ВИЗУАЛИЗАЦИЯ МНОГОМЕРНЫХ НАБОРОВ ДАННЫХ ПРИ ПОМОЩИ АЛГОРИТМОВ СНИЖЕНИЯ ПРОСТРАНСТВА ПРИЗНАКОВ PCA И t-SNE

VISUALIZATION OF MULTI-DIMENSIONAL DATASETS

USING PCA AND t-SNE FEATURE SPACE REDUCTION ALGORITHMS

УДК 004.5

Попова И.А., студент, popovai1@student.bmstu.ru
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана, кафедра «Системы обработки информации и управления»
Попова А.А., студент, annandreeva214@gmail.com
Россия, 125047, г. Москва, РХТУ им. Д.И. Менделеева, кафедра биотехнологии
Соболева Е.Д., студент, ls1997@yandex.ru
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана, кафедра «Системы обработки информации и управления»

Popova I.A., student, popovai1@student.bmstu.ru
Russia, 105005, Moscow, MSTU N.E. Bauman, Department of Information Processing and Management Systems
Popova A.A., student, annandreeva214@gmail.com
Russia, 125047, Moscow, MUCTR, Department of Biotechnology
Soboleva E.D., student, ls1997@yandex.ru
Russia, 105005, Moscow, MSTU N.E. Bauman, Department of Information Processing and Management Systems

Аннотация

Снижение размерности данных помогает облегчить анализ собранного набора данных, а именно сделать возможной его визуализацию. Именно визуализация является центральным компонентом исследовательского анализа данных - первого этапа анализа данных, цель которого состоит в том, чтобы понять смысл данных, прежде чем приступить к более целенаправленному моделированию и анализу. В данной статье сравниваются и исследуются несколько методов для снижения размерности и визуализации данных: анализ главных компонент (PCA) и t-стохастическое встраивание соседей (t-SNE).

Annotation

Reducing the dimensionality of the data helps to facilitate the analysis of the collected dataset, namely to make it possible to visualize it. Visualization is a central component of exploratory data analysis, the first phase of data analysis, the goal of which is to understand the meaning of the data before embarking on more targeted modeling and analysis. This article compares and explores several methods for dimensionality reduction and data visualization: principal component analysis (PCA) and t-stochastic neighbor embedding (t-SNE).

Ключевые слова: снижение размерности, алгоритм, визуализация, набор данных, модель.

Keywords: *dimensionality reduction, algorithm, visualization, data set, model.*

В современном обществе нас окружает большое количество данных из разных областей: цифровых коммуникаций, образования, экономики, медицины и так далее. Данные могут иметь сложную структуру: содержать множество признаков, которые могут быть нужными или не нести полезной информации для

исследователя. Данные высокой размерности могут затруднять анализ, делать невозможной визуализацию, а также быть дорогостоящими для обработки и хранения. Чтобы лучше визуализировать наборы данных, содержащие множество признаков, необходимо использовать методы снижения размерности данных.

Снижение размерности (Dimensionality Reduction) - это этап предварительной обработки для удаления избыточных признаков, зашумленных и нерелевантных данных с целью повышения точности обучения и сокращения времени обучения.

Чтобы извлечь важные признаки из набора данных, а значит оставить ценную информацию применяют два основных способа: извлечение признаков и уменьшение размерности данных.

Первый способ сохраняет наиболее важную информацию из исходного набора данных, а второй находит меньшее число новых признаков, каждый из которых представляет собой комбинацию исходных, которые содержат в основном ту же информацию, что и исходные признаки.

В данной статье будут рассмотрены методы снижения размерности данных PCA и t-SNE, а также данные алгоритмы будут применены к многомерному набору данных для сокращения числа признаков с целью последующей визуализации датасета.

Снижение размерности данных используется с целью:

- сокращения времени обработки данных и места для их хранения;
- увеличения точности модели машинного обучения и избежание переобучения;
- устранения избыточности данных и зашумленных данных;
- упрощения визуализации данных при сокращении многомерных наборов данных до двух либо трех основных признаков.

Анализ главных компонент (Principal Component Analysis, PCA)

Анализ главных компонент (PCA) - это метод машинного обучения без учителя для линейного снижения размерности, целью которого является сокращение размерности набора данных с потерей наименьшего количества информации. Данный метод был предложен Карлом Пирсоном в 1901 году. Это математическая процедура, которая преобразует ряд коррелированных переменных в меньшее число некоррелированных переменных, называемых главными компонентами. Первая и вторая главные компоненты учитывают, как можно большую дисперсию данных, и каждая последующая компонента учитывает, как можно больше оставшейся дисперсии. Алгоритм находит направления с максимальной дисперсией в многомерных данных и проецирует их на новое подпространство с такими же или меньшими размерами, чем исходное.

Алгоритм позволяет аппроксимировать n-размерное облако точек до эллипсоида (также будет n-размерным), при этом оси эллипсоида будут соответствовать будущим главным компонентам. Проекция на оси (снижение размерности) должна сохранить наибольшее число информации.

