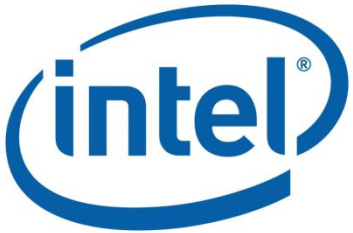
Лабораторная работа № 5
Машина опорных векторов



Бизнес-задача

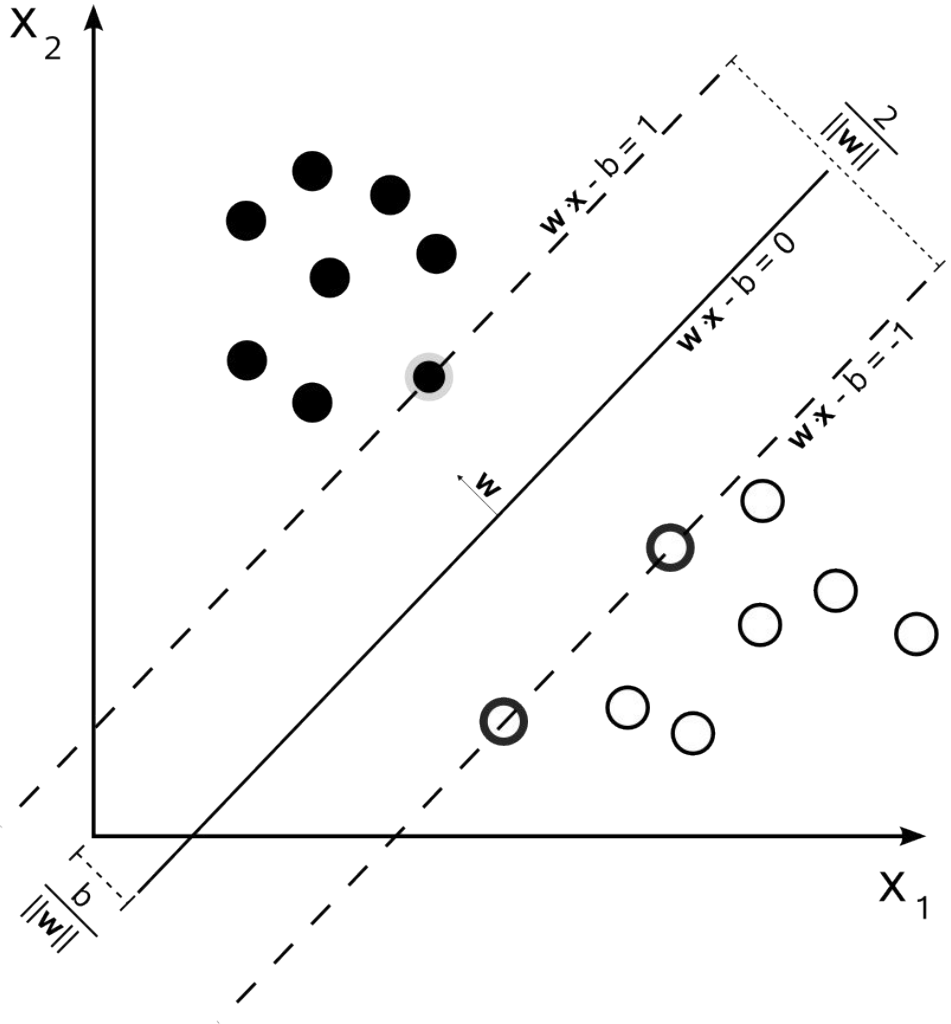
- Задача – отличить фальшивые банкноты от настоящих
- База Banknote authentification: https://archive.ics.uci.edu/ml/datasets/banknote+authentication
- Объекты представляют из себя характеристики изображений банкнот
- 1372 объекта
- 4 признака:
  - энтропия изображения
  - коэффициенты дисперсии, ассиметрии и эксцесса вейвлет-преобразования изображения

Бизнес-задача

- Способ решения – классификация
- Метод классификации – SVM (Support Vector Machine)
- Положительные стороны SVM:
  - быстрый метод классификации;
  - метод сводится к решению задачи квадратичного программирования в выпуклой области, которая обычно имеет единственное решение;
  - метод позволяет осуществлять более уверенную классификацию, чем другие линейные методы.

