

Metody odkrywania wiedzy: Forest Cover Type Prediction

Monika Pawluczuk, Bartosz Lemiec

14 czerwca 2016

1 Interpretacja tematu projektu

Temat dotyczy predykcji typu lasu i oparty jest na danych dostarczonych przez US Forest Service (USFS).

Celem jest przewidzenie zalesienia terenu, a dokładniej określeniu typu drzewa dominującego w danym terenie, z danych kartograficznych. Typ zalesienia został określony, na parcelach o powierzchni 30x30 metrów, przez US Forest Service (USFS). Dane są nie-przeskalowane i zawierają kolumny z wartościami binarnymi dla niezależnych zmiennych jakościowych takich jak obszar czy też typ gleby.

2 Opis algorytmów wykorzystanych w badaniach

2.1 Algorytm K-najbliższych sąsiadów

Algorytm może być stosowany zarówno do zadań klasyfikacji jak i regresji. Jest jednym z najbardziej znanych podejść aproksymacji funkcji na podstawie zapamiętywania. W celu wyznaczenia odpowiedzi na zapytanie dotyczące przykładu bierze się pod uwagę najbliższy mu, według przyjętej metryki, przykład trenujący.

Tak sformułowany algorytm nie wymaga żadnych założeń co do dziedziny i reprezentacji przykładów, oprócz określenia miary odległości. Funkcja miary będzie w rzeczywistości pojęciem, o przeciwdziedzynie będącej zbiorem kategorii.

Algorytm charakteryzuje się bardzo dużą szybkością uczenia się, a jego dokładność zależy od ilości przykładów trenujących oraz funkcji mierzącej odległość.

Zostanie wykorzystany pakiet R `class` (<https://cran.r-project.org/web/packages/class/class.pdf>). Zakłada on użycie jako metryki odległości euklidesowej.

2.2 Drzewo decyzyjne - algorytm C4.5

Algorytm wykorzystywany do generowania drzew decyzyjnych. Jest rozszerzeniem wcześniejszego algorytmu ID3. Drzewa decyzyjne generowane za pomocą tego algorytmu mogą zostać wykorzystane przy klasyfikacji i dlatego też algorytm ten określany jest często jako klasyfikator statystyczny.

Buduje on drzewo decyzyjne z zestawu danych trenujących, wykorzystując pojęcie entropii informacji. Zestaw trenujący zawiera zaklasyfikowane już przykłady, zawierające

p-wymiarowy wektor $(x_{1,i}, x_{2,i}, \dots, x_{p,i})$, gdzie x_j reprezentuje wartość atrybutu albo właściwości próbki, oraz klasę do której S_i należy.

Przy każdym węźle drzewa decyzyjnego, algorytm C4.5, wybiera atrybut danych, który z największą efektywnością rozdziela zbiór próbek na podzbiory o większej entropii dla którejs z klas. Kryterium podziału jest entropia względna. Atrybut o największym przyroście informacji jest wybierany, jako ten na podstawie którego tworzone będą kolejne węzły lub liście. Algorytm C4.5 następnie postępuje rekurencyjnie, aby stworzyć pomniejsze podlisty.

Algorytm z góry pokrywa kilka podstawowych przypadków:

- Wszystkie próbki w liście należą do tej samej klasy \Rightarrow w takim przypadku drzewo tworzy liść drzewa wybierający daną klasę.
- Żaden z atrybutów nie niesie ze sobą jakichkolwiek informacji \Rightarrow algorytm C4.5 wykorzystuje węzeł decydujący wyżej w strukturze drzewa i wartość oczekiwaną w nim zawartą.
- Instancja nigdy nie spotkanej wcześniej klasy zostaje znaleziona \Rightarrow algorytm tworzy węzeł wyżej w drzewie wykorzystując jego wartość oczekiwaną.

Zostanie wykorzystany pakiet R RWeka, zawierający implementację algorytmu J48 (<https://cran.r-project.org/web/packages/RWeka/RWeka.pdf>).

2.3 Naiwny klasyfikator bayesowski

Naiwny klasyfikator bayesowski reprezentuje hipotezy za pomocą oszacowań (tworzonych na podstawie zbioru trenującego) pewnych prawdopodobieństw i klasyfikuje przykłady, wybierając dla nich najbardziej prawdopodobne kategorie. Przypomina on optymalny klasyfikator bayesowski, z podstawową różnicą, że nie wykorzystuje żadnych innych hipotez (nawet w celach pomocniczych). Zakłada on ograniczenie się do zestawu atrybutów dyskretnych.

Na podstawie zbioru trenującego szacowane są prawdopodobieństwa poszczególnych kategorii pojęcia docelowego c oraz prawdopodobieństwa poszczególnych wartości atrybutów dla przykładów różnych kategorii. Będą nas interesować oszacowania prawdopodobieństw:

- Prawdopodobieństwo dla każdej możliwej kategorii pojęcia docelowego c
- Prawdopodobieństwo wystąpienia danej wartości atrybutu dla przykładu z danej klasy - dla każdej możliwej kategorii pojęcia docelowego i każdej wartości każdego atrybutu.

Aby obliczyć prawdopodobieństwo kategorii d pojęcia docelowego: wyznaczamy ile jest przykładów z tą klasą w zbiorze trenującym, a następnie dzielimy ją przez ilość wszystkich przykładów w zbiorze trenującym.

Podobnie, aby oszacować prawdopodobieństwo wartości atrybutu $a_i = v$ dla kategorii d , liczymy ile jest przykładów każdej kategorii dla poszczególnych wartości każdego atrybutu i dzielimy uzyskaną przez liczbę wszystkich przykładów tej kategorii.

Wadą tego podejścia jest przypadek, gdy w którejś z grup nie ma żadnego przykładu. Wówczas prawdopodobieństwo jest szacowane na 0, co jest bardzo radykalnym podejściem. W związku z tym stosuje się technikę wygładzania, która jest też zaimplementowana w pakiecie, który będzie użyty w projekcie.

Wygładzanie polega na zastąpieniu wartości zerowej pewną niewielką, lecz dodatnią wartością ϵ . Jest to proste i popularne rozwiązanie, które powinno być dostosowane do rozmiaru zbioru trenującego. Na pewno wartość ta powinna być mniejsza niż odwrotność liczebności tego zbioru (skoro grupa, której mamy tylko jeden przykład będzie miała prawdopodobieństwo $1/|T|$ - grupa o zerowej liczbie przykładów powinna mieć mniejsze prawdopodobieństwo).

Najczęstszą heurystyką jest przyjęcie epsilonu na poziomie połowy odwrotności liczebności zbioru trenującego $1/2 * |T|$.

Ta sama zasada obowiązuje również przy obliczaniu prawdopodobieństwa klas.

Zostanie wykorzystany pakiet R `e1071` (<https://cran.r-project.org/web/packages/e1071/e1071.pdf>). Implementacja zakłada niezależność rozkładów atrybutów (stąd “naïwność” w nazwie) i rozkład Gaussa. Niestety implementacja, w razie brakujących wartości atrybutów pomija przykład ze zbioru trenującego.

