**Методичка по курсу “Методы Машинного Обучения” (2015)**

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

составлено avasite по материалам лекторов

Данный документ является хорошим материалом для приближённого изучения тех тем, которые здесь описаны (без глубокой математики). Однако в целом не хватает плюсов и минусов тех или иных подходов.

*Курсивом выделено то, на что можно забить.*

Наблюдаются проблемы с орфографией и пунктуацией.

Ссылки на слайды имеются ввиду – версии слайдов 2015 года.

Оглавление

[1. Глава 1. Подготовительная информация 4](#_Toc440562441)

[1.1. Общие сведения из теорвера и матстата 4](#_Toc440562442)

[1.2. Специальные сведения из теорвера и матстата 4](#_Toc440562443)

[1.3. Прочее 4](#_Toc440562444)

[2. Глава 2. Часть от Майсурадзе 4](#_Toc440562445)

[2.1. Введение 4](#_Toc440562446)

[2.1.1. Структура области Интеллектуального Анализа Данных 4](#_Toc440562447)

[2.1.2. Методология анализа данных 6](#_Toc440562448)

[2.2. Задача анализа данных, качество классификации 7](#_Toc440562449)

[2.3. Элементы принятия решений. Парадоксы 10](#_Toc440562450)

[2.3.1. Парадокс Симпсона, Lion at the gate paradox 10](#_Toc440562451)

[2.3.2. Кажущееся vs целесообразное 10](#_Toc440562452)

[2.4. Скользящий контроль 12](#_Toc440562453)

[2.5. Особенности поиска кратчайшего пути. Алгоритм Дейкстры 14](#_Toc440562454)

[3. Глава 3. Часть от Сенько. (полным разумным конспектом являются слайды Сенько, в этой главе лишь некоторая вытяжка) 16](#_Toc440562455)

[3.1. Прогнозирование по прецедентам 16](#_Toc440562456)

[3.2. Способы поиска закономерностей 17](#_Toc440562457)

[3.3. Обобщающая способность 17](#_Toc440562458)

[3.4. Байесовский классификатор 18](#_Toc440562459)

[3.5. Эмпирические методы оценки обобщающей способности 19](#_Toc440562460)

[3.6. Линейная регрессия 19](#_Toc440562461)

[3.6.1. Линейная регрессия 19](#_Toc440562462)

[3.6.2. Обобщённая ошибка 20](#_Toc440562463)

[3.6.3. Способы регуляризации линейной регрессии 21](#_Toc440562464)

[3.7. Задачи прогнозирования 21](#_Toc440562465)

[3.8. Нейронные методы, персептрон Роззенблатта 22](#_Toc440562466)

[4. Глава X. Не обработано, но есть полезные записки с лекций 24](#_Toc440562467)

[4.1. Прочее 24](#_Toc440562468)

[4.2. Средний риск, Байесовское решающее правило 24](#_Toc440562469)

[4.3. Метод сравнения с эталоном, метод минимизации эмпирического риска, верхний предел точности 24](#_Toc440562470)

[4.4. Задача линейной регрессии (почему бы не перенести её в 3.6?) 24](#_Toc440562471)

[4.5. ? ? 25](#_Toc440562472)

[4.6. Нейронные сети, решающие деревья 26](#_Toc440562473)

[4.7. ? ? 27](#_Toc440562474)

[4.8. ? ? 27](#_Toc440562475)

[4.9. Способы генерации ансамблей из одной выборки 28](#_Toc440562476)

[4.10. Оценка вероятности выживания. 28](#_Toc440562477)

[4.11. Кластерный анализ. 29](#_Toc440562478)

[5. Дополнение 29](#_Toc440562479)

[5.1. Модели данных 29](#_Toc440562480)

[5.2. Объём и площадь шара 30](#_Toc440562481)

[5.3. Форматы хранения данных 31](#_Toc440562482)

[5.4. Кодировки 31](#_Toc440562483)

[5.5. Снежинка и звезда 32](#_Toc440562484)

!!! Надо придумать нормальные названия для некоторых глав и параграфов.

Предисловие

Существует 3 принципиальных точки зрения на ММО в целом:

1. наука, примеры:
	1. *теория распознавания, прогнозирования, машинного обучения (machine learning, pattern recognition)*
	2. *предварительная обработка (подготовка) исходной информации в задачах распознавания (preprocessing)*
	3. *изучение понятия «сходства», способы его эффективного представления и обработки (similarity, proximity)*
	4. *метрические методы интеллектуального анализа данных*
	5. *анализ связей, анализ графов и сетевых структур (в частности, сетей управления) (link mining, relation mining, network analysis)*
	6. *анализ развития сложных систем*
	7. *обработка и анализ слабоструктурированной (текстовой) информации*
	8. *методы визуализации информации: аналитические пространства, сети*
2. технологии (данные, аналитическая обработка), примеры:
	1. *Технологии сбора, преобразования и хранения информации (Консолидация данных, Хранилища, Оценка качества одного или группы хранилищ)*
	2. *Технологии представления информации (Системы отчетности, Интерактивные информационные панели, Оперативный анализ данных (OLAP))*
	3. *Системы интеллектуального анализа данных*
3. бизнес.

Целью курса Майсурадзе ставит подготовить из нас специалистов, которые смогут сотрудничать со специалистами ИАД.

Сенько ориентирован на медицину - химия, биология, …, хотя занимается разным всяким.

*Антропология и этнография.*

*Накоплена информация о встречаемости мифологических сюжетов в фольклорных традициях по всему мира. Использования современных методов анализ данных в приложении к этой информации позволяет выявить связь между различными этносами, реконструировать исторический процесс расселения народов.*

(!!! 0.3. Знакомство (Сенько) (big source) (extra) - я скипнул там про верификацию)

# Глава 1. Подготовительная информация

## Общие сведения из теорвера и матстата

!!! Надо сюда дописать определения соответствующих понятий, не больше 2-х страниц.

Надо вспомнить, что такое:

случайная величина, выборка, условная вероятность, формула байеса, математическое ожидание, дисперсия, нормальное распределение, несмещённая и оптимальная оценка, совместное распределение, ковариация, корреляция.

и ещё что-то там было про критерии хи-квадрат и гипотезы.

## Специальные сведения из теорвера и матстата

**Выборка** – набор объектов, для которых всё известно.

**Генеральная совокупность** – всё множество значений, на которых мы обучаемся.

[со слайда 6 первой лекции Сенько] --- множество объектов, которые потенциально могут возникать в рамках F (F - процесс или явление, для которого решается задача прогнозирования)

* ну тогда на ней мы не обучаемся. Генеральная совокупность - множество всевозможных объектов, которые могут возникать в нашй задаче. (!!!думаю тут стоит исправить определение или удалить эту переписку, если считате, что определение верно)

"**Паретто-оптимальное множество**" - это множество тех оценок, для каждой из оценки нету доминирующей оценки. Т.е. оценки, которая лучше по каждому параметру.

По множеству паретто-оптимальных точек строится паретто-кривая, после чего каждый раз выбирают из этой кривой те точки, которые устраивают бизнес. Фактически это наложение новых требований поверх тех, которые уже были проанализированы (например, так могут часто делать с ценой, сначало выбирая по каким-то основным содержательным параметрам, а потом выбирая приемлимую цену).

## Прочее

Стохастические модели (т.е. случайные модели) – самые распространённые.

Совместное распределение представляет из себя полное описание модели с точки зрения математики.

# Глава 2. Часть от Майсурадзе

## Введение

### Структура области Интеллектуального Анализа Данных

**Методология** - формулировки и понятия.

**Технология** - процедуры действий.

**Кибернетическая система** – система, поведение которой определено и детерминировано.

**Интеллектуальная система** – подразумевает принятие решений в течение своей работы.

**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).

Физика — моделирование явления (пример классического моделирования) (стоит в явном виде написать почему ИАД != АД)

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.*

**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. *...*

<!!! для полного счастья здесь не хватает в качестве подпунктов к каждому классу задач реальных названий методов и их кратких описаний (их сути), которые встречаются в курсе>

**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. (стоит в двух словах сказать при чём тут ИАД)
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
9. **Задача снижения размерности** - в этой задаче переходят от одних описаний к другим (один из популярных способов снижения размерности - это снижение через нейронную сеть, в которой выходы замкнуты на входы, чтобы нейронная сеть пыталась отражать тождественную функцию, при этом внутренний слой нейронной сети имеет заниженное количество нейронов, чтобы они и отражали снижение размерности, данных, которые через них походят)

Виды представления исходных данных:

1. **Признаковое описание объектов** - фиксированный набор признаков, каждому объекту ставится в соответствие вектор значений признаков.
2. ***Плоские таблицы****,* ***кросс-таблицы****, ...*
3. **Описание эталонами** - одна или несколько функций расстояния, каждому объекту ставится в соответствие набор всех расстояний до всех заданных эталонов.
4. **Метрическое описание пар объектов** - фиксированный набор функций расстояния, каждой паре объектов ставится в соответствие вектор значений расстояний.
5. **Транзакционные данные**, **формальные контексты** - фиксировано множество элементов, **транзакция** - конечное подмножество элементов, наличие элемента в транзакции — бинарный признак

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

*Необходимые технологии: базы данных, 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

Методология ~~философии~~ 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

## Задача анализа данных, качество классификации

Постановка задачи обучения:

