Wykrywanie występowanie chorób serca,porównanie algorytów uczenia maszynowego nadzorowanego na podstawie zbioru danych dotyczących chorób układu krążenia z repozytorium UCI

Magdalena Szulc

Toruń,2022-05-01

Contents

Abstrakt

The aim of the work is to compare selected algorithms of supervised machine learning and build a model based on medical data, which diagnoses the presence or absence of cardiovascular disorders.

Medical data is distinguished by the fact that it is difficult to access, in most cases it is restricted information not available for public use. Therefore, a key step is to choose the features taken into account when creating the model. The data obtained from the UCI repository has already undergone pre-processing. The dataset itself, due to its small size, allows checking effects of algorithms without getting rid of redundant and insignificant features.

The main motive is to answer the question of how data deficiency strongly influences the outcome and whether there is a difference between the use of selected supervised learning algorithms requires a comparison of the difficulty level of creating a model, accuracy, complexity and time to obtain an answer.

Wstęp

Uczenie maszynowe jako spopularyzowana dziedzna metod uczenia , zainteresowanie sobą w dużej mierze zawdzięcza rozwoju procesorów graficznych pozwalających na wykorzystanie algorytmów w optymalnym czasie. Ze względu na chodliwość tematu , powstało nowe oprogramowanie lub przystosowania ułatwiające wydajną pracę z obszernymi zasobami danych. Pojęcie sztucznej inteligencji pochodzi od próby odtowrzenia ludzkiego sposobu myślenia, jedną z bardziej znanych historycznie postacią z tym związaną jest psycholog Frank Rosenblatt z Cornell University. Badacz przyczynił się do powstania projektu zbudowania maszyny o nazwie "perceptron" mającej za zadnie rozpoznawać litery jest prawzorem nowoczesnych sztucznych sieci neuronowych. To dzięki jego badaniach nad perceptronem rozpropagowany został koncept algorytmu uczenia skończonej liczbie wywołań. Z czasem liczba przetwarzanych informacji stopniowo się zwiększała dzięki zastosowaniu procesów równoległych oraz pamięci. Wartą wpomnienia datą jest rok 2006 , w tym roku zaprezentowano opensource'owy odpowiednik MapReduce od Google o który dał sposobnść do przenoszenia między procesorami obróbki Big Data. Rok ten jest również znaczący ze względu na wydanie przez Nividia procesora graficzngeo monopolizującego rynek uczenia maszynowego. Zmiejszenie kosztów pamięci RAM zaoowocowała powstaniem kolejnych algorytmów uczenia,a istniejące podejścia są sukcesywnie ulepszane.[1] W całym ogromie powstałych już metod uczenia się, z których każda ma swoje zalety i niedogodności.[2]

Diagnnoza medycznej to rozległy temat i pod kątem uczenia maszynowego może być rozpatrywany na podstawie danych tekstowych czy obrazów. Zadanie postawienia diagnozy podlega to typowe zagadnienie klasyfikacji ,ale nie jedyny sposób wykorzystania. Przykładem systemu uczącego może być wycena akcji na giełdzie na podstawie danych z poprzedniego kwartału lub optymalizacja strony na podstawie historii odwiedzeń .[3]

Sztuczna inteligencja wśród szerokiego zakresu swoich zastosowań może zostać wykorzytana do analizy bardziej lub mniej złożonych danych medycznych, w celu przewidzenia wystąpienia choroby u konkretnej osoby, bez udziału procesu myślowego od stony specjalisty.

Do tego przeznaczenia istnieje możliwość zastosowania uczenia nadzorowanego (ang. *supervised learning*) tj. rodzaj uczenia maszynowego zakładający istnienie zbioru danych testowych zawierających odpowiedzi, na których podstawie wyszukiwane są zależności, cechy znaczące oraz budowany jest w ten sposób model służący przykładowo do przewidywania przyszłych wartości.

W przypadku danych dotyczących chorób zależności typujące występowanie choroby, bazują na podstawie konkretnych wyników badań zgormadzonych w repozytorium UCI.

W dzisiejszych czasach choroby sercowo-naczyniowe stanowią najczęstszą przyczynę zgonów, a liczba osób cierpiących na te dolegliwości stale rośnie. Głównymi przyczynami zachorowalności diagnozowanymi przez specjalistów są niski poziom świadomości i profilaktyki chorób serca.Dlatego prowadzone są intensywne prace nad zwiększeniem dostępności badań, które wspomogą diagnostykę kardiologiczną na jak najwcześniejszym etapie [4].

Powodem szukania dokładniejszych sposobów diagnozowania są również wysokie koszty leczenia generowane przez choroby układu krwionośnego. Według analityków firmy konsultingowej KPMG [5] w

2011 r. koszty diagnostyki i terapii chorób serca wyniosły ponad 15 miliardów polskich złotych.

Uczenie maszynowe poprzez przetwarzanie dużych zasobów klinicznych danych historycznych pod kątem zależności przyczynowo skutkowych, może zostać wykorzystane do wczesnej diagnostyki lub wspomagania leczenia pacjentów [6].

Słowa kluczowe: uczenie maszynowe, uczenie nadzorowane, lasy losowe, maszyna wektorów nośnych, k-najbliższych sąsiadów

Cel i zakres pracy

Celem pracy jest porównanie wybranych algorytmów uczenia maszynowego nadzorowanego, przy założeniu że dane wejściowe są wybrakowane, a w rezultacie zbudowanie modelu który na podstawie danych medycznych wystawia diagnozę o występowaniu zaburzeń sercowo-naczyniowych lub ich braku.

Dane medyczne wyróżniają się tym, że trudno uzyskać do nich dostęp, najczęściej nie są to informacje, które się udostępnia do użytku publicznego, z tego powodu, kluczowym krokiem jest wybór cech branych pod uwagę przy tworzeniu modelu. Dane pozyskane z repozytorium UCI przeszły już wstępną obróbkę , sam dataset ze względu na swoje niewielkie rozmiary pozwala na sprawdzenie działań algorytmów bez pozbywania się nadmiarowych i mało znaczących cech.

Zatem odpowiedź na pytanie jak wybrakowanie danych mocno wpływa na rezultat i czy istnieją róznicę między zastosowaniem wybranych algorytmów nauczania nadzorowanego wymaga przedstawienia porówniania łatwości tworzenia modelu, dokładności, złożoności oraz czasu uzyskania odpowiedzi.

W pracy opisano następujące algorytmu uczenia nadzorowanego:

- losowe lasy decyzyjne (ang. *random decision forests*)
- maszyna wektorów nośnych (ang. *support vector machines*, SVM)
- k-najbliższych sąsiadów (ang. *k-neares neighbours*, KNN)

Wykrywanie występowanie chorób serca,porównanie algorytów uczenia maszynowego nadzorowanego na podstawie zbioru danych dotyczących chorób układu krążenia z repozytorium UCI

Wprowadzenie teorertyczne

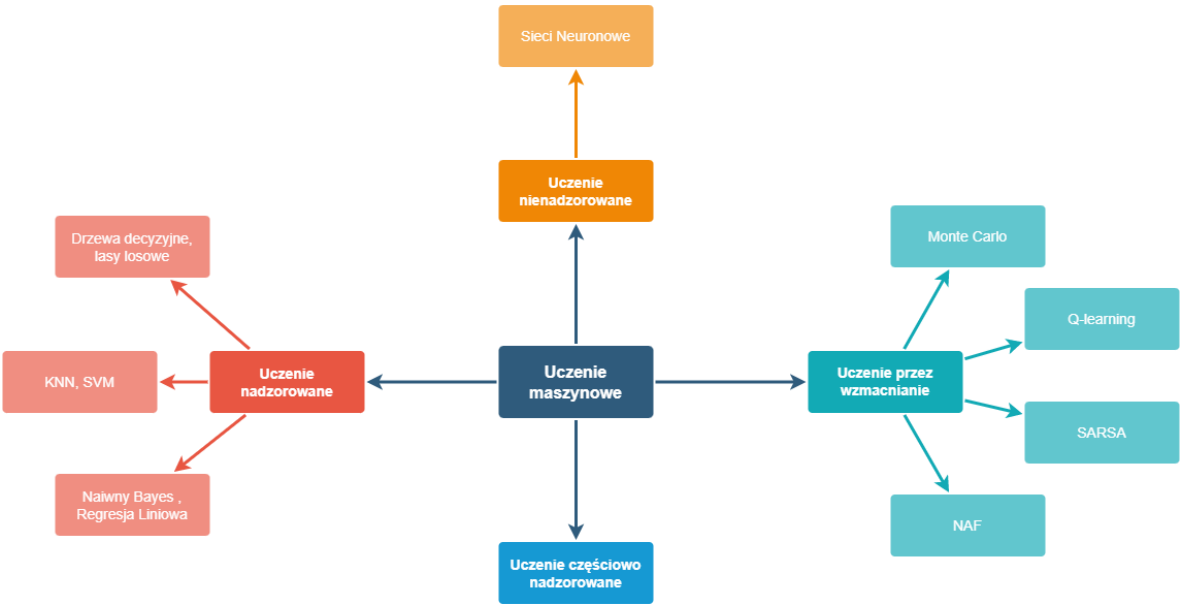
Omówienie teoretyczne rozpoczynają defninicje podstawowych pojęć wykorzystywanych w dalszych częściach, zaczynając od Algorytmu.

Algorytm to pojęcie matematyczne odpowiadające za szereg działań prowadzących do uzyskania żądanego rozwiązania, bazując na wynikach działań krokami opisujemy np problem znajdowania najktórszej drogi jak i zarówno tłumaczenie języka na podstawie analizy mowy.

Uczenie maszynowe (ang. *machine learning*, ML) to dziedzina zajmująca się tworzeniem modeli do analizy bardzo obszernych zasobów danch. Modele utworzone za pomocą algorytmów uczenia maszynowego są w stanie z wysokim prawdopodobieństwem wystawić predyckję lub dokonać klasyfikacji na temat zadanego problemu.

Model *klasyfikacjny* służy do przewidzenia etykiety klasy poprzez mapowanie na już z góry ustalony jednowymiarowy pozdział, model *regresywny* natomiast mapuje przestrzeń ustalając liczbę klas podziału oraz grupując wartości. [7] Istnieje możliwość przeksztaucenia problemu regresywnego na klasyfikacyje i na odwrót poprzez zamiane wartości oczekiwanego wyniku. Taką modyfikację zastosowano w praktycznej częsci projektu. Wyniki dla danych występowały w wartościach od 0 do 4 , dla wartości <1,4> przypadek testowy uznawany był za sklasyfikowany pozytywny (chory), dlatego przeksztaucenie z modelu regresywnego do modelu klasyfikacyjnego polega na konwersji wyników do wartości liczbowych 0 - brak stwierdzenia stanu choroboweo oraz 1 - stwierdzenie o chorobie układu krążenia.