3 Plan badań

3.1 Cel badań

Obszar przeznaczony do analizy zawiera cztery lokalizacje w Roosevelt National Forest, znajdującego się w północnym Colorado. Każda obserwacja przeprowadzona została na obszarze 30x30m. Celem jest predykcja typu lasu, który pokrywać będzie dany obszar, w postaci liczby, która to z kolei odpowiada jednemu z siedmiu typów:

Lp.	Nazwa
1	Świerk/jodła
2	Sosna wydmowa
3	Sosna żółta
4	Topola/wierzba
5	Topola osikowa
6	Daglezja zielona
7	Drzewo karłowate

3.2 Charakterystyka zbiorów danych i ich przygotowanie

Zestaw trenujący złożony z 15120 przykładów zawiera zarówno właściwości, jak i typ pokrycia. Zestaw testowy zawiera jedynie właściwości. Celem jest przewidzenie typu pokrycia w każdym z wierszy zestawu testowego (565892 obserwacje).

3.2.1 Kolumny danych

Lp.	Nazwa	Opis	Opis (j.ang.)
1	Elevation	Elewacja wyrażona w metrach	Elevation in meters
2	Aspect	Strona wyrażona w stopniach azymutu	Aspect in degrees azimuth
3	Slope	Nachylenie wyrażone w stopniach	Slope in degrees
4	Horizontal_Distance_To_Hydrology	Dystans horyzontalny do najbliższego zbiornika wodnego	Horizontal distance to nearest surface water features
5	Vertical_Distance_To_Hydrology	Dystans wertykalny do najbliższego zbiornika wodnego	Vertical distance to nearest surface water features
6	Horizontal_Distance_To_Roadways	Dystans horyzontalny do najbliższej drogi	Horizontal distance to nearest roadway
7	Hillshade_9am	Cieniowanie, zakres od 0 do 255, o godzinie 9, w dniu przesilenia	Hillshade index at 9am, summer solstice (0 to 255 index)
8	Hillshade_Noon	Cieniowanie, zakres od 0 do 255, w południe, w dniu przesilenia	Hillshade index at noon, summer solstice (0 to 255 index)
9	Hillshade_3pm	Cieniowanie, zakres od 0 do 255, o godzinie 3, w dniu przesilenia	Hillshade index at 3pm, summer solstice (0 to 255 index)
10	Horizontal_Distance_To_Fire_Points	Dystans horyzontalny do najbliższej stacji straży pożarnej	Horz Dist to nearest wildfire ignition points
11	Wilderness_Area	Obszar, wyrażony binarnie, w 4 kolumnach	Wilderness area designation
12	Soil_Type	Rodzaj gleby, wyrażony binarnie, w 40 kolumnach	Soil Type designation (40 binary columns, 0 = absence or 1 = presence)
13	Cover_Type	Docelowy typ zalesienia, wartości od 1 do 7	Forest Cover Type designation (7 types, integers 1 to 7)

3.2.2 Typy obszarów

Lp.	Nazwa
1	Rawah Wilderness Area
2	Neota Wilderness Area
3	Comanche Peak Wilderness Area
4	Cache la Poudre Wilderness Area

3.2.3 Rodzaje gleb

Lp.	Nazwa	Opis
1	Cathedral family	Rock outcrop complex, extremely stony
2	Vanet	Ratake families complex, very stony
3	Haploborolis	Rock outcrop complex, rubbly
4	Ratake family	Rock outcrop complex, rubbly
5	Vanet family	Rock outcrop complex complex, rubbly
6	Vanet - Wetmore families	Rock outcrop complex, stony
7	Gothic family	-
8	Supervisor	Limber families complex
9	Troutville family	Very stony
10	Bullwark - Catamount families - Rock outcrop complex, rubbly	-
11	Bullwark - Catamount families - Rock land complex, rubbly	-
12	Legault family	Rock land complex, stony
13	Catamount family	Rock land - Bullwark family complex, rubbly
14	Pachic Argiborolis	Aquolis complex
15	unspecified in the USFS Soil and ELU Survey	-
16	Cryaquolis	Cryoborolis complex
17	Gateview family	Cryaquolis complex
18	Rogert family, very stony	-
19	Typic Cryaquolis	Borochemists complex
20	Typic Cryaquepts	Typic Cryaquolls complex
21	Typic Cryaquolls	Leighcan family, till substratum complex
22	Leighcan family, till substratum, extremely bouldery	-
23	Leighcan family, till substratum - Typic Cryaquolls complex	-
24	Leighcan family, extremely stony	-
25	Leighcan family, warm, extremely stony	-
26	Granile	Catamount families complex, very stony
27	Leighcan family, warm	Rock outcrop complex, extremely stony
28	Leighcan family	Rock outcrop complex, extremely stony
29	Como	Legault families complex, extremely stony
30	Como family - Rock land	Legault family complex, extremely stony
31	Leighcan	Catamount families complex, extremely stony
32	Catamount family - Rock outcrop	Leighcan family complex, extremely stony
33	Leighcan - Catamount families	Rock outcrop complex, extremely stony
34	Cryorthents	Rock land complex, extremely stony
35	Cryumbrepts - Rock outcrop	Cryaquepts complex
36	Bross family - Rock land	Cryumbrepts complex, extremely stony
37	Rock outcrop - Cryumbrepts	Cryorthents complex, extremely stony
38	Leighcan - Moran families	Cryaquolls complex, extremely stony
39	Moran family - Cryorthents	Leighcan family complex, extremely stony
40	Moran family - Cryorthents	Rock land complex, extremely stony

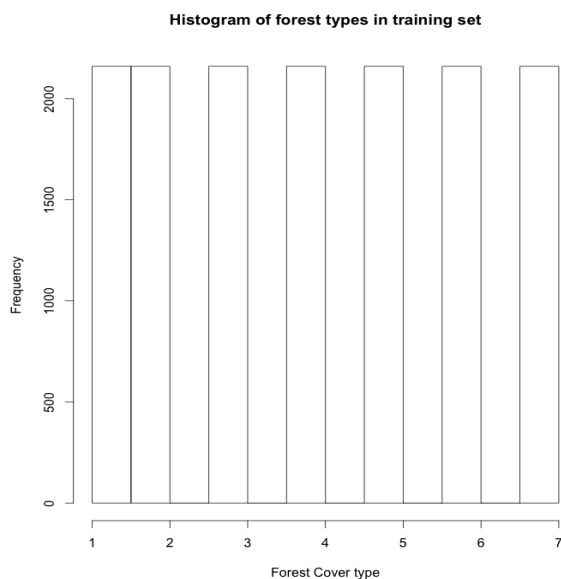
4 Wstępna analiza danych

Wstępna analiza danych została przedstawiona w tabeli poniżej, gdzie zostały policzone podstawowe wartości, takie jak ilość unikalnych wartości atrybutu, wartość minimalna, maksymalna, czy mediana.

