Санкт-Петербургский государственный университет

Кафедра системного программирования

Группа 23.Б11-мм

Поддержка экспериментального окружения для пакета статистических гипотез PySATL

Кузнецов Дмитрий Владимирович

Отчёт по учебной практике
в форме «Решение»

Научный руководитель:
доцент кафедры системного программирования, к. ф.-м. н., Гориховский В. И.

Консультант:
разработчик ПО, ООО «Юниверс Дата», Миронов А. В.

Санкт-Петербург
2025

Оглавление

Введение

В современном мире часто приходится работать с огромным объемом данных, которые поступают из различных сфер жизни - от медицины [2] до бизнеса [1]. Для анализа таких данных становится необходимым использовать инструменты для статистической обработки. В частности, важной подзадачей такой обработки является проверка соответствия законов распределения данных на основании статистических критериев.

Некоторые критерии реализованы в библиотеках на языке R (например, в exptest: [3]). Однако язык R достаточно плохо интегрируется в программные системы. Более того, часто появляются новые, более мощные критерии, которые могут отсутствовать в старых пакетах.

В рамках проекта PySATL был создан прототип инструмента автоматической проверки статистических гипотез на языке Python [5], в котором был реализован ряд критериев для проверки гипотезы согласия, а также экспериментальное окружение. Как будет показано далее, у прототипа есть ряд недостатков, поэтому в текущем виде он не подходит для широкого использования.

Данная работа посвящена улучшению экспериментального окружения по проверке статистических гипотез.

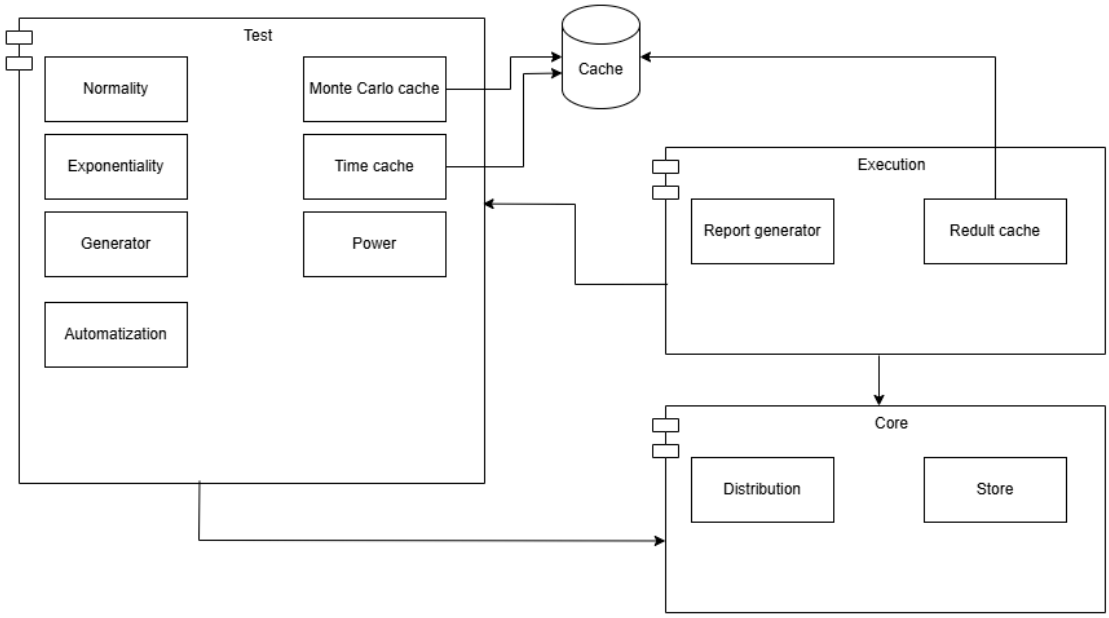

Рис. 1: Общая архитектура прототипа

1. Описание существующего прототипа окружения

В первую очередь потребовалось ознакомиться с уже реализованным прототипом инструмента экспериментального окружения проверки статистических гипотез.

1.1. Структура модулей прототипа pysatl-experiment

На данный момент прототип состоит из трех компонентов: ядра, модуля для тестирования и модуля для расчета и генерации результатов тестирования. Они продемонстрированы на рисунке 1.

1.1.1. Ядро

Ядро — модуль, предоставляющий общую функциональность для остальных модулей: генерация выборок разных размеров и типов рас-

пределения, расчет функции распределения, плотности распределения и т.д.

