

Wnioskowanie Statystyczne - projekt

Mikołaj Gilewicz, Łukasz Kwiatkowski

Wstęp

Import bibliotek i danych

```
library(ggplot2)
library(gridExtra)

diamonds <- read.csv('diamonds.csv', colClasses=c(X='NULL'))
```

W naszym projekcie wykorzystamy zbiór danych zawierający parametry niemal 54 tysięcy diamentów pochodzących z katalogu “Tiffany & Co.” z 2017 roku.

Opis zmiennych

zmienna	opis
carat	waga diamentu (w karatach)
cut	jakość szlifu diamentu (zmienna kategoryczna)
color	klasa koloru diamentu (zmienna kategoryczna)
clarity	poziom przejrzystości diamentu (zmienna kategoryczna)
depth	proporcja wysokości diamentu do jego obwodu (w %)
table	maksymalna szerokość górnej części diamentu (w mm)
price	cena katalogowa diamentu podana w USD
x	długość diamentu (w mm)
y	szerokość diamentu (w mm)
z	wysokość diamentu (w mm)

Wstępne przekształcenia zmiennych

Dostrzegliśmy, że niektóre diamenty nie mają podanych niektórych wymiarów (podana jest wartość 0 mm), dlatego będziemy musieli pozbyć się ich ze zbioru danych.

```
diamonds$x[diamonds$x == 0] = NA
diamonds$y[diamonds$y == 0] = NA
diamonds$z[diamonds$z == 0] = NA
```

Faktoryzujemy zmienne kategoryczne, a ich wartości ustawiamy w kolejności od “najlepszej” do “najgorszej”, co ułatwi potem odczytywanie wykresów.

```
diamonds$cut <- factor(diamonds$cut, c('Ideal', 'Premium', 'Very Good', 'Good', 'Fair'))
diamonds$color <- factor(diamonds$color, c('D', 'E', 'F', 'G', 'H', 'I', 'J'))
diamonds$clarity <- factor(diamonds$clarity, c('IF', 'VVS1', 'VVS2', 'VS1', 'VS2', 'SI1', 'SI2', 'I1'))
```

Uwaga: Zmienna ‘color’ określa jakość koloru danego diamentu. Nie chodzi więc tutaj o odcień czy barwę, a sklasyfikowanie koloru za pomocą używanej powszechnie skali od “D” (kolor najlepszy) do “Z” (kolor

najgorszy). W naszych danych najgorszą występującą klasą jest “J”.

Z tak przekształconego zbioru danych usuwamy wszystkie wiersze z wartościami NA.

```
nrow(diamonds)
```

```
## [1] 53940  
diamonds <- na.omit(diamonds)  
nrow(diamonds)
```

```
## [1] 53920
```

Z naszego zbioru, w którym znajdowało się początkowo 53 940 diamentów, usunęliśmy 20, które nie zawierały pewnych parametrów. Stąd naszą analizę przeprowadzimy na pozostałych 53 920 diamentach.

Analiza eksploracyjna

Zależności między wymiarami diamentu

Intuicja podpowiada, że wymiary diamentu powinny być ze sobą w jakiś sposób skorelowane. Tezę tę zdaje się potwierdzać macierz korelacji zmiennych x , y , z .

```
cor(diamonds[c(8:10)])
```

```
##           x          y          z  
## x 1.0000000 0.9749183 0.9754351  
## y 0.9749183 1.0000000 0.9567437  
## z 0.9754351 0.9567437 1.0000000
```