4.1 Podsumowanie dostępnych danych

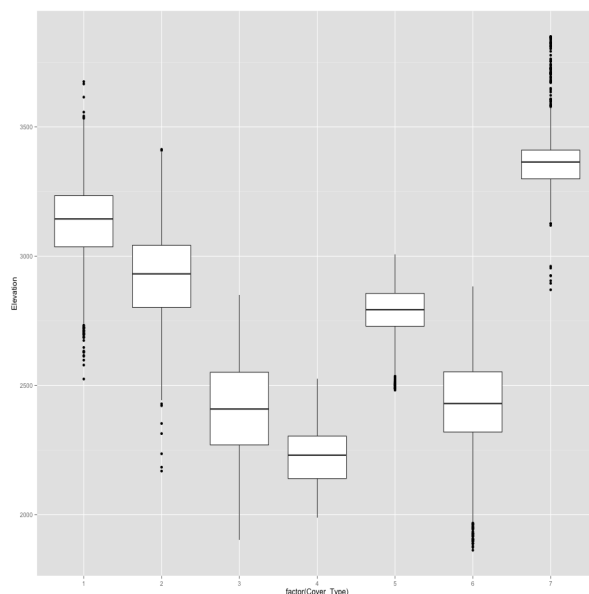
Liczba przykładów trenujących: 15120 Liczba kolumn: 56 Rozkład obszarów zalesienia:

Typ lasu	1	2	3	4	5	6	7
Ilość przykładów	2160	2160	2160	2160	2160	2160	2160



- Zbiór trenujący zawiera 15120 wierszy, wraz z 56 kolumnami
- Kolumna 1 reprezentuje ID danego wpisu
- Kolumny 2-11 reprezentują atrybuty ilościowe
- Kolumny 12-15 to reprezentacja binarna obszaru *Wilderness_Area* (4 kolumny wzajemnie się wykluczające: ComanchePeak, CachePodure, Neota, Rawah)
- Kolumny 16-55 to reprezentacja binarna typu gleby *Soil_Type* (40 kolumn wzajemnie się wykluczających)
- Kolumna *Cover_Type* zawiera 7 wartości (docelowa klasyfikacja)
- Kolumny binarne wzajemnie się wykluczają

Największą zależność, pod względem przypisania do klasy docelowej, wykazuje atrybut *Elevation*, który to został przedstawiony na poniższym wykresie pudełkowym:



Można więc się spodziewać, że na przykład w przypadku zastosowania algorytmu C4.5 będzie on stanowił atrybut wyjściowy, od którego algorytm będzie zaczynać budowanie drzewa, a dopiero analiza kolejnych wartości atrybutów pozwoli na dokładniejsze zaklasyfikowanie próbek.

4.2 Atrybut dyskretny reprezentujący pojęcie docelowe

Atrybutem reprezentującym pojęcie docelowe jest typ lasu. Jest to zbiór liczbowy, składający się z siedmiu wartości:

Lp.	Nazwa	Nazwa (j.ang.)
1	Świerk/jodła	Spruce/Fir
2	Sosna wydmowa	Lodgepole Pine
3	Sosna żółta	Ponderosa Pine
4	Topola/wierzba	Cottonwood/Willow
5	Topola osikowa	Aspen
6	Daglezja zielona	Douglas-fir
7	Drzewo karłowate	Krummholz

4.3 Przetworzenie do odpowiedniej postaci

Zbiór danych trenujących jest dostępny do pobrania w postaci pliku csv. Jest on wczytywany za pomocą komendy `read.csv` do formatu `data.frame`.

Algorytm najbliższych sąsiadów jako argument danych treningowych/testujących może przyjmować dane w tym właśnie formacie, podobnie jak naiwny klasyfikator bayesowski i algorytm C4.5.

4.3.1 Modyfikacja atrybutów

Ze względu na wzajemne wykluczanie się wartości przechowywanych w kolumnach 12-15, opisujących *Wilderness_Area*, atrybut ten został złączony w jeden, przyjmujący wartości dyskretne od 1 do 4, gdzie jego liczba porządkowa przypisana jest do typu obszaru opisanego w tabeli Typy obszarów.

Podobna optymalizacja została przeprowadzona dla typów gleb *Soil_Type* (kolumny 16-55), z tego samego powodu. Przyjmują one więc wartości dyskretne od 1 do 40 zgodnie z tabelą Rodzaje gleb.

4.3.2 Eliminacja/naprawa defektów danych

Dane nie zawierają brakujących danych, dlatego też nie będzie potrzebna estymacja wartości atrybutów dla danych brakujących. Ewentualne defekty w danych mogą występować ze względu na źródło danych, skąd zostały one pobrane (USFS), jednak przy realizacji tego projektu zakładamy ich poprawność, ze względu na trudność lokalizacji ewentualnych błędów.

4.3.3 Modyfikacja rozkładu klas

Klasy w zestawie danych testowych rozkładają się równomiernie, dlatego też nie zostanie przeprowadzona modyfikacja ich rozkładu.

4.3.4 Losowanie podzbiorów danych

Ze względu na równy podział klas, losowanie podzbiorów danych wykonywane będzie z uwzględnieniem sprawiedliwego podziału rozkładu klas pomiędzy zbiorami danych trenujących. Oznacza to, iż każdy podzbiór danych zbioru trenującego będzie zawierać równą ilość klas konkretnego typu tak, aby w modelu uczącym się żadna z klas nie mogła zdominować pod względem jej wyboru.

