Dawid Bartosiak 318361

WSI Ćwiczenie 4 - Regresja i klasyfikacja

1. Wstęp

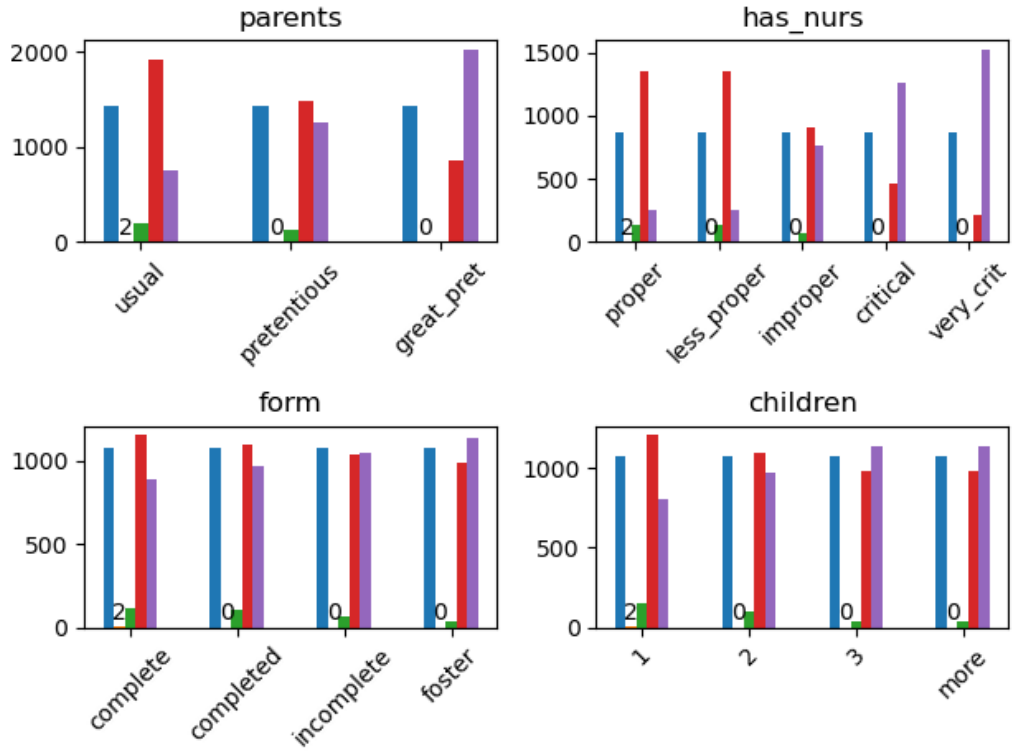
Celem ćwiczenia 4 było zaimplementowanie algorytmu lasu losowego którego zadaniem była klasyfikacja przyjęcia dzieci do przedszkola na podstawie informacji o strukturze i finansach rodziny. Warto tutaj zwrócić uwagę na szczególny rozkład wspomnianego zbioru – znajdują się w nim także klasy rzadkie (występują 2 i 328 razy na 12960 obserwacji). Może to stworzyć problem – gdy w zbiorze treningowym nie wylosuje się żadna obserwacja z wspominianych klas, będziemy mieli problem z ich przypisaniem (prawdopodobnie nie przypiszemy ich do poprawnych klas).

