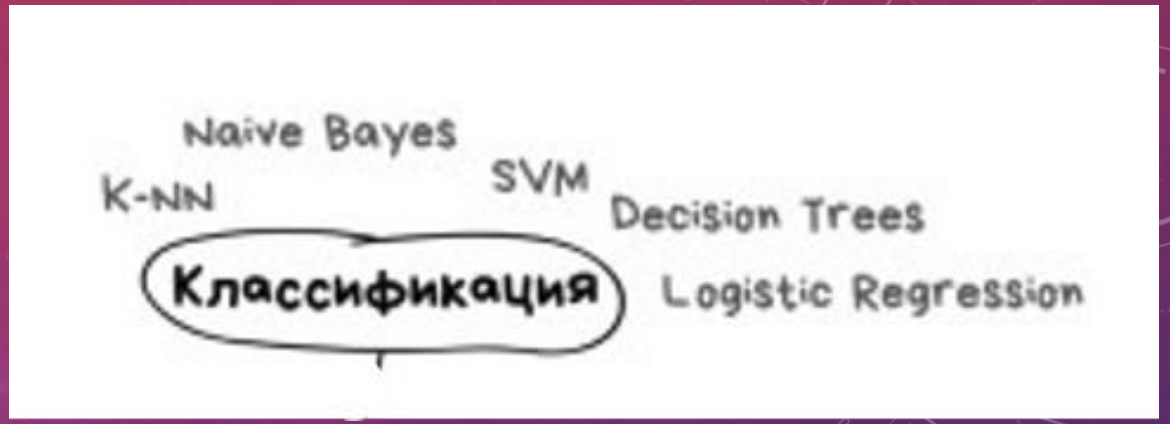
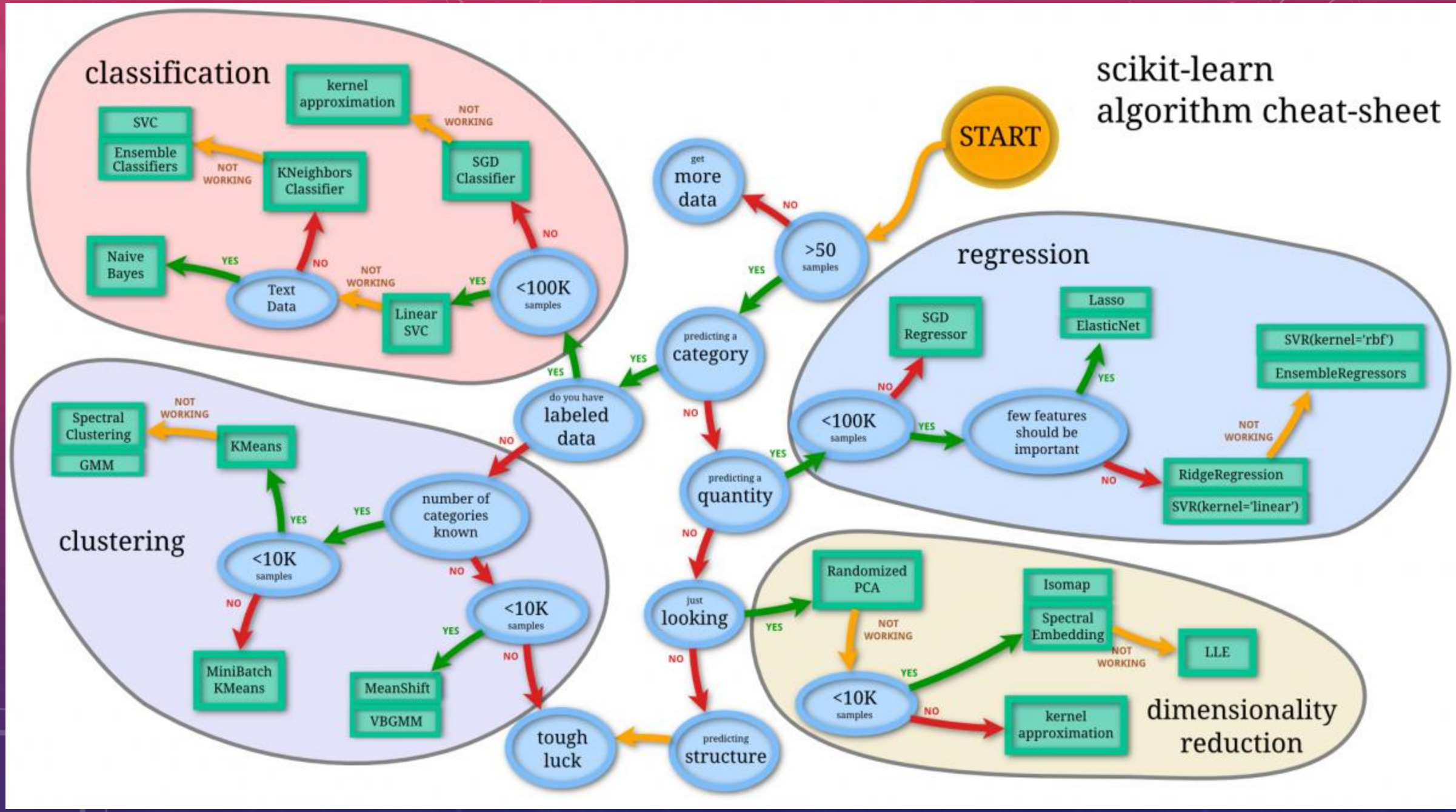
МЕТОДЫ МАШИННОГО ОБУЧЕНИЯ С УЧИТЕЛЕМ

