ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ БЮДЖЕТНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«ФИНАНСОВЫЙ УНИВЕРСИТЕТ ПРИ ПРАВИТЕЛЬСТВЕ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ»**

**(ФИНАНСОВЫЙ УНИВЕРСИТЕТ)**

Кафедра анализа данных и машинного обучения

Факультета информационных технологий и анализа больших данных

***Дисциплина: «Машинное обучение в семантическом и сетевом анализе»***

*Направление подготовки: «Прикладная математика и информатика»*

*Профиль: «Анализ данных и принятие решений в экономике и финансах»*

*Факультет информационных технологий и анализа больших данных*

*Форма обучения очная*

*Учебный 2023/2024 год, 6 семестр*

**Курсовая работа на тему:**

«Построение рекомендательной системы в области закупок с использованием графовых нейронных сетей»

Выполнил(а):

студентка группы ПМ21-1

Стаханова А.А.

*Научный руководитель:*

ассистент Бахматов А.В.

**Москва 2024**

**Содержание**

Оглавление

[Введение 3](#_Toc167665867)

[Глава 1. Теоретическая часть 3](#_Toc167665868)

[1.1 Рекомендательные системы 3](#_Toc167665869)

[1.2 Графы в контексте рекомендательных сетей 3](#_Toc167665870)

[1.3 Графовые нейронные сети 3](#_Toc167665871)

[1.4 Задача Link Prediction 3](#_Toc167665872)

[Глава 2. Практическая часть 4](#_Toc167665873)

[1.1 Описание датасета 4](#_Toc167665874)

[1.2 Предобработка данных 5](#_Toc167665875)

[1.3 Разделение датасета 6](#_Toc167665876)

[1.4 Обработка информации о продуктах 6](#_Toc167665877)

[1.5 Векторизация 7](#_Toc167665878)

[1.6 Создание графа 8](#_Toc167665879)

[1.7 Разделение на тренировочную и тестовую выборки 11](#_Toc167665880)

[1.8 Используемые метрики 12](#_Toc167665881)

[1.9 DotPredictor 12](#_Toc167665882)

[1.10 Модель GraphSAGE 13](#_Toc167665883)

[1.11 Модель GCN - 2 слоя 16](#_Toc167665884)

[1.12 Модель GCN - 3 слоя 18](#_Toc167665885)

[1.13 Модель с собственным SAGEConv 21](#_Toc167665886)

[1.14 Модель с собственным WeightedSAGEConv 24](#_Toc167665887)

[1.15 Результаты 27](#_Toc167665888)

[1.16 Построение рекомендаций 28](#_Toc167665889)

[Заключение 31](#_Toc167665890)

# Введение

В современных условиях конкурентного рынка, где информация и скорость принятия решений играют ключевую роль, компании все чаще обращаются к инновационным технологиям для оптимизации своих бизнес-процессов. Одним из таких направлений является использование рекомендательных систем в области закупок. Внедрение подобных технологий открывает новые горизонты для улучшения эффективности и снижения затрат различных предприятий.

Цель данной курсовой работы — исследование и разработка рекомендательной системы для закупок с использованием графовых нейронных сетей.

Задачи:

1. Рассмотреть теоретические основы рекомендательных систем и графовых нейронных сетей
2. Провести обзор существующих подходов и технологий, используемых в построении рекомендательных систем и графовых нейронных сетей
3. Разработать несколько моделей
4. Сравнить качество построенных моделей
5. Построить рекомендательную систему на наилучшей модели с акцентом на специфические задачи и особенности процесса государственных закупок

Настоящая работа направлена на демонстрацию потенциала графовых нейронных сетей в создании эффективных рекомендательных систем для закупок. Она основана на существующих достижениях: сверточные сети графов (GCN) и GraphSAGE.

# Теоретическая часть

## Рекомендательные системы

Система рекомендаций — это алгоритм искусственного интеллекта или ИИ, обычно связанный с машинным обучением, который использует большие данные, чтобы предлагать или рекомендовать потребителям дополнительные продукты. Рекомендательные алгоритмы собирают данные о пользователях, товарах и контексте (как на конкретном сайте, так и за его пределами), и посредством набора алгоритмических решений (algorithmic decisions) формируют для каждого пользователя, персонализированный опыт взаимодействия. Они могут основываться на различных критериях, включая прошлые покупки, историю поиска, демографическую информацию и другие.

На сегодняшний момент существует огромное количество алгоритмов и методик рекомендательных систем, большинство из них делятся на следующие широкие категории: совместная фильтрация, фильтрация содержимого и контекстная фильтрация.

***Алгоритмы совместной фильтрации*** (collaborative filtering) - этот подход использует сходство поведения предпочтений пользователей, учитывая предыдущие взаимодействия между пользователями и элементами. Рекомендательные алгоритмы учатся прогнозировать будущее взаимодействие. Существует два основных типа совместной фильтрации:

1. user-based collaborative filtering - основан на сходстве между пользователями. Собираются данные о предпочтениях пользователей. Для каждого пользователя определяется группа пользователей с похожими предпочтениями (соседи). На основе рейтингов соседей делаются предсказания для данного пользователя.

2. item-based collaborative filtering - фокусируется на сходстве между элементами. Собираются данные о взаимодействии пользователей с элементами. Для каждого элемента определяется группа элементов с похожими оценками. Если пользователь оценил какой-то элемент, то система рекомендует другие похожие элементы, основываясь на сходстве между элементами.

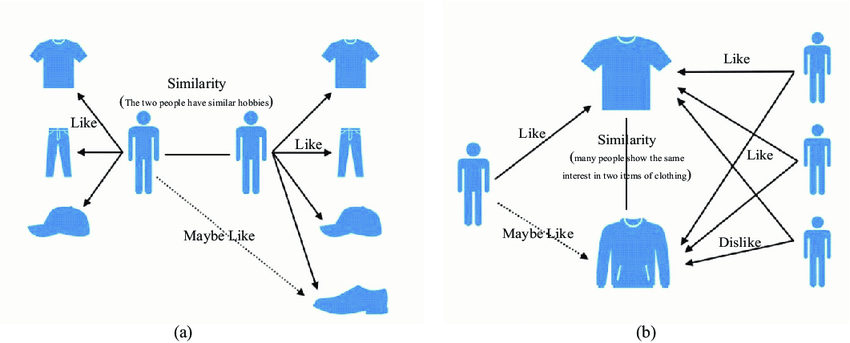


Рис. 1 Сollaborative filtering: (a) user-based; (b) item-based.

***Фильтрация, основанная на содержании*** (content-based filtering) этот подход использует атрибуты, свойства или функции элемента, чтобы рекомендовать другие элементы, аналогичные предпочтениям пользователя. Этот подход основан на сходстве элементов и пользовательских характеристик с учетом информации о пользователе и элементах, с которыми он взаимодействовал (например, возраст пользователя, категория кухни ресторана, средний обзор фильма), моделирует вероятность появления нового взаимодействие.

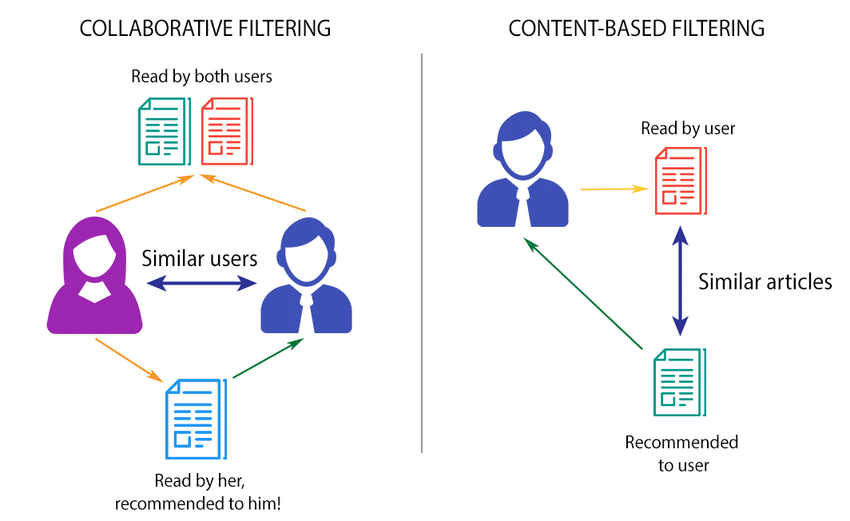


Рис. 2 Сontent-based filtering

***Контекстная фильтрация*** (context filtering) это подход учитывает не только предпочтения пользователя и характеристики элементов, но и контекст, в котором происходит взаимодействие. Контекст может включать различные факторы, такие как время, местоположение, устройство, настроение пользователя и другие окружающие условия. К примеру, система может предлагать разные типы контента в разное время дня: утром рекомендовать новости, а вечером фильмы или сериалы.

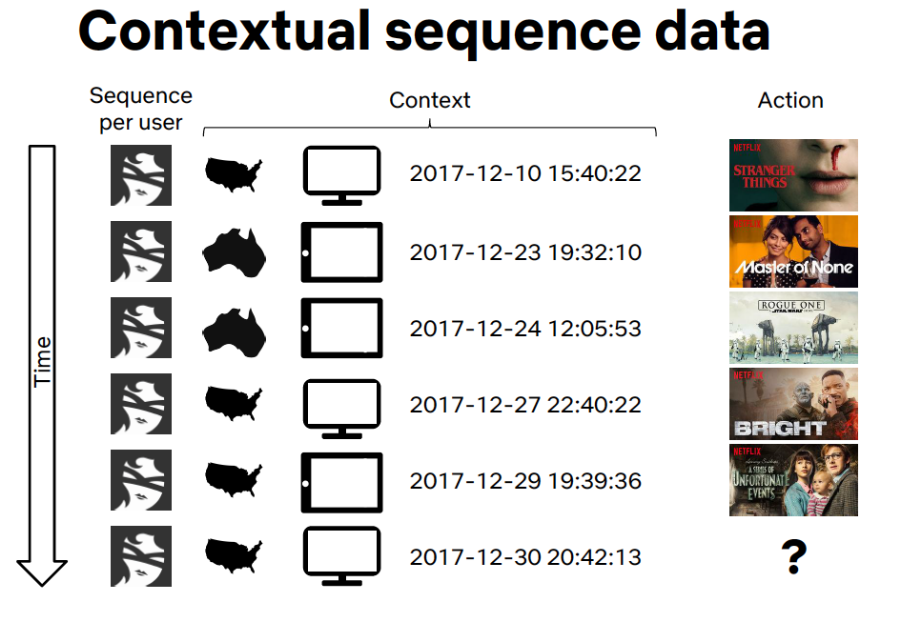


Рис. 3 Context filtering

## Графы в контексте рекомендательных сетей

Рекомендательные системы и графовые методы могут быть эффективно связаны для улучшения качества рекомендаций. Различные объекты (например, пользователи, элементы и атрибуты) явно или неявно связаны между собой и влияют друг на друга посредством различных отношений. Такая организация на основе графов дает преимущества за счет использования потенциальных свойств методов обучения графов (например, случайного блуждания и внедрения сети) для обогащения представлений пользователя и узлов элементов, что является важным фактором для успешных рекомендаций.

Система рекомендаций на основе графов имеет три основных преимущества:

* Масштабируемость: графовые методы позволяют эффективно обрабатывать графы различных размеров и структур, включая очень большие графы. Это делает системы рекомендаций на основе графов масштабируемыми и способными работать с большими объемами данных и пользователей
* Разнообразие моделирования отношений: может включать различные виды взаимодействий, связи между пользователем и элементом, а также контекстные или дополнительные атрибуты.
* Эффективность хранения данных: в отличие от методов, основанных на разреженных матрицах, которые требуют хранения больших объемов данных в памяти, графовые методы могут быть более эффективными в использовании памяти. Это особенно важно для больших графов, где хранение разреженных матриц может потребовать значительных ресурсов.

## Графовые нейронные сети

Графовые нейронные сети (graph neural networks; GNN) – класс моделей глубокого обучения, спроектированный для решения задач обучения на данных, представленных в виде графов. Они позволяют моделировать сложные зависимости между объектами и находить скрытые паттерны, что делает их идеальными для реализации задач в целях данной курсовой работы в сфере государственных закупок, где связи между продуктами, поставщиками и заказчиками играют важную роль.

В данной работе будут рассмотрены два вида моделей: GraphSAGE (Graph Sample and Aggregation) и GCN (Graph Convolutional Network), которые используются для обработки данных на графах и представляют собой методы для извлечения признаков из узлов графа, учитывая их структуру и связи с окружающими узлами.

GraphSAGE представляет собой архитектуру графовой нейронной сети, предложенную в статье "Inductive Representation Learning on Large Graphs" (Hamilton et al., 2017). Основная идея GraphSAGE заключается в том, чтобы извлечь признаки узлов, агрегируя информацию из их окружения. Для каждого узла в графе GraphSAGE выбирает набор соседних узлов (например, с помощью случайного выбора или сэмплирования), а затем агрегирует признаки соседних узлов, чтобы обновить признаки текущего узла. Этот процесс агрегации повторяется на нескольких слоях (обычно несколько раз), чтобы собрать информацию из различных расстояний от каждого узла. GraphSAGE подходит для задач с большими графами и позволяет обучать нейронную сеть на графе, не загружая его полностью в память.

