Методы машинного обучения (только 100% ответы :) )

[**1) Основные этапы построения моделей в задачах регрессии и классификации. (готов) 3**](#_eirttjk6rzg0)

[**2) Проверка гипотез. t-критерий с одной выборкой.(готов) 5**](#_78ijbshlot4q)

[**3) Выбор модели с использованием кросс-валидации.(готов) 6**](#_6cx9vnxlum51)

[**4) Оценка параметров. Метод максимального правдоподобия.(готов) 8**](#_oiusmnefaw2i)

[**5) Задача оптимизации. Градиентный спуск.(готов) 10**](#_d1yhj3wnrgfe)

[**6) Регрессия. Метод наименьших квадратов и градиентный спуск.(готов) 11**](#_imtdg2o75dt)

[**7) Регрессия. Градиентный спуск и стохастический градиентный спуск.(Готов) 13**](#_t624fhuuki62)

[**8) Регрессия и классификация. Метод ближайших соседей.(Готов) 14**](#_snwqoju99m91)

[**9) Регрессия. Деревья решений. Критерии деления.(copypaste) 17**](#_zenxz5n3k3zu)

[**10) Регрессия и классификация. Ансамблевые моделей. Бэггинг, бустинг и стекинг(Готов) 18**](#_xwym0b7w8tjd)

[**11) Регрессия и классификация. Деревья решений. Бэггинг, случайный лес и сверхслучайные деревья.(copypaste) 21**](#_i21z03iht9fa)

[**12) Регрессия. L1, L2 регуляризация.(Готов) 22**](#_bxeaky2llqxc)

[**13) Классификация. Логистическая регрессия (бинарная классификация).(готов) 24**](#_uycbtdcg7qtz)

[**14) Классификация. Полиномиальная логистическая регрессия (многоклассовая классификация).(готов) 26**](#_hdr4ig2j56z9)

[**15) Классификация. Методы определения качества классификаторов.(copypaste) 27**](#_tksk2kx8zdg7)

[**16) Классификация. Наивный байесовский классификатор с нормальным законом распределения.(Готов) 28**](#_lv0v9jg56rff)

[**17) Классификация. Метод опорных векторов.(copypaste) 31**](#_5r4wq9kwzzvo)

[**18) Классификация. Деревья решений. Критерии деления.(copypaste) 32**](#_18joxjp9a8yn)

[**19) Классификация. L1, L2 регуляризация. (Готов) 33**](#_xl76z0wynznr)

[**20) Классификация текстовых документов с использованием наивного байесовского классификатора. Модель Бернулли.(copypaste) 37**](#_dwjhh79pryvp)

[**21) Классификация текстовых документов с использованием наивного байесовского классификатора. Мультиномиальная модель.(copypaste) 38**](#_rh8tr8xdo256)

[**22) Классификация. Многослойная нейронная сеть прямого распространения.(готов) 39**](#_vxguq8wp2esw)

[**23) Кластеризация. Метод k-средних. Выбор начальных значений центров кластеров.(copypaste) 43**](#_cy3hwhub7hqy)

[**24) Кластеризация. Иерархическая кластеризация. Агломеративные методы.(copypaste) 45**](#_vjsqhws899e9)

[**25) Уменьшение размерности. Метод главных компонент PCA. (copypaste) 47**](#_ovz169xz3z24)

[**26) Рекомендательные системы. Коллаборативная фильтрация на основе сходства пользователей. (copypaste) 49**](#_yrbit9b3491r)

[**27) Рекомендательные системы. Коллаборативная фильтрация на основе сходства элементов.(copypaste) 51**](#_lzm1rfqwacj5)

[**28) Рекомендательные системы. Факторизация матрицы рейтингов. ALS.(copypaste) 52**](#_5m87cney62bt)

[**29) Распределенные алгоритмы. Расчет среднего значения и стандартного отклонения.(copypaste) 54**](#_dcbjetner7no)

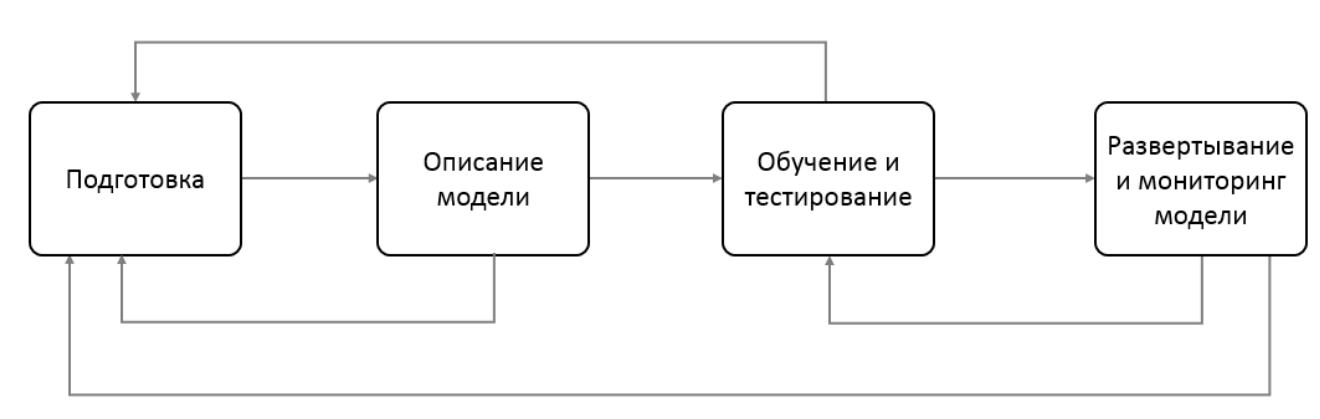
[**30) Распределенные алгоритмы. Расчет косинусной меры сходства.(copypaste) 55**](#_2ffk49345wvv)

[**31) Распределенные алгоритмы. Градиентный спуск.(copypaste) 56**](#_i6cvd0rrf96f)

[**32) Распределенные алгоритмы. Стохастический градиентный спуск.(copypaste) 57**](#_wupmg7geiiey)

[**33) Распределенные алгоритмы. Факторизация матрицы (один из вариантов).(copypaste) 58**](#_m8vwxm8azlsb)

# 1) Основные этапы построения моделей в задачах регрессии и классификации. (готов)



**1 Подготовка**

определение метрик,

входных и выходных данных,

Сбор и разметка данных

выбор оценок качества модели,

выбор базовой отметки (baseline)

(Для справки: Базовая модель(baseline) - это модель, которая просто предсказывает принадлежность каждого наблюдения в наборе данных к наиболее распространенному классу.

любая модель классификации, которая имеет более высокую точность, чем базовая модель, может считаться “полезной”, но очевидно, что чем больше разница в точности между нашей моделью и базовой моделью, тем лучше.)

**2 Описание модели**

Выбор моделей,

методов регуляризации,

функции потерь (loss function)

**3 Обучение и тестирование модели**

• Запуск обучения выбранных моделей

• Выбор значений гиперпараметров (количество итераций, коэффициента регуляризации)

• Оценка качества моделей на отложенной выборке

• Определение наилучшей модели

**4 Развертывание и мониторинг модели**

• Использование лучшей модели для предсказания

• Мониторинг производительности модели

• Сравнение выбранной модели с моделями из предыдущего этапа

• Периодическая калибровка модели

**Пример для классификации текста**

1. **Сбор данных:** В случае классификации текста это может быть корпус текстов с метками классов (например, положительные и отрицательные отзывы, категории новостей и т.д.).
2. **Предобработка данных:** Тексты требуют значительной предобработки, чтобы быть пригодными для машинного обучения.

- Очистка текста: Удаление HTML-тегов, спецсимволов, пунктуации.

- Нормализация: Приведение текста к нижнему регистру, стемминг, лемматизация.

- Удаление малозначимых слов: (например, "и", "в", "на").

1. **Преобразование текстов в числовые представления:**Машинные модели работают с числовыми данными, поэтому текст необходимо преобразовать в числовые векторы.

- Bag of Words: Преобразование текста в векторы, где каждый элемент соответствует **количеству раз**, которое каждое слово встречается в документе.

- TF-IDF: Взвешивание слов в документе на основе их **частоты** и их обратной частоты в корпусе.

- Word Embeddings: Использование более сложных представлений, таких как Word2Vec или GloVe, которые сохраняют **семантическую близость слов**.

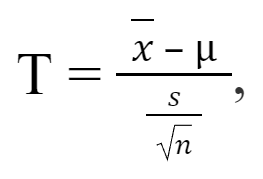
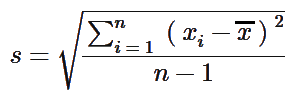
1. **Разделение данных на обучающую и тестовую выборки:**
2. **Обучение модели:**Выбор и обучение модели машинного обучения на обучающей выборке.
3. **Оценка модели:**Оценка качества модели на тестовой выборке с использованием различных метрик. Метрики: Accuracy, Precision, Recall, F1-score, ROC-AUC.
4. **Тонкая настройка и улучшение модели:**Подбор гиперпараметров и улучшение модели с помощью различных техник.Примеры: Кросс-валидация, Grid Search, Random Search.
5. **Внедрение и мониторинг:**Развертывание модели в производственной среде и мониторинг её работы.

# 2) Проверка гипотез. t-критерий с одной выборкой.(готов)

Проверка гипотез является статистическим методом, который используется при принятии решений с использованием экспериментальных данных

**t-критерий** с одной выборкой позволяет проверить гипотезу о равенстве выборочного среднего некоторому заданному числу μ

H0: = μ – основная гипотеза, H1: != μ – альтернативная гипотеза (двусторонняя), в случае, если основанная будет не верна, H1: < μ (левосторонняя), H1: > μ (правосторонняя)

– среднее значение, μ – гипотетическое среднее, s – выборочное стандартное отклонение несмещённое, n – размер выборки, T - это t-статистика

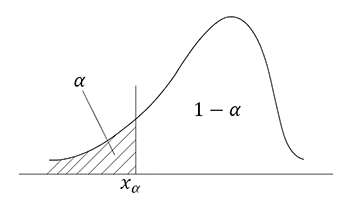
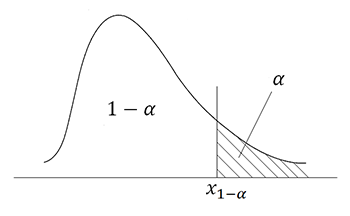
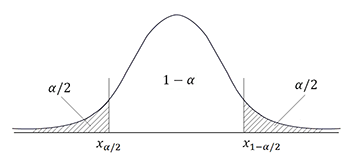
Далее задается уровень значимости α (обычно 0,05) и определяется критическая область tкрит = t0,05 \* n

После смотрится куда попало значение t, если в критическую область, то основная гипотеза отвергается и принимается альтернативная и наоборот

**p-значение (p-value)** показывает вероятность получения тестовой статистики (или более экстремального значения), если нулевая гипотеза H0 верна. Меньшее p-значение указывает на более сильные доказательства против H0​.

Для двустороннего t-теста p-value определяется как — вероятность того, что t-статистика в распределении Стьюдента с **df** степенями свободы больше абсолютного значения наблюдаемой t-статистики. (df=n−1)

Для одностороннего t-теста p-value определяется как

Виды критических областей: двусторонняя, левосторонняя, правосторонняя

# 3) Выбор модели с использованием кросс-валидации.(готов)

В качестве примера степень полинома, как в рк рассказываем, как строится каждая точка

Кросс-валидация – метод, который позволяет оценить модель, разбивая доступные данные на несколько частей и выполнить несколько итераций обучения и оценки на различных комбинациях этих частей.

