**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Технологии машинного обучения»

Отчет по лабораторной работе №1

«Разведочный анализ данных. Исследование и визуализация данных»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-62Б |  | преподаватель каф. ИУ5 |
| Щепетов Дмитрий |  | Гапанюк Ю.Е. |
| Подпись и дата: |  | Подпись и дата: |

2024 г.

**Описание задания**

* Выбрать набор данных (датасет).

Для лабораторных работ не рекомендуется выбирать датасеты большого размера.

* Создать ноутбук, который содержит следующие разделы:

1. Текстовое описание выбранного Вами набора данных.
2. Основные характеристики датасета.
3. Визуальное исследование датасета.
4. Информация о корреляции признаков.

* Сформировать отчет и разместить его в своем репозитории на github.

**Текст программы**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import numpy as np

#import df

#import plotly.express as px

plt.style.use('ggplot')

pd.set\_option('display.max\_columns',24)

pd.set\_option('display.max\_colwidth',None)

pd.set\_option('display.float\_format', lambda x: '%.4f' % x)

data=pd.read\_excel(r'C:\Users\Dima\Desktop\Homework\3 курс\ОАД\НИРС\Superstore.xlsx',sheet\_name='Orders', engine='openpyxl')

data.head(3)

data.info()

data=data.drop('Row ID',axis=1)

data=data[[

'Order ID',

'Order Date',

'Ship Date',

'Ship Mode',

#'Customer ID',

#'Customer Name',

'Segment',

#'Country',

'City',

'State',

#'Postal Code',

'Region',

#'Product ID',

'Category',

'Sub-Category',

'Product Name',

'Sales',

'Quantity',

'Discount',

'Profit']]

data.head(3) # final dataframe, after columns were removed

# feature engineering, extracts specific date values from the `Order Date` column, creates new features from existing features

data['month']=data['Order Date'].dt.month

data['year']=data['Order Date'].dt.year

data['year\_month']=data['Order Date'].dt.to\_period('M')

data['total\_discount\_in\_dollars']=data['Sales'] \* data['Discount'] # discount's equivalent to dollars

data['selling\_price']=data['Sales'] / data['Quantity'] # calculates selling price for the each product

data['(net)\_profit\_before\_discount']=data['Sales'] \* data['Discount'] + data['Profit'] # net profit before deducting discount

data['order\_fulfillment\_time']=data['Ship Date'] - data['Order Date'] # interval between order placed and order shipped

data['net\_profit\_per\_unit\_sold']=data['Profit'] / data['Quantity'] # net profit generated per unit sold

data=data.rename(columns={'Profit':'net\_profit'}) # renames Profit column with net\_profit, a more specific name

data['profit\_margin']=data['net\_profit'] / data['Sales'] \* 100 # for a 25% profit margin, the company makes .25 dollars per 1 dollar sale

data['discounted\_sales']=data['Sales'] - (data['Discount']\*data['Sales']) # extracts sales accounted for discount

print('Output dataframe:')

data.head(5)

data.info()

print('The following shows descriptive statistics on numeric data. This can also reveal potential errors or anomalies with the data.')

data.describe()

sns.barplot(x=data['Segment'], y=data['Sales'])

plt.show()

plt.figure(figsize=(8,5))

Category\_df = dict(data["Category"].value\_counts())

plt.pie(Category\_df.values(),labels=Category\_df.keys(),wedgeprops={"edgecolor":"black"},autopct="%1.1f%%")

plt.tight\_layout()

plt.figure(figsize=(12,6))

sns.countplot(x=data["Ship Mode"])

plt.tight\_layout()

data.groupby('year\_month')['Sales'].sum().plot(c='#003f5c',linewidth=1,figsize=(12,6))

plt.title('Total Monthly Sales')

plt.xlabel('Month')

plt.ylabel('Total Sales')

plt.tight\_layout()

plt.show()

data.groupby('month')['Sales'].sum().plot(kind='bar',color='#1d3557',figsize=(6,4.5),width=.89)

