НИУ ВШЭ СПБ

ПАДИИ, 2 курс

Отчёт об исследовательской работе: «Применение случайных графов для проверки гипотезы согласия»

Студенты: Пожидаев Филипп, Афоничев Артемий

2025

Оглавление

1 Введение **2**

2 Описание кода **3**

2.1 Построение KNN-графа и дистанционного графа, работа с их характеристиками . . . 3

2.1.1 KNN-граф . . . 3

2.1.2 Дистанционный граф . . . 3

2.1.3 Характеристики . . . 3

2.2 Распараллеливание метода Монте-Карло . . . 3

2.2.1 `monte_carlo_step()` . . . 4

2.2.2 `monte_carlo_multiprocessing()` . . . 4

2.3 Параллельная генерация датасета . . . 4

2.3.1 `generate_row(idx, seed)` . . . 4

2.3.2 `generate_dataset(num_samples, seed)` . . . 4

3 Описание экспериментов **5**

3.1 Исследование, как ведет себя числовая характеристика графа в зависимости от параметров процедуры построения графа . . . 5

3.2 Исследование, как ведет себя числовая характеристика графа в зависимости от параметров распределения . . . 5

3.2.1 `Exp(`λ`)`, `LogNormal(0,` σ`)` . . . 5

3.2.2 `Normal(0,` σ`)`, `SkewNormal(`α`)` . . . 6

3.3 Исследование важности характеристик, как признака классификации . . . 7

3.3.1 `Exp(`λ`)`, `LogNormal(0,` σ`)` . . . 7

3.3.2 `Normal(0,` σ`)`, `SkewNormal(`α`)` . . . 9

3.3.3 Общие наблюдения . . . 10

3.3.4 Выводы . . . 10

3.4 Применение нескольких классификационных алгоритмов для фиксированного n . . . 11

3.4.1 $n = 25$ . . . 11

3.4.2 $n = 100$ . . . 11

3.4.3 $n = 500$ . . . 12

4 Результаты **14**

5 Заключение **15**

Глава 1

Введение

Отчёт работы по исследованию свойств случайных графов (`KNN` и дистанционных), построенных на основе различных вероятностных распределений.
Цель работы: исследовать поведение числовых характеристик случайных графов в зависимости от параметров распределений и параметров построения графов.
Задачи:

1. Изучить поведение числа треугольников, хроматического и кликового числа в зависимости от параметров распределений;
2. Исследовать влияние параметров процедуры построения графа и размера выборки;
3. Провести эксперименты с ML классификаторами.

Глава 2

Описание кода

В данной главе мы рассмотрим алгоритмы и реализованные функции для проведения экспериментов.

2.1 Построение KNN-графа и дистанционного графа, работа с их характеристиками

2.1.1 KNN-граф

Функция `build_knn_graph(k, vertices)` реализует алгоритм построения KNN-графа на основе заданного набора вершин `vertices` и параметра k, определяющего количество ближайших соседей для каждой вершины.
Используется алгоритм `NearestNeighbors` из библиотеки `scikit-learn`, который для каждой вершины находит $k+1$ ближайших соседей (включая саму вершину). Создаётся граф с помощью библиотеки `networkx (nx.Graph())`.

2.1.2 Дистанционный граф

Функция `build_distance_graph(d, vertices)` строит граф, в котором вершины соединяются рёбрами, если расстояние между ними не превышает заданного порога d. Для каждой пары вершин (i, j) проверяется условие $|\texttt{v[}i\texttt{]} - \texttt{v[}j\texttt{]}| \leq d$. Если условие выполняется, между вершинами добавляется ребро.

2.1.3 Характеристики

Функция `compute_stats(arr)` вычисляет основные статистики массива данных: среднее значение, дисперсию, стандартное отклонение и стандартную ошибку.
Функции, предназначенные для вычисления минимальной степени, количества треугольников, хроматического числа, кликового числа, размера максимального независимого множества, числа доминирования, минимального размера кликового покрытия являются обёрткой над существующими в `networkx` методами класса `nx.Graph`.

2.2 Распараллеливание метода Монте-Карло

В данном разделе описывается реализация метода Монте-Карло с использованием параллельных вычислений для эффективного статистического анализа графовых структур. Предложенный подход позволяет ускорить проведение множественных экспери-

ментов за счёт распределения вычислений между несколькими ядрами процессора.
Алгоритм состоит из двух основных функций. Они принимают следующий набор аргументов — `n, distr_param, graph_param, gen_func, graph_func, res_func` (однако, второй в самом начале на вход ещё подаётся параметр `M`).