4.3.5 Możliwość zdefiniowania nowych atrybutów

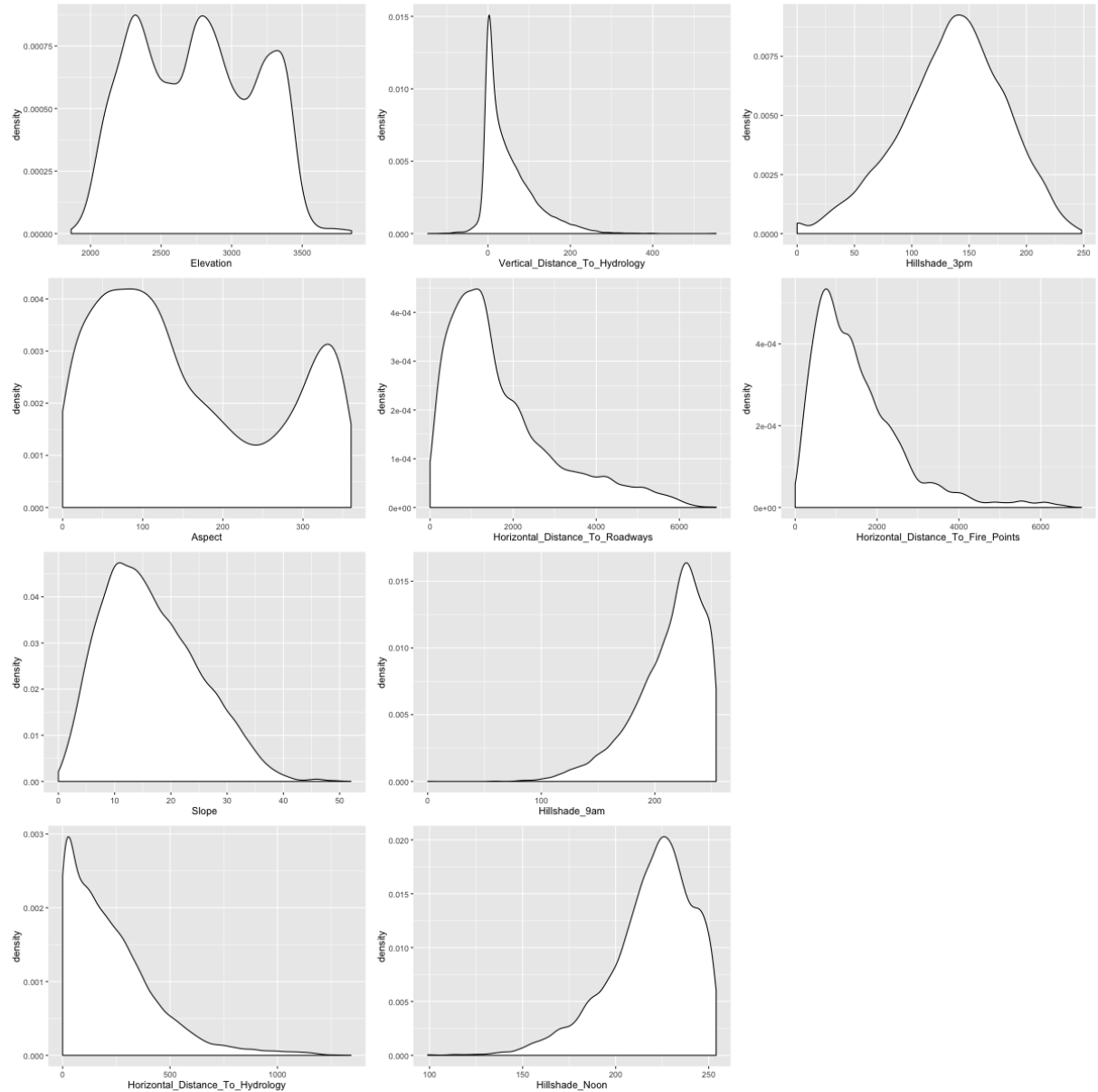
Możliwość definicji nowych atrybutów może wynikać z korelacji zachodzących pomiędzy istniejącymi atrybutami. Korelacja zaobserwowana będzie w trakcie dalszej części realizacji projektu, na podstawie graficznej reprezentacji atrybutów.

5 Implementacja i wyniki

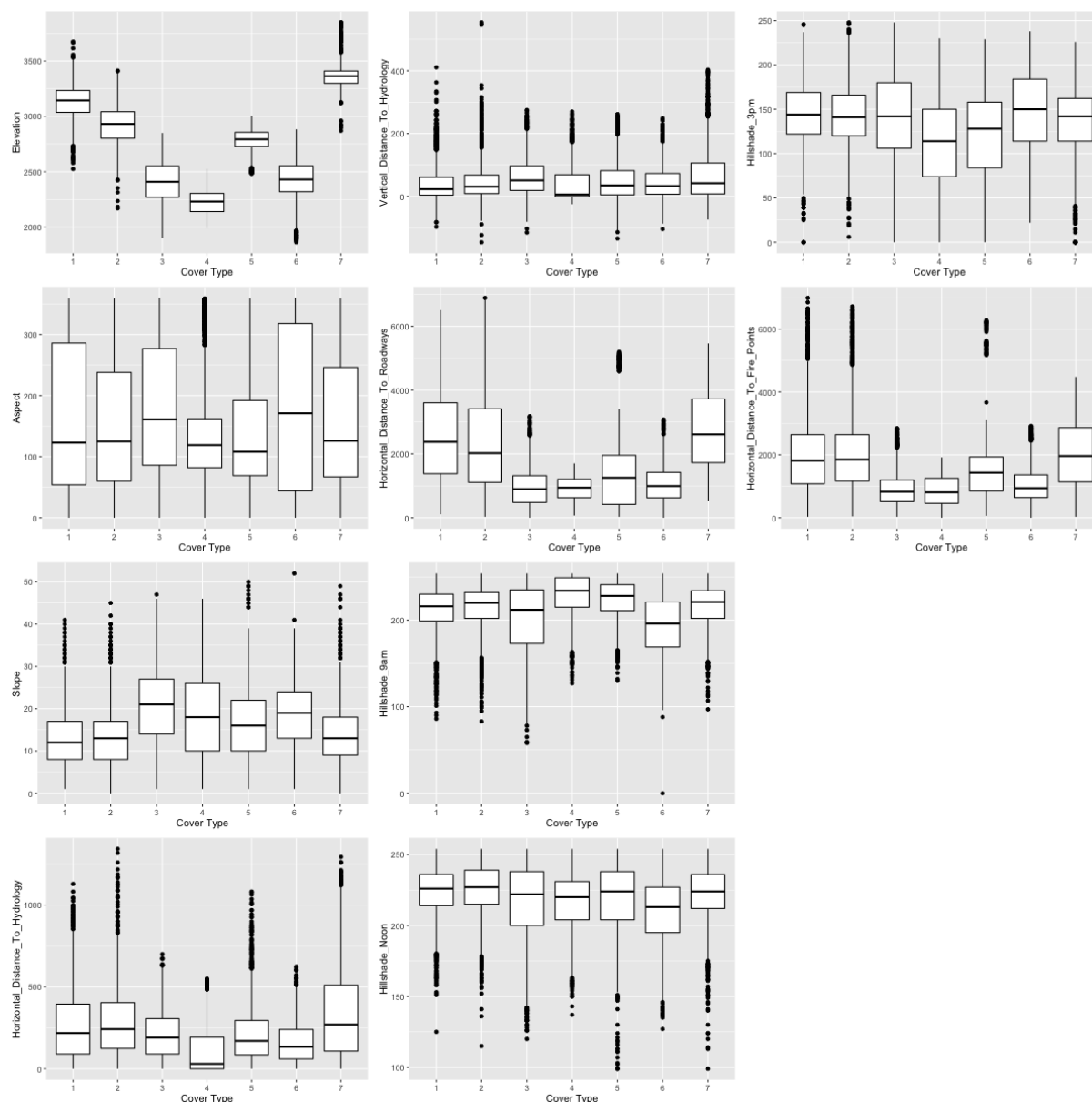
5.1 Analiza atrybutów

Zgodnie z założeniami, atrybuty dotyczące rodzaju gleby (*Soil Type*) oraz obszaru (*Wilderness Area*) zostały przekształcone na pojedyncze atrybuty dyskretne. Pozostałe atrybuty są wartościami numerycznymi, w związku z tym można było zbadać rozkład gęstości

tych atrybutów oraz związek ich wartości z klasą wynikową. Zbiorcze wykresy zostały przedstawione poniżej.



Z wykresów gęstości można zobaczyć, że atrybuty różnią się między sobą pod tym względem.



