Uczenie Maszynowe

Końcowe sprawozdanie z Projektu

Temat projektu: Techniki oceny klasyfikacji dla zestawów danych dotyczących raka piersi

Mateusz Krakowski

Bartosz Latosek

13 czerwca 2023

Spis treści

1 Opis Projektu

1.1 Treść zadania

Zaimplementuj techniki oceny klasyfikacji dla zestawów danych dotyczących raka piersi, które dostępne są w: http://archive.ics.uci.edu/ml/datasets.

1.2 Użyte dane

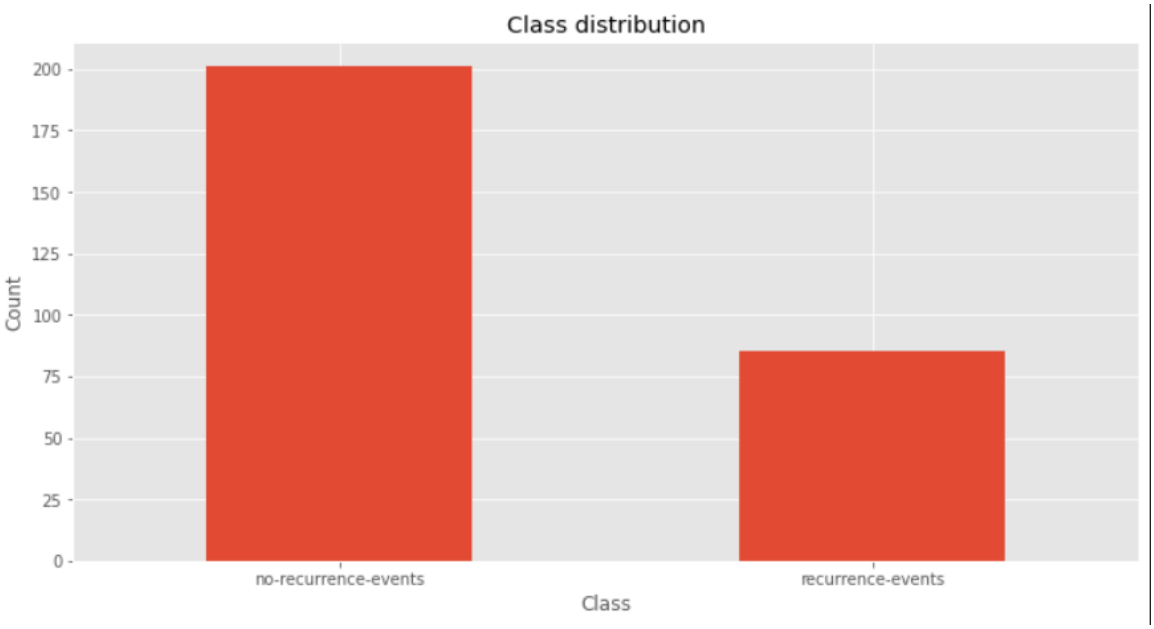
Wykorzystaliśmy dane z datasetu Breast Cancer Data Set [LINK] oraz Breast Cancer Wisconsin Data Set [LINK]

1.3 Analiza danych

Pełna analiza obydwu zbiorów danych znajduje się w plikach data_analisis.ipynb, w odpowiednich katalogach z danymi testowymi. Tutaj zamieszczamy skrót naszych odkryć.

1.3.1 Breast-Cancer

Pierwszym, co rzuciło nam się w oczy przy analizie tego zbioru danych była bardzo mała liczba przypadków. Mamy bowiem do czynienia ze zbiorem 286 przypadków, z czego 14 z nich to po prostu duplikaty z tymi samymi wartościami kolumn.
Problem pogłębia znaczna dysproporcja w zbiorze danych, którą przedstawia poniższy wykres.



Rysunek 1: Wizualizacja dysproporcji

Widzimy znaczną dysproporcję w zbiorze danych. Stosunek osób zdiagnozowanych jako zagrożonych do osób zdrowych wynosi 1 do 2.36 - dysproporcja jest zauważalna. Przy uczeniu

niektórych modeli należy poświęcić czas na odpowiednie rozwiązanie tego problemu - dobieranie przypadków uczących zważając na dysproporcję - np. WeightedRandomSampler.

W dalszej części analizy danych wykryliśmy, że atrybuty node-caps oraz breast-quad posiadają brakujące wartości. Braków było jednak za mało, by poddać je dalszej analizie - jaki był powód braków danych w tych atrybutach, więc założyliśmy, że są to braki typu Missing Completly At Random, a w związku z małą ilością danych - wadliwych wierszy nie usunęliśmy a zamiast tego zastąpiliśmy brakujące wartości najczęściej występującymi wartościami korespondujących atrybutów.

Przeanalizujmy teraz częstość występowania atrybutów i ich związki z klasą pacjenta.

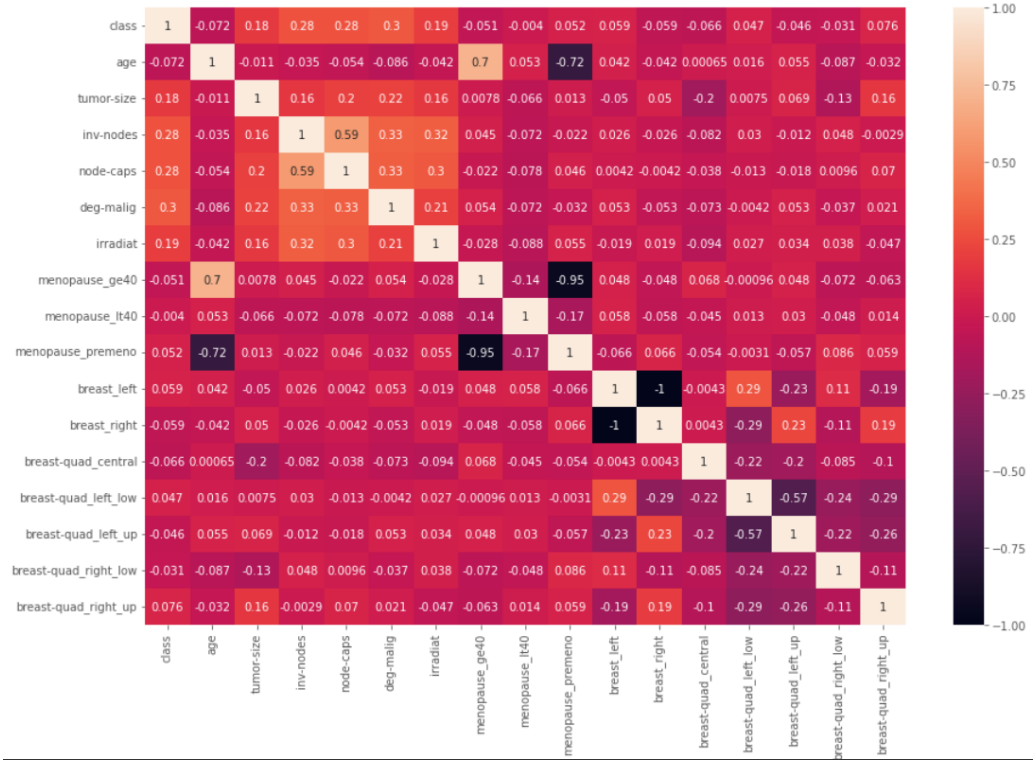
Rysunek 2: Stosunek wartości atrybutu do częstości występowania.

Wizualizacja atrybutów pozwoliła nam lepiej zrozumieć dystrybucje ich wartości w zbiorze danych. Wartości ich są niezbalansowane. W przypadku np. kolumny reprezentującej wiek pacjenta - przeważająca część pacjentów była w zakresie 40-60 lat. Bardzo ubogi jest zakres

młodych pacjentek w wieku od 20 do 40 lat, co może wpłynąć na jakość otrzymywanych modeli, jeżeli w danych testowych znajdzie się dużo pacjentek z tego zakresu wiekowego. Najsilniej widoczna dysproporcja jest zauważalna na wykresie inv-nodes. Z domeny medycznej - atrybut ten oznacza ilość pachowych węzłów chłonnych zawierających przerzuty raka piersi widoczne w badaniu histopatologicznym. Widzimy, że znaczna część pacjentów miała jedynie 0-2 takich węzłów, stąd informacja że dane dotyczą raczej osób we wczesnych fazach raka piersi.

W tym momencie analizy zbioru danych zebraliśmy wystarczająco dużo informacji, aby przygotować funkcję preprocessingu. W ramach jej działania pozbywamy się brakujących wartości z atrybutów, atrybuty dyskretne kodujemy przy pomocy one-hot encoding a atrybuty reprezentujące przedziały liczbowe uśredniamy i poddajemy normalizacji w celu polepszenia uczenia modeli.

Po preprocessingu danych możemy przejść do głębszej analizy zależności pomiędzy atrybutami.



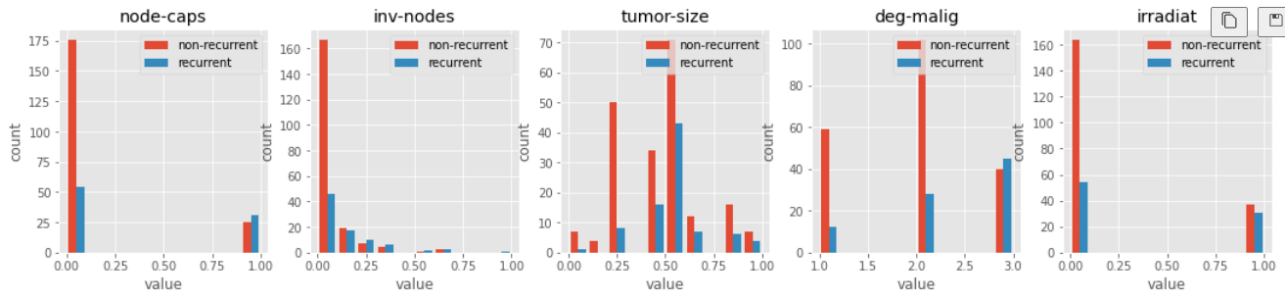
Rysunek 3: Tabela Korelacji Atrybutów

Z powyższego wykresu można odczytać informacje oczywiste - takie jak to, że gdy u pacjenta występuje guz w lewej piersi - to nie występuje on w prawej. Oczywiście z medycznego punktu widzenia, taka możliwość istnieje, ale nasz zbiór danych jest mocno zubożony i dotyczy pacjentek których badana była tylko jedna pierś. Analogicznie, widzimy silną zależność odwrotną pomiędzy menopause_premeno a wiekiem oraz menopause_ge40. Ma to sens, ponieważ przedwczesna menopauza oznacza, że pacjentka musiała być w młodym wieku i na pewno

nie mogła przechodzić me wykresu można zauważyć bardzo duży wpływ atrybutów Bare Nuclei i Cell Size na predykcje modelunopauzy w wieku >= 40 lat. Pozostają inne zależności takie jak breast_left i umiejscowienie guza w którymś miejscu lewej piersi, ale analiza tych zależności nic nie wniesie do projektu.

Co może nas faktycznie interesować, to to że istnieje silna zależność między klasą, a inv-nodes oraz klasą a node-caps, a dodatkowo te dwa atrybuty są silnie skorelowane ze sobą. Mogą one okazać się kluczowe dla naszych modeli i są warte dalszej analizy.

Porównanie liczności klas z wartościami silnie skorelowanych z nią atrybutów



Rysunek 4: Wykresy liczności

Z powyższego wykresu widać, że atrybuty tumor-size oraz deg-malig układają się zgodnie z rozkładem normalnym. Co więcej, deg-malig = degree of maligence - stopień złośliwości, pokazuje nam, że najczęściej spotykanym w danych poziomem złośliwości był poziom 2, i wraz z jego wzrostem rośnie liczba pacjentek znajdujących się w grupie ryzyka. Najliczniejszą grupę pacjentek pod względem atrybutu inv-nodes stanowią pacjentki z liczbą węzłów chłonnych z zakresu od 0 do 2. Wraz ze wzrostem tej liczby, maleje liczba pacjentek w grupie zagrożenia. Co do poprzednich atrybutów, rozłożenie klas jest zbalansowane i zgodne z rozkładem atrybutu.

Dodatkowo w internecie znaleźliśmy informacje o tym, że breast-quadrants zostały niepoprawnie przypisane do 'prawej piersi' pomimo tego, że dotyczyły one lewej piersi. W związku z tym, że dane są zakłamane i nie są szczególnie istotne dla naszego modelu, w ostatecznej wersji zdecydujemy się na pominięcie tych atrybutów w uczeniu modeli.

Wnioski z analizy danych:

- posiadamy dane 286 osób, jest to mało aby nauczyć dobry klasyfikator, ale nie będziemy się w tym projekcie na samym uczeniu dobrych klasyfikatorów, a na wizualizacji miar jakości modeli.
- Klasą większościową są osoby u których nie wystąpiły zdarzenia rekurencyjne (no-recursive-events), stanowią 70% całego datasetu. Reszta to osoby u których wystąpiły zdarzenia rekurencyjne.
- Udało nam się zauważyć silną korelację między wiekiem a posiadaniem menopałzy ge40 oraz niski wiek jest skorelowany z posiadaniem menopauzy premeno.