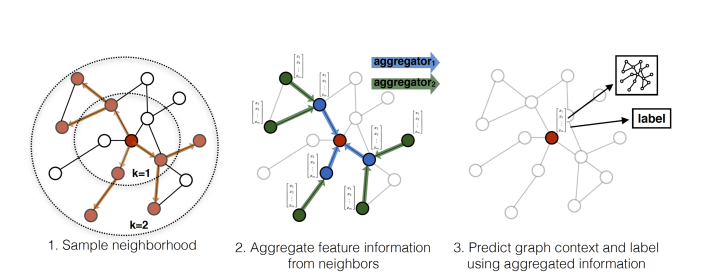
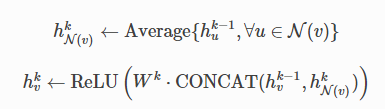


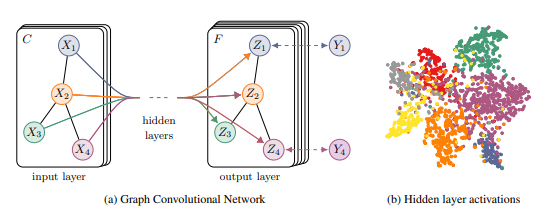
Рис. 4 Визуализация работы GraphSAGE

Данный алгоритм принимает следующую математическую форму:



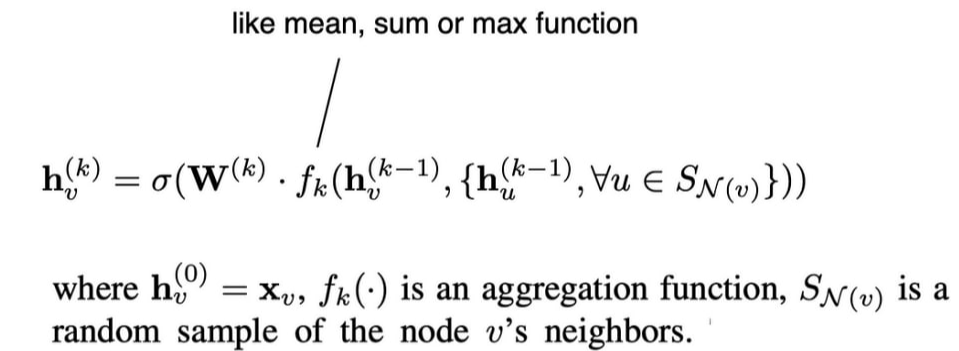
Передача сообщений является направленной: сообщение, отправленное от одного узла u к другому узлу v, не обязательно совпадает с другим сообщением, отправленным от узла v к узлу u в противоположном направлении.

GCN (Graph Convolutional Network) - это архитектура графовой нейронной сети, предложенная в статье "Semi-Supervised Classification with Graph Convolutional Networks" (Kipf & Welling, 2017). GCN основан на сверточных операциях, адаптированных для работы с графами. Он использует локальную информацию о структуре графа и агрегирует признаки соседних узлов, чтобы обновить признаки каждого узла. В отличие от GraphSAGE, GCN работает на основе фиксированных сверточных операций, которые могут быть вычислены эффективно с использованием матричных операций. Однако, GCN имеет ограничение, так как учитывает только признаки непосредственных соседей узла, и может иметь ограниченную способность захвата глобальной информации о графе.

Рис. 5 Многослойная сверточная сеть GCN

На рисунке 5 слева находится схематическое изображение многослойной сверточной сети графов (GCN) для полуконтролируемого обучения с входными каналами C и картами признаков F в выходном слое. Структура графа (ребра показаны черными линиями) является общей для всех слоев, метки обозначаются Yi. Справа: t-SNE (Maaten & Hinton, 2008) визуализация активации скрытого слоя двухслойной GCN, обученной на набор данных Cora (Sen et al., 2008) с использованием 5% меток. Цвета обозначают класс документа.

Данный алгоритм принимает следующую математическую форму:



## Задача Link Prediction

В контексте данной курсовой работы будет использована задача link prediction - задача предсказания отсутствующих ребер в графе на основе уже существующих связей и других свойств графа.

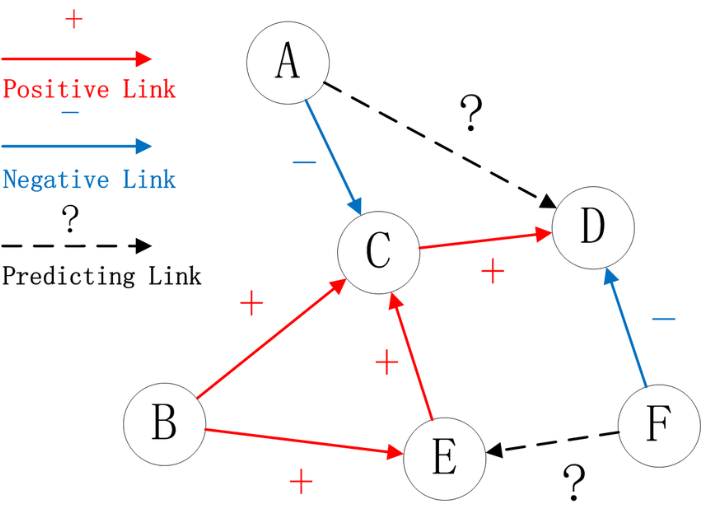


Рис. 6 link prediction

На вход алгоритма link prediction подаются данные о графе, который представляет собой совокупность узлов и связей между ними. Эти данные могут включать информацию о топологии графа (как узлы соединены между собой), атрибуты узлов и ребер, а также дополнительную контекстную информацию. Выбор модели: Для решения задачи link prediction используются различные модели, включая классические методы машинного обучения, но также и графовые нейронные сети, такие как GraphSAGE, GAT, GCN и др. Перед применением модели необходимо извлечь признаки из графа и его узлов. Эти признаки могут включать в себя структурные характеристики графа (например, центральность узлов, плотность графа), атрибуты узлов и ребер (например, возраст пользователя в социальной сети, жанр фильма в рекомендательной системе), а также другие контекстные признаки. После извлечения признаков модель обучается на обучающей выборке, которая включает в себя данные о графе с известными связями и их отсутствующими связями. В процессе обучения модель пытается оптимизировать определенный функционал потерь, который может варьироваться в зависимости от выбранной модели (например, бинарная кросс-энтропия, среднеквадратичная ошибка и т. д.). После завершения обучения модель может быть использована для предсказания отсутствующих связей в графе. Предсказанные значения могут интерпретироваться как вероятности наличия связи между парами узлов. Затем эти предсказанные значения могут быть использованы для принятия решений в конкретной прикладной задаче, такой как рекомендация товаров, прогнозирование взаимодействий и т. д.

# Практическая часть

## Описание датасета

Данные были загружены с сайта государственных закупок zakupki.gov.ru, на котором публикуется информация об актуальных и состоявшихся торгах с конкретными характеристиками работ или нужной продукции, где предприниматели, желающие участвовать в торгах, могут оставить заявку. С помощью специальных фильтров были выгружены уже выполненные контракты федеральных и муниципальных бюджетных учреждений, которые подчиняются требованиям 44-ФЗ и которые проводились в рублях.

В данном датасете имеются:

1. Номер контракта (ID) - contract\_number
2. Наименование заказчика - customer\_name
3. ИНН заказчика (который мы будем использовать в качестве ID) - customer\_INN
4. Предмет конракта - contract\_predmet
5. Объекты закупки - contract\_elems
6. Сумма всего контракта (в один конракт могут входить несколько позиций или же несколько закупок) - contract\_price
7. Специальный код закупки - order\_code
8. Сумма закупки - order\_price
9. Наименование поставщика - provider\_name
10. ИНН поставщика (который мы будем использовать в качестве ID) - provider\_INN

## Предобработка данных

Для начала с помощью функций work\_with\_INN, convert\_price данные были преобразованые в формат int. Функция process\_column необходима для корректного отображения цены закупки: если значение order\_price пустое, а contract\_price содержит в себе какую-то цену, то функция добавляет информацию о ней в пустующую ячейку в солбец order\_price. Был создан новый столбец 'product', в который помещена информация о предмете и объектах закупки. Функция add\_ID необходима для создания уникального кода закупки: 'order\_code' - это уникальный номер всего контракта, поэтому в некоторых строках он одинаковый, так как в контракте могут быть несколько объектов (закупок), но для каждой закупки нужен уникальный номер для удобства дальнейшей работы.

Итоги предобработки выглядят следующим образом:

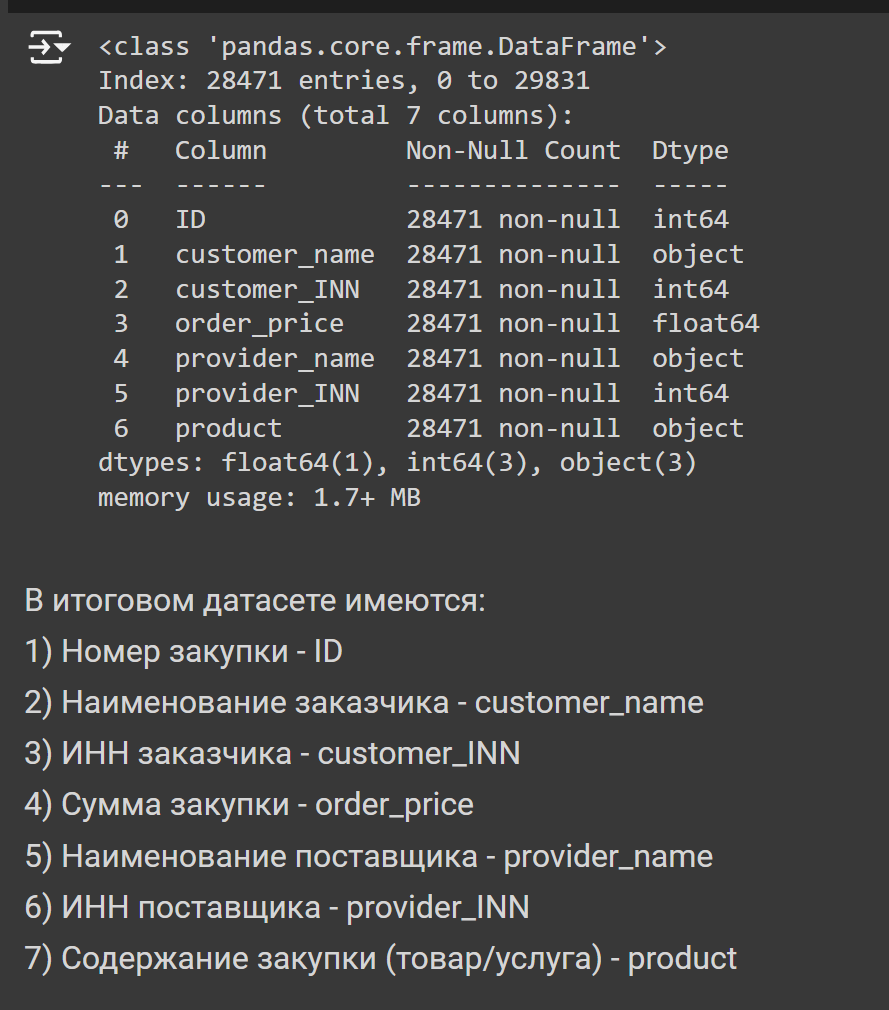


Рис. 7 Итоги предобработки датасета

## Разделение датасета

Для получения ребер графа взяты столбцы 'ID', 'customer\_INN', 'provider\_INN', 'order\_price'.

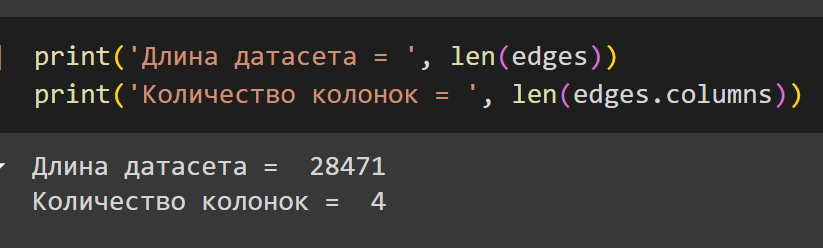


Рис. 8 Ребра будущего графа

Для получения вершин графа для начала использована фунция extract\_product\_info, которая выписывает все уникальные ИНН как заказчиков, так и поставщиков и ищет для них всю информацию о продуктав, с которыми они встречались. Кол-во уникальных ИНН поставщиков - 2026, а заказщиков - 1862.