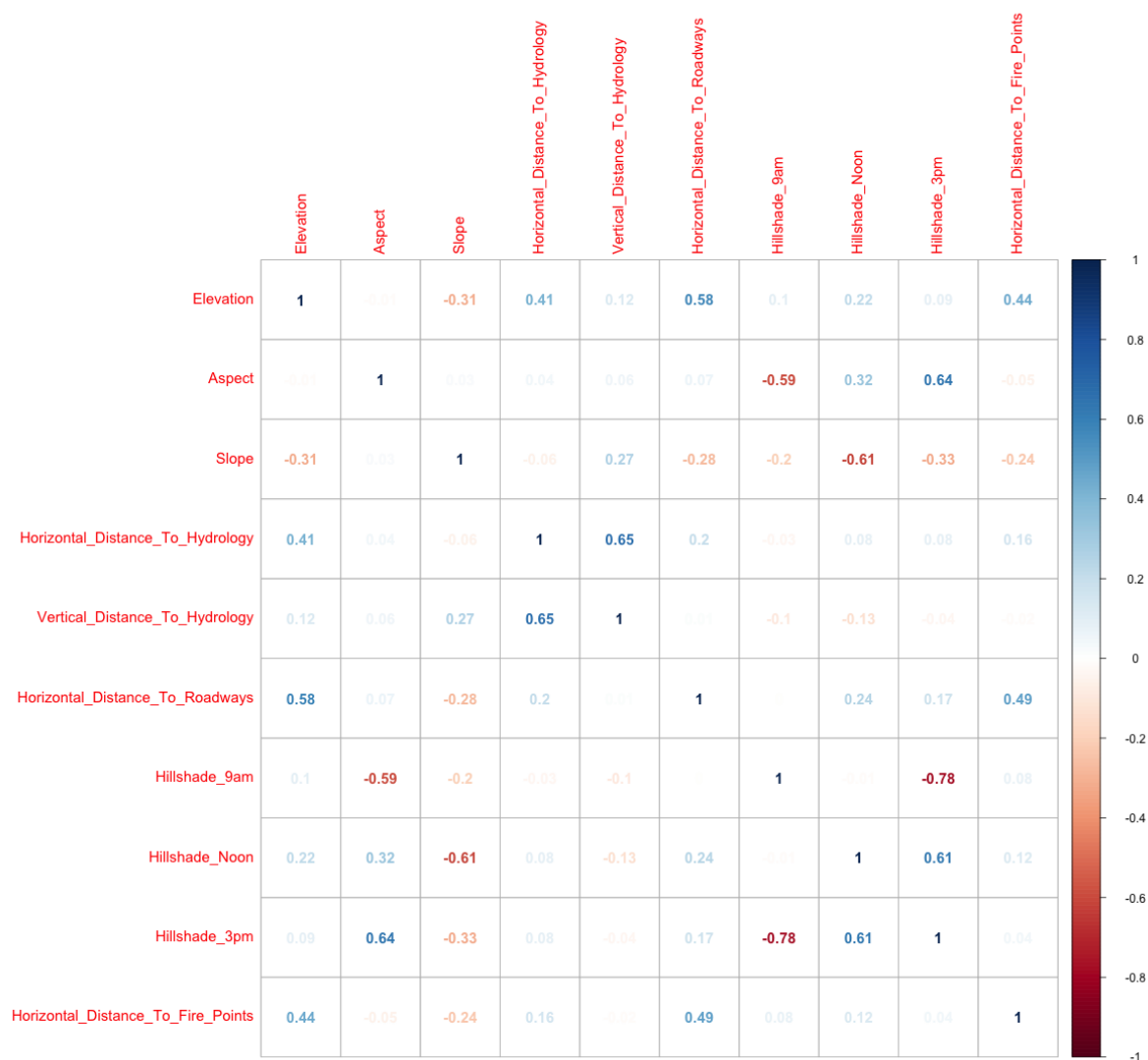
Z wykresów pudełkowych można zobaczyć, że większość atrybutów słabo rozdziela klasę docelową. Największą zależność, pod względem przypisania do klasy docelowej, wykazuje atrybut *Elevation*. Można więc się spodziewać, że na przykład w przypadku zastosowania algorytmu C4.5 będzie on stanowił atrybut wyjściowy, od którego algorytm będzie zaczynać budowanie drzewa, a dopiero analiza kolejnych wartości atrybutów pozwoli na dokładniejsze zaklasyfikowanie próbki. Pozostałe atrybuty mają ten sam zakres wartości dla różnych klas docelowych.

Interpretację poprzedniego wykresu potwierdza również test ważności atrybutów χ^2 . Atrybut *Elevation* ma największą wartość, która wynosi 0.613.

Lp.	Nazwa atrybutu	chi.square
1	Elevation	0.613
2	Soil_Type	0.600
3	Wilderness_Area	0.521
4	Horizontal_Distance_To_Roadways	0.301
5	Horizontal_Distance_To_Fire_Points	0.257
6	Horizontal_Distance_To_Hydrology	0.223
7	Hillshade_9am	0.200
8	Vertical_Distance_To_Hydrology	0.185
9	Slope	0.161
10	Hillshade_3pm	0.156
11	Aspect	0.155
12	Hillshade_Noon	0.120

Kolejnym etapem analizy, było sprawdzenie korelacji pomiędzy atrybutami. Idealnie, wszystkie atrybuty powinny być nieskorelowane (zwłaszcza dla klasyfikatora bayesowskiego, który zakłada taką właściwość). Gdyby atrybuty były silnie skorelowane, opieranie się na ich obu wartościach byłoby zbędne i wystarczyłoby wykorzystać tylko jeden atrybut przy modelu klasyfikacji.

Ustalono więc arbitralnie, że atrybuty skorelowane powyżej 75% mogą zostać usunięte (z pary takich atrybutów, do klasyfikacji użyjemy jednego z nich).



Korelacja powyżej 0.5 występuje dla następujących par atrybutów:

- Elevation, Horizontal_Distance_To_Roadways - 0,54
- Aspect, Hillshade_9am - -0,59
- Aspect, Hillshade_3pm - 0,64
- Slope, Hillshade_Noon - 0,61
- Horizontal_Distance_To_Hydrology, Vertical_Distance_To_Hydrology - 0,65

- Hillshade_9am, Hillshade_3am - 0,78
- Hillshade_9am, Hillshade_Noon - 0,61

Najwyższa wartość korelacji występuje między atrybutami *Hillshade_9am* i *Hillshade_3am* i wynosi 78%. Przy tak dużym stopniu korelacji, należałoby usunąć atrybut *Hillshade_3am*, jako że jest on najmniej istotny powołując się na test χ^2 , co usprawni działanie algorytmu i pomoże zapobiec zbytniemu dopasowywaniu się do modelu do danych treningowych.

5.2 Stworzone modele

Wszystkie modele zostały stworzone i testowane na połowie danych treningowych. Po została połowa danych zostanie użyta do ostatecznej oceny modelu - została ona odseparowana, aby modele w żaden sposób nie dopasowywały się do tej części danych.

