Metody Sztucznej Inteligencji 2 – projekt

Aplikacja klasyfikująca do jednej z czterech klas wielkości smogu na podstawie danych pogodowych

Autorzy: Bartosz Woźniak, Krzysztof Małaśnicki

Spis treści

[Spis treści 1](#_Toc494615118)

[Dane pogodowe 3](#_Toc494615119)

[Dane smogowe 4](#_Toc494615120)

[Dane 4](#_Toc494615121)

[Aplikacja 5](#_Toc494615122)

[Baza danych 7](#_Toc494615123)

[kNN 9](#_Toc494615124)

[Uczenie 9](#_Toc494615125)

[Testowanie: 9](#_Toc494615126)

[kNN Wyniki 10](#_Toc494615127)

[Las Losowy 11](#_Toc494615128)

[Inicjalizacja danych 11](#_Toc494615129)

[Budowanie lasu losowego 11](#_Toc494615130)

[Klasyfikacja 14](#_Toc494615131)

[Las Losowy – Wyniki 15](#_Toc494615132)

[Etap III 22](#_Toc494615133)

[XGBoost 22](#_Toc494615134)

[Przygotowanie danych 22](#_Toc494615135)

[Dane 22](#_Toc494615136)

[Wybór metryki w 3 etapie 23](#_Toc494615137)

[Wyniki i obserwacje 23](#_Toc494615138)

[Podział pracy 29](#_Toc494615139)

# Dane pogodowe

Dane związane z pogodą zostały pobrane ze strony internetowej: <https://dane.imgw.pl/>. Pobrano dane za okres od 01 stycznia 2015 roku do 28 lutego 2015 roku. Wybrano właśnie ten okres czasu, ponieważ 2015 rok jest ostatnim rokiem, z którego udostępnione są dane smogowe w wygodnym do parsowania formacie. A pierwsze dwa miesiące 2015 roku okazały się wystarczającą ilością danych. Dane pochodzą ze stacji meteorologicznej Kraków Balice. Stacja z Krakowa została wybrana ze względu na fakt ogólnej opinii występowania smogu w Krakowie. Dane te zostały pobrane przy użyciu własnoręcznie napisanego joba (aplikacji konsolowej).

Dane pogodowe były pobierane za pomocą API w formacie csv, a następnie zapisywane do bazy danych SQL (MS SQL Server). Aplikacja działała bardzo długo (kilka dni) ze względu na fakt, iż dla każdego dnia i dla każdej cechy konieczne było osobne zapytanie. Dodatkowo, pomiędzy poszczególnym zapytaniami konieczna była krótka przerwy, gdyż w przeciwnym wypadku pojawiał się błąd 429 (Too Many Requests). Pobrano dane z 59 dni, dla każdego z tych dni próbowano pobrać dane dla 782 cech, które teoretycznie dostępne są w bazie IMGW. Różne cechy zwracały różne odpowiedzi. Niektóre były mierzone 1 raz dziennie (1 rekord), a inne raz na minutę (1440 rekordów). Wszystkie te dane, bez obrabiania, zapisano do bazy danych, konwertując jedynie napisy na liczby. Dało to 2.5 mln rekordów w tabeli. Jednak tylko 218 unikalnych cech zwróciło jakiekolwiek dane. Pozostałe były zawsze puste.

Następnie dane zostały przetworzone. Przeanalizowano częstotliwość pomiarów każdej z cech, ręcznie analizując dane. Wyodrębniono następujące grupy częstotliwości mierzenia danych: raz na minutę, raz na godzinę, raz na 6 godzin, raz na 12 godzin, raz na dobę, raz na 10 dni, raz na miesiąc. Każdą z cech przypisano do jednej z wyżej wymienionych grup. Należy zaznaczyć, iż podstawą do późniejszych analiz będą pomiary raz na godzinę, dlatego też każdą z cech ustandaryzowano do takiego formatu. Z danych mierzonych raz na minutę przepisano te o pełnych godzinach. Dane mierzone raz na godzinę zostały przepisane bez zmian. Natomiast pozostałe dane zostały przepisane kilkukrotnie, aby zachować jednolity podział danych, np. dane mierzone raz na 6 godzin, były mierzone o 0, 6, 12, 18, dlatego też w godzinach 1 – 5 przepisano dane z godziny 0 itd. Wstawiono również rekordy z pustymi wartościami dla braków danych, których jest bardzo wiele w pobranych danych. Zignorowano również cechy mierzone raz na rok (dla wszystkich rekordów byłyby one jednakowe). Po przetworzeniu pozostało 179 unikalnych cech (z licznymi brakami danych). Po przetworzeniu w tabeli znalazło się 250 tys. rekordów (59 dni \* 24 h \* 179 cech).

Ostatnim etapem była normalizacja danych pogodowych – przeskalowanie wartości każdej z cech do przedziału [0, 1]. Dla każdej z wartości zastosowano następujący wzór:

(wartość – min(cecha)) / (max (cecha) – min(cecha))

Po normalizacji usunięto kolejnych 35 cech, które zawsze przyjmowały jedną i tę samą wartość. Dane pogodowe w takiej postaci były gotowe do zastosowania algorytmu kNN.

# Dane smogowe

Dane związane ze smogiem zostały pobrane ze strony internetowej: <http://powietrze.gios.gov.pl/pjp/archives>. Dane z każdego roku były dostępne w wygodnym formacie w postaci pliku Excel za cały rok dla każdego ze szkodliwych pyłów osobno, pomiary godzinowe lub dzienne. Najpopularniejsze obecnie są pyły PM2.5 i PM10. Zdecydowano się wykorzystać pomiary godzinowe pyłu PM2.5. Zsynchronizowano dane wyłącznie ze stycznia i lutego 2015 roku ze względu na fakt posiadania odpowiadających tym terminom danych pogodowych. W pliku znajdowały się dane dla wielu stacji pomiarowych z całej Polski. Zsynchronizowano wyłącznie dane z jednej stacji pomiarowej z Krakowa znajdującej się najbliżej stacji meteorologicznej Kraków Balice. Po tym procesie w bazie danych znajdowały się dane godzinowe za okres od 01 stycznia 2015 roku do 28 lutego 2015 mierzone dla pyłu PM2.5 w stacji pomiarowej w Krakowie (nieopodal Balic). W tabeli uzyskano 1.4 tys. rekordów. Dane zostały zsynchronizowane i podzielone na klasy przy użyciu własnoręcznie napisanej aplikacji konsolowej. Pojedyncze braki danych (kilka w całym pliku) zastąpiono wartościami średnimi z godziny wcześniejszej i późniejszej.

Następnie dane należało podzielić na 4 klasy. Na wyżej wymienionej stronie wyróżniono następujące klasy:

|  |  |
| --- | --- |
| Jakość powietrza | Wartość |
| Bardzo dobra | 000 – 012 |
| Dobra | 013 – 036 |
| Umiarkowana | 037 – 060 |
| Dostateczna | 061 – 084 |
| Zła | 085 – 120 |
| Bardzo zła | 121 – |

Analizując powyższą tabelę oraz starając się zredukować liczbę klas do 4, na potrzeby projektu wprowadzono i podzielono dane według następującego schematu:

|  |  |
| --- | --- |
| Jakość powietrza | Wartość |
| Dobra | 000 – 032 |
| Umiarkowana | 033 – 049 |
| Dostateczna | 050 – 079 |
| Zła | 080 – |

# Dane

Kolejnym krokiem było podzielenie danych na zbiór treningowy i zbiór testowy. Do zbioru testowego zaliczono dane od 8 do 14 stycznia oraz od 8 do 14 lutego. Pozostałe dane trafiły do zbioru treningowego. Podział ten został dokonany na zasadzie losowej. Dało to 1080 rekordów w zbiorze treningowym i 360 rekordów w zbiorze testowym. Podział na klasy w miarę równomierny. Dane smogu i pogody znajdują się w osobnych tabelach, łatwo je połączyć, ponieważ każdy obiekt charakteryzowany jest przez unikalną datę (traktowaną jako ID). W skład obiektu wchodzi wektor cech oraz klasa smogowa.