## Обработка информации о продуктах

Необходимо обработать информацию о товарах и продуктах контрактов. Функция preprocess\_text\_numba удаляет все символы кроме русских букв, приводит все к нижнему регистру, удаляет стоп-слова (stopwords.words('russian')), выполняется лемматизация слов с помощью pymorphy2.MorphAnalyzer().

Функция process\_words\_in\_column обрабатывает строки в столбце следующим образом: если в строке менее 6 слов, то добавляются самые часто встречающиеся слова, чтобы длина строки стала равной 6; если слов больше или равно 6, то выбирается топ 6 самых встречающихся. В результате, в каждой строке будет содержаться 6 слов в виде строки.

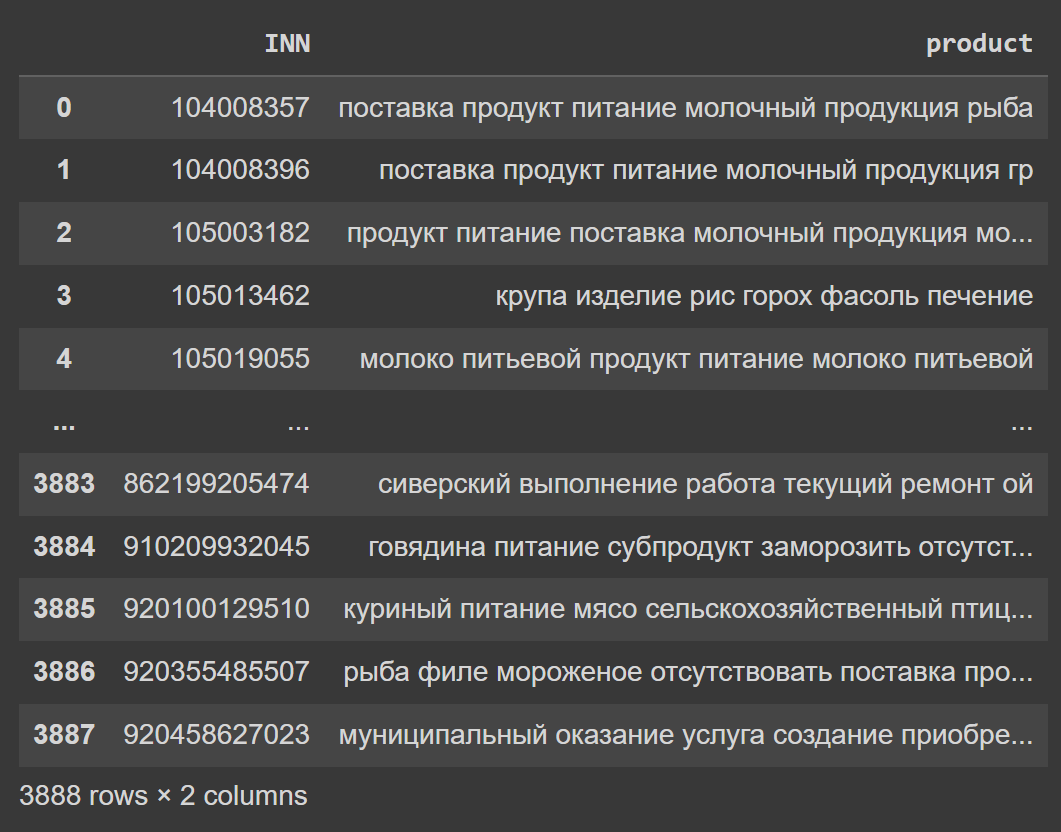


Рис. 9 Результат обработки информации о продуктах

## Векторизация

Свойства графа не могут содержать данные в виде строкового формата, поэтому столбец 'product' должен быть векторизирован. Для этого использован TF-IDF - алгоритм, который вычисляет числовые представления слов в тексте, учитывая частотность их употребления и уникальность в контексте документов. Он состоит из двух основных компонентов: частоты термина (TF) и обратной частоты документа (IDF).

*Term Frequency (TF):*

* Отражает частоту появления термина в документе. Определяет, насколько важно слово в конкретном документе.
* Вычисляется как отношение числа вхождений термина к общему числу слов в документе.

*Inverse Document Frequency (IDF):*

* Отражает важность термина в контексте всего корпуса документов. IDF понижает вес часто встречающихся слов и повышает вес редко встречающихся слов.
* Вычисляется как логарифм отношения общего числа документов к числу документов, содержащих данный термин.

Вычисление TF-IDF:

TF-IDF для термина t в документе d вычисляется как произведение TF и IDF:

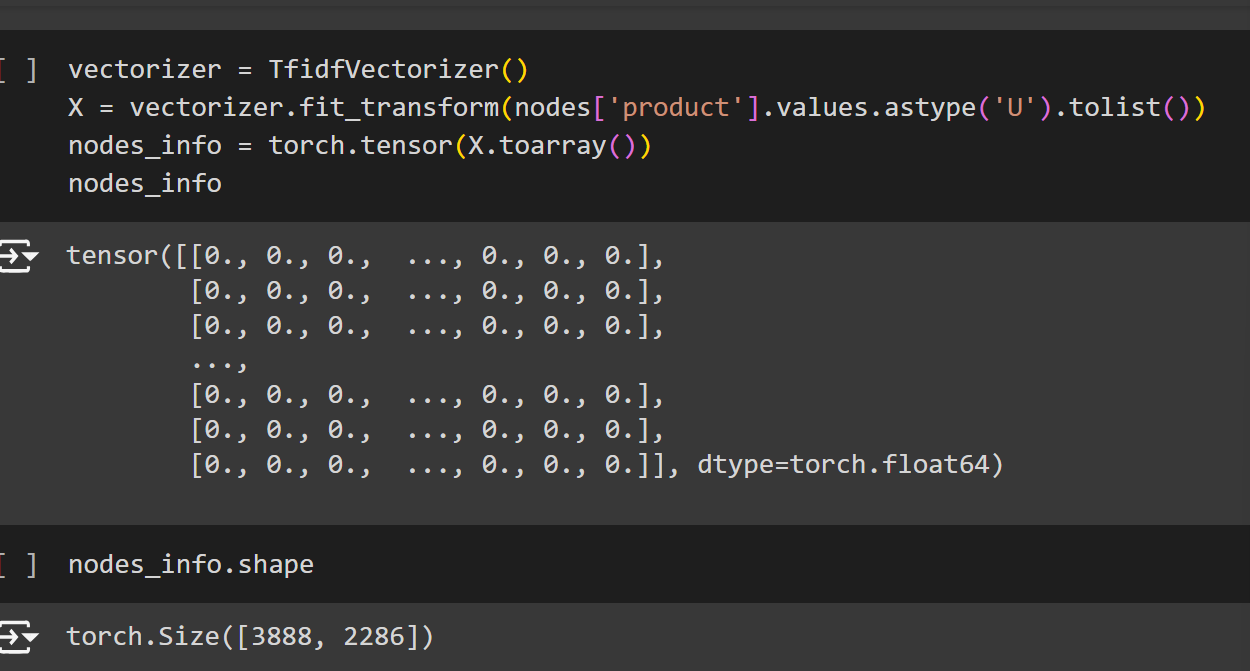


Рис. 10 Реализация векторизации

## Создание графа

Для начала создается отсортированный список всех уникальных ИНН заказчиков и поставщиков. Каждому значению присваивается порядковый номер.

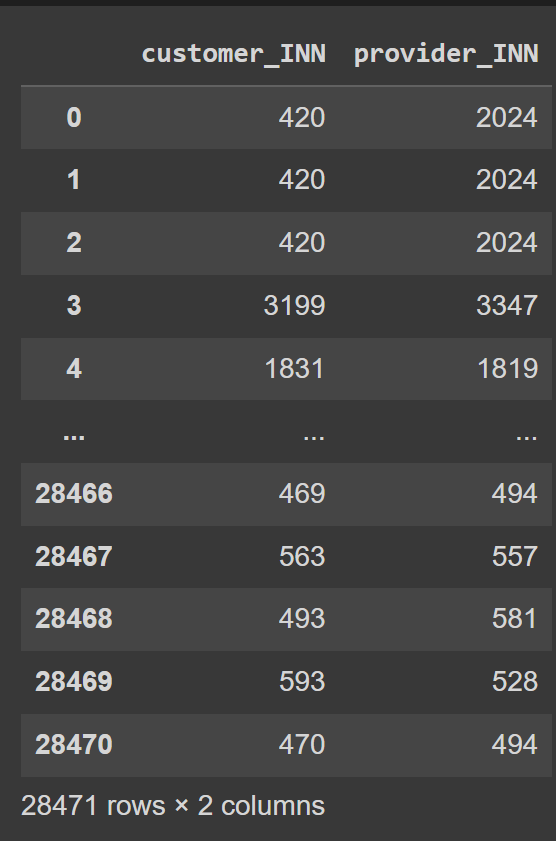
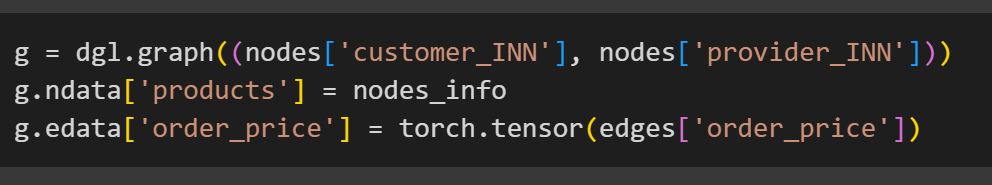


Рис. 11 Формирование вершин графа

Для создания графа и дальнейшей работы с ним используется библиотека DGL - фрэймворк для построения моделей глубокого обучения на графах, в котором граф представляет из себя отдельную графовую структуру, похожу на структуру графа из networkx. Ключевые свойства:

* простой в использовании
* высокопроизводительный
* масштабируемый

  
Рис. 12 Создание графа с помощью DGL

Вершины графа - ИНН поставщиков и заказчиков. Команда g.ndata['products'] = nodes\_info используется для присвоения данных узлов графа, хранящихся в переменной nodes\_info, под ключом 'products'. Команда g.edata['order\_price'] = torch.tensor(edges['order\_price']) используется для присвоения данных рёбер графа, хранящихся в переменной edges['order\_price'], под ключом 'order\_price'.

