

Uniwersytet Warszawski

Wydział Nauk Ekonomicznych

Wojciech Guszyła

Nr albumu: P-20611

**Analiza wybranych zbiorów danych oraz implementacja metod
uczenia maszynowego w pakiecie Shiny**

Praca dyplomowa na kierunku 'Data Science w zastosowaniach biznesowych. Warsztaty z wykorzystaniem programu R'

Recenzent

dr Piotr Wójcik

Wydział Nauk Ekonomicznych Uniwersytetu Warszawskiego

Wrzesień 2019

Świadom odpowiedzialności prawnej oświadczam, że nieniejsza praca dyplomowa została napisana przeze mnie samodzielnie i nie zawiera treści uzyskanych w sposób niezgodny z obowiązującymi przepisami.

30 wrzesień 2019

Wojciech Guszyła

1. Streszczenie

Aplikacja, którą opisuje niniejszy dokument, umożliwia użytkownikom analizę statystyczną i graficzną niektórych zbiorów danych dostępnych w pakiecie R oraz zastosowanie na nich wybranych metod uczenia maszynowego (ML) w celu wytrenowania algorytmów, które następnie stosują stworzony model do predykcji danych dotychczas niewidzianych. Użytkownicy mają również możliwość wczytania własnych plików w formacie Comma Separated Value (csv).

W ekosystemie R istnieje ogromna liczba pakietów dostępnych publicznie, które umożliwiają badaczom zrozumienie zależności między danymi. Jednakże stopień skomplikowania jakim cechuje się język R sprawia, że owe pakiety mogą być trudne w użyciu przez mniej doświadczonych użytkowników.

Autor pracy, używając biblioteki webowej Shiny (<https://shiny.rstudio.com>), stworzył aplikację z interfejsem graficznym, który umożliwia badaczom wykonywanie analiz bez znajomości zasad programowania.

Program ma na celu również zademonstrowanie i potwierdzenie umiejętności, które Autor nabył podczas studiów podyplomowych na kierunku 'Data Science w zastosowaniach biznesowych. Warsztaty z wykorzystaniem programu R' na Wydziale Nauk Ekonomicznych Uniwersytetu Warszawskiego.

Kod źródłowy aplikacji został opublikowany na portalu Github pod poniższym adresem:

https://github.com/wojtekgoo/data_science/tree/master/UW_thesis

Działająca wersja jest również uruchomiona na portalu Shinyapps:

https://wojtekgoo.shinyapps.io/UW_thesis/

2. Spis treści

1.	Streszczenie	3
2.	Spis treści.....	4
3.	Struktura aplikacji.....	6
4.	Zakładka 'Dataset'	8
	Wczytywanie danych.....	8
	Zamiana danych na NA.....	9
	Konwersja typów zmiennych.....	10
	Zamiana poziomów zmiennej jakościowej.....	11
	Usuwanie wierszy	11
	Analiza wartości brakujących	12
	Usuwanie zmiennych odstających.....	12
	Eksport danych do pliku CSV	14
5.	Zakładka 'Stats'	15
	Transformacja zmiennych	15
	Wykres rozrzutu	16
	Wykres pudełkowy i wykresy częstości.....	16
	Macierz wykresów.....	18
6.	Zakładka 'ML'	20
6.1	Regresja liniowa	20
	Wybór zmiennych.....	20
	Walidacja krzyżowa	21
	Trening modelu	21
	Predykcja	23
6.2	Regresja logistyczna dwumianowa.....	24
	Wybór zmiennych.....	24
	Trening modelu	24
	Predykcja	25
6.3	Liniowa Analiza Dyskryminacyjna.....	27
	Wybór zmiennych.....	27
	Trening modelu	27
	Predykcja	29
6.4	Drzewo decyzyjne.....	30
	Wybór zmiennych.....	30

Trening modelu	30
Predykcja	32

3. Struktura aplikacji

Aplikacja będąca przedmiotem niniejszej dokumentacji została stworzona z wykorzystaniem pakietu Shiny¹, który umożliwia tworzenie stron internetowych zasilanych przez działający na serwerze program napisany w języku R. Użytkownicy aplikacji mogą poprzez stronę HTML wybierać parametry przetwarzania, które ma miejsce na serwerze a jego wynik jest przedstawiany ponownie na stronie internetowej².

Uwaga: aplikacja oraz komentarze są napisane w języku angielskim. Praca ma na celu zademonstrowanie umiejętności autora w zakresie analizy danych, również przyszłym potencjalnym pracodawcom, dlatego kod źródłowy został umieszczony w publicznym repozytorium na portalu Github a aplikacja działa również na serwerze <https://www.shinyapps.io> – odnośniki do obu dokumentów będą widoczne w CV Autora. Dlatego w celu ułatwienia poruszania się po aplikacji osobom obcojęzycznym, Autor zdecydował o utrzymaniu wersji angielskiej.

Struktura aplikacji (format i zawartość zakładki) zapisana jest w głównym pliku Shiny 'app.R'. W pozostałych plikach z rozszerzeniem .R zawarte są poszczególne moduły wywoływane podczas działania programu.

Aplikacja po uruchomieniu instaluje wszystkie pakiety potrzebne do dalszego działania. Funkcja 'pkg', zdefiniowana przez Autora, sprawdza najpierw czy dany pakiet jest już dostępny w środowisku użytkownika. Jeżeli tak, to zostaje on załadowany bez konieczności ponownej instalacji. Jeżeli pakiet jest niedostępny, zostaje on najpierw pobrany z repozytorium R (<https://cran.r-project.org/mirrors.html>), zainstalowany a następnie załadowany funkcją 'library' – wymagany jest dostęp do internetu.

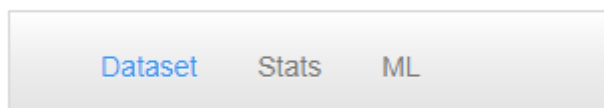
```
pkg = function(x)
{
  if ( !require(x, character.only = TRUE) )
  {
    if ( !(x %in% installed.packages()) )
    {
      print( paste0("[+] ", x, " package not installed. Installing..." ) )
      install.packages(x)
      library(x, character.only = TRUE)
    }
    else
    {
      if( !library(x) )
        print( paste0("[+] ", x, " package installed but could not be loaded" ) )
    }
  }
  else
  {
    print( paste0("[+] ", x, " package loaded" ) )
  }
}
```

¹ <https://shiny.rstudio.com>

² https://pbiecek.gitbooks.io/przewodnik/content/Programowanie/jak_tworzyc_aplikacje.html

Po uruchomieniu aplikacji użytkownikowi wyświetla się interfejs graficzny, składający się z trzech zakładek (tabów):

- Dataset
- Stats
- ML



Po uruchomieniu programu jesteśmy w części 'Dataset', w której mamy możliwość wyboru jednego z predefiniowanych zbiorów danych lub załadowania własnego pliku w formacie CSV. Po dokonaniu wyboru dane te będą dostępne do analizy w pozostałych dwóch częściach aplikacji (Stats i ML). Zakładka 'Dataset' umożliwia także dokonanie niektórych podstawowych operacji na danych, opisanych w punkcie 4. i zmodyfikowanie ich przed dalszą analizą.

W tabie 'Stats' dostępnych jest kilka statystyk opisowych oraz graficznych, które pozwalają zrozumieć zależności w wybranym zbiorze danych. Użytkownik ma możliwość interakcji ze zmiennymi i tworzenie wykresów wedle uznania, manipulując dostępnymi parametrami. Zmiany tu dokonane nie są zapisywane i nie zostaną uwzględnione przy przejściu do zakładki 'ML' – w takim przypadku konieczna jest modyfikacja początkowego zbioru danych, przed załadowaniem go do aplikacji w tabie 'Dataset'.