Sposób wykorzystania segreguje alorytmy uczenia maszynowego na dwie kategorie, jednak powszechnie stosowanym podziałem jest podział zależnie od sposobu *trenowania* algorytmu. Algorytmy dzieli się na min.: uczenie nadzorowane, uczenie częściowo nadzorowane, uczenie bez nadzoru oraz uczenie przez wzmacnianie [8] .



Dobór typu uczenia oraz algorytmu uzależniony jest od danych wejściowych oraz oczekiwanego rezultatu. Dane wyjściowe mogą przyjmować format odpowiedzi TAK/NIE , klasyfikacji do danego zbioru czy np procentowej oceny ryzyka.

Uczenie maszynowe nadzorowane (ang. *supervised learning*) to klasa algorytmów uczenia maszynowego, która bazuje na poetykietowanych danych. Nadzór polega na porównaniu rezultatów działania modelu z wynikami które są zawarte w danych wejściowych (*dane oznaczone*) [9]. Algorytm po osiągnięciu żądanej efektywnosci jest w stanie dokonać klasyfikacji przykładu dla którego nie posiada odpowiedzi. Sprawdza się to obecnie w rekomendacji produktów oraz diagnozie chorób. Z matematycznego puntu widzenia dopasowanie danych oznaczonych nazywane jest aproksymacją funkcji [8] .

Uczenie maszynowe bez nadzoru (ang. *unsupervised learning*) to klasa algroytmów uczenia maszynowego która wiodąco rozwiązuje problemy grupowania. Dane dostarczane do modelu nie zawierają *oznaczeń*, zatem nauczanie polega na wyciąganiu konkluzji z poprzednio wykonanych iteracji. Na skuteczność modeli budownych w oparciu o uczenie bez nadzoru wpływ ma rozmiar dostarczonego do nauki zbioru danch, im jest on większy tym bardziej wzrasta effektywność. Takie zbiory można uzyskać rejestrując dane na bierząco dlatego do najczęstrzych zastosowań tej klasy algorytmów, można zaliczyć rozpoznawanie mowy czy obrazu [8] .

Uczenie maszynowe przez wzmacnianie (ang. *reinforcement learning*) to klasa algorytmów uczenia maszynowego której nauczanie nie opiera się na danych wejściowych czy wyjściowych a rezultatach otrzymanych podczas testu nazywanych tzw. sygnałami wzmocnienia który może przyjmować wartość pozytwną lub negatywną. Algorytm generując dane wejściowe dostosowuje reguły by uzyskać zwrotnie sygnał pozytywny w jak największej liczbie przypadków. [10] .

Uczenie częściowo nadzorowane (ang. *semi-supervised learning*) to klasa algorytmów uczenia maszynowego która wykorzystuje zbór danych w większości niepoetykietowany na podstawie których tworzony jest model [11] , w wykorzystywana głównie w przypadkach niewydajności zastosowania osobno modeli nadzorowanych i nienadzorowanych. Zastosowanie tej klasy algorytmów pozwala również na maksymalizację wykorzystania zebranych informacji [2] .

Uczenie nadzorowane przedstawiając oficjalną matematyczą definicje :

$$DL = ((xi, yi))_{i=1}^{l} \qquad gdzie :$$

(xi, yi) to punkt z zakresu xi X oznaczony etykietą yi dla danych wejściowych oznaczonych jako X.

Zbiór (xi,yi) to tgzw. dane uczące na podstawie których metody uczenia próbują wywnioskować funkcję która ustali y dla nieoznakowanego x. Większy zasób punktów sprawdzający działanie ,czyli dane testowe definiujemy następująco [12] :

DU = (x i) l+u g d z i e : i=l+1

Klasyfikacja a Regresja

Oba przedstawione poniżej typy sysytematyzją w oparciu o dostarczone dane wejściowe i mają one wspólną część polegającą na budowaniu modelu separującego kategorie docelowe w użyteczny i dokładny sposób.[2]

Klasyfikacja - decyduje o przynależności do zbioru,kategorii, grupy lub klasy. *Regresja* - daję ciągłą prognoze korelacji między zmiennymi, standardowym przykładem zastosowania jest prognoza pogody. Realne pomiary temperatury, prędkości wiatru , ciśnienia wpływają na finalną odpowiedź. Sama regresja dzieli się równiez na kategorie ze względu na skomplikowania, najprostrzym przykładem jest oczywiście regresja liniowa.

Analogiczne typy istanieją dla uczenia bez nadzoru jak na przykład *grupowanie*, które klasyfikuje dane w zbiory. Rozbieżność z klasyfikacją polaga na wykorzystaniu do wykonania oceny korelacji podobnych cech, a nie wsad danych testowych.

Redukcja wymiarowa to jak nazwa wskazuje pozbycie się nieistotnych atrybutów i odrzuceniu duplikatów, a co za tym idzie wymiaru data set'u. Dobrym przykładem byłoby tutaj analizy zawartości skrzynki pocztowej i szukanie spamu.

Podział osób na kategorie cierpiące na choroby sercowo-naczyniwe oraz zdrowe, to dylemat klasyfikayjny nadający się do rozwiązania za pomoca algorytmów uczenia maszynowego nadzorowanego i na nich skupia się dalsza część pracy.

Ścieżka działania algorytmów uczenia maszynowego nadzorowanego

Bazowy schemat budowania i oceniania modeli uczenia nadzorowanego to kolejno:

1. Przygotowanie danych.
2. Implementacja modelu.
3. Trening oraz ocena precykcji.

Wykorzystanie utworzonego modelu wymaga:

1. Zbudowania modelu oraz jego wytrenowania
2. Wykonania predykcji na danych które nie posiadają oznaczenia[3]

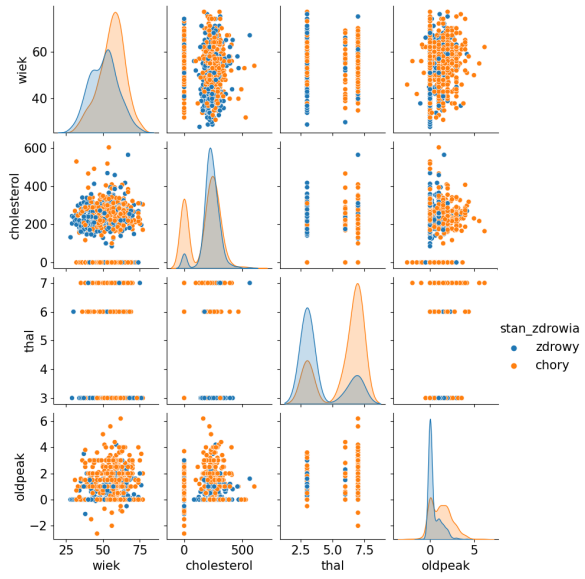
Ten schemat można zastosować do dowolnego modelu uczenia, wykonanie kolejnych bloków zadaniowych różnić się będzie specyfiką dla danego modelu:

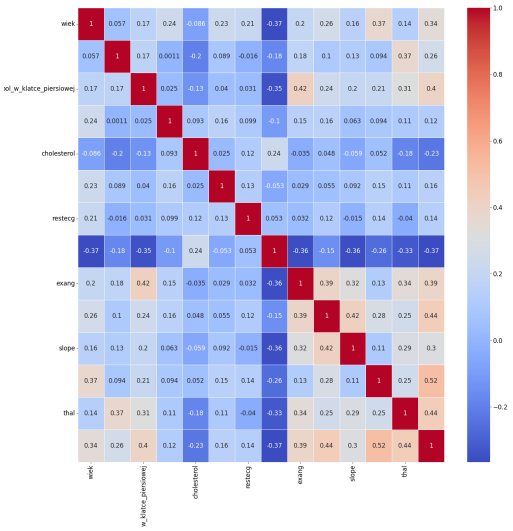
- przygotowanie danych dla algorytmów uczenia nadzorowanego musi zawierać również zebranie odpowiedzi/wyników dla danych testowych
- implementacja modelu za każdym razem jest specyficzna dla zastosowanego algorytmu który równiez zależy od typu uczenia
- ewaluacja modeli regresywnych różni się od ewluacji modeli klasyfikacyjnych ze względu na wykorzystanie innych miar oceny dokładności
- zastosowanie modeli może posiadać wielorakie formy realizacji.

Dane

Terminologia

Analiza uczenia maszynowego wymusza stosowanie rozróżnienia przy pojęciach parametru i argumentu. Cechy dla których szukamy optymalnych wartości , które stanowią podstawe modelu i ich dostrajanie wykonywane jest podczas treningu nazywane są parametrami i hiperparametrami. Argumenty natomiast to liczby znajdujące się w wierszach zbioru danych które podlgają zmianie tylko podczas preprocesingu Nazewnictwo hiperparametrów wykorzystywane jest w przypadku zastosoawnia dla nich walidacji krzyżowej.





Nadmierne dopasowanie(ang. overfitting *)*

Istnieje możliwość wytrenowania algorytmu tylko pod zadany zbiór testowy, wyniki dotyczące dokładności modelu pomimo iż będą wysokie nie będą świadczyły o rzeczywistej skuteczności gdyż będą uwzględniały tylko pojedyńczy przypadek zasymulowany daną próbką danych testowych. Przed wyborem cech do hiperparametryzacji warto sprawdzić macież korelacji cech aby nie wybrać tylko tych silnie ze sobą sprzężonych[13].

Repozytorium uczenia maszynowego UCI

Sensem wykorzystania uczenia maszynowego jest prognoza lub klasyfikacja rzeczywistych wartości z dużego zbioru danych które mogą znaleść zastosowanie w praktycznych dziedzinach. Im bardziej dokładne i rzeczywiste dane do testowania i tworzenia modelu tym większe prawdopodobieństwo otrzymania realnych wyników na końcu ścieżki uczenia.

