ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчёт по лабораторной работе № 4

«Задача отбора признаков для модели МО»

Выполнил работу

Старцев Иван Александрович

Академическая группа №J3110

Принято

Практик, Владислав Вершинин

Санкт-Петербург

2024

1. Введение

Цель работы: разработать алгоритм для отбора наиболее значимых “фич” для модели МО без сильной потери качества.

Задачи:

* Установить и разобраться с библиотекой “mlpack”
* Написать алгоритм перебора всех возможных фич для модели МО.
* Реализовать подсчет метрики для каждого набора фич.
* Проверить каждый результат метрики на соответствие условию задачи

1. Теоретическая подготовка

Самым главным теоретическим аспектом было понимание работы пакета “mlpack” и его установка. Для установки требовалось прочитать много документаций для установки таких пакетов и работы с ними. Алгоритмы не требовали серьезной подготовки так как требовалось написать тяжелый и неоптимизированный код. Для написания такого было необходимо хорошее знание циклов и рекурсии. Основными используемыми типами данных были матрицы - arma::mat для хранения датасеты и std::vector для хранения результатов работы программы, а также библиотека “ctime” для подсчета времени работы.

1. Реализация

На первом этапе устанавливался “mlpack” для возможности запускать дальнейший программный код. Вторым этапом была разработка неоптимизированного алгоритма перебора всех возможных фич для удаления с итоговой сложностью nn. Далее был реализован подсчет метрики в ходе каждой итерации нового набора фич с проверкой условия задачи.

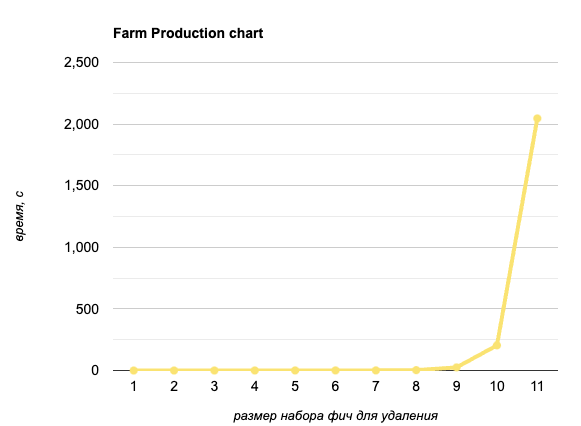
1. Экспериментальная часть

Согласно требованиям моего варианта, на вход к моему алгоритму подается датасет из 11 признаков для перебора. Теоретически заданная сложность задачи составляет O(nn \* n2 \* m). Для тестирования алгоритма была собрана статистика приведенная в таблице №1.

Таблица №1 – Подсчет сложности реализованного алгоритма

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Размер входного набора “фич” | 8 | 9 | 10 | 11 |
| Время выполнения программы, с | 3 | 25 | 204 | 2046 |

График представляющий визуально удобный формат данных из таблицы №1 представлен на изображении №1



Изображение №1 - график работы алгоритма

Скорость выполнения программы сильно зависит от кол-ва элементов. При увеличении входного набора фич время стремительно начинает увеличиваться, что является не очень хорошим показателем.

1. Заключение

В ходе выполнения работы мною был реализован алгоритм отбора минимального количества признаков для модели МО без сильной потери качества предсказания. Цель работы была достигнута путем тестирования модели на разных признаках из датасета.

В качестве дальнейших исследований можно предложить оптимизацию алгоритма перебора фич с точки зрения уменьшения итераций перебора разных наборов признаков.

1. Приложения

ПРИЛОЖЕНИЕ А

Листинг кода файла quick\_sort.cpp

#include <iostream>  
#include <fstream>  
#include <string>  
#include <sstream>  
#include <vector>  
#include <ctime>  
#define MLPACK\_PRINT\_INFO  
#define MLPACK\_PRINT\_WARN  
#include <mlpack.hpp>  
#include <mlpack/methods/linear\_regression/linear\_regression.hpp>  
#include "extra\_materials/modelling.cpp"  
  
  
void generate\_combi\_and\_train(arma::mat& dataset**,**float& result\_score**,**float& best\_score**,** std::vector<int>& fichi**,**const std::vector<int>& nums**,** std::vector<int>& combination) {  
 if (!combination.empty() && nums.size() != combination.size()) {  
 arma::mat drop\_dataset = drop\_columns(dataset**,** combination)**;** float score = evaluate\_dataset(drop\_columns(dataset**,** combination)**,** drop\_dataset.n\_rows - **2,** drop\_dataset.n\_rows - **1**)**;** if (-**0.1** \* best\_score <= score - best\_score && score - best\_score <= best\_score \* **0.1** && fichi.size() <= combination.size()) {  
 result\_score = score**;** fichi = combination**;** }  
 }  
 for (int i = **0;** i < nums.size()**;** ++i) {  
 if (std::find(combination.begin()**,** combination.end()**,** i) != combination.end()) {  
 continue**;** }  
 combination.push\_back(nums[i])**;** generate\_combi\_and\_train(dataset**,**result\_score**,** best\_score**,** fichi**,** nums**,** combination)**;** combination.pop\_back()**;** }  
}  
  
  
int feature\_selection() {  
 const char\* path = "../data/WineQT.csv"**;** arma::mat dataset**;** if (!mlpack::data::Load(path**,** dataset)) {  
 throw std::runtime\_error("Could not read \*.csv!")**;** }  
 std::vector<int> nums**;** for (int i = **0;** i < dataset.n\_rows - **7;** i++) {  
 nums.push\_back(i)**;** }  
 std::vector<int> combination**;** float best\_score = evaluate\_dataset(dataset**, 11, 12**)**;** float result\_score = **10;** std::vector<int> fichi**;** generate\_combi\_and\_train(dataset**,** result\_score**,** best\_score**,**fichi**,** nums**,** combination)**;** std::cout << "best score: " << best\_score << std::endl**;** std::cout << "result score: " << result\_score << std::endl**;** std::vector<int> features**;** std::cout << "features: " << std::endl**;** for (int i = **0;** i < dataset.n\_rows - **2;** i++) {  
 if (std::find(fichi.begin()**,** fichi.end()**,** i) == fichi.end() ) {  
 features.push\_back(i)**;** }  
 }  
 for (auto val : features) {  
 std::cout << " " << val**;** }  
 std::cout << std::endl**;** return **0;**}  
  
  
void run\_tests() {  
 feature\_selection()**;**}  
  
void run\_tests\_for\_modelling() {  
 const char\* path = "../data/WineQT.csv"**;** arma::mat dataset**;** if (!mlpack::data::Load(path**,** dataset)) {  
 throw std::runtime\_error("Could not read \*.csv!")**;** }  
 std::vector<int> nums = {**0, 2, 3, 4, 5, 6, 7, 8, 9**}**;** dataset = drop\_columns(dataset**,** nums)**;** float score = evaluate\_dataset(dataset**,** dataset.n\_rows - **2,** dataset.n\_rows - **1**)**;** std::cout << score << std::endl**;**}  
  
int main() {  
 unsigned int start = clock()**;** run\_tests()**;** std::cout << "ВРЕМЯ РАБОТЫ: "<<(clock() - start) / CLOCKS\_PER\_SEC << std::endl**;** return **0;**}