Szczegółowy opis zakładki 'Stats' znajduje się w punkcie 5.

Zakładka 'ML' umożliwia wybór X modeli Machine Learning'owych i zastosowanie ich do wczytanych danych. Użytkownik decyduje jaki procent zbioru jest przeznaczony na dane treningowe (train set) a jaki na dane testowe (test set). Model, po „nauczeniu się danych”, dokonuje predykcji i wyświetla metryki opisujące jej dokładność.

Szczegółowy opis zakładki 'ML' znajduje się w punkcie 6.

4. Zakładka 'Dataset'

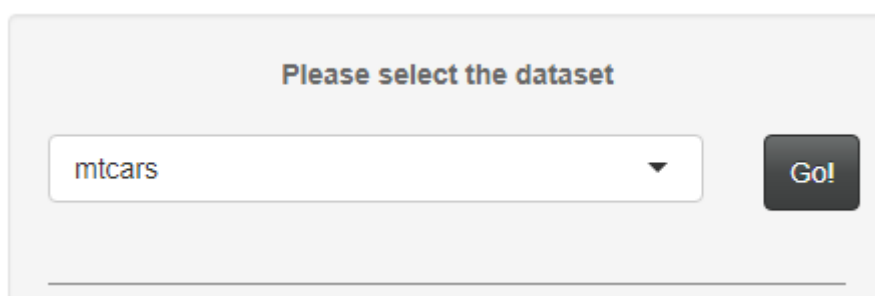
Wczytywanie danych

Zakładka pozwala na wczytanie jednego z kilku zbiorów danych dostępnych w podstawowym pakiecie R 'datasets'

- *aiqquality, iris, mtcars, rock, quakes*

oraz zbiorów danych z pakieru 'ISLR':

- *Auto, Caravan, Carseats, College, Credit, Defaults, Hitters, Khan, NCI60, OJ, Portfolio, Smarket, Wage, Weekly*



The screenshot shows a web interface for selecting a dataset. At the top, it says "Please select the dataset". Below this is a dropdown menu with "mtcars" selected. To the right of the dropdown is a dark button labeled "Go!".

łącznie jest to 19 możliwości. Zbiory te zostały wybrane przez Autora pod kątem ich przydatności w dalszej analizie, na podstawie ilości obserwacji oraz ilości i rodzaju zmiennych. Zbiory danych z małą ilością wierszy, danymi nie nadającymi się do faktoryzacji i niewieloma kolumnami zostały odrzucone. Pozwoliło to na wykorzystanie w pełni możliwości aplikacji i pokazanie całego spektrum jej zastosowań.

Użytkownik ma również możliwość wgrania własnego zbioru danych w formacie CSV i wyboru:

- separatora oddzielającego zmienne (przecinek, tabulator, średnik)
- czy plik posiada wiersz nagłówkowy
- poinstruowania R o traktowaniu tekstu jako zmienne jakościowe (factor)
- poinformowaniu R czy w danych występują cudzysłowy (nie, tak – pojedynczy cudzysłów, tak – podwójny cudzysłów)

Or upload your own CSV file

Browse...

No file selected

☒ Header

☐ Strings as factors

Separator

☒ Comma☐ Semicolon☐ Tab

Quote

☒ None☐ Double Quote☐ Single Quote

Zamiana danych na NA

Po wczytaniu danych jednym lub drugim sposobem można zamienić wybrane wartości na 'NA'. Stała NA w języku R oznacza brak danych i pozwoli nam na ich późniejsze pominięcie. Przykładem są tu wartości odstające (*outliers*), których chcemy się pozbyć z analizy.

Użytkownik najpierw zaznacza w tabeli kolumny, z których chce się pozbyć danej wartości, potem wpisuje w polu tekstowym w bocznym panelu wartość, która ma zostać zamieniona na NA, a następnie klika przycisk 'Go!'

Please select the dataset

mtcars

Go!

Or upload your own CSV file

Browse...

No file selected

☒ Header

☐ Strings as factors

 Separator

☒ Comma

☐ Semicolon

☐ Tab

 Quote

☒ None

☐ Double Quote

☐ Single Quote

Change value to NA

Enter value

4

Go!

mpg

cyl

Mazda RX4	21	6
Mazda RX4 Wag	21	6
Datsun 710	22.8	
Hornet 4 Drive	21.4	6
Hornet Sportabout	18.7	8
Valiant	18.1	6
Duster 360	14.3	8
Merc 240D	24.4	
Merc 230	22.8	
Merc 280	19.2	6

mpg

cyl

Showing 1 to 10 of 32 entries

Missing values

Variable	n
mpg	0
cyl	11
disp	0
hp	0

Outliers

Variable	outliers
mpg	33.9
cyl	no outl.
disp	no outl.
hp	335

Konwersja typów zmiennych

Funkcjonalność 'Convert variable type' pozwala na zamianę danych w wybranych kolumnach na dane jakościowe (*factor*), numeryczne (*numeric*), bądź tekstowe (*character*).

Convert variable type

Select variable:

cyl am gear

Choose type:

factor

Go!

Najpierw z lewego menu wybieramy interesujące nas zmienne, potem z prawego menu docelowy typ a następnie klikamy przycisk 'Go!'

Zamiana poziomów zmiennej jakościowej

Funkcja 'Recode factor levels' ma za zadanie zmienić poziomy w zmiennych jakościowych na nowe, wybrane przez użytkownika.

Recode factor levels

Factor variable:

From:

To:

Go!

cyl ▼

4

10

Najpierw użytkownik z lewego menu wybiera kolumnę, w której chce zmienić wartości, potem w lewym polu tekstowym wpisuje obecnym poziom do zmiany a w prawym polu tekstowym poziom docelowy.

Uwaga: w menu wyboru zmiennych pojawią się jedynie te, które mają typ 'factor'.

Usuwanie wierszy

Użytkownik może zaznaczyć w tabeli wiersze, które mają zostać usunięte a następnie wykonać operację przyciskiem 'Delete'

Show 10 ▼ entries

	mpg ⚙	cyl ⚙	disp ⚙	hp ⚙	drat ⚙	wt ⚙	qsec ⚙
Mazda RX4	21	6	160	110	3.9	2.62	16
Mazda RX4 Wag	21	6	160	110	3.9	2.875	17
Datsun 710	22.8	4	108	93	3.85	2.32	18
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19
Hornet Sportabout	18.7	8	360	175	3.15	3.44	17
Valiant	18.1	6	225	105	2.76	3.46	20
Duster 360	14.3	8	360	245	3.21	3.57	15
Merc 240D	24.4	4	146.7	62	3.69	3.19	
Merc 230	22.8	4	140.8	95	3.92	3.15	2
Merc 280	19.2	6	167.6	123	3.92	3.44	1
	mpg	cyl	disp	hp	drat	wt	qsec

Delete selected rows

Delete

Analiza wartości brakujących

Program analizuje wartości brakujące w zbiorze poniższym kodem:

```
output$showMissing = renderTable({
  req(df$x)
  # common_na_strings has bug: empty "" value spoils miss_scan_count results
  # recreate common_na_strings without ""
  na_strings = c("NA", "N A", "N/A", "NA ", " NA", "N /A", "N / A", "N / A ", "na", "n a", "n/a", "na
", " na", "n /a", "n / a", " a / a", "n / a ", "NULL", "null", "Not Available", "NOt available")
  na = sapply(df$x, function(y) sum(is.na(y))) # this will count NA's
  miss_scan = miss_scan_count(df$x, na_strings) # this won't count NA's
  result = miss_scan
  result$n = miss_scan$n + na # add NA's manually to miss_scan_count results
  result
})
```

