



Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Информационная безопасность»

ОТЧЕТ

по Рубежному контролю №2
по курсу «Программирование на Python»
на тему: «Анализ отдела аналитики»
Вариант № 6

Студент ИУ8-13М
(Группа)

(Подпись, дата)

Савватеев А. Э.
(И. О. Фамилия)

Преподаватель

(Подпись, дата)

Куликова А. В
(И. О. Фамилия)

2025 г.

СОДЕРЖАНИЕ

Введение	3
Цель работы	3
Исходные данные	3
Требования к функциональности	3
Основная часть	4
Архитектура решения	5
Используемые библиотеки	5
Методология расчета метрик	5
Описание визуализации	5
Результаты работы	6
Полученные данные	7
Аналитические выводы	7
Заключение	7

Введение

Данная работа заключается в разработке аналитической панели для мониторинга эффективности работы Отдела баз данных. В ходе выполнения контрольной работы был создан программный инструмент на языке Python, который осуществляет сбор, обработку и визуализацию данных о рабочей нагрузке отдела и уровне удовлетворенности пользователей.

Цель работы

Целью работы является разработка программного средства для аналитического анализа эффективности работы Отдела баз данных. Программа должна обеспечивать получение данных из API системы технической поддержки, их обработку и трансформацию, расчет ключевых метрик нагрузки на отдел, визуализацию динамики рабочей нагрузки и выявление корреляции между объемом нерешенных задач и уровнем удовлетворенности пользователей.

Исходные данные

Для выполнения работы использовался API endpoint `http://193.233.171.205:5000/api/v1/timeline` с учетными данными пользователя `analyst_db` (Отдел баз данных) и кодом доступа `PqR67zAb89St`. Период анализа составляет 30 дней. Система предоставляет следующие метрики: `date` (дата записи), `tickets_created` (количество созданных заявок), `tickets_resolved` (количество решенных заявок) и `satisfaction_rate` (индекс удовлетворенности пользователей в процентах).

Требования к функциональности

Разрабатываемая система должна обеспечивать подключение к API и получение данных за заданный период, а также обработку полученных данных с использованием библиотеки Pandas. Необходимо реализовать расчет дополнительных метрик, таких как ежедневное изменение количества тикетов и накопленный объем нерешенных тикетов (`backlog`). Визуализация должна представлять собой два графика: динамику нагрузки на отдел с отображением накопленного долга и новых заявок, а также корреляционный анализ

зависимости между нагрузкой и удовлетворенностью. Система также должна выводить расчетные данные для верификации результатов.

Основная часть

Архитектура решения

Программа представляет собой Python-скрипт, состоящий из трех основных компонентов. Модуль получения данных осуществляет HTTP-запрос к API с использованием библиотеки `requests`. Модуль обработки данных выполняет преобразование JSON-ответа в структуру DataFrame (Pandas) и производит необходимые расчеты. Модуль визуализации создает графики с помощью библиотек `matplotlib` и `seaborn`.

Используемые библиотеки

В работе используется библиотека `requests` для выполнения HTTP-запросов к API, `pandas` для работы с табличными данными и временными рядами, `matplotlib` как базовая библиотека визуализации данных и `seaborn` как высокоуровневая библиотека для создания статистических графиков.

Методология расчета метрик

Основной проблемой при анализе нагрузки на отдел является высокая волатильность показателя «создано заявок» — количество новых тикетов может значительно колебаться день ото дня. Для получения более стабильной картины используется метод расчета накопительного долга:

1. Ежедневный баланс:

$$\text{daily_change} = \text{tickets_created} - \text{tickets_resolved}$$

2. Накопленный объем нерешенных тикетов:

$$\text{total_active_tickets} = \sum_{i=1}^n \text{daily_change}_i$$

Данный подход позволяет отследить тренд накопления «долга» в отделе, когда количество поступающих задач систематически превышает производительность команды.

Описание визуализации

Программа генерирует комплексную панель из двух графиков.

График 1: Динамика нагрузки на отдел. Красная линия с маркерами показывает накопленный объем нерешенных тикетов, а серые столбцы отображают количество новых заявок за каждый день. Это позволяет визуально оценить, когда началось накопление задач и какими темпами растет очередь.

График 2: Корреляционный анализ. Точечная диаграмма с регрессионной прямой демонстрирует зависимость между накопленным объемом нерешенных тикетов (ось X) и уровнем удовлетворенности пользователей (ось Y). Линия тренда строится с доверительным интервалом 95%, а коэффициент корреляции Пирсона отображается в заголовке графика.

Результаты работы

Полученные данные

При запуске программа выводит в консоль таблицу с рассчитанными метриками за последние 30 дней, что позволяет верифицировать корректность расчетов. Таблица включает столбцы **date** (дата), **tickets_created** (созданные заявки), **tickets_resolved** (решенные заявки), **total_active_tickets** (накопленный backlog) и **satisfaction_rate** (удовлетворенность клиентов).

Аналитические выводы

График динамики нагрузки позволяет выявить периоды, когда отдел не справляется с объемом задач. Рост красной линии вверх свидетельствует о накоплении «долга» и требует управленческих решений. Корреляционный анализ показывает наличие связи между размером очереди нерешенных тикетов и уровнем удовлетворенности пользователей. Отрицательная корреляция указывает на то, что увеличение backlog негативно сказывается на качестве обслуживания.

Полученные метрики могут использоваться для планирования ресурсов отдела, выявления периодов повышенной нагрузки, оценки эффективности принимаемых управленческих решений и прогнозирования необходимости привлечения дополнительных специалистов.

Заключение

В ходе выполнения контрольной работы был успешно разработан аналитический инструмент для мониторинга эффективности работы Отдела баз данных. Программа демонстрирует комплексное применение современных библиотек Python для решения задач data science, включая интеграцию с REST API для получения актуальных данных, трансформацию и обработку временных рядов, расчет производных метрик для углубленного анализа и создание информативной визуализации с использованием профессиональных инструментов.

Разработанное решение позволяет менеджерам принимать обоснованные решения на основе данных, а не интуиции. Методология расчета накопительного долга показала свою эффективность в выявлении реальной картины нагрузки на отдел, скрытой за ежедневными колебаниями количества заявок. Программа может быть легко адаптирована для анализа других отделов технической поддержки путем изменения параметров аутентификации в конфигурации.

полный код и вывод можно посмотреть в репозитории на GitHub по ссылке: https://github.com/SlashLight/BMSTU_PYTHON_RK2.