Применение моделей

kNN

Применяется для автоматической классификации или регрессии.
Когда KNN используется для классификации, выходной сигнал может быть рассчитан как класс с самой высокой частотой из K наиболее похожих случаев. Каждый экземпляр в сущности голосует за свой класс, а за класс принимается большинство голосов.
В случае регрессии прогноз основывается на среднем или медиане K-подобных случаев. Перед применением необходимо определить метрику (расстояние). Обычно используются Евклидова, Hamming distance, Manhattan distance, расстояние Минковского.
Евклидово - это хорошая мера расстояния, которую можно использовать, если входные переменные имеют одинаковый тип (например, все измеренные значения ширины и высоты). Манхэттенское расстояние - хорошая мера для использования, если входные переменные не похожи по типу (например, возраст, пол, рост и т. Д.).
Значение k находится с помощью настройки или эмпирически.
Pros
Может быть применён к выборкам с большим количеством переменных, но существует проклятье размерности: По мере увеличения количества измерений объем входного пространства увеличивается с экспоненциальной скоростью.
Cons
Неэффективен в реальных задачах из-за низкой скорости классификации. Если в обучающей выборке N объектов, в тестовой выборе M объектов, а размерность пространства — K, то количество операций для классификации тестовой выборки может быть оценено как O(K*M*N).
Алгоритм работы kNN является хорошим примером для начала знакомства с ML.
Ссылки
Реализация на Python

Logistic Regression

Supervised learning classifier. В отличие от линейной регрессии, Loss function defined as ...
Решение использует метод градиентного спуска. Лучше всего подходит, когда выходная переменная принимает только два значения. Важность логистический регрессии обусловлена тем, что многие задачи анализа данных могут быть решены с помощью бинарной классификации или сведены к ней.
Links
In Python
Ссылки

Введение: (https://wiki.loginom.ru/articles/logistic-regression.html)

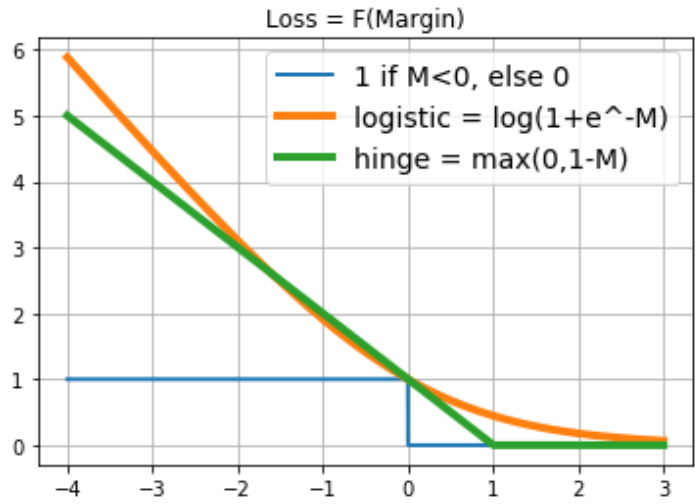
SVM

Классификатор. Алгоритмически находит уравнение разделяющей гиперплоскости.
Решение использует метод градиентного спуска, but we can solve hard-margin SVM problem without gradient descent.
The task of searching support vectors is reduced to search saddle point in the Lagrange function – this task refers to quadratic programming only.
Существует два основных типа классификации алгоритмов SVM Hard Margin и Soft Margin:

- Hard marge: стремится найти лучшую гиперплоскость, не допуская ни одной формы неправильной классификации.
- Soft marge: мы добавляем степень терпимости в SVM. Таким образом, мы позволяем модели произвольно ошибочно классифицировать несколько точек данных, если это может привести к идентификации гиперплоскости, способной лучше обобщать невидимые данные.

Loss function defined as https://habrastorage.org/getpro/habr/formulas/39d/c70/e22/39dc70e228cab0427f3ea55a08ed9415.svg

Сравнение функции потерь для logistic regression и SVM



Pros

- хорошо работает с данными небольшого объема
- хорошо работает с пространством признаков большого размера;
- алгоритм максимизирует разделяющую полосу, которая, как подушка безопасности, позволяет уменьшить количество ошибок классификации;
- так как алгоритм сводится к решению задачи квадратичного программирования в выпуклой области, то такая задача всегда имеет единственное решение (разделяющая гиперплоскость с определенными гиперпараметрами алгоритма всегда одна).

Cons

- долгое время обучения (для больших наборов данных);
- неустойчивость к шуму: выбросы в обучающих данных становятся опорными объектами-нарушителями и напрямую влияют на построение разделяющей гиперплоскости;
- не описаны общие методы построения ядер и спрямляющих пространств, наиболее подходящих для конкретной задачи в случае линейной неразделимости классов. Подбирать полезные преобразования данных – искусство.

Skikit-learn

В Scikit-learn расстояние от объектов до построенной алгоритмом LinearSVC границы можно получить с помощью метода decision_function класса LinearSVC.

Decision trees

Классификатор

Pros

- Порождение четких правил классификации, понятных человеку, например, «если возраст < 25 и интерес к мотоциклам, отказать в кредите»
- Деревья решений могут легко визуализироваться
- Относительно быстрые процессы обучения и классификации
- Малое число параметров модели
- Поддержка и числовых, и категориальных признаков

Cons

- Разделяющая граница, построенная деревом решений, имеет свои ограничения (состоит из гиперкубов), и на практике дерево решений по качеству классификации уступает некоторым другим методам
- Необходимость отсекать ветви дерева (pruning) или устанавливать минимальное число элементов в листьях дерева или максимальную глубину дерева для борьбы с переобучением. Впрочем, переобучение — проблема всех методов машинного обучения
- Нестабильность. Небольшие изменения в данных могут существенно изменять построенное дерево решений. С этой проблемой борются с помощью ансамблей деревьев решений (рассмотрим далее)
- Проблема поиска оптимального дерева решений NP-полна, поэтому на практике используются эвристики типа жадного поиска признака с максимальным приростом информации, которые не гарантируют нахождения глобально оптимального дерева

Links:

Введение (https://m.habr.com/ru/post/270449/)

Naive Bayes

Pros

- Очень быстро обучается, поэтому хорошо подходит для real time
- Многоклассовая классификация. Позволяет прогнозировать

вероятности для множества значений целевой переменной
- Классификация текстов, анализ тональности, фильтрация спама
- Рекомендательные системы

Cons

- Использование ансамблей (bagging, busting) не даёт результатов, т.к. они направлены на уменьшение дисперсии

Skikit-learn

Содержит 3 типа моделей на основе наивного байесовского алгоритма

- GaussianNB. Используется в случае нормально распределённых непрерывных признаков
- Multinomial. Используется в случая дискретных признаков. Например, в задаче классификации текстов признаки могут показывать, сколько раз каждое слово встречается в данном тексте.
- Bernoulli. Используется в случае двоичных признаков.

Links:

Хорошее введение (https://www.google.com/amp/s/craftappmobile.com/naive-bayes-classifier-and-bayes-classifiers/amp/)

Введение (http://datareview.info/article/6-prostyih-shagov-dlya-osvoeniya-naivnogo-bayesovskogo-algoritma-s-primerom-koda-na-python/)