1. Каждому объекту сопоставляется унифицированное описание (description) (определяется признаковое описание объекта)

Например: признаковое описание объекта (feature-based description) или свободные переменные (multiple predictor/response/independent variables, multivariable data)

1. Каждому объекту сопоставляется значение целевого признака или зависимой переменной (target feature, outcome/dependent variable)
	1. если множество значений целевого признака дискретное – задача классификации
	2. если множество значений целевого признака непрерывное – задача восстановления регрессии
	3. в простом случае целевого признака – это скаляр
2. Требуется найти отображение, переводящее описание объекта в значение целевого признака и действующее на генеральной совокупности объектов (feature extraction)

!!! не помню, к чему это, но смотрится важным

(Это что-то вроде процесса обучения - этапа Learning)

1. Sensor Level – «описание признаков», по которым можно сортировать и которые потом будут использоваться (датчики, измерения)
2. Sensor Fusion – «правила», по которым можно группировать в один признак объекты на входе (если есть несколько датчиков, то измерения должны проводиться в совокупности)

После первых двух этапов работаем с данными

1. Feature (descriptor) extraction – «выделение» признаков, которые будут использоваться алгоритмом для обработки в дальнейшем (применение дескрипторной функции, ряд определенных признаков)
	1. Feature Fusion - объединение по каким-либо признакам
	2. Score Fusion - объединение по каким либо оценкам признаков

После третьего этапа уже признаки

1. Распознавание/Оценки (Prediction Scoring)

Теперь оценки

1. Decision Making
	1. Decision Fusion (формализация того, как проводить выбор/принятие решения)

На выходе решение

2 этапа функционирования системы:

1. Создание системы (Learning) (настройка системы)

В ходе «обучения» могут появляться различные «сущности».

1. Эксплуатация системы (Prediction)

Базовые понятия Анализа Данных (АД):

1. **Описание объекта** – то, что поступит на вход отображения. *Бывает, что объект и его описание не всегда различаемы, потому что объект может быть с самого начала задан в виде своего описания.*
2. Отображение описания объекта в значение целевой функции, в зависимости от контекста, принято называть **классификатором** (classifier, predictor), **регрессией** (predictor function), алгоритмом…
3. **Информационная модель** или **модель данных** – множество отображений, из которого выбираем ответ. Обычно модель – параметрическое семейство отображений: выбрать отображение – указать значения всех параметров.
4. **Обучение, настройка алгоритма** – процесс выбора конкретного отображения из модели. (!!!скорее всего это определение не совсем точное. Обучение - процесс настройки метода обучения, согласно известным данным (прецедентам и не только))
5. **Метод обучения** – процедура, которая выбирает из модели конкретное отображение.

Метод обучения зависит от некоторой дополнительной информации – частичное описание отображения, ограничения универсальные или локальные.

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

Пример:

объект: человек

признаковое описание: (возраст - x1, рост - x2)

целевой признак: вес

Задача восстановления регрессии: необходимо найти отображение, которое по возрасту и росту определяет вес

Пример универсального ограничения: с увеличением роста вес должен расти

Информационная модель: M={f:R2→R | f(x1, x2)=a1x1+a2x2+a0} (это трёх-параметрическое семейство)

процесс обучения должен назначить значения параметрам ai, значения можно выбрать как угодно.

Подход к процессу обучения с точки зрения оптимизации:

1. ввести понятие качества отображения (performance metrics)
2. указать модель
3. выбрать из модели отображение с наилучшим качеством

таким образом задача обучения сводится к задаче оптимизации

Классы задач прикладной статистики (в ней в отличие от ИАД задача классификации и восстановления регрессии устанавливается поэтапно, а не сразу):

1. Дисперсионный анализ - установить наличие влияния заданного фактора на изучаемый процесс
2. Корреляционный анализ - оценить силу такой связи
3. Регрессионный анализ - выбрать конкретную математическую модель связи, оценить адекватность отражения ею установленной взаимосвязи

*(с точки зрения Майсурадзе - типичная ошибка, это взять метод решения не от той задачи)*

Для задачи обучения по прецедентам, даны прецеденты (**размеченная выборка** - конечный набор объектов с описаниями и истинными значениями целевого признака) и некоторый функционал качества (который агрегирует результаты алгоритма для всех объектах (и предсказанных им целевых признаков)).

Значение функционала качества зависит и от отображения, и от размеченной выборки.

Для задачи восстановления регрессии (*отражение на непрерывное множество*), алгоритм не даст значения равного целевому признаку, поэтому вводится невязка на основе некоторой метрики на множестве значений целевого признака.

Типичный функционал качества агрегирует множество невязок, полученных на прецедентах

Для задач снижения размерности целевой признак отсутствует, в ней переходят от одних описаний к другим. Принято вводить невязку через метрику на пространстве описаний.

Типичный функционал качества агрегирует множество невязок между исходными и новыми описаниями (!!! а исходные данные тут причём?)

Для задачи классификации, множество значений целевых признаков дискретно и поэтому можно говорить о точном равенстве истинного ответа с результатом распознавания.

**Корректный классификатор** - классификатор, который на наборе прецедентов не даёт ни одной ошибки.

На практике требовать полной корректности классификатора - не самое удачное решение. Для большинства показателей качества, ответ даваемый в рамках оптимизационного подхода не будет корректным, да и проблему переобучения никто не отменял.

**Матрица ошибок (МО)** - двумерное аналитическое пространство (истинные ответы (actual) x результат распознавания (predicted)) (*кажись это означает, что в клетках матрицы-таблицы стоят соответствующие этой клетке алгоритмы*)

Матрица ошибок позволяет рассчитать различные показатели качества классификатора на размеченной выборке.

На главной диагонали - правильно распознанные прецеденты.

Прецеденты (S1, c1), …, (Sm, cm), N - алгоритм, который предсказал имел истинный ответ j’, а предсказал j’’.



Бинарная классификация (важный частный случай классификации). так же о ней говорят, как о **методе диагностики**:

Классификация на два класса часто понимается как диагностики наличия (positive) / отсутствия (negative) какого-то свойства или состояния. *В диагностике (особенно медицинской) сложилась своя терминология, параллельная анализу данных.*

* «положительный» ответ для объектов с наличием состояния - истинно положительные случаи - true positive
* «положительный» ответ для объектов с отсутствием состояния - ложноположительные случаи - false positive - ошибка второго рода
* «отрицательный» ответ для объектов с наличием состояния - ложноотрицательные случаи - false negative - ошибка первого рода
* «отрицательный» ответ для объектов с отсутствием состояния - истинно отрицательные случаи - true negative

Важнейшие операционные характеристики метода диагностики:

* Чувствительность (Se, sensitivity)
* Специфичность (Sp, specificity)

Вспомогательные критерии информативности (эффективности) метода диагностики:

* Общая точность (Ac, accuracy)
* Прогностичность положительного результата (+VP, positive predictive value)
* Прогностичность отрицательного результата (-VP, negative predictive value)
* **Чувствительность (recall, полнота)** - вероятность положительного результата теста у лиц с заболеванием =TP/(TP+FN). Чем выше чувствительность, тем чаще выявляются патологии, тем больше ложных тревог. Высокая чувствительность полезна при массовом скрининге населения, чтобы отобрать возможных больных для дальнейшей диагностики.
* **Специфичность (**specificity**)** - вероятность отрицательного результата теста у лиц без заболевания =TN/(TN+FP). Чем выше специфичность, тем надежнее подтверждается патология, тем больше пропусков цели. Высокая специфичность полезна на втором этапе диагностики, когда надо доказать наличие уже предполагаемой болезни из малого числа диагнозов.

Метод с высокой специфичностью называется **дискриминатор**.

* **Диагностическая точность (precision)** - вероятность правильного ответа среди всех положительных ответов (какая доля действительно своих - странное высказывание, может его вообще убрать?) = TP/(TP+FP)
* **Общая точность (Aсcuracy)** – доля правильных ответов = (TP+TN)/m (!!!) ;(m = TP + TN + FP + FN)

Чтобы объединить несколько критериев в один, чтобы потом можно было работать с одним числом, используется **F-measure**: 1/F = (1/P + 1/R) /2 !!!пояснения? F - …, P- …, R - ...

Можно добавлять веса, но они будут идти исключительно из предметной области.

**Микроусреднение** – когда мы сразу считаем показатель качества по всем показателям.

**Макроусреднение** – когда мы сначала группируем по группам, а потом уже по показателям групп выводим показатель качества.

С точки зрения бизнеса всегда нужно получить ответ, но иногда на выходе добавляют возможный вариант ответа «не знаю». Часто обозначают Δ (a - для positive сказали “не знаю”).

Precision – диагностическая точность = TP/(TP+FP)

Recall – полнота = TP/(TP+FN) + a, но часто каждый раз придумывают по своему, могут добавить a/2, ...

## Элементы принятия решений. Парадоксы

### Парадокс Симпсона, Lion at the gate paradox

Модель принятия решений может быть чересчур опрощённой, в результате чего будет проигнорированная важная информация (например, если мы оцениваем кино, то лучше это делать отдельно для мужчин и женщин)

**Парадокс Симпсона**:

Явление в статистике, когда при наличии двух групп данных, в каждой из которых наблюдается одинаково направленная зависимость, при объединении этих групп направление зависимости меняется на противоположное.

