Съдържание

Въведение

Кратко описание на SVM модела

SVM (Support Vector Machines) е алгоритъм за машинно самообучение, използван за задачи на класификация и регресия. Той се основава на идеята за намиране на хиперплоскост, която максимално разделя две класа от данни.

Целта на SVM е да намери оптималната хиперплоскост, която разделя данните по най-добрия начин. Това се извършва чрез избиране на хиперплоскост, която има най-голямо разстояние (марджин) до най-близките данни от двата класа. Тези най-близки данни, наречени опорни вектори, са от съществено значение за построяването на хиперплоскостта.

SVM може да работи както с линейни, така и с нелинейни данни, чрез използване на различни ядра (kernel функции). Ядрото преобразува пространството на данните, което позволява SVM да работи в по-високоизмерно пространство, където е по-вероятно да намери разделяща хиперплоскост.

Едно от предимствата на SVM е, че може да се справя с малък брой примери за обучение и да извършва добре в сложни задачи на класификация. Освен това, SVM има математически обоснована теория, която гарантира неговата ефективност при правилно подбрани хиперпараметри.

В заключение, SVM е мощен алгоритъм за класификация и регресия, който може да се използва за разделяне на данни в линейни и нелинейни пространства.

Кратко описание на Deep Learning

Deep Learning е подраздел на машинното самообучение, който използва изкуствени невронни мрежи с дълбоки архитектури за обработка на данни и извършване на сложни задачи. Deep Learning моделите се характеризират със множество слоеве (неврони), които обработват данните последователно и изграждат постепенно все по-абстрактни представяния.

Основна съставна част на Deep Learning моделите са невронните мрежи, които са структури от взаимно свързани възли (неврони). Всяко ниво от възлите обработва информацията и я предава на следващото ниво, като така моделът може да изгражда постепенно все по-сложни абстракции. Това позволява на моделите да откриват връзки и закономерности в данните, които не са лесно достъпни от човешко зрение или традиционни методи на машинното самообучение.

Deep Learning моделите изискват голямо количество данни за обучение и често използват градиентно спускане за настройка на параметрите. За обучение и оптимизация на тези модели се използват различни алгоритми, като най-известният е задълбоченият обратно разпространяващ се градиент (backpropagation), който изчислява градиентите на параметрите спрямо целевата функция.

Deep Learning моделите са доказали своята сила в различни области на компютърното зрение, природен езиков процесинг, рекомендателни системи, геномика и други. Те могат да се използват за класификация, регресия, обектно разпознаване, генериране на съдържание, машинен превод и много други задачи.

Deep Learning моделите се градят с помощта на различни фреймуърки, като TensorFlow, Keras, PyTorch и други, които предоставят инструменти и библиотеки за създаване, обучение и използване на тези модели.

Задание

Да се обучат SVM и DeepLearning модели на зададеият набот от данни. Наборът от данни съдържа множество фактори, които се използват за класифициране на всеки запис в един от следните класове: *Без диабет, Пред диабет, Дабет тип 2*

Експериментална постановка

Среда за разработка

За целта на този проект е потготвена среда за разработка и изпълнение на базата на python, tesorflow и sklearn, публикувана в Github: https://github.com/Rafael-Kalachev/runm-coursework.git и имидж за докер контейнер публикуван в Docker Hub: https://hub.docker.com/repository/docker/rkalachev/runm_coursework/general

За използване на средата се използва платформа за контейнеризация (Docker) която извиква **docker_process.sh** Тъй като цялата работна директория се монтира в докер контейнера това ни позволява да бъдат изпълнявани множество заявки без работната среда да бъде прегенерирана.

Предварителна обработка на данните

Зададеният набор от данни съдържа 13850 записа и 52 входни фактора.

Отстраняване на празни полета

Тук се обръща внимание, че наборът от данни има голям борй липсващи стойности, като респективно по фактори (INC_THIRST: 9705, FREQ_URIN: 11248). След изследване се открива, че тези стойности са категорийни променливи “ДА/НЕ” които зависят от техните числени оцени респективно (INC_THIRSTN и FREQ_URINN). Обръща се внимание, че липсват стойностите за “НЕ”, които използвайки скрипта попълваме в набора от данни.

Обработка на факторите съдържащи текст

Голям брой от факторите съдържат текст, най-вероятно защото са категорийни променливи. За да придобиване на по-добро разбиране относно тези фактори представяме списък с всички текстови фактори и наборъ от уникални данни в тях.

