Оглавление (Ctrl+щелчок по разделу для быстрой навигации)

[Тестовые вопросы по MaxAbsScaler, MinMaxScaler, StandardScaler, RobustScaler 3](#_Toc187938870)

[Тестовые вопросы по StandardScaler 4](#_Toc187938871)

[Тестовые вопросы по MaxAbsScaler 5](#_Toc187938872)

[Тестовые вопросы по MinMaxScaler 6](#_Toc187938873)

[Тестовые вопросы по RobustScaler 7](#_Toc187938874)

[Тестовые вопросы по numpy.mean, numpy.std 8](#_Toc187938875)

[Тестовые вопросы по QuantileTransformer 9](#_Toc187938876)

[Тестовые вопросы по PowerTransformer 10](#_Toc187938877)

[Тестовые вопросы по KBinsDiscretizer 10](#_Toc187938878)

[Вопросы по PCA 12](#_Toc187938879)

[Вопросы по Kernel PCA 15](#_Toc187938880)

[Тестовые вопросы по Sparse PCA 16](#_Toc187938881)

[Тестовые вопросы по "Факторный анализ" 18](#_Toc187938882)

[Тестовые вопросы по sklearn.decomposition.FactorAnalysis 20](#_Toc187938883)

[Тестовые вопросы по "Ассоциативный анализ" 22](#_Toc187938884)

[Тестовые вопросы по mlxtend.frequent\_patterns.apriori 24](#_Toc187938885)

[Тестовые вопросы по FPGrowth 26](#_Toc187938886)

[Тестовые вопросы по association\_rules 28](#_Toc187938887)

[Тестовые вопросы по mlxtend.frequent\_patterns.fpmax 30](#_Toc187938888)

[Тестовые вопросы по KMeans 32](#_Toc187938889)

[Тестовые вопросы по sklearn KMeans 34](#_Toc187938890)

[Тестовые вопросы по "Метод Локтя" 37](#_Toc187938891)

[Тестовые вопросы по MiniBatchKMeans 43](#_Toc187938892)

[Тестовые вопросы по AgglomerativeClustering 46](#_Toc187938893)

[Тестовые вопросы по DBSCAN 50](#_Toc187938894)

[Тестовые вопросы по OPTICS 54](#_Toc187938895)

[Тестовые вопросы по GaussianNB 59](#_Toc187938896)

[Тестовые вопросы по MultinomialNB 62](#_Toc187938897)

[Тестовые вопросы по ComplementNB 64](#_Toc187938898)

[Тестовые вопросы по BernoulliNB 66](#_Toc187938899)

[Тестовые вопросы по GaussianNB, MultinomialNB, ComplementNB, BernoulliNB 67](#_Toc187938900)

[Тестовые вопросы по LinearDiscriminantAnalysis 69](#_Toc187938901)

[Тестовые вопросы по "Метод опорных векторов" 70](#_Toc187938902)

[Тестовые вопросы по sklearn SVC 71](#_Toc187938903)

[Тестовые вопросы по SVC, NuSVC и LinearSVC 72](#_Toc187938904)

Тестовые вопросы по MaxAbsScaler, MinMaxScaler, StandardScaler, RobustScaler

1. Какой метод масштабирования НЕ изменяет форму распределения данных?

????Странный вопрос, они все не изменяют форму графика (или так с графиками повезло), но MaxAbsScaler гарантированно не смещает нули, поэтому я думаю что это он

MaxAbsScaler просто масштабирует каждый признак result = x/|Xmax|

2. Какой метод масштабирования наиболее устойчив к выбросам?

RobustScaler

3. Какой метод масштабирования приводит данные к нулевому среднему и единичной дисперсии?

StandardScaler

4. Какой метод масштабирования использует медиану и межквартильный размах (IQR)?

RobustScaler – result=(x-median)/IQR

5. Какой метод масштабирования приводит данные к диапазону [-1, 1]?

MaxAbsScaler, если в данных есть и положительные, и отрицательные значения

MinMaxScaler, если указать feature\_range=(-1;1)

6. Какой метод масштабирования лучше всего подходит для разреженных данных?

MaxAbsScaler, потому что не смещает нули в данных

7. Если данные имеют длинный хвост и выбросы, какой метод масштабирован предпочтительнее?

RobustScaler меньше подвержен влиянию экстремальных значений/выбросов

8. Какой метод масштабирования использует минимум и максимум признака?

MinMaxScaler

Тестовые вопросы по StandardScaler

1. StandardScaler преобразует данные так, чтобы:

Среднее значение каждого признака стало 0, а стандартное отклонение std – 1, дисперсия - 1

2. StandardScaler чувствителен к выбросам?

Да, так как он использует чувствительные к выбросам стандартное отклонение и среднее значение для вычислений. **result = (x-mean)/std**

3. Какой атрибут объекта StandardScaler содержит средние значения, вычисленные для каждого признака?

scaler.mean\_

4. В каком случае рекомендуется использовать StandardScaler:

Когда данные должны быть стандартизированы для алгоритмов, чувствительных к масштабу (многие алгоритмы в машинном обучении)

5. Применение StandardScaler эквивалентно (как изменяются данные):

**result = (x-mean)/std** Данные центрируются и масштабируются до диапазона [0;1]

6. Что изменяет StandardScaler в исходных данных:

Среднее значение mean =0, стандартное отклонение std=1

7. Если данные уже имеют нулевое среднее и единичную дисперсию, применение

StandardScaler приводит к:

Минимальным изменениям из-за погрешности вычислений

x\_result=(x-mean)/std

mean=~0

std=~1

Тестовые вопросы по MaxAbsScaler

1. MaxAbsScaler масштабирует данные выполняя:

Деление каждого признака на его максимальное абсолютное значение

**x\_scaled=x/|Xmax|**

2. После применения MaxAbsScaler все значения признаков будут находиться в диапазоне:

[-1;1], если есть и отрицательные и положительные значения

[0;1], если есть только положительные значения

[-1;0], если есть только отрицательные значения

3. MaxAbsScaler хорошо подходит для данных:

Разреженных, т.к. не изменяет форму распределения данных

Когда надо сохранить нулевые элементы, знак значений и относительные расстояния между точками в признаке

4. В результате MaxAbsScaler сохраняет:

Разреженность данных, нулевые элементы, знак, относительные расстояния между точками в признаке

5. Если в данных есть только положительные значения, MaxAbsScaler будет работать

аналогично:

MinMaxScaler с feature\_range=(0, 1)

6. Что MaxAbsScaler изменяет в исходных данных:

Масштаб, нормируя данные относительно максимального абсолютного значения

7. Если максимальное абсолютное значение признака равно 1, то MaxAbsScaler:

Не изменит значений этого признака, т.к. этот признак уже масштабирован

**x\_scaled=x/|Xmax|**

**Xmax=1**

Тестовые вопросы по MinMaxScaler

1. MinMaxScaler масштабирует данные в диапазон:

[0;1] по умолчанию или указанный пользователем в feature\_range=(min,max)

2. Что MinMaxScaler использует для масштабирования:

**X\_scaled=((X-Xmin)/(Xmax-Xmin))\*(max-min)+min**

3. MinMaxScaler чувствителен к выбросам?

Да, он чувствительный к выбросам, т.к. использует чувствительные для выбросов характеристики – максимум и минимум

4. Какой атрибут объекта MinMaxScaler содержит минимальные значения, вычисленные для каждого признака?

scaled.min\_

5. Что MinMaxScaler сохраняет в исходных данных:

Отношения между значениями, относительные расстояния между точками внутри признака

6. Какой параметр MinMaxScaler позволяет задать нужный диапазон масштабирования?

feature\_range=(min,max)

7. Если данные уже находятся в диапазоне [0, 1], применение MinMaxScaler:

Не изменит данные

Тестовые вопросы по RobustScaler

1. Что RobustScaler использует для масштабирования:

X\_scaled=(X-Xmedian)/IQR=(X-Xmedian)/(Q3-Q1)

2. RobustScaler менее чувствителен к выбросам, чем:

StandardScaler, MinMaxScaler, MaxAbsScaler, т.к. использует устойчивые к выбросам медиану и межквартальный размах IQR

3. Какой параметр RobustScaler контролирует диапазон квантилей для вычисления IQR?

quantile\_range=(25,75) по умолчанию

4. ***with\_centering=False*** в RobustScaler означает:

Данные не будут центрироваться по медиане, только масштабируются по IQR, нули останутся нулями

5. RobustScaler рекомендуется использовать, когда:

Данные содержат выбросы

6. После применения RobustScaler с параметрами по умолчанию, медиана каждого признака будет:

Равна 0, т.к. данные центрируются по медиане

7. IQR (Interquartile Range) вычисляется как:

IQR=Q3-Q1, где

Q3 – третий квартиль (75-й процентиль) (25% чисел меньше этого значения)

Q1- первый квартиль (25-й процентиль) (75% чисел меньше этого значения)

Тестовые вопросы по numpy.mean, numpy.std

1. **Функция np.mean() вычисляет:** среднее арифметическое.
2. **Функция np.std() вычисляет:** стандартное отклонение.
3. **Для вычисления среднего значения элементов вдоль строк двумерного массива arr следует использовать:** np.mean(arr, axis=1).
4. **Для вычисления стандартного отклонения элементов вдоль столбцов двумерного массива arr следует использовать:** np.std(arr, axis=0).
5. **Параметр ddof в функции np.std() определяет:** степень свободы (по умолчанию 0, для выборочной дисперсии установить 1).
6. **Если в массиве есть значения NaN, np.mean() вернет:** NaN.
7. **Чтобы проигнорировать NaN при вычислении среднего значения, нужно использовать:** np.nanmean().
8. **Чтобы вычислить взвешенное среднее значение, следует использовать:** np.average().

Тестовые вопросы по QuantileTransformer

1. **QuantileTransformer преобразует данные так, чтобы:** распределение соответствовало выбранному (uniform или normal).
2. **Какой параметр QuantileTransformer определяет количество квантилей?** n\_quantiles.
3. **Какой параметр QuantileTransformer позволяет выбрать между равномерным и нормальным распределением преобразованных данных?** output\_distribution.
4. **Если output\_distribution='normal', QuantileTransformer преобразует данные так, чтобы они приблизительно соответствовали:** нормальному распределению.
5. **QuantileTransformer устойчив к выбросам?** Да.
6. **Параметр random\_state в QuantileTransformer используется для:** воспроизводимости результатов.
7. **Если в данных есть пропущенные значения (NaN), QuantileTransformer:** вызывает ошибку (их нужно обработать заранее).

Тестовые вопросы по PowerTransformer

* 1. **Основная цель PowerTransformer:** стабилизировать дисперсию и сделать данные более нормальными.
  2. **PowerTransformer использует следующую технику для преобразования данных:** нелинейные преобразования.
  3. **Какой параметр PowerTransformer определяет метод преобразования?** method.
  4. **Значение 'yeo-johnson' параметра method означает:** преобразование, подходящее для данных с отрицательными значениями.
  5. **Параметр standardize в PowerTransformer:** масштабирует данные после преобразования.
  6. **PowerTransformer подходит для данных:** с положительными и отрицательными значениями (в зависимости от метода).
  7. Box-Cox transformation можно применять только к: положительным значениям.