2.2.1 `monte_carlo_step()`

Выполняет отдельное испытание (одно повторение метода Монте-Карло). Является атомарной операцией для параллелизации.
Выполняет следующие шаги для одного испытания:

1. Генерирует набор вершин с помощью функции `gen_func` с заданными параметрами распределения `distr_param`;
2. Строит граф указанным методом (`graph_func`) с параметром `graph_param`;
3. Вычисляет и возвращает требуемую характеристику графа с помощью функции `res_func`.

2.2.2 `monte_carlo_multiprocessing()`

Организует параллельное выполнение множества испытаний. Использует библиотеку `joblib` для распараллеливания, задействует все доступные ядра процессора, а результат всех испытаний собирает в единый массив.

2.3 Параллельная генерация датасета

В данном разделе описывается алгоритм параллельной генерации датасета для исследования характеристик случайных графов. Реализация использует многопоточные вычисления для эффективного создания большого объема данных.

2.3.1 `generate_row(idx, seed)`

Генерирует дистанционный граф на n вершинах (n выбирается случайно из заданного набора N), считает ключевые характеристики.

2.3.2 `generate_dataset(num_samples, seed)`

Использует все ядра процессора, автоматически распределяет задачи, выводит прогресс бар с помощью модуля `tqdm`, собирает результат в единый `pd.DataFrame`.

Глава 3

Описание экспериментов

Теперь перейдем к самим экспериментам.

3.1 Исследование, как ведет себя числовая характеристика графа в зависимости от параметров процедуры построения графа

В случае с дистанционным графом было установлено, что при росте n и d числовая характеристика и метрики качества растут, но у d есть критический порог, после которого метрики растут незначительно или же вовсе не растут, этот порог для большинства n равен $d = 0.8$. Это касается всех распределений (`Exp, LogNormal, Normal, SkewNormal`), но при анализе `Normal` и `SkewNormal` был замечен сдвиг порога ближе к единице.
Дальнейшее исследование проводилось с фиксированным d, равным:

- 0.8 для `Exp, LogNormal`;
- 0.9 для `Normal, SkewNormal`.

В `KNN`-графе было замечено, что метрики качества сильно зависят от n, а при больших n качество оказалось не хуже, чем в дистанционном графе, поэтому именно он будет участвовать в дальнейших экспериментах, начиная с исследования важности характеристик для классификации.
Стоит отметить, что изначальные попытки извлечения какой-то информации о `KNN`-графе из его минимальной степени вершины являются артефактом условия исследовательской работы, поэтому было принято решение рассмотреть количество треугольников для всех распределений.

3.2 Исследование, как ведет себя числовая характеристика графа в зависимости от параметров распределения

3.2.1 `Exp`(λ), `LogNormal(0,` σ`)`

`KNN`-граф:

- λ особо не влияет на характеристику `KNN`-графа, при фиксированном σ и изменении λ метрики качества остаются примерно равными;

- σ довольно сильно влияет на результат, чем больше σ, тем «левее» значения `Exp(`λ`)` и «правее» значения `LogNormal(0, `σ`)` $\Rightarrow$ мы можем классифицировать их с большей точностью.

Дистанционный граф:

- Чем больше λ и σ, тем меньше мощность;
- λ влияет на характеристику дистанционного графа значительнее, чем σ;
- При достаточно больших λ, то есть $\lambda > 1$ мощность нулевая.

3.2.2 `Normal(0, `σ`), SkewNormal(`α`)`

`KNN`-граф:

- Мощность больше зависит от α, чем от σ. Ошибка первого рода почти везде одинаковая.

