Ответы на экзаменационные вопросы по курсу

“Методы Машинного Обучения” (2015)

лекции читали Майсурадзе А. И. и Сенько О. В.

ответы составлены группой 521 в 2015 году

[Вопрос 1](#h.unir0qrjnn5y)

[Вопрос 2](#h.yrblb6mqjtuh)

[Вопрос 3](#h.hhr88my2sefm)

[Вопрос 4](#h.i2mtjb8t7obk)

[Вопрос 5](#h.g9mdlspsae49)

[Вопрос 6](#h.cyny0xmwbzwk)

[Вопрос 7](#h.8t623rrn87i3)

[Вопрос 9](#h.ezghicq8lhdx)

[Вопрос 10](#h.lddv1msqmnhh)

[Вопрос 11](#h.luoprsxsgqwx)

[Вопрос 12](#h.bdmte64550a5)

[Вопрос 13](#h.ub6n4dqi4cbn)

[Вопрос 14](#h.ye3eadtgo3qu)

[Вопрос 16](#h.wxlfte5w0k6e)

[Вопрос 17](#h.qs8cy5wj0tia)

[Вопрос 18](#h.bw2fzitlewpn)

[Вопрос 19](#h.6it36jbf5c6y)

[Вопрос 20](#h.rduryp9005n3)

[Вопрос 21](#h.tt54tifc224b)

[Вопрос 22](#h.tj15u6ssyi1)

[Вопрос 23](#h.74cvic80sq78)

[Вопрос 24](#h.7hifx5yamf0k)

[Вопрос 25](#h.lfbvmvrcnr7)

[Вопрос 26.](#h.ruxaifobpt7u)

[Вопрос 27](#h.uhilftamgpzg)

[Вопрос 29](#h.mjmq35tfhrs7)

[Вопрос 30](#h.n9gjdgomxycd)

[Вопрос 31](#h.5l6c6dogyuls)

# Вопрос 1

Место и роль ИАД в современной структуре человеческой деятельности. Три уровня технологий анализа данных, их назначение. Понятие о моделировании реального мира в науке. Физическая модель. Модель “решателя”. Информационная модель. Эвристическая модель. Основной приём ИАД для связи предметных областей и фундаментальной математики (обучение и эксплуатация эвристической информационной модели).

??

**Интеллектуальная задача** - задача обработки информации, возникающая в плохо формализованной прикладной области. Адекватные математические модели реальных объектов отсутствуют.

**ИАД - Интеллектуальный анализ данных** - раздел Анализа Данных (АД).

Три направления решения интеллектуальных задач:

* строгое математическое моделирование прикладной области. Обычно создание адекватной модели невозможно
* моделирование процесса мышления. Человек справляется - пытаемся моделировать его способ мышления. Персептроны. Экспертные системы.
* реализуем процесс преобразования информации. Несмотря на отсутствие модели того, как решает человек, несмотря на отсутствие адекватной математической модели реальной ситуации, опираясь на обычный здравый смысл

Физика — моделирование явления (пример классического моделирования)

Основная идея ИАД: Моделирование данных, а не явлений. т.к. это “информационное моделирование”

*Основной задачей физики является построение математической модели некоторой системы или явления. Модель позволяет прогнозировать развитие явления или управлять системой.*

*ИАД — моделирование данных (неклассическое моделирование) Если мы не можем создать физическую модель явления, можно попытаться моделировать данные.*

**Информационная модель** или **модель данных** – множество отображений, из которого выбираем ответ. Обычно модель – параметрическое семейство отображений: выбрать отображение – указать значения всех параметров.

**Эвристическая информационная модель** — это модель, описывающая данные. Она не опирается на законы природы, хотя и может базироваться на некоторых разумных предпосылках.

*Формализация — параметрические семейства алгоритмов.*

1. *линейный классификатор*
2. *байесовские модели*
3. *нейронные сети*
4. *метрические модели (ближайшие соседи, парзеновские окна, потенциальные функции)*
5. *логические модели (решающие списки, решающие деревья, тестовый подход)*
6. *АВО - алгоритмы вычисления оценок*
7. *...*

*Историческое развитие: решение отдельных задач (50-е) -> появление моделей, оформление семейств алгоритмов -> коллективы алгоритмов -> операции над алгоритмами, алгебры над алгоритмами*

*Необходимые технологии:*

*базы данных,*

*консолидация данных (сбор и объединение данных из разных исходных областей)*

*хранилища данных (warehouses),*

----------------------------------------------

*OLAP (online analytical processing),*

*системы отчётности,*

*-------------------------------------------------------*

*ИАД (Data Mining, Machine Learning, ...)*

*Сбор данных*

*----------------------*

*проверка гипотез*

*----------------------*

*генерация гипотез*

В машинном обучении часто мы не вольны выбирать выборку (например, мы не можем сказать "заразите мне ещё людей")

Методология анализа данных:

1. Бизнес (управление) (снижение себестоимости, повышение производитльности труда, сохранение старых клиентов, привлечение новых, увеличение рыночной доли, освоение новых рынков…)
2. Индустрия (инженеры, технологи) (видят готовые решения, но не всегда класс задач)
3. Отраслевая наука
4. Прикладная наука
5. Фундаментальная наука (топологическое пространство, аналитическая функция, асимптотика распределения, локальный оптимум, электронная плотность, третичная структура белка…)

На каждом уровне:

Постановка в интересах верхнего уровня.

Метод решения на основе знаний нижнего уровня.

Использовать классы задач, которые понимают представители соседних уровней.

=> конкретные задачи верхних уровней превращаются в каскад всё более абстрактных задач нижнего уровня.

Уровни задач:

1. Предметная область - содержательные задачи
2. Прикладная наука - инструменты статистики и анализа, фундаментальные задачи ИАД
3. Фундаментальная наука - задачи по дисциплинам

# Вопрос 2

*Основные модели данных (dataframe, multidimensional, similarity tensor, transactional). Гомогенные и гетерогенные модели. Фундаментальные задачи ИАД и основные инструменты статистики.*

Данные могут быть гомогенными - т.е. однородными, и гетерогенными, т.е. неоднородными.

Модели для задания исходных данных (в целом все они конвертируемы из одной в другую)

1. **Data Matrix (признаковое описание)** (матрица значений атрибутов каждого объекта) – объекты гомогенные (однотипные).

*(классификация (обучение + расознавание), восстановление регрессий)*

1. **Multi Dimensional**.

Матрица Кросс-сочетания. – это n-мерный куб, получаемый за счёт декартова произведения матриц с некоторыми атрибутами (например, <simpson vs not simpson x male vs female x child vs adult>)

В каждой ячейке этой матрицы находятся объекты, которые обладают соответствующими признаками.

*Показатель* – это функция от множества объектов из ячейки.

Большинство систем OLAP (Online Analytical Processing) – используют такое представление.

Детализация кросс-таблицы называется Drill Down.

Для показателей (значений ячеек), могут быть функции агрегирования (подсчёт показателя для объединённых ячеек).

Пример полу-агрегируемого показателя – «первый объект множества».

*(story telling, data visualization, explorative analysis, reporting systems, проверка гипотез)*

1. **Similarity Tensor (описание эталонами)**

Используются гетерогенные объекты (т.е. разного типа)

Таблица (аналитическое пространство (тензор)) задаёт взаимодействие между конкретными объектами попарно. (т.е. фактически нам даны оценки похожести разнотипных объектов)

Т.е. каждому объекту ставится в соответствие набор расстояний до других объектов (может быть всех, а может и не всех).

*(кластеризация по входу: би-, ко- класт., кластеризация по выходу: плоская, иерархическая, нечёткая, стохастическая, ранжирующая, )*

* + 1. *Flat partitioning (плоская кластеризация) (выходные объекты принадлежат только одному классу)*
    2. *Иерархическая кластеризация – строится бинарное дерево, листья которого – объекты*
    3. *Нечёткая кластеризация – (например, объект A принадлежит множеству головастиков на 1/10 долю) – именно долю, а не «с вероятностью»*
    4. *Стохастическая кластеризация – у кластера есть описание (бывает самым разным) + распределение вероятностей, что объект принадлежит этому кластеру.*

*Именно вероятность, а не доля. Поэтому для стохастической кластеризации нельзя говорить, что объект принадлежит всем кластерам, а для нечёткой – можно.*

* + 1. *Ранжирующая кластеризация (это не задача ранжирования)*

*Тут не выделяют конкретные различимые кластеры, надо просто, чтобы рядом стоящие объекты были похожи, а далекостоящие – не похожи, т.е. фактически объекты выстраиваются в цепочку.*

1. **Transactional Data / Формальный контекст** (бинарный признак) **/ Транзакционные данные**

Пусть у предметной области есть понятие «элемент» и «носитель».

И каждому носителю соответствует некоторое количество элементов (называемое «транзакция» - это множество (дубликаты отсутствуют))

Формальный контекст – это когда все признаки имеют значение типа bool.

Теоретически одним из носителей в множестве – может быть время

*(поиск популярных наборов, поиск ассоциативных правил, поиск последовательных ассоциативных правил)*

***Транзакционные данные****,* ***формальные контексты*** *- фиксировано множество элементов, транзакция - конечное подмножество элементов, наличие элемента в транзакции — бинарный признак*

**Фундаментальные задачи ИАД** - задачи, на которые принято проводить декомпозицию более сложных задач.

1. Задачи обучения с учителем:
   1. **Задачи классификации** - надо получить алгоритм, который может отнести произвольный объект к одному из заранее заданных классов
      1. Бинарная классификация (частный случай)
   2. **Задачи восстановления регрессии** - надо получить алгоритм (регрессию), который каждому объекту распознавания сопоставит некоторое значение из бесконечного, нерперывного множества
   3. **Задачи обучения по прецедентам** - классификатор или регрессия настраиваются по заданному конечному набору прецедентов - объектов с заранее известными правильными ответами.
   4. **Задачи прогнозирования** - обычно прогнозирование сводится к классификации или восстановлению регрессии, когда один из признаков определяет время.
   5. **Задачи последовательного обучения** - прецеденты приходят последовательно во времени (один за другим). Алгоритм постоянно донастраивается.
   6. **Архивирование, настройка модели источника** - символы на архивацию приходят последовательно один за другим. (ИАД здесь нужен потому, что нужно уметь предсказывать будущие символы, чтобы кодировать самые популярные наименьшим количеством бит)
2. Задачи обучения без учителя:
   1. **Кластеризация (Сегментация)** - надо разбить все множество объектов на непересекающиеся подмножества (кластеры, сегменты), в которых объекты в каком-либо смысле похожи друг на друга.
   2. **Нечеткая кластеризация**, **бикластеризация** и т.д.
   3. **Иерархическая кластеризация (Таксономия)** - надо построить дерево подмножеств, в котором каждый последующий слой является измельчением предыдущего.
3. Задачи с частичным обучением - кроме прецедентной информации имеется информация о том, что некоторый набор объектов действительно существует и будет использован в ходе решения прикладной задачи. То есть для настройки алгоритма можно использовать прецедентную информацию и информацию о существовании данных объектов. (например: база данных фотографий, ищем фотографии с лицами)
4. Выявление отклонений, детектирование:
   1. **Выявление ошибок в данных** - (“так не может быть”). Поступающая информация может содержать ошибки. (например: неисправность измерительного прибора)
   2. **Выявление нетипичного поведения** - (“так раньше не было”). Наша атомная электростанция раньше никогда не взрывалась. Мы не знаем, как выглядит станция, собирающаяся взорваться, но систему мониторинга создать должны.
   3. **Устранение отклонений из обучения (фильтрация)** - необходимо выявить те прецедентные данные, которые мешают качественно настроить модель.
5. Задачи восстановления пропусков - решение обычно сводится к задачам классификации или восстановления регрессии
   1. **Заполнение пропусков в прецедентах** - выбранный метод обучения модели требует, чтобы присутствовали все данные без пропусков.
   2. **Заполнение пропусков в описаниях распознаваемых объектов** - настроенная модель требует, чтобы во вновь приходящих на обработку описаниях не было пропусков.
6. Анализ наборов (не учитываем время транзакции) Термин: анализ рыночной корзины
   1. **Поиск популярных наборов** (например: чай и мёд часто покупают вместе)
   2. **Поиск ассоциативных правил** (например: те, кто купил мёд, часто покупают чай)
7. **Анализ последовательностей** (учитываем время)
   1. Поиск последовательных правил (например: если сегодня купил принтер, то через месяц купит картридж)
8. Анализ формальных понятий - формализация описания понятия в виде пары (объём, содержание) (!!! надо бы примеров)
   1. Поиск формальных понятий
   2. Построение и анализ решёток понятий

Основные инструменты статистики:

1. Оценка параметров
2. Проверка гипотиз
3. Дисперсионный анализ - установить наличие зависимости
4. Корреляционный анализ - установить и определить силу зависимости
5. Регрессионный анализ - установить конкретный вид зависимости
6. Дискреминантный анализ (по сути - классификация)

*Основные инструменты статистики:*

1. *описательная математическая статистика обобщает данные и их простейшие свойства, группирует их (сбор данных, категоризация, обобщение данных, представление данных)*
2. *индуктивная статистика позволяет выявлять закономерности в данных и делать из этого выводы (элементарные операции - дисперсия, ковариация)*

# Вопрос 3

*Распределение фундаментальных задач ИАД и основных инструментов статистики по моделям данных: в разрезе исходных данных, в разрезе результатов.*

(здесь не хватает обозначений, где «разрез исходных данных», а где «разрез результатов»)

Задачи математической статистики: (скорее всего это относится к инструментам статистики ??)

1. Оценка параметров
2. Проверка гипотиз
3. Анализ зависимостей
   1. Дисперсионный анализ (установить наличие зависимостей)
   2. Корреляционный анализ (установить и определить илу зависимости)
   3. Регрессионный анализ (установить конкретный вид зависимости)
4. Дискриминантный анализ (по сути классификация)
5. DM - Data Matrix
   1. Классификация = *объекты с признаками* -> обучение -> *информационная модель* -> распознавание -> *объекты с предсказанными признаками*
   2. Восстаноление регрессий
6. MD - Multi Dimensional
   1. Storytelling
   2. Data visualization
   3. Explorative analysis (поисковый анализ данных)
   4. Reporting systems (системы отчёта)
   5. Проверка гипотиз
7. ST - Similarity Tensor
   1. Входные кластеризации
      1. би- ко- кластеризации
   2. Выходные кластеризации
      1. Flat partitioning (плоская кластеризация) (выходные объекты принадлежат только одному классу)
      2. Иерархическая кластеризация – строится бинарное дерево, листья которого – объекты
      3. Нечёткая кластеризация – (например, объект A принадлежит множеству головастиков на 1/10 долю) – именно долю, а не «с вероятностью»
      4. Стохастическая кластеризация – у кластера есть описание (бывает самым разным) + распределение вероятностей, что объект принадлежит этому кластеру.

Именно вероятность, а не доля. Поэтому для стохастической кластеризации нельзя говорить, что объект принадлежит всем кластерам, а для нечёткой – можно.

* + 1. Ранжирующая кластеризация (это **не** задача ранжирования)

Тут не выделяют конкретные различимые кластеры, надо просто, чтобы рядом стоящие объекты были похожи, а далекостоящие – не похожи, т.е. фактически объекты выстраиваются в цепочку.