# Aplikacja

Celem realizacji projektu przygotowano aplikację w C# 7.0 z wykorzystaniem IDE Visual Studio Enterprise 2017. Wykorzystano następujące nugety:

* Entity Framework – Object Relational Mapping, mapowanie encji z bazy danych do klas
* Entity Framework Bulk Insert – zapewnia szybkie wstawianie do bazy danych listy rekordów
* Log4net – logowanie do pliku podczas działania programu
* Autofac – Dependency Injection, wstrzykiwanie zależności
* Newtson Json – parsowanie plików json

Stworzono 3 projekty:

* SmogDetector.Task.App – generyczna aplikacja konsolowa wywoływana z różnymi parametrami. Każda komenda jest odpowiedzialna za wykonanie innego zadania. Każda komenda to osobna klasa wywołująca odpowiednie metody z projektu Business. Komenda musi implementować odpowiedni interfejs. Wstrzykiwane są za pomocą Autofaca.
* SmogDetector.Task.Business – projekt odpowiedzialny za logikę biznesową aplikacji, przetwarzanie danych. To tutaj zaimplementowane są wszystkie algorytmy, tutaj przetwarzane są dane. Każdemu zadaniu odpowiada osobna klasa oraz interfejs.
* SmogDetector.DataAccess – ORM, projekt odpowiedzialny za dostęp do bazy danych (Entity Framework), mapowanie relacyjno – obiektowe.

Dostępne są następujące komendy:

* Help – wyświetla listę dostępnych komend z opisem
* Import-imgw-classifications – importuje z pliku json (ze strony IMGW) wszystkie cechy do bazy danych (nazwę cechy, jednostkę, kod cechy)
* Import-imgw-stations – importuje z pliku json (ze strony IMGW) kod i nazwę stacji Kraków Balice
* Sync-imgw-data – synchronizuje dane pogodowe dla wszystkich stacji i wszystkich cech z bazy danych za okres czasu od 01 stycznia do 28 lutego 2015 roku
* Compute-classifictaions – przetwarza dane związane z cechami licząc ilość wystąpień każdej z cech, oraz liczbę jej wystąpień na godzinę (potrzebne w ręcznej analizie danych przed przetworzeniem danych pogodowych)
* Process-weather-data – ujednolica dane pogodowe, aby były to dane godzinowe, dopisuje dodatkowe rekordy i usuwa zbędne, tworzy puste rekordy dla braków danych
* Normalize-weather-data – normalizuje dane pogodowe
* Import-smog-data – importuje dane smogowe z pliku
* Process-smog-data – poprawia braki danych, dodaje podział na klasy
* Compute-knn-distances – oblicza odległości każdego obiektu ze zbiory testowego do każdego obiektu ze zbioru treningowego
* Find-knn-classes – dopasowuje każdy z obiektów ze zbioru testowego do danej klasy, zależnie od parametru algorytmu k
* Run-random-forest – buduje las losowy ze wskazaną liczbą drzew (domyślnie 100), testuje jego jakość na zbiorze oob, wyniki zapisuje do bazy danych oraz klasyfikuje obiekty ze zbioru testowego zapisując wyniki do bazy danych
* Test-random-forest – uruchamia opisaną powyżej metodę dla różnej liczby drzew: 1, 5, 10, 20, 50, 100, 200, 500, 1 000

# Baza danych

W bazie danych znajdują się następujące tabele:

* ImgwClassifications – atrybuty pogody, zsynchronizowane bezpośrednio ze strony IMGW, nazwa Classifcations została wzięta bezpośrednio ze strony IMGW
  + Id – unikalne ID w bazie danych
  + Code – kod atrybutu ze strony IMGW
  + Name – nazwa atrybutu, np. temperatura
  + Unit – jednostka pomiaru atrybutu, np. m/s
  + HoursCounter – liczność wystąpienia atrybutu na godzinę, gdy jego pomiar nie jest pusty, przydatne przy analizie danych, występują atrybuty mierzone np. raz na minutę lub raz na godzinę
  + Counter – liczność wystąpienia atrybutu, gdy jego pomiar nie jest pusty, przydatne przy analizie danych
  + PerHourCounter – liczność wystąpienia atrybutu w ciągu godziny uśredniona, gdy jego pomiar nie jest pusty, przydatne przy analizie danych
* ImgwStations – stacje meteorologiczne (konkretnie jedna z Balic)
  + Id – unikalne ID w bazie danych
  + Code – kod stacji ze strony IMGW
  + Name – nazwa stacji
* ImgwWeatherData – dane pogodowe zsynchronizowane bezpośrednio ze strony IMGW
  + Id – unikalne ID w bazie danych
  + Date – data z dokładnością, co do minuty
  + Value – wartość liczbowa
  + Status – status, wartość zawsze 0 dla poprawnych danych, pole ze strony IMGW
  + ClassificationId – Id klasyfikacji, czyli atrybutu
  + StationId – Id Stacji
  + ValueDate – wartość, data, nieliczne wartości są datami, a nie liczbami
  + CbdoCbdh – flaga, czy dane pochodzą z bazy Cbdo, czy Cbdh, są to bazy ze strony IMGW, wszystkie zsynchronizowane dane pochdzą ostatecznie z bazy Cbdh
* ProcessedWeatherData – przetworzone dane pogodowe, ustandaryzowane do czętotliwości pomiarów raz na godzinę, usunięcie dat (których były 2, prawie zawsze puste), uzupełnienie brakujących wartości rekordami z wartością null
  + Id – unikalne ID w bazie danych
  + Date – data z dokładnością, co do godziny
  + Classification – ID atrybutu pogody
  + Value – wartość dla tej daty i tego atrybutu
* NormalizedWeatherData – dane przepisane z tabeli powyżej, poddane normalizacji
  + Id – unikalne ID w bazie danych
  + Date – data z dokładnością, co do godziny
  + Classification – ID atrybutu pogody
  + Value – wartość dla tej daty i tego atrybutu
* SmogData – dane związane ze smogiem, wczytane z pliku
  + Id – unikalne ID w bazie danych
  + Value – wartość pomiaru
  + Type – typ szkodliwego pyłu, np. PM10
  + Station – stacja, w której dokonano pomiaru, np. Kraków Balice
  + Date – data pomiaru z dokładnością, co do godziny
* ProcessedSmogData – przetworzone dane smogowe z podziałem na klasy, zostawiono dane tylko dla pyłu PM2.5 i stacji najbliższe stacji meteorologicznej Kraków Balice
  + Id – unikalne ID w bazie danych
  + Date – data pomiaru z dokładnością, co do godziny
  + Value – wartość pomiaru
  + Class – klasa jakości powietrza, na podstawie wartości, klasy opisane powyżej w rozdziale dane
* KnnDistancesData – tabela zawierająca odległości pomiędzy wszystkimi elementami ze zbioru treningowego do wszystkich elementów ze zbioru testowego
  + Id – unikalne ID w bazie danych
  + TrainingSet – data elementu ze zbioru treningowego
  + TestingSet – data elementu ze zbioru testowego
  + Distance – odległość pomiędzy elementami
* KnnResultsData – tabela zawierająca wyniki uruchomienia KNN (dla k = 1, 5, 15) dla wszystkich elementów ze zbioru testowego, umożliwia późniejszą analizę danych
  + Id – unikalne ID w bazie danych
  + TestingSet – data elementu ze zbioru testowego
  + KnnClass1 – klasa przewidziana przez kNN dla k = 1
  + KnnClass5 – klasa przewidziana przez kNN dla k = 5
  + KnnClass15 – klasa przewidziana przez kNN dla k = 15
  + CorrectClass – prawdziwa klasa
  + Description – miejsce na komentarz
* RandomForestResults – tabela zawierająca wyniki uruchomienia Random Forest dla wszystkich elementów ze zbioru testowego, umożliwia późniejszą analizę danych
  + Id – unikalne ID w bazie danych
  + TestngSet – dana elementu ze zbioru testowego
  + CorrectClass – prawdziwa klasa
  + PrdictedClass – przewidziana klasa przez las losowy
  + Description – opis parametrów wywołania lasu losowego, np. ilość drzew
* RandomForestStats – tabela zawierająca statystyki, informacje o błędzie oob, dla wskazanej liczby drzew i innych parametrów
  + Id – unikalne ID w bazie danych
  + Trees – liczba drzew w lesie
  + Oob – błąd oob, liczba przypadków błędnie sklasyfikowanych do wszystkich przypadków
  + Description – opis parametrów wywołania lasu losowego, np. jak radzić sobie z wartościami ciągłymi