2. Implementacja

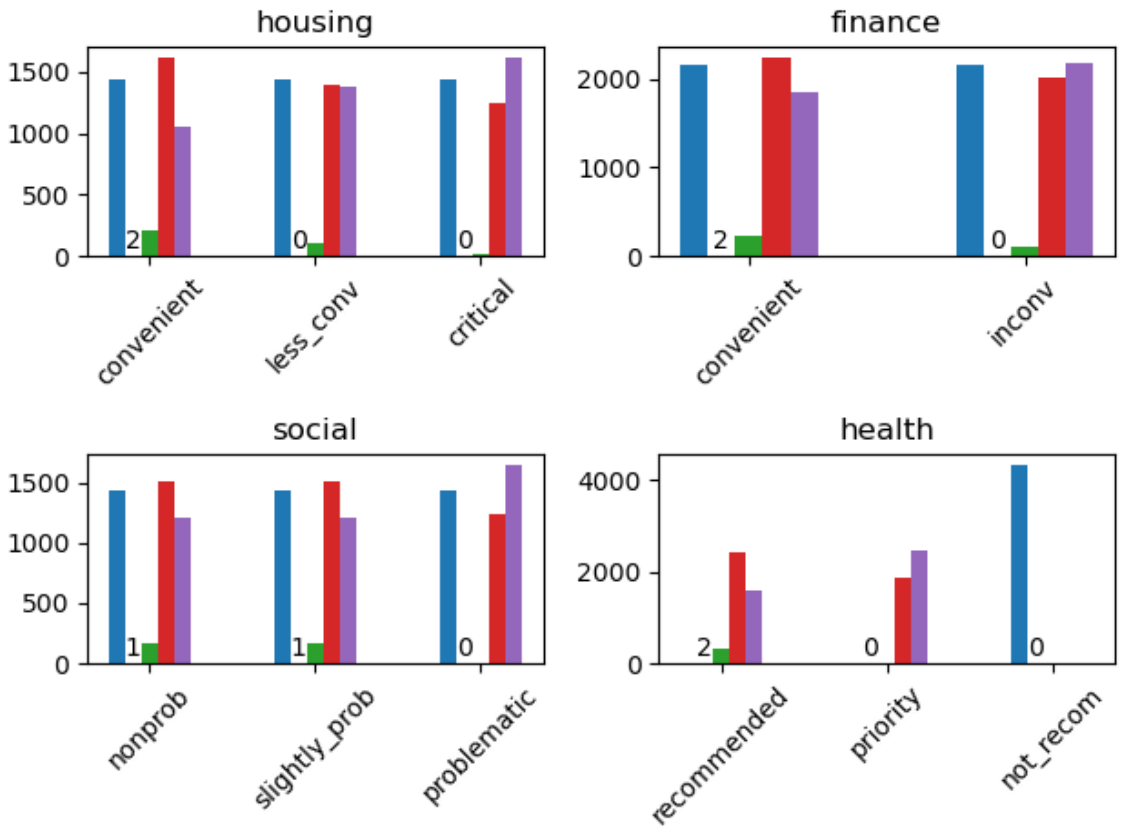
Wykorzystując programowanie obiektowe utworzona została utworzona klasa lasu losowego (RandomForest) oraz pomocnicze klasy Node i Leaf. Wybierana jest decyzja najczęściej wskazywana przez drzewa. Program został sparametryzowany pod względem:

-Ilości drzew
-Ilości używanych parametrów dla każdego z drzew
-Maksymalnej głębokości drzew
-Ilości danych użytych do trenowania
-Sposobu podawania danych treningowych (posortowane/potasowane)