1.1.2. Модуль для расчета и генерации результатов тестирования

Модуль для расчета и генерации результатов тестирования — модуль, отвечающий за запуск эксперимента и генерацию представления результатов вычислений в одном или нескольких файлах в читаемом формате и содержащий информацию о времени расчета тестирования и мощности критериев для разных уровней значимости.

1.1.3. Модуль для тестирования

Модуль для тестирования — модуль, в котором содержатся реализованные на данный момент статистические критерии, проверяющие выборку на соответствие выбранной гипотезе. Также здесь находится кэш для хранения предельных распределений критериев со временем их расчета, а также уже сгенерированные выборки. Содержащиеся в данном модуле критерии будут рассмотрены отдельно.

1.2. Структура классов статистических критериев

Как было упомянуто выше, в прототипе реализован ряд статистических критериев гипотезы согласия для нормального и экспоненциального распределения. Каждый из них представлен своим отдельным классом. Кроме того, существуют общие классы для гипотез. Вместе они образуют определенную иерархию. Всего в проекте представлено более 30 критериев, поэтому рассмотрим иерархию классов на примере критерия Колмогорова-Смирнова для экспоненциального распределения и критерия Лиллиефорса для нормального такую (диаграмма для соответствующих классов представлена на рисунке 2).

Во всех классах также присутствует метод, представляющий информацию о названии критерия (например, для записи в кэш).

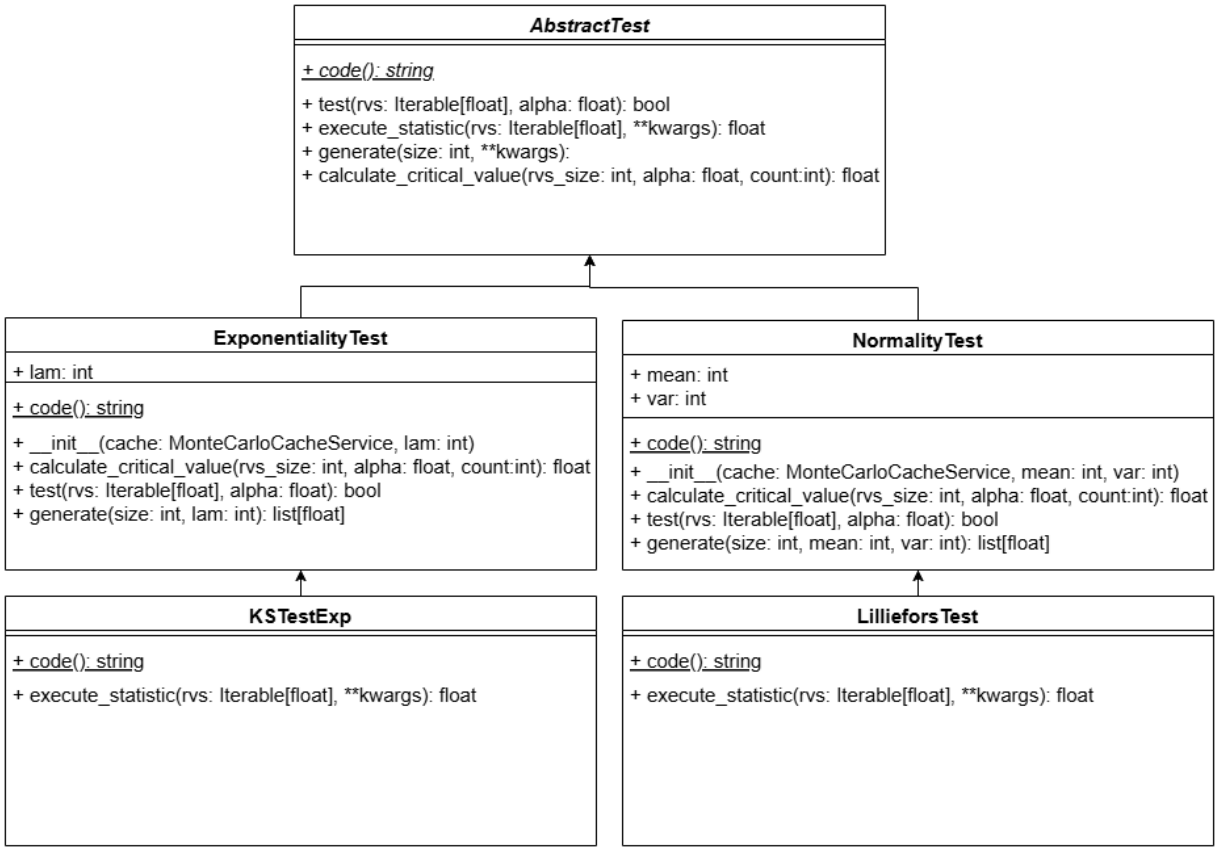

Рис. 2: Диаграмма классов для классов критериев KSTestExp и ChiSquareTest до внесения изменений

1.2.1. AbstractTest

Класс AbstractTest является абстрактным классом для статистических критериев и определяет общие методы запуска критерия, подсчета критических значений и статистики, а также генерации выборки. Реализации этих методов представлены в классах-наследниках.

