**Muslimov Arthur**

(since 08.12.2019)

**Прикладное машинное обучение…:**

Основная суть:

* Под машинным обучением (**ML**) понимается обеспечение лучшей обработки машинами определённой задачи за счёт обучения на основе данных вместо явного кодирования правил.
* Существует много разных типов систем машинного обучения: **с учителем и без, пакетное или динамическое, на основе образцов или моделей** и т.д.
* В проекте **ML** вы накапливаете данные в обучающем наборе и предоставляете его обучающему алгоритму. Если алгоритм основан на моделях, тогда он настраивает ряд параметров, чтобы подогнать модель к обучающему набору (т.е. выработать качественные прогнозы на самом обучающем наборе), и затем есть надежда, что алгоритм будет способен вырабатывать пригодные прогнозы также и на новых образцах. Если алгоритм основан на образцах, то он просто заучивает образцы на память и использует измерение сходства для обобщения на новые образцы.
* Система не будет хорошо работать, если обучающий набор слишком мал или данные не репрезентабельны, зашумлены либо загрязнены несущественными признаками (“Мусор на входе – мусор на выходе”). В конце концов, ваша модель не должна быть ни чересчур простой (и недообучаться), ни излишне сложной (и переобучаться).

Дальше идёт объяснение всего и вся.

Термины:

**Автономное обучение (offline learning) (**Т.е. ваша система сначала обучается, и только потом попадает в производственную среду, где она применяет полученные данные**);ф**

**Атрибут (**В **ML** представляет собой тип данных (Например, «Вид»). Однако часто люди используют слова **атрибут** и **признак** как синонимы**);**

**Визуализация (**т.е. просто вывод в двумерного или 3D представления данных**);**

**Внешнее обучение (out-of-core learning) (**Применяется для алгоритмов **динамического обучения**. Т.е. весь обучающий набор не должен находиться в памяти одной машины. Алгоритм загружает только часть данных, выполняет шаг обучения на этих данных и повторяет процесс, пока данные на иссякнут**);**

**Выбор признаков (feature selection) (**выбор самых полезных признаков для обучения из всех признаков (Согласитесь, не важно где именно находиться ягуар, будь то привычная всем саванна или заснеженный север. Он всё равно остаётся ягуаром)**);**

**Выведение (inference) (**т.е. применение модели, чтобы выработать прогноз на новых образцах**);**

**Выделение признаков (feature extraction) (**сложение нескольких полезных признаков в один более полезный (Пример с автомобилем: его пробег можно связать с износом, и получиться возраст). Тут поможет алгоритм понижения размерности**);**

**Гиперпараметр (**т.е. это параметр, который ограничивает ваш алгоритм (Чем выше – тем меньше свободы будет у алгоритма) (Допустим, вы выбираете короткий поводок для вашей собаки).**);**

**Глубинные анализ данных (~data mining) (**Тоже, что и **интеллектуальный анализ (~data mining));**

**Динамическое обучение (**т.е. система может обучаться постепенно, т.е. обучиться-поработать-обновиться-… . Внимание! Обычно это процесс происходит **автономно** **(offline learning)**, а не на ходу. Иногда вводит в заблуждение (так в книге написано)**);**

**Иерархическая кластеризация (**Умеет выделять группы в уже выделенных группах**);**

**Интеллектуальный анализ данных (~data mining) (**Применение систем **ML** для исследования крупных объёмов данныхможет помочь обнаружить паттерны, которые не были замечены ранее**);**

**Испытательный набор (test set) (**т.е. набор, на котором вы проверяете вашу систему для замерки её **производительности**. Принято разделять весь набор данных на **обучающий набор (training set) и испытательный набор (test set)** в соотношении 80% к 20% (или 4 к 1)**);**

**Кластеризация** (выделение групп из массы)**;**

**Логическая регрессия (logistic regression) (**Обычно используется для классификации, т.к. умеет выдавать вероятность в % по принадлежности к какому-либо классу**);**

**Метка (**Желаемое выходное значение для данного образца**);**

**Многомерная регрессия (multivariate regression) (**Это **регрессия** со множеством **признаков**, по которым модель будет вырабатывать прогноз**);**