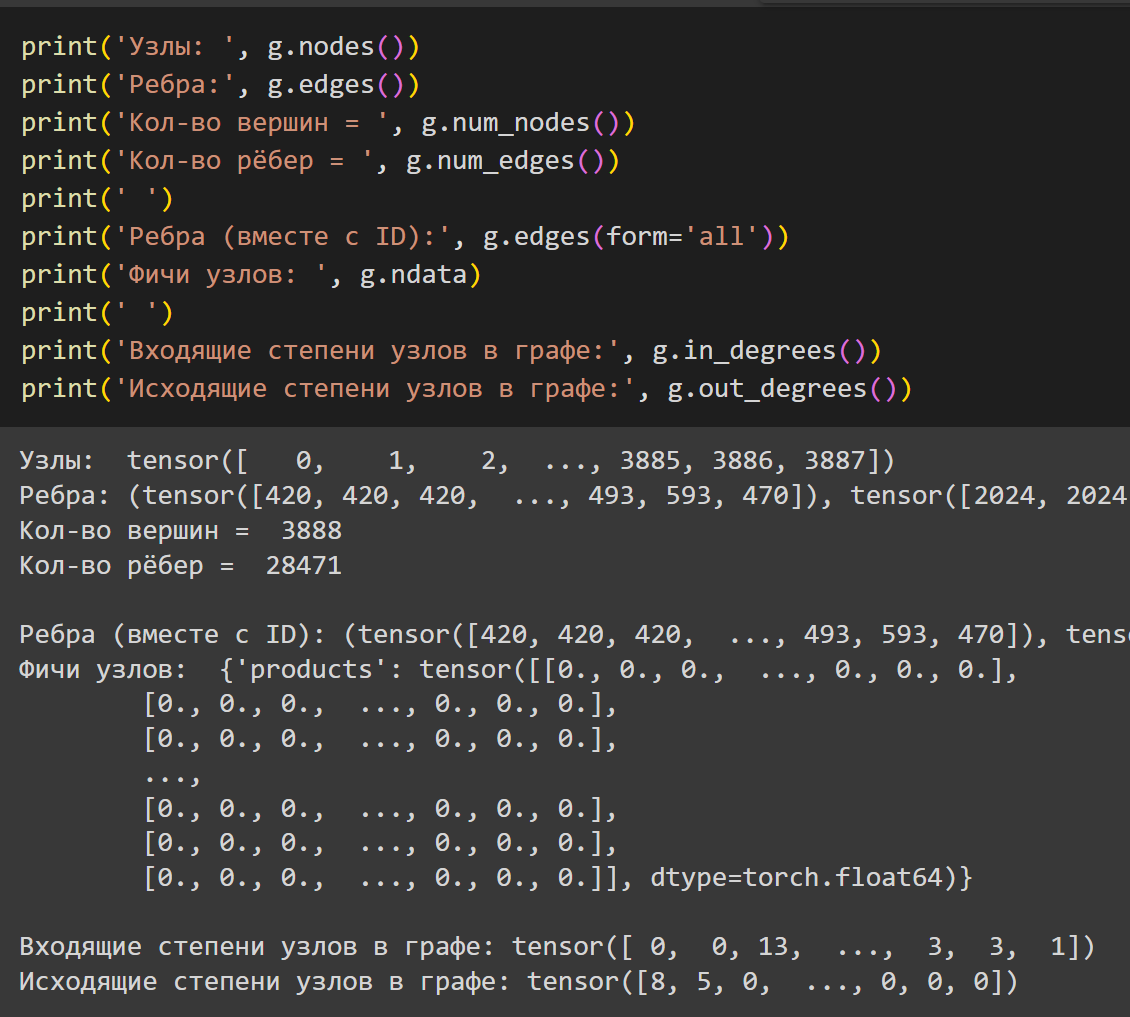


Рис. 13 Информация о графе

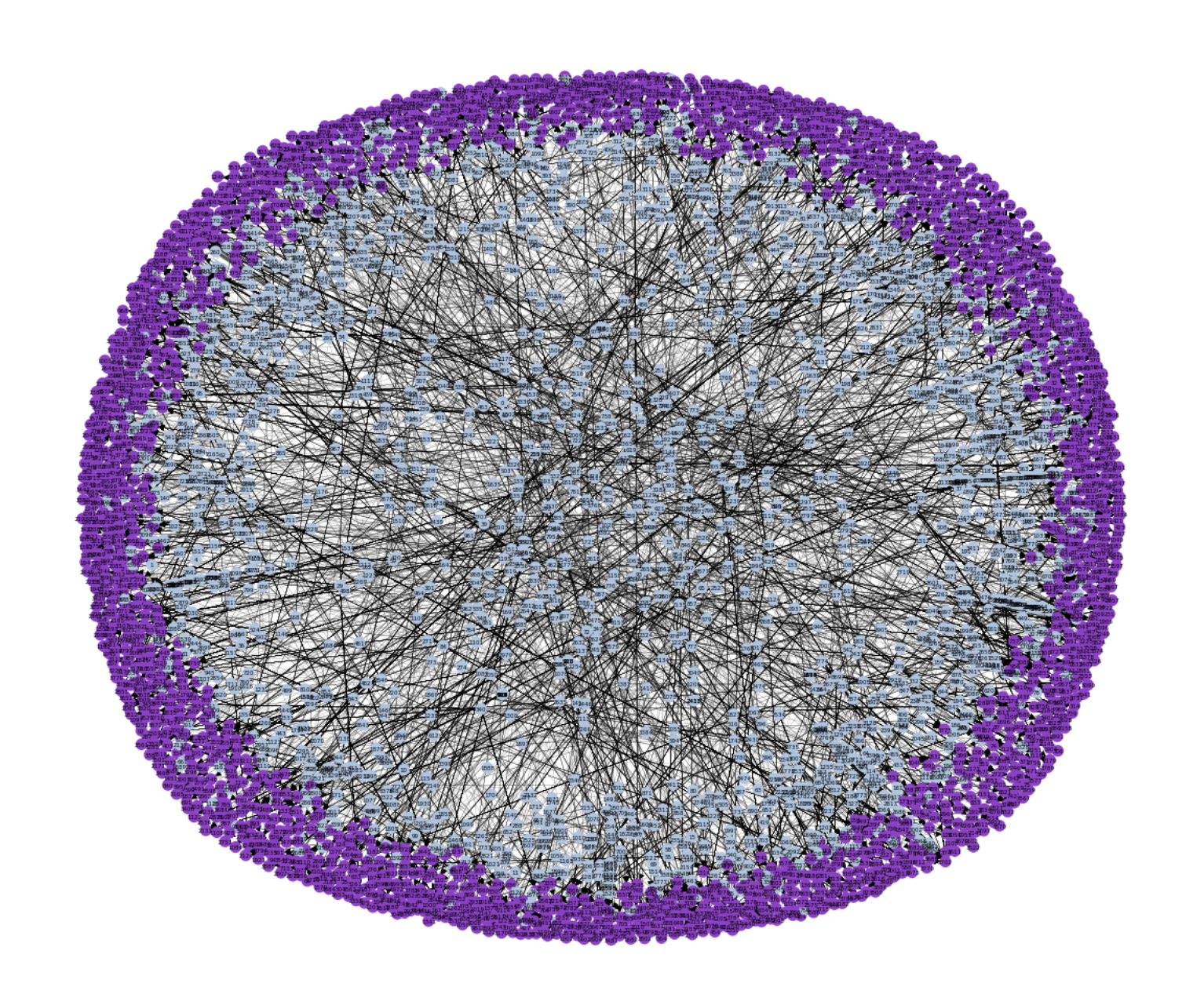


Рис. 14 Визуализация графа

Для визуализации графа использована библиотека networkx. Customer\_INN - вершины, помеченные голубым цветом, provider\_INN - фиолетовым цветом.

## Разделение на тренировочную и тестовую выборки

Функция split\_graph выполняет разбиение его рёбер на тренировочный и тестовый наборы, создавая также позитивные и негативные примеры рёбер. Доля рёбер, которые попадут в тестовый набор = 10%.

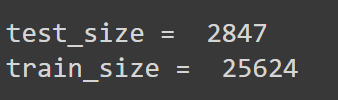


Рис. 15 Размеры тренировочной и тестовой выборок

Для создания именно негативных примеров ребер для начала создается adj - разреженная матрица смежности графа, затем adj\_negative - матрица, содержащая 1 в местах отсутствующих рёбер и 0 в местах существующих рёбер. Negative\_cust и negative\_prov - индексы узлов, которые формируют негативные рёбра (не существующие в графе рёбра). В итоге получаются:

* train\_g - тренировочный граф, созданный путём удаления тестовых рёбер из исходного графа
* test\_g - тестовый граф, созданный путём удаления тренировочных рёбер из исходного графа
* train\_pos\_g - граф, содержащий позитивные тренировочные рёбра
* train\_negative\_g - граф, содержащий негативные тренировочные рёбра
* test\_pos\_g - граф, содержащий позитивные тестовые рёбра
* test\_negative\_g - граф, содержащий негативные тестовые рёбра

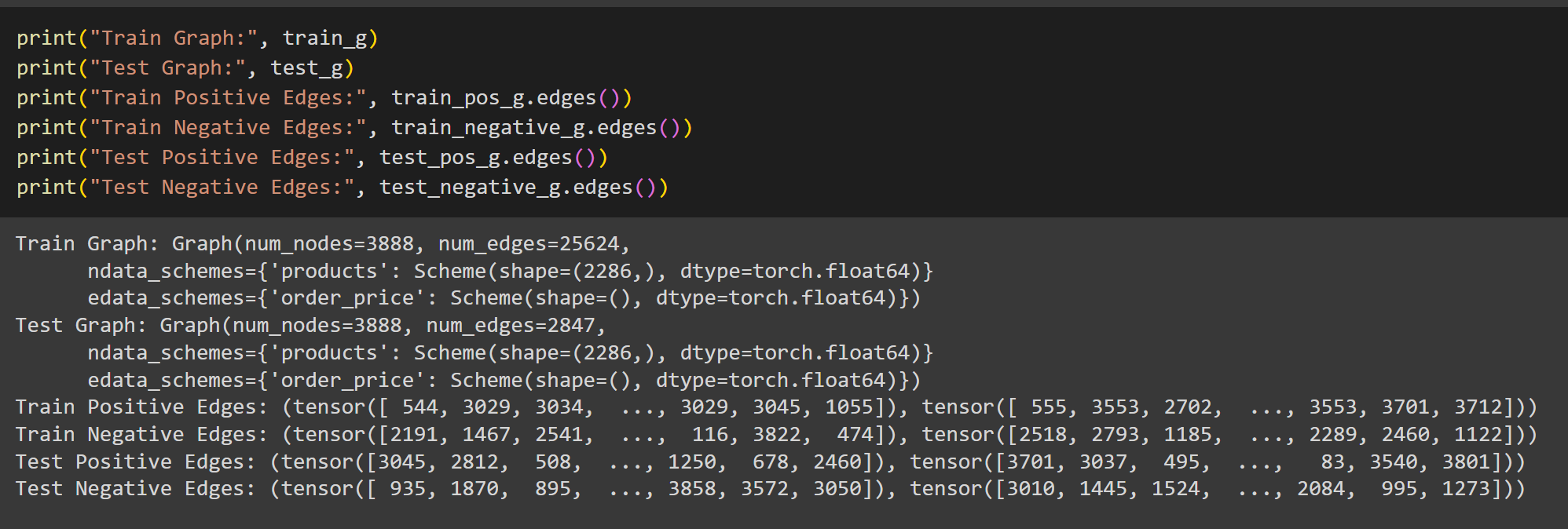


Рис. 16 Параметры после применения split\_graph

## Используемые метрики

Для вычисления метрик эффективности прописаные функции:

* compute\_loss вычисляет бинарную кросс-энтропию для заданных положительных и отрицательных оценок
* compute\_accuracy вычисляет точность классификации для положительных и отрицательных оценок (pred\_labels создаются на основе порога 0.5)
* compute\_f1 - вычисляет F1-меру для положительных и отрицательных оценок
* compute\_auc - вычисляет AUC (Area Under Curve), то есть площадь под ROC-кривой, которая показывает качество ранжирования модели для положительных и отрицательных оценок.
* compute\_conf\_matrix - вычисляет матрицу ошибок (confusion matrix) для положительных и отрицательных оцено

## DotPredictor

Класс DotPredictor является наследником nn.Module и предназначен для предсказания наличия ребра между двумя узлами. g.local\_scope() создаёт локальную копию данных узлов и рёбер графа. Все изменения данных узлов и рёбер внутри этого блока будут локальными и не повлияют на исходный граф. Далее g.ndata['h'] = h присваивает входные представления узлов (node embeddings) h атрибуту 'h' всех узлов графа g. Функция g.apply\_edges применяет другую функцию ко всем рёбрам в графе. В данном случае используется fn.u\_dot\_v('h', 'h', 'score'), что выполняет скалярное произведение между представлениями начальных (u) и конечных (v) узлов каждого ребра.'score' - атрибут, в который сохраняются результаты скалярного произведения.

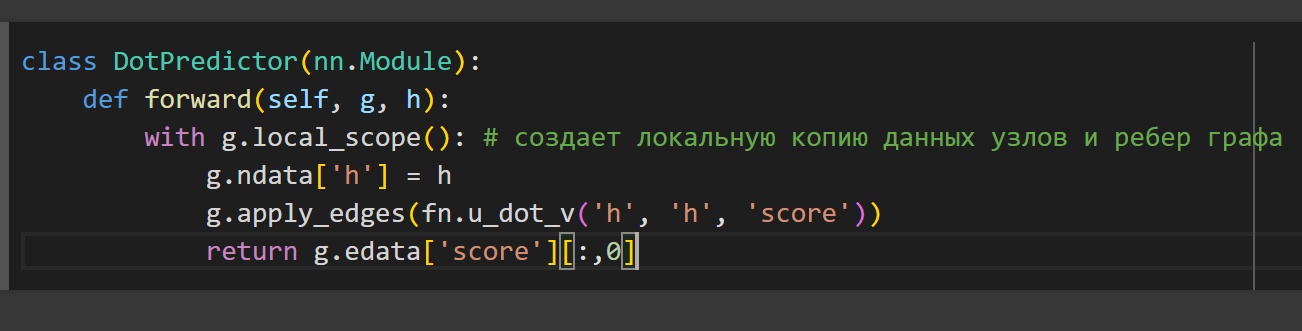


Рис. 17 DotPredictor

## Модель GraphSAGE

Первая модель GraphSAGE с двумя слоями SAGEConv с методом агрегации "mean" и функцией активации ReLU.