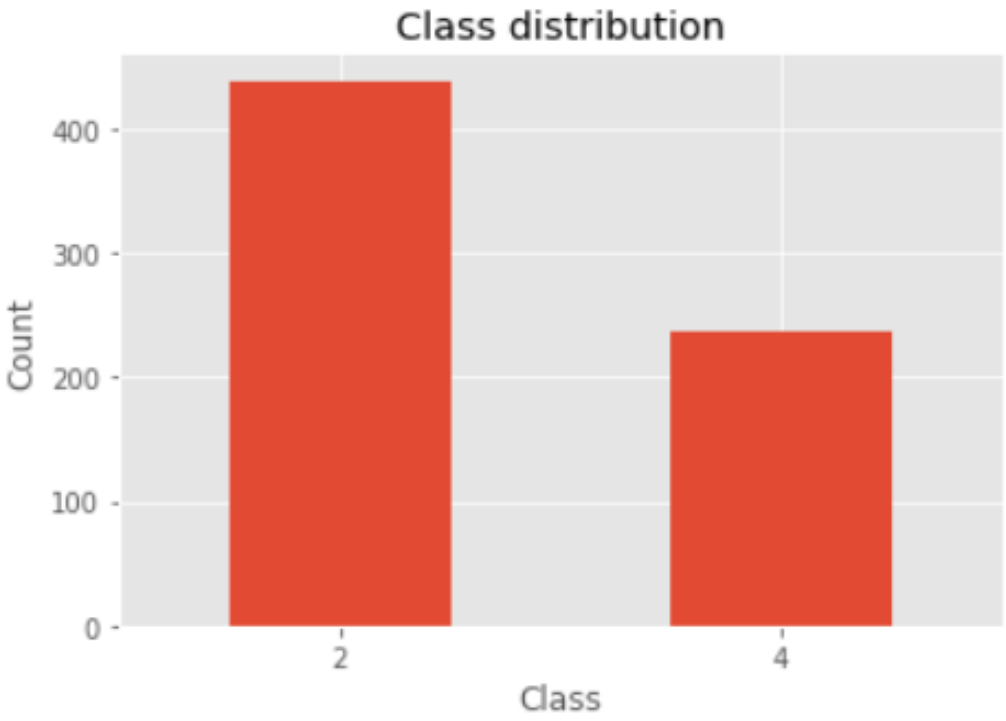
- Najbardziej skorelowane z atrybutem klasy są: deg-malig, node-caps, inv-nodes. Jest to jednak korelacja na poziomie 0.3, dodatkowo trzeba pamiętać że korelacja nie oznacza przyczynowości.
- Do trenowania niektórych modeli wymagane będzie użycie Samplera, ze względu na silną przewagę zdrowych pacjentów w zbiorze danych.
- Modele trenowane będą przy użyciu jedynie najbardziej znaczących atrybutów.

1.3.2 Breast-Cancer-Wisconsin

Przeprowadźmy analizę podobnego zbioru danych w celu stworzenia punktu odniesienia. Pierwsze, co rzuciło się w oczy przy analizie danych to ich ilość. Mamy do czynienia z 699 wierszami reprezentujących pacjentów. Jest to znacznie więcej niż w przypadku poprzedniego zbioru danych, w związku z czym spodziewamy się dużo lepszych wyników modeli uczonych na breast-cancer-wisconsin.

Tylko jeden atrybut jest obarczony brakami wartości - łącznie brakuje 16 rekordów. W tym przypadku możemy postąpić analogicznie do zbioru danych breast-cancer i przyjąć, że braki są MCAR oraz zastąpić je przez najczęściej występującą wartość atrybutu. Oprócz tego, tylko 8 rekordów jest zduplokowanych - co w skali całego zbioru danych jest czynnikiem pomijalnym.

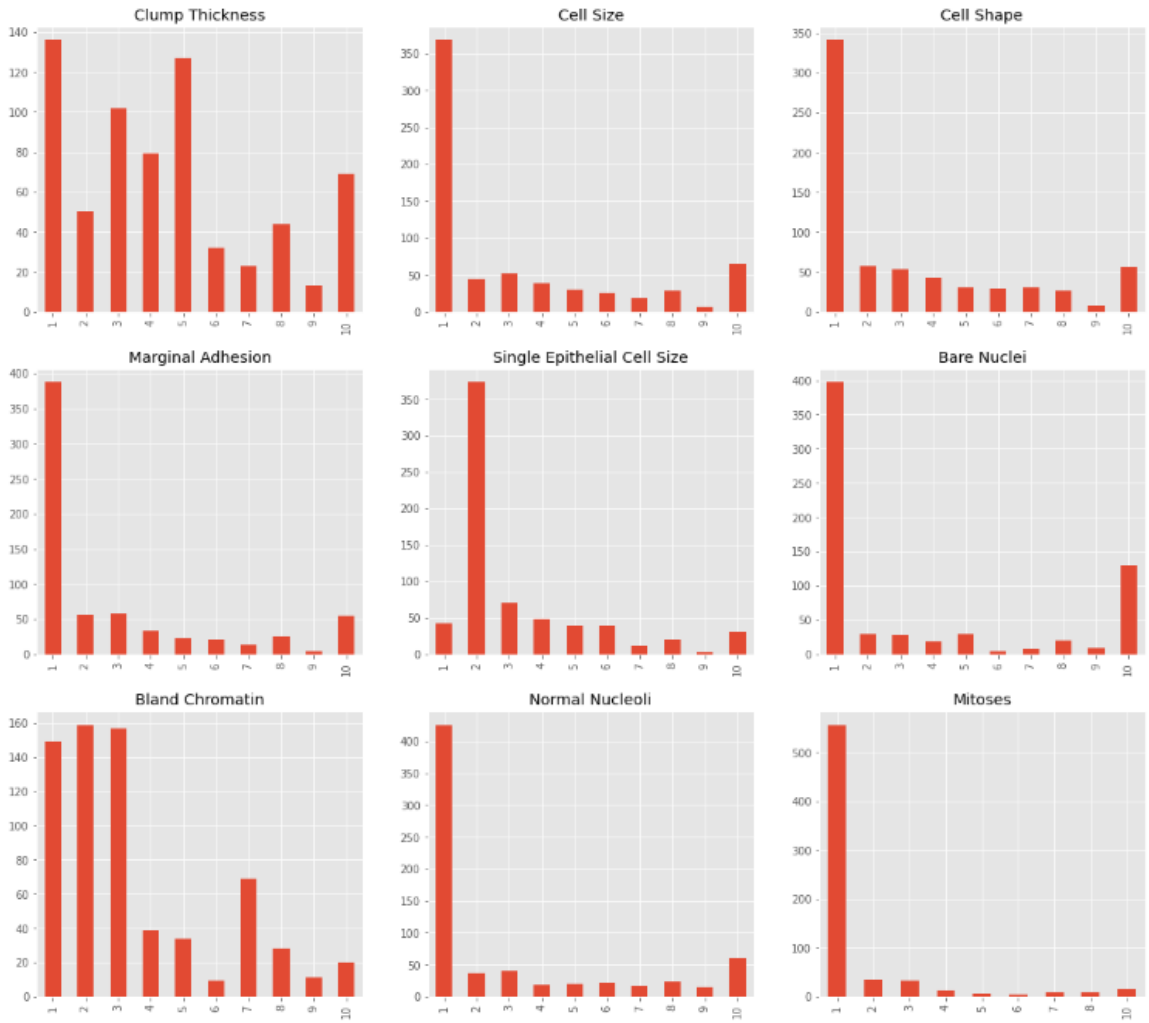
Zwizualizujmy dysproporcje w zbiorze danych.



Rysunek 5: Dysproporcje między klasami

Widzimy znacznie lepsze rozłożenie klas niż w przypadku zbioru danych 'breast-cancer'. W dalszym ciągu, jest dwa razy więcej pacjentów z klasą 0 (łagodny rak) niż z klasą 1 (rak złośliwy), ale ilość danych rekompensuje nam tę dysproporcję. W dalszym ciągu, przy uczeniu modeli dalej może być wymagane użycie samplera, ale tę decyzję projektową zostawimy na później - przy okazji procesu analizy modeli.

Przeanalizujmy teraz częstość występowania atrybutów i ich związki z klasą pacjenta.

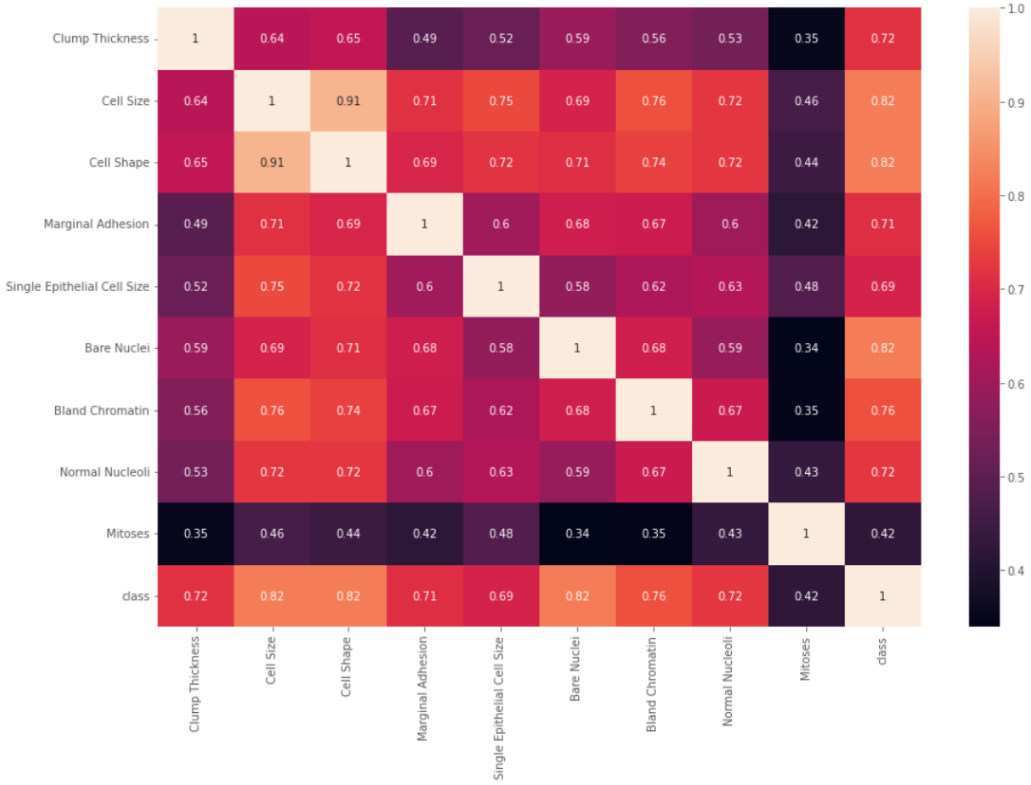


Rysunek 6: Stosunek wartości atrybutu do częstości występowania.

W przypadku wielu atrybutów jest zauważalna znaczna dysproporcja pomiędzy występującymi wartościami. W przypadku przeważającej części atrybutów (Normal Nucleoi, Mitosis, Marginal adhesion, Bare nuclei, Cell size, Cell Shape), 90% wszystkich wartości stanowi 1. Może to stanowić problemy przy uczeniu modeli i wymaga dodania normalizacji do preprocessingu.

Na tym etapie ponownie została stworzona funkcja do preprocessingu danych pełniąca analogiczne funkcje.

Po normalizacji danych możemy przejść do głębszej analizy zależności pomiędzy atrybutami.

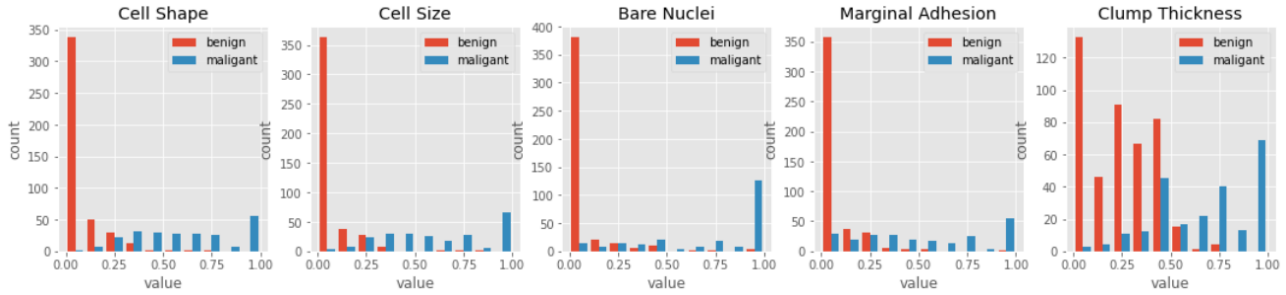


Rysunek 7: Tabela Korelacji Atrybutów

Widzimy silną zależność pomiędzy atrybutami cell shape i cell size. Jest to raczej logiczne - rozmiar komórki jest zależny od jej kształtu.

