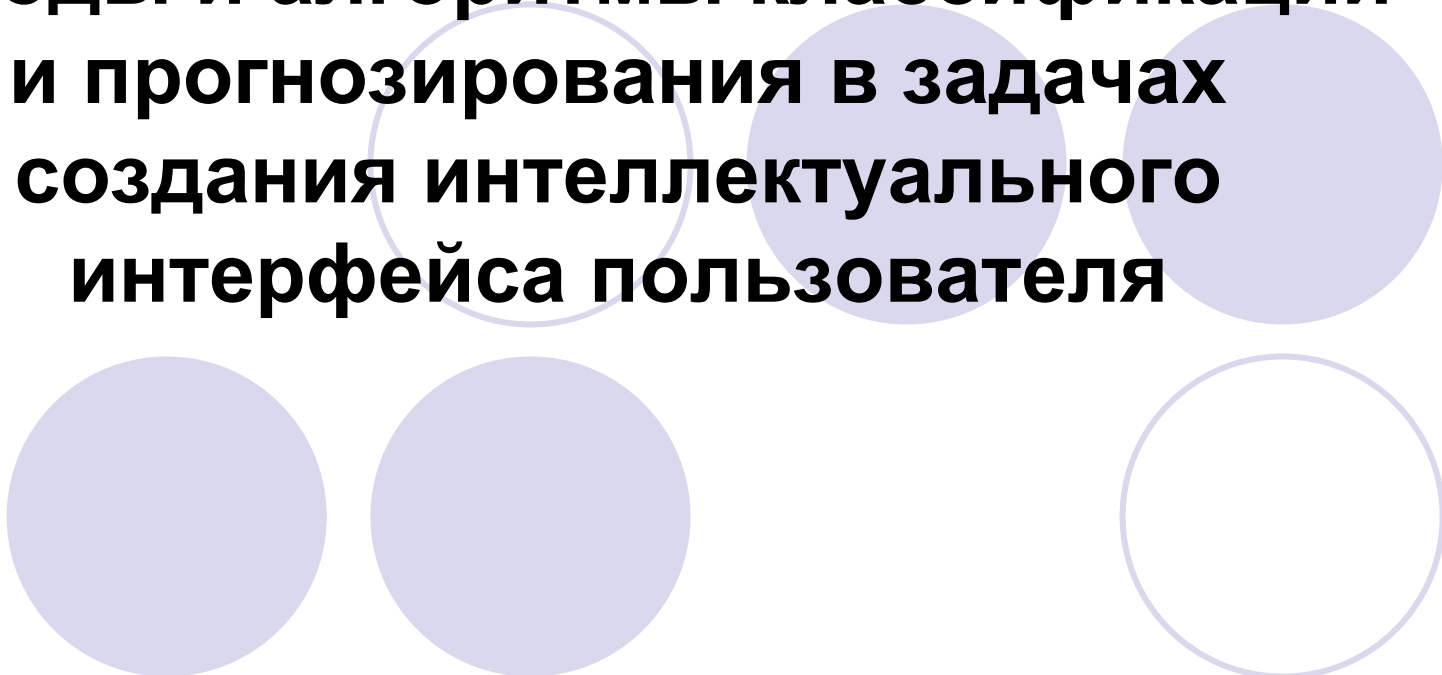


# Методы и алгоритмы классификации и прогнозирования в задачах создания интеллектуального интерфейса пользователя



Выполнил

Студент гр.ДА-72

Давыденко К.П.

Руководитель

к.т.н., с.н.с. Киселев Г.Д.

# Интеллектуальный интерфейс (определение)

- ИНТЕЛЛЕКТУАЛЬНЫЙ ИНТЕРФЕЙС (intelligent interface) - интерфейс, обеспечивающий взаимодействие пользователя с ЭВМ на естественном языке. Как правило, включает диалоговый процессор, интерпретирующий профессиональный язык пользователя, и планировщик, преобразующий описание задачи в программу ее решения на основе информации, хранящейся в базе знаний

# Интеллектуальный интерфейс (задачи)

- **"Интеллектуальные" интерфейсы** расширяют взаимодействие между человеком и компьютером с помощью:
  - увеличения диапазона способов ввода и вывода, посредством которых происходит взаимодействие;
  - обогащения грамматики ввода и вывода;
  - кооперации с пользователем в достижении целей задачи.

# Что такое классификация?

- **Задача классификации** — формализованная задача, в которой имеется множество объектов (ситуаций), разделённых некоторым образом на классы. Задано конечное множество объектов, для которых известно, к каким классам они относятся. Это множество называется выборкой. Классовая принадлежность остальных объектов не известна.
- **Классифицировать объект** — значит, указать номер (или наименование) класса, к которому относится данный объект.

# Методы классификации



- Метрические
- Логические
- Статические (бейесовские)
- Методы регрессионного анализа
- Нейросетевые методы
- Линейные методы

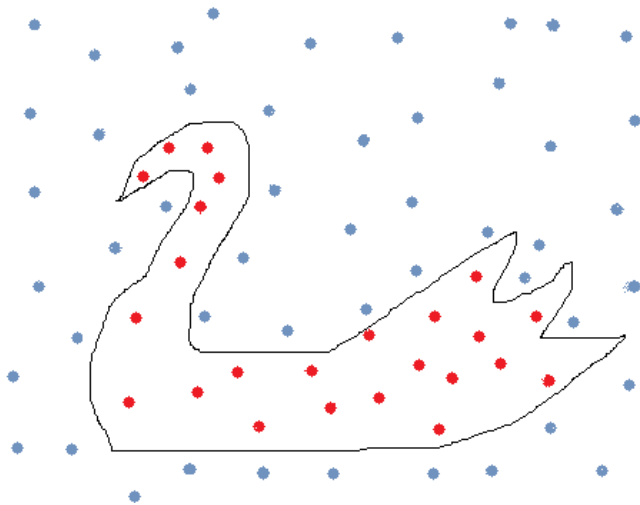
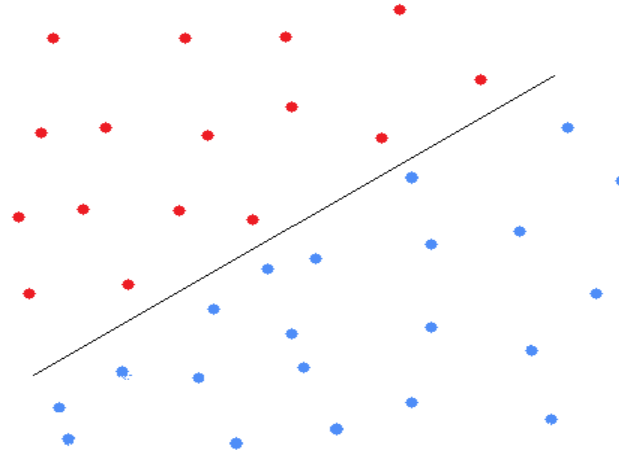


