Projekt CV

Kacper Ludwiczak

2023-03-14

# Regression Model Projekt - Kacper Ludwiczak, 221303

## Temat - Kształtowanie się oczekiwanej długości życia dla większości państw świata na podstawie modelu regresji i czynniki wpływające na jego wysokość.

Zaczynając projekt pracowałem na danych związanych z nowoczesnymi technologiami. Dokonałem czyszczenia danych, obliczeń w excelu oraz obliczeń w języku R. Projekty znajdują się pod nazwą “Stary projekt - Reggresion Models”, “Stary projekt – Zestawienie”, “Stary projekt w R”. Jednakże stwierdziłem, że wyniki nie są zadowalające i porzuciłem ten projekt.

Następnie zacząłem prace nad tym modelem. Dane pozyskałem ze strony kaggle.com. Według strony dane pochodzą ze strony internetowej WHO i ONZ z pomocą Deeksha Russella i Duana Wanga. Dane dotyczą większości państw świata, ich różnych zbiorów informacji, np. Długość życia. Plik z oryginałem danych jest pod nazwą “Life Expectancy Data”.

[Link do strony z oryginalnymi danymi](https://www.kaggle.com/datasets/kumarajarshi/life-expectancy-who?resource=download)

Po pierwsze chciałem skupić się na próbie 30 krajów. Wybrałem zatem 30 krajów, metodą losową. Dane obliczyłem zarówno w excelu jak i w języku R. Dane i obliczenia znajdują się w pliku “Projekt w R próba” oraz “Projekt w Excel próba”.

Uznałem, że lepszym pomysłem będzie stworzenie modelu z większości krajów na jakich dano było mi pracować. Zacząłem od pozostawienia tylko krajów z 2015 jako najbardziej aktualnych z danej bazy. Dane są pod nazwą “Dane z 2015”. Następnie zamieniłem dane za pomocą funkcji “Tekst jako kolumny”. Usunąłem kolumny “Year”, “Status”, ponieważ kolumny są mi niepotrzebne. Usunąłem kolumny “Alcohol”, “percentage expenditure”, “Total expenditure”, ponieważ były zauważające braki danych. Usunąłem kolumny “BMI”, “HIV/AIDS”, “thinness 1-19 years”, “thinness 5-9 years”, “Schooling”, ponieważ pojawiła mi się pewna anomalia. Podczas zamiany na kolumny niektóre dane są formacie daty, po zamienieniu ich na liczby pojawiały się błędne liczby. Również nie które liczby zamienione automatycznie są błędne. Po wielu nieudanych próbach rozwiązania tego problemu, dokonałem eliminacji tych kolumn. Również musiałem dokonać zamiany kropek na przecinki w liczbach oraz wypełnić puste pola średnią z reszty kolumny. Dane znajdują się w pliku “Dane zrobione”. Również stworzyłem osobny plik dla importowania do RStudio, pod nazwą “Dane do R”.

### Szczegóły dotyczące danych:

* Country - Kraj
* Year - Rok
* Status - Stan rozwinięty lub rozwijający się
* Life expectancy - Oczekiwana długość życia w latach
* Adult Mortality - Wskaźniki śmiertelności dorosłych obu płci (prawdopodobieństwo śmierci w wieku od 15 do 60 lat na 1000 mieszkańców)
* Infant deaths - Liczba zgonów niemowląt na 1000 ludności
* Alcohol - Spożycie alkoholu na mieszkańca (15+) (w litrach czystego alkoholu)
* Hepatitis B - Zasięg szczepień przeciw wirusowemu zapaleniu wątroby typu B (HepB) wśród 1-latków (%)
* Measles – Odra, liczba zgłoszonych przypadków na 1000 ludności
* BMI - Średni wskaźnik masy ciała całej populacji
* Under-five deaths - Liczba zgonów poniżej piątego roku życia na 1000 mieszkańców
* Polio - Zasięg szczepień przeciw polio (Pol3) wśród 1-latków (%)
* Total expenditure - Wydatki sektora instytucji rządowych i samorządowych na zdrowie jako odsetek wydatków sektora instytucji rządowych i samorządowych ogółem (%)
* Diphtheria - Odsetek szczepień przeciwko anatoksynie błonicy i tężcowi oraz krztuścowi (DTP3) wśród 1-latków (%)
* HIV/AIDS - Zgony na 1000 żywych urodzeń HIV/AIDS (0-4 lata)
* GDP - Produkt Krajowy Brutto per capita (w USD)
* Population - Ludność kraju
* Thinness 1-19 years - Rozpowszechnienie szczupłości wśród dzieci i młodzieży w wieku od 10 do 19 lat (%)
* Thinness 5-9 years - Występowanie szczupłości wśród dzieci w wieku od 5 do 9 lat (%)
* Income - Wskaźnik rozwoju społecznego pod względem struktury dochodów zasobów (wskaźnik w zakresie od 0 do 1)
* Schooling- Liczba lat nauki (lata)

### Projekt w RStudio jest pod nazwą “Projekt R markdown”

Zainstalowałem pakiet “readxl”, oraz uruchomiłem go przez funkcje “library”.

library(readxl)

## Warning: pakiet 'readxl' został zbudowany w wersji R 4.2.2

Zainstalowałem pakiet “rmarkdown”, oraz uruchomiłem go przez funkcje “library”.

library(rmarkdown)

## Warning: pakiet 'rmarkdown' został zbudowany w wersji R 4.2.2

Kod wczytuje plik excel “Dane do R.xlsx” i zapisuje jego zawartość w zmiennej “Dane”. Następnie wyświetla zawartość tej zmiennej za pomocą funkcji “View()”, wyświetla nazwy kolumn danych za pomocą funkcji “names()”, a na końcu wyświetla wartości kolumny “Life expectancy” za pomocą funkcji “print()”.

Dane <- read\_excel("C:/Users/Kacper/Desktop/Projekt Regression/Dane do R.xlsx")  
View(Dane)  
names(Dane)

## [1] "Life expectancy" "Adult Mortality" "Infant deaths" "Hepatitis B"   
## [5] "Measles" "Under-5 deaths" "Polio" "Diphtheria"   
## [9] "GDP" "Population" "Income"

print(Dane$"Life expectancy" )

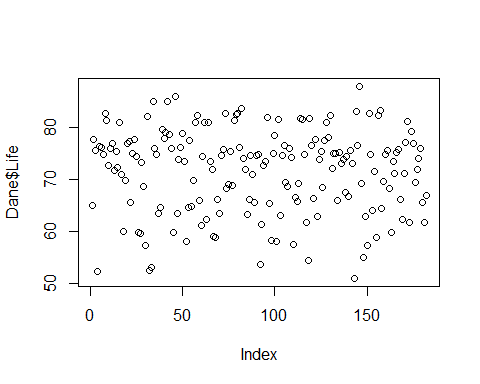
## [1] 65.0 77.8 75.6 52.4 76.4 76.3 74.8 82.8 81.5 72.7 76.1 76.9 71.8 75.5 72.3  
## [16] 81.1 71.0 60.0 69.8 77.0 77.4 65.7 75.0 77.7 74.5 59.9 59.6 73.3 68.7 57.3  
## [31] 82.2 52.5 53.1 85.0 76.1 74.8 63.5 64.7 79.6 78.0 79.1 85.0 78.8 76.0 59.8  
## [46] 86.0 63.5 73.9 76.2 79.0 73.5 58.2 64.7 77.6 64.8 69.9 81.1 82.4 66.0 61.1  
## [61] 74.4 81.0 62.4 81.0 73.6 71.9 59.0 58.9 66.2 63.5 74.6 75.8 82.7 68.3 69.1  
## [76] 75.5 68.9 81.4 82.5 82.7 76.2 83.7 74.1 72.0 63.4 66.3 74.7 71.1 65.7 74.6  
## [91] 74.9 53.7 61.4 72.7 73.6 82.0 65.5 58.3 75.0 78.5 58.2 81.7 63.1 74.6 76.7  
## [106] 69.4 68.8 76.1 74.3 57.6 66.6 65.8 69.2 81.9 81.6 74.8 61.8 54.5 81.8 76.6  
## [121] 66.4 77.8 62.9 74.0 75.5 68.5 77.5 81.1 78.2 82.3 72.1 75.0 75.0 66.1 75.2  
## [136] 73.2 74.0 67.5 74.5 66.7 75.6 73.2 51.0 83.1 76.7 88.0 69.2 55.0 62.9 57.3  
## [151] 82.8 74.9 64.1 71.6 58.9 82.4 83.4 64.5 69.7 74.9 75.7 68.3 59.9 73.5 71.2  
## [166] 75.3 75.8 66.3 62.3 71.3 77.1 81.2 61.8 79.3 77.0 69.4 72.0 74.1 76.0 65.7  
## [181] 61.8 67.0

Tutaj za pomocą funkcji “colnames” dokonałem zmiany nazw kolumn w celu szybszej pracy w dalszych etapach.

colnames(Dane)<-c("Life","Adult\_M","Infant\_D","H\_B","Measles","Under\_D","Polio","Dipht","GDP","Popl","Income")

Użyłem funkcji “plot” w celu zobaczeniu na wykresie dane z kolumny “Life”. Kolumna “Life” zostaje moją zmienną zależną, natomiast reszta kolumn zmiennymi niezależnymi.

plot(Dane$"Life")



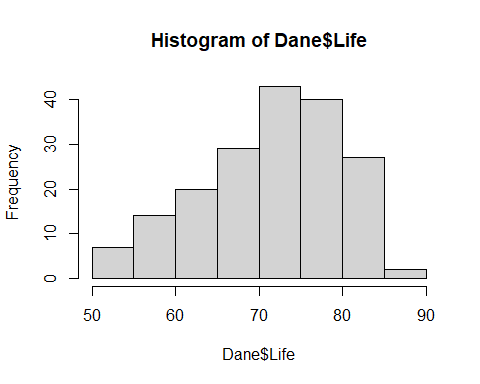
Sprawdziłem następujące wartości danych: \* Minimalna wartość = 51.00 \* Pierwszy kwartal = 65.85 \* Mediana = 73.95 \* Średnia = 71.72 \* Trzeci kwartal = 76.97 \* Maksymalna wartość = 88.00 , dla zmiennej zależnej.

summary(Dane$"Life" )

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 51.00 65.85 73.95 71.72 76.97 88.00

W celu zobrazowania graficznego użyłem funkcji “hist”. Według tego można zauważyć dominacje wartości w przedziale od 70 do 80.

hist(Dane$"Life" )



Za pomocą funkcji “cor” przedstawiłem korelacje wszystkich zmiennych. Im dana liczba jest większa tym większa jest korelacja między odpowiadającymi danymi w wierszu i kolumnie. Można z niej ustalić, że korelacja między zmienną zależną “Life” a zmienną: \* “Adult\_M” jest duża ujemnie \* “Infant\_D” jest bardzo mała ujemnie \* “H\_B” jest mała dodatnio \* “Measles” jest bardzo mała ujemnie \* “Under\_D” jest bardzo mała ujemnie \* “Polio” jest średnia dodatnio \* “Dipht” jest średnia dodatnio \* “GDP” jest mała dodatnio \* “Popl” jest bardzo mała ujemnie \* “Income” jest bardzo mała ujemnie

cor(Dane)

## Life Adult\_M Infant\_D H\_B Measles  
## Life 1.00000000 -0.77215206 -0.23984305 0.40450050 -0.077810464  
## Adult\_M -0.77215206 1.00000000 0.18604690 -0.23139939 0.054570792  
## Infant\_D -0.23984305 0.18604690 1.00000000 -0.08681527 0.801663443  
## H\_B 0.40450050 -0.23139939 -0.08681527 1.00000000 0.015656414  
## Measles -0.07781046 0.05457079 0.80166344 0.01565641 1.000000000  
## Under\_D -0.27495588 0.21446481 0.99348313 -0.11001898 0.764990118  
## Polio 0.52186030 -0.37773277 -0.12958871 0.59020367 -0.028359472  
## Dipht 0.50655358 -0.32738699 -0.11788072 0.90925537 -0.001622602  
## GDP 0.43089742 -0.31610931 -0.11953200 0.13188477 -0.076143290  
## Popl -0.04721323 0.03756295 0.26624483 -0.05672244 0.127925269  
## Income -0.13196089 0.13031897 0.02150182 -0.08670694 0.008493532  
## Under\_D Polio Dipht GDP Popl  
## Life -0.27495588 0.52186030 0.506553583 0.43089742 -0.047213234  
## Adult\_M 0.21446481 -0.37773277 -0.327386992 -0.31610931 0.037562946  
## Infant\_D 0.99348313 -0.12958871 -0.117880715 -0.11953200 0.266244826  
## H\_B -0.11001898 0.59020367 0.909255374 0.13188477 -0.056722444  
## Measles 0.76499012 -0.02835947 -0.001622602 -0.07614329 0.127925269  
## Under\_D 1.00000000 -0.15154462 -0.143729736 -0.12620291 0.303720175  
## Polio -0.15154462 1.00000000 0.661623315 0.22349572 -0.221071926  
## Dipht -0.14372974 0.66162332 1.000000000 0.21333277 -0.065038775  
## GDP -0.12620291 0.22349572 0.213332775 1.00000000 0.046037030  
## Popl 0.30372017 -0.22107193 -0.065038775 0.04603703 1.000000000  
## Income 0.05419071 -0.06262976 -0.100745256 -0.04756283 -0.007033844  
## Income  
## Life -0.131960889  
## Adult\_M 0.130318973  
## Infant\_D 0.021501816  
## H\_B -0.086706940  
## Measles 0.008493532  
## Under\_D 0.054190707  
## Polio -0.062629761  
## Dipht -0.100745256  
## GDP -0.047562832  
## Popl -0.007033844  
## Income 1.000000000

Kod ten oblicza korelację między kolumnami w obiekcie “Dane”, z wyjątkiem pierwszej kolumny, która jest pomijana za pomocą wyrażenia “[, -1]”. Jest to kolumna “Life”. Funkcja “cor” oblicza korelację Pearsona, czyli stopień zależności liniowej między dwoma zmiennymi. Wynik jest następnie zaokrąglany do trzech miejsc po przecinku za pomocą funkcji “round”.

round(cor(Dane[,-1]),3)

## Adult\_M Infant\_D H\_B Measles Under\_D Polio Dipht GDP Popl  
## Adult\_M 1.000 0.186 -0.231 0.055 0.214 -0.378 -0.327 -0.316 0.038  
## Infant\_D 0.186 1.000 -0.087 0.802 0.993 -0.130 -0.118 -0.120 0.266  
## H\_B -0.231 -0.087 1.000 0.016 -0.110 0.590 0.909 0.132 -0.057  
## Measles 0.055 0.802 0.016 1.000 0.765 -0.028 -0.002 -0.076 0.128  
## Under\_D 0.214 0.993 -0.110 0.765 1.000 -0.152 -0.144 -0.126 0.304  
## Polio -0.378 -0.130 0.590 -0.028 -0.152 1.000 0.662 0.223 -0.221  
## Dipht -0.327 -0.118 0.909 -0.002 -0.144 0.662 1.000 0.213 -0.065  
## GDP -0.316 -0.120 0.132 -0.076 -0.126 0.223 0.213 1.000 0.046  
## Popl 0.038 0.266 -0.057 0.128 0.304 -0.221 -0.065 0.046 1.000  
## Income 0.130 0.022 -0.087 0.008 0.054 -0.063 -0.101 -0.048 -0.007  
## Income  
## Adult\_M 0.130  
## Infant\_D 0.022  
## H\_B -0.087  
## Measles 0.008  
## Under\_D 0.054  
## Polio -0.063  
## Dipht -0.101  
## GDP -0.048  
## Popl -0.007  
## Income 1.000

Kod ten tworzy macierz z danych zawartych w obiekcie “Dane”. Wiersze z obiektu “Dane” są wczytywane za pomocą wyrażenia “[, -1]”, co oznacza, że pierwsza kolumna jest pomijana. Jest to kolumna “Life”. Następnie dane są konwertowane na macierz za pomocą funkcji “as.matrix”. Ostatecznie macierz jest przypisywana do obiektu “matrix”.

matrix <- as.matrix(Dane[,-1])

Kod oblicza własne wartości i wektory dla macierzy “matrix”. Macierz jest najpierw transponowana za pomocą funkcji “t”, a następnie mnożona przez siebie za pomocą operatora “%\*%“, co daje macierz kowariancji. Otrzymaną macierz kowariancji jest analizowana przez funkcję”eigen”, która oblicza własne wartości i wektory. Wynik jest przypisywany do obiektu “matrix\_eigen”. Dzięki temu mogę zobaczyć własne wartości macierzy

matrix\_eigen <- eigen(t(matrix) %\*% matrix)  
matrix\_eigen$val

## [1] 1.365373e+17 2.887223e+10 1.122436e+10 6.501808e+06 1.227256e+06  
## [6] 7.446202e+05 4.555312e+04 1.774313e+04 7.863299e+03 5.388979e+03

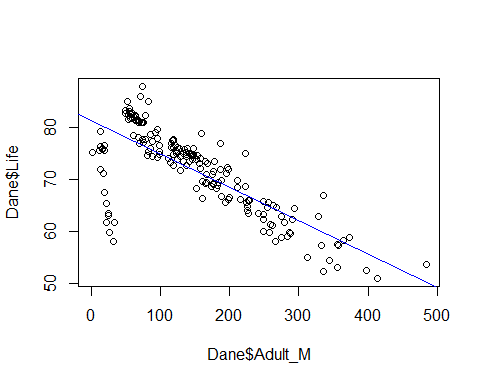
Kod wylicza pierwiastek kwadratowy elementów ilorazu dwóch wektorów za pomocą “sqrt”, “matrix\_eigenval[1]” jest pierwszym elementem wektora “matrix\_eigenval” jest całym wektorem. Dostając następujące wyniki. Wyniki powyżej 30 wskazują na brak współliniowości. W moim przypadku można zauważyć, że współliniowość nie występuje. Na tym etapie można wykluczyć jako by mój model był dobrym modelem.

sqrt(matrix\_eigen$val[1]/matrix\_eigen$val)

## [1] 1.000 2174.631 3487.746 144913.411 333547.800 428211.415  
## [7] 1731277.043 2774026.198 4166997.596 5033527.140

Sprawdziłem na wykresie jaka jest korelacje zmiennej zależnej i zmiennej “Adult\_M” w postaci graficznej. Tutaj przetestowałem funkcje “lm” za pomocą stworzenia modelu prostej regresji liniowej z użyciem jednej zmiennej niezależnej. Funkcja “abline” umożliwiła mi dodanie niebieskiej linii regresji, która najdokładniej próbuje dopasować się do istniejących danych. Można zauważyć, że linia jest przekrzywiona w dół po stronie lewej, wynika to z pojawiających się wartości odstających, poniżej wartości 100 dla zmiennej niezależnej.

plot(Dane$"Life"~Dane$"Adult\_M")  
SimpleModel<- lm(Dane$"Life"~Dane$"Adult\_M")  
abline(SimpleModel$coef,col="blue")



Wynik p-value wyszedł bardzo dobry. Jednakże wartość R-squared zbyt mała.

summary(SimpleModel)

##   
## Call:  
## lm(formula = Dane$Life ~ Dane$Adult\_M)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -21.2282 -1.6823 0.7664 2.9876 11.2813   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 81.492535 0.709494 114.9 <2e-16 \*\*\*  
## Dane$Adult\_M -0.064512 0.003957 -16.3 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 5.117 on 180 degrees of freedom  
## Multiple R-squared: 0.5962, Adjusted R-squared: 0.594   
## F-statistic: 265.8 on 1 and 180 DF, p-value: < 2.2e-16

Stworzenie pierwszego modelu. Używam metody “Backward” ręcznie. Eliminując zmienne o najwyższym wskaźniku p-value, tak długo aż model będzie zadawalający. Zakładam dla modelu, że poziom p-value poniżej 0.05 jest odpowiedni.

Podsumowaniem modelu regresji liniowej wygląda następująco. Zawiera on informacje na temat tego, jak dobrze modelem można wyjaśnić zmienną zależną (Life), korzystając z 7 zmiennych objaśniających (Adult\_M, Infant\_D, H\_B, Measles, Under\_D, Polio, Dipht, GDP, Popl, Income). Sekcja “Residuals” pokazuje rozkład reszt modelu regresji liniowej, czyli różnic między wartościami faktycznymi a wartościami przewidywanymi. “Min”, “1Q”, “Median”, “3Q”, i “Max” to odpowiednio najmniejsza, pierwszy kwartyl, medianę, trzeci kwartyl, i największą wartość reszt. Sekcja “Coefficients” pokazuje wartości współczynników regresji dla każdej zmiennej objaśniającej, wraz z błędem standardowym i wartością statystyki t. Wartość p dla każdej zmiennej określa, czy jest ona istotna statystycznie (p < 0,05 oznacza, że zmienna jest istotna). Wartość “Residual standard error” to średni kwadrat błędu reszty. “Multiple R-squared” to współczynnik determinacji, który określa, jak dobrze modelem można wyjaśnić zmienną zależną. Wartość R-kwadratu zawsze znajduje się w zakresie od 0 do 1, a wartość bliska 1 oznacza dobre dopasowanie modelu. “Adjusted R-squared” to współczynnik determinacji uwzględniający liczbę zmiennych objaśniających. Wartość “F-statistic” i “p-value” służą do testowania hipotezy zerowej, że wszystkie współczynniki regresji są równe 0.

Model <- lm(Life~Adult\_M+Infant\_D+H\_B+Measles+Under\_D+Polio+Dipht+GDP+Popl+Income,data=Dane)  
summary(Model)

##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + H\_B + Measles + Under\_D +   
## Polio + Dipht + GDP + Popl + Income, data = Dane)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -16.3833 -2.4605 0.2957 2.3275 11.8063   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.931e+01 1.761e+00 39.356 < 2e-16 \*\*\*  
## Adult\_M -4.718e-02 3.959e-03 -11.919 < 2e-16 \*\*\*  
## Infant\_D 9.576e-02 4.644e-02 2.062 0.040728 \*   
## H\_B -9.270e-03 3.379e-02 -0.274 0.784131   
## Measles 1.253e-05 7.851e-05 0.160 0.873375   
## Under\_D -8.295e-02 3.440e-02 -2.412 0.016940 \*   
## Polio 5.024e-02 1.887e-02 2.662 0.008500 \*\*   
## Dipht 6.339e-02 3.886e-02 1.631 0.104718   
## GDP 1.202e-04 3.239e-05 3.712 0.000278 \*\*\*  
## Popl 2.531e-08 1.461e-08 1.732 0.085120 .   
## Income 1.122e-02 3.531e-02 0.318 0.751148   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 4.346 on 171 degrees of freedom  
## Multiple R-squared: 0.7233, Adjusted R-squared: 0.7071   
## F-statistic: 44.69 on 10 and 171 DF, p-value: < 2.2e-16

W pierwszym modelu największy poziom p-value miała kolumna “Measles”. W drugim modelu już się nie pojawiła.

Model2 <- lm(Life~Adult\_M+Infant\_D+H\_B+Under\_D+Polio+Dipht+GDP+Popl+Income,data=Dane)  
summary(Model2)

##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + H\_B + Under\_D + Polio +   
## Dipht + GDP + Popl + Income, data = Dane)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -16.3917 -2.4905 0.3158 2.3285 11.8053   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.930e+01 1.755e+00 39.481 < 2e-16 \*\*\*  
## Adult\_M -4.721e-02 3.944e-03 -11.968 < 2e-16 \*\*\*  
## Infant\_D 9.971e-02 3.918e-02 2.545 0.011806 \*   
## H\_B -9.209e-03 3.369e-02 -0.273 0.784901   
## Under\_D -8.530e-02 3.099e-02 -2.752 0.006554 \*\*   
## Polio 5.024e-02 1.882e-02 2.670 0.008305 \*\*   
## Dipht 6.345e-02 3.875e-02 1.637 0.103391   
## GDP 1.201e-04 3.229e-05 3.720 0.000269 \*\*\*  
## Popl 2.538e-08 1.457e-08 1.743 0.083161 .   
## Income 1.204e-02 3.483e-02 0.346 0.730039   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 4.334 on 172 degrees of freedom  
## Multiple R-squared: 0.7232, Adjusted R-squared: 0.7087   
## F-statistic: 49.94 on 9 and 172 DF, p-value: < 2.2e-16

W drugim modelu największy poziom p-value miała kolumna “H\_B”. W trzecim modelu już się nie pojawiła

Model3 <- lm(Life~Adult\_M+Infant\_D+Under\_D+Polio+Dipht+GDP+Popl+Income,data=Dane)  
summary(Model3)

##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + Under\_D + Polio + Dipht +   
## GDP + Popl + Income, data = Dane)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -16.4264 -2.4996 0.3435 2.3178 11.8929   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.933e+01 1.747e+00 39.684 < 2e-16 \*\*\*  
## Adult\_M -4.735e-02 3.899e-03 -12.143 < 2e-16 \*\*\*  
## Infant\_D 9.943e-02 3.906e-02 2.546 0.011780 \*   
## Under\_D -8.510e-02 3.090e-02 -2.754 0.006521 \*\*   
## Polio 5.018e-02 1.876e-02 2.674 0.008208 \*\*   
## Dipht 5.437e-02 1.988e-02 2.735 0.006881 \*\*   
## GDP 1.211e-04 3.200e-05 3.785 0.000211 \*\*\*  
## Popl 2.531e-08 1.452e-08 1.743 0.083134 .   
## Income 1.203e-02 3.474e-02 0.346 0.729621   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 4.322 on 173 degrees of freedom  
## Multiple R-squared: 0.7231, Adjusted R-squared: 0.7103   
## F-statistic: 56.47 on 8 and 173 DF, p-value: < 2.2e-16

W trzecim modelu największy poziom p-value miała kolumna “Income”. W czwartym modelu już się nie pojawiła.