plt.title('Aggregated Monthly Sales')

plt.xticks(ticks=np.arange(0,12,1),labels=['Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec'],rotation=0)

plt.tight\_layout()

plt.show()

month\_subcat=pd.DataFrame(data.groupby(['month','Sub-Category'])['Sales'].sum().reset\_index())

month\_subcat

plt.figure(figsize=(12,7))

sns.barplot(data=month\_subcat,\

x='Sub-Category',\

y='Sales',\

hue='month',\

palette=['#e9d8a6','#e9d8a6','#e9d8a6',\

'#e9d8a6','#e9d8a6','#e9d8a6',\

'#e9d8a6','#e9d8a6','#f77f00',\

'#e9d8a6','#d62828','#003049'])

plt.title('Montly Sub-Category Sales (Sept, Nov, Dec)')

plt.xticks(rotation=25)

plt.tight\_layout()

data.query('Segment == "Consumer"').groupby('month')['Sales'].sum().plot(kind='bar',\

figsize=(6,4.5),\

width=.89,\

color='#1d3557')

plt.title('Consumer Segment Aggregated Monthly Sales')

plt.show()

plt.figure(figsize=(12,8))

plt.subplot(211)

data.groupby('year\_month')['Sales'].mean().plot(linewidth=1.5,color='#d62828')

plt.title('Average Monthly Sales')

plt.subplot(212)

data.groupby('year\_month')['Sales'].mean().plot(linewidth=1.5,color='#d62828')

data.groupby('year\_month')['Sales'].describe()['std'].plot(linewidth=1.5,color='#033270')

plt.title('Monthly Sales (Average & Std)')

plt.legend(['Average','Standard Deviation'])

plt.tight\_layout()

plt.show()

df\_sales=pd.DataFrame(data.groupby(['Category','Sub-Category'])['Sales'].sum()).reset\_index().sort\_values('Sales',ascending=False)

sns.barplot(x='Sales',y='Sub-Category',data=df\_sales,hue='Category',palette=['#003049','#d62828','#f77f00'])

plt.title('Total Sales per Sub-Category')

plt.show()

yearly\_sales=pd.DataFrame(data.groupby(['Sub-Category','year'])['Sales'].sum()).reset\_index()

yearly\_sales

plt.figure(figsize=(12,5))

sns.barplot(data=yearly\_sales,x='Sub-Category',y='Sales',hue='year',palette=['#177e89','#db3a34','#ffc857','#084c61'])

plt.xticks(rotation=50)

plt.title('Yearly Sales per Sub-category')

plt.tight\_layout()

plt.show()

yearly\_sales['yearly\_growth\_rate'] = yearly\_sales.groupby('Sub-Category')['Sales'].pct\_change() \* 100

print('Sales Annual Average Growth Rate:')

pd.DataFrame(yearly\_sales.groupby('Sub-Category')['yearly\_growth\_rate'].mean().sort\_values(ascending=False))

df3 = data

columns\_to\_remove = df3.select\_dtypes(['period[M]', 'timedelta64[ns]', 'object']).columns

df3 = df3.drop(columns=columns\_to\_remove)

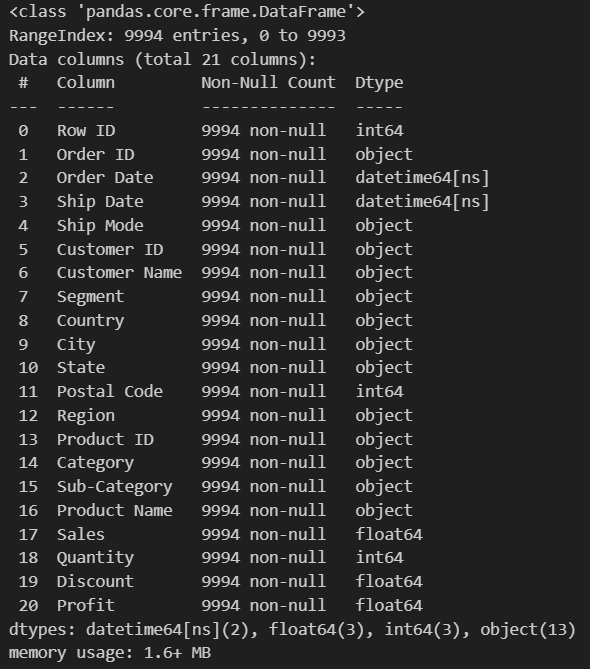
plt.figure(figsize=(15,6))