# Обучение с учителем

- Линейные методы классификации относятся к методам обучения с учителем, т.е. для того чтобы классифицировать какой-либо объект, необходимо сначала ввести шаблоны классов (обучающее множество).
- Для исследования были выбраны методы SVM (Support Vector Machine, Метод Опорных Векторов) и KDA (Kernel Discriminant Analysis, Ядерный Дискриминантный Анализ).

# Виды разделимости данных

1) Линейная  
разделимость



2) Нелинейная  
разделимость



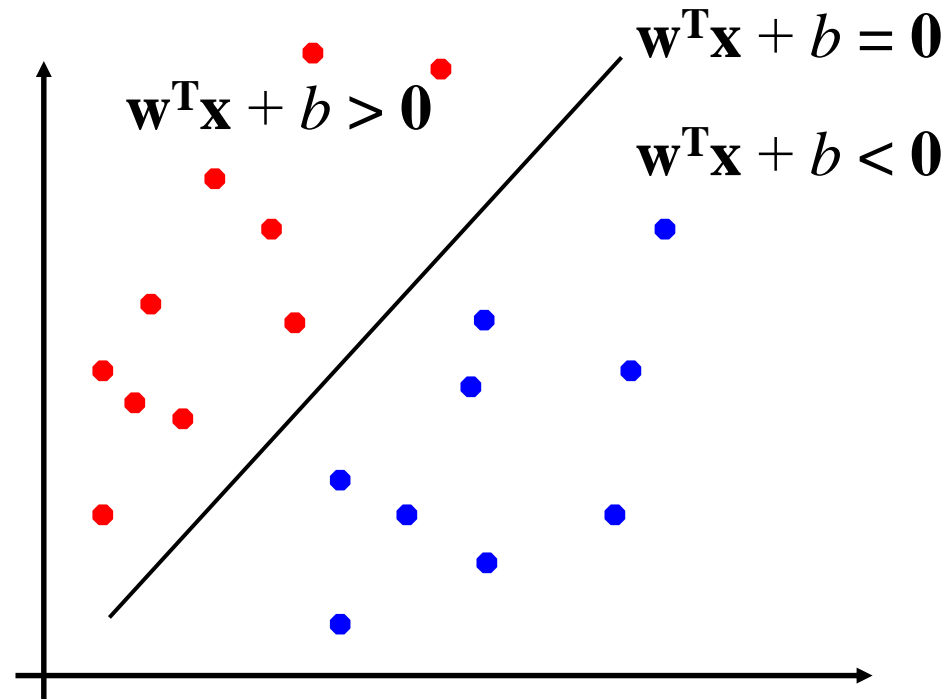
# Линейная разделимость (SVM)

- Самый простой классификатор в данном случае – гиперплоскость

$$S = \{ \mathbf{x} \mid \langle \mathbf{w}, \mathbf{x} \rangle + b = 0 \}$$

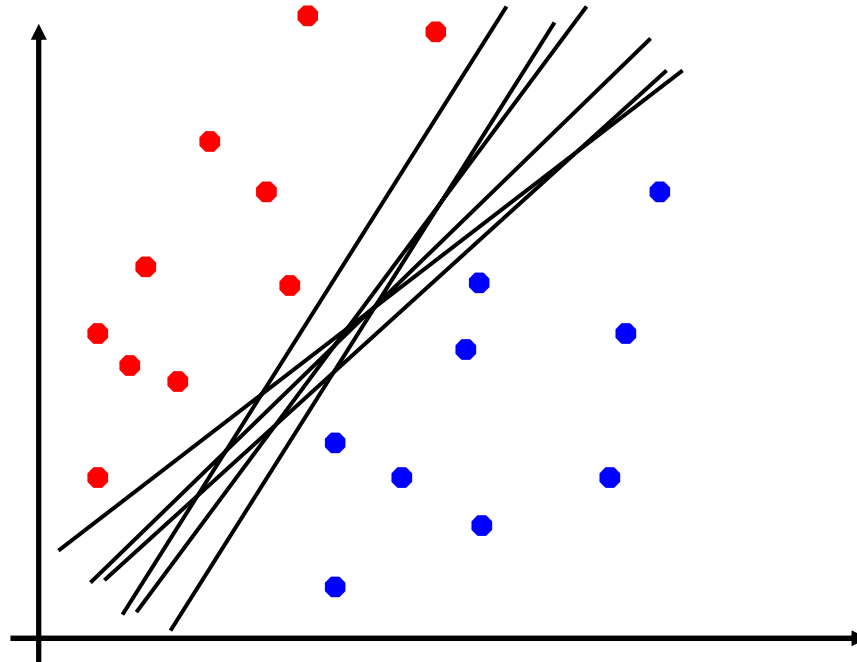
- Выбрать  $\mathbf{w}$  и  $b$  исходя из информации содержащейся в обучающем множестве
- Необходимо предсказать с какой стороны лежит новая точка

$$f_X(x_{\text{новый}}) = \text{sign}(\langle w, x_{\text{новый}} \rangle + b)$$





Какая разделяющая плоскость будет наилучшей?