**Недообучение (underfitting)**. **(**Т.е. Система может быть проста, чтобы заметить шаблоны в данных**);**

**Обнаружение аномалий (anomaly detection) (**Например, выявление дефектной турели с необычным набором **признаков** на конвейере**);**

**Обучающий набор (training set) (**На этом наборе ваша система и обучается**);**

**Обучающий образец (training instance** или **training sample) (**Обучающий пример из **обучающего набора (training set));**

**Обучение ассоциативным правилам (**Это обнаружение интересных зависимостей в данных. Например, может выясниться, что ваши клиенты часто покупают соус для барбекю и мясо для жарки, поэтому вам стоит расположить их по соседству на полках**) (association rule learning);**

**Обучение без учителя** (т.е. обучающий набор содержит просто образцы без ответов);

**Обучение на основе моделей (**т.е. мы строим модель, по которой система будет определять образцы**);**

**Обучение на основе образцов (**т.е. новые образцы сравниваются с базой**);**

**Обучение с подкреплением** (т.е. агент исследует среду, выбирает действие и получает какую-то награду или штраф. После чего корректирует политику для получения максимальной награды)**;**

**Обучение с учителем** (т.е. обучающий набор содержит образцы вида **образец – его** метка)**;**

**Ошибка выхода за пределы выборки (out-of-sample error) (**Тоже самое, что и **ошибка обобщения (generalization error));**

**Ошибка обобщения (generalization error) (**Тоже, что и **ошибка выхода за пределы выборки (out-of-sample error)** Частота ошибок на **испытательном наборе (test set)**. Т.е. ошибка, связанная с неполным **обучающим набором (training set)**. Избежать её почти что невозможно**);**

**Пакетное обучение (**т.е. чтобы обновить систему, надо заново скормить ей всю старую информацию + ещё и новую. Система помещается в производственную среду и работает без обучения. Это называется **автономным обучением** (**offline learning**)**)**;

**Переобучение** **(overfitting)** обучающих данных. Тут система может хорошо показывать себя только на обучающем наборе, но не обобщается как следует (Допустим, если вы приехали в новую страну как турист, и вас ограбил таксист, это не значит, что все таксисты здесь воры. Вы просто отнеслись к ним предвзято после этого случая). Ваша модель может быть просто слишком сложна, и она начнёт вглядываться в совершенно незначительные детали. Решается такая проблема **регуляризацией (regularization);**

**Понижение размерности (**т.е. упрощает данные желательно без потерь информации. Сюда относится метод выделение **признаков**, т.е. объединение похожих**);**

**Параметр модели (**Это значение, которое модель ищет в процессе своего обучения для последующего **выделения**. Это может быть угол для линейной **регрессии** или значение x для какой-либо функции**);**

**Перекрёстная проверка (cross-validation)** **(**Т.е. вы разбиваете весь **обучающий набор** **(training set)** на маленькие поднаборы и обучаете каждую модель на разных комбинациях этих поднаборов, затем проверяете их на оставшихся поднаборах. После выборки лучшей модели с гиперпараметрами, вы обучаете её на полном **обучающем набор (training set)**. **Ошибка обобщения (generalization error)** измеряется на **испытательном набор (test set));**

**Признак (**В **ML** представляет собой **атрибут +** его значение (Допусти, «Вид = Гепард»). Однако часто люди используют слова **атрибут** и **признак** как синонимы**);**

**Проверочный набор (validation set) (**На этом наборе вы проверяете свои решение для выявления лучшего**);**

**Прогнозаторы (predictors) (**Характеристики и **признаки** образца (марка, пробег, возраст автомобиля)**);**

**Производительность (**Т.е. насколько хорошо ваша система будет работать с новыми образцами. Измеряется путём прогона системы через **испытательный набор (training set));**

**Регуляризация (regularization)** **(**т.е. вы просто ограничиваете свободу **алгоритму** обучения (допустим, не позволяете себе слишком сильно злиться после какого-то инцидента), изменяя **гиперпараметр** (чем выше значение, чем меньше свободы будет у алгоритма**);**