Тестовые вопросы по KBinsDiscretizer

1. **KBinsDiscretizer используется для:** дискретизации непрерывных данных.
2. **Параметр n\_bins в KBinsDiscretizer определяет:** количество интервалов.
3. **Какие стратегии разбиения на интервалы поддерживает KBinsDiscretizer (параметр strategy)?** uniform, quantile, kmeans.
4. **Стратегия uniform в KBinsDiscretizer:** равные по ширине интервалы.
5. **Стратегия quantile в KBinsDiscretizer:** равное количество элементов в каждом интервале.
6. **Параметр encode в KBinsDiscretizer определяет способ кодирования преобразованных признаков:** ordinal, onehot, onehot-dense.
7. **encode='ordinal' в KBinsDiscretizer преобразует данные в:** целочисленные метки.
8. **KBinsDiscretizer может обрабатывать пропущенные значения (NaN)?** Нет (требуется предварительная обработка).

Вопросы по PCA

1. PCA используется для:

Снижения размерности данных при сохранении максимально возможной вариации.

2. PCA находит:

Ортогональные направления (главные компоненты), вдоль которых данные имеют наибольшую дисперсию.

3. Результатом работы PCA являются:

Главные компоненты (направления векторного пространства) и преобразованные данные в новом базисе.

4. Главные компоненты являются:

Линейными комбинациями исходных признаков.

5. Параметр n\_components в PCA определяет:

Количество главных компонент, которые нужно оставить.

6. Параметр svd\_solver в PCA определяет:

Алгоритм для выполнения сингулярного разложения (например, «auto», «full», «arpack», «randomized»).

7. explained\_variance\_ratio\_ в PCA показывает:

Долю дисперсии, объясненную каждой главной компонентой.

8. Перед применением PCA рекомендуется:

Стандартизировать данные, чтобы все признаки имели одинаковый масштаб.

Дополнительные вопросы по методу главных компонент (PCA)

1. Основная цель PCA:

Упростить данные путем снижения размерности с минимальной потерей информации.

2. PCA ищет:

Направления наибольшей вариации данных в многомерном пространстве.

3. Результатом PCA являются:

Матрица преобразованных данных, собственные значения и собственные векторы.

4. Главные компоненты:

Ортогональны друг другу и упорядочены по убыванию дисперсии.

5. n\_components в PCA определяет:

Число главных компонент или долю сохраненной дисперсии (если задано как число от 0 до 1).

6. explained\_variance\_ratio\_ в PCA показывает:

Какую часть общей дисперсии данных объясняет каждая главная компонента.

7. Перед применением PCA рекомендуется:

Привести данные к нулевому среднему и единичной дисперсии.

8. PCA чувствителен к масштабу признаков?

Да, поэтому стандартизация данных обязательна.

9. Какой метод чаще всего используется для вычисления главных компонент в PCA?

Сингулярное разложение (SVD).

10. Что такое собственные значения в контексте PCA?

Меры дисперсии, объясняемой каждой главной компонентой.

11. components\_ в PCA содержит:

Коэффициенты линейных комбинаций признаков для главных компонент.

12. mean\_ в PCA содержит:

Средние значения для каждого признака в данных.

13. PCA может использоваться для:

Визуализации данных, уменьшения размерности, устранения многоколлинеарности.

14. Если n\_components задан как число от 0 до 1, то это:

Доля дисперсии, которую необходимо сохранить.

15. whiten=True в PCA:

Преобразует данные так, чтобы каждая главная компонента имела единичную дисперсию.

16. PCA является:

Линейным методом снижения размерности.

17. inverse\_transform в PCA:

Восстанавливает данные из пространства главных компонент в исходное пространство.

18. Какой параметр отвечает за случайность в PCA?

Параметр random\_state при использовании randomized SVD.

19. svd\_solver=«arpack» рекомендуется для:

Малых объемов данных и низкого числа компонент.

20. PCA гарантирует:

Максимальную сохранность дисперсии данных в первых компонентах.

Вопросы по Kernel PCA

1. Какая из следующих функций ядра НЕ является допустимой для Kernel PCA?

Любая, которая не является положительно определенной (например, несимметричные функции).

2. Что из перечисленного является целью Kernel PCA?

Нелинейное снижение размерности путем преобразования данных в более высокомерное пространство.

3. Какое из следующих утверждений о Kernel PCA является верным?

Kernel PCA использует ядра для учета нелинейных зависимостей в данных.

4. Какая из следующих матриц используется для вычисления собственных значений и собственных векторов в Kernel PCA?

Матрица ядра (kernel matrix).

5. Что из перечисленного является преимуществом Kernel PCA?

Способность выявлять сложные, нелинейные структуры данных.

Тестовые вопросы по Sparse PCA

1. Какой параметр в SparsePCA используется для управления степенью разреженности? - **"alpha"** или **"regularization parameter"**. Он определяет уровень регуляризации, который применяется для достижения разреженных компонент.

2. Какой метод в SparsePCA используется для вычисления компонент? - метод **"Principal Component Analysis" (PCA)** с добавлением регуляризации, чтобы обеспечить разреженность компонент. Это может быть реализовано через методы, такие как **Lasso** или другие подходы к регуляризации.

3. Какой атрибут в SparsePCA содержит коэффициенты разреженных компонент? - **"components\_"**. Этот атрибут хранит значения весов для каждой из компонент.

4. Какой метод в SparsePCA используется для восстановления исходных данных из спроецированных данных? - метод **"inverse transform"**, который позволяет преобразовать спроецированные данные обратно в исходное пространство.

5. Что означает "разреженность" в контексте Sparse PCA? - "Разреженность" в контексте Sparse PCA означает наличие большого количества нулевых или близких к нулю значений в коэффициентах компонент. Это позволяет выделить только наиболее значимые признаки и игнорировать менее важные, что делает модель более интерпретируемой и устойчивой к шуму.

6. Какой из следующих методов регуляризации чаще всего используется в Sparse PCA? - метод регуляризации **L1**, который способствует созданию разреженных решений за счет добавления штрафа за величину коэффициентов.

7. Как влияет увеличение параметра регуляризации (например, alpha в Scikit-Learn) на разреженность компонент в Sparse PCA? - Увеличение параметра регуляризации (alpha) приводит к увеличению разреженности компонент, так как более высокий уровень регуляризации заставляет модель устанавливать больше коэффициентов равными нулю, что уменьшает количество активных признаков.

8. В каких случаях Sparse PCA может быть предпочтительнее стандартного PCA? - в случаях, когда необходимо выделить лишь несколько значимых признаков из большого набора данных, особенно если данные содержат много шумов или нерелевантных признаков.

9. Какой из следующих недостатков характерен для Sparse PCA? - Один из недостатков Sparse PCA заключается в том, что он может быть более чувствителен к выбору параметров регуляризации и может требовать больше вычислительных ресурсов по сравнению со стандартным PCA, особенно при работе с большими наборами данных.

Тестовые вопросы по "Факторный анализ"

1. Какова основная цель факторного анализа? - Основная цель факторного анализа заключается в уменьшении размерности данных и выявлении скрытых факторов, которые объясняют взаимосвязи между наблюдаемыми переменными. Это позволяет упростить данные и выявить структуры, которые могут быть неочевидны при прямом анализе.

2. Что такое "фактор" в факторном анализе? - "Фактор" в факторном анализе представляет собой скрытую переменную, которая объясняет корреляции между наблюдаемыми переменными. Факторы помогают понять, какие группы переменных имеют схожие характеристики и как они взаимодействуют друг с другом.

3. Какой тип данных подходит для факторного анализа? - Для факторного анализа подходят данные, которые являются количественными (например, интервальные или отношения). Однако также возможно применение методов для категориальных данных после их соответствующей кодировки.

4. Что такое "факторная нагрузка"? - Факторная нагрузка — это коэффициент, который показывает степень влияния конкретной переменной на определенный фактор. Она указывает на то, насколько сильно переменная связана с фактором, и помогает интерпретировать значение фактора.

5. Какой метод вращения факторов используется для упрощения интерпретации факторов? - Для упрощения интерпретации факторов часто используется метод **оборачивания (rotation)**, такой как **Varimax**, который максимизирует количество высоких нагрузок и минимизирует низкие нагрузки на факторы, делая их более интерпретируемыми.

6. Что такое "собственное значение" в факторном анализе? - "Собственное значение" (eigenvalue) в факторном анализе отражает количество вариации в данных, объясняемой конкретным фактором. Чем выше собственное значение, тем больше информации этот фактор содержит.

7. Какое правило используется для определения количества факторов, которые следует оставить? - Одно из распространенных правил — оставить те факторы, у которых собственное значение больше 1 (правило Кайзера). Также можно использовать график "локтя" для визуального определения точки, где добавление новых факторов не дает значительного увеличения объясненной вариации.

8. Что такое "общность" (communality) в факторном анализе? - Общность — это доля вариации каждой переменной, которая объясняется факторами в модели. Она показывает, насколько хорошо факторы объясняют каждую переменную; высокие значения общности указывают на то, что переменная хорошо представлена факторами.

9. Какой из следующих методов НЕ является методом извлечения факторов? - Методы извлечения факторов включают методы главных компонент (PCA), максимального правдоподобия и других подходов. Метод **регрессии** не является методом извлечения факторов.

10. Факторный анализ является \_\_\_ методом. - Факторный анализ является **методом уменьшения размерности** и **методом многомерной статистики**, который используется для анализа взаимосвязей между переменными и выявления скрытых структур в данных.

Тестовые вопросы по sklearn.decomposition.FactorAnalysis

1. Какой метод используется в ***sklearn.decomposition.FactorAnalysis*** для оценки параметров модели? - В sklearn.decomposition.FactorAnalysis используется метод **максимального правдоподобия** для оценки параметров модели.

2. Какой параметр в ***FactorAnalysis*** отвечает за количество факторов, которые необходимо извлечь? - Параметр, отвечающий за количество факторов, которые необходимо извлечь, называется **n\_components**.

3. Какой параметр в ***FactorAnalysis*** управляет типом вращения факторов? - Параметр, который управляет типом вращения факторов, называется **rotation**. Он может принимать значения, такие как 'varimax', 'promax' и другие.

4. Какой атрибут обученного объекта ***FactorAnalysis*** содержит матрицу факторных нагрузок? - Атрибут, который содержит матрицу факторных нагрузок в обученном объекте FactorAnalysis, называется **components\_**.

5. Какой метод в ***FactorAnalysis*** используется для трансформации новых данных с использованием обученной модели? - Метод, используемый для трансформации новых данных с использованием обученной модели, называется **transform()**.

6. Какой параметр в ***FactorAnalysis*** отвечает за инициализацию факторных нагрузок? - Параметр, отвечающий за инициализацию факторных нагрузок, называется **init**. Он определяет метод инициализации (например, 'random' или 'pca').

7. Какой параметр в ***FactorAnalysis*** позволяет контролировать максимальное количество итераций при оценке параметров? - Параметр, который позволяет контролировать максимальное количество итераций при оценке параметров, называется **max\_iter**.

8. Какой атрибут обученного объекта ***FactorAnalysis*** содержит ковариационную матрицу шума? - Атрибут, который содержит ковариационную матрицу шума в обученном объекте FactorAnalysis, называется **noise\_variance\_**.

9. Какой параметр в ***FactorAnalysis*** отвечает за допуск сходимости алгоритма? - Параметр, который отвечает за допуск сходимости алгоритма, называется **tol** (tolerance).