Ортогональные оси (главные компоненты) нового подпространства могут быть интерпретированы как направления с максимальной дисперсией с учетом условия, что новые оси-признаки являются ортогональными друг к другу. Как показано на Рисунке 1 X1 и X2 являются осями исходных признаков объектов, а PC1 и PC2 являются найденными главными компонентами.

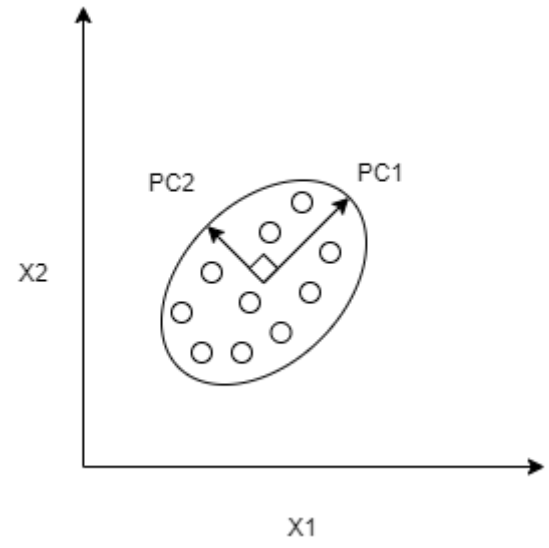

Рисунок 1. Найденные новые оси – главные компоненты PC1 и PC2.

PCA широко используется в ряде областей: при обработке изображений, для сжатия данных, в биоинформатике, эконометрике, а также в общественных науках.

Рассмотрим более подробно основные шаги алгоритма.

1. Стандартизировать набор n-мерных данных.

   Стандартизация - это масштабирование набора данных таким образом, чтобы все признаки и их значения находились в одинаковом диапазоне. Стандартизация осуществляется путем вычитания из каждого значения признака среднего значения для данного признака и деления результата на стандартное отклонение этого же признака.

$$x_{new} = \frac{x - \mu}{\sigma} \qquad (1)$$

   Так как набор данных был нормализован, в таком случае среднее значение для каждого признака равно 0, а стандартное отклонение равно 1.

2. Рассчитать ковариационную матрицу для объектов в наборе данных.

PCA помогает определить корреляцию и зависимости между признаками в наборе данных. Ковариационная матрица выражает корреляцию между различными переменными в наборе данных. Важно идентифицировать сильно зависимые переменные, потому что они содержат избыточную информацию, которая снижает общую производительность модели.

Математически ковариационная матрица представляет собой матрицу p × p, где p представляет размеры набора данных. Каждая запись в матрице представляет ковариацию соответствующих переменных. То есть (i,j)-элемент - корреляция признаков (Xi, Xj).

Формула для расчета ковариации следующая:

$$Cov(X_i, X_j) = M\left[(X_i - M(X_i)) * \left(X_j - M(X_j)\right)\right]$$
$$= M(X_i X_j) - M(X_i) * M(X_j) \qquad (2)$$

В данном случае $M(X_i) = M(X_j) = 0$, тогда:

$$Cov(X_i, X_j) = M(X_i\, X_j) \qquad (3)$$

Когда $X_i = X_j$:

$$Cov(X_i, X_j) = Var(X_i) \qquad (4)$$

Рассмотрим случай, когда у нас есть трехмерный набор данных с переменными f1, f2 и f3, ковариационная матрица представляет собой матрицу 3 × 3, как показано в Таблице 1:

Таблица 1. Ковариационная матрица для трехмерного набора данных.

	F1	F2	F3
F1	Var(f1)	Cov(f1, f2)	Cov(f1, f3)
F2	Cov(f2, f1)	Var(f2)	Cov(f2, f3)
F3	Cov(f3, f1)	Cov(f3, f2)	Var(f3)

В данном случае:

$$Var(X) = M((X - \mu)^2) = \frac{1}{n-1} * \sum_{i=1}^{n} (x_i - \bar{x})^2, \quad (5)$$

где $\bar{x}$ – среднее значение вектора x.

$$Cov(X, Y) = \frac{1}{n-1} * \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}), \quad (6)$$

где $\bar{x}$ – среднее значение вектора x и $\bar{y}$ - среднее значение вектора y.

В матрице по главной диагонали расположены дисперсии признаков, а в оставшихся ячейках находятся ковариации пар признаков. Ковариационная матрица будет симметрична. Причем ковариационная матрица будет описывать дисперсию случайной величины, то есть ее форму.

На основании ковариационной матрицы можно сделать следующие заключения:

- Значение ковариации указывает, насколько взаимозависимы две переменные относительно друг друга;
- Если значение ковариации отрицательно, это означает, что соответствующие переменные обратно пропорциональны друг другу;
- Положительная ковариация означает, что соответствующие переменные прямо пропорциональны друг другу.

3. Вычислить собственные значения и собственные вектора для ковариационной матрицы.

Собственный вектор – это ненулевой вектор $\vec{v}$, который при умножении на квадратную матрицу A равен такому же вектору, умноженному на скалярный коэффициент λ. Число λ – собственное число(значение)

матрицы A. Каждому собственному вектору соответствует свое собственное число λ.