Najsilniej skorelowane z atrybutem klasy są Cell Size, Cell Shape i Bare Nuclei. Złośliwość nowotworu jest zwykle związana z rozmiarem komórki, co uzasadnia korelację Cell Size i Cell shape z klasą. Bare nuclei odnosi się do jądra nieotoczonego cytoplazmą (resztą komórki) - zjawisko to zwykle wiąże się z niezłośliwością nowotworu, co uzasadnia tę korelację. Najsłabiej skorelowany z atrybutem klasy jest atrybut Mitoses - liczba komórek dzielących się, widocznych w histopatologii barwionej hematoksyliną i eozyną (H&E). Zwykle jest to dobry wyznacznik agresywności guza, ale w przypadku naszego zbioru danych nie stanowi on aż tak dobrego wyznacznika jak pozostałe atrybuty.

Porównanie liczności klas z wartościami silnie skorelowanych z nią atrybutów



Rysunek 8: Wykresy liczności

Na powyższym wykresie jeszcze dokładniej widoczne są granice między wartościami atrybutów a przyporządkowaną klasą. Jest to zjawisko pożądane -oznacza, że istnieje prawie liniowa granica między klasami wyznaczona pomiędzy parami atrybutów, a co za tym idzie modele powinny dobrze radzić sobie z klasyfikacją.

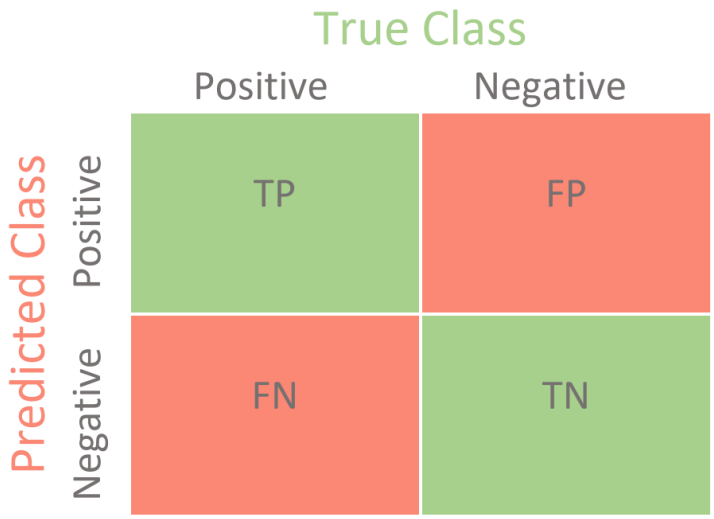
Wnioski z analizy

- Ten zbiór danych wydaje się być znacznie lepszy niż breast_cancer z wielu powodów: (liczność danych, mniej powtarzających się wierszy, mniej wartości brakujących, Wszystkie atrybuty da się znormalizować, Zależności między atrybutami a klasą są silne a granice podziału wyraźne)
- Najsilniej skorelowane z atrybutem klasy są atrybuty Cell Size, Cell Shape oraz Bare Nuclei
- W zbiorze danych dalej występują dysproporcje, ale przez liczność danych nie powinno stanowić to aż tak dużego problemu jak w przypadku zbioru danych breast_cancer

1.4 Miary jakości modelu

1.4.1 Macierz pomyłek

Macierz zawierająca 4 wartości mówiące o tym jak model poradził sobie z klasyfikacją danych



Rysunek 9: Macierz Pomyłek

Na podstawie macierzy pomyłek obliczane poniższe miary jakości.

1.4.2 Accuracy

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP}$$

Accuracy niesie informację o tym, jaki procent próbek testowych został poprawnie sklasyfikowany przez model.

1.4.3 Recall(Sensitivity)

$$Recall = Sensitivity = \frac{TP}{TP + FN}$$

Recall mówi nam, jak model radzi sobie z klasyfikowaniem przypadków pozytywnych danej klasy.

1.4.4 Specificity

$$Specificity = \frac{TN}{TN + FP}$$

Specificity mówi nam, jak model radzi sobie z klasyfikowaniem przypadków negatywnych danej klasy.

1.4.5 Precision

$$Precision = \frac{TP}{TP + FP}$$

Precision mówi nam, w jakich proporcjach model klasyfikuje próbki jako pozytywne w zależności od faktycznej klasy próbki.

1.4.6 Miara F1

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

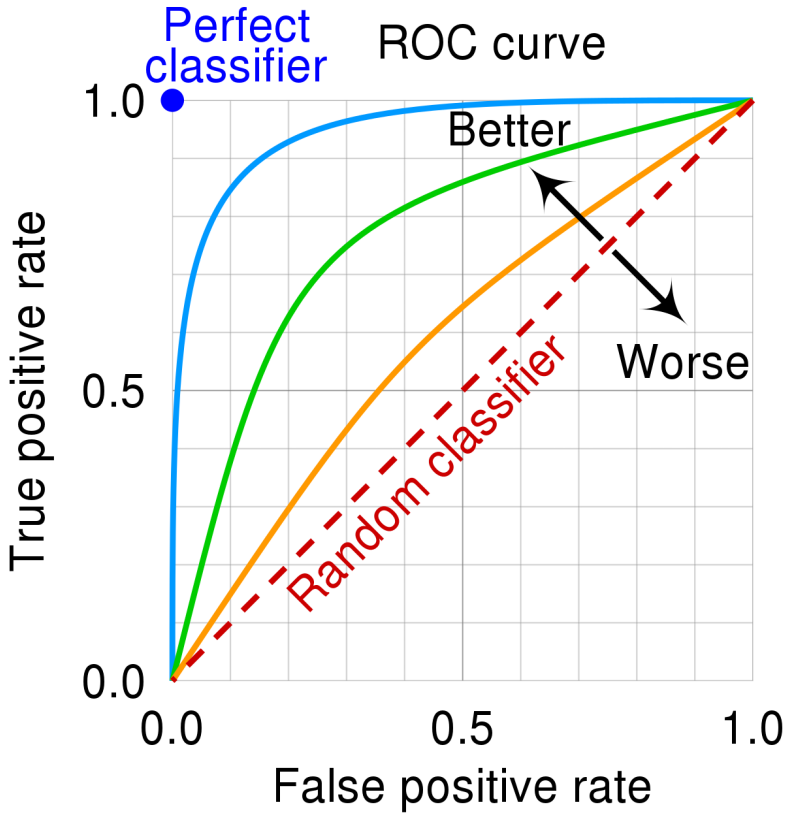
F1 Score mówi nam, jak dobrze model radzi sobie z klasyfikacją przypadków TP, przy tym minimalizując liczbę przypadków FP i FN.

1.4.7 Support

$$Support = TP + FN$$

Support mówi nam ile w danych ewaluacyjnych jest osób posiadających rekursywne zdarzenia.

1.4.8 Krzywe ROC i AUC



Rysunek 10: Krzywa ROC

Krzywa ROC pokazuje nam zależność TPR(true positive rate) od FPR(false positive rate).

$$TPR = Recall = Sensitivity = \frac{TP}{TP + FN}$$

$$FPR = 1 - Specificity = \frac{FP}{FP + TN}$$

Rysuje się ją poprzez sprawdzenie, jak algorytm klasyfikuje przypadki dla wybranych poziomów cutoff. Cutoff to liczba z zakresu [0, 1] która definiuje które predykcje modeli są klasyfikowane jako klasa pozytywna lub negatywna. AUC to pole pod wykresem narysowanym w powyższy sposób.

1.5 Założenia odnośnie metryk oceny

W naszym zadaniu samo Accuracy nie będzie dobrym wyznacznikiem jakości modelu. Ważne będzie, aby przy ocenie wziąć pod uwagę Sensitivity oraz Specificty. To, aby ustalić która z tych dwóch metryk jest ważniejsza, musi odpowiedzieć pytanie czy jesteśmy bardziej skłonni dopuścić do klasyfikacji przypadków fałszywie negatywnych, czy fałszywie pozytywnych. Osobiście zdaje mi się, że większe straty ponosimy w przypadku klasyfikacji FP, przypadki FN zawsze można wykluczyć w dogłębnym badaniu, na przykład sięgając po opinię specjalistów lub innego algorytmu. Dlatego zdaje mi się, że Recall będzie w tym problemie ważniejszą metryką. Dodatkowo dobrymi miarami do porównania algorytmów będzie miara F1 oraz krzywe ROC i AUC.

1.6 Modele badane miarami jakości

1.6.1 Model losowy

Model losowy zwracający wartość losową z zakresu [0, 1], oznaczającą prawdopodobieństwo sklasyfikowania osobnika jako przynależnego do klasy 1 (wystąpiły zdarzenia rekurencyjne). Zakładamy domyślny cutoff = 0.5, z tego wynika, że celność modelu powinna wahać się w granicy 50%. Można było również przyjąć model zwracający zawsze wartość 0 (pacjent zdrowy), co przez dysproporcje w zbiorze danych dawałoby wyższą celność, ale zdecydowaliśmy się użyć losowości by zasymulować podejmowanie decyzji.

Zalety:

- Dobry baseline do porównywania jakości innych modeli.

Wady:

- Oczywista, niska jakość klasyfikatora.

1.6.2 Naiwny Bayes

Naiwny klasyfikator Bayesowski to metoda klasyfikacji, która opiera się na teorii Bayesa. Pozwala ona na przypisanie nowych obiektów do jednej z kilku klas na podstawie cech, które posiadają.

Działanie naiwnego klasyfikatora Bayesowskiego polega na wyznaczeniu prawdopodobieństwa przynależności nowego obiektu do każdej z klas na podstawie jego cech. **Klasyfikator zakłada, że cechy obiektów są niezależne od siebie**, co może być uproszczeniem, ale w wielu przypadkach działa wystarczająco dobrze.

Aby wyznaczyć prawdopodobieństwo przynależności nowego obiektu do danej klasy, naiwny klasyfikator Bayesowski korzysta z wcześniej zebranych danych treningowych, w których już zostały przyporządkowane klasy dla innych obiektów o znanych cechach. Dzięki temu można wyznaczyć, jakie cechy są typowe dla danej klasy i jakie prawdopodobieństwo przynależności do niej mają obiekty posiadające te cechy. W celu uproszczenia stworzenia modelu wykorzystaliśmy GaussianNB z modułu *sklearn.naive_bayes*.

Zalety:

- Jest znacznie szybszy od pozostałych klasyfikatorów zarówno pod względem uczenia jak i dokonywania predykcji.
- Dobrze radzi sobie z wielowymiarowymi zbiorami danych.

Wady:

- Zakłada, że atrybuty są od siebie niezależne, co nie jest prawdą dla naszych zbiorów danych i może obniżać jakość klasyfikatora.

1.6.3 Random Forest

Główną ideą Random Forest jest połączenie wielu drzew decyzyjnych, które są trenowane na różnych podzbiorach danych treningowych, wykorzystując losowe próbkowanie ze zwracaniem. Każde drzewo w lesie ma swoją niezależną losowość, co oznacza, że losuje tylko pewną liczbę cech spośród wszystkich dostępnych. W ten sposób tworzone są różnorodne drzewa, które uzupełniają się nawzajem.

W drzewie decyzyjnym kolejne warunki są reprezentowane jako węzły, a decyzje jako gałęzie, które łączą węzły. Drzewo zaczyna się od węzła nadrzędnego, nazywanego korzeniem, który reprezentuje główne pytanie.

Z każdego węzła wychodzą gałęzie prowadzące do kolejnych węzłów, które reprezentują różne możliwe decyzje lub wyniki. Przechodząc w dół po drzewie, na każdym etapie podejmujemy decyzje na podstawie cech lub zmiennych, aż osiągniemy końcowy węzeł, który reprezentuje ostateczną decyzję lub w naszym przypadku - klasę. W celu uproszczenia stworzenia modelu wykorzystaliśmy RandomForestClassifier z modułu *sklearn.ensemble.RandomForestClassifier*.

Zalety:

- Wysoce dokładne prognozowanie: Las losowy jest znany z wysokiej dokładności prognozowania zarówno w zadaniach klasyfikacji, jak i regresji. Może efektywnie radzić sobie z danymi o dużym rozmiarze i złożoności.
- Odporność na nadmierne dopasowanie (overfitting): Dzięki zastosowaniu wielu drzew decyzyjnych i losowaniu podzbiorów cech dla każdego drzewa, RandomForest jest mniej podatny na nadmierne dopasowanie niż reszta testowanych klasyfikatorów.

Wady:

- Brak interpretowalności: W przypadku dużych zestawów drzew decyzyjnych, zrozumienie konkretnego wyniku predykcyjnego może być trudne
- Wymagane zasoby obliczeniowe: Tworzenie lasu losowego i prognozowanie na podstawie wielu drzew może być czasochłonne, szczególnie w przypadku dużych zestawów danych.

1.6.4 XGBoost

Gradient boosting to technika uczenia maszynowego, która polega na budowaniu sekwencji słabych modeli predykcyjnych i łączeniu ich w silny model. Słabe modele są trenowane iteracyjnie w celu minimalizacji funkcji kosztu, a każdy kolejny model dostosowuje się do reszt, jakie pozostawiają poprzednie modele.

XGBoost implementuje tę technikę w sposób zoptymalizowany pod kątem wydajności i skuteczności. W przeciwieństwie do innych metod gradient boosting, XGBoost wykorzystuje regresję logistyczną jako funkcję kosztu, co zapewnia stabilność procesu uczenia. W celu uproszczenia stworzenia modelu wykorzystaliśmy XGBoost z modułu *xgboost.xgb*.