Do uruchomienia kNN lub Random Forest potrzebne są dane z dwóch tabel: NormalizedWeatherData – dane związane z pogodą oraz ProcessedSmogData – dane związane ze smogiem. Pozostałe tabele, to tabele pomocnicze, używane przy przetwarzaniu danych lub przechowujące wyniki klasyfikacji.

# kNN

## Uczenie

Zostało zaimplementowano na poziomie przetwarzania i normalizacji danych.

1. Dokonaj normalizacji danych.
2. Zapamiętaj cały zbiór treningowy (zapisany w bazie danych SQL).

## Testowanie:

1. Przetwarzanie, normalizacja danych:
   1. Dokonaj normalizacji danych testowych
2. Metoda Compute kNN Distances:
   1. Policz odległości pomiędzy wszystkimi wektorami testowymi a wszystkimi wektorami zbioru treningowego
3. Metoda Find kNN Classes:
   1. Dla każdego wektora testowego wykonaj:
      1. Posortuj odległości od tego wektora do wszystkich wektorów ze zbioru treningowego od największej do najmniejszej
      2. Zobacz etykiety k – najbliższych wektorów do wektora testowego. Zrób histogram częstości poszczególnych etykiet spośród „k-najbliższych” (Policz ile i których etykiet było spośród k najbliższych)
      3. Przypisz najczęściej występującą etykietę jako etykietę wektora testowego
      4. Jeśli wystąpił impas (dwie klasy miały taką samą liczbę głosów) rozwiąż problem losowo

Na poziomie obliczania odległości pomiędzy poszczególnymi obiektami występowały liczne braki danych. Zdecydowano się zastosować wartość średnią dla danej cechy z obiektów ze zbioru treningowego w takim przypadku. Do obliczenia odległości wykorzystano metrykę euklidesową.

# kNN Wyniki

Skuteczność kNN przetestowano dla k = 1, k = 5 oraz k = 15. Zbiór treningowy miał ponad 1000 rekordów. Zbiór testowy 300. Każdy obiekt zawierał ponad 100 cech. Czas wykonania kNN (policzenia odległości i klasyfikacja) to kilkanaście sekund.

Poniżej przedstawiono, jaki procent wszystkich obiektów został poprawnie sklasyfikowany:

* Dla k = 1, poprawnie sklasyfikowano około 47% przypadków
* Dla k = 5, poprawnie sklasyfikowano około 55% przypadków
* Dla k = 15, poprawnie sklasyfikowano około 53% przypadków

Poniżej przedstawiono, jaki procent przypadków został pomylony o co najwyżej jedną klasę (obiekt sklasyfikowany np. do klasy 2 zamiast 1), czyli błąd klasyfikatora był stosunkowo niewielki, gdyż pomylenie klasy z dobrą i umiarkowaną jakością powietrza to zdecydowanie mniejszy błąd, niż pomylenie dobrej i złej jakości powietrza:

* Dla k = 1, 22% przypadków zostało pomylonych tylko o jedną klasę
* Dla k = 5, 15% przypadków zostało pomylonych tylko o jedną klasę
* Dla k = 15, 16% przypadków zostało pomylonych tylko o jedną klasę

Przeprowadzone testy pokazują, iż kNN uzyskuje trafność dopasowania na poziomie 50%. Dodatkowo, 20% przypadków jest nieznacznie błędnie klasyfikowanych, pomyłka o jedną klasę. Wyniki te są dalekie od idealnego rozwiązania, dającego zadowalającą jakość klasyfikacji. Jednak należy mieć na uwadze, iż dane pogodowe nie muszą mieć jednoznacznego i trywialnego przełożenia na dane smogowe, dlatego też uzyskanie wyższego poziomu klasyfikacji może być trudne. Przypomnijmy, iż do klasyfikacji były brane pod uwagę wyłącznie cechy pogodowe z danej godziny. Kolejnym etapem rozwoju, byłby pomysł przetestowania działania kNN, nie tylko w oparciu o bieżące dane, ale może o dane z ostatniej doby lub ostatnich trzech dni. Istnieje przypuszczenie, że mogłoby to poprawić jakość klasyfikacji. Można by wówczas zastosować średnią wykładniczą dla cech i na tej podstawie obliczać odległości. KNN dla k = 5 i 15 uzyskał podobne rezultaty i były one nieznacznie lepsze, niż dla k = 1.

# Las Losowy

## Inicjalizacja danych

Przygotowanie listy elementów zbioru uczącego, listy elementów zbioru testowego, listy dostępnych atrybutów oraz listy klas.

* Lista klas to lista występujących w zbiorze etykiet klas, w naszych danych jest to 1, 2, 3, 4.
* Lista atrybutów to lista wszystkich id atrybutów występujących w zbiorach, w naszych danych są to liczby naturalne z przedziału 100 – 1000. Dostępne są 143 różne atrybuty.
* W skład zbioru uczącego wchodzą co godzinne pomiary smogu i pogody z następujących przedziałów czasu: 01 – 07.01.2015 oraz 15.01.2015 – 07.02.2015 oraz 15 – 28.02.2015. Lista elementów zbioru uczącego to lista obiektów zawierających następujące pola:
  + data – jednoznacznie identyfikuje obiekt, jest to id
  + klasa – etykieta klasy
  + Słownik z atrybutami, gdzie id atrybutu to klucz, a wartość w słowniku to wartość dla podanego atrybuty dla danego obiektu. Wartość to liczna zmiennoprzecinkowa.
* W skład zbioru testowego wchodzą co godzinne pomiary smogu i pogody z następujących przedziałów czasu: 08 – 14.01.2015 oraz 08 – 14.02.2015. Lista elementów zbioru testowego to lista obiektów zawierających następujące pola:
  + data – jednoznacznie identyfikuje obiekt, jest to id
  + klasa – etykieta klasy
  + przewidziana klasa – klasa wytypowana przez las losowy
  + Słownik z atrybutami, gdzie id atrybutu to klucz, a wartość w słowniku to wartość dla podanego atrybuty dla danego obiektu. Wartość to liczna zmiennoprzecinkowa.