Model4 <- lm(Life~Adult\_M+Infant\_D+Under\_D+Polio+Dipht+GDP+Popl,data=Dane)  
summary(Model4)

##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + Under\_D + Polio + Dipht +   
## GDP + Popl, data = Dane)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -16.4750 -2.5123 0.3676 2.3051 11.8873   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.935e+01 1.742e+00 39.819 < 2e-16 \*\*\*  
## Adult\_M -4.730e-02 3.887e-03 -12.169 < 2e-16 \*\*\*  
## Infant\_D 9.561e-02 3.737e-02 2.558 0.011365 \*   
## Under\_D -8.206e-02 2.955e-02 -2.777 0.006095 \*\*   
## Polio 5.021e-02 1.872e-02 2.683 0.008003 \*\*   
## Dipht 5.425e-02 1.982e-02 2.737 0.006843 \*\*   
## GDP 1.211e-04 3.192e-05 3.795 0.000203 \*\*\*  
## Popl 2.472e-08 1.439e-08 1.718 0.087513 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 4.312 on 174 degrees of freedom  
## Multiple R-squared: 0.7229, Adjusted R-squared: 0.7118   
## F-statistic: 64.85 on 7 and 174 DF, p-value: < 2.2e-16

W czwartym modelu największy poziom p-value miała kolumna “Popl”. W piątym modelu już się nie pojawiła.

Model5 <- lm(Life~Adult\_M+Infant\_D+Under\_D+Polio+Dipht+GDP,data=Dane)  
summary(Model5)

##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + Under\_D + Polio + Dipht +   
## GDP, data = Dane)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -16.951 -2.437 0.477 2.551 11.735   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.980e+01 1.731e+00 40.319 < 2e-16 \*\*\*  
## Adult\_M -4.821e-02 3.872e-03 -12.451 < 2e-16 \*\*\*  
## Infant\_D 7.326e-02 3.523e-02 2.080 0.03901 \*   
## Under\_D -6.291e-02 2.753e-02 -2.286 0.02347 \*   
## Polio 4.199e-02 1.819e-02 2.308 0.02218 \*   
## Dipht 5.995e-02 1.965e-02 3.051 0.00264 \*\*   
## GDP 1.264e-04 3.194e-05 3.957 0.00011 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 4.336 on 175 degrees of freedom  
## Multiple R-squared: 0.7182, Adjusted R-squared: 0.7086   
## F-statistic: 74.34 on 6 and 175 DF, p-value: < 2.2e-16

W piątym modelu po usunięciu kolumny “Popl”, nastąpił duży wzrost wartości p-value dla kolumny “Under\_D” oraz “Polio”. Jest efekt niepożądany dla mojego modelu, dlatego wybranym modelem zostaje model czwarty. Pomimo wartości p-value dla kolumny “Popl” w okolicy 0.08. Również poziom R-squared wynosi 0.7229, możemy interpretować to jako dobry model, choć na granicy złego (zmienność oczekiwanej długości życia jest wyjaśniony w 72% przez model). Dodatkowo Adjusted Rsquared jest na poziomie 0.7118, jest to najlepszy wynik ze wszystkich pięciu policzonych.

Model <- lm(Life~Adult\_M+Infant\_D+Under\_D+Polio+Dipht+GDP+Popl,data=Dane)  
summary(Model)

##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + Under\_D + Polio + Dipht +   
## GDP + Popl, data = Dane)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -16.4750 -2.5123 0.3676 2.3051 11.8873   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.935e+01 1.742e+00 39.819 < 2e-16 \*\*\*  
## Adult\_M -4.730e-02 3.887e-03 -12.169 < 2e-16 \*\*\*  
## Infant\_D 9.561e-02 3.737e-02 2.558 0.011365 \*   
## Under\_D -8.206e-02 2.955e-02 -2.777 0.006095 \*\*   
## Polio 5.021e-02 1.872e-02 2.683 0.008003 \*\*   
## Dipht 5.425e-02 1.982e-02 2.737 0.006843 \*\*   
## GDP 1.211e-04 3.192e-05 3.795 0.000203 \*\*\*  
## Popl 2.472e-08 1.439e-08 1.718 0.087513 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 4.312 on 174 degrees of freedom  
## Multiple R-squared: 0.7229, Adjusted R-squared: 0.7118   
## F-statistic: 64.85 on 7 and 174 DF, p-value: < 2.2e-16

W celu potwierdzenia wybrania dobrego modelu, użyłem funkcji “step” za równo direction “backward” jak i “both”. Obie funkcje dały mi wynik identyczny dla mojego. Kod tworzy nowy model regresji liniowej “Model\_nowy\_backward” przy użyciu procedury backward selection. Procedura ta polega na usuwaniu najmniej istotnych zmiennych objaśniających, aż pozostaną tylko te, które są istotne statystycznie. Na początku tworzony jest pełny model regresji liniowej. Następnie jest on przekształcany za pomocą funkcji “step” z argumentem “direction = ‘backward’”. W końcowym kroku wyświetlane jest podsumowanie modelu “Model\_nowy\_backward”, a następnie “Model\_nowy” jest ustawiany na “Model\_nowy\_backward” i wyświetlany jest ponownie podsumowanie modelu.

Model\_backward<-step(lm(Life~Adult\_M+Infant\_D+H\_B+Measles+Under\_D+Polio+Dipht+GDP+Popl+Income,data=Dane),direction="backward")

## Start: AIC=545.5  
## Life ~ Adult\_M + Infant\_D + H\_B + Measles + Under\_D + Polio +   
## Dipht + GDP + Popl + Income  
##   
## Df Sum of Sq RSS AIC  
## - Measles 1 0.48 3230.9 543.53  
## - H\_B 1 1.42 3231.9 543.58  
## - Income 1 1.91 3232.3 543.61  
## <none> 3230.4 545.50  
## - Dipht 1 50.26 3280.7 546.31  
## - Popl 1 56.66 3287.1 546.66  
## - Infant\_D 1 80.32 3310.8 547.97  
## - Under\_D 1 109.87 3340.3 549.59  
## - Polio 1 133.91 3364.3 550.89  
## - GDP 1 260.29 3490.7 557.60  
## - Adult\_M 1 2683.87 5914.3 653.56  
##   
## Step: AIC=543.53  
## Life ~ Adult\_M + Infant\_D + H\_B + Under\_D + Polio + Dipht + GDP +   
## Popl + Income  
##   
## Df Sum of Sq RSS AIC  
## - H\_B 1 1.40 3232.3 541.60  
## - Income 1 2.24 3233.2 541.65  
## <none> 3230.9 543.53  
## - Dipht 1 50.36 3281.3 544.34  
## - Popl 1 57.05 3288.0 544.71  
## - Infant\_D 1 121.67 3352.6 548.25  
## - Polio 1 133.94 3364.9 548.92  
## - Under\_D 1 142.29 3373.2 549.37  
## - GDP 1 259.99 3490.9 555.61  
## - Adult\_M 1 2690.72 5921.6 651.79  
##   
## Step: AIC=541.6  
## Life ~ Adult\_M + Infant\_D + Under\_D + Polio + Dipht + GDP + Popl +   
## Income  
##   
## Df Sum of Sq RSS AIC  
## - Income 1 2.24 3234.6 539.73  
## <none> 3232.3 541.60  
## - Popl 1 56.75 3289.1 542.77  
## - Infant\_D 1 121.08 3353.4 546.30  
## - Polio 1 133.62 3365.9 546.98  
## - Dipht 1 139.79 3372.1 547.31  
## - Under\_D 1 141.69 3374.0 547.41  
## - GDP 1 267.74 3500.1 554.09  
## - Adult\_M 1 2755.08 5987.4 651.80  
##   
## Step: AIC=539.73  
## Life ~ Adult\_M + Infant\_D + Under\_D + Polio + Dipht + GDP + Popl  
##   
## Df Sum of Sq RSS AIC  
## <none> 3234.6 539.73  
## - Popl 1 54.89 3289.5 540.79  
## - Infant\_D 1 121.68 3356.2 544.45  
## - Polio 1 133.81 3368.4 545.11  
## - Dipht 1 139.26 3373.8 545.40  
## - Under\_D 1 143.32 3377.9 545.62  
## - GDP 1 267.75 3502.3 552.21  
## - Adult\_M 1 2752.98 5987.5 649.80

summary(Model\_backward)

##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + Under\_D + Polio + Dipht +   
## GDP + Popl, data = Dane)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -16.4750 -2.5123 0.3676 2.3051 11.8873   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.935e+01 1.742e+00 39.819 < 2e-16 \*\*\*  
## Adult\_M -4.730e-02 3.887e-03 -12.169 < 2e-16 \*\*\*  
## Infant\_D 9.561e-02 3.737e-02 2.558 0.011365 \*   
## Under\_D -8.206e-02 2.955e-02 -2.777 0.006095 \*\*   
## Polio 5.021e-02 1.872e-02 2.683 0.008003 \*\*   
## Dipht 5.425e-02 1.982e-02 2.737 0.006843 \*\*   
## GDP 1.211e-04 3.192e-05 3.795 0.000203 \*\*\*  
## Popl 2.472e-08 1.439e-08 1.718 0.087513 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 4.312 on 174 degrees of freedom  
## Multiple R-squared: 0.7229, Adjusted R-squared: 0.7118   
## F-statistic: 64.85 on 7 and 174 DF, p-value: < 2.2e-16

Model\_both<-step(lm(Life~Adult\_M+Infant\_D+H\_B+Measles+Under\_D+Polio+Dipht+GDP+Popl+Income,data=Dane),direction="both")

## Start: AIC=545.5  
## Life ~ Adult\_M + Infant\_D + H\_B + Measles + Under\_D + Polio +   
## Dipht + GDP + Popl + Income  
##   
## Df Sum of Sq RSS AIC  
## - Measles 1 0.48 3230.9 543.53  
## - H\_B 1 1.42 3231.9 543.58  
## - Income 1 1.91 3232.3 543.61  
## <none> 3230.4 545.50  
## - Dipht 1 50.26 3280.7 546.31  
## - Popl 1 56.66 3287.1 546.66  
## - Infant\_D 1 80.32 3310.8 547.97  
## - Under\_D 1 109.87 3340.3 549.59  
## - Polio 1 133.91 3364.3 550.89  
## - GDP 1 260.29 3490.7 557.60  
## - Adult\_M 1 2683.87 5914.3 653.56  
##   
## Step: AIC=543.53  
## Life ~ Adult\_M + Infant\_D + H\_B + Under\_D + Polio + Dipht + GDP +   
## Popl + Income  
##   
## Df Sum of Sq RSS AIC  
## - H\_B 1 1.40 3232.3 541.60  
## - Income 1 2.24 3233.2 541.65  
## <none> 3230.9 543.53  
## - Dipht 1 50.36 3281.3 544.34  
## - Popl 1 57.05 3288.0 544.71  
## + Measles 1 0.48 3230.4 545.50  
## - Infant\_D 1 121.67 3352.6 548.25  
## - Polio 1 133.94 3364.9 548.92  
## - Under\_D 1 142.29 3373.2 549.37  
## - GDP 1 259.99 3490.9 555.61  
## - Adult\_M 1 2690.72 5921.6 651.79  
##   
## Step: AIC=541.6  
## Life ~ Adult\_M + Infant\_D + Under\_D + Polio + Dipht + GDP + Popl +   
## Income  
##   
## Df Sum of Sq RSS AIC  
## - Income 1 2.24 3234.6 539.73  
## <none> 3232.3 541.60  
## - Popl 1 56.75 3289.1 542.77  
## + H\_B 1 1.40 3230.9 543.53  
## + Measles 1 0.46 3231.9 543.58  
## - Infant\_D 1 121.08 3353.4 546.30  
## - Polio 1 133.62 3365.9 546.98  
## - Dipht 1 139.79 3372.1 547.31  
## - Under\_D 1 141.69 3374.0 547.41  
## - GDP 1 267.74 3500.1 554.09  
## - Adult\_M 1 2755.08 5987.4 651.80  
##   
## Step: AIC=539.73  
## Life ~ Adult\_M + Infant\_D + Under\_D + Polio + Dipht + GDP + Popl  
##   
## Df Sum of Sq RSS AIC  
## <none> 3234.6 539.73  
## - Popl 1 54.89 3289.5 540.79  
## + Income 1 2.24 3232.3 541.60  
## + H\_B 1 1.40 3233.2 541.65  
## + Measles 1 0.80 3233.8 541.69  
## - Infant\_D 1 121.68 3356.2 544.45  
## - Polio 1 133.81 3368.4 545.11  
## - Dipht 1 139.26 3373.8 545.40  
## - Under\_D 1 143.32 3377.9 545.62  
## - GDP 1 267.75 3502.3 552.21  
## - Adult\_M 1 2752.98 5987.5 649.80

summary(Model\_both)

##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + Under\_D + Polio + Dipht +   
## GDP + Popl, data = Dane)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -16.4750 -2.5123 0.3676 2.3051 11.8873   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.935e+01 1.742e+00 39.819 < 2e-16 \*\*\*  
## Adult\_M -4.730e-02 3.887e-03 -12.169 < 2e-16 \*\*\*  
## Infant\_D 9.561e-02 3.737e-02 2.558 0.011365 \*   
## Under\_D -8.206e-02 2.955e-02 -2.777 0.006095 \*\*   
## Polio 5.021e-02 1.872e-02 2.683 0.008003 \*\*   
## Dipht 5.425e-02 1.982e-02 2.737 0.006843 \*\*   
## GDP 1.211e-04 3.192e-05 3.795 0.000203 \*\*\*  
## Popl 2.472e-08 1.439e-08 1.718 0.087513 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 4.312 on 174 degrees of freedom  
## Multiple R-squared: 0.7229, Adjusted R-squared: 0.7118   
## F-statistic: 64.85 on 7 and 174 DF, p-value: < 2.2e-16

Kodu przedstawia analizę reszt statystycznych (residuals) na podstawie oszacowania opartego na estymatorach. \* “RS <- rstudent(Model)” - definiuje nową zmienną “RS” jako wartości R-student reszt modelu. \* “plot(RS, ylab=”R-Student residual”, main=“R-Student residual”)” - tworzy wykres R-Student reszt. \* “RS[abs(RS)==max(abs(RS))]” - znajduje największą wartość absolutną w zmiennej “RS”. \* “dim(Dane)” - zwraca wymiary macierzy “Dane”. \* “dim(Dane)[1]” - zwraca liczbę wierszy macierzy “Dane”. \* “0.05/dim(Dane)[1]*2” - oblicza prawdopodobieństwo dwustronnego testu.* ”dim(Dane)[1]-4-1” - oblicza stopień swobody. \* “qt(0.05/(dim(Dane)[1]*2),(dim(Dane)[1]-4-1))” - oblicza kwantyl dla dwustronnego testu.* ”(-abs(RS[abs(RS)==max(abs(RS))])<qt(0.05/(dim(Dane)[1]\*2),(dim(Dane)[1]-4-1)))” - sprawdza, czy największa wartość absolutna w “RS” jest mniejsza niż kwantyl dwustronnego testu.

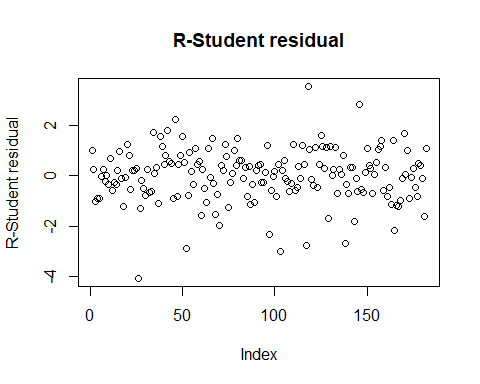
Wynik jest “TRUE”, czyli największa wartość absolutna w “RS” jest mniejsza niż kwantyl dwustronnego testu.

(RS <- rstudent(Model))

## 1 2 3 4 5 6   
## 1.023334203 0.262412754 -1.013383482 -0.903483486 -0.913156781 -0.015213520   
## 7 8 9 10 11 12   
## 0.242952203 -0.229492019 0.001261876 -0.332966301 0.671634892 -0.561060502   
## 13 14 15 16 17 18   
## -0.270781679 -0.339882040 0.214714241 0.976191332 -0.113823635 -1.230626553   
## 19 20 21 22 23 24   
## -0.046351605 1.231480550 0.793732960 -0.547296389 0.221400169 0.229791685   
## 25 26 27 28 29 30   
## 0.273818513 -4.061306753 -1.275146284 -0.173241386 -0.491721295 -0.785892147   
## 31 32 33 34 35 36   
## 0.241631341 -0.651209707 -0.638224944 1.730399228 0.076582343 0.344153583   
## 37 38 39 40 41 42   
## -1.100803597 1.564525379 1.159958707 0.445940067 0.799042649 1.809026896   
## 43 44 45 46 47 48   
## 0.580420746 0.501151287 -0.914841163 2.242821108 -0.813938531 0.449812365   
## 49 50 51 52 53 54   
## 0.817504905 1.547108690 0.541282555 -2.883727628 -0.773276590 0.918851445   
## 55 56 57 58 59 60   
## 0.173950471 -0.353599366 1.064451492 0.466158405 0.552824600 -1.578945019   
## 61 62 63 64 65 66   
## 0.234763297 -0.480943724 -1.050109286 1.096449459 -0.079188624 1.467786615   
## 67 68 69 70 71 72   
## -0.295376009 -1.528178983 -0.744291906 -1.950954126 0.420017839 0.214408844   
## 73 74 75 76 77 78   
## 1.254123467 0.773044602 -1.270032847 -0.243978236 0.096259248 0.986343745   
## 79 80 81 82 83 84   
## 0.408143258 1.477295155 0.596832705 0.626567232 -0.100794361 0.315345557   
## 85 86 87 88 89 90   
## -0.824873415 0.367743613 -1.134733535 -0.339728572 -1.053871009 0.199180650   
## 91 92 93 94 95 96   
## 0.410473037 0.452682429 -0.271775293 -0.250182884 0.142002855 1.203761565   
## 97 98 99 100 101 102   
## -2.325931428 -0.575460598 -0.044116138 0.156755345 -0.836429828 0.438391304   
## 103 104 105 106 107 108   
## -2.990215824 0.196806713 0.607760674 -0.106705083 -0.223317263 -0.605692702   
## 109 110 111 112 113 114   
## -0.306043882 1.227272336 -0.589963621 -0.469085169 0.384342768 -0.114487775   
## 115 116 117 118 119 120   
## 1.184321370 0.455125603 -2.777418278 3.543884752 1.026745667 -0.154999832   
## 121 122 123 124 125 126   
## -0.396854086 1.109353454 -0.462503565 0.437154698 1.607319731 1.155742908   
## 127 128 129 130 131 132   
## 0.293728314 1.138540223 -1.683538561 1.148938315 0.031105432 0.241567069   
## 133 134 135 136 137 138   
## 1.131679473 -0.682442145 0.266168378 0.035215507 0.796320550 -2.669991967   
## 139 140 141 142 143 144   
## -0.337072015 -0.679764799 0.320581874 0.342732468 -1.811657272 -0.098739108   
## 145 146 147 148 149 150   
## -0.624008836 2.830762674 -0.539010518 -0.642895473 0.151362342 1.085328421   
## 151 152 153 154 155 156   
## 0.398716556 0.285371202 -0.694530663 0.047931289 0.544787550 1.027560398   
## 157 158 159 160 161 162   
## 1.153994165 1.386293181 -0.568225813 0.339170801 -0.826027842 -0.446803060   
## 163 164 165 166 167 168   
## -1.115845717 1.415353364 -2.149828953 -1.180955476 -1.222909747 -0.967192297   
## 169 170 171 172 173 174   
## -0.095609992 1.684867748 0.055532210 0.988162446 -0.880918068 -0.050072552   
## 175 176 177 178 179 180   
## 0.289617284 -0.441532692 -0.828668802 0.501050943 0.425020876 -0.123891612   
## 181 182   
## -1.592907350 1.067742383

Przedstawienie na wykresie

plot(RS,ylab="R-Student residual",main="R-Student residual")



Uzyskanie wartości odstających.

RS[abs(RS)==max(abs(RS))]

## 26   
## -4.061307

dim(Dane)

## [1] 182 11

dim(Dane)[1]

## [1] 182

0.05/dim(Dane)[1]\*2

## [1] 0.0005494505

dim(Dane)[1]-4-1

## [1] 177

qt(0.05/(dim(Dane)[1]\*2),(dim(Dane)[1]-4-1))

## [1] -3.712521

(-abs(RS[abs(RS)==max(abs(RS))])<qt(0.05/(dim(Dane)[1]\*2),(dim(Dane)[1]-4-1)))

## 26   
## TRUE

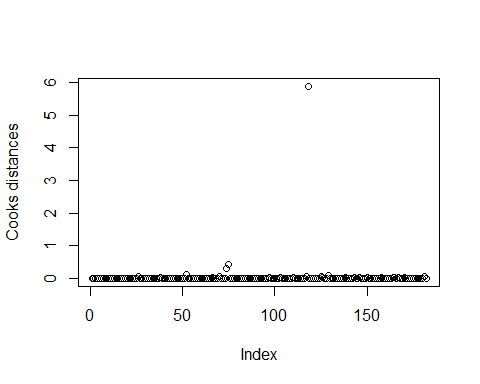
Dzięki odległości cooka, mogę usunąć wartości odstające i stworzyć nowy model o nazwie “Model\_no\_outliers”. Kod wykorzystuje funkcję cooks.distance do obliczenia odległości Cooka dla każdego punktu danych. Odległość Cooka jest miarą, jak bardzo punkt danych wpływa na wynik modelu regresji liniowej. Wartość odległości Cooka jest wyznaczana jako iloczyn między zmienną zależną (Life) i skorygowaną estymatą modelu bez tego punktu danych. Wartości odległości Cooka powyżej 1 są uważane za “outliers”, tj. punkty danych, które mają nieproporcjonalnie duży wpływ na wynik modelu. Następnie wykonywane jest wykreślenie wartości odległości Cooka, a następnie tworze nowy model regresji liniowej bez punktów danych, które są outliers. W rezultacie, wynik końcowy jest wynikiem modelu regresji liniowej bez wartości odstających.