DIABETES, [‘no’ ‘diabetes type 2’ ‘pre-diabetes’]

GENDER, [‘female’ ‘male’]

ETHNIC, [‘hispanic or latino’ ‘white’ ‘black or african american’ ‘american indian or alaska native’ ‘native hawaiian or other pacific islander’ ‘asian’]

INC_THIRST, [‘yes’ ‘no’]

UNIT_THIRST, [‘l/day’] ((Drop))

FREQ_URIN, [‘no’ ‘yes’]

UNIT_URIN, [‘times/day’] ((Drop))

INC_HUNGER, [‘no’ ‘yes’]

WGHT_LOSS, [‘no’ ‘yes’]

FATIGUE, [‘yes’ ‘no’]

BLUR_VISION, [‘no’ ‘yes’]

SLOW_HEALING, [‘no’ ‘yes’]

FREQ_INFECTIONS, [‘no’ ‘yes’]

UNIT_A1C, [‘%’] ((Drop))

UNIT_FASTING_SUGAR, [‘mg/dl’] ((Drop))

UNIT_GLUCOSE_TOLERANCE, [‘mg/dl’] ((Drop))

UNIT_RANDOM_SUGAR, [‘mg/dl’] ((Drop))

UNIT_DIASTOLIC, [‘mmhg’] ((Drop))

UNIT_SYSTOLIC, [‘mmhg’] ((Drop))

UNIT_LDL_CHOL, [‘mg/dl’] ((Drop))

UNIT_HDL_CHOL, [‘mg/dl’] ((Drop))

UNIT_TRIGLYCERIDES, [‘mg/dl’] ((Drop))

UNIT_ALBUMIN, [‘g/dl’] ((Drop))

UNIT_NERVE_VELOCITY, [‘m/sec’] ((Drop))

UNIT_BMI, [‘kg/m^2’] ((Drop))

UNIT_GUMLINE, [‘mm’] ((Drop))

UNIT_GLUCOSE_SCREENING, [‘mg/dl’] ((Drop))

Факторите маркирани с ((Drop)) биват отстранени, тъй като съдържат постоянна стойност за всички изследвани данни, което не би внесло никаква динамика в изхода. Отстранените фактори са основно мерните единици използвани, които са еднакви за всички записи. Тук е важно да се отбележи, че при използването на модела се налага всички данни да бъдат конвертирани в горепосочените мерни единици.

Факторът SUBJID се отстранява, тъй като това е номерът на записа, който е нерелевантен за експеримента

Факторите GENDER и ETHNIC имат предварително подготвена цифрова репрезентация респективно GENDERN и ETHNICN, следователно могат да бъдат отстранени.

Оставащите фактори са факторите с “ДА”/“НЕ”. тези фактори се конвертират в 1 и 0 респективно за сотйнотите “ДА” и “НЕ”

Обработка за нуждите на SVM

Support Vector Machines (SVM) са проектирани да се справят с проблеми с класификацията, когато етикетите на класа са категорийни. Класовете обикновено се представят с дискретни цели числа, като всеки клас има уникален целочислен етикет. В изходната променлива DIABETESN Това е вече направено, следователно променливата DIABETES може да бъде отстранена.