3. Dane wejściowe



Rys. 1: Wykresy danych wejściowych

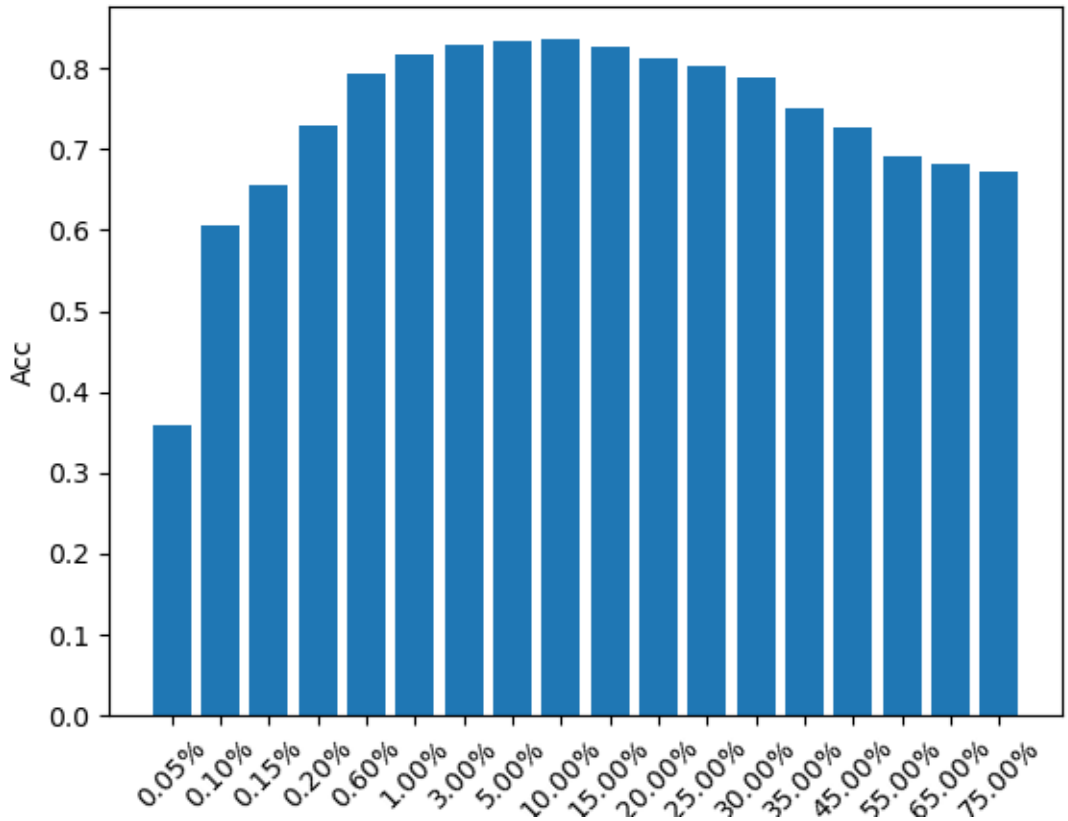


Rys. 2: Wykresy danych wejściowych

Na wyżej przedstawionych wykresach widać rozkład danych wejściowych. Zauważyć łatwo, że dane dzielą się na 5 klas z czego 3 są „zbalansowane” a dwie pozostałe mają odpowiednio mało (328) i bardzo mało (2) obserwacji, co stanowi szczególne wyzwanie w kwestii zadania klasyfikacji. Szczególnie ciekawe są tutaj wykresy dotyczące zdrowia oraz aktualnego przedszkola dziecka, gdyż na nich jest największe zróżnicowanie w klasach (z czego aktualne przedszkole ma najwięcej „stanów”).

4. Wpływ danych wejściowych.

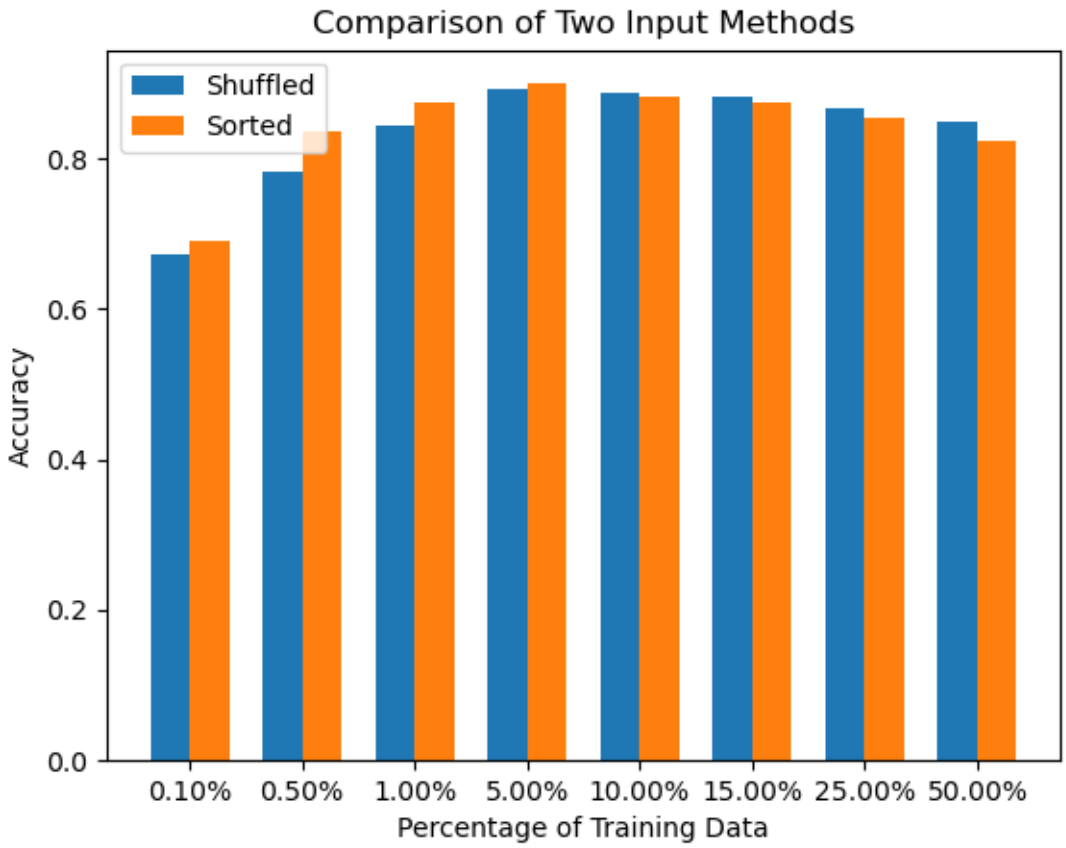
Dla identycznych parametrów wejściowych zmieniałem jedynie ilość danych użytych do trenowania, dla tego samego ziarna losowości oraz pozostałych parametrów, które wynosiły: max 5 parametrów na drzewo, 10 drzew, max głębokość 5.