Zalety:

- Wysoce efektywny.
- Skuteczność w rozwiązywaniu skomplikowanych problemów: XGBoost radzi sobie dobrze z problemami o wysokiej złożoności, z dużą liczbą cech, dużymi zestawami danych i nieliniowymi zależnościami między cechami.
- Odporność na overfitting przez wbudowane mechanizmy regularyzacji.

Wady:

- Konieczność dostosowania hiperparametrów: XGBoost ma wiele hiperparametrów, które należy dostosować.
- Zależność od jakości danych.

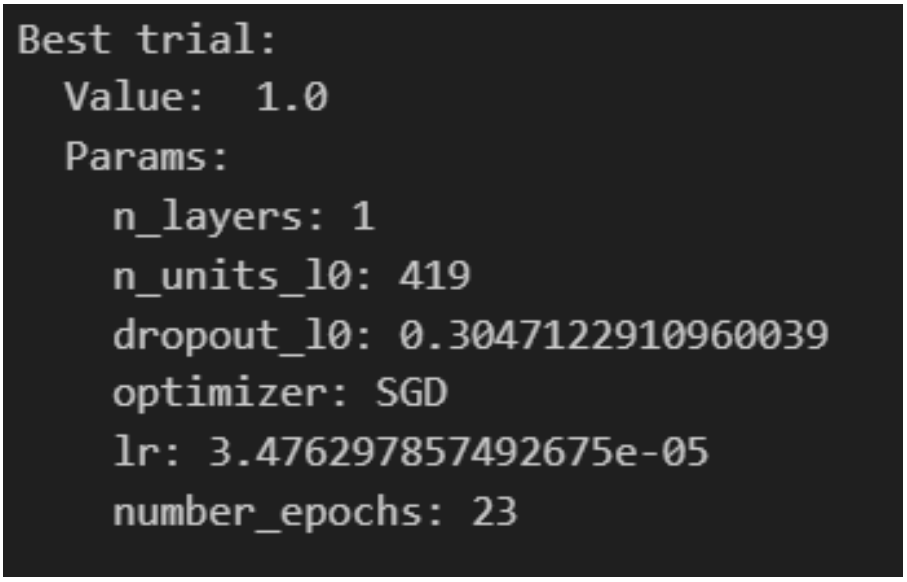
W celu uzyskania modelu o jak najwyższej jakości, przed wdrożeniem zostało przeprowadzone strojenie hiperparametrów (dla obydwu zbioru danych), które zawarte jest w pliku *___xgboost.ipynb*. Poniżej zestawiona jest lista najoptymalniejszych parametrów znaleziona w procesie dla zbioru danych breast-cancer:

```
Best trial:
  Value:  0.8023255813953488
  Params:
    max_depth: 9
    learning_rate: 0.021293532451690887
    n_estimators: 366
    min_child_weight: 2
    gamma: 2.5049861036687497e-06
    subsample: 0.16750648356581507
    colsample_bytree: 0.04229281082115818
    reg_alpha: 3.969227080574277e-07
    reg_lambda: 3.744263342403242e-05
```

Rysunek 11: Parametry XGBoost

1.6.5 Sieć neuronowa

W celu uproszczenia zdecydowaliśmy się na implementację własnej sieci neuronowej. W celu optymalizacji i zapobieganiu przeuczeniu - pomiędzy warstwami ukrytymi wykorzystaliśmy metodykę dropoutu oraz batchnormu. Ich szczegóły jak i inne parametry sieci takie jak liczba warstw ukrytych, czy optymalizator zostały ustalone przy pomocy strojenia hiperparametrów, a najlepsze modele zostały zapisane. Poniżej zestawiona jest lista najoptymalniejszych parametrów znaleziona w procesie dla zbioru danych breast-cancer:



Rysunek 12: Parametry Sieci neuronowej

Zalety:

- Wysoce adaptacyjne: Sieci neuronowe są zdolne do adaptacji i uczenia się złożonych i nieliniowych zależności w danych.
- Wysoce adaptacyjne: Sieci neuronowe są zdolne do adaptacji i uczenia się złożonych i nieliniowych zależności w danych.

Wady:

- Wymagane zasoby obliczeniowe.
- Potrzeba dużej ilości danych treningowych - co dla naszego problemu może okazać się krytyczne.

1.6.6 Support Vector Machines

SVM to algorytm który dąży do odnalezienia hiperpłaszczyzny która najlepiej odseparowuje dane należące do obydwu klas. Hiperpłaszczyzna ta ma maksymalizować margines, czyli minimalną odległość między hiperpłaszczyzną, a najbliższymi punktami z obu klas.

Zalety:

- Skuteczność w przestrzeni o wysokiej wymiarowości
- Efektywność w przypadku małych zbiorów danych treningowych - co będzie korzystne dla naszych zbiorów danych.

Wady:

- Złożoność obliczeniowa: Trenowanie SVM, szczególnie w przypadku użycia funkcji jądra nieliniowego, może być zasobożerne obliczeniowo.

- Trudność interpretacji: SVM może być trudny do interpretacji, szczególnie w przypadku wykorzystania funkcji jądra.

2 Opis Funkcjonalny

Poniżej przedstawiona zostanie struktura projektu.

2.1 Dane

Wszystkie modele klasyfikujące raka piersi, są trenowane a następnie sprawdzane przy użyciu danych, opisanych w sekcji Dostępne Dane.

Do reprezentacji danych w projekcie używamy abstrakcji w postaci klasy Data. W niej jako atrybuty przechowywane są surowe dane jak i obrobione dane, wymagane przez niektóre z modeli. W wyniku obróbki danych atrybuty nominalne zakodowane są za pomocą one-hot encoding. Dodatkowo uprościliśmy dane reprezentowane przez zakres do pojedynczej wartości będącej środkiem przedziału. Następnie wszystkie tak obrobione dane zostały znormalizowane do zakresu [0, 1].

Analiza danych pod kątem atrybutów została przeprowadzona na podstawie pliku:

data_analysys.ipynb

znajdującego się w odpowiednim folderze dla obydwu zbiorów danych.

2.2 Modele

Wszystkie modele w projekcie dziedziczą po abstrakcyjnej klasie BinaryClassificationModel, wprowadzającej wygodny w użyciu interfejs do późniejszej analizy i oceny jakości modeli. Wszystkie modele wewnętrznie wyznaczają miary oceny jakości a następnie przechowują je w atrybutach, do których w prosty sposób można się dostać. Posiadają one również metody służace do rysowania confusion matrix oraz ROC curve.

W projekcie zaimplementowane zostały następujące modele, znajdujące się w plikach odpowiadającym ich nazwom:

1. Model losowy
2. Naiwny Bayes
3. Sieć neuronowa
4. Random Forest
5. XGBoost
6. Support Vector Machine

2.3 Main i MainWisconsin

W notatniku *main.ipynb* oraz *main-wisconsin.ipynb* znajdują się najważniejsze części projektu, łączące ze sobą wszystkie modele i przedstawiające wyniki analizy miar jakości testowanych modeli dla obydwu zbiorów danych.

2.4 ModelEvaluator

Jest klasą, która wylicza metryki oceny jakości modelu dla podanego modelu oraz wizualizuje niektóre z nich jak macierz pomyłek czy krzywe ROC.

2.5 Testy jednostkowe

Stworzyliśmy testy jednostkowe w celu przetestowania działania naszych klas oraz metod. Dodatkowo w trakcie pisania testów okazało się, że nie obsłużyliśmy błędu polegającego na tym że dla niektórych danych, niektóre funkcje obliczające metryki zwracały błąd DevisionByZeroError, obsłużyliśmy błędy, gdy miałoby dojść do dzielenia przez zero to wtedy metryka nie jest obliczana, pozostaje Nullem. Testy jednostkowe dodatkowo dobrze pokazują jak należy korzystać z klas które przygotowaliśmy. Testy znajdują się w plikach test_BinaryClassificationModel.py i data/test_Data.py.

3 Opis metryk jakości

	model	accuracy	recall(sensitvity)	specificity	precision	f1_score	support	auc	execution time
0	RandomModel	0.476744	0.44	0.491803	0.261905	0.328358	25	0.391803	0.000000
1	NaiveBayes	0.720930	0.44	0.836066	0.523810	0.478261	25	0.726885	0.003001
2	NeuralNetwork	0.837209	0.72	0.885246	0.720000	0.720000	25	0.849508	14.137046
3	RandomForest	0.732558	0.44	0.852459	0.550000	0.488889	25	0.672459	0.554088
4	XGBoost	0.802326	0.36	0.983607	0.900000	0.514286	25	0.822951	2.151674
5	SupportVectorMachine	0.790698	0.36	0.967213	0.818182	0.500000	25	0.694754	0.010015

Rysunek 13: Przykładowa tabela metryk

3.1 Accuracy

Metryka Accuracy jest najprostszym i najbardziej intuicyjnym wyznacznikiem jakości modelu. Oznacza ona jaki procent obiektów ze zbioru testowego został poprawnie sklasyfikowany przez model. Ta metryka to dobry wyznacznik jakości modelu, który w wielu przypadkach wystarcza do wybrania najlepszego modelu, chociaż należy pamiętać, że w naszym przypadku zbiór danych jest bardzo ubogi i faktyczne Accuracy osiągnięte przez model może się różnić od uśrednionego Accuracy osiąganego przez ten model na tym zbiorze danych nawet o kilkanaście procent.

3.2 Recall

Recall mówi nam o tym, jak model radzi sobie z klasyfikowaniem przypadków pozytywnych danej klasy. Jest to szczególnie ważna w naszym problemie metryka, ponieważ zależy nam najbardziej na wykryciu raka wśród pacjentów, którzy faktycznie są w grupie ryzyka.

3.3 Specificity

Specificity mówi nam, jak model radzi sobie z klasyfikowaniem przypadków negatywnych danej klasy. W przypadku naszego problemu jest to metryka mniej istotna od Recall, nie mniej jednak powinniśmy dążyć do osiągnięcia modelu, który poprawnie klasyfikuje osoby zdrowe. Ponownie, model losowy osiąga wartość ok. 50% co pokrywa się z oczekiwaniami, widzimy jednak, że pozostałe modele bardzo dobrze radzą sobie z rozpoznawaniem osób zdrowych - osiągają Specificity ok. 90%. Jest to ponownie związane z ubogim zbiorem treningowym i dysproporcją klas - dostajemy więcej przypadków osób zdrowych niż tych, którzy znajdują się w grupie ryzyka, przez co modele uczą się precyzyjniej rozpoznawać zdrowych pacjentów.

3.4 Precision

Precision mówi nam o tym, jaki procent przypadków klasy przewidzianej jako pozytywna jest faktycznie pozytywna- czyli ile procent osób które zaklasyfikowaliśmy jako chore jest faktycznie chore. Nie jest to bardzo ważna metryka. Zależy nam bardziej na tym, żeby klasyfikować osoby zagrożone, niż błędnie klasyfikować osoby zdrowe jako znajdujące się w grupie ryzyka). Nie mniej jednak powinno nam również zależeć na tym, aby osoby zdrowe jak najrzadziej były klasyfikowane jako zagrożone.

3.5 F1 Score

F1 Score mówi nam, jak dobrze model radzi sobie z klasyfikacją przypadków TP, przy tym minimalizując liczbę przypadków FP i FN. Oznacza to, że modele z wyższym F1 lepiej klasyfikują osoby w grupie ryzyka, popełniając przy tym mniej błędnych klasyfikacji.

3.6 Support

Support jest metryką, która odnosi się bardziej do zbioru danych testowych niż do samych modeli. Informuje nas o tym, ile wśród wszystkich danych testowych było osobników chorych - w grupie ryzyka, więc nie powinno nas dziwić to, że dla każdego modelu support jest taki sam.

3.7 Execution time

Mówi nam jak długo algorytm uczył się dopasowania do zbioru danych w sekundach.

