|  |
| --- |
|  |
| 1. МИНОБРНАУКИ РОССИИ |
| 1. Федеральное государственное бюджетное образовательное учреждение 2. высшего образования 3. «МИРЭА - Российский технологический университет» 4. РТУ МИРЭА |

|  |  |
| --- | --- |
| ДИСЦИЛИНА | **Системы искусственного интеллекта и большие данные** |
|  | (полное наименование дисциплины без сокращений) |
| ИНСТИТУТ | **Искусственного интеллекта** |
| КАФЕДРА | **Технологии искусственного интеллекта** |
|  | полное наименование кафедры |
| ВИД УЧЕБНОГО | **Практические занятия** |
| МАТЕРИАЛА | (в соответствии с пп.1-11) |
| ПРЕПОДАВАТЕЛЬ | **Питинов Артем Вадимович** |
|  | (фамилия, имя, отчество) |
| СЕМЕСТР | **3 семестр 2025** |
|  | (указать семестр обучения, учебный год) |

Основы Python для обработки данных. Установка и настройка среды (Jupyter, Colab, VSCode). Загрузка данных из CSV/Excel. Источники экономических данных (макро: Росстат, ЦБ, МВФ; микро: CRM, транзакции, опросы, открытые данные). Загрузка данных Загрузка данных Росстата, ЦБ РФ, биржевых котировок. (CSV, Excel). Введение в NumPy. Индексация и срезы в многомерных массивах. Операции над данными в библиотеке Pandas. Базовые операции с DataFrame: выборка данных по условиям, добавление и удаление столбцов.Пример: расчет доходности портфеля активов; вычисление корреляционной матрицы для финансовых инструментов.

Ссылка на блокнот

<https://colab.research.google.com/drive/1J-GrZr3dUUGDu0fDMYYprKJg1O89LPwZ?usp=sharing>

# \*\*0. Установка и запуск среды\*\*

🎯 Выберите любой из трёх вариантов.

a) Google Colab

1. Зайдите на https://colab.research.google.com и создайте новый notebook.

2. У Colab уже предустановлены pandas, numpy, openpyxl (для Excel).

b) Jupyter Notebook (Anaconda)

1. Скачайте и установите Anaconda Distribution.

2. Запустите Anaconda Navigator → Jupyter Notebook.

3. Создайте notebook и, при необходимости, установите недостающие пакеты:

conda install pandas openpyxl xlrd

c) VS Code + Jupyter extension

1. Установите VS Code и расширение \*Jupyter\*.

2. Откройте .ipynb-файл — появится интерактивный интерфейс.

3. Через терминал выполните:

pip install pandas openpyxl xlrd

Где можно брать данные для заданий?

\* 🛠️ Соревнования Kaggle](https://www.kaggle.com/datasets)

\* 🛠️ Google Datasets](https://datasetsearch.research.google.com/)

\* 🎓 Сайт huggingface](https://huggingface.co/datasets)

\* 🛠️ Федеральная служба государственной статистики](https://rosstat.gov.ru/folder/10705)

\* 🎓 ЕМИСС Государственная статистика](https://fedstat.ru/)

Инструменты

Будем использовать библиотеки:

\* [[doc] 🛠️ NumPy](https://numpy.org/) — поддержка больших многомерных массивов и быстрых математических функций для операций с этими массивами.

\* [[doc] 🛠️ Scikit-learn](https://scikit-learn.org/stable/) — ML алгоритмы.

\* [[doc] 🛠️ Pandas](https://pandas.pydata.org/) — удобная работа с табличными данными.

\* [[doc] 🛠️ PyTorch](https://pytorch.org/) —  фреймворк машинного обучения.

\* [[doc] 🛠️ Matplotlib](https://matplotlib.org/) —  библиотека для визуализации. Вывод различных графиков.

\* [[doc] 🛠️ Seaborn](https://seaborn.pydata.org/) — еще одна библиотека для визуализации .