Пример:

|  |  |  |  |
| --- | --- | --- | --- |
| лекарство 1 | лекарство 2 |  |  |
| 1 из 5 мужчин вылечился = 20% | 3 из 10 вылечилось = 30% | 3/4 = 75% | 5/7 = 71% |
| 4 из 9 женщин вылечилось = 44% | 1 из 2 вылечился = 50% | 2/7 = 28% | 1/4 = 25% |
| в сумме 5 из 14 = 36% | в сумме 4 из 12 = 33% | 5/11 = 45% | 6/11 = 54% |

*Но как это возможно? Суть в том, что выборки были не одинакового размера, и есть 4 разных категории: маленькая выборка с плохим показателем, маленькая выборка с хорошим показателем, и аналогично для большой выборки, - а дальше, когда мы складываем выборки (мужчин и женщин), то к первой подмешивается вторая выборка и из-за относительного размера выборок (всех 4-х) выборка для лекарства 1 портится одним образом (например туда добавляется ), а выборка для лекарства 2 портится другим образом.*

**Lion at the gate paradox** (!!! какое настоящее название парадокса, а то это не гуглится?):

Парадокс заключается в том, что не важно, сколько непрямых признаков мы знаем, когда имеем дело с редким эффектом.

Пример:

в нашей местности вероятность встретить опасного животного 0.0001%

у 95% опасных животных и у 2% неопасных животных есть острые клыки

90% опасных животных и 5% неопасных рычат

Выйдя за ворота вы встретили льва, он рычит и имеет острые клыки. Он опасен?

Вероятность, что он опасен = P = 0,95\*0,90\*0,0001 / (0,95\*0,90\*0,0001 + 0,02\*0,05\*0,9999) =

0,0000855 / (0,0000855 + 0,0009999) = 8%

Если мы будем считать, что всё, что рычит и имеет острые клыки - опасно, то мы будем ошибаться с вероятностью в 92%.

### Кажущееся vs целесообразное

Важен размер выборки, на малых выборках степень уверенности, о верности принятого решения - меньше. (например, мы вылечили 8 из 10 или 800 из 1000).

Люди на практике часто принимают решения, не являющиеся оптимальными в соответствии со статистикой.

Так же людям часто приходится принимать решения в неочевидных ситуациях, в которых применение статистики - сомнительно, а выбирать-то надо (например, на очень маленькой выборке)

Логически нужно выбрать то решение, которое даст лучший результат (!!! с чего бы это?)

Вообще человек как правило выбирает решение, балансируя между 2-мя критериями: больше profit vs надёжность. В зависимости от ситуации (формулировки вопроса, важности цели, настроения, ...) человек будет по-разному полагаться на “авось повезёт”.

**Теорема фон Неймана-Моргенштерна** (Von Neumann–Morgenstern utility theorem):

Человек следующий рациональному поведению должен вести себя в соответствии с 4-мя аксиомами:

1. completeness (у индивида есть предпочтение между любыми 2-мя лотереями (<, >, ==)) - т.е. на множестве лотерей задано некоторое отношение
2. transitivity - заданное отношение транзитивно
3. continuity - есть некоторый переломный момент, когда индивид начинает склоняться от одной лотереи к другой (пусть L <= M <= N, существует 0 <= p <= 1, такое что pL + (1-p)N == M)
4. independence - пусть L < M, тогда для любого N и 0 < p <= 1, выполнено pL + (1-p)N < pM + (1-p)N (какой смысл в этой аксиоме?)

**Санкт-Петербургский парадокс**:

Питер подбрасывает монетку, если орёл, то платит 1$, если решка, то н подрасывает ещё раз и если орёл, то платит 2$, … по степеням двойки

Среднее ожидаемое значение = ½ \*1 + ¼ \* 2 + ⅛ \* 4 + 1/16 \* 8 + … = ½ + ½ + ½ + … = бесконечности

Но обычно окружающие люди согшалаются играть за 20$, не смотря на то, что они бы были в плюсе при постоянной игре, при абсолютно любой начальной ставке.

**Elsberg paradox (парадокс неопределённости)**:

Известные вероятности предпочтительнее неизвестных.

Примеры:

1. Выбрая между лотереей, где человек получит 100 баксов в 1 из 80 случаев (т.е. средний выигрыш = 1.25$) и просто гарантированной разовой выплатой в 1$, люди выбирают первое.

(!!! возможно на слайде лажа, и нужно было сделать так, чтобы было 80 баксов на 100 случаев, чтобы ожидаемая выплата была меньше бакса, тогда если человек выберет лотерею - то это ещё более противоречиво, в то время как пример выше - это в целом нормальное поведение) (ну или люди выберут 2 вариант, то есть гарантированный выигрыш, что вполне коррелирует с вышесказанным)

1. Пример на парадокс Элсберга: пусть есть 2 урны, в одной 50 белых и 50 чёрных, а в другой просто есть 100 каких-то шариков. Сначала вам дано задание вынять белый шарик, люди лезут в первую урну, потом их просят вынят чёрный - и они снова лезут в первую урну. Хоте их первый выбор урны предполагал, что в первой урне белых шариков больше, чем во второй (иначе зачем было брать именно первую урну), и тогда второй раз они должны были полезть рукой во вторую урну, но нет.
2. Выбирая между лотереей, в которой гарантированно дадут 3000$ и лотереей, в которой дадут 4000$ с вероятностью 80% или ничего. 80% людей выбирают первое.

Выбирая между лотереей, в которой 3000$ с вероятностью 25% и лотереей 4000$ с ероятностью 20%, 35% выбрали первое, а 65% выбрали второе.

Теперь проанализируем средний выигрыш для первого выбора и второго выбора: 3000 против 3200 vs 750 против 800.

Логично предположить, что человек либо выбирает то, где лучше средний выигрыш, либо выбирает то, где выше надёжность, однако на практике люди в первом выборе выбрали одно, а во втором выбрали дргое.

Этот эффект называется **эффектом определённости** (certainty effect).

Т.е. людям пресуще не рациональное поведение в соответствии с аксиомами теоремы фон неймана-...

1. Аналогично предыдущему случаю с certainty effect. Людям так же свойственно переоценивать важность малых (например, 0.001) вероятностей. - **overweighting of small probabilities**.
2. 1 выбор: что выбирете? “спасти 400 наверняка” или “не спасти никого с вероятностью ⅓ или всех 600 с вероятностью ⅔”.

2 выбор: что выбирете? “200 человек умрёт наверняка” или “спасти всех 600 с вероятностью ⅔ или никого”

Выборы абсолютно одинаковы, но в первом случае выбрают 9 к 17, а во втором 10 к 17.

Этот эффект называется **framing effect**.

**Эффект определённости (certainty effect)** - людей в среднем надёжность близкая к 1, привлекает больше обычного.

**Overweighting of small probabilities** - переоценка людьми малых вероятностей (например, 0.001)

**Framing effect** - отет людей в среднем зависит от того, ка поставить вопрос

## Скользящий контроль

**Скользящий контроль** или **кросспроверка** или **кроссвалидация** (**cross-validation, CV**) — процедура эмпирического оценивания обобщающей способности алгоритмов, обучаемых по прецедентам.

Фиксируется некоторое множество разбиений исходной выборки на две подвыборки: **обучающую** и **контрольную**. Для каждого разбиения выполняется настройка алгоритма по обучающей подвыборке, затем оценивается его средняя ошибка на объектах контрольной подвыборки. **Оценкой скользящего контроля** называется средняя по всем разбиениям величина ошибки на контрольных подвыборках.

Различные варианты скользящего контроля отличаются видами функционала качества и способами разбиения выборки.

Если выборка независима, то средняя ошибка скользящего контроля даёт несмещённую оценку вероятности ошибки. Это выгодно отличает её от средней ошибки на обучающей выборке, которая может оказаться смещённой (оптимистически заниженной) оценкой вероятности ошибки, что связано с явлением переобучения.

Скользящий контроль является стандартной методикой тестирования и сравнения алгоритмов классификации, регрессии и прогнозирования.

*Математические выкладки, о скользящем контроле можно прочесть в файле “4.1. Скользящий контроль - MachineLearning.pdf” или ссылке:*

*“*[*http://www.machinelearning.ru/wiki/index.php?title=%D0%A1%D0%BA%D0%BE%D0%BB%D1%8C%D0%B7%D1%8F%D1%89%D0%B8%D0%B9\_%D0%BA%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D1%8C*](http://www.machinelearning.ru/wiki/index.php?title=%D0%A1%D0%BA%D0%BE%D0%BB%D1%8C%D0%B7%D1%8F%D1%89%D0%B8%D0%B9_%D0%BA%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D1%8C)*”*

*Доверительного оценивания - вроде не было на лекциях*

Суть в том, что по результатам скользящей выборки мы получаем некоторое распределение качества для различных вариантов разбиения.

**Утверждение**: Если разбиения осуществлялись независимо и равновероятно то (N - количество разбиений, случайная величина функционала качества Q, Q(i) - упорядоченные по возрастанию функционалы качества разных разбиений)

1. с вероятностью t/(N+1) значение случайной величины Q не превосходит Q(N -t +1)
	1. можно взять частный случай t=1 и тогда можно будет посчитать верхнюю оценку, в частности для получения верхней оценки с надёжностью 95% достаточно взять N = 20 разбиений.
2. с вероятностью 2t/(N+1) значение случайной величины Q лежит в интервале [Q(t);Q(N-t+1)]
	1. можно взять частный случай t=1 и тогда можно будет посчитать оценку для интервала [Q(1);Q(N)], в частности для получения верхней оценки с надёжностью 95% достаточно взять N = 40 разбиений.