10. Какой метод в ***FactorAnalysis*** используется для вычисления логарифмической

вероятности данных? - Метод, используемый для вычисления логарифмической вероятности данных в FactorAnalysis, называется **score()**.

Тестовые вопросы по "Ассоциативный анализ"

1. Что является основной целью ассоциативного анализа?

Целью ассоциативного анализа, является нахождения всех правил XY(зависимость между связанными событиями или элементами), причем поддержка и достоверность этих правил должны быть выше некоторых наперед определенных порогов, называемых соответственно минимальной поддержкой (minsupport) и минимальной достоверностью (minconfidence).

2. Какой тип данных чаще всего используется в ассоциативном анализе?

Для реализации работы с алгоритмами выделения ассоциаций чаще всего используется специальные типы данных, относящиеся к объектам трех классов: входной массив транзакций (transactions) и на выходе - часто встречающиеся фрагменты данных (itemsets) и правила (rules).

3. Что такое "поддержка" (support) правила в ассоциативном анализе?

"Поддержка" (support) – это одна из метрик, которая используется для измерения распространенности или частоты набора элементов в базе данных.

4. Что такое "достоверность" (confidence) правила в ассоциативном анализе?

"Достоверность" (confidence) - это одна из метрик, которая измеряет вероятность того, что если в транзакции присутствует антецедент (условие), то в этой же транзакции будет присутствовать и консеквент (следствие).

5. Что такое "подъем" (lift) правила в ассоциативном анализе?

"Подъем" (lift) - это одна из метрик, которая измеряет, насколько более вероятно появление консеквента (следствия) в транзакции, содержащей антецедент (условие), по сравнению с вероятностью появления консеквента в произвольной транзакции.

6. Какой алгоритм часто используется для поиска ассоциативных правил?

Наиболее часто используемым алгоритмом для поиска ассоциативных правил является алгоритм Apriori.

7. Что такое "антецедент" в ассоциативном правиле?

В ассоциативных правилах антецедент указывает условия, которые, если выполняются, могут привести к выполнению другого условия.

8. Что такое "консеквент" в ассоциативном правиле?

Консеквент представляет собой результатом или событие, которое вероятно произойдет, если выполнены условия, заданные антецедентом (предпосылкой).

9. Какое значение "подъема" (lift) указывает на независимость антецедента и консеквента?

Значение подъема (lift), равное 1, указывает на независимость антецедента и консеквента в ассоциативном правиле.

10. Ассоциативный анализ чаще всего применяется в: Анализ корзины покупок, формирование товарных рекомендации, планирование и размещение товаров в магазине.

Тестовые вопросы по mlxtend.frequent\_patterns.apriori

1. Какой тип данных принимает функция ***apriori*** из библиотеки ***mlxtend.frequent\_patterns*** ?

DataFrame

2. Какой параметр в функции ***apriori*** отвечает за минимальную поддержку?

min\_support

3. Какой параметр в функции ***apriori*** позволяет использовать названия столбцов DataFrame вместо индексов?

use\_colnames

4. Какой тип данных возвращает функция ***apriori*** ?

DataFrame

5. Какой столбец в возвращаемом DataFrame содержит найденные часто встречающиеся наборы элементов?

itemsets

6. Какой столбец в возвращаемом DataFrame содержит значение поддержки для каждого часто встречающегося набора?

support

7. Как можно отфильтровать результаты, полученные с помощью ***apriori*** , по длине наборов элементов?

Использовать фильтрацию по столбцу itemsets, извлекая длину набора

8. Для чего используется параметр ***low\_memory*** в функции ***apriori*** ?

Для снижения потребления памяти при обработке больших наборов данных.

9. Какой параметр в функции ***apriori*** позволяет задать максимальную длину наборов

элементов?

max\_len

10. Что нужно сделать перед использованием функции ***apriori*** , если данные представлены в виде списка транзакций с текстовыми элементами?

Данные надо закодировать так, чтобы их можно было представить в виде матрицы. Для кодированния данных используем TransactionEncode.

Тестовые вопросы по FPGrowth

1. Что является основной задачей алгоритма FPGrowth?

Основной задачей алгоритма FPGrowth - поиск часто встречающихся наборов элементов.

2. Какое основное преимущество FPGrowth по сравнению с Apriori?

Основное преимущество FPGrowth по сравнению с Apriori - позволяет избежать затратной процедуры генерации кандидатов, характерной для алгоритма Apriori.

3. Какая структура данных используется в алгоритме FPGrowth для хранения информации о транзакциях?

Структура данных FP-дерево (Frequent Pattern Tree).

4. Что такое "условная база паттернов" (conditional pattern base) в алгоритме FPGrowth?

Набор транзакций, содержащих определенный элемент, с удалением элементов, предшествующих ему в дереве.

5. Что такое "условное FP-дерево" (conditional FP-tree) в алгоритме FPGrowth?

FP-дерево, построенное на основе условной базы паттернов.

6. Как влияет порядок элементов в транзакциях на построение FP-дерева?

Порядок влияет на структуру дерева и его эффективность.

7. Какой параметр определяет минимальную поддержку в алгоритме FPGrowth?

min\_support

8. FPGrowth, как и Apriori, используется на первом этапе поиска:

Частых наборов элементов.

9. Какой из следующих недостатков может быть у алгоритма FPGrowth?

Может быть ресурсозатратным при очень больших данных или высокой частоте элементов

10. После нахождения часто встречающихся наборов с помощью FPGrowth, что обычно делается для получения ассоциативных правил?

Генерация ассоциативных правил на основе найденных частых наборов.

Тестовые вопросы по association\_rules

1. Для чего используется функция ***association\_rules*** в библиотеке ***mlxtend.frequent\_patterns*** ?

Формирование ассоциативных правил

2. Какой тип данных принимает функция ***association\_rules*** в качестве основного аргумента?

Основным аргументом функции является DataFrame, содержащий частые наборы (frequent itemsets), который обычно получается в результате работы функций, таких как apriori или fpgrowth. Этот DataFrame должен содержать столбцы с элементами наборов (itemsets) и их поддержкой (support).

3. Какой параметр в функции ***association\_rules*** отвечает за метрику, используемую для оценки правил?

Параметр metric определяет метрику, которая будет использоваться для оценки правил.

metric="confidence"

4. Какие метрики можно использовать для оценки ассоциативных правил с помощью функции ***association\_rules*** ?

**Поддержка (Support):** Доля транзакций, содержащих определенный набор элементов.

Например, если набор {молоко, хлеб} встречается в 20% транзакций, его поддержка

равна 0.2.

**Доверие (Confidence):** Условная вероятность того, что транзакция, содержащая набор

элементов A, также содержит набор элементов B. Вычисляется как support(A ∪ B) /

support(A) . Например, если поддержка {молоко, хлеб} равна 0.2, а поддержка {молоко}

равна 0.5, то доверие правила {молоко} → {хлеб} равно 0.2 / 0.5 = 0.4.

**Подъем (Lift):** Показывает, насколько часто элементы A и B встречаются вместе по

сравнению с тем, как часто они встречались бы, если бы были независимыми.

Вычисляется как support(A ∪ B) / (support(A) \* support(B)) . Значение подъема больше

1 указывает на положительную корреляцию, меньше 1 - на отрицательную, а равное 1 – на независимость.

leverage (рычаг)

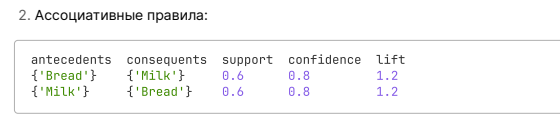
conviction (убежденность)

5. Какой параметр в функции ***association\_rules*** отвечает за минимальный порог метрики, при котором правило считается интересным?

Параметр min\_threshold

6. Какой тип данных возвращает функция ***association\_rules*** ?

Функция возвращает объект pandas.DataFrame, содержащий сгенерированные ассоциативные правила и их метрики.



7. Какие столбцы обычно присутствуют в DataFrame, возвращаемом функцией ***association\_rules*** ?

antecedents (левые части правил)

consequents (правые части правил)

support (поддержка всего правила)

confidence (доверие)

lift (подъем)

antecedent support (поддержка левой части правила)

consequent support (поддержка правой части правила)

Дополнительные метрики: leverage, conviction (если применимо).

8. Что представляют собой столбцы antecedents и consequents в DataFrame, возвращаемом функцией ***association\_rules*** ?

antecedents (левые части правил)

consequents (правые части правил)

antecedents — это наборы элементов, которые выступают в качестве условий (левой части правила).

consequents — это наборы элементов, которые следуют из условий (правой части правила).

9. Как можно отфильтровать правила, сгенерированные функцией ***association\_rules*** , по значению метрики (например, lift)?

rules = association\_rules(frequent\_itemsets, metric="lift", min\_threshold=1.0)

filtered\_rules = rules[rules['lift'] > 1.5]

10. Что нужно сделать перед использованием функции ***association\_rules*** , если у вас есть только исходные транзакционные данные?

Генерация частых наборов с порогом поддержки с помощью библиотеки apriori

Преобразовать транзакционные данные в формат one-hot encoding (например, с помощью TransactionEncoder).

Сгенерировать частые наборы элементов с помощью функции apriori или fpgrowth, указав минимальный порог поддержки (min\_support).

Использовать полученные частые наборы в функции association\_rules.

Тестовые вопросы по mlxtend.frequent\_patterns.fpmax

1. Что является основной задачей алгоритма FPMax?

FP-Max — это вариант FP-Growth, который фокусируется на получении максимальных наборов элементов. **Набор элементов X называется максимальным, если он является часто встречающимся и не существует часто встречающегося супершаблона, содержащего X.**

2. Что такое "максимальный часто встречающийся набор элементов" в контексте FPMax?

**Набор элементов X называется максимальным, если он является часто встречающимся и не существует часто встречающегося супершаблона, содержащего X.**

Другими словами, часто встречающийся шаблон X не может быть подшаблоном более крупного часто встречающегося шаблона, чтобы соответствовать определению *максимального набора элементов*.

3. Какой тип данных принимает функция fpmax из библиотеки mlxtend.frequent\_patterns ?

Функция fpmax принимает DataFrame в формате one-hot encoding, где строки представляют транзакции, а столбцы — возможные элементы, со значениями 1 (элемент присутствует) и 0 (элемент отсутствует).

4. Какой параметр в функции fpmax отвечает за минимальную поддержку?

Параметр min\_support

5. Какой параметр в функции fpmax позволяет использовать названия столбцов DataFrame вместо индексов?

Параметр use\_colnames=True (по умолчанию: False)

6. Какой тип данных возвращает функция fpmax ?

Функция возвращает объект pandas.DataFrame, который содержит максимальные часто встречающиеся наборы элементов и их поддержку.

7. Какой столбец в возвращаемом DataFrame содержит найденные максимальные часто встречающиеся наборы элементов?

Столбец itemsets

8. Какой столбец в возвращаемом DataFrame содержит значение поддержки для каждого максимального часто встречающегося набора?

Столбец support

9. По сравнению с алгоритмом FPGrowth, FPMax обычно:

Находит только максимальные часто встречающиеся наборы, что сокращает объем данных.

Может быть более эффективным в задачах, где интересны только максимальные наборы.

Упускает информацию о поддержке подмножеств, в отличие от FPGrowth.