#Если работаете в локальном Jupyter/VS Code и модулей нет — раскомментируйте

#!pip install pandas openpyxl xlrd -q

import pandas as pd

# ****1. Загрузка тренировочного набора данных****

**Обзор доступных данных**

Датасет: IBM HR Analytics Employee Attrition & Performance Описание: Данные о сотрудниках компании IBM, включая признаки увольнения (Attrition), удовлетворенности, условий труда, зарплаты и др. Актуальность для инноватики: высокая текучесть кадров снижает эффективность инновационных проектов.

# Загрузка архива с Google Drive

!gdown 1ssiVlJiAYRfkpnVtQwg-UE253RvrrdDx -O IBM\_HR\_dataset.zip

# Распаковка

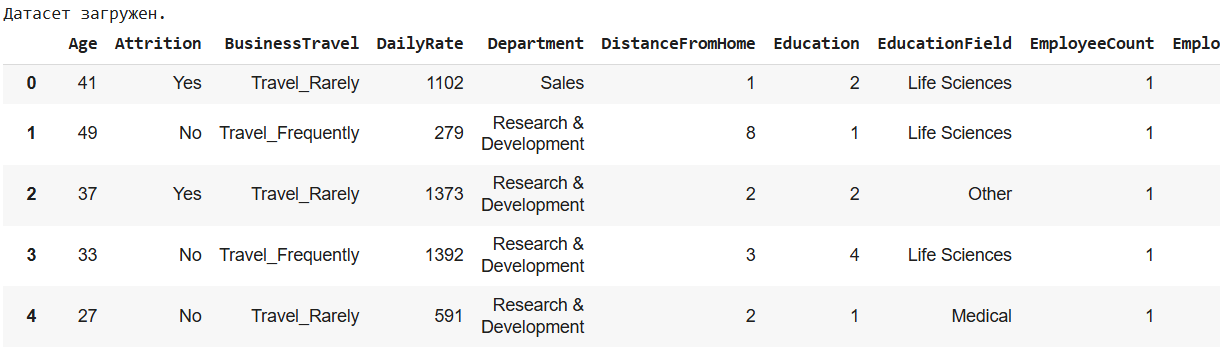
!unzip IBM\_HR\_dataset.zip

# Загрузка данных в DataFrame

df = pd.read\_csv('WA\_Fn-UseC\_-HR-Employee-Attrition.csv')  # Убедитесь, что имя файла верное

print("Датасет загружен.")

df



# ****2. Первичный обзор структуры данных****

Методы, которые мы изучим:

• shape — размеры ((строки, столбцы))

• columns — список имён столбцов

• head(n) / tail(n) — первые / последние n строк

• sample(n) — случайные n строк

• info() — типы данных, пропуски, память

• describe() — базовые статистики столбцов

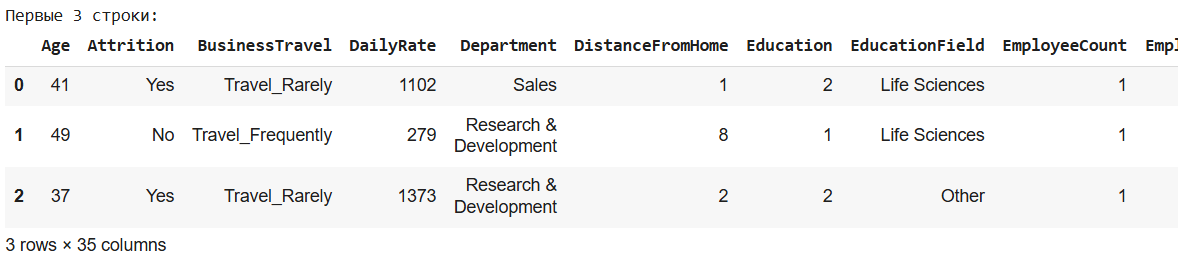
print("Размер датасета:", df.shape)

print(" Заголовки столбцов:")

print(df.columns.tolist())

