



МИНИСТЕРСТВО СЕЛЬСКОГО ХОЗЯЙСТВА РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ
– МСХА имени К.А. ТИМИРЯЗЕВА»
(ФГБОУ ВО РГАУ - МСХА имени К.А. Тимирязева)

Открытый Мир. Старт в науку

Тема работы: «Визуализация результатов анализа больших данных»

Выполнил
студент 1 курса колледжа
группы ТК-И-09-21
Маврин Никита Сергеевич

Научный руководитель:
ассистент кафедры статистики и кибернетики
РГАУ-МСХА имени К.А. Тимирязева
Ульянкин Александр Евгеньевич

Москва,
2023

Введение

Python сегодня входит в число наиболее популярных языков программирования, и является языком программирования высокого уровня общего назначения. Сфера его применения весьма широка. В разработке приложений и веб-сайтов он тоже активно используется. Этот интерпретируемый объектно-ориентированный язык имеет открытый исходный код и динамическую семантику. Сам Python создавался посредством языка “Си”.

В 1989 г. Гвидо Ван Россум создал новый язык программирования под названием Python, а в 1991 г выпустил его. Главная цель, которую ставил перед собой автор — это упрощение процесса программирования. Чтобы писать код было проще, он должен стать более читабельным и понятным для человека. У Python открытый исходный код. Одно из преимуществ этого языка — возможность запускать программы на нём как на ОС Windows, так и на macOS и Linux.

Код, написанный на “питоне”, действительно легко читать и разбираться в его структурных элементах. Это делает язык программирования Python подходящим для начинающих. Но его возможности не сводятся лишь к базовым: на Python написаны и эффективно

поддерживаются сложнейшие веб-платформы и высоконагруженные приложения.

Python поддерживает динамическую типизацию, парадигмы объектно-ориентированного программирования и императивного программирования. Поэтому его выбирают программисты, практикующие RAD-модель (быструю разработку).

Глава 1 Python в анализе данных

• Сферы применения Python

Python – это скриптовый язык программирования. Он универсален, поэтому подходит для решения разнообразных задач и многих платформ, начиная с iOS и Android и заканчивая серверными ОС.

Это интерпретируемый язык — он не компилируется, то есть до запуска представляет из себя обычный текстовый файл. Программировать можно практически на всех платформах, язык хорошо спроектирован и логичен. Разработка идёт в разы быстрее, потому что кода здесь куда меньше, чем на других языках. И ещё Python отлично подходит новичкам. Его можно встретить в вебе и на мобильных устройствах, в приложениях и решениях, связанных с машинным обучением (нейросети и искусственный интеллект), а также в качестве встроенной системы.

Чаще всего Python используется в веб-разработке. Всю серверную часть веб-сайта можно написать на «питоне». Но не на чистом Python, а на популярных фреймворках (Django, Flask), которые, в свою очередь, написаны на нём. Эти фреймворки упрощают процессы генерации html-страниц, которые пользователь видит в своём браузере, запросы к базе данных, обработку адресов.

К сегодняшнему дню уже написана масса дополнительных инструментов для реализации веб-приложений. Например, с помощью HTMLGen авторства сторонних разработчиков можно добавлять готовые классы для html-страницы на Python; пакет `mod_python` помогает запускать Apache-скрипты и при этом обеспечивать стабильное функционирование шаблонов Python Server Pages.

Визуальный интерфейс. В области графики многие задачи тоже решаются с помощью языка программирования Python. Если необходимо адаптировать создаваемый графический интерфейс под стилистику операционной системы, где будет запускаться приложение, то можно использовать Python с дополнительными графическими библиотеками Dabo и PythonCard, которые значительно упростят процесс разработки.

Базы данных. Современная версия “питона” создавалась таким образом, чтобы максимально просто и ясно взаимодействовать с любыми базами данных. В частности, рабочая среда языка содержит программный интерфейс для работы с базами прямо в скрипте посредством SQL-запросов. Код на Python потребует лишь минимальных доработок, если нужно будет использовать его для БД Oracle и MySQL.

Системное программирование и администрирование. У Python есть интерфейсы для управления службами различных ОС, в которых он работает — Linux, Windows и др. Поэтому на «питоне» удобно писать портативные приложения для ПК. Уже давно язык программирования Python используют системные администраторы для написания своих программ. Посредством Python можно ускорять открытие и поиск файлов в папках, запуск программ, вычисления и другие типичные задачи.