10. Что нужно сделать перед использованием функции fpmax , если данные представлены в виде списка транзакций с текстовыми элементами?

Преобразовать данные в формат one-hot encoding, используя TransactionEncoder или аналогичный инструмент.

Убедиться, что DataFrame соответствует требованиям функции fpmax (значения 0 и 1).

Тестовые вопросы по KMeans

1. KMeans - это алгоритм:

KMeans — это алгоритм кластеризации, который относится к методам обучения без учителя (unsupervised learning).

2. Что является основной целью алгоритма KMeans?

KMeans — разделить данные на k кластеров таким образом, чтобы объекты внутри одного кластера были максимально похожи, а объекты из разных кластеров — максимально различны.

3. Как KMeans определяет сходство между объектами?

KMeans определяет сходство между объектами с помощью **евклидова расстояния** (по умолчанию), измеряя расстояние между точками данных и центроидами кластеров.

4. Что такое центроиды в алгоритме KMeans?

Центроиды — это центральные точки кластеров, которые представляют собой среднее значение всех объектов, принадлежащих к данному кластеру.

5. Как KMeans выбирает начальное расположение центроидов?

Начальное расположение центроидов выбирается случайным образом из данных. Для улучшения качества начальной инициализации часто используется метод **KMeans++**, который выбирает стартовые центроиды таким образом, чтобы они были максимально удалены друг от друга.

6. Что такое метод локтя (elbow method) в KMeans?

**Метод локтя** — это подход для определения оптимального числа кластеров k. Он основан на анализе зависимости значения инерции (суммы квадратов расстояний от точек до их центроидов) от числа кластеров. Оптимальное k соответствует "изгибу" на графике, где уменьшение инерции становится менее значительным.

7. Что такое метод силуэта (silhouette method) в KMeans?

Метод силуэта измеряет качество кластеризации, оценивая, насколько каждый объект хорошо относится к своему кластеру по сравнению с соседними кластерами.

Коэффициент силуэта принимает значения от -1 до 1:

* Значения близкие к 1 означают, что объект правильно классифицирован.
* Значения близкие к 0 указывают на неопределенность.
* Отрицательные значения означают, что объект классифицирован неверно.

8. Какой из следующих недостатков характерен для KMeans?

Чувствительность к выбросам и шуму.

Неоптимальность при неправильной инициализации центроидов.

Требование заранее задавать число кластеров k.

Предположение, что кластеры имеют сферическую форму и примерно равны по размеру.

9. Как можно уменьшить влияние выбросов на KMeans?

Использовать метод KMedoids, который минимизирует сумму расстояний до медианы вместо среднего.

Применять предварительную обработку данных (удаление выбросов).

Ограничить влияние выбросов с помощью алгоритмов, подобных MiniBatchKMeans.

10. KMeans гарантированно находит глобальный оптимум?

Нет, KMeans не гарантирует нахождение глобального оптимума, поскольку результат зависит от начальной инициализации центроидов.

Тестовые вопросы по sklearn KMeans

1. Какой класс в Scikit-Learn используется для реализации алгоритма KMeans?

В библиотеке Scikit-Learn для реализации алгоритма K-Means используется класс sklearn.cluster.KMeans.

2. Какой параметр в ***KMeans*** отвечает за количество кластеров?

Параметр n\_clusters отвечает за количество кластеров. Это обязательный параметр, который задаёт число кластеров, вокруг которых алгоритм будет находить центроиды.

3. Какой параметр в ***KMeans*** отвечает за метод инициализации центроидов?

Метод инициализации центроидов задаётся параметром init.

Возможные значения:

* 'k-means++' (по умолчанию): улучшенная стратегия, которая выбирает начальные центроиды, чтобы ускорить сходимость.
* 'random': случайная инициализация центроидов.
* Также можно задать массив с предопределёнными координатами начальных центроидов.

Соответственно, пример настройки алгоритма K-Means может выглядеть так:

from sklearn.cluster import KMeans

kmeans = KMeans(

n\_clusters=3, *# Количество кластеров*

init='k-means++' *# Метод инициализации*

)

4. Какие значения может принимать параметр ***init*** в ***KMeans*** ?

Параметр init отвечает за метод инициализации центроидов. Он может принимать следующие значения:

* + 'k-means++' (по умолчанию): выбирает центроиды умным образом, чтобы ускорить сходимость.
  + 'random': случайный выбор начальных центроидов.
  + Массив (ndarray) с заранее заданными координатами начальных центроидов. Размерность должна быть (n\_clusters, n\_features).

Пример использования параметра init:

kmeans = KMeans(init='random') *# Используется случайная инициализация*

5. Какой параметр в ***KMeans*** отвечает за максимальное количество итераций алгоритма?

Параметр max\_iter отвечает за максимальное количество итераций алгоритма k-means для одного запуска. По умолчанию max\_iter=300.

6. Какой атрибут обученного объекта ***KMeans*** содержит метки кластеров для каждой точки данных?

Атрибут labels\_ содержит метки кластеров для каждой точки данных после выполнения алгоритма. Метки представляют собой индексы (целые числа), указывающие, к какому кластеру принадлежит каждая точка.

Пример:

kmeans = KMeans(n\_clusters=3)

kmeans.fit(data) *# Обучение модели*

print(kmeans.labels\_) *# Метки кластеров для каждой точки данных*

7. Какой атрибут обученного объекта ***KMeans*** содержит координаты центроидов кластеров?

Атрибут cluster\_centers\_ содержит координаты центроидов кластеров после выполнения алгоритма. Это массив размером (n\_clusters, n\_features), где каждая строка представляет координаты центра конкретного кластера.

Пример:

kmeans = KMeans(n\_clusters=3)

kmeans.fit(data)

print(kmeans.cluster\_centers\_) *# Координаты центроидов кластеров*

8. Какой метод в ***KMeans*** используется для предсказания кластера для новых точек данных?

Метод predict() используется для предсказания кластера, к которому принадлежит каждая новая точка данных. Этот метод принимает массив новых данных и возвращает массив меток кластеров.

Пример:

new\_points = [[1, 2], [3, 4]]

predicted\_labels = kmeans.predict(new\_points)

print(predicted\_labels) *# Предсказанные кластеры для новых точек*

9. Какой метод в ***KMeans*** используется для вычисления суммы квадратов расстояний от каждой точки до ближайшего центроида?

Метод inertia\_ (атрибут) используется для возвращения суммы квадратов расстояний от каждой точки до ее ближайшего центроида. Это значение оценивает, насколько хорошо точки сгруппированы в кластеры (чем меньше значение, тем лучше). inertia\_ автоматически вычисляется после обучения модели.

Пример:

kmeans = KMeans(n\_clusters=3)

kmeans.fit(data)

print(kmeans.inertia\_) *# Сумма квадратов расстояний до ближайших центроидов*

10. Какой параметр в ***KMeans*** позволяет задать случайное состояние для воспроизводимости результатов?

Параметр random\_state позволяет задать начальное состояние генератора случайных чисел для обеспечения воспроизводимости. Он принимает int или None (по умолчанию).

Пример:

kmeans = KMeans(n\_clusters=3, random\_state=42) *# Установка random\_state для воспроизводимых результатов*

Тестовые вопросы по "Метод Локтя"

Пример визуализации метода локтя:

import matplotlib.pyplot as plt

from sklearn.cluster import KMeans

from sklearn.datasets import make\_blobs

*# Генерация данных для примера*

data, \_ = make\_blobs(n\_samples=300, centers=4, random\_state=42)

inertia = []

for k in range(1, 10):

kmeans = KMeans(n\_clusters=k, random\_state=42)

kmeans.fit(data)

inertia.append(kmeans.inertia\_)

*# Построение графика*

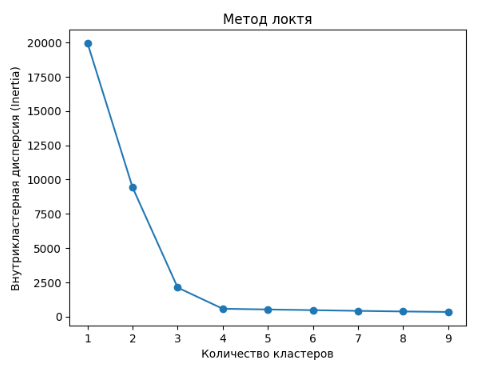
plt.plot(range(1, 10), inertia, marker='o')

plt.title('Метод локтя')

plt.xlabel('Количество кластеров')

plt.ylabel('Внутрикластерная дисперсия (Inertia)')

plt.show()



В этом коде график покажет "локоть", который поможет определить оптимальное число кластеров для заданного набора данных.

1. Для чего используется метод локтя?

Метод локтя используется для определения оптимального количества кластеров в модели кластеризации. Цель заключается в том, чтобы найти такое количество кластеров, при котором дальнейшее увеличение числа кластеров не приводит к значительному снижению внутрикластерной дисперсии (или инерции, inertia\_), показывая компромисс между количеством кластеров и качеством их формирования.

2. На каком графике основан метод локтя?

Метод локтя основан на графике зависимости внутрикластерной дисперсии (inertia\_) от количества кластеров (n\_clusters). На графике отображается:

* Ось X: количество кластеров (n\_clusters).
* Ось Y: значение параметра инерции (сумма квадратов расстояний до ближайших центроидов).

3. Как определяется оптимальное количество кластеров с помощью метода локтя?

Оптимальное количество кластеров определяется как точка, где происходит "локоть" графика — т.е. место, где снижение внутрикластерной дисперсии резко замедляется при увеличении числа кластеров. Это указывает на то, что добавление новых кластеров после этой точки даёт лишь незначительное улучшение качества кластеризации.

Пример определения:

1. Постройте график зависимости inertia\_ от n\_clusters.
2. Найдите "перелом" линии, где выгода от увеличения кластеров уменьшается.

4. Является ли метод локтя точным методом определения оптимального количества кластеров?

Нет, метод локтя не является точным методом. Он ориентирован на визуальный анализ данных и построение логических предположений. Иногда "локоть" может быть плохо выражен, из-за чего выбор оптимального количества кластеров становится субъективным.

5. Какие недостатки имеет метод локтя?

Метод локтя имеет несколько недостатков:

1. Субъективность: Определение "локтя" зачастую носит субъективный характер, особенно если перелом в графике выражен нечетко.
2. Ограниченность применения: Метод подходит только для анализа кластеризации, основанной на минимизации внутрикластерной дисперсии (например, KMeans), и для других методов кластеризации может быть неприменим.
3. Чувствительность к данным: График может быть сильно искажен шумами и выбросами в данных, что затруднит нахождение оптимального числа кластеров.
4. Масштабируемость: При большом количестве кластеров метод может стать менее информативным, поскольку значения дисперсии уменьшаются медленно.

6. Какие альтернативные методы можно использовать для определения оптимального количества кластеров?