5.2.1 Algorytm C4.5

Algorytm C4.5 został zaimplementowany jako funkcja *J48* w pakiecie RWeka. Model, oprócz danych treningowych, przyjmuje również jako argumenty opcje, którymi można dostroić algorytm. Główne parametry budowania klasyfikatora C4.5 to:

- *U* - Stosowanie nieprzyciętych drzew
- *O* - Nie zapadaj drzewa
- *C* - Granica prawdopodobieństwa rozkładu klas dla którego przycinamy drzewo (domyślnie 0,25)
- *M* - Minimalna liczba przykładów, aby utworzyć liść w drzewie (domyślnie 2)
- *R* - Używaj przycinania na podstawie redukcji błędu
- *N* - Na ile zbiorów powinny zostać podzielone dane treningowe, jeśli używamy przycinania na podstawie redukcji błędu (domyślnie 3)
- *B* - Używaj jedynie podziałów binarnych w drzewie

Tworzenie samego modelu drzewa decyzyjnego zostało przetestowane dla różnych opcji:

- wszystkie opcje klasyfikatora z domyślnymi wartościami

Correctly Classified Instances	9311	61.5807 %
Incorrectly Classified Instances	5809	38.4193 %

- (Weka control: Do not collapse tree, O = TRUE)

Correctly Classified Instances	9311	61.5807 %
Incorrectly Classified Instances	5809	38.4193 %

- tworzenie jedynie binarnych podziałów (Weka control: Use binary splits only, B = TRUE)

Correctly Classified Instances	12529	82.8638 %
Incorrectly Classified Instances	2591	17.1362 %

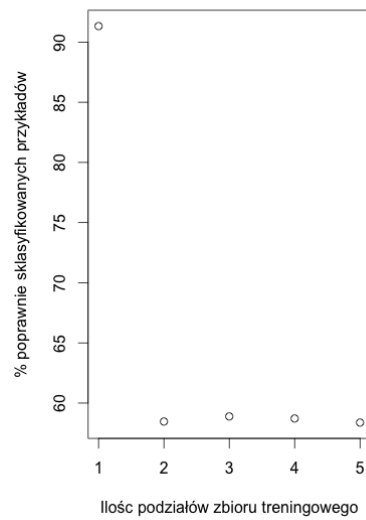
- nieprzycinanie drzew (Weka control: Use unpruned tree, U = TRUE)

Correctly Classified Instances	13809	91.3294 %
Incorrectly Classified Instances	1311	8.6706 %

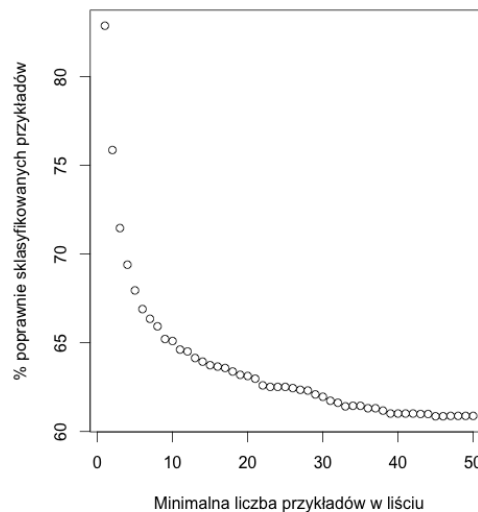
Dodatkowo sprawdzono, czy usunięcie atrybutu *Hillshade_3am* (mocno skorelowany z *Hillshade_noon*) wpływa na stabilność atrybutu. Wyniki bez użycia tego atrybutu były porównywalne, ale jednak minimalnie lepsze.

Spośród przetestowanych modeli wybrano początkowo pozornie najlepszy. Model ten został następnie przetestowany za pomocą metody cross-validation: dzielono zestaw danych treningowych na k podzbiorów o równej liczności, a następnie uczono model na jednym z tych zbiorów i testowano na $(k-1)$ pozostałych. Przetestowano tą metodę dla każdego $k \in \{1, \dots, |training_data|\}$. Sprawdzono w ten sposób stabilność algorytmu.

Wydawałoby się, że najlepsze wyniki zostały uzyskane przy zablokowaniu przycinania drzew - daje ono największy odsetek dobrze sklasyfikowanych przykładów. Istnieje jednak niebezpieczeństwo, że jest to kwestia nadmiernego dopasowania drzewa do danych treningowych. Za pomocą metody cross-validation widać, że obawy potwierdzają się - przy dzieleniu danych na mniejsze zbiory, jakość klasyfikatora natychmiast spada. Poniżej przedstawiono wykres zależności stopnia poprawnie sklasyfikowanych przykładów od ilości zbiorów na które podzielono zbiór treningowy.



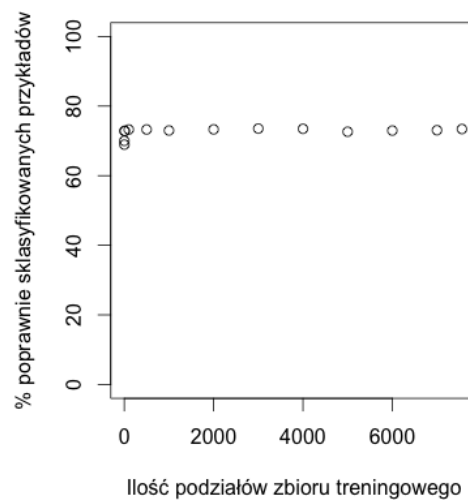
W związku z tym, za kolejną najlepszą metodę uznano uzyskiwanie drzewa jedynie przy podziałach binarnych. Następnym etapem jest maksymalne przycięcie drzewa, które jednocześnie nie powodowałoby dużej straty jakości modelu. Jest to ustalane za pomocą opcji *M*, które ustala minimalną liczbę przykładów, dla których może być utworzony liść - domyślnie jest to liczba 2. Powiększanie tej wartości ma mocny wpływ na jakość modelu, zwłaszcza dla początkowych wartości, a później zaczyna się stabilizować (przy wartości $M \geq 10$) - co można zobaczyć na wykresie poniżej.



W związku z tym należy wybrać taką liczbę liści, która będzie miała mniejszą skuteczność, ale w zamian model będzie cechować się większą stabilnością.

Ostatecznie, wybrano model łączący kilka ustawień. Drzewo ma zastosowane tylko podziały binarne i stosuje przycinanie na podstawie redukcji błędu. Uczenie na podstawie redukcji błędu odbywa się na danych treningowych podzielonych na 5 zbiorów i zakłada minimalną liczbę przykładów w liściu równą 10. Można założyć, że przy testowaniu modelu na danych testowych wyniki poprawią się, więc obecna jakość na poziomie 70-75% wydaje się zadowalająca.