# Линейная разделимость (SVM)

Определение линейно разделимых множеств:

$$\left. \begin{array}{l} \langle \mathbf{w}, \mathbf{x}_i \rangle + b \geq 1 \text{ для } y_i = +1 \\ \langle \mathbf{w}, \mathbf{x}_i \rangle + b \leq -1 \text{ для } y_i = -1 \end{array} \right\} y_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$$

Для поиска оптимальной гиперплоскости необходимо максимизировать :

$$\rho(\mathbf{w}_o, b_o) = \frac{2}{\|\mathbf{w}_o\|}$$

Т.е. найти  $\mathbf{w}$  и  $b$  удовлетворяющих

$$y_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 \text{ для } i = 1, 2, \dots, N$$

И минимизировать ф-ию

$$\Phi(\mathbf{w}) = \frac{1}{2} \mathbf{w}^T \mathbf{w}$$

# Поиск минимума

Формулируя эту задачу в терминах метода Лагранжа, получаем, что необходимо найти минимум по  $w$ ,  $b$ , и максимум по  $\lambda_i$  функции

$$\frac{1}{2}w \cdot w - \sum_i \lambda_i (y_i (w \cdot x_i - b) - 1),$$

при условии  $\lambda_i \geq 0$ .

Необходимым условием метода Лагранжа является равенство нулю производных Лагранжиана по переменным  $w$  и  $b$ . Взяв производную целевой функции по  $w$ , выражаем вектор  $w$  через множители Лагранжа:

$$w = \sum_i \lambda_i y_i x_i.$$

Теперь уравнение разделяющей гиперплоскости выглядит так:

$$\sum_i \lambda_i y_i x_i \cdot x - b = 0,$$

где  $x_i$  - это объект, который мы хотим классифицировать.

# Линейная разделимость (KDA)

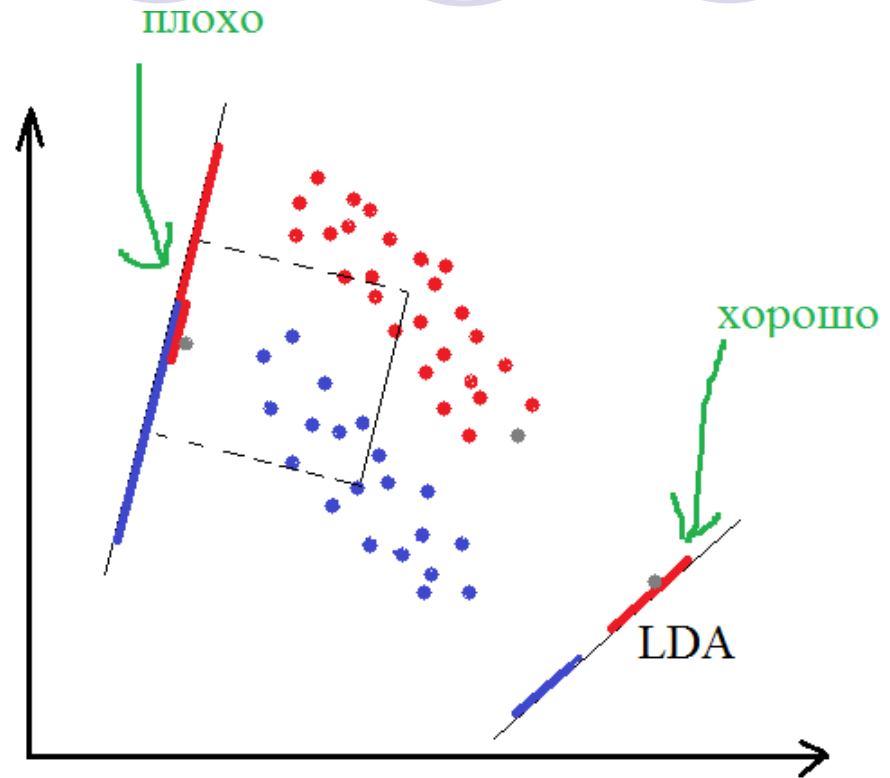
Простейшее решение - провести прямую, соединяющую центры классов и проецировать точки на неё - не подходит, так как при этом классы практически полностью перекрываются. Решение состоит в том, чтобы найти ось, проекция на которую максимизирует величину  $J(a)$  - отношение общей дисперсии выборки к сумме дисперсий внутри отдельных классов:

$$J(a) = \frac{a^T S_b a}{a^T S_w a}$$

$$S_b = \sum_{k=1}^c m_k (\mu^{(k)} - \mu)(\mu^{(k)} - \mu)^T$$

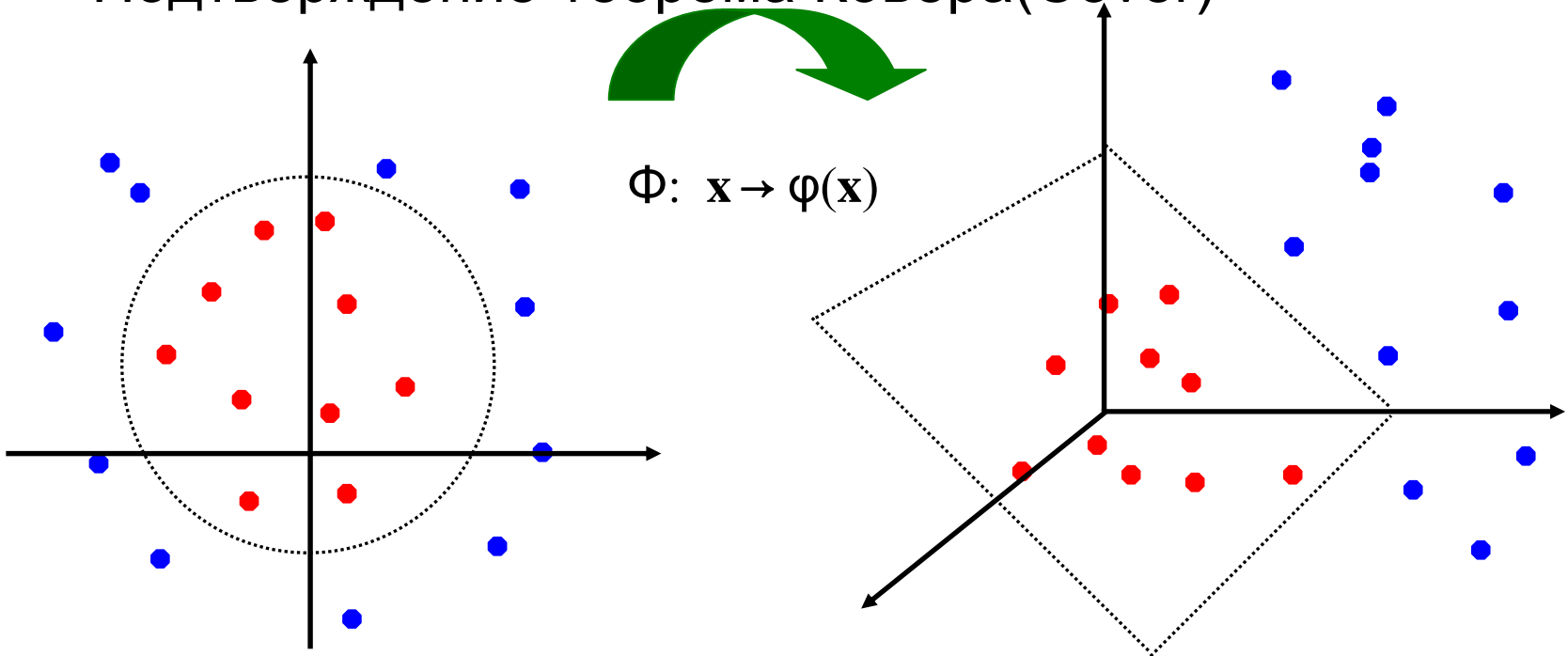
$$S_w = \sum_{k=1}^c \left( \sum_{i=1}^{m_k} (x_i^{(k)} - \mu^{(k)})(x_i^{(k)} - \mu^{(k)})^T \right)$$

где  $\mu$  - глобальный центроид,  $m_k$  - количество объектов в  $k$ -ом классе,  $\mu^{(k)}$  - центроид  $k$ -го класса,  $x_i^{(k)}$  -  $i$ -ый объект в  $k$ -ом классе



# Идея Kernel SVM (нелинейный случай)

- Исходное пространство может быть отображено в пространство более высокой размерности, где множество станет линейно-разделимым.
- Подтверждение-теорема Ковера(Cover)



# Kernel Trick

- Каждое ядро должно быть представимо в виде

$$K(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle$$

- Теорема Мерсера: каждая симметричная, положительно полуопределенная функция является ядром, т.е. матрица  $K$  должна быть положительно полуопределенной

$K =$

|                                 |                                 |                                 |     |                                 |
|---------------------------------|---------------------------------|---------------------------------|-----|---------------------------------|
| $K(\mathbf{x}_1, \mathbf{x}_1)$ | $K(\mathbf{x}_1, \mathbf{x}_2)$ | $K(\mathbf{x}_1, \mathbf{x}_3)$ | ... | $K(\mathbf{x}_1, \mathbf{x}_n)$ |
| $K(\mathbf{x}_2, \mathbf{x}_1)$ | $K(\mathbf{x}_2, \mathbf{x}_2)$ | $K(\mathbf{x}_2, \mathbf{x}_3)$ |     | $K(\mathbf{x}_2, \mathbf{x}_n)$ |
|                                 |                                 |                                 |     |                                 |
| ...                             | ...                             | ...                             | ... | ...                             |
| $K(\mathbf{x}_n, \mathbf{x}_1)$ | $K(\mathbf{x}_n, \mathbf{x}_2)$ | $K(\mathbf{x}_n, \mathbf{x}_3)$ | ... | $K(\mathbf{x}_n, \mathbf{x}_n)$ |

- Также нам не требуется знать как выглядит на самом деле пространство, где строится гиперплоскость, нужны только значения ядра как меры близости между двумя векторами

# Примеры ядерных функций

- Линейное ядро:  $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$
- Полиномиальное:  $K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i^T \mathbf{x}_j)^p$

- Гауссовское (RBF) ядро:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$$

- Сигмоидальное:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\beta_0 \mathbf{x}_i^T \mathbf{x}_j + \beta_1)$$

# Нелинейный случай (SVM)

- Решение:

$$f(x) = \sum_i \alpha_i y_i K(x_i, x_j) + b$$

- Вычисления в пространствах огромных размерностей становится возможным благодаря использованию ядер
- Методы для определения  $\alpha_i$  остаются неизменными.



# Нелинейный случай (KDA)

Используя линейное отображение  $\phi : \mathbb{R}^n \rightarrow F$

$$S_b^\phi = \sum_{k=1}^c m_k (\mu_\phi^{(k)} - \mu_\phi)(\mu_\phi^{(k)} - \mu_\phi)^T$$

$$S_w^\phi = \sum_{k=1}^c \left( \sum_{i=1}^{m_k} (\phi(x_i^{(k)}) - \mu_\phi^{(k)})(\phi(x_i^{(k)}) - \mu_\phi^{(k)})^T \right)$$

