МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г. ЧЕРНЫШЕВСКОГО»

Кафедра дифференциальных уравнений и математической экономики

ОТЧЕТ ПО ПРОИЗВОДСТВЕННОЙ (БАЗОВОЙ) ПРАКТИКЕ

студента 4 курса 451 группы
направления 38.03.05 – Бизнес-информатика

механико-математического факультета
Ковина Семёна Дмитриевича

Место прохождения: АО “Неофлекс Консалтинг”
Сроки прохождения: с 29.06.2021 г. по 26.07.2021 г.
Оценка:

Руководитель практики от СГУ

профессор, д.ф.-м.н., доцент _____ А.Ю. Трынин

Руководитель практики от организации

руководитель направления по подготовке молодых специалистов филиала _____ А.К. Кузьмин

Саратов 2021

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Тема производственной базовой практики является: "Проведение аналитики утечки кадров "Neoflex".

Целью производственной практики являлось углубление и закрепление знаний, полученных в процессе обучения и приобретение необходимых навыков по реализации полученных знаний в трудовой деятельности с использованием современных программно-аппаратных средств в составе производственного коллектива "Neoflex", отделе "Бизнес-направление Data Science".

Организация "Neoflex" предоставила необходимые условия для прохождения практики. Была проведена предварительная встреча с компанией, в которой было рассказано о самой компании и ее деятельности.

Основной вид деятельности отдела "Бизнес-направление Data Science" состоит в разработке и внедрении информационных систем.

При прохождении практики передо мной была поставлена задача:

По следующим разбивкам:

- грейды
- филиалы
- подразделения
- стаж работы в компании

провести аналитику и построить графики текучки кадров (% уволившихся сотрудников по кварталам) и среднего срока работы в компании.

Работа выполнялась на языке Python с использованием библиотек Numpy, Pandas и Matplotlib.

При решении поставленных задач и при подготовке отчета были использованы источники [1-5].

1 Аналитические специальности в сфере IT

Аналитик – это собирательное название профессии, суть которой сводится к сбору большого количества цифровых данных, их анализу и трактовке полученной информации. Что это за данные – зависит уже от специфики работы такого сотрудника. Аналитик может быть специалистом в области финансов, инвестиций, конкретных рыночных сегментов, инженерии, химии, компьютерного программного обеспечения, рекламы, социологии и так далее.

Существуют множество профессий в IT аналитике, которые тесно связаны между собой, тяжело не запутаться в них. В соответствии с рисунком 1 представлена диаграмма Эйлера соотношений областей знаний по некоторым профессиям:

Рисунок 1 – области знаний различных IT профессий

Далее в качестве примеров приведены основные специальности с краткими описаниями:

Бизнес-аналитика — это процесс обнаружения, интерпретации и информирования о найденных закономерностях в данных, а также использование средств, которые помогают компании анализировать любые данные в любой среде и на любых устройствах. Бизнес-аналитика предлагает и дополнительные возможности для достижения желаемых результатов, такие как оптимизация, снижение затрат и взаимодействие с заказчиками.

Data Science – междисциплинарная область, которая охватывает практически все, что связано с данными: от их подготовки до очистки и анализа. Data Science использует научные методы и алгоритмы для работы как со структурированными, так и с неструктурированными данными. Эта область сочетает в себе статистику, математику, машинное обучение, решение проблем и многое другое.

Big Data – область, в которой рассматриваются различные способы анализа и систематического извлечения больших объемов данных. Эта специальность включает применение механических или алгоритмических процессов получения оперативной информации для решения сложных бизнес-задач. Специалисты по “Большим Данным” работают с сырыми неструктурированными данными, результаты анализа которых используются для поддержки принятия решений в бизнесе.

Аналитика больших данных включает проверку, преобразование, очистку и моделирование данных.

Аналитика данных — это процесс извлечения из необработанной информации важных данных, имеющих практическое значение. Для компаний это означает применение правильных методов анализа для выявления скрытых закономерностей и тенденций, которые можно использовать для улучшения своего бизнеса.

Остановимся на этой специальности, так как она ближе всего к теме производственной практики. Далее она описана более подробно.

1.1 Анализ данных

Итак, аналитика данных — это процесс преобразования первичных данных в полезные знания, которые можно использовать.

Аналитик данных выполняет следующие задачи:

- собирает данные (формирует запрос сам или получает задачу от менеджеров);
- знакомится с параметрами набора (какие типы данных собраны, как их можно отсортировать);
- проводит предварительную обработку (очищает от ошибок и повторов, упорядочивает);
- интерпретирует (анализирует, собственно решает задачу);
- делает вывод;
- визуализирует (так, чтобы на основе вывода можно было принять решение, подтвердить или опровергнуть гипотезу).

В соответствии с рисунком 2 представлены необходимые знания и навыки для специалиста по аналитике данных:

Рисунок 2 – области знаний аналитика данных

2 Python и его библиотеки

2.1 Высокоуровневый язык программирования Python

Python (в русском языке встречаются названия питóн или пáйтон) — высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ. Язык является полностью объектно-ориентированным — всё является объектами.[1] Необычной особенностью языка является выделение блоков кода пробельными отступами. Синтаксис ядра языка минималистичен, за счёт чего на практике редко возникает необходимость обращаться к документации. Сам же язык известен как интерпретируемый и используется в том числе для написания скриптов. Недостатками языка являются зачастую более низкая скорость работы и более высокое потребление памяти написанных на нём программ по сравнению с аналогичным кодом, написанным на компилируемых языках, таких как Си или C++.

Стандартная библиотека включает большой набор полезных переносимых функций, начиная от функционала для работы с текстом и заканчивая средствами для написания сетевых приложений. Дополнительные возможности, такие как математическое моделирование, работа с оборудованием, написание веб-приложений или разработка игр, могут реализовываться посредством обширного количества сторонних библиотек, а также интеграцией библиотек, написанных на Си или C++, при этом и сам интерпретатор Python может интегрироваться в проекты, написанные на этих языках.

Python стал одним из самых популярных языков, он используется в анализе данных, машинном обучении, DevOps и веб-разработке, а также в других сферах, включая разработку игр.

2.2 Библиотека Numpy

Предок NumPy, Numeric, был разработан Джимом Хугунином. Также был создан пакет Numarray с дополнительной функциональностью. В 2005 году Трэвис Олифант выпустил пакет NumPy, добавив особенности Numarray в Numeric.[2] Это проект с исходным кодом, и в его развитии поучаствовало уже много человек.

NumPy или Numerical Python — это библиотека Python, которая предлагает следующее:

- Мощный N-мерный массив,
- Высокоуровневые функции,
- Инструменты для интеграции кода C/C++ и Fortran,
- Использование линейной алгебры, Преобразований Фурье и возможностей случайных чисел.

Она также предлагает эффективный многомерный контейнер общих данных. С ее помощью можно определять произвольные типы данных.

Основная часть функционала Numpy описана в приложении А.

2.3 Библиотека Pandas

Pandas — программная библиотека на языке Python для обработки и анализа данных. Работа pandas с данными строится поверх библиотеки NumPy, являющейся инструментом более низкого уровня.[3] Предоставляет специальные структуры данных и операции для манипулирования числовыми таблицами и временными рядами. Название библиотеки происходит от эконометрического термина «панельные данные», используемого для описания многомерных структурированных наборов информации.