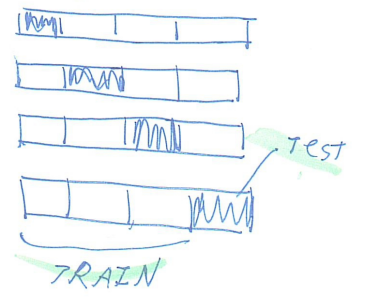
1 Разделение данных на обучающую выборку и отложенную выборку.

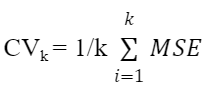
2 Применение кросс-валидации на обучающей выборке для оценки моделей.

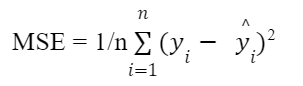
3 Выбор модели, которая показывает наилучшие показатели на основе результатов кросс-валидации.

4 Дообучение выбранной модели на полном обучающем наборе данных.

5 Оценка выбранной модели на отложенной выборке.

Пример разбиения датасета на фолды



По этой формуле строится график, находим среднее по всем ошибкам (для тренировочных и проверочных данных)

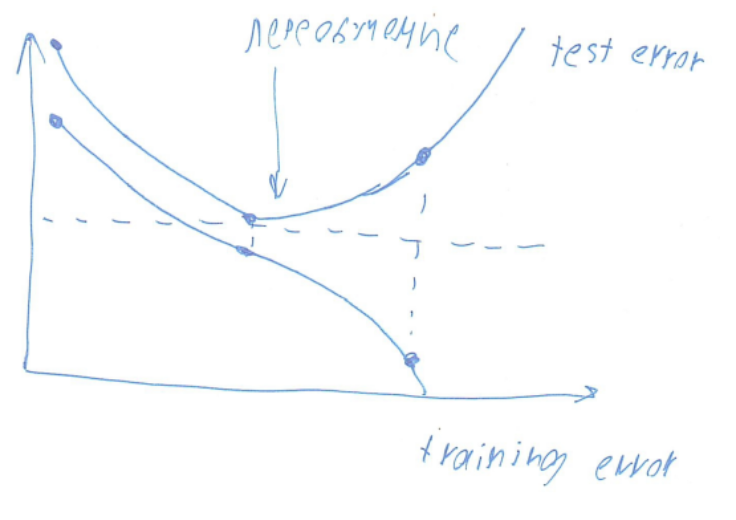
среднеквадратическая ошибка

Y без шапки это реальные данные из фолдов

Y с шапкой посчитанные теоретические (результат регрессии)

Формула линейной регрессии: 

y - зависимая переменная, xi - независимые переменные, βi - параметры регрессии, ε - ошибка модели. Степень полинома здесь это количество членов (которые напихает Папулин)



Нижний график trainLoss верхний validationLoss

нижняя ось степень полинома

Вертикальная ось ошибка

# 4) Оценка параметров. Метод максимального правдоподобия.(готов)

**Оценка параметров** — это процесс нахождения числовых значений параметров распределения, которое предположительно описывает данные.

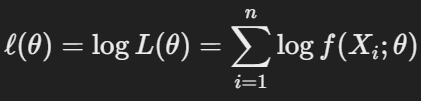
(Дополнительно: основывается на принципе нахождения таких значений параметров модели, которые максимизируют вероятность (правдоподобие) наблюдаемых данных при данной модели.)

Одним из методов оценки параметров является:**Метод максимального правдоподобия (ММП)**. Он основывается на максимизации функции правдоподобия, чтобы найти оценки параметров, которые делают наблюдаемые данные наиболее вероятными.

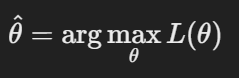
Идея метода заключается в следующем:

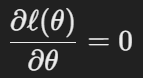
**Определение функции правдоподобия**:

Пусть X1,X2,…,Xn​ — независимые и одинаково распределенные случайные величины с плотностью вероятности f(x;θ), где θ — вектор неизвестных параметров.

Функция правдоподобия L(θ) определяется как совместная плотность вероятности всех наблюдаемых данных:

Для удобства еще есть Логарифмическая функция правдоподобия



Максимизация функции правдоподобия

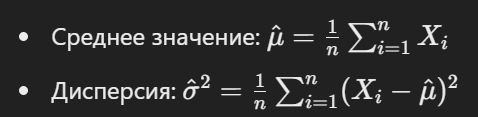
Производные логарифмической функции правдоподобия

Решение этого уравнения дает ММП-оценки параметров

**Пример применения ММП для нормального распределения**

**Данные**: Допустим, у нас есть выборка из нормального распределения.

Формулы для оценки методом максимального правдоподобия будут такими



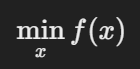
Эти оценки являются наиболее вероятными значениями параметров, которые делают наблюдаемые данные максимально вероятными согласно модели нормального распределения.

**Папулин: В общем виде про метод, в качестве примера нормальный закон распределения и как в нем найти параметры.**

# 5) Задача оптимизации. Градиентный спуск.(готов)

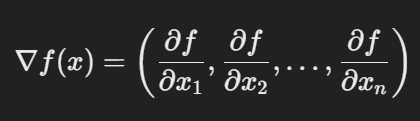
**Задача оптимизации** заключается в нахождении минимума или максимума функции при заданных ограничениях (если они есть). В общем виде задача оптимизации формулируется следующим образом:

* **Целевая функция** f(x) : Это функция, которую нужно минимизировать или максимизировать.
* **Ограничения** (необязательно): Могут быть неравенствами или равенствами, которые переменные x должны удовлетворять.

**Безусловная оптимизация** относится к задаче оптимизации без каких-либо ограничений. Формально, задача безусловной минимизации функции f(x) записывается так.

**Градиентный спуск** — это один из наиболее популярных методов для решения задач безусловной оптимизации. Он используется для минимизации функции путем итеративного перемещения по направлению наибольшего убывания функции.

**Основные концепции градиентного спуска**



**Градиент** ∇f(x) : Вектор частных производных функции f. Он указывает направление наибольшего увеличения функции.

**Шаг итерации** **α**: Малое положительное число, определяющее **длину шага при перемещении в направлении антиградиента**.

**Правило обновления**: На каждом шаге алгоритма текущее значение переменной **x** обновляется по правилу.

где Xk — текущее значение переменной на k-й итерации, Xk+1​ — обновленное значение.

**Процесс градиентного спуска**

1. **Инициализация**: Выберите начальное значение **X0** и шаг итерации **α**.
2. **Итерация**:

- Вычисление градиента функции в текущей точке xk​: 

- Обновление значений переменной по правилу выше

1. **Проверка условия остановки**: Останавливаемся, если градиент достаточно мал (или если выполнено другое условие, например, достигнуто максимальное число итераций).

**Папулин: Градиентный спуск, безусловная оптимизация, как происходит подстройка X без привязки к линейной регрессии**

# 6) Регрессия. Метод наименьших квадратов и градиентный спуск.(готов)

**Линейная регрессия** – это метод моделирования зависимости между зависимой переменной y и одной или несколькими независимыми переменными X. Цель состоит в том, чтобы найти линейную функцию, которая наилучшим образом аппроксимирующую данные.

линейная регрессия выражается следующим уравнением

(получаем предсказанные значения)

- параметры модели (**коэффициенты регрессии**), ​ - независимые переменные, **ϵ** - ошибка модели.

**Метод наименьших квадратов (Ordinary Least Squares, OLS) -** используется для оценки параметров линейной регрессии. Он минимизирует сумму квадратов ошибок (разниц между фактическими и предсказанными значениями).

Функция потерь для метода наименьших квадратов

где yi​ - фактические значения (из датасета).

- предсказанные значения (формула выше)

**Градиентный спуск (Gradient Descent) -** это итеративный метод оптимизации, используемый для нахождения минимума функции. Он часто применяется, когда прямое решение уравнений невозможно или затруднительно.

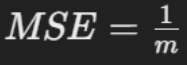
**Шаги градиентного спуска**

**1. Инициализация параметров**:

- параметры модели устанавливаются в нули или случайные небольшие числа

**2. Вычисление функции потерь**:

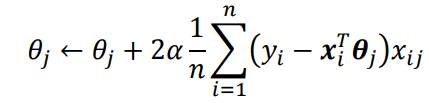
Для каждой точки данных вычисляем, насколько наши предсказания отличаются от реальных значений y. Это дает нам общую ошибку (здесь через среднекв.ошибку MSE).

\*

**3. Вычисление градиента**:

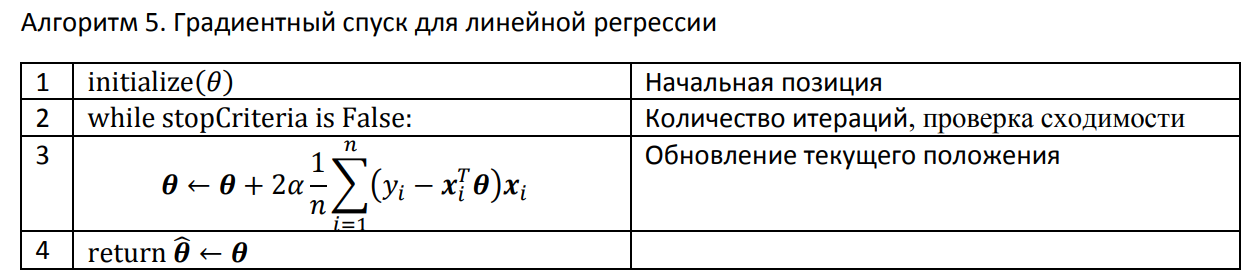
Градиент — это вектор, указывающий направление и скорость, с которой нужно изменять параметры, чтобы уменьшить ошибку.

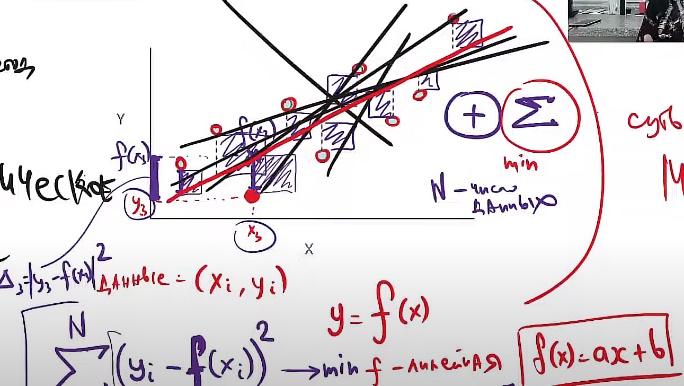
**4. Обновление параметров**:

Обновляем параметры в направлении, противоположном градиенту, чтобы уменьшить ошибку.(вместо **тета** пишем **бета**)

**α** - шаг обучения (learning rate)который мы делаем в направлении градиента.

**5. Повторение**: Повторяем шаги 2-4, пока изменения в функции потерь не станут незначительными, что означает, что мы приблизились к минимальному значению функции потерь.



**Папулин:** 

Методы поиска наименьших параметров есть такие… вводная часть Понимание как работает алгоритм

Сказать как оно расчитывается формула без регуляризации

Градиентный спуск -> функция потерь

# 7) Регрессия. Градиентный спуск и стохастический градиентный спуск.(Готов)

Регрессия — это метод статистического анализа, используется для моделирования и анализа отношений между переменными. Основная цель — определить, как зависимая переменная изменяется при изменении одной или нескольких независимых переменных.

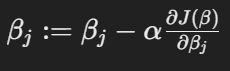
**Градиентный спуск** — это итеративный метод оптимизации, который используется для нахождения минимума функции.

**Основные шаги:**

- Инициализировать веса β случайными значениями.

- Вычислить градиент функции стоимости по каждому весу.

- Обновить веса согласно правилу обновления.