W wartościach atrybutów występują liczne braki danych (dane pogody). W jednym z wariantów przeprowadzonych obliczeń zamiast bezpośrednich danych, obliczono i zastosowano średnią wykładniczą z ostatnich 24 godzin, zarówno dla zbioru treningowego, jak i testowego. W tym przypadku zastąpiono również braki danych, wartościami średnimi.

## Budowanie lasu losowego

1. Dla każdego i od i = 0 do i < liczba drzew  Wybuduj drzewo losowe i dodaj korzeń do listy, gdzie liczba drzew to parametr.
2. Budowa drzewa losowego:
   1. Ze zbioru uczącego wybierz losowo 2/3 elementów, jako elementy uczące, a pozostałe 1/3 elementów dodaj do zbioru oob, na podstawie którego liczony będzie błąd klasyfikacji.
   2. Wylosuj ze zbioru atrybutów następującą liczbę atrybutów: Floor(Sqrt(liczba\_atrybutów)).
   3. Dla wylosowanego zbioru uczącego i dla wylosowanego zbioru atrybutów, zbuduj drzewo algorytmem C4.5.
   4. Oblicz błąd klasyfikacji, na podstawie stworzonego wcześniej zbioru oob. Błąd klasyfikacji to liczba przypadków błędnie sklasyfikowanych do wszystkich przypadków.
   5. Usprawnienie: kroki b, c, d powtórz k razy i wybierz drzewo o najmniejszym błędzie klasyfikacji. Dane testowano dla k = 1 oraz k = 10. Dzięki temu krokowi wyniki się poprawiają, dla k = 10, istnieje małe prawdopodobieństwo stworzenia drzewa o wysokim współczynniku oob, czyli takie, które może źle klasyfikować.
   6. Zwróć korzeń drzewa.
3. C4.5 – rekurencyjnie:
   1. Jeśli w zbiorze treningowym nie ma elementów, zwróć pusty węzeł.
   2. Jeśli wszystkie elementy w zbiorze treningowym należą do tej samej klasy, zwróć węzeł z etykietą tej klasy.
   3. Jeśli zbiór atrybutów jest pusty, zwróć węzeł z etykietą klasy, której elementów jest najwięcej w obecnym zbiorze treningowym.
   4. Podziel obecny zbiór na podzbiory:
      1. Wybierz atrybut dzielący
      2. Podziel zbiór według tego atrybutu
      3. Wywołaj rekurencyjnie algorytm dla każdego z podzbiorów
   5. Zwróć korzeń tak zbudowanego drzewa.
4. Wybór atrybutu, podział węzła, sposób 1 – sprowadzenie wartości ciągłych do dyskretnych:
   1. Wybierz atrybut z maksymalnym zyskiem informacji Gain.
   2. Jeśli nie znaleziono atrybutu dzielącego zbiór na co najmniej dwa podzbiory, zwróć węzeł z etykietą klasy najczęściej występującą w zbiorze.
   3. Jeśli wartości atrybutu, to wartości ciągłe, a nie dyskretne (wartości ciągłe i dyskretne są określone na podstawie ręcznej analizy danych), sprowadź je do 4 dyskretnych wartości:
      1. Uporządkuj wartości w kolejności rosnącej i podziel na 4 równoliczne podzbiory, pierwszy przedział zawiera wartości od minus nieskończoności do k1, drugi od k1 do k2, trzeci od k2 do k3, czwarty większe od k3. Punkty te są łatwo obliczane, tak aby zachować równoliczność podzbiorów (liczba elementów / 4 \* 1 lub \* 2 lub \* 3).
   4. Podziel zbiór według dyskretnych wartości (prawdziwych lub sztucznie stworzonego przedziału). Dla każdej wartości dyskretnej stwórz osobny podzbiór.
   5. Wywołaj rekurencyjnie algorytm dla każdego z podzbiorów z pominięciem wybranego wcześniej atrybutu.
5. Wybór atrybutu, podział węzła, sposób 2 – wartości dyskretne traktuj jak powyżej, wartości ciągłe dziel na dwa podzbiory:
   1. Wybierz atrybut oraz punkt dzielący z maksymalnym zyskiem informacji Gain.
   2. Jeśli nie znaleziono atrybutu dzielącego zbiór na co najmniej dwa podzbiory, zwróć węzeł z etykietą klasy najczęściej występującą w zbiorze.
   3. Podziel zbiór według atrybutu. Jeden podzbiór to wartości mniejsze lub równe od punktu dzielącego, drugi podzbiór to pozostałe wartości.
   4. Wywołaj rekurencyjnie algorytm dla każdego z podzbiorów.
6. Obliczanie Gain dla wartości dyskretnych:

$I(P) = -(p_1* \log(p_1) + p_2* \log(p_2) + \ldots + p_n* \log(p_n))$

$Info(X,T) = \sum_{i = 1}^{n} \frac{\Vert T_i\Vert}{\Vert T\Vert} * Info(T_i)$

* 1. Oblicz I(P).
  2. Dla każdego atrybutu:
     1. Podziel zbiór według tego atrybutu na podzbiory.
     2. Oblicz Info(X, T).
     3. Oblicz Gain jako różnicę dwóch powyższych.
     4. Jeśli Gain > GainMax, uznaj obecny atrybut jako najlepszy.

1. Obliczanie Gain dla wartości ciągłych:
   1. Oblicz I(P).
   2. Dla każdego atrybutu:
      1. Dla każdej wartości atrybutu:
         1. Podziel zbiór według tego atrybutu na podzbiory (wartości mniejsze i większe od danej wartości).
         2. oblicz Info(X, T).
         3. Oblicz Gain jako różnicę dwóch powyższych.
         4. Jeśli Gain > GainMax, uznaj obecny atrybut jako najlepszy oraz zapamiętaj najlepszy punkt podziału.
2. Braki danych:
   1. Są pomijane przy obliczaniu Gain.
   2. Przy podziale zbioru według atrybutu są wrzucane do dostępnych podzbiorów według następującego schematu:
      1. Jeśli mamy dwa podzbiory, w jednym 4 elementy, a w drugim 6 oraz 5 elementów z brakami danych, to 3 losowo wybrane elementy z brakami danych trafią do podzbioru drugiego, a 2 do pierwszego (prawdopodobieństwo według liczności zbiorów).
3. Szczegóły implementacyjne:
   1. Zaimplementowano dwie wersje podziału węzła dla wartości ciągłych:
      1. Stworzenie przedziałów – wartości dyskretnych i traktowanie danych dalej jako wartości dyskretne.
      2. Dzielenie wartości ciągłych na dwa podzbiory według wybranej wartości dzielącej.
   2. Istnieje możliwość zastąpienia wartości atrybutów pogody, średnią wykładniczą za ostanie 24 godziny, przy jednoczesnym zastąpieniu braków danych średnią lub przetwarzanie zwykłych danych, a nie średniej wykładniczej.
   3. Budowę drzewa można powtórzyć k razy dla różnych podzbiorów atrybutów (przeprowadzono testy dla k = 1 i k = 10) i wybrać drzewo o najmniejszym błędzie klasyfikacji oob i tylko to drzewo dodać do lasu i tylko je później używać do klasyfikacji. Powinno to poprawić jakość klasyfikacji, gdyż istnieje mniejsze prawdopodobieństwo wylosowania słabo dzielących atrybutów.
   4. Pozostałe parametry:
      1. Seed = 1487 – wykorzystywany przy losowaniu.
      2. Ilość ostatnich godzin, za które liczone jest średnia wykładnicza (jeśli w ogóle) = 24.
      3. Ilość klas dyskretnych, na które dzielone są wartości ciągłe = 4. Wybrano podział na 4 dyskretne przedziały, ponieważ mamy 4 klasy, w pewnej znalezionej publikacji sugerowano, że ma to sens. Spostrzeżenie: prawdziwe wartości dyskretne w naszym zbiorze danych mają zwykle 10 unikalnych wartości.
      4. Współczynnik, ile drzew trafi do zbioru uczącego, a ile do oob = 2 / 3.
      5. Funkcja, ile atrybutów ze zbioru atrybutów zostanie wylosowanych = Floor(Sqrt(liczba\_atrybutów).
      6. Atrybuty, które przyjmują wartości dyskretne – określone na podstawie ręcznej analizy.
      7. Liczba drzew.