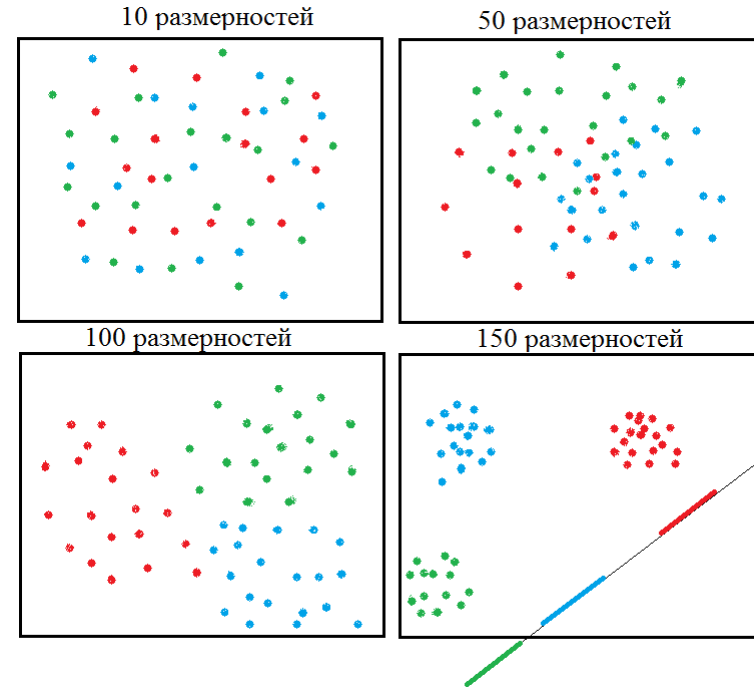
$$J(a) = \frac{a^T S_b^\phi a}{a^T S_w^\phi a}$$

Что эквивалентно

$$J(a) = \frac{a^T K W K a}{a^T K K a}$$

где  $K$ -ядерная матрица ( $K_{ij} = K(x_i, x_j)$ ),  $W$ -весовая матрица

$$W_{ij} = \begin{cases} 1/m_k, & \text{если } x_k \text{ и } x_j \text{ оба принадлежат } k\text{-ому классу} \\ 0, & \text{в другом случае} \end{cases}$$