Метод Support Vector Machine

- Изобретён в 1963 году, авторы – Вапник, Червоненкис
- Современная постановка в 1995 году
- Классификатор – разделяющая гиперплоскость
- Гиперплоскость является наилучшей в смысле ширины «разделяющей полосы»
- Точки на границах «полосы» называются «опорными» - отсюда название
- В чём смысл увеличивать ширину полосы?


x2
x1
w x - b = 1
w x - b = 0
w x - b = -1
||w|| 2
w
b
||w||

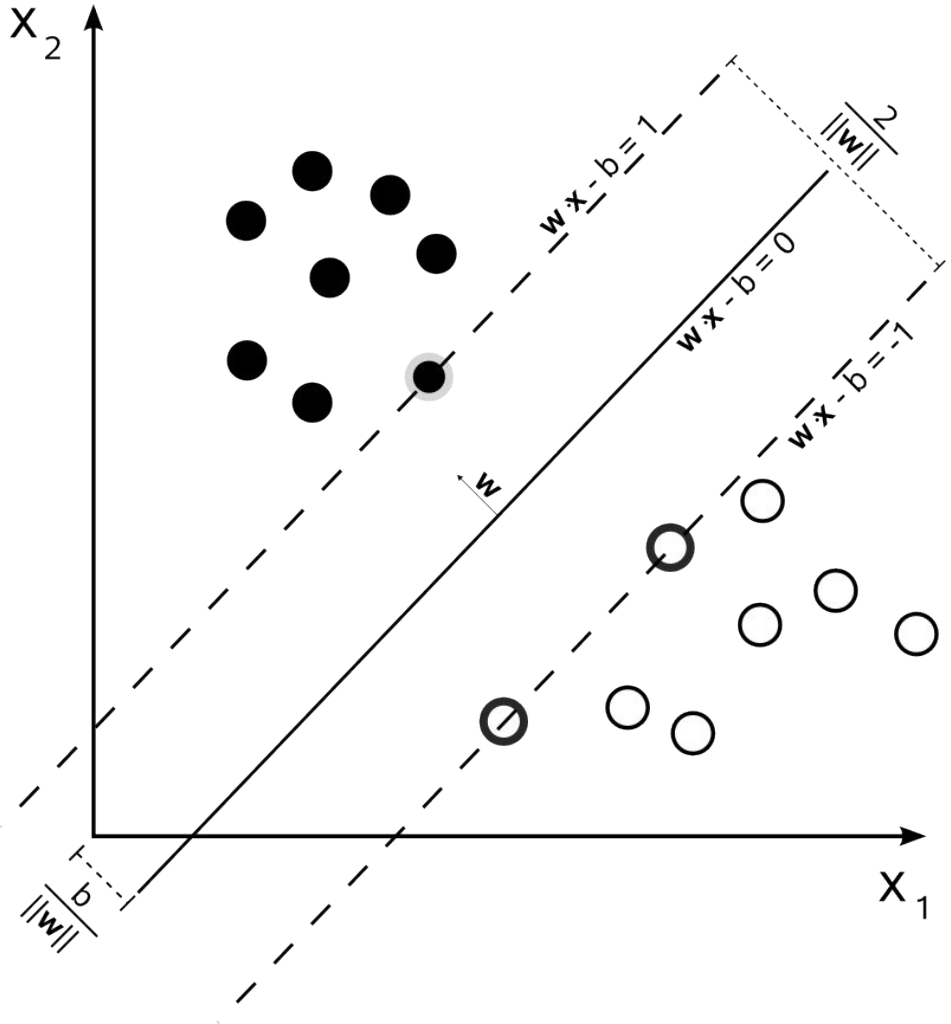
Математическая постановка

- Пусть даны два линейно разделимых класса объектов
- Мы можем описать все точки разделяющей гиперплоскости используя вектор-нормаль к этой гиперплоскости:

$$< w, x > -b = 0$$

- Данная разделяющая гиперплоскость находится в «разделяющей полосе», которую мы также можем задать уравнениями:

$$< w, x > -b = -1 \qquad < w, x > -b = 1$$


x2
x1
w x - b = 1
w x - b = 0
w x - b = -1
||w|| 2
w
b
||w||

Математическая постановка

- Можно найти ширину данной полосы как $\frac{2}{\|w\|}$
- Для машины опорных векторов необходимо найти разделяющую гиперплоскость, которая задает полосу максимальной ширины. Задача оптимизации:

$$\frac{1}{2}\|w\|^2 \to min$$

$$y_i(< w, x_i > -b) \geq 1, \forall i = 1, \dots, n$$

Решение оптимизационной задачи

- Метод множителей Лагранжа:

$$L_P = \frac{\|w\|^2}{2} - \sum_{i=1}^{n} \lambda_i (y_i (< w, x_i > + b) - 1) \rightarrow \max_{\lambda_i \geq 0} \min_{w,b}$$