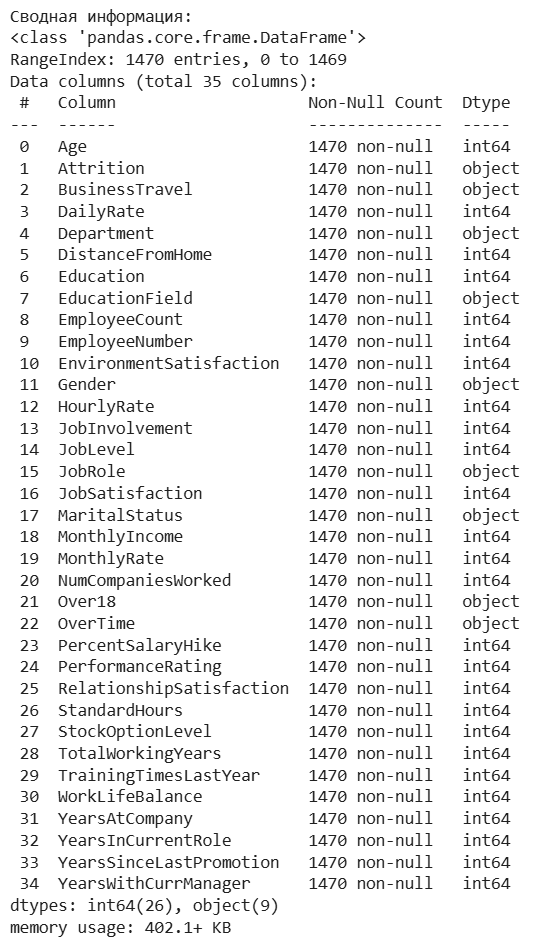
print("Первые 3 строки:")

df.head(3)



print("Сводная информация:")

df.info()



# Выведем пять последних строк (tail).

df.tail()

#Выведем 3 случайных столбца  из этой выборки (sample).

df.sample(3)

Введение в NumPy

NumPy представляет собой библиотеку нацеленную на работу с многомерными массивами и матричными структурами данных. Также она может быть задействована для выполнения различных математических операций с массивами.

import numpy as np

mass1 = np.array([1, 2, 3], int)

mass2 = np.array([4, 5, 6], int)

print(mass1[1])

print(mass1 + mass2)

print(mass1 - mass2)

print(mass1 \* mass2)

print(mass1 / mass2)

print(mass1 % mass2)

print(mass1 \*\* mass2)

# Пример инициализации двумерного массива:

mass3 = np.array([[1, 2], [3, 4]], float)

mass4 = np.array([[5, 6], [7, 8]], float)

print(mass3 \* mass4)

mass5 = np.array([[1, 2], [3, 4], [5, 6]], float)

mass6 = np.array([-1, 3], float)

print(mass5 \* mass6)

Базовые операции над массивами

mass7 = np.array([2, 4, 5], float)

print ('СУмма = ' + str(mass7.sum()))

print ('Результат перемножения = ' + str(mass7.prod()))

print ('Среднее значение арифметическое = ' + str(mass7.mean()))

print ('Минимальное = ' + str(mass7.min()))

print ('Индекс Минимального = ' + str(mass7.argmin()))

mass8 = np.array([7, 4, 5, 1], float)

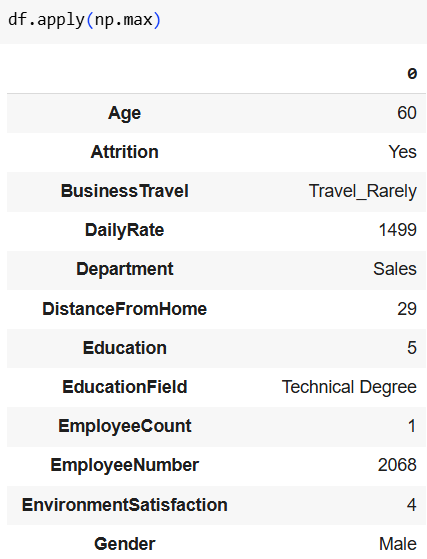
print (sorted(mass8))

mass9 = np.array([[1, 2], [3, 4], [5, 6]], float)

for x in mass9:

  print(x)

df.apply(np.max)