[14]

W celu gromadzenia miarodajnej bazy dostępnych zbiorów danych testowych powstało repozytorium uczenia maszynowego UCI. Jak podaje strona informacyjna :

> … było ono cytowane ponad 1000 razy, co czyni je jednym ze 100 najczęściej cytowanych „artykułów" w całej informatyce … [14]

Repozytorium gromadzi dane z wielu rozbieżnych dziedzin , dane medyczne umieszczone w repozytorium nie zawierają wrażliwych danych pacjentów , a niektóre zbiory są poddane już wstępnej obróbce tak jak zbiór danych *"Heart Disease Databases"* wykorzystany w tym dokumencie, który powstał na podstawie realnych danych medycznych zebrany z lokalizacji :

1. Fundacja Cleveland Clinic [15]
2. Węgierski Instytut Kardiologii, Budapeszt [16]
3. V.A. Centrum medyczne, Long Beach, Kalifornia [15]
4. Szpital Uniwersytecki, Zurych, Szwajcaria [17].

Stratyfikacja

Wyróżniono 14 atrybutów spośród 76 zebrancyh do wykorzystania w algorytmach uczenia maszynowego, wszystkie z nich mają wartośi liczbowe. Lista atrybutów wykorzystanych w algorytmie:

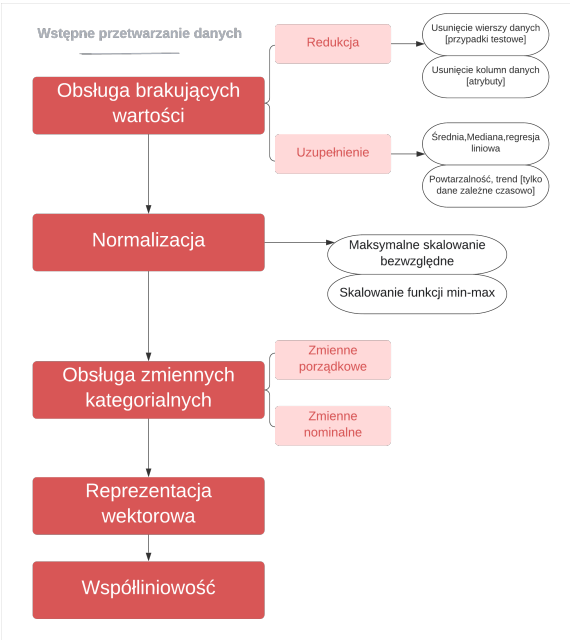
- wiek
- płeć
- rodzaj bólu w klatce piersiowej
- spoczynkowe ciśnienie krwi
- cholesterol w surowicy w mg/dl
- poziom cukru we krwi na czczo > 120 mg/dl
- spoczynkowe wyniki elektrokardiograficzne
- osiągnięto maksymalne tętno
- dławica piersiowa wywołana wysiłkiem fizycznym
- obniżenie odcinka ST wywołane wysiłkiem fizycznym w stosunku do odpoczynku
- nachylenie szczytowego odcinka ST ćwiczenia
- liczba głównych naczyń pokolorowanych fluorozopią
- skan serca z talem lub test wysiłkowy
- stan (brak choroby serca/choroba serca)

Rozkład chorób serca w danych testowych to 44.67% chorych czyli 509 prób pozytywnych oraz 411 negatywnych. W danych testowych znajduje się 726 przypadków osób płci męskiej oraz 194 żeńskiej. Dla zachorowań widać nierówność ale jest ona spowodowana rzeczywistą statystyką. Tylko u 25.77% badanych kobiet stwierdzono występowanie chorób wieńcowych, natomiast wśród badanych mężczyzn jest to aż 63.22%. [14]

W przypadku danych testowych z repozytorium UCI, fakt iż dane pochodziły z różnych lokalizacji ma duże znaczenie ,gdyż od placówki medycznej zależy jakim badaniom poddani zostali pacjęci a co za tym idzie w jakich kolumnach tabelarycznego przedstawienia będą mieć uzupełnione bądź puste wartości. Scalenie ze sobą wyników badań dostarcza większej różnorodności również dzięki temu że dane pochodzą z wielu krajów. Jeżeli zestaw wejściowy zostałby ograniczony do jednej lokalizacji to cecha dla której nie uzupełniono wartości zostałaby pominięta podczas treningu ze względu na brak danych, co skutowało by uboższym modelem i możliwe że pominięciem kluczowej cechy wpływającej na działanie.

Wstępna obróbka danych

Proces przetwarzania danych może skladać sie z wielu różnych kroków zależenie od typu, w uczeniu nadzorowanym operującym na danych tekstowo-liczbowych poprawnym będzie zastosowanie schematu przedstawionego poniżej:



Po złączeniu można przeprowadzić szereg działań w celu sztucznego uzupełnienia pustych wartości bazując na wartościach które już istnieją.

Obsługa brakujących wartości

Możliwościami obsługi brakujących wartości są : mniej polecana ze względu na utrate danych, redukcja zestawu danych lub uzupełnienie go zgodnie z wybrany przez siebie założeniem. Biblioteki do nauczania maszynowego dostarczają już gotowe rozwiązania do upuszczenie wierszy lub kolumn zawierających wartości *null*. Uzupełnienie danych inaczej *imputacja*, rozwiązuje problem w mniej stratny sposób i tak samo jak do redukcji są już gotowe rozwiązania w bibliotece sklearn. Istnieją 4 różne strategie uzupełniania wykorzystujące proste matematyczne obliczenia takie jak :

- średnia,
- mediana,
- stała,
- najczęściej występująca wartość.

Do wyznaczenia wartości uzupełniających można również użyć regresji liniowej.

Standaryzacja

Przeksztaucenie danych również bazujące na statystycznych założeniach i również ustandaryzowane w popularnych bibliotekach. Dążymy aby średnia wartoś wynosiła 0, a odchylenie standardowe 1 dla liczbowyh reprezentacji danych. Z matematyczne punktu widzenia wykonujemy działanie

$$\frac{\bar{X}}{\sqrt{\frac{\sum_{i=1}^{n}(X-\bar{X})^2}{N-1}}}$$

[18]

Obsługa zmiennych kategorialnych

Cechy kategorialne dzielą się na dwie zasadnicze grupy ze względu na mozliwość uprorządkowania ,

dane takie jak wyksztaucenie , rozmiar podlegają mapowaniu , dane typu kolor lub płeć podlegaja kodowaniu. W ten sposób dane kategoryczne staja się wartosciami liczbowymi.

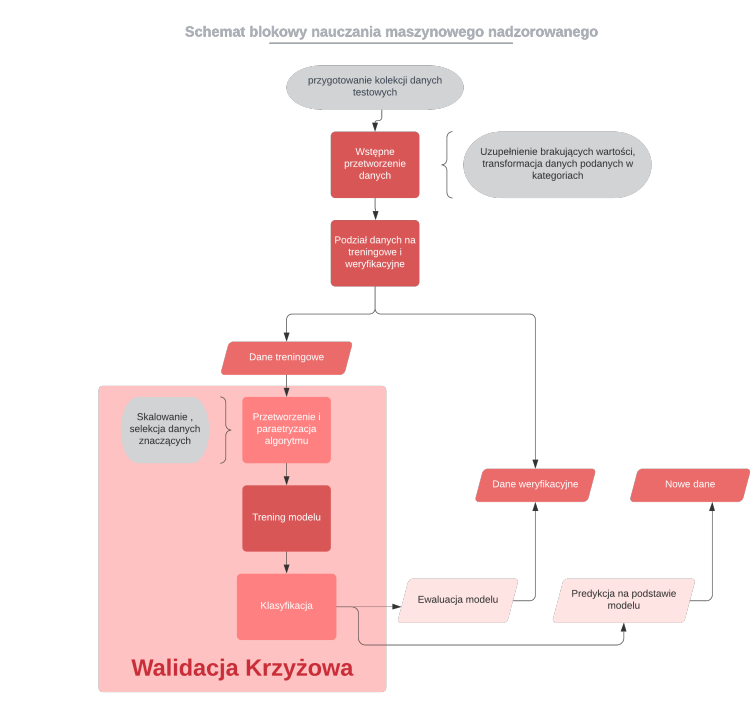
Reporezentacja wektorowa

Obsługa danych kategorialnych pozwoliła zmapować/zakodować je w postaci liczbowej, ale można pójść o krok dalej i te same dane mieć w postaci 0 lub 1 na odpowiedniej kolumnie. Rozwiązanie reprezentacji wektorowej polega na utworzeniu tylu kolumn ile jest unikalnych wartości dla kategorii i wpisanie 0 lub 1 dla każdego rekordu danych [19] .

Współliniowość cech

Aby znaleść korelacje współliniowości należy szukać liniowej zależności pomiędzy danymi, najłatwiej zauważyć to tworząc wykresy z danych testowych dla każdej pary [19].

Zgodnie z poniższym schematem po przeprocesowaniu wejściowego zbioru danych, należy go podzielić na dane treingowe oraz ewaluacyjne. Powszechnie stosowana K krzyżowa walidacja umożliwia maksymalne wykorzystanie dostarczonego wejścia do dostrajania parametrów modelu, ponieważ optymalizacja hiperparametrów połączone z ciągłą weryfikacją poprawności to sedno treningu.



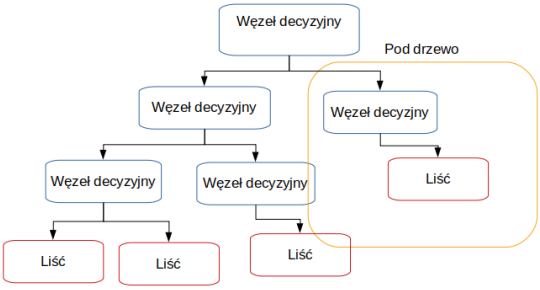
K-krotna walidacja krzyżowa (ang. *K-fold Cross Validation*, KCV) - metoda weryfikacji działająca poprzez podział zbioru danych na k podzbiorów z których każdy przynajmniej raz jest zbiorem oceniającym wydajność , zaznaczając że K musi być równe lub mniejsze niż liczba elementów w zbiorze [20], [21] .

Kluczowym elementem jest ewaluacja która odbywa się na końcu każdej z k-1 iteracji w celu dostosowania parametrów, po osiągnięciu wymaganych lub ustalonych wartości dokładności modelu lub weryfikacji wszystkich możliwych opcji i znalezienie najlepszego modelu można go wykorzystać do weryfikacji na danych spoza zestawu testowego.

Wybrane algorytmy uczenia maszynowego nadzorowanego

Losowe lasy decyzyjne

Drzewa decyzyjne (ang. *decisions trees*) są uznawane za najprostyszy i najbliższy ludzkiemu zrozumieniu algorytm uczenia, który swoją nazwę zawdzięcza graficznej reprezentacji w postaci drzewa. Każdy węzeł oznacza atrybut, na podstawie którego następuje rozróżnienie. W modelu kluczowa jest kolejnośc cech, które występują po sobie ponieważ determinuje to otrzymany rezultat [8], [22] .