## Klasyfikacja

Klasyfikacja elementów zbioru testowego przez las losowy:

1. Dla każdego elementu zbioru testowego wykonaj:
   1. Dla każdego drzewa z lasu znajdź klasę dla tego elementu.
   2. Wybierz najczęściej występującą klasę w lesie.
   3. W przypadku remisu, wybierz losowo.
2. Klasyfikacja elementu przez drzewo:
   1. Jeśli węzeł jest liściem, zwróć klasę.
   2. W przypadków wartości dyskretnych, wybierz odpowiedni węzeł – dziecko. W przypadku wartości ciągłych, określ, czy wartość jest mniejsza, czy większa i wybierz odpowiedni węzeł – dziecko. Wywołaj rekurencyjnie dla tego węzła.
   3. W przypadku braku danych, wybierz najbardziej liczny węzeł – dziecko.

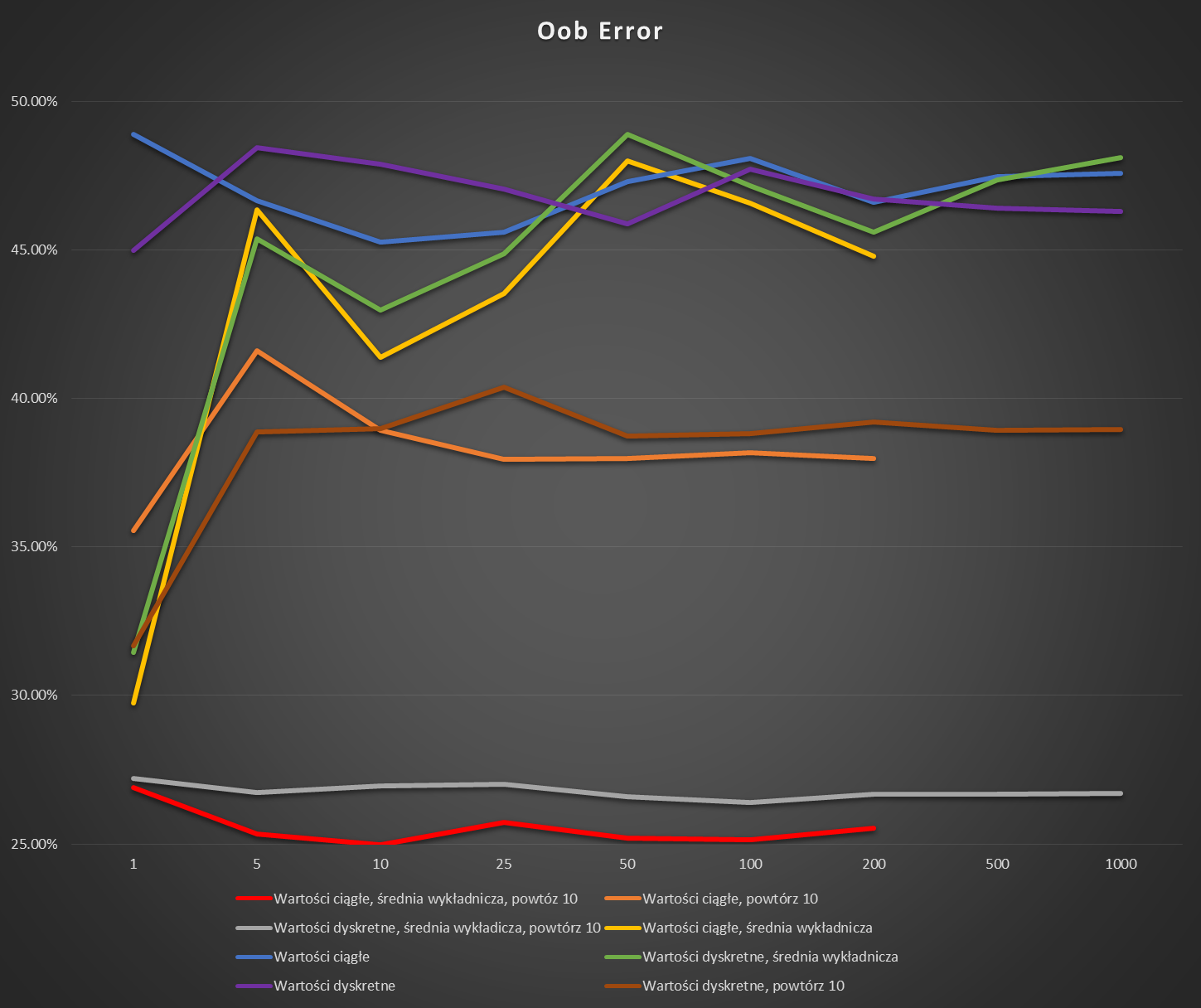
# Las Losowy – Wyniki

Poniższa tabela i wykres przedstawiają wyniki Oob Error, czyli liczbę przypadków błędnie sklasyfikowanych do liczby wszystkich przypadków. Im mniejszy procent, tym lepszy poziom klasyfikacji. Testy przeprowadzono dla następującej liczby drzew: 1, 5, 10, 25, 50, 100, 200, 500, 1 000 oraz dla wszystkich kombinacji następujących parametrów:

* Wartości dyskretne/wartości ciągłe – sposób podziału węzła przy budowie drzewa, dla wartości ciągłych. Opisane dokładnie w opisie budowania lasu losowego.
* Średnia wykładnicza/(puste) – zastosowanie średniej wykładniczej za ostatnie 24 godziny do atrybutów pogody lub pozostawienie nieprzetworzonych danych
* Powtórz 10/(puste) – powtórz budowanie drzewa 1 lub 10 razy i do lasu dodaj tylko drzewo z najmniejszym błędem Oob