Funkcja zlicza wszystkie najczęściej występujące wartości oznaczające obserwację brakującą:

"NA", "N A", "N/A", "NA ", " NA", "N /A", "N / A", "N / A ", "na", "n a", "n/a", "na ", " na", "n /a", "n / a", " a / a", "n / a ", "NULL", "null", "Not Available", "NOt available"

Do tego dodawane są wartości NA i wyświetlany jest rezultat, np.:

Missing values

Variable	n
mpg	0
cyl	11
disp	0
hp	0
drat	0
wt	0
qsec	0
vs	0
am	0
gear	0
carb	10

Usuwanie zmiennych odstających

Aplikacja analizuje załadowany zbiór danych pod kątem wartości odstających (*outliers*) i umożliwia ich usunięcie. Użyta jest do tego funkcja *'find_outliers'*³, która oblicza przedział między pierwszym a trzecim kwartylem (IQR) a następnie wyznacza wartości leżące 1.5 raza niżej niż pierwszy kwartył i 1.5 raza wyżej niż kwartył trzeci:

³ <http://todo-science.blogspot.com/2014/11/remove-outliers-in-r-using.html>

```

find_outliers = function(x, na.rm = TRUE) {
  x <- x[!is.na(x)] # remove NA's
  if(is.factor(x)) return("n/a") # factors not applicable
  if(is.character(x)) return("n/a") # characters not applicable

  qnt <- quantile(x, probs=c(.25, .75), na.rm = TRUE)
  H <- 1.5 * IQR(x, na.rm = na.rm)
  y = x
  low = y[x < (qnt[1] - H)]
  high = y[x > (qnt[2] + H)]
  result = c(low, high)
  if(length(result) > 0) {
    return(paste(result, collapse = " "))
  } else return("no outl.")
}

```

W obliczeniach brane są pod uwagę jedynie kolumny numeryczne – ilościowe i tekstowe są odrzucane. Wyniki funkcji jest wyświetlany w postaci tabeli:

Missing values

Variable	n
mpg	0
cyl	0
disp	0
hp	0
drat	0
wt	0
qsec	0
vs	0
am	0
gear	0
carb	0

Outliers

data
33.9
n/a
no outl.
335
no outl.
5.25 5.424 5.345
22.9
no outl.
no outl.
no outl.
8

- 'n/a' oznacza, że kolumna nie jest numeryczna
- 'no outl.' oznacza, że w kolumnie nie ma wartości odstających

Następnie użytkownik ma możliwość usunięcia wartości odstających i zamiany ich na NA:

Remove outliers

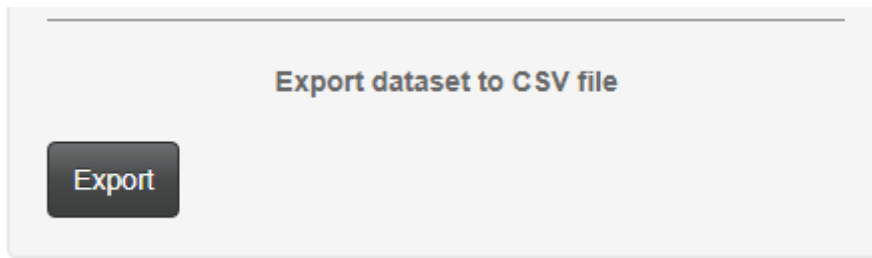
Choose column:

wt

Go!

Eksport danych do pliku CSV

Użytkownik ma możliwość zapisania danych do pliku z rozszerzeniem Comma Separated Value:

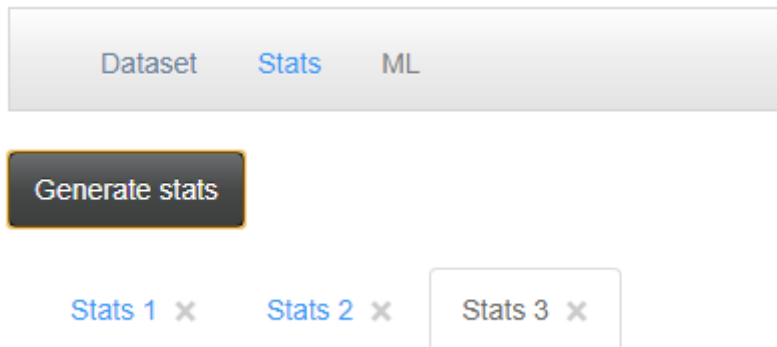


Plik zapisywany jest w katalogu aplikacji pod domyślną nazwą *'exportedDataset.csv'*.

Funkcjonalność ta nie występuje w demonstracyjnej wersji aplikacji na portalu Shinyapps.

5. Zakładka 'Stats'

Początkowo widoczny jest jedynie przycisk 'Generate stats', który tworzy przy każdym naciśnięciu nowy panel ze statystykami:



Panele są numerowane kolejno od 1. Użytkownik ma możliwość ich zamknięcia klikając 'x'.

Do każdego panelu zostanie przekazany zbiór danych załadowany w zakładce 'Dataset'. Aplikacja umożliwia analizę różnych zbiorów w różnych panelach. Wczytanie nowego zbioru danych nie ma wpływu na dotychczas utworzone panele.

Po wygenerowanie statystyk użytkownikowi wyświetla się zestaw modułów, z których każdy zapisany jest w osobnym pliku na dysku i ma inną funkcję. Modułowość ta pozwala na wykorzystanie kodu w innych aplikacjach.

The screenshot displays the 'Stats 1' panel. On the left, under the 'mtcars' dataset, there are two summary tables. The first table is for the 'mpg' variable, showing 32 observations with a mean of 20.09 and a standard deviation of 6.796. The second table is for the 'cyl' variable, showing 32 observations with a mean of 7.47 and a standard deviation of 3.44. On the right, there is a table titled 'Shapiro-Wilk normality test for numeric columns' showing the test results for various numeric variables in the dataset.

Statistic	p-value
mpg	0.9475647 1.228814e-01
dis	0.9200127 2.080657e-02
hp	0.9334153 4.08024e-02
drat	0.9458839 1.100608e-01
wt	0.9432577 9.265499e-02
qsec	0.9732509 5.935176e-01
vs	0.6322635 9.737376e-00
gear	0.5435628 0.000379e-07
carb	0.6680732 7.888032e-06

Pierwsze dwie statystyki dotyczą całego zbioru danych. Moduł po lewej wyświetla podsumowanie każdej zmiennej i zestaw wartości opisujących zmienną.

Po prawej, dla każdej zmiennej numerycznej liczony jest test Shapiro-Wilka, pomagający ocenić czy zmienna ma rozkład zbliżony do rozkładu normalnego.

Transformacja zmiennych

Poniżej, po lewej stronie znajdują się trzy kolejne moduły:

Module : apply_function

Choose variable
mpg

Function
☒ log
☐ abs
☐ sqrt

Apply function !