Prawie każdy algorytm uczenia maszynowego nadzorowanego można podzielić na dwa etapy. W pierwszym opracowywany jest wzorzec, na którym bazuję późniejsza predykcja. Uczenie składa się z dwóch części w wariancie drzew decyzyjnych uczenie to tworzenie rozgałęzień reprezentujących atrybuty dzielące zwstaw testowy aż dalszy podział jest niemożliwy.Takie drzewo może mieć dowlnie długą droge po węzłach, niestety taki sposób rozwiązania jest przyczyną powstawnia przyapdku _overfitting'u. Ograniczenie głebokości drzewa lub minimalna liczba wartości w liściu zminiejsza ale nie niweluje ryzyka. [3]

Na metodologie drzew decyzyjnych oparta jest dokładniejsza forma nauczania nadzorowanego: *losowe lasy decyzyjne.*

Losowe lasy decyzyjne (ang. *random decision forests*) to technika polegająca na połączeniu wielu drzew decyzyjnych w celu uniknięcia problemu z *nadmiernym dopasowaniem* do treningowego zestawu danych na którym został przeszkolony.

Utworzony szablon aby poprawnie działać na danych testowych i służacych weryfikacji, nie może stać się charakterystycznym przypadkiem rozwiązującym przypadek testowy [8], [22]. W tym celu dla loswoych lasów decyzyjnych najpierw stosuję się **agregację bootstrap'ową**. Z treningowego zestawu danch losuję się, z możliwymi powtorzeniami, wiersze danych dla których trenowany będzie model. Jako rezutat brana jest większość lub średnia wartości uzyskanych wyników dla poszczególnych drzew dezycyjnch. Dodatkowo dla drzew decyzyjnych w lasach losowych, atrybuty odpowiadające za kategoryzację są wybierane z wylosowanego podzbioru.[23]

Działanie biblioteki sklearn dla lasoów losowych wygląda następująco : 0. Wylosowanie podzbioru cech włączanych w dane drzewo. 1. Typowanie kombinacji podziału obliczając prawdopodobieństwo wyboru i uśredniając wyniki. Wybierany jest podział na klase o najwyższym średnim prawdopodobieństwie . 2. Utworzenie nowego węzła . 3. W każdym podwęźle należy zweryfikować powiązania i jeżeli spełniony jest warunek wartości docelowych są wystarczająco podobne, zwracana jest prognozowana wartość w przeciwnym razie należy powtórzyć od kroku 1.

Technika bootstrap

Główną wartością z jej zastosowania jest nadanie losowości tworzenia drzew , podział można wykonać pobierając próbki z zwracaniem lub bez. Brak możliwości ponownego wyboru wcześniejszej cechy uniezależnia je od siebie . Metoda z zwracaniem wymaga powtarzania aż do wyrania próbki liczącej

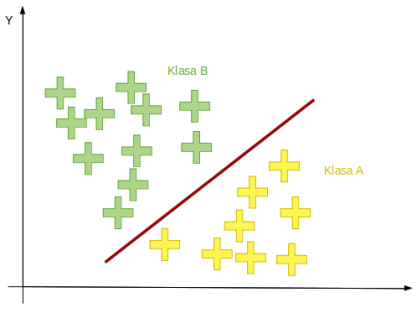
tyle samo co macieżysta kolekcja. Potem po podliczeniu statystyki i ich średnich dla każdego wykonania proces powtarzany jest aż do uzyskania warunku końcowego. [3]

Ekstremalne lasy losowe Ekstremalność polega na wyselekcjonowania losowego podzbioru punktów podziału i dla tego podzbioru przeprowadzana jest ewaluacja. W związku z tym do kroków procesu oprócz losowania cech , dodatkowo losowo określane są tez podziały.

Wśrod zalet lasów losowych nalezy wyróżnić iż potrafią one trafnie wykalkulować brakujace wartości cech. Idealnie znajdują zastosowanie dla realnych danych, których zasadniczym problemem jest ich niekompletność.
Dane medyczne posiadają szeroką wariację zmiennych z dużym prawdopodobieństwem wybrakownia, zastosowanie do nich lasów decyzyjnych ma potencjał na pozytywne rezulataty.

Maszyna wektorów nośnych

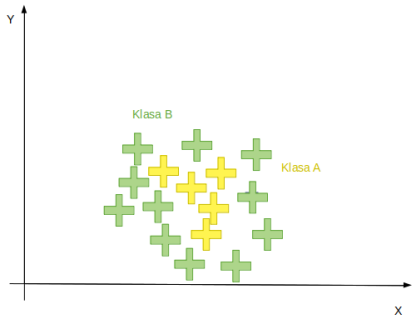
Metoda wektorów nośnych (ang. *support vector machines* , skr. ***SVM***) to algorytm uczenia maszynowego nadzorowanego, który każdy parametr z dostępnych cech dla danych wejściowych, traktuje jako punkt w przestrzeni. Na podstawie ułożenia punktów dzieli się je na 2 klasy. Graficznie jest to reprezentowane przez prostą dla której odległość między najbliższymi dwoma punktami dla wektorów jest możliwie największa.



[^schemat_wzorowany]

Taka prosta nazywana jest *prostą marginalną* i powstaje ona poprzez generowanie i selekcję tych prostych które rzetelnie szufladkują klasy danych [8], [22].

Techinka ta gwarantuje precyzyjniejsze rezulatay niż drzewa decyzyjne, niestety dla dużych zbiorów danych czas trwania szkolenia znacznie się wydłuża oraz istnieją przypadki dla których podział jedną prostą jest niewykonalny, taki przypadek reprezentuje rozkład na schemacie nr. 2.



[^schemat_wzorowany]

Z powyższego schematu widać że prosta marginalna ma zastosowanie w przypadku dwóch wymiarów, dla większej ilości stosowane jest przekształcenie do innego systemu współrzędnych i szukanie hiperpłaszczyzny brzegowej dzielącej tak samo jak prosta punkty w przestrzeni na dwa zbiory.[24]

Wyszukiwanie podziału

Idea działania maszyny wektorów nośnych opiera się na wyznaczenia minimalnej wartości wektora wag oraz przesunięcia (ang. *bias*) który geometrycznie opisuje współrzędne hiperpłaszczyzny.

$$\min_{w,b} \frac{1}{2}||w||^2$$

$$\text{pod warunkiem } y_i(w \cdot x_i + b) - 1 \geqslant 0,\ i = 1, \ldots, n.$$

[25]

W przypdaku prostej wersji podziału poprzez prostą optymalizacja polega na redukcji danych potrzebnych do uzyskania rozbicia.Margines między kategoriami powinien być maksymalny żeby zminimalizować błąd dla próby testowej oraz proste uogólnianie. Nawet w przypadku zastosowania hiperpłaszczyzny punkty dzielone sa na 2 klasy dlatego nie stosuje się go do grupowania i klasywikacji dla większej ilosci.W biblitece sklearn konwencja przyjmuje stosowanie zasady podziału ang. *one-versus-rest* i oferuje ponad trzy podejścia formułowania estymatora klasyfikacji: *SVC*,*LinearSVC*,*NuSVC* itd. .

K najbliższych sąsiadów

K najbliższych sąsiadów (ang. *k nearest neighbours*, skr. ***KNN***) to algorytm uczenia maszynowego nadzorowanego operający swoje estymacje dla konkretnego przypadku danch na wartościach jego K najbliższych sąsiadów(punktów) liczonych min. dla przestrzeni Euklidesowej [8]. Koncepcja polega na :

0. Wyszukanie k prób do porównania zgodnie z metryką.
1. Analiza podobieństwa dla wybranej próbki .
2. Selekcja najczęściej pojawiającej się odpowiedzi i oznaczenie sklasyfikowanych wartości.[3]

Do wyznaczenia odległości w metryce Euklidesowej stosowany jest wzór:

$$d_{(x,y)} = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

[26]

popularne są również przestrzenie Manhattan:

$$d_{(x,y)} = \sum_{i=1}^{n}|(x_i - y_i)|$$

[26]

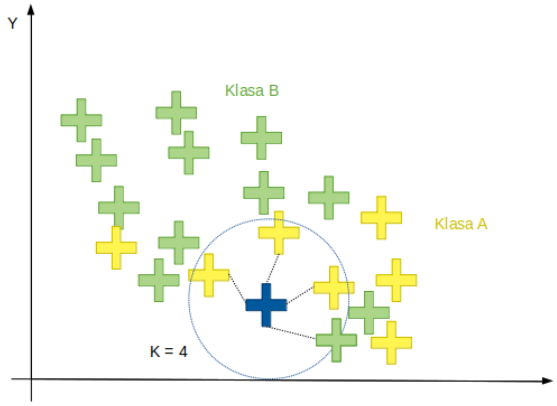
oraz Mińkowskiego:

$$d = \left(\sum_{i=1}^{m}|u_i - v_i|^p\right)^{1/p}$$

[27]

Atrybut który nastraja proces uczenia się modelu i ma na niego największy wpływ określany jest jako hiperparametr. Dla KNN jest to liczba sąsiadów i może przyjmować maksymalnie wartości do rozmiaru zbioru cech. Im większa ilość jednostek mających wpływ, tym potęguje się niestety złożoność czasowa algorytmu, znacząco już większa od przedstawionych powyżej innych algoryrtmów,[8] oraz tym bardziej wzrasta ryzyko nadmiernego dopasowania do modelu testowanego.

W celu przewidzenia wartości dla nowych danych, należy odnaleść K najbliższych punktów wyliczając odległości, a następnie przpisać odpowiedź implikowaną przez większość sąsiadów. Dla wartości K równej jeden, metoda ta nazywana jest algorytmem najbliższego sąsiada.



[^schemat_wzorowany]

Dla lekarza wartością dodatnią jest wykrycie zależności które decyzują o uznaniu lub zaprzeczeniu występowania choroby. Zastosowanie algorytmu KNN może nie tylko zakwalifikować osoby chorujące na serce, ale również ułatwić swoją graficzną reprezentacją wpływ cech na ostateczny osąd próbki.

Opis praktycznej częsci projektu

Narzędzia i biblioteki zastosowane w pojekcie

Biblioteki w większości posiadają otwarty kod źródłowy, napisany w języku Python [28].