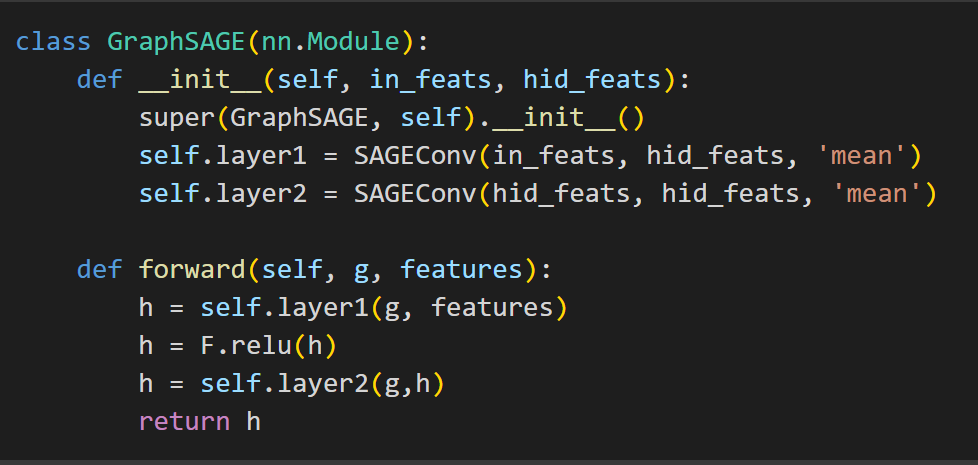
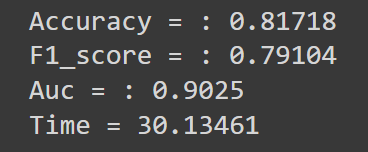
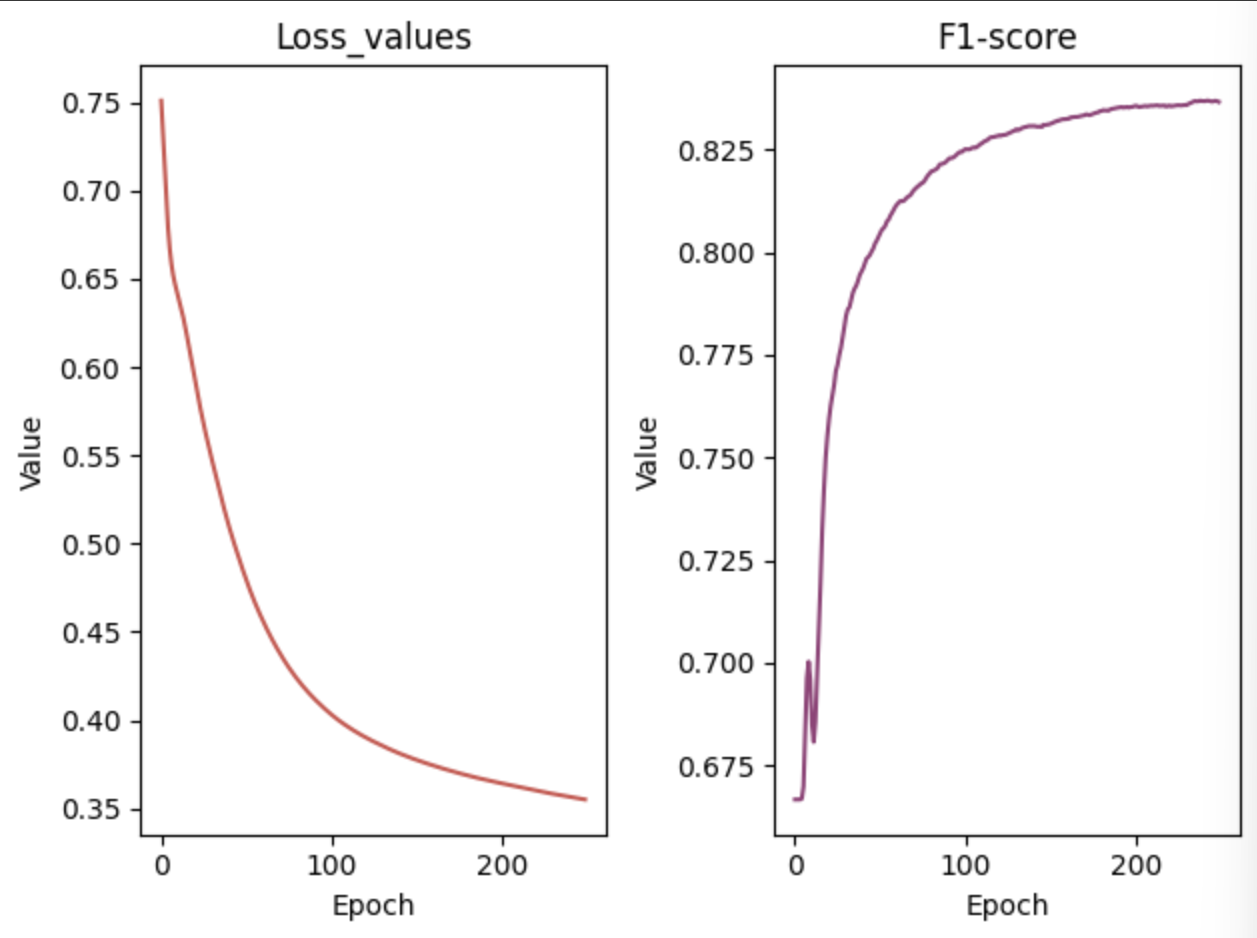


Рис. 18 Архитектура GraphSAGE

Метрики выглядят следующим образом:





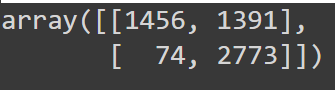


Рис. 19 Метрики GraphSAGE

TP = 1456: истинно позитивных предсказаний.  
FP = 1391: ложно позитивных предсказаний.  
FN = 74: ложно негативных предсказаний.  
TN = 2773: истинно негативных предсказаний.

- Высокое количество TN: Модель правильно классифицирует большинство отрицательных примеров

- Достаточно высокое TP: Модель правильно классифицирует много положительных примеров

- Однако FP тоже высокое: Модель часто классифицирует негативные предсказания как позитивные

- Низкое FN: Модель редко ошибается, классифицируя отрицательные примеры, как положительные

Во всех моделях была использована кросс-валидация (cross-validation) - это метод оценки производительности модели машинного обучения, который позволяет оценить, насколько успешно модель обобщается на новые данные, которые она ранее не видела.

Метрики

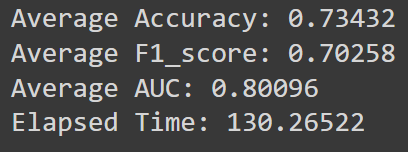


Рис. 19 Метрики кросс-валидации GraphSAGE

Средняя точность = 0.73432 указывает на то, что в среднем модель правильно классифицировала около 73.432% примеров на всех фолдах. То есть, примерно 3 из 4 примеров были правильно классифицированы.

Среднее значение F1-меры = 0.70258 показывает среднюю гармоническую среднюю точность и полноту модели на всех фолдах. Это значение указывает на сбалансированность модели между точностью и полнотой.

Среднее значение AUC = 0.80096 представляет собой среднюю площадь под кривой ROC (Receiver Operating Characteristic) для всех фолдов. Это показывает, насколько хорошо модель разделяет классы. В данном случае модель это делает достаточно хорошо.

## Модель GCN - 2 слоя

Вторая модель GCN с двумя слоями GraphConv, с функцией активации leaky ReLU, слоем нормализации и слоем dropout для регуляризации и предотвращения возможности переобучения.

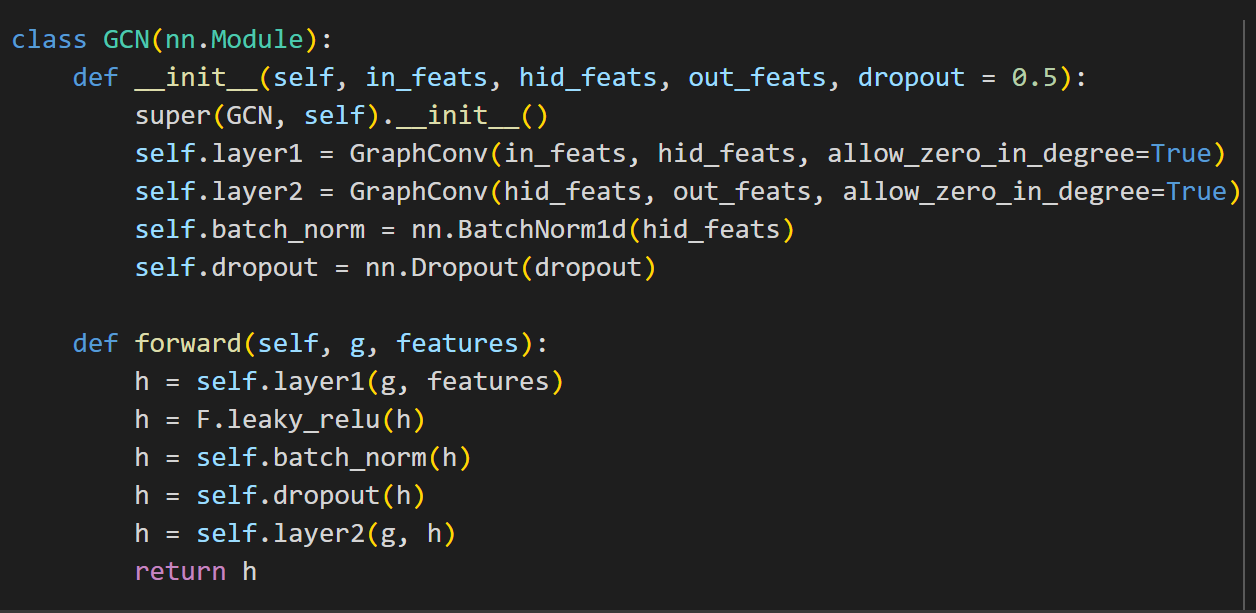
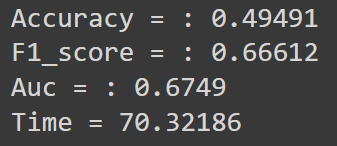
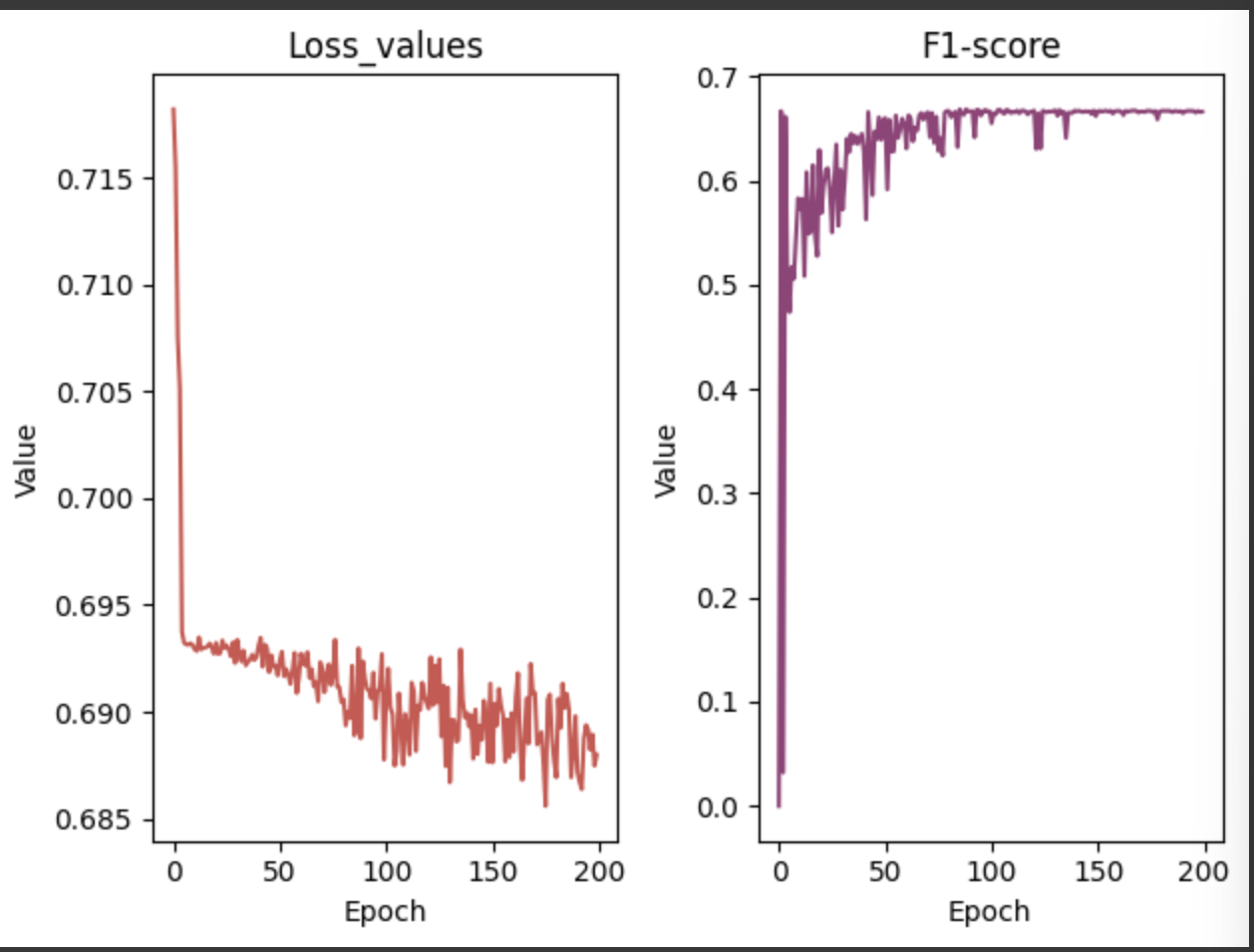


Рис. 20 Архитектура GCN\_2





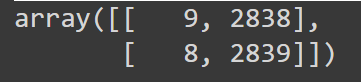


Рис. 21 Метрики GCN\_2

TP = 9: истинно положительных предсказаний.  
FP = 2838: ложно позитивных предсказаний.  
FN = 8: ложно негативных предсказаний.  
TN = 2839: истинно позитивных предсказаний.