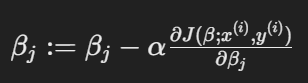
- Повторять до сходимости.

Формула обновления весов в градиентном спуске

**βj​** — текущий вес, **α** — шаг итерации,

— частная производная функции стоимости (ошибки) **J(β)** по **βj**​.

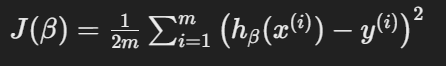
**Стохастический градиентный спуск (SGD) -** отличается от обычного градиентного спуска тем, что обновление весов происходит после каждого примера в наборе данных, а не после прохождения всего набора данных.(то есть мы не сразу охватываем весь набор обучающих данных а делаем это для каждой точки)

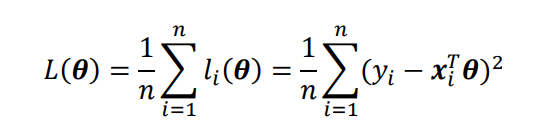
Алгоритм тот же, но вычисляем градиент и обновляем веса, для каждого примера в наборе данных

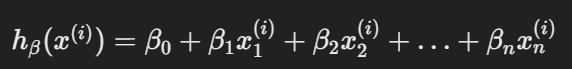
Основная Формула обновления весов в стохастическом градиентном спуске

- i-й пример из обучающего набора данных.

**Применение к конкретной форме функции**

Мы можем применить градиентный спуск и стохастический градиентный спуск к любой функции, для которой можно вычислить производную. Например, если у нас есть **функция стоимости (ошибки)** **J(β)** в линейной регрессии, такая как среднеквадратичная ошибка (Mean Squared Error, MSE):



Где  лучше h везде заменить на y

**(Опционально) Линейная регрессия** – это метод моделирования зависимости между зависимой переменной y и одной или несколькими независимыми переменными X. Цель состоит в том, чтобы найти линейную функцию, которая наилучшим образом аппроксимирующую данные.

линейная регрессия выражается следующим уравнением

(получаем предсказанные значения)

- параметры модели (коэффициенты регрессии), ​ - независимые переменные, ϵ - ошибка модели.

**Папулин: По стахостическому -> можем применить к определенной форме нашей функции**

**Папулин: MSE = градиент приравненный к 0**

**Папулин: стохастический можно применить к **

# 8) Регрессия и классификация. Метод ближайших соседей.(Готов)

**Отсюда вводную, а на следующей странице все остальное**

**Регрессия** - это метод который используется для предсказания значения целевой переменной на основе значений одной или нескольких независимых переменных. Существует несколько типов регрессии, таких как линейная регрессия, полиномиальная регрессия, логистическая регрессия и т.д.

**(Опционально)Линейная регрессия** моделирует зависимость между переменными, используя линейную функцию. Формула линейной регрессии имеет вид:

**y^**​ - предсказанное значение целевой переменной, **β0​** - свободный член (интерсепт),

**βi** - коэффициенты регрессии, **xi​** - свободные переменные.

Для нахождения коэффициентов регрессии используется метод наименьших квадратов, который минимизирует **сумму квадратов ошибок**:

**yi**​ - наблюдаемое значение,

**m** - количество наблюдений.

**Классификация -** это задача машинного обучения, которая заключается в присвоении объекта к одному из нескольких заранее определённых классов. Одним из популярных методов классификации является логистическая регрессия.

**(Опционально)**

**Логистическая регрессия** используется для бинарной классификации и предсказывает вероятность принадлежности к одному из двух классов. Модель логистической регрессии выражается следующим образом:

логистическая функция (сигмоида),

 - вероятность того, что y равно 1 при данных x.

**Логистическая функция потерь:**



Здесь:  и **yi** - истинное значение класса (0 или 1).

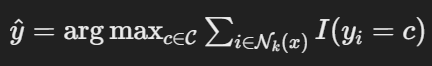
### 

### 

**Метод ближайших соседей, k-NN**( k-Nearest Neighbors) - это метод обучения, который использует близость к объектам в обучающем наборе для классификации или регрессии.

(дополнительно: основывается на предположении, что объекты, которые находятся близко друг к другу в пространстве признаков, имеют похожие значения целевой переменной. Классификация или предсказание значения целевой переменной осуществляется на основе ближайших соседей в обучающей выборке.)

Для **классификации k-NN** объекту присваивается класс, который наиболее часто встречается среди его **k** ближайших соседей. Формально, класс **y^**​ объекта **x** определяется так:



**C** - множество всех классов, **Nk​(x)** - множество индексов **k** ближайших соседей объекта **x**,

**I(⋅)** - индикаторная функция, которая равна 1, если условие истинно, и 0 иначе.

**Регрессия k-NN**

Для регрессии k-NN предсказанное значение **y^**​ объекта **x** определяется как среднее значение целевой переменной среди его **k** ближайших соседей:

**Nk​(x)** - множество индексов **k** ближайших соседей объекта **x**,

**yi​** - значение целевой переменной для соседа **i**.

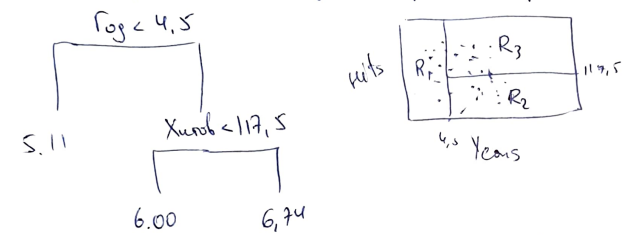
Метод k-NN не требует явного обучения модели, но может быть вычислительно затратным для больших наборов данных.

**Папулин: Формулы регрессии классификации и ближ соседей**

**Папулин: Метод ближайших соседей, формулы для регр классификации**

# 9) Регрессия. Деревья решений. Критерии деления.(copypaste)

Деревья решений – параметрический контролируемый метод обучения, используемый для классификации и регрессии.

Цель – создать модель, которая предсказывает значения целевой переменной, изучая простые правила принятия решений

Обучение – разбиение пр-ва признаков для мн-во возможных значений x1, x2,…, xp на J уникальных непересекающихся регионов R1, R2,…, RJ

Предсказание – для каждого наблюдения, которое попадает в регион RJ, мы предсказываем одно и то же значение, которое соот-ет среднему целевому значению для обучающих наблюдений из RJ

Общая задача: )2 🡪 min, где – среднее целевое значение наблюдений из обучающего набора в j-ой области

Используется жадный (потому что не оценив. послед. шаги, а только на основе тек. данных) подход – рекурсивное бинарное деление

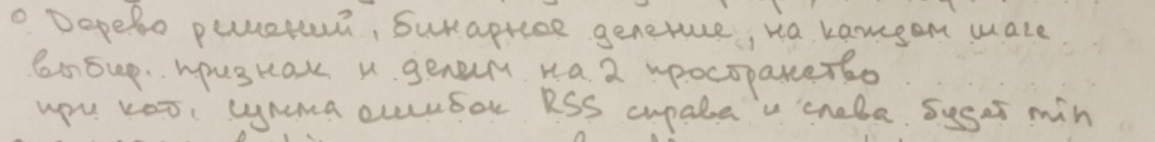
Построение дерева

Выбирается признак j и точка деления s такие, что при делении пр-ва на регионы {x|xj < s} и {x|xj >= s} дает самое маленькое RSS = )2

Для признака j и точки деления s получаем: R1(j,s) = {x|xj<s} и R2(j,s) = {x|xj >= s}

Необходимо минимизировать выражение: = []

**Папулин:**Про деревья про бинарное деление на каждом шаге выбираем признак и точку деления что каждый раз подпространство делим на два и что суммы Rss слева и справа минимальна

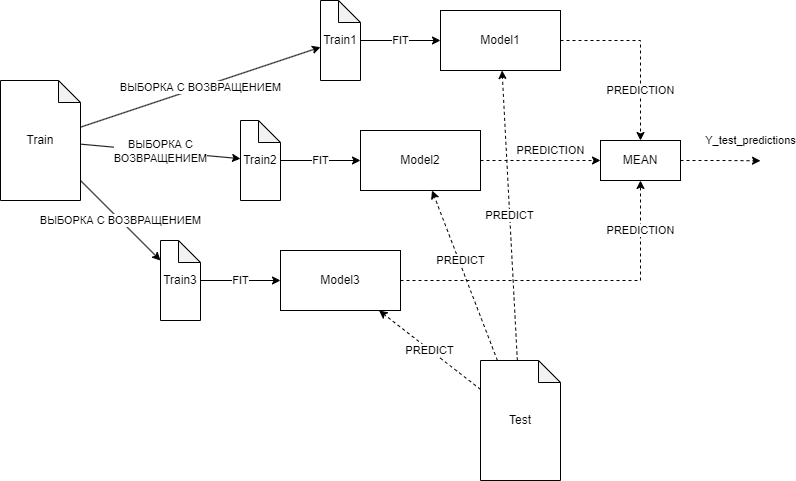
**Папулин:**

# 10) Регрессия и классификация. Ансамблевые моделей. Бэггинг, бустинг и стекинг(Готов)

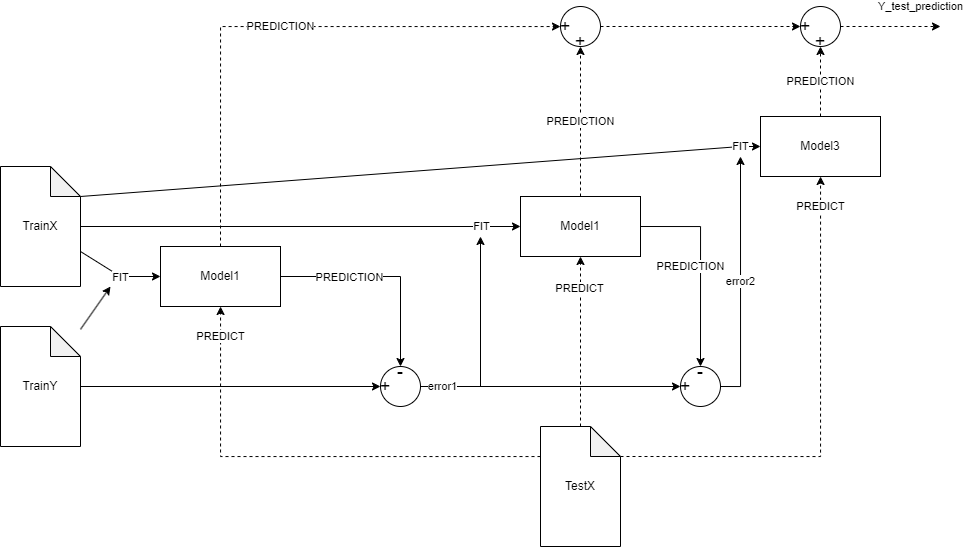
Ансамблевые методы – подход, в котором несколько моделей объединяются для улучшения качества предсказания, если применение одной модели дало неудовлетворительный результат.

Бэггинг – метод, в котором используется комбинация из нескольких моделей, каждая из которых обучается на случайной выборке. Каждая из них дает прогноз, а финальный результат получается путем усреднения.

Обучающие множества формируются посредством техники бутстреп из одного исходного обучающего множества.



Бустинг – метод, в котором модели строятся последовательно, и каждая следующая формируется на исправлении ошибок, допущенных предыдущей



Стекинг – метод, когда несколько базовых моделей обучаются независимо, а затем их прогнозы используются в качестве входных данных для метамодели

В общем виде стекинг состоит из двух видов моделей:

• Базовые модели (две и более): линейная регрессия, метод опорных векторов, деревья решений и пр.

• Метамодель (одна): как правило, линейная регрессия или логистическая регрессия При этом входными значениями для метамодели являются предсказания от базовых моделей, выполняя роль входных признаков.