Module : apply_scale

Choose variable
hp

Scale data

Module : funHistory

- log mpg
- scale hp

- Moduł 'apply_function' pozwala zastosować wybraną transformację (logarytm naturalny, wartość bezwzględna, pierwiastek kwadratowy) do zmiennej numerycznej. Po zastosowaniu funkcji, zbiór danych w pozostałych modułach zostanie odświeżony i przekalkulowany ponownie
- Moduł 'apply_scale' pozwala na skalowanie wybranej zmiennej poprzez odjęcie od każdej wartości średniej arytmetycznej całego wektora i podzielenie przez odchylenie standardowe
- Moduł 'funHistory' tworzy historię dotychczas użytych funkcji z dwóch powyższych modułów

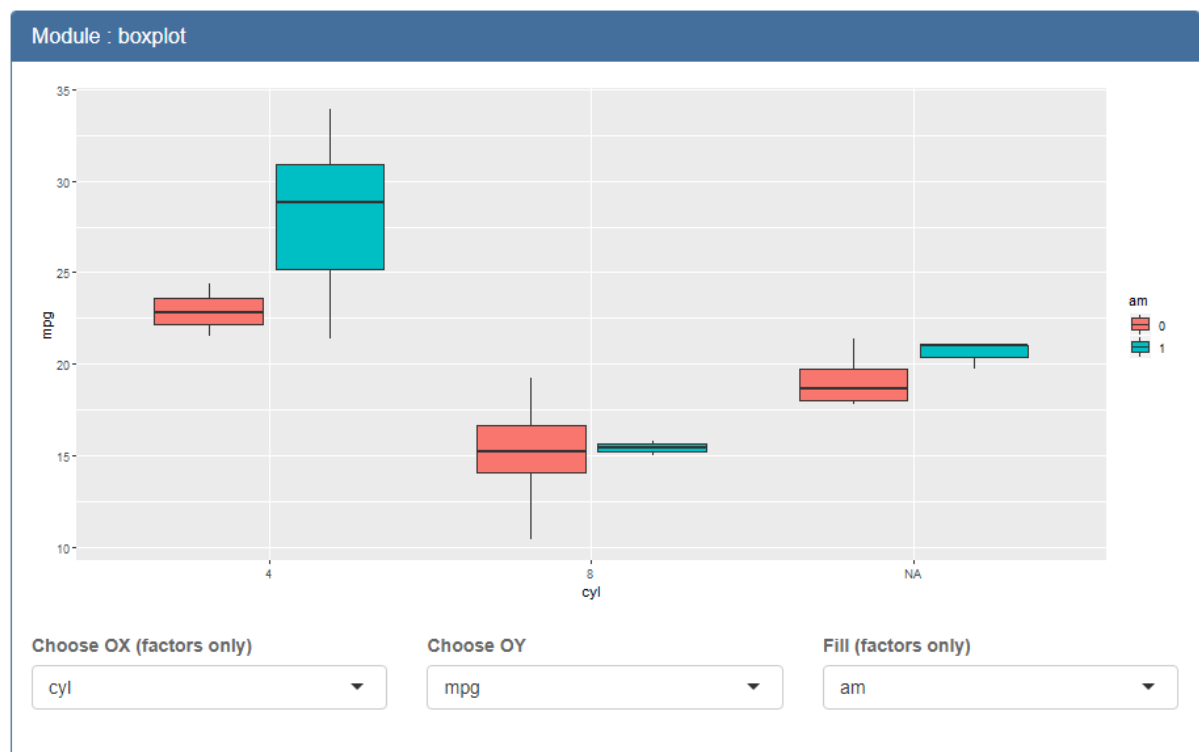
Wykres rozrzutu

Po prawej stronie ekranu możemy narysować wykres rozrzutu (*scatterplot*) dwóch zmiennych na osiach OX i OY, wedle wyboru użytkownika. Dodatkowo moduł pozwala na naniesienie na wykres kolejnych zmiennych za pomocą pokolorowania istniejących punktów (*colour*), zmiany ich rozmiaru (*size*) lub kształtu (*shape*). Użytkownik może również manipulować przezroczystością punktów na wykresie (*alpha/transparency*), co jest przydatne przy dużej ilości danych. Przy rysowaniu wykresu aplikacja automatycznie usuwa wiersze, w których występują wartości NA.

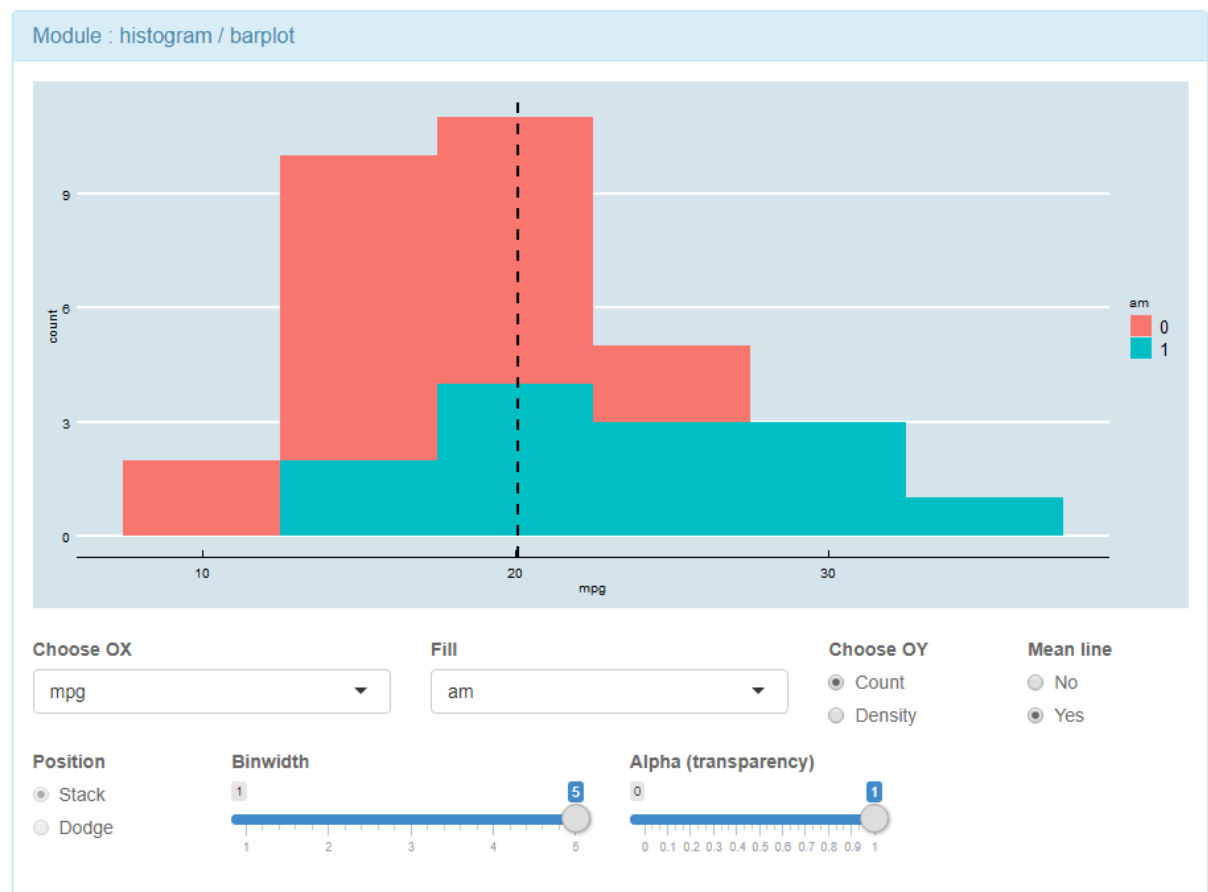
Wykres pudełkowy i wykresy częstości

Kolejne dwa moduły pozwalają na tworzenie wykresu pudełkowego (*boxplot*) i histogramu lub wykresu kolumnowego (*barplot*).

W boxplocie na osi OX możemy wybrać jedynie zmienne jakościowe, podobnie w menu wypełnienia (*fill*). Na osi OY możemy umieścić dowolną zmienną.