(cooks\_dis <- cooks.distance(Model))

## 1 2 3 4 5 6   
## 1.084358e-02 1.158679e-04 4.041842e-03 1.033076e-02 2.245975e-03 6.714757e-07   
## 7 8 9 10 11 12   
## 8.708425e-05 9.492557e-04 1.517303e-08 1.796033e-04 4.433065e-04 7.350522e-04   
## 13 14 15 16 17 18   
## 1.569365e-04 1.499690e-04 7.685861e-05 1.557227e-03 1.463874e-05 2.546840e-03   
## 19 20 21 22 23 24   
## 4.061338e-06 2.160770e-03 9.745246e-04 7.601425e-04 6.239910e-05 8.403537e-05   
## 25 26 27 28 29 30   
## 6.198951e-05 7.059110e-02 5.146900e-03 3.544015e-05 3.074601e-04 3.036962e-03   
## 31 32 33 34 35 36   
## 5.586022e-04 2.971755e-03 2.307590e-03 4.006873e-03 3.053602e-05 4.743459e-04   
## 37 38 39 40 41 42   
## 2.171488e-03 3.032421e-02 2.001798e-03 2.217774e-04 1.148955e-03 6.854474e-03   
## 43 44 45 46 47 48   
## 4.264796e-04 2.664669e-04 6.059988e-03 6.830475e-03 1.129106e-03 1.611891e-04   
## 49 50 51 52 53 54   
## 1.043956e-03 2.760880e-03 3.194454e-04 1.194298e-01 1.451466e-03 1.059199e-03   
## 55 56 57 58 59 60   
## 1.317752e-04 1.898576e-04 1.694939e-03 1.352787e-03 5.445132e-03 6.684669e-03   
## 61 62 63 64 65 66   
## 5.857327e-05 3.574510e-03 2.628266e-03 2.569498e-03 6.967644e-06 2.505006e-02   
## 67 68 69 70 71 72   
## 3.121076e-04 5.797469e-03 8.629252e-04 6.519839e-02 2.265100e-04 5.534666e-05   
## 73 74 75 76 77 78   
## 2.819776e-03 2.942617e-01 4.297374e-01 7.765277e-05 3.144929e-05 1.405190e-03   
## 79 80 81 82 83 84   
## 1.037284e-03 4.876417e-03 3.273077e-04 2.235635e-03 1.675840e-05 2.216640e-04   
## 85 86 87 88 89 90   
## 2.464354e-03 1.889457e-03 4.743857e-03 1.333189e-04 1.179606e-03 4.804916e-05   
## 91 92 93 94 95 96   
## 2.710941e-04 5.348596e-03 2.247925e-04 6.615451e-05 2.661073e-05 3.096057e-03   
## 97 98 99 100 101 102   
## 2.188943e-02 1.773639e-03 2.081422e-06 3.897792e-05 2.152629e-03 5.399223e-04   
## 103 104 105 106 107 108   
## 3.302102e-02 4.206278e-05 3.321832e-04 1.328257e-05 1.027624e-04 9.295949e-04   
## 109 110 111 112 113 114   
## 1.390226e-04 1.826201e-02 4.237468e-04 4.554140e-04 2.388011e-03 1.274920e-04   
## 115 116 117 118 119 120   
## 2.201075e-03 3.247714e-04 6.103057e-02 5.882092e+00 1.626442e-03 3.471825e-05   
## 121 122 123 124 125 126   
## 2.344821e-03 1.830146e-03 5.171257e-04 2.358737e-04 5.984783e-02 2.495201e-02   
## 127 128 129 130 131 132   
## 2.016877e-04 2.352226e-03 7.660300e-02 2.083046e-03 8.080530e-07 6.191294e-05   
## 133 134 135 136 137 138   
## 2.459306e-03 1.123319e-03 8.086594e-05 1.403479e-06 1.259203e-03 2.029298e-02   
## 139 140 141 142 143 144   
## 1.730489e-04 5.820324e-04 1.073002e-04 1.683776e-04 2.505885e-02 1.529818e-04   
## 145 146 147 148 149 150   
## 9.258432e-04 1.282444e-02 4.503165e-04 1.881848e-03 8.713101e-05 2.343858e-02   
## 151 152 153 154 155 156   
## 7.590964e-04 1.035859e-04 1.344452e-03 2.241222e-06 7.078227e-03 1.847264e-03   
## 157 158 159 160 161 162   
## 2.119699e-03 1.997196e-02 4.627543e-04 1.352773e-04 1.739250e-03 2.357439e-04   
## 163 164 165 166 167 168   
## 3.401139e-03 3.024241e-02 1.269319e-02 4.242884e-03 1.544484e-02 1.758608e-03   
## 169 170 171 172 173 174   
## 2.480695e-05 2.067676e-02 5.010622e-06 1.404717e-03 3.072742e-03 6.328392e-06   
## 175 176 177 178 179 180   
## 1.041605e-04 3.131750e-04 3.372494e-03 2.059599e-04 1.914949e-04 2.645504e-05   
## 181 182   
## 4.527136e-02 5.129188e-03

Tworzenie wykresu

plot(cooks\_dis,ylab="Cooks distances")



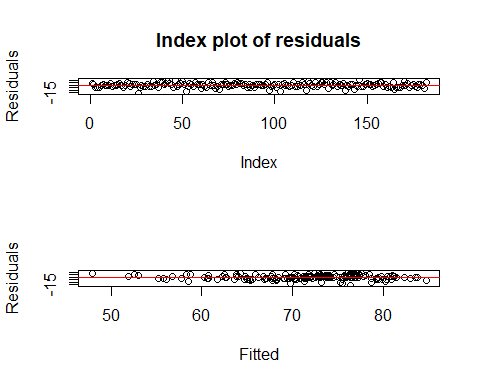
“Model\_no\_outliers” ma lepsze wyniki niż Model.

Model\_no\_outliers <-lm(Life~Adult\_M+Infant\_D+Under\_D+Polio+Dipht+GDP+Popl,data=Dane,subset=(cooks\_dis<max(cooks\_dis)))  
summary(Model\_no\_outliers)

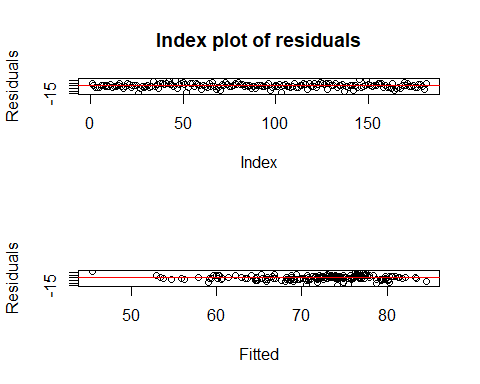
##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + Under\_D + Polio + Dipht +   
## GDP + Popl, data = Dane, subset = (cooks\_dis < max(cooks\_dis)))  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -14.2143 -2.3349 0.1912 2.2708 11.9048   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 7.042e+01 1.714e+00 41.099 < 2e-16 \*\*\*  
## Adult\_M -4.508e-02 3.816e-03 -11.813 < 2e-16 \*\*\*  
## Infant\_D 2.787e-01 6.307e-02 4.418 1.75e-05 \*\*\*  
## Under\_D -2.341e-01 5.158e-02 -4.539 1.05e-05 \*\*\*  
## Polio 4.282e-02 1.824e-02 2.347 0.020060 \*   
## Dipht 5.017e-02 1.923e-02 2.609 0.009875 \*\*   
## GDP 1.194e-04 3.091e-05 3.863 0.000158 \*\*\*  
## Popl 1.072e-08 1.448e-08 0.740 0.460100   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 4.175 on 173 degrees of freedom  
## Multiple R-squared: 0.7349, Adjusted R-squared: 0.7242   
## F-statistic: 68.51 on 7 and 173 DF, p-value: < 2.2e-16

Kod tworzy wykres diagnostycznych reszt dla modelu statystycznego. \* “par(mfrow=c(2,1))” - ustawia parametr okna, aby rysować dwa wykresy na jednej stronie. \* “plot(Model$res, ylab="Residuals", main="Index plot of residuals")" - tworzy wykres indexu reszt dla modelu "Model". \* "abline(h=0, col="red")" - dodaje linie na wykresie, reprezentującą wartość 0, w kolorze czerwonym. \* "plot(Model$fit, Model$res, xlab="Fitted", ylab="Residuals")" - tworzy wykres reszt, gdzie na osi x jest "Fitted", a na osi y jest "Residuals". \* "abline(h=0, col="red")" - dodaje linie na wykresie, reprezentującą wartość 0, w kolorze czerwonym. \* "plot(Model\_no\_outliers$res, ylab=”Residuals”, main=“Index plot of residuals”)” - tworzy wykres indexu reszt dla modelu “Model\_no\_outliers”. \* “abline(h=0, col=”red”)” - dodaje linie na wykresie, reprezentującą wartość 0, w kolorze czerwonym. \* “plot(Model\_no\_outliersres, xlab=”Fitted”, ylab=“Residuals”)” - tworzy wykres skojarzony reszt, gdzie na osi x jest “Fitted”, a na osi y jest “Residuals”, dla modelu “Model\_no\_outliers”. \* “abline(h=0, col=”red”)” - dodaje linie na wykresie, reprezentującą wartość 0, w kolorze czerwonym

par(mfrow=c(2,1))  
plot(Model$res,ylab="Residuals",main="Index plot of residuals")  
abline(h=0,col="red")  
plot(Model$fit,Model$res,xlab="Fitted",ylab="Residuals")  
abline(h=0,col="red")



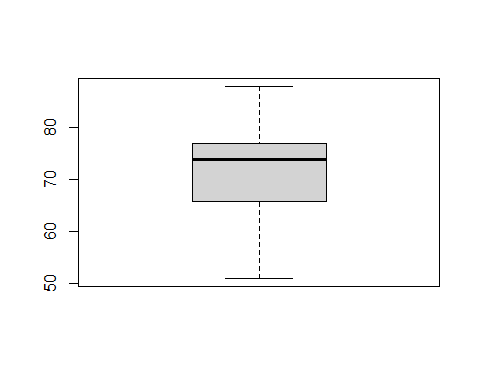
par(mfrow=c(2,1))  
plot(Model\_no\_outliers$res,ylab="Residuals",main="Index plot of residuals")  
abline(h=0,col="red")  
plot(Model\_no\_outliers$fit,Model\_no\_outliers$res,xlab="Fitted",ylab="Residuals")  
abline(h=0,col="red")



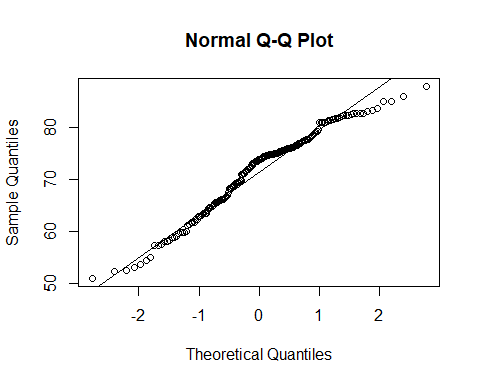
Test rozkładu normalnego zacząłem od graficznego przedstawienia wszystkich zmiennych w modelu. Za pomocą funkcji “boxplot” oraz “qqnorm”. Kod przedstawia. \* “boxplot(Dane$Life)" - rysuje wykres typu boxplot, który pokazuje statystyki centralne i rozrzut danych dla zmiennej "Life". \* "qqnorm(Dane$”Life” )” - rysuje wykres typu Q-Q plot, który pokazuje, jak bardzo dane odpowiadają rozkładowi normalnemu. \* “qqline(Dane$”Life” )” - dodaje linię do wykresu typu Q-Q plot, która pokazuje, jak dane powinny odpowiadać rozkładowi normalnemu.

Jeśli dane są zgodne z rozkładem normalnym, to na wykresie typu Q-Q plot powinny być one rozłożone wokół linii. Jeśli nie są, oznacza to, że dane nie odpowiadają rozkładowi normalnemu. Wykres pudełkowy składa się z osi. Nad osią umieszczony jest prostokąt (pudełko), którego lewy bok jest wyznaczony przez pierwszy kwartyl, zaś prawy bok przez trzeci kwartyl. Szerokość pudełka odpowiada wartości rozstępu ćwiartkowego. Wewnątrz prostokąta znajduje się pionowa linia, określająca wartość mediany. W pierwszym przykładzie można zauważyć, że dane z kolumny “Life” całkiem dobrze wypadają na rozkładzie normalnym.

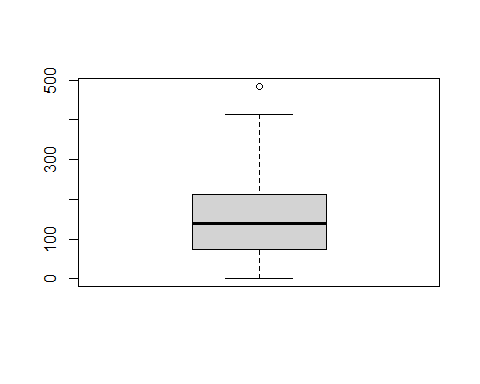
boxplot(Dane$Life)



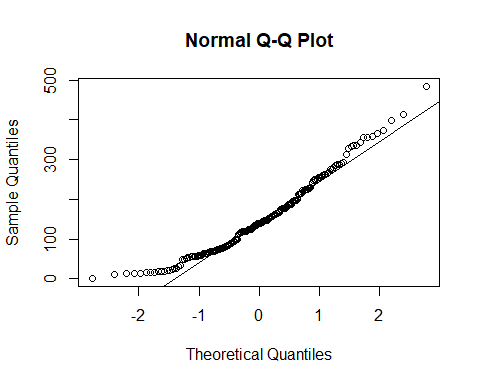
qqnorm(Dane$"Life" )  
qqline(Dane$"Life" )



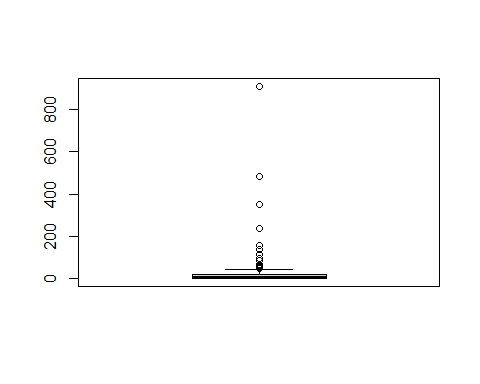
boxplot(Dane$Adult\_M)



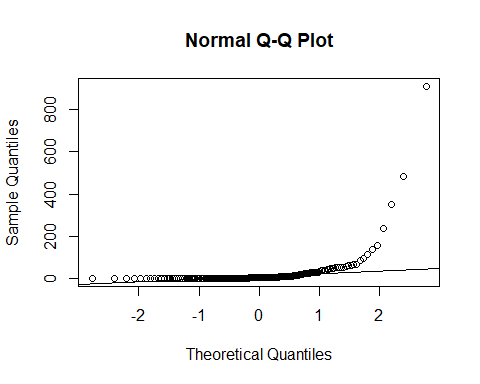
qqnorm(Dane$"Adult\_M" )  
qqline(Dane$"Adult\_M" )



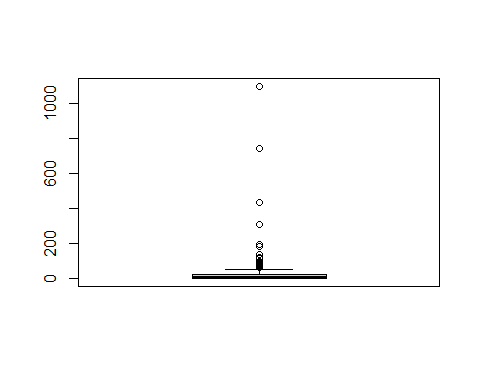
boxplot(Dane$Infant\_D)



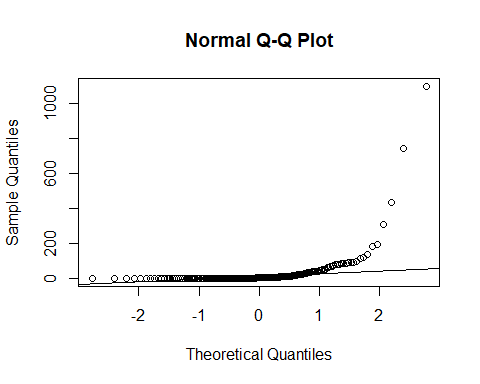
qqnorm(Dane$"Infant\_D")  
qqline(Dane$"Infant\_D")



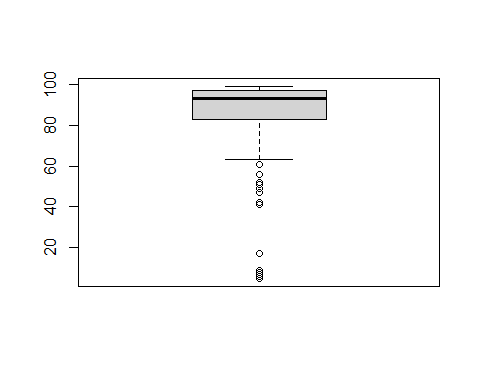
boxplot(Dane$Under\_D)



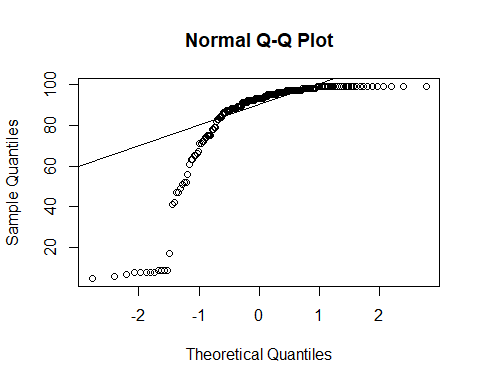
qqnorm(Dane$"Under\_D")  
qqline(Dane$"Under\_D")



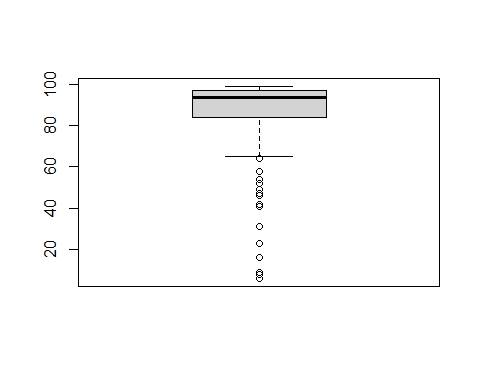
boxplot(Dane$Polio)



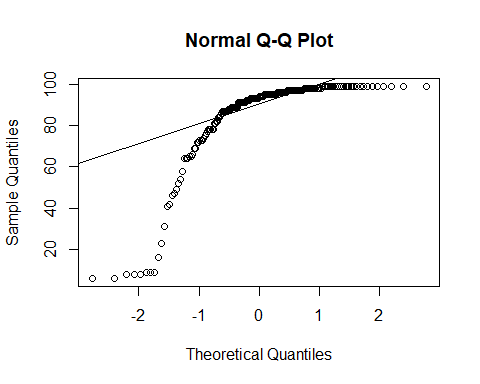
qqnorm(Dane$"Polio")  
qqline(Dane$"Polio")



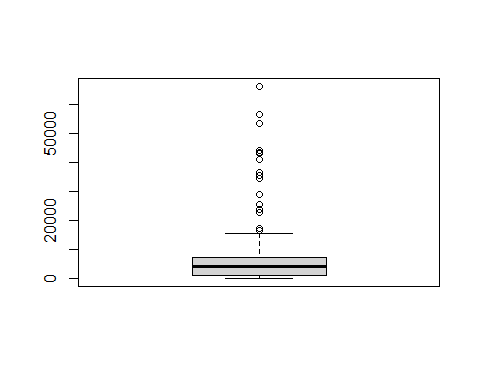
boxplot(Dane$Dipht)



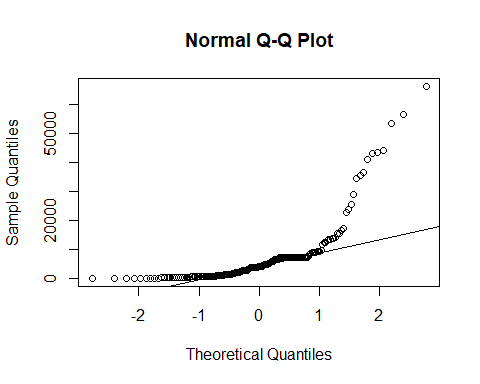
qqnorm(Dane$"Dipht")  
qqline(Dane$"Dipht")



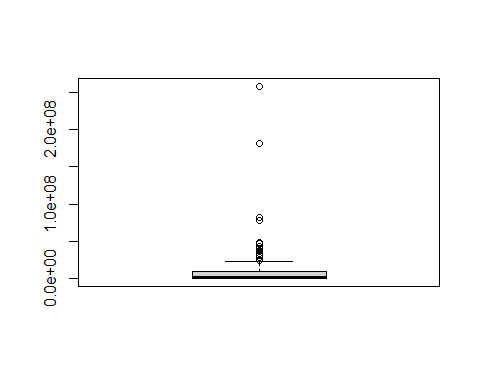
boxplot(Dane$GDP)



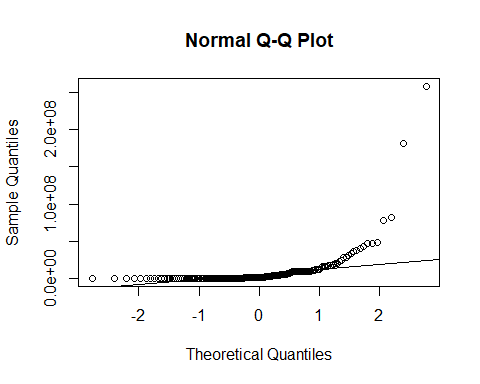
qqnorm(Dane$"GDP")  
qqline(Dane$"GDP")



boxplot(Dane$Popl)



qqnorm(Dane$"Popl")  
qqline(Dane$"Popl")



Również przedstawiłem graficznie to samo za pomocą pakietu “ggpubr”, oraz funkcji “ggqqplot”. Wykres Q-Q jest rysowany między daną próbą a rozkładem normalnym. Wykreślana jest również 45- stopniowa linia odniesienia, aby ocenić, jak bliskie są wartości próbki rozkładowi normalnemu.

library("ggpubr")

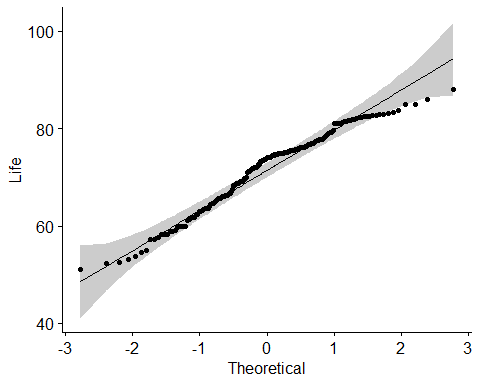
## Warning: pakiet 'ggpubr' został zbudowany w wersji R 4.2.2

## Ładowanie wymaganego pakietu: ggplot2

## Warning: pakiet 'ggplot2' został zbudowany w wersji R 4.2.2

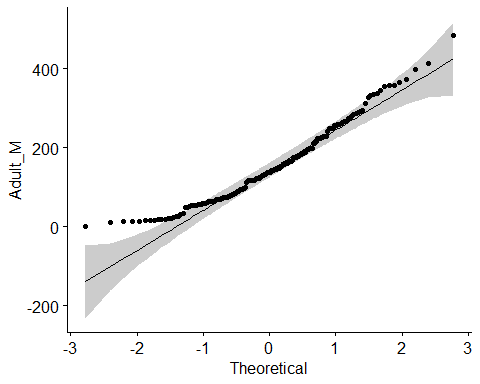
ggqqplot(Dane$Life, ylab = "Life")

## Warning: The following aesthetics were dropped during statistical transformation: sample  
## ℹ This can happen when ggplot fails to infer the correct grouping structure in  
## the data.  
## ℹ Did you forget to specify a `group` aesthetic or to convert a numerical  
## variable into a factor?  
## The following aesthetics were dropped during statistical transformation: sample  
## ℹ This can happen when ggplot fails to infer the correct grouping structure in  
## the data.  
## ℹ Did you forget to specify a `group` aesthetic or to convert a numerical  
## variable into a factor?



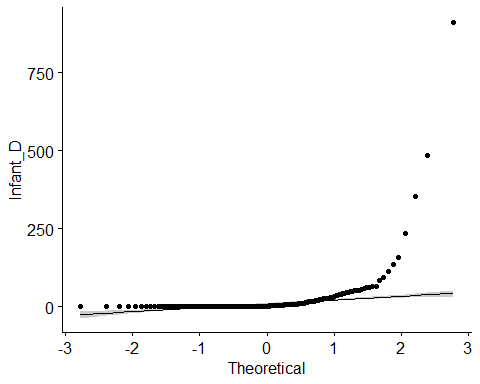
ggqqplot(Dane$Adult\_M, ylab = "Adult\_M")

## Warning: The following aesthetics were dropped during statistical transformation: sample  
## ℹ This can happen when ggplot fails to infer the correct grouping structure in  
## the data.  
## ℹ Did you forget to specify a `group` aesthetic or to convert a numerical  
## variable into a factor?  
## The following aesthetics were dropped during statistical transformation: sample  
## ℹ This can happen when ggplot fails to infer the correct grouping structure in  
## the data.  
## ℹ Did you forget to specify a `group` aesthetic or to convert a numerical  
## variable into a factor?



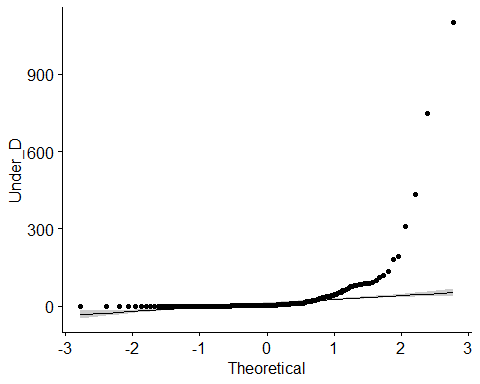
ggqqplot(Dane$Infant\_D, ylab = "Infant\_D")

## Warning: The following aesthetics were dropped during statistical transformation: sample  
## ℹ This can happen when ggplot fails to infer the correct grouping structure in  
## the data.  
## ℹ Did you forget to specify a `group` aesthetic or to convert a numerical  
## variable into a factor?  
## The following aesthetics were dropped during statistical transformation: sample  
## ℹ This can happen when ggplot fails to infer the correct grouping structure in  
## the data.  
## ℹ Did you forget to specify a `group` aesthetic or to convert a numerical  
## variable into a factor?



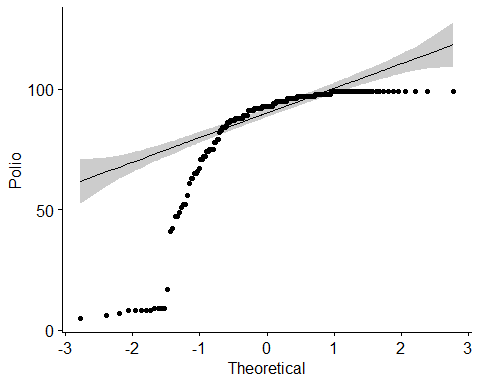
ggqqplot(Dane$Under\_D, ylab = "Under\_D")

## Warning: The following aesthetics were dropped during statistical transformation: sample  
## ℹ This can happen when ggplot fails to infer the correct grouping structure in  
## the data.  
## ℹ Did you forget to specify a `group` aesthetic or to convert a numerical  
## variable into a factor?  
## The following aesthetics were dropped during statistical transformation: sample  
## ℹ This can happen when ggplot fails to infer the correct grouping structure in  
## the data.  
## ℹ Did you forget to specify a `group` aesthetic or to convert a numerical  
## variable into a factor?