Если метод локтя не подходит или недостаточно точен, можно использовать следующие альтернативные методы:

1. Метод "Силуэтов" (Silhouette Method):
   * Оценивает, насколько хорошо каждая точка данных принадлежит своему кластеру, сравнивая средние расстояния внутри и между кластерами.
   * Оптимальное количество кластеров — то, где коэффициент силуэта (silhouette\_score) максимален.
2. Критерий Гапа (Gap Statistic):
   * Сравнивает суммарную дисперсию в кластеризации с ее ожидаемым значением в случайных данных.
   * Оптимальное число кластеров — значение, при котором происходит наибольшее отклонение от случайного распределения.
3. Информационные критерии (AIC и BIC):
   * Используются для кластеризации моделей, таких как Gaussian Mixture Model (GMM).
   * Основаны на максимизации правдоподобия и penalization (штрафам) за большое количество кластеров.
4. Индекс Дэвиса-Болдуина (Davies-Bouldin Index):
   * Этот индекс оценивает компактность и разделимость кластеров. Чем меньше его значение, тем лучше качество кластеризации и тем более чётким считается разбиение.
   * Оптимальное количество кластеров — то, при котором Davies-Bouldin Index минимален.
5. Перекрёстная проверка на стабильность кластеров:
   * Известна как метод устойчивости (Cluster Stability). При этом методе данные случайным образом разделяются на подгруппы, выполняется кластеризация, и проверяются различия между кластерами. Устойчивые результаты указывают на подходящее количество кластеров.
6. Визуализация в сниженной размерности:
   * Использование методов снижения размерности (например, PCA или t-SNE) для визуального анализа кластеров. Это может дать интуитивное представление о числе кластеров, если данные имеют хорошо выраженные разбиения.

7. Что такое инерция в контексте метода локтя?

Инерция (inertia\_) в контексте метода локтя — это сумма квадратов расстояний от каждой точки данных до ближайшего центроида.

Inertia = 

Где:

* xi— точка данных,
* cj— координаты центроида ближайшего кластера.

Инерция измеряет "компактность" кластеров: чем меньше инерция, тем плотнее группы данных вокруг своих центроидов. Проблема в том, что инерция всегда уменьшается при увеличении числа кластеров, поэтому метод локтя используется для нахождения оптимального компромисса.

8. Как метод локтя связан с принципом уменьшения размерности?

Метод локтя прямого отношения к уменьшению размерности не имеет, но его часто дополняют методами снижения размерности (например, PCA, t-SNE или UMAP) для визуализации данных в двумерном или трёхмерном пространстве. Это помогает:

1. Наглядно показать структуру данных.
2. Оценить, являются ли кластеры действительно разделёнными, то есть присутствуют ли логичные разбиения в данных.
3. Упростить и ускорить расчёты для высокоразмерных данных.

С помощью снижения размерности можно также лучше интерпретировать результаты метода локтя, особенно если визуально становится очевидным определённое число кластеров.

9. Можно ли использовать метод локтя для оценки качества кластеризации, полученной с помощью алгоритмов, отличных от KMeans?

Да, метод локтя можно адаптировать для других алгоритмов кластеризации, если они минимизируют внутрикластерную дисперсию. Однако не все алгоритмы основаны на таком принципе:

* Для алгоритмов KMeans и его модификаций (MiniBatchKMeans и т.д.) метод локтя работает хорошо, так как они используют концепцию центроидов.
* Для других алгоритмов, таких как DBSCAN или иерархическая кластеризация: метод локтя не всегда подходит, поскольку такие алгоритмы не используют подход на основе центроидов или инерции. Чтобы оценить качество кластеризации в этих случаях, можно применять метод силуэтов или другие метрики, например, индекс Дэвиса-Болдуина.

10. Что делать, если на графике метода локтя нет четко выраженной точки "локтя"?

Если точка локтя плохо выражена (например, график линейно убывает или инерция снижается постепенно), можно использовать следующие подходы:

1. Использовать другие метрики или методы для определения числа кластеров:
   * Примените метод силуэтов или критерий Гапа, которые дают количественные оценки без необходимости визуальной интерпретации.
2. Рассмотрение нескольких значений кластеров:
   * Проверьте разные значения n\_clusters и оцените их на предмет интерпретируемости (например, с помощью визуализации или анализа бизнеса).
3. Комплект метрик (Silhouette + Inertia):
   * Рассмотрите другие метрики вместе с инерцией. Например, выберите количество кластеров, где значение коэффициента силуэта достаточно высокое.
4. Снижение размерности и визуализация:
   * Примените метод PCA или t-SNE к данным, чтобы визуально определить, сколько кластеров имеет смысл.
5. Зависимость от задачи:
   * В некоторых случаях подходящее количество кластеров определяется исходя из прикладной задачи (например, если есть заранее известная структура данных или бизнес-цель).

Тестовые вопросы по MiniBatchKMeans

1. MiniBatchKMeans — это модификация алгоритма KMeans, которая:

MiniBatchKMeans — это модификация алгоритма KMeans, предназначенная для ускорения процесса кластеризации, особенно на больших наборах данных. Вместо того чтобы использовать весь набор данных на каждом шаге пересчёта центра кластеров, MiniBatchKMeans работает с подвыборками данных (мини-батчами). Эти мини-батчи случайным образом выбираются из полного набора данных, чтобы минимизировать вычислительные затраты и использовать меньше памяти.

2. Какое основное отличие MiniBatchKMeans от KMeans?

Основное отличие заключается в том, какие данные используются на каждом шаге обновления центроидов кластеров:

* KMeans: На каждой итерации использует все данные для пересчёта центроидов.
* MiniBatchKMeans: На каждой итерации берёт случайные подвыборки (мини-батчи) из данных, что уменьшает объём вычислений.

Это делает MiniBatchKMeans более подходящим для работы с большими наборами данных, где использование всего датасета за один раз может быть очень затратным.

3. Как MiniBatchKMeans влияет на скорость работы алгоритма?

MiniBatchKMeans значительно увеличивает скорость работы алгоритма, особенно на больших наборах данных. Поскольку на каждой итерации обрабатывается только небольшой мини-батч данных, время вычислений на шагах обновления центроидов существенно сокращается. В результате алгоритм способен быстрее сходиться, не обрабатывая весь набор данных сразу. Эта модификация также снижает использование памяти.

4. Как MiniBatchKMeans влияет на точность кластеризации?

MiniBatchKMeans может несколько снизить точность кластеризации по сравнению с классическим KMeans, поскольку на каждом шаге обновления центроидов используется только часть данных. Это делает обновления менее стабильными и точными. Однако на больших наборах данных разница в точности обычно незначительна, так как сэмплирование из мини-батчей по-прежнему сохраняет общую структуру данных.

В целом, MiniBatchKMeans представляет собой компромисс между скоростью и точностью: он быстрее, но может быть немного менее точным.

5. Какой параметр в MiniBatchKMeans отвечает за размер мини-батча?

Размер мини-батча задаётся параметром batch\_size.

* Этот параметр определяет, сколько элементов используется для формирования подвыборки (мини-батча) на каждой итерации. По умолчанию batch\_size=100, но это значение можно изменить в зависимости от размера данных и доступных вычислительных ресурсов.
* Более крупные мини-батчи увеличивают вычислительные затраты, но могут улучшить точность, так как предоставляют более репрезентативную выборку данных на каждом шаге.

6. Какой параметр в MiniBatchKMeans отвечает за количество итераций по всему набору данных?

Параметр max\_iter отвечает за определение максимального количества итераций по всему набору данных.

* Этот параметр в MiniBatchKMeans устанавливает максимальное число обновлений центроидов, которые будут выполняться в процессе обучения.
* По умолчанию max\_iter=100, но это значение можно изменить в зависимости от задач, размера данных и необходимой точности.

7. В каких случаях MiniBatchKMeans может быть предпочтительнее KMeans?

MiniBatchKMeans предпочтительнее использовать в следующих случаях:

1. Обработка больших наборов данных. Если объём данных слишком велик, и полный анализ всех точек за раз (как в KMeans) становится неэффективным по времени или памяти.
2. Ограниченные вычислительные ресурсы. MiniBatchKMeans требует меньше памяти и меньше времени на вычисления за счёт использования мини-батчей.
3. Реальное время (streaming). Если данные поступают потоками или доступны не все сразу, MiniBatchKMeans может эффективно обрабатывать данные по частям.
4. Быстрая приблизительная кластеризация. Когда важнее скорость, чем точная кластеризация, например, для предварительных оценок или подключения алгоритмов, чувствительных к времени работы.

8. MiniBatchKMeans гарантированно находит глобальный оптимум?

Нет, MiniBatchKMeans не гарантирует нахождение глобального оптимума.

* Как и классический KMeans, MiniBatchKMeans может застрять в локальном минимуме, так как алгоритм является жадным и зависит от начальных условий (случайное начальное положение центроидов).
* Использование случайной подвыборки данных (мини-батчей) делает алгоритм ещё более "приблизительным" по сравнению с KMeans, а значит, найти глобальный оптимум становится сложнее.
* Чтобы улучшить результаты, можно воспользоваться несколькими запусками алгоритма с разными начальными позициями центроидов с помощью параметра n\_init.

9. Можно ли использовать метод локтя для определения оптимального количества кластеров в MiniBatchKMeans?

Да, метод локтя можно использовать в MiniBatchKMeans для определения оптимального количества кластеров.

* Метод локтя работает на основе оценки значения относительной инерции (атрибута inertia\_) для различных значений числа кластеров и выбора оптимального значения на "переломной" точке графика.
* Поскольку итоговая инерция в MiniBatchKMeans может быть менее стабильной из-за использования мини-батчей, точность метода локтя может немного снизиться. Однако для крупных наборов данных различие между результатами MiniBatchKMeans и KMeans обычно несущественно.

10. Какой атрибут обученного объекта MiniBatchKMeans содержит координаты центроидов кластеров?

Атрибут cluster\_centers\_ хранит координаты центроидов кластеров.

* Это двумерный массив numpy, где каждая строка представляет координаты одного из центроидов в пространстве признаков.
* Формат: (n\_clusters, n\_features), где n\_clusters — количество кластеров, а n\_features — количество признаков.

Тестовые вопросы по AgglomerativeClustering

1. AgglomerativeClustering - это алгоритм:

**AgglomerativeClustering** — это алгоритм иерархической кластеризации, который реализует метод "снизу вверх". Он начинает с каждого объекта данных как отдельного кластера и объединяет пары кластеров, основываясь на выбранной метрике сходства, пока не будет достигнуто заданное количество кластеров или другие критерии остановки.

Основные параметры:

**n\_clusters**  
Количество кластеров, к которым алгоритм должен прийти.  
По умолчанию n\_clusters=2.

**affinity**  
Мера расстояния между точками данных. Возможные значения:

* 'euclidean' (по умолчанию),
* 'l1' (манхэттенское расстояние),
* 'l2' (евклидово расстояние),
* 'cosine' (косинусное расстояние),
* 'precomputed' (предварительно вычисленные расстояния).

**linkage**  
Метод для определения расстояния между кластерами. Возможные значения:

* 'ward' (минимизация суммарной дисперсии, работает только с 'euclidean'),
* 'complete' (максимальное расстояние между точками кластеров),
* 'average' (среднее расстояние между точками кластеров),
* 'single' (минимальное расстояние между точками кластеров).

**distance\_threshold**  
Максимальное расстояние между кластерами, при котором они объединяются. Используется, если n\_clusters не задан.

**compute\_full\_tree**  
Если True, строится полное дерево кластеризации, даже если задано n\_clusters.

**Иерархическая кластеризация** помогает визуализировать данные в виде дендрограммы, что полезно для анализа структуры данных.\

Плюсы:

Не требует задания начальных центров кластеров.

Подходит для работы с небольшими наборами данных.

Минусы:

Высокая вычислительная сложность при большом объеме данных.