- Производные:

- Двойственная проблема:

$$L_D = \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j y_i y_j < x_i, x_j > \rightarrow \max_{\lambda_i \geq 0}$$

Случай линейно неразделимой выборки

- Для неразделимых классов вводится функция потерь

$$\varepsilon_i = \max(0, 1 - y_i(< w, x_i > - b))$$

- Задача оптимизации:

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\varepsilon_i \to min$$

$$y_i(< w, x_i > - b) \geq 1 - \varepsilon_i, \quad \varepsilon_i \geq 0, \forall i = 1, \ldots, n$$

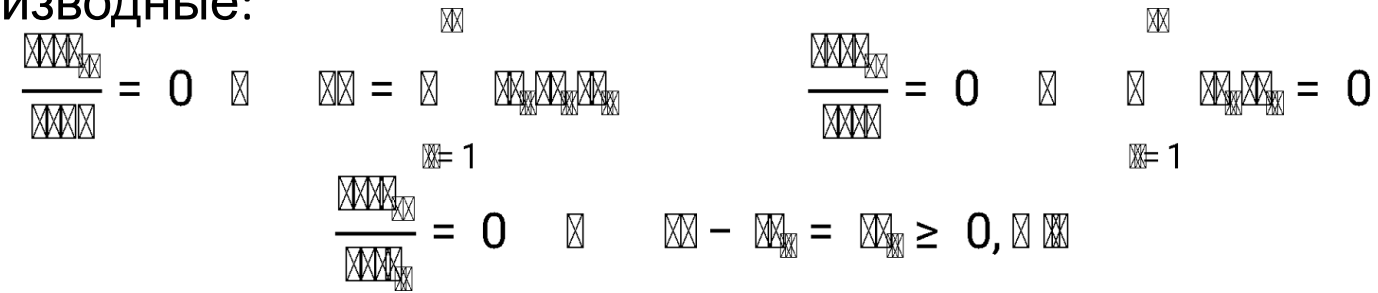
- Данная форма SVM называется C-classification.

Случай линейно неразделимой выборки

- Метод множителей Лагранжа:

$$L_P = \frac{\|w\|^2}{2} + C\sum_{i=1}^{n}\varepsilon_i - \sum_{i=1}^{n}\lambda_i(y_i(< w, x_i > +b) - 1 + \varepsilon_i) - \sum_{i=1}^{n}\beta_i\varepsilon_i \to \max_{\lambda_i \geq 0}\min_{w,b,\varepsilon_i}$$

- Производные:



$$\frac{\partial L_P}{\partial w} = 0 \quad w = \sum_{i=1}^{n}\lambda_i y_i x_i \qquad \frac{\partial L_P}{\partial b} = 0 \quad \sum_{i=1}^{n}\lambda_i y_i = 0$$

$$\frac{\partial L_P}{\partial \varepsilon_i} = 0 \quad C - \beta_i = \lambda_i \geq 0,$$

- Двойственная проблема:

$$L_D = \sum_{i=1}^{n}\lambda_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\lambda_i\lambda_j y_i y_j < x_i, x_j > \to \max_{0 \leq \lambda_i \leq C}$$

Предсказание

- Для предсказания результата алгоритма, используется функция sign:

$$F(z) = sign(\sum_{i=1}^{n} \lambda_i y_i \langle x_i, z \rangle + b)$$

- Для $\lambda_i = 0$ точка x_i не является «опорной», таким образом, в сумму, которая определяет класс нового объекта, влияние вносят только «опорные» точки.

Использование метода SVM

- Использован набор данных Banknote Authentification
- В качестве тестовой выборки взяты 107 последних объектов класса «0» (настоящие банкноты) и 119 объектов класса «1» (фальшивки). Остальные объекты используются в качестве обучающей выборки.
- Исходные параметры алгоритмов SVM взяты одинаковыми для разных библиотек