**Стратификация выборки** — это способ уменьшить разброс (дисперсию) оценок скользящего контроля, в результате чего получаются более узкие доверительные интервалы и более точные (tight) верхние оценки. Стратификация заключается в том, чтобы заранее поделить выборку на части (страты), и при разбиении на обучение длины m и контроль длины k гарантировать, что каждая страта будет поделена между обучением и контролем в той же пропорции m/k.

Стратификация классов в задачах классификации означает, что каждый класс делится между обучением и контролем в пропорции m/k.

Стратификация по вещественному признаку. Объекты выборки сортируются согласно некоторому критерию, например, по возрастанию одного из признаков. Затем выборка разбивается на k последовательных страт одинаковой (с точностью до 1) длины. При формировании контрольных выборок из каждой страты выбирается по одному объекту, либо с заданным порядковым номером внутри страты, либо случайным образом.

Разновидности скользящего контроля:

1. **Полный скользящий контроль** - т.е. строятся все возможные разбиения при некоторм фиксированном k (k - количество элементов для контроля, остальное - идёт в обучающую выборку)
	1. частный случай при k = 1 - “контроль по отдельным объектам” (leave-one-out CV) (Было показано, что контроль по отдельным объектам является ассимптотически оптиальным при некоторых условиях) (один из самых распросранённых случаев) (имеет средний уровень вычислительно сложности, потому что нужно переобучиться L раз (столько, сколько элементов в выборке))
	2. общий случай при k > 2 имеет количество комбинаций N = CLk - что слишком много и затрудняет применение метода полного скользящего контроля

Бывают исключения - например для метода k-ближайших соседей есть эффективная вычислительная формула, и поэтому полный скользящий контроль использовать вычислительно допустимо.

1. **Случайные разбиения** - разбиения выбираются равновероятно и независимо, именно для такого случая работают формулы описанные выше в утверждении. На практике эти формулы часто переносятся и на другие способы разбиения.
2. **Контроль на отложенных данных (hold-out CV)** - оценка скользящего контроля строится по одному случайному разбиению (N = 1). Недостатки:
	1. Приходится слишком много объектов оставлять в контрольной подвыборке. Уменьшение длины обучающей подвыборки приводит к смещённой (пессимистически завышенной) оценке вероятности ошибки.
	2. Оценка существенно зависит от разбиения, тогда как желательно, чтобы она характеризовала только алгоритм обучения.
	3. Оценка имеет высокую дисперсию, которая может быть уменьшена путём усреднения по разбиениям.

Следует различать скользящий контроль по отложенным данным и контроль по тестовой выборке. Если во втором случае оценивается вероятность ошибки для классификатора, построенного по обучающей подвыборке, то в первом случае для классификатора, построенного по полной выборке (то есть доля ошибок вычисляется не для того классификатора, который выдается в качестве результата решения задачи). <!!! что за муть тут написана?>

1. **Контроль по q блокам (q-fold CV)** - выборка разбивается на q блоков примерно одинаковой длинны, после этого каждый блок по очереди становится контрольной выборкой. Это компромисное решение между left-one-out, hold-out и случайным разбиениями. Обычно выборку разбивают на 10-20 блоков.
2. **Контроль по r\*q блокам (r\*q-fold CV)** - контроль ко q блокам повторяется r раз, каждый раз выборка случайным образом разбивается на q непересекающихся блоков.

Данный вариант является одним из стандартных и используется в системах WEKA и “Полигон алгоритмов”

Скользящий контроль в задачах прогнозирования.

В задачах прогнозирования, динамического обучения, обучения с подкреплением и активного обучения прецеденты изначально линейно упорядочены по времени их появления.

1. **Контроль при нарастающей длине обучения**

Выбираем некоторый “настоящий момент”, после чего, всё что было до него - это в обучающую выборку, а всё что после - в контрольную выборку. Также можно взять некоторую **величину задержки прогнозирования**, и в контрольную выборку попадут только те объекты, которые находятся после настоящего момента + величина задержки прогнозирования.

Так как длинна обучения со временем увеличиваться (всё больше объектов мы относим к обучающе выборке) - то точность прогнозов может улучшаться, но это плохо для оценивания качества алгоритма.

1. **Контроль при фиксированной длине обучения** - отличается от предыдущего лишь тем, что длинна обучения фиксируется, ограничиваясь лишь поседними m прецедентами.

Недостатки скользящего контроля:

1. Задачу обучения приходится решать N раз, что сопряжено со значительными вычислительными затратами.
2. Оценка скользящего контроля предполагает, что алгоритм обучения уже задан. Она ничего не говорит о том, какими свойствами должны обладать «хорошие» алгоритмы обучения, и как их строить. Такого рода подсказки дают, например, теоретические оценки обобщающей способности.
3. Попытка использовать скользящий контроль для обучения, в роли оптимизируемого критерия, приводит к тому, что он утрачивает свойство несмещённости, и снова возникает риск переобучения.
4. Скользящий контроль дает несмещенную точечную, но не интервальную оценку риска. В настоящее время не существует методов построения на основе скользящего контроля точных доверительных интервалов для риска, то есть математического ожидания потерь (в частности, вероятности ошибочной классификации).

**Применения скользящего контроля**:

На практике скользящий контроль применяется для оптимизации некоторых критически важных параметров, как правило, определяющих структуру или сложность используемой модели алгоритма, и имеющих относительно небольшое число возможных значений.

Примеры:

* Выбор модели алгоритмов из небольшого множества альтернативных вариантов
* Оптимизация параметра регуляризации, в частности:
	+ параметра регуляризации в гребневой регрессии
	+ параметра сокращения весов (weight decay) в нейронных сетях
	+ параметра C в методе опорных векторов
* Оптимизация ширины окна в методах
	+ парзеновского окна
	+ ближайшего соседа
	+ ядерного сглаживания
* Оптимизация числа нейронов в скрытом слое многослойной нейронной сети.
* Выбор информативного набора признаков.
* Редукция решающего дерева.
* Структурная минимизация риска.

## Особенности поиска кратчайшего пути. Алгоритм Дейкстры

Постановка задачи:

1. Задан неориентированный или ориентированный граф G=(V,E).
2. Заданы длины рёбер ℓ(v,w)≥0. Если ребра нет, то можно считать ℓ(v,w)=+∞.
3. Длины рёбер могут не удовлетворять неравенству треугольника (не метрика).
4. Длина пути ℓ(P) равна сумме длин ребёр.
5. Требуется найти кратчайший путь между заданными вершинами s и t.
6. Длины dist(s, t) кратчайших путей – метрика.

Практические требования задачи - миллионы вершин и рёбер, задачу решать надо за миллисекунды (а значит можем просмотреть лишь пару сотен вершин).

*Современные алгоритмы точного поиска кратчайшего пути:* (!!! пояснения к тому, что ниже?)

* *Arc flags [Lauther 04, Kohler et al. 06].*
* *A∗ with landmarks [Goldberg & Harrelson 05].*
* *Reach [Gutman 04, Goldberg et al. 06].*
* *Highway [Sanders & Schultes 05] and contraction [Geisberger 08] hierarchies.*
* *Transit nodes [Bast et al. 06].*
* *Combinations (pruning and directing algorithms).*

Метод сканирования (1963):

1. Для каждой вершины v храним «метку» (расстояние) d(v) и статус S(v) ∈ {unreached, labeled, scanned}.
2. Сначала все вершины имеют статус unreached, расстояние от источника до них d(v) = +∞.
3. Когда d(v) уменьшается, v получает статус labeled.
4. Начинаем с уменьшения d(v) для некоторых v.
5. Пока есть вершины labeled, выбрать одну из них и обработать. Вершина получает статус scanned.
6. Обработка отмеченной вершины v: для каждой дуги (v,w), если d(w) > d(v)+ℓ(v,w), то d(w) = d(v)+ℓ(v,w).

Начинать можно по-разному.

Выбирать можно по-разному.

Критерий останова надо аккуратно определять.

**Алгоритм Дейкстры** (1959):

1. Ищем путь из вершины s в вершину t
2. Начинаем обработку с вершины s: d(s) = 0
3. На каждом шаге обрабатываем вершину с минимальной меткой (!!!вообще в алгоритме дейкстры обрабатываются все вершины, не только та, у которой метка меньше. Так что у меня большие сомнения по поводу того, что тут написано. Откуда это взято?)
4. Останавливаемся, когда для обработки выбрана вершина t
5. Сложность линейна относительно размера просмотренного подграфа.

Модификации алгоритма дейкстры:

1. **Обратный алгоритм Дейкстры** - начинаем сканировать от финальной вершины t (переориентируем ребра ориентированного графа)

(как и в обычном алгоритме дейкстры на примере просмотрены почти все вершины)

1. **Двунаправленный алгоритм Дейкстры** - одновременно начинаем искать от вершин s и t. Проблема - нетривиальный критерий останова:

Пусть Ds и Dt – уже достигнутый минимум для выбираемых меток от соответствующего источника. Растут от нуля. Никакая unreached/labeled далее не получит менее этих значений. Никакая scanned уже не превышает их. Когда появляется первая scanned\_s И scanned\_t вершина (вершина, просканированная обоими алгоритмами), это ещё не остановка.