**Регрессия (regression) (**Возвращение к среднему. Применяется для прогнозирования целевого числового значения, имея набор характеристик и признаков (пробег, возраст, марка автомобиля)**);**

**Сигнал (**Небольшая порция информации, поступающая в проект **ML**. Ссылаясь на теорию информации Шеннона: вам нужно высокое соотношение **сигнал/**шум**);**

**Скорость обучения (learning rate) (**Это важная часть систем динамического обучения. Это показатель скорости адаптации на новых **обучающих образцах (training instance)**. Если вы установите высокую скорость, то модель будет быстро адаптироваться к новым данным, но постепенно забудет старые. Если низкую, то ваша модель будет обладать большой инертностью, т.е. будет долго обучаться, но будет менее восприимчива к шуму в данных или нерепрезативных точек данных**);**

**Смещение выборки (sampling bias) (**Образцы могут быть нерепрезентативными в случае дефектного метода выборки. Есть один знаменитый пример **смещения выборки:** При президентских выборах 1936 года журнал Liberty Digest прогнозировал, что с 57% голосов выиграет Лэндон. Но, как мы знаем, победил Рузвельт с результатом в 62%. Выяснилось, что наш журнал опрашивал людей из различных клубов, подписчиков различных журналов, т.е., в основном, богатых и состоятельных людей, которым выгоднее голосовать за республиканца. Ещё одна ошибка была в том, что из опрашиваемых людей ответило только 25%, т.е. те, кто не интересовался политикой или кому просто не нравился журнал Liberty Digest не отвечали. Это называется **погрешностью, вызванной неполучением ответов (nonresponse bias));**

**Точность (accuracy) (**Это конкретная оценка **производительности);**

**Транспонирование (**Когда строки и столбцы в матрице меняются местами**);**

**Функция издержек (**измеряет расстояние между прогнозами нашей модели и обучающими примерами. Цель – минимизировать это расстояние**)**

**Функция полезности** **(utility function)**/**функция приспособленности** **(fitness function)** (измеряет, насколько хороша наша модель);

**Функция стоимости (cost function) (**определяет, насколько наша модель плоха**);**

**Частичное обучение** (т.е. обучающий набор содержит много простых образцов и несколько помеченных. Большинство таких алгоритмов — это смесь алгоритмов **обучения без учителя** и **Обучения с учителем**. Например, у вас есть уже выделенные группы объектов, но помечены из них только несколько. Алгоритм сам пометит оставшиеся в группах объекты, но если это что-то сложное, допустим люди на фотографиях, то часто приходиться вручную что-то исправлять)**;**

**Шум выборки (sampling noise) (**т.е. мы нам достаётся мало данных, с которыми мы и работаем. Это может привезти к тому, что мы не будем видеть полной картины, и мы получим неточную модель**);**

**ML** замечательно подходит для:

* задач, существующие решения которых требуют большого объёма ручной настройки или длинных списков правил – один алгоритм **ML** часто способен упростить код и выполниться лучше;
* сложных задач, для которых традиционных подход вообще не предлагает хороших решений – лучшие приёмы **ML** могут найти решение;
* Изменяющихся сред – система **ML** способна адаптироваться к новым данным;
* получения сведений о сложных задачах и крупных объёмах данных;

Существуют несколько критериев классификации системы **ML**:

1. Или **Обучение с учителем,** или **Обучение без учителя**, или **Частичное обучение**, или **Обучение с подкреплением**.

Основные алгоритмы:

**k-ближайших соседей (k-nearest neighbors);**

**линейная регрессия (lineal regression);**

**логистическая регрессия (logistic regression);**

**деревья опорных векторов (Support Vector Machine - SVM);**

**нейронные сети (neural network);**

Основные алгоритмы для **Обучения без учителя**:

**Кластеризация:**

**k-средние (k-means);**

**иерархический кластерный анализ (Hierarchical Cluster Analysis - HCA);**

**максимизация ожиданий (Expectation maximization);**

**максимизация ожиданий;**

**Визуализация** и **понижение размерности:**

**анализ главных компонентов (Principal Component Analysis – PCA);**

**ядерных анализ главных компонентов (kernel PCA);**

**локальное линейное вложение (Locally-Linear Embedding - LLE);**

**стохастическое вложение соседей с t-распределением;**

Типичные задачи для **Обучения без учителя:**

**Визуализация;**

Объединение данных в группы**;**

Понижение размерности**;**

**Обнаружение аномалий (anomaly detection);**

Обучение ассоциативным правилам **(association rule learning);**

1. Или **Пакетное обучение** Или **Динамическое обучение**.
2. Или **Обучение на основе образцов**,Или **Обучение на основе моделей:**

Простая модель:

**удовлетворённость\_жизнью = θ0 +  θ1\*ВВП\_на\_душу\_населения.**

Для этой функции нам понадобиться указать измерение производительности(?):

**Функция полезности** (**utility function**)/**Функция приспособленности (fitness function);**

**Функция издержек –** Её обычно и применяют**;**

**Функция стоимости (cost function);**

Мы применяем **функцию издержек.** Мы создали **Пакетно-обучающуюся систему на основе образцов с Обучением с учителем.**

Типичные проект **ML**:

1. Вы исследовали данные;
2. Вы выбрали модель;
3. Вы обучили модель на обучающих данных (т.е. в данном случае обучающий алгоритм искал значения параметров модели, которые доводят до минимума функцию издержек);
4. Наконец, вы применили модель, чтобы получить прогноз на новых образцах (это называется **выведением (inference),** надеясь на то, что эта модель будет хорошо обобщаться);

Т.к. наши главные задачи - это выбрать алгоритм и обучить его на данных, то две вещи, которые могут пойти не так являются:

1. “Плохой алгоритм”;

Тут есть несколько вещей, которые подпортят вам жизнь:

* **Переобучение** **(overfitting)** обучающих данных. Тут система может хорошо показывать себя только на обучающем наборе, но не обобщается как следует (Допустим, если вы приехали в новую страну как турист, и вас ограбил таксист, это не значит, что все таксисты здесь воры. Вы просто отнеслись к ним предвзято после этого случая.).

Ваша модель может быть просто слишком сложна, и она начнёт вглядываться в совершенно незначительные детали. Решается такая проблема **регуляризацией (regularization).** Т.е. вы просто ограничиваете **алгоритму** обучения (т.е. не позволяете себе слишком сильно злиться после данного инцидента), изменяя **гиперпараметр** (чем выше значение, чем меньше свободы будет у алгоритма). Регулировка **гиперпараметра** является важной частью построения системы **ML;**

* **Недообучение (underfitting)**. Тут наоборот. Система может быть слишком проста, чтобы заметить шаблоны в данных. Решается эта проблема просто:
  + Можно выбрать более мощную модель с большим числом параметров;
  + Можно предоставить алгоритму более лучших признаков (т.е. сконструировать новые признаки);
  + Ещё можно попробовать снизить ограничения на модель (например, уменьшить значение **гиперпараметра**);

1. “Плохие данные”. Причины могут быть разные:

Недостаточный размер обучающих данных. Тут всё ясно. Из-за неполной картины видения модель выдаст неточный результат;

Нерепрезентативные обучающие данные. Тут может быть **шум выборки (sampling noise)** или **смещение выборки (sampling bias)**;

Данные плохого качества. Тут тоже всё ясно. Если на вход идут образцы с ошибками, выбросами и шумом, то они помешают системе выделить точные признаки паттернов. Это приводит к снижению производительности системы. Есть два возможных пути решения проблемы:

* Если некоторые примеры несомненно являются мусором, то можно или отсеивать обучающий набор от таких образцов, или попытаться их собственноручно исправить;
* Если некоторые примеры просто неполные (допустим, 2-3% кандидатов не указали свой возраст), то можно просто отказаться от такого рода атрибутов, или отсеивать такие образцы, или попытаться самим дополнить их (выставить средний возраст для них). Ещё можно обучить две модели: одну с исправлением обучающего набора, другую без. Ну или самим придумать что-то;