1.2.2. ExponentialityTest и NormalityTest

Данные классы являются общими для статистических критериев гипотезы экспоненциального и нормального распределений соответственно. В них реализуются методы из абстрактного класса.

Стоит отметить, что в текущей реализации есть множество недостатков. Так, из названий классов сейчас не прослеживается чёткая связь между конкретными статистическими критериями и гипотезами, для которых они предназначены; это усложняет понимание архитектуры и затрудняет расширение функциональности в будущем.

Также сейчас части кода, отвечающие за работу с нормальным и экспоненциальным распределениями, содержат дублирующиеся фрагменты, что увеличивает вероятность ошибок при внесении изменений в старые критерии и добавлении новых. Общую функциональность стоит вынести в промежуточный класс.

Помимо этого, в классе NormalityTest присутствует функциональность для подсчета времени его работы. Данное решение было необходимо в прототипе, однако для широкого использования его следует при необходимости также вынести в отдельный класс.

1.2.3. KSTestExp и LillieforsTest

Данные классы непосредственно реализуют функцию подсчета статистики для соответствующих критериев. Также в них присутствует метод для генерации выборки, который можно реализовать отдельно.

1.3. Юнит-тестирование

В текущей реализации юнит-тесты написаны только для функциональности, связанной с нормальным распределением. Отсутствие тестов для других типов распределений (например, экспоненциального) снижает надёжность кода и увеличивает риск появления ошибок. Необходимо отдельно проверить процесс тестирования в приложении.

1.4. Запуск экперимента

В текущей версии окружения для запуска эксперимента каждый раз нужно писать скрипт на Python вручную с указанием необходимых классов. Пример такого скрипта имеется в репозитории.

Это достаточно неудобно для обычного пользователя приложения, возможно, не настолько хорошо знакомого с языком Python. Для облегчения ввода будет удобно, например, использовать конфигурацию, в которой можно указать используемые классы и параметры.

Наконец, в текущем виде приложение не подходит для развертывания на отдельном устройстве, что также создает трудности при запуске эксперимента.

1.5. Выводы по текущему состоянию прототипа

Таким образом, на текущем этапе разработанный прототип обладает рядом недостатков, препятствующих его широкому использованию и масштабированию. Оставшаяся часть работы будет посвящена исправлению этих недостатков.

2. Постановка цели и задач

Таким образом, целью данной работы является исправление выявленных недостатков в экспериментальном окружении пакета PySATL для её подготовки к широкому использованию.

В рамках её достижения были выделены следующие задачи:

1. проведение рефакторинга архитектуры классов статистических критериев;
2. организация процесса unit-тестирования внутри критериев;
3. добавление возможности конфигурирования пакета;
4. добавление возможности удалённого запуска эксперимента.

3. Описание предложенных изменений в архитектуре прототипа

В данном разделе описаны изменения в прототипе. В частности, здесь описывается рефакторинг архитектуры статистических критериев.

3.1. Рефакторинг архитектуры статистических критериев

Рис. 3: Диаграмма классов для классов критериев KSTestExp и ChiSquareTest после внесения изменений

3.1.1. AbstractGoodnessOfFitStatistic

Для решения проблемы несвязности критериев и гипотезы, а также объединения функциональности был написан общий класс для гипотезы согласия. Метод записи в кэш также был переопределён для

соответствия структуре наследования.

Наличие данного класса позволит упростить внедрение новых критериев для гипотезы согласия в будущем; в частности, критерии для распределения Вейбулла. Также подобная структура упростит добавление и других гипотез (например, гипотезы однородности).

3.1.2. Изменения в других классах

Все классы были переименованы для соответствия новой структуре в проекте. Это позволит упростить понимание архитектуры и ускорит добавление новых критериев.