# Программа (обучение)

Form1

класс 1

класс 2

класс 3

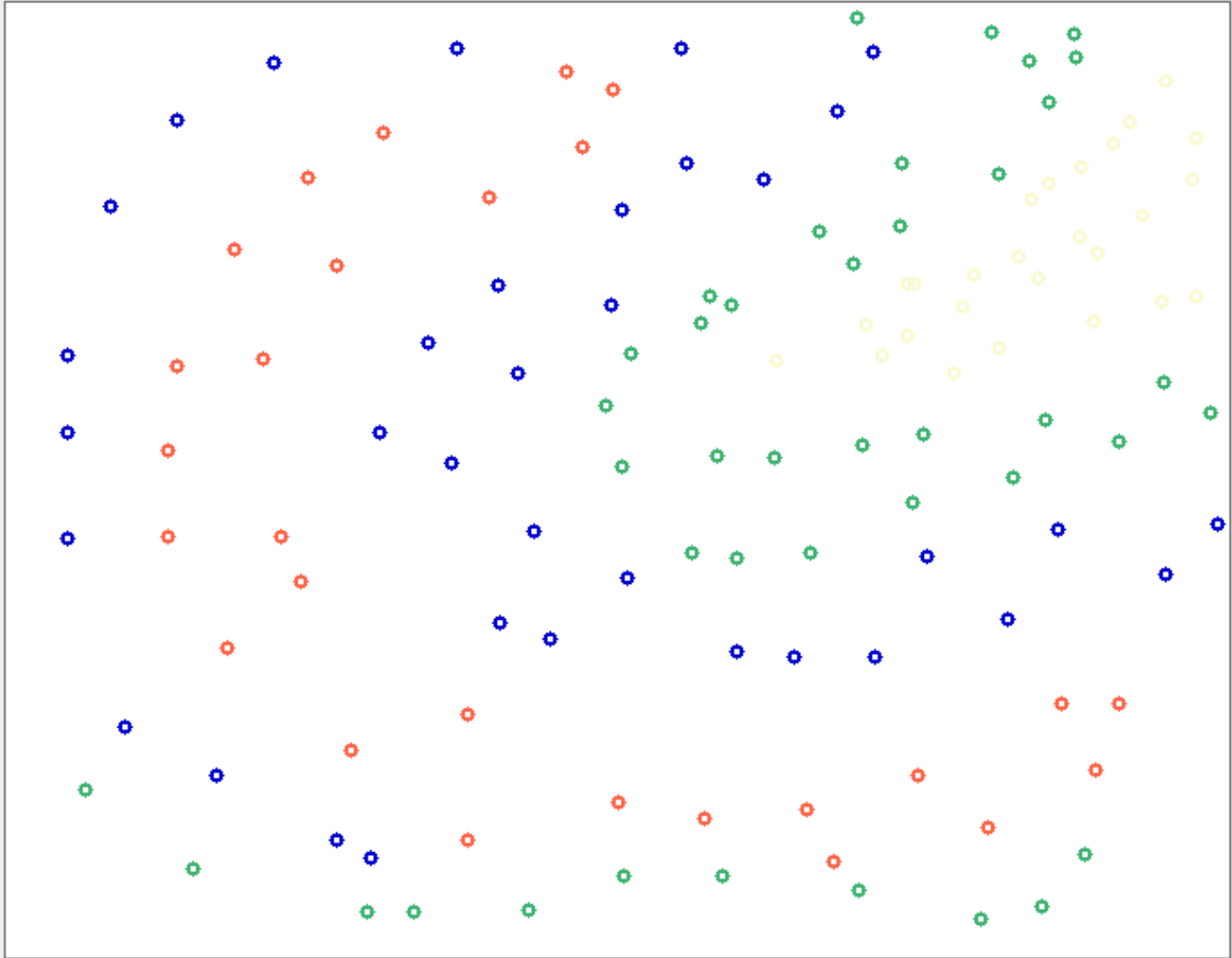
класс 4

Ядро

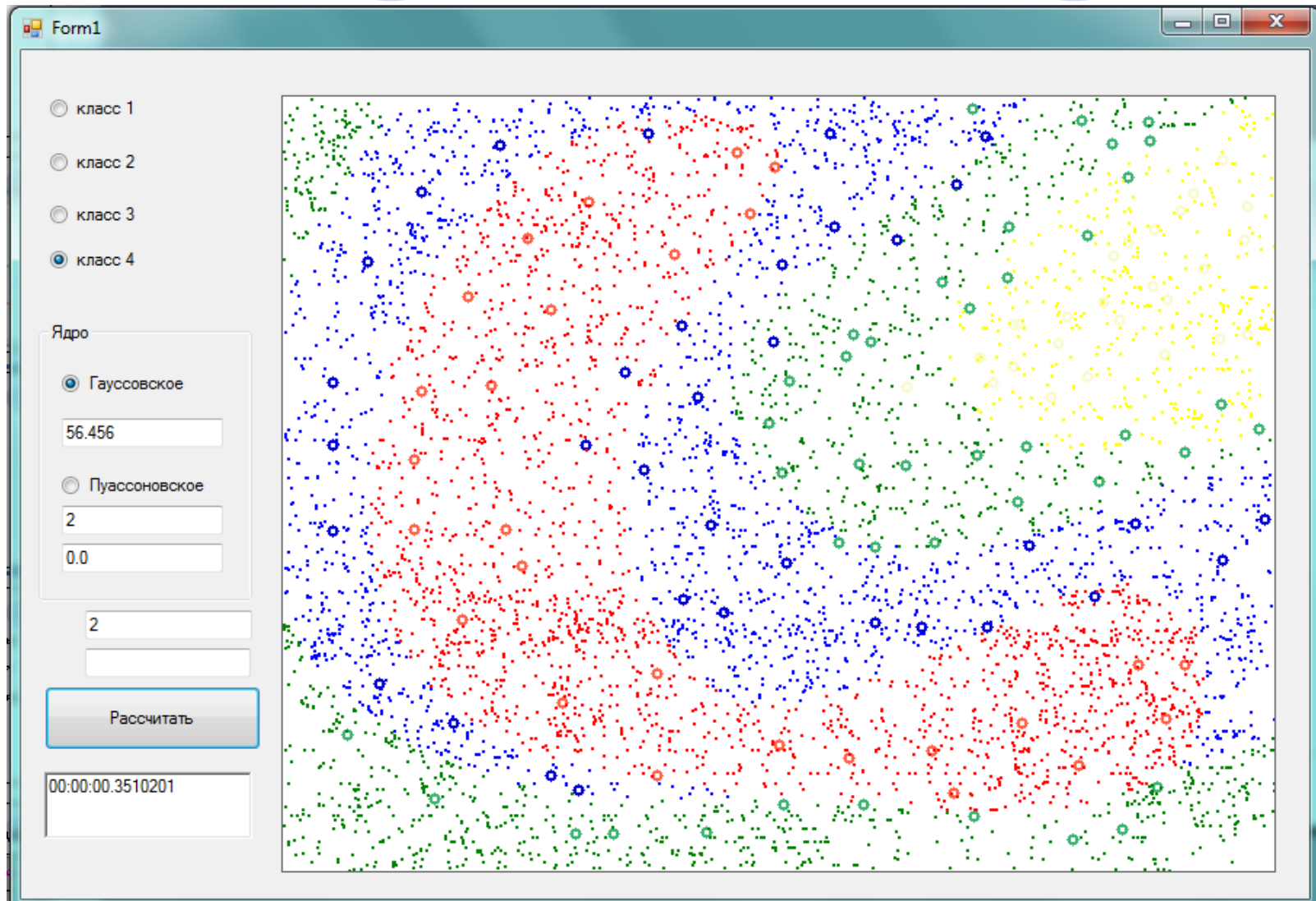
Гауссовское

Пуассоновское

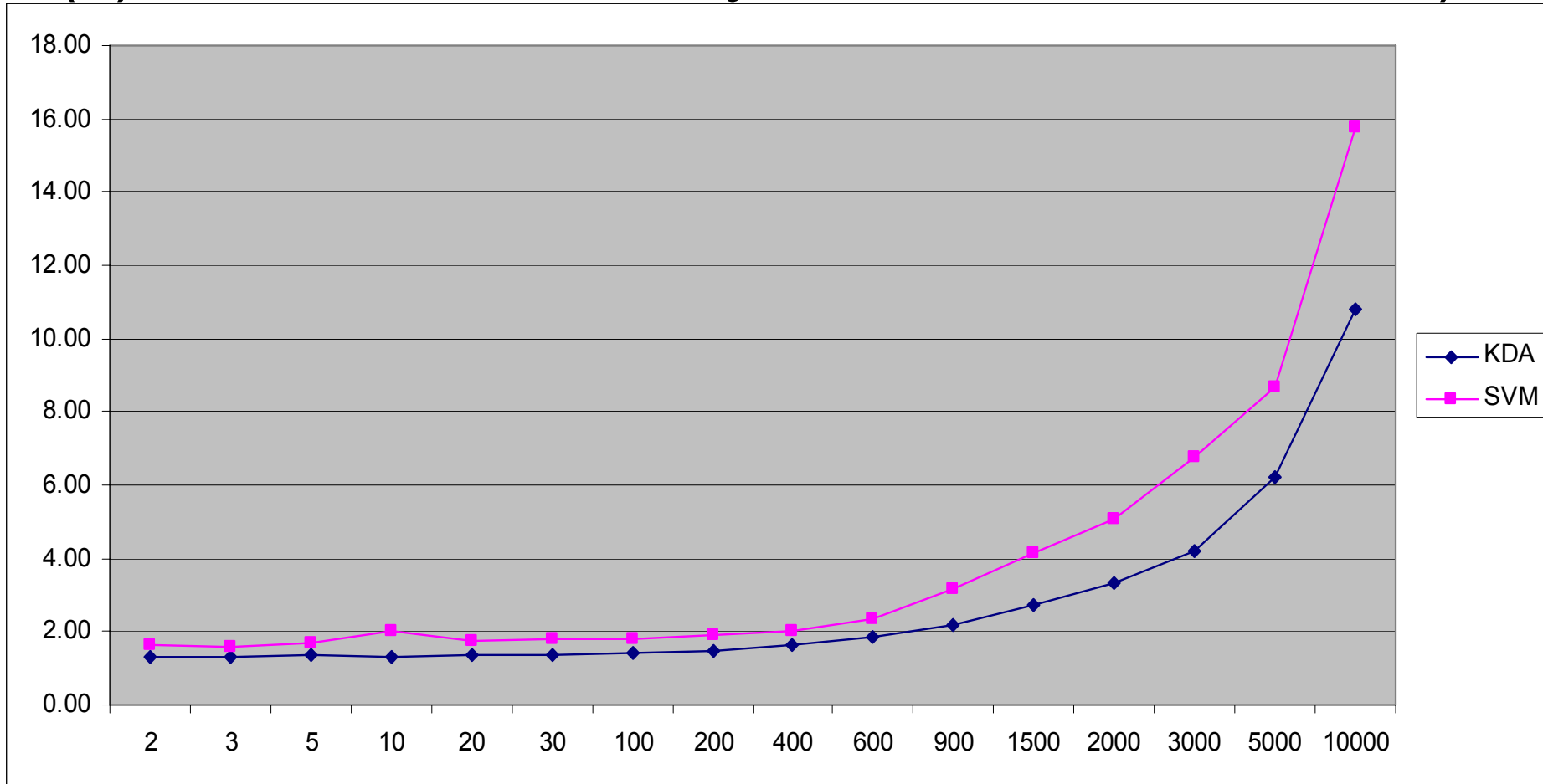
Рассчитать



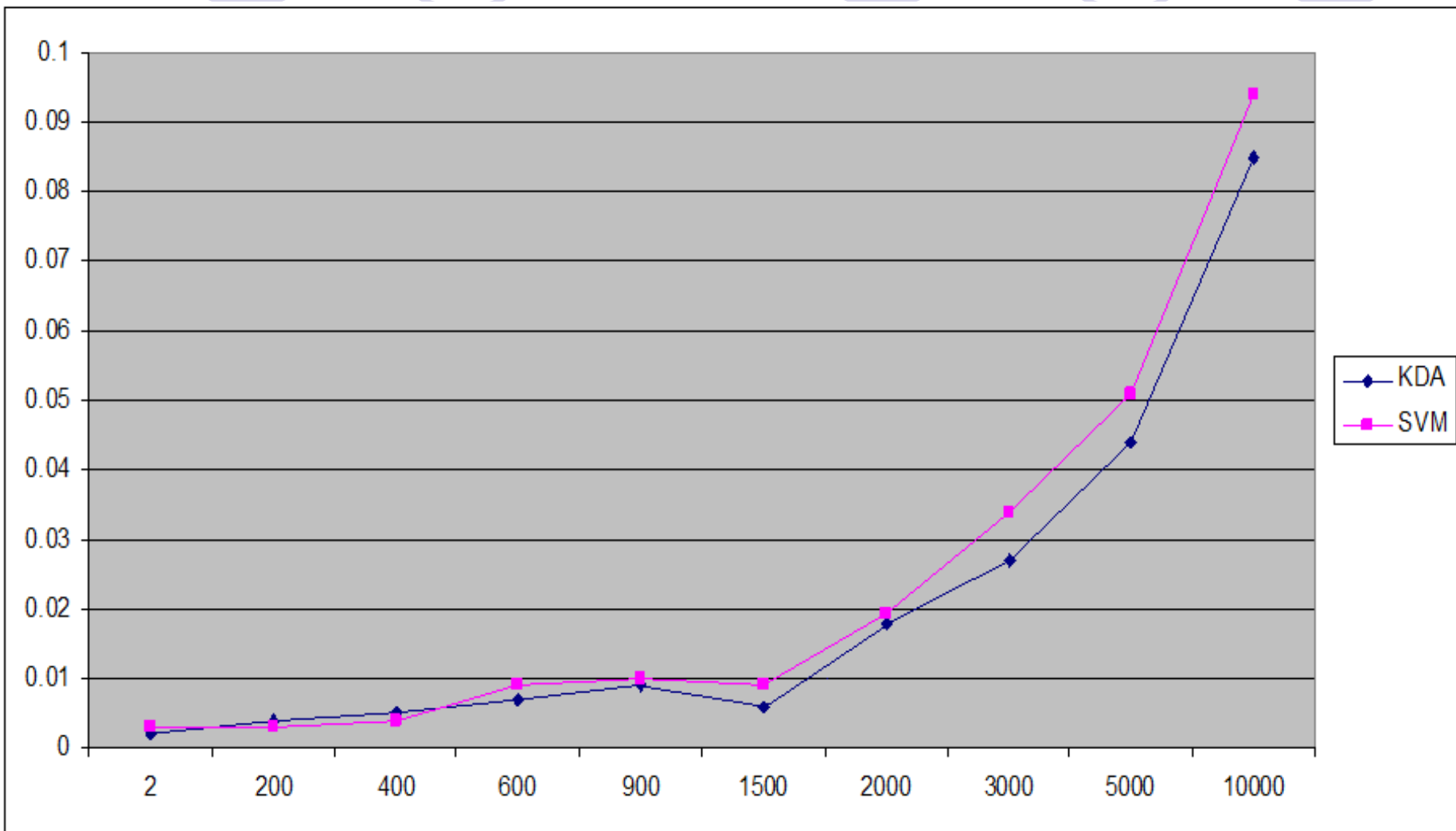
# Программа (классификация)



# Сравнение SVM и KDA (скорость нахождения гиперплоскости и проекции (с) от количества обучающего множества)




# Сравнение SVM и KDA (время классификации (с) от количества обучающего множества)



# Классификации рукописных букв

```
proj... #include "stdafx.h"
```

```
C:\Windows\system32\cmd.exe  
cascade=00624AA0  
storage=03106FB0  
hm?  
sigma=1000 Accuracy:0.42857 147/343 match  
sigma=2000 Accuracy:0.71429 245/343 match  
sigma=3000 Accuracy:0.71429 245/343 match  
sigma=4000 Accuracy:0.69388 238/343 match  
sigma=5000 Accuracy:0.75510 259/343 match  
sigma=6000 Accuracy:0.73469 252/343 match  
sigma=7000 Accuracy:0.79592 273/343 match  
sigma=8000 Accuracy:0.81633 280/343 match  
sigma=9000 Accuracy:0.77551 266/343 match  
sigma=10000
```