Процесс обучения можно представить в виде двух стадий:

• Обучение базовых моделей на обучающем множестве

• Обучение метамодели на предсказаниях базовых моделей.

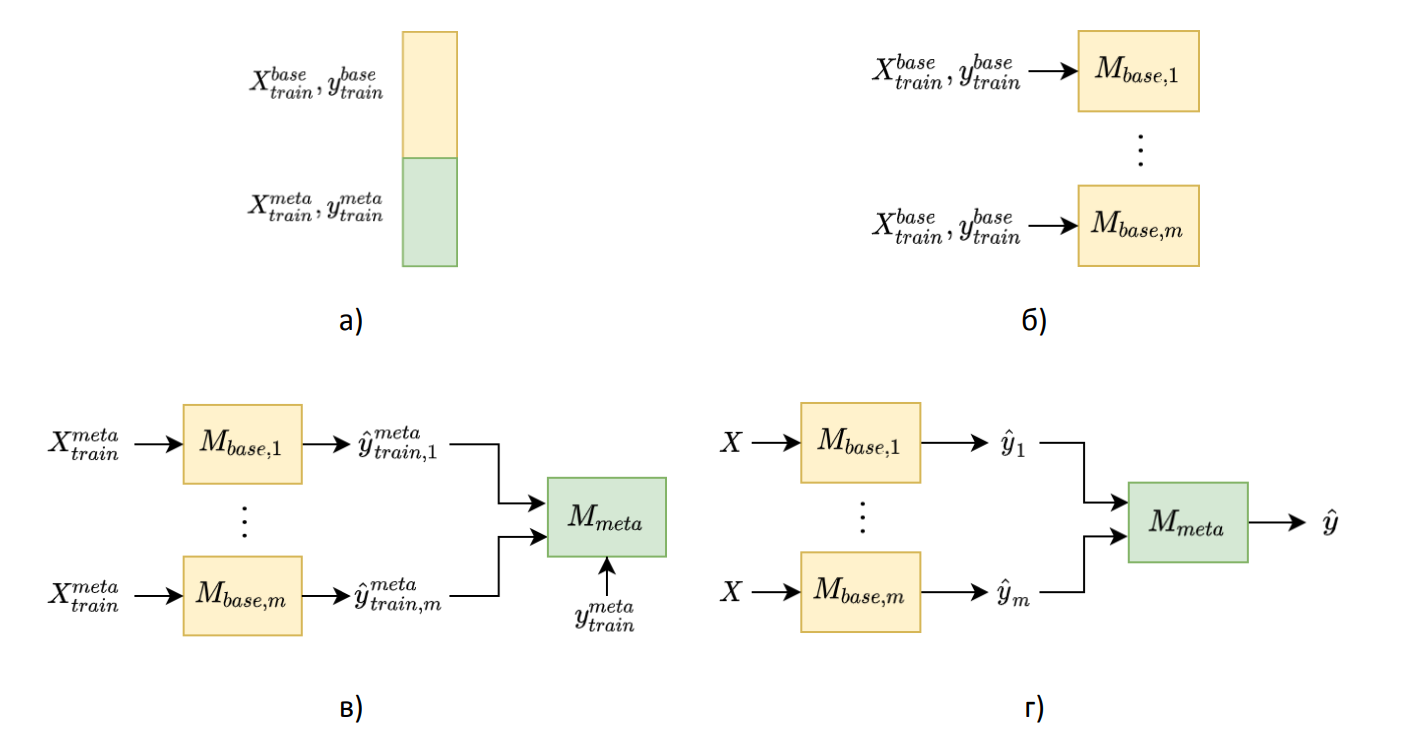
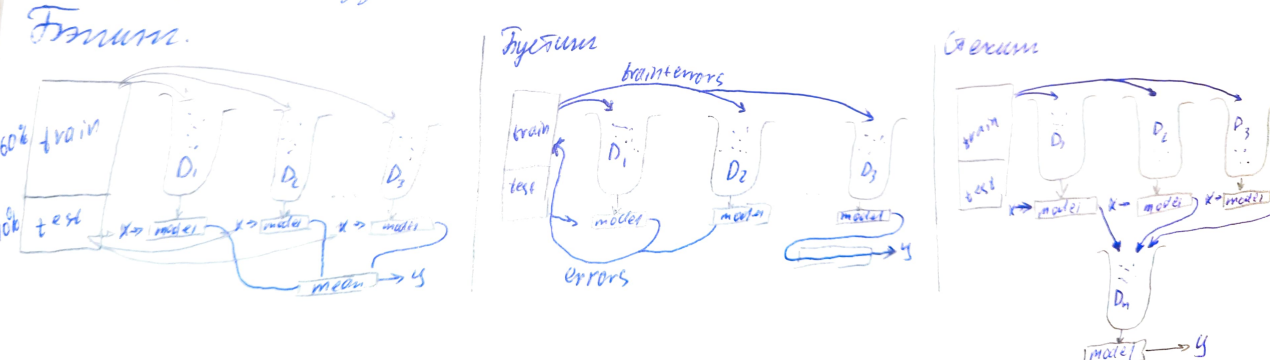


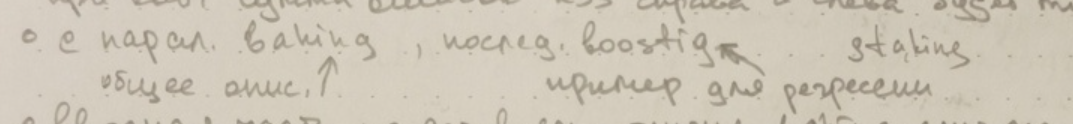
Рисунок – Обучение и предсказание в стекинге: а) обучающее множество для базовых моделей и метамодели; б) обучение базовых моделей; в) обучение метамодели; г) предсказание на новых данных



**Папулин:**Параллельное обучение и последовательное обучения

В качестве бустинга обучение лин регрессии

**Папулин:**



# 11) Регрессия и классификация. Деревья решений. Бэггинг, случайный лес и сверхслучайные деревья.(copypaste)

**Беггинг** для регрессии: = , где В – кол-во базовых моделей (деревьев), обученных на В различных обучающих мн-вах

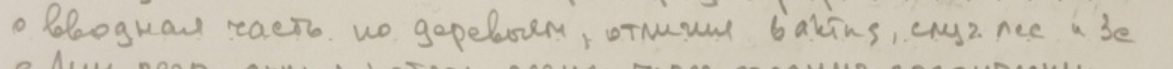
**Случайный лес** – построение мн-ва деревьев из обучающих выборок, полученных посредством случайного разбиения исходного обучающего набора. Когда необходимо сделать разделение, из всех р признаков выбирается случ. образом m кандидатов, m =

Если в наборе данных есть очень сильный признак при других менее сильных признаках, то в коллекции деревьев большинство или все из них будут использовать сильный признак на верхнем сплите. Следовательно все деревья будут похожи между собой

**В сверхслучайных деревьях** случайность в том, как вычисляются разделения в узлах. Как и в случайных лесах, используется случайное подмножество признаков, но вместо поиска наиболее оптимальных порогов, они произвольно выбираются для каждого возможного признака, и наилучший из этих случайно генерируемых порогов выбирается как лучшее правило для разделения узла

Уменьшение дисперсии за счёт увеличения смещения + случайный выбор порогового значения (смещение – разница между предсказанным значением и фактическим)

**Папулин:** Все в контексте деревьев сделать вводную часть по деревьям

**Папулин:**

# 12) Регрессия. L1, L2 регуляризация.(Готов)

**Линейная регрессия**

Линейная регрессия моделирует зависимость между входными переменными **X** и целевой переменной **y** с помощью линейной функции:

где **w** - вектор весов, **b** - смещение (intercept).

**L1 и L2 регуляризация**

Регуляризация помогает избежать переобучения, путем добавления штрафа за большие значения весов, делая модель более обобщенной и устойчивой к шуму в данных. Она также может способствовать sparsity (разреженности) в весах, что особенно полезно в случае L1 регуляризации, когда многие веса могут стать нулевыми.

**L1 регуляризация (Lasso)** Добавляет сумму абсолютных значений весов к функции потерь:

где λ - коэффициент регуляризации.

**L2 регуляризация (Ridge)** Добавляет сумму квадратов весов к функции потерь:



**Например** для линейной регрессии функция потерь (MSE) выглядит так:



Для логистической регрессии функция потерь (кросс-энтропия) выглядит так:



**Регуляризация через ограничения**

Регуляризацию можно интерпретировать как наложение ограничений на веса модели. Например, для L2 регуляризации:

* **L2 регуляризация** может быть представлена как ограничение на сумму квадратов весов:

где **C** - некоторое фиксированное значение.

**L1 регуляризация** как ограничение на сумму абсолютных значений весов:



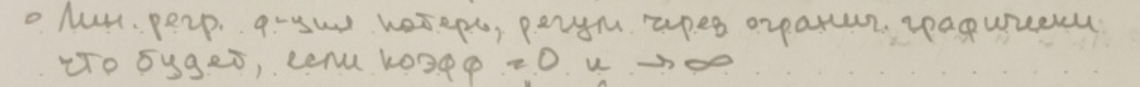
**Эффект коэффициента регуляризации λ**

* **Если λ=0:** Регуляризация отсутствует, и модель может переобучаться на тренировочных данных.
* **Если λ стремится к бесконечности:** Все веса стремятся к нулю, что приводит к сильно упрощенной модели (например, константное предсказание), что может привести к недообучению.

Регуляризация - это мощный инструмент для улучшения обобщающей способности модели и её устойчивости к шуму в данных.

**Папулин:**Функцию потерь л1 л2 показать регуляризацию через ограничения

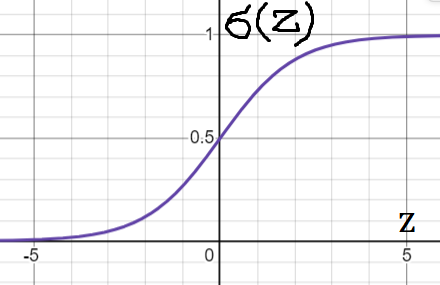
Для чего нужна что он контролирует что если он 0 что если он бесконечность

**Папулин:**

# 13) Классификация. Логистическая регрессия (бинарная классификация).(готов)

**Классификация** – задача МО, которая заключается в соотнесении наблюдения к дискретному классу из конечного множества.

**Логистическая регрессия** является методом классификации и относится к классу методов обучения с учителем. Для обучения используется множество с известными целевыми значениями (метками) **𝑦** для каждого наблюдения/экземпляра. Логистическая регрессия относится к параметрическим методам, поэтому необходимо найти значения параметров, при которых модель будет иметь наименьшую ошибку классификации.

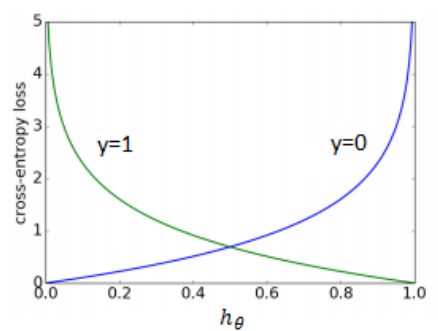


Логистическая функция (сигмоида) в общем виде:

,

где — параметры модели;

— вектор признаков входных данных.