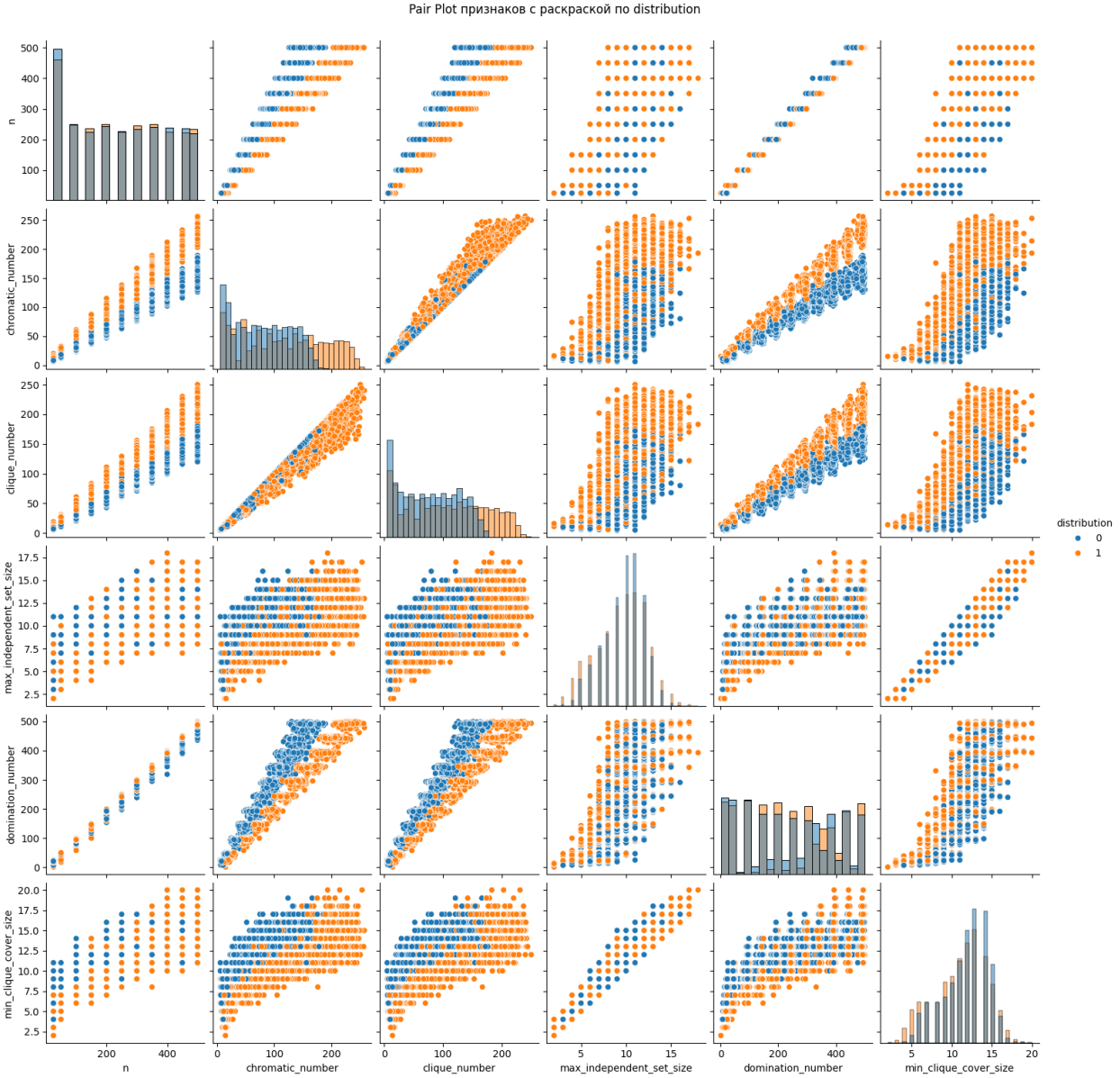
Дистанционный граф:

- Чем больше α и σ, тем больше мощность. Обе переменные вносят хороший вклад в рост характеристик;
- Можно заметить, как увеличение α сдвигает график для `SkewNormal(`α`)` «правее», а увеличение σ сдвигает график для `Normal(0, `σ`)` «левее». Этот факт может помочь в будущем с точностью классификации.

3.3 Исследование важности характеристик, как признака классификации

3.3.1 Exp(λ), LogNormal(0, σ)

Посмотрим на распределение таргета относительно признаков:



Выводы:

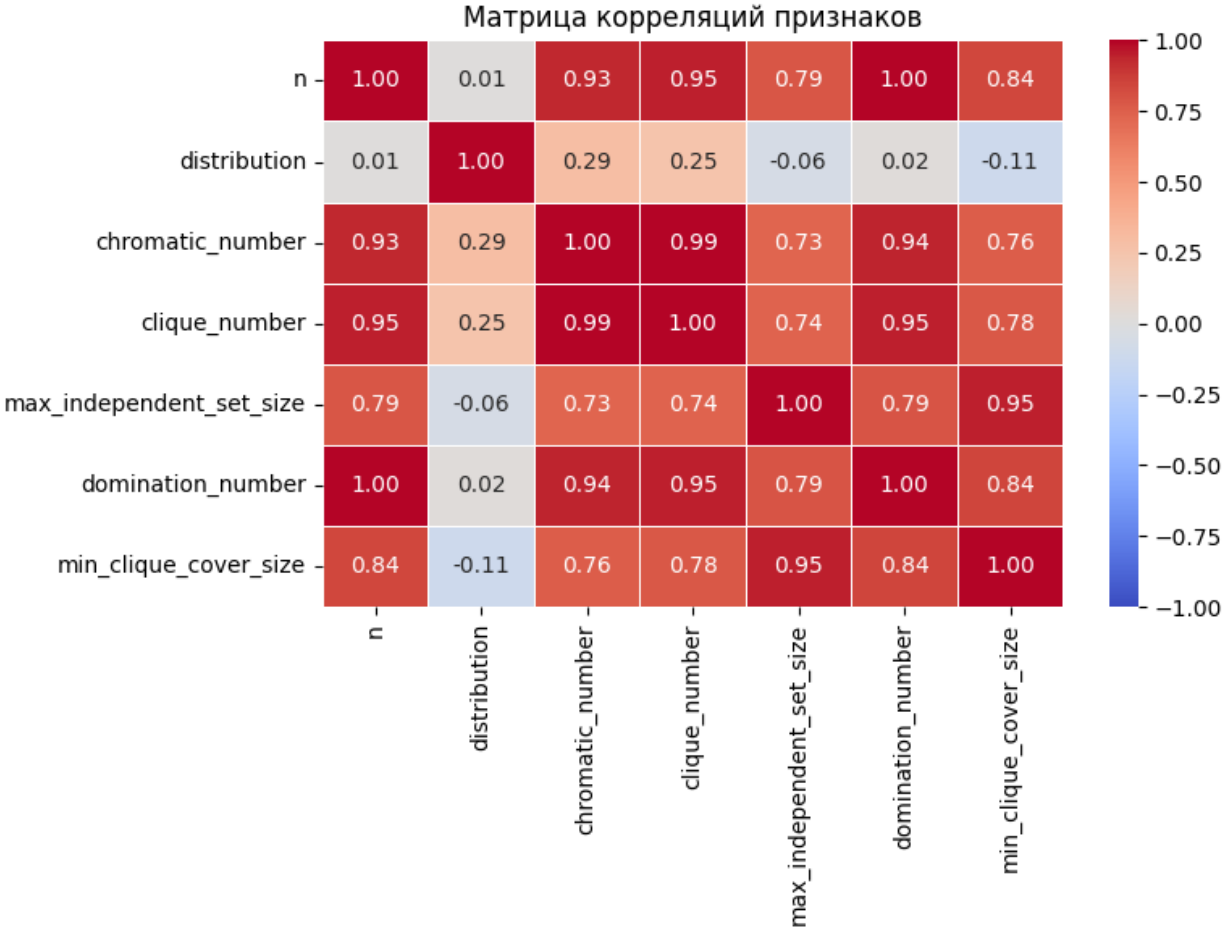
- Самые важные признаки для классификации: `chromatic_number` и `clique_number` (в нашем случае это одно и то же), они достаточно хорошо разделяют данную выборку на два класса при всех n;
- При росте n важность характеристик не меняется, по-прежнему самый важный - `chromatic_number`.

Между признаками прослеживаются зависимости:

- `domination_number` линейно зависит от `chromatic_number`;
- `max_independent_set_size` и `min_clique_cover_size` (в нашем случае это одно и то же) квадратично зависят от `domination_number`;

- `max_independent_set_size` квадратично зависит от `chromatic_number`.