Основная область применения — обеспечение работы в рамках среды Python не только для сбора и очистки данных, но для задач анализа и моделирования данных, без переключения на более специфичные для статобработки языки (такие, как R и Octave).

Также активно ведётся работа по реализации «родных» категориальных типов данных.

Пакет прежде всего предназначен для очистки и первичной оценки данных по общим показателям, например среднему значению, квантилям и так далее; статистическим пакетом он в полном смысле не является, однако наборы данных типов DataFrame и Series[4] применяются в качестве входных в большинстве модулей анализа данных и машинного обучения.

Основные возможности библиотеки:

- Объект *DataFrame* для манипулирования индексированными массивами двумерных данных
- Инструменты для обмена данными между структурами в памяти и файлами различных форматов
- Встроенные средства совмещения данных и способы обработки отсутствующей информации
- Переформатирование наборов данных, в том числе создание сводных таблиц
- Срез данных по значениям индекса, расширенные возможности индексирования, выборка из больших наборов данных
- Вставка и удаление столбцов данных
- Возможности группировки позволяют выполнять трёхэтапные операции типа «разделение, изменение, объединение»
- Слияние и объединение наборов данных
- Иерархическое индексирование позволяет работать с данными высокой размерности в структурах меньшей размерности
- Работа с временными рядами: формирование временных периодов и изменение интервалов.

Примеры работы функционала Pandas приведены в приложении В.

2.4 Библиотека Matplotlib

Matplotlib — библиотека на языке программирования Python для визуализации данных двумерной (2D) графикой (3D графика также поддерживается). Получаемые изображения могут быть использованы в качестве иллюстраций в публикациях.

Matplotlib написан и поддерживался в основном Джоном Хантером (англ. John Hunter) и распространяется на условиях BSD-подобной лицензии. Генерируемые в различных форматах изображения могут быть использованы в интерактивной графике, в научных публикациях, графическом интерфейсе пользователя, веб-приложениях, где требуется построение диаграмм (англ. plotting). В документации автор признаётся, что Matplotlib начинался с подражания графическим командам MATLAB, но является независимым от него проектом.[5]

Версия 2.1.1 — последняя стабильная — требует Python версии 2.7 или от 3.4 и выше и версию NumPy от 1.7.1 и выше.

Библиотека Matplotlib построена на принципах ООП, но имеет процедурный интерфейс pylab, который предоставляет аналоги команд MATLAB.

Matplotlib является гибким, легко конфигурируемым пакетом, который вместе с NumPy, SciPy и IPython предоставляет возможности, подобные MATLAB. В настоящее время пакет работает с несколькими графическими библиотеками, включая wxWindows и PyGTK.

Пакет поддерживает многие виды графиков и диаграмм:

- Графики (line plot)
- Диаграммы разброса (scatter plot)
- Столбчатые диаграммы (bar chart) и гистограммы (histogram)
- Круговые диаграммы (pie chart)
- Ствол-лист диаграммы (stem plot)
- Контурные графики (contour plot)

- Поля градиентов (quiver)
- Спектральные диаграммы (spectrogram)

Пользователь может указать оси координат, решетку, добавить надписи и пояснения, использовать логарифмическую шкалу или полярные координаты.

Несложные трёхмерные графики можно строить с помощью набора инструментов (toolkit) mplot3d. Есть и другие наборы инструментов: для картографии, для работы с Excel, утилиты для GTK и другие.

С помощью Matplotlib можно делать и анимированные изображения.

Набор поддерживаемых форматов изображений, векторных и растровых, можно получить из словаря FigureCanvasBase.filetypes. Типичные поддерживаемые форматы:

- Encapsulated PostScript (EPS)
- Enhanced Metafile (EMF)
- JPEG
- PDF
- PNG
- Postscript
- RGBA («сырой» формат)
- SVG
- SVGZ
- TIFF

Пример построения графика и функции Matplotlib приведены в приложении С.

3 Практическая часть

3.1 Знакомство с наборами данных

В качестве производственной практики мне была дана задача провести аналитику текучки кадров (% уволившихся сотрудников) и нарисовать графики по годам(кварталам), в разбивке:

1) По грейдам
2) По филиалам
3) По подразделениям (БН)
4) По стажу работы в компании

Задание выполнялось на платформе Kaggle (приложение D).

Для реализации задачи было предоставлено 3 файла:

1) Employees – данные о сотрудниках, включая Табельный номер, Сотрудник(ФИО зашифровано в виде id), Дата рождения, Дата приема, Дата увольнения и Пол(в соответствии с таблицей 1).

Табельный номер	Сотрудник	Дата рождения	Дата приема	Дата увольнения	Физическое лицо.Пол
	06c8118e2c3726b66efb4b8d113e2635	15.03.1976	21.05.2009	30.06.2009	Мужской
6	06c8118e2c3726b66efb4b8d113e2635	15.03.1976	01.07.2009	28.06.2019	Мужской
	06c8118e2c3726b66efb4b8d113e2635	15.03.1976	01.07.2009		Мужской
7	4e24fb1750996e0e8f91861df9ff65e7	01.05.1979	02.07.2009		Женский
	4e24fb1750996e0e8f91861df9ff65e7	01.05.1979	02.07.2009		Женский
	80af072e4e86c60e06fede7a8de6fb79	11.04.1969	14.07.2009	14.09.2012	Мужской
11	23ec45eb079c9ef3fa719b144067ac66	01.06.1960	14.07.2009	29.11.2019	Мужской
	23ec45eb079c9ef3fa719b144067ac66	01.06.1960	14.07.2009		Мужской
	5e7b2efe12281cee893cb63580a64c77	20.10.1975	17.07.2009	07.04.2011	Мужской
	b97a0a84585008373abe3b888e86b359	09.12.1983	21.07.2009	25.10.2013	Мужской
	2d77f813f4db00eef993aca0642a13f0	05.05.1984	21.07.2009	20.03.2014	Мужской
	198643fbcec1dadd44bba2d09d80df71	02.05.1983	21.07.2009	27.02.2015	Мужской
	2ee0b7faeafbd6c0eac0577b7d819469	19.11.1984	21.07.2009	29.05.2015	Женский
20	cc2e17ac3e5f8bd371e26a70c0ddc88f	24.11.1983	22.07.2009		Мужской
	cc2e17ac3e5f8bd371e26a70c0ddc88f	24.11.1983	22.07.2009		Мужской
	35e9db6439531c57043e45beee0d2cdc	12.02.1985	28.07.2009	14.01.2011	Мужской
	0573f5a2f09a64ac8cebabf50af4e8b8	29.03.1978	28.07.2009	18.11.2016	Мужской
	944bb5ee61fdf3abc29020b8da5c3f79	11.11.1985	30.07.2009	16.01.2015	Мужской
	1f9e8f1c2348efebe8513c5672e22872	18.11.1987	04.08.2009	15.01.2013	Мужской
	a0bc671ff2f46bc854a410f7214c338b	25.09.1986	20.08.2009	28.01.2015	Женский
	48f4f0cc2e170758fd1840fabf3953ea	23.06.1984	15.09.2009	10.01.2014	Мужской
40	e1e061c61daee1d518a9d8181e64c196	19.10.1969	23.10.2009		Женский
	e1e061c61daee1d518a9d8181e64c196	19.10.1969	23.10.2009		Женский

Таблица 1 – фрагмент документа Employees.csv

2) Cities – города и филиалы, в которых работаю сотрудники(в соответствии с таблицей 2).