Идея собственных векторов состоит в том, чтобы использовать ковариационную матрицу, чтобы понять, где в данных больше всего дисперсии. Требуется найти такие вектора, чтобы размер(дисперсия) проекции выборки была максимальная.

Пусть A - квадратная матрица (в нашем случае ковариационная матрица, вычисленная ранее), v - вектор, а λ - скаляр, удовлетворяющий равенству $Av = \lambda v$, тогда λ называется собственным значением, связанным с собственным вектором v линейного оператора A.

Запишем в виде уравнения:

$$Av - \lambda v = 0 \qquad (7)$$

$$(A - \lambda I) = 0 \qquad (8)$$

Где A – квадратная ковариационная матрица, v – собственный вектор, λ – соответствующее собственное число, I – единичная матрица.

Так как v - ненулевой вектор, тогда в единственном случае уравнение может быть равно нулю, если:

$$\det(A - \lambda I) = 0 \qquad (9)$$

Собственные числа найдем из выражения:

$$A - \lambda I = 0 \qquad (10)$$

Собственный вектор для каждого собственного значения можно найти, используя выражение:

$$(A - \lambda I)v = 0 \qquad (11)$$

Следовательно, собственный вектор задает направление максимальной дисперсии у проекции, а собственное значение, соответствующее этому вектору равно величине этой дисперсии. Число собственных векторов будет равно размеру матрицы ковариации.

4. Отсортировать собственные значения и соответствующие им собственные векторы.

После того, как были вычислены собственные векторы и собственные значения, все, что нужно сделать, это упорядочить их в порядке убывания, где собственный вектор v с наибольшим собственным значением λ является наиболее значимым и, таким образом, образует первый главный компонент.

Максимальная вариация исходного набора данных будет достигаться вдоль собственного вектора матрицы, которому соответствует максимальное собственное значение. Следовательно, главные компоненты, на которые требуется спроецировать данные – это собственные вектора, которым соответствуют топ-k штук собственных значений соответствующей матрицы.

5. Выбрать k собственных значений и сформируйте матрицу собственных векторов.

Выберете k первых собственных значений и соответствующих им собственных векторов.

Таким образом, главные компоненты меньшего значения могут быть удалены, чтобы уменьшить размеры данных.

6. Преобразовать исходную матрицу.

Последний шаг в вычислении главных компонент состоит в том, чтобы сформировать матрицу, известную как матрица признаков, которая содержит все значимые признаки, содержащие максимальное количество ценной информации.

Получим матрицу новых признаков следующим образом:

Исходная матрица признаков * выбранные k собственных векторов = матрица новых признаков.

Таким образом была получена проекция исходного набора данных в ортогональном базисе найденных главных компонент. Полученные переменные являются независимыми друг от друга. Они содержат большую часть полезной информации, которая была разбросана по исходным признакам.

Рассмотрим использование PCA, реализованного в sklearn. В конструктор класса PCA передается параметр - n_components, который задает новое число измерений датасета, то есть n собственных векторов с наибольшими собственными числами.

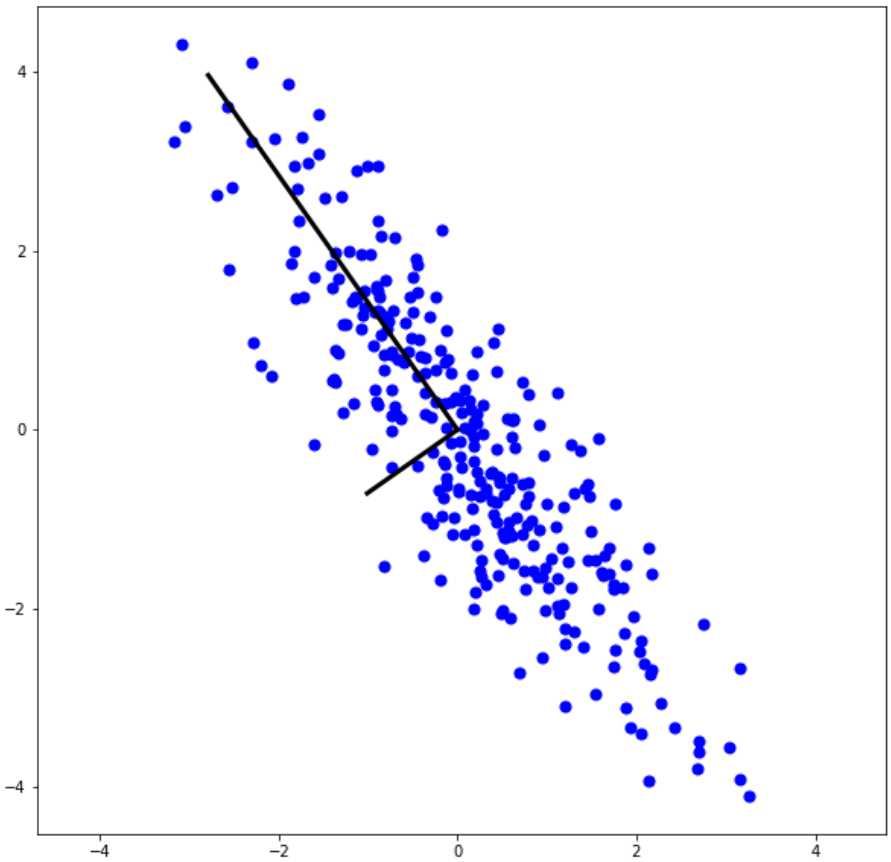

Рисунок 2. Статистическая выборка в двумерном пространстве и ее главные компоненты.