W module po prawej aplikacja zwraca dwa typy wykresu: kolumnowy - gdy zmienna na osi OX jest jakościowa, histogram – gdy zmienna na osi OX jest ilościowa



Po wybraniu osi OX pojawiają się dodatkowe opcje. Przykładowo, gdy rysujemy barplot możemy dodać dodatkową zmienną za pomocą wypełnienia (*fill*) i zdecydować czy kolumny mają być wyświetlane sąsiadująco (*position = dodge*) czy nałożone na siebie (*position = stack*). Opcje te są niedostępne przy rysowaniu histogramu.

Gdy w wyborze zmiennej na osi OX zaznaczymy wartość numeryczną, aplikacja narysuje histogram i uaktywni kilka dodatkowych opcji manipulacji wykresem:

- Na osi OY możemy zaznaczyć czy ma być wyświetlana liczba obserwacji (*OY = count*) czy gęstość (*OY = density*)
- Można dodać znacznik wartości średniej zmiennej OX (*mean line = yes*)
- Możemy ustawić pożądaną szerokość przedziałów (suwak *binwidth*). Wartość maksymalna jest liczona automatycznie, w zależności od wartości zmiennej OX, metodą Freedmana-Diaconisa⁴:

```
breaks = pretty(range(unlist(dataset()[input$xvar])), n = nclass.FD(unlist(dataset()[input$xvar])), min.n = 1)
bwidth$max <- breaks[2]-breaks[1]
```

Macierz wykresów

Na samym dole zakładki 'Stats' aplikacja tworzy macierz wykresów pokazujących współzależności między wszystkimi zmiennymi w analizowanym zbiorze danych:



W zależności od typów zmiennych, tworzone będą różne wykresy.

Uwaga: wygenerowanie całej macierzy może zająć dłuższą chwilę – w zależności od mocy obliczeniowej komputera i ilości zmiennych. Dlatego domyślnie jest zaznaczony opcja 'stop refreshing' w lewym

⁴ https://en.wikipedia.org/wiki/Freedman–Diaconis_rule

górnym rogu modułu. Gdy chcemy zobaczyć wykres, należy ją odznaczyć – Shiny zacznie wtedy generować dane.

6. Zakładka 'ML'

Do wyboru mamy jeden z czterech modeli uczenia maszynowego (*machine learning*). Po wybraniu odpowiedniego i kliknięciu 'Generate model', wygenerowany zostanie moduł odpowiedni do wybranego algorytmu.

6.1 Regresja liniowa

Wybór zmiennych

Budowę modelu regresji liniowej zaczynamy od wyboru zmiennej objaśnianej (*outcome variable*). Możliwe do wybrania są jedynie zmienne numeryczne (o klasie *numeric*). Następnie wybieramy zmienne niezależne (*independent variables*) i dodajemy ewentualne interakcje między zmiennymi, np:

- *cyl:hp*
- *cyl + hp*

Select model

Linear regression ▼

Generate model

Model 1 ✕

Create formula

Choose outcome variable

mpg ▼

Choose independent variables

cyl disp hp drat

Add custom term to your model (eg. $x_1 \cdot x_2$ or $x_1 : x_2$)

cyl:hp

Allocate observations to the train set

train set %

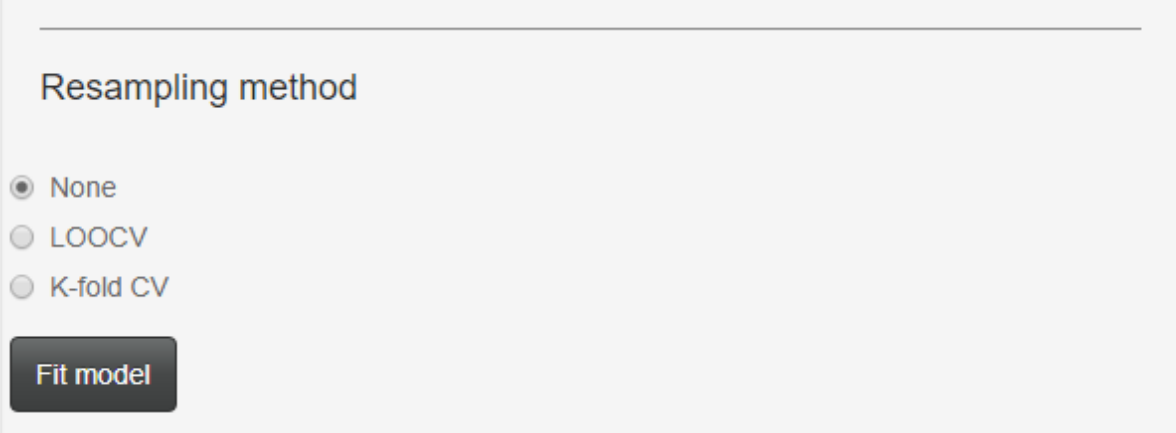
50

Potem decydujemy ile procent obserwacji z analizowanego zbioru danych ma trafić do zbioru uczącego (reszta trafi do zbioru testowego).

Walidacja krzyżowa

Wybierając losowo dane do zbioru uczącego ryzykujemy, że w zbiorze testowym znajdą się obserwacje wyjątkowo trudne lub wyjątkowo łatwe do predykcji. Taki model nie będzie się dobrze sprawdzał gdy zechcemy go użyć do predykcji na nowym zbiorze danych. Do oceny naszego modelu możemy zastosować metody ponownego próbkowania (*resampling methods*). Aplikacja oferuje dwie takie funkcje:

- K-krotną walidację krzyżową (*K-fold CV*): W tej metodzie, oryginalna próba jest dzielona na K podzbiorów. Następnie kolejno każdy z nich bierze się jako zbiór testowy, a pozostałe razem jako zbiór uczący i wykonuje analizę. Analiza jest więc wykonywana K razy. K rezultatów jest następnie uśrednianych (lub łączonych w inny sposób) w celu uzyskania jednego wyniku.
- walidację Leave-One-Out (*LOOCV*): Jest to odmiana walidacji K-krotnej, gdy N-elementowa próba jest dzielona na N podzbiorów, zawierających po jednym elemencie⁵



Resampling method

☒ None

☐ LOOCV

☐ K-fold CV

Fit model

Trening modelu

Po naciśnięciu przycisku *'Fit model'* aplikacja używa pakietu *'caret'* do wytrenowania modelu. Podsumowanie wyświetlone zostaje w panelu *'Fitted model summary'*. Można stąd odczytać m.in. oszacowane parametry modelu włącznie z błędami standardowymi czy statystyki F bądź R2.