1. TD - Transactional Data
   1. Поиск популярных наборов (например, какие товары покупают вместе)
   2. Поиск ассоциативных правил
   3. Поиск последовательных ассоциативных правил

# Вопрос 4

***Модель данных «признаковое описание объектов». Понятие о шкалах значений атрибутов. Представление реляционными технологиями. Схемы «звезда» и «снежинка».***

**Признаковой описание объектов**

Source:<http://bit.ly/1N7BtxT> – machinelearning.ru

Признаковое описание объекта — это вектор, составленный из значений фиксированного набора признаков на данном объекте. Признаки в общем случае могут иметь различные типы, причём не обязательно числовые.

Пусть X — множество объектов (ситуаций, прецедентов). Что есть объект, определяется спецификой предметной области.

Признаком (feature) называется результат измерения некоторой характеристики объекта. Формально, признак — это отображение f: X→Df, где Df — множество допустимых значений признака. В зависимости от природы этого множества признаки делятся на следующие типы:

бинарный признак: Df = {0, 1};

номинальный признак: Df — конечное множество;

порядковый признак: Df — конечное упорядоченное множество;

количественный признак: Df = R (действ. числа).

В прикладных задачах встречаются и более сложные случаи. Значениями признаков могут быть числовые последовательности, изображения, тексты, функции, графы, результаты запросов к базе данных, и т. д.

Если все признаки имеют одинаковый тип, то исходные данные называются однородными, в противном случае — разнородными.

Пусть имеется набор признаков f1,...,fn. Вектор (f1(x),...,fn(x)) называется признаковым описанием объекта x<-X. В машинном обучении не делается различия между объектом и его признаковым описанием; полагается, что X = Df1 \* … \* Dfn (декартово произведение).

**Шкалы значений атрибутов**

Числовые признаки/атрибуты могут быть упорядочены в отличие от категориальных (нет в типах выше, но означает номинальный признак с акцентом на неупорядоченность). Т.к. многие алгоритмы так или иначе используют близость пары наборов признаков, важной является задача о выборе шкалы для значений числовых признаков. (Скорее всего стоит говорить о том, что несбалансированные шкалы могут влиять на характеристики, вычисляемые над признаками).

**Представление реляционными технологиями. Схемы “звезда” и “снежинка”**

Реляционная модель данных (РМД) — логическая модель данных, прикладная теория построения баз данных, которая является приложением к задачам обработки данных таких разделов математики, как теория множеств и логика первого порядка. Кроме того, в состав реляционной модели данных включают теорию нормализации. Термин «реляционный» означает, что теория основана на математическом понятии отношение (relation).

OLAP (англ. *online analytical processing*, аналитическая обработка в реальном времени) — технология обработки данных, заключающаяся в подготовке суммарной (агрегированной) информации на основе больших массивов данных, структурированных по многомерному принципу.

Причина использования OLAP для обработки запросов — скорость. Реляционные БД хранят сущности в отдельных таблицах, которые обычно хорошо нормализованы. Эта структура удобна для операционных БД (системы OLTP), но сложные многотабличные запросы в ней выполняются относительно медленно.

OLAP-структура, созданная из рабочих данных, называется OLAP-куб. Куб создаётся из соединения таблиц с применением схемы звезды или схемы снежинки. В центре схемы звезды находится таблица фактов, которая содержит ключевые факты, по которым делаются запросы. Множественные таблицы с измерениями присоединены к таблице фактов. Отличием схемы снежинки является то, что таблицы измерений нормализованы с рядом других связанных измерительных таблиц, — в то время как в схеме звезды таблицы измерений полностью денормализованы, с каждым измерением представленным в виде единой таблицы, без соединений на связанные таблицы в схеме снежинки. Чем больше степень нормализации таблиц измерений, тем сложнее выглядит структура схемы снежинки. Создаваемый «эффект снежинки» затрагивает только таблицы измерений, и не применим к таблицам фактов.

Таблицы измерений показывают, как могут анализироваться агрегированные реляционные данные. Количество возможных агрегирований определяется количеством способов, которыми первоначальные данные могут быть иерархически отображены.

# 

# 

# Вопрос 5

Линейная регрессия:

Распространённым средством решения задач прогнозирования непрерывной величины *Y* по переменным *X1,...,Xn* является использование метода множественной линейной регрессии. В данном методе связь переменной *Y* с переменными *X1,..., Xn* задаётся с помощью линейной модели

*Y =β0 +β1X1 +...+βnXn +ε,*

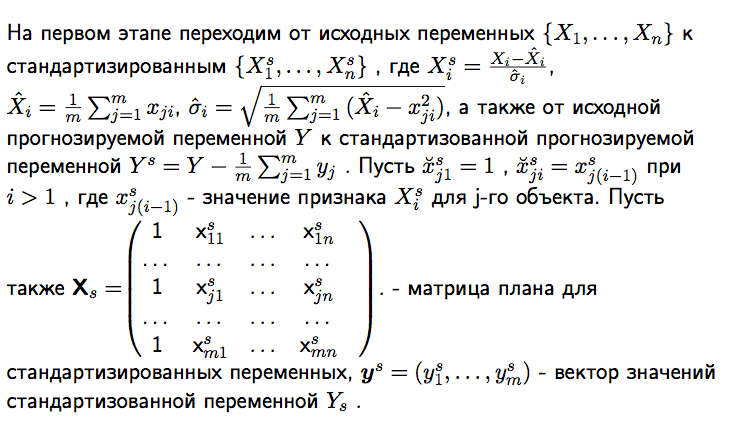
где *β0,β1,...,βn* - вещественные регрессионные коэффициенты, *ε* - случайная величина, являющаяся ошибкой прогнозирования. Регрессионные коэффициенты ищутся по обучающей выборке

*S ̃t = {s1 = (y1,x1),...,sm = (ym,xm)}*,

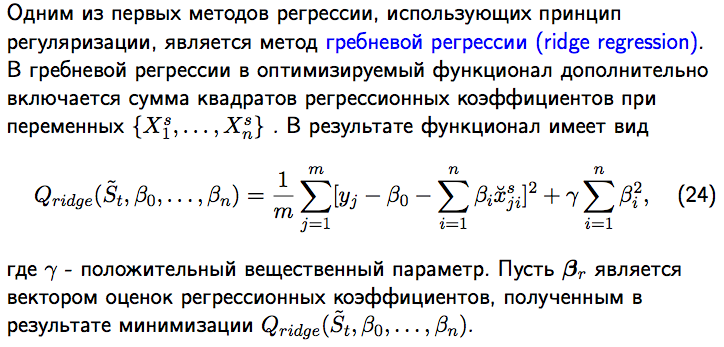
где *xj =(xj1,...,xjn)* вектор значений переменных *X1,...,Xn* для объекта *sj.*

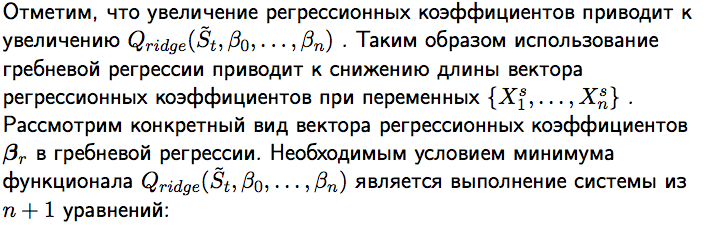
Одним из возможных способов борьбы с неустойчивостью является использование методов, основанных на включение в исходный оптимизируемый функционал Q(S ̃t,β0, β1,...,βn) дополнительной штрафной компоненты. Введение такой компоненты позволяет получить решение, на котором Q(S ̃t,β0,β1,...,βn) достаточно близок к своему глобальному минимуму. Однако данное решение оказывается значительно более устойчивым и благодаря устойчивости позволяет достигать существенно более высокой обобщающей способности. Подход к получению более эффективных решений с помощью включения штрафного слагаемого в оптимизируемый функционал принято называть регуляризацией по Тихонову.

**Первый Этап:**

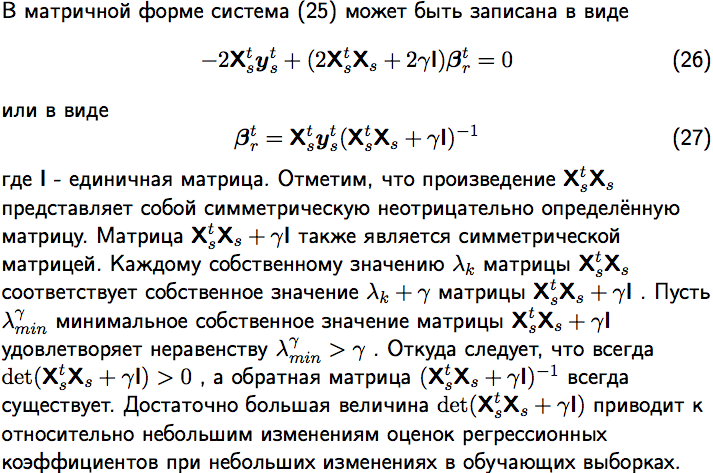


**Гребневая регрессия:**

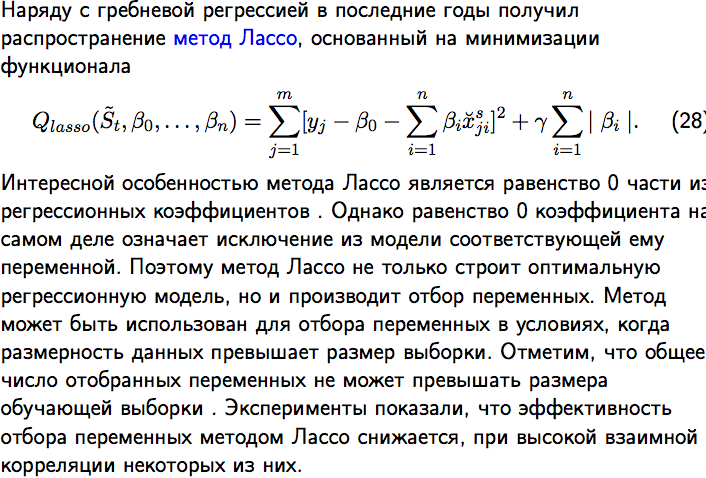








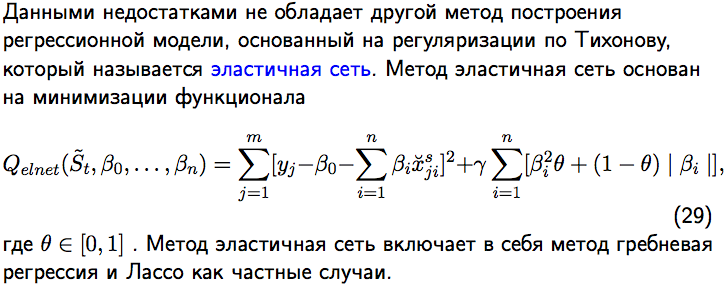
**Лассо**



Немного дополнительной инфы про метод гребневую регрессию и лассо:

<http://www.machinelearning.ru/wiki/images/5/52/Strijov-Krymova10Model-Selection.pdf>

**Эластичная сеть**



# Вопрос 6

Транзакционная модель данных. Связанные с ней задачи.

Транзакционная модель представления данных описывает событие или транзакцию, которое имело место быть. При этом существует некоторое ограниченное множество элементов в системе и вхождение или нет конкретного элемента в конкретную транзакцию определяется бинарным признаком (0 входит, 1 нет).

Обычно, транзакции присуще различных покупкам, документообороту и прочей бизнес ереси.

Пример нескольких транзакций (а точнее, нескольких транзакционных записей, в нашем случае описывающих единовременные покупки):

1. Клавиатура, мышь, очки;
2. Мышь, ноутбук, внешний HDD.
3. …

Здесь возникает несколько задач анализа данных, которые могут искать:

1. Зависимости в транзакциях между некоторыми элементами множества объектов (что покупали и с чем);
2. Какие группы элементов множества были популярны;
3. Без чего жизнь боль (без какого продукта россияне не будут жить. правильный ответ без водки, но это нужно проАНАЛизировать из транзакционных данных, снятых с кассовых аппаратов по всей России);
4. …

Конкретно из **формальных задач**, которые описывал Арчил (указаны в Билете 2):

1. Анализ наборов (не учитываем время транзакции) Термин: анализ рыночной корзины
   1. Поиск популярных наборов (например: чай и мёд часто покупают вместе)
   2. Поиск ассоциативных правил (например: те, кто купил мёд, часто покупают чай)
2. Анализ последовательностей (учитываем время)
   1. Поиск последовательных правил (например: если сегодня купил принтер, то через месяц купит картридж)

Вообще говоря, многие (том числе на [stackoverflow](http://stackoverflow.com/questions/18884728/difference-between-transactional-and-non-transactional)) говорят о том, что транзакционные данные нужны для систем [OLTP](https://ru.wikipedia.org/wiki/OLTP)).

**OLTP** (Online Transaction Processing), транзакционная система — обработка транзакций в реальном времени. Способ организации БД, при котором система работает с небольшими по размерам транзакциями, но идущими большим потоком, и при этом клиенту требуется от системы минимальное время отклика.

Термин OLTP применяют также к системам (приложениям). OLTP-системы предназначены для ввода, структурированного хранения и обработки информации (операций, документов) в режиме реального времени.

OLTP-приложениями охватывается широкий спектр задач во многих отраслях — автоматизированные банковские системы, ERP-системы (системы планирования ресурсов предприятия), банковские и биржевые операции, в промышленности — регистрация прохождения детали на конвейере, фиксация в статистике посещений очередного посетителя веб-сайта, автоматизация бухгалтерского, складского учёта и учёта документов и т. п. Приложения OLTP, как правило, автоматизируют структурированные, повторяющиеся задачи обработки данных, такие как ввод заказов и банковские транзакции. OLTP-системы проектируются, настраиваются и оптимизируются для выполнения максимального количества транзакций за короткие промежутки времени. Как правило, большой гибкости здесь не требуется, и чаще всего используется фиксированный набор надёжных и безопасных методов ввода, модификации, удаления данных и выпуска оперативной отчётности. Показателем эффективности является количество транзакций, выполняемых за секунду. Обычно аналитические возможности OLTP-систем сильно ограничены (либо вообще отсутствуют).

Разница между транзакционными данными и не транзакционными — четкое время. У первых оно есть (то бишь это нечто, произошедшее в определенный момент), а у других нет четкого времени (к примеру адрес это не транзакционные данные, так как ты не секунду живешь тут).

# Вопрос 7

Общая задача классификации. Понятие об обучении и использовании. Объект, модель, алгоритм-классификатор. /\**Универсальные ограничения. Локальные ограничения.*\*/ Оптимизационный подход.

Постановка задачи классификации:

1. Каждому объекту сопоставляется унифицированное описание (description) (определяется признаковое описание объекта). Например: признаковое описание объекта (feature-based description).
2. Каждому объекту сопоставляется значение целевого признака. Для задачи классификации это множество значений - дискретно.
3. Требуется найти отображение (алгоритм), переводящее описание объекта в значение целевого признака и действующее на генеральной совокупности объектов.

Объект (то же, что и описание объекта) - это то, что поступит на вход отображения (классификатора).

Информационная модель (модель данных) – множество отображений, из которого выбираем ответ. Обычно модель – параметрическое семейство отображений: выбрать отображение – указать значения всех параметров.

Обучение (настройка алгоритма) - это процесс выбора конкретного отображения из модели. Соответственно, метод обучения (алгоритм) - это процедура, которая выбирает конкретное отображение.