На рисунке 2 видно, что первая главная компонента, которой соответствует более длинный вектор черного цвета, объясняет большую часть дисперсии данных – около 95%, а значит она содержит почти всю информацию о расположении данной выборки в пространстве.

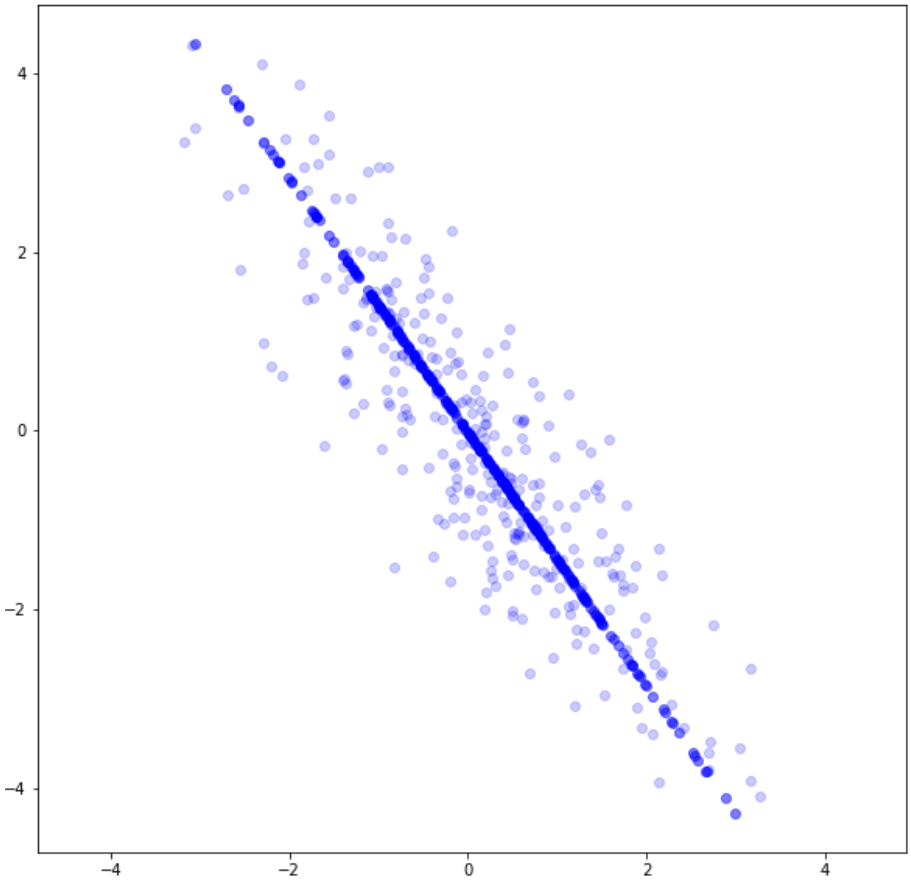

Рисунок 3. Проекция статистической выборки на первую главную компоненту.

Таким образом при помощи алгоритма PCA была снижена размерность исходного набора данных в 2 раза, полученные признаки были визуализированы на двумерном графике. При снижении размерности удалось сохранить наиболее значимую информацию, содержащуюся в исходном наборе данных. Таким образом был приближен многомерный набор данных с помощью данных меньшей размерности.

Стохастическое вложение соседей с t-распределением (t-distributed stochastic neighbor embedding, t-SNE)

t-SNE – алгоритм машинного обучения без учителя, используемый преимущественно для визуализации многомерных наборов данных. Алгоритм разработан в 2008 году исследователям в области AI Лоуренсом ван дер Маатеном и Джеффри Хинтоном. Алгоритм базируется на методе SNE, который был опубликован Хинтоном и Ровейсом в 2002 году.

Техника SNE имеет некоторые недостатки в поиске оптимальной функции стоимости и сталкивается с проблемой скученности (crowding problem) смысл которой состоит в том, что расстояние между двумя точками в пространстве отображения, которые соответствуют двум точкам в исходном многомерном пространстве, должно быть больше, чем расстояние, моделируемое при помощи распределения Гаусса (нормального распределения). Чтобы избежать этих проблем, Хинтон и Роуис немного изменили функцию стоимости t-SNE. Два основных изменения:

- применение симметричной версии SNE для упрощения градиентных спусков;
- применение t-распределения Стьюдента в низкоразмерном пространстве вместо нормального распределения (распределения Гаусса) для устранения проблемы скученности (crowding problem).

t-SNE сокращает сотни измерений к меньшему числу, при этом сохраняются важные закономерности в данных, а именно: чем ближе объекты находятся друг к другу в исходном многомерном пространстве, тем меньше расстояние между ними в пространстве более низкой размерности. В отличие от PCA, который используется как для визуализации, так и для снижения размерности данных, t-SNE применяется только для визуализации многомерных наборов данных и по своей природе является нелинейной техникой. Это означает, что алгоритм может фиксировать нелинейные закономерности в данных. Поскольку это

вероятностный метод, следовательно, при разных запусках алгоритма на одном и том же наборе данных можно получить разные результаты.