Вероятность принадлежности к классу 1:

Вероятность принадлежности к классу 0:

Функция принятия решения:

Идеальный классификатор присвоит 1 корректному исходу и 0 для некорректного

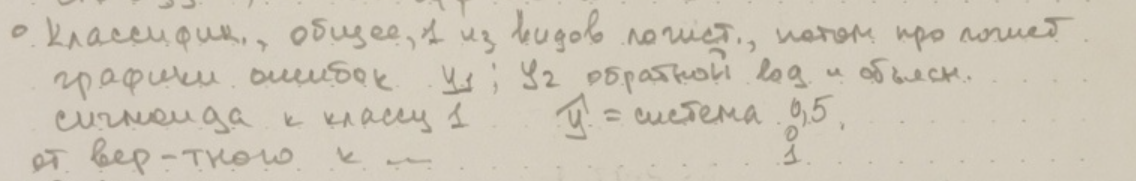
Обучение (оценка параметров) = ,  
где — функция потерь, перекрёстная энтропия.

logloglog,  
где — вероятность принадлежности к классу i при параметрах

**Папулин:** Функция потерь лог регрессии графики ошибок

Форстропия

Написать Yшапка с системой если, перейти от вероятностного к системе уравнений с условием

**Папулин:** 

# 14) Классификация. Полиномиальная логистическая регрессия (многоклассовая классификация).(готов)

**Классификация** – задача МО, которая заключается в соотнесении наблюдения к дискретному классу из конечного множества.

**Логистическая регрессия** является методом классификации и относится к классу методов обучения с учителем. Для обучения используется множество с известными целевыми значениями (метками) 𝑦 для каждого наблюдения/экземпляра. Логистическая регрессия относится к параметрическим методам, поэтому необходимо найти значения параметров, при которых модель будет иметь наименьшую ошибку классификации.

**Полиномиальная логистическая регрессия** является методом классификации, который обобщает логистическую регрессию для решения задачи многоклассовой классификации, то есть задачи соотнесения наблюдения к дискретному классу (мн-во натуральных чисел без нуля).

Вероятность принадлежности классу при параметрах модели заданном векторе признаков входных данных :

При условии, что

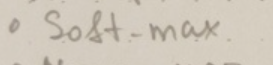
и

Функция принятия решения:

Обучение (оценка параметров) = ,  
где — функция потерь, перекрёстная энтропия.

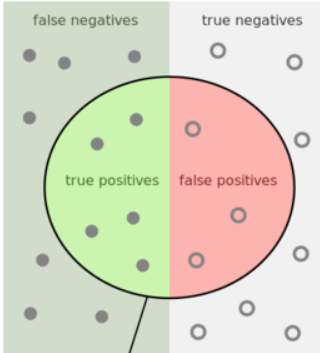
loglog,  
где

**Папулин:**

****

# 15) Классификация. Методы определения качества классификаторов.(copypaste)

Доля ошибок классификации: ErrorRate: ), где I) = {

Доля правильных классификации: Accuracy: ), 

где I) = {

|  | Y = 1 | Y = 0 |
| --- | --- | --- |
| Y^ = 1 | TP | FP |
| Y^ = 0 | FN | TN |

Первая буква – что получилось

Вторая буква – что ожидалось

ТР – ожидалось 1, получилось 1

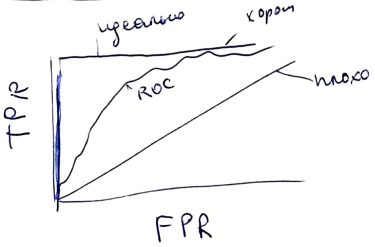
ТN – ожидалось 0, получился 0

FP – ожидался 1, получился 0

FN – ожидался 0, получилась 1

**Accuracy** = (TP + TN)/(TP + TN + FP + FN) – **доля правильных ответов** (на всем множестве)

**Recall** = TP/(TP + FN) – количество объектов, что и ожидалось 1 и получилось 1, от всех, что получилось 1 (**доля правильных ответов на множестве {y\_i = 1}**)

**Precision** = TP/(TP + FP) – количество объектов, что ожидались 1 и реально получились 1, от всех ожидавших 1 (**доля правильных ответов на множестве {y\_i^ = 1}**)

**F1** = 2 \* Precision \* Recall / (Precision + Recall)

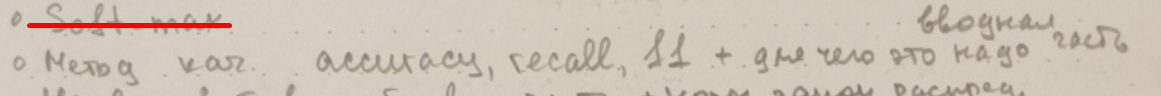
**ROC** = Recall / FPR, где FPR = FP / (FP + TN) – количество получившихся 0, а ожидали 1 от всех, что ожидали 0 (отношение доли правильных ответов на множестве {y\_i = 1} к доли неправильных ответов на множестве {y\_i = 0})

**TPR** = TP / (TP + FР)

**Папулин:**accuracy precision recall f1 f2

Вводная часть

**Папулин:**



# 16) Классификация. Наивный байесовский классификатор с нормальным законом распределения.(Готов)

Теорема Байеса:

,  
где — вероятность гипотезы A при наступлении события B

— вероятность наступления события B при истинности гипотезы A

— вероятность гипотезы А вообще

— вероятность наступления события B вообще

В контексте классификации для одного наблюдения:

,  
где – вектор признаков; y – метка класса.

Так как не зависит от y, то , следовательно

Наивный байесовский классификатор называется “наивным”, потому что считается истинным предположение, что признаки являются независимыми. Отсюда следует, что

(16.1)

Для нахождения и используем метод максимального правдоподобия. Функция правдоподобия для логистической регрессии:

,  
где n — количество независимых одинаково распределенных наблюдений; X — матрица признаков.

Из (16.1) следует

После нахождения можно рассчитать и

Предсказание имеет вид:

При нормальном распределении (Гаусса) предсказания для х будет выглядеть так:

,  
где ;

— функция плотности нормального распределения;

— среднее значение j-ого признака для класса k;

— дисперсия j-ого признака для класса k.

При бинарной классификации:

, где где n1 – количество объектов класса 1; n0 – кол-во объектов класса 0.

= xij – оценка среднего значения признака j для класса 1 равна среднему арифметическому значений признака j объектов y = 1, а I – индикаторная ф-ия

= xij

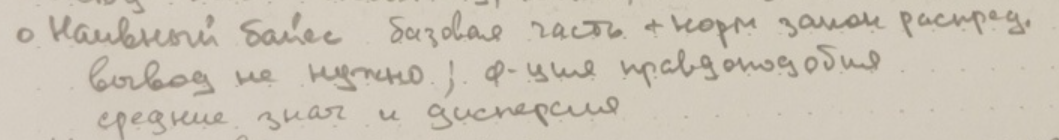
= 2 – оценка дисперсии признака j для объектов класса 1. Она равна среднему арифметическому квадратов разницы значений признака j

= 2

**Папулин:**Общая формула базовую часть а потом для разных законов + нормального распределения формула

Параметры полученные это сред значение для каждого класса и дисперсия

**Папулин:**

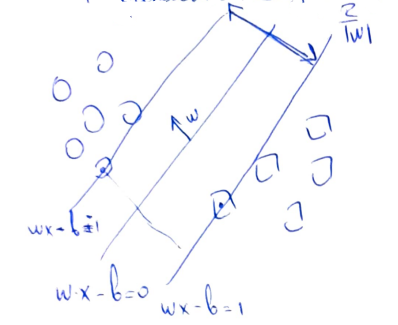


# 17) Классификация. Метод опорных векторов.(copypaste)

Цель SVM – найти гиперплоскость, которая максимально разделяет классы данных.

Зазор – максимальное расстояние до ближайших точек каждого кластера.

Опорный вектор – это точки данных, которые лежат ближе всего к гиперплоскости разделения, они находятся на границе класса и определяют положение гиперплоскости.

Гиперплоскость в n-мерном пространстве может быть представлена уравнением: **w⋅x+b=0**

w - вектор весов (нормаль к гиперплоскости),

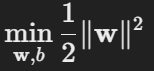
x - вектор признаков, 𝑏 - смещение (bias).

Ширина полосы: **2/|w|**

Чтобы классифицировать новое наблюдение **x**, используется решающее правило: **f(x)=sign(w⋅x+b)**

sign — знак функции (возвращает +1 или -1 в зависимости от знака аргумента).

Условие классификации для линейно разделимых данных: **yi​(w⋅xi​+b)≥1∀i** где **yi ∈{−1,+1}** — метка класса для i-го наблюдения.

SVM решает задачу максимизации зазора с ограничением:

при условии выше

Случай с мягким зазором (Soft Margin):

Для учета ошибок классификации вводятся штрафные переменные **ξi** , **C** - параметр, контролирующий степень штрафа

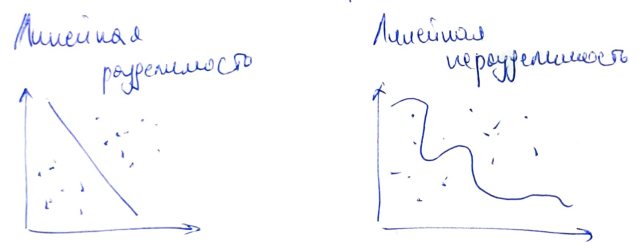
условие:

**Для первого условия которое выше**

Если yi (wx + b) больше 1, то правильная классификация

Если yi (wx + b) равно 1, то на границе

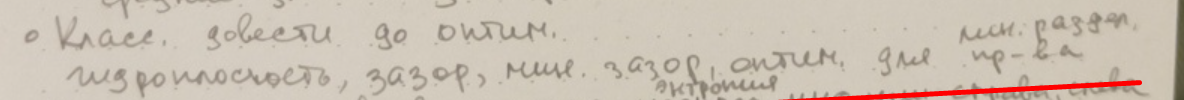
Если yi (wx + b) меньше 1, то неправильная (не с той стороны зазора)



**Папулин:** До задачи оптимизации прямой что такое гиперплоскость

Что такое зазор

Для линейно разделимого пространства достаточно

**Папулин:** 

# 18) Классификация. Деревья решений. Критерии деления.(copypaste)

Дерево решений для задачи классификации строится аналогично как для задачи регрессии. Однако оптимизация не происходит по RSS, ведь теперь категориальные значения

– ошибка классификации – доля обучающих наблюдений в регионе, которые не принадлежат классу с наибольшим кол-ом эл-ов:

1 - , где = , где R – кол-во данных класса k в регионе m

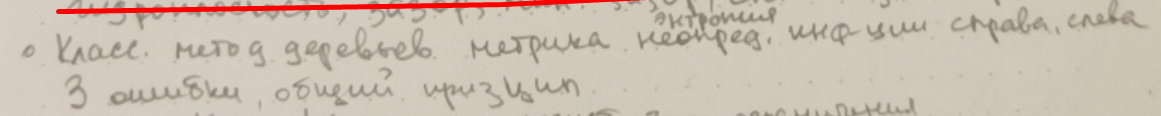
– Gini-индекс – вычисляется как сумма дисперсий по каждому классу, , где – доля обучающих наблюдений k-го класса в регионе m

– Энтропия – определяет чистоту региона, –

Задача заключается в том, чтобы найти такой признак j и точку деление s, которые дадут наибольшую определенность после разделения: = [Ec – (Eleft + Eright)], где = (j, s) – параметр деления, nc – общее кол-во элементов в выборке, Ес – мера неопределённости (энтропия)

**Папулин:**В регрессии ищем признак

Улучшение распределенности информации справа и слева

**Папулин:** 

# 19) Классификация. L1, L2 регуляризация. (Готов)

**Линейная регрессия**

Линейная регрессия моделирует зависимость между входными переменными X и целевой переменной y с помощью линейной функции:

где **w** - вектор весов, **b** - смещение (intercept).