- Высокое количество FP: Модель ошибочно классифицирует много отрицательных примеров как положительные

- Высокое количество TN: Модель правильно классифицирует большинство отрицательных примеров

- Низкое количество TP: Модель не может предсказывать положительные примеры

- Низкое FN: Модель практически не ошибается, классифицируя положительные примеры как отрицательные, и наоборот.

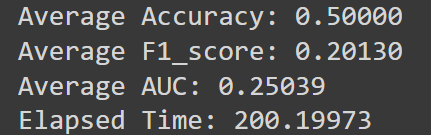


Рис. 22 Метрики кросс-валидации GCN\_2

Потери стабилизируются около 0.693, что очень близко к значению для случайного угадывания в бинарной классификации

Точность в каждой эпохе и каждом фолде стабилизируется около 0.499-0.4995, что указывает на случайное угадывание. Средняя точность по всем фолдам равна 0.50000, что является показателем случайного угадывания в бинарной классификации.

Средний F1-Score очень низкий — 0.2013. Это говорит о том, что модель плохо различает классы и не умеет правильно предсказывать положительные и отрицательные классы.

Средний AUC равен 0.25039, что значительно ниже 0.5 и указывает на плохое качество классификации. AUC ниже 0.5 означает, что модель хуже, чем случайное угадывание.

## Модель GCN - 3 слоя

Третья модель аналогично прошлой, но в ней используется 3 слоя GraphConv

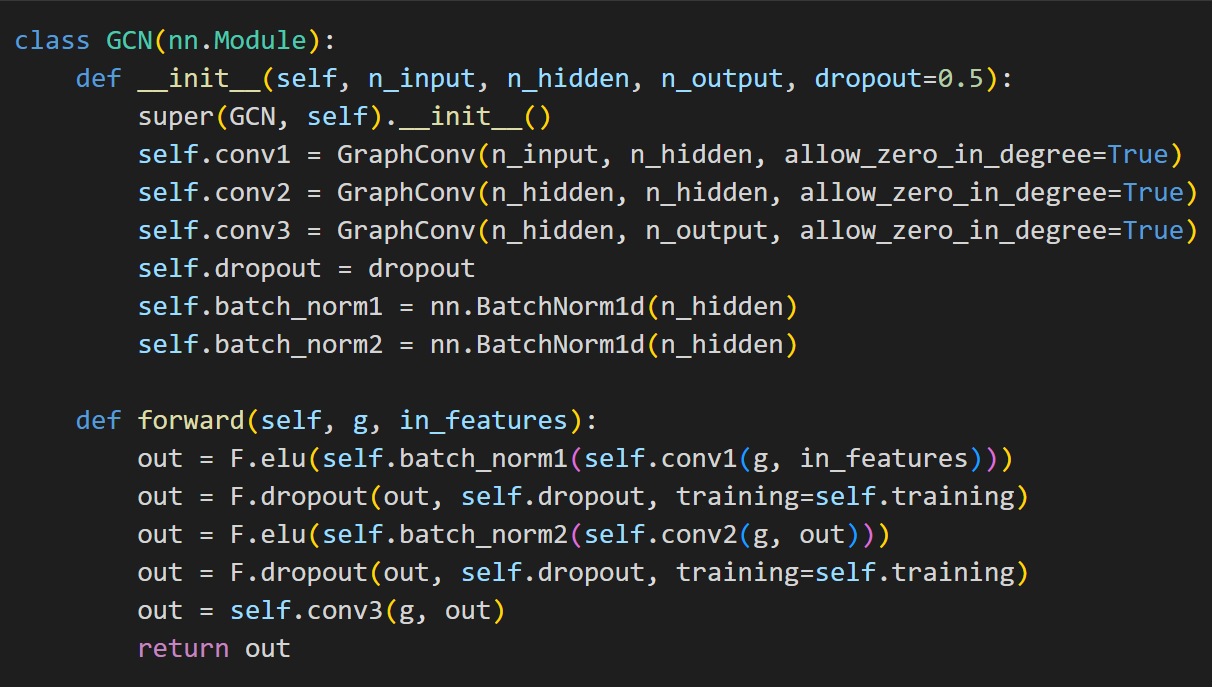
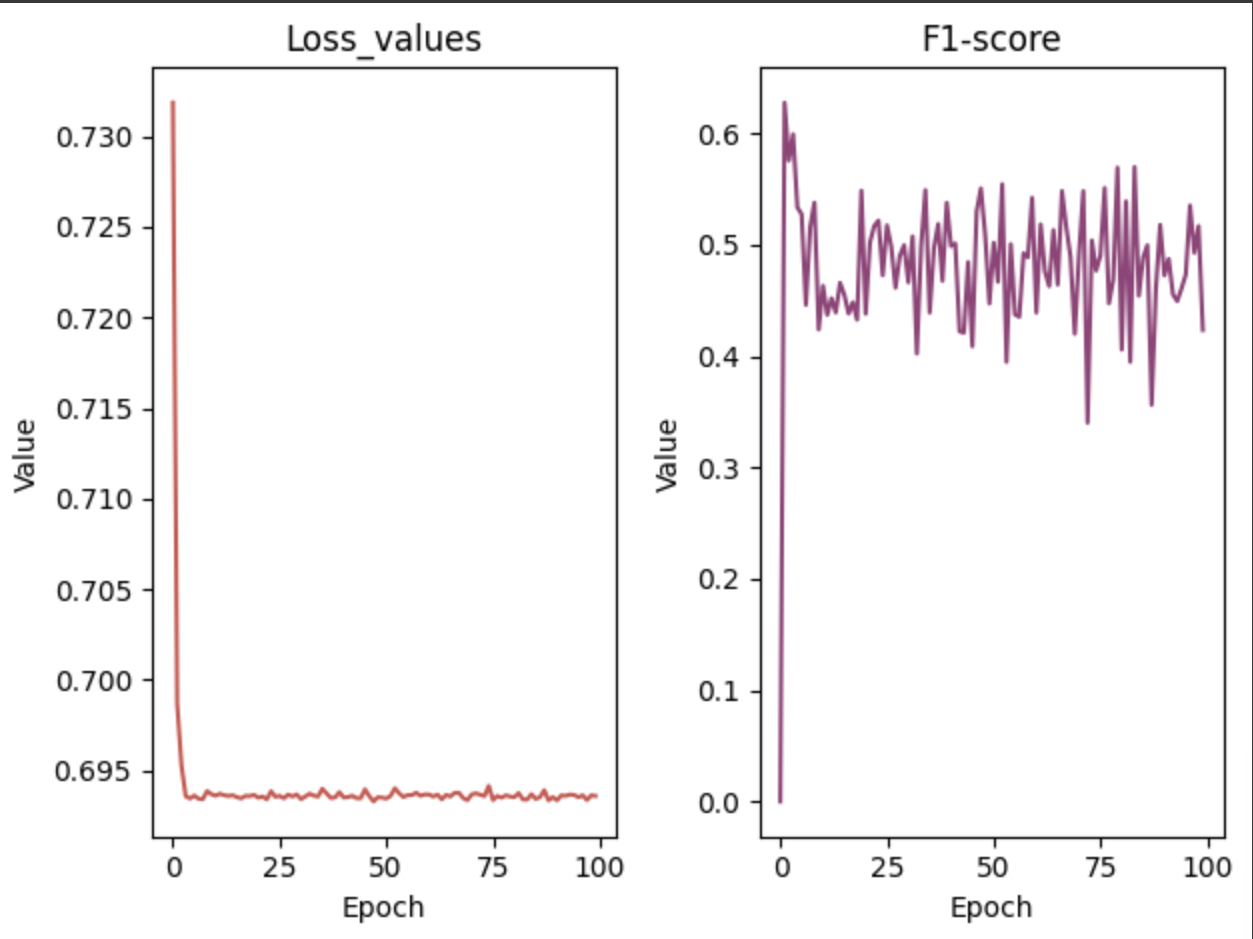


Рис. 23 Архитектура GCN\_3





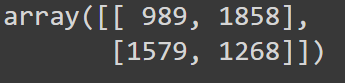


Рис. 24 Метрики GCN\_3

TP = 989: истинно позитивных предсказаний

FP = 1858: ложно позитивных предсказаний.  
FN = 1579: ложно негативных предсказаний.  
TN = 1268: истинно негативных предсказаний.

- Высокое количество FP и FN: Модель ошибочно классифицирует много отрицательных примеров как положительные и наоборот

- Высокое количество TN: Модель правильно классифицирует достаточно много отрицательных примеров

- Низкое количество TP: Модель плохо предсказывает положительные примеры

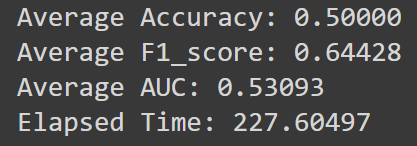


Рис. 25 Метрики кросс-валидации GCN\_3

Accuracy = 0.50000 показывает, что модель правильно классифицирует примерно половину примеров. Это не лучше случайного угадывания для задачи бинарной классификации.

F1-score значение 0.64428 указывает на более высокую производительность по сравнению с точностью, особенно учитывая сбалансированный вклад точности и полноты. F1-score выше, чем точность, может указывать на то, что модель лучше работает с одним из классов.

AUC значение 0.53093 немного выше 0.5, что указывает на чуть лучшую производительность, чем случайное угадывание.

## Модель с собственным SAGEConv

SAGEConv - это класс, реализующий самостоятельную свертвку для модели GraphSAGE.

* Принимаются входные данные: граф g и признаки h.
* Каждый узел отправляет свои признаки своим соседям (операция message\_func=fn.copy\_u("h", "m")).
* Каждый узел принимает признаки своих соседей и вычисляет их среднее (операция reduce\_func=fn.mean("m", "h\_N")).
* update\_all сообщает, что нужно инициировать сообщение и сократить функции для всех узлов и ребер.
* Применяется линейное преобразование, нормализацию и dropout к объединенным признакам.
* Возвращаются обработанные признаки узлов.