4 Analiza wyników dla zbioru danych Breast Cancer Data Set

Breast Cancer Data Set [LINK]

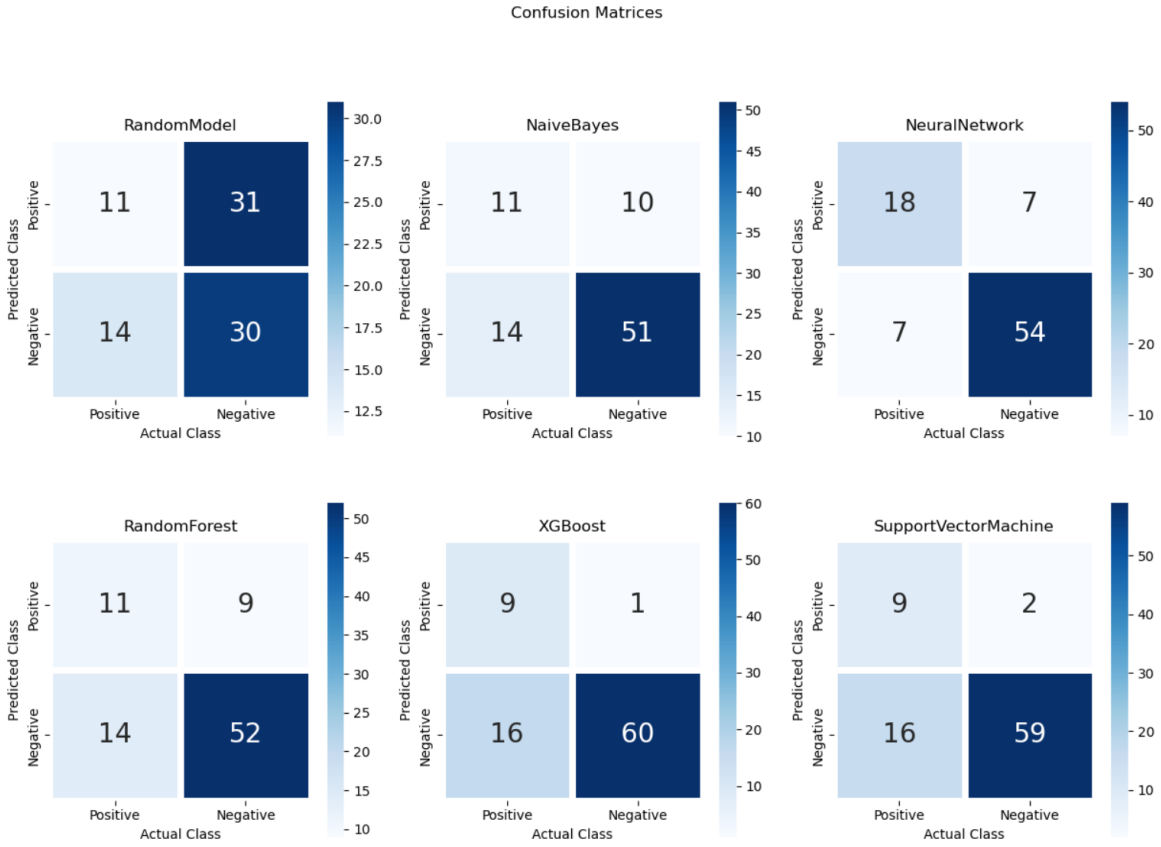
4.1 Wstępne założenia

W naszym zadaniu samo Accuracy nie będzie dobrym wyznacznikiem jakości modelu. Ważne będzie, aby przy ocenie wziąć pod uwagę Sensitivity oraz Specificty. To, aby ustalić która z tych dwóch metryk jest ważniejsza, musi odpowiedzieć pytanie czy jesteśmy bardziej skłonni dopuścić do klasyfikacji przypadków fałszywie negatywnych, czy fałszywie pozytywnych. Osobiście zdaje mi się, że większe straty ponosimy w przypadku klasyfikacji FP, przypadki FN zawsze można wykluczyć w dogłębnym badaniu, na przykład sięgając po opinię specjalistów lub innego algorytmu. Dlatego zdaje mi się, że Recall będzie w tym problemie ważniejszą metryką. Dodatkowo dobrymi miarami do porównania algorytmów będzie miara F1 oraz krzywe ROC i AUC, co wynika z ich natury.

4.2 Wpływ danych na wyniki

Niestety, zbiór danych na którym operujemy jest mały, zaledwie 286 obserwacji. Dlatego można się spodziewać, ze wariancja wyników będzie duża. Możemy również spodziewać się zjawiska przetrenowania modelów takich jak sieć neuronowa czy XGBoost. Pomimo tego, że osiągają one dobre wyniki na zbiorze treningowym - w rzeczywistym zastosowaniu na zbiorze testowym radzą sobie średnio. Kolejną negatywną cechą wynikającą z ubogiego zbioru danych jest mała powtarzalność przeprowadzonych eksperymentów przez duże dysproporcje w wylosowanych zbiorach treningowym i testowym.

4.3 Macierze pomyłek



Rysunek 14: Macierze pomyłek

Macierze pomyłek przedstawiają nam wyniki predykcji naszych modeli. Na ich podstawie możemy wyliczać wcześniej opisane metryki. Rzucającą się w oczy rzeczą jest to, że dla każdego modelu najwięcej jest przypadków true negative, wiąże się to z tym, że zbiór danych jest niezbalansowany - jest więcej przypadków negatywnych klasy docelowej, do których modele (pomimo Samplera) mocniej się przystosowywują.

4.4 Analiza wyników

Dane ze zbioru zostały podzielone na zbiór treningowy i testowy w proporcji 7 : 3. Na 286 wierszy danych daje nam to 200 wierszy, na których uczone są nasze modele. Jest to znacznie za mało danych aby wytrenowane modele były stabilne i zawsze osiągnęły zadowalający wynik, ponieważ w przypadku chęci nauczenia modelu przez powielanie danych, może szybko zacząć się on przeuczać.

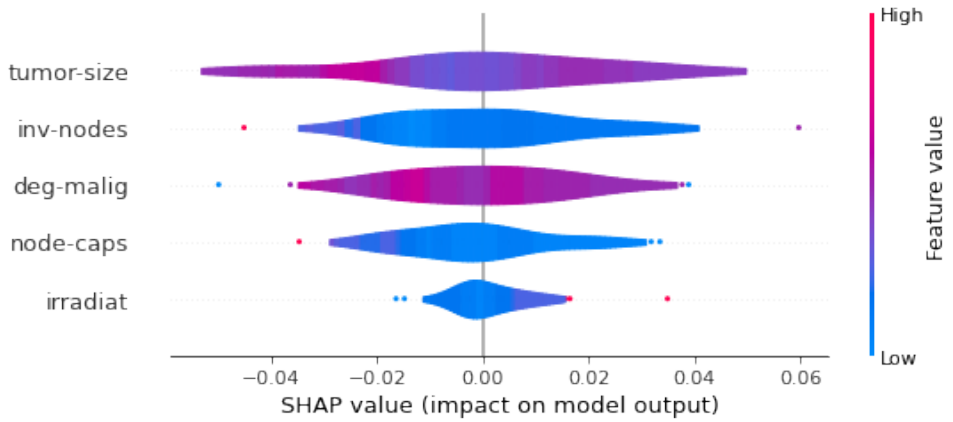
4.5 Analiza wyników poszczególnych algorytmów

	model	accuracy	recall(sensitvity)	specificity	precision	f1_score	support	auc	execution time
0	RandomModel	0.476744	0.44	0.491803	0.261905	0.328358	25	0.391803	0.000000
1	NaiveBayes	0.720930	0.44	0.836066	0.523810	0.478261	25	0.726885	0.003001
2	NeuralNetwork	0.837209	0.72	0.885246	0.720000	0.720000	25	0.849508	14.137046
3	RandomForest	0.732558	0.44	0.852459	0.550000	0.488889	25	0.672459	0.554088
4	XGBoost	0.802326	0.36	0.983607	0.900000	0.514286	25	0.822951	2.151674
5	SupportVectorMachine	0.790698	0.36	0.967213	0.818182	0.500000	25	0.694754	0.010015

Rysunek 15: Tabela metryk

4.5.1 RandomModel

Model losowy działa - losowo, posłuży on nam jako model podstawowy, jeśli jakiś algorytm pod względem danych metryk radzi sobie gorzej niż model losowy, możemy uznać go za bardzo słaby.



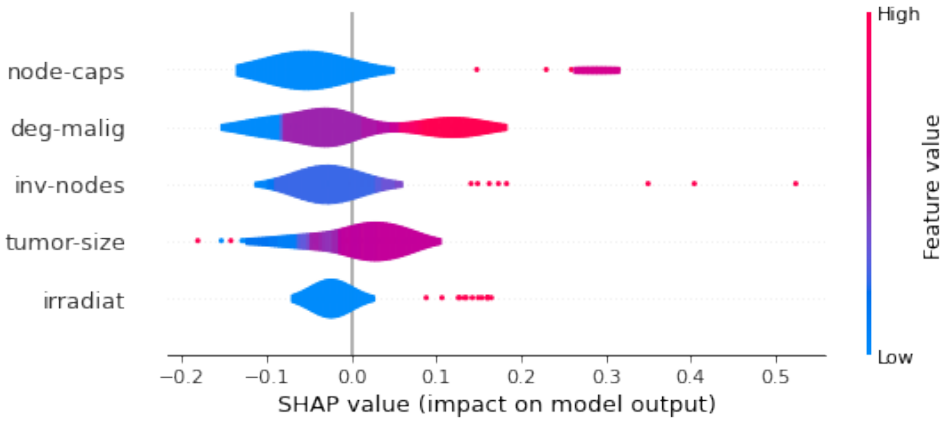
Rysunek 16: RandomModel: Wpływ wartości atrybutów na predykcje modelu

Analiza SHAP w przypadku modelu losowego nie niesie za sobą głębszego sensu, ponieważ atrybuty nie mają żadnego znaczenia przy odpowiedzi modelu.

4.5.2 NaiveBayes

Naiwny Bayes zakłada niezależność atrybutów, co w przypadku naszego problemu może znacznie pogarszać wyniki modelu, który nie będzie w stanie wykryć skomplikowanych zależności między atrybutami, przez co gorzej poradzi sobie z klasyfikacją. Pomimo tego, model poradził sobie dość dobrze. Działa lepiej niż strategia naiwna polegająca na klasyfikowaniu wszystkich przypadków jako 0. Jest porównywalny zarówno pod względem accuracy jak i recall. Naiwny Bayes klasyfikuje poprawnie około połowę pacjentów z rakiem jako chorych. Nie jest to wynik zadowalający. Z zalet modelu - jest bardzo szybki jeśli chodzi o prędkość obliczeń. Ten model

również posłuży nam jako baseline, jeśli jakiś algorytm będzie radził sobie gorzej pod względem recall i specificity od Naiwnego Bayesa, to daje nam sygnał że model nie poradzi sobie z zadaniem.

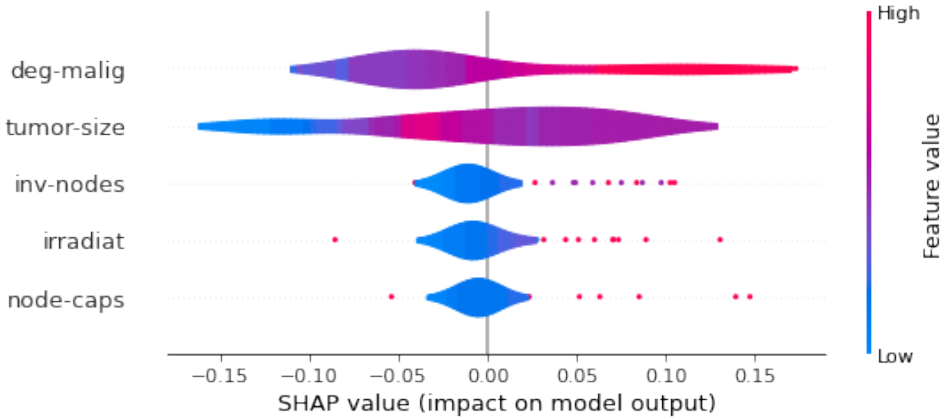


Rysunek 17: NaiveBayes: Wpływ wartości atrybutów na predykcje modelu

Wykres Shap wskazuje nam że największy wpływ na predykcje modelu NaiveBayes miały atrybuty node-caps, deg-malig i inv-nodes, co pokrywa się z założeniami wynikającymi ze wstępnej analizy danych.

4.5.3 NeuralNetwork

Sieć neuronowa, poradziła sobie zadowalająco z problemem, pomimo słabej jakości danych. Przy małym zasobie czasowym (15s) wynikającym z uczenia się sieci neuronowej otrzymaliśmy model który poprawnie klasyfikuje 84% przypadków. Dodatkowo, model osiąga wysokie wartości metryk recall (72%) oraz specificity (89%). Model ten wyróżnia się dodatkowo najlepszym wynikiem f1 i auc, świadczy to o najlepszej jakości klasyfikacji uzyskanych za pomocą tego modelu. Niski precision w porównaniu do XGBoost i SVM jest ceną jaką ponosimy za wysoki recall - nie ma to natomiast dużego znaczenia przy ogólnej ocenie jakości modelu, gdyż na tle innych modeli sieć neuronowa najczęściej przydziela klasę 1. Powoduje to częstrze występowanie przypadków FP, ale za to zwiększa też przypadki TP, które są głównym wyznacznikiem jakości modelu na tym zbiorze danych.

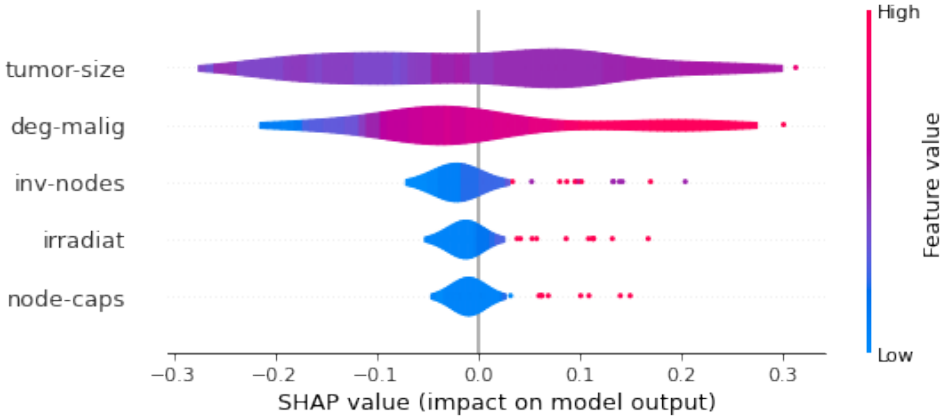


Rysunek 18: NeuralNetwork: Wpływ wartości atrybutów na predykcje modelu

Analiza wykresu mówi nam, że duży wpływ na predykcje sieci neuronowej miał atrybut deg-malig oraz tumor-size. Ponownie pokrywa się to z wnioskami uzyskanymi na podstawie wstępnej analizy dostępnych danych, jedynym odstępstwem jest słabe znaczenie atrybutu node-caps, na którym model nie skupiał się podczas klasyfikacji.