Дата	Сотрудник	Филиал	Город
08.02.2005	60af5ad43f95c071e5f3f3588281834c	Москва	Москва
28.02.2005	e1e061c61daee1d518a9d8181e64c196	Москва	Москва
28.02.2005	8dcd51df55acd1afc191d263848e73d1	Москва	Москва
01.03.2005	16e30d6f664c5e4085b1e41e77b5aac0	Москва	Москва
02.08.2005	f66dac3272ab7d2398c79cbf9da03363	Москва	Москва
16.01.2006	7bd61b9428039f706f7c1e0602a5d5b1	Москва	Москва
07.12.2006	b2a67e7335e3072c1c6a01d5e7004c72	Москва	Москва
11.12.2006	82ac8834435110f81bb4e2dc2b12b567	Москва	Москва
02.04.2007	bead7915fc661af66f0862fff75eaa56	Москва	Москва
26.04.2007	2a8662c5d555b73331b45012e71e6b20	Москва	Москва
03.07.2007	1948c3b85be4caefd86d1cad0b8039d0	Москва	Москва
01.08.2007	9e25e42c243c8196768a74d2ecbc661f	Москва	Москва
30.10.2007	e75f54c484d022e8ef77c4780ce5ddd9	Москва	Москва
09.01.2008	dab3b281513447b57bc7a5dee370ed01	Москва	Москва
26.02.2008	4722d70b32de398f3e338452be305055	Москва	Москва
18.03.2008	aec526a464ad6813d6a6c00bbdf00f8d	Москва	Москва
21.07.2008	901efb68151020c160b9663c18a850aa	Москва	Москва
04.08.2008	0581ace14c9042669649774adf9f00ee	Москва	Москва
18.08.2008	09decd2b58fbcbdbe9bc2c70ef5491c7	Москва	Москва
25.08.2008	cc2e17ac3e5f8bd371e26a70c0ddc88f	Москва	Москва
11.12.2008	e1bc2eb75426666e817cdf797f25dd23	Москва	Москва
26.03.2010	d8499032297beae6f0d9f51ede4e302d	Москва	Москва
07.06.2011	c16a210c3ca84c82ccd772c97c3ebfbd	Москва	

Таблица 2 – фрагмент документа Cities.csv

3) Grades – данные о грейдах (статус в градации от -1 до 9, где -1 сотрудник на стажировке и 9 сотрудник Senior) и подразделениях (БН) сотрудников (в соответствии с таблицей 3).

Дата	Сотрудник	БН	grade
01.01.2019	8a195bab54f69a76d6be02a70692dc8e	ФАС	-1
01.01.2019	60af5ad43f95c071e5f3f3588281834c	Руководитель	9
01.01.2019	2e32f3195c2fd2d6660a4e48f5dbd40a	SOA-ВТБ	3
01.01.2019	c338b675efc7c06ca9f54c03c312ead1	BigData Solutions	2
01.01.2019	6f216663ad0259436e16b60c5202ce86	ФАС	-1
01.01.2019	68915ee9dd402ea9ab8745d3f1f1eca9	ФАС	-1
01.01.2019	b6c7af6f64f2e92c160caa1b4425247a	Водитель	-1
01.01.2019	1f3e490038d8dc794aaefa8802b34c0a	NFO	3
01.01.2019	bad50bad803a978392acd9d6922b7877	Департамент развития бизнеса	6
01.01.2019	b7eac6d9b1c222f3579036123b2b2256	SOA-ВТБ	2
01.01.2019	b7e281fa1e924b529d2917ab0c31091c	SOA-ВТБ	2
01.01.2019	66c677b4fd8615725fc90215db6ac2f9	SOA-ВТБ	3
01.01.2019	bd9b3c3f046542cb5d1fe520cde799c1	SOA-ВТБ	6
01.01.2019	786812d6349132b15312420e3c102a2f	SOA-ВТБ	2
01.01.2019	5a5a8e991404d751e6739cde7ec01ceb	SOA-ВТБ	3
01.01.2019	1f37a1d187ebf9822e7775fb11ea6174	SOA-ВТБ	1
01.01.2019	96270b54525e045688d1dfec3330dfdb	SOA-ВТБ	4
01.01.2019	c04e5bc0a3afbb5ca2365c558708d6ef	SOA-ВТБ	2
01.01.2019	e75f54c484d022e8ef77c4780ce5ddd9	SOA-ВТБ	5
01.01.2019	a5cc17f5ed657b28ed7c7886e98906d2	SOA-ВТБ	2
01.01.2019	2070bc632fdde2c9a3e5ba771708ec56	SOA-ВТБ	3
01.01.2019	86558e19d13f72ba259001dcfb98f96f	SOA-ВТБ	4
01.01.2019	5f2c4436b116c685330c61c8b0ac8065	SOA-ВТБ	5

Таблица 3 – фрагмент документа Grades.csv

3.2 Обработка данных

Так как работа выполнялась на Kaggle, библиотеки Numpy и Pandas подгружены априори, а вот Matplotlib необходимо импортировать. Чтобы начать работу загружаем наши файлы на Kaggle. Присваиваем название каждому файлу и получаем готовые датафреймы для работы:

```
employees = pd.read_csv("../input/empgrad/employees.csv")
grades = pd.read_csv("../input/empgrad/grades.csv")
cities = pd.read_csv("../input/cities/cities.csv")

import matplotlib.pyplot as plt
import matplotlib.dates as dt
```

Чтобы получить краткую статистику о каждой таблице используем функцию .describe():

```
employees.describe()
```

	Табельный номер	Сотрудник	Дата рождения	Дата приема	Дата увольнения	Физическое лицо.Пол
count	1259	2337	2335	2337	1001	2337
unique	1259	1895	1663	1039	686	2
top	0000001140	34bff4ba2fb785b91777d3d6f908de5d	1993-10-09	2010-01-15	2015-01-30	Мужской
freq	1	4	6	24	10	1560

Видим, что в таблице employees 2337 строк, не хватает двух дат рождений, 1895 сотрудников(уникальных) и 1001 раз сотрудники увольнялись(помним, что сотрудники могут повторятся).

```
cities.describe()
```

	Дата	Сотрудник	Филиал	Город
count	6295	6295	5986	5081
unique	975	1385	5	37
top	2020-12-01	1e21147cd02c35aa5f39adb0d3fce216	Москва	Саратов
freq	541	22	2658	1974

В таблице cities 6295 строк, 5 уникальных филиалов, но сотрудники работают из 37 городов, 1385 уникальных сотрудников.

```
grades.describe()
```

	grade
count	7541.000000
mean	2.556690
std	1.864687
min	-1.000000
25%	2.000000
50%	3.000000
75%	4.000000
max	9.000000

Так как таблица grades содержит числовые значения, наша функция показывает кол-во строк, среднее значение, среднее отклонение, и значения от 0% до 100%. Можно сделать вывод, что строк 7541 и среднее значение грейдов 2,56.

Для удобства работы я переименовал все столбцы каждой таблицы:

```
employees.columns = ['N','Employee','Birthday','Joindate','Retiredate','Gender']
cities.columns = ['Date','Employee','District','City']
grades.columns = ['Date','Employee','Depart','Grade']
```

3.3 Построение графиков

Работу я проводил поочередно с каждым датасетом, начал с таблицы employees.