Оценка длины кратчайшего пути через вершину v:

* Scanned\_s И scanned\_t => Окончательно d(s,v)+d(t,v) (≤ Ds+Dt).
* не scanned\_s И не scanned\_t => Достижимо ≥ Ds+Dt.
* Только scanned\_s => Достижимо ≥ d(s,v)+Dt.
* Только scanned\_t => Достижимо ≥ Ds+d(t,v).

M – длина уже известного кратчайшего пути. Сначала M=+∞.

Условие отсечения вершины: через неё невозможен путь < M.

Условие останова: все необработанные вершины отсечены.

(на примере в сумме просмотрена лишь четверть вершин)

Для согращения перебора нужно отбрасывать мнрожества вершин, которые нам не нужны, для этого нужны хорошие критерии отсечения.

Критерии отсечения (доказано, что все дают точные результаты):

1. **Highway/contraction hierarchies (CH)**: кратчайший путь сначала выходит на всё более крупные магистрали, а в конце сходит на всё более мелкие.

Проводим предварительную обработку, эвристически удаляя ненужные вершины, и вставляя потерянные кратчайшие пути (некоторые пути сохранять не придётся).



1. Эвристически упорядочиваем вершины.
2. Дуги направляем от младших к старшим вершинам.
3. Пополняем граф дугами для сохранения кратчайших путей.

Идея: Старшие вершины важнее (“глобальнее”). Запрос выполняется двунаправленным Дейкстрой, каждый поиск развивается только от младших к старшим вершинам.

Пусть (V, E∪E+) – граф с дополнительными дугами. Вершины упорядочены. Пусть Gf – орграф с дугами (v,w): (v,w)∈E∪E+, v<w, а Gr – орграф с дугами (w, v): (v,w)∈E∪E+, v>w.

По заданным s и t запускаем модифицированный двунаправленный алгоритм Дейкстры, причем поиск вперёд идет в графе Gf, а поиск назад - в Gr. (Оба поиска развиваются от младших к старшим.)

При останове dist(s, t) = minv(d(s, v)+d(v, t)).

*Для графов дорог предложены эвристические способы упорядочивания, которые работают быстро и добавляют относительно мало новых дуг. Эксперименты показывают, что для графов дорог даже в худшем случае просматриваются очень маленькие подграфы.*

1. **Reach pruning (RE)**: малозначительные перекрестки вдали от начала/конца пути можно игнорировать.

Рассмотрим вершину v, которая делит путь P на части P1 и P2.

r(P,v) := min(ℓ(P1), ℓ(P2)) – расстояние до ближайшего конца для данного пути P.

r(v) := max(r(P,v)) – максимум по всем кратчайшим путям P, проходящим через v.

Reach – «радиус действия», «охват». Если s и t находятся от w далее r(w), то кратчайший путь точно не проходит через w.

Сокращение перебора в алгоритме Дейкстры:

если r(w) < min(d(s, v)+ℓ(v, w), LB(w, t)), то отсечь вершину w.

Оценку снизу LB(w, t), например, «даром» имеем при использовании двунаправленного алгоритма Дейкстры.



Добавление замыканий (делается итерационно несколько раз):



В этом графе большие Reach, трудно отсекать.

Добавим вспомогательное «замыкание».

Из двух путей одной длины предпочитаем пути из меньшего числа дуг. Reach снизился, легче отсекать.



Небольшое число замыканий обычно позволяет существенно понизить значения Reach для многих вершин.

=> Запускаем двунаправленый алгоритм Дейкстры.

1. **Transit nodes (TN)**: Все дальние кратчайшие пути в/из некоторого региона обязательно проходят через небольшое подмножество вершин.
* Все кратчайшие маршруты в/из региона проходят только через точки доступа.
* Карты обычно удаётся разбить на регионы так, что в среднем у региона 10 точек доступа.
* Пар точек доступа из разных регионов немного, пути можно запомнить.
* Внутри каждого региона быстро ищем путь от заданной точки до всех точек доступа.
* Поиск пути между точками из разных регионов сводится к просмотру таблицы размера 10х10: D(i,j)=dist(s,TN1i)+dist(TN1i,TN2j)+dist(TN2j,t).

# Глава 3. Часть от Сенько. (полным разумным конспектом являются слайды Сенько, в этой главе лишь некоторая вытяжка)

## Прогнозирование по прецедентам

(Если возможность сделать искабельную версию его лекций? ctrl+f работает, но через раз, а точнее через десять =\ )

Задача прогнозирования решается для явления F.

Множество объектов, которые потенциально могут возникать в рамках F, называется **генеральной совокупностью**.

Генеральная совокупность обычно рассматривается как множество элементарных событий, на котором заданы алгебра событий и вероятностная мера, т.е. генеральная совокупность рассматривается как вероятностное пространство.

Каждому объекту из множества объектов генеральной совокупности - (X1, …, Xn) соответствует прогнозируемая величина Y, она известна не для всех и задача прогнозирования - построение алгоритма, вычисляющего соответствующие Y. Зная некоторые из прогнозируемых величин для некоторых объектов.

**Обучающая выборка** - выборка прецедентов из генеральной совокупности с известными значениями прогнозируемой величины.

Обычающая выборка обычно рассматривается как независимая выборка объектов из генеральной совокупности.

**Методы машинного обучения** - методы, основанные на обучении по прецедентам.

Прогнозируемая величина может:

1. принимать значения из отрезка непрерывной оси
2. принимать значения из конечного множества - для данного случая задача называется **задачей распознавания**.
3. являться кривой, описывающей вероятность, возникновения некоторого критического события до различных моментов времени.

## Способы поиска закономерностей

Один из способов поиска закономерностей - это поиск в некотором априори заданном семействе алгоритмов наилучшего. (тут получается, что поиск закономерностей == поиску наилучшего алгоритма в семействе? разве поиск закономерностей - это не разбиение на семейства алгоритмов?)

Один из способов поиска алгоритма - **минимизация функционала эмпирического риска** на обучающей выборке.

**Средний риск** = $∭\_{x\_{i} j}^{}(j-A(x, θ))^{2} dP(x\_{i}, j)$ подинтегральное выражение часто обозначают Q(x, j, A(θ))

Задача в том, чтобы минимизировать средний риск относительно параметра θ. При решении задачи получаем качество (величину интеграла) и надёжность (вероятность по исходной информации найти оптимальное решающее правило).

**Метод минимизации эмпирического риска** – допустим определена функция среднего риска, допустим количество классов, среди которых нужно выбрать - конечно*, есть выборка объектов из классов, как они выбирались:*

*есть совместное распределение, из которого выбирались объекты одним из 2-х способов:*

1. *Сначала объект, потом класс (модель учитель)*
2. *Сначала класс, а потом объект*

После этого эмпирический риск – это средний риск, но интеграл по всем значениям заменён на сумму по выбранным объектам и классам. $P\_{эмп}(θ)=\frac{1}{m}\sum\_{i=1}^{m}Q(s^{i}, c^{i}, A(θ))\rightarrow min\_{θ}$

Q - называется **величиной потерь** (он просто отражает, на сколько наш алгоритм промазал мимо правильного значения). Величину потерь можно задавать разными способами, например, как квадрат или модуль ошибки, или как булево значение - есть/нет ошибки.

**Метод наименьших квадратов** - это метод минимизации эмпирического риска, для выбора корректного алгоритма из некоторого множества, где в качестве функции потерь взят квадрат ошибки.

*Примером семейств алгоритмов, для которых нужно провести метод минимизации эмпирического риска может быть семейство линейных функций, или семейство кубических функций, ...*

## Обобщающая способность

**Обобщающая способность** - точность алгоритма прогнозирования на всевозможных новых, не использованных для обучения объектах, т.е. это точность по всей генеральной совокупности.

**Мерой обобщающей способности** явяется математическое ожидание потерь по всей генеральной совокупности.

Цель задачи прогнозирования - максимизация обобщающей способности. Но нужно понимать, что подсчитать её достаточно проблематично, т.к. обобщающая способность определяется через множество всей генеральной совокупности.

**Эффект переобучения** - это эффект, при котором расширение модели алгоритма и увеличение её сложности приводит к повышению точности аппроксимации на обучающей выборке, т.е. к снижению эмпирического риска, однако при этом не приводит к улучшению обобщающей способности, или приводит к её ухудшению.

Пример:

можно пытаться аппроксимировать кубическим полиномом и получить красивую кривую, а можно взять полином более высокого прядка и кривую будет шатать, при этом она пройдёт через все точки обучающй выборки, но на генеральной совокупности будет хуже.

<картинки на слайде 22 из лекций 1>

Верхний предел точности:

E(Y|X) - среднее по всей совокупности

E(Y -A(X))2 = ошибка алгоритма А = E(Y - E(Y|X) + E(Y|X) - A(X))2 =

= ∑{ E(Y - E(Y|X))2 + 2\*E(Y - E(Y|X))[=0, т.к. EY = P(X)\*E(Y|X) и EE(Y|X) = P(X)E(Y|X)] \* E(E(Y|X) - A(X)) + E(E(Y|X) - A(X))2 } =

= ∑{ E(Y - E(Y|X))2 + E(E(Y|X) - A(X))2 }

Таким образом, алгоритм, который вычисляет отклонение Y от X (т.е. A(X) = E(Y|X)) является лучшим и имеет ошибку равную выражению выше.