Python

Język programowania wysokopoziomowego umożliwiający programowanie zorientowane obiektowo. Pozwala na tworzenie klas , dziedziczenie , polimorfizm odraz hermetyzacje atrybutów w klasie. Python wykorzystywany jest również do pisania skryptów . sam interpreter można rozbudowywać o nowe typy danych zaimplementowane w C lub C++. Na oficjalnej stronie *https://www.python.org/* można zajrzeć do kodu źródłowego który jest udostepniony publicznie. Darmowa biblioteka ułatwia tworzenie szybkich aplikacji oraz integrację.[11]

Scikit-learn

Praktyczna część pracy napisana została w języku Python z wykorzystaniem *scikit-learn*, obsługującym wiele algorytmów maszynowego uczenia się w tym uczenia nadzorowanego i docelowo wybranych algorytmów przedstawionych w teoretycznej części pracy.

Biblioteka rozwijana przez ponad 10lat opiera się o *Numpy* oraz *Scipy*, daje zestaw narzędzi do obliczeń na macierzach, wektorach oraz umożliwiający metody numeryczne takie jak całkowanie, różniczkowanie i temu podobne [29]. W rezultacie można za jej pomoca wykonać elementy procesu nauczania algorytmu, takie jak: przetwarzanie wstępne, redukcja wymiarowości, klasyfikacja, regresja. [28] Pomimo cieszenia się dużym zaufaniem, implementacja nie koniecznie musi być najoptymalniejsza dodatkowo korzystanie z Numpy zwiększa ryzyko błędów samej biblioteki. Za jej pomocą można wygenerować przykładowa dane, wyliczyć metryki wydajności oraz zinterpretowac wyniki klasyfikacji.[13]

Pandas Do przygotowania danych wykorzystano zestaw narzędzi *Pandas*, ułatwiający tworzenie struktur danych i ich analizę.

Matplotlib W celu wizualizacji wyników w postaci wykresów zastosowano, opartą na *Matplotlib*, bibliotekę *Seaborn* powszechnie stosowaną do rysowania estetycznej grfiki statystycznej.

Flask Część prezentacyjna czyli możliwość wprowadzenia danych w formularzu na stronie i weryfikacja wyniku dla wyuczonych już modeli wykorzystuje bibliotkę *Flask*. Framework Flask ułatwia pisanie aplikacji internetowch i jest rozwiązaniem które daje duży zakres dowolności oraz możliwości. Flask

sam z siebie nie definiuje warstwy bazy danych czy formularzy, pozwala za to na obsługę rozszerzeń które ubogacają aplikację o wybraną funkcjonalność. [30]

JsonPickle i *JobLib* Przekazywanie obiektów o bardziej skomplikowanej budowie i ich *serializacja* oraz *deserializacja* do formatu JSON wykonane są za pomocą biblioteki *jsonpickle*, a zapis modeli wykonano za pomocą *joblib* która zapewnia obsługę obiektów Pythona i jest zoptymalizowana pod kątem pracy na dużych tablicach Numpy. [28]

Środowisko wykonania

Wykonanie programu i analizę danych testowych wykonano na maszynie o parametrach :

```
Procesor      Intel(R) Core(TM) i5-6300U CPU @ 2.40GHz   2.50 GHz
Zainstalowana ęćpami RAM    8,00 GB (ędostpne: 7,82 GB)
Typ systemu 64-bitowy system operacyjny, procesor x64
```

Wersjie bibliotek wykorzystanych w projekcie:

```
setuptools~=49.2.1
Flask~=2.0.1
matplotlib~=3.4.2
numpy~=1.20.3
pandas~=1.2.4
scikit-learn~=0.24.2
pytest~=6.2.5
joblib~=1.0.1
scipy~=1.6.3
seaborn~=0.11.2
jsonpickle~=2.1
```

Interpreter Python w wersji *3.9*.

Moduły projektu:

Wstęp

- Config - zawiera statyczne zasoby oraz konfigurację logowania projektu
- Data - moduł odpowiada za wczytywanie i obróbkę danych testowych, zawiera definicje objektów wykorzystywanych przy uczeniu oraz zapisu modelu oraz przekazywaniu wyników prezentowanych na stronie
- Management:
  - PlotGeneration - moduł odpowiedzialny za prezentację wyników w postaci wykresów porównujących algorytmy oraz odpowiedzi na zadany problem oraz przechowuje obiekty które przy wywoływaniu funkcji od strony www są jedynie formatowane i zwrcane
  - Prediction - przygotowanie parametryzacji oraz implepmentacja treningu dla każdego z 3 algorytmów :
    * RF - przygotowanie, trening i zapis modelu sprecyzowany dla klasyfikatora lasów losowych
    * KNN - przygotowanie, trening i zapis modelu sprecyzowany dla klasyfikatora k-najbliższych sąsiadów

    - SVM - przygotowanie, trening i zapis modelu sprecyzowany dla maszyny wektorów nośnych

- Static - folder z grafikami, plikami stylów, skryptami javascript oraz jQuerry
- Templates - folder z stronami html wykorzystującymi dyrektywy Flask

Projekt posiada dwa tryby pracy :

- tryb nauczania na podstawie danych testowych
  machine learning z wykorzystaniem 3 algorytmów (*Run_Learning_Proces.xml*) , musi być wykonany przynajmniej raz przed wykorzystaniem programu jako aplikacja

- tryb aplikacji web
  wykorzystanie Flask do prezentacji i wykorzystania utworzonych modeli (*Run_Web_Application.xml*) , strona prezentuje analizę danych oraz uczenia algorytmów , przy czym głównym zadaniem jest wykonanie predykcji na podstawie danych wpisanych do formularza.

Trening algorytmu

Wstęp

Głównym zadaniem trybu nauczania jest utworzenie i wytrenowanie modeli dla 3 algorytmów nauczania nienazdorowanego, w tym celu wykonywany jest preprocesing danych czyli kolejno:

Przygotowanie danych

Proces przygotowania danych zastosowany w prajekcie składa się z następujących kroków:

1.Załadowanie i konkatenacja dataset'u , standardowo również wybranie cech znaczących głownie odbywające się poprzez odrzucenie nadmiarowych parametrów ,ale istnieje też możliwość dodania nowych np utowrzenie powierzchni na podstawie wymiarów zawartych w danych testowych.. Następnie należy wykonać wyeliminowanie cech nie wpływających na odpowiedź i dopiero po dokonaniu selecji przystapić do przetwarzania zebranych informacji. 2.Uzupełnienie pustych wartości - dla późniejszego porównania w projekcie tworzone są imputery dla 4 różnych form uzupełnienia, ale są to najbardziej podstawowe działania typu średnia wartość , najczęściej występująca wartość. Idealnym rozwiązaniem było by w przypadku posiadania eksperckiej wiedzy z danej dziedziny uzupełnienia brakujących wartości własnymi propozycjami. Innym sposobem może być wykorzystanie heurystyk specyficznych dla tworzonego modelu[3]. 3.Standaryzacja 4.Konwersja danych dla kategorii 5.Normalizacja z wykorzystaniem MinMaxScaler, zmiana skali w formie przykładu to na przykład przeliczenie temperatury z stopni Celcjiusza do FahrenHeita. Wykonuje się ją by przesunąć wartości skrajne i pozbyć się nierówności w zbiorze[3].

Trening algorytmu

Podział danych na dane treningowe i testowe wykorzystuje zdefiniowane w bibliotece sklearn predefiniową funkcje train_test_split zwracającą cztery obiekty tablic dla x_testowych , y_testowch , x_treingowych oraz y_treningowych[3]. Tak spreparowany zestaw danych poddawany jest treningowi modelu kolejno dla każdego z algorytmów. Do dostrojenia parametrów oraz znalezienia najlepszego modelu wykorzystuwany jest:

GridSearchCV

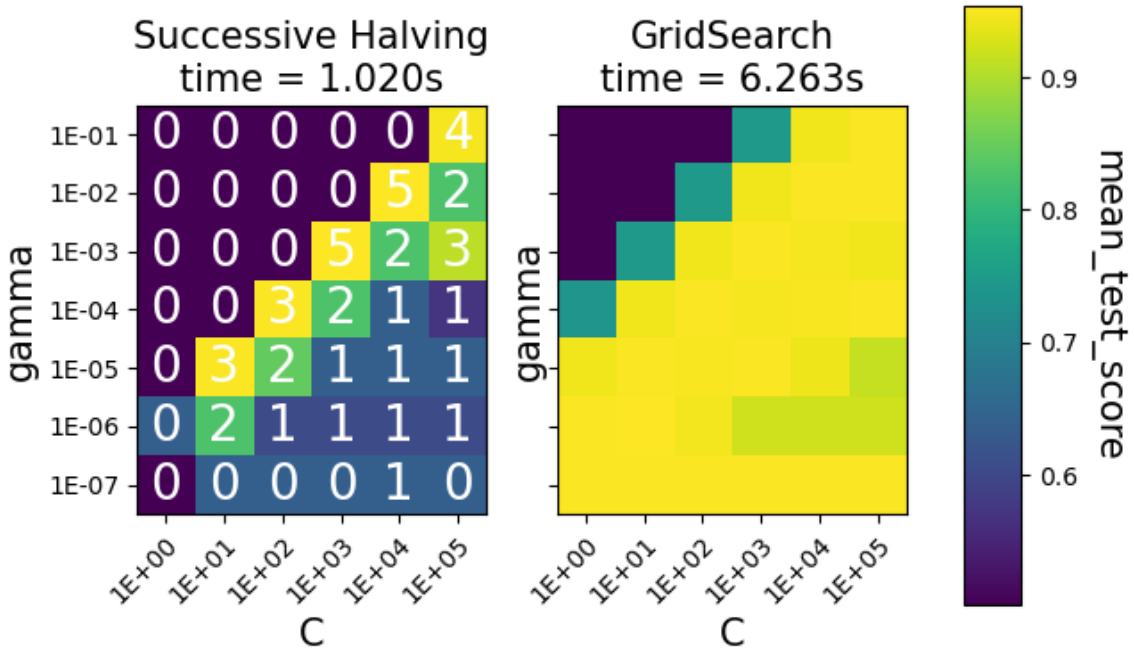
Model trenowany jest na podstawie siatki parametrów i sprawdzana jest każda kombinacja by uzyskać konfiguracje parametrów dla najlepszego estymatora.