**L1 и L2 регуляризация**

Регуляризация помогает избежать переобучения, путем добавления штрафа за большие значения весов, делая модель более обобщенной и устойчивой к шуму в данных. Она также может способствовать sparsity (разреженности) в весах, что особенно полезно в случае L1 регуляризации, когда многие веса могут стать нулевыми.

**L1 регуляризация (Lasso)** Добавляет сумму абсолютных значений весов к функции потерь:

где λ - коэффициент регуляризации.

**L2 регуляризация (Ridge)** Добавляет сумму квадратов весов к функции потерь:



Например для линейной регрессии функция потерь (MSE) выглядит так:



Для логистической регрессии функция потерь (кросс-энтропия) выглядит так:



**Классификация и L1, L2 регуляризация**

Классификация – это задача машинного обучения, при которой модель учится предсказывать категорию или класс объекта. В отличие от регрессии, где целевая переменная является непрерывной, в классификации целевая переменная является дискретной (категориальной).

**Логистическая регрессия** используется для задачи классификации и использует логистическую функцию для предсказания вероятностей классов:

**P^** - предсказанная вероятность

**Регуляризация через ограничения**

Регуляризацию можно интерпретировать как наложение ограничений на веса модели. Например, для L2 регуляризации:

* **L2 регуляризация** может быть представлена как ограничение на сумму квадратов весов:

где **C** - некоторое фиксированное значение.

**L1 регуляризация** как ограничение на сумму абсолютных значений весов:



**Эффект коэффициента регуляризации λ**

* **Если λ=0:** Регуляризация отсутствует, и модель может переобучаться на тренировочных данных.
* **Если λ стремится к бесконечности:** Все веса стремятся к нулю, что приводит к сильно упрощенной модели (например, константное предсказание), что может привести к недообучению.

Регуляризация - это мощный инструмент для улучшения обобщающей способности модели и её устойчивости к шуму в данных.

**- – — – - – — – - – — – - – — – - Дополнительно - – — – - – — – - – — – - – — – -**

**Типы классификации**

**1. Бинарная классификация:** Задача, в которой необходимо предсказать один из двух возможных классов.

**2. Мультиклассовая классификация:** Задача, в которой необходимо предсказать один из более чем двух классов.

**3. Многоклассовая классификация:** Задача, в которой каждый объект может принадлежать сразу нескольким классам.

**Модели классификации**

**1. Логистическая регрессия:** Логистическая регрессия используется для бинарной классификации и расширяется для мультиклассовой задачи с помощью методов, таких как one-vs-all или softmax.

**2. SVM (машины опорных векторов):** Метод, который ищет гиперплоскость, разделяющую классы с максимальным зазором (margin).

**3. Деревья решений и ансамблевые методы:**

**Деревья решений** создают правила на основе значений признаков.

**Ансамблевые методы** (например, случайный лес, градиентный бустинг) комбинируют несколько деревьев для улучшения производительности.

**4. Нейронные сети:** Используются для более сложных задач классификации, особенно для изображений и текста. Глубокие нейронные сети могут обучаться на больших объемах данных и извлекать сложные паттерны.

Регуляризация – методика МО для контроля сложности модели и предотвращения переобучения. Она вводит доп. ограничения на модель, чтобы сделать её более устойчивой

Переобучение возникает, когда модель имеет большую дисперсию, что м.б. вызвано большой гибкостью выбранной модели или использования большого количества признаков, сопост. с количеством наблюдений

Чтобы избежать переобучение, в формулу потерь вводится доп. компонент R()

1)Гребневая регрессия (L2)

MSE = 1/n RSS, где RSS = )2 = yi - jxij]2 – оценка параметров (обучение)

Функция потерь: L() = RSS + 𝜆 = yi - - jxij]2 + 𝜆 (первое – RSS, второе - штраф), где – коэффициент сдвига, j – параметры модели

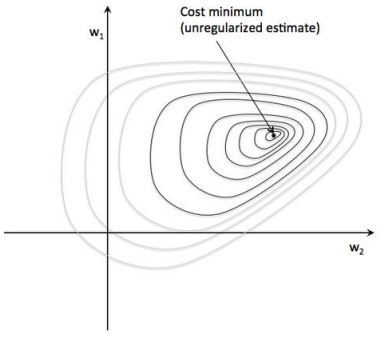
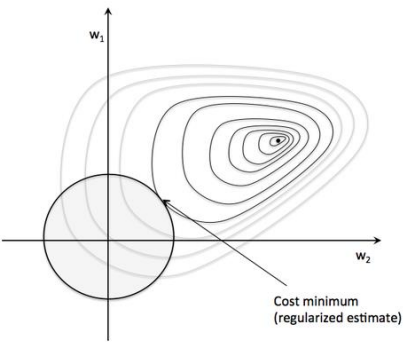
Оценка параметров: = yi - - jxij]2 + 𝜆 – при возрастании 𝜆 увеличивается штраф

Для ЛР: = -yilog – (1 - yi)log (1 - ]2 + 𝜆, где hθ,i​ - вероятность того, что i-й пример принадлежит к классу 1 при заданном векторе параметров θ.

2) Лассо (L1)

Добавляет сумму абсолютных значений коэффициентов модели к ф-ии потерь. Лассо стремится уменьшить оценку параметров до 0, в отличии от L2, которая только штрафует большое значение весов.

Всё те же три формулы, только штраф: 𝜆j|

Гиперпараметр держит баланс между предотвращением переобучения и обобщением новых данны

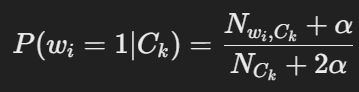
**Папулин:** 

# 20) Классификация текстовых документов с использованием наивного байесовского классификатора. Модель Бернулли.(copypaste)

Модель Бернулли — это вариант наивного байесовского классификатора, где документы представляются как двоичные векторы, указывающие на присутствие или отсутствие слов.

Основные формулы для модели Бернулли

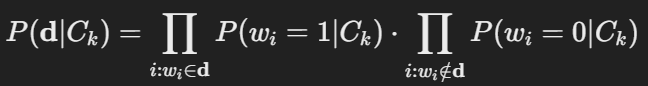
Вероятность слова в классе (присутствие/отсутствие):



**wi​=1** — слово **wi​** присутствует в документе,

**wi=0**— слово **wi​** отсутствует в документе.

Вероятность документа в классе:

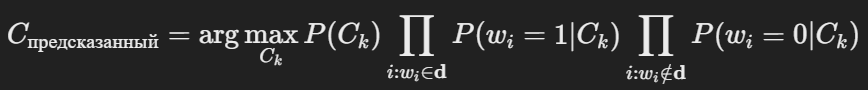
**i : wi ∈ d** — слова, присутствующие в документе, **i: wi ∉ d** — слова, отсутствующие в документе.

**Процесс классификации**

1. Предобработка текста: аналогично модели "Мультиномиальной".

2. Обучение модели: вычисление 𝑃(𝐶𝑘) и 𝑃(𝑤𝑖∣𝐶𝑘) для всех классов.

3. Классификация: для нового документа **𝑑** вычисление апостериорной вероятности для каждого класса и выбор класса с максимальной вероятностью.



Набор текстовых документов – набор векторов размером N = |V|, где |V| - размер словаря. d 🡪 b = (b1,…,bN), где d – текстовый документ, а b – вектор, а элементы – 1 или 0 определенному слову/терму в документе d

Если х является вектор из бинарных чисел, то вер-ть появления х при заданном классе y будет p(x|y) = , где – это р( = 1|y)

Для одного документа: p(d|c) ~ p(b|c) =

Для коллекции документов MLE будет иметь вид:

L(y|D;) = , где ND – количество документов в коллекции; сi – класс документа di

Оценка вероятности встретить документ класса с: = , где Nc – количество документов класса с

Оценка вероятности встретить **терм tj**в документах класса с: =где – количество документов класса с, содержащих терм tj

Предсказание: = p(c)p(d|c)

Учитывается только присутствие и отсутствие какого-то терма, а в Мультиномиальной ещё учитывается частота встречаемости этого терма

**Папулин:** В контексте текстовых методов с наивным байесом в файлах старосты(связанном с рк 2) (Minor\_Lecture\_5\_NB вроде)

Общая постановка задачи и классификация параметров

**Папулин:** 

# 21) Классификация текстовых документов с использованием наивного байесовского классификатора. Мультиномиальная модель.(copypaste)

Если х является вектор из натуральных чисел, то вероятность появления х при заданном классе y будет p(x|y) =

Апостериорная вероятность (вероятность произойдет на основе обновляющихся данных): L(y|X;) = , где пи – вероятность того, что переменная y равна y\_i, может быть задана ранее или взята из данных, а р – кол-ва признаков, а n – кол-во наблюдений

Поиск максимальных параметров : = log L(

d 🡪 x = (x1,…,xN) для одного документа: р(d|c) = – вероятность, что документ d принадлежит классу c

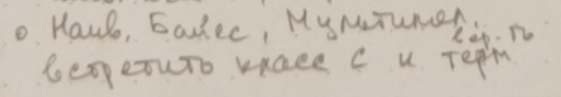
Обучение: L(y|D;) =

Оценка вероятности встретить документ класса с: =, где Nc – количество документов класса с

Оценка вероятности встретить терм tj в документах класса с: =где – количество терм tj в документах класса с, nc – количество термов в доку-ах класса с

Предсказание: = p(c) \* p(d|c)

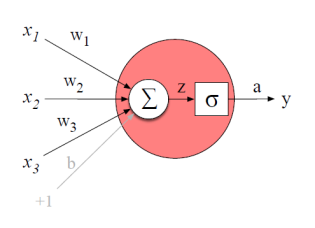
**Папулин:**Как в 20

**Папулин:** 

# 22) Классификация. Многослойная нейронная сеть прямого распространения.(готов)

**Классификация** – задача МО, которая заключается в соотнесении наблюдения к дискретному классу из конечного множества.

**Модели на основе искусственных нейронных сетей** — это способ представления особенности работы нейронов человека в виде упрощенной математической модели для решения сложных нелинейных задач компьютерными средствами.

**Компьютерная нейронная сеть** – это набор вычислительных единиц, каждая из которых принимает на вход вектор значений и возвращает одно выходное значение. Особенностью нейронной сети прямого распространения является передача значений от одного слоя к другому без обратных связей и внутрислойного взаимодействия.

Нейрон принимает взвешенную сумму входных значений:

,  
где b — смещение;

— входное значение;

— вес входа .

Сокращенная запись: ,  
где b — смещение;

— вектор входных значений;

— вектор весов входных значений.

В качестве выходного значения нейрона 𝑦 используют значение нелинейной функции 𝑎, называемой функцией активации. Функция активации принимает значение 𝑧, то есть значение линейной по 𝑥 функции: , где — функция активации.

Функции активации:

| Наименование | Формула | Производная | График |
| --- | --- | --- | --- |
| Сигмоида |  |  |  |
| Гиперболический тангенс |  |  |  |
| Кусочно-линейная ReLU |  |  |  |

Производные функций сигмоиды и гиперболического тангенса близки к 0 при больших значениях аргумента, поэтому для скрытых слоев в качестве функции активации часто используют кусочно-линейную функцию ReLU, производная которой при больших значениях аргумента равна 1.

Выходное значение Перцептрона:

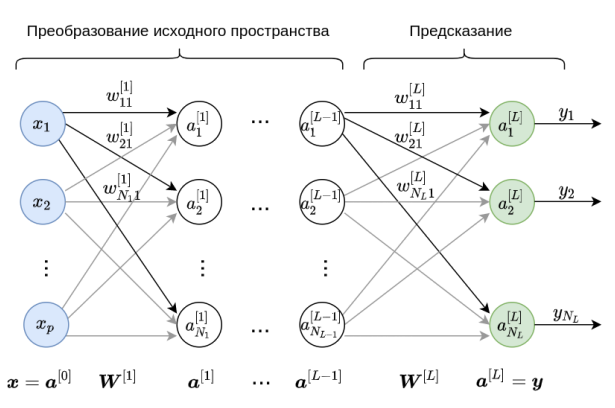


Рисунок — Многослойная нейронная сеть прямого распространения

Гибкость модели увеличивается с количеством нейронов в скрытом слое.