КЛАССИФИКАЦИЯ


Машинное обучение
Классическое обучение
Без учителя
С учителем
Кластеризация
DBSCAN
K-Means
Agglomerative
Mean-Shift
Fuzzy C-Means
Классификация
Naive Bayes
K-NN
SVM
Decision Trees
Logistic Regression
Поиск правил
Euclat
Apriori
FP-Growth
Регрессия
Linear Regression
Polynomial Regression
Ridge/Lasso Regression
Уменьшение размерности (обобщение)
t-SNE
PCA
LSA
SVD
LDA
Ансамблевые методы
Стекинг
Беггинг
Random Forest
Бустинг
XGBoost
AdaBoost
LightGBM
CatBoost
Обучение с подкреплением
Генетический Алгоритм
Q-Learning
SARSA
Deep Q-Network (DQN)
A3C
Нейросети и глубокое обучение
Convolutional Neural Networks (CNN)
DCNN
Перцептроны (MLP)
Recurrent Neural Networks (RNN)
LSM
LSTM
GRU
Generative Adversarial Networks (GAN)
Autoencoders
seq2seq


Naive Bayes
K-NN
SVM
Decision Trees
Классификация
Logistic Regression


scikit-learn
algorithm cheat-sheet
START
get more data
>50 samples
predicting a category
do you have labeled data
predicting a quantity
just looking
predicting structure
tough luck
classification
kernel approximation
SVC
Ensemble Classifiers
KNeighbors Classifier
SGD Classifier
Naive Bayes
Text Data
Linear SVC
<100K samples
clustering
Spectral Clustering
GMM
KMeans
number of categories known
<10K samples
MiniBatch KMeans
MeanShift
VBGMM
regression
SGD Regressor
Lasso
ElasticNet
SVR(kernel='rbf')
EnsembleRegressors
<100K samples
few features should be important
RidgeRegression
SVR(kernel='linear')
dimensionality reduction
Randomized PCA
Isomap
Spectral Embedding
LLE
<10K samples
kernel approximation
YES
NO
NOT WORKING

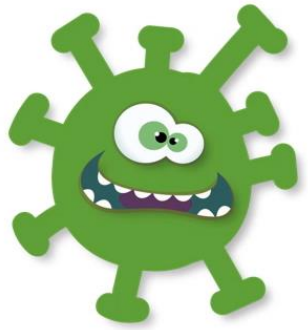
КЛАССИФИКАЦИЯ

- **Задача классификации в машинном обучении** — это задача отнесения объекта к одному из заранее определенных классов на основании его формализованных признаков. Каждый из объектов в этой задаче представляется в виде вектора в N-мерном пространстве, каждое измерение в котором представляет собой описание одного из признаков объекта. Для обучения классификатора необходимо иметь набор объектов, для которых заранее определены классы. Это множество называется **обучающей выборкой**, её разметка производится вручную, с привлечением специалистов в исследуемой области
- Задача классификации (как, впрочем и следует из названия) сводится к отнесению конкретного объекта к одному из заранее известных классов. Вот эта “метка” - название класса - и выступает в роли **целевой переменной**. И, совершенно естественно, что классов может быть только конечное количество, поэтому целевая переменная - дискретная.

ПРИМЕРЫ БИНАРНОЙ КЛАССИФИКАЦИИ

ИЛИ

ИЛИ



ИЛИ

АЛГОРИТМЫ КЛАССИФИКАЦИИ

Scikit-Learn обеспечивает легкий доступ к многочисленным различным алгоритмам классификации.

К числу таких классификаторов относятся:

- К-Ближайшие Соседи
- Машины опорных Векторов
- Классификаторы дерева решений / Случайные леса
- Наивный Байес
- Линейный дискриминантный анализ
- Логистическая регрессия

МЕТОД БЛИЖАЙШИХ СОСЕДЕЙ (K NEAREST NEIGHBORS)