Rys. 3: Wykres poprawności modelu klasyfikacyjnego w zależności od ilości danych użytych do trenowania

Na Rysunku 3 przedstawiona została dokładność klasyfikacji (Accurracy) w zależności od procentowej ilości użytych danych do trenowania. Widać że najlepszy wynik otrzymaliśmy dla użycia 10% danych jako dane treningowe i dokładność ta wyniosła 83,56%. Dla niższych wartości widoczne jest niedouczenie, a dla większych przeuczenie.

Kolejnym eksperymentem który należy przeprowadzić, jeżeli rozważamy wpływ danych wejściowych jest sprawdzenie jak zachowa się nasz algorytm gdy dodatkowo posortujemy wstępnie dane (po kolumnie odpowiadającej za wynik).

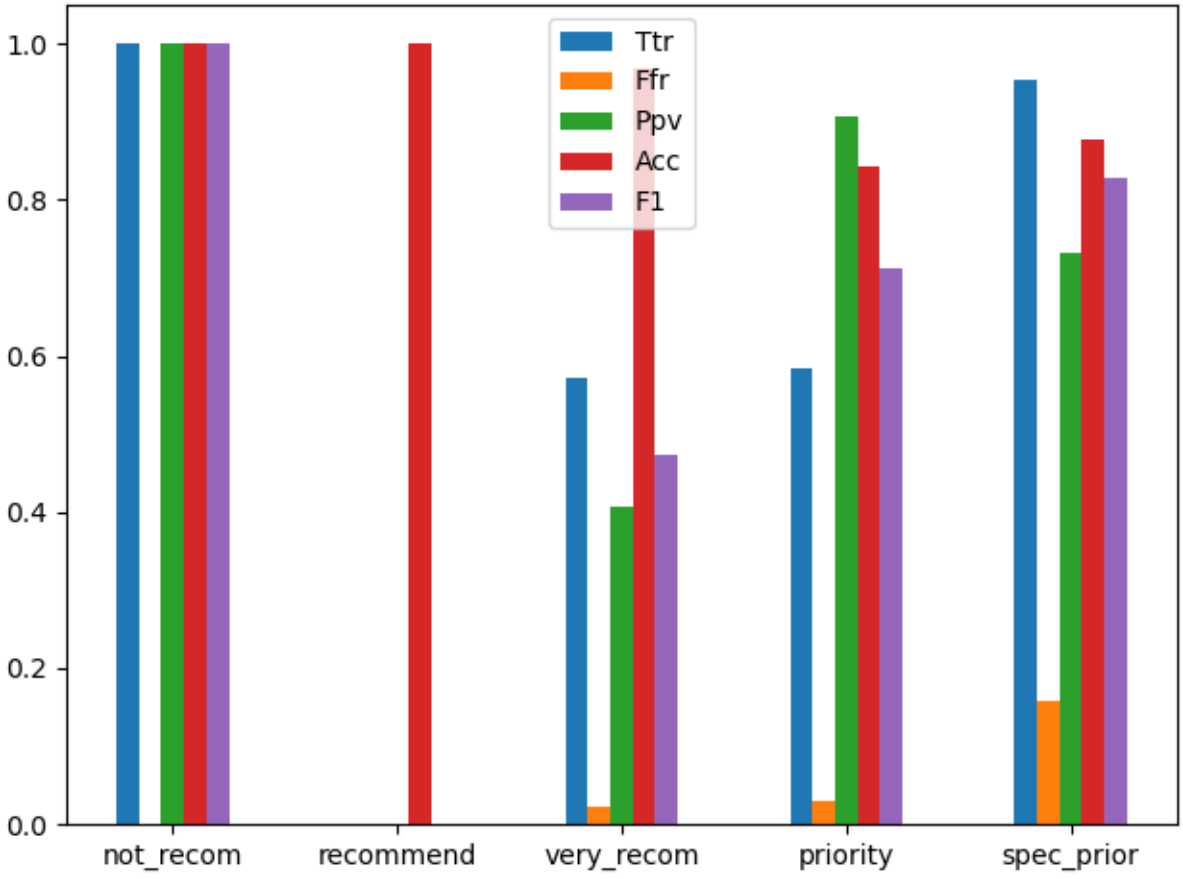


Rys. 4: Wykres poprawności modelu klasyfikacyjnego w zależności od ilości danych użytych do trenowania oraz sposobu podania tych danych

Widać, że nauka dla danych posortowanych jest szybsza, ale za to model wcześniej zaczyna mieć problemy z generalizacją i następuje overfitting. Najlepszy wynik otrzymujemy dla 5% (87,51% dokładności), dla danych posortowanych. Program wykonuje się szybko, a różnice w czasach wykonania są nieznaczne, wyniki mogłyby być zaburzone przez błędy pomiarowe.