Посмотрим на корреляции признаков:

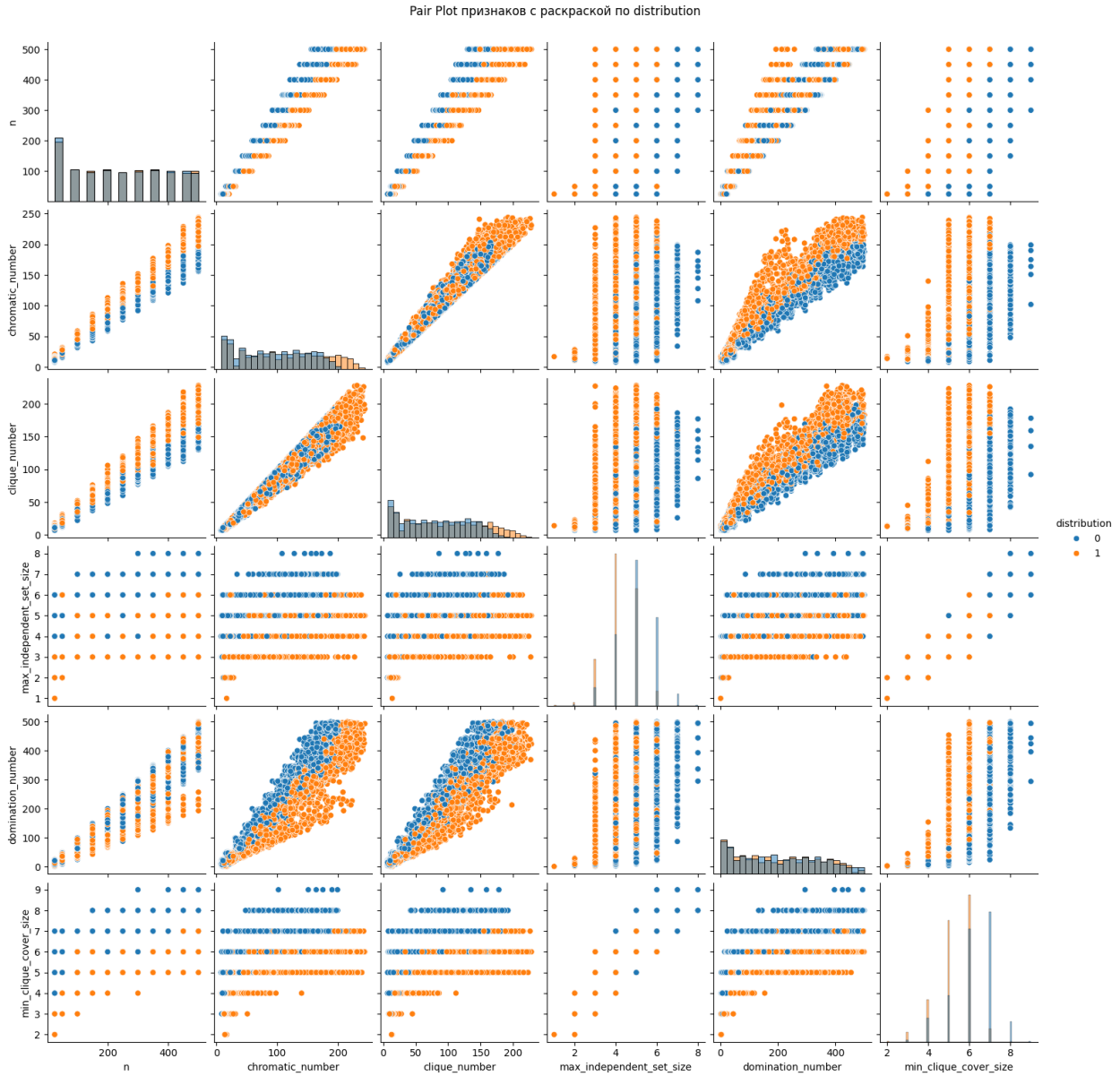


Выводы:

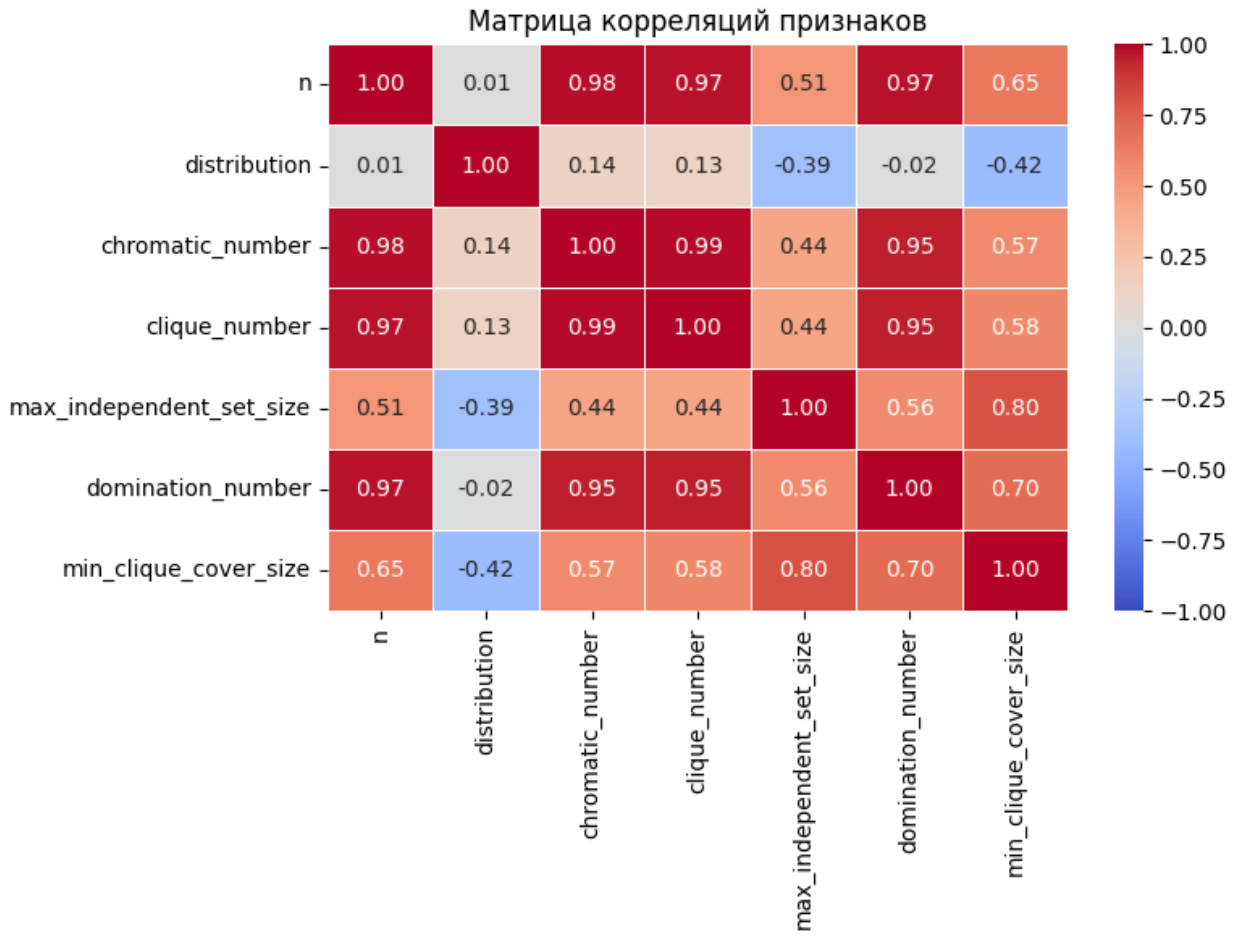
- В целом, тут мы можем найти подтверждения нашим выводам по pair plot;
- Больше всего с таргетом `distribution` коррелируют `chromatic_number` и `clique_number`;
- `domination_number` имеет сильную корреляцию со всеми остальными признаками и очень слабую с таргетом;
- `max_independent_set_size` и `min_clique_cover_size` имеют слабую корреляцию с таргетом, но довольно сильно зависят от других характеристик.

3.3.2 Normal(0, σ), SkewNormal(α)

Посмотрим на распределение таргета относительно признаков:



Аналогичные наблюдения.
Посмотрим на корреляции признаков:



Выводы:

- Получилось, что самая сильная корреляция с таргетом у `max_independent_set_size` и `min_clique_cover_size`;
- `domination_number` имеет сильную корреляцию со всеми остальными признаками и очень слабую с таргетом.

3.3.3 Общие наблюдения

Были предприняты попытки сгенерировать больше признаков путём нормирования, деления, возведения в квадрат и других операций, которые показались логичными в контексте конкретных признаков и их зависимости. В итоге особого прироста эффективности данная эвристика не дала, поэтому было принято решение обучать модели на изначальном датасете, взяв n в качестве гиперпараметра модели.

3.3.4 Выводы

- С поставленной задачей лучше всего справляется листанционный граф с параметром $d = 0.8$ для `Exp(`λ`)`, `LogNormal(0,` σ`)` и $d = 0.9$ для `Normal(0,` σ`)`, `SkewNormal(`α`)`;
- Значимость признаков не зависит от n, но при больших n разделение становится более явным.