Zbudujemy jeszcze modele badające zależności między tymi zmiennymi.

```
model_0.xy <- lm(x ~ y, diamonds)  
summary(model_0.xy)
```

```
##  
## Call:  
## lm(formula = x ~ y, data = diamonds)  
##  
## Residuals:  
##      Min       1Q   Median       3Q      Max  
## -48.532  -0.059  -0.013   0.053   2.489  
##  
## Coefficients:  
##                 Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 0.2421057  0.0055026     44    <2e-16 ***  
## y          0.9572152  0.0009411    1017    <2e-16 ***  
## ---  
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  
##  
## Residual standard error: 0.2491 on 53918 degrees of freedom  
## Multiple R-squared:  0.9505, Adjusted R-squared:  0.9505  
## F-statistic: 1.035e+06 on 1 and 53918 DF,  p-value: < 2.2e-16  
plot_1.xy <- ggplot(diamonds, aes(x, y)) +  
  geom_point() +  
  geom_smooth(method='lm', formula='y~x') +  
  labs(title='Wykresy zależności między wymiarami diamentów')
```

```

model_0.xz <- lm(x ~ z, diamonds)
summary(model_0.xz)

##
## Call:
## lm(formula = x ~ z, data = diamonds)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -44.535 -0.083 -0.021  0.064  4.889
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.229427  0.005456  42.05  <2e-16 ***
## z           1.554273  0.001512 1028.20  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2466 on 53918 degrees of freedom
## Multiple R-squared:  0.9515, Adjusted R-squared:  0.9515
## F-statistic: 1.057e+06 on 1 and 53918 DF, p-value: < 2.2e-16

plot_1.xz <- ggplot(diamonds, aes(x, z)) +
  geom_point() +
  geom_smooth(method='lm', formula='y~x')

model_0.yz <- lm(y ~ z, diamonds)
summary(model_0.yz)

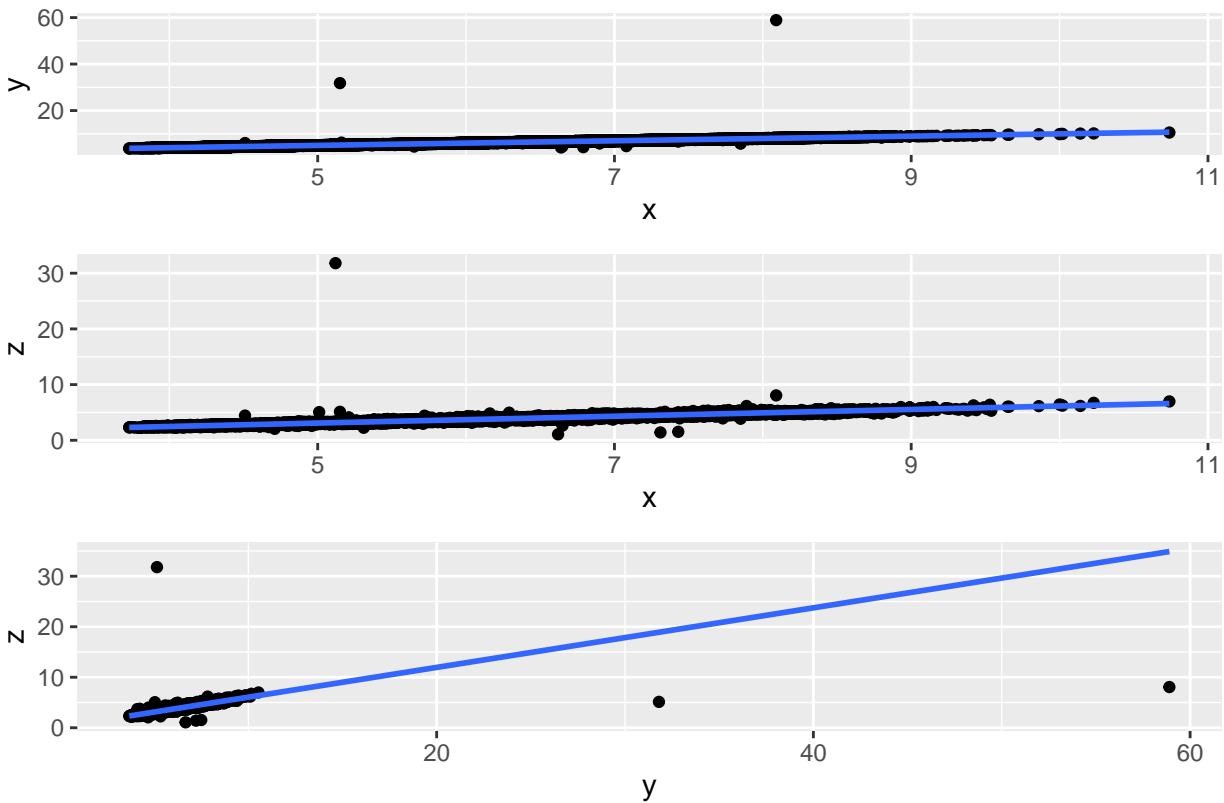
##
## Call:
## lm(formula = y ~ z, data = diamonds)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -44.464 -0.081 -0.016  0.066  46.147
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.238311  0.007339  32.47  <2e-16 ***
## z           1.552685  0.002033  763.61  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3317 on 53918 degrees of freedom
## Multiple R-squared:  0.9154, Adjusted R-squared:  0.9154
## F-statistic: 5.831e+05 on 1 and 53918 DF, p-value: < 2.2e-16

plot_1.yz <- ggplot(diamonds, aes(y, z)) +
  geom_point() +
  geom_smooth(method='lm', formula='y~x')

grid.arrange(plot_1.xy, plot_1.xz, plot_1.yz, nrow=3)

```

Wykresy zależności miedzy wymiarami diamentów



Wszystkie trzy wymiary diamentu są od siebie liniowo zależne. Modele mają istotność statystyczną, a i wariancja jest nimi objaśniana w znacznej części. Na wykresach obserwacje (poza pojedynczymi wyjątkami) układają się wzdłuż linii regresji.