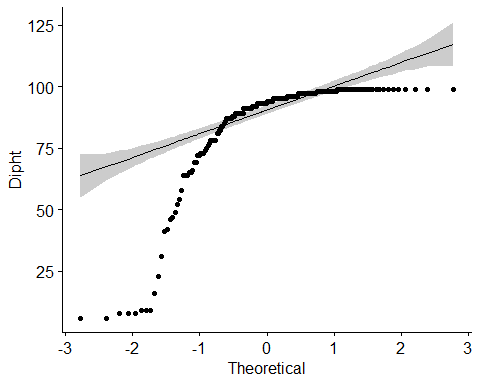
ggqqplot(Dane$Polio, ylab = "Polio")

## Warning: The following aesthetics were dropped during statistical transformation: sample  
## ℹ This can happen when ggplot fails to infer the correct grouping structure in  
## the data.  
## ℹ Did you forget to specify a `group` aesthetic or to convert a numerical  
## variable into a factor?  
## The following aesthetics were dropped during statistical transformation: sample  
## ℹ This can happen when ggplot fails to infer the correct grouping structure in  
## the data.  
## ℹ Did you forget to specify a `group` aesthetic or to convert a numerical  
## variable into a factor?



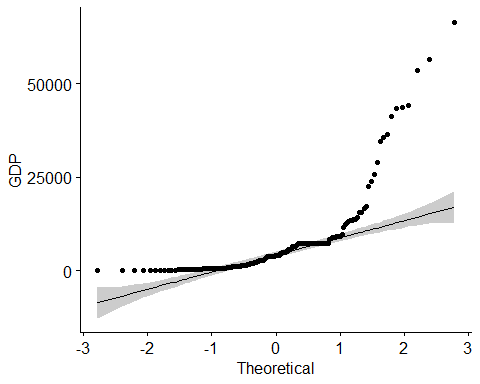
ggqqplot(Dane$Dipht, ylab = "Dipht")

## Warning: The following aesthetics were dropped during statistical transformation: sample  
## ℹ This can happen when ggplot fails to infer the correct grouping structure in  
## the data.  
## ℹ Did you forget to specify a `group` aesthetic or to convert a numerical  
## variable into a factor?  
## The following aesthetics were dropped during statistical transformation: sample  
## ℹ This can happen when ggplot fails to infer the correct grouping structure in  
## the data.  
## ℹ Did you forget to specify a `group` aesthetic or to convert a numerical  
## variable into a factor?



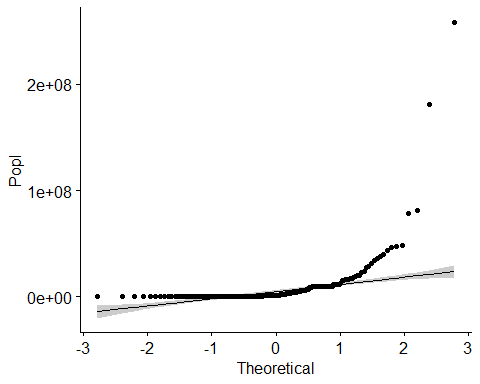
ggqqplot(Dane$GDP, ylab = "GDP")

## Warning: The following aesthetics were dropped during statistical transformation: sample  
## ℹ This can happen when ggplot fails to infer the correct grouping structure in  
## the data.  
## ℹ Did you forget to specify a `group` aesthetic or to convert a numerical  
## variable into a factor?  
## The following aesthetics were dropped during statistical transformation: sample  
## ℹ This can happen when ggplot fails to infer the correct grouping structure in  
## the data.  
## ℹ Did you forget to specify a `group` aesthetic or to convert a numerical  
## variable into a factor?



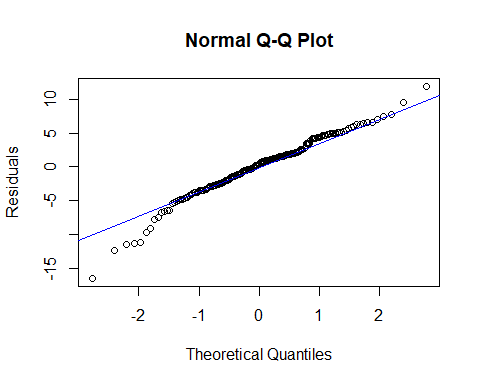
ggqqplot(Dane$Popl, ylab = "Popl")

## Warning: The following aesthetics were dropped during statistical transformation: sample  
## ℹ This can happen when ggplot fails to infer the correct grouping structure in  
## the data.  
## ℹ Did you forget to specify a `group` aesthetic or to convert a numerical  
## variable into a factor?  
## The following aesthetics were dropped during statistical transformation: sample  
## ℹ This can happen when ggplot fails to infer the correct grouping structure in  
## the data.  
## ℹ Did you forget to specify a `group` aesthetic or to convert a numerical  
## variable into a factor?



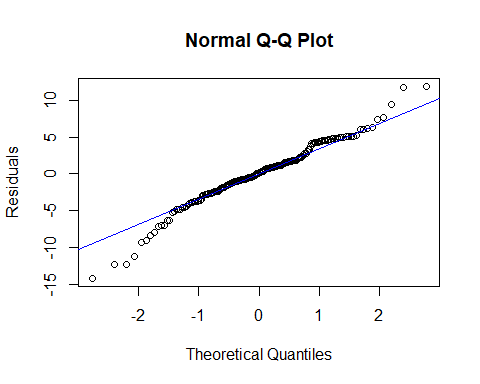
Przedstawienie na wykresie rozkładu normlanego modeli “Model” oraz “Model\_no\_outliers”. Kod pokazuje wizualizację danych i sprawdza, czy reszty regresji są zgodne z rozkładem normalnym. \* “par(mfrow=c(1,1))” - oznacza, że na każdej stronie wyników będzie tylko jeden wykres. \* “qqnorm(Model$res,ylab="Residuals")" - rysuje wykres typu Q-Q plot dla reszt regresji Model, z opisem osi y jako "Residuals". \* "qqline(Model$res, col=”blue”)” - dodaje linię do wykresu typu Q-Q plot, która pokazuje, jak dane powinny odpowiadać rozkładowi normalnemu. Kolor linii to niebieski.

par(mfrow=c(1,1))  
qqnorm(Model$res,ylab="Residuals")  
qqline(Model$res, col="blue")

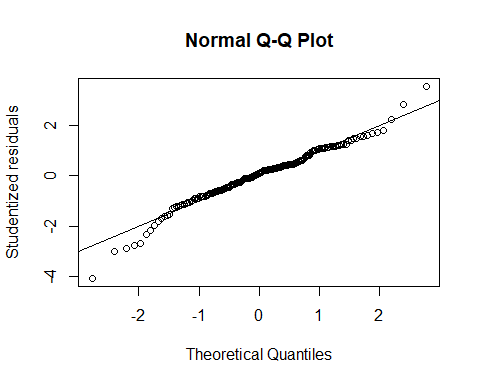


Przedstawienie na wykresie rozkładu normlanego modeli “Model” oraz “Model\_no\_outliers” z użyciem funkcji “rstudent”. \* “qqnorm(Model\_no\_outliers$res,ylab="Residuals")" - rysuje wykres typu Q-Q plot dla reszt regresji Model\_no\_outliers, z opisem osi y jako "Residuals". \* "qqline(Model\_no\_outliers$res, col=”blue”)” - dodaje linię do wykresu typu Q-Q plot, która pokazuje, jak dane powinny odpowiadać rozkładowi normalnemu. Kolor linii to niebieski. \* “qqnorm(rstudent(Model),ylab=”Studentized residuals”)” - rysuje wykres typu Q-Q plot dla studentyzowanych reszt regresji Model, z opisem osi y jako “Studentized residuals”. \* “abline(0,1)” - dodaje prostą do wykresu typu Q-Q plot, która pokazuje, jak dane powinny odpowiadać rozkładowi normalnemu. \* “qqnorm(rstudent(Model\_no\_outliers),ylab=”Studentized residuals”)” - rysuje wykres typu Q-Q plot dla studentyzowanych reszt regresji Model\_no\_outliers, z opisem osi y jako “Studentized residuals”. \* “abline(0,1)” - dodaje prostą do wykresu typu Q-Q plot, która pokazuje, jak dane powinny odpowiadać rozkładowi normalnemu.

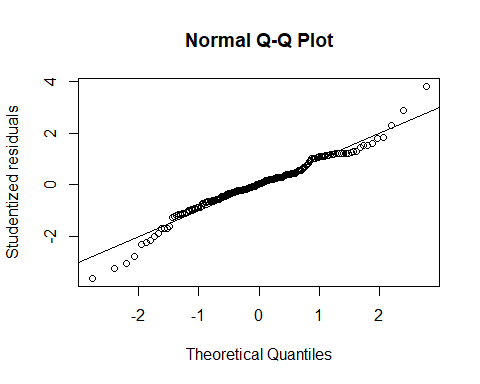
par(mfrow=c(1,1))  
qqnorm(Model\_no\_outliers$res,ylab="Residuals")  
qqline(Model\_no\_outliers$res, col="blue")



qqnorm(rstudent(Model),ylab="Studentized residuals")  
abline(0,1)



qqnorm(rstudent(Model\_no\_outliers),ylab="Studentized residuals")  
abline(0,1)



W celu sprawdzenia rozkładu normlanego użyje testu Shapiro-Wilka. P-value w teście Shapiro-Wilka jest miarą siły dowodu przeciwko hipotezie, że dane pochodzą z rozkładu normalnego. Im mniejsze p-value, tym mocniejszy jest dowód przeciwko hipotezie normalności. Jeśli p-value jest mniejsze niż poziom istotności (np. 0,05), to odrzucamy hipotezę normalności i stwierdzamy, że dane nie pochodzą z rozkładu normalnego. W przeciwnym razie przyjmujemy hipotezę normalności. Za pomocą “shapiro.test” mogłem uzyskać następująće wyniki \* Dla modelu “Model” wynik wyszedł W = 0.97386, p-value = 0.001701. Poniżej 0,05 p-value rozkład nie jest normalny. \* Dla modelu “Model2” wynik wyszedł W = 0.97416, p-value = 0.001852. Poniżej 0,05 p-value rozkład nie jest normalny. \* Dla modelu “Model3” wynik wyszedł W = 0.97468, p-value = 0.002148. Poniżej 0,05 p-value rozkład nie jest normalny. \* Dla modelu “Model4” wynik wyszedł W = 0.97386, p-value = 0.001701. Poniżej 0,05 p-value rozkład nie jest normalny. \* Dla modelu “Model5” wynik wyszedł W = 0.97026, p-value = 0.0006324. Poniżej 0,05 p-value rozkład nie jest normalny. \* Dla modelu “Model\_no\_outlires” wynik wyszedł W = 0.97233, p-value = 0.001155. Poniżej 0,05 p-value rozkład nie jest normalny. Wszystkie modele uzyskały wynik p-value poniżej 0,05, co czynni je niepochodzących z rozkładu normlanego. Jednocześnie wyniki są bardzo zbliżone. Nie zmieniam modelu, a modelem jest model “Model4”.

shapiro.test(Model$res)

##   
## Shapiro-Wilk normality test  
##   
## data: Model$res  
## W = 0.97386, p-value = 0.001701

shapiro.test(Model2$res)

##   
## Shapiro-Wilk normality test  
##   
## data: Model2$res  
## W = 0.97416, p-value = 0.001852

shapiro.test(Model3$res)

##   
## Shapiro-Wilk normality test  
##   
## data: Model3$res  
## W = 0.97468, p-value = 0.002148

shapiro.test(Model4$res)

##   
## Shapiro-Wilk normality test  
##   
## data: Model4$res  
## W = 0.97386, p-value = 0.001701

shapiro.test(Model5$res)

##   
## Shapiro-Wilk normality test  
##   
## data: Model5$res  
## W = 0.97026, p-value = 0.0006324

shapiro.test(Model\_no\_outliers$res)

##   
## Shapiro-Wilk normality test  
##   
## data: Model\_no\_outliers$res  
## W = 0.97233, p-value = 0.001155

Test niezależności Kod ładuje bibliotekę “randtests” i wywołuje funkcję “runs.test” na zmiennej “Model$res" oraz Model\_no\_outliers$res”. Funkcja “runs.test” jest jednym z testów dla oceny losowości w sekwencji wartości. Test jest wykonywany na “residulas” z modelu statystycznego. Wynik testu przedstawia czy wartości “residuals” są losowe, czy są skorelowane. Jeśli wartości danych byłyby losowe, można oczekiwać, że liczba ciągów wzrostów i spadków wartości byłaby podobna. Jeśli natomiast wartości danych byłyby skorelowane, można oczekiwać, że będzie więcej ciągów wzrostów niż spadków, lub na odwrót. W obu przypadkach moje model są powyżej 0.05 p-value, oznacza to, że nie odrzucaja hipotezę zerową i wartości są losowe.

library(randtests)

runs.test(Model$res)

##   
## Runs Test  
##   
## data: Model$res  
## statistic = 0.14866, runs = 93, n1 = 91, n2 = 91, n = 182, p-value =  
## 0.8818  
## alternative hypothesis: nonrandomness

runs.test(Model\_no\_outliers$res)

##   
## Runs Test  
##   
## data: Model\_no\_outliers$res  
## statistic = 0, runs = 91, n1 = 90, n2 = 90, n = 180, p-value = 1  
## alternative hypothesis: nonrandomness

Kod oznacza wykonanie analiz regresji dla zmiennych wyjaśniających w zbiorze danych Dane. W pierwszej linijce tworzy się macierz explanatory z wykluczeniem pierwszej kolumny Dane. Następnie tworzy się model regresji dla pierwszej kolumny explanatory jako zmiennej zależnej i pozostałych kolumn explanatory jako zmiennych niezależnych. Podsumowanie modelu jest wyświetlane za pomocą funkcji summary(). Wartość współczynnika determinacji dla tego modelu jest pobierana z jego podsumowania.

explanatory<-as.matrix(Dane[,-1])

summary(lm(explanatory[,1]~explanatory[,-1]))

##   
## Call:  
## lm(formula = explanatory[, 1] ~ explanatory[, -1])  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -232.418 -50.191 0.923 43.754 246.229   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.636e+02 2.733e+01 9.647 < 2e-16 \*\*\*  
## explanatory[, -1]Infant\_D -1.762e+00 8.844e-01 -1.992 0.04791 \*   
## explanatory[, -1]H\_B 1.131e+00 6.450e-01 1.753 0.08145 .   
## explanatory[, -1]Measles -7.701e-04 1.511e-03 -0.510 0.61098   
## explanatory[, -1]Under\_D 1.557e+00 6.518e-01 2.389 0.01799 \*   
## explanatory[, -1]Polio -1.039e+00 3.547e-01 -2.930 0.00385 \*\*   
## explanatory[, -1]Dipht -1.291e+00 7.421e-01 -1.739 0.08384 .   
## explanatory[, -1]GDP -1.698e-03 6.102e-04 -2.783 0.00600 \*\*   
## explanatory[, -1]Popl -4.823e-07 2.791e-07 -1.728 0.08575 .   
## explanatory[, -1]Income 3.595e-01 6.796e-01 0.529 0.59749   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 83.72 on 172 degrees of freedom  
## Multiple R-squared: 0.2791, Adjusted R-squared: 0.2414   
## F-statistic: 7.4 on 9 and 172 DF, p-value: 4.237e-09

summary(lm(explanatory[,1]~explanatory[,-1]))$r.squared

## [1] 0.2791155

W tej części kodu zaimportowana jest biblioteka “car” i wykonane jest obliczenie Variance Inflation Factor (VIF) dla modelu “Model” automatycznie. VIF jest miarą multicollinearity (wzajemnej korelacji) pomiędzy zmiennymi niezależnymi w modelu regresji. Wartość VIF powyżej 10 sugeruje występowanie wzajemnej korelacji i konieczność usunięcia jednej z kolumn explanatory. W moim modelu takimi wartościami są “Infant\_D” oraz “Under\_D”.

library(car)

## Warning: pakiet 'car' został zbudowany w wersji R 4.2.2

## Ładowanie wymaganego pakietu: carData

## Warning: pakiet 'carData' został zbudowany w wersji R 4.2.2

vif(Model)

## Adult\_M Infant\_D Under\_D Polio Dipht GDP Popl   
## 1.359195 95.660047 99.401120 2.015937 1.889497 1.147135 1.347827

Za pomocą “Model”, mogłem dostać współczynniki dla wszystkich zmiennych.

Model$coefficients

## (Intercept) Adult\_M Infant\_D Under\_D Polio   
## 6.935148e+01 -4.730148e-02 9.560769e-02 -8.205823e-02 5.021274e-02   
## Dipht GDP Popl   
## 5.425434e-02 1.211242e-04 2.472045e-08

Przykładowa interpretacja współczynnika dla “Adult\_M”: Jeśli zwiększymy oczekiwane prawdopodobieństwo śmierci między 15 a 60 rokiem życia na 1000 mieszkańców o rok, to oczekiwana długość życia w kraju zmniejszy się o 4,73 lata, przy założeniu niezmienionych wartości pozostałych zmiennych objaśniających tzn. zgodnie z zasadą ceteris Paribus. Stworzyłem nowe zmienne do których przypisałem współczynniki. Dostając bety z odpowiadającymi współczynnikami.

beta\_0<-Model$coefficients['(Intercept)']  
beta\_0

## (Intercept)   
## 69.35148

beta\_adultm<-Model$coefficients['Adult\_M']  
beta\_adultm

## Adult\_M   
## -0.04730148

beta\_infantd<-Model$coefficients['Infant\_D']  
beta\_infantd

## Infant\_D   
## 0.09560769

beta\_underd<-Model$coefficients['Under\_D']  
beta\_underd

## Under\_D   
## -0.08205823

beta\_polio<-Model$coefficients['Polio']  
beta\_polio

## Polio   
## 0.05021274

beta\_dipht<-Model$coefficients['Dipht']  
beta\_dipht

## Dipht   
## 0.05425434

beta\_gdp<-Model$coefficients['GDP']  
beta\_gdp

## GDP   
## 0.0001211242

beta\_popl<-Model$coefficients['Popl']  
beta\_popl

## Popl   
## 2.472045e-08

Dzięki stworzeniu bet, mogę dokonać predykcji. Stworzyłem 3 różne predykcje z różnymi liczbami. Przykładowo w “forecast” zakładając, że: \* “Adult Mortality” jest na poziomie 100 \* “Infant deaths” jest na poziomie 56 \* “Under-five deaths” jest na poziomie 80 \* “Polio” jest na poziomie 70 \* “Diphtheria” jest na poziomie 80 \* “GDP” jest na poziomie 300 \* “Population” jest na poziomie 10000000 Wynik oczekiwanej długości życia według modelu wynosi 71.54948.

(forecast<-beta\_0+beta\_adultm\*100+beta\_infantd\*56+beta\_underd\*80+beta\_polio\*70+beta\_dipht\*80+beta\_gdp\*300+beta\_popl\*10000000)

## (Intercept)   
## 71.54948

(forecast2<-beta\_0+beta\_adultm\*500+beta\_infantd\*86+beta\_underd\*90+beta\_polio\*73+beta\_dipht\*94+beta\_gdp\*456+beta\_popl\*9080000)

## (Intercept)   
## 55.58289

(forecast3<-beta\_0+beta\_adultm\*60+beta\_infantd\*47+beta\_underd\*52+beta\_polio\*64+beta\_dipht\*45+beta\_gdp\*291+beta\_popl\*13768000)

## (Intercept)   
## 72.77058

W już końcowym etapie projektu dostrzegłem możliwość innego wprowadzenia danych. Użyłem sposobu zmiany danych w RStudio. Musiałem użyć nowego pakietu “tidyverse”, aby odczytało moje dane w formacie “csv”. Tak dostałem dane z prawidłowym odczytem kolumn, które w excelu nie działały, dzięki temu dostałem dostęp do danych, które bardzo mnie ciekawiły i chciałem użyć wcześniej. Postanowiłem nie usuwać poprzedniej pracy w celu sprawozdania pracy i dostania możliwych ciekawych wyników. Instalacja pakietu i uruchomienie go, aby użyć funkcji “read\_csv”

library(tidyverse)

## Warning: pakiet 'tidyverse' został zbudowany w wersji R 4.2.2

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──  
## ✔ tibble 3.1.8 ✔ dplyr 1.1.0  
## ✔ tidyr 1.2.1 ✔ stringr 1.5.0  
## ✔ readr 2.1.3 ✔ forcats 0.5.2  
## ✔ purrr 1.0.1

## Warning: pakiet 'tibble' został zbudowany w wersji R 4.2.2

## Warning: pakiet 'tidyr' został zbudowany w wersji R 4.2.2

## Warning: pakiet 'readr' został zbudowany w wersji R 4.2.2

## Warning: pakiet 'purrr' został zbudowany w wersji R 4.2.2

## Warning: pakiet 'dplyr' został zbudowany w wersji R 4.2.2

## Warning: pakiet 'stringr' został zbudowany w wersji R 4.2.2

## Warning: pakiet 'forcats' został zbudowany w wersji R 4.2.2

## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ✖ dplyr::recode() masks car::recode()  
## ✖ purrr::some() masks car::some()

Stworzenie nowego wektora z nową bazą danych, o nazwie “Dane\_nowe”

Dane\_nowe <- read\_csv("C:/Users/Kacper/Desktop/Projekt Regression/Dane z 2015.csv")

## Rows: 182 Columns: 22  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (2): Country, Status  
## dbl (20): Year, Life expectancy, Adult Mortality, infant deaths, Alcohol, pe...  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

View(Dane\_nowe)  
names(Dane\_nowe)

## [1] "Country" "Year"   
## [3] "Status" "Life expectancy"   
## [5] "Adult Mortality" "infant deaths"   
## [7] "Alcohol" "percentage expenditure"   
## [9] "Hepatitis B" "Measles"   
## [11] "BMI" "under-five deaths"   
## [13] "Polio" "Total expenditure"   
## [15] "Diphtheria" "HIV/AIDS"   
## [17] "GDP" "Population"   
## [19] "thinness 1-19 years" "thinness 5-9 years"   
## [21] "Income composition of resources" "Schooling"

Działania kodu: Ładuje bibliotekę “dplyr”, która zapewnia zestaw narzędzi do pracy z ramkami danych w R. Używa biblioteki “dplyr” do modyfikowania ramki danych “Dane\_nowe”. Funkcja “select” jest używana do wybierania kolumn z ramki danych, a argument “-c(Country, Year, Status, Alcohol, percentage expenditure, Total expenditure)” jest używany, aby wykluczyć określone kolumny z ramki danych. Rezultat jest przypisywany z powrotem do “Dane\_nowe” za pomocą operatora rury “%>%”. Wywołuje funkcję “View”, aby wyświetlić zawartość zmodyfikowanej ramki danych “Dane\_nowe” w formacie tabeli. Wywołuje funkcję “names”, aby wyświetlić nazwy kolumn w zmodyfikowanej ramce danych “Dane\_nowe”.

library("dplyr")  
Dane\_nowe <- Dane\_nowe %>% select(-c(Country, Year, Status, Alcohol, `percentage expenditure`, `Total expenditure`))  
View(Dane\_nowe)  
names(Dane\_nowe)

## [1] "Life expectancy" "Adult Mortality"   
## [3] "infant deaths" "Hepatitis B"   
## [5] "Measles" "BMI"   
## [7] "under-five deaths" "Polio"   
## [9] "Diphtheria" "HIV/AIDS"   
## [11] "GDP" "Population"   
## [13] "thinness 1-19 years" "thinness 5-9 years"   
## [15] "Income composition of resources" "Schooling"

Przydzielanie nowych nazw kolumną.

colnames(Dane\_nowe)<-c("Life","Adult\_M","Infant\_D","H\_B","Measles","BMI", "Under\_D","Polio","Dipht","H/A", "GDP","Popl", "t\_1-19", "t\_5-9", "Income", "Schooling")

Działania kodu: Tworzy nową kolumnę w ramce danych “Dane\_nowe” o nazwie “H\_B”. Używa funkcji “ifelse” do wypełnienia brakujących wartości (reprezentowanych przez “NA”) w kolumnie “H\_B”. Jeśli wartość w danej komórce kolumny “H\_B” jest “NA”, oblicza się średnią wszystkich niebrakujących wartości w kolumnie “H\_B” za pomocą funkcji “mean” z argumentem “na.rm = TRUE”, który usuwa brakujące wartości z obliczeń. Jeśli wartość w danej komórce kolumny “H\_B” nie jest “NA”, wartość pozostaje niezmieniona. Rezultat jest przypisywany z powrotem do kolumny “Dane\_nowe$H\_B”.

Dane\_nowe$H\_B = ifelse(is.na(Dane\_nowe$H\_B),  
 ave(Dane\_nowe$H\_B, FUN = function(x) mean(x, na.rm = TRUE)),  
 Dane\_nowe$H\_B)

Działania te są wykonywane na reszcie kolumn z brakującymi danymi.