Алгоритм t-SNE используется для визуализации результатов исследования в области компьютерной безопасности, биоинформатике, медицине.

Рассмотрим более подробно основные шаги алгоритма.

На первых шагах алгоритма вычисляется вероятность подобия точек в многомерном пространстве и вычисляется вероятность подобия точек в соответствующем низкоразмерном пространстве. Сходство точек рассчитывается как условная вероятность того, что точка A выбрала бы точку B в качестве своего соседа, если бы соседи были выбраны пропорционально их плотности вероятности согласно нормальному распределению (распределению Гаусса) с центром в точке A. Используется T-тест из t-распределения.

Затем алгоритм пытается минимизировать разницу между этими условными вероятностями (или сходствами) в многомерном и низкоразмерном пространствах для идеального представления точек данных в низкоразмерном пространстве.

Для измерения минимизации суммы разности условной вероятности t-SNE минимизирует сумму расхождения Кульбака-Лейблера общих точек данных с использованием метода градиентного спуска.

Дивергенция Кульбака-Лейблера или расхождение KL - это мера того, как одно распределение вероятностей отклоняется от второго ожидаемого распределения вероятностей.

Дивергенция Кульбака-Лейблера определяется выражением:

$$D_{KL}(P||Q) = -\sum_{x \in X} P(x) \log\left(\frac{Q(x)}{P(x)}\right) \qquad (12)$$

Цель состоит в том, чтобы распределения $Q(x)$ и $P(x)$ совпадали как можно больше. T-SNE минимизирует сумму расхождений KL по всем точкам данных, используя метод градиентного спуска.

Под точкой данных будем понимать точку x_i в исходном многомерном пространстве признаков R^D. Где D – размерность исходного пространства. Тогда точка отображения - точка y_i в низкоразмерном пространстве отображения R^2. Предполагается, что каждой точке отображения соответствует одна исходная точка, то есть имеет место биекция.

Главная цель такого преобразования – сохранить изначальную структуру данных. Таким образом, если две точки исходного набора данных расположены близко, тогда эти же точки должны располагаться рядом в пространстве отображения.

Положим, что $|x_i - x_j|$ евклидово расстояние между двумя исходными точками данных, тогда $|y_i - y_j|$ – расстояние между соответствующими точками в пространстве отображения.

Необходимо рассчитать, насколько точка x_j находится близко к точке x_i, при этом точка x_i берется из нормального распределения с заданным стандартным отклонением σ. Причем дисперсия выбирается следующим образом: точки, расположенные в областях с большей плотностью должны иметь меньшую дисперсию в отличие от точек, расположенных в областях с малой плотностью.

Условное сходство для двух точек данных определяется следующим образом:

$$p_{j|i} = \frac{\exp\left(\frac{-\left|x_i - x_j\right|^2}{2 * \sigma_i^2}\right)}{\sum_{k \neq i} \exp\left(\frac{-|x_i - x_k|^2}{2 * \sigma_i^2}\right)} \quad (13)$$

При этом сходство p_{ij} вычисляется по формуле:

$$p_{ij} = \frac{p_{i|j} + p_{j|i}}{2 * n} \quad (14)$$

Где n – общее число точек в исходном наборе данных. В результате получим матрицу сходства для исходных данных, которая будет являться постоянной.

Определим матрицу сходства для точек отображения:

$$q_{ij} = \frac{(1 + \left\|y_i - y_j\right\|^2)^{-1}}{\sum_{k \neq l}(1 + \|y_k - y_l\|^2)^{-1}} \quad (15)$$

Для решения проблемы скученности (crowding problem) используется t-распределения Стьюдента с одной степенью свободы или распределение Коши для точек из пространства отображения. При использовании нормального распределения для точек отображения, как и для точек данных может возникнуть дисбаланс в распределении расстояний для соседних точек. Используемое распределение Стьюдента имеет более «тяжелый» хвост, таким образом позволяет компенсировать этот дисбаланс.

Будем стремиться к тому, чтобы обе матрицы были наиболее близки, так как это будет означать, что похожие точки данных будут иметь похожие отображения.

Целевая функция, используемая в алгоритме, имеет следующий вид:

$$C = \sum_i KL(P||P) = \sum_i \sum_j p_{ij} log \frac{p_{ij}}{q_{ij}} \quad (16)$$

Это выражение определяет, насколько близки две матрицы сходства. Чтобы минимизировать целевую функцию используется метод градиентного спуска (gradient descent). Формула для вычисления градиента:

$$\frac{\partial C}{\partial y_i} = 4 * \sum_j (p_{ij} - q_{ij}) * (y_i - y_j) * (1 + \|y_i - y_j\|^2)^{-1} \qquad (17)$$

Другими словами, точки пространства отображения будут стягиваться для близлежащих точек в многомерном пространстве признаков и отталкиваться для удаленных точек. t-SNE отличается от PCA тем, что сохраняет только небольшие попарные расстояния или локальные сходства, тогда как PCA сохраняет большие попарные расстояния для максимизации дисперсии.

