**Регрессия**

Linear regression

Основная формула

Для минимизации использует метод найменьших квадратов 

Ridge

Та же самая линейная регрессия, но с L2 регуляризацией.

Та же основная формула +a\*. Параметр α отвечает за сжатие коэффициентов: с увеличением α параметры модели стремятся к нулю.

Lasso regression

То же самое что и ridge, но с l1 регуляризацией.

DecisionTreeRegressor

Ноль инфы. НАДО ИСКАТЬ!

За каким принципом разбивается дерево?  
может ли разбится больше чем на 2 ветки?  
Как именно получается ответ в регресии?

В целом метод очень рпост и интуитивно понятно как он работает. Как минимум на классификации. На регресии правда есть свои нюансы.

Для разделения на ветки используется метод найменших квадратов. Возможно строит линейные зависимости и делить датасет на часть(ветку)Б и пото ветки так же разбиваются на части, и т.д. и т.д.

Random forest

Ансамбль деревьев принятия решений.

Лучше выбирать рандомный признаки, чем рандомные деревья.

kNN  
Метод ближайших соседей.  
Один из самых простых методов. Нет смысла даже что-то рассказывать. Хотя надо бы поискать методы кнн.

Как именно берутся н ближайших соседей, не можем же мы отсортировать тупо по какой то переменной и всё.

Perceptron

Можно ли использовать его для регресии, и тогда как именно он работает.

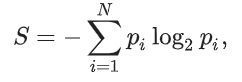
GradientBoostingRegressor

SGDRegressor

**Классификация**

DecisionTreeClassifier

Для самого разбиение можно использовать:

2. Энтропия Шеннона , где pi – вероятности нахождения системы в i-ом состоянии. Само число s указывает на состоянии хаоса, чем ближе число к 1, тем более в хаотичном состоянии оно находится.

S0 – число означающее хаус в «исходном дереве», перебираем все **предикаты** (набор функций/значений, которые разбивают множество на подмножество (больше, меньше, равно, т.п.)), ищем числа S1 и S2, хаосы в разделяемых ветках, и выбираем признак, где дельта S найбольшее. Одним из возможных критериев остановки может быть небольшое значение *∆S*

2. Неопределенность Джини (Gini impurity): G=1−∑k(pk)^2

LogisticRegression

[Всё то же самое, что и линейная регрессия, только берется логистическая функция (сигмоида) [9](http://datareview.info/wp-content/uploads/2015/01/9.png)](http://datareview.info/wp-content/uploads/2015/01/9.png)

SVM

Пример подходит для бинарной классификации, так же можно использовать для мультиклассовой класс., но надо слегка преобразовать, хз как(мб, использовать один против всех).

Находятся такие параметры w, что wX=Y. Далее ищем максимальное расстояние от опорных векторов(2 ближайших точок к этой самой ‘прямой’) к нашей прямой. Данное расстояние будет равно https://habrastorage.org/storage/habraeffect/31/86/3186bd99eb78eadf2af35359b431c58e.png. Проблема нахождения максимума https://habrastorage.org/storage/habraeffect/31/86/3186bd99eb78eadf2af35359b431c58e.png эквивалентна проблеме нахождения минимума https://habrastorage.org/storage/habraeffect/05/47/0547d2ad77f3163ca1641d382a173ad2.png. Эта задаач решается с помощью множителей Лагранжа.

Когда данные линейно неразделимы - все элементы обучающей выборки вкладываются в пространство **X** более высокой размерности с помощью специального отображения https://habrastorage.org/storage/habraeffect/ea/d4/ead4cdea29783cc66b34f5d3706f86e8.png. Классифицирующая функция *F* принимает вид https://habrastorage.org/storage/habraeffect/46/9f/469fa9d46d14e1afb2d5b26ff598851b.png. Выражение https://habrastorage.org/storage/habraeffect/8f/3b/8f3bee609b3f807563af6f7e4ae5ba37.png называется *ядром* классификатора. С математической точки зрения ядром может служить любая положительно определенная симметричная функция двух переменных.

Чаще всего на практике встречаются следующие ядра:  
**Полиномиальное:** https://habrastorage.org/storage/habraeffect/8d/73/8d734dc93985270c76bfc0e059556264.png  
**Радиальная базисная функция:** https://habrastorage.org/storage/habraeffect/df/c6/dfc66ea3b4a833ef4033ba07362b31d3.png  
**Гауссова радиальная базисная функция:** https://habrastorage.org/storage/habraeffect/4d/b0/4db09c6643fd0bb3d19f4458707209ec.png  
**Сигмоид:** https://habrastorage.org/storage/habraeffect/35/37/353764f90f9af20442bcc97c6a9b4a08.png

RVM

Работает так же как и SVM, но выдаёт вероятность.

