Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное 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 «Программное обеспечение ЭВМ и информационные технологии»

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №8
по дисциплине ««Основы систем ИИ»

Тема Уменьшение размерности

Студент Батуев А.Г.

Группа ИУ7-36Б

Преподаватели Строганов Ю.В.

Москва, 2024

Содержание

ВВЕДЕНИЕ

Цель: выполнить уменьшение размерности векторов для черепашьего набора данных с использованием заданного метода и проанализировать, как это влияет на близость документов.

Задачи:

— применить метод уменьшения размерности к векторам документов.

— провести анализ изменения близости между документами после уменьшения размерности.

1 Аналитическая часть

Метод главных компонент (Principal Component Analysis, PCA) — это техника уменьшения размерности, которая используется для преобразования набора коррелированных переменных в меньший набор некоррелированных переменных, называемых главными компонентами. Метод подразумевает сохранение как можно большей полезной информации об исходных данных при уменьшении их количества.

1.1 Метод главных компонент

Основные шаги PCA:

1) центрирование данных. В каждом измерении (признаке) вычисляется среднее значение, которое затем вычитается из значений этого признака для всех наблюдений, чтобы данные были центрированы вокруг нуля.

2) вычисление ковариационной матрицы. Определяется ковариационная матрица данных, которая показывает, как различаются признаки относительно друг друга.

3) получение собственных векторов и собственных значений. Вычисляются собственные векторы и собственные значения ковариационной матрицы. Собственные векторы указывают направление главных компонент, а собственные значения показывают, сколько дисперсии данных захвачено в каждом направлении.

4) выбор главных компонент. Главные компоненты выбираются на основе величины их собственных значений. Обычно выбирают те компоненты, которые объясняют наибольшую долю дисперсии данных.

5) преобразование данных. Данные проектируются на пространство, определенное выбранными главными компонентами, что приводит к уменьшению размерности.

Перед применением PCA необходимо иметь вектора документов в формате слово : кол-во повторений в тексте. Из этого вектора мы извлекаем только кол-во повторений да так, чтобы все вектора получились упорядоченные и одной размерности. Главные компоненты — это новые переменные (основные оси), которые вычисляются так, чтобы максимально описывать разброс данных. После применения PCA часть информации может быть потеряна, поэтому результаты мер близости могут немного отличаться от тех, которые могли бы быть получены, если бы работали с исходными данными.

1.2 Способ сравнения до PCA и после

Сравнение того, как повлиял PCA на степень близости документов, производится с помощью косинусной меры близости. В качестве входных данных для неё будут подваваться матрицы близости документов (до и после PCA). Результаты будут градироваться следующим образом:

— близко к 1. Означает, что векторы полностью совпадают или находятся в одном

направлении (максимальная схожесть).

— близко к 0. Означает, что векторы ортогональны, т.е. они не имеют ничего общего (нет схожести).

— близко к -1. Означает, что векторы направлены в противоположные стороны (полная противоположность).

Косинусная мера независима от масштаба векторов, что делает её удобной для сравнения структуры матриц, поскольку она учитывает только направление векторов, игнорируя их длину.

Вывод

В разделе был рассмотрен метод главных компонент (PCA) как способ уменьшения размерности данных. Описаны основные этапы применения PCA, начиная с центрирования данных и заканчивая проекцией на новое пространство главных компонент. Также проведен анализ того, как применение PCA влияет на степень близости между документами, с использованием косинусной меры, которая была выбрана для сравнения близости документов до и после уменьшения размерности.

2 Конструкторская часть

Для анализа близости используется вектор документа, способ получения которого описывался в лабораторной работе №7.

2.1 Подготовка данных

Для метода главных компонент вектора дополняются словами из других векторов (их количество равно 0 в данном векторе). Это делается для соблюдения единой размерность векторов.

2.2 Метрики

Косинусоидальное сходство вычисляет сходство двух векторов как косинус угла между двумя векторами. Это определяет, направлены ли два вектора примерно в одном направлении. Таким образом, если угол между векторами равен 0 градусам, то косинусоидальное сходство равно 1.