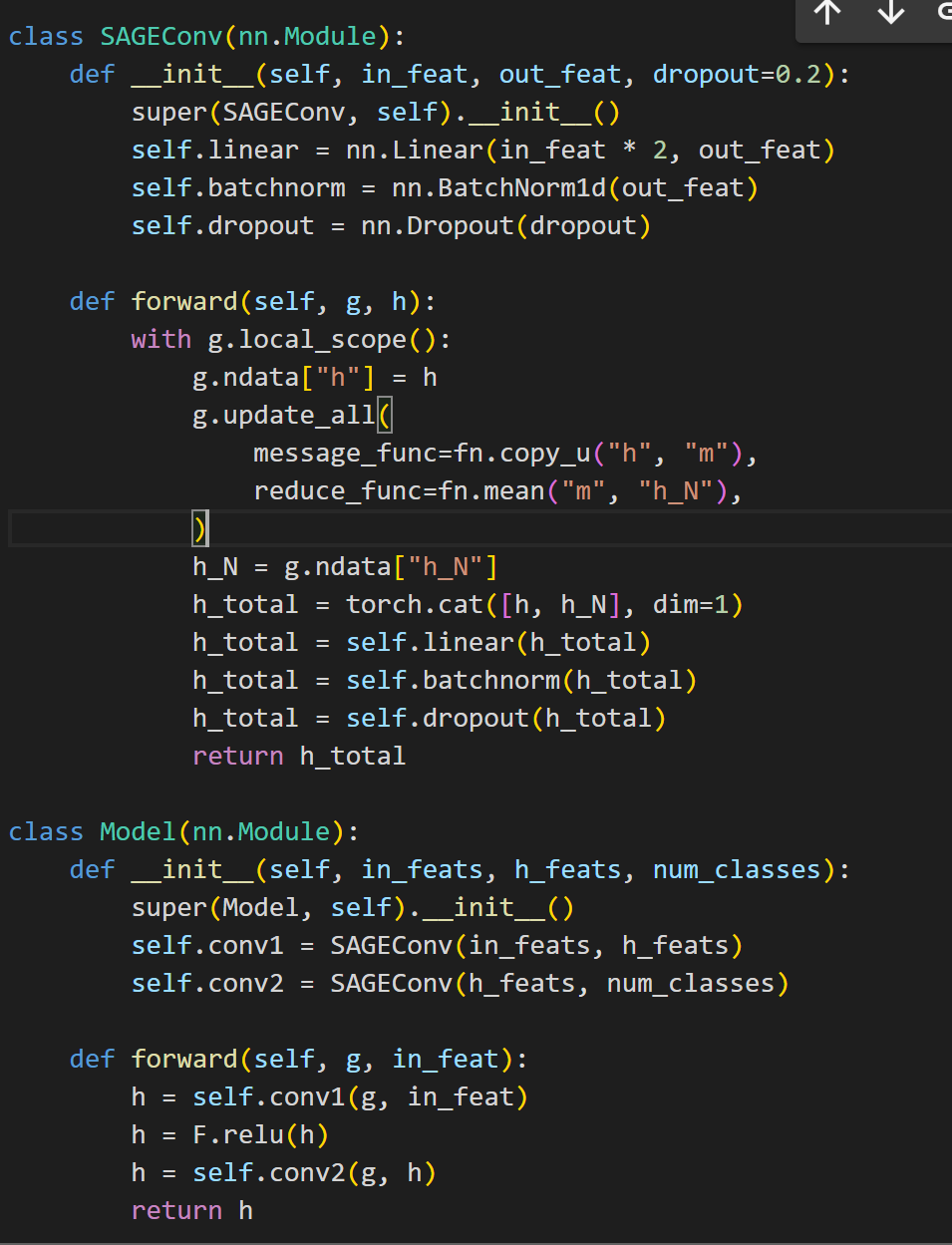
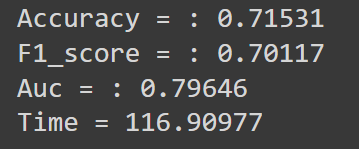
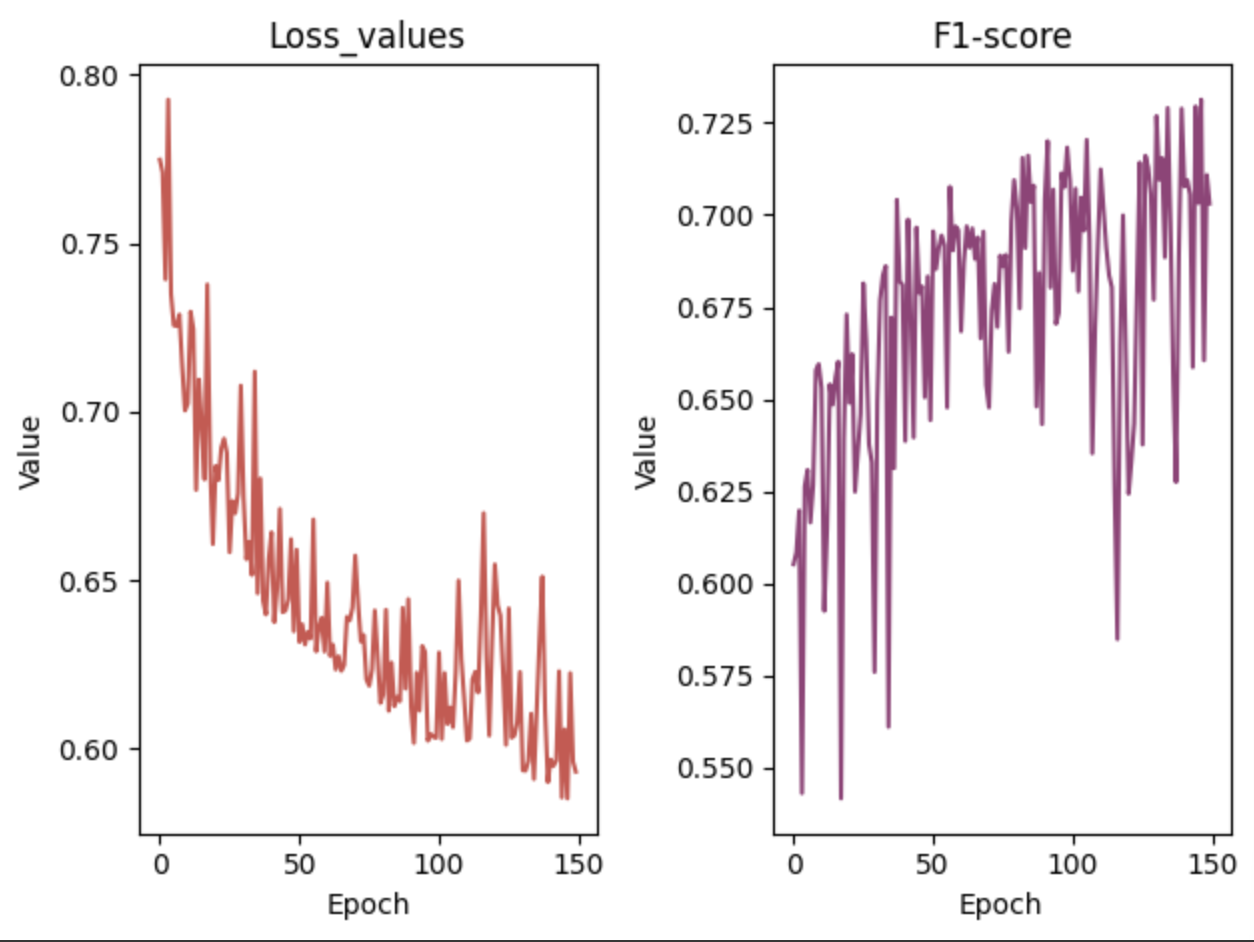


Рис. 26 Архитектура SAGEConv





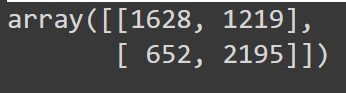


Рис. 27 Метрики SAGEConv

TP = 1628: истинно позитивных предсказаний  
FP = 1219: ложно позитивных предсказаний.  
FN = 652: ложно негативных предсказаний.  
TN = 2195: истинно негативных предсказаний.

- Высокое количество TP и TN: Модель правильно классифицирует много положительных и отрицательных примеров (однако отрицательные примерны классифицирует лучше)

- Высокое количество FP: Модель ошибочно классифицирует много отрицательных примеров как положительные

- Низкое количество FN: Модель правильно классифицирует отрицательные примеры.

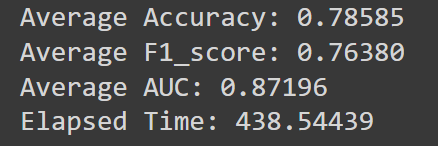


Рис. 28 Метрики кросс-валидации SAGEConv

Accuracy = 0.78585 показывает, что модель правильно классифицирует примерно 79% примеров.

F1-score значение 0.7638, что значит, что модель достаточно хорошо сбалансирована между точностью и полнотой

AUC значение 0.87196 указывает на хорошую способность модели различать классы

## Модель с собственным WeightedSAGEConv

Класс WeightedSAGEConv представляет собой слой графовой свёртки для модели GraphSAGE с учетом весов ребер графа.

* Принимает входные данные: граф g, признаки h узлов и веса w ребер.
* Каждому ребру графа присваивается вес из тензора весов w.
* Выполняется графовая операция - каждый узел отправляет свои признаки своим соседям, умноженные на веса ребер (операция message\_func=fn.u\_mul\_e("h", "w", "m")).
* Каждый узел принимает признаки своих соседей и вычисляет их среднее (операция reduce\_func=fn.mean("m", "h\_N")).
* Применяется линейное преобразование, нормализация и dropout к объединенным признакам.
* Возвращаются обработанные признаки узлов.

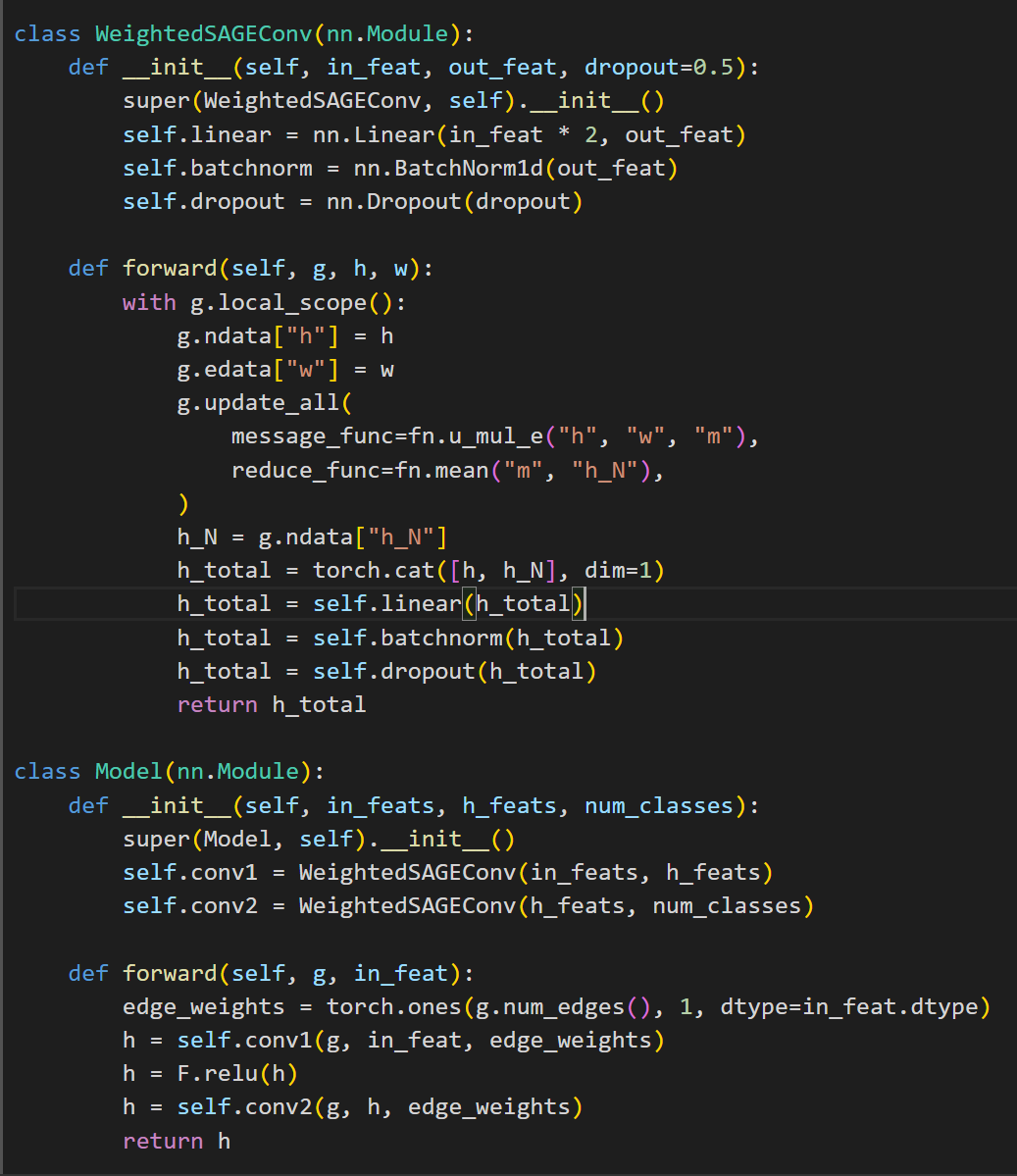
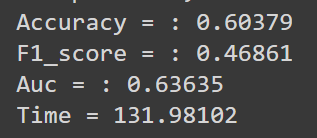
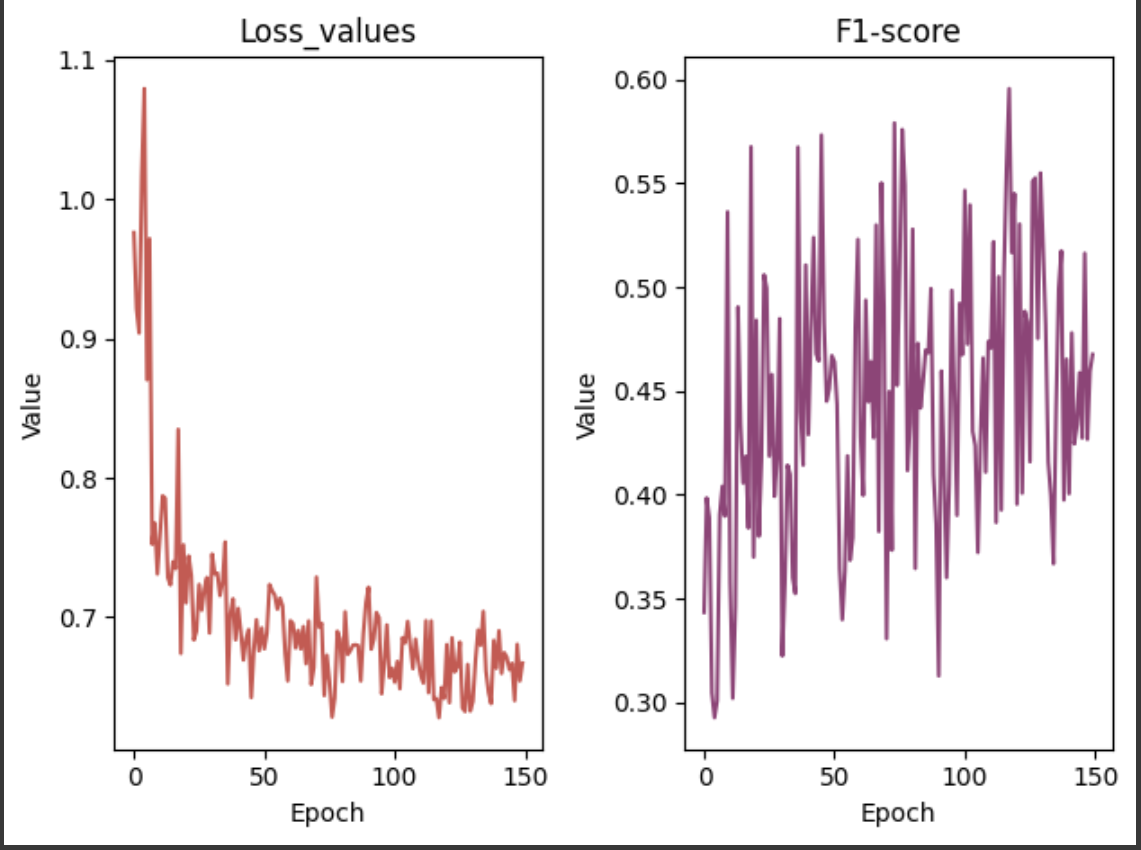


Рис. 29 Архитектура WeightedSAGEConv





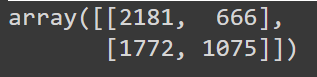


Рис. 30 Метрики WeightedSAGEConv

TP = 2181: истинно позитивных предсказаний  
FP = 666: ложно позитивных предсказаний.  
FN = 1772: ложно негативных предсказаний.  
TN = 1075: истинно негативных предсказаний.