Обработка за нуждите на DeepLearning

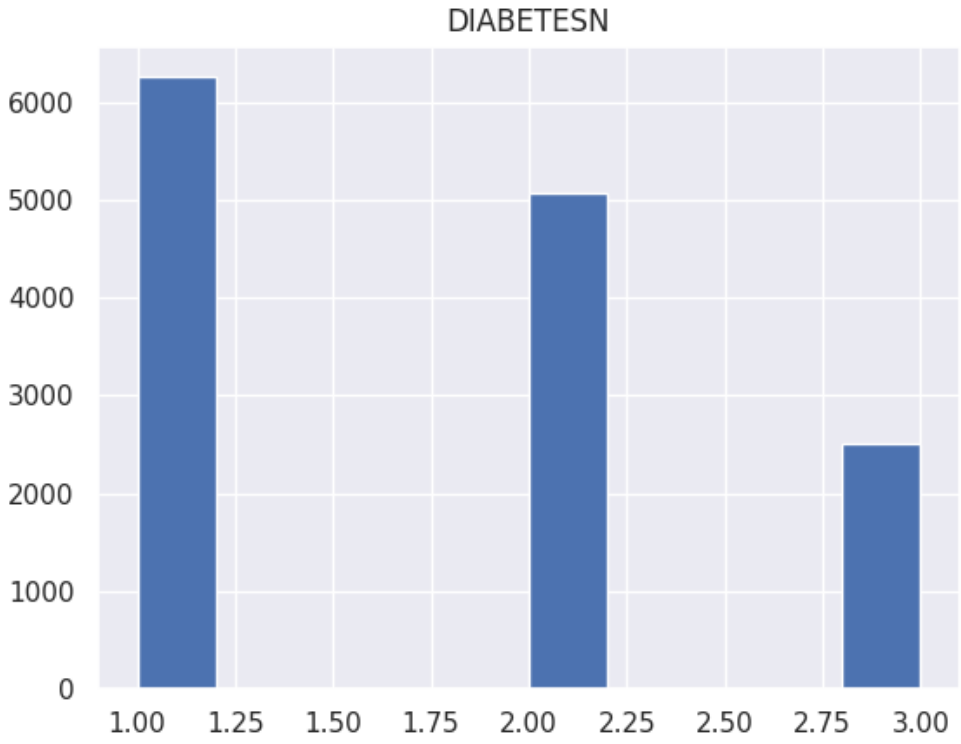
При дълбокото обучение е обичайно категоричните променливи да се представят като еднократно кодирани вектори, където всяка категория е представена от двоичен вектор с 1 в позицията, съответстваща на категорията и 0 на друго място. Това е често срещан начин за представяне на категорични променливи в невронни мрежи, включително модели за дълбоко обучение.

Поради това обикновено се препоръчва да се представят категориални променливи, като се използва еднократно кодиране или подобни техники, вместо просто да се използват самите числови стойности като входни данни за модел на дълбоко обучение. Това ще гарантира, че моделът третира променливите като категорични и ще може да научи подходящи представяния за различните категории.

За целта изходният категориален фактор DIABETES се представя с 3 бинарни фактора респективно: “Диабет тип 2”, “Без Диабет”, “Преддиабет”

Категориалният фактор ETHNIC се заменя с 6 бинарни категориални променливи респективно: “Испански/Латино”, “Бял”, “Черен или афроамерикански”, “Америко индиански или с произход от Аляска”, “С Хавайски произход или друг тихоокеански произход” или “Азиатски”

Числените бинарни променливи зададени с 1 и 2 са променени на 0 и 1.



Фигура 1: Хистограма на Изходен фактор

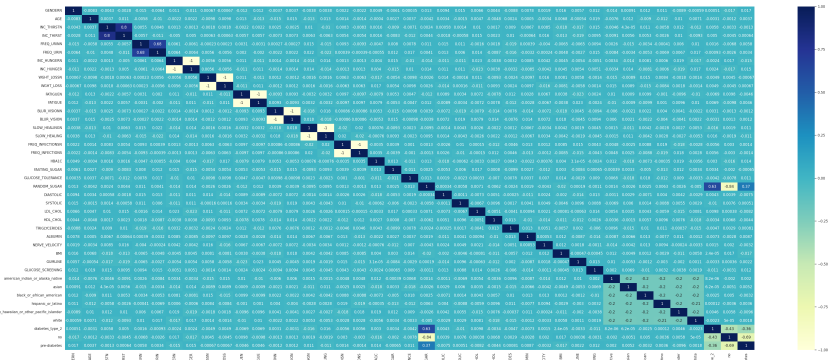
Оглед на факторите

Изходен фактор

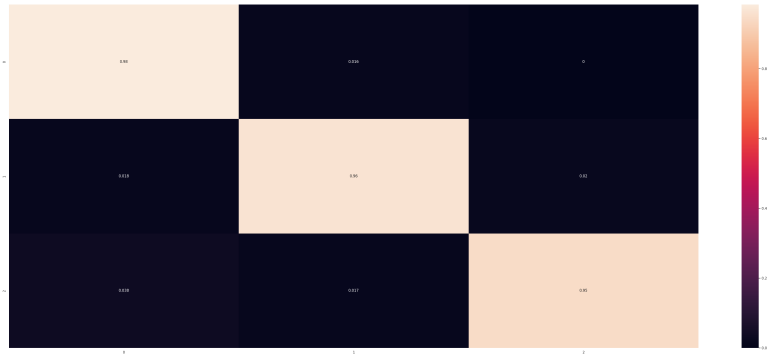
Обръщаме внимание, че изходният фактор показан на фигура 1 има известен класов дисбаланс.

