Анализ Алгоритмов

Рубежный контроль №1

«Метод классификации Support Vector Machine»

Юмаев Артур Русланович

ИУ7-55

Преподаватель: Волкова Л.Л.

Оглавление

[Введение 3](#_Toc26271884)

[Постановка задачи 4](#_Toc26271885)

[1. Аналитическая часть 5](#_Toc26271886)

[Квадратичная оптимизация и поиск оптимальной гиперплоскости 5](#_Toc26271887)

[Вывод 6](#_Toc26271888)

[2. Конструкторская часть 7](#_Toc26271889)

[Схемы алгоритмов 8](#_Toc26271890)

[Вывод 11](#_Toc26271891)

[3. Технологическая часть 12](#_Toc26271892)

[4. Исследовательская часть 14](#_Toc26271893)

[Заключение 16](#_Toc26271894)

[Литература 17](#_Toc26271895)

# Введение

В настоящее время сфера искусственного интеллекта проникла в нашу повседневную жизнь и все больше компаний и исследовательских центров применяют ИИ для решения разного рода задач. Классические методы ИИ были придуманы еще Тьюрингом в 40е годы. Позже появилась первая попытка воссоздать искусственный нейрон человека. Модель искусственного нейрона была предложена Розенблатом в 50х года. Тогда Розенблат еще не знал, насколько сильно это изменит наш мир. Также в 19 веке были изобретены первые методы классификации, такие как линейный дискриминантный анализ Фишера. В 1992 году Вапником [1] была предложена модель классификации данных, которая была названа методом опорных векторов (SVM – Support Vector Machine). Она могла обучаться на некотором наборе данных и классифицировать его на 2 класса. Далее этот метод был доработан для произвольного количества классов. Сейчас машина опорных векторов активно используется для классификации сложных многомерных данных.

# Постановка задачи

Цель: изучение метода классификации SVM.

Задачи:

* реализовать метод;
* протестировать метод;
* привести листинг на одном из языков программирования.

# Аналитическая часть

Машина опорных векторов – это линейная система, идея которой состоит в построении гиперплоскости, выступающей в качестве поверхности решений, максимально разделяющей положительные и отрицательные примеры. В частности, алгоритм настройки опорных векторов можно использовать для построения следующих трех типов обучаемых машин.

* Полиномиальные обучаемые машины.
* Сети на основе радиальных базисных функций.
* Двухслойные персептроны.

Рассмотрим множество обучающих примеров

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

где – входной образ для примера i; – соответствующий ему желаемый отклик (целевой выход). Предположим, что классы +1 и -1, составляющие множество d линейно разделимы. Уравнение поверхности решений в форме гиперплоскости, выполяющей это разделение, записывается следующим образом:

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

где **x –** входной вектор; **w** – настраиваемый вектор весов; *b* – порог. Таким образом можно записать:

|  |  |  |
| --- | --- | --- |
|  | , | (3) |

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

Оптимальный вектор весовых коэффициентов обозначим как , а оптимальный порог обозначим как

## Квадратичная оптимизация и поиск оптимальной гиперплоскости

Задачу условной оптимизации можно решить с помощью *метода множителей Лагранжа* (Method of Langrange multipliers) [2].

Построим сначала функцию Лагранжа:

|  |  |  |
| --- | --- | --- |
|  |  | (5) |

где дополнительные неотрицательные элементы – множители Лагранжа.

Опуская некоторые преобразования запишем итоговую формулу для получения вектора весовых коэффициентов и порога.