4.5.4 RandomForest

Las losowy nie wyróżnia się niczym na tle pozostałych modeli. Jakość predykcji jest zbliżona do modelu losowego, czy Naiwnego Klasyfikatora Bayesowskiego.



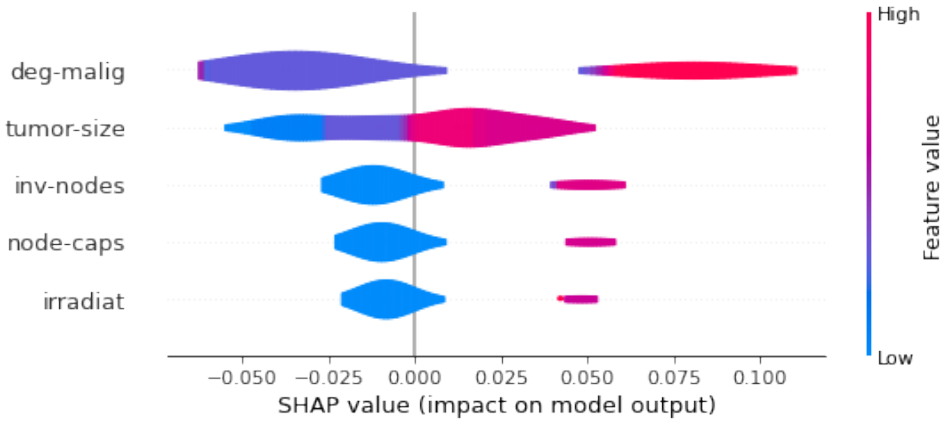
Rysunek 19: RandomForest: Wpływ wartości atrybutów na predykcje modelu

Podobnie w przypadku sieci neuronowej, najistotniejsze dla modelu atrybuty to deg-malig oraz tumor-size.

4.5.5 XGBoost

XGBoost pod względem accuracy poradził sobie porównywalnie dobrze co sieć nieuronowa. XGBoost charakteryzuje się najwyższym na tle pozostałych modeli specificity - najlepiej kla-

syfikuje osoby w grupie zagrożenia jako faktycznie zagrożone. Z drugiej strony najniższa ze wszystkich modeli wartość recall pokazuje, że XGBoost słabo radzi sobie z klasyfikacją osób zagrożonych. W przypadku naszego problemu jest to silnie niepożądane. Wysokie precision i niski recall mówi nam o tym, iż XGBoost klasyfikuje pacjentów jako chorych tylko wtedy, gdy jest tego pewien. Skutkuje to w klasyfikowaniu wielu chorych jako zdrowych. To zjawisko skutkuje w niskim F1 score.

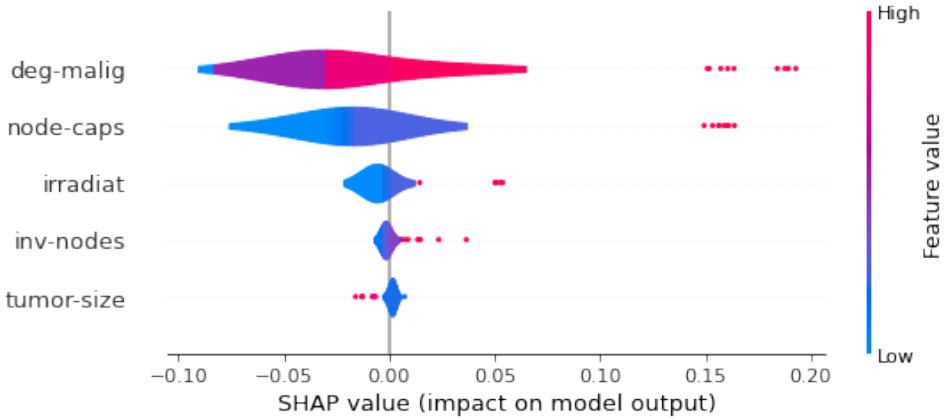


Rysunek 20: XGBoost: Wpływ wartości atrybutów na predykcje modelu

Analiza wykresu mówi nam, że znaczący wpływ na wyjście modelu ma deg-malig i tumor-size. Ciekawym zjawiskiem jest długa przerwa między skrajnymi wartościami większości atrybutów. Pokrywa się to z wnioskami płynącymi z analizy metryk - klasyfikuje pacjentów jako chorych tylko wtedy, gdy jest tego pewien. Niestety, do uzyskania wyżej wymienionej pewności model potrzebuje skrajnych wartości atrybutów, a mniej wyraźne zależności po prostu zignoruje.

4.5.6 SVM

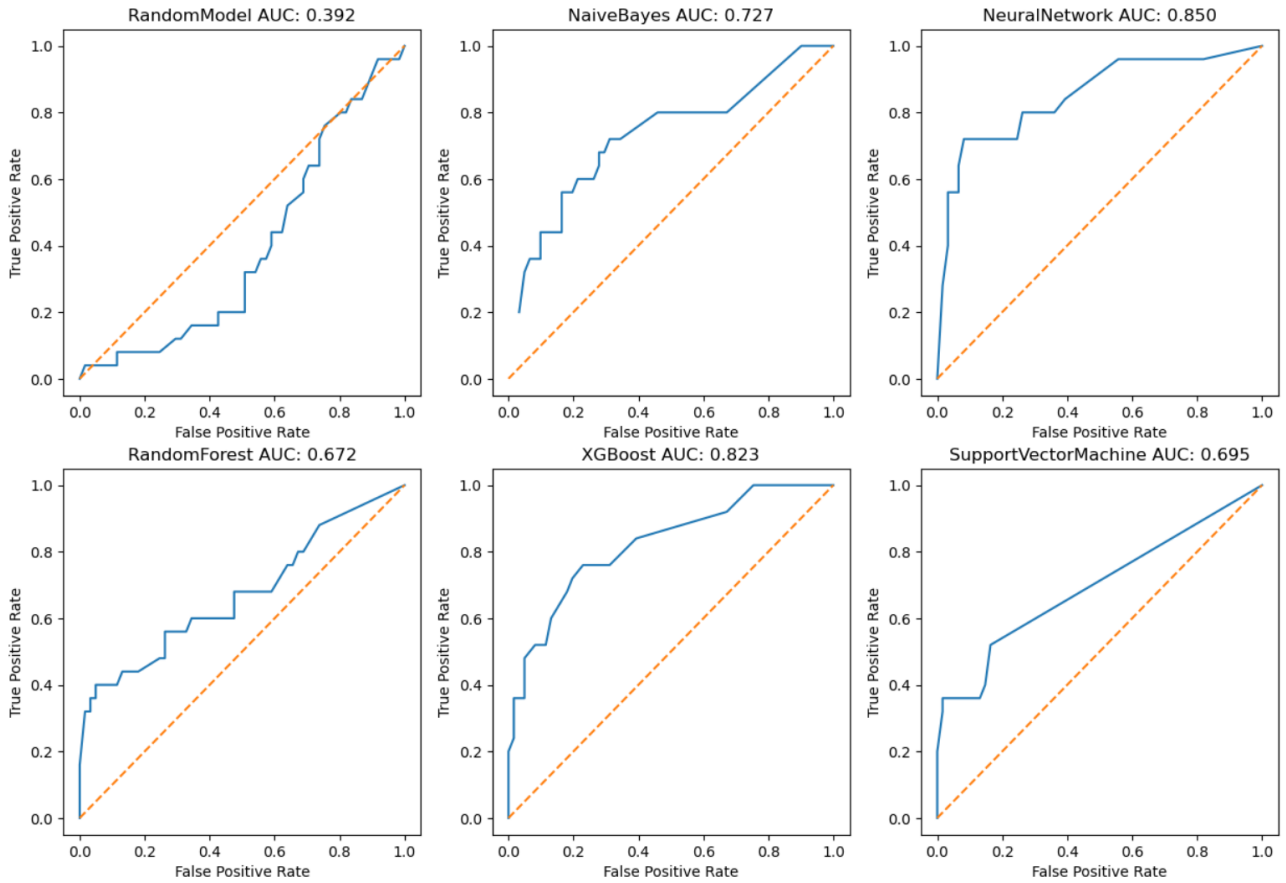
SVM posiada ten sam problem co XGBoost, wiele przypadków osób chorych jest klasyfikowanych jako zdrowe. SVM klasyfikuje jako chore osoby tylko te, co do których jest pewien że są chore. Wysokie specificity otrzymujemy za cene pomijania wielu przypadków osób chorych.



Rysunek 21: SVM: Wpływ wartości atrybutów na predykcje modelu

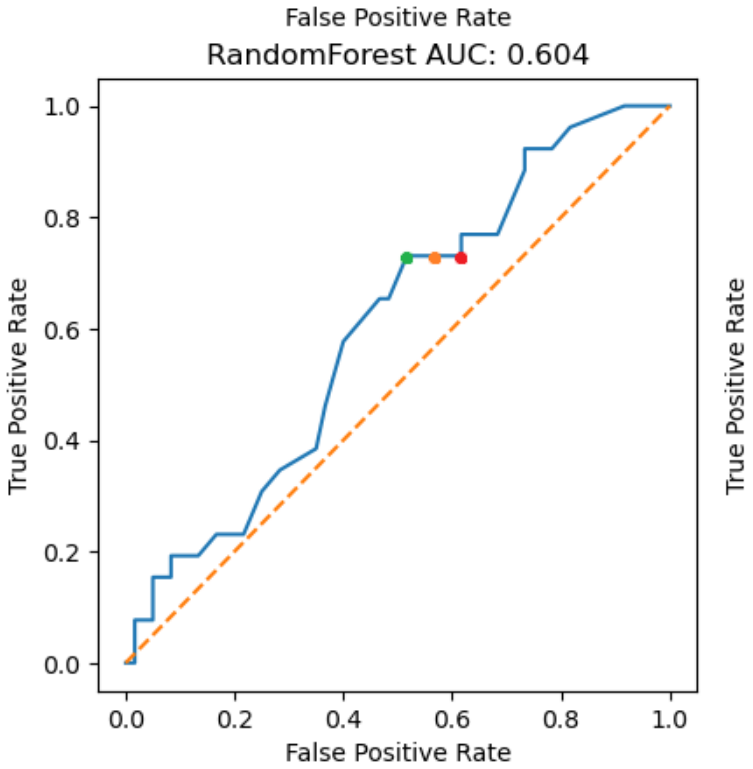
Na drodze analizy wykresu stwierdzamy, że niemal cały wpływ na wyjście modelu SMV ma wartość atrybutów deg-malig i node-caps, co ponownie pokrywa się z wnioskami wyciągniętymi na podstawie zbioru danych.

4.6 Krzywe ROC, AUC



Rysunek 22: Przykładowe krzywe ROC testowanych modeli

Krzywe ROC przedstawiają zależność True Positive Rate od False Positive Rate dla różnych poziomów odcięć (poziomów, od których rzutujemy prawdopodobieństwo klasy na klasę). Krzywe ROC rysuje się poprzez sprawdzenie, jak algorytm klasyfikuje przypadki dla wybranych poziomów odcięć. AUC to pole pod wykresem narysowanym w powyższy sposób. Do obliczenia pola pod wykresem użyliśmy algorytmu całkowania metodą trapezów. Większa wartość AUC modelu zwykle koreluje z lepszą jakością. Porównując nasze modele pod względem AUC, najlepiej wypada Sieć neuronowa, tuż po niej XGBoost - co pokrywa się z wnioskami wynikającymi z analizy metryk w poprzednich częściach sprawozdania.



Rysunek 23: Porównanie modeli na krzywej ROC

Dodatkowo możemy pod względem jakości klasyfikacji ocenić model zależnie od tego, jaki cutoff ustawiliśmy. Na rysunku przedstawiono punkty odpowiadające różnym wartościom cutoff, na podstawie tego możemy wnioskować, że model zaznaczony na zielono jest lepszy od modelu zaznaczonego pomarańczową kropką i czerwoną kropką.

4.7 Wnioski końcowe

Najlepiej z problemem poradziła sobie sieć neuronowa. We wnioskach początkowych ustaliliśmy, że najważniejszym czynnikiem wpływającym na ocenę jakości modelu jest recall - mówi on o tym, jak model radzi sobie z klasyfikacją osób chorych. Oprócz tego, sieć neuronowa charakteryzuje się ogólnie wyższymi wartościami takich metryk jak accuracy czy AUC. Jedynym minusem jest długi, w porównaniu do reszty modeli czas uczenia, co wynika z charakterystyki tego podejścia. Efekt ten można jednak łatwo zniwelować, stosując do klasyfikacji sieć neuronową która została wcześniej wytrenowana.