- Высокое количество TP: Модель правильно классифицирует много положительных примеров

- Умеренное количество TN: Модель нормально классифицирует отрицательные примеры

- Высокое количество FN: Модель ошибочно классифицирует много положительных примеров как отрицательные

- Низкое количество FP: Модель мало ошибается, классифицируя негативные примеры как позитивные.

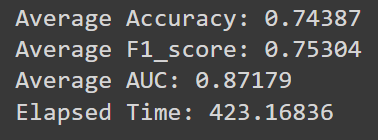


Рис. 31 Метрики кросс-валидации WeightedSAGEConv

Accuracy = 0.74387 показывает, что модель правильно классифицирует примерно 74% примеров.

F1-score значение 0.75304, что значит, что модель достаточно хорошо сбалансирована между точностью и полнотой

AUC значение 0.87179 указывает на хорошую способность модели различать классы

## 1.15 Результаты

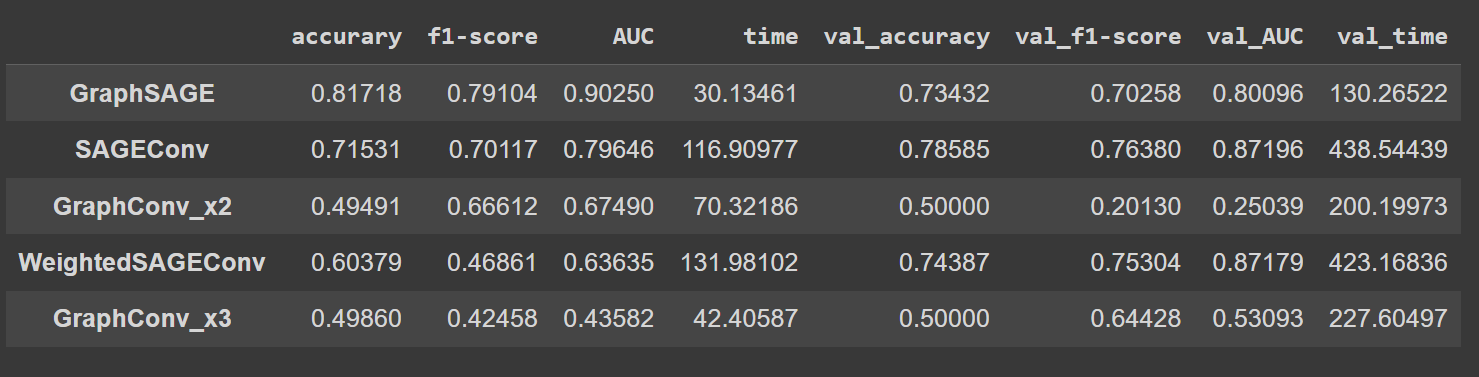


Рис. 32 Результаты

Была сформирована итоговая таблица, в которую помещены результаты по оцениваемым метрикам, а также информация по поводу кросс-валидации. Строчки таблицы отсортированы по значению метрики F1-score. Наилучшая модель получилась GrapgSAGE – в ней самые большие метрики accuracy, f1-score, AUC, наименьшее время выполнения и хорошие значения на кросс-валидации.

## 1.16 Построение рекомендаций

Функция info\_customer() генерирует случайное число из списка ИНН заказчиков. По этому значению из большого датафрейма all\_data достается наименование этого заказчика и соответствующая вершина в графе. Функция get\_from\_history ищет всех поставщиков, с которыми контактировал выбранный заказчик, далее также достаются наименование и ИНН поставщиков. Основная функция rec\_system принимает на вход количество рекомендаций, которое необходимо получить в конце. Создает граф g\_without без вершины выбранного заказчика. Так как в прошлом пункте определено, что наиболее эффективной моделью оказалась GraphSAGE, то с ее помощью получены вероятности для всех возможных поставщиков. prob\_edges объединяет значения вероятностей и сами ребра графа. Далее значения сортируются в порядке убывания.

Так, для примера был случайно выбран заказчик со следующей информацией:

* ИНН случайно выбранного заказчика: 9703042554
* Наименование выбранного заказчика: ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ ГОРОДА МОСКВЫ "УПРАВЛЕНИЕ ОБЕСПЕЧЕНИЯ ДЕЯТЕЛЬНОСТИ КОМПЛЕКСА ГРАДОСТРОИТЕЛЬНОЙ ПОЛИТИКИ И СТРОИТЕЛЬСТВА ГОРОДА МОСКВЫ"
* Вершина в графе = 3426

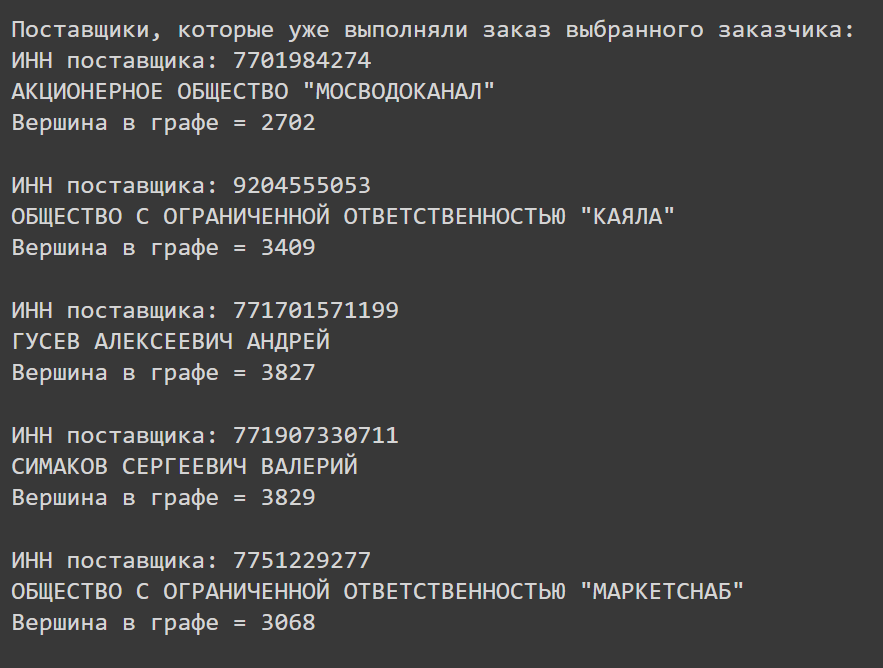


Рис. 33 Результат работы get\_from\_history

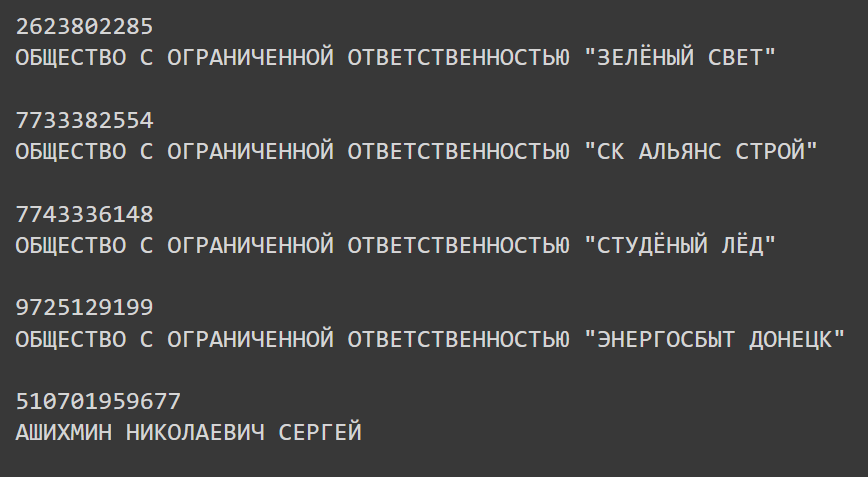


Рис. 34 Найденные рекомендации

Из информации о прошлых поставщиках видно, что запросы на указания услуг связаны свкорее всего с энергоресурсами, водоснабжением и т.д.

На выходе получены рекомендации о 5 поставщиках, которые брали похожие заказы

# Заключение

В ходе выполнения курсовой работы были изучены основные концепции и методы, связанные с графами, рекомендательными системами и графовыми нейронными сетями. Это включало в себя изучение теоретических основ, методов анализа графовых данных, а также основных принципов работы рекомендательных систем. Были рассмотрены два основных алгоритма такие, как GraphSAGE и GCN, применяемые для построения рекомендательных системах в задачах link prediction.

В практической части работы были построены и сравнены пять моделей:

* Модель GraphSAGE
* Модель GCN - 2 слоя
* Модель GCN - 3 слоя
* Модель с собственным SAGEConv
* Модель с собственным WeightedSAGECon

Путем анализа метрик эффективности таких, как accuracy, F1-score, AUC, time, а также с помощью применения кросс-валидации было выявлено, что первая модель GraphSAGE демонстрирует наилучшее качество среди рассмотренных. На основе этой модели была разработана и реализована рекомендательная система, которая может быть использована заказчиками для поиска n-количества поставщиков для выполнения государственных закупок согласно 44-ФЗ.

В целом, выполнение курсовой работы позволило углубить понимание методов анализа графовых данных, принципов построения рекомендательных систем и их интеграции с графовыми методами.

**Список использованных источников**

1. Lemei Zhang1 · Peng Liu1 · Jon Atle Gulla1 Recommending on graphs: a comprehensive review from a data perspective. Received: 29 March 2021 / Accepted in revised form: 15 January 2023. – URL: <https://arxiv.org/pdf/2212.12230>/ (дата обращения: 20.05.2024)
2. Thomas N. Kipf, Max Welling SEMI-SUPERVISED CLASSIFICATION WITH GRAPH CONVOLUTIONAL NETWORKS. Published as a conference paper at ICLR 2017 University of Amsterdam, 2017 – URL: <https://arxiv.org/pdf/1609.02907/> (дата обращения: 20.05.2024)
3. William L. Hamilton, Rex Ying, Jure Leskovec Inductive Representation Learning on Large Graphs. Department of Computer Science Stanford University Stanford, CA, 94305, 2017 — URL: <https://cs.stanford.edu/people/jure/pubs/graphsage-nips17.pdf> (дата обращения: 21.05.2024).
4. Recommendation System // NVIDIA URL: https://www.nvidia.com/en-us/glossary/recommendation-system/#:~:text=A%20recommendation%20system%20is%20an,demographic%20information%2C%20and%20other%20factors (дата обращения: 22.05.2024).
5. Link Prediction using Graph Neural Networks¶ // docs.dgl URL: <https://docs.dgl.ai/en/2.0.x/tutorials/blitz/4_link_predict.html> (дата обращения: 22.05.2024).
6. Recommender Systems with PyTorch, Part III [Code Included] // PureAI URL: <https://pureai.substack.com/p/recommender-systems-with-pytorch> (дата обращения: 23.05.2024).
7. ЕИС Закупки URL: <https://zakupki.gov.ru/epz/main/public/home.html> (дата обращения: 20.04.2024).
8. CS224W: Machine Learning with Graphs // stanford URL: <https://web.stanford.edu/class/cs224w/> (дата обращения: 20.05.2024).

**Приложение А. Программный код**

Ссылка на код: <https://github.com/nastushks/kursovaja.git>