<подробнее см вычисления на слайде 24 лекции 1>

## Байесовский классификатор

Байесовский классификатор имеет наименьшую ошибку распознавания, если предположить, что элементы выборки имеют всю доступную информацию о распределении объектов по классам.

(!!! какую именно “доступную информацию”?, 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)

**Байесовский классификатор** относит объект x к тому классу, в котором максимизируется условная вероятность P(Ki, x).

(мы при этом знаем долю объектов принадлежащих каждому из классов K1, … KL)

Задача поиска минимума ошибки сводится к задаче линейного программирования. <вычисления на слайде 28 лекции 1>

Задача линейного программирования даёт некоторое множество, и одна из вершин является решением (минимизацией), при этом если решение задачи достигается в одной точке, то объект нужно будет классифицировать принадлежащим к этому классу, если в нескольких точках, то можно выбрать любой класс, в которых достигается экстремум задачи лп (лин. прогр.).

*Для вычисления необходимых для Байесовского классификатора условных вероятностей можно использовать метод максимального правдоподобия (ММП).*

*функция максимального правдоподобия = произведение плотности вероятности на объектах обучающей выборки.*

*Важно, что выборка у нас - это независимые одинаково распределённые случайные величины.*

*Оценка для θ у функции максимльного правдоподобия совпадает с задачей линейного программирования выше, небходимо функцию правдоподобия максимизировать.*

Проблема байесовского классификатора: на практике часто неизвестны ни общий вид распределения, ни значения их конкретных параметров.

Методы прогнозирования:

* Статистические методы
* Линейные модели регрессионного анализа
* Различные методы, основанные на линейной разделимости
* Методы, основанные на ядерных оценках
* Нейросетевые методы
* Комбинаторно-логические методы и алгоритмы вычисления оценок
* Алгебраические методы
* Решающие или регрессионные деревья и леса
* Методы, основанные на опорных векторах

*Байесовское решающее правило является частным случаем байесовского классификатора, для 2-х классов*

***Байесовское решающее правило*** *(является оптимальным) P(j=1 | x) / P (j=2 | x) <?> P1 / P2 – где P(j=1 | x) это вероятность того, что данный элемент x будет классифицирован как 1, а P1 – вероятность что мы держим в руках элемент из класса 1.*

*(выводится из среднего риска для случая с 2-мя классами)*

**Наивный Байесовский классификатор**. Это байесовский классификатор с предположением, что все переменные независимы. (благодаря этому условная вероятность распадается на произведение условных вероятностей отдельно по каждому из признаков, что очень помогает в вычислениях) (точность около 0,67, т.е. так себе)

## Байесовский классификатор для многомерного нормального распределения

<Сенько, лекция 4, слайды 1 (частично были пропущены)>

Если мы работаем с наивным байесовским классификатором и предположим, что случайные величины распределены по нормальному распределению, то объект с признаковым описанием x будет отнесён построенной аппроксимацией байесовского классификатора к классу, для которого оценка gi(x) является максимальной.





Следует отметить, что построенный классификатор в общем случае является квадратичным по признакам. Однако классификатор превращается в линейный, если оценки ковариационных матриц разных классов оказываются равными.

## Линейный дискриминант Фишера (ЛДФ)

Задача распознавания 2-х классов.

Метод основан на том, что в многомерном признаковом пространстве ищется направление w, чтобы средние значения проекции на него объектов обучающей выборки из классов максимально различались.

<Сенько лекция 4 недоделана>

## Эмпирические методы оценки обобщающей способности

**Контрольная выборка** - часть объектов из исходной выборки, по которой будет оцениваться обобщающая способность алгоритма. Она не должна входить в обучающую выборку.

**Функционал риска** - $P\_{эмп}(θ)=\frac{1}{m}\sum\_{i=1}^{m}Q(s^{i}, c^{i}, A(θ))\rightarrow min\_{θ}$, где сумма идёт по элементам контрольной выборки.

Подробнее про скользящий контроль читай параграф про [скользящий контроль](#h.z1487fo39g8)

Несмещённость оценки скользящего контроля означает, что математическое ожидание от величины потерь при обучении на всей выборке равна мат ожиданию по всем итерациям скользящего контроля (т.е. среднее значение по всем m итерациям скользящего контроля) от математического ожидания величины потерь при обучении на m-1 элементе выборки.

<слайд 44 лекции 1>

## Линейная регрессия

## Линейная регрессия

**Линейная регрессия** - когда функция классификации для объектов генеральной совокупности ищется среди класса линейных функций.

Традиционный способ поиска коэффициентов линейной регрессии является **метод наименьших квадратов** (т.е. когда для выборки минимизируется срений риск, а в качестве функции величины потерь - берётся квадрат разности)

Необходимое условие минимума для уравнений всегда одно - равенство нулю производных.

На конце линейных функций линейной регрессии добавляется некоторая ε, которая является случайной величиной, которая нормально распределена имеет нулевое мат. ожидание на выборке и её дисперсия не зависит случайных величин - выборки.

В этом случае метод наименьших квадратов совпадает с методом максимлаьного правдоподобия.

Метод ММП (метод максимального правдоподобия) позволяет восстанавливать плотность распределения вероятностей по случайным выборкам, если общий вид плотности вероятностного распределения известен.

ММП заключается в том, чтобы найти такое θ, при котором функция максимального правдоподобия максимизируется, если взять !!! (это же логическая ошибка, как мы можем говорить, об эквивалентности ММП и МНК, если мы сами по ходу решения подставили линейную функцию в ММП) (важно не то, что мы подставили линейную функцию, а то, что мы предположили, что у линейной функции на конце висит случайная величина с нормальным распределением, так что всё норм) функцию линейного вида из линейной регрессии, то записа функцию МП и взяв логарифм, видно, что её точка максимума достигается тогда же, когда достигается точка максимумав методе МНК (наименьших квадратов).

Для одномерной модели линейной регрессии (у объекта лишь один признак):

Коваривация показывает, как зависит Y от X, если она >0, то функции одновременно растут (или падают), если <0, то одна растёт, а другая падает.

Коэффициент корреляции Пирсона = $\frac{cov(Y, X)}{\sqrt{D(X)D(Y)}}$

+- 1 означаетлинейную зависимость между X и Y

0 означает отсутствие линейной зависимости

интересный вывод, для линейной регрессии b0 + xb1

b1 = cov(Y, X) / D(X)

b0 = y - b1\*x

<не понимаю, к чему на слайдах с 10 лекции 3 рассказывается про дисперсию и ковариацию - вывод-то какой?>

Многомерная регрессия:

При многомерной регрессии строят **матрицу плана** - строка матрицы - это вектор значений переменных из обучающей выборки, и первый столбец - забит “1”.

тогда можно записать y = βX + ε, где все буквы - это вектора, а X - это матрица плана.

После взятия производной уравнения выглядят так: -2XTyT + 2XTXβT = 0

Решение для βT = (XTX)-1 \* XTyT

В данном случае, чтобы решение существовало, нужно чтобы определитель матрицы был не равен нулю, т.е. чтобы m-мерные вектора были линейно независимы.

Если ветора будут сильно коррелированны, то значение определителя будет близко к нулю, и такое решение обычно является не устойчивым к малым изменениям исходных данных. Это свойство называется **мультиколлинеарностью**.

На практике высокая устойчивость достигается только когда число объектов в выборках по крайней мере в 3-5 раз превышает число переменных.

Регрессионный анализ в изначальном своём виде не позволяет отбирать признаки.

Гребневая (ridge)

Лассо слишком жёстко отбирает признаки

<Сенько слайды 18 - 30 лекции 3> - там много выкладок, которые приводят к выводам связанным, с кореляцией, ковариацией, ...

## Обобщённая ошибка

**Обобщёные потери** - математическое ожидание по всевозжным обучающим выборкам от матожидания по величине потерь.

**Обобщённая квадратичная ошибка** - обобщённая потеря, в случае когда в качестве величины потери используется квадрат ошибки.



<Сенько слайд 23, потом шло много вычислений, которые я скипаю>

В итоге:



Структура обобщённой ошибки:

1. **Шумовая компонента N** - является минимально достижимой квадратичной ошибкой прогноза, которая не может быть устранена с использованием только математических средств
2. **Составляющая сдвига (Bias)** - высокое значение компоненты сдвига соответствует отсутствию в модели алгоритмов, достаточно хорошо аппроксимирующих объективно существующую зависимость Y от переменных X1, …, Xn.

Составляющая сдвига может быть снижена, например, путём расширения модели за счёт включения в него дополнительных более сложных алгоритмов, что обычно позволяет повысить точность аппроксимации данных.

1. **Дисперсионная составляющая (Variance)** - характеризует неустойчивость обученных прогнозирующих алгоритмов при статистически возможных изменениях в обучающих выборках.

Дисперсионная составляющая возрастает при небольших размерах обучающей выборки.

Дисперсионная составляющая может быть снижена путём выбора сложности модели, соответствующей размеру обучающих данных.

**Bias-Variance дилема**: 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.

## Способы регуляризации линейной регрессии

Регуляризация используется для улучшения устойчивости в регрессионном анализе для частного случая - МНК (Метод Наименьших Квадратов).

Суть метода регуляризации: включение дополнительной шумовой компоненты в исходный оптимизируемый функционал.