Dane\_nowe$BMI = ifelse(is.na(Dane\_nowe$BMI ),  
 ave(Dane\_nowe$BMI , FUN = function(x) mean(x, na.rm = TRUE)),  
 Dane\_nowe$BMI )  
Dane\_nowe$GDP = ifelse(is.na(Dane\_nowe$GDP ),  
 ave(Dane\_nowe$GDP , FUN = function(x) mean(x, na.rm = TRUE)),  
 Dane\_nowe$GDP )  
Dane\_nowe$Popl = ifelse(is.na(Dane\_nowe$Popl ),  
 ave(Dane\_nowe$Popl , FUN = function(x) mean(x, na.rm = TRUE)),  
 Dane\_nowe$Popl )  
Dane\_nowe$`t\_1-19` = ifelse(is.na(Dane\_nowe$`t\_1-19` ),  
 ave(Dane\_nowe$`t\_1-19` , FUN = function(x) mean(x, na.rm = TRUE)),  
 Dane\_nowe$`t\_1-19` )  
Dane\_nowe$`t\_5-9` = ifelse(is.na(Dane\_nowe$`t\_5-9` ),  
 ave(Dane\_nowe$`t\_5-9` , FUN = function(x) mean(x, na.rm = TRUE)),  
 Dane\_nowe$`t\_5-9` )  
Dane\_nowe$Income = ifelse(is.na(Dane\_nowe$Income ),  
 ave(Dane\_nowe$Income , FUN = function(x) mean(x, na.rm = TRUE)),  
 Dane\_nowe$Income )  
Dane\_nowe$Schooling = ifelse(is.na(Dane\_nowe$Schooling ),  
 ave(Dane\_nowe$Schooling , FUN = function(x) mean(x, na.rm = TRUE)),  
 Dane\_nowe$Schooling )

Sprawdzanie współliniowości. Działanie kodu: Ten kod tworzy nową macierz “matrix\_nowy” z danymi “Dane\_nowe” bez pierwszej kolumny. Następnie oblicza macierz wartości własnych “matrix\_eigen\_nowy” jako iloczyn transponowanej macierzy “matrix\_nowy” i “matrix\_nowy”. W końcowym kroku wartości pierwszej wartości własnej dzielone są przez wszystkie wartości własne, a następnie wynik jest pierwiastkowany. Wyniki powyżej 30 pokazują brak współliniowości.

matrix\_nowy <- as.matrix(Dane\_nowe[,-1])  
  
matrix\_eigen\_nowy <- eigen(t(matrix\_nowy) %\*% matrix\_nowy)  
matrix\_eigen\_nowy$val

## [1] 1.406867e+17 2.824722e+10 1.119625e+10 6.451890e+06 1.254072e+06  
## [6] 7.715249e+05 6.910248e+04 4.464387e+04 7.941367e+03 5.960368e+03  
## [11] 3.273808e+03 1.132892e+03 1.812325e+02 7.168875e+01 9.788778e-01

sqrt(matrix\_eigen\_nowy$val[1]/matrix\_eigen\_nowy$val)

## [1] 1.000000e+00 2.231714e+03 3.544788e+03 1.476668e+05 3.349387e+05  
## [6] 4.270232e+05 1.426854e+06 1.775193e+06 4.208999e+06 4.858363e+06  
## [11] 6.555410e+06 1.114377e+07 2.786175e+07 4.429972e+07 3.791074e+08

Kod tworzy nowy model regresji liniowej “Model\_nowy” z zmienną objaśnianą “Life” i kilkoma zmiennymi objaśniającymi: “Adult\_M”, “Infant\_D”, “H\_B”, “Measles”, “BMI”, “Under\_D”, “Polio”, “Dipht”, “H/A”, “GDP”, “Popl”, “t\_1-19”, “t\_5-9”, “Income”, “Schooling”. Dane są wczytywane z data frame’u “Dane\_nowe”. W końcowym kroku jest wyświetlany podsumowanie modelu regresji liniowej “Model\_nowy”.

Model\_nowy <- lm(Life~Adult\_M+Infant\_D+H\_B+Measles+BMI+Under\_D+Polio+Dipht+`H/A`+GDP+Popl+`t\_1-19`+`t\_5-9`+Income+Schooling,data=Dane\_nowe)  
summary(Model\_nowy)

##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + H\_B + Measles + BMI +   
## Under\_D + Polio + Dipht + `H/A` + GDP + Popl + `t\_1-19` +   
## `t\_5-9` + Income + Schooling, data = Dane\_nowe)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -9.0242 -1.4732 0.0279 1.5076 7.6316   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 5.325e+01 2.112e+00 25.214 < 2e-16 \*\*\*  
## Adult\_M -2.512e-02 3.473e-03 -7.233 1.65e-11 \*\*\*  
## Infant\_D 9.068e-02 3.411e-02 2.659 0.00862 \*\*   
## H\_B 2.476e-02 2.359e-02 1.050 0.29526   
## Measles -4.671e-05 5.591e-05 -0.835 0.40471   
## BMI -2.990e-03 1.395e-02 -0.214 0.83062   
## Under\_D -6.832e-02 2.482e-02 -2.753 0.00656 \*\*   
## Polio 1.469e-02 1.332e-02 1.103 0.27170   
## Dipht 1.143e-02 2.743e-02 0.417 0.67733   
## `H/A` -4.438e-01 2.287e-01 -1.940 0.05402 .   
## GDP 5.588e-06 2.414e-05 0.231 0.81721   
## Popl 4.571e-09 1.029e-08 0.444 0.65730   
## `t\_1-19` -1.883e-01 2.496e-01 -0.754 0.45171   
## `t\_5-9` -2.071e-02 2.462e-01 -0.084 0.93307   
## Income 2.713e+01 4.619e+00 5.872 2.28e-08 \*\*\*  
## Schooling 4.810e-02 2.037e-01 0.236 0.81364   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.97 on 166 degrees of freedom  
## Multiple R-squared: 0.8746, Adjusted R-squared: 0.8633   
## F-statistic: 77.19 on 15 and 166 DF, p-value: < 2.2e-16

Użycie metody “backward”, aby ustalić najlepszy model.

Model\_nowy\_backward<-step(lm(Life~Adult\_M+Infant\_D+H\_B+Measles+BMI+Under\_D+Polio+Dipht+`H/A`+GDP+Popl+`t\_1-19`+`t\_5-9`+Income+Schooling,data=Dane\_nowe),direction="backward")

## Start: AIC=411.43  
## Life ~ Adult\_M + Infant\_D + H\_B + Measles + BMI + Under\_D + Polio +   
## Dipht + `H/A` + GDP + Popl + `t\_1-19` + `t\_5-9` + Income +   
## Schooling  
##   
## Df Sum of Sq RSS AIC  
## - `t\_5-9` 1 0.06 1463.8 409.44  
## - BMI 1 0.40 1464.2 409.48  
## - GDP 1 0.47 1464.3 409.49  
## - Schooling 1 0.49 1464.3 409.49  
## - Dipht 1 1.53 1465.3 409.62  
## - Popl 1 1.74 1465.5 409.65  
## - `t\_1-19` 1 5.02 1468.8 410.05  
## - Measles 1 6.15 1469.9 410.19  
## - H\_B 1 9.72 1473.5 410.63  
## - Polio 1 10.72 1474.5 410.76  
## <none> 1463.8 411.43  
## - `H/A` 1 33.20 1497.0 413.51  
## - Infant\_D 1 62.32 1526.1 417.02  
## - Under\_D 1 66.85 1530.6 417.56  
## - Income 1 304.08 1767.9 443.78  
## - Adult\_M 1 461.32 1925.1 459.29  
##   
## Step: AIC=409.44  
## Life ~ Adult\_M + Infant\_D + H\_B + Measles + BMI + Under\_D + Polio +   
## Dipht + `H/A` + GDP + Popl + `t\_1-19` + Income + Schooling  
##   
## Df Sum of Sq RSS AIC  
## - BMI 1 0.36 1464.2 407.48  
## - GDP 1 0.48 1464.3 407.50  
## - Schooling 1 0.48 1464.3 407.50  
## - Dipht 1 1.65 1465.5 407.64  
## - Popl 1 1.75 1465.6 407.65  
## - Measles 1 6.09 1469.9 408.19  
## - H\_B 1 9.72 1473.6 408.64  
## - Polio 1 10.68 1474.5 408.76  
## <none> 1463.8 409.44  
## - `H/A` 1 33.73 1497.6 411.58  
## - `t\_1-19` 1 61.02 1524.9 414.87  
## - Infant\_D 1 62.81 1526.7 415.08  
## - Under\_D 1 67.25 1531.1 415.61  
## - Income 1 304.19 1768.0 441.80  
## - Adult\_M 1 466.98 1930.8 457.83  
##   
## Step: AIC=407.48  
## Life ~ Adult\_M + Infant\_D + H\_B + Measles + Under\_D + Polio +   
## Dipht + `H/A` + GDP + Popl + `t\_1-19` + Income + Schooling  
##   
## Df Sum of Sq RSS AIC  
## - GDP 1 0.42 1464.6 405.53  
## - Schooling 1 0.46 1464.7 405.54  
## - Popl 1 1.76 1466.0 405.70  
## - Dipht 1 1.88 1466.1 405.72  
## - Measles 1 5.94 1470.2 406.22  
## - H\_B 1 9.39 1473.6 406.65  
## - Polio 1 10.94 1475.2 406.84  
## <none> 1464.2 407.48  
## - `H/A` 1 33.46 1497.7 409.59  
## - Infant\_D 1 62.45 1526.7 413.08  
## - `t\_1-19` 1 62.48 1526.7 413.09  
## - Under\_D 1 66.89 1531.1 413.61  
## - Income 1 308.71 1772.9 440.30  
## - Adult\_M 1 467.07 1931.3 455.87  
##   
## Step: AIC=405.53  
## Life ~ Adult\_M + Infant\_D + H\_B + Measles + Under\_D + Polio +   
## Dipht + `H/A` + Popl + `t\_1-19` + Income + Schooling  
##   
## Df Sum of Sq RSS AIC  
## - Schooling 1 0.38 1465.0 403.58  
## - Popl 1 1.87 1466.5 403.77  
## - Dipht 1 1.98 1466.6 403.78  
## - Measles 1 5.95 1470.6 404.27  
## - H\_B 1 9.21 1473.8 404.67  
## - Polio 1 10.96 1475.6 404.89  
## <none> 1464.6 405.53  
## - `H/A` 1 33.11 1497.7 407.60  
## - `t\_1-19` 1 62.11 1526.8 411.09  
## - Infant\_D 1 62.21 1526.8 411.10  
## - Under\_D 1 66.67 1531.3 411.64  
## - Income 1 342.80 1807.4 441.81  
## - Adult\_M 1 467.28 1931.9 453.93  
##   
## Step: AIC=403.58  
## Life ~ Adult\_M + Infant\_D + H\_B + Measles + Under\_D + Polio +   
## Dipht + `H/A` + Popl + `t\_1-19` + Income  
##   
## Df Sum of Sq RSS AIC  
## - Popl 1 1.89 1466.9 401.82  
## - Dipht 1 1.94 1467.0 401.82  
## - Measles 1 5.77 1470.8 402.30  
## - H\_B 1 9.64 1474.7 402.78  
## - Polio 1 10.73 1475.7 402.91  
## <none> 1465.0 403.58  
## - `H/A` 1 32.77 1497.8 405.61  
## - Infant\_D 1 62.04 1527.0 409.13  
## - `t\_1-19` 1 62.90 1527.9 409.23  
## - Under\_D 1 66.57 1531.6 409.67  
## - Adult\_M 1 471.29 1936.3 452.34  
## - Income 1 1276.02 2741.0 515.60  
##   
## Step: AIC=401.82  
## Life ~ Adult\_M + Infant\_D + H\_B + Measles + Under\_D + Polio +   
## Dipht + `H/A` + `t\_1-19` + Income  
##   
## Df Sum of Sq RSS AIC  
## - Dipht 1 2.09 1469.0 400.08  
## - Measles 1 5.73 1472.6 400.52  
## - Polio 1 9.05 1476.0 400.94  
## - H\_B 1 10.03 1476.9 401.06  
## <none> 1466.9 401.82  
## - `H/A` 1 33.82 1500.7 403.96  
## - Infant\_D 1 61.21 1528.1 407.26  
## - `t\_1-19` 1 65.55 1532.5 407.77  
## - Under\_D 1 67.29 1534.2 407.98  
## - Adult\_M 1 472.28 1939.2 450.61  
## - Income 1 1303.36 2770.3 515.53  
##   
## Step: AIC=400.08  
## Life ~ Adult\_M + Infant\_D + H\_B + Measles + Under\_D + Polio +   
## `H/A` + `t\_1-19` + Income  
##   
## Df Sum of Sq RSS AIC  
## - Measles 1 5.55 1474.5 398.76  
## - Polio 1 12.40 1481.4 399.60  
## <none> 1469.0 400.08  
## - `H/A` 1 35.09 1504.1 402.37  
## - Infant\_D 1 60.74 1529.7 405.45  
## - `t\_1-19` 1 64.18 1533.2 405.86  
## - Under\_D 1 66.95 1535.9 406.19  
## - H\_B 1 71.09 1540.1 406.68  
## - Adult\_M 1 473.15 1942.1 448.89  
## - Income 1 1380.67 2849.7 518.67  
##   
## Step: AIC=398.76  
## Life ~ Adult\_M + Infant\_D + H\_B + Under\_D + Polio + `H/A` + `t\_1-19` +   
## Income  
##   
## Df Sum of Sq RSS AIC  
## - Polio 1 12.53 1487.1 398.30  
## <none> 1474.5 398.76  
## - `H/A` 1 34.90 1509.4 401.02  
## - `t\_1-19` 1 58.91 1533.5 403.89  
## - Infant\_D 1 62.70 1537.2 404.34  
## - Under\_D 1 64.84 1539.4 404.59  
## - H\_B 1 68.51 1543.0 405.03  
## - Adult\_M 1 470.40 1944.9 447.15  
## - Income 1 1407.00 2881.5 518.70  
##   
## Step: AIC=398.3  
## Life ~ Adult\_M + Infant\_D + H\_B + Under\_D + `H/A` + `t\_1-19` +   
## Income  
##   
## Df Sum of Sq RSS AIC  
## <none> 1487.1 398.30  
## - `H/A` 1 37.51 1524.6 400.84  
## - `t\_1-19` 1 59.79 1546.9 403.48  
## - Infant\_D 1 63.46 1550.5 403.91  
## - Under\_D 1 65.73 1552.8 404.17  
## - H\_B 1 134.08 1621.2 412.01  
## - Adult\_M 1 483.96 1971.0 447.58  
## - Income 1 1514.79 3001.9 524.14

summary(Model\_nowy\_backward)

##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + H\_B + Under\_D + `H/A` +   
## `t\_1-19` + Income, data = Dane\_nowe)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -9.7571 -1.4736 -0.1144 1.6313 8.0334   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 53.331379 1.970747 27.062 < 2e-16 \*\*\*  
## Adult\_M -0.025318 0.003364 -7.525 2.71e-12 \*\*\*  
## Infant\_D 0.069914 0.025657 2.725 0.007088 \*\*   
## H\_B 0.040848 0.010313 3.961 0.000109 \*\*\*  
## Under\_D -0.054905 0.019797 -2.773 0.006154 \*\*   
## `H/A` -0.458897 0.219038 -2.095 0.037614 \*   
## `t\_1-19` -0.192444 0.072761 -2.645 0.008919 \*\*   
## Income 29.053282 2.182283 13.313 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.923 on 174 degrees of freedom  
## Multiple R-squared: 0.8726, Adjusted R-squared: 0.8675   
## F-statistic: 170.3 on 7 and 174 DF, p-value: < 2.2e-16

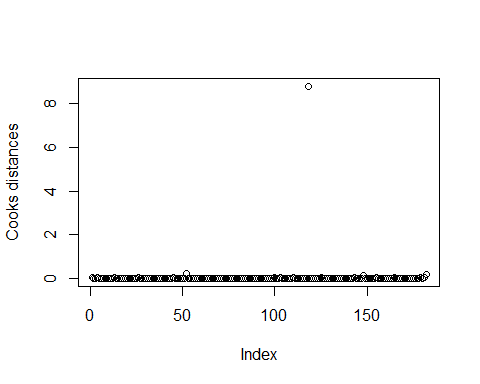
Finalny wynik jest bardzo optymistyczny, ponieważ wynik R-squared wynosi 0.8726 (zmienność oczekiwanej długości życia jest wyjaśniony w 87% przez model), a “Adjusted R-squared” wynosi 0.8675.

Model\_nowy <- Model\_nowy\_backward  
summary(Model\_nowy)

##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + H\_B + Under\_D + `H/A` +   
## `t\_1-19` + Income, data = Dane\_nowe)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -9.7571 -1.4736 -0.1144 1.6313 8.0334   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 53.331379 1.970747 27.062 < 2e-16 \*\*\*  
## Adult\_M -0.025318 0.003364 -7.525 2.71e-12 \*\*\*  
## Infant\_D 0.069914 0.025657 2.725 0.007088 \*\*   
## H\_B 0.040848 0.010313 3.961 0.000109 \*\*\*  
## Under\_D -0.054905 0.019797 -2.773 0.006154 \*\*   
## `H/A` -0.458897 0.219038 -2.095 0.037614 \*   
## `t\_1-19` -0.192444 0.072761 -2.645 0.008919 \*\*   
## Income 29.053282 2.182283 13.313 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.923 on 174 degrees of freedom  
## Multiple R-squared: 0.8726, Adjusted R-squared: 0.8675   
## F-statistic: 170.3 on 7 and 174 DF, p-value: < 2.2e-16

Sprawdzenie outlierów oraz stworzenie nowego modelu z usunietymi wartościami odstającymi.

cooks\_dis <- cooks.distance(Model\_nowy)  
plot(cooks\_dis,ylab="Cooks distances")



Model\_nowy\_no\_outliers <-lm(Life~Adult\_M+Infant\_D+H\_B+Under\_D+`H/A`+`t\_1-19`+Income,data=Dane\_nowe,subset=(cooks\_dis<max(cooks\_dis)))

summary(Model\_nowy\_no\_outliers)

##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + H\_B + Under\_D + `H/A` +   
## `t\_1-19` + Income, data = Dane\_nowe, subset = (cooks\_dis <   
## max(cooks\_dis)))  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -8.7930 -1.5945 -0.0056 1.5528 8.2047   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 55.166752 1.991738 27.698 < 2e-16 \*\*\*  
## Adult\_M -0.024734 0.003274 -7.554 2.33e-12 \*\*\*  
## Infant\_D 0.243635 0.057474 4.239 3.65e-05 \*\*\*  
## H\_B 0.038465 0.010047 3.829 0.00018 \*\*\*  
## Under\_D -0.198851 0.047024 -4.229 3.80e-05 \*\*\*  
## `H/A` -0.485941 0.213008 -2.281 0.02375 \*   
## `t\_1-19` -0.185696 0.070736 -2.625 0.00943 \*\*   
## Income 26.920332 2.213949 12.159 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.841 on 173 degrees of freedom  
## Multiple R-squared: 0.8773, Adjusted R-squared: 0.8723   
## F-statistic: 176.6 on 7 and 173 DF, p-value: < 2.2e-16

summary(Model\_nowy)

##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + H\_B + Under\_D + `H/A` +   
## `t\_1-19` + Income, data = Dane\_nowe)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -9.7571 -1.4736 -0.1144 1.6313 8.0334   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 53.331379 1.970747 27.062 < 2e-16 \*\*\*  
## Adult\_M -0.025318 0.003364 -7.525 2.71e-12 \*\*\*  
## Infant\_D 0.069914 0.025657 2.725 0.007088 \*\*   
## H\_B 0.040848 0.010313 3.961 0.000109 \*\*\*  
## Under\_D -0.054905 0.019797 -2.773 0.006154 \*\*   
## `H/A` -0.458897 0.219038 -2.095 0.037614 \*   
## `t\_1-19` -0.192444 0.072761 -2.645 0.008919 \*\*   
## Income 29.053282 2.182283 13.313 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.923 on 174 degrees of freedom  
## Multiple R-squared: 0.8726, Adjusted R-squared: 0.8675   
## F-statistic: 170.3 on 7 and 174 DF, p-value: < 2.2e-16

Sprawdzenie czy rozkład jest normalny za pomocą Testu Shapiro. W przypadku “Model\_nowy” model nie ma rozkładu normalnego. Natomiast w modelu “Model\_nowy\_no\_outliers” p-value wynosi ponad 0.05, oznacza to, że istnieje rozkład normalny.

shapiro.test(Model\_nowy$res)

##   
## Shapiro-Wilk normality test  
##   
## data: Model\_nowy$res  
## W = 0.98429, p-value = 0.03897

shapiro.test(Model\_nowy\_no\_outliers$res)

##   
## Shapiro-Wilk normality test  
##   
## data: Model\_nowy\_no\_outliers$res  
## W = 0.98966, p-value = 0.2134

Na końcu porównałem moje wszystkie modele

“Model\_nowy” ma lepszy “R-squared” niż “Model” - 0.72 < 0.87. “Model\_nowy” ma lepszy “Adjusted R-squared” niż “Model” - 0.71 < 0.86. “Model\_nowy” ma lepsze zmienne o wartości “p-value” mniejszych niż 0.05, niż “Model”: \* “Model” trzy wartości z przedziału 0 – 0,001 \* “Model\_nowy” cztery wartości z przedziału 0 – 0,001 \* “Model” trzy wartości z przedziału 0,001 – 0,01 \* “Model\_nowy” trzy wartości z przedziału 0,001 – 0,01 \* “Model” jedną wartość z przedziału 0,01 – 0,05 \* “Model\_nowy” jedną wartość z przedziału 0,01 – 0,05 \* “Model” jedną wartość z przedziału 0,05 – 0,1 \* “Model\_nowy” zero wartości z przedziału 0,05 – 0,1

summary(Model)

##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + Under\_D + Polio + Dipht +   
## GDP + Popl, data = Dane)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -16.4750 -2.5123 0.3676 2.3051 11.8873   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.935e+01 1.742e+00 39.819 < 2e-16 \*\*\*  
## Adult\_M -4.730e-02 3.887e-03 -12.169 < 2e-16 \*\*\*  
## Infant\_D 9.561e-02 3.737e-02 2.558 0.011365 \*   
## Under\_D -8.206e-02 2.955e-02 -2.777 0.006095 \*\*   
## Polio 5.021e-02 1.872e-02 2.683 0.008003 \*\*   
## Dipht 5.425e-02 1.982e-02 2.737 0.006843 \*\*   
## GDP 1.211e-04 3.192e-05 3.795 0.000203 \*\*\*  
## Popl 2.472e-08 1.439e-08 1.718 0.087513 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 4.312 on 174 degrees of freedom  
## Multiple R-squared: 0.7229, Adjusted R-squared: 0.7118   
## F-statistic: 64.85 on 7 and 174 DF, p-value: < 2.2e-16

summary(Model\_nowy)

##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + H\_B + Under\_D + `H/A` +   
## `t\_1-19` + Income, data = Dane\_nowe)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -9.7571 -1.4736 -0.1144 1.6313 8.0334   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 53.331379 1.970747 27.062 < 2e-16 \*\*\*  
## Adult\_M -0.025318 0.003364 -7.525 2.71e-12 \*\*\*  
## Infant\_D 0.069914 0.025657 2.725 0.007088 \*\*   
## H\_B 0.040848 0.010313 3.961 0.000109 \*\*\*  
## Under\_D -0.054905 0.019797 -2.773 0.006154 \*\*   
## `H/A` -0.458897 0.219038 -2.095 0.037614 \*   
## `t\_1-19` -0.192444 0.072761 -2.645 0.008919 \*\*   
## Income 29.053282 2.182283 13.313 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.923 on 174 degrees of freedom  
## Multiple R-squared: 0.8726, Adjusted R-squared: 0.8675   
## F-statistic: 170.3 on 7 and 174 DF, p-value: < 2.2e-16

“Model\_nowy\_no\_outliers” ma lepszy “R-squared” niż “Model\_no\_outliers” - 0.73 < 0.87. “Model\_nowy\_no\_outliers” ma lepszy “Adjusted R-squared” niż “Model\_no\_outliers” - 0.72 < 0.87. “Model\_nowy\_no\_outliers” ma lepsze zmienne o wartości “p-value” mniejszych niż 0.05, niż “Model\_no\_outliers”: \* “Model\_no\_outliers” pięć wartości z przedziału 0 – 0,001 \* “Model\_nowy\_no\_outliers” sześć wartości z przedziału 0 – 0,001 \* “Model\_no\_outliers” jedną wartość z przedziału 0,001 – 0,01 \* “Model\_nowy\_no\_outliers” jedną wartość z przedziału 0,001 – 0,01 \* “Model\_no\_outliers” jedną wartość z przedziału 0,01 – 0,05 \* “Model\_nowy\_no\_outliers” jedną wartość z przedziału 0,01 – 0,05 \* “Model\_no\_outliers” zero wartości z przedziału 0,05 – 0,1 \* “Model\_nowy\_no\_outliers” zero wartości z przedziału 0,05 – 0,1 \* “Model\_no\_outliers” ma jedną wartość z przedziału 0,1 – 1

summary(Model\_no\_outliers)