⁵ https://pl.wikipedia.org/wiki/Sprawdzian_krzyżowy

Create formula

Choose outcome variable

mpg

Choose independent variables

cyl disp hp qsec gear

Add custom term to your model (eg. x1*x2 or x1:x2)

cyl:hp

Allocate observations to the train set

train set %

80

Resampling method

☐ None
☐ LOOCV
☒ K-fold CV

of k

4

Fitted model summary

Call:

lm(formula = .outcome ~ ., data = dat)

Residuals:

Min	1Q	Median	3Q	Max
-3.6093	-1.3947	-0.0601	0.2634	4.8498

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	43.30021	12.46229	3.474	0.00411 **
cyl8	-12.12760	9.50242	-1.276	0.22419
disp	-0.02509	0.01355	-1.851	0.08704 .
hp	-0.15976	0.05365	-2.978	0.01068 *
qsec	-0.12019	0.65221	-0.184	0.85663
gear4	0.69868	4.93478	0.142	0.88958
gear5	2.26348	3.61547	0.626	0.54212
`cyl8:hp`	0.13269	0.05955	2.228	0.04415 *

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.783 on 13 degrees of freedom

Multiple R-squared: 0.8953, Adjusted R-squared: 0.8389

F-statistic: 15.87 on 7 and 13 DF, p-value: 1.966e-05

Prediction metrics

Po prawej stronie ekranu widzimy panel z istotnością zmiennych (*variable importance*):

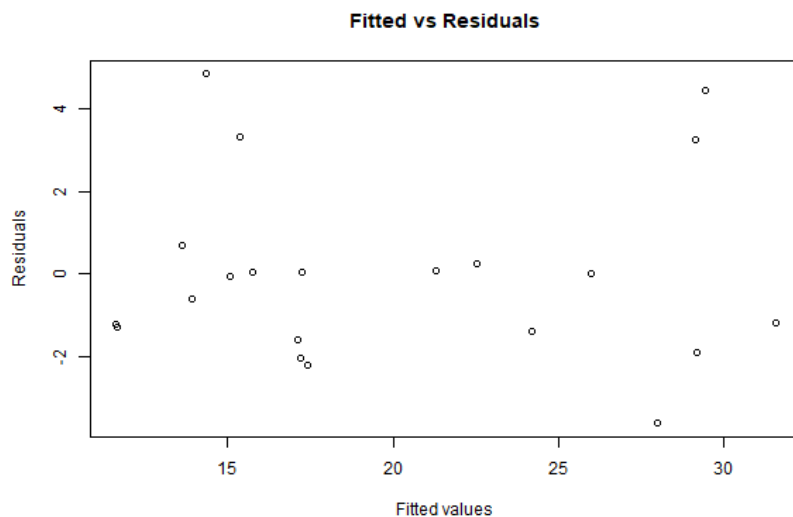
Variable importance

```
lm variable importance
```

	Overall
hp	100.000
`cyl8:hp`	73.568
disp	60.265
cyl8	40.006
gear5	17.081
qsec	1.506
gear4	0.000

Statystyka pokazuje, które zmienne najbardziej przyczyniają się do wyjaśnienia wariacji w modelu.

Aplikacja rysuje również wykres zależności między wartościami dopasowanymi (*fitted values*) a wartościami rezydualnymi (*residuals*)



Można go wykorzystać np. do zbadania heteroskedastyczności modelu.

Predykcja

Na obserwacjach należących do zbioru testowego dokonujemy predykcji z wykorzystaniem wytrenowanego modelu. Wyniki przedstawione są w formie tabeli z trzema kolumnami, które przedstawiają odpowiednio nazwę obserwacji, predykcję i wartość rzeczywistą. Dodatkowo aplikacja liczy trzy metryki:

- **RMSE:** pierwiastek średniego błędu kwadratowego, informuje o ile średnio prognozy różnią się od wartości rzeczywistych.
- **Rsquared:** współczynnik determinacji, miara licząca jak dobrze wartości przewidziane są dopasowane do wielkości rzeczywistych. Wielkość należy interpretować jako procent.
- **MAE:** średni błąd absolutny, informuje o ile średnio wynosi odchylenie od wartości rzeczywistej

Prediction metrics

RMSE	Rsquared	MAE
3.9778575	0.7038649	2.9918149

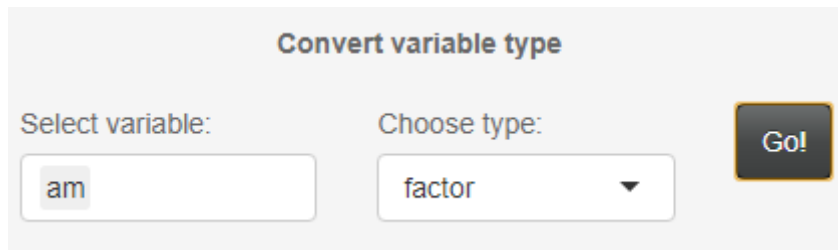
Names	Predictions	Actuals
Merc 450SE	17.29	16.40
Chrysler Imperial	11.81	14.70
Toyota Corona	22.39	21.50
Lotus Europa	23.09	30.40

6.2 Regresja logistyczna dwumianowa

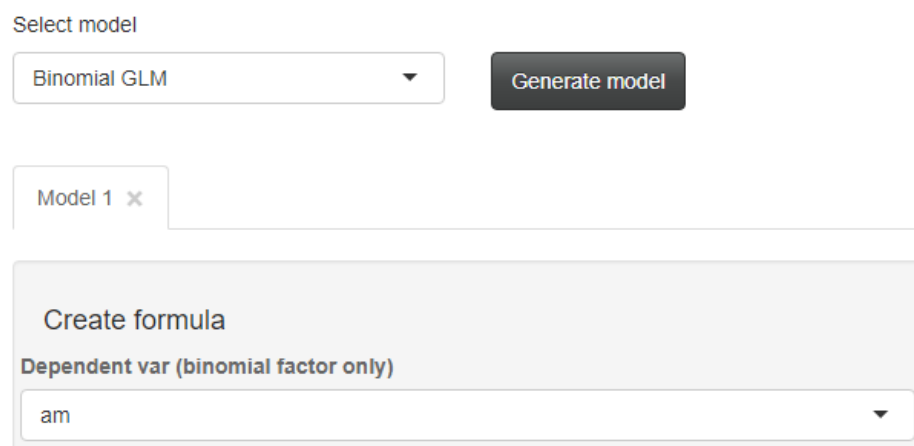
Wybór zmiennych

Regresję logistyczną dwumianową (*binomial logistic regression*) stosujemy do predykcji zmiennej jakościowej o dwóch klasach. Dlatego przed rozpoczęciem analizy musimy się upewnić, że nasza zmienna zależna ma klasę 'factor' i dokładnie dwa poziomy. Jeżeli nie, zmian możemy dokonać w zakładce 'Dataset'. Przykład poniżej dla zmiennej 'am' ze zbioru 'mtcars':

- najpierw zamieniamy zmienną dwuwartościową na jakościową



- następnie pojawia się ona w menu wyboru w zakładce 'ML'



Wyboru zmiennych objaśniających dokonujemy w menu niżej i dodajemy ewentualne interakcje między zmiennymi.

Potem decydujemy ile procent obserwacji z analizowanego zbioru danych ma trafić do zbioru uczącego (reszta trafi do zbioru testowego).

Walidacja krzyżowa działa na takiej samej zasadzie jak opisano w modelu regresji liniowej.

Trening modelu

Po naciśnięciu przycisku 'Fit model' aplikacja używa pakietu 'caret' do wytrenowania modelu. Podsumowanie wyświetlone zostaje w panelu 'Fitted model summary'. Można stąd odczytać m.in. oszacowane parametry modelu włącznie z błędami standardowymi czy inne statystyki.

Fitted model summary

```
Call:
NULL

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-7.597e-06 -4.560e-06 -1.136e-06  1.096e-07  6.276e-06

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.847e+02  6.017e+06      0      1
cyl          -2.811e+01  1.497e+05      0      1
disp          8.635e-02  1.657e+03      0      1
hp            2.593e-01  3.643e+03      0      1
drat          7.883e+00  3.884e+05      0      1
wt           -1.722e+01  2.283e+05      0      1
qsec         -8.686e-01  2.676e+05      0      1
vs           -4.356e+01  4.160e+05      0      1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 1.7995e+01  on 15  degrees of freedom
Residual deviance: 2.9452e-10  on  8  degrees of freedom
AIC: 16

Number of Fisher Scoring iterations: 24
```

Po prawej stronie ekranu widzimy panel z istotnością zmiennych (*variable importance*):

Variable Importance

```
glm variable importance

Overall
cyl 100.000
vs  54.987
wt  39.114
hp  36.812
disp 26.488
drat 9.241
qsec 0.000
```

Statystyka pokazuje, które zmienne najbardziej przyczyniają się do wyjaśnienia wariancji w modelu.

