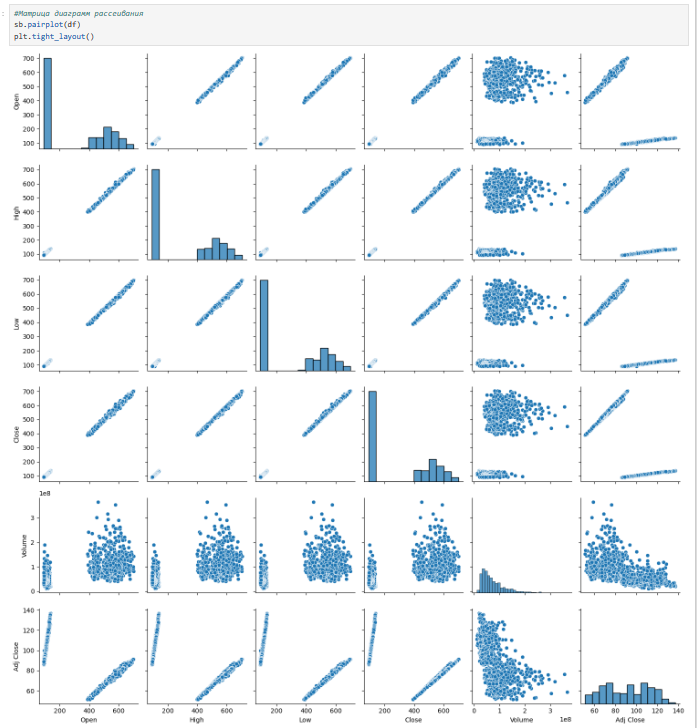


Рисунок 4 – Матрица диаграмм рассеивания.

Тепловая карта используется для наглядного анализа собранных данных и изображается с помощью команды .heatmap. На рисунке 5 изображена цветовая карта зависимости цены начала дня от даты.

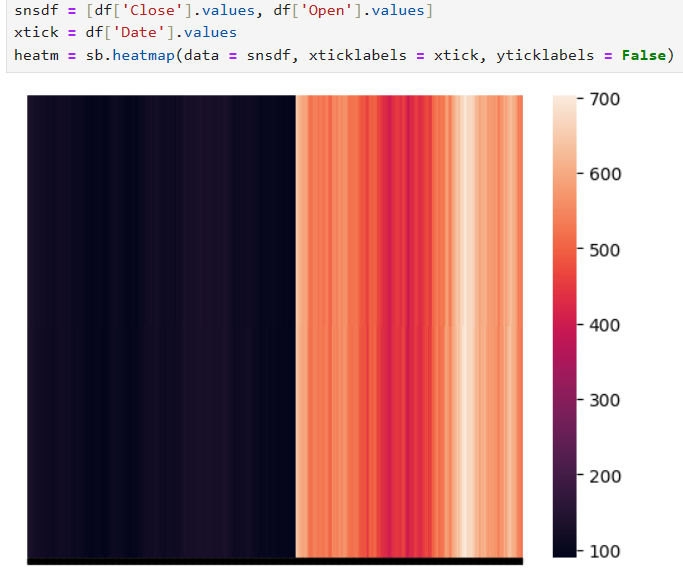


Рисунок 5 – Цветовая карта значений цены от даты.

Функция регрессии используется для определения направления разброса собранных данных. Для изображения этой функции используются алгоритмы, изображенные на рисунке 6. На рисунке 7 изображена функция линейной регрессии.

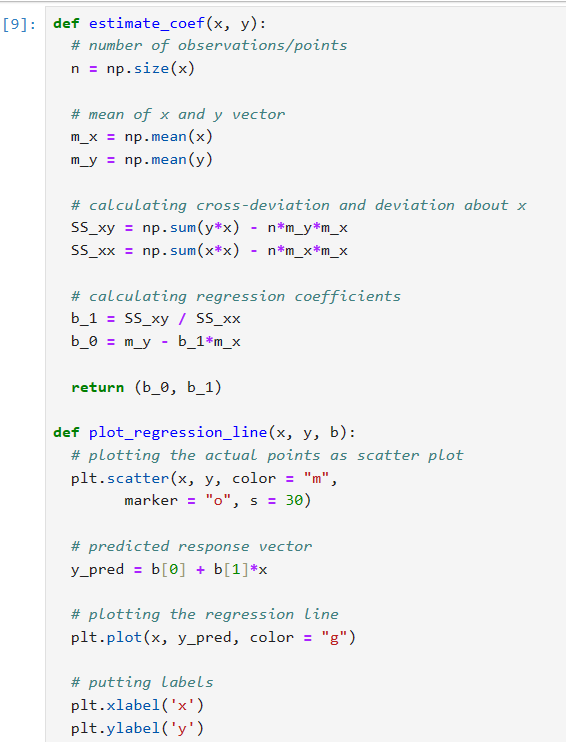


Рисунок 6 – Алгоритмы для нахождения уравнения регрессии.