##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + Under\_D + Polio + Dipht +   
## GDP + Popl, data = Dane, subset = (cooks\_dis < max(cooks\_dis)))  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -14.2143 -2.3349 0.1912 2.2708 11.9048   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 7.042e+01 1.714e+00 41.099 < 2e-16 \*\*\*  
## Adult\_M -4.508e-02 3.816e-03 -11.813 < 2e-16 \*\*\*  
## Infant\_D 2.787e-01 6.307e-02 4.418 1.75e-05 \*\*\*  
## Under\_D -2.341e-01 5.158e-02 -4.539 1.05e-05 \*\*\*  
## Polio 4.282e-02 1.824e-02 2.347 0.020060 \*   
## Dipht 5.017e-02 1.923e-02 2.609 0.009875 \*\*   
## GDP 1.194e-04 3.091e-05 3.863 0.000158 \*\*\*  
## Popl 1.072e-08 1.448e-08 0.740 0.460100   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 4.175 on 173 degrees of freedom  
## Multiple R-squared: 0.7349, Adjusted R-squared: 0.7242   
## F-statistic: 68.51 on 7 and 173 DF, p-value: < 2.2e-16

summary(Model\_nowy\_no\_outliers)

##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + H\_B + Under\_D + `H/A` +   
## `t\_1-19` + Income, data = Dane\_nowe, subset = (cooks\_dis <   
## max(cooks\_dis)))  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -8.7930 -1.5945 -0.0056 1.5528 8.2047   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 55.166752 1.991738 27.698 < 2e-16 \*\*\*  
## Adult\_M -0.024734 0.003274 -7.554 2.33e-12 \*\*\*  
## Infant\_D 0.243635 0.057474 4.239 3.65e-05 \*\*\*  
## H\_B 0.038465 0.010047 3.829 0.00018 \*\*\*  
## Under\_D -0.198851 0.047024 -4.229 3.80e-05 \*\*\*  
## `H/A` -0.485941 0.213008 -2.281 0.02375 \*   
## `t\_1-19` -0.185696 0.070736 -2.625 0.00943 \*\*   
## Income 26.920332 2.213949 12.159 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.841 on 173 degrees of freedom  
## Multiple R-squared: 0.8773, Adjusted R-squared: 0.8723   
## F-statistic: 176.6 on 7 and 173 DF, p-value: < 2.2e-16

W tym etapie z ciekawości użyłem metody “Machine Learning”, aby zobaczyć jakie wyniki mogę uzyskać działając na moich danych. Dokonałem podziału danych na część “test”, którą będę testować na podstawie części “training”. Podział jest 80% dla “training” i 20% dla “test” z całości danych. Zainstalowałem pakiet “caTools”, oraz go uruchomiłem za pomocą funkcji “library”. Następnie uruchomienie funkcji “set.seed”, która odpowiada za generowanie losowych liczb. Funkcja “sample.split”, która podzieliła moją bazą danych według ustalonego wcześniejszego podziału. Kod przedstawia: \* Pierwsza linia - ładowanie biblioteki caTools \* Druga linia - ustawianie seeda losowego na wartość 123 \* Trzecia linia - dzielenie danych na zbiór treningowy i testowy z proporcją 80/20 \* Czwarta linia - tworzenie zbioru treningowego na podstawie danych i dzielenia \* Piąta linia - tworzenie zbioru testowego na podstawie danych i dzielenia

library(caTools)

## Warning: pakiet 'caTools' został zbudowany w wersji R 4.2.2

set.seed(123)  
split = sample.split(Dane$Life, SplitRatio = 0.8)  
training\_set = subset(Dane, split == TRUE)  
test\_set = subset(Dane, split == FALSE)

Stworzenie modelu składającego z bazy danych “training\_set”. Kod przedstawia: \* Pierwsza linia - tworzenie modelu regresji liniowej na zbiorze treningowym, gdzie zmienna objaśniana jest Life a pozostałe zmienne są uwzględniane jako zmienne objaśniające \* Druga linia - wyświetlenie podsumowania modelu

regressor = lm(formula = Life ~ .,  
 data = training\_set)  
  
summary(regressor)

##   
## Call:  
## lm(formula = Life ~ ., data = training\_set)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -12.8832 -2.3786 0.2148 2.2319 11.7683   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 7.095e+01 2.035e+00 34.871 < 2e-16 \*\*\*  
## Adult\_M -4.333e-02 4.427e-03 -9.788 < 2e-16 \*\*\*  
## Infant\_D 3.582e-01 8.639e-02 4.146 5.96e-05 \*\*\*  
## H\_B -2.775e-03 4.473e-02 -0.062 0.95064   
## Measles 4.439e-05 9.019e-05 0.492 0.62341   
## Under\_D -3.029e-01 6.925e-02 -4.373 2.44e-05 \*\*\*  
## Polio 2.729e-02 2.103e-02 1.297 0.19670   
## Dipht 6.012e-02 4.980e-02 1.207 0.22947   
## GDP 1.214e-04 3.811e-05 3.186 0.00179 \*\*   
## Popl 3.350e-09 1.539e-08 0.218 0.82795   
## Income 8.854e-02 4.126e-02 2.146 0.03367 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 4.311 on 134 degrees of freedom  
## Multiple R-squared: 0.7194, Adjusted R-squared: 0.6985   
## F-statistic: 34.36 on 10 and 134 DF, p-value: < 2.2e-16

Stworzyłem przewidziane y, dla “regressor” Kod przedstawia: \* Pierwsza linia - prognozowanie wartości Life na zbiorze testowym \* Druga linia - wyświetlenie prognozowanych wartości

y\_pred = predict(regressor, newdata = test\_set)  
y\_pred

## 1 2 3 4 5 6 7 8   
## 76.670226 54.765098 80.158901 83.354213 73.581444 76.470558 72.056293 74.566890   
## 9 10 11 12 13 14 15 16   
## 76.538567 81.451813 56.818147 77.422626 73.219638 68.548345 64.410685 73.990682   
## 17 18 19 20 21 22 23 24   
## 60.204301 66.129969 69.850911 79.137619 73.871221 70.555095 74.053635 70.754243   
## 25 26 27 28 29 30 31 32   
## 71.222304 68.897333 82.096634 8.478208 63.420292 74.277451 71.569561 75.935574   
## 33 34 35 36 37   
## 79.217138 80.146481 62.553392 74.047578 68.050284

Stworzenie “training\_set\_nowy” i “test\_set\_nowy” na podstawie “Dane\_nowe”, według tych samych zasad.

split\_nowy = sample.split(Dane\_nowe$Life, SplitRatio = 0.8)  
training\_set\_nowy = subset(Dane\_nowe, split == TRUE)  
test\_set\_nowy = subset(Dane\_nowe, split == FALSE)

Stworzenie modelu regresji liniowej “regressor\_nowy” na zbiorze treningowym “training\_set\_nowy”, gdzie zmienna objaśniana jest Life a pozostałe zmienne są uwzględniane jako zmienne objaśniające

regressor\_nowy = lm(formula = Life ~ .,  
 data = training\_set\_nowy)

Porównałem dwa modele. “regressor\_nowy” ma lepszy “R-squared” niż “regressor” - 0.71 < 0.87. “regressor\_nowy” ma lepszy “Adjusted R-squared” niż “regressor” - 0.69 < 0.86.

summary(regressor\_nowy)

##   
## Call:  
## lm(formula = Life ~ ., data = training\_set\_nowy)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -9.1604 -1.3642 -0.1081 1.5677 7.6245   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 5.454e+01 2.398e+00 22.748 < 2e-16 \*\*\*  
## Adult\_M -2.577e-02 3.856e-03 -6.684 6.34e-10 \*\*\*  
## Infant\_D 1.996e-01 6.846e-02 2.916 0.00418 \*\*   
## H\_B 3.779e-02 3.070e-02 1.231 0.22056   
## Measles -3.437e-05 6.371e-05 -0.539 0.59053   
## BMI -1.023e-02 1.462e-02 -0.699 0.48554   
## Under\_D -1.599e-01 5.467e-02 -2.924 0.00408 \*\*   
## Polio 1.100e-04 1.446e-02 0.008 0.99394   
## Dipht 1.486e-03 3.413e-02 0.044 0.96534   
## `H/A` -3.577e-01 2.432e-01 -1.471 0.14378   
## GDP 6.981e-06 2.874e-05 0.243 0.80849   
## Popl -7.987e-09 1.065e-08 -0.750 0.45476   
## `t\_1-19` 7.336e-03 3.198e-01 0.023 0.98173   
## `t\_5-9` -1.954e-01 3.153e-01 -0.620 0.53647   
## Income 2.683e+01 5.226e+00 5.133 1.02e-06 \*\*\*  
## Schooling 8.662e-02 2.353e-01 0.368 0.71343   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.911 on 129 degrees of freedom  
## Multiple R-squared: 0.8768, Adjusted R-squared: 0.8625   
## F-statistic: 61.22 on 15 and 129 DF, p-value: < 2.2e-16

summary(regressor)

##   
## Call:  
## lm(formula = Life ~ ., data = training\_set)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -12.8832 -2.3786 0.2148 2.2319 11.7683   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 7.095e+01 2.035e+00 34.871 < 2e-16 \*\*\*  
## Adult\_M -4.333e-02 4.427e-03 -9.788 < 2e-16 \*\*\*  
## Infant\_D 3.582e-01 8.639e-02 4.146 5.96e-05 \*\*\*  
## H\_B -2.775e-03 4.473e-02 -0.062 0.95064   
## Measles 4.439e-05 9.019e-05 0.492 0.62341   
## Under\_D -3.029e-01 6.925e-02 -4.373 2.44e-05 \*\*\*  
## Polio 2.729e-02 2.103e-02 1.297 0.19670   
## Dipht 6.012e-02 4.980e-02 1.207 0.22947   
## GDP 1.214e-04 3.811e-05 3.186 0.00179 \*\*   
## Popl 3.350e-09 1.539e-08 0.218 0.82795   
## Income 8.854e-02 4.126e-02 2.146 0.03367 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 4.311 on 134 degrees of freedom  
## Multiple R-squared: 0.7194, Adjusted R-squared: 0.6985   
## F-statistic: 34.36 on 10 and 134 DF, p-value: < 2.2e-16

Prognozowanie wartości Life na zbiorze testowym “test\_set\_nowy” i modelu “regressor\_nowy”.

y\_pred\_nowy = predict(regressor\_nowy, newdata = test\_set\_nowy)  
y\_pred\_nowy

## 1 2 3 4 5 6 7 8   
## 77.35855 58.70143 79.16632 82.99941 75.57473 80.84683 72.02276 75.67873   
## 9 10 11 12 13 14 15 16   
## 79.42603 80.71335 52.37017 79.52112 72.87348 61.31732 66.19483 74.75147   
## 17 18 19 20 21 22 23 24   
## 58.20621 60.07876 69.03170 77.48792 71.30342 67.42520 75.22256 70.60603   
## 25 26 27 28 29 30 31 32   
## 72.00059 65.47484 82.47891 34.28289 66.12623 73.64672 72.81358 78.07606   
## 33 34 35 36 37   
## 80.81572 81.14668 66.82476 71.37072 66.73114

Porównanie “y\_pred” i “y\_pred\_nowy” z “Life”.

1. “y\_pred\_nowy” jest bardziej przybliżony, różnica 0,45
2. “y\_pred” jest bardziej przybliżony, różnica 2,37
3. “y\_pred\_nowy” jest bardziej przybliżony, różnica 2,77
4. “y\_pred\_nowy” jest bardziej przybliżony, różnica 0,19
5. “y\_pred\_nowy” jest bardziej przybliżony, różnica 0,53
6. “y\_pred\_nowy” jest bardziej przybliżony, różnica 0,25
7. “y\_pred” jest bardziej przybliżony, różnica 4,95
8. “y\_pred\_nowy” jest bardziej przybliżony, różnica 1,72
9. “y\_pred” jest bardziej przybliżony, różnica 1,16
10. “y\_pred” jest bardziej przybliżony, różnica 0,75
11. “y\_pred\_nowy” jest bardziej przybliżony, różnica 0,13
12. “y\_pred\_nowy” jest bardziej przybliżony, różnica 5,48
13. “y\_pred” jest bardziej przybliżony, różnica 5,78
14. “y\_pred\_nowy” jest bardziej przybliżony, różnica 3,38
15. “y\_pred\_nowy” jest bardziej przybliżony, różnica 0,19
16. “y\_pred” jest bardziej przybliżony, różnica 0,39
17. “y\_pred\_nowy” jest bardziej przybliżony, różnica 0,8
18. “y\_pred\_nowy” jest bardziej przybliżony, różnica 1,18
19. “y\_pred\_nowy” jest bardziej przybliżony, różnica 2,83
20. “y\_pred\_nowy” jest bardziej przybliżony, różnica 2,79
21. “y\_pred\_nowy” jest bardziej przybliżony, różnica 0,20
22. “y\_pred\_nowy” jest bardziej przybliżony, różnica 1,73
23. “y\_pred” jest bardziej przybliżony, różnica 0,55
24. “y\_pred\_nowy” jest bardziej przybliżony, różnica 1,2
25. “y\_pred” jest bardziej przybliżony, różnica 2,42
26. “y\_pred\_nowy” jest bardziej przybliżony, różnica 1,13
27. “y\_pred” jest bardziej przybliżony, różnica 0,20
28. “y\_pred\_nowy” jest bardziej przybliżony, różnica 20,22, anomalia
29. “y\_pred\_nowy” jest bardziej przybliżony, różnica 2,37
30. “y\_pred” jest bardziej przybliżony, różnica 0,72
31. “y\_pred\_nowy” jest bardziej przybliżony, różnica 1,19
32. “y\_pred” jest bardziej przybliżony, różnica 1,44
33. “y\_pred” jest bardziej przybliżony, różnica 2,52
34. “y\_pred\_nowy” jest bardziej przybliżony, różnica 1,65
35. “y\_pred” jest bardziej przybliżony, różnica 0,75
36. “y\_pred” jest bardziej przybliżony, różnica 1,95
37. “y\_pred\_nowy” jest bardziej przybliżony, różnica 4,93

“y\_pred\_nowy” miał lepsze przybliżenie wyników w 23/37. W punkcie 28 doszło do anomalii, a różnica wyniku przypadku “y\_pred\_nowy” okazało się oddalone od wartości ze zbioru testującej w 20,22, a “y\_pred” w 46,02.

y\_pred = predict(regressor, newdata = test\_set)  
y\_pred

## 1 2 3 4 5 6 7 8   
## 76.670226 54.765098 80.158901 83.354213 73.581444 76.470558 72.056293 74.566890   
## 9 10 11 12 13 14 15 16   
## 76.538567 81.451813 56.818147 77.422626 73.219638 68.548345 64.410685 73.990682   
## 17 18 19 20 21 22 23 24   
## 60.204301 66.129969 69.850911 79.137619 73.871221 70.555095 74.053635 70.754243   
## 25 26 27 28 29 30 31 32   
## 71.222304 68.897333 82.096634 8.478208 63.420292 74.277451 71.569561 75.935574   
## 33 34 35 36 37   
## 79.217138 80.146481 62.553392 74.047578 68.050284

y\_pred\_nowy = predict(regressor\_nowy, newdata = test\_set\_nowy)  
y\_pred\_nowy

## 1 2 3 4 5 6 7 8   
## 77.35855 58.70143 79.16632 82.99941 75.57473 80.84683 72.02276 75.67873   
## 9 10 11 12 13 14 15 16   
## 79.42603 80.71335 52.37017 79.52112 72.87348 61.31732 66.19483 74.75147   
## 17 18 19 20 21 22 23 24   
## 58.20621 60.07876 69.03170 77.48792 71.30342 67.42520 75.22256 70.60603   
## 25 26 27 28 29 30 31 32   
## 72.00059 65.47484 82.47891 34.28289 66.12623 73.64672 72.81358 78.07606   
## 33 34 35 36 37   
## 80.81572 81.14668 66.82476 71.37072 66.73114

# Rozdział 2 - Machine Learning

## Multiple Linear Regression

Ten kod w języku R ma na celu odczytanie pliku CSV z danymi z lokalizacji “C:Regression 2.0z 2015.csv” i przypisanie wynikowego zestawu danych do zmiennej dataset. Następnie kod wyświetla widok (tabelę) danych w przeglądarce.

Ostatecznie, funkcja names(dataset) zwraca wektor zawierający nazwy kolumn w tym zestawie danych.

dataset <- read\_csv("C:\\Users\\Kacper\\Desktop\\Projekt Regression 2.0\\Dane z 2015.csv")

## Rows: 182 Columns: 22  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (2): Country, Status  
## dbl (20): Year, Life expectancy, Adult Mortality, infant deaths, Alcohol, pe...  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

View(dataset)  
names(dataset)

## [1] "Country" "Year"   
## [3] "Status" "Life expectancy"   
## [5] "Adult Mortality" "infant deaths"   
## [7] "Alcohol" "percentage expenditure"   
## [9] "Hepatitis B" "Measles"   
## [11] "BMI" "under-five deaths"   
## [13] "Polio" "Total expenditure"   
## [15] "Diphtheria" "HIV/AIDS"   
## [17] "GDP" "Population"   
## [19] "thinness 1-19 years" "thinness 5-9 years"   
## [21] "Income composition of resources" "Schooling"

Ten kod wprowadza pakiet dplyr, a następnie zmienia zestaw danych dataset, tak że wybiera wszystkie kolumny z wyjątkiem tych, które zostaną wymienione po -c (czyli Country, Year, Status, Alcohol, percentage expenditure, i Total expenditure).

Następnie widok zmienionego zestawu danych jest wyświetlany w przeglądarce przy pomocy funkcji View(dataset).

Ostatecznie, funkcja names(dataset) zwraca wektor zawierający nazwy pozostałych kolumn w tym zestawie danych.

library("dplyr")  
dataset <- dataset %>% select(-c(Country, Year, Status, Alcohol,  
`percentage expenditure`, `Total expenditure`))  
View(dataset)  
names(dataset)

## [1] "Life expectancy" "Adult Mortality"   
## [3] "infant deaths" "Hepatitis B"   
## [5] "Measles" "BMI"   
## [7] "under-five deaths" "Polio"   
## [9] "Diphtheria" "HIV/AIDS"   
## [11] "GDP" "Population"   
## [13] "thinness 1-19 years" "thinness 5-9 years"   
## [15] "Income composition of resources" "Schooling"

Ten kod zmienia nazwy kolumn w zestawie danych dataset na podane wektorowo nazwy, zgodnie z kolejnością kolumn w pliku CSV.

Każda nowa nazwa kolumny jest przypisywana do wektora c(“Life”,“Adult\_M”,“Infant\_D”,“H\_B”,“Measles”,“BMI”, “Under\_D”,“Polio”,“Dipht”,“H/A”, “GDP”,“Popl”, “t\_1-19”, “t\_5-9”, “Income”, “Schooling”), a następnie przypisywana do dataset za pomocą funkcji colnames(), która zmienia nazwy kolumn.

Ostatecznie, funkcja names(dataset) zwraca wektor zawierający nowe nazwy kolumn w tym zestawie danych.

colnames(dataset)<-c("Life","Adult\_M","Infant\_D","H\_B","Measles","BMI",  
"Under\_D","Polio","Dipht","H/A", "GDP","Popl", "t\_1-19", "t\_5-9", "Income",  
"Schooling")  
names(dataset)

## [1] "Life" "Adult\_M" "Infant\_D" "H\_B" "Measles" "BMI"   
## [7] "Under\_D" "Polio" "Dipht" "H/A" "GDP" "Popl"   
## [13] "t\_1-19" "t\_5-9" "Income" "Schooling"

Ten kod wprowadza kilka zmian do zestawu danych dataset, aby zastąpić brakujące wartości średnią wartości w danej kolumnie. Konkretnie, używa funkcji ifelse() do sprawdzenia, czy wartość w danej komórce jest brakująca, a następnie zastępuje ją średnią wartości z danej kolumny, wykorzystując funkcję ave().

Poniżej znajdują się szczegółowe wyjaśnienia dla każdej kolumny:

* H\_B: jeśli wartość jest brakująca, to zastępuje ją średnią wartości z kolumny H\_B.
* BMI: jeśli wartość jest brakująca, to zastępuje ją średnią wartości z kolumny BMI.
* GDP: jeśli wartość jest brakująca, to zastępuje ją średnią wartości z kolumny GDP.
* Popl: jeśli wartość jest brakująca, to zastępuje ją średnią wartości z kolumny Popl.
* t\_1-19: jeśli wartość jest brakująca, to zastępuje ją średnią wartości z kolumny t\_1-19.
* t\_5-9: jeśli wartość jest brakująca, to zastępuje ją średnią wartości z kolumny t\_5-9.
* Income: jeśli wartość jest brakująca, to zastępuje ją średnią wartości z kolumny Income.
* Schooling: jeśli wartość jest brakująca, to zastępuje ją średnią wartości z kolumny Schooling.

Na koniec, kod wyświetla zmieniony zestaw danych w przeglądarce przy pomocy funkcji View(dataset).

dataset$H\_B = ifelse(is.na(dataset$H\_B),  
 ave(dataset$H\_B, FUN = function(x) mean(x, na.rm =  
TRUE)),  
 dataset$H\_B)  
  
dataset$BMI = ifelse(is.na(dataset$BMI ),  
 ave(dataset$BMI , FUN = function(x) mean(x, na.rm =  
TRUE)),  
 dataset$BMI )  
  
dataset$GDP = ifelse(is.na(dataset$GDP ),  
 ave(dataset$GDP , FUN = function(x) mean(x, na.rm =  
TRUE)),  
 dataset$GDP )  
  
dataset$Popl = ifelse(is.na(dataset$Popl ),  
 ave(dataset$Popl , FUN = function(x) mean(x, na.rm  
= TRUE)),  
dataset$Popl )  
  
dataset$`t\_1-19` = ifelse(is.na(dataset$`t\_1-19` ),  
 ave(dataset$`t\_1-19` , FUN = function(x) mean(x,  
na.rm = TRUE)),  
 dataset$`t\_1-19` )  
  
dataset$`t\_5-9` = ifelse(is.na(dataset$`t\_5-9` ),  
 ave(dataset$`t\_5-9` , FUN = function(x) mean(x,  
na.rm = TRUE)),  
 dataset$`t\_5-9` )  
  
dataset$Income = ifelse(is.na(dataset$Income ),  
 ave(dataset$Income , FUN = function(x)  
mean(x, na.rm = TRUE)),  
 dataset$Income )  
  
dataset$Schooling = ifelse(is.na(dataset$Schooling ),  
 ave(dataset$Schooling , FUN = function(x)  
mean(x, na.rm = TRUE)),  
 dataset$Schooling )  
  
View(dataset)

Ten kod dzieli zbiór danych dataset na zbiór treningowy i testowy w stosunku 80:20, wykorzystując funkcję sample.split() z pakietu caTools. W szczególności, wywołuje funkcję sample.split(dataset Life, SplitRatio = 0.8), która losowo dzieli dane na dwa zbiory zgodnie z daną proporcją. Argument dataset$Life oznacza, że wykorzystujemy kolumnę “Life” jako wektor klasyfikacyjny, który decyduje, do którego zbioru zostanie przypisany każdy wiersz.

Następnie wykorzystuje funkcję subset() do utworzenia dwóch podzbiorów danych:

training\_set: zbiór treningowy, który zawiera 80% losowo wybranych wierszy z dataset (tj. te, które mają wartość TRUE w wektorze split). test\_set: zbiór testowy, który zawiera pozostałe 20% wierszy (tj. te, które mają wartość FALSE w wektorze split). Ostatecznie, nazwy utworzonych podzbiorów danych nie są zmieniane, ale możesz je zmienić, jeśli chcesz, wykorzystując funkcję colnames().

library(caTools)  
set.seed(123)  
split = sample.split(dataset$Life, SplitRatio = 0.8)  
training\_set\_mult = subset(dataset, split == TRUE)  
test\_set\_mult = subset(dataset, split == FALSE)

Kod wyświetla zestaw training\_set w przeglądarce przy pomocy funkcji View().

View(training\_set\_mult )

Kod wyświetla zestaw test\_set w przeglądarce przy pomocy funkcji View().

View(test\_set\_mult )

Kod standaryzuje dane w training\_set za pomocą funkcji scale, a następnie otworzył przeglądarkę danych do wizualnej inspekcji wynikowych przeskalowanych danych.

training\_set\_mult = scale(training\_set\_mult)  
View(training\_set\_mult )

Kod standaryzuje dane w test\_set za pomocą funkcji scale, a następnie otworzył przeglądarkę danych do wizualnej inspekcji wynikowych przeskalowanych danych.

test\_set\_mult = scale(test\_set\_mult)  
View(test\_set\_mult )

Ten kod R służy do dopasowania modelu regresji liniowej do danych znajdujących się w training\_set i uzyskania podsumowania wyników.

Linia regressor = lm(formula = Life ~ ., data = as.data.frame(training\_set)) tworzy obiekt modelu regresji liniowej regressor, który jest dopasowany do danych znajdujących się w training\_set. W tym przypadku Life jest zmienną zależną, a . oznacza, że używane są wszystkie pozostałe kolumny jako zmienne niezależne.

mult\_reg = lm(formula = Life ~ .,  
 data = as.data.frame(training\_set\_mult))

Ten kod wykonuje regresję liniową między zmiennej odpowiedzi Life (wskaźnik oczekiwanej długości życia) a zestawem zmiennych objaśniających w ramce danych training\_set.