Ilość poprawnie sklasyfikowanych przykładów przy podanych paramaterach dla rosnącej liczby zbiorów metody cross-validation jest ustabilizowana i wygląda następująco:



5.2.2 Algorytm k-najbliższych sąsiadów

Algorytm k-najbliższych sąsiadów został zaimplementowany jako funkcja *IBk* w pakiecie RWeka. Model, oprócz danych treningowych, przyjmuje również jako argumenty opcje, którymi można dostroić algorytm. Główne parametry budowania algorytmu k-najbliższych sąsiadów to:

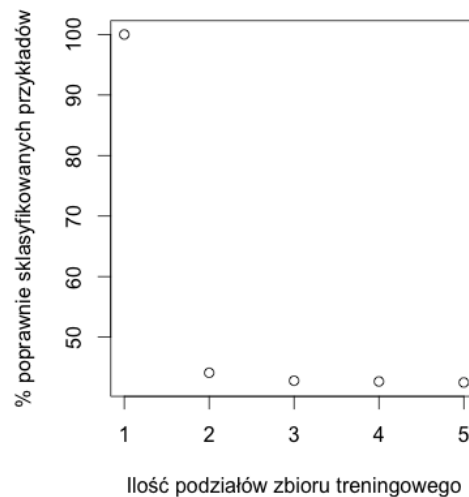
- K - liczba najbliższych sąsiadów branych pod uwagę przy klasyfikacji (domyślnie 1)
- I - waga sąsiadów definiowana jako odwrotność odległości pomiędzy wierzchołkiem, a badanym sąsiadem
- F - waga sąsiadów definiowana jako 1 - odległość pomiędzy wierzchołkiem, a badanym sąsiadem

Tworzenie samego modelu klasyfikacji, podobnie jak w przypadku drzew decyzyjnych, zostało przetestowane dla różnych opcji:

- wszystkie opcje klasyfikatora z domyślnymi wartościami

Correctly Classified Instances	15120	100 %
Incorrectly Classified Instances	0	0 %

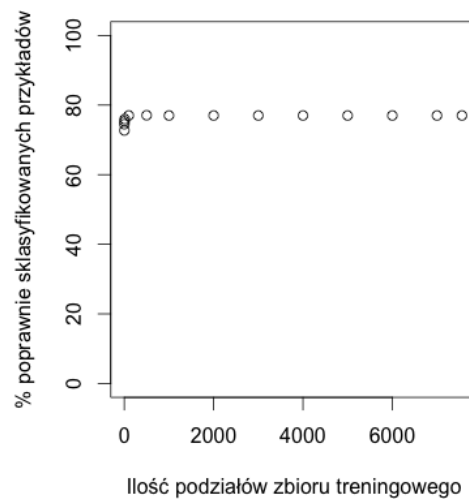
Model dający idealną klasyfikację jest jednak zbyt dopasowany, co udowodnić można poprzez sprawdzenie modelu metodą cross-validation. Dla coraz większej ilości podziałów zbioru treningowego, jego jakość drastycznie spada:



Przy zwiększaniu liczby branych pod uwagę sąsiadów, jakość klasyfikatora również spada, ale nie tak drastycznie:



Możnaby więc wybrać kompromis, i zmniejszyć jakość modelu, poprawiając jego stabilność. Jako najlepszy, został wybrany model biorący pod uwagę 4 najbliższe wierzchołki. Dla takich parametrów, algorytm radzi sobie bardzo dobrze przy coraz większej ilości podziałów zbiorów treningowych i zachowuje stałą wartość poprawnie sklasyfikowanych przykładów:



Model klasyfikuje przykłady poprawnie na poziomie 74-77%, więc zachowuje się on trochę lepiej niż drzewo decyzyjne.

5.2.3 Naiwny klasyfikator Bayesowski

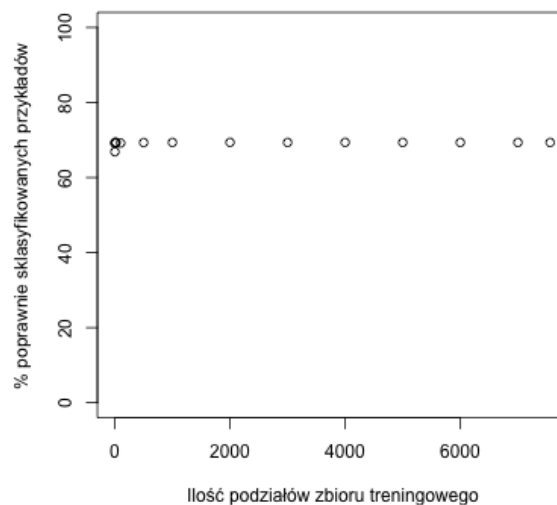
Algorytm klasyfikatora Bayesowskiego został zaimplementowany jako funkcja `"weka/classifiers/bayes/NaiveBayes"` w pakiecie RWeka. Korzysta on z pakietu, który został opisany w dokumentacji początkowej, e1017.

Parametry, które mogą pozwolić na lepsze wyniki algorytmu:

- K - dla atrybutów ciągłych, używać funkcji gęstości zamiast rozkładu normalnego
- D - dyskretyzacja atrybutów ciągłych

Zastosowanie obydwu opcji daje lepsze i jednocześnie porównywalne wyniki. Ze względu na specyfikę danych - tzn. większość atrybutów, to atrybuty ciągłe, wybrano zastosowanie funkcji gęstości.

Model zachowuje się całkiem stabilnie i nie zmienia praktycznie swojej skuteczności przy zwiększaniu podziałów w metodzie cross-validation. Jednocześnie, daje najgorsze wyniki, na poziomie 69% - prawdopodobnie jest to skutek występujących między atrybutami korelacji (nie bierzemy pod uwagę tylko jednego atrybutu z bardzo wysoką korelacją, pozostałe zostały w zbiorze rozpatrywanych atrybutów).



5.3 Ocena stworzonych modeli

Modele zostaną ocenione według następujących miar w dwóch kategoriach:

1. Trafność klasyfikacji (Classification / Predictive accuracy)

(a) Ogólna trafność klasyfikacji

Ogólne statystyki związane z predykcją modelu.

(b) Wyniki względem klas

Statystyki związane z predykcją poszczególnych klas: miary jakości takie jak TP rate, FP rate.

(c) Macierz pomyłek

Kolumny macierzy oznaczają kolejne klasy przykładów, a wiersze klasy oznaczone przez model. Dana komórka w macierzy oznacza ile przykładów danej klasy model oznaczył jako inną klasę - zależy nam, aby wartości dla wierszy i kolumn oznaczających tę samą klasę były jak największe.

(d) Macierz pomyłek podczas testowania stabilności

Podczas tworzenia modelu, dane były testowane w ramach stabilności i wydajności. Kolumny macierzy oznaczają kolejne klasy przykładów, a wiersze klasy oznaczone przez model. Dana komórka w macierzy oznacza ile przykładów danej klasy model oznaczył jako inną klasę - zależy nam, aby wartości dla wierszy i kolumn oznaczających tę samą klasę były jak największe.

Modele zostały ocenione poprzez testowanie na połowie dostępnych danych treningowych. Dane te zostały zupełnie odseparowane od pozostałych, na których modele uczyły się, aby zapewnić obiektywną ocenę modeli.