5. Efekty działania programu.

Aby sprawdzać działanie programu na wielu poziomach należy sprawdzić macierz błędów, jednak sam program w następnym kroku będę optymalizować głównie z uwagi na poprawność przewidywanych danych (Accuracy). Z uwagi na mnogość możliwych wyników potrzebne byłoby 5 macierzy, w ramach poprawienia czytelności decyduję się na ukazanie danych na wykresie słupkowym i tabeli, dzieląc je z uwagi na klasę. Dla aktualnie rozważanego przypadku (parametry z punktu czwartego, posortowane wejście, 5% danych przeznaczone na trening) wykres macierzy błędów i jej wartości prezentują się następująco:



Rys. 5: Wykres metryk z macierzy błędów dla rozważanego algorytmu lasu losowego

Tabela 1: Dane z macierzy błędów

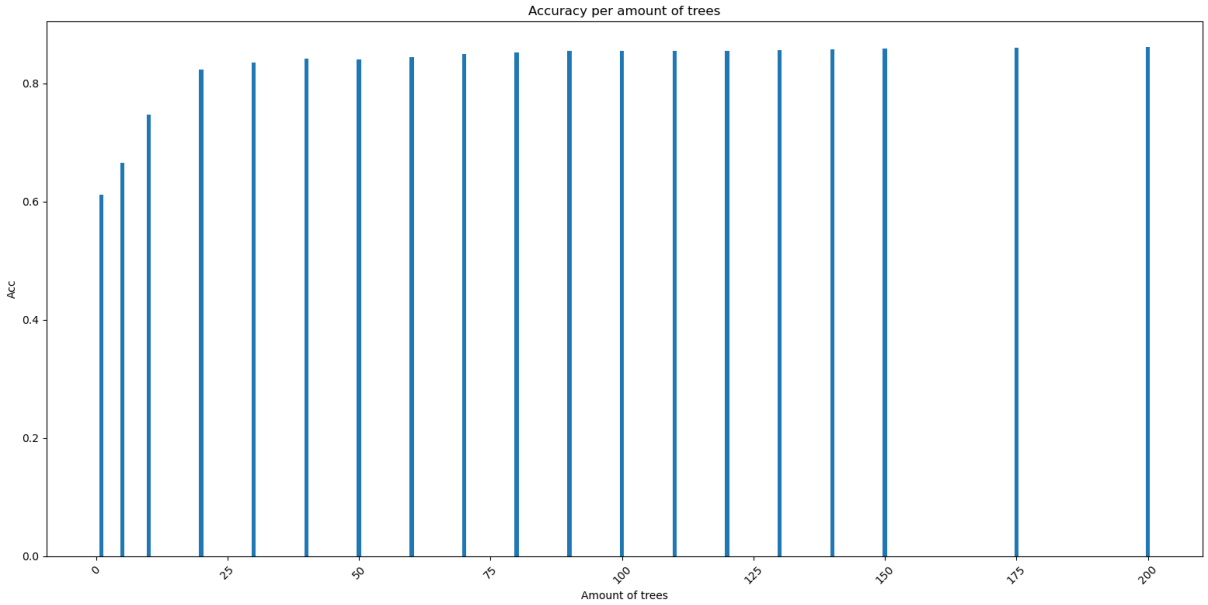
Klasa	TP	FP	FN	TN
not_recom	4111	0	0	8201
recommend	0	0	0	12310
very_recom	177	259	133	11743
piority	2378	241	1685	8008
spec_prior	3649	1334	177	7152

Na rysunku 5 zauważyć możemy, że najlepiej algorytm poradził sobie w przypadku nie-zalecanym i zalecanym (są w nim 2 przypadki, obydwa w zbiorze treningowym, gdyż widoczny na wykresie jest tylko diagram gdzie nie jest wymagany żaden przypadek prawdziwie pozytywny). Ciężko jest powiedzieć coś więcej o modelu z uwagi na brak korelacji – dla klasy priority mamy więcej przypadków niepoprawnego sklasyfikowania do innej klasy zaś dla very_recom i spec_prior mamy więcej przypadków niepoprawnego sklasyfikowania do tych właśnie klas.