# ****3. Выборка строк и столбцов в датафрейме****

Базовые способы:

1. **По имени столбца** - df['col'] (например, df['housing\_median\_age'])
2. **Срез строк по индексу** - df[5:10] (это строки 5-9 )
3. **loc (метка)** - df.loc[строки, столбцы] (например, df.loc[0:4, 'housing\_median\_age'] )
4. **iloc (позиция)** - df.iloc[строки, столбцы] (например, df.iloc[:5, :3] )

# Сотрудники с высокой удовлетворённостью (4 из 4)

high\_sat = df[df['JobSatisfaction'] == 4]

print(f"Сотрудников с высокой удовлетворённостью: {len(high\_sat)}")

# Уволившиеся, несмотря на высокую удовлетворённость

attrited\_high\_sat = df[(df['Attrition'] == 'Yes') & (df['JobSatisfaction'] == 4)]

print(f"Уволились, несмотря на высокую удовлетворённость: {len(attrited\_high\_sat)}")

# Сотрудники с высоким доходом и малым стажем

high\_income\_new = df[(df['MonthlyIncome'] > df['MonthlyIncome'].quantile(0.75)) & (df['YearsAtCompany'] < 2)]

print(f"Высокооплачиваемые новички: {len(high\_income\_new)}")

Добавление и удаление столбцов

# Добавим признак: "молодой сотрудник" (до 30 лет)

df['YoungEmployee'] = df['Age'] < 30

df[['Age', 'YoungEmployee']].head(5)

# Добавим столбец: "потенциальный лидер" — высокая вовлечённость + высокий доход

df['PotentialLeader'] = (df['JobInvolvement'] == 4) & (df['MonthlyIncome'] > df['MonthlyIncome'].median())

df[['JobInvolvement', 'MonthlyIncome', 'PotentialLeader']].head(5)

# Переименование столбца

df.rename(columns={'PotentialLeader': 'HighPotential'}, inplace=True)

df = df.drop(columns=['HighPotential'])

df

Получение строк, удовлетворяющих определенному условию

# Фильтрация: сотрудники с высоким уровнем стресса (долго работают)

print("Сотрудники, работающие сверхурочно:")

overtime\_workers = df[df['OverTime'] == 'Yes']

overtime\_workers[['EmployeeNumber', 'Age', 'Department', 'MonthlyIncome']].head()

# Преобразуем данные в массивы NumPy

satisfaction = df['JobSatisfaction'].values  # 1–4

involvement = df['JobInvolvement'].values   # 1–4

performance = df['PerformanceRating'].values # 1–4

# Расчёт корреляции Пирсона

corr\_si = np.corrcoef(satisfaction, involvement)[0, 1]

corr\_sp = np.corrcoef(satisfaction, performance)[0, 1]

print(f"Корреляция: Удовлетворённость ↔ Вовлечённость: {corr\_si:.3f}")

print(f"Корреляция: Удовлетворённость ↔ Эффективность: {corr\_sp:.3f}")

# Выбор ключевых числовых переменных

hr\_numeric = df[['Age', 'MonthlyIncome', 'JobSatisfaction', 'JobInvolvement',

                 'WorkLifeBalance', 'YearsAtCompany', 'PerformanceRating']]

# Выбор ключевых числовых переменных

hr\_numeric = df[['Age', 'MonthlyIncome', 'JobSatisfaction', 'JobInvolvement',

                 'WorkLifeBalance', 'YearsAtCompany', 'PerformanceRating']]

# Расчёт корреляционной матрицы

corr\_matrix = hr\_numeric.corr().round(2)