Dla niektórych przypadków nie wykonano obliczeń, gdyż czas wykonania lasu losowego był zbyt długi – powyżej godziny. Dla pozostałych przypadków, czas wykonania to zwykle, kilka sekund lub kilka minut. Czas rośnie liniowo względem liczby drzew w lesie. Jeśli dokonujemy podziałów w oparciu o wartości ciągłe, a nie staramy się tworzyć sztucznych dyskretnych przedziałów, złożoność jest większa o co najmniej jeden rząd wielkości, ponieważ musimy sprawdzić wszystkie możliwe podziały zbioru. Powtarzanie budowy drzewa również wpływa na złożoność czasową, ponieważ w rzeczywistości budujemy i liczymy błąd oob dla 10 razy więcej drzew.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Oob Error | | | | | | | | |
| Ilość drzew | Wartości ciągłe, średnia wykładnicza, powtórz 10 | Wartości ciągłe, powtórz 10 | Wartości dyskretne, średnia wykładnicza, powtórz 10 | Wartości ciągłe, średnia wykładnicza | Wartości ciągłe | Wartości dyskretne, średnia wykładnicza | Wartości dyskretne | Wartości dyskretne, powtórz 10 |
| 1 | 26.91% | 35.56% | 27.20% | 29.75% | 48.89% | 31.44% | 45.00% | 31.67% |
| 5 | 25.33% | 41.61% | 26.74% | 46.35% | 46.67% | 45.38% | 48.44% | 38.89% |
| 10 | 24.99% | 38.94% | 26.97% | 41.39% | 45.28% | 42.97% | 47.89% | 39.00% |
| 25 | 25.73% | 37.96% | 27.01% | 43.55% | 45.61% | 44.88% | 47.04% | 40.39% |
| 50 | 25.19% | 37.99% | 26.61% | 48.00% | 47.31% | 48.90% | 45.89% | 38.73% |
| 100 | 25.14% | 38.18% | 26.41% | 46.58% | 48.09% | 47.18% | 47.74% | 38.83% |
| 200 | 25.54% | 37.97% | 26.67% | 44.79% | 46.62% | 45.62% | 46.73% | 39.20% |
| 500 |  |  | 26.69% |  | 47.46% | 47.37% | 46.40% | 38.94% |
| 1000 |  |  | 26.70% |  | 47.60% | 48.13% | 46.30% | 38.95% |

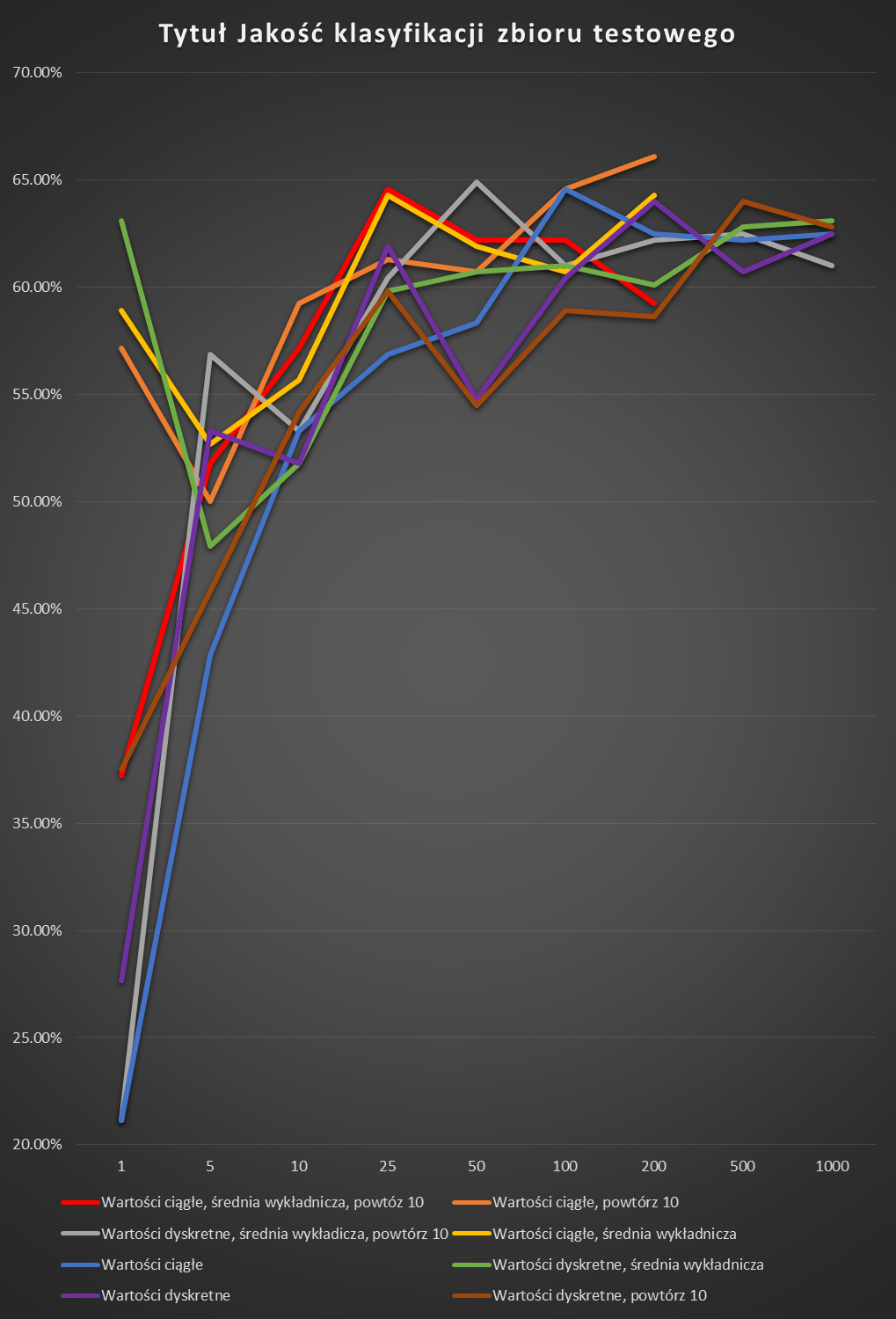


Analizując powyższy wykres I tabelę, łatwo zauważyć następujące wnioski:

* Tradycyjny podział wartości ciągłych daje nieco lepsze wyniki, niż próba łączenia w dyskretne przedziały (jednak wyniki są nieznacznie lepsze, a czas wykonania znacznie dłuższy).
* Zastosowanie średniej wykładniczej znacznie poprawia wyniki, nieznacznie wydłużając czas działania programu. Może to być spowodowane nie tylko liczeniem średniej wykładniczej, ale również zastąpienie wartości pustych, wartościami średnimi.
* Powtórzenie budowania drzewa 10 razy i wybranie tego z najmniejszym błędem oob, znacznie poprawia wyniki zmniejszając błąd klasyfikacji oob, czas wzrasta 10-krotnie.
* Liczba około 50 – 200 drzew wydaje się optymalna, utrzymują się stabilne wyniki, większa liczba drzew ich nie poprawia.