Машинное обучение. В машинном обучении активно используются не только основной инструментарий языка Python, но и фреймворки, а также дополнительные библиотеки, созданные специально специально под машинное обучение. Наиболее популярные среди них — это TensorFlow (низкоуровневая библиотека, позволяющая пользователю самому создавать

алгоритмы) и scikit-learn (который уже содержит наиболее часто встречающиеся алгоритмы обучения).

На языке программирования Python пишется такой функционал для машинного обучения, с помощью которого работают системы распознавания голосов и лиц. Python используют специалисты по глубокому обучению и создатели нейронных сетей.

Автоматизация бизнес-процессов. Одна из наиболее востребованных ниш в IT, где используется Python – это написание коротких скриптов для автоматизации ряда рабочих процессов и стандартных процедур. К примеру, это небольшой код, автоматически обрабатывающий входящие письма: он сортирует их по наличию заданных ключевых фраз, чем сильно упрощает жизнь пользователю (делать то же самое вручную было бы сложно и долго).

В чём же секрет эффективности языка Python для программирования таких скриптов из нескольких строк? Это, в первую очередь, простой и понятный синтаксис, составлять сценарии на котором — одно удовольствие. И, во вторую очередь, отсутствие этапа компиляции и возможность сразу запустить и отладить код.

К индустрии игр многие относятся несерьёзно, и совершенно зря. Именно игровая индустрия привела к появлению многих удобных и полезных гаджетов, дала мощный толчок развитию цифровой графики и другим разработкам. Но создать крупный проект на одном только Python не получится — он занимает довольно скромное место и выполняет узкий набор функций.

- **Анализ данных на Python**

Анализ данных – это обработка и преобразование большого количества неструктурированных или неорганизованных данных с целью генерирования ключевой информации об этих данных, которые могли бы помочь в принятии обоснованных решений.

Существуют различные инструменты для анализа данных: Python, Microsoft Excel, Tableau, SaS и т.д. Data Science-специалистам для работы необходим простой, но в тоже время, функциональный язык. Многие опытные аналитики делают выбор в пользу Python. В рассматриваемой сфере он имеет множество сильных сторон.

Высокая продуктивность разработки – у Python простой синтаксис, это позволяет писать код быстрее, чем на других языках программирования (например, Java или C). При этом, код на Python получается читабельным и легко интерпретируемым.

Низкий порог входа для изучения. Анализом данных обычно занимаются те, кто участвуют в управлении бизнесом — предприниматели, аналитики, экономисты и т.п. Когда возникает потребность изучения языка программирования, остается мало времени на решение важных задач. Поэтому отказываются от Java, C и подобных — так как их изучение занимает много времени.

«Интерактивность» языка. У Питона есть встроенный интерпретатор, позволяющий «кодить» на ходу. То есть, аналитики могут проверять многочисленные гипотезы в интерактивном режиме. Работая с другими языками программирования, добиться аналогичного результата сложнее.

Интегрированные возможности для оптимизации кода. Встроенный интерпретатор пригодится специалистам, работающим с Big Data. Кроме анализа данных, приходится часто улучшать алгоритмы их обработки. И так как Python предлагает неявную и динамическую типизацию данных, сходу определить оптимизацию не получится — это можно сделать только в процессе исполнения кода.

Для этого и нужен интерпретатор. Он автоматически преобразует исходный код в машинные инструкции. На основе этого генерируются идеи по оптимизации. Например, если сравнить две инструкции, выполняющие одинаковые функции — получится быстро понять, почему одна работает быстрее другой.

Динамичное развитие языка. Java и C++ сильно отстают по скорости развития от Python. Последний больше открыт для комьюнити: любой разработчик может предлагать свои идеи, которые впоследствии могут быть добавлены в обновление. Благодаря этому с каждой новой версией производительность языка повышается, а синтаксис совершенствуется.

Визуализация данных. Matplotlib и Seaborn широко используются для визуализации данных. Они помогают конвертировать огромные списки чисел в удобные графики, гистограммы, диаграммы, тепловые карты и так далее.

Конечно, библиотек куда больше. Python предлагает бесчисленное количество инструментов для проектов в сфере анализа данных и может помочь при выполнении любых задач в процессе.