Несущественные признаки. “Мусор на входе – мусор на выходе”. Система будет обучаться, если она поймёт, что вы от неё хотите. Если вы хотите, чтобы система смогла отличать ягуара от леопарда (этого часто не могут сделать даже люди), то вам стоит позаботиться об просто огромнейшем репрезентабельном наборе данных. Чтобы на каждом снимке каждое пятнышко прям блистало на их мехе. Иначе, система не успеет выделить ключевые паттерны и её производительность будет оставлять желать лучшего. Важная часть успеха проекта **ML** вытекает из хорошего набора признаков, на котором производиться обучение. Такой процесс, называемый **конструированием признаков (feature engineering)**,включает в себя**:**

* **выбор признаков (feature selection) –** выбор самых полезных признаков для обучения из всех признаков (Согласитесь, не важно где именно находиться ягуар, будь то привычная всем саванна или заснеженный север. Он всё равно остаётся ягуаром);
* **выделение признаков (feature extraction) –** сложение нескольких полезных признаков в один более полезный (У меня не подходящий пример, чтобы объяснить на пальцах. Пример с автомобилем: его пробег можно связать с износом, и получиться возраст). Тут поможет алгоритм понижения размерности;
* **создание новых признаков.** Тут нужно просто собрать побольше данных (Когда-нибудь система поймёт, что у ягуара хвост длиннее, чем у леопарда).

Испытание:

Оценка модели:

Единственным способ узнать **производительность** вашей модели является её проверка на новых образцах. Для этого вашу модель можно поместить в реальную производственную среду и понаблюдать за ней. Но ваши клиенты могут быть недовольны, если система покажет себя плохо. Есть способ лучше: испытать её на **испытательном наборе (test set)**. Частота ошибок на новых образцах называется **ошибкой обобщения (generalization error)**.

Если ошибка обучения низкая, т.е. она хорошо себя показала на обучающем наборе, но **ошибка общения (generalization error)** высока, то ваша модель **переобучена (overfitting)**.

Выбор модели:

Если вы никак не можете выбрать между линейнойиполиномиальной регрессией, то как вариант, можно опробовать обе, а замети решить, что лучше.

**Регуляризация**:

Допустим, вы выбрали вашу модель. Во избежание **переобучения** **(overfitting)**, вы настраиваете **гиперпараметры**. Как вариант, вы можете выбрать 100 значений для них и обучить 100 разных версий вашей модели и выбрать модель с наилучшей **производительностью.**

**Проверочный набор (validation set)**:

Вот вы уже выбрали вашу модель. Но оказалось, что в производственной среде она показывает себя несколько хуже, чем по паспорту. Дело в том, что вы выбрали наилучшие параметры конкретно для **испытательного набора** **(test set)**. В общем случае эта проблема решается с помощью дополнительного **проверочного набора (validation set)**. Вместо подгонки модели под **испытательный набор (test set)**, вы подгоняете её под **проверочный (test set)**. И уже, выбрав лучшее решение, вы прогоняете модель по **испытательному набору (test set)** и замеряете финальную **производительность**.

Во избежание большой траты данных частым случаем является применение приёма **перекрёстной проверки (cross-validation)**. Вы разбиваете весь **обучающий набор** **(training set)** на маленькие поднаборы и обучаете каждую модель на разных комбинациях этих поднаборов, затем проверяете их на оставшихся поднаборах. После выборки лучшей модели с гиперпараметрами, вы обучаете её на полном **обучающем набор (training set)**. **Ошибка обобщения (generalization error)** измеряется на **испытательном набор (test set).**

**Теорема об отсутствии бесплатных завтраков (No Free Lunch - NFL):**

Модель является упрощённой версией наблюдений. Упрощения означают отбрасывание избыточных деталей, которые вряд ли обобщатся на новые образцы. Однако чтобы решить, какие данные отбрасывать, а какие оставлять, вы должны делать предположения*.* Например, линейная модель выдвигает предположение о том, что данные фундаментально линейны и расстояние между образцами и прямой линией – просто шум, который можно безопасно игнорировать.