6. Dobór optymalnych parametrów.

Optymalne parametry dobierane były metodą inżynierską w oparciu o obserwację zachowania algorytmu dla wielu próbek. W celu lepszej wizualizacji zamieszać będę do każdego z parametrów wykresy uzasadniające jego wybór. Każda pętla zaczyną

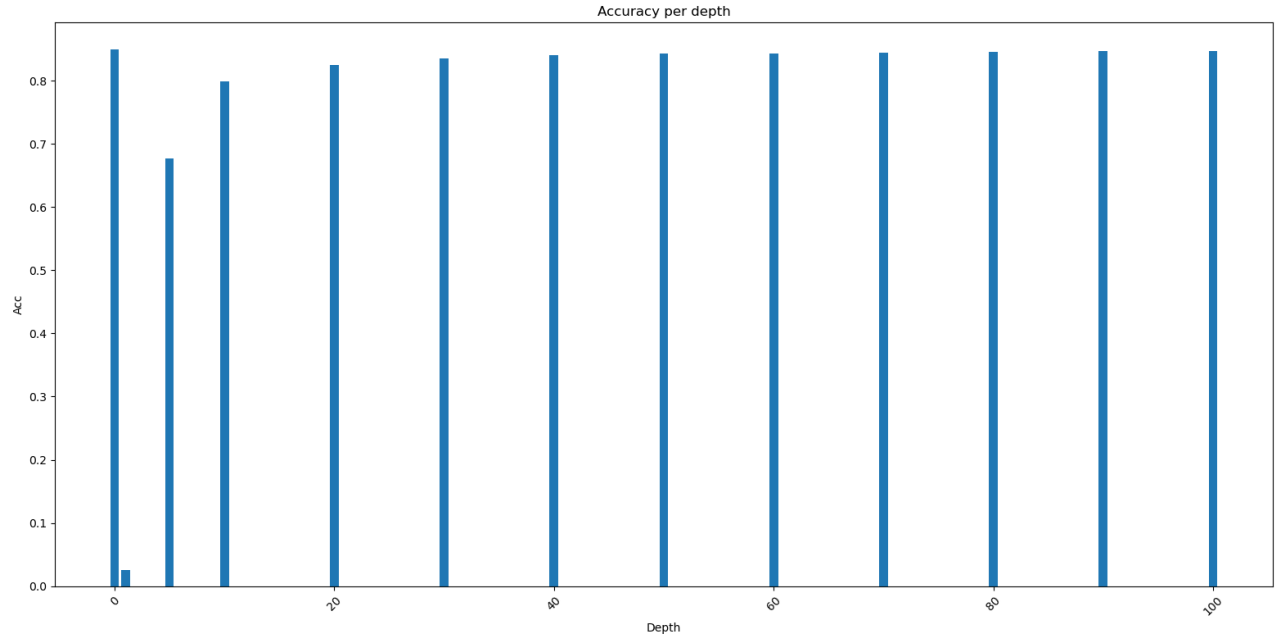
a) Dobór ilości drzew.



Rys. 6: Wyniki działania algorytmu dla różnej ilości drzew

Tutaj niestety ciężko o jednoznaczną odpowiedź. Im więcej drzew tym lepszy wynik, ale biorąc pod uwagę wymaganą moc obliczeniową decyduję się na wybór 40 drzew z dokładnością 84,18%

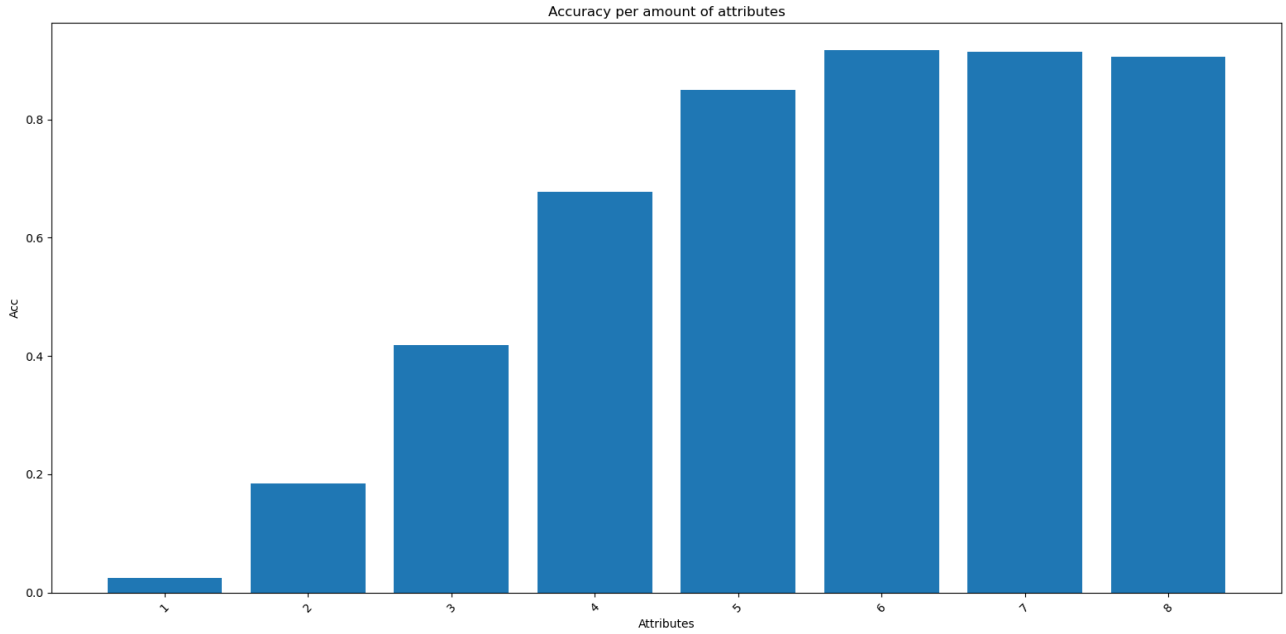
b) Dobór maksymalnej głębokości drzew.