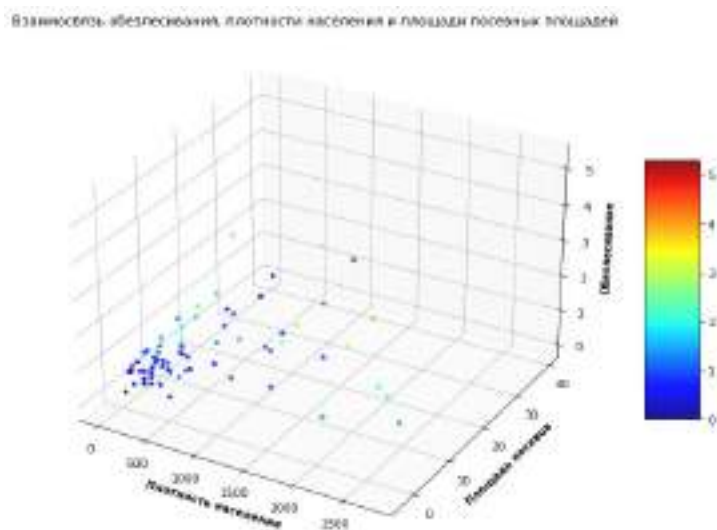


Рисунок 1 – Визуализация данных на Python на примере 3D-диаграммы

Глава 2 Примеры использования Python в анализе больших данных

- **Визуализация больших данных на Python**

Для демонстрации возможностей визуализации, был использован набор данных с информацией о наблюдениях НЛО, который включает в себя более 600 тыс. наблюдений и более 10 показателей.

Первым делом необходимо подключить нужные библиотеки:

```
import pandas as pd
import numpy as np
import pycountry
import matplotlib.pyplot as plt
from matplotlib import cm
from mpl_toolkits.mplot3d import Axes3D
# Используем класс YandexTranslate из модуля yandex_translate
from yandex_translate import YandexTranslate
# Используем класс YandexTranslateException из модуля yandex_translate
from yandex_translate import YandexTranslateException
```

Рисунок 2 – Загрузка библиотек

Для чтения файла с таблицей используется метод `read_csv` модуля `pd`. На вход функции подается имя csv файла, и чтобы подавить предупреждения при чтении файла, задаются параметры `escapechar` и `low_memory`:

```
df = pd.read_csv('./scrubbed.csv', escapechar='`', low_memory=False)
```

Рисунок 3 – Чтение данных

Теперь можно построить график наблюдений по странам. Для построения графиков используется библиотека `pyplot`:

```
# Построим график наблюдений по странам
country_count = pd.value_counts(df['country'].values, sort=True)
country_count_keys, country_count_values = dict_sort(dict(country_count))
TOP_COUNTRY = len(country_count_keys)
plt.title('Страны, где больше всего наблюдений', fontsize=PLOT_LABEL_FONT_SIZE)
plt.bar(np.arange(TOP_COUNTRY), country_count_values, color=getColors(TOP_COUNTRY))
plt.xticks(np.arange(TOP_COUNTRY), country_count_keys, rotation=0, fontsize=12)
plt.yticks(fontsize=PLOT_LABEL_FONT_SIZE)
plt.ylabel('Количество наблюдений', fontsize=PLOT_LABEL_FONT_SIZE)
plt.show()
```

Рисунок 4 – Код построения графика наблюдений по странам

Результат представлен на рисунке 5:

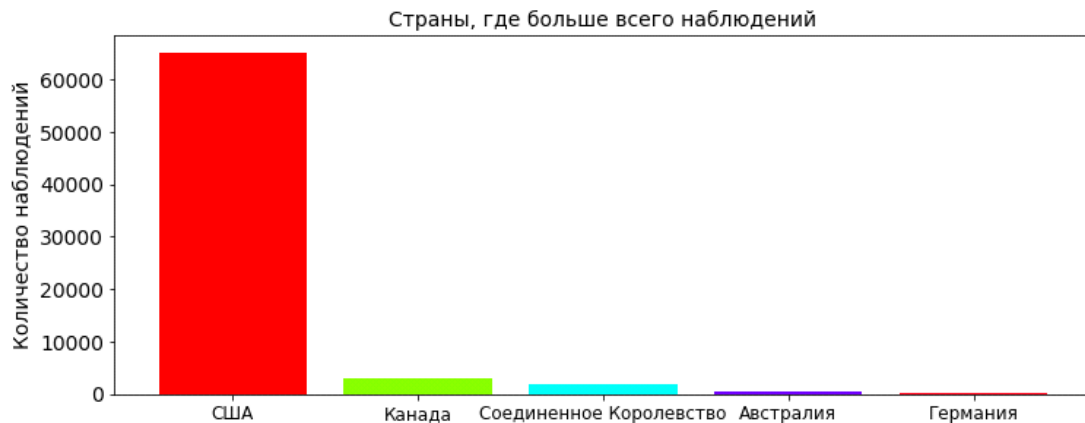


Рисунок 5 – График наблюдений НЛО по странам

Из графика видно, что больше всего наблюдений в США. И это не удивительно, ведь больше всего людей, которые следят за НЛО проживают в штатах.