Interpretacja wyników:

* Współczynniki: estymator wskazuje, jak zmienia się wartość zmiennej odpowiedzi wraz ze zmianami jednostki zmiennej objaśniającej. Współczynniki z wartością p poniżej 0,05 są uważane za istotne, co oznacza, że istnieje istotny statystycznie związek między tą zmienną objaśniającą a zmienną odpowiedzi.
* W przypadku naszych danych Adult\_M (współczynnik -0,3161) oraz Income (współczynnik 0,5088) są silnie związane z długością życia.
* Residuals: Wartości te reprezentują różnice między rzeczywistymi wartościami zmiennej odpowiedzi (Life) a wartościami przewidywanymi przez model.
* Residual standard error: Mierzy, jak dokładnie model dopasowuje się do danych.
* Multiple R-squared: Wartość ta wskazuje, jak dobrze dopasowany jest model do danych. W przypadku naszych danych, współczynnik R kwadrat wynosi 0,8768, co oznacza, że 87,68% zmienności zmiennej odpowiedzi jest wyjaśnione przez zmienne objaśniające w naszym modelu.
* F-statistic: Test ten porównuje nasz model z modelem, który nie zawiera żadnych zmiennych objaśniających. Ma on duże znaczenie w statystycznej istotności naszego modelu i wynosi 61,22 z 15 i 129 stopniami swobody.
* p-value: Wartość p informuje nas o prawdopodobieństwie, że nie ma żadnego związku między zmiennymi objaśniającymi a zmienną odpowiedzi. Dla każdej zmiennej objaśniającej widzimy wartość p. Najmniejsze wartości p (<0,05) są uważane za istotne, co oznacza, że istnieje istotny statystycznie związek między tą zmienną objaśniającą a zmienną odpowiedzi.

summary(mult\_reg )

##   
## Call:  
## lm(formula = Life ~ ., data = as.data.frame(training\_set\_mult))  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.16693 -0.17378 -0.01378 0.19970 0.97128   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -2.025e-16 3.079e-02 0.000 1.00000   
## Adult\_M -3.161e-01 4.730e-02 -6.684 6.34e-10 \*\*\*  
## Infant\_D 2.163e+00 7.418e-01 2.916 0.00418 \*\*   
## H\_B 1.046e-01 8.494e-02 1.231 0.22056   
## Measles -3.715e-02 6.888e-02 -0.539 0.59053   
## BMI -2.717e-02 3.884e-02 -0.699 0.48554   
## Under\_D -2.121e+00 7.253e-01 -2.924 0.00408 \*\*   
## Polio 3.437e-04 4.518e-02 0.008 0.99394   
## Dipht 3.977e-03 9.135e-02 0.044 0.96534   
## `H/A` -6.256e-02 4.254e-02 -1.471 0.14378   
## GDP 8.917e-03 3.671e-02 0.243 0.80849   
## Popl -2.492e-02 3.323e-02 -0.750 0.45476   
## `t\_1-19` 3.922e-03 1.709e-01 0.023 0.98173   
## `t\_5-9` -1.066e-01 1.720e-01 -0.620 0.53647   
## Income 5.088e-01 9.913e-02 5.133 1.02e-06 \*\*\*  
## Schooling 3.056e-02 8.304e-02 0.368 0.71343   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3708 on 129 degrees of freedom  
## Multiple R-squared: 0.8768, Adjusted R-squared: 0.8625   
## F-statistic: 61.22 on 15 and 129 DF, p-value: < 2.2e-16

Ten kod wykonuje kierunkową selekcję zmiennych z wykorzystaniem algorytmu schodkowego (“stepwise”) i kierunkiem “backward”. W każdej iteracji algorytmu, zmienna, która ma najwyższą wartość p-wartości (najmniej istotną statystycznie), zostaje usunięta z modelu. Proces ten jest powtarzany aż do momentu, gdy żadna zmienna nie zostanie usunięta lub usunięcie jednej zmiennej nie zwiększy znacząco jakości modelu.

W wyniku tego kodu zostanie wybrany najlepszy model regresji liniowej z zmiennymi wybrane w procesie selekcji z kierunkiem “backward”. Model ten będzie miał najmniejszą liczbę zmiennych, które są istotne statystycznie w celu wyjaśnienia zmienności zmiennej zależnej “Life”.

mult\_reg <-   
step(lm(Life ~ .,  
 data=as.data.frame(training\_set\_mult)),direction="backward")

## Start: AIC=-272.65  
## Life ~ Adult\_M + Infant\_D + H\_B + Measles + BMI + Under\_D + Polio +   
## Dipht + `H/A` + GDP + Popl + `t\_1-19` + `t\_5-9` + Income +   
## Schooling  
##   
## Df Sum of Sq RSS AIC  
## - Polio 1 0.0000 17.737 -274.65  
## - `t\_1-19` 1 0.0001 17.738 -274.65  
## - Dipht 1 0.0003 17.738 -274.65  
## - GDP 1 0.0081 17.746 -274.59  
## - Schooling 1 0.0186 17.756 -274.50  
## - Measles 1 0.0400 17.777 -274.33  
## - `t\_5-9` 1 0.0528 17.790 -274.22  
## - BMI 1 0.0673 17.805 -274.10  
## - Popl 1 0.0773 17.815 -274.02  
## - H\_B 1 0.2084 17.946 -272.96  
## <none> 17.737 -272.65  
## - `H/A` 1 0.2974 18.035 -272.24  
## - Infant\_D 1 1.1691 18.907 -265.40  
## - Under\_D 1 1.1757 18.913 -265.35  
## - Income 1 3.6227 21.360 -247.71  
## - Adult\_M 1 6.1421 23.880 -231.54  
##   
## Step: AIC=-274.65  
## Life ~ Adult\_M + Infant\_D + H\_B + Measles + BMI + Under\_D + Dipht +   
## `H/A` + GDP + Popl + `t\_1-19` + `t\_5-9` + Income + Schooling  
##   
## Df Sum of Sq RSS AIC  
## - `t\_1-19` 1 0.0001 17.738 -276.65  
## - Dipht 1 0.0003 17.738 -276.65  
## - GDP 1 0.0081 17.746 -276.59  
## - Schooling 1 0.0186 17.756 -276.50  
## - Measles 1 0.0401 17.778 -276.32  
## - `t\_5-9` 1 0.0529 17.790 -276.22  
## - BMI 1 0.0674 17.805 -276.10  
## - Popl 1 0.0872 17.825 -275.94  
## - H\_B 1 0.2089 17.946 -274.95  
## <none> 17.737 -274.65  
## - `H/A` 1 0.2975 18.035 -274.24  
## - Infant\_D 1 1.1702 18.908 -267.39  
## - Under\_D 1 1.1775 18.915 -267.33  
## - Income 1 3.6345 21.372 -249.62  
## - Adult\_M 1 6.2399 23.977 -232.95  
##   
## Step: AIC=-276.65  
## Life ~ Adult\_M + Infant\_D + H\_B + Measles + BMI + Under\_D + Dipht +   
## `H/A` + GDP + Popl + `t\_5-9` + Income + Schooling  
##   
## Df Sum of Sq RSS AIC  
## - Dipht 1 0.0003 17.738 -278.65  
## - GDP 1 0.0081 17.746 -278.59  
## - Schooling 1 0.0186 17.756 -278.50  
## - Measles 1 0.0401 17.778 -278.32  
## - BMI 1 0.0674 17.805 -278.10  
## - Popl 1 0.0871 17.825 -277.94  
## - H\_B 1 0.2092 17.947 -276.95  
## <none> 17.738 -276.65  
## - `H/A` 1 0.2987 18.036 -276.23  
## - `t\_5-9` 1 0.7031 18.441 -273.02  
## - Infant\_D 1 1.1779 18.915 -269.33  
## - Under\_D 1 1.1863 18.924 -269.26  
## - Income 1 3.6415 21.379 -251.58  
## - Adult\_M 1 6.2497 23.987 -234.89  
##   
## Step: AIC=-278.65  
## Life ~ Adult\_M + Infant\_D + H\_B + Measles + BMI + Under\_D + `H/A` +   
## GDP + Popl + `t\_5-9` + Income + Schooling  
##   
## Df Sum of Sq RSS AIC  
## - GDP 1 0.0083 17.746 -280.58  
## - Schooling 1 0.0185 17.756 -280.50  
## - Measles 1 0.0400 17.778 -280.32  
## - BMI 1 0.0680 17.806 -280.10  
## - Popl 1 0.0881 17.826 -279.93  
## <none> 17.738 -278.65  
## - `H/A` 1 0.2988 18.037 -278.23  
## - `t\_5-9` 1 0.7031 18.441 -275.01  
## - Infant\_D 1 1.1777 18.916 -271.33  
## - Under\_D 1 1.1861 18.924 -271.26  
## - H\_B 1 1.3357 19.074 -270.12  
## - Income 1 3.7121 21.450 -253.10  
## - Adult\_M 1 6.3262 24.064 -236.42  
##   
## Step: AIC=-280.58  
## Life ~ Adult\_M + Infant\_D + H\_B + Measles + BMI + Under\_D + `H/A` +   
## Popl + `t\_5-9` + Income + Schooling  
##   
## Df Sum of Sq RSS AIC  
## - Schooling 1 0.0139 17.760 -282.47  
## - Measles 1 0.0391 17.785 -282.26  
## - BMI 1 0.0650 17.811 -282.05  
## - Popl 1 0.0850 17.831 -281.89  
## <none> 17.746 -280.58  
## - `H/A` 1 0.2936 18.040 -280.20  
## - `t\_5-9` 1 0.6948 18.441 -277.01  
## - Infant\_D 1 1.1702 18.916 -273.32  
## - Under\_D 1 1.1783 18.924 -273.26  
## - H\_B 1 1.3458 19.092 -271.98  
## - Income 1 4.5507 22.297 -249.48  
## - Adult\_M 1 6.3214 24.068 -238.40  
##   
## Step: AIC=-282.47  
## Life ~ Adult\_M + Infant\_D + H\_B + Measles + BMI + Under\_D + `H/A` +   
## Popl + `t\_5-9` + Income  
##   
## Df Sum of Sq RSS AIC  
## - Measles 1 0.0364 17.796 -284.17  
## - BMI 1 0.0628 17.823 -283.96  
## - Popl 1 0.0837 17.844 -283.79  
## <none> 17.760 -282.47  
## - `H/A` 1 0.2931 18.053 -282.10  
## - `t\_5-9` 1 0.6974 18.457 -278.88  
## - Infant\_D 1 1.1816 18.942 -275.13  
## - Under\_D 1 1.1932 18.953 -275.04  
## - H\_B 1 1.4108 19.171 -273.38  
## - Adult\_M 1 6.4562 24.216 -239.51  
## - Income 1 15.9446 33.705 -191.57  
##   
## Step: AIC=-284.17  
## Life ~ Adult\_M + Infant\_D + H\_B + BMI + Under\_D + `H/A` + Popl +   
## `t\_5-9` + Income  
##   
## Df Sum of Sq RSS AIC  
## - BMI 1 0.0545 17.851 -285.73  
## - Popl 1 0.0853 17.882 -285.48  
## <none> 17.796 -284.17  
## - `H/A` 1 0.2960 18.092 -283.78  
## - `t\_5-9` 1 0.6619 18.458 -280.88  
## - Under\_D 1 1.1614 18.958 -277.00  
## - Infant\_D 1 1.1724 18.969 -276.92  
## - H\_B 1 1.3962 19.193 -275.22  
## - Adult\_M 1 6.4199 24.216 -241.51  
## - Income 1 16.0854 33.882 -192.81  
##   
## Step: AIC=-285.73  
## Life ~ Adult\_M + Infant\_D + H\_B + Under\_D + `H/A` + Popl + `t\_5-9` +   
## Income  
##   
## Df Sum of Sq RSS AIC  
## - Popl 1 0.0850 17.936 -287.04  
## <none> 17.851 -285.73  
## - `H/A` 1 0.2803 18.131 -285.47  
## - `t\_5-9` 1 0.6124 18.463 -282.84  
## - Under\_D 1 1.1229 18.974 -278.88  
## - Infant\_D 1 1.1339 18.985 -278.80  
## - H\_B 1 1.4690 19.320 -276.26  
## - Adult\_M 1 6.4933 24.344 -242.74  
## - Income 1 17.2628 35.114 -189.63  
##   
## Step: AIC=-287.04  
## Life ~ Adult\_M + Infant\_D + H\_B + Under\_D + `H/A` + `t\_5-9` +   
## Income  
##   
## Df Sum of Sq RSS AIC  
## <none> 17.936 -287.04  
## - `H/A` 1 0.2636 18.199 -286.92  
## - `t\_5-9` 1 0.5744 18.510 -284.47  
## - Under\_D 1 1.1042 19.040 -280.38  
## - Infant\_D 1 1.1116 19.048 -280.32  
## - H\_B 1 1.4954 19.431 -277.43  
## - Adult\_M 1 6.5408 24.477 -243.96  
## - Income 1 17.3220 35.258 -191.04

Finalnie model ten próbuje przewidzieć wartości zmiennej zależnej “Life” na podstawie siedmiu zmiennych niezależnych: “Adult\_M”, “Infant\_D”, “H\_B”, “Under\_D”, “H/A”, “t\_5-9” i “Income”.

Współczynniki dla każdej zmiennej niezależnej wyrażone są w jednostkach wartości zmiennej zależnej (w tym przypadku “Life”) na jednostkę zmiennej niezależnej. Na przykład, każdy wzrost o jedną jednostkę zmiennej “Adult\_M” spowoduje spadek średniej wartości zmiennej “Life” o 0,3141 jednostki.

R kwantyfikuje jakość dopasowania modelu do danych, używając współczynnika determinacji R-kwadrat (Multiple R-squared), który w tym przypadku wynosi 0,8754. Oznacza to, że zmienne niezależne wyjaśniają 87,54% zmienności zmiennej zależnej “Life”. Współczynnik ten może zostać skorygowany przy użyciu współczynnika korekty R-kwadrat (Adjusted R-squared), który w tym przypadku wynosi 0,8691. F-statystyka i p-wartość F-statystyki służą do testowania ogólnej istotności modelu, a w tym przypadku wartość F-statystyki wynosi 137,6 i jest bardzo istotna (p-wartość <2,2e-16).

summary(mult\_reg )

##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + H\_B + Under\_D + `H/A` +   
## `t\_5-9` + Income, data = as.data.frame(training\_set\_mult))  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.17819 -0.19158 -0.01222 0.19080 0.98895   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.956e-16 3.005e-02 0.000 1.000000   
## Adult\_M -3.141e-01 4.444e-02 -7.068 7.27e-11 \*\*\*  
## Infant\_D 1.965e+00 6.742e-01 2.914 0.004170 \*\*   
## H\_B 1.126e-01 3.333e-02 3.380 0.000945 \*\*\*  
## Under\_D -1.962e+00 6.757e-01 -2.904 0.004293 \*\*   
## `H/A` -5.832e-02 4.111e-02 -1.419 0.158200   
## `t\_5-9` -8.690e-02 4.148e-02 -2.095 0.038039 \*   
## Income 5.349e-01 4.650e-02 11.503 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3618 on 137 degrees of freedom  
## Multiple R-squared: 0.8754, Adjusted R-squared: 0.8691   
## F-statistic: 137.6 on 7 and 137 DF, p-value: < 2.2e-16

Wynik predykcji to lista wartości numerycznych, które są oszacowanymi wartościami dla zmiennej objaśnianej (Life) na podstawie modelu regresji liniowej wytrenowanego na danych treningowych. Ta lista wartości numerycznych reprezentuje prognozowane wartości dla danych testowych, które zostały przekazane do funkcji predict(). Można porównać te wartości z rzeczywistymi wartościami dla zmiennej objaśnianej w danych testowych, aby oszacować jakość predykcji modelu.

y\_pred\_mult = predict(mult\_reg, newdata = as.data.frame(test\_set\_mult))  
y\_pred\_mult

## 1 2 3 4 5 6   
## 0.62492040 -1.34629012 0.84872804 1.28481681 0.44194619 1.07520596   
## 7 8 9 10 11 12   
## -0.01424005 0.43887772 0.87887451 1.07128296 -2.39636775 0.90903954   
## 13 14 15 16 17 18   
## 0.32924114 -1.33925475 -0.69116309 0.28076952 -1.59815890 -1.53614746   
## 19 20 21 22 23 24   
## -0.39849026 0.66637433 -0.05284672 -0.58992575 0.31480026 -0.16424161   
## 25 26 27 28 29 30   
## 0.02826906 -0.67066331 1.23944668 -1.66920465 -0.43525302 0.19850107   
## 31 32 33 34 35 36   
## 0.12488377 0.76276030 1.07305848 1.12072452 -0.27980829 -0.02389218   
## 37   
## -0.50657336

Kod który generuje tabelę porównawczą między rzeczywistymi wartościami Life w “test\_set”, a przewidywanymi wartościami “Life” na podstawie modelu “regressor”.

test\_set\_mult\_table <- data.frame(test\_set\_mult[,1], y\_pred\_mult)  
colnames(test\_set\_mult\_table) <- c("test dataset", "test predict")  
head(test\_set\_mult\_table)

## test dataset test predict  
## 1 0.7187034 0.6249204  
## 2 -2.1628557 -1.3462901  
## 3 0.5598773 0.8487280  
## 4 1.2859394 1.2848168  
## 5 0.5258431 0.4419462  
## 6 1.0930792 1.0752060

## Polynomial Regression

dataset\_poly <- read\_csv("C:\\Users\\Kacper\\Desktop\\Projekt Regression 2.0\\Dane z 2015.csv")

## Rows: 182 Columns: 22  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (2): Country, Status  
## dbl (20): Year, Life expectancy, Adult Mortality, infant deaths, Alcohol, pe...  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

library("dplyr")  
dataset\_poly <- dataset\_poly %>% select(-c(Country, Year, Status, Alcohol, `percentage expenditure`, `Total expenditure`))  
  
colnames(dataset\_poly)<-c("Life","Adult\_M","Infant\_D","H\_B","Measles","BMI", "Under\_D","Polio","Dipht","H/A", "GDP","Popl", "t\_1-19", "t\_5-9", "Income", "Schooling")  
  
dataset\_poly$H\_B = ifelse(is.na(dataset\_poly$H\_B),  
 ave(dataset\_poly$H\_B, FUN = function(x) mean(x, na.rm =  
TRUE)),  
 dataset\_poly$H\_B)  
  
dataset\_poly$BMI = ifelse(is.na(dataset\_poly$BMI ),  
 ave(dataset\_poly$BMI , FUN = function(x) mean(x, na.rm =  
TRUE)),  
 dataset\_poly$BMI )  
  
dataset\_poly$GDP = ifelse(is.na(dataset\_poly$GDP ),  
 ave(dataset\_poly$GDP , FUN = function(x) mean(x, na.rm =  
TRUE)),  
 dataset\_poly$GDP )  
  
dataset\_poly$Popl = ifelse(is.na(dataset\_poly$Popl ),  
 ave(dataset\_poly$Popl , FUN = function(x) mean(x, na.rm  
= TRUE)),  
dataset\_poly$Popl )  
  
dataset\_poly$`t\_1-19` = ifelse(is.na(dataset\_poly$`t\_1-19` ),  
 ave(dataset\_poly$`t\_1-19` , FUN = function(x) mean(x,  
na.rm = TRUE)),  
 dataset\_poly$`t\_1-19` )  
  
dataset\_poly$`t\_5-9` = ifelse(is.na(dataset\_poly$`t\_5-9` ),  
 ave(dataset\_poly$`t\_5-9` , FUN = function(x) mean(x,  
na.rm = TRUE)),  
 dataset\_poly$`t\_5-9` )  
  
dataset\_poly$Income = ifelse(is.na(dataset\_poly$Income ),  
 ave(dataset\_poly$Income , FUN = function(x)  
mean(x, na.rm = TRUE)),  
 dataset\_poly$Income )  
  
dataset\_poly$Schooling = ifelse(is.na(dataset\_poly$Schooling ),  
 ave(dataset\_poly$Schooling , FUN = function(x)  
mean(x, na.rm = TRUE)),  
 dataset\_poly$Schooling )  
  
View(dataset\_poly)

dataset\_poly$Adult\_M = dataset\_poly$Adult\_M^2  
dataset\_poly$Infant\_D = dataset\_poly$Infant\_D^2  
dataset\_poly$H\_B = dataset\_poly$H\_B^2  
dataset\_poly$Measles = dataset\_poly$Measles^2  
dataset\_poly$BMI = dataset\_poly$BMI^2  
dataset\_poly$Under\_D = dataset\_poly$Under\_D^2  
dataset\_poly$Polio = dataset\_poly$Polio^2  
dataset\_poly$Dipht = dataset\_poly$Dipht^2  
dataset\_poly$`H/A` = dataset\_poly$`H/A`^2  
dataset\_poly$GDP = dataset\_poly$GDP^2  
dataset\_poly$Popl = dataset\_poly$Popl^2  
dataset\_poly$`t\_1-19` = dataset\_poly$`t\_1-19`^2  
dataset\_poly$`t\_5-9` = dataset\_poly$`t\_5-9`^2  
dataset\_poly$Income = dataset\_poly$Income^2  
dataset\_poly$Schooling = dataset\_poly$Schooling^2

# install.packages('caTools')  
library(caTools)  
set.seed(123)  
split = sample.split(dataset\_poly$Life, SplitRatio = 0.8)  
training\_set\_poly = subset(dataset\_poly, split == TRUE)  
test\_set\_poly = subset(dataset\_poly, split == FALSE)

Kod pozostawia pierwsze dane nie spotęgowane.

#dataset\_poly$Adult\_M = poly(dataset\_poly$Adult\_M, degree = 2, raw = TRUE)  
#dataset\_poly$Infant\_D = poly(dataset\_poly$Infant\_D, degree = 2, raw = TRUE)  
#dataset\_poly$H\_B = poly(dataset\_poly$H\_B, degree = 2, raw = TRUE)  
#dataset\_poly$Measles = poly(dataset\_poly$Measles, degree = 2, raw = TRUE)  
#dataset\_poly$BMI = poly(dataset\_poly$BMI, degree = 2, raw = TRUE)  
#dataset\_poly$Under\_D = poly(dataset\_poly$Under\_D, degree = 2, raw = TRUE)  
#dataset\_poly$Polio = poly(dataset\_poly$Polio, degree = 2, raw = TRUE)  
#dataset\_poly$Dipht = poly(dataset\_poly$Dipht, degree = 2, raw = TRUE)  
#dataset\_poly$`H/A` = poly(dataset\_poly$`H/A`, degree = 2, raw = TRUE)  
#dataset\_poly$GDP = poly(dataset\_poly$GDP, degree = 2, raw = TRUE)  
#dataset\_poly$Popl = poly(dataset\_poly$Popl, degree = 2, raw = TRUE)  
#dataset\_poly$`t\_1-19` = poly(dataset\_poly$`t\_1-19`, degree = 2, raw = TRUE)  
#dataset\_poly$`t\_5-9` = poly(dataset\_poly$`t\_5-9`, degree = 2, raw = TRUE)  
#dataset\_poly$Income = poly(dataset\_poly$Income, degree = 2, raw = TRUE)  
#dataset\_poly$Schooling = poly(dataset\_poly$Schooling, degree = 2, raw = TRUE)

poly\_reg = lm(formula = Life ~ .,  
 data = training\_set\_poly)

summary(poly\_reg)

##   
## Call:  
## lm(formula = Life ~ ., data = training\_set\_poly)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -11.3219 -1.6298 -0.0525 1.4530 7.7421   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.081e+01 1.318e+00 46.144 < 2e-16 \*\*\*  
## Adult\_M -8.967e-05 1.078e-05 -8.320 1.08e-13 \*\*\*  
## Infant\_D 5.511e-04 2.616e-04 2.107 0.0371 \*   
## H\_B 1.443e-04 2.435e-04 0.593 0.5545   
## Measles -3.011e-10 1.643e-09 -0.183 0.8549   
## BMI 1.730e-05 1.951e-04 0.089 0.9295   
## Under\_D -3.759e-04 1.735e-04 -2.167 0.0321 \*   
## Polio 1.145e-05 1.584e-04 0.072 0.9425   
## Dipht 1.982e-04 2.882e-04 0.688 0.4928   
## `H/A` 6.832e-02 3.601e-02 1.897 0.0600 .   
## GDP -1.970e-10 5.264e-10 -0.374 0.7088   
## Popl -2.810e-17 4.597e-17 -0.611 0.5421   
## `t\_1-19` -7.706e-03 2.399e-02 -0.321 0.7485   
## `t\_5-9` 7.836e-03 2.338e-02 0.335 0.7381   
## Income 2.150e+01 3.832e+00 5.612 1.17e-07 \*\*\*  
## Schooling 2.590e-03 9.191e-03 0.282 0.7785   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.848 on 129 degrees of freedom  
## Multiple R-squared: 0.8821, Adjusted R-squared: 0.8684   
## F-statistic: 64.34 on 15 and 129 DF, p-value: < 2.2e-16

poly\_reg <-   
step(lm(Life ~ .,  
 data=training\_set\_poly, direction="backward"))

## Warning: W poleceniu 'lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...)':  
## dodatkowy argument 'direction' zostanie odrzucony