Нахожу соотношение мужского/женского пола среди сотрудников:

```
g = pd.DataFrame(employees.Gender)
f = g.value_counts()
print(f)
```

Получил: Мужской 1560, Женский 777. Построил диаграмму:

```
fig, ax = plt.subplots()
labels = 'male','female'
ax.pie(f, labels = labels, autopct='%0.2f%%',startangle=90)
plt.show()
```

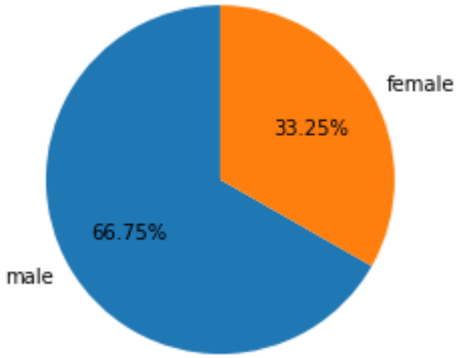


Диаграмма 1 – соотношение мужского/женского пола среди сотрудников

Найдем стаж сотрудников (количество проработанных дней) в зависимости от даты увольнения по кварталам за 2015 год. Импортируем библиотеку datetime, чтобы работать с датами.

```
from datetime import date
new = fired[['Joindate', 'Retiredate']]
filtered_df = new[new["Retiredate"].isin(pd.date_range('2015', '2016'))]
days = filtered_df['Retiredate'] - filtered_df['Joindate']
filtered_df['Days'] = days
q1_2015 = filtered_df[filtered_df["Retiredate"].isin(pd.date_range('2015', '2015-04'))]
q1_2015.reset_index(drop=True)
q1_2015["Days"] = (q1_2015["Days"]).dt.days
dates = dt.date2num(q1_2015.Retiredate)
q1_2015.set_index('Retiredate',inplace=True)
q1_2015.index.values.tolist()
print(q1_2015)
plt.gcf().autofmt_xdate()
fig, ax = plt.subplots()
ax.bar(q1_2015.index, q1_2015.Days)

#set ticks every week
ax.xaxis.set_major_locator(dt.WeekdayLocator())
#set major ticks format
ax.xaxis.set_major_formatter(dt.DateFormatter('%b %d'))
plt.gcf().autofmt_xdate()
fig.set_figwidth(30)
fig.set_figheight(15)
ax.set(xlabel="Дата увольнения",ylabel="Стаж работы в днях", title="1 квартал 2015г.")
plt.setp(ax.get_xticklabels(), rotation=45)

plt.show()
```

Получили данные за первый квартал 2015 года, по образцу выше строим столбчатые диаграммы для остальных кварталов(диаграммы 2-5):

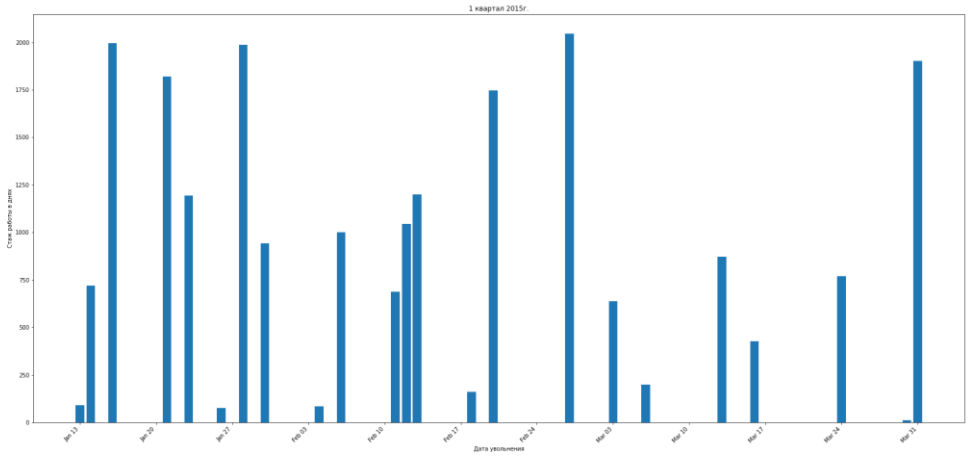


Диаграмма 2 – стаж работы по дате увольнения за 1 квартал 2015г.

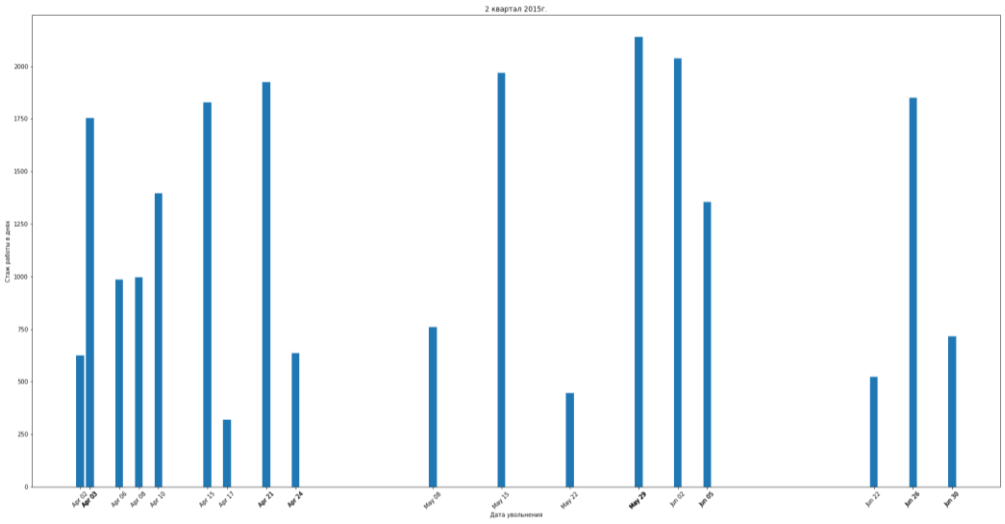


Диаграмма 3 – стаж работы по дате увольнения за 2 квартал 2015г.

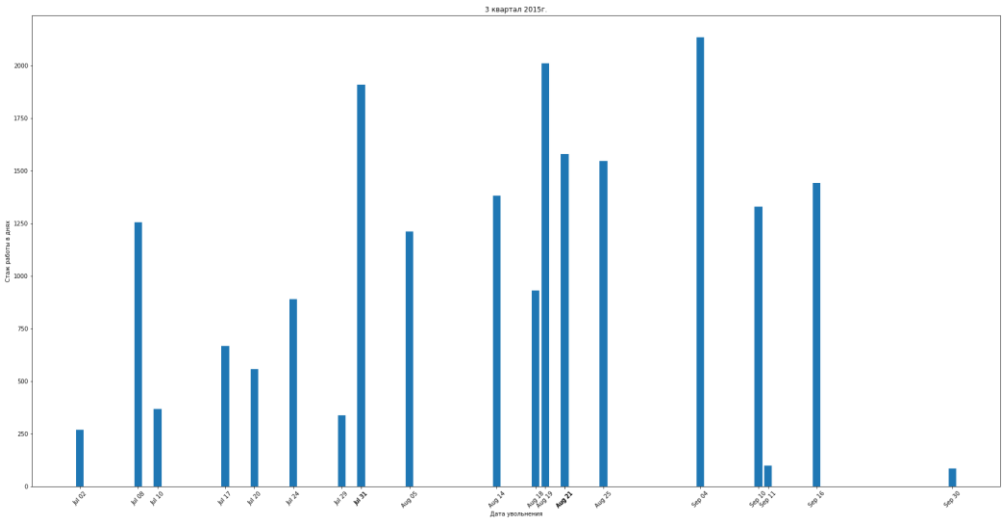


Диаграмма 4 – стаж работы по дате увольнения за 3 квартал 2015г.