Обучение модели, Intel DAAL

```
//Создаем данные, на которых обучаем выборку
nFeatures = trainDataSource.getNumberOfColumns();
NumericTable trainData = new HomogenNumericTable(context, Double.class,nFeatures - 1, 0,
NumericTable.AllocationFlag.NotAllocate);
NumericTable trainGroundTruth = new HomogenNumericTable(context, Double.class, 1, 0,
NumericTable.AllocationFlag.NotAllocate);
MergedNumericTable mergedData = new MergedNumericTable(context);
mergedData.addNumericTable(trainData);
mergedData.addNumericTable(trainGroundTruth);
//Подразумевается, что столбец с классами располагается последним в исходных данных

trainDataSource.loadDataBlock(mergedData);
TrainingBatch algorithm = new TrainingBatch(context, Double.class, TrainingMethod.boser);

//Установка параметров алгоритма и вычисление
algorithm.parameter.setKernel(new com.intel.daal.algorithms.kernel_function.linear.Batch(context, Double.class));
algorithm.input.set(InputId.data, trainData);
algorithm.input.set(InputId.labels, trainGroundTruth);
training = algorithm.compute();
```

Предсказание, Intel DAAL

```
//Создаем данные, для которых будет сделано предсказание класса
NumericTable testData = new HomogenNumericTable(context, Double.class, nFeatures - 1, 0,
NumericTable.AllocationFlag.NotAllocate);
NumericTable testGroundTruth = new HomogenNumericTable(context, Double.class, 1, 0,
NumericTable.AllocationFlag.NotAllocate);
MergedNumericTable mergedData = new MergedNumericTable(context);
mergedData.addNumericTable(testData);
mergedData.addNumericTable(testGroundTruth);
//Подразумевается, что столбец с классами располагается последним в исходных данных

testDataSource.loadDataBlock(mergedData);
PredictionBatch algorithm = new PredictionBatch(context, Double.class, PredictionMethod.defaultDense);
//Установка параметров алгоритма и вычисление
algorithm.parameter.setKernel(new com.intel.daal.algorithms.kernel_function.linear.Batch(context, Double.class));

//Получение обученной модели
Model model = training.get(TrainingResultId.model);
algorithm.input.set(NumericTableInputId.data, testData);
algorithm.input.set(ModelInputId.model, model);
testing = algorithm.compute();
```

Предсказание, Intel DAAL

```
NumericTable predictedLabels = testing.get(PredictionResultId.prediction); //Получение результатов предсказания
//Создание метрики – таблицы сопряжённости, установка параметров
QualityMetricSetBatch metric = new QualityMetricSetBatch(context);
BinaryConfusionMatrixInput input = metric.getInputDataCollection().getInput(QualityMetricId.confusionMatrix);
input.set(BinaryConfusionMatrixInputId.predictedLabels, predictedLabels);
input.set(BinaryConfusionMatrixInputId.groundTruthLabels, testGroundTruth);
quality = metric.compute();

//Получение матрицы сопряжённости
BinaryConfusionMatrixResult qualityMetricResult = quality.getResult(QualityMetricId.confusionMatrix);
NumericTable binaries = qualityMetricResult.get(BinaryConfusionMatrixResultId.binaryMetrics);
DoubleBuffer buf = DoubleBuffer.allocate(6);
buf = binaries.getBlockOfRows(0,1,buf);

//Распечатка известных метрик по таблице сопряжённости
System.out.println("Accuracy: " + buf.get(0));
System.out.println("Precision: " + buf.get(1));
System.out.println("Recall: " + buf.get(2));
System.out.println("fscore: " + buf.get(3));
System.out.println("Specifity: " + buf.get(4));
System.out.println("AUC: " + buf.get(5));
```

Алгоритм SVM, Python, R

```
import numpy as np
import pandas as np
from sklearn import svm

#Считывание файлов с данными
train = np.genfromtxt("data.csv", delimiter=',')
test = np.genfromtxt("test.csv", delimiter=',')

#Построение модели SVM и предсказание
classifier = svm.SVC()
classifier.fit(train[:,0:4],train[:,4])
prediction = classifier.predict(test[:,0:4])

#Анализ результатов с помощью матрицы сопряжённости
tab = pd.crosstab(index = prediction, columns= test[:,4])
print(tab)
```

```
#Считывание файлов с данными
data.train = read.csv("data.csv",header = FALSE)
data.test = read.csv("test.csv",header = FALSE)

#Построение модели SVM и предсказание
svm.model = svm(V5~., data = data.train)
prediction = predict(svm.model,data.test)


#Анализ результатов с помощью матрицы сопряжённости
table(prediction,data.test$V5)
```

Результаты вычислений для данных Banknote Authentification

- Результаты вычислений(суммарное время получения модели + предсказание, количество ошибок) с одинаковыми параметрами:
  - DAAL: ~0.021 секунды, 2 ошибки
  - Python: ~0.007 секунды, 2 ошибки
  - R: ~0.14 секунды, 7 ошибок