В известной работе 1996 года [“The Lack of A Priori Distinctions Between Learning Algorithms”] Дэвид Вольперт продемонстрировал, что если вы не делаете абсолютно никаких предположений о данных, тогда нет оснований отдавать предпочтение одной модели перед любой другой. Это теорема **об отсутствии бесплатных завтраков (No Free Lunch - NFL)**. Для одних наборов данных наилучшей моделью является линейная, в то время как для других ею будет нейронная сеть. Нет модели, которая априори гарантировала бы лучшую работу (отсюда и название теоремы). Единственный способ достоверно знать, какая модель лучше, предусматривает оценку всех моделей. Поскольку это невозможно, на практике вы делаете некоторые разумные предположения о данных и оцениваете лишь несколько рациональных моделей. Например, для простых задач вы можете оценивать линейные модели с различными уровнями регуляризации, а для сложных задач – разнообразные нейронные сети.

Не бывает глупых вопросов!

Введение в машинное обучение:

1. *Как бы вы определили машинное обучение?*

Я бы посмотрел на то, как работает программа. Если она справляется только с определёнными образцами, то **ML** тут не причастен. Если система сама справляется с ещё и с похожими образцами, то, скорее всего это оно самое.

1. *Можете ли вы назвать четыре типа задач, где* ***ML*** *показывает блестящие результаты?*

Думаю да. Первым на ум приходит задачи, где традиционные методы программирования вообще не предлагают хороших решений. Из этого вытекает ещё один тип задач, где приходится делать очень много рутинной работы и писать длиннющие правила. Также, думаю, что **ML** хорошо себя проявляет в меняющейся обстановке, например в интернете, где постоянно что-то происходит. Ну и, раз уж речь пошла об интернете, то нет лучшего решения для анализа Big Data, чем хорошо обученная система с **ML** на борту.

1. *Что такое помеченный обучающий набор?*

Это всего лишь обучающий набор данных, где вместе с образцами идут уже и ответы, по которым алгоритм **ML** и должен строить свои шаблоны.

1. *Каковы две наиболее распространённых задачи обучения с учителем?*

Одна из них – это банальная классификация. Например, спама. Ну и прогнозирование чего-то.

1. *Можете ли вы назвать четыре распространённых задачи обучения без учителя?*

Одна из них - это **кластеризация**. Это всего лишь группировка объектов с общими признаками. Например, подростков и стариков отличают разные интересы. Они выделяются в разные группы. Ещё часто встречается проект, нацеленный на **понижение размерности**. Это когда входные данные надо упростить. Например, зачем нам лишняя информация о размере экрана телевизора, если и так понятно, что старые модели довольно малы? В принципе, если нам не важны габариты, то мы можем просто выбросить этот **атрибут** и судить о размере экрана по его возрасту без потерь **производительности**. Также к этому методу обучения относится и **обнаружение аномалий** **(anomaly detection)** и выделение ассоциаций, для которого могут использовать уже упомянутый алгоритм **понижения размерности**.

1. *Какой тип алгоритма* ***ML*** *вы бы использовали, чтобы сделать возможным прохождение роботом по разнообразным неизведанным территориям?*

Судя по описанию, это должно быть **обучения с подкреплением**. Действия на основе политики могут достигнуть многого.

1. *Какой тип алгоритма* ***ML*** *вы бы использовали для сегментирования своих заказчиков в несколько групп?*

Думаю, будет **кластеризация**, которую я упомянул выше**.**

1. *Как бы вы представили задачу выявления спама – как задачу обучения с учителем или как задачу обучения без учителя?*

Конечно как задачу **обучения с учителем**. Не думаю, что какой-либо алгоритм **обучения без учителя** подошёл бы под эту роль.

1. *Что такое система динамического обучения?*

Это система, которая может обучаться поэтапно, т.е. обучиться-поработать-доучиться, при этом не забывая уже накопленный опыт.

1. *Что такое внешнее обучение?*

Это когда обучающий набор находится не на одной машине, а на многих (наверное потому что он просто гигантский и не умещается в памяти одной машины). Система загружает данные по частям, обучается на них, и повторяет этот процесс, пока не пройдёт все данные.