5 Analiza wyników dla zbioru danych Breast Cancer Wisconsin Data Set

Breast Cancer Wisconsin Data Set [LINK]

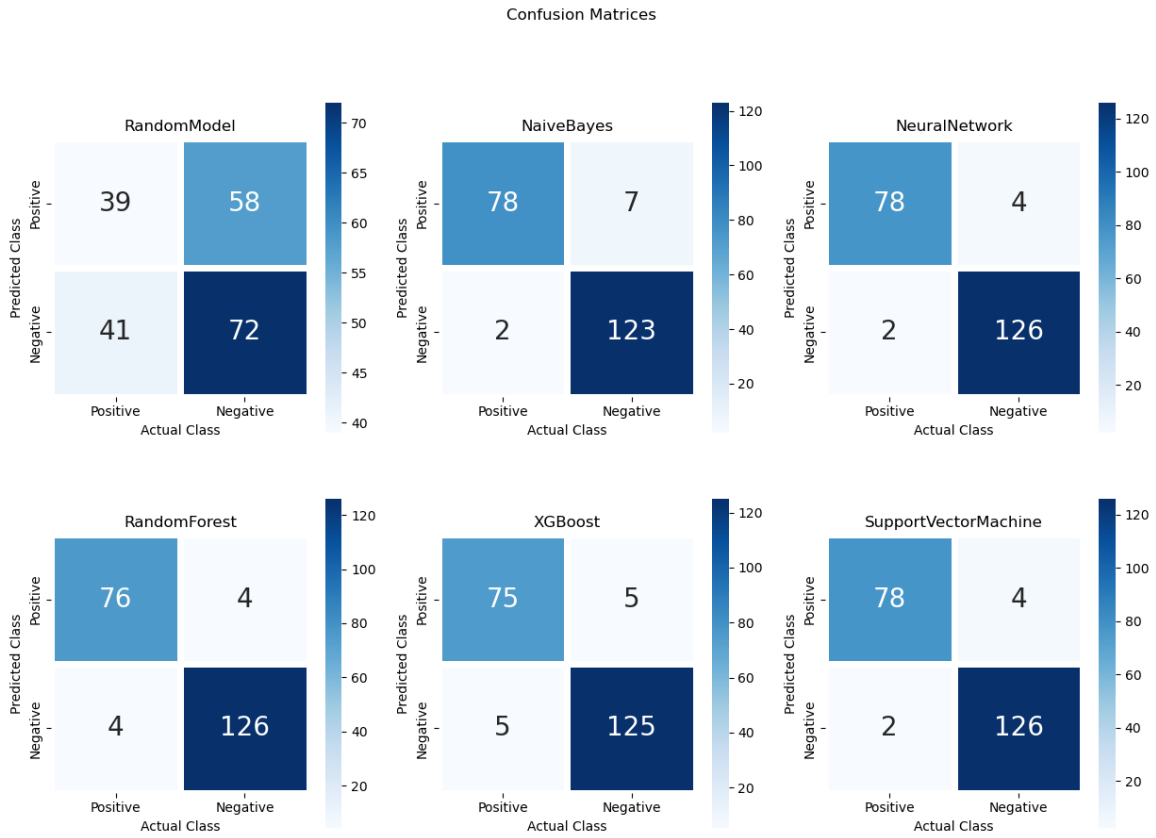
5.1 Wstępne założenia

W naszym zadaniu samo Accuracy nie będzie dobrym wyznacznikiem jakości modelu. Ważne będzie, aby przy ocenie wziąć pod uwagę Sensitivity oraz Specificty. To, aby ustalić która z tych dwóch metryk jest ważniejsza, musi odpowiedzieć pytanie czy jesteśmy bardziej skłonni dopuścić do klasyfikacji przypadków fałszywie negatywnych, czy fałszywie pozytywnych. Osobiście zdaje mi się, że większe straty ponosimy w przypadku klasyfikacji FP, przypadki FN zawsze można wykluczyć w dogłębnym badaniu, na przykład sięgając po opinię specjalistów lub innego algorytmu. Dlatego zdaje mi się, że Recall będzie w tym problemie ważniejszą metryką. Dodatkowo dobrymi miarami do porównania algorytmów będzie miara F1 oraz krzywe ROC i AUC.

5.2 Wpływ danych na wyniki

W porównaniu do oryginalnego zbioru danych, na którym miały być trenowane modele - breast_cancer_wisconsin charakteryzuje się przede wszystkim większą ilością przykładów w całym zbiorze jak i znacznie mniejszą dysproporcją pomiędzy klasami. Spodziewamy się, że spowoduje to zniwelowanie zjawiska przetrenowania modeli a tym samym poprawę ogólnej jakości klasyfikacji wszystkich modeli. Ważnym aspektem jest też to, że wszystkie atrybuty mają ujednoliconą formę i w łatwy sposób mogą zostać znormalizowane.

5.3 Macierze pomyłek



Rysunek 24: Macierze pomyłek

Macierze pomyłek przedstawiają nam wyniki predykcji naszych modeli. Na ich podstawie możemy wyliczać wcześniej opisane metryki.

5.4 Analiza wyników

Dane ze zbioru zostały podzielone na zbiór treningowy i testowy w proporcji 7 : 3. Na 699 przykładów daje nam to 489 wierszy, na których uczone są nasze modele. Jest to znacznie więcej danych niż w przypadku poprzedniego datasetu. Dodatkowo wcześniejsza analiza danych wykazała, że mamy do czynienia z wartościowymi danymi. Zakładamy, że uda nam się stworzyć modele które lepiej poradzą sobie z klasyfikacją.

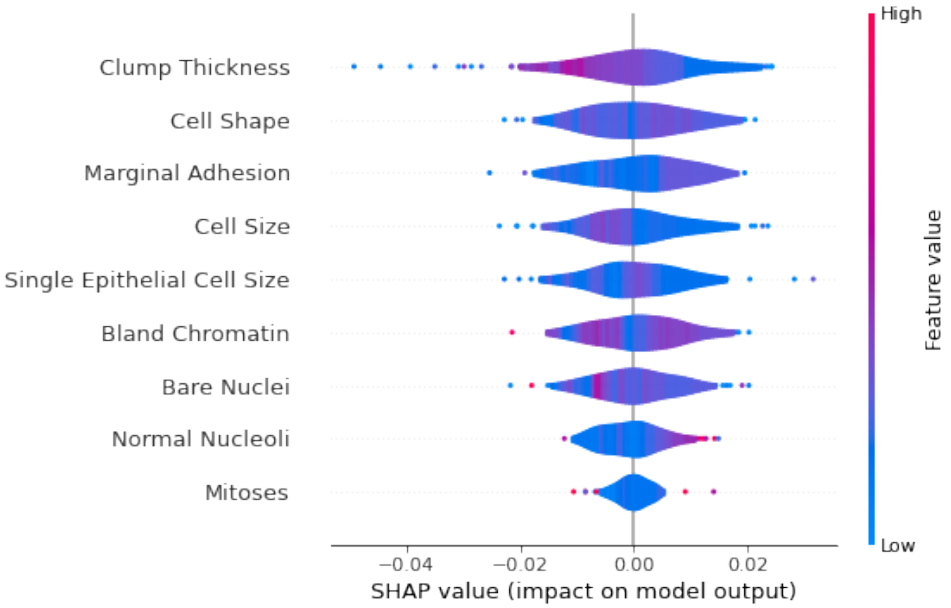
5.5 Analiza wyników poszczególnych algorytmów

	model	accuracy	recall(sensitvity)	specificity	precision	f1_score	support	auc	execution time
0	RandomModel	0.528571	0.4875	0.553846	0.402062	0.440678	80	0.494423	0.000000
1	NaiveBayes	0.957143	0.9750	0.946154	0.917647	0.945455	80	0.940192	0.002000
2	NeuralNetwork	0.971429	0.9750	0.969231	0.951220	0.962963	80	0.976827	11.707561
3	RandomForest	0.961905	0.9500	0.969231	0.950000	0.950000	80	0.986875	0.548198
4	XGBoost	0.952381	0.9375	0.961538	0.937500	0.937500	80	0.986971	0.131013
5	SupportVectorMachine	0.971429	0.9750	0.969231	0.951220	0.962963	80	0.992933	0.021005

Rysunek 25: Tabela metryk

5.5.1 RandomModel

Model losowy, ponownie, działa - losowo, posłuży on nam jako model podstawowy, jeśli jakiś algorytm pod względem danych metryk radzi sobie gorzej niż model losowy, możemy uznać go za zły.

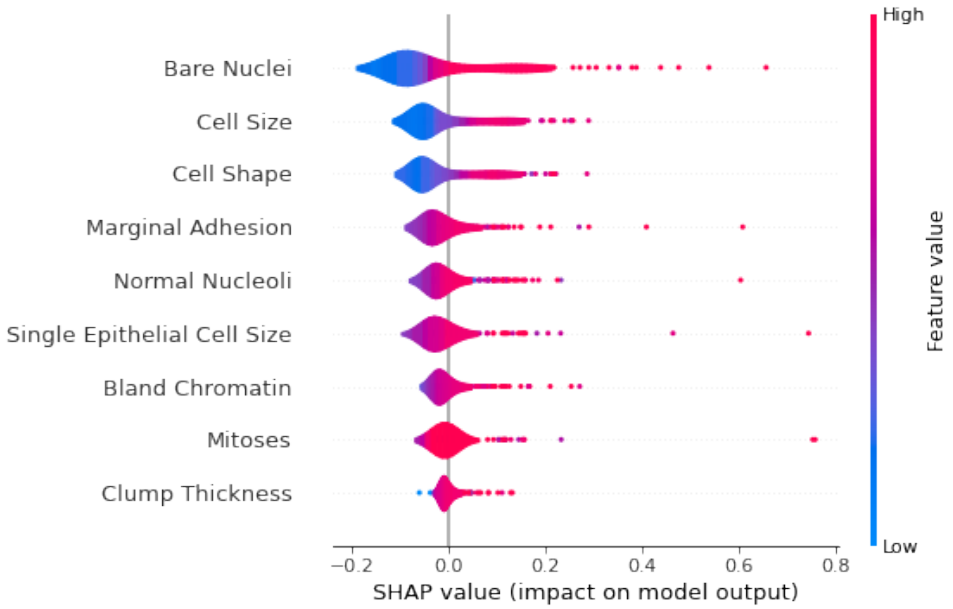


Rysunek 26: RandomModel: Wpływ wartości atrybutów na predykcje modelu

Analiza SHAP w przypadku modelu losowego nie niesie za sobą głębszego sensu, ponieważ atrybuty nie mają żadnego znaczenia przy odpowiedzi modelu.

5.5.2 NaiveBayes

Naiwny Bayes nie wyróżnia się niczym na tle innych modeli uczonych na tym zbiorze danych. Posiada wysokie miary wszystkich metryk, co pokazuje jak dobrze poradził sobie z problemem.

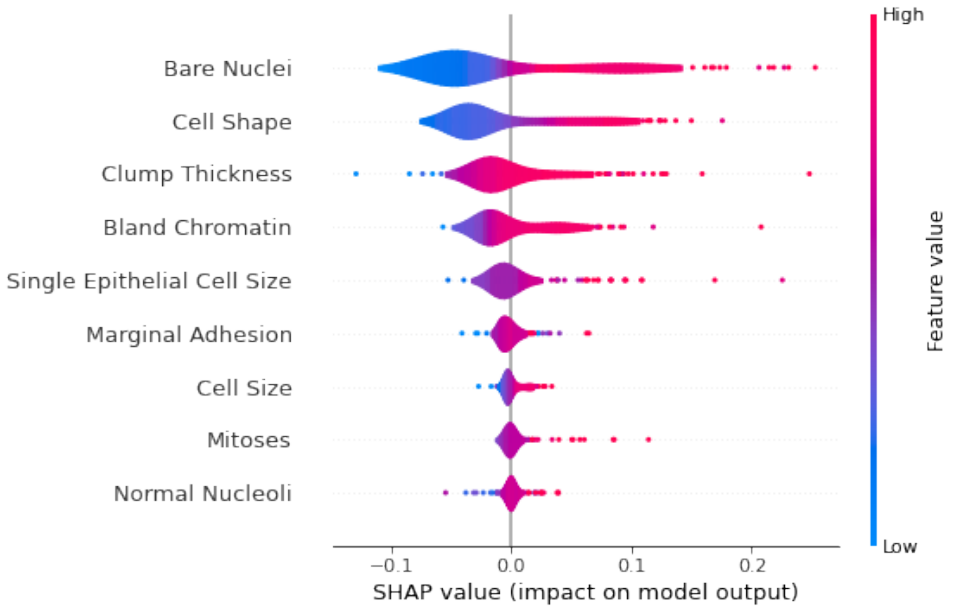


Rysunek 27: NaiveBayes: Wpływ wartości atrybutów na predykcje modelu

Naiwny Bayes korzysta z tego co odkryliśmy podczas analizy danych, największe znaczenie w predykcji mają atrybuty najbardziej skorelowane z klasą wyjściową - Bare Nuclei, Cell Size i Cell Shape.

5.5.3 NeuralNetwork

Sieć Neuronowa nie wyróżnia się niczym na tle innych modeli uczonych na tym zbiorze danych. Posiada wysokie miary wszystkich metryk, co pokazuje jak dobrze poradził sobie z problemem. Jedynym, co negatywnie wyróżnia to podejście na tle innych jest wysoki czas uczenia wynikający z natury modelu.