/\*

*Метод обучения зависит от некоторой дополнительной информации – частичное описание отображения, ограничения универсальные или локальные. Это лучше не писать, если не знаешь, что это за ограничения. У Майсурадзе есть один пример универсального ограничения: с увеличением роста вес должен расти. Больше ничего*

\*/

Для одной и той же модели и одной и той же информации можно предложить много разных методов обучения, которые могут дать разные ответы.

Процесс обучения должен назначить значения параметрам модели.

Взять эти значения «с потолка» – один из методов обучения, не требует дополнительной информации.

Оптимизационный подход:

1. Ввести понятие качества отображения
2. Указать модель
3. Выбрать из модели отображение с наилучшим качеством

Оптимизационный подход сводит задачу обучения к задаче оптимизации качества модели.

# Вопрос 9

Понятие байесовского классификатора как оптимального алгоритма распознавания.

Основное понятие, которое хочет услышать Арчил в данном вопросе -- Байесовский подход к классификации основан на теореме, утверждающей, что если плотности распределения каждого из классов известны, то искомый алгоритм можно выписать в явном аналитическом виде. Более того, этот алгоритм оптимален, то есть обладает минимальной вероятностью ошибок.

Далее материал (который мне показался подходящим) с [machinelearning](http://www.machinelearning.ru/wiki/index.php?title=Байесовский_классификатор).

Пусть X — множество описаний объектов, Y — множество номеров (или наименований) классов. На множестве пар «объект, класс» X \times Y определена вероятностная мера \mathsf P. Имеется конечная [обучающая выборка](http://www.machinelearning.ru/wiki/index.php?title=%D0%92%D1%8B%D0%B1%D0%BE%D1%80%D0%BA%D0%B0) независимых наблюдений X^m = \{(x_1,y_1),\ldots,(x_m,y_m)\}, полученных согласно вероятностной мере \mathsf P.

Задача классификации заключается в том, чтобы построить алгоритм a:\; X\to Y, способный классифицировать произвольный объект x \in X.

В байесовской теории классификации эта задача разделяется на две:

1. **Построение оптимального классификатора при известных плотностях классов**. Эта подзадача имеет простое и окончательное решение.
2. **Восстановление плотностей классов по обучающей выборке**. В этой подзадаче сосредоточена основная сложность байесовского подхода к классификации.

**Построение классификатора при известных плотностях классов**

Пусть для каждого класса y \in Y известна априорная вероятность P_y того, что появится объект класса y, и плотности распределения p_y(x) каждого из классов, называемые также функциями правдоподобия классов. Требуется построить алгоритм классификации a(x), доставляющий минимальное значение функционалу среднего риска.

Средний риск определяется как математическое ожидание ошибки:

R(a) = \sum_{y\in Y} \sum_{s\in Y} \lambda_{y} P_y \mathsf{P}_{(x,y)}\bigl\{a(x)=s|y\bigr\},

где \lambda_{y} — цена ошибки или штраф за отнесение объекта класса y к какому-либо другому классу.

**Теорема**. Решением этой задачи является алгоритм a(x) = \mathrm{arg}\max_{y\in Y} \lambda_{y} P_y p_y(x).

Значение P\{y|x\} = P_y p_y(x) интерпретируется как апостериорная вероятность того, что объект x принадлежит классу y.

Если классы равнозначимы, \lambda_{y} P_y = \mathrm{const}(y), то объект x просто относится к классу с наибольшим значением плотности распределения в точке x.

**Восстановление плотностей классов по обучающей выборке**

По заданной подвыборке объектов класса y построить эмпирические оценки априорных вероятностей P_y и функций правдоподобия p_y(x).

В качестве оценки априорных вероятностей берут, как правило, долю объектов данного класса в обучающей выборке.

**Восстановление плотностей** (функций правдоподобия каждого из классов) является наиболее **трудной задачей**. Наиболее распространены три подхода: *параметрический*, *непараметрический* и *разделение смеси вероятностных распределений*. Третий подход занимает промежуточное положение между первыми двумя, и в определённом смысле является наиболее общим.

* **Параметрическое** восстановление плотности при дополнительном предположении, что [плотности нормальные (гауссовские)](http://www.machinelearning.ru/wiki/index.php?title=%D0%9C%D0%BD%D0%BE%D0%B3%D0%BE%D0%BC%D0%B5%D1%80%D0%BD%D0%BE%D0%B5_%D0%BD%D0%BE%D1%80%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D1%80%D0%B0%D1%81%D0%BF%D1%80%D0%B5%D0%B4%D0%B5%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5), приводит к [нормальному дискриминантному анализу](http://www.machinelearning.ru/wiki/index.php?title=%D0%9D%D0%BE%D1%80%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%B4%D0%B8%D1%81%D0%BA%D1%80%D0%B8%D0%BC%D0%B8%D0%BD%D0%B0%D0%BD%D1%82%D0%BD%D1%8B%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7&action=edit) и [линейному дискриминанту Фишера](http://www.machinelearning.ru/wiki/index.php?title=%D0%9B%D0%B8%D0%BD%D0%B5%D0%B9%D0%BD%D1%8B%D0%B9_%D0%B4%D0%B8%D1%81%D0%BA%D1%80%D0%B8%D0%BC%D0%B8%D0%BD%D0%B0%D0%BD%D1%82_%D0%A4%D0%B8%D1%88%D0%B5%D1%80%D0%B0).
* **Непараметрическое** восстановление плотности приводит, в частности, к [методу парзеновского окна](http://www.machinelearning.ru/wiki/index.php?title=%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%BF%D0%B0%D1%80%D0%B7%D0%B5%D0%BD%D0%BE%D0%B2%D1%81%D0%BA%D0%BE%D0%B3%D0%BE_%D0%BE%D0%BA%D0%BD%D0%B0).
* [Разделение смеси распределений](http://www.machinelearning.ru/wiki/index.php?title=%D0%A0%D0%B0%D0%B7%D0%B4%D0%B5%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D1%81%D0%BC%D0%B5%D1%81%D0%B8_%D1%80%D0%B0%D1%81%D0%BF%D1%80%D0%B5%D0%B4%D0%B5%D0%BB%D0%B5%D0%BD%D0%B8%D0%B9&action=edit) может быть сделано с помощью [EM-алгоритма](http://www.machinelearning.ru/wiki/index.php?title=EM-%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC). Дополнительное предположение, что плотности компонент смеси являются радиальными функциями, приводит к [методу радиальных базисных функций](http://www.machinelearning.ru/wiki/index.php?title=%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D1%80%D0%B0%D0%B4%D0%B8%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D1%85_%D0%B1%D0%B0%D0%B7%D0%B8%D1%81%D0%BD%D1%8B%D1%85_%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B9&action=edit). Обычно в качестве компонент смеси берут, опять-таки, гауссовские плотности.

Таким образом, формула байесовского классификатора приводит к большому разнообразию байесовских алгоритмов, отличающихся только способом восстановления плотностей.

[***Наивный байесовский классификатор***](http://www.machinelearning.ru/wiki/index.php?title=%D0%9D%D0%B0%D0%B8%D0%B2%D0%BD%D1%8B%D0%B9_%D0%B1%D0%B0%D0%B9%D0%B5%D1%81%D0%BE%D0%B2%D1%81%D0%BA%D0%B8%D0%B9_%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%82%D0%BE%D1%80) (naїve Bayes) основан на той же формуле и дополнительном предположении, что объекты описываются независимыми признаками. Предположение о независимости существенно упрощает задачу, так как оценить n одномерных плотностей гораздо легче, чем одну n-мерную плотность. (благодаря этому условная вероятность распадается на произведение условных вероятностей отдельно по каждому из признаков) (точность около 0,67, т.е. так себе).

**Наивный байесовский классификатор** может быть как параметрическим, так и непараметрическим, в зависимости от того, каким методом восстанавливаются одномерные плотности.

Объекты x\in Xописываются n статистически независимыми признаками:

x \equiv \bigl( \xi_1,\ldots,\xi_n\bigr) \equiv \bigl( f_1(x),\ldots,f_n(x) \bigr).

Предположение о независимости означает, что **функции правдоподобия классов** представимы в виде

p_y(x) = p_{y1}(\xi_1) \cdot \ldots \cdot p_{yn}(\xi_n),

где p_{yj}(\xi_j) — плотность распределения значений j-го признака для класса y.

# Вопрос 10

Классификаторы, основанные на использовании формулы Байеса. Линейный дискриминант Фишера.

Про классификаторы на основе формулы Байеса см. вопрос 11.

Дикриминант Фишера на machinelearning.ru: [goo.gl/uvK4Rx](http://goo.gl/uvK4Rx)

[Из лекций Сенько]

Линейный дискриминант Фишера (ЛДФ) для распознавания двух классов *K1* и *K2* .

В основе метода лежит поиск в многомерном признаковом пространстве такого направления *w*, чтобы средние значения проекции на него объектов обучающей выборки из классов *K1* и *K2* максимально различались. Проекцией произвольного вектора *x* на направление *w* является отношение 

В качестве меры различий проекций классов на *w* используется функционал



где - среднее значение проекции векторов переменных *X1*, … ,*Xn*, описывающих объекты из класса *Ki.*

* выборочная дисперсия проекций векторов, описывающих объекты из класса *Ki*, *i* {1, 2}.

Смысл функционала ясен из его структуры. Он является по сути квадратом отличия между средними значениями проекций классов на направление *w* нормированным на сумму внутриклассовых выборочных дисперсий. Можно показать, что Φ(w, St ) достигает максимума при



где Таким образом оценка направления для оптимального

распознавания K1 и K2 может быть выражена в этом виде.

Распознавание нового объекта *s∗* по векторному описанию *x∗* производится по величине его проекции на направление *w*:

При этом используется простое пороговое правило: при *γ(x∗ ) >*  объект *s∗* относится к классу *K1* и *s∗* относится к классу *K2* в противном случае.

Граничный параметр подбирается по обучающей выборке таким образом, чтобы проекции объектов разных классов на оптимальное направление *w* оказались бы максимально разделёнными. Простой, но эффективной, стратегией является выбор в качестве порогового параметра средней проекции объектов обучающей выборки на направление *w*. Метод ЛДФ легко обобщается на случай с несколькими классами. При

этом исходная задача распознавания классов *K1* , . . . , *KL* сводится к последовательности задач с двумя классами *K1’* и *K2’*: *Ki и ,* для каждой из которых ищется оптимальное направление. Распознаваемый объект относится к классу, соответствующему максимальной величине проекции.

Эвристика линейного дискриминанта Фишера является в некотором роде упрощением квадратичного дискриминанта. Она используется с целью получить более устойчивый алгоритм классификации. Наиболее целесообразно пользоваться линейным дискриминантом Фишера, когда данных для обучения недостаточно. Вследствие основной гипотезы, на которой базируется алгоритм, наиболее удачно им решаются простые задачи классификации, в которых по формам классы "похожи" друг на друга.

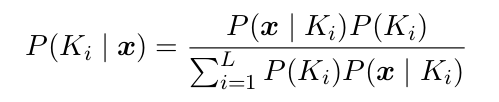
# Вопрос 11

Классификаторы, основанные на использовании формулы Байеса. Логистическая регрессия.

[из слайдов Сенько, 88-99, 102-106]

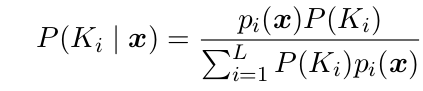
Байесовские методы обучения основаны на аппроксимации условных вероятностей классов в точках признакового пространства с использованием формулы Байеса (учитывая утверждение, что максимальную точность распознавания обеспечивает байесовское решающее правило, относящее распознаваемый объект, описываемый вектором x переменных (признаков) X1, . . . , Xn к классу K∗ , для которого условная вероятность P(K∗ | x) максимальна).

Рассмотрим задачу распознавания классов K1 , . . . , KL . Условные вероятности классов в точке признакового пространства могут быть рассчитаны с использованием формулы Байеса. В случае, если переменные X1, . . . , Xn являются дискретными, формула Байеса может быть записана в виде:



где P(K1), . . . , P(KL) - вероятность классов K1, . . . , KL безотносительно к признаковым описаниям (априорная вероятность).

В случае, если переменные X1, . . . , Xn являются непрерывными, формула Байеса может быть записана с ледующим образом:



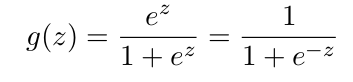
где p1(x), . . . , pL(x) - значения плотностей вероятностей классов K1, . . . , KL в пространстве Rn .

Примеры классификаторов, основанных на использовании формулы Байеса:

* Наивный байесовский классификатор
* Линейный дискриминант Фишера
* Логистическая регрессия

**Логистическая регрессия** - метод построения линейного классификатора, позволяющий оценивать апостериорные вероятности принадлежности объектов классам.

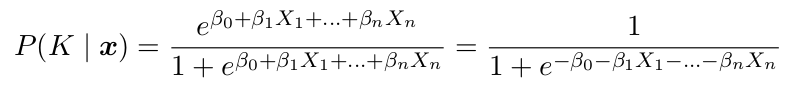
Целью логистической регрессии является аппроксимация плотности условных вероятностей классов в точках признакового пространства. При этом аппроксимация производится с использованием логистической функции:



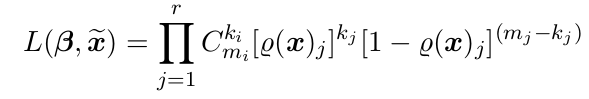
В методе логистическая регрессия связь условной вероятности класса K с прогностическими признаками осуществляется через переменную Z , которая задаётся как линейная комбинация признаков:

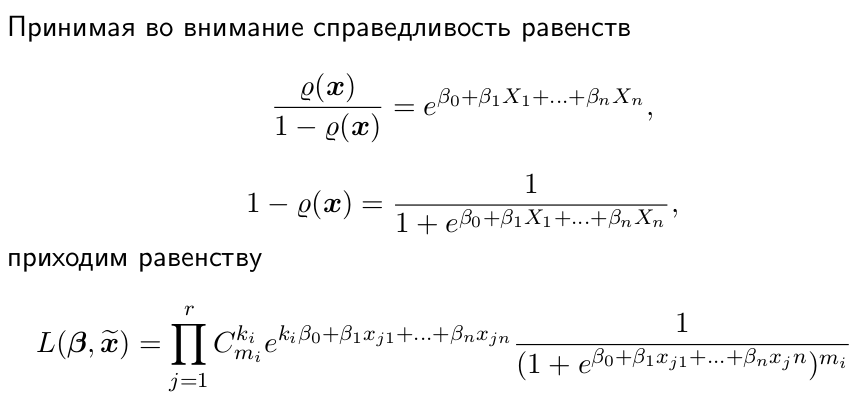
z = β0 + β1\*X1 + . . . + βn\*Xn .