Алгоритм вычисления в многослойной нейронной сети прямого распространения:

| 1 |  | Загрузка наблюдения |
| --- | --- | --- |
| 2 |  | Вектор признаков одного наблюдения |
| 3 | for i in 1, …, n: | Цикл по всем слоям |
| 4 |  | Линейная функция |
| 5 |  | Функция активации |
| 6 |  | Предсказание |
| 7 | return |  |

Функции активация для выходного слоя:

| Задача | Функция активации | Функция потерь |
| --- | --- | --- |
| Регрессия | (нет функции активации) |  |
| Бинарная классификация |  |  |
| Многоклассовая классификация |  |  |

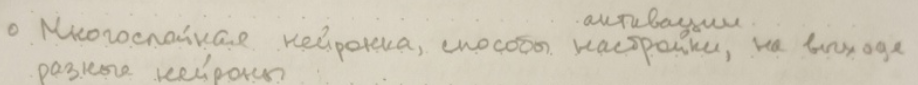
**Папулин:**В общем виде про классиф неиронную сеть

И алгоритм функция активации и какие они бываают

Для разных задач разный входной слой

Напр для бинарной классификации сигмоид

Про обучение не надо

**Папулин:** 

# 23) Кластеризация. Метод k-средних. Выбор начальных значений центров кластеров.(Готов)

**Кластеризация** – задача разбиения мн-ва объектов на кластеры (внутри каждого кластера должны остаться похожие объекты)

**Методы выбора центров:**

1. Метод К-средних – разделение наблюдений на заранее определенное кол-во кластеров
2. Иерархическая кластеризация – каждый объект кластер и после они последовательно объединяются
3. DBSCAN – кластеры формируются на основе близости между наблюдениями
4. Смесь ф-ий нормальных распределений – задается кол-во кластеров и для каждого на основе наблюдений подбираются параметры ф-ии норм-го распр-ия

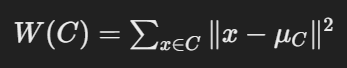
**Метод К-средних**: подход к разделению данных на мн-во **К** различных непересекающихся кластеров. Сначала надо задать кол-во кластеров **К**. Затем алгоритм на каждой итерации присваивает каждому из наблюдений один из кластеров до тех пор, пока не будет выполнен критерий остановки.

Каждое наблюдение должно принадлежать только одному кластеру и внутри каждого кластера внутрекластерная вариация (близость объектов друг к другу) **W(Ck)** для кластера **Сk** минимальна. Задача минимизации:

**С\*** = 2 – сумма отклонения значения признака внутри класса от среднего, где С\* = (,…,)

**Алгоритм:**

1. Случайным образом присвоить значения от 1 до **К** каждому наблюдению
2. Повторять до тех пор, пока не перестанут изменяться наблюдения в кластерах
   1. Вычислить центроид для каждого кластера (центроид – вектор средних значений размера р по каждому наблюдения данного кластера)
   2. Присвоить каждое набл. кластеру, чей центроид ближе

внутренняя кластерная вариация:

**x** — это точки данных, принадлежащие кластеру **C**

**μC​** — это центроид кластера **C**

**∥x−μC​∥** — это евклидово расстояние между точкой **x** и центроидом **μC​**.

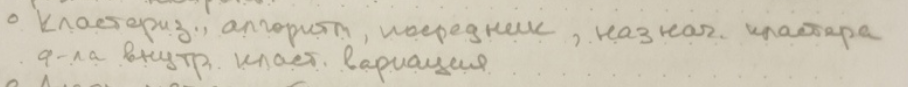
Для всего набора данных, разделенного на k кластеров, общая внутренняя кластерная вариация Wk



**Ci​** — кластер **i**, **μCi** — центроид кластера **i**.

**Папулин:**Алгоритм к средних описать алгоритм и формулу

Формулу внутренне кластерной вариации

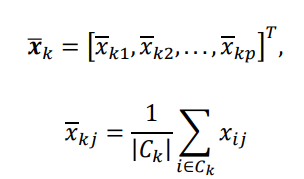
**Папулин:** 

# 24) Кластеризация. Иерархическая кластеризация. Агломеративные методы.(copypaste)

Недостаток метода k-средних – это необходимость при запуске алгоритма задавать количество кластеров 𝐾, иерархический подход этого не требует

**Агломеративная кластеризация** – тип иерархической кластеризации, при которой строится древообразная структура (дендрограмма) начиная с листьев до основания дерева, постепенно объединяя наблюдения в кластеры до тех пор, пока все они не будут принадлежать одному кластеру

Центройд — вектор средних значений признаков внутри кластера

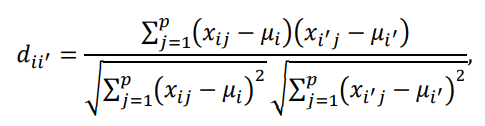


**Мера близости между двумя наблюдениями**

Евклидово расстояние



расстояние на основе корреляции (например, Пирсона)



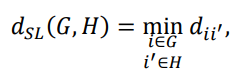
где – среднее значение признаков 𝑖-ого наблюдение

**Алгоритм:**

1. Определить расстояние схожести/различия всех пар наблюдений ( = n(n-1)/2. Каждое наблюдение рассматривать как отдельный кластер
2. Для i = n, n – 1,…, 2
   1. Определить наиболее близкие пары кластеров, объединить эти кластеры. Величина схожести/различия указывает на высоту в дендограмме, где объединение должно быть показано
   2. Определить межкластерное расстояние схожести/различия для всех пар кластеров

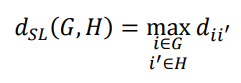
**Варианты кластеризации**

>> Метод одиночной связи (наименьшее расстояние): dSL(G, H)= dii’, где G и H – два кластера; dii’ – расстояние между двумя наблюдениями разных кластеров (евклидово расстояние = ||xi – xi’||)

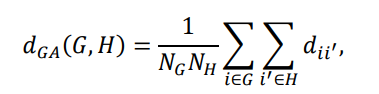


где 𝐺 и 𝐻 – два кластера; 𝑑𝑖𝑖 ′ – расстояние между двумя наблюдениями разных кластеров;

>> Метод полной связи (наибольшее расстояние): dCL(G, H)= dii’



>> Метод средней связи (среднее расстояние): dAL(G, H)= , где NG и NH – количество наблюдений в кластерах G и H



>> Центройдная связь (схожесть между центройдами кластеров)

Центроидная связь имеет значительный недостаток, связанный с инверсией. Под инверсией подразумевается ситуация, когда два кластера объединяются на высоте ниже каких-либо из его составляющих кластеров. Это приводит к сложности в визуализации и интерпретации дендрограмы.

Особенностью иерархической кластеризации является:

* меры схожести/различия между наблюдениями
* выбор метода связи между группами наблюдений
* построение дендрограммы
* выбор высоты среза дендрограммы для получения кластеров

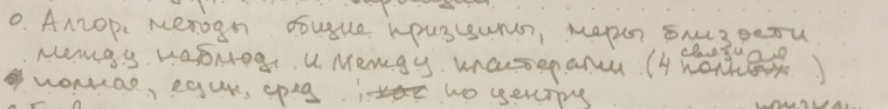
**Папулин:**Общие принципы,

Меры близости между наблюдениями

Про обьединение кластеров

4 варианта: полные связи, центроидные,…

Что такое полные связи

**Папулин:** 

# 25) Уменьшение размерности. Метод главных компонент PCA. (copypaste)

(Дополнительно: Если в наборе данных содержится более трёх переменных, то может быть очень сложно графически визуализировать данные в многомерном пространстве. Возникает задача снижения размерности. Метод главных компонент (МГК) решает эту задачу путём извлечения важной информации, содержащейся в многомерных данных. МГК отражает сохранённую информацию в значительно меньшем количестве новых переменных (факторов), называемых главными компонентами. Новые факторы являются линейной комбинацией исходных переменных.)

Метод главных компонент (principal component analysis, PCA) – на вход поступает мн-во **p** размерных данных и идёт поиск линейного подпр-ва размерности **m < p**, чтобы данные лежали в этом подпр-ве с сохранением разброса

Особенность РСА – линейное преобразование исходного пр-ва: **Z = PX**, где **Р** – матрица трансформации (вращения) **Х** в **Z**. Задача заключается в том, чтобы найти некоторое преобразование **𝑃**, при котором признаки нового пространства будут независимы

При этом ковариационная матрица **SZ = ZZT**должна быть диагональной

После преобразование **Р** над исходным **Х** получаем **SZ = D**, где **D** – диагональная матрица собственных значений

Строки итоговой матрицы **Р** и есть главные компоненты

Алгоритм нахождения **Р**, чтобы **SZ**было диагональным:

1. Центрирование данных
2. Вычисление ков. матрицы **SZ**(эл-ты по диагонали есть дисперсия признаков)
3. Вычисление собств. значений и собств. векторов **SZp = 𝜆p**, где **𝜆** – собственные значения, а **р = []** – собственные вектора
4. Выбор компонент для проекции
5. Проецирование данных

**Папулин:**Про размерность

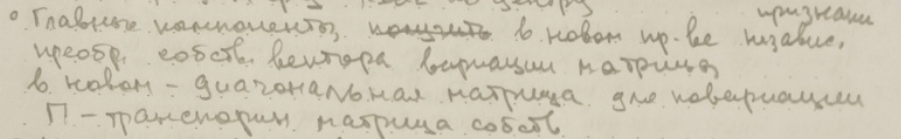
Исход набор данных в новом пространстве, в котором наши признаки независимы

Для этого использовать преобразование, это вектора нашей матрицы

Нужна диагональная корреляционная матрица в итоге, такое должно быть преобразование (суть в том что корреляции только на диагонали)

Вывод не надо писать

Про п вектора

**Папулин:** 

# 26) Рекомендательные системы. Коллаборативная фильтрация на основе сходства пользователей. (copypaste)

Рекомендательные системы предсказывают альтернативы, которые могут заинтересовать пользователя

Несколько подходов к построению рекомендательных систем:

1. На основании контента
2. Коллаборотивная фильтрация
3. Факторизация матрицы рейтингов

Недостатки

– холодный старт – новые объекты никому не рекомендуются

– тривиальность предсказаний

Основные задачи:

– предсказания: определение степени заинтересованности пользователя u в объекте i ∉ Iu обозначается как

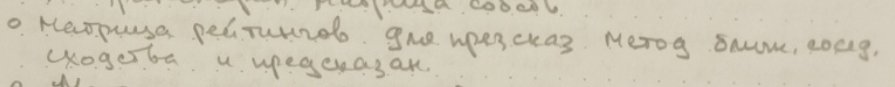
– рекомендации: определение списка из N объектов

Метод рек. на основе схожести пользователей предсказывает рейтинг Ru,i пользователя u для нового товара i исп. рейтингов, проставленных товару i наиболее похожими на u пользователями (ближ. соседи)

Определение сходства между пользователями (косинусное сх-во): su,i = sim(u,v) = – значение сходства между пользователями u и v

Предсказание (с учётом весов степени схожести соседей): = , где – объекты пользователя u, наиболее похожие на i