Generalization

**Кластеризация**

Метода расстояний:

1. Евклидово расстояние https://habrastorage.org/getpro/habr/post_images/aa4/6e7/d7b/aa46e7d7b544dbaa221a43bb671fb43c.jpg
2. Квадрат евклидового расстояние https://habrastorage.org/getpro/habr/post_images/5dc/84a/1d6/5dc84a1d66392e9f11ba13be381ad036.jpg
3. Манхєттенское расстояние https://habrastorage.org/getpro/habr/post_images/930/f68/0a2/930f680a27caed2bed04f6b09a70a785.jpg
4. Чебешиво расстояние https://habrastorage.org/getpro/habr/post_images/ccd/79d/96b/ccd79d96ba5a93373b4daab5650ea7d2.jpg
5. Степенное расстояние https://habrastorage.org/getpro/habr/post_images/4cf/36c/f90/4cf36cf908a6c2c27a0b87039d82d6e6.jpg r и p задаются пользователем

Классификая алгоритмов:

1. Иерархические и плоские.
   1. Иерархические – на выходе дают дерево(тоесть разделяют на всё большее и большее кол кластеров),
   2. Плоские алгоритмы строят одно разбиение объектов на кластеры.
2. Четкие и нечеткие.
   1. Четкий – каждому объекту выборки ставят в соответствие номер кластера.
   2. Нечеткие – – каждому объекту выборки ставят показывающую степень отношения объекта к кластерам

Объединение кластеров:

1. Одиночная связь

Расстояние между двумя наиболее близкими объектами (ближайшими соседями) в различных кластерах

1. Полная связь

Наибольше расстояние между любыми двумя объектами в различных кластерах

1. Невзвешенное попарное среднее

Среднее расстояние между всеми парами объектов в них

1. Взвешенное попарное среднее

То же самой, что и предыдущий, но размер кластера используется в качестве весового коэффициента.

1. Невзвешенный центроидный метод

Расстояние между их центрами тяжести.

1. Взвешенный центроидный метод

То же самое, что и невзвешенный, но размер кластера используется в качестве весового коэффициента.

Иерархичные можно поделить на две группы: нисходящие и восходящие.

Восходящие более распротраненные, с начала они разбиаются на большое количество кластеров, а потом сходятся к одному. Нисхдящие – наоборот.

Метод к средних

Алгоритм строит заданное число кластеров

Работа алгоритма делится на несколько этапов:

1. Случайно выбрать *k* точек, являющихся начальными «центрами масс» кластеров.
2. Отнести каждый объект к кластеру с ближайшим «центром масс».
3. Пересчитать «центры масс» кластеров согласно их текущему составу.
4. Если критерий остановки алгоритма не удовлетворен, вернуться к п. 2.

В качестве критерия остановки работы алгоритма обычно выбирают минимальное изменение среднеквадратической ошибки. Так же возможно останавливать работу алгоритма, если на шаге 2 не было объектов, переместившихся из кластера в кластер.

Метод с средних

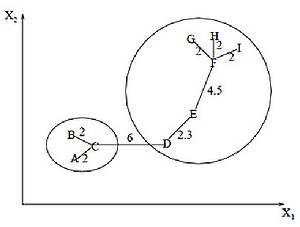
Он представляет собой модификацию метода k-средних, но выдаёт вероятность принадлежания к классу.

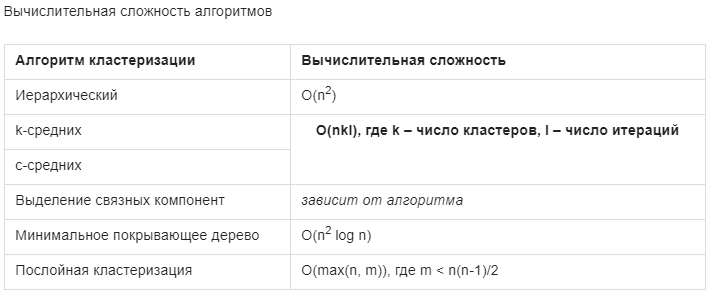
Шаги работы алгоритма:

1. Выбрать начальное нечеткое разбиение *n* объектов на *k* кластеров путем выбора матрицы принадлежности *U* размера *n x k*.
2. Используя матрицу U, найти значение критерия нечеткой ошибки:  
   https://habrastorage.org/getpro/habr/post_images/21c/c5d/72b/21cc5d72b0c9bee8ad0476798eb4093c.jpg,  
   где *ck* — «центр масс» нечеткого кластера *k*:  
   https://habrastorage.org/getpro/habr/post_images/5fb/811/641/5fb811641b4fa3c84ba2490811ff53ea.jpg.
3. Перегруппировать объекты с целью уменьшения этого значения критерия нечеткой ошибки.
4. Возвращаться в п. 2 до тех пор, пока изменения матрицы *U* не станут незначительными.

Алгоритмы, основанные на теории графов  
Суть таких алгоритмов заключается в том, что выборка объектов представляется в виде графа *G=(V, E)*, вершинам которого соответствуют объекты, а ребра имеют вес, равный «расстоянию» между объектами. Достоинством графовых алгоритмов кластеризации являются наглядность, относительная простота реализации и возможность вносения различных усовершенствований, основанные на геометрических соображениях. Основными алгоритмам являются алгоритм выделения связных компонент, алгоритм построения минимального покрывающего (остовного) дерева и алгоритм послойной кластеризации.