Условная вероятность K в точке векторного пространства x∗ = (x1∗ , . . . , xn∗) задаётся в виде

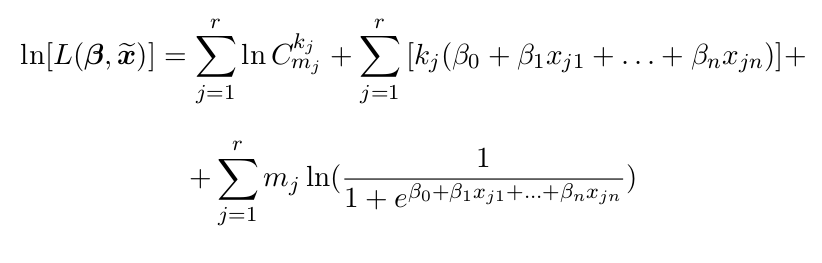


Оценки регрессионных параметров β0 , β1 , . . . , βn могут быть вычислены по обучающей выборке с помощью различных вариантов метода максимального правдоподобия. Предположим, что объекты обучающей выборки сосредоточены в точках признакового пространства из множества x = {x1 , . . . , xr } . При этом распределение объектов обучающей выборка по точкам задаётся с помощью набора пар {(m1 , k1 ), . . . , (mr , kr)}, где mi - общее число объектов в точке xi , ki - число объектов класса K в точке xi. Вероятность данной конфигурации подчиняется распределению Бернулли. Введём обозначение = P(K|x) . Оценка вектора регрессионных параметров β = (β0 , . . . , βn ) может быть получена с помощью метода максимального правдоподобия. Функция правдоподобия может быть записана в виде:





Поиск оптимального значения параметров удобнее производить, решая задачу максимизации логарифма функции правдоподобия, который в нашем случае принимает вид:



Дополнительные ресурсы:

<http://www.machinelearning.ru/wiki/index.php?title=%D0%9B%D0%BE%D0%B3%D0%B8%D1%81%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%80%D0%B5%D0%B3%D1%80%D0%B5%D1%81%D1%81%D0%B8%D1%8F>

<https://ru.wikipedia.org/wiki/%D0%9B%D0%BE%D0%B3%D0%B8%D1%81%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%80%D0%B5%D0%B3%D1%80%D0%B5%D1%81%D1%81%D0%B8%D1%8F>

# Вопрос 12

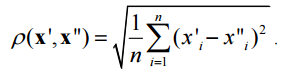
Метод k-ближайших соседей.

Используется для решения задач распознавания.

Основным принципом **метода ближайших соседей** является то, что объект присваивается тому классу, который является наиболее распространённым среди соседей данного элемента.

Оценка условных вероятностей ведётся по ближайшей окрестности точки x, содержащей k признаковых описаний объектов обучающей выборки.

В качестве оценки за класс выступает отношение , где - число признаковых описаний объектов обучающей выборки из внутри .

Окрестность задаётся с помощью функции расстояния заданной на декартовом произведении , где - область допустимых значений признаковых описаний. В качестве функции расстояния может быть использована стандартная эвклидова метрика

Для задач с бинарными признаками в качестве функции расстояния может быть использована метрика Хэмминга, равная числу совпадающих позиций в двух сравниваемых признаковых описаниях.

Окрестность ищется путём поиска в обучающей выборке векторных описаний, ближайших в смысле выбранной функции расстояний, к описанию распознаваемого объекта s\*.

Единственным параметром, который может быть использован для настройки (обучения) алгоритмов в методе *k*–ближайших соседей является собственно само число ближайших соседей.

Для оптимизации параметра *k* обычно используется метод, основанный на скользящем контроле. Оценка точности распознавания производится по обучающей выборке при различных и выбирается значение данного параметра, при котором полученная точность максимальна.

# Вопрос 13

Есть машинное представление целых чисел и чисел с плавающей точкой.

Текстовые файлы бывают human readable и нет.

Форматы хранения табличных данных:

**comma separated values** - допускается, что разделитель может встречаться внутри полей, поэтому нужен механизм quotes (возможность заключения в кавычки (для защиты от проблемы с использованием кавычек с самих полях ведётся подсчёт чётности справа и слева от запятой (вообще говоря справа, потому что lookahead быстрее))) (регулярными выражениями легче разбирается, чем экранированием)

Пример:

1996,Jeep,Grand Cherokee,"MUST SELL! air, moon roof, loaded",4799.00

отображается как

1996 | Jeep | Grand Cherokee | MUST SELL! air, moon roof, loaded | 4799.00

**tab delimited values** - подразумевается, что разделитетль (обычно это табуляция) в полях не встречается

С помощью экранирования можно распознавать служебные символы в качестве обычных. Например,если нужно в регулярном выражении использовать символ “$”, то для этого нужно записать /$, где символ “/” - экран.

**Диаграммы для многомерной модели данных** - то что в Excel.

**Системы отчётноси** - ПО, которое позволяют создавать (а так же менять переходить от одного к другому) и визуализировать аналитические пространства.

Диаграмма дополняет аналитическое пространство (она может вводить на категориях дополнительные пордяки) (например, как расставить метки на осях) + она добавляет виды кодирования (категории могут кодироваться размером/цветом/формой/положением)

На диаграмме отдельные объекты - на предпоследней строчке.

Диаграммы для наборов точек из Rn - например

***Кодировки:***

Абстрактному символу (абстрактному понятию в человеческой голове) ставится в соответствие некоторый код.

Например это делает кодировка unicode или ascii.

Дальше код символа некоторым образом записывается в файл, при этом код может быть преобразован в какое-то другое число.

Например это делает кодировка utf-8 или utf-16, ...

Особенности ascii - она разбита на первые 128 символов, которые зафиксированны, и на слеующие 128 символов, которые в разной стране могут быть какими угодно.

Особенности unicode - это большая несколько-байтная кодировка, которая побита на куски, и каждый кусок под что-то отведён, есть кусок для китайцев, есть кусок для русских, …

Для номеров с U+0000 по U+007F кодировка UTF-8 полностью соответствует 7-битному [US-ASCII](https://ru.wikipedia.org/wiki/ASCII) c 0 в старшем бите и занимает один байт.

Алгоритм UTF-8 технически позволяет записывать код любой длины. Но для эффективной и надёжной работы алгоритма необходимо ограничение длины кода.

Порядок *от старшего к младшему* или ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *big-endian*, дословно: «тупоконечный»): A_n,\dots,A_0, запись начинается со старшего и заканчивается младшим. Этот порядок является стандартным для протоколов [TCP/IP](https://ru.wikipedia.org/wiki/TCP/IP), он используется в заголовках пакетов данных и во многих протоколах более высокого уровня, разработанных для использования поверх TCP/IP. Поэтому, порядок байтов от старшего к младшему часто называют сетевым порядком байтов

Порядок *от младшего к старшему* или ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *little-endian*, дословно: «остроконечный», о происхождении термина [ниже](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%80%D1%8F%D0%B4%D0%BE%D0%BA_%D0%B1%D0%B0%D0%B9%D1%82%D0%BE%D0%B2#.D0.98.D0.BD.D1.82.D0.B5.D1.80.D0.B5.D1.81.D0.BD.D1.8B.D0.B5_.D1.84.D0.B0.D0.BA.D1.82.D1.8B)): A_0,\dots,A_n, запись начинается с младшего и заканчивается старшим. Этот порядок записи принят в памяти персональных компьютеров с [x86](https://ru.wikipedia.org/wiki/X86)-процессорами, в связи с чем иногда его называют [*интеловский*](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%BB) *порядок байт*

*Смешанный порядок байтов (*[*англ.*](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *middle-endian) иногда используется при работе с числами,* [*длина которых превышает машинное слово*](https://ru.wikipedia.org/wiki/%D0%94%D0%BB%D0%B8%D0%BD%D0%BD%D0%B0%D1%8F_%D0%B0%D1%80%D0%B8%D1%84%D0%BC%D0%B5%D1%82%D0%B8%D0%BA%D0%B0)*. Число представляется последовательностью* [*машинных слов*](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5_%D1%81%D0%BB%D0%BE%D0%B2%D0%BE)*, которые записываются в формате, естественном для данной архитектуры, но сами слова следуют в обратном порядке.*

Некоторые кодировки в качестве первых байт записывают значение, которое фиксирует little или big, а некоторые их не пользуют, потому что всегда работают лишь в одном режиме.

**Схема «звезды»**, схема звёздного соединения, звездоподобная схема, звёздная схема (от [англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *star schema*) — специальная организация [реляционных таблиц](https://ru.wikipedia.org/w/index.php?title=%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%82%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0&action=edit&redlink=1), удобная для хранения многомерных показателей. Лежит в основе реляционного [OLAP](https://ru.wikipedia.org/wiki/OLAP).

Модель данных состоит из двух типов таблиц: одной [таблицы фактов](https://ru.wikipedia.org/wiki/%D0%A2%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0_%D1%84%D0%B0%D0%BA%D1%82%D0%BE%D0%B2) (*fact table*) — центр «звезды» — и нескольких [таблиц измерений](https://ru.wikipedia.org/wiki/%D0%A2%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0_%D0%B8%D0%B7%D0%BC%D0%B5%D1%80%D0%B5%D0%BD%D0%B8%D0%B9) (*dimension table*) по числу измерений в модели данных — лучи «звезды».

Таблица фактов обычно содержит одну или несколько колонок типа DECIMAL, дающих числовую характеристику какому-то аспекту предметной области (например, объём продаж для торговой компании или сумма платежей для банка), и несколько целочисленных колонок-ключей для доступа к таблицам измерений.

Таблицы измерений расшифровывают ключи, на которые ссылается таблица фактов; например, таблица «products» измерения «товары» базы данных торговой компании может содержать сведения о названии товара, его производителе, типе товара. За счёт использования специальной структуры таблицы измерений реализуется иерархия измерений, в том числе ветвящаяся.

Обычно данные в таблицах-измерениях денормализованы: ценой несколько неэффективного использования дискового пространства удается уменьшить число участвующих в операции соединения таблиц, что обычно приводит к сильному уменьшению времени выполнения запроса. Иногда, тем не менее, требуется произвести [нормализацию](https://ru.wikipedia.org/wiki/%D0%9D%D0%BE%D1%80%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%84%D0%BE%D1%80%D0%BC%D0%B0#.D0.9D.D0.BE.D1.80.D0.BC.D0.B0.D0.BB.D0.B8.D0.B7.D0.B0.D1.86.D0.B8.D1.8F_.D0.B1.D0.B0.D0.B7_.D0.B4.D0.B0.D0.BD.D0.BD.D1.8B.D1.85) таблиц-измерений; такая схема носит название «снежинка»

**Схема снежинки** получила свое название за свою форму, в виде которой отображается логическая схема таблиц в многомерной базе данных. Так же как и в [схеме звезды](https://ru.wikipedia.org/wiki/%D0%A1%D1%85%D0%B5%D0%BC%D0%B0_%D0%B7%D0%B2%D0%B5%D0%B7%D0%B4%D1%8B), схема снежинки представлена централизованной [таблицей фактов](https://ru.wikipedia.org/wiki/%D0%A2%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0_%D1%84%D0%B0%D0%BA%D1%82%D0%BE%D0%B2), соединенной с [таблицами измерений](https://ru.wikipedia.org/wiki/%D0%A2%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0_%D0%B8%D0%B7%D0%BC%D0%B5%D1%80%D0%B5%D0%BD%D0%B8%D0%B9). Отличием является то, что здесь таблицы измерений нормализованы с рядом других связанных измерительных таблиц, — в то время как в схеме звезды таблицы измерений полностью денормализованы, с каждым измерением представленным в виде единой таблицы, без соединений на связанные таблицы в схеме снежинки. Чем больше степень нормализации таблиц измерений, тем сложнее выглядит структура схемы снежинки. Создаваемый «эффект снежинки» затрагивает только таблицы измерений, и не применим к таблицам фактов.

Схема снежинки, также как и схема звезды, наиболее часто встречается в таких [хранилищах данных](https://ru.wikipedia.org/wiki/%D0%A5%D1%80%D0%B0%D0%BD%D0%B8%D0%BB%D0%B8%D1%89%D0%B5_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), для которых скорость получения данных более важна, чем эффективность их манипуляции. Следовательно, таблицы должны быть нормализованы в малой степени, и зачастую разрабатываются с применением не выше [третьего уровня нормализации](https://ru.wikipedia.org/wiki/%D0%A2%D1%80%D0%B5%D1%82%D1%8C%D1%8F_%D0%BD%D0%BE%D1%80%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%84%D0%BE%D1%80%D0%BC%D0%B0).

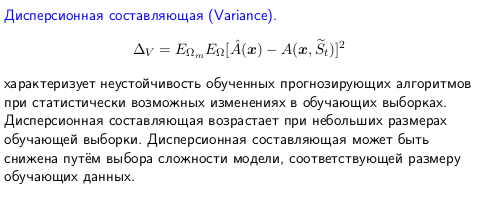
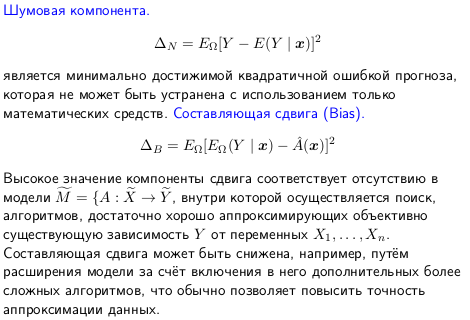
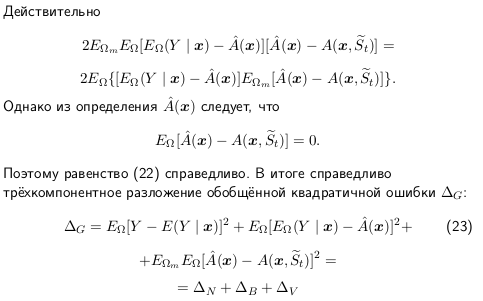
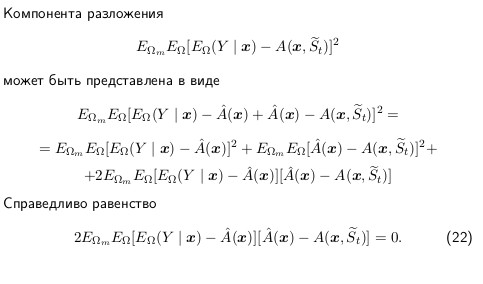
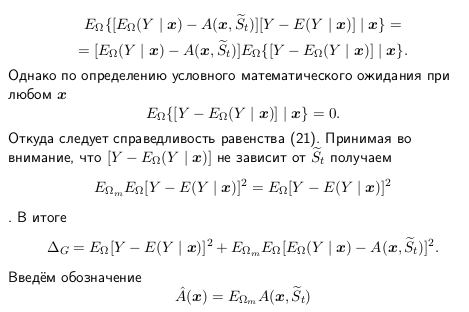
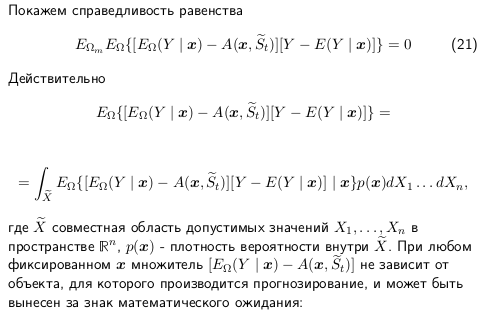
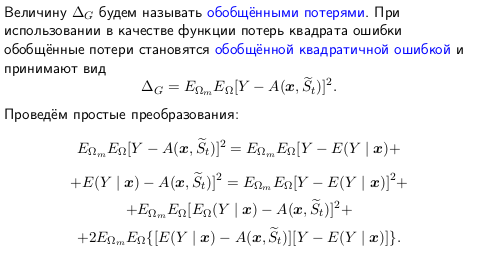
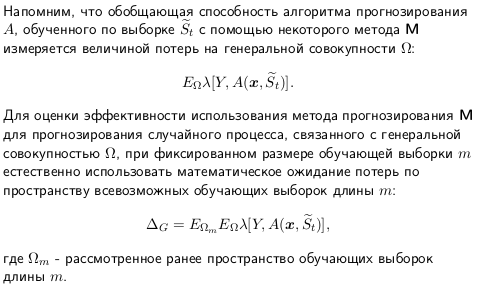
# Вопрос 14

См. вопрос 13 (~ с пятого абзаца)

# Вопрос 16

*Трёхкомпонентное разложение ошибки регрессионных моделей.*

Сенько (all in one) - 69-77 стр.