Celem ułatwienia dalszej analizy stworzymy zmienną `volume`, czyli objętość najmniejszego prostopadłościanu, w jakim zmieściłby się dany diament. Zmienną `volume` będziemy dalej nazywać "rozmiarem" diamentu.

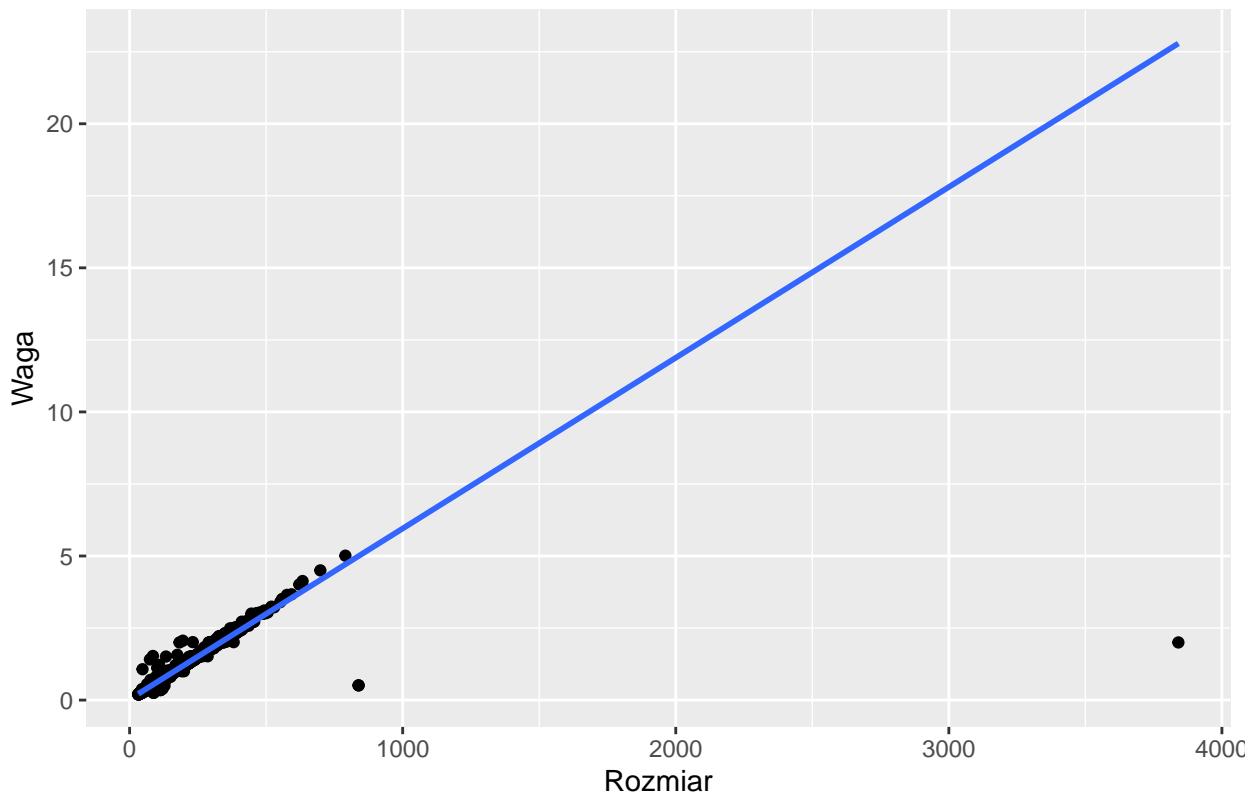
```
iamonds$volume = diamonds$x * diamonds$y * diamonds$z
```

Zależność wagi od rozmiaru

Kolejną zależnością, która wydaje się dość oczywista jest wpływ rozmiaru diamentu na liczbę karatów.

```
ggplot(diamonds, aes(volume, carat)) +  
  geom_point() +  
  geom_smooth(method='lm') +  
  labs(title='Wykres zależności wagi diamentu od jego rozmiaru') +  
  xlab('Rozmiar') +  
  ylab('Waga')
```