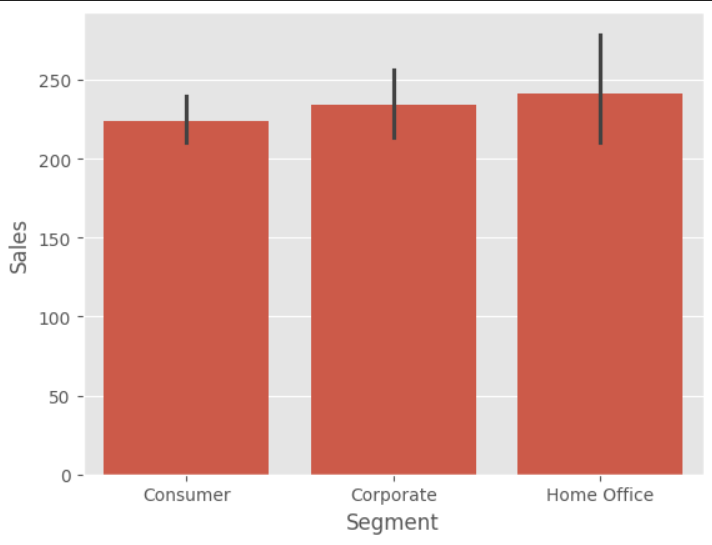
sns.heatmap(df3.corr(), annot=True)

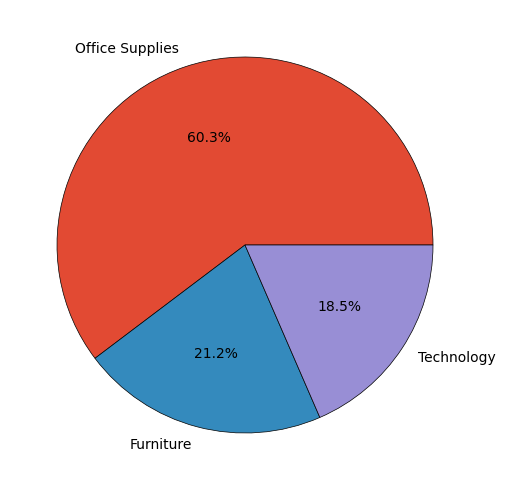
plt.show()

