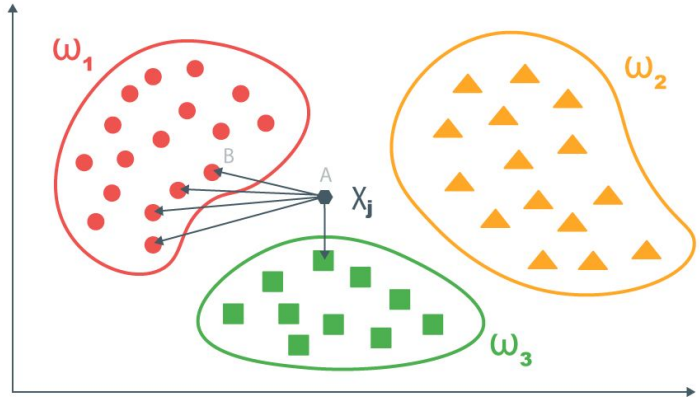
# TMVA и классификация данных

(На примере написанной программы)

# Задача классификации

Задача классификации в общем виде: для неопределенных данных построить алгоритм, который будет соотносить эти данные с данными, имеющимися в известной выборке. Например, алгоритм, определяющий на основе некоторого количества четных чисел, относится ли произвольное число к четным числам.



# TMVA

TMVA - The toolkit for multivariate analysis. Root - интегрированная среда для классификации и анализа данных. Из TMVA - скриптов были использованы:

- TMVAClassification.C
- TMVAClassificationApplication.C

# Сигнал/Бэкграунд

- Сигнал - события, которые нужно выделить среди остальных событий и которые имеют ценность для поставленной задачи.
- Бэкргграунд - фоновые события, которые не имеют ценности для их классификации и служат для выделениях из них “сигнальных” событий.

# Постановка задачи:

Проанализировать два файла, в одном из которых общие данные, а в другом данные с явно выраженной зависимостью между ними. Натренировать на основе этих файлов алгоритм, показывающий, есть ли подобная зависимость для любых других данных.

# Программа:

Написанная программа состоит из 3-х файлов:

- Neuro.hpp - файл-заголовок.
- Neuro.cpp - файл-описание.
- Neuro-pilots.cpp - Главная программа.

Программа состоит из структуры, в которую занесены переменные - характеристики “пилотов”, такие как рост, вес итд, функций, создающих рандомную генерацию этих характеристик и функции, записывающей сгенерированные характеристики в структуру “дерево”.

# Работа TMVAClassification.C

Данный скрипт считывает из “дерева” переменные - данные, записанные в дерево, работает с этими переменными различными классификаторами, и предоставляет информацию, такую, как: Матрицы корреляций переменных, зависимости между выходными переменными, график эффективности работы каждого классификатора итд. Так же, формирует директорию, в которую записываются “веса” для каждой переменной - значения определенной функции.

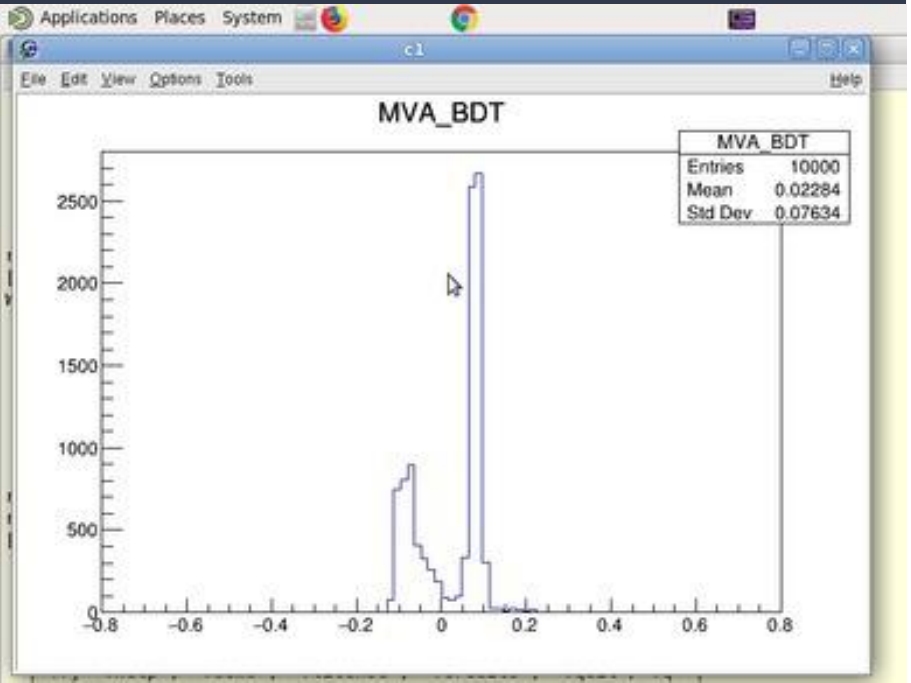
# Алгоритм работы:

- Указывается директория и имя файла .root. Файл открывается и объявляется“целью”.
- Создаются указатели на объект “дерево” с таким же именем дерева, какое находится в открытом файле.
- Создается выходной файл, в котором хранятся примеры, гистограмы итд.
- Создается объект “factory”, который принимает используемые классификаторы и обеспечивает взаимодействие с модулями TMVA.
- Методом AddVariable добавляются переменные, которые будут участвовать в классификации.
- Добавляется общее число событий, которое разбивается на количество “сигнальных” и “бэкграундных” событий.

# Работа TMVAClassificationApplication.C

Скрипт получает набор данных в виде дерева из файла .root и применяет к каждому событию веса, посчитанные скриптом TMVAClassification.C, после чего записывает гистограмму для каждого примененного метода, в которой указывается распределение весов для каждого события для этих данных. Этот скрипт является проверочным, а не тренировочным.

# Алгоритм работы:




- Создается TMVAReader - добавляются переменные, имена которых должны совпадать с переменными, находящимися в “дереве”.
- Создается адрес на объект “дерево”.
- Указывается адрес “ветки” или “веток” в дереве, таким образом, адреса переменных в указанном дереве привязываются к адресам переменных, для которых были составлены веса.
- В цикле каждое событие считывается из “дерева” и обрабатывается для выбранных классификаторов. На выходе получаются гистограммы со значениями функций для событий от каждого классификатора.

# Результат работы:

Сравнивая гистограммы, полученные скриптом TMVAClassificationApplication.C и гистограммы для тренировочного набора данных, показывающие разделения сигнальных и бэкграундных событий, можно сказать, какой диапазон значений классификатора соответствует сигнальным событиям в тестовом наборе данных.