Интересно еще посмотреть в какое время года наблюдали больше всего объектов:

```
# Количество наблюдений по временам года
MONTH_COUNT = [0,0,0,0,0,0,0,0,0,0,0,0]
MONTH_LABEL = ['Январь', 'Февраль', 'Март', 'Апрель', 'Май', 'Июнь',
               'Июль', 'Август', 'Сентябрь', 'Октябрь', 'Ноябрь', 'Декабрь']

for i in df['datetime']:
    m,d,y_t = i.split('/')
    MONTH_COUNT[int(m)-1] = MONTH_COUNT[int(m)-1] + 1

plt.bar(np.arange(12), MONTH_COUNT, color=getColors(12))
plt.xticks(np.arange(12), MONTH_LABEL, rotation=90, fontsize=PLOT_LABEL_FONT_SIZE)
plt.ylabel('Частота появления', fontsize=PLOT_LABEL_FONT_SIZE)
plt.yticks(fontsize=PLOT_LABEL_FONT_SIZE)
plt.title('Частота появления объектов по месяцам', fontsize=PLOT_LABEL_FONT_SIZE)
plt.show()
```

Рисунок 6 – Код построения графика наблюдений по месяцам

Результат показан на рисунке 7:

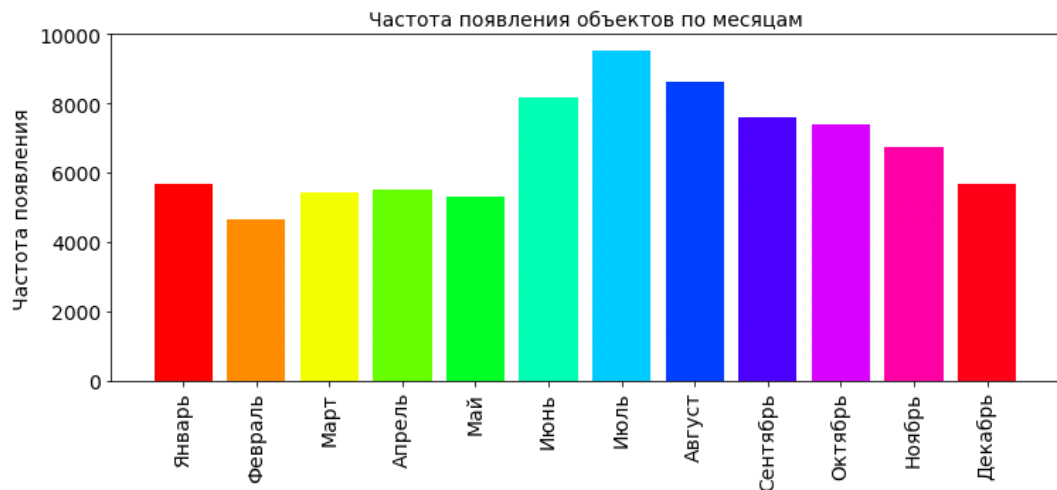


Рисунок 7 – График наблюдений НЛО по месяцам

Из графика видно, что больше всего наблюдений происходит в летнее время, когда небо ясное и звездное небо всегда видно.

Также можно посмотреть какие формы объектов на небе видели и сколько раз:

```
# Формы наблюдаемых объектов
shapes_type_count = pd.value_counts(df['shape'].values)
shapes_type_count_keys, shapes_count_values = dict_sort(dict(shapes_type_count))
OBJECT_COUNT = len(shapes_type_count_keys)
plt.title('Типы объектов', fontsize=PLOT_LABEL_FONT_SIZE)
bar = plt.bar(np.arange(OBJECT_COUNT), shapes_type_count_values, color=getColors(OBJECT_COUNT))
plt.xticks(np.arange(OBJECT_COUNT), shapes_type_count_keys, rotation=90, fontsize=PLOT_LABEL_FONT_SIZE)
plt.yticks(fontsize=PLOT_LABEL_FONT_SIZE)
plt.ylabel('Сколько раз видели', fontsize=PLOT_LABEL_FONT_SIZE)
plt.show()
```