W projekcie dla każdego algorytmy zapróbkowano większość dostępnych dla danego modelu klasyfikacji hiperparametrów przekazywane w param_grid. Wykorzystane parametry wykonania GridSearchCv [29]:

- estimator: implementacja interfejsu obiekt estymatora scikit-learn,
- param_grid: słownik parametrów które są potem testowane w dowolnej sekwencji ustawień,
- refit: dopasowanie best_estimator_ ,best_index_, best_score_ i best_params_ dla najlepszej sekwencji ustawień parametrów
- cv: parametr k dla KFold walidacji krzyżowej,
- verbose: obszerność logowanych informacji

HalvingGridSearchCV

Sciikit-learn udostępnia również inne implmentacj zastosowania walidacy krzyżowej np.: *HalvingGridSearchCV* lub *HalvingRandomSearchCV*. HalvingGridSearchCV polega na zmniejszaniu o połowe (z ang. *half*) zbioru parametrów po każdej iteracji algorytmu krzyżowego. Ta strategia wyszukiwania sukcesywnie zmniejsza ilość wymaganych iteracji dla danego zestawienia przez co wykonania jest szybsze niż w przypadku zwykłego GridSearchCv. Na poniższym wykresie przedstawiającym średni wynik dla algorytmu SVC widać że czas wykonania zmniejszył się ponad 6 krotnie w stosunku do GridSearch.



[29]

Umieszczone oznaczenia od 0 do 5 informują o tym w której iteracji kombinacja parmaretów została oznaczona jako najlepsze zestawienie. Implementacja ta nie została wykorzystana ze względu na nadal pozycjonowanie jej jako eksperymentalnej.

Podczas uczenia i wykonania funkcji *fit* w resultacie otrzymujemy zadane wcześniej informację jakie hiperparametry po przejciu sprawdzianu krzyżowego zostały uznane za wystarczająco precyzyjne do utworzenia modelu[3].

Pierwszym z wymaganych argumentów *GridSearchCV* są estymatory. W projekcie ich impementacja pochodząca z bilioteki oraz dostępną dla nich parametryzacje daje w wyniku następujące zestawienie najlepszych osiągniętych estymatorów:

KNeighborsClassifier [29] :

- n_neighbors: [todo] - liczba sąsiadów z których wnioskowany jest jednostkowy resultat
- weights: - wagi na podsatwie których wyliczana jest predykcja , można zastosować wagę 1:1 lub nałożyć wagi zgodnie z dystansem.
- algorithm: - algorytm zastosowany do znalezienia najbliższych sąsiadów, w projekcie wykorzystano : brute-force oraz auto
- leaf_size: - rozmiar liścia dla algorytmów BallTree or KDTree
- p: - wykorzystanie miar odległości dla manhattan
- metric: -metryka odległości

RandomForestClassifier [29] :

- criterion: [todo] - funkcja pomiaru dokładności rozgałęzienia
- min_samples_leaf: [todo] -minimalna liczba probek wymagana na liściu.
- min_weight_fraction_leaf: [todo] -minimalny ułamek sumy wag wymagany na liściu
- min_impurity_decrease: [todo] - większe lub równe zmniejszenie zanieczyszczenia powoduje podział danego węzła
  Zmniejszenie zanieczyszczenia liczone jest zgodnie z wzorem:

```
N_t / N * (impurity - N_t_R / N_t * right_impurity
                    - N_t_L / N_t * left_impurity)
```

gdzie N to całkowita liczba próbek, N_t to liczba próbek w bieżącym węźle, N_t_L to liczba próbek w lewym liściu, a N_t_R to liczba próbek w prawym liściu.

- max_features: [todo] - liczba funkcji najlepszego podziału
- random_state: [todo] - wykorzystywany przy próbkowaniu cech przy poszukiwaniu najlepszego podziału w węźle
- cpp_aplha: [todo] - zastosowanie to przycinanie drzewa o największej złożoności mniejszej niż cpp_alpha

SVC [29] :

- C: [todo]float, default=1.0 Regularization parameter. The strength of the regularization is inversely proportional to C. Must be strictly positive. The penalty is a squared l2 penalty.
- kernel: [todo]- jądro wykorzystane w algorytmie
- degree: [todo] - stopień dla funkcji jądra *poly*
- gamma: [todo] - współczynnik jądra dla wartości *scale* parametr jądra ustawiany jest na wartość:

```
1 / (n * X.var())
```

dla wartości auto jest to :

```
1 / n
```

gdzie n to liczba cech.

- coef0: [todo] - niezależny parametr funkcji jądra , wykorzystywany tylko przy jądrach *poly* i *sigmoid*.
- shrinking: [todo] - heurystyka kurcząca
- cache_size: [todo] - cache jądra (w MB).

SVC używa jednego podstawowego parametru, C, do kontrolowania kompromisu między obciążeniem a wariancją. Bezpośrednia interpretacja tego parametru jest trudna. Klasa NuSVC wykonuje to

samo zadanie co SVC, ale za pomocą innych obliczeń. Zwracam na to uwagę, ponieważ podstawowy parametr klasy NuSVC, (ny, ang. nu), ma proste znaczenie: przynajmniej procent danych jest zachowywanych jako wektory nośne. Ma to wpływ na błędy, przy czym są one określonego rodzaju — są to błędy marginesu. Błędy marginesu to punkty, które albo (1) znajdują się po złej stronie separatora (jest to błąd klasyfikacji), albo (2) znajdują się po właściwej stronie (zostały poprawnie sklasyfikowane), ale na marginesie. Inne znaczenie parametru jest takie, że dla danych treningowych akceptowany jest maksymalnie procent błędów marginesu. W określonych warunkach procent błędów marginesu rośnie do , a procent danych w wektorach nośnych spada do . Wartości znajdują się w przedziale [0, 1] i są interpretowane jako wartość procentowa od 0% do 100%. Choć klasa SVC jest trudniejsza do interpretacji, ma lepszą charakterystykę działania niż NuSVC. Parametry klasyfikatorów SVC są nieco magiczne. Są jak tajniki znane tylko alchemikom. Aby zrozumieć ich znaczenie, trzeba zagłębić się w zaawansowaną matematykę. Można jednak analizować skutki używania tych parametrów na niewielkich przykładach (napisałem „niewielkich", a nie „prostych"). Zobacz, jak zmiany wartości parametrów C i wpływają na granice między klasami.

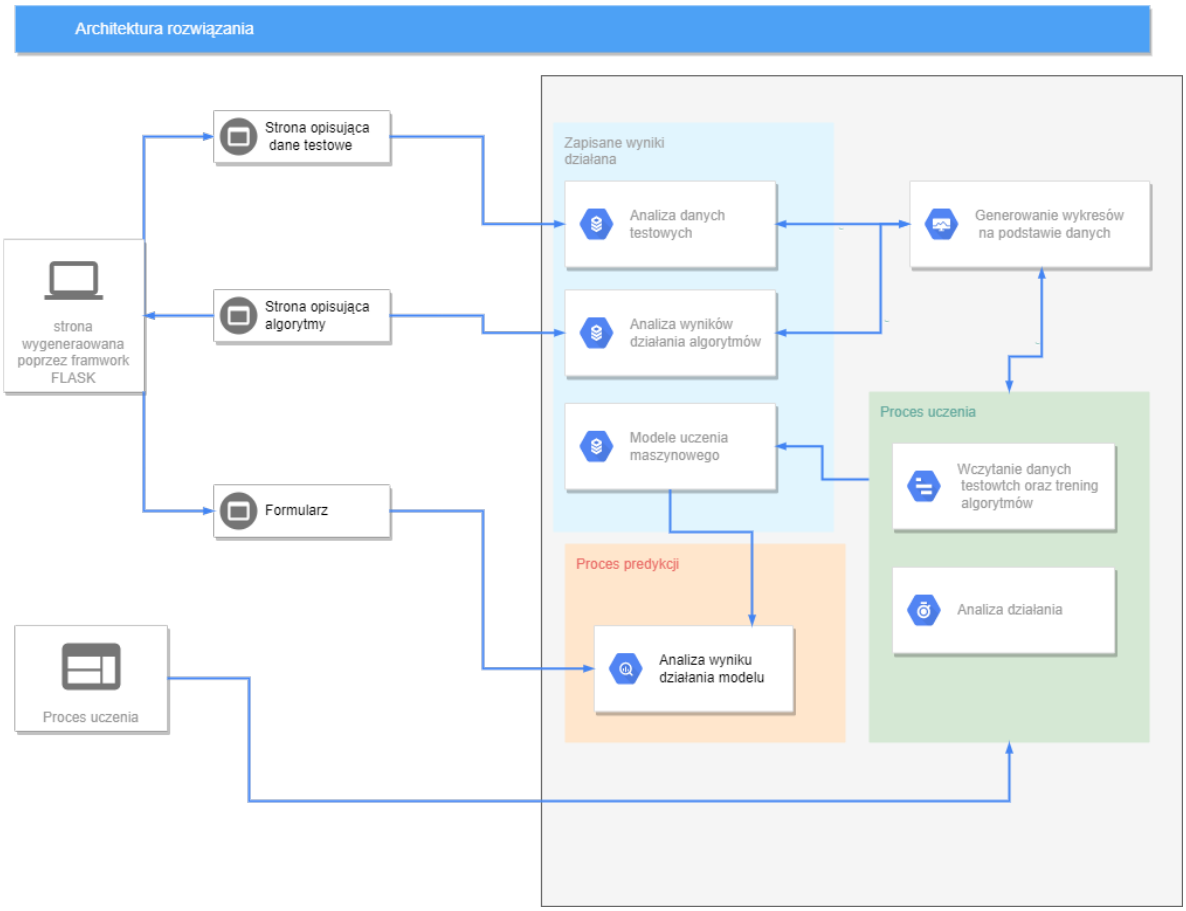
Z tego przykładu należy zapamiętać dwie rzeczy: 1. Zwiększanie i zmniejszanie C dają mniej więcej ten sam efekt. Jednak skale dla obu tych parametrów są wyraźnie różne. C zmienia się o rzędy wielkości, natomiast zmieniane jest liniowo w krokach o wielkości 1/10. 2. Duże i małe C mogą prowadzić do klasyfikatora SVC, który w pewnym zakresie ignoruje błędy klasyfikacj[3]

Po odnalezieniu najlepszego estymatora model jest zapisywany oraz generowane sa wykresy dla trybu aplikacji webowej:

- wykresy modeli datasetu wejściowego i rozłożenia cech
- wykresy prezentujące zestawienia danych zebranych na temat algorytmu podczas wykonywania treningu.