Ядра (Kernel Trick)

- Ядро – функция специального вида:

$$K(x, x') = < \varphi(x), \varphi(x') >$$

  - Симметричная
  - Неотрицательно определенная
- Ядро используется вместо линейного скалярного произведения точек:

$$L_D = \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j y_i y_j * K(x_i, x_j) \rightarrow \max_{0 \leq \lambda_i \leq C}$$

Kernel Trick

- Функция ядра переводит точки в пространство большей размерности. Пример:

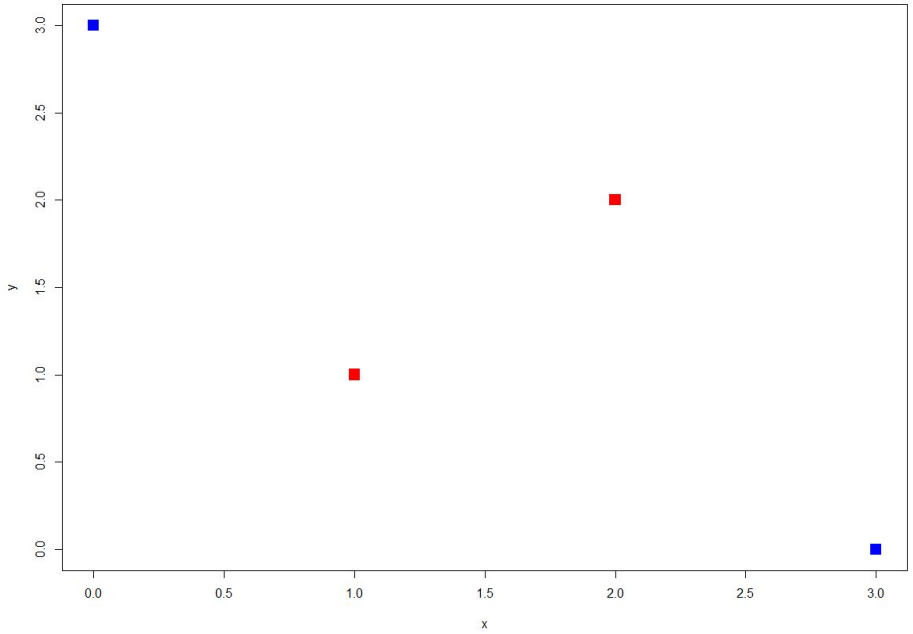
$$K(x,y) = (< x, y >)^2$$

$$K(x,y) = (x_1 y_1 + x_2 y_2)^2 = x_1^2 y_1^2 + x_2^2 y_2^2 + 2 x_1 y_1 x_2 y_2$$

$$x = (x_1, x_2) \qquad \varphi(x) = (x_1^2, x_2^2, \sqrt{2} x_1 x_2)$$

Пример использования ядра

- Пусть даны точки из двух классов:
  - A: (0,3);(3,0)
  - B: (1,1);(2,2)
  - Неразделимы
- Для полиномиального ядра:
  - A: (0,9,0);(9,0,0)
  - B: $(1,1,\sqrt{2});(4,4,4\sqrt{2})$
  - Разделимы: $z = \frac{\sqrt{2}}{2}$



Виды ядер

- Полиномиальное: $k(x, x') = (< x, x' > + c)^d$
- Гауссова радиальная базисная функция:

$$k(x, x') = e^{-\gamma \|x - x'\|^2}, \gamma > 0$$

- Сигмоид:

$$k(x, x') = \tanh(k * < x, x' > + c), k > 0, c < 0$$

Предсказание

- Для предсказания результата алгоритма, используется функция sign:

$$F(z) = sign(\sum_{i=1}^{n} \lambda_i y_i * K(x_i, z) + b)$$

- Для $\lambda_i = 0$ точка x_i не является «опорной», таким образом, в сумму, которая определяет класс нового объекта, влияние вносят только «опорные» точки.

Результаты вычислений для данных Banknote Authentification

- Результаты вычислений (суммарное время получения модели + предсказание, количество ошибок):
  - DAAL: ~2.5 секунды, 2 ошибки
  - Python: ~0.014 секунды, 0 ошибок
  - R: ~0.065 секунды, 1 ошибка
- Почему результаты получились разные?