Все объекты расположены в многомерном пространстве

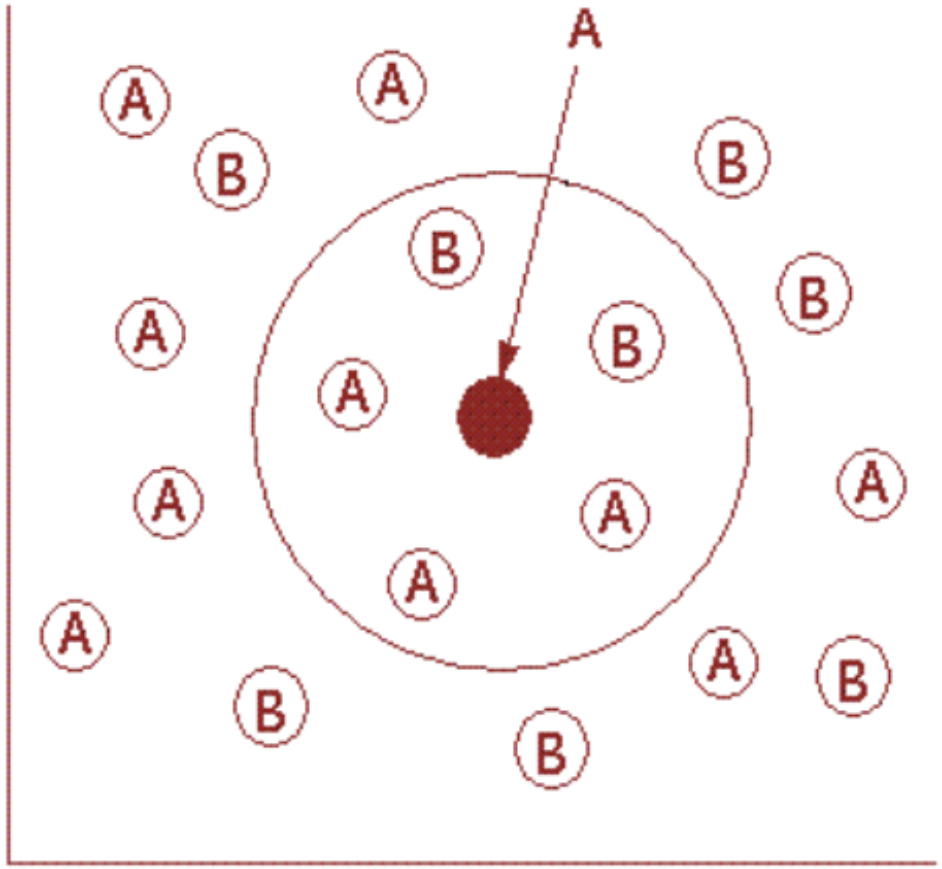
1. Задаем число *k* – количество ближайших соседей.

2. Ищем *k* объектов с минимальным расстоянием до нашего нового объекта. Используем меру для расчета расстояний.

3.1 Простое невзвешенное голосование. Считаем сколько объектов с классами присутствует внутри заданного расстояния. Например, если число объектов с классом А большинство, то новый объект относится к классу А.

3.2. Взвешенное голосование

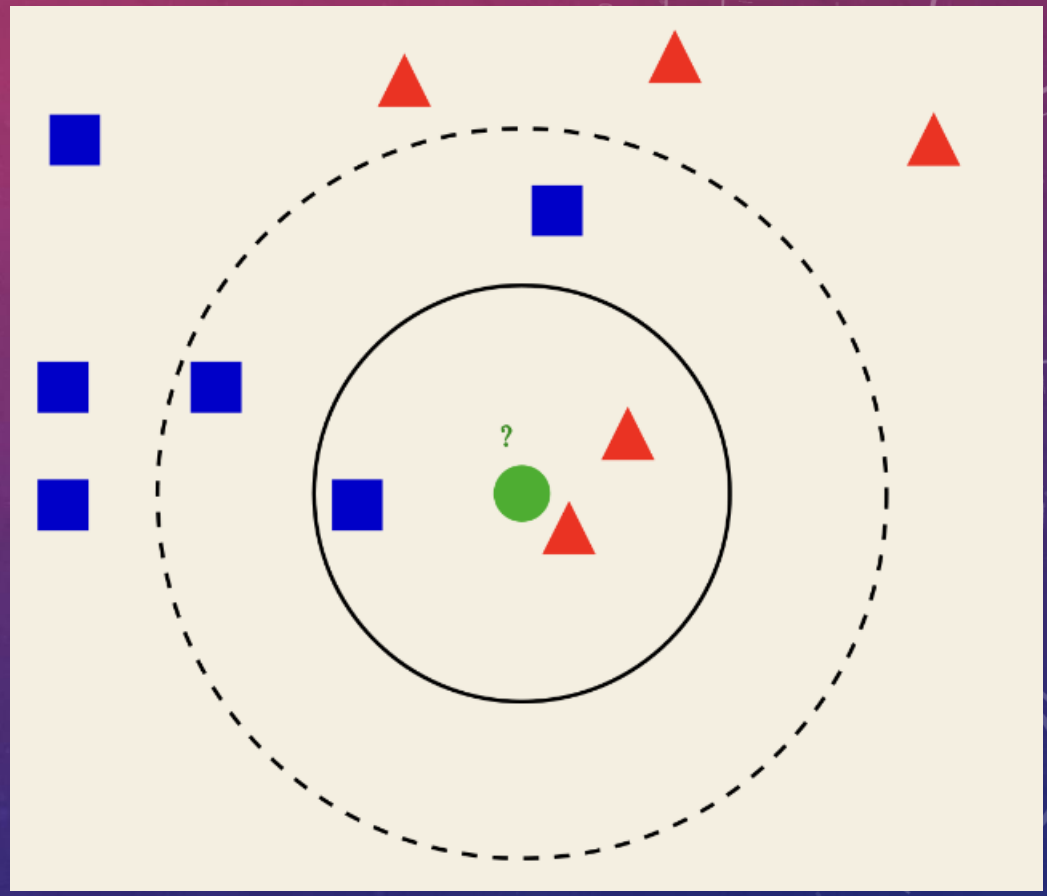
В такой ситуации учитывается также и расстояние до новой записи. Чем меньше расстояние, тем более значимый вклад вносит голос.