|  |  |  |
| --- | --- | --- |
|  |  | (6) |

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

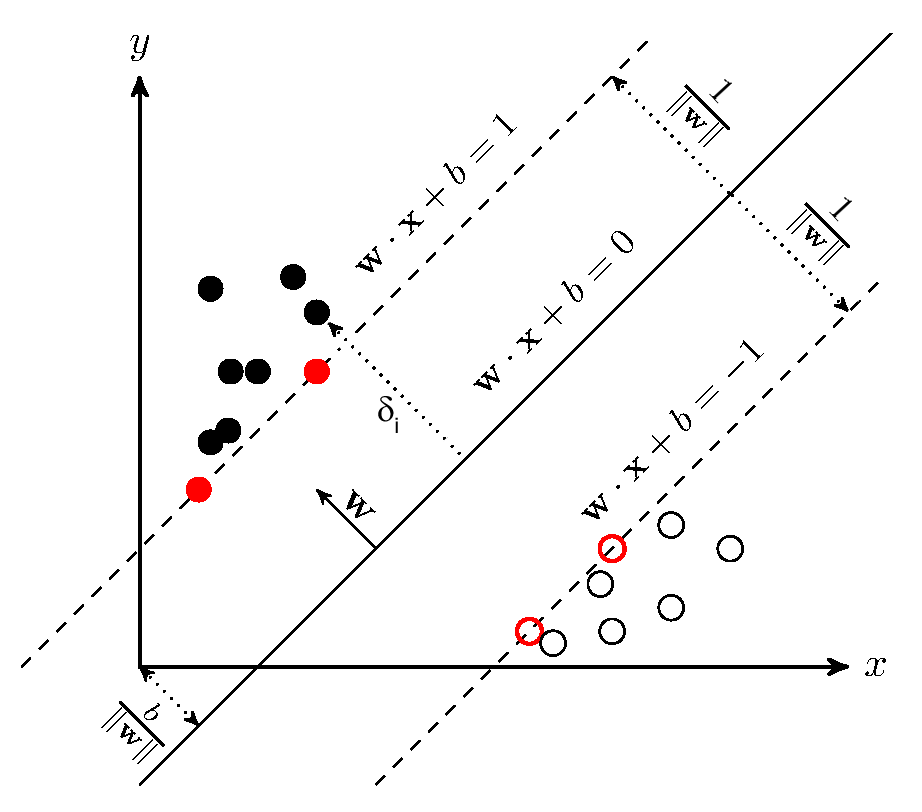


Рисунок 1. Визуализация оптимальной разделяющей гиперплоскости на плоскости

## Вывод

Был проведен краткий математический обзор машины опорных векторов. Приведена целевая функция для оптимизации и поиска оптимальных значений весовых коэффициентов и порога.

# 2. Конструкторская часть

## Схемы алгоритмов

В данном разделе будут приведены схема алгоритма Винограда и схема функция распараллеливания. На рисунке 1 изображена схема функции распараллеливания.

## Вывод

Была приведена схема алгоритма Винограда, а также схема для функции для распараллеливания этого алгоритма. Также была приведена оценка трудоемкости алгоритмов Винограда.

# 3. Технологическая часть

В данном разделе будет приведен листинг метода опорных векторов на языке Python 3.7 (листинги 1), так как имеется большой опыт работы с ним.

Листинг 1. Метод опорных векторов

|  |
| --- |
| class SVM:  def \_\_init\_\_(self):  pass  def fit(self, data):  self.data = data  opt\_dict = {}  transforms = [[ 1, 1],  [-1, 1],  [-1,-1],  [ 1,-1]]  all\_data = []  for class\_label in self.data:  for featureset in self.data[class\_label]:  for feature in featureset:  all\_data.append(feature)  self.max\_feature\_value = max(all\_data)  self.min\_feature\_value = min(all\_data)  # no need to keep this memory.  all\_data=None  step\_sizes = [self.max\_feature\_value \* 0.1,  self.max\_feature\_value \* 0.01,  self.max\_feature\_value \* 0.001]  b\_range\_multiple = 5  b\_multiple = 5  latest\_optimum = self.max\_feature\_value\*10  # stepping down the vector  for step in step\_sizes:  w = np.array([latest\_optimum, latest\_optimum])  optimized = False  while not optimized:  for b in np.arange(-1 \* \  (self.max\_feature\_value \* \  b\_range\_multiple),  self.max\_feature\_value \* \  b\_range\_multiple,  step \* b\_multiple):  for transformation in transforms:  w\_t = w \* transformation  found\_option = True  for class\_label in self.data:  for xi in self.data[class\_label]:  yi = class\_label  if not yi \* \  (np.dot(w\_t, xi) + b) \  >= 1:  found\_option = False    if found\_option:  opt\_dict[np.linalg.norm(w\_t)] = [w\_t, b]  if w[0] < 0:  optimized = True  else:  w = w - step  norms = sorted([n for n in opt\_dict])  opt\_choice = opt\_dict[norms[0]]  self.w = opt\_choice[0]  self.b = opt\_choice[1]  latest\_optimum = opt\_choice[0][0] + step \* 2  def predict(self, features):  self.features = features  prediction = np.sign(np.dot(np.array(features),  self.w) + self.b)  return prediction |

# 4. Исследовательская часть

Заключение

Была рассмотрена и реализована многопоточная версия алгоритма Винограда, которая показала большую эффективность по сравнению с классическим алгоритмом Винограда. Была оценена трудоёмкость алгоритма Винограда, сделаны схемы алгоритмов. Дана асимптотическая оценка. Задача выполнена.

# Литература

[1] Boser B., I. Guyon and V.N. Vapnik. “A training algorithm for optimal margin classifiers”, Fifth Annual Workshop on Computational Learning Theory, 1992, p. 144-152. San Mateo, CA: Morgan Kauffman.

[2] Bertsekas D.P. Dynamic Programming and Optimal Control, 1995, vol. I and vol. II, Belmont, MA: Athenas Scientific.