Таким образом существует Bias-Variance дилемма Составляющая сдвига может быть снижена путём увеличения разнообразия модели. Однако увеличение разнообразия модели при недостаточном объёме обучающих данных ведёт к росту компоненты сдвига. Наиболее высокая точность прогноза достигается, при поддержании правильного баланса между разнообразием используемой модели и объёмом обучающих данных

# Вопрос 17

Многомерная модель - **Multi Dimensinal**

Информация в **многомерной модели** *представляется в виде многомерных массивов, называемых гиперкубами*. В одной базе данных, построенной на многомерной модели, может храниться множество таких кубов, на основе которых можно проводить совместный анализ показателей. Конечный пользователь в качестве внешней модели данных получает для анализа определенные срезы или проекции кубов, представляемые в виде обычных двумерных таблиц или графиков.

В статистике, экономике, и смежных областях, многомерный анализ представляет собой процесс анализа данных, который группирует данные в двух или более категорий: размеры и данных измерений. Например, набор данных, состоящий из числа выигрышей для одной футбольной команды в каждый год на протяжении нескольких лет является одномерным (в данном случае, продольным) набор данных. Набор данных, состоящий из числа выигрышей для нескольких футбольных команд в течение одного года также одномерный (в данном случае, в поперечном сечении) набор данных. Набор данных, состоящий из ряда побед для нескольких футбольных команд в течение нескольких лет является двумерным набор данных.

Во многих дисциплинах, двумерные массивы данных также называют панельные данные. В то время как, строго говоря, два и более многомерные наборы данных - "многомерные", термин "многомерные", как правило, применяется только к данным наборы с тремя или более измерениях. Например, некоторые наборы данных обеспечивают прогноз прогнозы на нескольких целевых периодов. Они проводятся несколькими синоптиками. Три измерения позволяют предоставить более подробную информацию, чем можно почерпнуть из двумерных наборов данных панели.

Агрегирование данных (data aggregation): процесс сбора, обработки и представления информации в окончательном виде. Агрегирование данных в основном выполняется для формирования отчетов, выработки политики, управления здравоохранением, научных исследований, статистического анализа и изучения здоровья населения

Про многомерные данные лучше всего расказано:

<http://habrahabr.ru/post/126810/>

**Куб**

Возьмем для примера таблицу Invoices1, которая содержит заказы фирмы. Поля в данной таблице будут следующие:

* Дата Заказа
* Страна
* Город
* Название заказчика
* Компания-доставщик
* Название товара
* Количество товара
* Сумма заказа

Какие агрегатные данные мы можем получить на основе этого представления? Обычно это ответы на вопросы типа:

* Какова суммарная стоимость заказов, сделанных клиентами из определенной страны?
* Какова суммарная стоимость заказов, сделанных клиентами из определенной страны и доставленных определенной компанией?
* Какова суммарная стоимость заказов, сделанных клиентами из определенной страны в заданном году и доставленных определенной компанией?

Все эти данные можно получить из этой таблицы вполне очевидными SQL-запросами с группировкой.

Результатом этого запроса всегда будет столбец чисел и список атрибутов его описывающих (например, страна) – это одномерный набор данных или, говоря математическим языком, – вектор.

Представим себе, что нам надо получить информацию по суммарной стоимости заказов из всех стран и их распределение по компаниям доставщиков – мы получим уже таблицу (матрицу) из чисел, где в заголовках колонок будут перечислены доставщики, в заголовках строк – страны, а в ячейках будет сумма заказов. Это – двумерный массив данных. Такой набор данных называется сводной таблицей (pivot table) или кросс-таблицей.

Если же нам захочется получить те же данные, но еще в разрезе годов, тогда появится еще одно изменение, т.е. набор данных станет трехмерным (условным тензором 3-го порядка или 3-х мерным «кубом»).

Очевидно, что максимальное количество измерений – это количество всех атрибутов (Дата, Страна, Заказчик и т.д.), описывающих наши агрегируемые данные (сумму заказов, количество товаров и т.п).

Так мы приходим к понятию многомерности и его воплощению – многомерному кубу. Такая таблица будет у нас называться «таблицей фактов». Измерения или Оси куба (dimensions) – это атрибуты, координаты которых – выражаются индивидуальными значениями этих атрибутов, присутствующих в таблице фактов. Т.е. например, если информация о заказах велась в системе с 2003 по 2010 год, то эта ось годов будет состоять из 8 соответствующих точек. Если заказы приходят из трех стран, то ось стран будет содержать 3 точки и т.д. Независимо от того, сколько стран заложено в справочнике Стран. Точки на оси называются ее «членами» (Members).

Сами агрегируемые данные в данном случае буду назваться «мерами» (Measure). Чтобы избежать путаницы с «измерениями», последние предпочтительней называть «осями». Набор мер образует еще одну ось «Меры» (Measures). В ней столько членов (точек), сколько мер (агрегируемых столбцов) в таблице фактов.

Члены измерений или осей могут быть объединены одной или несколькими иерархиями (hierarchy). Что такое иерархия, поясним на примере: города из заказов могут быть объединены в районы, районы в области, области страны, страны в континенты или другие образования. Т.е. налицо иерархическая структура – континент-*страна-область-район-город* – 5 уровней (Level). Для района данные агрегируются по всем городам, которые в него входят. Для области по всем районам, которые содержат все города и т.п. Зачем нужно несколько иерархий? Например, по оси с датой заказа мы можем хотеть группировать точки (т.е. дни) по иерархии *Год-Месяц-День* или по *Год-Неделя-День*: в обоих случаях по три уровня. Очевидно, что Неделя и Месяц по-разному группируют дни. Бывают также иерархии, количество уровней в которых не детерминировано и зависит от данных. Например, папки на компьютерном диске.

Агрегация данных может происходить с использованием нескольких стандартных функций: сумма, минимум, максимум, среднее, количество.

Drill – это детализация отчета с помощью уменьшения степени агрегации данных, совмещенное с фильтрацией по какой-нибудь другой оси (или нескольким осям). Сверление бывает нескольких видов:

* drill-down – фильтрация по одной из исходных осей отчета с выводом детальной информации по потомкам в рамках иерархии выбранного фильтрующего члена. Например, если имеется отчет по распределению заказов в разрезе Стран и Годов, то при щелчке на 2007-м году выведется отчет в разрезе тех же Стран и месяцев 2007 года.
* drill-aside – фильтрация под одной или нескольким выбранным осям и снятие агрегации по одной или нескольким другим осям. Например, если имеется отчет по распределению заказов в разрезе Стран и Годов, то при щелчке на 2007-м году выведется другой отчет в разрезе, например, Стран и Поставщиков с фильтрацией по 2007 году.
* drill-trough – снятие агрегации по всем осям и одновременная фильтрация по ним же – позволяет увидеть исходные данные из таблицы фактов, из которых получено значение в отчете. Т.е. при щелчке по значению ячейки выводится отчет со всеми заказами, которые дали эту сумму. Эдакое мгновенное бурение в самые «недра» куба.

# Вопрос 18

Постановка задачи кластеризации.

Пусть X — множество объектов, Y — множество номеров (имён, меток) кластеров. Задана функция расстояния между объектами \rho(x,x'). Имеется конечная обучающая выборка объектов X^m = \{ x_1, \dots, x_m \} \subset X. Требуется разбить выборку на непересекающиеся подмножества, называемые *кластерами*, так, чтобы каждый кластер состоял из объектов, близких по метрике \rho, а объекты разных кластеров существенно отличались. При этом каждому объекту x_i\in X^m приписывается номер кластера y_i.

*Алгоритм кластеризации* — это функция a:\, X\to Y, которая любому объекту x\in X ставит в соответствие номер кластера y\in Y. Множество Y в некоторых случаях известно заранее, однако чаще ставится задача определить оптимальное число кластеров, с точки зрения того или иного *критерия качества* кластеризации.

Кластеризация ([обучение без учителя](http://www.machinelearning.ru/wiki/index.php?title=%D0%9E%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5_%D0%B1%D0%B5%D0%B7_%D1%83%D1%87%D0%B8%D1%82%D0%B5%D0%BB%D1%8F)) отличается от [классификации](http://www.machinelearning.ru/wiki/index.php?title=%D0%9A%D0%BB%D0%B0%D1%81%D1%81%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F) ([обучения с учителем](http://www.machinelearning.ru/wiki/index.php?title=%D0%9E%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5_%D1%81_%D1%83%D1%87%D0%B8%D1%82%D0%B5%D0%BB%D0%B5%D0%BC)) тем, что метки исходных объектов y_i изначально не заданы, и даже может быть неизвестно само множество Y.

Решение задачи кластеризации принципиально неоднозначно, и тому есть несколько причин:

* Не существует однозначно наилучшего критерия качества кластеризации. Известен целый ряд эвристических критериев, а также ряд алгоритмов, не имеющих чётко выраженного критерия, но осуществляющих достаточно разумную кластеризацию «по построению». Все они могут давать разные результаты.
* Число кластеров, как правило, неизвестно заранее и устанавливается в соответствии с некоторым субъективным критерием.
* Результат кластеризации существенно зависит от метрики, выбор которой, как правило, также субъективен и определяется экспертом.

Плоская кластеризация не подразумевает вложенности кластеров, в то время как иерархическая допускает вложенность.

Результатом иерархической кластеризации является дерево, корнем которого будет вся выборка, а листьями наиболее мелкие кластеры. На каждом этапе решается задача плоской кластеризации. Обычно критерий останова (когда уже не нужно объединять соседние кластеры) не формализован а подбирается экспериментально - либо уже достигнуто число кластеров, либо эксперт посчитал, что результат уже нормальный.

Метод k средних является алгоритмом плоской кластеризации.

Нечеткие методы кластеризации допускают принадлежность объекта сразу к нескольким классам с разной степенью схожести. Четкие методы допускают принадлежность только одному кластеру, что не всегда справедливо для объектов близко к границам кластера.

# Вопрос 19

Плоская кластеризация. Задача и метод k-means.

Решаем задачу кластеризации объектов из обучающей выборки *X = {x1, x2, … , xm}* (“обучение без учителя”). Предположим, что обычная евклидова метрика на имеющихся признаках будет достаточно разумно описывать “похожесть” объектов (на практике все признаки изначально, естественно, надо центрировать и нормировать).

М[#heading=h.yrblb6mqjtuh](https://docs.google.com/document/d/1vmzS-rtEfsPm65qv__RlIe26qmiXZJLWOwhsTeMve6Q/edit#heading=h.yrblb6mqjtuh)ы хотим разбить имеющиеся объекты на *K* кластеров (сгустков) (*K* -- параметр алгоритма). Каждый кластер *{s1, s2, …, sK}* будем задавать с помощью специально введённого элемента из этого же евклидова пространства -- центроида (*mi* -- центр кластера *yi*). В результате кластеризации каждый объект будем относить к одному из кластеров

s(xi) -> {1, …, K} кластеру, соответствующему ближайшему центроиду:

Такая логика далеко не всегда оказывается адекватной. [Здесь](http://www.machinelearning.ru/wiki/images/2/28/Voron-ML-Clustering-slides.pdf) в районе 7-го слайда есть картиночки, объясняющие почему универсальной кластеризации не бывает.

Мы хотим найти:

1. Оптимальную разметку объектов по кластерам с точки зрения некоторого функционала (см. ниже)
2. Положения центроидов *mi*

В самом-самом классическом подходе минимизируется функционал:



(то есть сумма квадратов отклонений объектов от центров кластеров).

Вообще говоря можно ставить задачу для самых разных метрик, но в этом году Майсурадзе так не считает.

**А теперь главный прикол билета:**

В общем виде минимизировать функционал (функция от положения центроидов и разметки) указанный выше никто не умеет, зато:

1. Если центроиды фиксированы, то разметка находится тривиально

*s(xi) = argmin\_{mj} ||xi - mj||*

(каждый объект припишем к ближайшему кластеру)

1. Если фиксирована разметка, то легко можно найти оптимальное положение центроидов:

mj = 1/|Sj| \* sum\_{xi \in Sj|} xi

(ставим центроид в среднее арифметическоее соответствующих объектов)

Доказано, что этот процесс к чему-нибудь будет сходиться (блочно-покоординатный спуск) и даже за конечное число итераций. Но ясно, что часто можем попадать в локальные оптимумы и нужно запускаться из разных случайных начальных приближений и в конце брать наилучший результат.

Несмотря на большое число проблем, метод часто работает хорошо (по крайней мере для первичного анализа данных).

Из недостатков: нужно самим выбирать K ну и страдать от топологических проблем расположения объектов (кривые кластеры, не похожие на сферы или сферы, но с оочень разными внутрикластерными расстояниями).

[пишите в свои вопросы, ну ребят. хотя не, го флуд тут, в 19ом]

========== (wikipedia + http://www.machinelearning.ru/wiki/images/2/28/Voron-ML-Clustering-slides.pdf)

k-means. Формальная постановка задачи:

. d - размерность признака. n - количество примеров. Необходимо разбить элементы множества на кластеров так, что сумма квадратичных отклонений от центров кластеров будет минимальной. Формально:

\underset{\mathbf{S}} {\operatorname{arg\,min}}  \sum_{i=1}^{k} \sum_{\mathbf x \in S_i} \left\| \mathbf x - \boldsymbol\mu_i \right\|^2 

Поставленная задача будет рассматриваться в евклидовой метрике. Решение оптимизационной задачи:

Предположим, что уже известно некоторое приближение центроидов кластеров. Итерационно применяются следующие преобразования:

1. (E step) вычисление новых меток элементов выборки. Для каждого примера вычисляется расстояние до всех текущих центроидов кластеров и выбирается ближайший кластер. Номер кластера будет новой меткой этого примера.
2. (M step) Вычисляются новые центроиды кластеров на основе полученных меток

Алгоритм завершается либо в случае отсутствия изменений на последней итерации, либо по достижении фиксированного числа итераций. Начальные центры кластеров могут быть заданы случайно (случайные примеры из выборки).

Проблемы метода - сходимость локальному минимуму. Решение зависит от выбора начального приближения. Форма кластеров зависит от метрики.

# Вопрос 20csf_forel_w.png

Последовательная плоская кластеризация. Метод ФОРЕЛЬ.

[тут нужна форель] ->

Ещё один способ построить кластеризацию, но на этот раз уже точно для произвольной метрики (например, расстояние Левенштейна над объектами-строками):



Логика: последовательно добавляем в пространство объектов “сферу” радиуса R и с центром в одном из неразмеченных элементов (R -- параметр) и дальше подгоняемм положение “центра” так, чтобы отхватить побольше объектов. После этого удаляем все объекты, попавшие в добавленную сферу-кластер и создаём новую сферу.