Рассмотрим использование t-SNE, реализованного в sklearn. В конструктор класса TSNE передается параметр - n_components, который задает новое число измерений датасета, а также параметр perplexity (перплексия), который задает чувствительность модели к локальным или глобальным паттернам данных. Параметр в некотором смысле является предположением о количестве ближайших соседей каждой точки. Рекомендуется задавать значения перплексии в интервале от 5 до 50.

Рассмотрим работу встроенного алгоритма на наборе данных Iris Dataset, установим значение перплексии (параметр perplexity) равным 20.

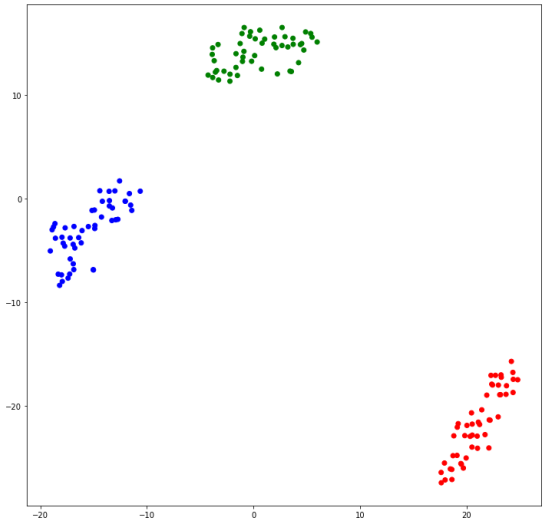

Рисунок 4. Результат работы алгоритма t-SNE на наборе данных Iris Dataset с параметром равной perplexity 20.

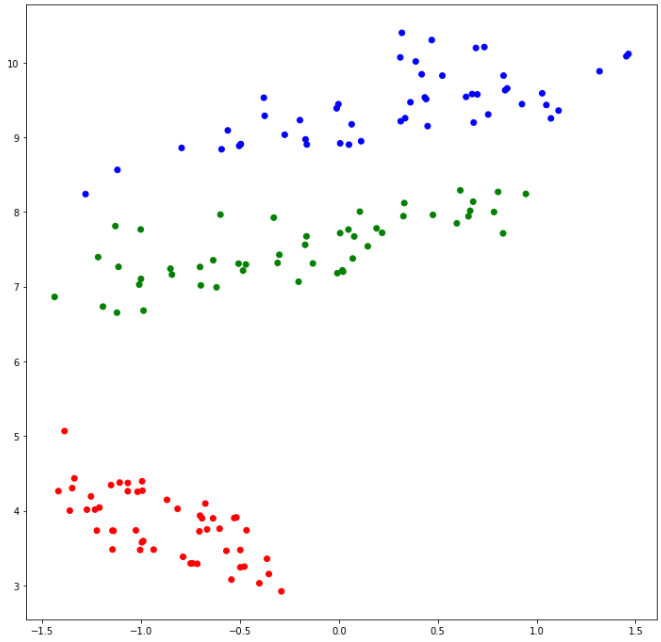

Рисунок 5. Результат работы алгоритма t-SNE на наборе данных Iris Dataset с параметром равной perplexity 80.

Можно сделать вывод о том, что, если перплексия принимает не большое значение, модель будет достаточно хорошо вычленять небольшие группы похожих объектов, если большое — большие группы. Таким образом можно

сохранить большую часть общей структуры данных при большом значении перплексии.

Чтобы сравнить работу алгоритмов PCA и t-SNE применим их для визуализации многомерного набора данных MNIST. Это набор данных для компьютерного зрения, состоящий из изображений рукописных цифр от 0 до 9. Всего база данных MNIST содержит 60000 изображений для обучения модели и 10000 изображений для тестирования работы обученной модели. Каждый образец представляет собой черно-белое изображение рукописной цифры размером 28 × 28 пикселей. Таким образом будем работать с набором данных, где каждый образец имеет 784 признака, а, следовательно, 784 измерения. Затем сравним, чем отличаются алгоритмы и выберем один из них, который лучше справился с поставленной задачей. Применяя данные методы к одному и тому же набору данных можно заметить, чем они отличаются друг от друга.

MNIST с PCA

Будем отображать точки данных в многомерном пространстве в двумерное пространство с двумя главными компонентами.