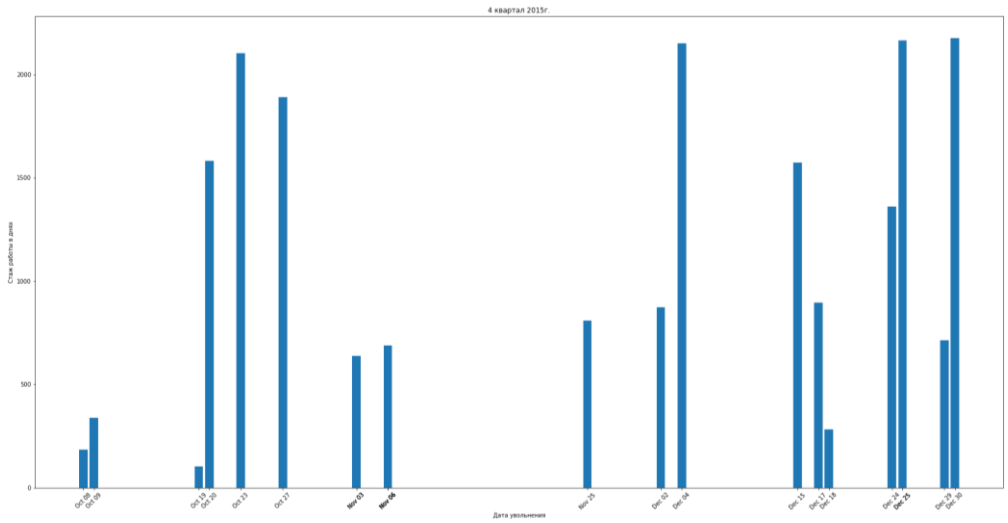


Диаграмма 5 – стаж работы по дате увольнения за 4 квартал 2015г.

Далее нашел количество сотрудников по грейдам:

```
grades.Date = pd.to_datetime(grades.Date)
q1_2015 = grades[grades["Date"].isin(pd.date_range('2015-01', '2015-03-30'))]
q2_2015 = grades[grades["Date"].isin(pd.date_range('2015-04', '2015-06-30'))]
q3_2015 = grades[grades["Date"].isin(pd.date_range('2015-7', '2015-9-30'))]
q4_2015 = grades[grades["Date"].isin(pd.date_range('2015-10', '2015-12-31'))]
q1_2015.reset_index(inplace=True,drop=True)
q2_2015.reset_index(inplace=True,drop=True)
q3_2015.reset_index(inplace=True,drop=True)
q4_2015.reset_index(inplace=True,drop=True)
q3_2015.tail()
gg1 = q1_2015.groupby('Depart').Grade.mean()
gg2 = q2_2015.groupby('Depart').Grade.mean()
gg3 = q3_2015.groupby('Depart').Grade.mean()
gg4 = q4_2015.groupby('Depart').Grade.mean()
grades.groupby('Depart').Grade.mean().sort_values(ascending=False)
gg1.sort_values(ascending=True,inplace=True)
gg2.sort_values(ascending=True,inplace=True)
gg3.sort_values(ascending=True,inplace=True)
gg4.sort_values(ascending=True,inplace=True)
q1count = q1_2015.Depart.value_counts()
q2count = q2_2015.Grade.value_counts()
q3count = q3_2015.Grade.value_counts()
q4count = q4_2015.Grade.value_counts()
fig, ax = plt.subplots()
ax.barh(gg1.index.values, gg1, color='y')
plt.xticks(q1count.index.values[::1])
```

Диаграммы 6-9 показывают среднее количество сотрудников по грейдам за каждый квартал 2015 года:

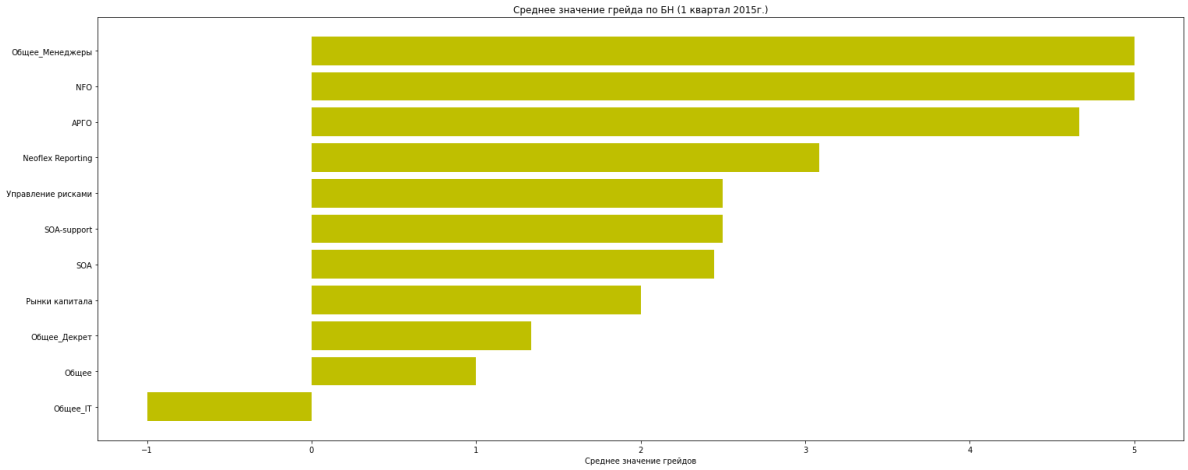


Диаграмма 6 – средние значения грейдов 1 квартала 2015г.

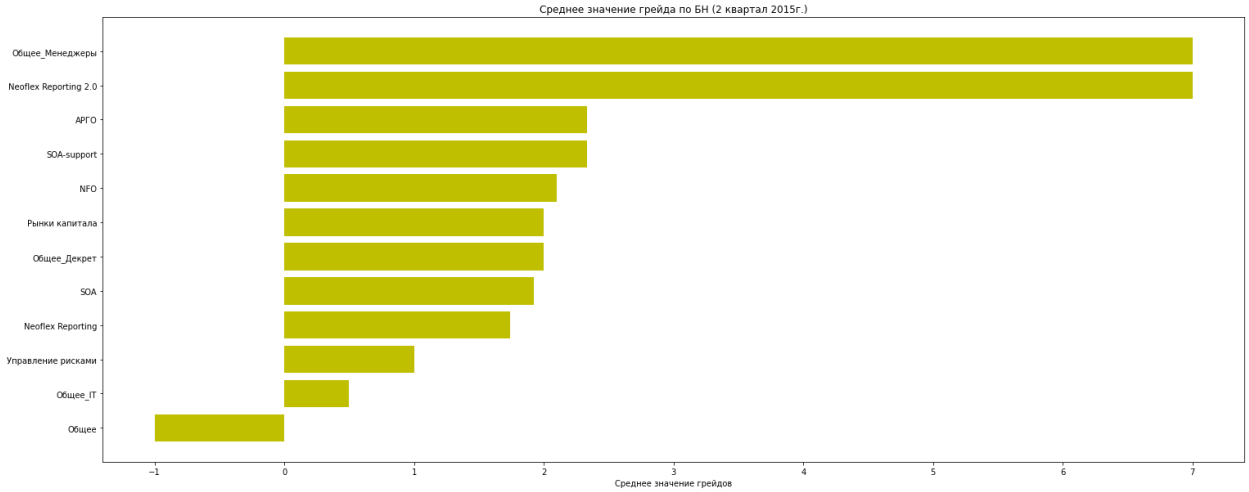


Диаграмма 7 – средние значения грейдов 2 квартала 2015г.