Чувствительность к выбросам.

2. Какой подход использует AgglomerativeClustering для построения кластеров?

**AgglomerativeClustering** использует **агломеративный подход** (метод "снизу вверх") для построения кластеров. Этот подход заключается в следующем:

 **Инициализация:**  
Каждый объект данных рассматривается как отдельный кластер.

 **Объединение кластеров:**  
На каждом шаге алгоритм объединяет два наиболее близких кластера, основываясь на выбранной метрике сходства и методе связи (linkage).

 **Повторение:**  
Процесс продолжается до тех пор, пока все объекты не объединятся в один кластер или не будет достигнуто заданное количество кластеров (n\_clusters), либо заданный порог расстояния (distance\_threshold).

 **Результат:**  
Итоговая структура кластеров может быть представлена в виде **дендрограммы**, что позволяет визуально проанализировать, на каком этапе объединения были сформированы кластеры.

Методы связи (linkage):

* **Ward:** минимизирует увеличение дисперсии внутри кластеров.
* **Complete:** использует максимальное расстояние между объектами разных кластеров.
* **Average:** считает среднее расстояние между объектами кластеров.
* **Single:** минимизирует минимальное расстояние между объектами кластеров.

3. Что такое дендрограмма в контексте AgglomerativeClustering?

Дендрограмма — это графическое представление иерархического процесса кластеризации. Она показывает, как данные объединяются в кластеры на каждом шаге агломеративного алгоритма. Вертикальные оси отображают уровни расстояния или несходства между кластерами, на которых происходят их объединения. Это позволяет выбрать оптимальное количество кластеров, "обрезая" дендрограмму на определенном уровне.

4. Какой параметр в AgglomerativeClustering отвечает за количество кластеров?

Параметр n\_clusters задаёт количество кластеров, которые алгоритм должен выделить. Значение по умолчанию — 2.

5. Какой параметр в AgglomerativeClustering отвечает за метод связывания (linkage)?

Параметр linkage определяет метод, используемый для расчёта расстояния между кластерами.

6. Какие методы связывания (linkage) доступны в AgglomerativeClustering ?

Доступны следующие методы:

* ward: минимизация увеличения дисперсии внутри кластеров.
* complete: максимальное расстояние между точками разных кластеров.
* average: среднее расстояние между точками разных кластеров.
* single: минимальное расстояние между точками разных кластеров.

7. Какой метод связывания (linkage) минимизирует дисперсию внутри кластеров?

Метод ward минимизирует увеличение дисперсии внутри кластеров.

8. Какой атрибут обученного объекта AgglomerativeClustering содержит метки кластеров для каждой точки данных?

Атрибут labels\_ содержит метки кластеров для каждой точки данных.

9. Можно ли использовать AgglomerativeClustering для кластеризации данных с большим количеством признаков?

Теоретически можно, но на практике это может быть затруднительно, так как алгоритм имеет высокую вычислительную сложность (обычно O(n3)O(n^3)O(n3)) и плохо масштабируется на большие объемы данных и множество признаков. В таких случаях может потребоваться предварительное уменьшение размерности данных.

10. AgglomerativeClustering гарантированно находит глобальный оптимум?

Нет, алгоритм AgglomerativeClustering не гарантирует нахождения глобального оптимума, так как он следует жадному подходу, объединяя на каждом шаге только наиболее близкие кластеры.

Тестовые вопросы по DBSCAN

1. DBSCAN - это алгоритм:

**DBSCAN (Density-Based Spatial Clustering of Applications with Noise)** — это алгоритм кластеризации, который группирует точки на основе плотности их расположения. Его основное преимущество заключается в том, что он автоматически определяет количество кластеров и хорошо справляется с выявлением кластеров сложной формы, а также устойчив к шумовым данным и выбросам.

* 1. Точки в кластере должны быть достаточно "плотно" сгруппированы, то есть каждая точка кластера должна находиться в пределах определённого радиуса от других точек.
  2. Точки, которые находятся в малонаселённых областях, считаются шумовыми.

Основные параметры DBSCAN:

eps (ε, радиус окрестности):

1. Определяет максимальное расстояние между двумя точками, чтобы они считались соседями.
2. Чем больше значение eps, тем больше вероятность, что точки будут объединены в один кластер.

min\_samples (минимальное количество точек):

1. Минимальное число точек в пределах радиуса eps, необходимое для того, чтобы точка считалась "основной" (core point).
2. Если число соседей меньше min\_samples, то точка считается либо граничной (border point), либо шумовой (noise point).

Типы точек в DBSCAN:

1. Основная точка (core point):

Точка, вокруг которой находится не менее min\_samples других точек в пределах радиуса eps.

1. Граничная точка (border point):

Точка, которая лежит в пределах радиуса eps от основной точки, но сама не имеет достаточно соседей, чтобы быть основной.

1. Шумовая точка (noise point):

Точка, которая не является ни основной, ни граничной и не входит в какой-либо кластер.

**Этапы работы алгоритма:**

 Выбирается начальная точка из набора данных.

 Проверяется количество соседей этой точки в пределах радиуса eps.

* Если число соседей >= min\_samples, то точка становится **основной**, и начинается формирование нового кластера.
* Если число соседей < min\_samples, то точка помечается как **шумовая**.

 Для каждой основной точки алгоритм рекурсивно добавляет соседние точки в кластер, пока не будут покрыты все плотные области данных.

 Повторяется, пока не будут обработаны все точки.

Преимущества DBSCAN:

**Выявление кластеров произвольной формы:** В отличие от KMeans, DBSCAN не ограничивается кластеризацией на основе сферических границ.

**Автоматическое определение шума:** Алгоритм автоматически исключает шумовые точки.

Не требует заранее задавать количество кластеров.

Недостатки DBSCAN:

**Чувствительность к параметрам eps и min\_samples:** Неправильный выбор параметров может привести к плохой кластеризации.

**Проблемы с высокоразмерными данными:** В таких данных сложно подобрать оптимальное значение eps.

**Плохая производительность на больших наборах данных:** Алгоритм требует вычисления расстояний между всеми парами точек.

Атрибуты обученного объекта DBSCAN:

**labels\_** — метки кластеров для каждой точки. Шумовые точки получают метку -1.

**core\_sample\_indices\_** — индексы основных точек.

**components\_** — координаты основных точек (ядра кластеров).

2. Какие два основных параметра используются в DBSCAN?

eps (эпсилон):

Радиус окрестности точки, в пределах которого алгоритм определяет плотность. Точки, находящиеся внутри этого радиуса, рассматриваются как потенциальные соседи. Чем меньше eps, тем более плотные кластеры будут обнаружены.

min\_samples (минимальное количество образцов):

Минимальное число точек, включая саму точку, которые должны находиться в пределах радиуса eps, чтобы точка считалась "основной точкой" (core point). Этот параметр определяет минимальную плотность, необходимую для формирования кластера.

3. Что такое "основная точка" (core point) в DBSCAN?

Основная точка — это точка, вокруг которой находится как минимум min\_samples точек в радиусе eps.

4. Что такое "граничная точка" (border point) в DBSCAN?

Граничная точка — это точка, которая находится в окрестности радиуса eps от основной точки, но вокруг неё самого недостаточно точек, чтобы она стала основной.

5. Что такое "шумовая точка" (noise point) в DBSCAN?

Шумовая точка — это точка, которая не является ни основной, ни граничной. Она считается выбросом и не входит ни в один кластер.

6. Какой параметр в DBSCAN отвечает за радиус окрестности?

Параметр eps.

7. Какой параметр в DBSCAN отвечает за минимальное количество точек в окрестности?

Параметр min\_samples.

8. Какие преимущества имеет DBSCAN по сравнению с KMeans?

 Не требует заранее задавать количество кластеров.

 Может обнаруживать кластеры произвольной формы.

 Устойчив к выбросам и шуму.

9. Какие недостатки имеет DBSCAN?

 Чувствителен к выбору параметров eps и min\_samples.

 Сложно работает с кластерами переменной плотности.

 Высокая вычислительная сложность на больших наборах данных.

10. Какой атрибут обученного объекта DBSCAN содержит метки кластеров для каждой точки данных?

Атрибут labels\_.

Тестовые вопросы по OPTICS

1. OPTICS - это алгоритм:

OPTICS (Ordering Points To Identify the Clustering Structure) — алгоритм кластеризации на основе плотности, предназначенный для выявления кластеров произвольной формы и шумовых данных. Он похож на DBSCAN, но предоставляет более гибкий подход к кластеризации, так как не требует фиксированного значения радиуса eps для всех кластеров.

Основная идея алгоритма заключается в построении упорядоченного списка точек данных на основе их плотности. Этот список используется для определения структуры кластеров и выделения областей с различной плотностью.

Как работает OPTICS:

Инициализация.

Алгоритм начинает с произвольной точки и вычисляет её плотность соседей (количество точек в радиусе eps).

Определение основных и достижимых расстояний.

1. **Основное расстояние (core distance):** минимальное расстояние, при котором точка становится "основной" (имеет не менее min\_samples соседей).
2. **Достижимость (reachability distance):** расстояние, необходимое для достижения другой точки через текущую.

Упорядочение точек.

Алгоритм упорядочивает точки на основе их достижимости, создавая упорядоченный список, который используется для визуализации и анализа.

Кластеризация.

Кластеры выделяются из упорядоченного списка, анализируя изменения в значениях достижимости.

Основные параметры OPTICS:

min\_samples:

1. Минимальное количество точек в радиусе eps, чтобы точка считалась "основной" (core point).
2. Этот параметр определяет минимальную плотность кластера.

max\_eps:

1. Максимальное расстояние, в пределах которого алгоритм рассматривает точки как соседей.
2. Если расстояние между точками превышает max\_eps, они не считаются связанными.

Metric:

1. Мера расстояния, используемая для определения близости между точками (например, Евклидово расстояние, Манхэттенское расстояние и т.д.).
2. По умолчанию используется Евклидово расстояние.

cluster\_method:

1. Метод выделения кластеров из упорядоченного списка.
2. xi — выделение кластеров на основе изменения плотности.
3. dbscan — выделение кластеров, как в DBSCAN.

eps (для dbscan метода):

Радиус окрестности, используемый для выделения кластеров, если выбран метод dbscan.

Xi:

Параметр для определения плотности в методе xi. Используется для выделения кластеров с различной плотностью.

leaf\_size:

Размер листа для оптимизации вычислений в KD-дереве или BallTree, которые используются для ускорения поиска соседей.

Основные преимущества OPTICS:

1. Способен выявлять кластеры с различной плотностью без необходимости фиксированного значения eps.
2. Идентифицирует шумовые точки.
3. Работает с данными произвольной формы.
4. Подходит для задач с большими объёмами данных и многомерными признаками.

Пример визуализации результата OPTICS:

Основным инструментом анализа результатов является **диаграмма достижимости (reachability plot)**. Эта диаграмма показывает, как изменяется достижимость точек, что позволяет визуально выделить кластеры (как "долины" на графике) и шумовые точки (как пики).

Недостатки OPTICS:

1. Алгоритм требует больше вычислительных ресурсов, чем DBSCAN.
2. Результаты чувствительны к выбору параметров, таких как min\_samples и xi.
3. Визуализация и интерпретация результатов может быть сложной при большом количестве данных.

2. Какое основное преимущество OPTICS по сравнению с DBSCAN?