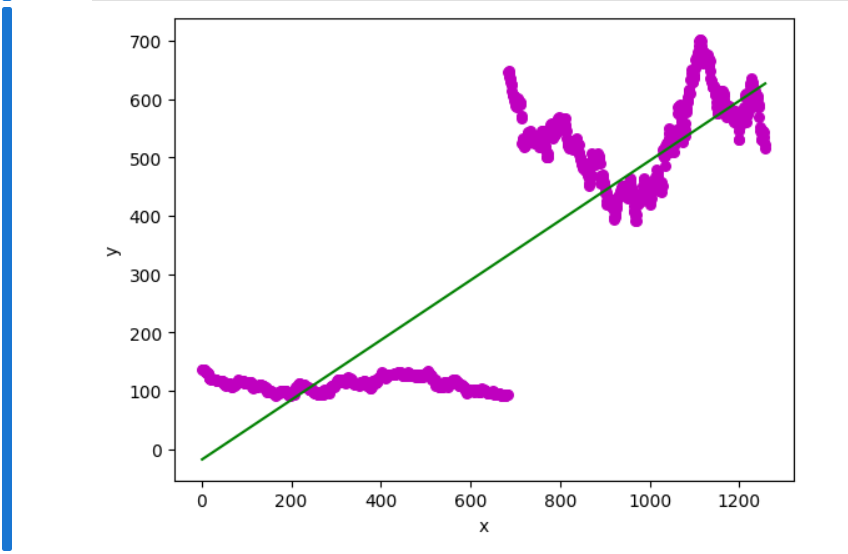


Рисунок 7 – Изображение линейной функции регрессии

Для нахождения месяца, в котором наблюдался наибольший рост курса акций на конец месяца по сравнению с началом месяца был использован следующий алгоритм:

*year\_list = []*

*for value in df['Date'].values:*

*for year in year\_list:*

*if value.year != year:*

*pass*

*else:*

*break*

*else:*

*year\_list.append(value.year)*

*month\_list = []*

*for value in df['Date'].values:*

*for month in month\_list:*

*if value.month != month:*

*pass*

*else:*

*break*

*else:*

*month\_list.append(value.month)*

*current\_difference = 0*

*max\_date = df['Date'].values[0]*

*current\_min = 0*

*current\_max = 0*

*for year in year\_list:*

*for month in month\_list:*

*day\_list = []*

*for value in df['Date'].values:*

*if value.year == year and value.month == month:*

*for day in day\_list:*

*if value.day != day:*

*pass*

*else:*

*break*

*else:*

*day\_list.append(value.day)*

*if len(day\_list) > 0:*

*i = 0*

*k = 0*

*current\_min = 0*

*current\_max = 0*

*for value in df['Date'].values:*

*if value.year == year and value.month == month:*

*if value.day == max(day\_list):*

*#print(day\_list)*

*#print(year, month)*

*current\_max = df['Close'].values[i]*

*k = i*

*if value.day == min(day\_list):*

*current\_min = df['Open'].values[i]*

*#print(current\_max)*

*#print(current\_min)*

*i = i + 1*

*if current\_max - current\_min > current\_difference:*

*current\_difference = (current\_max - current\_min)*

*max\_date = df['Date'].values[k]*

*#print(max\_date)*

*#print(current\_difference)*

*print(max\_date)*

*print(current\_difference)*

На рисунке 8 изображен результат выполнения алгоритма.

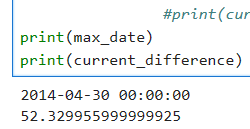


Рисунок 8 – Результат выполнения задания согласно варианту

Для восстановления значения в пропущенные дни за январь месяц 2017 года используется следующий алгоритм:

*work\_day\_list = [31, 30, 27, 26, 25, 24, 23, 20, 19, 18, 17, 16, 13, 12, 11, 10, 9, 6, 5, 4, 3, 2]*

*work\_day\_list\_chekpoints = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]*

*year = 2017*

*month = 1*

*i = 0*

*for day in work\_day\_list:*

*for value in df['Date'].values:*

*if value.year == year and value.month == month and value.day == day:*

*work\_day\_list\_chekpoints[i] = 1*

*i = i + 1*

*print(work\_day\_list\_chekpoints)*

*i = 0*

*for day in work\_day\_list\_chekpoints:*

*if day == 0:*

*information = []*

*information.append(datetime.datetime.strptime(str(year)+'-'+str(month)+'-'+str(work\_day\_list[i]), '%Y-%m-%d'))*

*k = 0*

*for value in df['Date'].values:*

*#if value.year == year and value.month == month:*

*# print(value.day, work\_day\_list[i-1])*

*if value.year == year and value.month == month and value.day == work\_day\_list[i-1]:*

*Open = (df['Open'].values[k] + df['Open'].values[k+2])/2*

*high = (df['High'].values[k] + df['High'].values[k+2])/2*

*low = (df['Low'].values[k] + df['Low'].values[k+2])/2*

*close = (df['Close'].values[k] + df['Close'].values[k+2])/2*

*volume = (df['Volume'].values[k] + df['Volume'].values[k+2])/2*

*adj\_close = (df['Adj Close'].values[k] + df['Adj Close'].values[k+2])/2*

*information.append(Open)*

*information.append(high)*

*information.append(low)*

*information.append(close)*

*information.append(volume)*

*information.append(adj\_close)*

*print(information)*

*k = k + 1*

*i = i + 1*

На рисунке 9 изображен результат выполнения алгоритма:

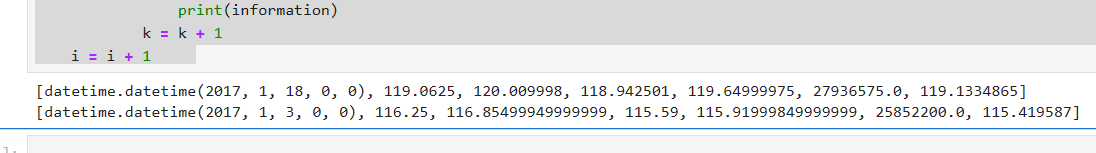


Рисунок 9 – Результат выполнения задания согласно варианту

**Вывод:** в результате выполнения лабораторной работы выполнен анализ данных в csv файле, изображены графики зависимости данных друг от друга.