Математическая формула: $\cos(A, B) = \frac{A*B}{\|A\| \times \|B\|,}$ [3]

Коэффициент корреляции Пирсона. Этот коэффициент показывает, насколько сильно и в каком направлении одна переменная (или компоненты одного вектора) зависит от другой.

Если говорить о сравнении векторов, то каждый вектор может быть интерпретирован как набор значений переменной. Корреляция Пирсона вычисляет степень линейной зависимости между соответствующими компонентами этих векторов. [4]

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{2.1}$$

Сходство Жаккара находит число общих элементов в них и делит найденное число на количество элементов обеих последовательностей.

Математическая формула: $Jac = \frac{len(A \cup B)}{len(A \cap B)}$

2.3 Получение результатов

Используя полученные вектора, применяем методы их анализа для получения матрицы схожести до и после PCA, которая, в свою очередь, используется для построения 3 графиков, отражающие результаты исследования.

Вывод

В разделе был рассмотрен процесс подготовки данных для метода главных компонент, включающий дополнение векторов словами для приведения их к единой размерности. Также были описаны три метрики: косинусоидальное сходство, коэффициент корреляции Пирсона и

сходство Жаккара, с помощью которых проводится анализ близости между векторами документов. На основе этих метрик были получены матрицы схожести до и после применения PCA, результаты которых визуализированы на трёх графиках.

3 Технологическая часть

3.1 Средства написания программы

Для реализации программного обеспечения были использованы следующие средства:

— среда разработки PyCharm 2023 community edition [16]

— язык разработки Python 3.10 [17]

Используемые библиотеки:

— os - для доступа к файлам системы и создания папок/файлов [5]

— numpy - для удобной и быстрой работы с массивами, сохранения промежуточных данных и связи их с другими библиотеками [6]

— re - для выделения слов [7]

— collections.defaultdict - для создания начальных словарей в формате (строка : число) [8]

— bs4.BeautifulSoup - для чтения html файлов [9]

— docx.Document - для чтения docx файлов [10]

— odf.opendocument.load и odf.text.P - для чтения odt файлов [11]

— pymorphy3 - для выделения начальных форм слова [12]

— matplotlib.pyplot - для отображения графиков [13]

— seaborn - для построения графиков [14]

— sklearn.metrics.pairwise.cosine similarity - косинусная мера близости [15]

— sklearn.decomposition.PCA - метод главных компонент [15]

— sklearn.metrics.jaccard_score - метод Жаккарда [15]

3.2 Подготовка к анализу

3.3 Метрики

3.3.1 Косинусная мера близости

Листинг 3.1 — Косинусная мера близости

```
    def cosine_metric(v1, v2):
        return cosine_similarity([v1], [v2])[0][0]
```

3.3.2 Метод Жаккарда

Особенностью этой функции можно назвать бинаризацию вектора, полученный после-работы метода PCA. Так как после применения PCA появляются как положительные, так и отрицательные значения.

Листинг 3.2 — Метод Жаккарда

```
def jac_metric(v1, v2, to_binary=None):
    if to_binary:
        v1_binary = [int(value >= 0) for value in v1]
        v2_binary = [int(value >= 0) for value in v2]
        return jaccard_score(v1_binary, v2_binary)
    else:
        return jaccard_score(v1, v2, average='macro')
```

3.3.3 Метод главных компонент

Листинг 3.3 — Метод главных компонент

```
def call_pca(v1s, v2s):
    sk_pca = PCA()

    all_keys_v1 = set().union(*[vec.keys() for vec in v1s])
    all_keys_v2 = set().union(*[vec.keys() for vec in v2s])

    v1s_values = [[v1.get(key, 0) for key in all_keys_v1] for v1 in
       v1s]
    v2s_values = [[v2.get(key, 0) for key in all_keys_v2] for v2 in
       v2s]

    v1_transformed = sk_pca.fit_transform(v1s_values)
    v2_transformed = sk_pca.fit_transform(v2s_values)

    return v1_transformed, v2_transformed, v1s_values, v2s_values
```

3.4 Анализ

3.4.1 Матрица близости

Матрица близости - это матрица, которая в каждой ячейке (i, j) содержит сравнительный анализ двух векторов.