Poniższa tabela i wykres mają analogiczną postać i strukturę, do opisanych wcześniej. Z tą różnicą, iż tym razem przedstawia ona jakość klasyfikacji, czyli procent przypadków poprawnie sklasyfikowanych oraz testy zostały przeprowadzone na pierwotnym zbiorze testowym, a nie na zbiorze oob podczas budowania lasu. Tym razem, im większy współczynnik, tym lepszy poziom klasyfikacji.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Jakość klasyfikacji zbioru testowego | | | | | | | | |
| Ilość drzew | Wartości ciągłe, średnia wykładnicza, powtórz 10 | Wartości ciągłe, powtórz 10 | Wartości dyskretne, średnia wykładnicza, powtórz 10 | Wartości ciągłe, średnia wykładnicza | Wartości ciągłe | Wartości dyskretne, średnia wykładnicza | Wartości dyskretne | Wartości dyskretne, powtórz 10 |
| 1 | 37.20% | 57.14% | 21.13% | 58.93% | 21.13% | 63.10% | 27.68% | 37.50% |
| 5 | 51.79% | 50.00% | 56.85% | 52.68% | 42.86% | 47.92% | 53.27% | 45.83% |
| 10 | 57.14% | 59.23% | 53.27% | 55.65% | 53.27% | 51.79% | 51.79% | 54.17% |
| 25 | 64.58% | 61.31% | 60.42% | 64.29% | 56.85% | 59.82% | 61.90% | 59.82% |
| 50 | 62.20% | 60.71% | 64.88% | 61.90% | 58.33% | 60.71% | 54.76% | 54.46% |
| 100 | 62.20% | 64.58% | 61.01% | 60.71% | 64.58% | 61.01% | 60.42% | 58.93% |
| 200 | 59.23% | 66.07% | 62.20% | 64.29% | 62.50% | 60.12% | 63.99% | 58.63% |
| 500 |  |  | 62.50% |  | 62.20% | 62.80% | 60.71% | 63.99% |
| 1000 |  |  | 61.01% |  | 62.50% | 63.10% | 62.50% | 62.80% |



Przeanalizujmy najpierw wpływ testowanych parametrów na jakość klasyfikacji. Wykorzystajmy do tego wyniki dla 100 oraz 200 drzew, gdyż wyglądają one na najbardziej stabilne.

1. Tradycyjny podział wartości ciągłych, budowa drzewa powtórzona 10 razy  65% poprawnie sklasyfikowanych przypadków
2. Tradycyjny podział wartości ciągłych, drzewo budowane tylko raz  63%
3. Tradycyjny podział wartości ciągłych, ze średnią wykładniczą  62%
4. Sprowadzenie wartości ciągłych do dyskretnych  62%
5. Sprowadzenie wartości ciągłych do dyskretnych, ze średnią wykładniczą, powtórzone 10 razy  62%
6. Sprowadzenie wartości ciągłych do dyskretnych, ze średnią wykładniczą  61%
7. Tradycyjny podział wartości ciągłych, ze średnią wykładniczą, powtórzony 10 razy  61%
8. Sprowadzenie wartości ciągłych do dyskretnych, budowa drzewa powtórzona 10 razy  59% (jednak dla większej liczby drzew mamy tu 63%)

Analizując powyższe wyniki, łatwo zauważyć, iż nie potwierdzają one jednoznacznie wniosków i spodziewanych efektów wyciągniętych na podstawie analizy zbioru oob. Wszystkie wyniki są do siebie bardzo zbliżone, różnice są niewielkie.

* Tradycyjny podział wartości ciągłych w węźle rzeczywiście daje nieznacznie lepsze wyniki (należy pamiętać o większej złożoności).
* Powtórzenie budowania drzewa 10 razy, nie poprawiło jakości klasyfikacji, a 10-krotnie wydłużyło czas, co nie potwierdza wcześniejszych ustaleń na zbiorze oob.
* Nie można też potwierdzić, iż zastosowanie średniej wykładniczej korzystnie wpływa na poprawę klasyfikacji.
* Wydaje się, iż liczba drzew 100-500 jest optymalna, gdyż wyniki na tym poziomie się stabilizują.

Łatwo wytłumaczyć, dlaczego powtórzenie budowania drzewa i wybranie najlepszego korzystnie wpływa na jakość klasyfikacji zbioru oob, natomiast niekoniecznie dla zbioru testowego. Podczas budowania lasu są wtedy wybierane drzewa, które najlepiej dopasowują się do zbioru oob, jednak nie ma to przełożenia na zbiór testowy. Zbiór testowy może inaczej na to reagować, więc lepiej w tym konkretnym przypadku pozostawić większą losowość i większy błąd oob, gdyż jakość klasyfikacji zbioru testowego na tym nie ucierpiała.

Średnia wykładnicza daje porównywalne wyniki, co obliczenia na podstawie zwykłych danych, co można uzasadnić, iż uśrednianie danych i obliczenia średniej wykładniczej nie wpływają wprost na poprawę jakości klasyfikacji. Po raz kolejny dane ze zbioru treningowego były do siebie bardziej podobne, dlatego w przypadku zbioru oob, powodowało to poprawę.

Warto zauważyć, iż wyniki klasyfikacji dla liczby drzew mniejszej, niż 50 są fatalne, zależą od kwestii przypadku, wylosowania udanego zbioru atrybutów, a zbiór oob mógł sugerować, iż wyniki będą dużo lepsze. Jednak jeszcze raz należy zaznaczyć, iż przy tak małej liczbie drzew wyniki mogą być bardzo losowe i różnić się w zależności od seeda.

Analizując oob Error można z pewną dokładnością przewidzieć, jaki będzie poziom jakości klasyfikacji dla zbioru testowego. Powtarzanie budowy drzewa 10 razy zaburza tę predykcję, gdyż sztucznie zawyżamy oob. Porównajmy:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Porównanie Oob Error i jakości klasyfikacji dla 200 drzew | | | | | | | | |
|  | Wartości ciągłe, średnia wykładnicza, powtórz 10 | Wartości ciągłe, powtórz 10 | Wartości dyskretne, średnia wykładnicza, powtórz 10 | Wartości ciągłe, średnia wykładnicza | Wartości ciągłe | Wartości dyskretne, średnia wykładnicza | Wartości dyskretne | Wartości dyskretne, powtórz 10 |
| Oob Error | 25.54% | 37.97% | 26.67% | 44.79% | 46.62% | 45.62% | 46.73% | 39.20% |
| Spodziewana jakość | 74% | 62% | 63% | 55% | 53% | 54% | 54% | 61% |
| Prawdziwa jakość | 59.23% | 66.07% | 62.20% | 64.29% | 62.50% | 60.12% | 63.99% | 58.63% |

Tak jak zauważono wcześniej, w przypadkach, w których powtarzano budowanie drzewa 10 razy, osiągnięto niższą jakość klasyfikacji na zbiorze testowym, niż spodziewana. Natomiast zaskakująco, w pozostałych przypadkach osiągnięto o około 10% lepszą jakość klasyfikacji niż ta, która wynikała by z testowania na zbiorze oob. Może to być spowodowane bardziej różnorodnymi danymi w zbiorze testowym, więc drzewa dla nich nie są z pewnością prze uczone.

Las Losowy a kNN

kNN poprawnie klasyfikował około 53% procent elementów ze zbioru testowego, natomiast las losowy (w zależności od konfiguracji parametrów) około 62%. Widzimy, więc, iż las losowy jest lepszym klasyfikatorem niż kNN, co jest zgodne z naszym intuicyjnym przypuszczeniem.

# Etap III

## XGBoost

W 3 etapie chcieliśmy osiągnąć jak najlepszy wynik czułości na zbiorze testowym. W tym celu skorzystaliśmy z biblioteki xgboost i spróbowaliśmy różnych podejść do rozwiązania problemu.