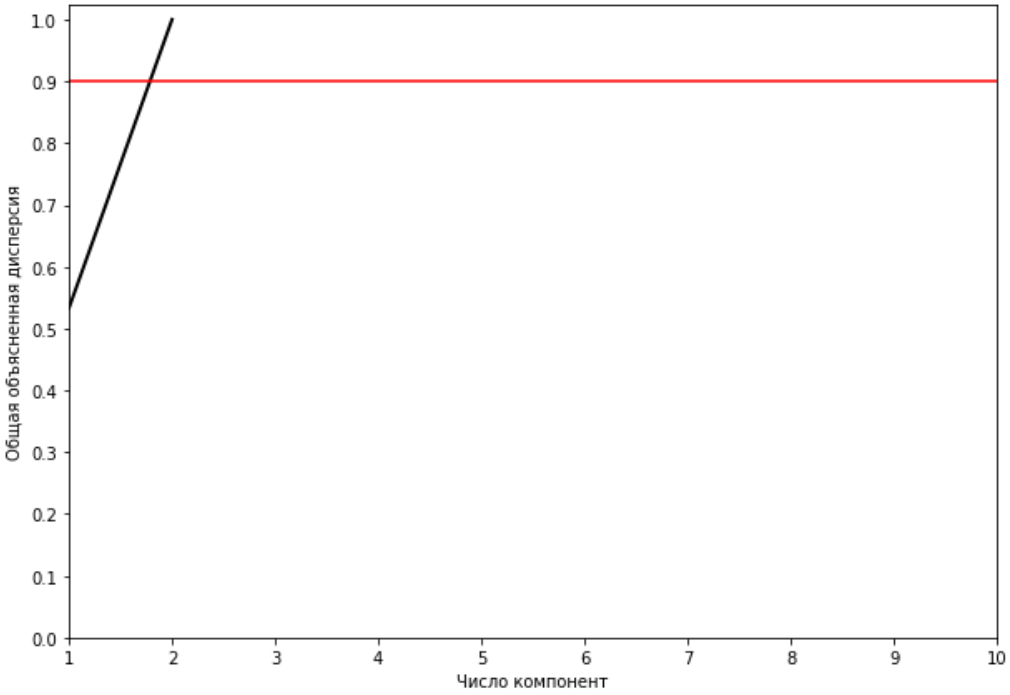

Рисунок 6. Количество объясненной дисперсии, содержащейся в первых двух компонентах.



Рисунок 7. Визуализация MNIST при помощи PCA.

По графику на Рисунке 7 видно, что нет возможности четко разделить цифры или сгруппировать их. Данные MNIST не упорядочены для приятной визуализации. На графике можно заметить зависимость между различными цифрами. Например, кластер с цифрой 8 наиболее распределен по другим кластерам. Это значит, что в наборе данных присутствует много разных вариаций ее написания, которые делают эту цифру схожей с другими цифрами. Однако алгоритм справился с задачей достаточно быстро. Причина неудачной визуализации заключается в том, что в многомерном пространстве MNIST слишком сложен для

PCA, инструмента линейной проекции, чтобы отображать его в двухмерном пространстве.

MNIST с t-SNE

Предыдущий метод работают не очень хорошо, потому что он способен выявлять простые (линейные) структуры данных; с другой стороны, t-SNE - это метод, который может работать со сложными структурами данных. t-SNE пытается сохранить расстояния между соседними точками. Для каждой точки данных t-SNE оценивает и выбирает соседей, а затем пытается сделать так, чтобы у всех точек было одинаковое количество соседей. Рассмотрим двумерный график, который был построен с помощью t-SNE.



Рисунок 8. Визуализация MNIST при помощи t-SNE.

На рисунке видно, что t-SNE очень успешно выявляет кластерные структуры, объединяя точки данных в одну категорию и разделяя каждый кластер. Однако t-SNE потребовалось значительно больше времени для решения задачи нежели PCA. Это хорошая техника для визуализации данных. Однако у алгоритма есть один недочет: застревание в локальном минимуме, как показано на рисунке 8.