МЕТОД К-БЛИЖАЙШИХ СОСЕДЕЙ

Преимущества и недостатки

- Этот алгоритм довольно прост в своей реализации и устойчив к зашумленным обучающим данным. Даже если обучающие данные велики, они достаточно эффективны. Единственным недостатком алгоритма KNN является то, что нет необходимости определять значение K, а вычислительные затраты довольно высоки по сравнению с другими алгоритмами.



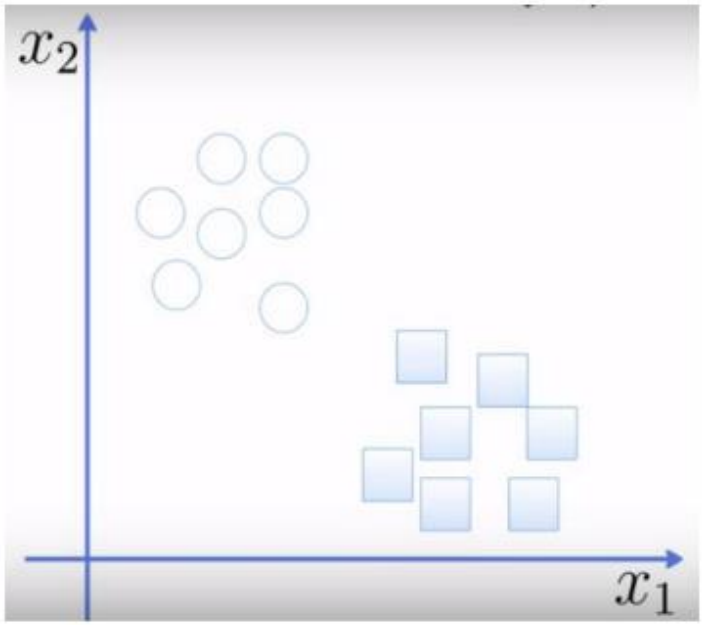
Примеры использования

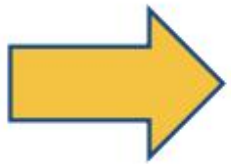
- Промышленное применение для поиска аналогичных задач по сравнению с другими
- Приложения для обнаружения рукописного ввода
- Распознавание изображений
- Распознавание видео
- Анализ запасов

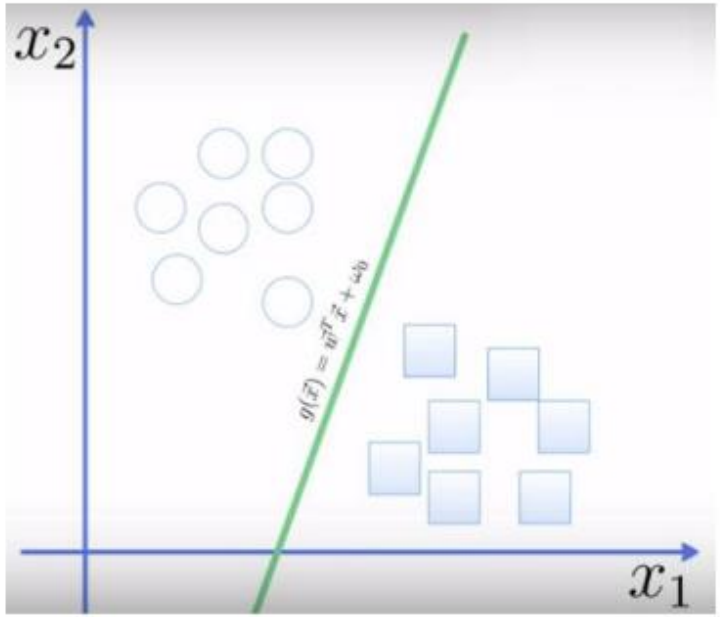
МЕТОД ОПОРНЫХ ВЕКТОРОВ (SVM)

Каждый объект данных (например, документ, котировки ценных бумаг или компании) представлен как вектор в **P** мерном пространстве (последовательность чисел). Пусть у нас есть тестовая коллекция, в которой есть набор объектов (features) и есть набор классов. Математическая задача обучения заключается в том что бы найти функцию, которая адекватно сопоставляла объекты и классы, то есть найти такую функцию, которая эффективно разделяла бы объекты в пространстве features.

Рассмотрим пример на плоскости: У нас есть два класса с двумя features (x1, x2). Нужной найти прямую линию, которая оптимально разделяла два класса.