Рисунок 8 – Код построения графика наблюдений форм НЛО

Результат показан на рисунке 8:

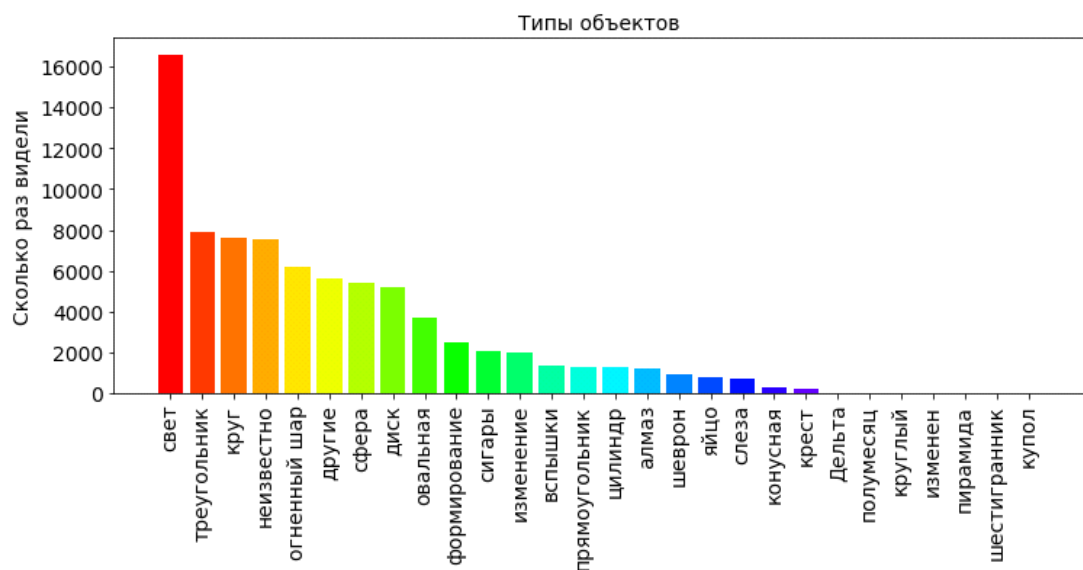


Рисунок 8 – График типов наблюдаемых объектов НЛО

Из графика видно, что больше всего на небе видели просто свет, который в принципе необязательно является НЛО. Этому явлению существует внятное объяснение, например, ночное небо отражает свет от прожекторов. Также это вполне может быть северным сиянием, которое появляется не только в полярной зоне, но и в средних широтах, а изредка даже и вблизи экватора.

Вообще, атмосфера Земли пронизана множеством излучений различной природы, например электрическими и магнитными полями. Также на ночном небе можно наблюдать как движутся космические спутники или самолеты, которые в большинстве случаев и принимают за НЛО.

Заключение

Таким образом, с помощью Python и некоторых библиотек, можно с легкостью визуализировать различного рода данные не зависимо от объема, будь то 100 тыс. наблюдений или 3-4 показателя.

В работе, рассмотрена лишь малая часть возможностей Python. На «питоне» можно строить не только графики или таблицы, но и также создать нейросеть по распознаванию лиц или других предметов в реальном времени.

Благодаря простоте и гибкости языка Python, его можно рекомендовать пользователям, не являющимся программистами, но использующими вычислительную технику и программирование в своей работе.

Список использованной литературы

- Практикум по статистике/ А.П. Зинченко, А.Е. Шибалкин, О.Б. Тарасова и др.; под ред. А.П. Зинченко. – 2-е изд., перераб. и доп. – М.:

Колос С, 2007. – 413 с.: ил. – (Учебники и учеб. пособия для студентов высш. учеб. заведений).

- Интеллектуальные системы: учебное пособие / Г. Р. Кадырова. – Ульяновск: УлГТУ, 2017. – 113 с. ISBN 978-5-9795-1745-2
- Обзор методов классификации в Python с помощью Scikit-Learn [Электронный ресурс]. – URL: <https://pythobyte.com/overview-of-classification-methods-in-python-with-scikit-learn-47463>
- Vaex – Faster Pandas Alternate in Python [Электронный ресурс]. – URL: <https://www.machinelearningplus.com/python/vaex>
- Анализ данных с использованием Python [Электронный ресурс]. / Хабр – URL: <https://habr.com/ru/post/353050/>