Алгоритм минимального покрывающего дерева

Алгоритм минимального покрывающего дерева сначала строит на графе минимальное покрывающее дерево, а затем последовательно удаляет ребра с наибольшим весом. На рисунке изображено минимальное покрывающее дерево, полученное для девяти объектов.  
  
  
  
Путём удаления связи, помеченной CD, с длиной равной 6 единицам (ребро с максимальным расстоянием), получаем два кластера: {A, B, C} и {D, E, F, G, H, I}. Второй кластер в дальнейшем может быть разделён ещё на два кластера путём удаления ребра EF, которое имеет длину, равную 4,5 единицам.





**Метрики**

**Классификация**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | | **Actual class** | | | | **Cat** | **Dog** | **Rabbit** | | **Predicted class** | **Cat** | **5** | 2 | 0 | | **Dog** | 3 | **3** | 2 | | **Rabbit** | 0 | 1 | **11** | |

TP – 5 котов отобразило как котов

FN – 3 кота отобразило как не котов

FP – 2 не котов отобразило как котов

TN – 17 не котов отобразило как не котов

|  | y=1 | y=0 |
| --- | --- | --- |
| y^=1 | True Positive (TP) | False Positive (FP) |
| y^=0 | False Negative (FN) | True Negative (TN) |

Первое значение(T,F) – означет правильно ли опредилило

Второе значение(P,N) – означает предиктивные 1 или 0.

Negative должно быть больше(если задача про рак – здоровые будут негатив)

Самый простой способ:

**accuracy**= TP+TN / TP+TN+FP+FN

не подходит для неравномерно разбитых данных.

**Precision**

Какая доля объектов полученных от классификатора положительных ответов являются правильными.

Precision = TP / TP+FP

**Recall**

Какая доля правильных ответов из всего множества объектов, c позитивными отметками.

Recall = TP / TP+FN

**F мера** -  среднее гармоническое precision и recall.

Fβ= . β – вес точности в метрике. при β они гармоничны.

F-мера достигает максимума при полноте и точности, равными единице, и близка к нулю, если один из аргументов близок к нулю.

**AUC-ROC**

Используют, когда на выходе значения от 0 до 1(вероятности).  
Данная кривая представляет из себя линию от (0,0) до (1,1) в координатах True Positive Rate (TPR) и False Positive Rate (FPR):

TPR=TP / TP+FN, TPR - полнота  
FPR=FP / FP+TN,  FPR - доля из объектов negative класса, которую алгоритм предсказал неверно.

В идеальном случае, когда классификатор не делает ошибок (FPR = 0, TPR = 1) мы получим площадь под кривой, равную единице; Площадь под кривой в данном случае показывает качество алгоритма (больше — лучше), кроме этого, важной является крутизна самой кривой — мы хотим максимизировать TPR, минимизируя FPR.

Критерий AUC-ROC устойчив к несбалансированным классам, но есть и исключения.

**AUC PR**

То же самое, строит площадь под кривой precision-recall

#### **Logistic Loss**

**Регрессия**

#### Mean Absolute Error (MAE)

Метрика измеряет среднюю сумму абсолютной разницы между фактическим значением и прогнозируемым значением.

[https://3.bp.blogspot.com/-B5qlRjsefZs/Wuq1z17vHbI/AAAAAAAACZM/ZOHviQntHLAxIMNSBgDwZrwsThkaEo34gCK4BGAYYCw/s400/MAE.png](http://3.bp.blogspot.com/-B5qlRjsefZs/Wuq1z17vHbI/AAAAAAAACZM/ZOHviQntHLAxIMNSBgDwZrwsThkaEo34gCK4BGAYYCw/s1600/MAE.png)

Mean Squared Error (MSE)

[https://4.bp.blogspot.com/-8-nlPIjzBEU/Wuq14EZbqAI/AAAAAAAACZY/XLScxPJAbBIpY8fEc_YZuW7wR0QxYqfhQCK4BGAYYCw/s400/MSE.png](http://4.bp.blogspot.com/-8-nlPIjzBEU/Wuq14EZbqAI/AAAAAAAACZY/XLScxPJAbBIpY8fEc_YZuW7wR0QxYqfhQCK4BGAYYCw/s1600/MSE.png)

#### Root Mean Squared Error (RMSE)

[https://4.bp.blogspot.com/-SuftZtem2_E/Wuq152CYCXI/AAAAAAAACZg/o1hLEvFE_Cgf3JEdVZiMvlloK_LFKvrwQCK4BGAYYCw/s400/RMSE.png](http://4.bp.blogspot.com/-SuftZtem2_E/Wuq152CYCXI/AAAAAAAACZg/o1hLEvFE_Cgf3JEdVZiMvlloK_LFKvrwQCK4BGAYYCw/s1600/RMSE.png)