Типы регуляризаций:

1. **регуляризация по Тихонову** - добавление штрафной компоненты к оптимизируемому функционалу (по формулам там была модификация значений признаков объектов)
2. **гребневая регрессия (ridge)** - добавляет сумму квадратов регрессионных компонентов к функционалу (это приводит к положительности дискриминанта и улучшается устойчивость)
3. **метод лассо** - добавляет модули регрессионных компонент к функционалу (делает отбор переменных (т.к. некоторые регрессионные переменные приравниваются к 0) и применение на маленьких выборках возможным) (при высокой корреляции некоторых переменных, на практике метод лассо ухудшает свои показатели)
4. **эластичная сеть** - добавление суммы квадрата регрессионных компонент с коэффициентом θ и модуля регрессионных компонент с коэффициентом (1-θ) (вбирает все лучшее из первых двух).



## Задачи прогнозирования

## Нейронные методы, персептрон Роззенблатта

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 персептронов 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

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

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

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

Настройка параметра происходит по обучающей 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 персептрона:

1. Случайно выбирается нулевое приближение вектора весовых коэффициентов
2. Преобразованные описания обучающей выборки последовательно передаются на вход персептрона
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 персептрона построит линейную гиперплоскость, разделяющую объекты двух классов за конечное число шагов.

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

**Многослойный персептрон** - 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 соблюдаются 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 конфигурация 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 персептрон 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 фиксированы). 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 - параметр, задающий размер каждого шага

Алгоритм обучения:

1. Выбираем количество слоёв и количество нейронов в слоях. Присваиваем весам случайные значения. Дальше подаём последовательно объекты обучающей выборки (их векторные описания) и производим 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 протяжении нескольких последних итераций оказывается меньшим некоторого порогового значения
	3. общее время обучения превышает допустимый предел

# Глава X. Не обработано, но есть полезные записки с лекций

## Прочее

Покомпонентная ошибка = шум + variance + bias

Шум – отклонение от условного мат ожидания.

Varienсe – на сколько варьируется прогноз в каждой точке

Bias – отклонение в каждой точке наилучшего прогноза от среднего по выборке.

Регуляризация увеличивает устойчивость, т.е. уменьшает variance

Коллективное решение снижает и variance и bias

Теорема. Пусть есть конечная модель алгоритмов (нужно выбрать оптимальный параметр) мощности N, пусть среди алгоритмов есть корректный. Пусть мы находим алгоритм методом минимизации эмпирического риска. Тогда С вероятностью (1 – η) можно утверждать, что вероятность ошибочной классификации с помощью полученного алгоритма составляет величину меньшую ε, если длинна обучающей последовательности m не меньше чем m >= ceil( (ln N – ln η) / -ln(1-ε) )

Из этой формулы следует **проблема переобучения** (**overfit**) – когда точность падает с увеличением выборки.

## Средний риск, Байесовское решающее правило

Часто часть из q1, ..., qn (функционалы качества) идут в "target performance metrics" (целевой функционал качества), а часть qi идут в множество "constraints" (ограничения).

## Метод сравнения с эталоном, метод минимизации эмпирического риска, верхний предел точности

**Выборка** – набор объектов, для которых всё известно.

**Генеральная совокупность** – всё множество значений, на которых мы обучаемся.

**Корректный алгоритм** – никогда не ошибается на генеральной совокупности.

**Метод сравнения с эталоном** – для каждого класса есть представляющий его элемент, тогда для классифицируемого объекта нужно посчитать расстояние до каждого и выбрать класс, до представителя которого расстояние меньше.

**Обобщающая способность** – это точность на генеральной совокупности.

**Проблема обобщающей способности** – проблема того, чтобы объект оптимизированный на выборке работал на всём множестве возможных объектов.

Способы оценить обобщающую способность:

1. Разбить обучающую выборку на 2 части, после чего на первой части обучиться, а на второй части проверить себя (потом можно попытаться сделать наоборот)
2. **Скользящий контроль** – выбрасываем объект из выборки, на остальных обучаемся, потом на этом объекте проверяемся, и так по очереди с каждым объектом.

## Задача линейной регрессии (почему бы не перенести её в 3.6?)

**Задача линейной регрессии** – пусть есть обучающая выборка. При обучении, нужно чтобы хотя бы на обучающей выборке алгоритм работал хорошо.

Q(x, j, A(θ)) = Q(St, b0, b1) = 1/m \* sum(1<j<m, (yj – b0 – xjb1)2), где b1 = (cov (Y,X | St) / D(X | St)), b0 = Y – b1X

*??? это случайно не метод наименьших квадратов?*

Свойства ковариации: если ковариация >0, то Y растёт вместе с X, если <0, то Y убывает, когда X растёт, а если =0, то величины X и Y не связаны.

**Коэффициент корреляции Пирсона** = cov (X, Y) / (D(X)\*D(Y))^(1/2)

Если = +- 1, то это значит, что величины связаны линейно, = 0 означает отсутствие линейной связности.

Размытое облако вокруг прямой – это примерно +-0.5

**Метод максимального правдоподобия**.

Функция максимального правдоподобия – произведение плотности вероятности.

Берётся функция макс правдоподобия, после чего логарифмируется и ищется максимум.

**Матрица плана** – матрица где строка отражает элемент выборки. При малом определителе матрицы, получается неустойчивость, чтобы от неё избавиться, можно либо убить какой-нибудь столбец или строку, либо применить метод регуляризации.

**Метод регуляризации**. (суть в том, чтобы вместо исходного функционала взять похожий).

**Эластичная сеть**. (один из лучших способов изменения функционала) (быстрое обучение)

Свойства оптимальных регрессий:

1. см. книжку/конспекты

//================================================================================================

## ? ?

**Байесовский классификатор**.

Относит объект в тот класс, для которого условная вероятность P(класс i | x) максимальна. (при решении как правило нужно считать по формуле Байеса)

**Наивный Байесовский классификатор**. Это байесовский классификатор с предположением, что все переменные независимы. (благодаря этому условная вероятность распадается на произведение условных вероятностей отдельно по каждому из признаков) (точность около 0,67, т.е. так себе)(перенесено в раздел 3.4, думаю, можно удалять отсюда)

**Линейный дискриминант Фишера**. Суть в том, чтобы найти такое направление, в котором проекция на него классов отличается больше всего. Алгоритм для 2-х классов, если классов больше одного, то надо по очереди K1 vs K2, K3, … KL, потом K2 vs K3, K4, …

**Логистическая регрессия**.

Логистическая функция = 1/(1 + ez)

Линейная комбинация признаков z = b0 + b1x1 + … + bnxn

Цель в подборе bi так, чтобы для класса K1 z стремилось к 1, а для класса K2 z стремилась к 2. И

P(K | x) = логистической функции с линейной комбинацией признаков.

Для подбора используется метод максимального правдоподобия (см. слайды)

**k ближайших соседей**. P(Ki | x) оценка ведётся по ближайшей окрестности точки x.

Берём k наиболее похожих, и берём их доли и берём тот класс, который больше представлен.

Критерий Неймана-Пирсона.

//================================================================================================

**ROC – анализ**.

Суть в том, что структура любого алгоритм распознавания = R (распознающий оператор) \* C (решающее правило)

R вычисляет «вероятность» (это любое число) для каждого класса, что объект лежит в этом классе.

**Метод линейной машины**.

2 стадии: вычисление оценок для класса, + принятие решения.

**Тривиальное решающее правило**: объект будет отнесён к классу Ki если оценка за него максимальная.

Из правила вытекает система уравнений между оценками всех классов, Система в общем случае может быть решена через релаксационный алгоритм.

//================================================================================================

**Метод опорных векторов**.

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 спосбности. Процесс называется «**Подбор ядра**»).

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

//================================================================================================

## Нейронные сети, решающие деревья

**Перцептрон Розенблатта** – это нейрон с разными входами, активационной функцией, вычисляющей результирующее значение, которое может разветвляться в другие нейроны.

Пример активационной функции: 1 в случае, если сумма входов >0, -1 в случае, если сумма входов <0.

Процесс обучения: при правильной классификации выдаваемое значение не трогаем, при неправильной, выдаваемое значение корректируем на некоторую дельта. Вопрос правильной классификации может например проверяться на основе умножения значения перцептрона на значение свойства выборки.

Рекурсивно обучаем есть, пока не:

1. достигается полное разделение объектов из классов.
2. Повторение итерации не приводит к улучшению разделения классов
3. Просто превышен лимит итераций.

Многослойный перцептрон состоит из слоя реагирующих нейронов, слоя внутренних нейронов и слоя рецепторов-нейронов. Активационная функция может быть такой же как раньше, но только сумма входов сравнивается с “b”.

У многослойной сети есть 2 недостатка: её тяжелее обучать, и она неустойчива.

//================================================================================================

**Решающие деревья** - дерево иерархической системы вопросов.

Необходимо снижать индексы неоднородности:

Индекс Джини / энтропийный индекс / индекс ошибочной классификации.

Сначала придумывается список порогов (вопросов для узлов дерева)

На каждом шаге выбирается вопрос для узла так, чтобы индекс неоднородности его потомков был минимален.

Остановиться при построении дерева можно либо когда

1. В каждом листе не более K классов
2. Выборка в листе уже не репрезентативна

Дерево не следует достраивать до конца, чтобы не случилось переобучение.