Листинг 3.4 — Получение матрицы близости

```
def get_corr_matrix(vectors, trans=None):
    vecs_len = len(vectors)
    na = np.zeros((vecs_len, vecs_len))
    nb = np.zeros((vecs_len, vecs_len))
    nc = np.zeros((vecs_len, vecs_len))
    for i in range(vecs_len):
```

```
            for j in range(vecs_len):
                na[i][j] = metrics.pearson_metric(vectors[i], vectors[j])
                nb[i][j] = metrics.cosine_metric(vectors[i], vectors[j])
                nc[i][j] = metrics.jac_metric(vectors[i], vectors[j],
                    trans)
        return na, nb, nc
```

3.4.2 Постройка тепловых карт и сравнение матриц близостей

Листинг 3.5 — Функция вывода результатов

```
    def create_heatmaps(vec, n_vec, old_vec, old_n_vec, files):
        matrix_pears, matrix_cosine, matrix_jac = get_corr_matrix(vec, 1)
        matrix_pears_2, matrix_cosine_2, matrix_jac_2 = get_corr_matrix(
            old_vec)

        plt.figure(figsize=(19.2, 10.8))
        sea.heatmap(matrix_cosine, xticklabels=files, yticklabels=files,
            square=True, linewidths=.3, cmap="YlGnBu",
                    annot=True, annot_kws={"size": 4}, fmt=".2f")
        plt.savefig(os.path.join(RES_DIR, "HeatmapCos.png"), dpi=100)
        plt.clf()

        plt.figure(figsize=(19.2, 10.8))
        sea.heatmap(matrix_pears, xticklabels=files, yticklabels=files,
            square=True, linewidths=.3, cmap="YlGnBu",
                    annot=True, annot_kws={"size": 4}, fmt=".2f")
        plt.savefig(os.path.join(RES_DIR, "HeatmapPearson.png"), dpi=100)
        plt.clf()

        plt.figure(figsize=(19.2, 10.8))
        sea.heatmap(matrix_jac, xticklabels=files, yticklabels=files,
            square=True, linewidths=.3, cmap="YlGnBu",
                    annot=True, annot_kws={"size": 4}, fmt=".2f")
        plt.savefig(os.path.join(RES_DIR, "HeatmapJac.png"), dpi=100)
        plt.clf()

        matrix_pears, matrix_cosine, matrix_jac = get_corr_matrix(n_vec,
            1)
```

```
    matrix_pears_2, matrix_cosine_2, matrix_jac_2 = get_corr_matrix(
        old_n_vec)

    plt.figure(figsize=(19.2, 10.8))
    sea.heatmap(matrix_cosine, xticklabels=files, yticklabels=files,
        square=True, linewidths=.3, cmap="YlGnBu",
                annot=True, annot_kws={"size": 4}, fmt=".2f")
    plt.savefig(os.path.join(RES_DIR, "HeatmapCosNorm.png"), dpi=100)
    plt.clf()

    plt.figure(figsize=(19.2, 10.8))
    sea.heatmap(matrix_pears, xticklabels=files, yticklabels=files,
        square=True, linewidths=.3, cmap="YlGnBu",
                annot=True, annot_kws={"size": 4}, fmt=".2f")
    plt.savefig(os.path.join(RES_DIR, "HeatmapPearsonNorm.png"), dpi
        =100)
    plt.clf()

    plt.figure(figsize=(19.2, 10.8))
    sea.heatmap(matrix_jac, xticklabels=files, yticklabels=files,
        square=True, linewidths=.3, cmap="YlGnBu",
                annot=True, annot_kws={"size": 4}, fmt=".2f")
    plt.savefig(os.path.join(RES_DIR, "HeatmapJacNorm.png"), dpi=100)
    plt.clf()
```

3.4.3 Функция постройки всех тепловых карт

На основе матрицы, полученной из предыдущей функции, строится график в seaborn. Данный график является результатом всей работы.