Rys. 7: Wyniki działania algorytmu dla różnej głębokości

W przypadku głębokości czas wykonania pozostaje praktycznie bez zmian zatem decyduję się na brak ograniczenia (ustawienie 0) z dokładnością 84,94%.

c) Dobór ilości atrybutów.



Rys. 8: Wyniki działania algorytmu dla różnej ilości atrybutów

Dobór ilości atrybutów jest prosty, tutaj wynik najlepszy otrzymałem dla 6 atrybutów i wynosi on 91,72% dokładności

Ostateczne nastawy naszego programu to:

-Zbiór testowy to 5% danych

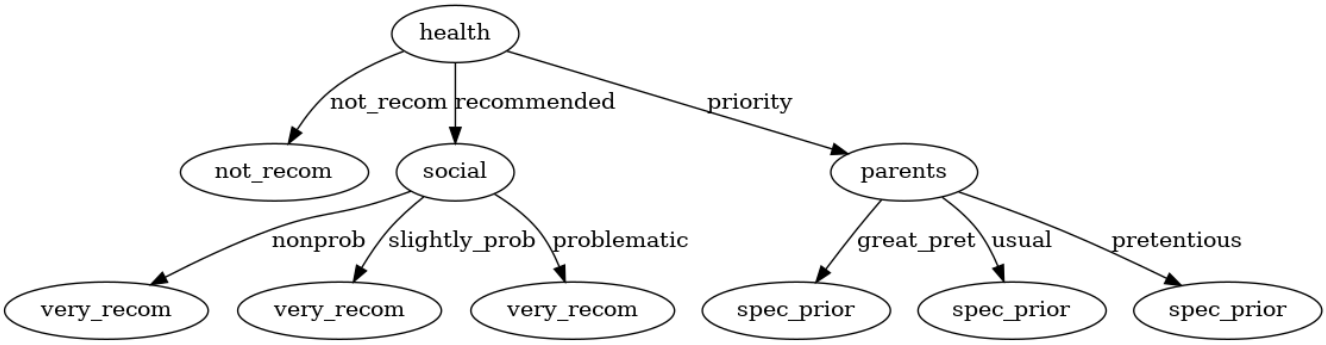
-Ilość drzew – 40

-Maksymalna głębokość – 0 (nieograniczona)

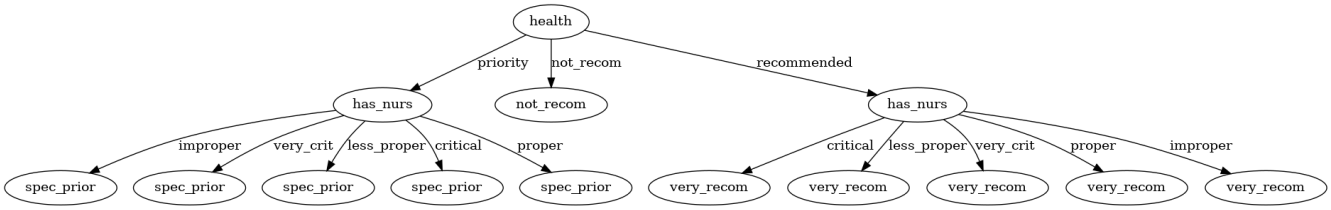
-Ilość parametrów - 6

-Dane wejściowe posortowane

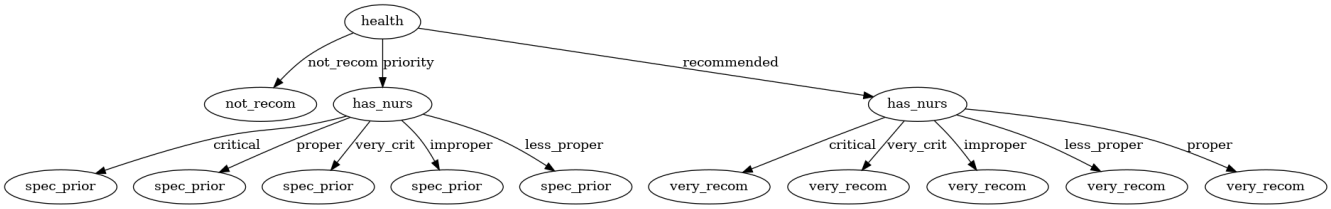
Przykładowe drzewo decyzyjne z programu nie jest w stanie się zmieścić w dokumencie w sensownej rozdzielczości, zatem załączam je jako dodatkowy plik – tree-example.png, a tutaj wrzucam przykładowe drzewa o głębokości 2:



Rys. 9: Przykładowe drzewo decyzyjne



Rys. 10: Przykładowe drzewo decyzyjne



Rys. 11: Przykładowe drzewo decyzyjne

Na powyższych rysunkach widoczna jest moja teza którą powiedziałem w punkcie pierwszym, że szczególnie ważne w danych wydaje się zdrowie i aktualne przedszkole dziecka. Są to dane, które najczęściej występują w drzewach decyzyjnych.

7. Wnioski.

Wniosek 1: Problem zbalansowania klas

W analizie przedstawionej w punkcie pierwszym zauważyłem wyzwanie związane z nierównomiernym rozkładem klas w zbiorze danych. Przewaga jednych klas nad drugimi może prowadzić do problemów z poprawnym przypisaniem rzadkich klas w przypadku, gdy nie występują w zbiorze treningowym. To zjawisko wymaga szczególnej uwagi w doborze parametrów, aby możliwie jak najbardziej uniknąć błędów wynikających z niezrównoważonej liczby obserwacji w poszczególnych klasach, starając się przede wszystkim o otrzymanie odpowiedniego zbioru uczącego (zmuszanie algorytmu do ekstrapolacji nie jest zjawiskiem zamierzonym).

Wniosek 2: Skuteczność algorytmu w zależności od danych treningowych

Przeprowadziłem eksperyment, zmieniając procentowy udział danych treningowych. Wykres przedstawiający dokładność klasyfikacji wskazuje, że dla 10% danych treningowych osiągnięto najlepsze rezultaty. Jednakże, warto zauważyć, że zarówno niedouczenie, jak i przeuczenie mogą występować przy skrajnych wartościach procentowych danych treningowych. Optymalny poziom zdaje się być zależny od charakterystyki konkretnego zbioru danych.

Wniosek 3: Wpływ posortowania danych na proces uczenia

Eksperyment z posortowanymi danymi wykazał, że nauka dla posortowanych danych jest szybsza, ale równocześnie model wcześniej zaczyna mieć problemy z generalizacją. Istnieje zauważalne ryzyko overfittingu, co może prowadzić do gorszej wydajności algorytmu na nowych danych. To potwierdza konieczność ostrożnego dostosowania modelu, aby uniknąć przetrenowania.

Wniosek 4: Analiza macierzy błędów

Przeanalizowane zostały wyniki algorytmu poprzez przedstawienie macierzy błędów i dokładności dla poszczególnych klas. Wykresy i tabele pozwalają zauważyć, że algorytm radzi sobie najlepiej z klasami "not_recom" i "recommend", ale dla pozostałych klas trudno jednoznacznie ocenić skuteczność modelu z powodu braku korelacji.

Wniosek 5: Optymalizacja parametrów algorytmu

Przeprowadzona została analiza wpływu różnych parametrów na skuteczność algorytmu. Ostateczne ustawienia obejmują 40 drzew, brak ograniczenia głębokości, 6 używanych atrybutów i posortowane dane wejściowe. Dla tych parametrów osiągnięto dokładność na poziomie 91,72%. Ostateczny wybór parametrów powinien być dostosowany do specyfiki danych i zasobów obliczeniowych.

Wniosek 6: Wpływ zdrowia i aktualnego przedszkola na decyzje algorytmu

Analiza struktury drzew decyzyjnych wskazuje, że zdrowie i aktualne przedszkole dziecka są kluczowymi atrybutami wpływającymi na decyzje algorytmu. To potwierdza wcześniejszą obserwację dotyczącą istotności tych danych w procesie klasyfikacji.

Wnioski ogólne:

Ćwiczenie 4 skupiło się na implementacji algorytmu lasu losowego do klasyfikacji przyjęcia dzieci do przedszkola na podstawie danych o rodzinie. Przeprowadzony został szereg eksperymentów, i analiza wpływu różnych czynników na skuteczność algorytmu. Wyniki wskazują na konieczność uwzględnienia specyfiki danych i ostrożnego dostosowywania parametrów modelu.