Также из класса гипотезы экспоненциальности за ненадобностью была удалена функциональность для подсчета времени работы статистических критериев, а также метод для тестирования критериев. Их в дальнейшем планируется реализовать отдельно, но точное решение пока что не принято. Методы генерации также были убраны.

Диаграмма классов с учётом всех изменений на рассмотренном выше примере показана на рисунке 3.

3.2. Результат работы по прототипу

Промежуточным результатом работы стало создание версии прототипа, которую можно было перенести в отдельный репозиторий для дальнейшего расширения функциональности. Это и было предложено в ходе дискуссии внутри группы проекта PySATL. Совместно был создан репозиторий pysatl-experiment[7], в который были перенесены текущие наработки отдельным пулл-реквестом. Отдельно также было решено перенести код статистических критериев в репозиторий pysatl-criterion[6].

4. Налаживание процесса тестирования

В данном разделе описываются изменения, связанные с процессом тестирования в проекте.

4.1. Методика юнит-тестирования

Для повышения надежности и качества кода потребовалось расширить имеющиеся юнит-тесты для подсчета статистики в классах критериев. Изначально для прототипа использовалась библиотека pytest[4], поэтому было решено оставить её и в новом репозитории.

По аналогии с имеющимися тестами для критериев нормального распределения с положительной выборкой, были реализованы тесты для экспоненциального распределения. Также в прототипе отсутствовали тесты для выборок с нулевыми и отрицательными значениями, которые также были добавлены.

4.2. Результаты по тестам

В ходе написания тестов выяснилось, что текущая реализация критериев не всегда работает исправно. Так, для некоторых критериев не совсем понятно, что должно ожидаться при работе метода подсчета статистики при некорректных нулевых значениях. Об этой проблеме было сообщено в группе проекта, её решение планируется в дальнейшем. Также на текущем этапе не были рассмотрены другие модули исходного прототипа, так как к ним было предложено реализовать вероятностное тестирование.

Текущие юнит-тесты можно посмотреть в репозитории pysatl-crtiterion.

5. Разработка автоконфигурации

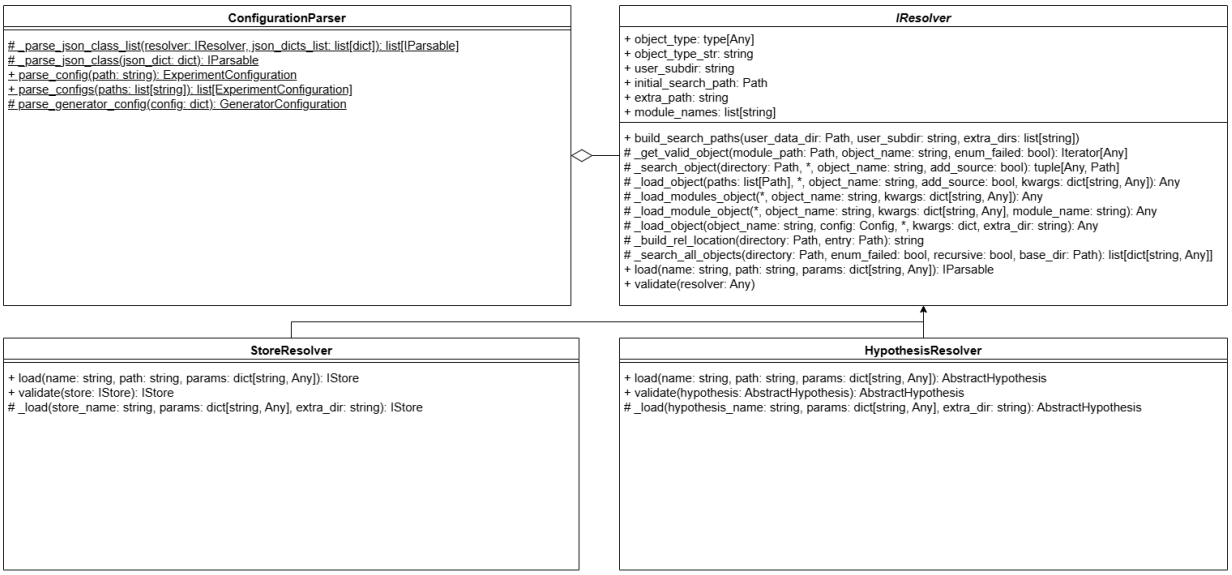

Рис. 4: Диаграмма классов парсера ConfigurationParser и некоторых ресолверов

Данный раздел посвящен решению проблемы упрощения запуска эксперимента для пользователя путем решения задачи автоконфигурации.