**Экранные формы с примерами выполнения программы**

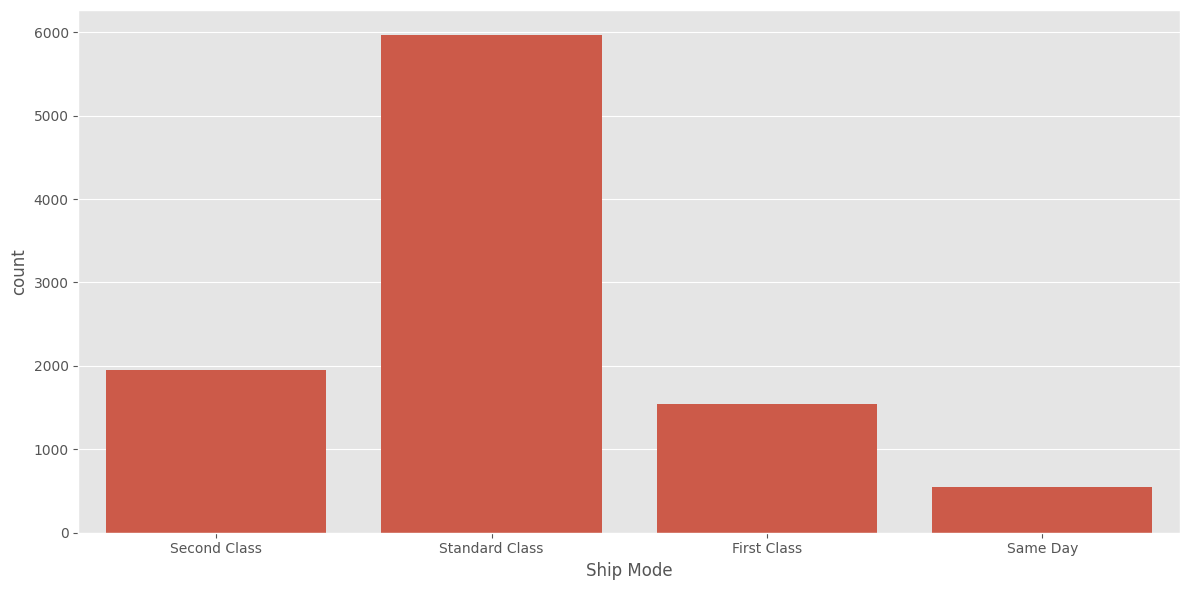
Просмотр информации о датасете



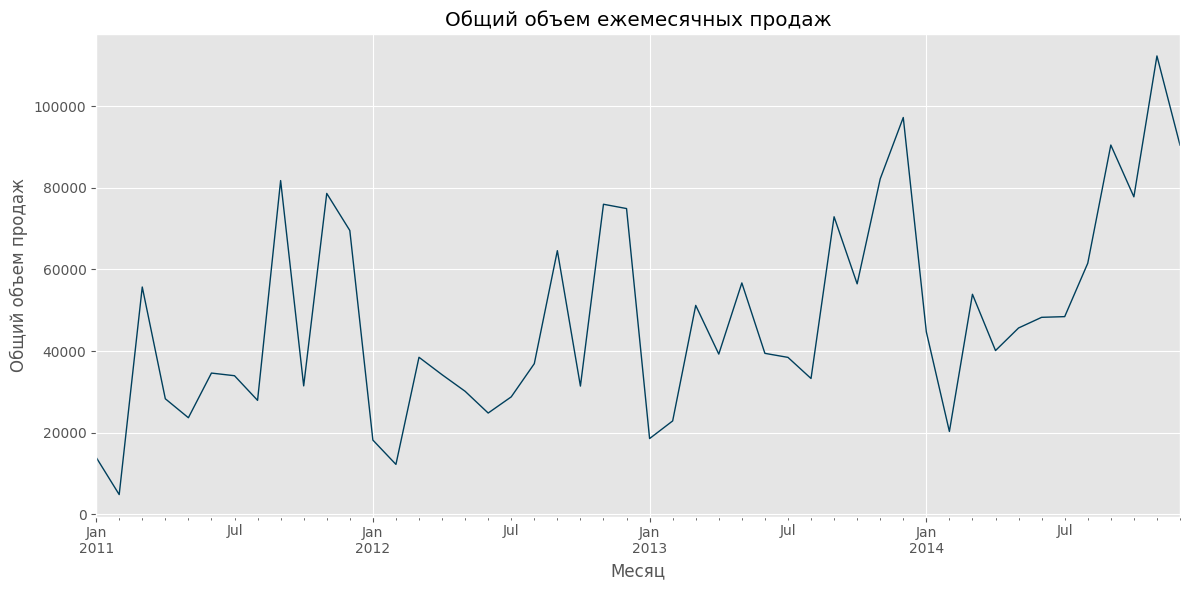
Столбчатая диаграмма (средние продажи по сегментам)