*проверить исходные параметры алгоритмов, такие как ядро, параметр C

Результаты вычислений для данных Banknote Authentification

- Результаты вычислений(суммарное время получения модели + предсказание, количество ошибок) с ядром radial и стандартными параметрами(200 MB кэш, большое количество итераций):
  - DAAL: ~0.024 секунды, 0 ошибок
  - Python: ~0.007 секунды, 0 ошибок
  - R: ~0.065 секунды, 1 ошибка

Результаты вычислений для данных Banknote Authentification

- Результаты вычислений(суммарное время получения модели + предсказание, количество ошибок) с ядром radial и одинаковыми заданными параметрами (gamma = 1,iterations = no_limit):
  - DAAL: ~0.035 секунды, 0 ошибок
  - Python: ~0.061 секунды, 0 ошибок
  - R: ~0.052 секунды, 0 ошибок

Набор данных Adult Income*

- Набор данных Adult Income содержит объекты, каждый объект – социальные характеристики некоторого человека (возраст, пол, профессиональная деятельность и т.п.)
- Задача – классифицировать объекты по уровню дохода (<50k, >=50k).

*https://archive.ics.uci.edu/ml/datasets/Adult

Признаки

Признак	Описание
Возраст	Целое число
Рабочий класс	
Вес	Характеристика, рассчитанная из известных социо-экономических показателей
Образование1	Уровень полученного образования
Образование2	Числовая характеристика уровня образования
Семейное положение	
Род занятий	Сфера рабочей деятельности

Признак	Описание
Специальность	Тип профессии
Отношения	Кем является объект (муж, жена,…)
Расовая принадлежность	
Пол	
Рост капитала	
Потеря капитала	
Рабочие часы в неделю	
Страна рождения	

Набор данных Adult Income*

- В обучающей выборке содержатся 30162 объекта (22653 объекта класса <50k), в тестовой выборке содержатся 15060 объектов (11360 объектов класса <50k)
- Случайный классификатор даёт математическое ожидание точности как 0.63.

Результаты вычислений для данных Adult Income

- Результаты вычислений:
  - DAAL: ~400 секунд, точность 0.63 (линейное ядро, после бинаризации, 200000 итераций, точность 0.001)
  - Python: ~20 секунд, точность 0.75(линейное ядро, после бинаризации, 100000 итераций, точность 0.001, сходимость не достигнута)
  - Python: ~40 секунд, точность 0.56(линейное ядро, после бинаризации, 200000 итераций, точность 0.001, сходимость не достигнута)
  - Python: ~2400 секунд, точность 0.79(линейное ядро, после бинаризации, без ограничения на количество итераций, точность 0.001, сходимость достигнута)
  - Python ~150 секунд, точность 0.75(ядро rbf, после бинаризации,

Результаты вычислений для данных Adult Income

DAAL 0.63 Result\Source	<=50k	>50k
<=50k	8718	2871
>50k	2642	829

Python 0.79 Result\Source	<=50k	>50k
<=50k	10819	2604
>50k	541	1096

R 0.84 Result\Source	<=50k	>50k
<=50k	10660	1588
>50k	700	2112

Плюсы и минусы SVM

- Плюсы:
  - это наиболее быстрый метод нахождения решающих функций;
  - метод сводится к решению задачи квадратичного программирования в выпуклой области, которая всегда имеет единственное решение;
  - метод находит разделяющую полосу максимальной ширины (для заданных параметров), что позволяет в дальнейшем

Практическое задание

1. Проанализировать разные результаты для набора данных Banknote Authentification, в чём разница базовых настроек алгоритма в разных инструментах?
2. Найти наилучшие параметры для данных Banknote Authentification, используя технику кросс-валидации.
3. Возможно ли улучшить точность алгоритма для данных Adult Income, используя другие параметры (gamma, C, параметры связанные

Ссылки на реализации алгоритма

1. http://www.csie.ntu.edu.tw/~cjlin/papers/libsvm.pdf - *LIBSVM: A Library for Support Vector Machines*, Chih-Chung Chang and Chih-Jen Lin; статья с описанием реализации SVM – модели, алгоритмы(R,Python).

2. http://www.csie.ntu.edu.tw/~cjlin/papers/quadworkset.pdf - *Working Set Selection Using Second Order Information for Training Support Vector Machines*, Rong-En Fan, Pai-Hsuen Chen, Chih-Jen

Литература

1. Charu C. Aggarwal. *Data Mining. The Textbook.* Springer International Publishing Switzerland, 2015.

2. http://www.ccas.ru/voron/download/SVM.pdf - К. В. Воронцов. *Лекции по методу опорных векторов*, 2007