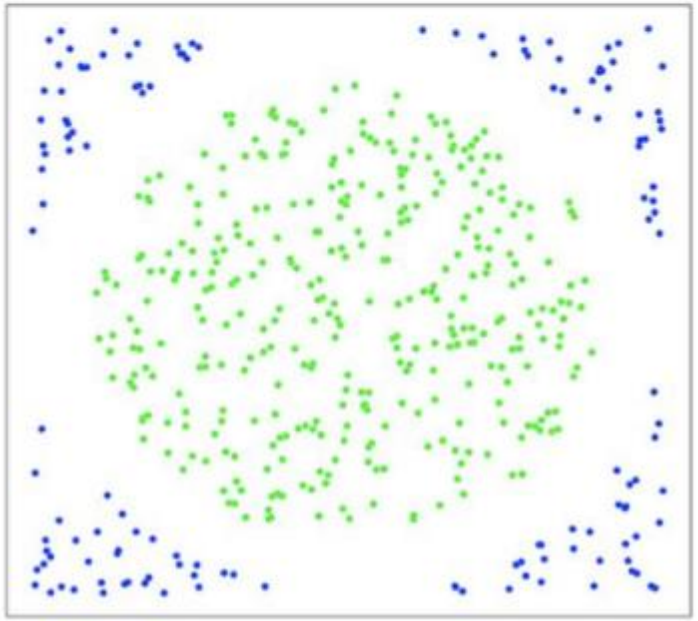


МЕТОД ОПОРНЫХ ВЕКТОРОВ (SVM)

Нахождение уравнения плоскости является стандартной задачей квадратичного программирования и решается с помощью множителей Лагранжа. Собственно в этом заключается процесс обучения.

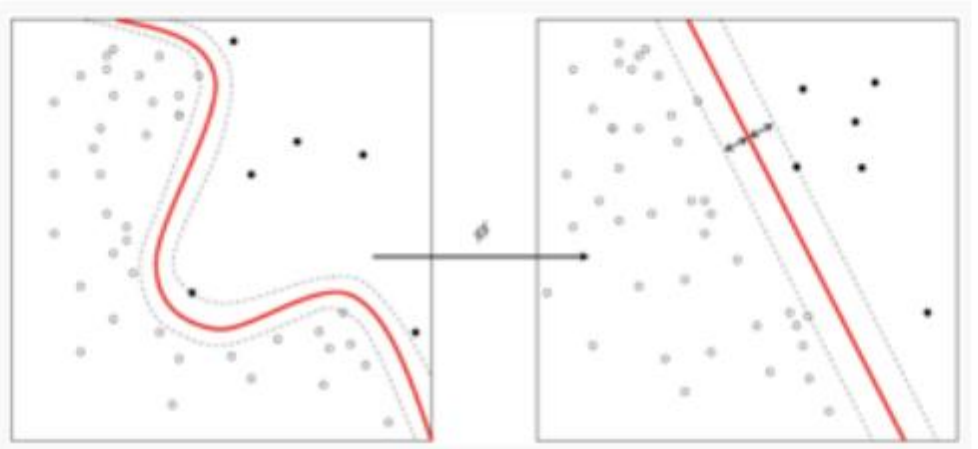
Как только плоскость найдена, берем новый объект и смотрим где он расположен относительно плоскости. Если справа, то принадлежит одному классу, если слева, то наш объект принадлежит другому классу.

Однако, ка правило на практике встречаются случаи когда объекты расположены, так что на плоскости невозможно провести разделяющую прямую. В этом случае плоскость вкладывается в пространство большей размерности. При вложении плоскость трансформируется таким образом, что бы появилась возможность провести разделяющую плоскость.