Листинг 3.6 — Функция main

```
def build_heatmaps_recursive(directory):
vectors = list()
norm_vectors = list()
files = list()
for folder in os.listdir(directory):
    folder_path = os.path.join(directory, folder)
    vectors_dir_path = os.path.join(folder_path, "vectors")
    norm_vectors_dir_path = os.path.join(folder_path, "norm_vectors")
    for filename in os.listdir(vectors_dir_path):
        files.append(filename)
        file_path = os.path.join(vectors_dir_path, filename)
```

```
            file_path_norm = os.path.join(norm_vectors_dir_path, filename)
            vectors.append(get_vectors(file_path))
            norm_vectors.append(get_vectors(file_path_norm))

    v1_transformed, v2_transformed, v1s_values, v2s_values = metrics.
        call_pca(vectors, norm_vectors)
    create_heatmaps(v1_transformed, v2_transformed, v1s_values, v2s_values
        , files)
```

Вывод

В разделе были рассмотрены средства разработки программы, а также библиотеки, использованные для обработки данных, работы с файлами и построения графиков. Описаны функции для расчета различных метрик (косинусная мера близости, метод Жаккарда) и метода главных компонент (PCA). На основе этих методов были построены матрицы близости до и после применения PCA, визуализированные в виде тепловых карт. В результате работы получены три графика, отражающие степень близости между файлами, как с обычными, так и с нормализованными векторами.

4 Исследовательская часть

Цель исследования: провести уменьшение вектора файла методом главных компонент.

4.1 Оборудование

Характеритстики ноутбука:

— процессор intel-core i5-12500H

— озу 16 Гб DDR4

— ос Windows 11 [18]

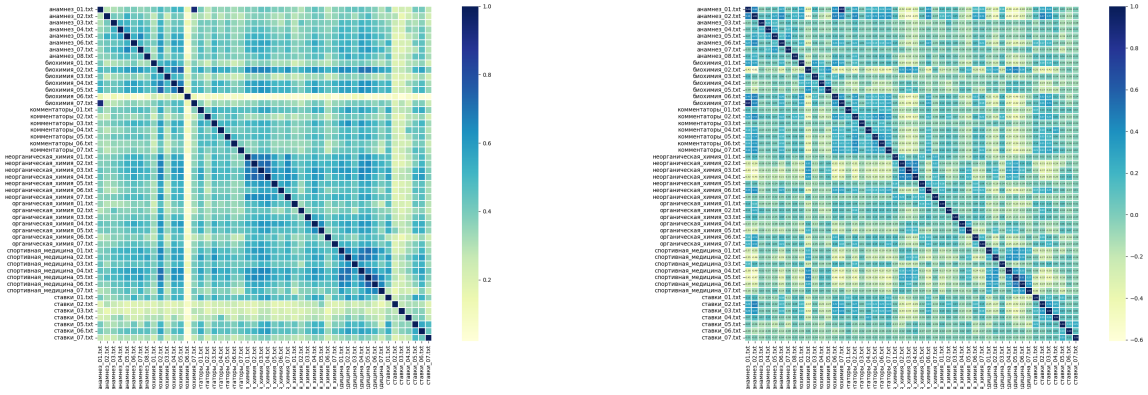
4.2 Степень различия до и после PCA

Для выяснения степени различия между двумя матрицами используется косинусная мера близости, в которую передается предварительно преобразованная в одномерный вектор матрица близости. Числа в таблице получаются при сравнении матриц до и после PCA.