Чуть более формально:

1. Случайно выбираем текущий объект из выборки
2. Помечаем объекты выборки, находящиеся на расстоянии менее, чем R от текущего
3. Вычисляем их центр тяжести, помечаем этот центр как новый текущий объект
4. Повторяем шаги 2-3, пока новый текущий объект не совпадет с прежним
5. Помечаем объекты внутри сферы радиуса R вокруг текущего объекта как кластеризованные, выкидываем их из выборки
6. Повторяем шаги 1-5, пока не будет кластеризована вся выборка

Несколько замечаний с [machinelearning.ru](http://www.machinelearning.ru/wiki/index.php?title=%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%A4%D0%9E%D0%A0%D0%95%D0%9B%D0%AC):

* Доказана сходимость алгоритма за конечное число шагов
* В линейном прстранстве центром тяжести может выступать произвольная точка пространства, в метрическом - только объект выборки
* Чем меньше R, тем больше таксонов (кластеров)
* В линейном пространстве поиск центра происходит за время О(n), в метрическом O(n²)
* Наилучших результатов алгоритм достигает на выборках с хорошим выполнением условий компактности
* При повторении итераций возможно уменьшение параметра R, для скорейшей сходимости
* Кластеризация сильно зависит от начального приближения (выбора объекта на первом шаге)
* Рекомендуется повторная прогонка алгоритма для исключения ситуации "плохой" кластеризации, по причине неудачного выбора начальных объектов

Можно отметить, что если потом нам приспичит заниматься не плоской, а иерархической кластеризацией, то можно взять нагенерированные центры сфер в роли новых “гиперобъектов” и теперь кластеризовывать их с бОльшим значением R’.

http://www.machinelearning.ru/wiki/index.php?title=%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC\_%D0%A4%D0%9E%D0%A0%D0%95%D0%9B%D0%AC

# Вопрос 21

Иерархическая кластеризация. Дивизивная. Агломеративная, функционалы связи (linkage).

Кластеризация - задача классификации объектов одной природы в несколько групп так, чтобы объекты в одной группе обладали одним и тем же свойством.

Под свойством подразумевается близость друг к другу относительно выбранной метрики.

Иерархическая кластеризация подразумевает наличие дерева вложенных кластеров: последовательно строим кластеры из ранее полученных кластеров.

Для того, чтобы осуществить иерархическую кластеризацию необходимо сначала задать расстояние

между произвольными кластерами

Способы задания расстояния:

1. расстоянием между двумя кластерами является минимальное расстояние между двумя объектами, один из которых принадлежит , а второй
2. расстоянием между двумя кластерами является максимальное расстояние между двумя объектами, один из которых принадлежит , а второй
3. расстояние между центрами кластеров
4. среднее расстояние между объектами из двух кластеров

В случае, когда все кластеры состоят только из одного объекта, расстояния между ними всегда равны расстояниям между этими единственными объектами.

На начальном этапе кластерами являются объекты

На каждом последующем шаге происходит объединение двух ближайших кластеров из набора, образованного на предыдущем шаге.

Процесс завершается при достижении одного из следующих условий:

1. Кластеры, образованные на новом шаге теряют компактность. Тогда мы оставляем в силе кластеризацию, полученную на предыдущем шаге.
2. Образуется требуемое число кластеров
3. Процесс завершается, если достигнутая кластеризация удовлетворяет требованиям эксперта исследователя.

Классическими подходами к построению иерархической кластеризации являются агломеративные и дивизивные алгоритмы кластеризации.

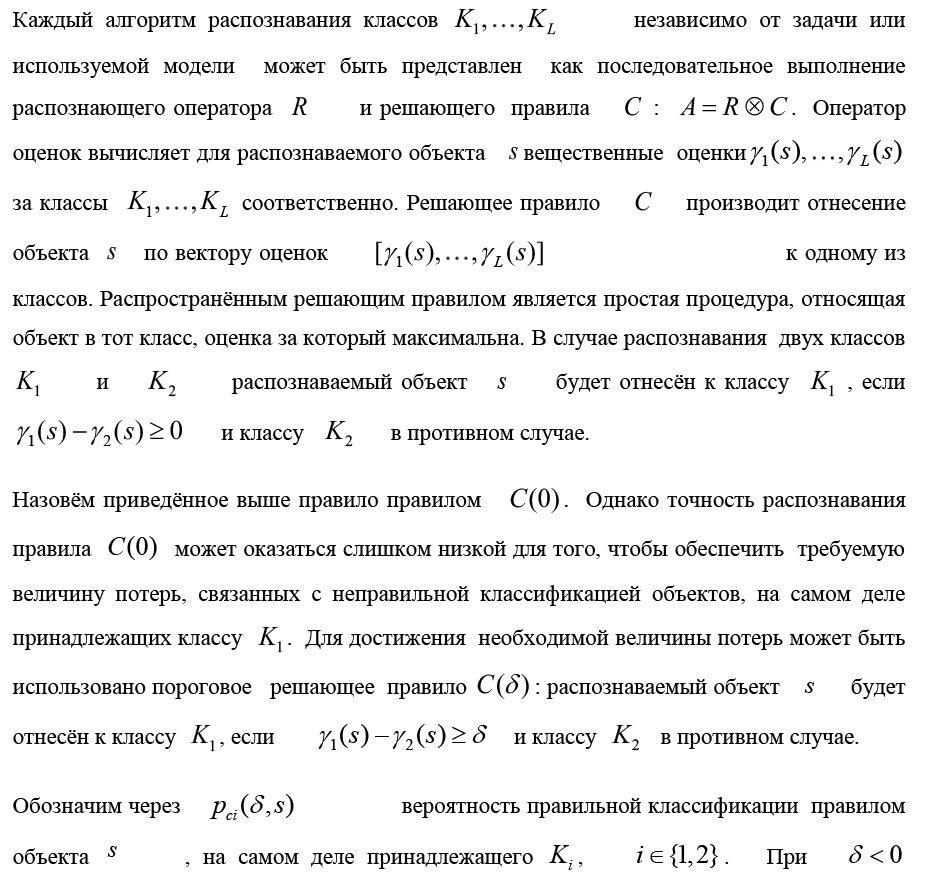
При построении иерархической кластеризации с помощью ***агломеративных*** алгоритмов объекты постепенно объединяются во всё более крупные кластеры. Таким образом из конфигурации, когда каждый объект является отдельным кластером, получается один кластер, содержащий все объекты. При использовании ***дивизивных*** алгоритмов же, наоборот, из более крупных кластеров получаются более мелкие. При этом из одного кластера, содержащего все объекты выборки, получаются кластеры из отдельных объектов.

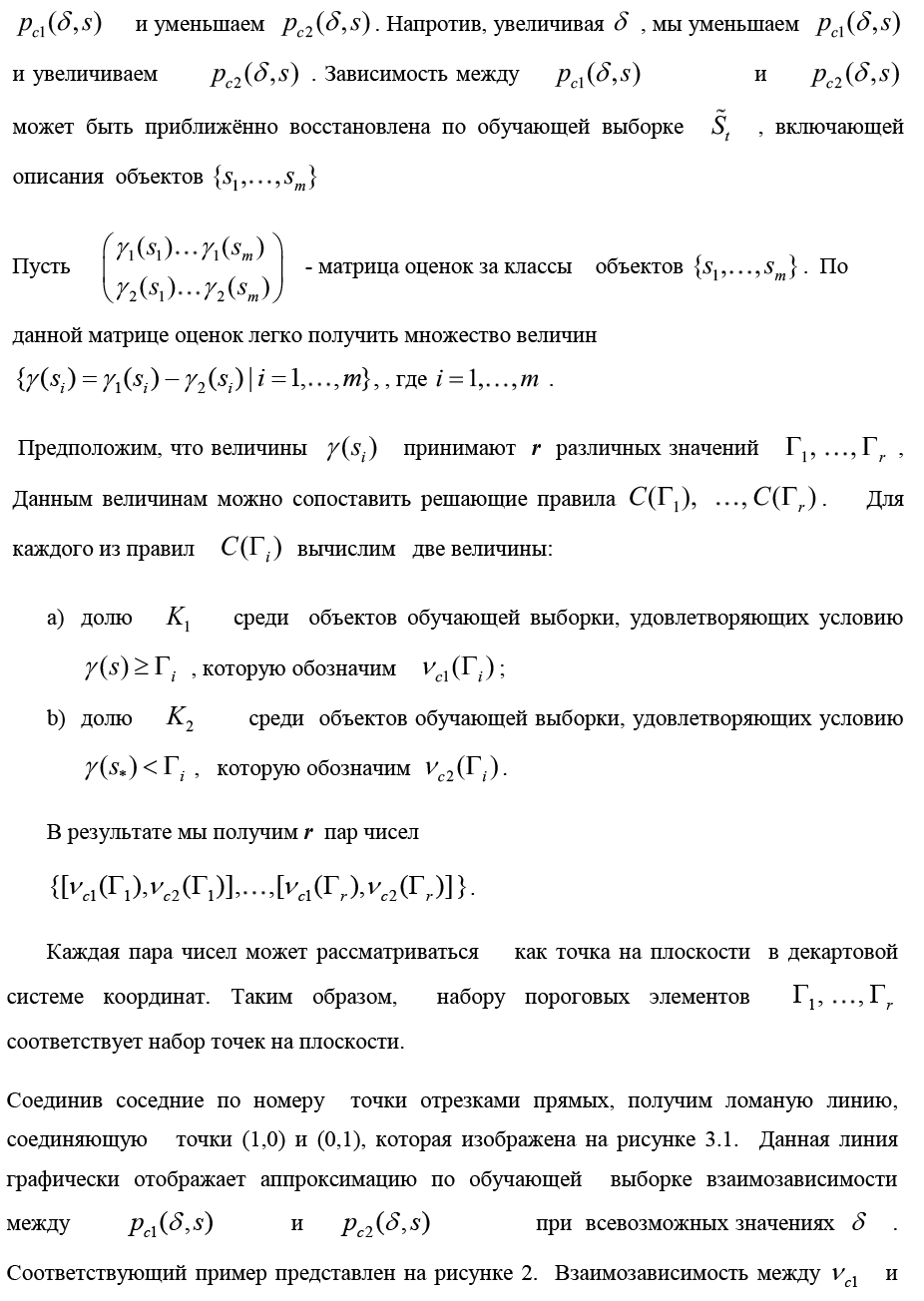
# Вопрос 22

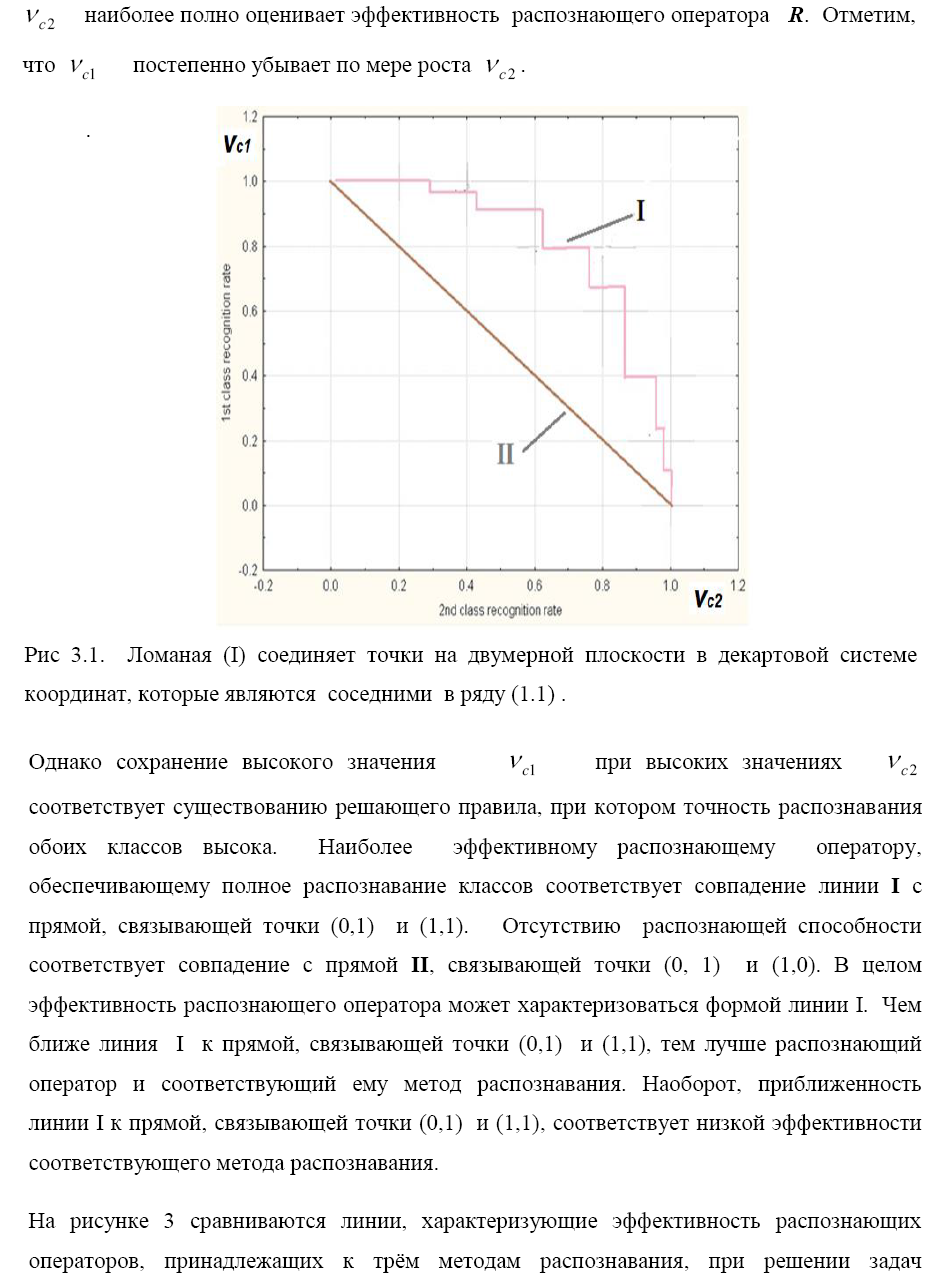
***ROC анализ***

<http://habrahabr.ru/post/228963/>

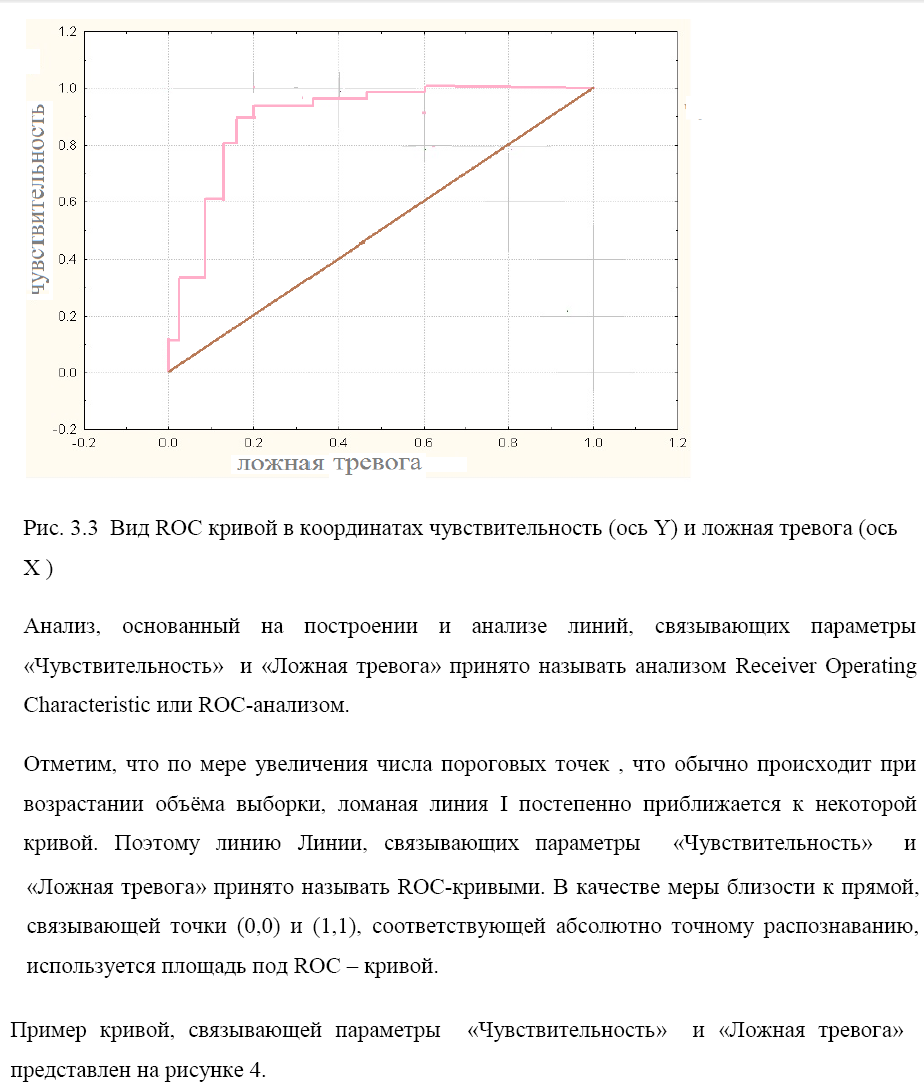
<https://alexanderdyakonov.wordpress.com/2015/10/09/%D0%B7%D0%B0%D0%B4%D0%B0%D1%87%D0%BA%D0%B8-%D0%BF%D1%80%D0%BE-auc-roc/>







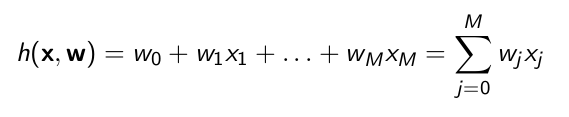




# **Вопрос 23**

Линейная модель. Линейная машина как метод обучения линейной модели.