5.3.1 Algorytm C4.5

Trafność klasyfikacji

Overall Statistics	
Name	Value
Accuracy	0.7575
95% CI	(0.7477, 0.7672)
No Information Rate	0.1475
P-Value [Acc i NIR]	2.2e-16
Kappa	0.7171
Mcnemar's Test P-Value	NA

Wyniki względem klas

	Class: 1	Class: 2	Class: 3	Class: 4	Class: 5	Class: 6	Class: 7
TPRate	0.60538	0.57009	0.66289	0.9519	0.8631	0.7173	0.9346
TNRate	0.96043	0.94530	0.94832	0.9789	0.9602	0.9511	0.9729
Pos Pred Value	0.72581	0.63212	0.67630	0.8851	0.7864	0.7094	0.8460
Neg Pred Value	0.93363	0.93025	0.94526	0.9917	0.9764	0.9529	0.9894
Prevalence	0.14749	0.14153	0.14008	0.1456	0.1450	0.1427	0.1376
Detection Rate	0.08929	0.08069	0.09286	0.1386	0.1251	0.1024	0.1286
Detection Prevalence	0.12302	0.12765	0.13730	0.1566	0.1591	0.1443	0.1520
Balanced Accuracy	0.78291	0.75770	0.80560	0.9654	0.9117	0.8342	0.9537

Macierz pomyłek

	Reference						
Prediction	1	2	3	4	5	6	7
1	675	193	0	0	0	0	62
2	233	610	18	0	87	14	3
3	2	34	702	37	36	227	0
4	0	0	91	1048	0	45	0
5	40	174	21	0	946	19	3
6	12	35	227	16	27	774	0
7	153	24	0	0	0	0	972

Macierz pomyłek - test stabilności

Klasa vs Klasa przewidziana	1	2	3	4	5	6	7
1	608	245	3	0	32	1	156
2	206	659	21	0	132	44	28
3	0	10	707	113	24	247	0
4	0	0	53	952	0	54	0
5	0	249	24	0	762	29	0
6	0	25	316	68	41	631	0
7	130	14	0	0	2	0	974

5.3.2 Algorytm k-najbliższych sąsiadów

Trafność klasyfikacji

Overall Statistics	
Name	Value
Accuracy	0.8126
95% CI	(0.8036, 0.8213)
No Information Rate	0.1475
P-Value [Acc i NIR]	2.2e-16
Kappa	0.7813
Mcnemar's Test P-Value	NA

Wyniki względem klas

	Class: 1	Class: 2	Class: 3	Class: 4	Class: 5	Class: 6	Class: 7
TPRate	0.7695	0.5850	0.8036	0.9455	0.9097	0.7405	0.9337
TNRate	0.9483	0.9715	0.9579	0.9797	0.9675	0.9688	0.9874
Pos Pred Value	0.7204	0.7719	0.7564	0.8882	0.8260	0.7982	0.9221
Neg Pred Value	0.9596	0.9342	0.9677	0.9906	0.9844	0.9573	0.9894
Prevalence	0.1475	0.1415	0.1401	0.1456	0.1450	0.1427	0.1376
Detection Rate	0.1135	0.0828	0.1126	0.1377	0.1319	0.1057	0.1284
Detection Prevalence	0.1575	0.1073	0.1488	0.1550	0.1597	0.1324	0.1393
Balanced Accuracy	0.8589	0.7783	0.8807	0.9626	0.9386	0.8547	0.9605

Macierz pomyłek

Klasa vs Klasa przewidziana	1	2	3	4	5	6	7
1	858	264	0	0	7	4	58
2	118	626	7	0	43	8	9
3	0	17	851	40	28	189	0
4	0	2	70	1041	0	59	0
5	59	113	16	0	997	20	2
6	10	36	115	20	21	799	0
7	70	12	0	0	0	0	971

Macierz pomyłek - test stabilności

Klasa vs Klasa przewidziana	1	2	3	4	5	6	7
1	768	129	3	0	30	6	109
2	324	578	23	1	98	39	27
3	1	11	912	66	12	99	0
4	0	0	20	992	0	47	0
5	139	81	29	0	718	97	0
6	10	38	153	61	51	768	0
7	128	89	0	0	6	0	897

5.3.3 Naiwny klasyfikator Bayesowski

Trafność klasyfikacji

Overall Statistics	
Name	Value
Accuracy	0.7155
95% CI	(0.7052, 0.7256)
No Information Rate	0.1475
P-Value [Acc i NIR]	2.2e-16
Kappa	0.668
Mcnemar's Test P-Value	NA

Wyniki względem klas

	Class: 1	Class: 2	Class: 3	Class: 4	Class: 5	Class: 6	Class: 7
TPRate	0.63229	0.54673	0.51747	0.9092	0.8002	0.69880	0.9029
TNRate	0.94414	0.94576	0.95785	0.9701	0.9449	0.93242	0.9727
Pos Pred Value	0.66197	0.62433	0.66667	0.8384	0.7113	0.63255	0.8406
Neg Pred Value	0.93687	0.92677	0.92416	0.9843	0.9654	0.94896	0.9843
Prevalence	0.14749	0.14153	0.14008	0.1456	0.1450	0.14272	0.1376
Detection Rate	0.09325	0.07738	0.07249	0.1324	0.1160	0.09974	0.1242
Detection Prevalence	0.14087	0.12394	0.10873	0.1579	0.1631	0.15767	0.1478
Balanced Accuracy	0.78821	0.74625	0.73766	0.9396	0.8726	0.81561	0.9378

Macierz pomyłek

	Reference						
Prediction	1	2	3	4	5	6	7
1	705	234	0	0	33	0	93
2	205	585	7	0	114	22	4
3	1	19	548	61	19	174	0
4	0	0	130	1001	0	63	0
5	45	165	76	0	877	66	4
6	5	43	298	39	53	754	0
7	154	24	0	0	0	0	939

Macierz pomyłek - test stabilności

Klasa vs Klasa przewidziana	1	2	3	4	5	6	7
1	582	253	1	0	49	4	156
2	193	661	20	0	166	30	20
3	0	10	604	155	83	249	0
4	0	0	58	950	0	51	0
5	56	285	14	0	657	52	0
6	0	97	223	92	67	602	0
7	81	38	0	0	1	0	1000

5.4 Podsumowanie

Opierając się na wynikach na danych testowych, z klasyfikacją najlepiej poradził sobie model oparty na algorytmie k-najbliższych sąsiadów. Choć sklasyfikował on średnio 81% przykładów poprawnie, z wyników poszczególnych klas widać, że różne klasy klasyfikował z inną skutecznością. Przykładowo, prawidłowo przykłady z klasy 4 w 95% przypadków, ale klasę 2 już tylko w 59%.

Najgorzej w porównaniu wypadł klasyfikator Bayesowski, zgodnie z przewidywaniami. W przypadku dużej ilości atrybutów, które są ze sobą skorelowane, podstawowe założenia algorytmu są niespełnione, więc trudno by algorytm dawał najlepsze wyniki.

5.5 Pakiet R

Pakiet został nazwany *forestPredictionPkg*. Po zainstalowaniu pakietu ze źródeł, funkcja *start()* jest odpowiedzialna za załadowanie wymaganych bibliotek oraz skryptów.

Następnie wywoływana jest w niej funkcja *run_classifiers*, która dane na zbiór treninowy i testowy, a następnie wywołuje odpowiednie generatory opracowywanych modeli. Podczas generowania modeli, generuje również statystyki dotyczące tych modeli. Następnie modele są zapisywane w postaci plików z rozszerzeniem *.Rds*.

Po wygenerowaniu modeli, w funkcji *start()* zostaje wywołana kolejna funkcja, *test_classifiers()*, która odpowiada za przetestowanie wygenerowanych modeli na zestawie danych testowych, poprzez predykcję klas dla przykładów, a następnie opracowanie wartości miar jakości opisanych wcześniej w dokumentacji.