A video window titled 'video' displays a handwritten digit '3' on a white background. The digit is drawn with thick black strokes and has a slightly irregular, cursive appearance.

```
/// <param name="sigma">The standard deviation for the Gaussian dist.  
public: void Gaussian(double sigma)  
{  
    sigma = sigma;
```

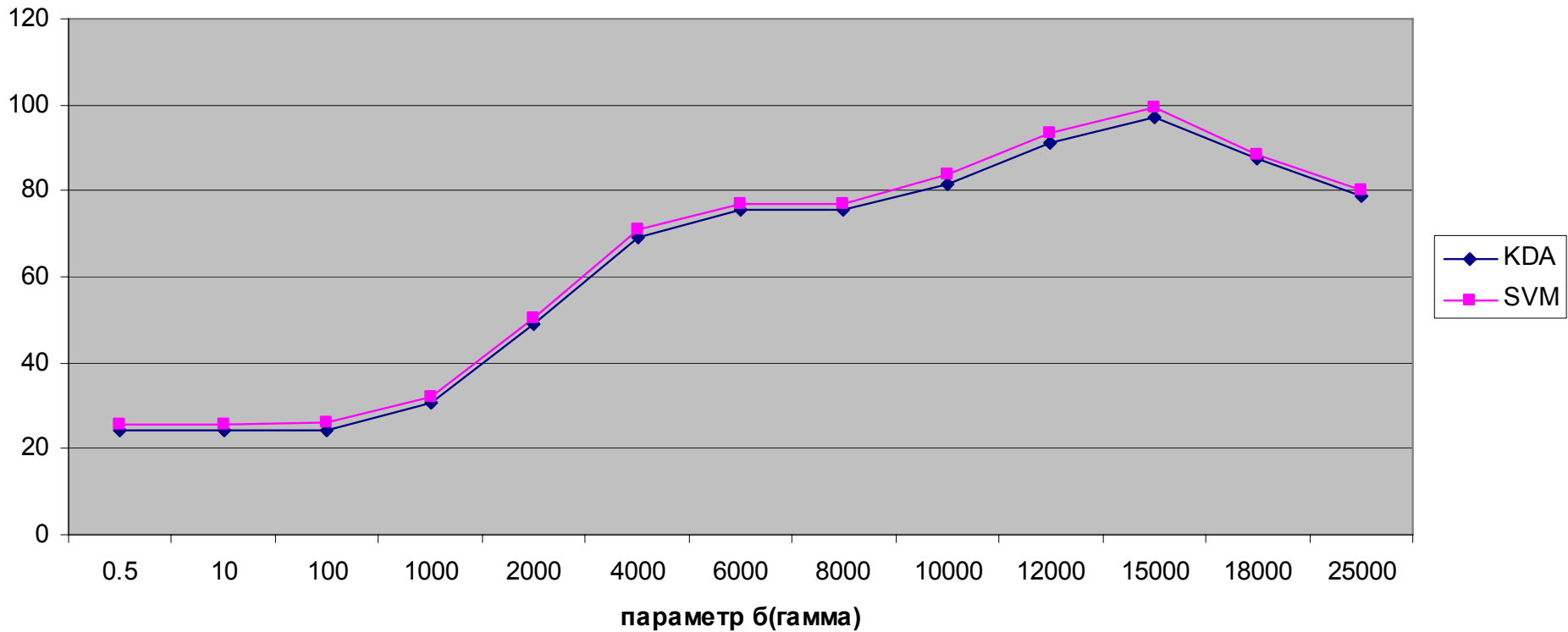
# Выбор параметра ядра в задаче классификации рукописных букв

В данной задаче каждый объект обучающего множества букв имел 25600 признаков, т.е. объект можно представить в пространстве, имеющем 25600 размерностей ( картинка 160x160 пикселей). Так же количество объектов множества было большое (более 100).

Были проведено тестирование, в результате которого были получены значения параметров ядра, при которых была получена наибольшая точность классификации.

# Выбор параметра ядра

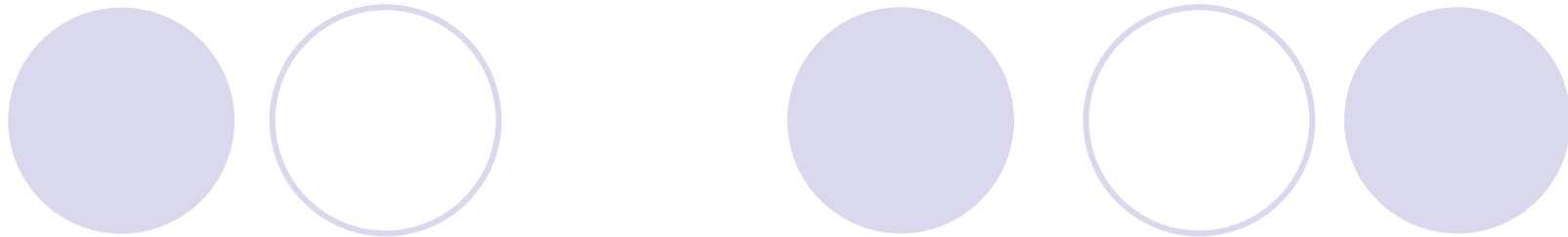
Зависимость точности классификации от параметра ядра (Гауссовское)







- Проведен анализ алгоритмов классификации для применения в задачах построения интеллектуального интерфейса
- Разработаны экспериментальные версии программ, реализующие алгоритмы SVA и KDA.
- Проведено тестирование реализованных алгоритмов при различных используемых параметрах.



**Спасибо за внимание!**