Основное преимущество OPTICS заключается в том, что он устраняет необходимость точного задания параметра eps. OPTICS строит упорядоченный список точек с указанием их достижимости (reachability distance), что позволяет выделять кластеры с различной плотностью. Это делает алгоритм более универсальным и подходящим для задач, где плотность кластеров может варьироваться.

3. Что такое "достижимость" (reachability distance) в OPTICS?

Достижимость — это расстояние, необходимое для достижения точки из другой точки через плотные области. Формально, это максимальное значение между основным расстоянием (core distance) и евклидовым расстоянием до соседней точки. Достижимость указывает, насколько "плотно" связаны точки.

4. Что такое "основное расстояние" (core distance) в OPTICS?

Основное расстояние — это минимальное расстояние eps, при котором точка становится основной (core point), то есть имеет не менее min\_samples соседей в пределах этого радиуса. Если точка не является основной, основное расстояние не определено.

5. Какой параметр в OPTICS отвечает за минимальное количество точек в окрестности?

Параметр **min\_samples** задаёт минимальное количество точек в радиусе eps, чтобы точка считалась основной (core point).

6. Какой параметр в OPTICS отвечает за максимальное расстояние для рассмотрения соседей?

Параметр **max\_eps** определяет максимальное расстояние, в пределах которого рассматриваются соседи для вычисления плотности.

7. Что такое "упорядоченный список" (ordering) в OPTICS?

Упорядоченный список — это последовательность точек данных, отсортированных по достижимости. Этот список отражает структуру кластеров и их плотность, что позволяет визуально оценить распределение точек и выделить кластеры.

8. Как можно визуализировать результаты OPTICS?

Результаты OPTICS визуализируют с помощью **диаграммы достижимости (reachability plot)**, где по оси X откладывается индекс точки из упорядоченного списка, а по оси Y — её достижимость. Кластеры отображаются как "долины" на графике, а шумовые точки — как пики.

9. Как можно извлечь кластеры из упорядоченного списка OPTICS?

Кластеры можно извлечь путём анализа диаграммы достижимости или с использованием алгоритма **DBSCAN** на основе значений достижимости. Некоторые реализации OPTICS, такие как в библиотеке Scikit-learn, позволяют автоматически выделять кластеры на основе пороговых значений.

10. OPTICS чувствителен к выбору параметров?

OPTICS менее чувствителен к параметрам, чем DBSCAN, благодаря тому, что он не требует строгого задания фиксированного eps. Однако параметры **min\_samples** и **max\_eps** всё же могут значительно влиять на результаты, особенно в случаях сложной структуры данных.

Тестовые вопросы по GaussianNB

**1. GaussianNB - это алгоритм:**

GaussianNB (Gaussian Naive Bayes) — это алгоритм классификации, основанный на теореме Байеса и предполагающий, что признаки (фичи) независимы друг от друга. Он используется в задачах машинного обучения и статистики для классификации данных.

**2. На каком принципе основан GaussianNB?**

GaussianNB (Gaussian Naive Bayes) основан на принципе теоремы Байеса, которая описывает, как обновлять вероятность гипотезы (в данном случае, принадлежности объекта к классу) на основе новых данных.

**3. Какое предположение делает GaussianNB о распределении признаков?**

GaussianNB делает предположение о том, что распределение признаков (фич) в каждом классе нормально (гауссово). Это означает, что для каждого признака Xi в классе C предполагается, что его значения следуют нормальному распределению, которое определяется двумя параметрами:

Среднее (μ): Это среднее значение признака в пределах данного класса.

Стандартное отклонение (σ): Это мера разброса значений признака вокруг среднего в пределах данного класса.

**4. Какой параметр в GaussianNB отвечает за сглаживание (smoothing)?**

***Из документации на метод***

var\_smoothing : float, default=1e-9

Portion of the largest variance of all features that is added to variances for calculation stability.

***Нейросетка***

В GaussianNB параметр, отвечающий за сглаживание, называется "параметр Лапласа" (Laplace smoothing) или "параметр сглаживания". Однако в контексте GaussianNB, который основан на нормальном распределении, сглаживание не применяется так, как это делается в других вариантах наивного байесовского классификатора, таких как Multinomial Naive Bayes.

В GaussianNB сглаживание может быть связано с добавлением небольшого значения к стандартному отклонению, чтобы избежать деления на ноль или проблем с вычислениями, когда стандартное отклонение слишком мало. Однако в стандартной реализации GaussianNB в библиотеке scikit-learn нет явного параметра для сглаживания, как это есть в других алгоритмах.

Если вы имеете в виду сглаживание в контексте других наивных байесовских классификаторов, таких как MultinomialNB, то там используется параметр alpha, который контролирует степень сглаживания. В GaussianNB же основное внимание уделяется оценке параметров нормального распределения (среднего и стандартного отклонения) для каждого признака в каждом классе.

**5. Для чего используется сглаживание в GaussianNB?**

Основные причины, по которым может потребоваться сглаживание, включают:

**Избежание деления на ноль:** Если стандартное отклонение для какого-либо признака в классе оказывается равным нулю (что может произойти, если все значения этого признака в классе одинаковы), это может привести к проблемам при вычислении вероятностей. Сглаживание помогает избежать этой ситуации, добавляя небольшое значение к стандартному отклонению.

**Устойчивость к выбросам:** Сглаживание может помочь сделать модель более устойчивой к выбросам в данных. Если в данных есть аномальные значения, они могут сильно повлиять на оценку среднего и стандартного отклонения. Сглаживание может помочь уменьшить влияние таких выбросов.

**Общая стабильность модели:** Сглаживание может улучшить общую стабильность и производительность модели, особенно в случаях, когда данные имеют небольшие объемы или когда некоторые классы имеют очень мало примеров. Это может помочь избежать переобучения и улучшить обобщающую способность модели.

**6. Какой метод в GaussianNB используется для обучения модели?**

В GaussianNB для обучения модели используется метод максимального правдоподобия (Maximum Likelihood Estimation, MLE). Этот метод позволяет оценить параметры нормального распределения (среднее и стандартное отклонение) для каждого признака в каждом классе на основе обучающего набора данных

**7. Какой метод в GaussianNB используется для предсказания класса для новых данных?**

В GaussianNB для предсказания класса для новых данных используется метод, основанный на теореме Байеса, в частности, вычисление апостериорной вероятности для каждого класса. Процесс предсказания включает следующие шаги:

Вычисление правдоподобия

Вычисление апостериорной вероятности

Выбор класса

**8. Какой атрибут обученного объекта GaussianNB содержит средние значения признаков для каждого класса?**

В обученном объекте GaussianNB атрибут, который содержит средние значения признаков для каждого класса, называется class\_prior\_. Однако, для получения средних значений признаков для каждого класса, вам нужно использовать атрибут theta\_.

theta\_: Это массив, который содержит средние значения признаков для каждого класса. Он имеет размерность (n\_classes,n\_features), где n\_classes — количество классов, а n\_features — количество признаков.

**9. Какой атрибут обученного объекта GaussianNB содержит дисперсии признаков для каждого класса?**

В обученном объекте GaussianNB атрибут, который содержит дисперсии признаков для каждого класса, называется sigma\_.

sigma\_: Это массив, который содержит дисперсии (или стандартные отклонения в квадрате) признаков для каждого класса. Он имеет размерность (n\_classes,n\_features), где n\_classes — количество классов, а n\_features — количество признаков.

**10. GaussianNB подходит для работы с категориальными признаками?**

GaussianNB не является оптимальным выбором для работы с категориальными признаками, поскольку он предполагает, что признаки распределены нормально (гауссово). Это предположение подходит для непрерывных признаков, но не для категориальных.

Тестовые вопросы по MultinomialNB

**1. MultinomialNB - это алгоритм:**

MultinomialNB (Multinomial Naive Bayes) — это алгоритм машинного обучения, который используется для классификации текстов и других задач, где данные представлены в виде категориальных признаков. Он основан на теореме Байеса и предполагает, что признаки независимы друг от друга.

**2. На каком принципе основан MultinomialNB?**

MultinomialNB (Multinomial Naive Bayes) основан на теореме Байеса и принципе условной вероятности.

Теорема Байеса: Основная идея заключается в том, что для классификации объекта (например, текста) мы можем использовать условные вероятности.

Предположение о независимости признаков: MultinomialNB предполагает, что все признаки (например, слова в тексте) независимы друг от друга, что упрощает вычисление правдоподобия. Это означает, что вероятность наблюдения признаков при условии класса можно выразить как произведение вероятностей каждого признака

Модель на основе частот: В MultinomialNB используется информация о частотах признаков для оценки вероятностей.

**3. Какое предположение делает MultinomialNB о распределении признаков?**

MultinomialNB делает предположение о том, что распределение признаков следует мультиномиальному распределению. Это означает, что алгоритм предполагает, что данные представляют собой набор категориальных признаков, которые могут принимать целочисленные значения, отражающие количество вхождений каждого признака (например, слов) в документе.

**4. Какой параметр в MultinomialNB отвечает за сглаживание (smoothing)?**

***Из документации***

*В MultinomialNB параметр, отвечающий за сглаживание, называется alpha.*

Сглаживание используется для предотвращения проблемы нулевых вероятностей, которая может возникнуть, если какой-либо признак (например, слово) не встречается в обучающем наборе данных для определенного класса.

**5. Для чего используется сглаживание в MultinomialNB?**

Влияние на модель: Сглаживание помогает улучшить обобщающую способность модели, особенно в случаях, когда обучающий набор данных мал или когда некоторые признаки могут отсутствовать в определенных классах.

**6. Какой метод в MultinomialNB используется для обучения модели?**

В MultinomialNB для обучения модели используется метод максимального правдоподобия. Этот метод позволяет оценить параметры модели, основываясь на обучающем наборе данных.

**7. Какой метод в MultinomialNB используется для предсказания класса для новых данных?**

В MultinomialNB для предсказания класса для новых данных используется метод, основанный на теореме Байеса, который позволяет вычислить апостериорные вероятности классов для данного набора признаков.

**8. Какой атрибут обученного объекта MultinomialNB содержит вероятности признаков для каждого класса?**

В обученном объекте MultinomialNB атрибут, который содержит вероятности признаков для каждого класса, называется feature\_log\_prob\_.

Содержимое: Этот атрибут представляет собой массив, где каждая строка соответствует классу, а каждый столбец соответствует признаку. Значения в этом массиве — это логарифмы условных вероятностей признаков при условии класса. Использование логарифмов помогает избежать проблем с числовой стабильностью при работе с очень малыми вероятностями.

**9. MultinomialNB подходит для работы с отрицательными значениями признаков?**

Нет, MultinomialNB не подходит для работы с отрицательными значениями признаков. Этот алгоритм предназначен для работы с дискретными и неотрицательными признаками, такими как частоты или счетчики.

**10. В каких задачах MultinomialNB часто показывает хорошие результаты?**

В целом, MultinomialNB особенно эффективен в задачах, где данные могут быть представлены в виде частот или счетчиков, и где важна независимость признаков.

1. Классификация текстов

2.Обработка естественного языка (NLP)

3. Классификация категорий

4. Медицинская диагностика

5. Рекомендательные системы

Тестовые вопросы по ComplementNB

**1. ComplementNB - это алгоритм:**

ComplementNB (Complement Naive Bayes) — это алгоритм машинного обучения, который является модификацией стандартного наивного байеса, в частности, MultinomialNB. Он был разработан для улучшения производительности на задачах многоклассовой классификации, особенно когда классы несбалансированы.