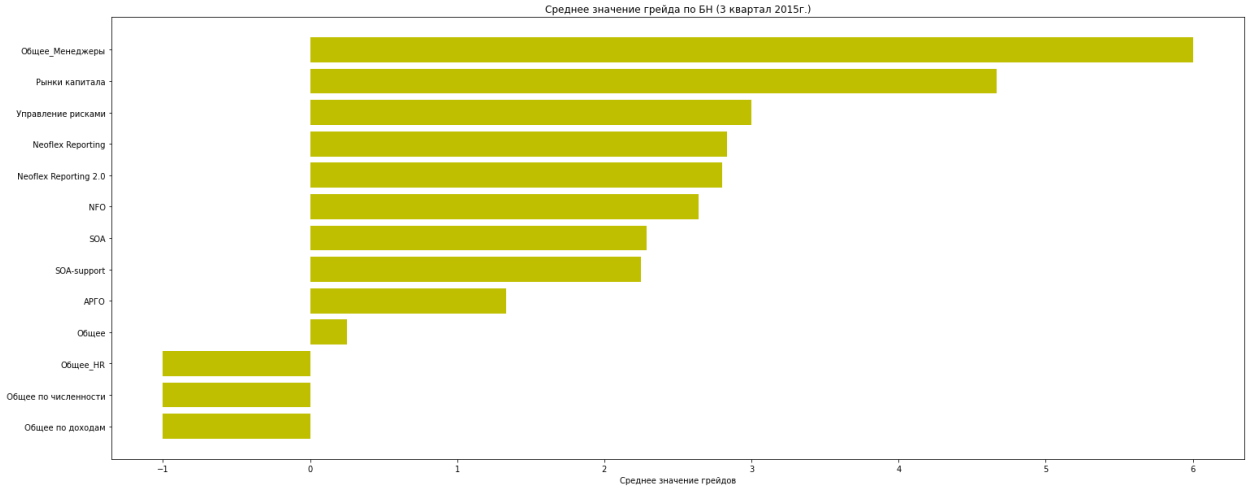


Диаграмма 8 – средние значения грейдов 3 квартала 2015г.

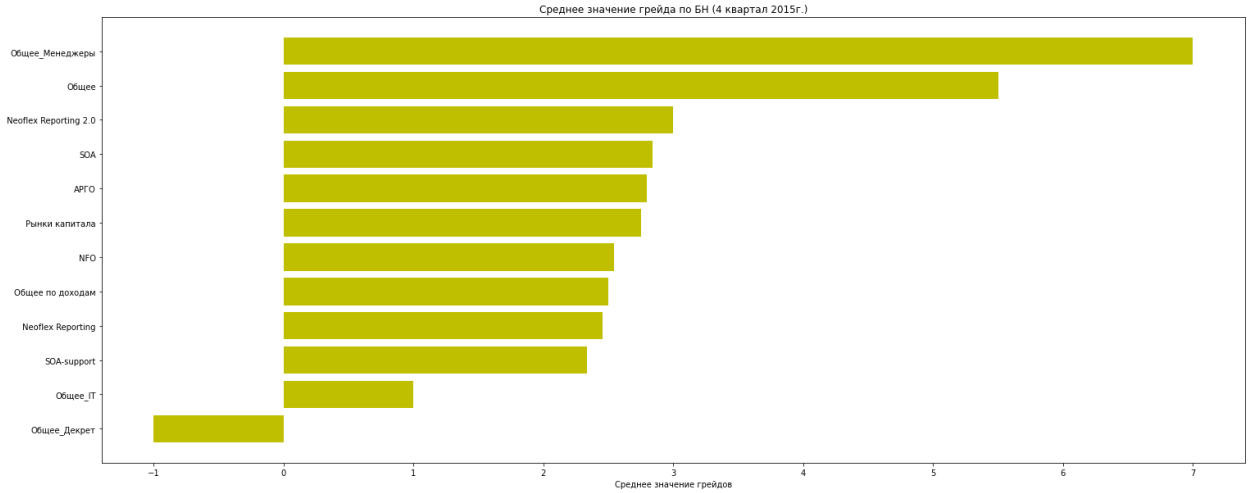


Диаграмма 9 – средние значения грейдов 4 квартала 2015г.

Далее нашел распределение кадров по городам(диаграмма 10):

```
cities3 = cities.dropna(subset=['District'])
dcount = cities3.District.value_counts()
explode = (0, 0, 0, 0, 0.7)
labels = dcount.index.values
plt.pie(dcount, labels=labels, autopct='%1.1f%%', explode=explode, startangle=180)
```

Диаграмма 10 – распределение кадров по городам

После этого нахожу количество сотрудников по подразделениям(БН, диаграммы 11-14):

```
ax.bar(q1count.index.values, q1count)
ax.bar(q2count.index.values, q2count)
ax.bar(q3count.index.values, q3count)
ax.bar(q4count.index.values, q4count)
```

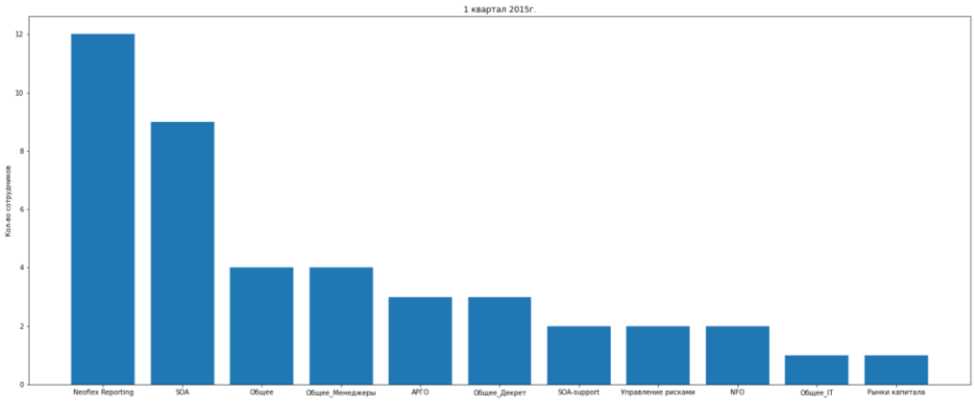


Диаграмма 11 – кол-во сотрудников по БН за 1 квартал 2015г.