## Start: AIC=318.55  
## Life ~ Adult\_M + Infant\_D + H\_B + Measles + BMI + Under\_D + Polio +   
## Dipht + `H/A` + GDP + Popl + `t\_1-19` + `t\_5-9` + Income +   
## Schooling  
##   
## Df Sum of Sq RSS AIC  
## - Polio 1 0.04 1046.3 316.56  
## - BMI 1 0.06 1046.3 316.56  
## - Measles 1 0.27 1046.5 316.59  
## - Schooling 1 0.64 1046.9 316.64  
## - `t\_1-19` 1 0.84 1047.1 316.67  
## - `t\_5-9` 1 0.91 1047.1 316.68  
## - GDP 1 1.14 1047.4 316.71  
## - H\_B 1 2.85 1049.1 316.94  
## - Popl 1 3.03 1049.2 316.97  
## - Dipht 1 3.84 1050.1 317.08  
## <none> 1046.2 318.55  
## - `H/A` 1 29.20 1075.4 320.54  
## - Infant\_D 1 36.00 1082.2 321.46  
## - Under\_D 1 38.07 1084.3 321.73  
## - Income 1 255.39 1301.6 348.22  
## - Adult\_M 1 561.38 1607.6 378.84

## Warning: W poleceniu 'lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...)':  
## dodatkowy argument 'direction' zostanie odrzucony

##   
## Step: AIC=316.56  
## Life ~ Adult\_M + Infant\_D + H\_B + Measles + BMI + Under\_D + Dipht +   
## `H/A` + GDP + Popl + `t\_1-19` + `t\_5-9` + Income + Schooling  
##   
## Df Sum of Sq RSS AIC  
## - BMI 1 0.05 1046.3 314.56  
## - Measles 1 0.29 1046.5 314.60  
## - Schooling 1 0.64 1046.9 314.64  
## - `t\_1-19` 1 0.83 1047.1 314.67  
## - `t\_5-9` 1 0.89 1047.2 314.68  
## - GDP 1 1.14 1047.4 314.71  
## - H\_B 1 2.90 1049.2 314.96  
## - Popl 1 3.75 1050.0 315.07  
## - Dipht 1 4.84 1051.1 315.22  
## <none> 1046.3 316.56  
## - `H/A` 1 29.29 1075.5 318.56  
## - Infant\_D 1 36.26 1082.5 319.50  
## - Under\_D 1 38.29 1084.5 319.77  
## - Income 1 257.66 1303.9 346.48  
## - Adult\_M 1 564.92 1611.2 377.16

## Warning: W poleceniu 'lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...)':  
## dodatkowy argument 'direction' zostanie odrzucony

##   
## Step: AIC=314.56  
## Life ~ Adult\_M + Infant\_D + H\_B + Measles + Under\_D + Dipht +   
## `H/A` + GDP + Popl + `t\_1-19` + `t\_5-9` + Income + Schooling  
##   
## Df Sum of Sq RSS AIC  
## - Measles 1 0.34 1046.7 312.61  
## - Schooling 1 0.69 1047.0 312.66  
## - `t\_1-19` 1 0.81 1047.1 312.67  
## - `t\_5-9` 1 0.85 1047.2 312.68  
## - GDP 1 1.13 1047.4 312.72  
## - H\_B 1 2.94 1049.2 312.97  
## - Popl 1 3.88 1050.2 313.10  
## - Dipht 1 4.79 1051.1 313.22  
## <none> 1046.3 314.56  
## - `H/A` 1 29.26 1075.6 316.56  
## - Infant\_D 1 37.93 1084.2 317.73  
## - Under\_D 1 39.92 1086.2 317.99  
## - Income 1 260.59 1306.9 344.81  
## - Adult\_M 1 570.58 1616.9 375.67

## Warning: W poleceniu 'lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...)':  
## dodatkowy argument 'direction' zostanie odrzucony

##   
## Step: AIC=312.61  
## Life ~ Adult\_M + Infant\_D + H\_B + Under\_D + Dipht + `H/A` + GDP +   
## Popl + `t\_1-19` + `t\_5-9` + Income + Schooling  
##   
## Df Sum of Sq RSS AIC  
## - Schooling 1 0.65 1047.3 310.70  
## - `t\_1-19` 1 0.85 1047.5 310.73  
## - `t\_5-9` 1 0.96 1047.6 310.74  
## - GDP 1 1.15 1047.8 310.77  
## - H\_B 1 2.93 1049.6 311.01  
## - Popl 1 3.92 1050.6 311.15  
## - Dipht 1 4.74 1051.4 311.26  
## <none> 1046.7 312.61  
## - `H/A` 1 29.46 1076.1 314.63  
## - Infant\_D 1 44.53 1091.2 316.65  
## - Under\_D 1 44.71 1091.4 316.67  
## - Income 1 265.63 1312.3 343.40  
## - Adult\_M 1 570.30 1617.0 373.68

## Warning: W poleceniu 'lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...)':  
## dodatkowy argument 'direction' zostanie odrzucony

##   
## Step: AIC=310.7  
## Life ~ Adult\_M + Infant\_D + H\_B + Under\_D + Dipht + `H/A` + GDP +   
## Popl + `t\_1-19` + `t\_5-9` + Income  
##   
## Df Sum of Sq RSS AIC  
## - `t\_1-19` 1 1.02 1048.3 308.84  
## - `t\_5-9` 1 1.15 1048.5 308.86  
## - GDP 1 1.95 1049.2 308.97  
## - H\_B 1 3.35 1050.6 309.16  
## - Popl 1 3.86 1051.2 309.23  
## - Dipht 1 4.47 1051.8 309.32  
## <none> 1047.3 310.70  
## - `H/A` 1 29.13 1076.4 312.68  
## - Infant\_D 1 45.34 1092.6 314.84  
## - Under\_D 1 45.53 1092.8 314.87  
## - Adult\_M 1 587.26 1634.6 373.25  
## - Income 1 1202.62 2249.9 419.58

## Warning: W poleceniu 'lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...)':  
## dodatkowy argument 'direction' zostanie odrzucony

##   
## Step: AIC=308.84  
## Life ~ Adult\_M + Infant\_D + H\_B + Under\_D + Dipht + `H/A` + GDP +   
## Popl + `t\_5-9` + Income  
##   
## Df Sum of Sq RSS AIC  
## - `t\_5-9` 1 0.18 1048.5 306.86  
## - GDP 1 2.03 1050.3 307.12  
## - H\_B 1 3.32 1051.6 307.30  
## - Popl 1 3.88 1052.2 307.38  
## - Dipht 1 4.40 1052.7 307.45  
## <none> 1048.3 308.84  
## - `H/A` 1 29.11 1077.4 310.81  
## - Infant\_D 1 46.56 1094.9 313.14  
## - Under\_D 1 46.77 1095.1 313.17  
## - Adult\_M 1 589.06 1637.4 371.50  
## - Income 1 1207.93 2256.2 417.99

## Warning: W poleceniu 'lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...)':  
## dodatkowy argument 'direction' zostanie odrzucony

##   
## Step: AIC=306.86  
## Life ~ Adult\_M + Infant\_D + H\_B + Under\_D + Dipht + `H/A` + GDP +   
## Popl + Income  
##   
## Df Sum of Sq RSS AIC  
## - GDP 1 1.99 1050.5 305.14  
## - H\_B 1 3.31 1051.8 305.32  
## - Popl 1 3.98 1052.5 305.41  
## - Dipht 1 4.47 1053.0 305.48  
## <none> 1048.5 306.86  
## - `H/A` 1 29.01 1077.5 308.82  
## - Infant\_D 1 46.38 1094.9 311.14  
## - Under\_D 1 46.61 1095.1 311.17  
## - Adult\_M 1 593.94 1642.4 369.94  
## - Income 1 1346.47 2395.0 424.64

## Warning: W poleceniu 'lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...)':  
## dodatkowy argument 'direction' zostanie odrzucony

##   
## Step: AIC=305.14  
## Life ~ Adult\_M + Infant\_D + H\_B + Under\_D + Dipht + `H/A` + Popl +   
## Income  
##   
## Df Sum of Sq RSS AIC  
## - H\_B 1 3.06 1053.5 303.56  
## - Popl 1 3.98 1054.5 303.69  
## - Dipht 1 4.78 1055.3 303.80  
## <none> 1050.5 305.14  
## - `H/A` 1 28.72 1079.2 307.05  
## - Infant\_D 1 47.19 1097.7 309.51  
## - Under\_D 1 47.42 1097.9 309.54  
## - Adult\_M 1 595.08 1645.6 368.22  
## - Income 1 1438.83 2489.3 428.24

## Warning: W poleceniu 'lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...)':  
## dodatkowy argument 'direction' zostanie odrzucony

##   
## Step: AIC=303.56  
## Life ~ Adult\_M + Infant\_D + Under\_D + Dipht + `H/A` + Popl +   
## Income  
##   
## Df Sum of Sq RSS AIC  
## - Popl 1 3.81 1057.4 302.09  
## <none> 1053.5 303.56  
## - `H/A` 1 28.05 1081.6 305.37  
## - Infant\_D 1 47.96 1101.5 308.02  
## - Under\_D 1 48.19 1101.7 308.05  
## - Dipht 1 86.16 1139.7 312.96  
## - Adult\_M 1 593.43 1647.0 366.34  
## - Income 1 1510.31 2563.9 430.52

## Warning: W poleceniu 'lm.fit(x, y, offset = offset, singular.ok = singular.ok, ...)':  
## dodatkowy argument 'direction' zostanie odrzucony

##   
## Step: AIC=302.09  
## Life ~ Adult\_M + Infant\_D + Under\_D + Dipht + `H/A` + Income  
##   
## Df Sum of Sq RSS AIC  
## <none> 1057.4 302.09  
## - `H/A` 1 28.72 1086.1 303.97  
## - Infant\_D 1 46.61 1104.0 306.34  
## - Under\_D 1 46.83 1104.2 306.37  
## - Dipht 1 89.02 1146.4 311.81  
## - Adult\_M 1 593.96 1651.3 364.73  
## - Income 1 1508.45 2565.8 428.63

summary(poly\_reg)

##   
## Call:  
## lm(formula = Life ~ Adult\_M + Infant\_D + Under\_D + Dipht + `H/A` +   
## Income, data = training\_set\_poly, direction = "backward")  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -11.3777 -1.5289 -0.1597 1.5155 7.7818   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.095e+01 1.125e+00 54.201 < 2e-16 \*\*\*  
## Adult\_M -8.933e-05 1.015e-05 -8.805 4.88e-15 \*\*\*  
## Infant\_D 5.384e-04 2.183e-04 2.466 0.014876 \*   
## Under\_D -3.689e-04 1.492e-04 -2.472 0.014642 \*   
## Dipht 3.560e-04 1.045e-04 3.409 0.000856 \*\*\*  
## `H/A` 6.715e-02 3.469e-02 1.936 0.054924 .   
## Income 2.197e+01 1.566e+00 14.031 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.768 on 138 degrees of freedom  
## Multiple R-squared: 0.8808, Adjusted R-squared: 0.8757   
## F-statistic: 170 on 6 and 138 DF, p-value: < 2.2e-16

y\_pred\_poly = predict(poly\_reg, newdata = as.data.frame(test\_set\_poly))  
y\_pred\_poly

## 1 2 3 4 5 6 7 8   
## 76.70962 57.62485 77.93285 83.00904 75.94669 81.55025 71.24129 74.91428   
## 9 10 11 12 13 14 15 16   
## 80.29849 82.08973 51.56411 79.32044 72.24539 62.20242 67.36020 74.55588   
## 17 18 19 20 21 22 23 24   
## 58.34224 61.46925 68.98470 77.88104 71.47106 67.84500 75.73046 69.25193   
## 25 26 27 28 29 30 31 32   
## 71.84390 66.83225 82.59163 -22.05823 66.96636 76.18356 71.93482 79.36955   
## 33 34 35 36 37   
## 79.77702 81.11281 66.72750 72.93146 69.10447

accuracy = sqrt(mean(y\_pred\_poly / test\_set\_poly$Life)^2)  
accuracy

## [1] 0.9699852

## SVR

dataset <- read\_csv("C:\\Users\\Kacper\\Desktop\\Projekt Regression 2.0\\Dane z 2015.csv")

## Rows: 182 Columns: 22  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (2): Country, Status  
## dbl (20): Year, Life expectancy, Adult Mortality, infant deaths, Alcohol, pe...  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

library("dplyr")  
dataset <- dataset %>% select(-c(Country, Year, Status, Alcohol,  
`percentage expenditure`, `Total expenditure`))  
colnames(dataset)<-c("Life","Adult\_M","Infant\_D","H\_B","Measles","BMI",  
"Under\_D","Polio","Dipht","H/A", "GDP","Popl", "t\_1-19", "t\_5-9", "Income",  
"Schooling")  
dataset$H\_B = ifelse(is.na(dataset$H\_B),  
 ave(dataset$H\_B, FUN = function(x) mean(x, na.rm =  
TRUE)),  
 dataset$H\_B)  
  
dataset$BMI = ifelse(is.na(dataset$BMI ),  
 ave(dataset$BMI , FUN = function(x) mean(x, na.rm =  
TRUE)),  
 dataset$BMI )  
  
dataset$GDP = ifelse(is.na(dataset$GDP ),  
 ave(dataset$GDP , FUN = function(x) mean(x, na.rm =  
TRUE)),  
 dataset$GDP )  
  
dataset$Popl = ifelse(is.na(dataset$Popl ),  
 ave(dataset$Popl , FUN = function(x) mean(x, na.rm  
= TRUE)),  
dataset$Popl )  
  
dataset$`t\_1-19` = ifelse(is.na(dataset$`t\_1-19` ),  
 ave(dataset$`t\_1-19` , FUN = function(x) mean(x,  
na.rm = TRUE)),  
 dataset$`t\_1-19` )  
  
dataset$`t\_5-9` = ifelse(is.na(dataset$`t\_5-9` ),  
 ave(dataset$`t\_5-9` , FUN = function(x) mean(x,  
na.rm = TRUE)),  
 dataset$`t\_5-9` )  
  
dataset$Income = ifelse(is.na(dataset$Income ),  
 ave(dataset$Income , FUN = function(x)  
mean(x, na.rm = TRUE)),  
 dataset$Income )  
  
dataset$Schooling = ifelse(is.na(dataset$Schooling ),  
 ave(dataset$Schooling , FUN = function(x)  
mean(x, na.rm = TRUE)),  
 dataset$Schooling )  
  
View(dataset)

# install.packages('caTools')  
library(caTools)  
set.seed(123)  
split = sample.split(dataset$Life, SplitRatio = 0.8)  
training\_set\_svm = subset(dataset, split == TRUE)  
test\_set\_svm = subset(dataset, split == FALSE)

View(training\_set\_svm)

View(test\_set\_svm)

training\_set\_svm = scale(training\_set\_svm)  
test\_set\_svm = scale(test\_set\_svm)

library(e1071)

## Warning: pakiet 'e1071' został zbudowany w wersji R 4.2.2

svm\_reg= svm(formula = Life ~ .,  
 data = training\_set\_svm,  
 type = 'eps-regression',  
 kernel = 'radial')

summary(svm\_reg)

##   
## Call:  
## svm(formula = Life ~ ., data = training\_set\_svm, type = "eps-regression",   
## kernel = "radial")  
##   
##   
## Parameters:  
## SVM-Type: eps-regression   
## SVM-Kernel: radial   
## cost: 1   
## gamma: 0.06666667   
## epsilon: 0.1   
##   
##   
## Number of Support Vectors: 106

y\_pred\_svm = predict(svm\_reg, newdata = as.data.frame(test\_set\_svm))  
y\_pred\_svm

## 1 2 3 4 5 6   
## 0.98576683 -1.32725304 1.08924043 1.19304550 0.41403343 1.18823712   
## 7 8 9 10 11 12   
## 0.15378448 0.73332863 0.86961408 1.21525367 -1.85063426 1.10522712   
## 13 14 15 16 17 18   
## 0.09865020 -1.13237688 -1.05459853 0.43768603 -1.50229750 -1.36362243   
## 19 20 21 22 23 24   
## -0.49014004 0.75328795 0.14008582 -0.64445081 0.34254885 0.07690509   
## 25 26 27 28 29 30   
## 0.06520564 -0.54618963 1.36386364 -0.35480675 -0.71525864 0.29719394   
## 31 32 33 34 35 36   
## 0.33876754 0.36831984 1.36225500 1.16300575 -1.03186833 0.13250720   
## 37   
## -1.27545481

## Decision Tree Regression

dataset <- read\_csv("C:\\Users\\Kacper\\Desktop\\Projekt Regression 2.0\\Dane z 2015.csv")

## Rows: 182 Columns: 22  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (2): Country, Status  
## dbl (20): Year, Life expectancy, Adult Mortality, infant deaths, Alcohol, pe...  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

library("dplyr")  
dataset <- dataset %>% select(-c(Country, Year, Status, Alcohol,  
`percentage expenditure`, `Total expenditure`))  
colnames(dataset)<-c("Life","Adult\_M","Infant\_D","H\_B","Measles","BMI",  
"Under\_D","Polio","Dipht","H/A", "GDP","Popl", "t\_1-19", "t\_5-9", "Income",  
"Schooling")  
dataset$H\_B = ifelse(is.na(dataset$H\_B),  
 ave(dataset$H\_B, FUN = function(x) mean(x, na.rm =  
TRUE)),  
 dataset$H\_B)  
  
dataset$BMI = ifelse(is.na(dataset$BMI ),  
 ave(dataset$BMI , FUN = function(x) mean(x, na.rm =  
TRUE)),  
 dataset$BMI )  
  
dataset$GDP = ifelse(is.na(dataset$GDP ),  
 ave(dataset$GDP , FUN = function(x) mean(x, na.rm =  
TRUE)),  
 dataset$GDP )  
  
dataset$Popl = ifelse(is.na(dataset$Popl ),  
 ave(dataset$Popl , FUN = function(x) mean(x, na.rm  
= TRUE)),  
dataset$Popl )  
  
dataset$`t\_1-19` = ifelse(is.na(dataset$`t\_1-19` ),  
 ave(dataset$`t\_1-19` , FUN = function(x) mean(x,  
na.rm = TRUE)),  
 dataset$`t\_1-19` )  
  
dataset$`t\_5-9` = ifelse(is.na(dataset$`t\_5-9` ),  
 ave(dataset$`t\_5-9` , FUN = function(x) mean(x,  
na.rm = TRUE)),  
 dataset$`t\_5-9` )  
  
dataset$Income = ifelse(is.na(dataset$Income ),  
 ave(dataset$Income , FUN = function(x)  
mean(x, na.rm = TRUE)),  
 dataset$Income )  
  
dataset$Schooling = ifelse(is.na(dataset$Schooling ),  
 ave(dataset$Schooling , FUN = function(x)  
mean(x, na.rm = TRUE)),  
 dataset$Schooling )  
  
dataset = data.frame(dataset)  
  
View(dataset)

library(caTools)  
set.seed(123)  
split = sample.split(dataset$Life, SplitRatio = 0.8)  
training\_set\_dec = subset(dataset, split == TRUE)  
test\_set\_dec = subset(dataset, split == FALSE)

training\_set\_dec= scale(training\_set\_dec)  
test\_set\_dec = scale(test\_set\_dec)

View(training\_set\_dec)

View(test\_set\_dec)

library(rpart)

## Warning: pakiet 'rpart' został zbudowany w wersji R 4.2.2

training\_set\_dec = data.frame(training\_set\_dec)  
dec\_reg = rpart(formula = Life ~ .,  
 data = training\_set\_dec,  
 control = rpart.control(minsplit = 1))

summary(dec\_reg)

## Call:  
## rpart(formula = Life ~ ., data = training\_set\_dec, control = rpart.control(minsplit = 1))  
## n= 145   
##   
## CP nsplit rel error xerror xstd  
## 1 0.01 0 1 0 0  
##   
## Node number 1: 145 observations  
## mean=8.422382e-17, MSE=0.9931034

dec\_reg

## n= 145   
##   
## node), split, n, deviance, yval  
## \* denotes terminal node  
##   
## 1) root 145 144 8.422382e-17 \*

y\_pred\_dec = predict(dec\_reg, newdata = as.data.frame(test\_set\_dec))  
y\_pred\_dec

## [1] 8.422382e-17 8.422382e-17 8.422382e-17 8.422382e-17 8.422382e-17  
## [6] 8.422382e-17 8.422382e-17 8.422382e-17 8.422382e-17 8.422382e-17  
## [11] 8.422382e-17 8.422382e-17 8.422382e-17 8.422382e-17 8.422382e-17  
## [16] 8.422382e-17 8.422382e-17 8.422382e-17 8.422382e-17 8.422382e-17  
## [21] 8.422382e-17 8.422382e-17 8.422382e-17 8.422382e-17 8.422382e-17  
## [26] 8.422382e-17 8.422382e-17 8.422382e-17 8.422382e-17 8.422382e-17  
## [31] 8.422382e-17 8.422382e-17 8.422382e-17 8.422382e-17 8.422382e-17  
## [36] 8.422382e-17 8.422382e-17

y\_pred\_dec = predict(dec\_reg, data.frame(Adult\_M = 100, Infant\_D = 56, H\_B = 90, Measles = 150, BMI = 42, Under\_D = 80, Polio = 70, Dipht = 80, `H/A` = 3.5, GDP = 300, Popl = 10000000, `t\_1-19` = 6, `t\_5-9` = 4, Income = 0.5, Schooling = 4.5))  
y\_pred\_dec

## [1] 8.422382e-17

## Random Forest Regression

dataset <- read\_csv("C:\\Users\\Kacper\\Desktop\\Projekt Regression 2.0\\Dane z 2015.csv")

## Rows: 182 Columns: 22  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (2): Country, Status  
## dbl (20): Year, Life expectancy, Adult Mortality, infant deaths, Alcohol, pe...  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

library("dplyr")  
dataset <- dataset %>% select(-c(Country, Year, Status, Alcohol,  
`percentage expenditure`, `Total expenditure`))  
colnames(dataset)<-c("Life","Adult\_M","Infant\_D","H\_B","Measles","BMI",  
"Under\_D","Polio","Dipht","H/A", "GDP","Popl", "t\_1-19", "t\_5-9", "Income",  
"Schooling")  
dataset$H\_B = ifelse(is.na(dataset$H\_B),  
 ave(dataset$H\_B, FUN = function(x) mean(x, na.rm =  
TRUE)),  
 dataset$H\_B)  
  
dataset$BMI = ifelse(is.na(dataset$BMI ),  
 ave(dataset$BMI , FUN = function(x) mean(x, na.rm =  
TRUE)),  
 dataset$BMI )  
  
dataset$GDP = ifelse(is.na(dataset$GDP ),  
 ave(dataset$GDP , FUN = function(x) mean(x, na.rm =  
TRUE)),  
 dataset$GDP )  
  
dataset$Popl = ifelse(is.na(dataset$Popl ),  
 ave(dataset$Popl , FUN = function(x) mean(x, na.rm  
= TRUE)),  
dataset$Popl )  
  
dataset$`t\_1-19` = ifelse(is.na(dataset$`t\_1-19` ),  
 ave(dataset$`t\_1-19` , FUN = function(x) mean(x,  
na.rm = TRUE)),  
 dataset$`t\_1-19` )  
  
dataset$`t\_5-9` = ifelse(is.na(dataset$`t\_5-9` ),  
 ave(dataset$`t\_5-9` , FUN = function(x) mean(x,  
na.rm = TRUE)),  
 dataset$`t\_5-9` )  
  
dataset$Income = ifelse(is.na(dataset$Income ),  
 ave(dataset$Income , FUN = function(x)  
mean(x, na.rm = TRUE)),  
 dataset$Income )  
  
dataset$Schooling = ifelse(is.na(dataset$Schooling ),  
 ave(dataset$Schooling , FUN = function(x)  
mean(x, na.rm = TRUE)),  
 dataset$Schooling )  
  
dataset = data.frame(dataset)  
  
View(dataset)

library(caTools)  
set.seed(123)  
split = sample.split(dataset$Life, SplitRatio = 0.8)  
training\_set\_frt = subset(dataset, split == TRUE)  
test\_set\_frt = subset(dataset, split == FALSE)

training\_set\_frt= scale(training\_set\_frt)  
test\_set\_frt = scale(test\_set\_frt)

library(randomForest)

## Warning: pakiet 'randomForest' został zbudowany w wersji R 4.2.2

## randomForest 4.7-1.1

## Type rfNews() to see new features/changes/bug fixes.

##   
## Dołączanie pakietu: 'randomForest'

## Następujący obiekt został zakryty z 'package:dplyr':  
##   
## combine

## Następujący obiekt został zakryty z 'package:ggplot2':  
##   
## margin

set.seed(1234)  
frt\_reg= randomForest(Life ~ .,   
 data = training\_set\_frt,  
 ntree = 500)

summary(frt\_reg)

## Length Class Mode   
## call 4 -none- call   
## type 1 -none- character  
## predicted 145 -none- numeric   
## mse 500 -none- numeric   
## rsq 500 -none- numeric   
## oob.times 145 -none- numeric   
## importance 15 -none- numeric   
## importanceSD 0 -none- NULL   
## localImportance 0 -none- NULL   
## proximity 0 -none- NULL   
## ntree 1 -none- numeric   
## mtry 1 -none- numeric   
## forest 11 -none- list   
## coefs 0 -none- NULL   
## y 145 -none- numeric   
## test 0 -none- NULL   
## inbag 0 -none- NULL   
## terms 3 terms call