# Вывод матрицы

print("Корреляционная матрица ключевых HR-показателей:")

corr\_matrix

# Визуализация — тепловая карта

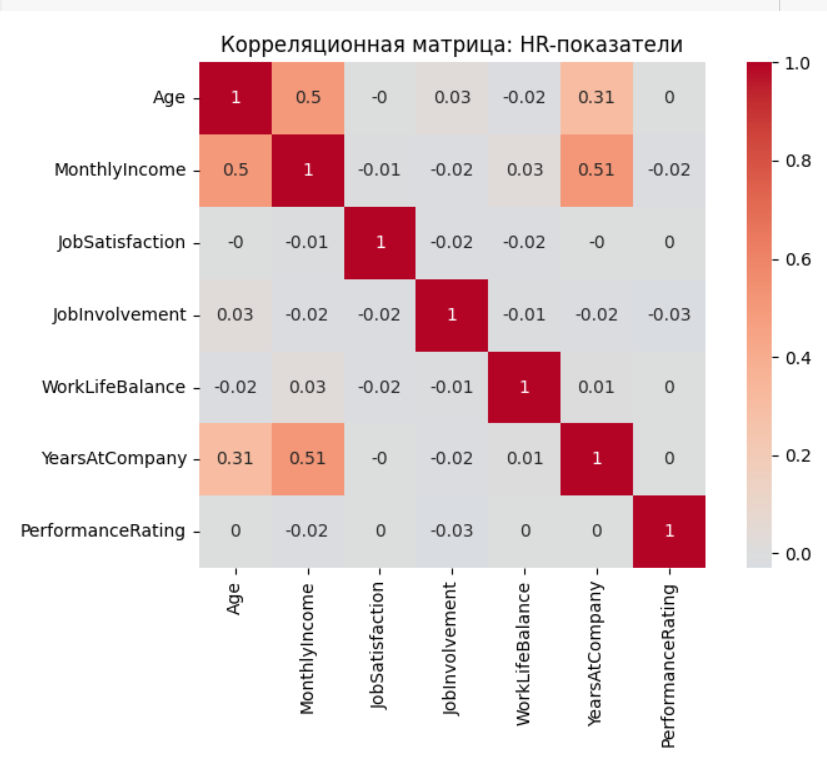
plt.figure(figsize=(8, 6))

sns.heatmap(corr\_matrix, annot=True, cmap='coolwarm', center=0, square=True)

plt.title('Корреляционная матрица: HR-показатели')

plt.tight\_layout()

plt.show()



# Уволившиеся сотрудники

print("Уволившиеся (Attrition = 'Yes'):")

attrited = df[df['Attrition'] == 'Yes']

attrited[['Age', 'JobRole', 'MonthlyIncome', 'YearsAtCompany']].head()

print(df[df['Education'] > 2])

# Группировка по отделу

hr\_summary = df.groupby('Department').agg(

    AvgIncome=('MonthlyIncome', 'mean'),

    AttritionRate=('Attrition', lambda x: (x == 'Yes').mean()),

    Count=('EmployeeNumber', 'size')

).round(2)

print("Сводка по отделам:")

hr\_summary

# ****4. Сохранение результата****

**Мы хотим выгрузить обработанный датафрейм в:**

• CSV (to\_csv)

• Excel (to\_excel)

**#Без индекса в CSV**

# Сохранение в CSV без индекса

df.to\_csv('hr\_processed.csv', index=False)

print("Сохранено как 'hr\_processed.csv'")

# Сохранение в Excel

df.to\_excel('ibm\_hr\_processed.xlsx', index=False)

print("Данные сохранены в 'ibm\_hr\_processed.xlsx'")

# Сохранение в Excel

df.to\_excel('hr\_processed.xlsx', index=False)

print("Сохранено как 'hr\_processed.xlsx'")

**Задание**

1. (анализ увольнений) Создайте датасет high\_performers\_attrition.csv, включающий только сотрудников с высокой производительностью (PerformanceRating == 4) и уволившихся (Attrition == 'Yes'). Сохраните его.
2. Откройте файл high\_performers\_attrition.csv и добавьте столбец CareerStagnation, если сотрудник проработал в текущей должности более 3 лет (YearsInCurrentRole > 3).