Центрирование по среднему: =

**Папулин:** 

# 27) Рекомендательные системы. Коллаборативная фильтрация на основе сходства элементов.(copypaste)

Идея в определении сходства между объектами i и j заключается в отборе пользователей с проставленными рейтингами для обоих объектов и применение способа вычисления сходства si,j

Косинусное сходство: si,j = sim(u, j) = – значение сходства между объектами j и i, где U – мн-во пользователей

Предсказание (с учётом весов степени схожести соседей): =

Центрирование по среднему: =

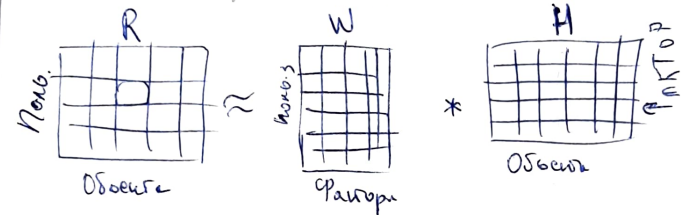
R - данные из набора R^ - предсказанные данные

**Папулин:**

# 28) Рекомендательные системы. Факторизация матрицы рейтингов. ALS.(copypaste)

**Матричная факторизация** – метод в рекомендательных системах для выявления скрытых фактов в данных, которые можно использовать для прогнозирования

**Основная задача** – разложить матрицу рейтингов **R** на произведения двух матриц **W** и **H**. При этом предсказание будет вычисляться как произведение строки **W** на столбец **H**



Задача оптимизации: = L(W, H)

Функция потерь L(W, H) = , где Li,j(W,H) = l(Ri,j, Wi\*,H\*j) - Это функция потерь для одного элемента данных. Она оценивает разницу между истинным значением Ri,j и значением, которое было предсказано после разложения матриц W и H, где мн-во Ω представляет собой мн-во всех таких позиций (i, j), для которых у вас есть некоторые наблюдения

Без регуляризации: L(W, H) = Wi\*H\*j)2

ALS (alternating least squares) — итеративный алгоритм разложения матрицы предпочтений на произведение 2 матриц

Алгоритм:

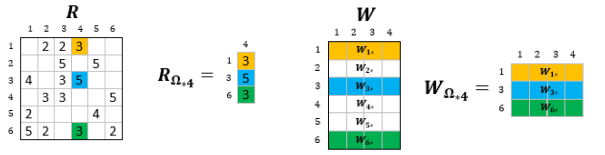
1. H 🡨 средний рейтинг для первой строки и малые случайные значения для остальных.
2. Цикл: критерий остановки
   1. W 🡨 вычисление при фиксированном H

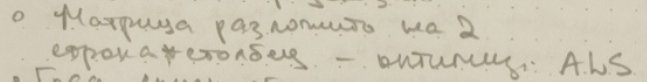
МНК для задач ЛР: = (XTX)-1XTy => для матрицы рейтингов: = (-1, где n – это итерация



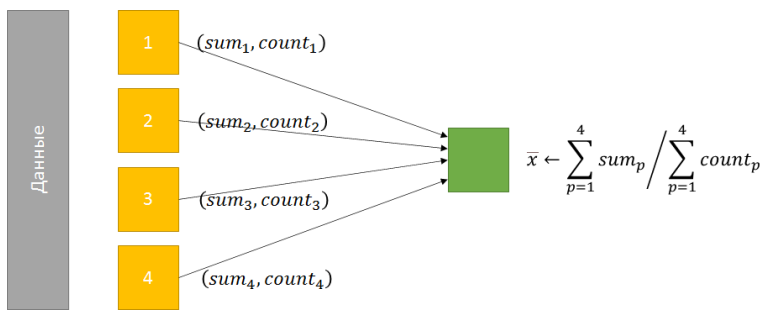
* 1. H 🡨 вычисление при фиксированном W

= (-1



**Папулин:** 

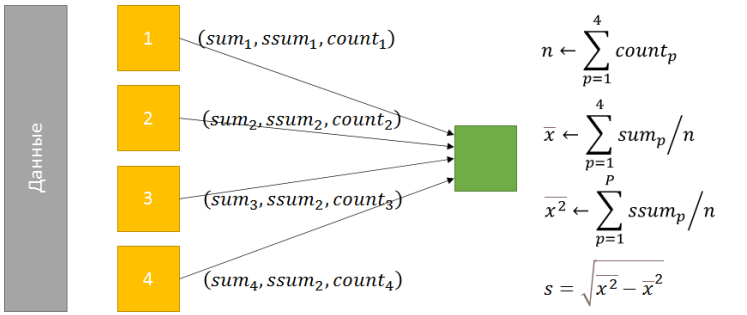
# 29) Распределенные алгоритмы. Расчет среднего значения и стандартного отклонения.(copypaste)

Среднее значение: = 

Алгоритм:

1. For p in 1..P in parallel:
2. Sump 🡨 0
3. Countp 🡨 0
4. For x in Dp:
5. Sump 🡨 sump + x
6. Countp 🡨 countp + 1
7. Collect ((sum1, count1),…,(sump, countp))
8. = /

Стандартное отклонение: s =

Алгоритм:

1. For p in 1..P in parallel:
2. Sump 🡨 0
3. Ssump 🡨 0
4. Countp 🡨 0
5. For x in Dp:
6. Sump 🡨 sump + x
7. Ssump 🡨 ssump +
8. Countp 🡨 countp + 1
9. Collect ((sum1, ssum1, count1),…,(sump, ssump, countp))
10. N 🡨 ; 🡨 /n; 🡨 /n
11. S =

**Папулин:**

# 30) Распределенные алгоритмы. Расчет косинусной меры сходства.(copypaste)

Косинусное сходство: **cosine(i,j) =**

Алгоритм. Часть 1 – Map (трансформация данных):

1. For m in 1..M in parallel: по всем задачам m (map) от 1 до M
2. For x in Dm: Dm – наблюдения в части m
3. For (xj, xk) in all\_pairs\_x: x = (x1,…,xp) – одно наблюдение
4. V 🡨 xj \* xk
5. Emit((j,k) 🡪 v) key 🡪 value

Алгоритм. Часть 2 – Reduce (агрегирование):

1. For m in 1..R in parallel: R – reduce – задач с разделением по ключу

По всем задачам m (map) от 1 до R (Reduce)

1. Reveive(||c1||,…,||cp||) Заранее вычисленные значения
2. Reveive pairs from map part
3. For (i,j) 🡪 [v1,…,vk] in pairs: vk – ненулевые значения из map части
4. 🡨
5. Cosine(i,j) 🡨

**Папулин:**

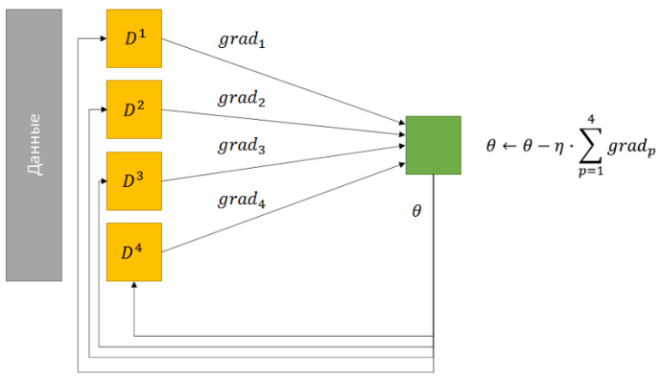
# 31) Распределенные алгоритмы. Градиентный спуск.(copypaste)

Градиентный спуск: 🡨 – η∇L(y, x, ), где ∇L(y, x, ) =

Алгоритм (обычный):

1. Initialize ()
2. For I in 1..T:
3. Grad 🡨
4. For (x,y) in D:
5. Grad 🡨 grad + (y – xT)x
6. 🡨 – η\*grad
7. Return

Распределенный градиентный спуск: 🡨 – η

Алгоритм (распределенный): 

1. Initialize ()
2. For I in 1..T:
3. Broadcast ()
4. For p in 1..P in parallel:
5. Gradp 🡨 0
6. For (x, y) in Dp:
7. Gradp 🡨 gradp + (y – xT)x
8. Collect(grad1,…,gradp)
9. 🡨 – ηp
10. Return

**Папулин:** 

# 32) Распределенные алгоритмы. Стохастический градиентный спуск.(copypaste)

Стаохостический градиентный спуск: 🡨 – η

Алгоритм (обычный) с мини-пакетами:

1. Initialize ()
2. For I in 1..T:
3. Shuffle(D)
4. Batches 🡨 split (D, bsize)
5. For B in batches:
6. Grad 🡨 0
7. For (x, y) in B:
8. Grad 🡨 grad + ∇l(y, x, )
9. 🡨 – η\*grad
10. Return

Распределенный стаохостический градиентный спуск: 🡨 – η

Алгоритм (распределенный) с мини-пакетами:

1. Initialize ()
2. For p in 1..T:
3. Broadcast ()
4. For p in 1..P in parallel:
5. Bp 🡨 sample (Dp, batchsize)
6. Gradp 🡨 0
7. For (x, y) in Bp:
8. Gradp 🡨 gradp + ∇l(y, x, )
9. Collect (grad1,…,gradp)
10. 🡨 – η\*
11. Return

**Папулин:** 

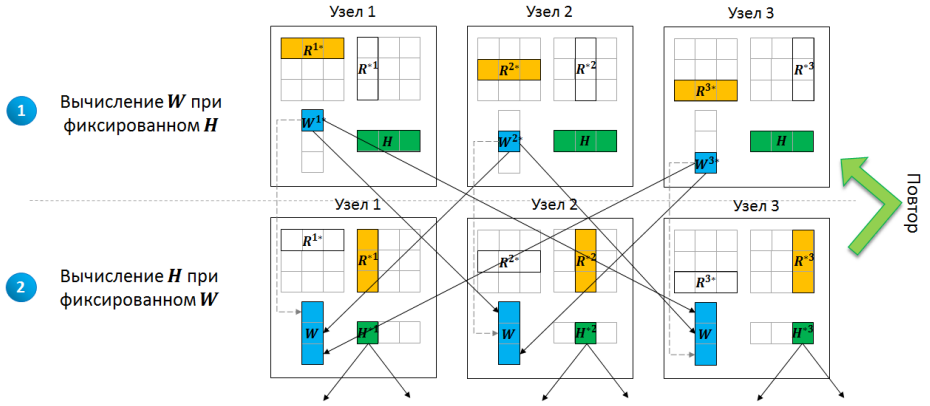
# 33) Распределенные алгоритмы. Факторизация матрицы (один из вариантов).(copypaste)

Алгоритм:

1. H 🡨 средний рейтинг для первой строки и малые случайные значения для остальных
2. Цикл: критерий остановки
   1. W 🡨 вычисление при фиксированном H распределено на d узлах
   2. Обновление W на всех узлах
   3. H 🡨 вычисление при фиксированном W распределено на d узлах
   4. Обновление H на всех узлах

= (-1, где n – это итерация

= (-1





——————————————————————————————————

Доп штуки

От папулина

2)

В экзаменационном вопросе про метрики классификации помимо Aсс, P, R, F1 необходимо понимать принципы построения ROC, как рассматривали на семинаре https://github.com/MLMethods/Practice/blob/master/notebooks/C6\_Metrics.ipynb

3)

Ниже ссылка с разъяснением стандартных ошибок при обучении моделей. В дз достаточно часто встречались подобные ошибки

https://scikit-learn.org/stable/common\_pitfalls.html

4)

При проверке дз я ошибочно полагал, что запись cv=4 при каждом запуске будет давать разное разбиение. Поэтому снижал в районе 0.2 балла. Не стал пересматривать все работы. Если на экзамене будет спорная ситуация с оценкой, то скажите, посмотрим, возможно, на 1-2 балла итоговая будет больше