Opis działania aplikacji webowej

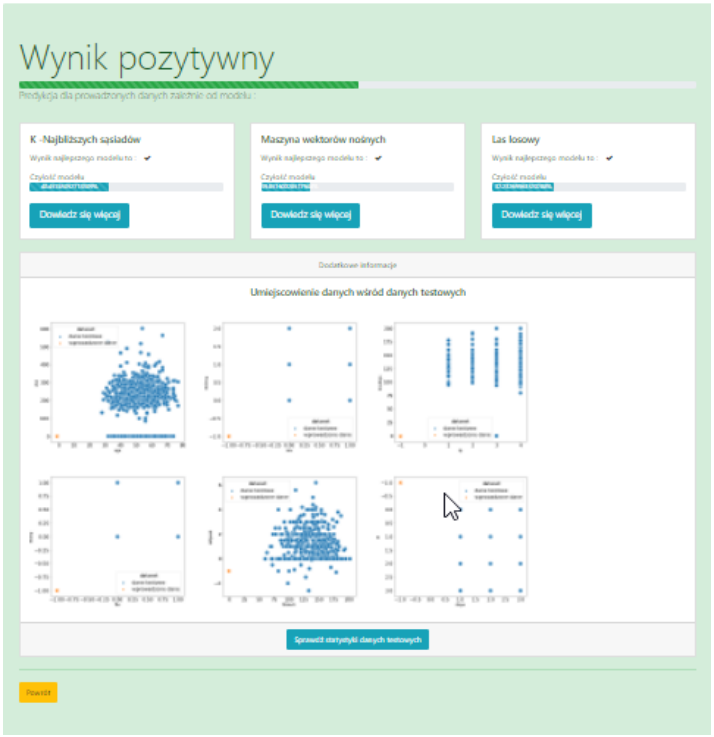
Poniżej przedstawiono architektówe działania:



Aplikacja posiada 4 widoki :

- widok głowny strony
- widok prezentacji danych wejściowych
- widok omówienia treningu algorytmów
- widok formularza pozwalającego na wykonanie predycji na wyuczonych modelach na podstawie własnych danych wejściowych

Zatwierdzenie formularza wyzwala odczytanie zapisanych modeli , iteracje i wykonanie predykcji na każdym z nich , następnie prezentowane są wyniki dla najlepszych estymatorów oraz wykresy wskazujące na umiejscowienie nowych danych na tle zbior testowego.



Porównanie działania modeli

Wstęp

Chodzi o to, że domyślna metoda oceny k-NN to średnia dokładność. Dokładność ma pewne fundamentalne ograniczenia i niebawem przejdziemy do tematu precision, recall, 4c2836bbe011706b10838f45bf5d766f 6.2. Więcej niż dokładność — wskaźniki dla klasyfikacji 187 roc_auc i f1 z rozległej listy powyżej. Po co omawiać te wskaźniki i co jest nie tak z dokładnością? Czy nie jest dokładna? Cieszę się, że pytasz. Przejdźmy od razu do odpowiedzi na Twoje pytania. Oto szybki przykład tego, jaki jest problem z dokładnością. Pamiętaj: dokładność polega w zasadzie na zliczeniu, ile razy odpowiedź jest poprawna. Wyobraźmy sobie zestaw danych, wśród których mamy 100 pacjentów i rzadką chorobę. Cieszymy się, że choroba jest rzadka, niemniej jest bardzo niebezpieczna. W naszym zestawie danych mamy 98 zdrowych pacjentów i 2 chore osoby. Weźmy prostą strategię bazową przepowiadającą, że wszyscy są zdrowi. Nasza dokładność wynosi 98%. To świetnie, prawda? Cóż, nie do końca. Znaleźliśmy dokładnie zero spośród ludzi, których mieliśmy znaleźć — tych, którzy wymagają leczenia. To poważny problem. Gdybyśmy mieli bardziej złożony system uczący się, który w taki sposób nie trafia z wynikami, bylibyśmy bardzo z niego niezadowoleni. W tym podrozdziale zamieszczone zostały wyniki oraz wykresy wygenerowane podczas treningu i weryfikacji danych testowych.

Dokładność w %								
Parametryzacja		parametry domyślne			wyznaczanie parametrow			
Algorytm / Imputer	średnia	mediana	stała	naj. wartość	średnia	mediana	stała	naj. wartość
Losowe lasy decyzyjne	100	100	100	100	100	100	100	100

Dokładność w %								
Maszyna wektorów nośnych	100	100	100	100	100	100	100	100
K-najbliższych sąsiadów	100	100	100	100	100	100	100	100

•

Możemy zadawać pytania i uzyskiwać odpowiedzi za pomocą macierzy błędu. Na przykład, jeżeli jesteśmy lekarzami, interesuje nas, jak dobrze jesteśmy w stanie identyfikować ludzi, którzy rzeczywiście są chorzy. Skoro zdefiniowaliśmy chorobę jako nasz wynik pozytywny, są to nasze osoby z pierwszego rzędu. Pytamy, jak dobrze nam poszło dla RzeczP: ile spośród rzeczywiście chorych osób wykryliśmy poprawnie: RzeczP TP TP FN TP . Określamy to terminem czułość. Można to określić następująco: „Jak dobrze test jest dopasowany do znajdowania chorych ludzi" — nacisk kładziony jest na chorych ludzi. Jest to też nazywane skutecznością wyszukiwania przez ludzi zawodowo zajmujących się wyszukiwaniem informacji. Różne społeczności wymyśliły tę koncepcję niezależnie, dlatego nadały jej inne nazwy. Aby zrozumieć skuteczność wyszukiwania, pomyśl o ruchu wyszukiwarki internetowej. Spośród interesującego nas ruchu, RzeczP, ile zapytań odnaleźliśmy lub wyszukaliśmy poprawnie? Jeszcze raz dla czułości i skuteczności wyszukiwania interesuje nas poprawność w przypadku rzeczywistych pozytywnych lub interesujących nas przykładów. Czułość ma jeszcze jeden powszechnie używany synonim: współczynnik poprawnie pozytywnych (TPR — ang. true positive rate). Nie szukaj ołówka, aby to zapisać; za moment przedstawię Ci tabelę terminów. TPR jest współczynnikiem przypadków poprawnie pozytywnych w odniesieniu do rzeczywistości. Jest jeszcze kwestia związana z chorymi osobami, które odnaleźliśmy poprawnie: chore osoby, które oceniliśmy błędnie. Taki błąd nazywany jest błędnym negatywnym (ang. false negative — FN). Zgodnie ze wzorem chorzy ludzie, w przypadku których się pomyliliśmy, to TP FN FN . Możemy dodać chorych ludzi, których oceniliśmy poprawnie, i chorych ludzi, których nie odnaleźliśmy, aby uzyskać liczbę wszystkich chorych ludzi. Matematycznie wygląda to tak: 1 100% TP FN TP FN TP FN FN TP FN TP . Te równania mówią, że mogę rozbić wszystkich chorych ludzi na (1) chorych, o których myślę, że są chorzy, i (2) chorych, o których myślę, że są zdrowi. Możemy również zapytać: „Jak dobrze idzie nam ze zdrowymi ludźmi?". Nasza uwaga przenosi się teraz do drugiego rzędu RzeczN. Jeżeli jesteśmy lekarzami, chcemy wiedzieć, jaka jest wartość naszego testu, kiedy ludzie są zdrowi. O ile ryzyko jest inne, wciąż występuje ryzyko błędnej diagnozy zdrowej osoby. My — bawiący się przez chwilę w lekarzy — nie chcemy mówić ludziom, że są chorzy, kiedy są zdrowi. Nie tylko straszymy ich i dajemy powód do zmartwienia, ale możemy przepisać im zabiegi i lekarstwa, których nie potrzebują! Taki przypadek nazywany jest błędnym pozytywnym (ang. false positive — FP). Możemy go ocenić, patrząc na to, jak dobrze poszło nam ze zdrowymi osobami: RzeczN TN FP TN TN .Terminem diagnostycznym jest tutaj specyficzność testu: czy test zgłasza alert tylko w specyficznych 4c2836bbe011706b10838f45bf5d766f 190 Rozdział 6. Ocena klasyfikatorów przypadkach, o które nam chodzi. Specyficzność jest także nazywana współczynnikiem poprawnie negatywnych (ang. true negative rate — TNR). W istocie jest to częstotliwość występowania przypadków prawdziwie negatywnych w odniesieniu do rzeczywistości. Ostatnia kombinacja macierzy błędu bierze predykcje jako część główną, a rzeczywistość jako drugorzędną. Ja widzę to jako odpowiedź na pytanie: „Jaka jest wartość naszego testu, kiedy wynik jest pozytywny?" lub bardziej zwięźle: „Jaka jest wartość trafienia?". Cóż, trafienia to PredP. W ramach PredP obliczymy liczbę poprawnych: TP. Wtedy otrzymamy TP FP TP PredP TP , nazywane dokładnością. Spróbuj dziesięć razy szybko powiedzieć: „Jak dokładne są nasze predykcje pozytywne?". Rysunek 6.1 przedstawia macierz błędu wraz ze wskaźnikami jej oceny. Rysunek 6.1. Macierz błędu poprawnych i niepoprawnych predykcji Jeszcze jeden komentarz odnośnie do macierzy błędu. Podczas czytania o niej możesz zauważyć, że niektórzy autorzy zamieniają osie: ustawiają rzeczywistość w kolumnach, a predykcje w wierszach. Wtedy TP i TN będą tymi samymi komórkami,

ale FP i FN będą zamienione. Nieświadomy czytelnik może być bardzo, bardzo skonsternowany. Uznaj się za ostrzeżonego i bądź uważny, czytając inne omówienia macierzy błędu.[3]

Zestawienie efektywności działania algorytmów

Konfrontacja technik ucznia maszynowego zaleznie od zestawu danch będzie dawała odmienne wyniki ze względu na ich predyspozycje do zajmowania się odpowiednimi zbiorami danych.

Potencjał algorytmów dla niewielkiego kompletu danych zawierającego wartości

Zczynając od drzew decyzyjnych,można od razu stwierdzić ich niski potencjał. Istnieje zbyt duże prawdopodobieństwo dopasowania się do modelu treningowego, gdyż wspomniany zbiór dancyh wejściowych nie jest wystarczająco liczny. Dlatego w pracy omówione zostały lasy decyzyjne.

Większej dokładności można się spodziewać po metodzie wektorów nośnych, ale jego złożoność czasowa oraz pamięciowa mogą zaniżyc jego ogólną klasyfikację.