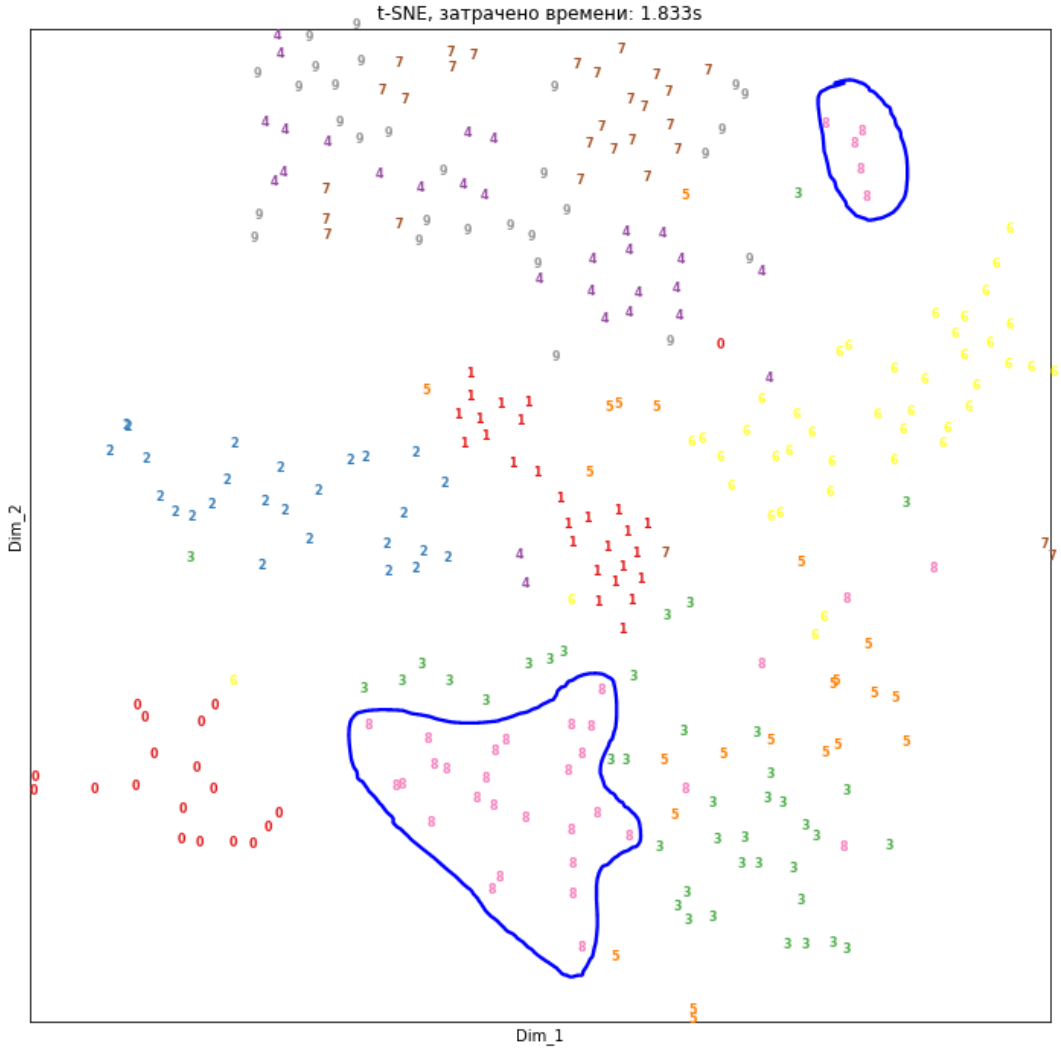

Рисунок 9. T-SNE застревает в локальном минимуме.

На Рисунке 9 есть два выделенных синим контуром кластера, которые находятся далеко друг от друга. Это означает, что точки данных в этом классе

застревают в локальном минимуме, когда t-SNE начинает выполняться. Есть несколько способов, которые могут помочь преодолеть локальный минимум:

1. добавить больше образцов в набор данных;
2. использовать имитацию отжига и тщательно подобрать гиперпараметры.

Вывод

Таким образом были рассмотрены два наиболее известных и используемых при решении практических задач машинного обучения метода снижения размерности данных: PCA и t-SNE. Достаточно сложно сказать, действительно ли один метод лучше другого, однако следует выбрать тот, который соответствует конкретному набору данных и требованиям, поставленных к решению. Анализ главных компонент (PCA) имеет тенденцию группировать точки данных одного класса вместе в длинную, но узкую полосу. PCA является основным алгоритмом, который применяется для уменьшения размерности данных. Он базируется на таких простых понятиях, как дисперсия, матрица ковариации, собственные векторы и собственные числа. Однако PCA может удалить много информации, которая действительно нужна, что делает нежелательным дальнейшее уменьшение размерности. Для алгоритма t-SNE характерно образовывать разрывы между разными классами для лучшей кластеризации и визуализации. t-Stochastic Neighbor Embedding может сократить сотни измерений к меньшему числу, при этом сохранить важные закономерности в данных, но иногда он застревает в локальном минимуме. Однако t-SNE может фиксировать нелинейные закономерности в данных. Понимая, как работают разные методы, а также зная их сильные и слабые стороны, позже, при решении соответствующих практических задач мы сможем уверенно использовать эти алгоритмы для анализа собранных данных.

Литература

1. Ван дер Маатен Л., Постма Э., Ван ден Херик Х.Я. Снижение размерности: сравнительный обзор // Journal of Machine Learning Research. 2010. №10. С. 1-41.

2. Шленс Д. Учебник по анализу главных компонент. Лаборатория системной нейробиологии, Калифорнийский университет в Сан-Диего, 2015. 12 с.

3. Хинтон Дж., Роуис С. Стохастическое вложение соседей - Нью-Йоркский университет, 2003. 20с.

4. Ван дер Маатен Л., Хинтон Дж. Визуализация данных с помощью t-sne// Journal of Machine Learning Research, 2008. 27с.

5. Девасси Б.М., Сони Дж. Уменьшение размерности и визуализация данных гиперспектральных чернил с использованием t-SNE // Science Direct. 2020. №311. С. 25-34.

Literature

1. Van der Maaten L., Postma E., Van Den Herik H.J. Dimensionality reduction: A comparative review // Journal of Machine Learning Research. 2010. №10. P. 1-41.

2. Shlens J. A tutorial on principal component analysis. Systems Neurobiology Laboratory, University of California at San Diego, 2015. 12 p.

3. Hinton G., Roweis S. Stochastic neighbor embedding - New York University, 2003. 20 p.

4. Van der Maaten L., Hinton G. Visualizing data using t-sne // Journal of Machine Learning Research, 2008. 27p.

5. Devassy B.M., Sony G. Dimensionality reduction and visualisation of hyperspectral ink data using t-SNE // Science Direct. 2020. №311. P. 25-34.