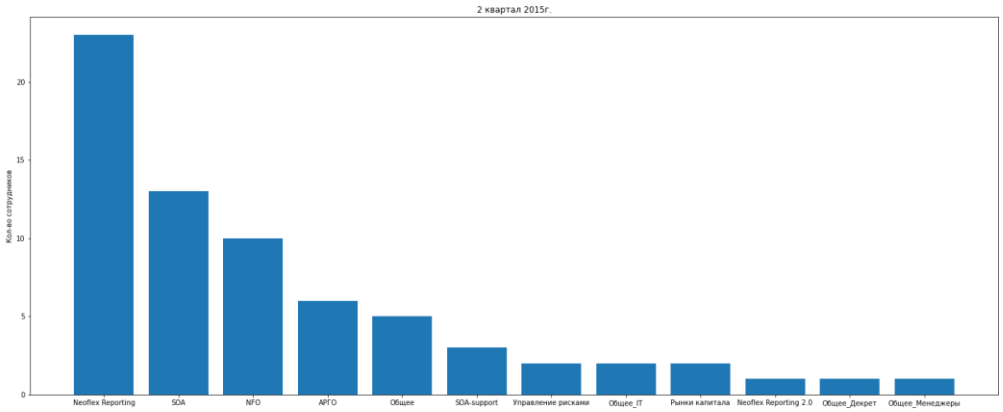


Диаграмма 12 – кол-во сотрудников по БН за 2 квартал 2015г.

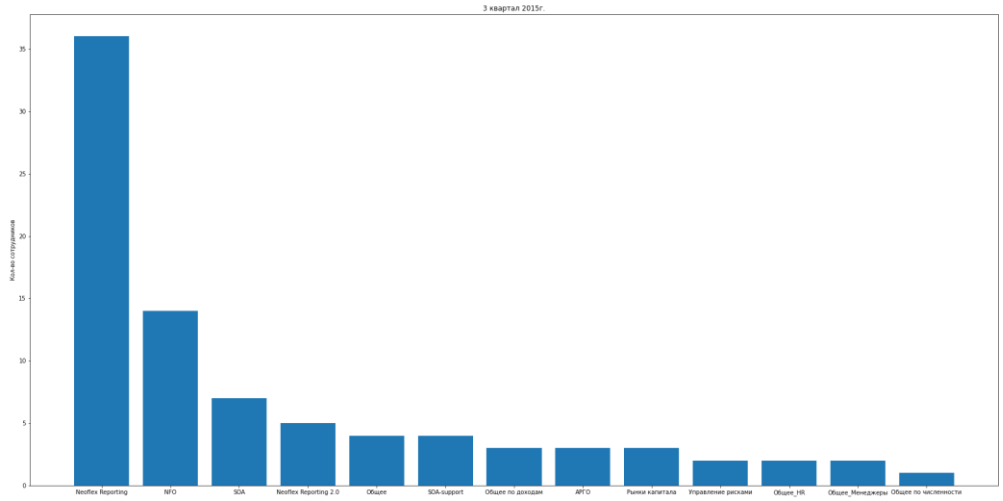


Диаграмма 13 – кол-во сотрудников по БН за 3 квартал 2015г.

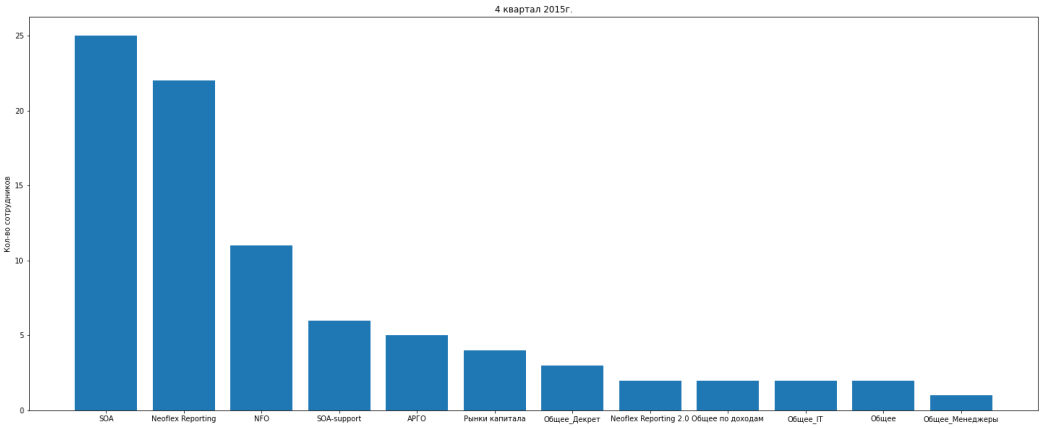


Диаграмма 14 – кол-во сотрудников по БН за 4 квартал 2015г.

ЗАКЛЮЧЕНИЕ

В результате прохождения производственной практики в компании «Неофлекс консалтинг» были получены знания и сформированы профессиональные навыки, необходимые в трудовой деятельности компании "Neoflex". Также были решены поставленные задачи по аналитике текучки кадров.

Производственная практика способствовала формированию следующих компетенций:

- ОК - 6 Способность работать в коллективе, толерантно воспринимая социальные, этнические, конфессиональные и культурные различия;
- ОПК – 2 способностью находить организационно-управленческие решения и готов нести за них ответственность; готов к ответственному и целеустремленному решению поставленных профессиональных задач во взаимодействии с обществом, коллективом, партнерами;
- ОПК - 3 Способность работать с компьютером как средством управления информацией, работать с информацией из различных источников, в том числе в глобальных компьютерных сетях;
- ПК - 13 Умение проектировать и внедрять компоненты ИТ-инфраструктуры предприятия, обеспечивающие достижение стратегических целей и поддержку бизнес-процессов;
- ПК - 19 Умение готовить научно-технические отчеты, презентации, научные публикации по результатам выполненных исследований.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1) Python [Электронный ресурс]: свободная энциклопедия / текст доступен по лицензии Creative Commons Attribution-ShareAlike ; Wikimedia Foundation, Inc, некоммерческой организации. Электрон. дан. (712413 статей, 2479181 страниц, 117 104 загруженных файлов). - Wikipedia®,2001- . - URL: https://ru.wikipedia.org/wiki/Python (дата обращения: 25.10.2021).
2) Numpy [Электронный ресурс]: - URL: https://pythonworld.ru/numpy(дата обращения: 25.10.2021).
3) Pandas [Электронный ресурс]: свободная энциклопедия / текст доступен по лицензии Creative Commons Attribution-ShareAlike ; Wikimedia Foundation, Inc, некоммерческой организации. Электрон. дан. (712413 статей, 2479181 страниц, 117 104 загруженных файлов). - Wikipedia®,2001- . - URL: https://ru.wikipedia.org/wiki/Pandas (дата обращения: 25.10.2021).
4) Laura Igual, Santi Segui Introduction to Data Science. / Igual Laura. - Cham: SpringerLink, 2017. - 227 p.
5) Pandas [Электронный ресурс]: свободная энциклопедия / текст доступен по лицензии Creative Commons Attribution-ShareAlike ; Wikimedia Foundation, Inc, некоммерческой организации. Электрон. дан. (712413 статей, 2479181 страниц, 117 104 загруженных файлов). - Wikipedia®,2001- . - URL: https://ru.wikipedia.org/wiki/Matplotlib (дата обращения: 25.10.2021).

Приложение А

Основным объектом NumPy является однородный многомерный массив (в numpy называется numpy.ndarray). Это многомерный массив элементов (обычно чисел), одного типа.