K-najbliższego sąsiada może być przydatny w przypadku danych nieliniowych oraz łatwo wykorzystany w problemach regresji. Wartość wyjściowa obiektu jest obliczana przez średnią k wartości najbliższych sąsiadów. Niestety tak samo jak w przypadku maszyny wektorów nośnych jest wolniejsza i bardziej kosztowna pod względem czasu i pamięci. Wymaga dużej pamięci do przechowywania całego zestawu danych treningowych do przewidywania oraz nie nadaje się również do dużych danych wymiarowych.

Wskaźniki wydajności

Określenie stopnia, w jakim skonstruowany model z powodzeniem realizuje wyznaczone zadanie należy do wskaźnika wydajności. Przykładem nieprawidłowego wyboru może być próba przewidzenia wystąpienia rzadkiej choroby u pacjenta i określenie głownym miernikiem *dokładność*. W takim scenariuszu klasyfikacja wszystkich pacjentów jako zdrowych , daje niewiele odbiegającą od perfekcji dokładność, a jednocześnie błędnie osądzać każde wystąpienie choroby.

Losowe lasy decyzyjne

##todo liczenie błędów macież pomysłek

Losowe lasy decyzyjne

###OCENA PODELI ORAZ UŻYTYCH PARAMETRÓW -OCENA SZYBKOŚCI WYKONANIA -OCENA ZALEŻNIE OD UZUPELNIANIA DANYCH -OCENA ZALEŻNIE OD DOBRANEJ PARAMERYZACJI : - które parametry mają i wpływ i dlaczego: - ZALEŻNIE OD METRYKI(SHORT OPIS METRYK)

Maszyna wektorów nośnych

###OCENA PODELI ORAZ UŻYTYCH PARAMETRÓW -OCENA SZYBKOŚCI WYKONANIA -OCENA ZALEŻNIE OD UZUPELNIANIA DANYCH -OCENA ZALEŻNIE OD DOBRANEJ PARAMERYZACJI : - które parametry mają i wpływ i dlaczego: - ZALEŻNIE OD METRYKI(SHORT OPIS METRYK)

K-najbliższych sąsiadów

###OCENA PODELI ORAZ UŻYTYCH PARAMETRÓW -OCENA SZYBKOŚCI WYKONANIA -OCENA ZALEŻNIE OD UZUPELNIANIA DANYCH -OCENA ZALEŻNIE OD DOBRANEJ PARAMERYZACJI : - które parametry mają i wpływ i dlaczego: - ZALEŻNIE OD METRYKI(SHORT OPIS METRYK)

Porównianie całościowe algorytmów : złożoność czasowa , dokładność , złożoność implementacyjna , wpływ danych wykorzytywanych w modelu

Wybrane najlepsze modele klasyfikacji:

[CV 5/15] END C=0.1, cache_size=200, coef0=0.0, degree=1, gamma=scale, kernel=linear, shrinking=False;, score=0.982 total time= 0.0s [CV 4/15] END C=0.1, cache_size=200, coef0=0.0, degree=1, gamma=scale, kernel=poly, shrinking=True;, score=0.982 total time= 0.0s [CV 4/15] END C=0.1, cache_size=200, coef0=0.0, degree=1, gamma=scale, kernel=poly, shrinking=False;, score=0.982 total time= 0.0s [CV 4/15] END C=10, cache_size=500, coef0=0.3, degree=5, gamma=scale, kernel=linear, shrinking=True;, score=0.982 total time= 0.0s [CV 4/15] END C=10, cache_size=200, coef0=0.3, degree=5, gamma=auto, kernel=rbf, shrinking=True;, score=0.982 total time= 0.0s

Wybrane najgorsze modele klasyfikacji :

[CV 10/15] END C=100, cache_size=200, coef0=0.3, degree=5, gamma=scale, kernel=poly, shrinking=True;, score=0.800 total time= 0.0s [CV 10/15] END C=100, cache_size=500, coef0=0.3, degree=5, gamma=scale, kernel=poly, shrinking=False;, score=0.800 total time= 0.0s [CV 10/15] END C=100, cache_size=200, coef0=0.3, degree=5, gamma=scale, kernel=poly, shrinking=False;, score=0.800 total time= 0.0s

Najlepsze modele i wartości dla regresji :

Najgorsze modele i wartości dla regresji :

Maszyna wektorów nośnych

K-najbliższych sąsiadów

###OCENA PODELI ORAZ UŻYTYCH PARAMETRÓW

problem multiklasyfikacji - problem regresji kategrycznej - zwykła regresja , mierzyć będe metoda prównania - tzrea było wprowadzić reguły do float na int -> inne metody do liczenia błędów na dzień dobry widzimy nie dokładność ze wględu na klasyfiakcję po przecinku regresja kategoryczna -> rzutowanie przedziału wartości na wartość graniczną

This question is a little wrong-worded. You cannot get worse after optimization, otherwise it wouldn't be optimization! (At worst you are at the same performance like before, getting the exact same parameters you already had)

As Grzegorz points out in a comment, first of all your parameter list isn't complete and doesn't contain the values you use later. For example the learning rate, but also max_depth. Secondly, a grid search where you don't really know where to look should contain a much larger variance for the parameters. You check [0.1, 0.01, 0.05] for the learning rate, but did you check [0.0001, 0.001, 1.]? The learning rate might be a bad example here but I hope it gets the point across, you might want to check magnitude/scale first, e.g. powers of ten, before checking small variations.

Depending on your dataset, the difference between runs with the same values might also come from different seeds! Check that you either always set the same seed, or try it enough times with different seeds to get a comparable answer (for example with KFold).

Is your model even converging for every training? Where do you make sure that you train long enough? You can plot the loss for the training and test sample and check if it's converging or not. This can be controlled with n_estimators in xgboost I believe.

There is nothing wrong in your code or process. Often times in machine learning performance on the test dataset is lower than than performance on the training data set. Your model is not generalizing perfectly to the data it has not seen before (i.e., the test dataset).

-2

When you do hyper-parameter tuning, you improve the regularization of the model. Before you did optimization, you could be overfitting. After optimization you regularized your model and now it performs just right.

enter image description here

Then your model score will be worse after optimization on training set. Your model score will also be bad for test set if your model heavily relies on one feature for classification.

You can use learning curve to see how the curve changes when you use optimization vs when you don't. And you can use df.corr() to see the correlation matrix for the correlation between feature values and target values. ++++++++++++++ Spis ilustracji{.unnumbered} ======== [^schemat_wzorowany]:Na podstwie materiałów opublikowanych na https://www.datacamp.com

Spis tabel

Bibliografia

[1] A. L. Fradkov, "Early history of machine learning," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 1385–1390, 2020, doi: https://doi.org/10.1016/j.ifacol.2020.12.1888.

[2] 2. Jesper E. van Engelen1 · Holger H. Hoos1, *A survey on semi-supervised learning.*

[3] M. E. FENNER, *Machine learning with python for everyone.*

[4] W. Modrzejewski and W. J. Musiał, "Stare i nowe i czynniki ryzyka sercowo-naczyniowego - jak zahamować epidemię miażdżycy? Część i. Klasyczne czynniki ryzyka," *Forum Zaburzeń Metabolicznych*, vol. 1(2), pp. 106–114, 2010.

[5] "Międzynarodowa sieć firm audytorsko-doradczych ze szczególnym uwzględnieniem branży dóbr konsumpcyjnych, usług finansowych, nieruchomości i budownictwa, technologii informacyjnych, mediów i komunikacji (TMT), transportowej (TSL), produkcji przemysłowej, a także sektora publicznego."

[6] K. Karol, *Uczenie maszynowe w opiece zdrowotnej.* Roczniki Kolegium Analiz Ekonomicznych/Szkoła Główna Handlowa 56, 2019, pp. 305–316.

[7] V. Nasteski, "An overview of the supervised machine learning methods."

[8] J. Grus, *Data science from scratch:first principles with python.* pp. r11 pp140.

[9] T. Łysiak, *The use of machine learning methods in predicting stock prices on the stock exchange.*

[10] J. Laska, "An overview of machine learning methods used in sentiment analysis."

[11] H. van Engelen J. E., "A survey on semi-supervised learning," *Technologie informatyczne w administracji publicznej i służbie zdrowia*, pp. 373–440, 2020.

[12] G. ;. G. Pedregosa F.; Varoquaux, *Scikit-learn: Machine learning in PythonScikit-learn: Machine learning in python.* 2011, pp. 2825–2830.

[13] *J. P. M. and J. S. Bojanowski, "Comparison of the novel probabilistic self-optimizing vectorized earth observation retrieval classifier with common machine learning algorithms."

[14] D. Dua and C. Graff, "UCI machine learning repository [http://archive.ics.uci.edu/ml]," *Machine Learning Technical Reports*, vol. 1(1), pp. 1–6, 2019.

[15] dr n. med. Robert Detrano,.

[16] M. Andras Janosi,.

[17] W. S. M. M. P. MD,.

[18] R. H. F. Peshawa J. Muhammad Ali, "Data normalization and standardization: A technical report," *Machine Learning Technical Reports*, vol. 1(1), pp. 1–6, 2014.

[19] D. Kumar, "Introduction to data preprocessing in machine learning beginners guide for data preprocessing."

[20] A. G. D. Anguita L. Ghelardoni, "The 'k' in k-fold cross validation," *Journal of Machine Learning Research*.

[21] G. Bonaccorso, *Mastering machine learning algorithms: Expert techniques to implement popular machine learning algorithms and fine-tune your models*.

[22] K. Matthew, *Thoughtful machine learning with python a test-driven approach*. pp. r.1 pp8.

[23] T. D. Breiman L., *Random forests, machine learning*. StatSoft Polska Sp. z o. o, 2001.

[24] S. HUANG, N. CAI, P. P. PACHECO, S. NARRANDES, Y. WANG, and W. XU, "Applications of support vector machine (SVM) learning in cancer genomics," *Cancer Genomics & Proteomics*, vol. 15, no. 1, pp. 41–51, 2018, Available: https://cgp.iiarjournals.org/content/15/1/41

[25] A. Pielowska, "Maszyna wektorów nośnych."

[26] H. S. R. Sudip Karki, "Comparison of a*, euclidean and manhattan distance using influence map in ms. Pac-man," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[27] C. B. Binbin Lua Martin Charltonb and P. Harrisc, "The minkowski approach for choosing the distance metric in geographically weighted regression."

[28] V. S. W. Samrudhi Rajendra Kaware, *Podejście porównawcze do algorytmów uczenia się maszynowego*. Reading, Massachusetts: Addison-Wesley, 1993.

[29] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[30] M. Grinberg, *Flask web development: Developing web applications with python*. " O'Reilly Media, Inc.", 2018.