Линейная модель (то, в каком виде будем искать результат, этого не произностить!):



Линейная машина — алгоритм классификации, основанный на построении линейной разделяющей поверхности.

…

Решение отнесения в класс l, если

В случае одного класса - функция одна (делит пространство на полупространства. Прямая делит плоскость на две полуплоскости, в одной лежат объекты первого класса, в другое - второго).

Обучение состоит в нахождении матрицы W, при которой максимальное количество элементов обучающей выборки окажется правильно распознанным. (или наоборот, минимальное количество неправильно)



Для каждого объекта должно быть выполнено правило - только одна функция должна подходить, т.е. должна быть больше по значению всех остальных функций.

Решение задачи сводится к решению системы неравенств. Для трех функций. например

f1 > f2;

f1 > f3;

f2 > f3

Для решения используется релаксационный метод.

**Конкретно, про линейную машину на этом все. Дальше в целом про линейные классификаторы:**

В случае двух классов, Y=\{-1,+1\}, удобно определить для произвольного обучающего объекта x_i\in X^m величину [*отступа*](http://www.machinelearning.ru/wiki/index.php?title=%D0%9E%D1%82%D1%81%D1%82%D1%83%D0%BF&action=edit) (margin):

M(x_i) = y_i \langle x_i,w \rangle.

В случае произвольного числа классов отступ определяется выражением

M(x_i) = \langle x_i,w_{y_i} \rangle - \!\max_{y\in Y,\, y\neq y_i}\! \langle x_i,w_y \rangle.

Отступ можно понимать как «степень погруженности» объекта в свой класс. Чем меньше значение отступа M(x_i), тем ближе объект подходит к границе классов, тем выше становится вероятность ошибки. Отступ M(x_i) отрицателен тогда и только тогда, когда алгоритм a(x) допускает ошибку на объекте x_i. Это наблюдение позволяет записать функционал [эмпирического риска](http://www.machinelearning.ru/wiki/index.php?title=%D0%AD%D0%BC%D0%BF%D0%B8%D1%80%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D1%80%D0%B8%D1%81%D0%BA) в следующем виде:

Q(w) = \sum_{i=1}^m \bigl[ M(x_i) < 0 \bigr].

Минимизация функционала Q(w) по вектору весов сводится к поиску максимальной совместной подсистемы в системе неравенств. Эта задача является NP-полной и может иметь очень много решений, поскольку минимальное число ошибок может реализоваться на различных подмножествах объектов.

Наиболее известные методы обучения линейного классификатора связаны с заменой пороговой функции потерь её различными непрерывными аппроксимациями:

\bigl[ M < 0 \bigr] \leq L(M),

где L:\:\mathbb{R}\to\mathbb{R}_{+} — непрерывная или гладкая функция, как правило, невозрастающая.

После замены функции потерь минимизируется не сам функционал [эмпирического риска](http://www.machinelearning.ru/wiki/index.php?title=%D0%AD%D0%BC%D0%BF%D0%B8%D1%80%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D1%80%D0%B8%D1%81%D0%BA), а его верхняя оценка.

Q(w) \leq \tilde Q(w) = \sum_{i=1}^m L\bigl( M(x_i) \bigr).

Если функционал Q(w) заменить пороговой функцией потерь можно получить другие методы:

1. Линейный дискриминант Фишера 
2. Однослойный перцептрон 
3. Метод опорных векторов 
4. Логистическая регрессия 

# 

# 

# Вопрос 24

***Метод опорных векторов***

**Кратко:**

Допустим классы разделимы. Начинается всё с того, что строятся 2 параллельные гиперповерхности для разделения всего на 2 множества, (между гиперплоскостями элементов быть не должно). Строим величину расстояния между плоскостями, которую нужно максимизировать при выполнении условий, что одни точки по одну сторону от гиперплоскости, а другие по другую, используем Лагранжа для нахождения максимума.

Если классы не разделимы, то добавляется параметр, который приводит к странной модификации расстояний (но это не эквивалентно переходу к другому пространству), после чего всё должно стать разделимым. (Параметр придумывается как-то заранее)

Либо можно ввести преобразование, для перехода в другое пространство, где всё станет разделимым. Самое распространённое преобразование – Гауссиана (для выбора параметров Гауссианы обычно проводят скользящий контроль или оценки для контрольной выборки, чтобы достичь максимальной обобщающей способности. Процесс называется «Подбор ядра »).

Метод опорных векторов плох, если использовать только расстояния без весов, т.к. все признаки обезличены.

**Подробнее:**

<http://bit.ly/22ZKlkF> - с machinelearning.ru (РЕКОМЕНДОВАНО)

<http://bit.ly/22ZHfNI> – лекция 7 со страницы 8 (про постановку задачи и ее преобразование в другие задачи оптимизации)

# Вопрос 25

*Нейронные сети. Модель перцептрона Розенблатта. Метод его обучения. Теорема Новикова. Переход от сдвига к фиктивному признаку. Многослойные перцептроны. Метод обратного распространения ошибки. Функции активации, удобные для распространения ошибки. Возможность разделения любых множеств.*

В основе нейросетевых методов лежит попытка скопировать компьютером процесс мышления животных.

**Нейрон** - некоторая сущность, которая суммирует входящие сигналы с учётом весов, применяет к результату активационную функцию и выдаёт результат на выход.

Сигналы, приходящие на вход перцептронов рецепторов интерпретируются как входные признаки.

3 типа нейронов:

1. нейроны-рецепторы
2. внутренние нейроны - имеет множество входных связей от рецепторов или внутренних нейронов
3. реагирующие нейроны - имеет множество входных связей от рецепторов или внутренних нейронов

Все сигналы, входящие в нейрон обрабатываются: z = w0 + ∑wiui , после чего применяется **активационная функция** Ф(z) - и это значение становится результатом нейрона и выдаётся другим нейронам (один сигнал с выхода может передаваться многим другим на вход) (ui - значения с других нейронов, wi - **веса** внутри нейрона, w0 - **параметр сдвига**)

(**фиктивный признак** - это когда z = ∑wiui, а в качестве u0 берётся всегда 1)

Примеры активационных функций (в целом, они удобны для подсчёта распространения ошибки):

1. пороговая функция (функция = 0, если z >= const, иначе = 1)
2. **сигмоидная** функция 1 / (1 + e-az)
3. гиперболический тангенс
4. тождественная функция Ф(z) = z

**Перцептрон Роззенблатта** (1957) - нейросетевая модель с одним единственным нейроном

Используется для задачи расознавания с 2-мя классами, активационная функция Ф=1, если z>=0, Ф= -1, если z<0

Особенность перцептрона - простая и эффективная процедура обучения (вычисления wi)

Настройка параметра происходит по обучающй выборке. (предварительно признаковое описание преобразуется в векторные сигналы, а затем вектора описаний из класса K2 умножаются на -1, а из K1 не меняются)

Процедура обучения перцептрона:

1. Случайно выбирается нулевое приближение вектора весовых коэффициентов
2. Преобразованные описания обучающей выборки последовательно передаются на вход перцептрона
3. Если писание x(k), поданное на k-м шаге классифицируется неправильно, то происходит коррекция w(k+1) = w(k) + x(k), если классификация верна, то ничего не меняется.
4. Повторяем до тех пор, пока:
   1. достигается полное разделение объектов из классов K1 и K2
   2. повторение подряд заданного числа операций не приводит к улучшению разделения
   3. оказывается исчерпанный заранее заданный лимит итераций

Теорема Новикова:

В случае, если описания объектов обучающей выборки линейно разделимы в пространстве признаковых описаний, то процедура обучения перцептрона построит линейную гиперплоскость разделяющую объекты двух классов за конечное число шагов.

Отсутствие линейной разделимости двух классов приводит к бесконечному зацикливанию процедуры обучения перцептрона.

**Многослойный перцептрон** - нейросетевые методы распознавания, задаваемые комбинациями связанных между собой нейронов.

Обладает существенно более высокой аппроксимирующей способностью

Обычно сеть формируется в виде слоёв: слои внутренних нейронов осуществляют преобразование сигналов. Слой реагирующих нейронов производит окончательную классификацию объектов на основании сигналов, поступающих от нейронов, принадлежащих внутренним слоям.

Обычно облюдаются следующие правила формирования структуры:

1. Допускаются связи между только между нейронами, находящимися в соседних слоях.
2. Связи между нейронами внутри одного слоя отсутствуют.
3. Активационные функции для всех внутренних нейронов идентичны

Для решения задачи распознавания с L классами используется конфигруация с L реагирующими нейронами.

Сигналы вычисляемые на выходе реагирующих нейронов, интерпретируются как оценки за классы.

Один реагирующий нейрон позволяет аппроксимировать области, являющиеся полупространствами, ограниченными гиперплоскостями.

Нейронная сеть с одним внутренним слоем позволяет аппроксимировать произвольную выпуклую область в многомерном признаковом пространстве (открытую или закрытую). Было доказано также, что Многослойный перцептрон с двумя внутренними слоями позволяет аппроксимировать произвольные области многомерного признакового пространства.

Способ обучения нейронных алгоритмов - **Метод обратного распространения ошибки**.

Потери классификации объекта будем считать как сумму квадрата разности между выходом реагирующего нейрона и тем, что он должен был выдать (т.е. 0, когда объект не принадлежит классу, и 1, иначе)

**Качество аппроксимации** на обучающей выборке - это сумма потерь для каждого объекта выборки. (веса фиксированны). Цель - подобрать такие веса, чтобы улучшить качество аппроксимации.

Основа обучения - метод градиентного спуска.

Метод градиентного спуска - оптимизирует произвольный фукнционал F(θ), θ(k) = θ(k-1) + n \* grad(F(θ(k-1)))

n - парамер, задающий размер каждого шага

Алгоритм обучения:

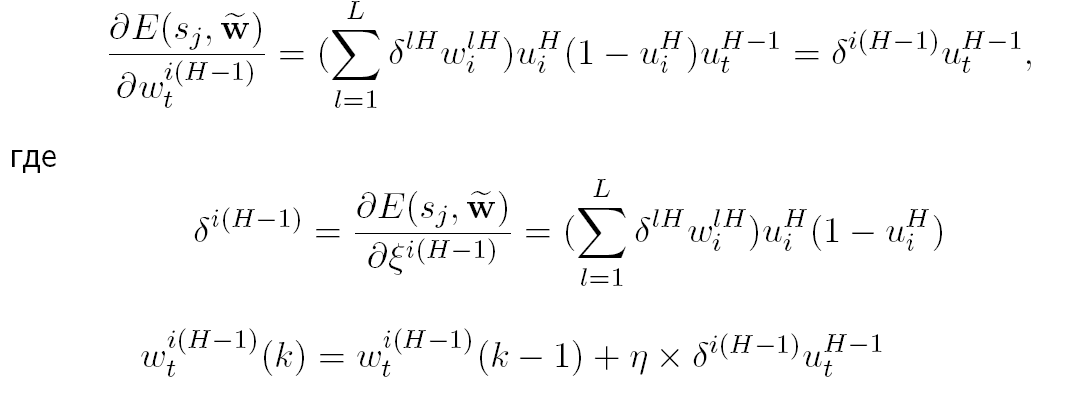
1. Выбираем количество слоёв и количество нейонов в слоях. Присваиваем весам случайные значения. Дальше подаём последовательно объекты обучающей выборки (их векторные описания) и проивзодим коррекцию весовых коэффициентов, как в пунктах ниже:
2. Предполагаем, что все активационные функции - сигмоиды = 1 / (1 + e-z)
3. Проводим коррекцию весовых коэффициентов i-го реагирующего нейрона с нейронами предшествующего слоя

Если взять производную качества аппроксимации по весам одного рассматриваемого реагирующего нейрона, то там по пути получаются удобные математические трансформации.