При обучение на модели, класовият дисбаланс може да се превърне в предизвикателство, особено когато по-редките класове имат много по-малък брой примери в сравнение с по-често срещаните класове. Това може да доведе до нежелани ефекти като ниска прецизност, ниска откриваемост на редките класове или ненадеждни резултати.

Приема се че класовият дисблаланс не е значим за целите на експеримента и не се коменсира, тъй акто имаме сравнително високо количество представителни извадки от всички класове.



Фигура 2: Корелационна матрица на факторите



Фигура 3: Матрица на объркване на SVM модел

Корелации между факторите

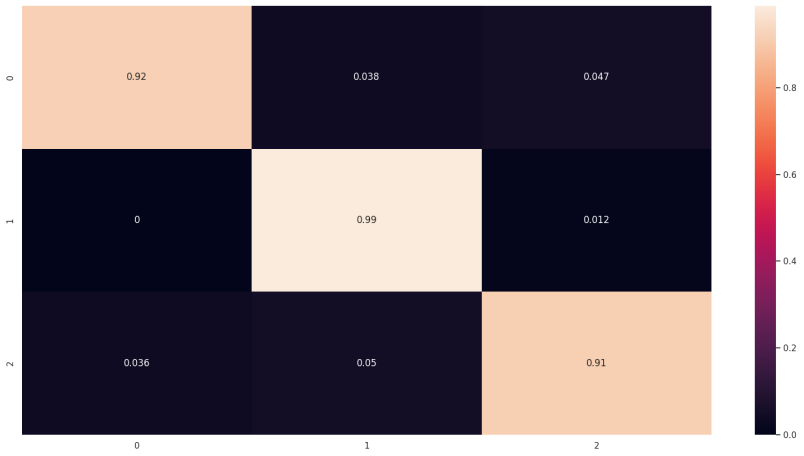
Обучение

Обучение на SVM

Използвайки скрипта **training.py** е обучен SVM модел.

За настройка на модела е използвано ‘linear’ (линейно) ядро като функция на преобразуване.

Получен е модел, който предоставя **96.89%** точност. Като на фигура 3 е показана матрицата на объркване на модела.



Фигура 4: Матрица на объркване на DeepLearning модел

Обучение на DeepLearnig модел

Използвайки скрипта **training_dl.py** е обучен SVM модел.

Sequential модел, който ни позволява да стековаме множество слоеве линейно. Първият слой в невронната мрежа. Той има 64 неврона и използва активационната функция ReLU, също така задава input_shape в зависимост от броя на факторите във входните данни.

Модела има три слоя след първия, които са скрити слоеве, всеки със 64 неврона и активационната функция ReLU. Тези слоеве помагат на модела да научи сложни модели и представления. Последният слой има 3 изходни неврона, които съответстват на трите целеви променливи.

Получен е модел, който предоставя **94.8%** точност. Като на фигура 4 е показана матрицата на объркване на модела.

Извод и заключение

За представеният случай забелязваме, че SVM дава малко по-добри резултати от DeepLearning модела. Също по време на обучение се наблщдава по бързо обучение при използването на невронна мрежа от колкото при използването на SVM.

Дълбокото обучение (Deep Learning) е мощен метод за обработка на данни, особено за сложни иерархични структури на данни, като изображения, звукови сигнали или текстове. Deep Learning моделите, основани на невронни мрежи, са в състояние да научат сложни представления и да извличат високоабстрактни характеристики от данните. Те могат

да се справят с големи и сложни набори от данни, но се нуждаят от значителни ресурси за обучение и обработка.

SVM е ефективен алгоритъм за класификация, който работи добре в случаи, когато има ясно разделяща граница между класовете и когато броят на признаците е малък или среден. SVM може да се справи добре и с по-малки набори от данни, като предоставя ясно интерпретируем модел.

Библиография

[1] В. Младенов and С. Йорданова, Размито управление и невронни мрежи. Ту-София, 2016.

[2] C. C. Aggarwal, Neural Networks and Deep Learning A Textbook. Cham Springer International Publishing, 2018.

[3] sklearn, “sklearn.svm.SVC” https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

[4] Wikipedia, “Support vector machine.” https://en.wikipedia.org/wiki/Support_vector_machine

[5] Wikipedia, “Deep Learning.” https://en.wikipedia.org/wiki/Deep_learning

АПЕНДИКС

Всички проектни файлове могат да бъдат намерени на Github: https://github.com/Rafael-Kalachev/runm-coursework.git Обучените модели както и допълнителна информация относно процеса са налични в прикаченият out.zip файл.