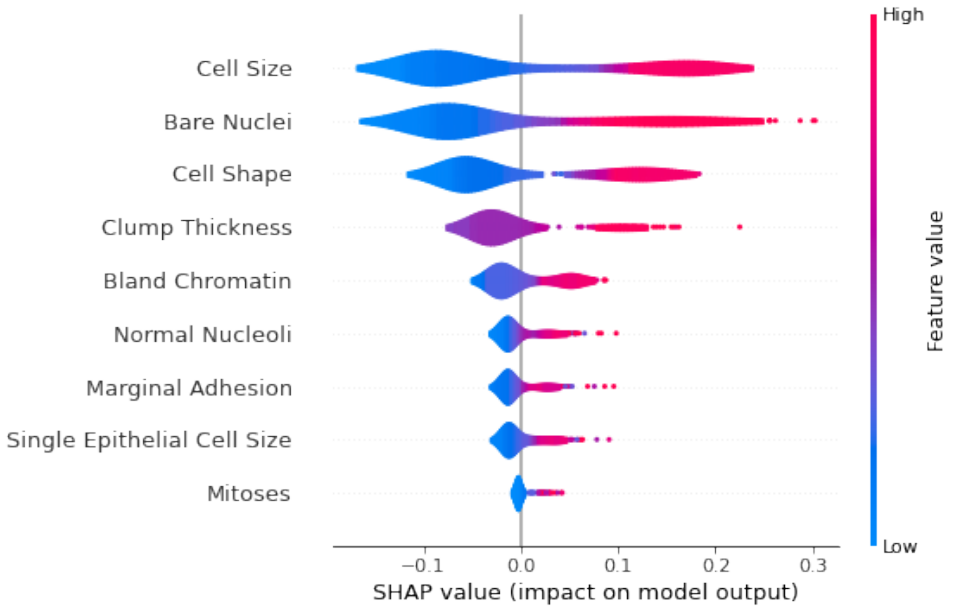


Rysunek 28: NeuralNetwork: Wpływ wartości atrybutów na predykcje modelu

W przypadku sieci neuronowej największy wpływ na predykcje miały atrybuty: Bare Nuclei, Cell Shape i Clump Thickness. Cell Size ma mały wpływ na wyjście, co jest dość zaskakujące. Analizując dane atrybut Cell Size wykazał wysokie skorelowanie z atrybutem klasy, najwidoczniej sieć neuronowa nie korzysta z niego w swoich predykcjach.

5.5.4 RandomForest

Las losowy nie wyróżnia się niczym na tle innych modeli uczonych na tym zbiorze danych. Posiada wysokie miary wszystkich metryk, co pokazuje jak dobrze poradził sobie z problemem.

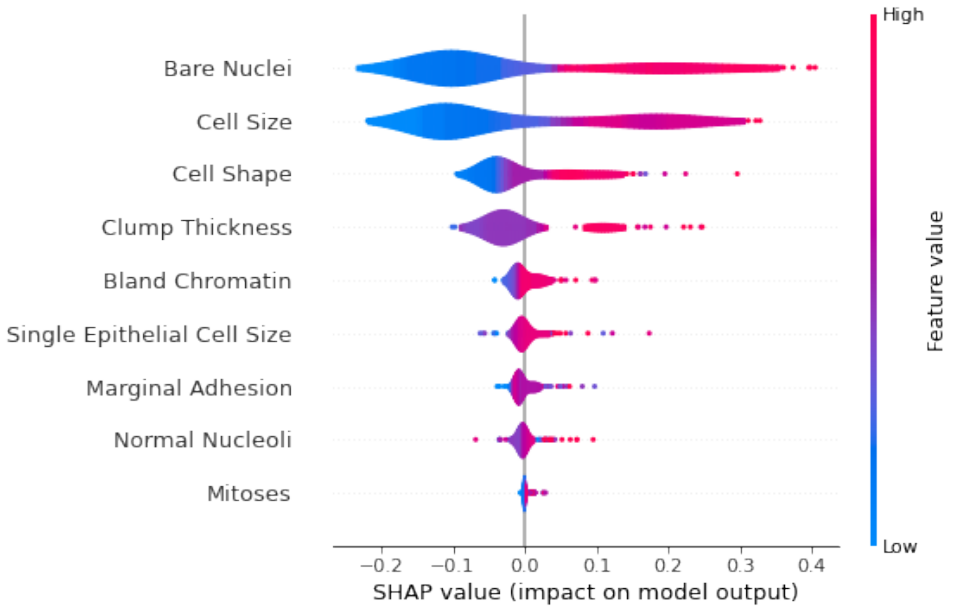


Rysunek 29: RandomForest: Wpływ wartości atrybutów na predykcje modelu

Zauważamy duży wpływ atrybutów Cell Size, Bare Nuclei i Cell Size na predykcje modelu.

5.5.5 XGBoost

XGBoost nie wyróżnia się niczym na tle innych modeli uczonych na tym zbiorze danych. Posiada wysokie miary wszystkich metryk, co pokazuje jak dobrze poradził sobie z problemem.

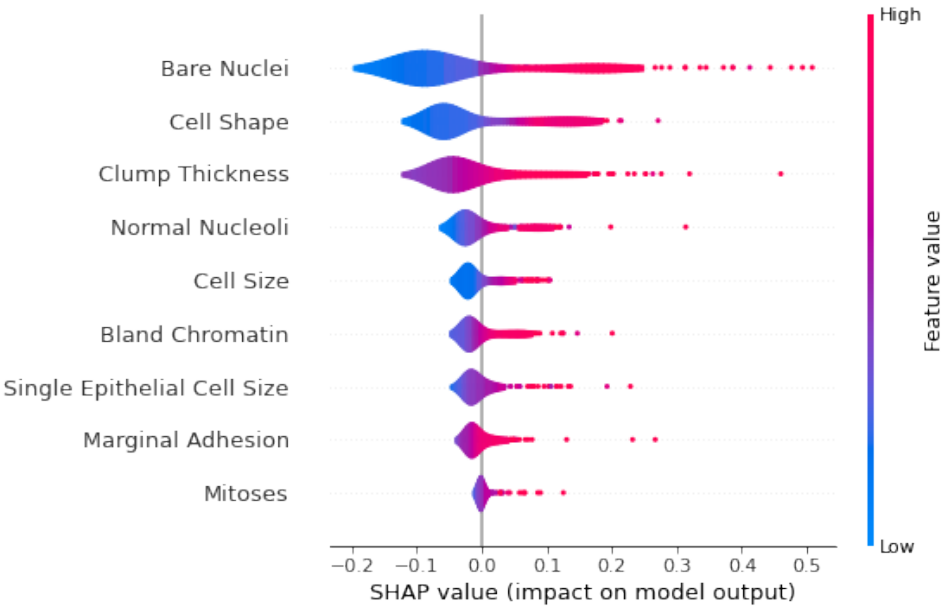


Rysunek 30: XGBoost: Wpływ wartości atrybutów na predykcje modelu

Z analizy wykresu można zauważyć bardzo duży wpływ atrybutów Bare Nuclei i Cell Shape na predykcje modelu.

5.5.6 SVM

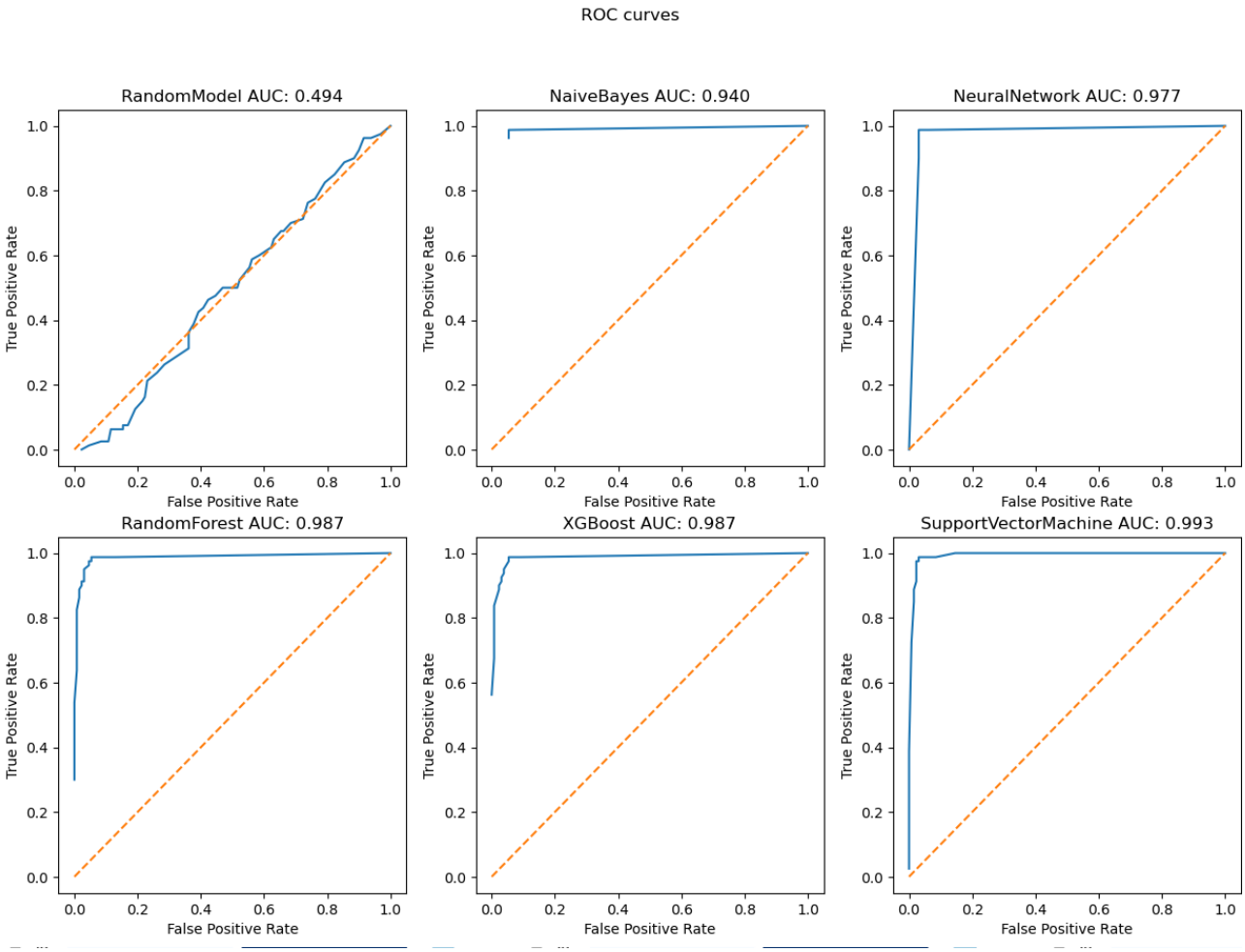
SVM nie wyróżnia się niczym na tle innych modeli uczonych na tym zbiorze danych. Posiada wysokie miary wszystkich metryk, co pokazuje jak dobrze poradził sobie z problemem. Zważając na to, że wszystkie metody poradziły sobie z problemem klasyfikacji dość podobnie pod względem metryk oceny, to właśnie SVM można nazwać najlepszym modelem ze względu na szybkość rozwiązania problemu (zaledwie 20ms).



Rysunek 31: SVM: Wpływ wartości atrybutów na predykcje modelu

Z analizy wykresu można zauważyć bardzo duży wpływ atrybutów Bare Nuclei i Cell Shape na predykcje modelu. Mały wpływ wydaje się mieć Cell Size, mimo dużej korelacji z klasą wyjściową.

5.6 Krzywe ROC, AUC



Rysunek 32: Przykładowe krzywe ROC testowanych modeli

Krzywe ROC przedstawiają zależność True Positive Rate od False Positive Rate dla różnych poziomów cutoff. Krzywe ROC rysuje się poprzez sprawdzenie, jak algorytm klasyfikuje przypadki dla wybranych poziomów cutoff. Cutoff to liczba z zakresu [0, 1] która definiuje które predykcje modeli są klasyfikowane jako klasa pozytywna, a które jako negatywna. AUC to pole pod wykresem narysowanym w powyższy sposób. Do obliczenia pola pod wykresem użyliśmy algorytmu całkowania metodą trapezów. Im większy AUC tym model zazwyczaj model można uznać za lepszy, porównując nasze modele pod względem AUC, najlepiej wypada SVM, lecz jest to na tyle mała przewaga, że nie warto jej brać pod uwagę.

5.7 Wnioski końcowe

Na drodze analizy wyników wysuwa się wniosek, że wszystkie modele oprócz modelu losowego poradziły sobie z problemem klasyfikacji bardzo dobrze. Jest to spowodowane tym, że zbiór danych jest wyższej jakości, zarówno pod względem liczności jak i silnych korelacji atrybutów z

klasą wyjściową. Na miano najlepszego modelu zasługuje SVM, z uwagi na szybkość rozwiązania problemu klasyfikacji.

6 Użyte narzędzia i biblioteki

Projekt wykonaliśmy w języku Python. Wykorzystaliśmy biblioteki:

- matplotlib==3.7.1 - do wykresów
- numpy==1.22.0 - do obliczeń
- pandas==1.3.5 - do działań na wektorze danych
- scikit_learn==1.1.2 - do niektórych modeli
- seaborn==0.11.2 - do wizualizacji danych
- torch==1.12.1 - do obliczeń na tensorach i sieci neuronowej
- xgboost==1.5.2 - do gotowej implementacji xgboost
- pytest==7.3.1 - do testów jednostkowych