### Przygotowanie danych

1. Wyciągnięcie danych z bazy danych do plików odt (ręcznie, folder SmogDetector.Database/tables)
2. Przekształcenie plików odt do csv (ręcznie) i podział pliku z pogodą na plik z danymi testowymi i treningowymi (folder SmogDetector.Database/tables)
3. Przekształcenie plików csv (trenigowego i testowego z pogodą i smogowego) do plików w formacie libsvm (robi to plik SmogDetector.Python/prepare\_data.py)
4. Napisanie odpowiednich plików robiących regresję i klasyfikację- format „get\_result\_ver\_X.py” (folder SmogDetector.Python, X to liczba)

### Dane

|  |  |  |
| --- | --- | --- |
| klasa | Dane treningowe | Dane testowe |
| 1 | 263 | 157 |
| 2 | 243 | 45 |
| 3 | 291 | 58 |
| 4 | 283 | 76 |
| Wszystkie klasy | 1080 | 336 |

Tabela: Rozkład danych na klasy w zbiorach treningowych i testowych

Zbadano następujące opcje:

* klasyfikacja dla klas smogu <1,2,3,4>
* klasyfikacja dla wartości smogu <1...200>
* regresja dla wartości smogu
* regresja dla wartości smogu
* regresja dla klas smogu
* klasyfikacja dla klas smogu z budowaniem drzewa DART
* regresja dla wartości smogu z budowaniem drzewa DART
* klasyfikacja dla klas smogu z użyciem funkcji liniowej
* regresja dla wartości smogu z użyciem funkcji liniowej

### Wybór metryki w 3 etapie

Początkowo zastosowano metryki:

1. dla klasyfikacji: mlogloss i merror: są to odpowienio metryki logloss i czułość(accuracy, recall). W poprzednich etapach jedyną metryką była czułość, stąd on była najważniejszą metryką
2. dla regresji: początkowo metryką była rmse (pierwiastek z błędu średniokwadratowego), następnie dodano nową metrykę, która zamieniała wartości na klasy i liczyła czułość

### Wyniki i obserwacje

Tabela. Wartości czułości na zbiorze testowym w zależności od podejścia do problemu i głębokości drzewa (oprócz funkcji liniowej) dla domyślnej metryki

Wykres. Czułość na zbiorze testowym przy uczeniu z zastosowaniem domyślnych metryk (dla klasyfiakcji merror, dla regresji rmse)

Tabela. Wartości czułości na zbiorze testowym w zależności od podejścia do problemu i głębokości drzewa (oprócz funkcji liniowej) dla metryki czułości dla klasyfikacji

Wykres. Czułość na zbiorze testowym z zastowaniem konwesji wartości do klas I wyznaczaniem błędów dla klas

Podejście klasyfikacja dla wartości smogu dawało słabe wyniki, jak I trwało bardzo długo, dlatego też zrezygnowano z jego dalszej analizy.

Należy zauważyć, że

* Choć wraz z kolejnymi iteracjami na zbiorze treningowym był coraz niższy, to błąd na zbiorze testowym zmieniał się nieprzewidywalnie
* Już od pierwszej iteracji błąd na zbiorze testowym utrzymywał się na podobnym poziomie
* Bardzo często już pierwsze iteracje dawały bardzo dobre rezultaty
* Dla klasyfikacji głębokość drzewa nie sprzyjała lepszym wynikom na zbiorze testowym, zaś dla regesji na wartościach smogu dla budowanie drzew GBM i DART lepsze wartości dawały płytsze drzewa
* Okazało się, że najlepiej sprawdza się klasyfikacja dla klas smogu, zarówno z typowym budowaniem dla xgboosta jak i strategią DART



Analiza tabeli pomaga stwierdzić, że:

* Klasa 0 była bardzo dobrze rozpoznawana ok 20%(kolumna c0\_r)
* Klasa 1 była bardzo słabo rozpoznawana - ok 80-90%(kolumna c1\_r)
* Klasa 2 była bardzo rozpoznawana w ok 50% przypadków (kolumna c1\_r)
* Klasa 3 była bardzo rozpoznawana w ok 80% przypadków (kolumna c1\_r)

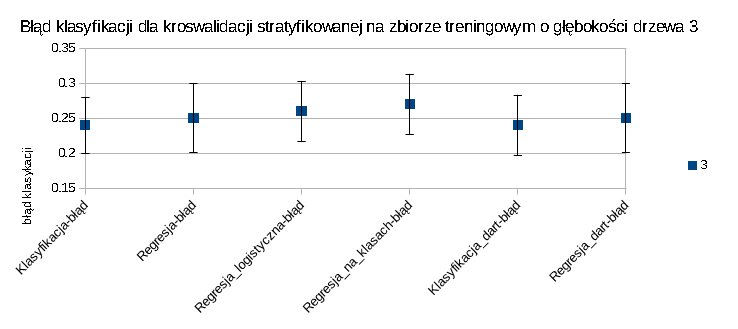
Z dużym prawdopodobieństwem jest to spowodowane rozkładem na zbiorze testowym. Tam gdzie rozpoznawanie jest lepsze, tam jest więcej elementów danej klasy.

Wykres. Porównanie działania dla regresji dla strategii budowania drzew GBM i DART z uwzględnieniem metryki

Zgodnie z przewidywaniami zastosowanie metryki w której wartości predykcji są przekształcane do wartości klas daje lepsze wyniki- wynika to z faktu, że optymalizujemy metrykę, którą ostatecznie sprawdzamy



Tabela. Błąd klasyfikacji, odchylenie standardowe i najlepsza iteracja dla kroswalidacji stratyfikowanej na zbiorze treningowym dla głębokości drzewa 3

Wykres. Błąd klasyfikacji dla kroswalidacji stratyfikowanej na zbiorze treningowym dla głębokości drzewa 3

W celu sprawdzenia, czy nie zachodzi w przypadku wyników zjawisko overfitingu, sprawdziliśmy wyniki dla kroswalidacji na zbiorze treningowym. Okazało się, że wyniki nie potwierdzają problemu overfittingu, wartości błędu wynoszą ok 25%, a najlepszy wynik czułości na zbiorze testowym to 71% (błąd 29%). Można więc zastanowić się, czy wynik na zbiorze testowym nie jest za niski- sądzimy, że jest to związane z podziałem danych- pochodzą z różnych okresów, a wynik jest zadowalający

# Podział pracy

* Bartosz Woźniak
  + Analiza danych związanych z pogodą
  + Synchronizacja danych związanych z pogodą
  + Przetworzenie danych związanych z pogodą
  + Normalizacja danych związanych z pogodą
  + Analiza danych związanych ze smogiem
  + Synchronizacja danych związanych ze smogiem
  + Przetworzenie danych związanych ze smogiem, podział na klasy
  + Stworzenie architektury bazy danych
  + Stworzenie architektury aplikacji
  + Implementacja kNN
  + Analiza wyników kNN
  + Analiza różnych rozwiązań stosowanych w lasach losowych
  + Implementacja lasu losowego
  + Rozwiązanie problemów, np. związanych z brakami danych
  + Analiza wyników lasu losowego
  + Stworzenie dokumentacji, raportu
* Krzysztof Małaśnicki
  + przeniesienie danych z bazy danych MS SQL do plików tekstowych
  + dostosowanie danych do wymagań dla wejścia dla biblioteki xgboost
  + zaimplementowanie korzystania z biblioteki xgboost wraz z wariantami
  + zaimplementowanie kroswalidacji z biblioteki xgboost wraz z wariantami
  + analiza wyników: wyszukiwanie trendów, stawianie hipotez i ich weryfikacja
  + wyjaśnienie przyczyn,
  + rozbudowa dokumentacji i raportu