Predykcja

Na obserwacjach należących do zbioru testowego dokonujemy predykcji z wykorzystaniem wytrenowanego modelu. Wyniki przedstawione są w formie tabeli z trzema kolumnami, które przedstawiają odpowiednio nazwę obserwacji, predykcję i wartość rzeczywistą.

Names	Predictions	Actuals
Hornet Sportabout	0	0
Duster 360	1	0
Camaro Z28	1	0
Pontiac Firebird	0	0
Ferrari Dino	1	1

Dodatkowo aplikacja oblicza tzw. confusion matrix czyli tabelę z szeregiem metryk ewaluujących nasz model:

Confusion matrix

```

Confusion Matrix and Statistics

      Reference
Prediction 0 1
0 2 0
1 2 1

    Accuracy : 0.6
      95% CI : (0.1466, 0.9473)
  No Information Rate : 0.8
  P-Value [Acc > NIR] : 0.9421

      Kappa : 0.2857

  McNemar's Test P-Value : 0.4795

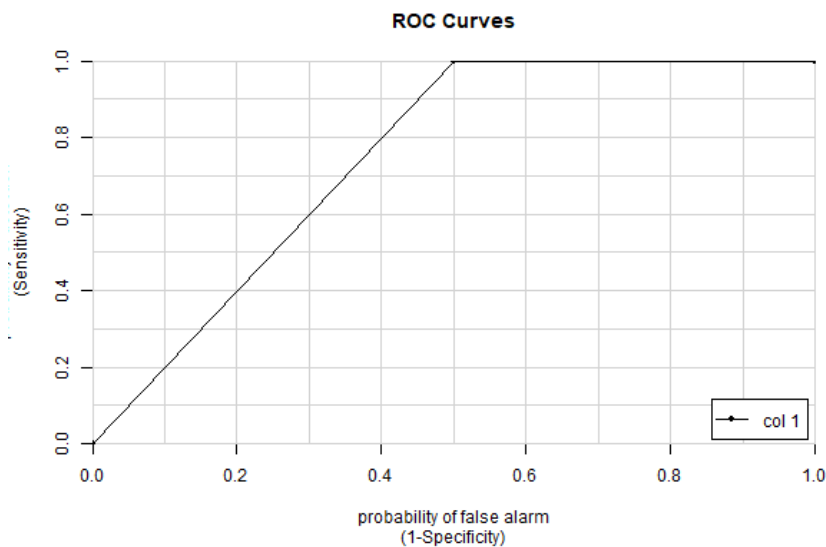
    Sensitivity : 0.5000
    Specificity : 1.0000
   Pos Pred Value : 1.0000
   Neg Pred Value : 0.3333
    Prevalence : 0.8000
    Detection Rate : 0.4000
  Detection Prevalence : 0.4000
   Balanced Accuracy : 0.7500

   'Positive' Class : 0

```

Po prawej stronie ekranu rysowana jest krzywa ROC, która jest graficzną reprezentacją efektywności modelu predykcyjnego poprzez wykreślenie charakterystyki jakościowej klasyfikatorów binarnych powstałych z modelu przy zastosowaniu wielu różnych punktów odcięcia⁶

⁶ <https://mathspace.pl/matematyka/receiver-operating-characteristic-krzywa-roc-czyli-ocena-jakosci-klasyfikacji-czesc-7/>



6.3 Liniowa Analiza Dyskryminacyjna

Wybór zmiennych

Liniową analizę dyskryminacyjną (*Linear Discriminant Analysis, LDA*) stosujemy podobnie jak regresję logistyczną w problemach klasyfikacyjnych, z tym że LDA lepiej się sprawdza przy zmiennych zależnych mających więcej niż dwie klasy.

Sposób działania zakładki jest podobny jak w poprzednim modelu z tą różnicą, że do Y możemy przypisać zmienną jakościową o dowolnej liczbie poziomów. Jeżeli w naszym zbiorze danych nie ma żadnych zmiennych typu *'factor'*, to wybór *'dependent variable'* nie będzie możliwy – w tym przypadku należy skorzystać z konwersji typów w zakładce *'Dataset'*.

Interakcje między zmiennymi i podział na próbę uczącą i testową dokonuje się tak samo, jak opisano wcześniej.

Trening modelu

Po naciśnięciu przycisku *'Fit model'* aplikacja używa pakietu *'caret'* do wytrenowania modelu. Podsumowanie wyświetlone zostaje w panelu *'Fitted model summary'*. Można stąd odczytać m.in. oszacowane parametry modelu włącznie z błędami standardowymi czy inne statystyki.

Fitted model summary

	Length	Class	Mode
prior	3	-none-	numeric
counts	3	-none-	numeric
means	33	-none-	numeric
scaling	22	-none-	numeric
lev	3	-none-	character
svd	2	-none-	numeric
N	1	-none-	numeric
call	3	-none-	call
xNames	11	-none-	character
problemType	1	-none-	character
tuneValue	1	data.frame	list
obsLevels	3	-none-	character
param	0	-none-	list

Linear Discriminant Analysis

21 samples
10 predictors
3 classes: '4', '6', '8'

No pre-processing
Resampling: Leave-One-Out Cross-Validation
Summary of sample sizes: 20, 20, 20, 20, 20, 20, ...
Resampling results:

Accuracy	Kappa
0.6666667	0.5

Po prawej stronie ekranu aplikacja oblicza tzw. confusion matrix czyli tabelę z szeregiem metryk ewaluujących nasz model:

Confusion matrix

Confusion Matrix and Statistics

```

      Reference
Prediction 4 6 8
      4 3 0 0
      6 1 2 0
      8 0 0 5

```

Overall Statistics

```

      Accuracy : 0.9091
      95% CI : (0.5872, 0.9977)
No Information Rate : 0.4545
P-Value [Acc > NIR] : 0.00243

```

```

      Kappa : 0.859

```

```

McNemar's Test P-Value : NA

```

Statistics by Class:

	Class: 4	Class: 6	Class: 8
Sensitivity	0.7500	1.0000	1.0000
Specificity	1.0000	0.8889	1.0000
Pos Pred Value	1.0000	0.6667	1.0000
Neg Pred Value	0.8750	1.0000	1.0000
Prevalence	0.3636	0.1818	0.4545
Detection Rate	0.2727	0.1818	0.4545
Detection Prevalence	0.2727	0.2727	0.4545
Balanced Accuracy	0.8750	0.9444	1.0000

Predykcja

Na obserwacjach należących do zbioru testowego dokonujemy predykcji z wykorzystaniem wytrenowanego modelu. Wyniki przedstawione są w formie tabeli z trzema kolumnami, które przedstawiają odpowiednio nazwę obserwacji, predykcję i wartość rzeczywistą. Niżej kalkulowana jest także dokładność predykcji.

Names	Predictions	Actuals
Hornet 4 Drive	6	6
Hornet Sportabout	8	8
Merc 240D	6	4
Merc 280C	6	6
Merc 450SE	8	8
Merc 450SLC	8	8
Lincoln Continental	8	8
Honda Civic	4	4
Toyota Corona	4	4
Dodge Challenger	8	8
Volvo 142E	4	4

Prediction accuracy

```
[1] 90.91
```

6.4 Drzewo decyzyjne

Wybór zmiennych

Drzewo decyzyjne (*decision tree*) stosujemy w problemach zarówno regresyjnych jak i klasyfikacyjnych.