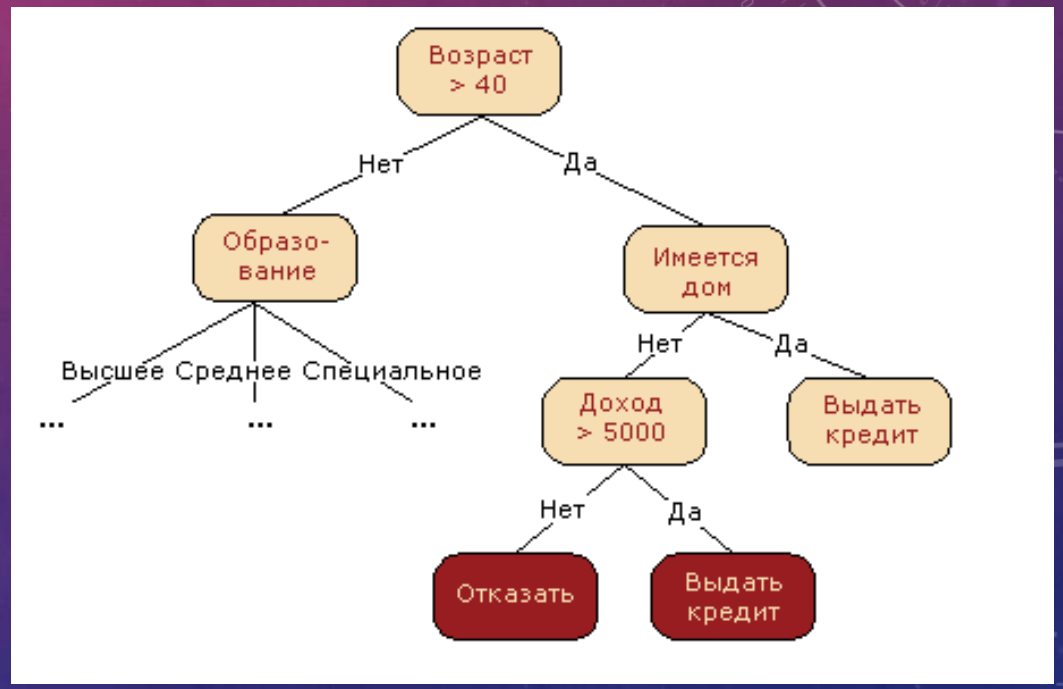
Демонстрация подобного преобразования

https://youtu.be/3liCbRZPrZA



КЛАССИФИКАТОР ДЕРЕВА РЕШЕНИЙ (DECISION TREE CLASSIFIER)

- Этот классификатор разбивает данные на всё меньшие и меньшие подмножества на основе разных критериев, т. е. у каждого подмножества своя сортирующая категория. С каждым разделением количество объектов определённого критерия уменьшается.
- Классификация подойдёт к концу, когда сеть дойдёт до подмножества только с одним объектом. Если объединить несколько подобных деревьев решений, то получится так называемый *Случайный Лес* (англ. *Random Forest*).



КЛАССИФИКАТОР ДЕРЕВА РЕШЕНИЙ (DECISION TREE CLASSIFIER)

Преимущества и недостатки

- Дерево решений дает преимущество в простоте понимания и визуализации, а также требует очень небольшой подготовки данных. Недостатком дерева решений является то, что оно может создавать сложные деревья, которые могут эффективно классифицировать боты. Они могут быть довольно нестабильными, потому что даже простое изменение данных может нарушить всю структуру дерева решений.

Примеры использования

- Исследование данных
- Распознавание образов
- Ценообразование опционов в финансах
- Выявление угроз заболеваний и рисков

НАИВНЫЙ БАЙЕСОВСКИЙ КЛАССИФИКАТОР (NAIVE BAYES)

- Такой классификатор вычисляет вероятность принадлежности объекта к какому-то классу. Эта вероятность вычисляется из шанса, что какое-то событие произойдёт, с опорой на уже на произошедшие события.
- Каждый параметр классифицируемого объекта считается независимым от других параметров.

НАИВНЫЙ БАЙЕСОВСКИЙ КЛАССИФИКАТОР (NAIVE BAYES)

Преимущества и недостатки

- Наивный байесовский классификатор требует небольшого объема обучающих данных для оценки необходимых параметров для получения результатов. Они чрезвычайно быстры по своей природе по сравнению с другими классификаторами.
- Единственным недостатком является то, что они, как известно, являются плохими оценщиками.

Примеры использования

- Предсказания заболеваний
- Классификация документов
- Спам-фильтры
- Анализ настроений

ЛИНЕЙНЫЙ ДИСКРИМИНАНТНЫЙ АНАЛИЗ (LINEAR DISCRIMINANT ANALYSIS)

- Этот метод работает путём уменьшения размерности набора данных, проецируя все точки данных на линию. Потом он комбинирует эти точки в классы, базируясь на их расстоянии от центральной точки.
- Этот метод, как можно уже догадаться, относится к линейным алгоритмам классификации, т. е. он хорошо подходит для данных с линейной зависимостью.

ЛОГИСТИЧЕСКАЯ РЕГРЕССИЯ В ЗАДАЧАХ КЛАССИФИКАЦИИ

Логистическая регрессия – это разновидность множественной регрессии, общее назначение которой состоит в анализе связи между несколькими независимыми переменными (называемыми также регрессорами или предикторами) и зависимой переменной. Бинарная логистическая регрессия, как следует из названия, применяется в случае, когда зависимая переменная является бинарной (т.е. может принимать только два значения). Иными словами, с помощью логистической регрессии можно оценивать вероятность того, что событие наступит для конкретного испытуемого (больной/здоровый, возврат кредита/дефолт и т.д.).

$$y = a + b_1 x_1 + b_2 x_2 + \ldots + b_n x_n$$

Для решения проблемы задача регрессии может быть сформулирована иначе: вместо предсказания бинарной переменной, мы предсказываем непрерывную переменную со значениями на отрезке [0,1] при любых значениях независимых переменных. Это достигается применением следующего регрессионного уравнения (логит-преобразование):