Метрика	**Обычные вектора**	**Нормализированные вектора**
Косинусная мера близости	0.1746	0.2269
Корреляция Пирсона	0.1780	0.2363
Жаккард	0.8032	0.7783

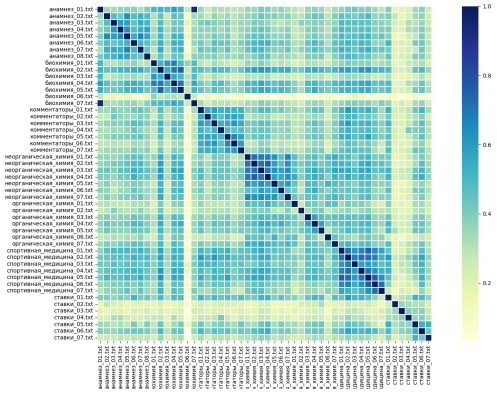
Таблица 4.1 — Сравнение степеней различия для обычных и нормализированных векторов

4.3 Графики

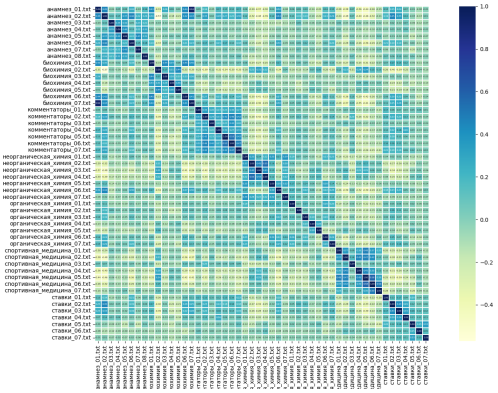


а) До применения PCA б) После применения PCA

Рисунок 4.1 — Сравнение косинусной меры близости на обычных векторах до применения PCA и после применения PCA

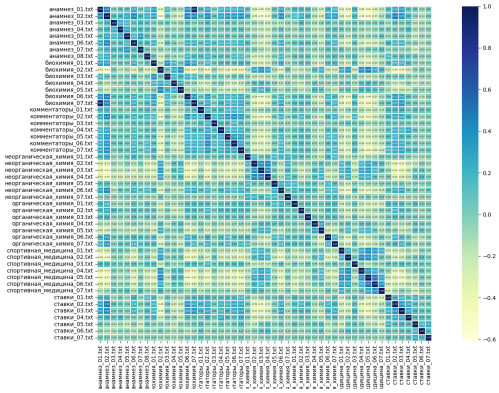


а) До применения PCA

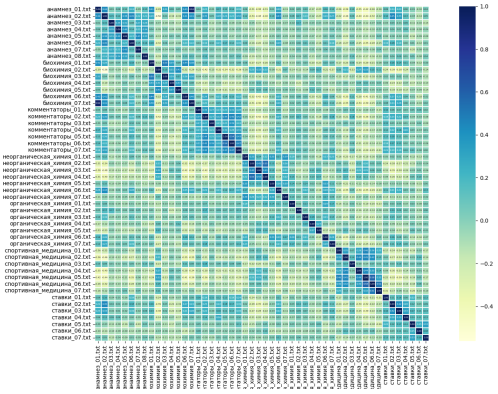


б) После применения PCA

Рисунок 4.2 — Сравнение косинусной меры близости на нормализированных векторах до применения PCA и после применения PCA

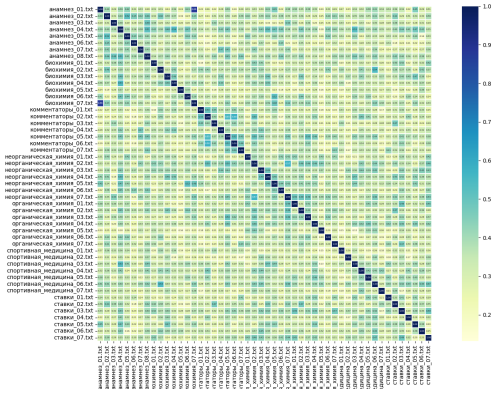


а) Тепловая карта, построеная с использование корреляции Пирсона на обычных векторах

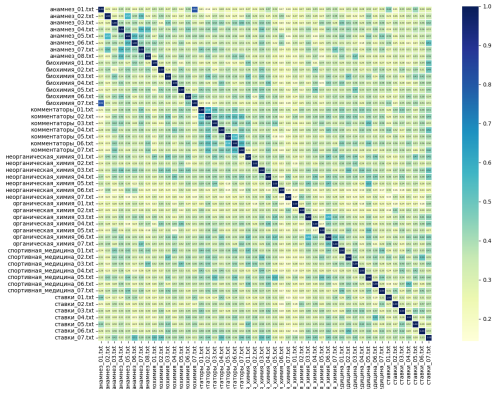


б) Тепловая карта, построеная с использование корреляции Пирсона на нормализированных векторах