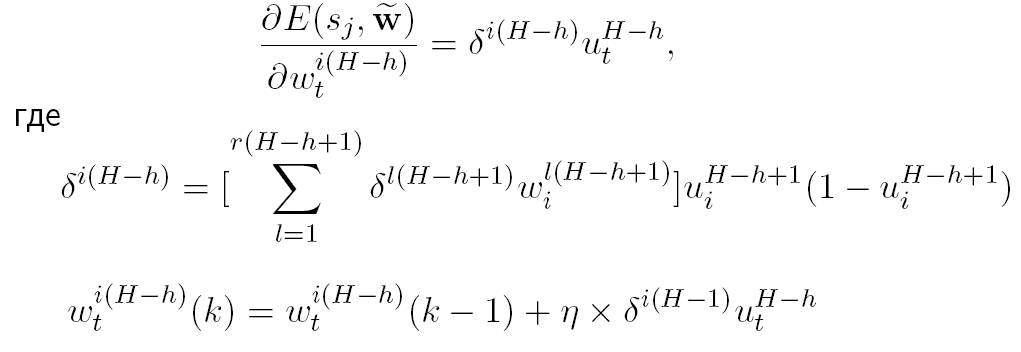
<Сенько, лекция 6, слайд 18> и градиентная коррекция будет выглядеть:

w(k) = w(k-1) + n \* u \* (-2 \* (α - g(x))\*(1 - g(x))\*g(x)), где g - сигнал на выходе реагирующего нейрона, α - нужное значение на выходе реагирующего нейрона, u - значение на выходе того нейрона, вес для которого мы пересчитываем (т.е. входное значение соответсвующее w)

1. Теперь аналогично рассматривая на один слой выше, тоже продифференцировав и немного сделав мат выкладок <Сенько, лекция 6, слайд 20> получим следующую формулу:



1. После этого были математические выкладки для общего случая, когда мы берём некоторый слой нейронов, проводились они аналогично:

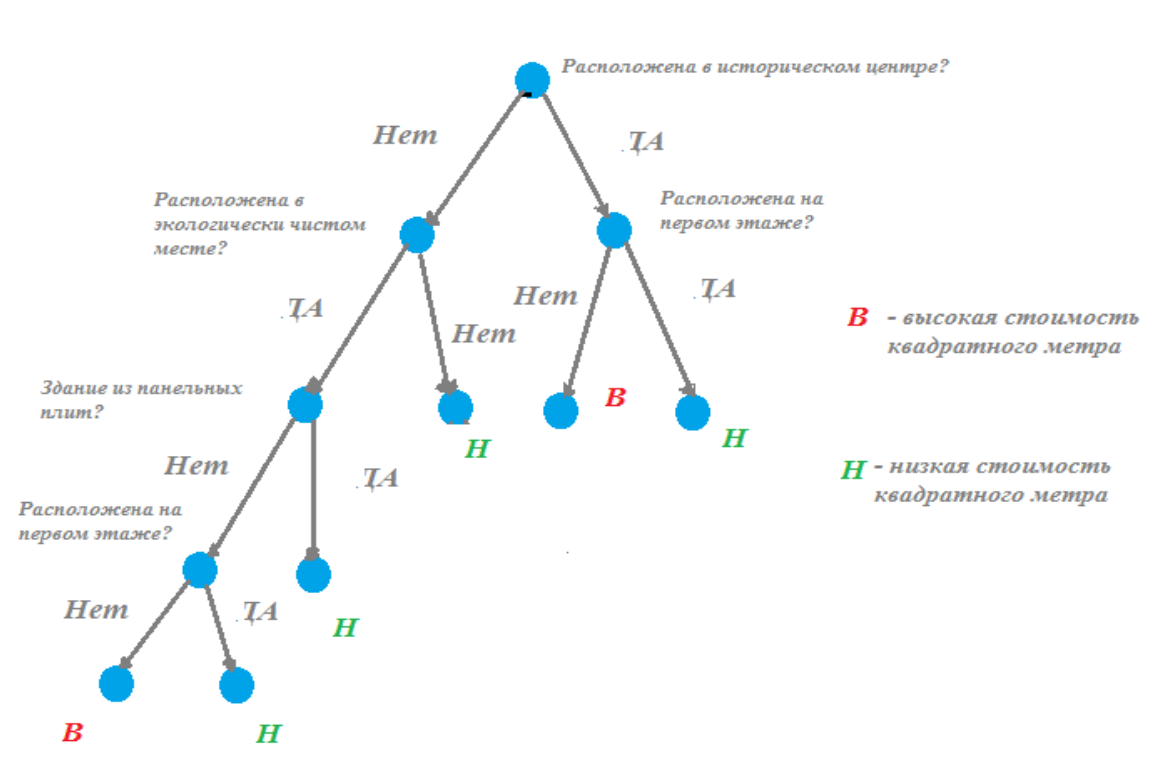


1. Таким образом, задаётся общая схема метода обратного распространения ошибки
2. Обучение заканчивается при выполнении одного из заранее заданных условий:
   1. Величина функционала ошибки оказывается меньше выбранного порогового значения
   2. Изменения функционала ошибки на протяжений нескольких последних итераций оказывается меньшим некоторого порогового значения
   3. общее время обучения превышает допустимый предел

# Вопрос 26.

Решающие деревья

Решающие деревья воспроизводят логические схемы, позволяющие получить окончательное решение о классификации объекта с помощью ответов на иерархически организованную систему вопросов. Причём вопрос, задаваемый на последующем иерархическом уровне, зависит от ответа, полученного на предыдущем уровне. Подобные логические модели издавна используются в ботанике, зоологии, минералогии, медицине и других областях. Пример, решающего дерева, позволяющая грубо оценить стоимость квадратного метра жилья в предполагаемом городе приведена на рисунке 1.



Схеме принятия решений, изображённой на рисунке 1, соответствует связный ориентированный ациклический граф – ориентированное дерево. Дерево включает в себя корневую вершину, инцидентную только выходящим рёбрами, внутренние вершины, инцидентную одному входящему ребру и нескольким выходящим, и листья – концевые вершины, инцидентные только одному входящему ребру.

Каждой из вершин дерева за исключением листьев соответствует некоторый вопрос, подразумевающий несколько вариантов ответов, соответствующих выходящим рёбрам. В зависимости от выбранного варианта ответа осуществляется переход к вершине следующего уровня. Концевым вершинам поставлены в соответствие метки, указывающие на отнесение распознаваемого объекта к одному из классов. Решающее дерево называется бинарным, если каждая внутренняя или корневая вершина инцидентна только двум выходящим рёбрам. Бинарные деревья удобно использовать в моделях машинного обучения.

Распознавание с помощью решающих деревьев. Предположим, что бинарное дерево T используется для распознавания объектов, описываемых набором признаков X[1],...,X[n]. Каждой вершине v дерева T ставится в соответствие предикат, касающийся значения одного из признаков. Непрерывному признаку X[j] соответствует предикат вида - некоторый пороговый параметр.

Категориальному признаку , принимеющему значения из множества является элементом дихотомического разбиения Выбор одного из двух, выходящих из вершины v ребер производится в зависимости от значения предиката.

Процесс распознавания заканчивается при достижении концевой вершины (листа). Объект относится классу согласно метке, поставленной в соответсттвие данному листу.

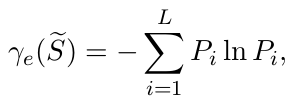
**Обучение решающих деревьев**

Рассмотрим задачу распознавания с классами K[1], …, K[L]. Обучение производится по обучающей выборке и включает в себя поиск оптимальных пороговых параметров или оптимальных дихометрических разбиений для признаков Х[1],...,X[n]. При этом поиск производится исходя из требования снижения среднего индекса неоднородности в выборках, порождаемых искомым дихометрическим разбиением обучающей выборки .

Индекс неоднородности вычисляется для произвольной выборки , содержащей объекты из классов К[1],...,K[L]. При этом используется несколько видов индексов, включая:

1. энтропийный индекс неоднородности
2. индекс Джини
3. индекс ошибочной классификации.

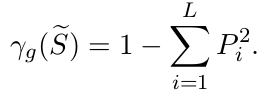
Энтропийный индекс неоднородности вычисляется по формуле





При этом принимается, что 0ln(0) = 0. Наибольшее значениепринимет при равенстве долей классов. Наименьшее значение достигается при принадлежности всех объектов одному классу.

Индекс Джини вычисляется по формуле



Индекс ошибочной классификации вычисляется по формуле



Нетрудно понять, что индексы (2) и (3) также достигают минимального значения при принадлежности всех объектов обучающей выборке одному классу.

Слайды Сенько

[ <http://www.machinelearning.ru/wiki/images/a/a5/MOTP14_8.pdf> ]

# Вопрос 27

**Принцип частичной прецедентности**. При распознании диагностике важны некоторые комбинации (именно конкретные наборы, а не целые классы)

**Тест** – это набор признаков, позволяющих разделить классы. (т.е. любые 2 объекта из любых 2-х классов отличаются хотя бы по одному признаку из теста)

**Тупиковый тест** – тест, из которого нельзя удалить ни один из признаков.

Выбор класса при классификации происходит на основании голосования по тесту (учёт совпавших признаков, причём могут учитываться какие-то веса)

Поиск теста – это задача о покрытии, NP-трудная.

# Вопрос 29

Алгоритмы, основанные на голосования по наборам закономерностей.

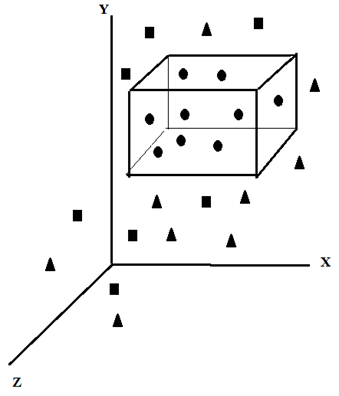
[[ Лекция 10]] > Коллективные методы, основанные на голосовании по системам *и далее.*

*[[ Лекция 5]] > Представительные наборы*

или метода Сенько ([[ММО3]]пункт 4.5).

Одним из эффективных подходов к решению задач прогнозирования и распознавания является использование **коллективных решений по системам закономерностей**. Под **закономерностью** понимается распознающий или прогностический алгоритм, определённый на некоторой подобласти признакового пространства или связанный с некоторым подмножеством признаков.

В качестве примера закономерностей могут быть приведены **представительные наборы**, являющиеся по сути подмножествами признаковых описаний, характерных для одного из распознаваемых классов. Аналогом представительный наборов в задач с вещественнозначной информацией являются логические закономерности классов. Под **логической закономерностью** класса *Kl* понимается область признакового пространства, имеющая форму гиперпараллелепипеда и содержащая только объекты из *Kl*



Математически логическая закономерность класса *Kl*, которую мы будем обозначать *r(l)*, описывается с помощью наборов предикатов вида:



где *i = 1, ..., n*.

Напомним, что предикатом называется утверждение, принимающее значения «ИСТИНА» или «ЛОЖЬ» в зависимости от значений входящих в них переменных. Полностью логическая закономерность задаётся конъюнкцией предикатов вида (1):



Очевидно, что множество векторов (x1, …, xn), для которых, как раз представляет собой гиперпараллелепипед в многомерном пространстве признаков. Не все признаки являются на самом деле существенными для закономерности *r(l)*. Для несущественного признака  отрезок  совпадает с отрезком, из которого принимает значения признак .

На этапе обучения для каждого класса *Kl* строится множество логических закономерностей . Границы  подбираются таким образом, чтобы равенство P[*r(l)*] = «ИСТИНА» выполнялось бы на максимально большом числе объектов обучающей выборки из класса *Kl* и равенство P[*r(l)*] = «ЛОЖЬ» выполнялось бы на всех объектах обучающей выборки из класса *Kl*. Наряду с полными логическими закономерностями, удовлетворяющими последним условиям, используются также частичные логические закономерности, для которых допускается попадание в них небольшой доли объектов чужих классов. Методы построения логических закономерностей подробно излагаются в работе [11], а также книге [9].

Предположим, что нам требуется распознать новый объектs\* .Для каждого класса*Kl*ищется число закономерностейв , для которых P[*r(l)*] = «ИСТИНА». В качестве оценки за класс используется доля таких закономерностей в . Классификация s\* производится с помощью стандартного решающего правила. То есть объект относится в тот класс, оценка за который максимальна.

# Вопрос 30

Bagging. Boosting. Решающие леса (random forest).

[[Лекция 10 Сенько]]

Bootstrap -> bagging || Boosting

Кратко:  
Можно из одной обучающей выборки генерировать разные наборы и на них обучаться, причем в сгенерированной выборке могут оказаться одинаковые элементы, а некоторые

элементы в неё могут вовсе не попасть. Этот процесс называется бутстрап. Существует два основных подхода бустрапа - бэггинг и бустинг. При бэггинге выборки генеируются сразу, а затем на каждой из них производится обучение, на выходе N алгоритмов.   
При бустинге, генерация выборок происходит иттеративно T раз. Причем на каждом шаге для каждого элемента рассчитывается вес. В этом случае, некоторые шумовые элементы, могут быть исключены.

После бустрапа решение принимается на основе ансамбля.

Одним из способов получения ансамбля является использование алгоритмов, обученных по разным обучающим выборкам, которые возникают в результате случайного процесса, лежащего в основе исследуемой задачи.

Обычно при решении прикладной задачи в распоряжении исследователя имеется обучающая выборка St = {s1, .. , sm} ограниченного объема*.* Однако процесс генерации семейства выборок из генеральной совокупности?? может быть имитирован с помощью процедуры бутсрэп (bootstrap), которая основана на выборках с возвращениями из St . В результате получаются выборки S\*bg, включающие объекты из обучающей выборки St . Следует заметить, что некоторые объекты из St могут встречаться в S\*bg более одного раза.

Предположим, что с помощью процедуры bootstrap получено T выборок. С помощью заранее выбранного метода, используемого для обучения отдельных алгоритмом распознавания, получим множество, включающее T распознающих алгоритмов Abg = {A1bg, … , ATbg}

Для получения коллективного решения может быть использован простейший комитетный метод, относящий объект в тот класс, куда отнесло большинство алгоритмов. Данная процедура носит название бэггинг (bagging), что является сокращенным названием Bootstrap Agregating. Процедура бэггинга показывает высокий прирост обобщающей способности по сравнению с алгоритмами, обучения с помощью базового метода по исходной обучающей выборке St, в тех случаях, когда вариационная составляющая ошибки базового метода высока.   
 При использовании в качестве базового метода решающих деревьев процедура бэггинга приводит к построению ансамблей решающих деревьев (решающие лесов).

Основной идеей алгоритма бустинг является пошаговое наращивание ансамбля алгоритмов. Алгоритм, который присоединяется к ансамблю на шаге k обучается по выборке, которая формируется из объектов обучающей выборки St.

В отличие от метода бэггинг объекты выбираются не равноправно, а исходя из некоторого вероятностного распределения, заданного на выборке St.

# Вопрос 31

***Из 2015 Теормин.ММО:***

**Задача анализа выживаемости (краткое определение)**

Анализ выживаемости - позволяющих оценить вероятность наступления события. Это математическая модель зависимости функции риска от независимых переменных-факторов. В анализе выживаемости решается задача оценки функции выживания или функций, производных от нее.

Выбор метода для анализа выживаемости зависит от исходных данных:

* наличия зависимой и независимых переменных-предикторов;
* наличия или отсутствия цензурированных данных.

Из слайдов: **Целью таких задач** является восстановление вероятности того, что ожидаемое критическое событие с исследуемым объектом произойдёт не ранее произвольного момента времени. Таким критическим событием может быть отказ изделия в технике, гибель испытуемого организма в биологии или смерть пациента в медицине. Таким образом целью анализа является вычисление функции (кривой) выживаемости S(t) = P{T > t} , где через T обозначено время наступления критического события, P {T > t} обозначает вероятность того, что критическое событие произойдёт позже момента t.

***Из 2015 Методичка:***

Оценка вероятности выживания.

**Survival метод оптимизации**.

Цель – выяснить, что некоторое критическое событие случится не раньше некоторого времени (т.е. определить вероятность того, что случится событие P(T > t), где T – критическое событие – эта вероятность называется **вероятность выживания**)

На ходе есть выборка:

{(индикатор типа информации, момент времени последнего получения информации, начальные показатели объекта), (), …}

(например: (жив, 14:00, съел таблетки типа А) )

**Метод Каплан-Майера**.

nj – те, кто в начале нового интервала был жив.

dj – количество критических событий на интервале.

S(t) = П i j=1((nj – dj) / nj), где j = 1,i – это разбиение интервала времени на кусочки.

Существует много методов сравнения оценок S(t), самая популярная – модель Кокса.

**Модель Кокса**.

**Мгновенный риск** – (вероятность гибели на интервале от T до T+∆t, при условии, что до этого критическое событие не наступало). Потом применяется факторизация Кокса с введением кучи параметров b1, b2, … bn, которые ищутся при помощи метода максимального правдоподобия (метод был признан мутным и рассмотрен не был)

**Оценка качества** проводится на основе сравнения реальной функции распределения критических событий и производной вероятности выживания (производная S(t))