Репрезентативность можно определять с помощью статического теста (хи-2, Якобсона)

1. Делаем контрольную выборку и строим дерево до тех пор, пока для построения и контрольная выборка не разойдутся во мнениях.
2. Прекратить построение, если индекс неоднородности почти не уменьшается.

«Эффект горизонта» - когда нельзя построить дерево делая выбор по очереди сначала по x потом по y, или наоборот, нужно именно сразу. Чтобы с этим совладать надо строить «глубокое» дерево, а потом его подрезать.

Окончательные решения в деревьях обычно всегда плохие.

//================================================================================================

## ? ?

**Поиск коллективных решений / голосование / взвешенное голосование**.

**Ансамбль** – множество алгоритмов, используемых для коллективного решения.

Т.е. есть группа прогнозирующих алгоритмов, на основании решения которых принимается окончательное решение. При этом алгоритмы могут учитываться по-разному.

Есть доказательство, что ошибка взвешенного решения не больше суммы взвешенных ошибок.

//================================================================================================

**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.

**Алгоритм типа Кора** – когда ищется не только набор признаков, но набор признаков со своим описанием.

**Представительный набор** – набор описания (зафиксированные признаки), которого присутствуют только в одном классе.

Из существования теста следует существование набора, но не наоборот.

**Голосование по закономерности** – если в классе есть представительный набор, то проверяется, сколько совпадений между классифицируемым объектом и представительным набором.

Для перехода к непрерывным признакам от дискретных, достаточно для непрерывных просто ввести некоторые границы, задав интервалы.

//================================================================================================

## ? ?

**Алгоритмы вычисления оценок**.

1. Выделяется множество подмножеств, которое называется «**система опорных векторов**».
2. Функция близости сравнивает 2 объекта по некоторому опорному множеству. (например =1 если все признаки совпали, или =0 если нет (а может учитываться какой-то порог, …))

**Логическая закономерность** – это некоторый параллелепипед в пространстве, который содержит в себе объекты только одного класса. Задача – найти логические закономерности.

По логической закономерности можно выбрать класс, которому принадлежит классифицируемый объект. Если объект попал в несколько закономерностей, то выбираем ту, чья доля пересечения больше. Закономерности при этом можно взвешивать (например, ставить вес, пропорционально количеству входящих объектов).

Может возникать проблема неустойчивости.

**Статистически взвешенные синдромы** – суть в построении оптимального разбиения так, чтобы наилучшим образом разделить объекты. (за счёт перебора всевозможных границ, чтобы выполнялся некоторый функционал (см. слайды/лекции))

**Метод комитетов** – пусть есть некоторый набор линейных функций (классификаторов). Объект следует отнести к тому классу, к которому его отнесёт большинство классификаторов.

Теоретически доказано существование комитета для непротиворечивых данных.

//================================================================================================

## Способы генерации ансамблей из одной выборки

**Метод Бэггинга** – bootstrap aggregating – случайным образом генерируем новую выборку на основе начальной удалением случайных элементов и добавлением дублей других случайных элементов. Так получаем новые выборки и по ним строим новые классификаторы для ансамбля.

**Метод Бустинга** – случайное изменение весов некоторых объектов в выборке, тем самым получение новых классификаторов для ансамбля.

Эти методы снижают ошибку распознавания.

**Монотонный логический классификатор**. На входе: обучающая выборка, список классов, ансамбль классификаторов. Для конкретного классифицируемого объекта выбирается число k (как параметр алгоритма), после этого находятся какие-то любые классификаторы из ансамбля в количестве k штук, которые классифицируют этот объект, как объект конкретного класса. Если после этого есть ещё хотя бы один классификатор, который тоже классифицирует этот объект к тому же классу, то считаем, что классификация удалась.

(фиг знает, чем это отличается то того, чтобы взять сразу k+1 классификатор, мы так Сенько и не поняли)

**Алгебраическая коррекция** – вспомним, что классификация это ROC, т.е. R \* C, где R – это оператор, ставящий каждому прецеденту некоторую характеристику – вероятность отнесения к каждому из классов. А С – это оператор, решающий по характеристикам к какому классу, будем принадлежать.

Журавлёв предложил ввести операции:

1. Почленное умножение матриц
2. Посленное сложение матриц
3. Почленное умножение на число

Тогда S (матрица, где столбец – прецендент из обучающей выборки) \* R = Г для каждого классификатора нашего ансамбля можно рассматривать как набор. Для которого существует замыкание из множества всех матриц Г, которые получаются из 1-3 пункта. Тогда существует в этом наборе Гcorr - **корректная матрица** (матрица, для которой корректно работает стандартное решающее правило).

В итоге надо в замыкании найти корректную матрицу.

//================================================================================================

## 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))

//================================================================================================

## Кластерный анализ.

Задача разбить выборки на группы. (подразумевается, что есть метрика)

**Метод внутригрупповых средних**.

Пусть сначала разбивка произвольная и задано некоторое предполагаемое число кластеров.

Берём центр каждого кластера и относим объекты к тому кластеру, к чьему центру они ближе.

И так повторяем пока не получится, на очередном шаге никаких изменений не требуется.

**Иерархическая кластеризация**.

На первом шаге – все объекты кластеры. На каждом следующем шаге объединяются 2 ближайших кластера в один кластер.

Повторяем пока не выполнится какое-нибудь условие, например,

1. Собралось нужное число кластеров
2. Эксперту понравилась кластеризация
3. При дальнейшей кластеризации, кластеры будут терять компактность

**Метод Фарэля**.

Берём шар радиуса r и в точке x, вычисляем среднее геометрическое для всех попавших объектов в шар, двигаем туда новый центр шара и повторяем.

# Дополнение

## Модели данных

Задачи математической 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)

Данные могут быть **гомогенными** (однотипными) и **гетерогенными** (разнотипными).

Модели данных:

1. **DM - Data Matrix - Признаковое описание** - матрица значений атрибутов каждого объекта - (объекты гомогенные).

Примеры задач:

* 1. Классификация = *объекты с признаками* -> обучение -> *информационная модель* -> распознавание -> *объекты с предсказанными признаками*
	2. Восстаноление регрессий
1. **MD - Multi Dimensional - матрица кросс-сочетания** - это n-мерный куб, получаемый за счёт декартова произведения векторов с некоторыми атрибутами (вектор - это множество значений конкретного признака прикладной области) (объекты - гомогенные)

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

Показатель – это функция от множества объектов из ячейки, т.е. в матрице кросс-сочетания в ячейках подсчитываются показатели, после чего сами объекты выкидываются..

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

Детализация кросс-таблицы называется Drill Down.

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

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

Примеры задач:

* 1. Storytelling
	2. Data visualization
	3. Explorative analysis (поисковый анализ данных)
	4. Reporting systems (системы отчёта)
	5. Проверка гипотиз
1. **ST - Similarity Tensor - Описание эталонами** - таблица (аналитическое пространство (тензор)) задаёт попарное взаимодействие между конкретными объектами (задана некоторая метрика, оценка схожести).

Примеры задач:

* 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 - Транзакционные данные -** Формальный контекст (бинарный признак)

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.

Примеры задач:

* 1. Поиск популярных наборов (например, какие товары покупают вместе)
	2. Поиск ассоциативных правил
	3. Поиск последовательных ассоциативных правил

***Муть непонятная, здесь Майсурадзе пытался привести примеры задач, которые используют то или иные модели данных***:

Дисперсионно-факторный анализ (частый клиент - психология). ST -> DM

ST - Задача реализации метрики – в основном используется метод главных компонент.

MT -> ST →(шаг – многомерное шкалирование)→ DM – Анализ соответствия

DM -> ST -> DM – Сокращение размерности (частный случай метода главных компонент)

ST -> ST – Тематическое моделирование (преимущественно используется стохастическая кластеризация)

## Объём и площадь шара

Объём шара и площадь сферы в разных метрических пространствах:

Замкнутый шар (радиус r, центр с) = {x ∈ X | dist(x, c) <= r}

Сфера (радиус r, центр с) = {x ∈ X | dist(x, c) = r}

Граница шара и сфера в общем случае не совпадают. Это принципиально разные множества.

Гипотеза: если метрика в Rn порождена нормой, удовлетворяющей классическому определению, то объем шара (в традиционной мере) зависит от радиуса только множителем rn.

(Дополнительные формулы можно найти в файле “5.2 Объем шара и площадь сферы в разных метрических пространствах.docx”)

Самый интересный вывод: при росте размерности объём шара стремится к объёму куба.

## Форматы хранения данных

Есть машинное представление целых чисел и чисел с плавающей точкой.

Текстовые файлы бывают human readable и нет.

Форматы хранения 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)

**tab delimited values** - подразумевается, что разделитетль в полях не встречается

Надо рассказать про понятие экранирования

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, …

Особенности utf-8 - посмотрите википедию, там наглядно.

Ещё бывает понятие little-endian и big-endian - это вопрос того, в каком порядке записывать в файл байты несколько-байтных символов.

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.

## Снежинка и звезда

**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».

**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.

Чтобы схема не подходила ни под снежинку, ни под звезду, нужно чтобы зависимости были, например, следующими: A -> B, A -> C, B-> D, C -> D. (типа как ромбовидное наследование виртуальных классов в с++)

Данные необходимо группировать. Есть следующие этапы: модель данных на входе, модель данных на выходе и наличие целевого признака.