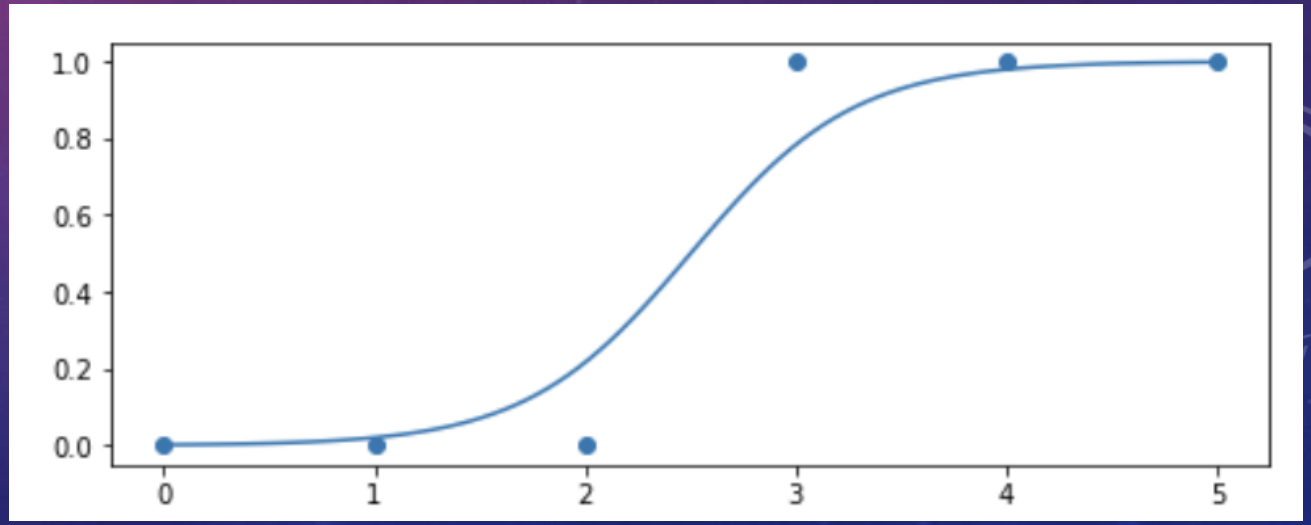
$$p = \frac{1}{1+e^{-y}},$$

где P – вероятность того, что произойдет интересующее событие; e – основание натуральных логарифмов 2,71…; y – стандартное уравнение регрессии.

ЛОГИСТИЧЕСКАЯ РЕГРЕССИЯ

- Логистическая регрессия выводит прогнозы о точках в бинарном масштабе — нулевом или единичном. Если значение чего-либо равно либо больше 0.5, то объект классифицируется в большую сторону (к единице). Если значение меньше 0.5 — в меньшую (к нулю).

- У каждого признака есть своя метка, равная только 0 или только 1. Логистическая регрессия является линейным классификатором и поэтому используется, когда в данных прослеживается какая-то линейная зависимость.



ЛОГИСТИЧЕСКАЯ РЕГРЕССИЯ

Преимущества и недостатки

- Логистическая регрессия специально предназначена для классификации, она полезна для понимания того, как набор независимых переменных влияет на результат зависимой переменной.
- Основным недостатком алгоритма логистической регрессии является то, что он работает только тогда, когда прогнозируемая переменная является двоичной, он предполагает, что в данных нет пропущенных значений, и предполагает, что предикторы независимы друг от друга.

Примеры использования

- Выявление факторов риска заболеваний
- Классификация слов
- Прогноз погоды
- Заявки на голосование

ПОДКЛЮЧАЕМЫЕ БИБЛИОТЕКИ. ОБУЧЕНИЕ МОДЕЛИ

```
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
```

```
knn = KNeighborsClassifier()
knn.fit(X_train, y_train)
```

МЕТРИКИ КАЧЕСТВА КЛАССИФИКАЦИИ

Матрица ошибок (Confusion matrix)

Матрица ошибок — это способ разбить объекты на четыре категории в зависимости от комбинации истинного ответа и ответа алгоритма.

Основные термины:

- TP — истино-положительное решение;
- TN — истино-отрицательное решение;
- FP — ложно-положительное решение (Ошибка первого рода);
- FN — ложно-отрицательное решение (Ошибка второго рода).

n=165	**Predicted: NO**	**Predicted: YES**	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
	55	110	

Accuracy — доля правильных ответов:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{P + N} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

Данная матрика имеет существенный недостаток — её значение необходимо оценивать в контексте баланса классов. Если в выборке 950 отрицательных и 50 положительных объектов, то при абсолютно случайной классификации мы получим долю правильных ответов 0.95. Это означает, что доля положительных ответов сама по себе не несет никакой информации о качестве работы алгоритма a(x), и вместе с ней следует анализировать соотношение классов в выборке.

Гораздо более информативными критериями являются точность (precision) и полнота (recall).

Точность показывает, какая доля объектов, выделенных классификатором как положительные, действительно является положительными:

$$Precision = \frac{TP}{TP + FP}$$

Полнота показывает, какая часть положительных объектов была выделена классификатором:

$$Recall = \frac{TP}{TP + FN}$$

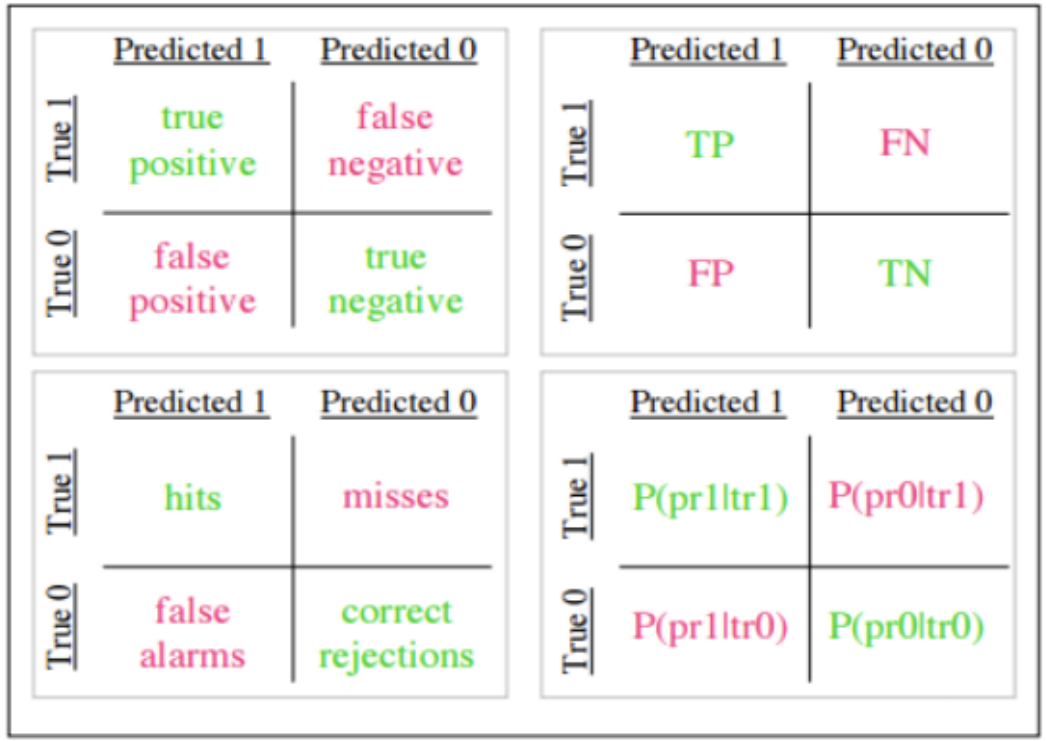
Меры качества классификаторов для бинарных классов

$$\text{Precision} = \frac{\text{TP}}{TP + FP}$$

Precision: число правильно предсказанных положительных значений поделенных на число предсказанных классификатором положительных значений.

$$\text{Recall} = \frac{\text{TP}}{TP + FN}$$

Recall : число правильно предсказанных положительных значений поделенных на число положительных ответов в данных.



Confusion Matrix

TP – число правильно предсказанных положительных значений

FN – число неправильно предсказанных положительных значений

FN – число правильно предсказанных негативных значений

FP – число неправильно предсказанных негативных значений

```python
# Оценка точности – простейший вариант оценки работы классификатора
print(accuracy_score(rtree_prediction, y_test))

# Но матрица неточности и отчёт о классификации дадут больше информации о производительности
print(confusion_matrix(rtree_prediction, y_test))
print(classification_report(rtree_prediction, y_test))
```

```
0.9888268156424581
[[109   1]
 [  1  68]]
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       110
           1       0.99      0.99      0.99        69

    accuracy                           0.99       179
   macro avg       0.99      0.99      0.99       179
weighted avg       0.99      0.99      0.99       179
```

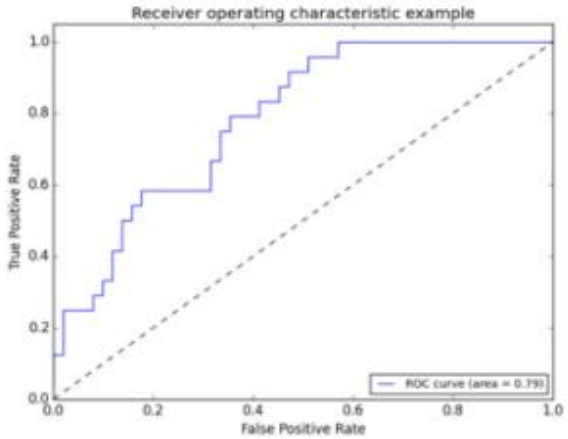
МЕРЫ КАЧЕСТВА КЛАССИФИКАТОРОВ ДЛЯ БИНАРНЫХ КЛАССОВ

F − measure

$$F - measure = \frac{(\beta^2 + 1) * precision * recall}{\beta^2 * precision + recall} \Longleftrightarrow \frac{(\beta^2+1)tp}{(\beta^2+1)tp+\beta^2 fn+fp}$$

β– обычно берут равной 1. F measure = 2 * (precision * recall) / (precision + recall)

The F measure (F1, Fscore) можно интерпретировать как взвешенное среднее precision и recall. Если F1=1, то классификатор отработал на 100% и F1=0 тогда классификатор не справился с задачей.



$$ROC = \frac{P(x|positive)}{P(x|negative)}$$

Рассчитывает отношение числа правильно распознанных случаев к числу не правильных. Процесс расчета таков: берутся данные, последовательно, и в них вычисляется это отношение. В какой то момент отношение становится константой.

AUC – интеграл под кривой.

СПАСИБО ЗА ВНИМАНИЕ!