5.1. ConfigurationParser

В ходе работы был спроектирован и реализован класс ConfigurationParser. Его ключевой метод parse_config по пути к файлу конфигурации в формате JSON возвращает созданную конфигурацию для эксперимента. Также в классе присутствуют вспомогательные методы, использующие обобщённую функциональность ресолверов, а также версия метода для нескольких файлов.

5.2. Resolvers

Для распознавания соответствующих классов из строк файла конфигурации были написаны ресолверы. Все ресолверы реализуют интерфейс IResolver, в котором также определена общая функциональность.

В каждом ресолвере переопределяется метод load, который нужен для вывода разных типов.

Ресолверы значительно облегчают задачу запуска эксперимента, а также позволяют пользователям самим добавлять свои критерии в проект.

Ресолверы написаны для классов гипотез, генераторов, билдера, критериев, воркеров, а также хранилищ выборок. Пример структуры классов ресолверов и их взаимодейтсвие с парсером конфигурации представлено на рисунке 4.

6. Оркестрация приложения

В данном разделе описываются проделанные изменения по задаче оркестрации приложения.

6.1. Dockerfile

Для обеспечения переносимости и воспроизводимости проекта было предложено использовать технологию контейнеризации Docker. Это позволит в будущем упростить запуск проекта на любых устройствах (включая удалённые серверы), изолировать зависимости проекта от системных библиотек, а также облегчить добавление связанных сервисов, таких как база данных.

В качестве примера в репозиторий был добавлен Dockerfile, содержащий инструкции для автоматизированной сборки Docker-образа со всеми необходимыми зависимостями и настройками для запуска приложения. Также был написан отдельный скрипт, принимающий путь к файлу конфигурации и содержащий точку входа для последующего вызова парсера конфигурации.

Кроме того, в проект был добавлен файл dockercompose.yml, содержащий настройки для многоконтейнерной оркестрации. Полноценную поддержку развертывания приложения на нескольких контейнерах предполагается реализовать в дальнейшем.

Заключение

В ходе работы за текущий семестр удалось решить следующие задачи:

1. проведен рефакторинг архитектуры классов статистических критериев;
2. добавлены юнит-тесты для классов статистических критериев;
3. добавлена возможность конфигурирования пакета внутри эксперимента;
4. добавлен пример удаленного запуска эксперимента в Docker-контейнере.

В ходе работы были также выявлены направления для дальнейшей работы в следующем семестре:

- добавление большей функциональности для развертывания приложения (в частности, добавление возможности сохранения и подгрузки предельных распределений из базы данных);
- дальнейшее расширенное тестирование в проекте;
- документирование функциональности приложения.

Конечные изменения по работе можно посмотреть по следующим ссылкам ниже:

- Изменения в стат. критериях: https://github.com/PySATL/pysatl-experiment/pull/5
- Юнит-тесты: https://github.com/PySATL/pysatl-criterion/pull/16
- Парсер конфигурации и пример Dockerfile: https://github.com/PySATL/pysatl-experiment/pull/15

Список литературы

[1] Brenner RJ Kroner KF. . Arbitrage, Cointegration, and Testing the Unbiasedness Hypothesis in Financial Markets // Journal of Financial and Quantitative Analysis. — 1995. — URL: https://www.cambridge.org/core/journals/journal-of-financial-and-quantitative-analysis/article/abs/arbitrage-cointegration-and-testing-the-unbiasedness-hypothesis-i37829E4A7BD29E3F7D74F3FEBAEDFCC9 (дата обращения: 15 марта 2025 г.).

[2] Jelle J. Goeman Aldo Solari. Multiple hypothesis testing in genomics // Statistics in Medicine. — 2014. — URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/sim.6082 (дата обращения: 15 марта 2025 г.).

[3] Библиотека критериев для экспоненциального распределения exptest. — URL: https://rdrr.io/cran/exptest/ (дата обращения: 15 марта 2025 г.).

[4] Документация по библиотеке pytest. — URL: https://docs.pytest.org/en/stable/ (дата обращения: 31 марта 2025 г.).

[5] Прототип инструмента для автоматической проверки гипотез. — URL: https://github.com/alex98247/statistic-test (дата обращения: 1 апреля 2025 г.).

[6] Репозиторий pysatl-criterion. — URL: https://github.com/PySATL/pysatl-criterion (дата обращения: 1 апреля 2025 г.).

[7] Репозиторий pysatl-experiment. — URL: https://github.com/PySATL/pysatl-experiment (дата обращения: 1 апреля 2025 г.).