1. *Какой тип обучающего алгоритма при выработке прогнозов полагается на измерение сходства?*

Лёгкий вопрос. Конечно же это алгоритмы **обучения на основе образцов.**

1. *В чём разница между параметром модели и гиперпараметром обучающего алгоритма?*

Параметром модели является значение, которое модель и настраивает, чтобы потом использовать для **выделения. Гиперпараметр** же служит для другой цели – настройки степени свободы модели. Его задают ещё до начала обучения, и он остаётся постоянным на всём его протяжении.

1. *Что ищут алгоритмы обучения на основе моделей? Какую наиболее распространённую стратегию они используют для достижения успеха? Как они вырабатывают прогнозы?*

Как я уже и отвечал в предыдущем вопросе, они ищут значения **параметров** **модели** для своей функции. Обычно с использованием **функции издержек**, которая показывает, насколько далека ваша модель от решений **обучающих образцов**. Подставляя известные входные данные и найденные **параметры**, они получают ответ, который и является прогнозом.

1. *Можете ли вы назвать четыре основных проблемы в машинном обучении?*

Ну… я попробую. Первое – это **переобучение (overfitting)**, когда модель хорошо показывает себя на **обучающем наборе (training set)**, но гораздо хуже на новых **образцах**. Это связано с её сложностью. Она начинает вглядываться в несущественные признаки и судить по ним также, как и по более важным. Также есть обратное – **недообучение**. Это когда модель наоборот, слишком проста, чтобы обобщаться на данные. Вы же не станете использовать линейную функцию для определения кошек на фотографиях, верно? Но есть проблемы, где модель не причём. Например, недостаточный объём **обучающего набора (training set)**. Тут всё просто, система может с первого раза не понять, что от неё хотят. Ещё бывает так, что данные могут быть не репрезентабельны. Т.е. когда в них многовато-то шума или «бракованных» образцов, или всё вместе. Я скажу даже больше! Есть ещё одна проблема, связанная с данными: плохое качества. Тут может быть ошибка **смещения выборки (sampling bias)** или **шум выборки (sampling noise)**.

1. *Что происходит, если ваша модель хорошо работает с обучающими данными, но плохо обобщается на новые образцы? Можете ли вы назвать три возможных решения?*

Если возникает такое недопонимание, то знайте, это **переобучение (overfitting)**. Я разбирал его выше. Решается данная проблема следующими путями. Можно просто скормить ей побольше обучающих данных, чтобы она перестала обращать внимание на шум. Если дополнительных данных нет, то можете просто **регуляризировать** модель путём настройки **гиперпараметра**. Наверное, самый действенный вариант – это просто упростить модель. Серьёзно, зачем использовать глубокую нейросеть там, где достаточно будет и линейной **регрессии**?

1. *Что такое* ***испытательный набор (test set)*** *и почему он может применяться?*

Как понятно из его названия, по нему судят о **производительности** вашей модели. По нему также выбирают и настраивают тип модели и её **гиперпараметры** для лучшей производительности.

1. *В чём заключается цель* ***проверочного набора******(validation set)****?*

После выбора лучшего решения для **испытательного набора (test set)**, вы заметите, что в производственной среде ваша модель работает несколько хуже, в **испытательном наборе (test set)**. Это случается потому что вы подгоняли решение под **испытательный набор (test set)**. В общем случае, решение этой проблемы требует дополнительного **проверочного набора (validation set)**. Вы выбираете лучшее решение под него, но **производительность** измеряете на испытательном.

1. *Что может пойти не так при настройке гиперпараметров с использованием испытательного набора?*

Как я уже и описал, вы настроите модель под конкретную пачку примеров. Другие наборы образцов могут потребовать немного других решений.

1. *Что такое* ***перекрестная проверка******(cross-validation)*** *и почему ей отдаётся предпочтение перед* ***проверочным набором******(validation set)****?*

Так называется такой приём, когда вы разбиваете весь ваш обучающий набор на маленькие поднаборики и обучаете разные модели на разные их комбинации, затем проверяете их на оставшихся кусочках данных. Дальше вы обучаете лучшую модель на полном наборе данных и используете в качестве измеряете производительность на **испытательном наборе (test set).** Это делается для экономии данных. Иногда бывает проблематично добыть больше данных.