Sposób działania zakładki jest podobny jak w pozostałych modelach – wybór zmiennych, interakcje oraz podział na próbę uczącą i testową dokonuje się tak samo, jak opisano wcześniej.

Dodatkowymi parametrami są opcje '*Max tree depth*' i '*cp*'.

Pierwszy pozwala ustawić maksymalną głębokość drzewa, drugi (*complexity parameter*) oznacza ile musi wynosić minimalna poprawa w funkcji kosztowej modelu w każdej gałęzi.

Trening modelu

Po naciśnięciu przycisku '*Fit model*' aplikacja używa pakietu '*rpart*' do wytrenowania modelu. Podsumowanie wyświetlone zostaje w panelu '*Fitted model summary*'. Można stąd odczytać m.in. oszacowane parametry modelu włącznie z błędami standardowymi czy inne statystyki.

Fitted model summary

```
Call:
rpart(formula = formula, data = train_set, method = "class",
      model = TRUE, control = rpart.control(maxdepth = ifelse(input$TI_maxDepth ==
      "", 10, as.numeric(input$TI_maxDepth)), cp = ifelse(input$TI_cp ==
      "", 0.001, as.numeric(input$TI_cp)), xval = xval))
n= 27

      CP nsplit rel error      xerror      xstd
1 0.600      0      1.0 1.0000000 0.1721326
2 0.001      1      0.4 0.4666667 0.1518067

Variable importance
disp  hp  mpg  wt drat  vs
  20  18  18  17  13  13

Node number 1: 27 observations,      complexity param=0.6
predicted class=8 expected loss=0.5555556 P(node) =1
class counts:      9      6      12
probabilities: 0.333 0.222 0.444
left son=2 (15 obs) right son=3 (12 obs)
Primary splits:
  disp < 266.9 to the left, improve=10.133330, (0 missing)
   hp  < 136.5 to the left, improve= 9.058608, (0 missing)
  mpg < 17.55 to the right, improve= 8.708333, (0 missing)
   wt  < 3.49  to the left, improve= 7.450980, (0 missing)
   vs  < 0.5   to the right, improve= 6.594697, (0 missing)
Surrogate splits:
  mpg < 17.55 to the right, agree=0.963, adj=0.917, (0 split)
   hp  < 136.5 to the left, agree=0.963, adj=0.917, (0 split)
   wt  < 3.49  to the left, agree=0.926, adj=0.833, (0 split)
 drat < 3.58  to the right, agree=0.852, adj=0.667, (0 split)
   vs  < 0.5   to the right, agree=0.852, adj=0.667, (0 split)

Node number 2: 15 observations
predicted class=4 expected loss=0.4 P(node) =0.5555556
class counts:      9      6      0
probabilities: 0.600 0.400 0.000

Node number 3: 12 observations
predicted class=8 expected loss=0 P(node) =0.4444444
class counts:      0      0      12
probabilities: 0.000 0.000 1.000
```

Gdy wybierzemy walidację krzyżową aplikacja generuje po prawej stronie ekranu tabelę z wartościami błędów kroswalidacyjnych. Pozwalają one na otrzymanie drzew optymalnych w zależności od wartości parametru *cp*. Funkcja zwraca wartość *xerror*, która jest ilorazem SSE_{cv} dla danego drzewa i SSE dla korzenia. Zwraca też błąd standardowy *std* (*xstd*). W celu optymalnego doboru drzewa można

następnie wybrać np. drzewo z najmniejszą wartością xerror (xerrormin). Oprocz tabeli generowana jest również reprezentacja graficzna⁷:

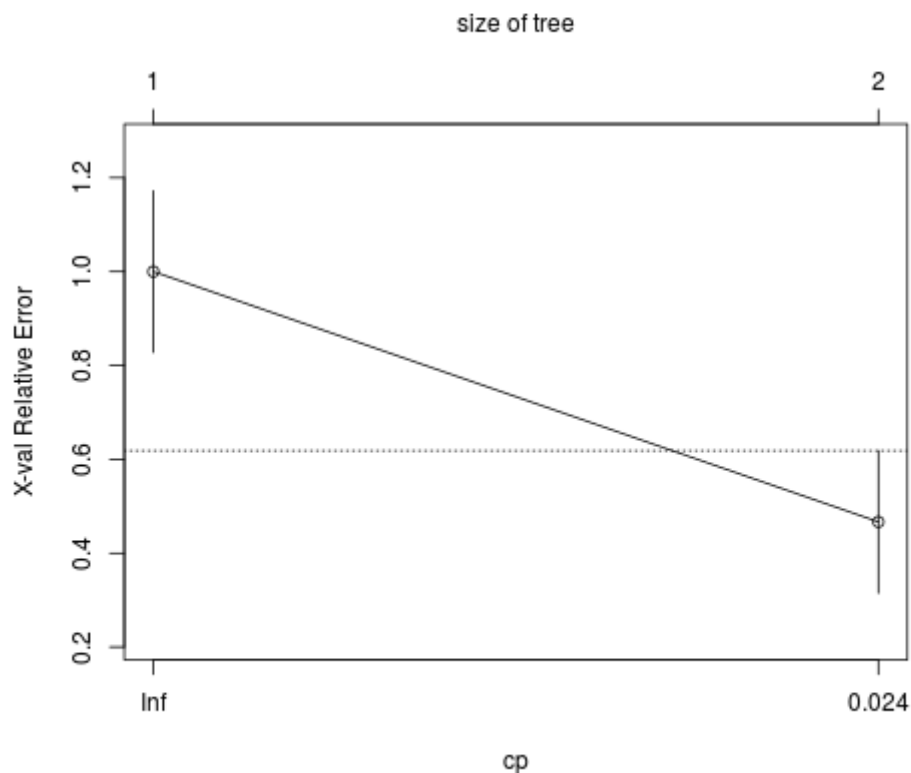
Complexity parameter

```
Classification tree:
Variables actually used in tree construction:
[1] disp
```

```
Root node error: 15/27 = 0.55556
```

```
n= 27
```

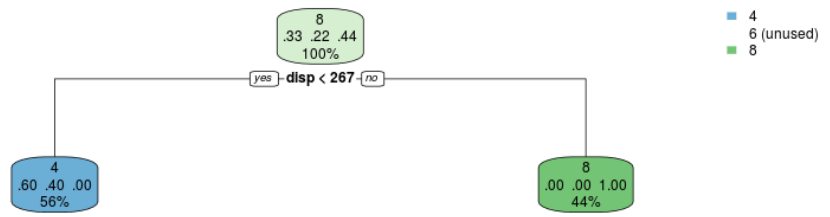
	CP	nsplit	rel error	xerror	xstd
1	0.600	0	1.0	1.00000	0.17213
2	0.001	1	0.4	0.46667	0.15181



Predykcja

Na obserwacjach należących do zbioru testowego dokonujemy predykcji z wykorzystaniem wytrenowanego modelu. Wyniki przedstawione są w formie tabeli z trzema kolumnami, które przedstawiają odpowiednio nazwę obserwacji, predykcję i wartość rzeczywistą. Shiny generuje także graficzną reprezentację drzewa losowego:

⁷ http://zsi.tech.us.edu.pl/~nowak/odzw/smad_lab10a.pdf



Names	Predictions	Actuals
Hornet Sportabout	8	8
Merc 280	4	6
Lincoln Continental	8	8
Honda Civic	4	4
Fiat X1-9	4	4