3.4 Применение нескольких классификационных алгоритмов для фиксированного n

В данном исследовании строился дистанционный граф с параметром $d = 0.8$ для `Exp(`λ`)`, `LogNormal(0, `σ`)` и $d = 0.9$ для `Normal(0, `σ`)`, `SkewNormal(`α`)`, затем считались его характеристики и для фиксированного n применялись три модели:

- `LogisticRegression`
- `SVM`
- `RandomForestClassifier`

3.4.1 n = 25

`Exp(`λ`)`, `LogNormal(0, `σ`)`

Таблица 3.1: Метрики разных моделей

модель	мощность	ошибка первого рода
LogisticRegression	0.86	0.15
SVM	0.86	0.14
RandomForest	0.79	0.16

В таблице 3.1 приведены метрики для `Exp(`λ`)`, `LogNormal(0, `σ`)`, выводы:

- Лучшие метрики показала SVM;
- Самый важный признак - хроматическое число.

`Normal(0, `σ`)`, `SkewNormal(`α`)`

Таблица 3.2: Метрики разных моделей

модель	мощность	ошибка первого рода
LogisticRegression	0.59	0.23
SVM	0.64	0.28
RandomForest	0.68	0.32

В таблице 3.2 приведены метрики для `Normal(0, `σ`)`, `SkewNormal(`α`)`, выводы:

- Лучшие метрики показала логистическая регрессия;
- Самый важный признак - хроматическое число.

3.4.2 n = 100

`Exp(`λ`)`, `LogNormal(0, `σ`)`

В таблице 3.3 приведены метрики для `Exp(`λ`)`, `LogNormal(0, `σ`)`, выводы:

- Лучшие метрики показала SVM;
- Самый важный признак - хроматическое число.

Таблица 3.3: Метрики разных моделей

модель	мощность	ошибка первого рода
LogisticRegression	0.92	0.05
SVM	0.94	0.06
RandomForest	0.92	0.07

`Normal(0, σ), SkewNormal(α)`

Таблица 3.4: Метрики разных моделей

модель	мощность	ошибка первого рода
LogisticRegression	0.80	0.19
SVM	0.81	0.22
RandomForest	0.79	0.19

В таблице 3.4 приведены метрики для `Normal(0, σ), SkewNormal(α)`, выводы:

- Лучшие метрики показала логистическая регрессия с новыми признаками;
- Самый важный признак - минимальное кликовое покрытие разделить на хроматическое число.

3.4.3 $\mathbf{n} = 500$

`Exp(λ), LogNormal(0, σ)`

Таблица 3.5: Метрики разных моделей

модель	мощность	ошибка первого рода
LogisticRegression	1.0	0
SVM	1.0	0
RandomForest	1.0	0

В таблице 3.5 приведены метрики для `Exp(λ), LogNormal(0, σ)`, выводы:

- При достаточно больших $n = 500$ все алгоритмы могут определить гипотезу с очень хорошей вероятностью;
- Самый важный признак - хроматическое число.

`Normal(0, σ), SkewNormal(α)`

В таблице 3.6 приведены метрики для `Normal(0, σ), SkewNormal(α)`, выводы:

- Лучшие метрики показала логистическая регрессия с новыми признаками;
- Самый важный признак - минимальное кликовое покрытие разделить на хроматическое число.

Таблица 3.6: Метрики разных моделей

модель	мощность	ошибка первого рода
LogisticRegression	0.98	0.02
SVM	0.98	0.02
RandomForest	0.96	0.03

Глава 4

Результаты

Сделаем небольшие сводные таблицы с наилучшими результатами.

Таблица 4.1: Результаты измерений для `Exp(`λ`), LogNormal(0, `σ`)`

n	мощность	ошибка первого рода
25	0.86	0.14
100	0.94	0.06
500	1.0	0

Таблица 4.2: Результаты измерений для `Normal(0, `σ`), SkewNormal(`α`)`

n	мощность	ошибка первого рода
25	0.59	0.23
100	0.80	0.19
500	0.98	0.02

Глава 5

Заключение

В ходе работы было установлено, что числовые характеристики случайных графов достаточно сильно влияют на распределение первоначальной выборки.

Основные выводы:

- Рост n значительно влият на качество классификатора, при больших n мы можем делать более точные предсказания;
- Для некоторых распределений помогает преобразование старых и генерация новых признаков;
- Основные характеристики - хроматическая число и ее производные.