**2. На каком принципе основан ComplementNB?**

ComplementNB (Complement Naive Bayes) основан на принципах теоремы Байеса и модифицирован для улучшения производительности в задачах многоклассовой классификации, особенно когда классы несбалансированы.

**3. В чем основное отличие ComplementNB от MultinomialNB?**

В отличие от MultinomialNB, ComplementNB вычисляет вероятности для каждого класса, используя информацию о всех остальных классах. Он оценивает вероятности на основе дополнения к классу, что позволяет учитывать влияние других классов на предсказания. Это делается для того, чтобы улучшить производительность в условиях несбалансированных классов.

**4. Для каких типов задач ComplementNB может быть предпочтительнее MultinomialNB?**

В общем, ComplementNB предпочтителен в тех случаях, когда классы имеют разное количество примеров, и когда важно учитывать влияние других классов на предсказания. Это делает его особенно полезным в задачах многоклассовой классификации и текстовой классификации.

1. Несбалансированные классы
2. Многоклассовая классификация
3. Текстовая классификация
4. Анализ тональности
5. Классификация документов
6. Обработка естественного языка (NLP)

**5. Какой параметр в ComplementNB отвечает за сглаживание (smoothing)?**

***Из документации***

*В ComplementNB параметр, отвечающий за сглаживание, называется alpha.*

Этот параметр используется для предотвращения проблемы нулевых вероятностей, которая может возникнуть, если какой-либо признак (например, слово) не встречается в обучающем наборе данных для определенного класса.

**6. Какой метод в ComplementNB используется для обучения модели?**

В ComplementNB для обучения модели используется метод максимального правдоподобия, аналогично другим алгоритмам наивного байеса. Однако, в отличие от MultinomialNB, ComplementNB применяет модифицированный подход, который учитывает информацию о всех классах, а не только о целевом классе.

**7. Какой метод в ComplementNB используется для предсказания класса для новых данных?**

В ComplementNB для предсказания класса для новых данных используется метод, основанный на теореме Байеса, который позволяет вычислить апостериорные вероятности классов для данного набора признаков.

**8. Какой атрибут обученного объекта ComplementNB содержит веса признаков для каждого класса?**

В обученном объекте ComplementNB атрибут, который содержит веса признаков для каждого класса, называется coef\_.

Содержимое: Этот атрибут представляет собой массив, где каждая строка соответствует классу, а каждый столбец соответствует признаку. Значения в этом массиве представляют собой логарифмы условных вероятностей признаков для каждого класса, что позволяет оценить влияние каждого признака на предсказания.

**9. ComplementNB подходит для работы с отрицательными значениями признаков?**

Нет, ComplementNB не подходит для работы с отрицательными значениями признаков. Этот алгоритм, как и другие модели наивного байеса, предназначен для работы с неотрицательными признаками, такими как частоты или счетчики.

**10. ComplementNB может быть использован для решения задач регрессии?**

Нет, ComplementNB (Complement Naive Bayes) не предназначен для решения задач регрессии. Этот алгоритм является классификатором, который используется для задач классификации, где цель состоит в том, чтобы предсказать категориальный класс на основе входных признаков.

Тестовые вопросы по BernoulliNB

1. BernoulliNB - это алгоритм:

Наивного байесовского классификатора, основанный на модели Бернулли.

2. На каком принципе основан BernoulliNB?

На применении теоремы Байеса и предположении о независимости признаков.

3. Какое предположение делает BernoulliNB о распределении признаков?

Признаки бинарны и следуют распределению Бернулли (принимают значения 0 или 1).

4. Какой параметр в BernoulliNB отвечает за сглаживание (smoothing)?

alpha

5. Для чего используется сглаживание в BernoulliNB?

Для предотвращения проблемы нулевой вероятности, которая возникает, если какой-либо признак отсутствует в данных для определённого класса.

6. Какой метод в BernoulliNB используется для обучения модели?

fit(X, y)

7. Какой метод в BernoulliNB используется для предсказания класса для новых данных?

predict(X)

8. Какой атрибут обученного объекта BernoulliNB содержит вероятности появления признаков в каждом классе?

feature\_log\_prob\_

9. BernoulliNB подходит для работы с непрерывными признаками?  
Нет, он предназначен для бинарных данных. Непрерывные признаки нужно преобразовать, например, через пороговое разбиение

10. В каких задачах BernoulliNB часто показывает хорошие результаты?

В задачах классификации текстов, например, для фильтрации спама или анализа тональности, где данные представлены в виде бинарных вхождений (наличие/отсутствие слов)

Тестовые вопросы по GaussianNB, MultinomialNB, ComplementNB, BernoulliNB

1. Какой из алгоритмов наивного Байеса подходит для работы с непрерывными признаками, предполагая их нормальное распределение?

GaussianNB

2. Вы работаете над задачей классификации текстов, используя представление "мешок слов" (bag-of-words). Какой алгоритм наивного Байеса будет наиболее подходящим?

MultinomialNB

3. Ваш набор данных для классификации сильно несбалансирован, один класс значительно преобладает над другими. Какой алгоритм наивного Байеса может быть наиболее эффективен в этой ситуации?

ComplementNB

4. Ваши признаки представляют собой бинарные значения (0/1), указывающие на наличие или отсутствие определенного атрибута. Какой алгоритм наивного Байеса следует использовать?

BernoulliNB

5. Какой параметр используется во всех перечисленных алгоритмах наивного Байеса для сглаживания вероятностей и предотвращения проблем с нулевыми вероятностями?

alpha

6. Какой из алгоритмов наивного Байеса использует дополнение (complement) вероятностей при расчете вероятности класса, что делает его более устойчивым к шуму в данных?

ComplementNB

7. Вы работаете над задачей классификации изображений, где признаки представляют собой значения пикселей. Какой алгоритм наивного Байеса может быть подходящим выбором?

GaussianNB

8. Вы разрабатываете систему для фильтрации спама, где признаки представляют собой наличие или отсутствие определенных слов в электронном письме. Какой алгоритм наивного Байеса может быть наиболее эффективен?

BernoulliNB

9. Вам нужно предсказать вероятность заболевания, основываясь на наличии или отсутствии определенных симптомов у пациента. Какой алгоритм наивного Байеса можно использовать?

BernoulliNB

10. Все перечисленные алгоритмы наивного Байеса (GaussianNB, MultinomialNB, ComplementNB, BernoulliNB) являются:

Классификаторами, основанными на теореме Байеса с предположением о независимости признаков (наивные байесовские классификаторы).

Тестовые вопросы по LinearDiscriminantAnalysis

1. LinearDiscriminantAnalysis (LDA) - это метод:

Классификации и снижения размерности.

2. Какова основная цель LDA в контексте классификации?

Найти линейные комбинации признаков, которые лучше всего разделяют классы.

3. Какое предположение делает LDA о распределении данных?

Данные каждого класса следуют многомерному нормальному распределению с одинаковой ковариационной матрицей.

4. Что такое дискриминантные функции в LDA?

Линейные комбинации признаков, которые максимизируют различие между классами.

5. Какой параметр в LinearDiscriminantAnalysis отвечает за количество дискриминантных функций (компонент)?

n\_components

6. Какой метод в LinearDiscriminantAnalysis используется для обучения модели?

fit(X, y)

7. Какой метод в LinearDiscriminantAnalysis используется для предсказания класса для новых данных?

predict(X)

8. Какой атрибут обученного объекта LinearDiscriminantAnalysis содержит коэффициенты дискриминантных функций?

coef\_

9. LDA может быть использован для снижения размерности данных?

Да, путем проекции данных на пространство с меньшим числом измерений, основанное на дискриминантных функциях

10. В каких случаях LDA может быть предпочтительнее PCA для снижения размерности?

Когда классовая информация важна, и требуется сохранить разделение между классами. LDA учитывает метки классов, в отличие от PCA, который использует только вариацию данных

Тестовые вопросы по "Метод опорных векторов"

 **Метод опорных векторов (SVM)** - это алгоритм:

* Машинного обучения, используемый для задач классификации и регрессии.

 **Основная идея метода опорных векторов**:

* Найти гиперплоскость, которая максимально разделяет классы данных с максимальным зазором (margin).

 **Опорные векторы**:

* Это точки из набора данных, которые лежат на границах классов и определяют положение гиперплоскости.

 **Зазор (margin) в методе опорных векторов**:

* Это расстояние между гиперплоскостью и ближайшими к ней точками данных двух классов.

 **Ядро (kernel) в методе опорных векторов**:

* Это функция, позволяющая преобразовать данные в пространство более высокой размерности для лучшей линейной разделимости.

 **Типы ядер, обычно используемые в методе опорных векторов**:

* Линейное, полиномиальное, RBF (радиальная базисная функция), сигмоидное.

 **Параметр C в методе опорных векторов**:

* Контролирует баланс между максимизацией зазора и минимизацией ошибки классификации. Малое значение делает модель более гибкой, а большое - жесткой.

 **Метод в sklearn.svm.SVC для обучения модели**:

* fit()

 **Метод в sklearn.svm.SVC для предсказания класса для новых данных**:

* predict()

 **Метод опорных векторов чувствителен к масштабированию признаков?**

* Да, потому что значения признаков влияют на положение гиперплоскости.

Тестовые вопросы по sklearn SVC

1. **Класс в Scikit-Learn для метода опорных векторов для классификации**:
   * sklearn.svm.SVC
2. **Параметр в SVC для выбора типа ядра**:
   * kernel
3. **Значения параметра kernel в SVC**:
   * linear, poly, rbf, sigmoid, а также пользовательская функция.
4. **Параметр в SVC для управления степенью регуляризации**:
   * C
5. **Влияние увеличения значения параметра C на сложность модели**:
   * Увеличивает сложность, уменьшая зазор.
6. **Параметр в SVC для задания степени полинома**:
   * degree (используется для полиномиального ядра).
7. **Параметр в SVC для задания ширины гауссова ядра (RBF)**:
   * gamma
8. **Метод в SVC для обучения модели**:
   * fit()
9. **Метод в SVC для предсказания класса для новых данных**:
   * predict()
10. **Атрибут обученного объекта SVC, содержащий опорные векторы**:
    * support\_vectors\_

Тестовые вопросы по SVC, NuSVC и LinearSVC

 **Класс, предназначенный только для линейных ядер**:

* LinearSVC

 **Параметр для управления степенью регуляризации в SVC и NuSVC, но отсутствующий в LinearSVC**:

* C

 **Параметр в NuSVC, контролирующий долю ошибок и опорных векторов**:

* nu

 **Класс, который обучается быстрее на больших наборах данных с линейно разделимыми классами**:

* LinearSVC

 **Класс, поддерживающий больше типов ядер, включая нелинейные**:

* SVC

 **Параметр в LinearSVC для выбора типа регуляризации (L1 или L2)**:

* penalty

 **Класс, использующий liblinear для оптимизации**:

* LinearSVC

 **Класс, использующий libsvm для оптимизации**:

* SVC и NuSVC

 **Класс, не поддерживающий вероятностные оценки классов (predict\_proba)**:

* LinearSVC

 **Параметр в SVC и NuSVC для использования пользовательского ядра**:

* kernel с указанием пользовательской функции.