Круговая диаграмма на основе данных категорий товаров 

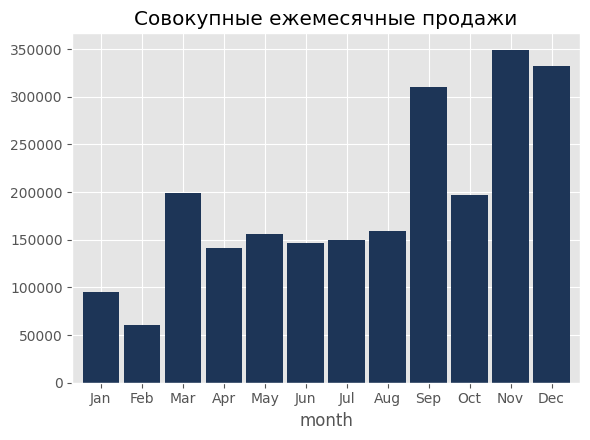
Столбчатая диаграмма для подсчета количества записей в зависимости от режима доставки товаров



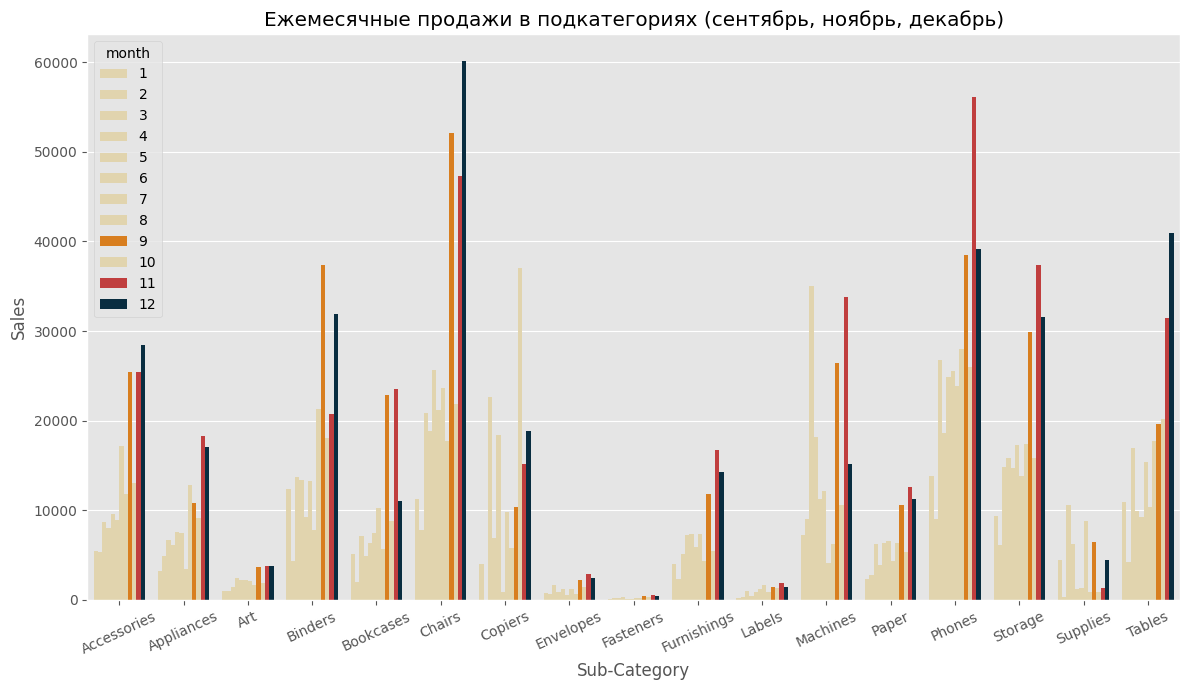
Линейный график для отображения ежемесячной суммы продаж по годам



Столбчатая диаграмма, показывающая совокупные ежемесячные продажи



Столбчатая диаграмма, показывающая ежемесячные продажи по подкатегориям товаров, при этом каждый столбец представляет собой отдельную подкатегорию, а разные месяцы выделены разными цветами



Столбчатая диаграмма, показывающая агрегированные ежемесячные продажи в сегменте "Consumer"



Корреляционная матрица