Рисунок 4.3 — Сравнение корреляцией Пирсона после применения PCA на обычных векторах и нормализированных векторах



а) Тепловая карта, построеная с использование корреляции Жаккарда на обычных векторах



б) Тепловая карта, построеная с использование корреляции Жаккарда на нормализированных векторах

Рисунок 4.4 — Сравнение метода Жаккарда после применения PCA на обычных векторах и нормализированных векторах

4.4 Вывод

Проведённое уменьшение векторов файлов методом главных компонент изменяет их взаимное отношение, что подтверждается изменениями в тепловых картах и числовыми показателями таблицы (см. 4.2). Несмотря на уменьшение размерности, исходные структуры векторов сохраняются, однако степень схожести после применения метода PCA варьируется в зависимости от типа метрики и нормализации векторов.

ЗАКЛЮЧЕНИЕ

Цель исследования достигнута: выполнено уменьшение вектора файла методом главных компонент, что позволило сохранить ключевую информативность данных и провести сравнительный анализ векторов до и после преобразования.

Построенные графики показывают степень изменений в векторах. Несмотря на различия в структуре нормализированных и обычных векторов, значительная часть исходных взаимосвязей сохраняется. Применение метода PCA выявило более выраженные связи между документами, что наглядно отражено на тепловых картах.

Самыми информативными метриками для анализа оказались косинусная мера близости и корреляция Пирсона. Эти метрики показывают практически идентичные результаты, как на тепловых картах, так и при сравнении с матрицами до уменьшения размерности. Метод Жаккарда показал менее согласованные результаты, что затрудняет их интерпретацию по сравнению с предыдущими метриками.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Seaborn: statistical data visualization URL: https://seaborn.pydata.org/ (Дата обращения 22.09.2024)
2. Principal component analysis (PCA). URL: https://scikit-learn.org/dev/modules/generated/sklearn.decomposition.PCA.html (Дата обращения 01.10.2024)
3. Метод эффективного расчета матрицы ближайших соседей для полнотекстовых документов. URL: https://cyberleninka.ru/article/n/metod-effektivnogo-rascheta-matritsy-blizhayshih-sosedey-dlya-polnotekstovyh-dokumentov (Дата обращения 01.10.2024)
4. Note on Regression and Inheritance in the Case of Two Parents (Карл Пиросон) - Лондон: Royal Society, 1904 - №75. - стр. 347–352
5. Модуль os в Python, доступ к функциям ОС URL: https://docs-python.ru/standart-library/modul-os-python/
6. NumPy Documentation URL: https://numpy.org/doc/
7. Использование регулярных выражений в Python URL: https://docs-python.ru/standart-library/modul-re-python/
8. Класс defaultdict() модуля collections в Python URL: https://docs-python.ru/standart-library/modul-collections-python/klass-defaultdict-modulja-collections/
9. Документация Beautiful Soup URL: https://www.crummy.com/software/BeautifulSoup/bs4/doc.ru/bs4r
10. Объект Document модуля python-docx в Python URL: https://docs-python.ru/packages/modul-python-docx-python/klass-document/
11. Документация odf URL: https://pypi.org/project/odfpy/
12. Документация pymorphy3 URL: https://pypi.org/project/pymorphy3/
13. Matplotlib 3.9.2 documentation URL: https://matplotlib.org/stable/index.html
14. seaborn: statistical data visualization URL: https://seaborn.pydata.org/
15. Документация sklearn URL: https://scikit-learn.org/stable/
16. Среда разработки Pycharm 2023 Community edition URL: https://www.jetbrains.com/ru-ru/pycharm/download/other.html

17. Язык разработки Python 3.10 URL: https://www.python.org/downloads/release/python-3100/

18. ОС Windows 11 URL: https://www.microsoft.com/ru-ru/software-download/windows11