Наиболее важные атрибуты объектов ndarray:

ndarray.ndim - число измерений (чаще их называют "оси") массива.

ndarray.shape - размеры массива, его форма. Это кортеж натуральных чисел, показывающий длину массива по каждой оси. Для матрицы из n строк и m столбов, shape будет (n,m). Число элементов кортежа shape равно ndim.

ndarray.size - количество элементов массива. Очевидно, равно произведению всех элементов атрибута shape.

ndarray.dtype - объект, описывающий тип элементов массива. Можно определить dtype, используя стандартные типы данных Python. NumPy здесь предоставляет целый букет возможностей, как встроенных, например: bool_, character, int8, int16, int32, int64, float8, float16, float32, float64, complex64, object_, так и возможность определить собственные типы данных, в том числе и составные.

ndarray.itemsize - размер каждого элемента массива в байтах.

ndarray.data - буфер, содержащий фактические элементы массива. Обычно не нужно использовать этот атрибут, так как обращаться к элементам массива проще всего с помощью индексов.

В NumPy существует много способов создать массив. Один из наиболее простых - создать массив из обычных списков или кортежей Python, используя функцию numpy.array() (запомните: array - функция, создающая объект типа ndarray):

```
>>> import numpy as np
>>> a = np.array([1, 2, 3])
>>> a
array([1, 2, 3])
>>> type(a)
```

```
<class 'numpy.ndarray'>
```

Функция array() трансформирует вложенные последовательности в многомерные массивы. Тип элементов массива зависит от типа элементов исходной последовательности (но можно и переопределить его в момент создания).

```
>>> b = np.array([[1.5, 2, 3], [4, 5, 6]])
>>> b
array([[ 1.5,  2. ,  3. ],
       [ 4. ,  5. ,  6. ]])
```

Можно также переопределить тип в момент создания:

```
>>> b = np.array([[1.5, 2, 3], [4, 5, 6]], dtype=np.complex)
>>> b
array([[ 1.5+0.j,  2.0+0.j,  3.0+0.j],
       [ 4.0+0.j,  5.0+0.j,  6.0+0.j]])
```

Функция zeros() создает массив из нулей, а функция ones() — массив из единиц. Обе функции принимают кортеж с размерами, и аргумент dtype:

```
>>> np.zeros((3, 5))
array([[ 0.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0.]])
>>> np.ones((2, 2, 2))
array([[[ 1.,  1.],
        [ 1.,  1.]],
       [[ 1.,  1.],
        [ 1.,  1.]]])
```

Функция eye() создаёт единичную матрицу (двумерный массив)

```
>>> np.eye(5)
array([[ 1.,  0.,  0.,  0.,  0.],
       [ 0.,  1.,  0.,  0.,  0.],
```

```
       [ 0.,  0.,  1.,  0.,  0.],
       [ 0.,  0.,  0.,  1.,  0.],
       [ 0.,  0.,  0.,  0.,  1.]])
```

Для создания последовательностей чисел, в NumPy имеется функция arange(), аналогичная встроенной в Python range(), только вместо списков она возвращает массивы, и принимает не только целые значения:

```
>>> np.arange(10, 30, 5)
array([10, 15, 20, 25])
>>> np.arange(0, 1, 0.1)
array([ 0. ,  0.1,  0.2,  0.3,  0.4,  0.5,  0.6,  0.7,  0.8,  0.9])
```

Математические операции над массивами выполняются поэлементно. Создается новый массив, который заполняется результатами действия оператора.

```
>>> a = np.array([20, 30, 40, 50])
>>> b = np.arange(4)
>>> a + b
array([20, 31, 42, 53])
>>> a - b
array([20, 29, 38, 47])
>>> a * b
array([  0,  30,  80, 150])
```

Приложение В

Объект DataFrame лучше всего представлять себе в виде обычной таблицы и это правильно, ведь DataFrame является табличной структурой данных. В любой таблице всегда присутствуют строки и столбцы. Столбцами в объекте DataFrame выступают объекты Series, строки которых являются их непосредственными элементами.

DataFrame проще всего сконструировать на примере питоновского словаря:

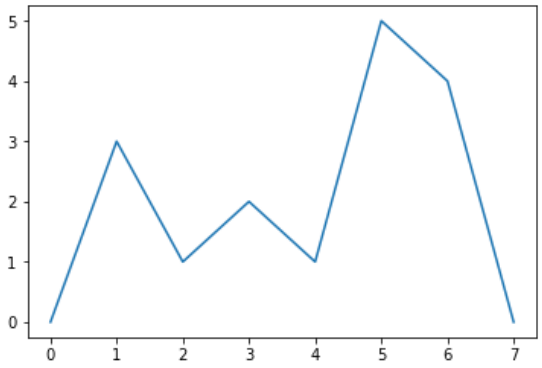
```
>>> df = pd.DataFrame({
...     'country': ['Kazakhstan', 'Russia', 'Belarus', 'Ukraine'],
...     'population': [17.04, 143.5, 9.5, 45.5],
...     'square': [2724902, 17125191, 207600, 603628]
... })
>>> df
   country  population    square
0  Kazakhstan       17.04   2724902
1      Russia      143.50  17125191
2     Belarus        9.50    207600
3     Ukraine       45.50    603628
```

Можем извлечь по отдельности любой столбец:

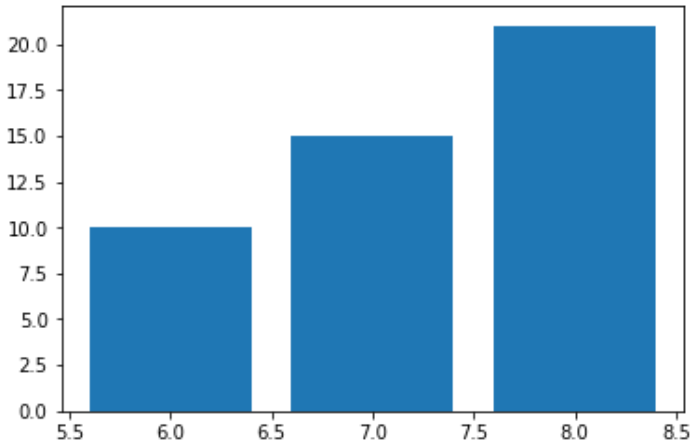
```
>>> df['country']
0    Kazakhstan
1        Russia
2       Belarus
3       Ukraine
Name: country, dtype: object
>>> type(df['country'])
<class 'pandas.core.series.Series'>
```

Приложение С

```
import matplotlib.pyplot as plt
plt.plot((0, 1, 2, 3, 4, 5, 6, 7), (0, 3, 1, 2, 1, 5, 4, 0))
plt.show()
```




```
plt.bar([6, 7, 8], [10, 15, 21])
plt.show()
```



Приложение D

Kaggle — система организации конкурсов по исследованию данных, а также социальная сеть специалистов по обработке данных и машинному обучению. Принадлежит корпорации Google (с марта 2017 года).

Среда организована как публичная веб-платформа, на которой пользователи и организации могут публиковать наборы данных, исследовать и создавать модели, взаимодействовать с другими специалистами по данным и инженерами по машинному обучению, организовывать конкурсы по исследованию данных и участвовать в них. В системе размещены наборы открытых данных, предоставляются облачные инструменты для обработки данных и машинного обучения. Также реализованы обучающие ресурсы, имеется раздел для размещения вакансий работодателями, где тоже возможна организация конкурсов для отбора наилучших кандидатов.

В Kaggle встроены языки Python и R.