Wykres zaleznosci wagi diamentu od jego rozmiaru



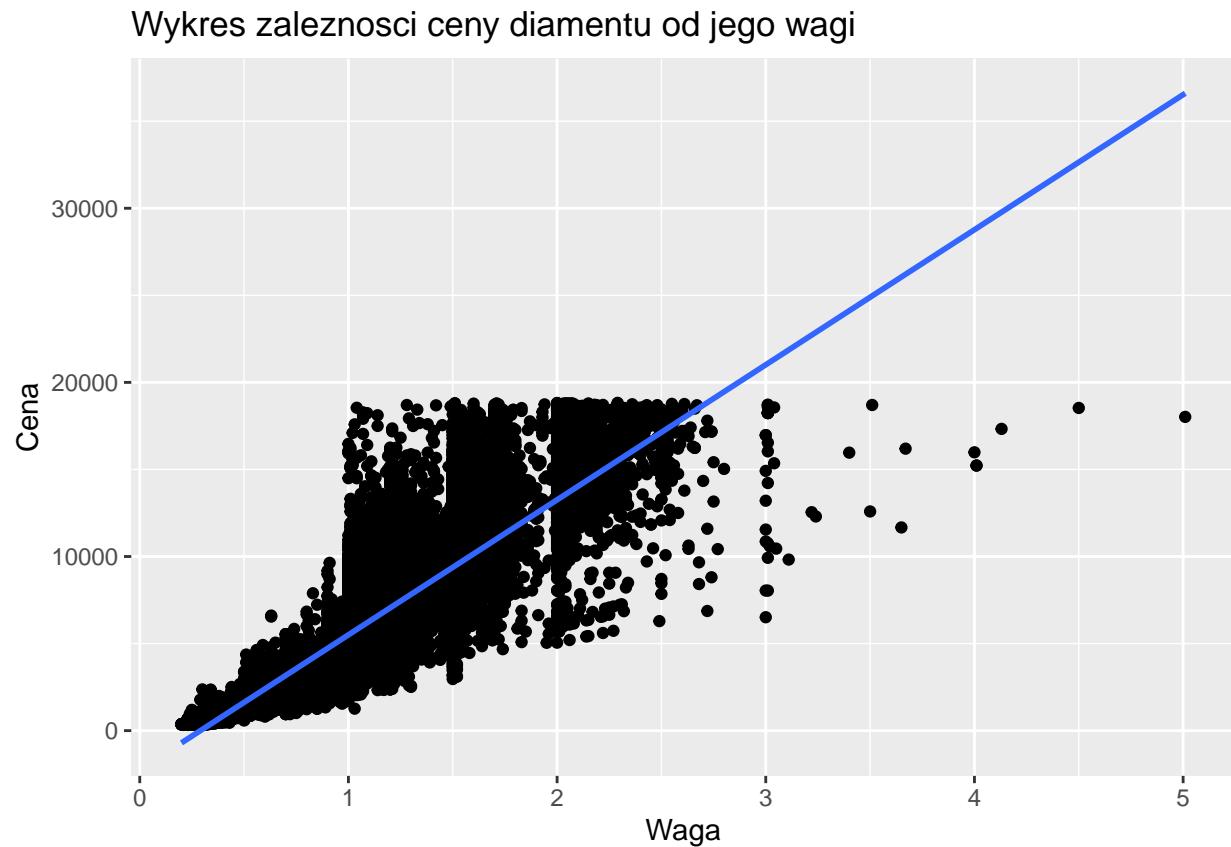
```
model_0.car.vol <- lm(carat ~ volume, diamonds)
summary(model_0.car.vol)

##
## Call:
## lm(formula = carat ~ volume, data = diamonds)
##
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -20.7861  -0.0183  -0.0086   0.0123   0.9968 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 2.797e-02 8.199e-04  34.11   <2e-16 ***
## volume      5.926e-03 5.407e-06 1095.87   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.09821 on 53918 degrees of freedom
## Multiple R-squared:  0.957, Adjusted R-squared:  0.957 
## F-statistic: 1.201e+06 on 1 and 53918 DF,  p-value: < 2.2e-16
```

Wykres i parametry modelu potwierdzają ogromną zależność wagi od rozmiaru diamentu, gdyż R^2 wynosi aż 95,7%.

Zależność ceny od wagi

```
ggplot(diamonds, aes(carat, price)) +  
  geom_point() +  
  geom_smooth(method='lm') +  
  labs(title='Wykres zależności ceny diamentu od jego wagi') +  
  xlab('Waga') +  
  ylab('Cena')
```



Na powyższym wykresie widoczna jest zależność ceny od wagi. Sprawdźmy to jeszcze za pomocą modelu liniowego:

```
model_price_carat <- lm(price ~ carat, diamonds)  
summary(model_price_carat)
```

```
##  
## Call:  
## lm(formula = price ~ carat, data = diamonds)  
##  
## Residuals:  
##      Min       1Q   Median       3Q      Max  
## -18582.6   -804.6    -18.9    537.0  12731.8  
##  
## Coefficients:  
##                 Estimate Std. Error t value Pr(>|t|)  
## (Intercept) -2255.77     13.05 -172.8   <2e-16 ***  
## carat        7755.77    14.07  551.3   <2e-16 ***
```

```

## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1548 on 53918 degrees of freedom
## Multiple R-squared:  0.8493, Adjusted R-squared:  0.8493
## F-statistic: 3.039e+05 on 1 and 53918 DF,  p-value: < 2.2e-16

```

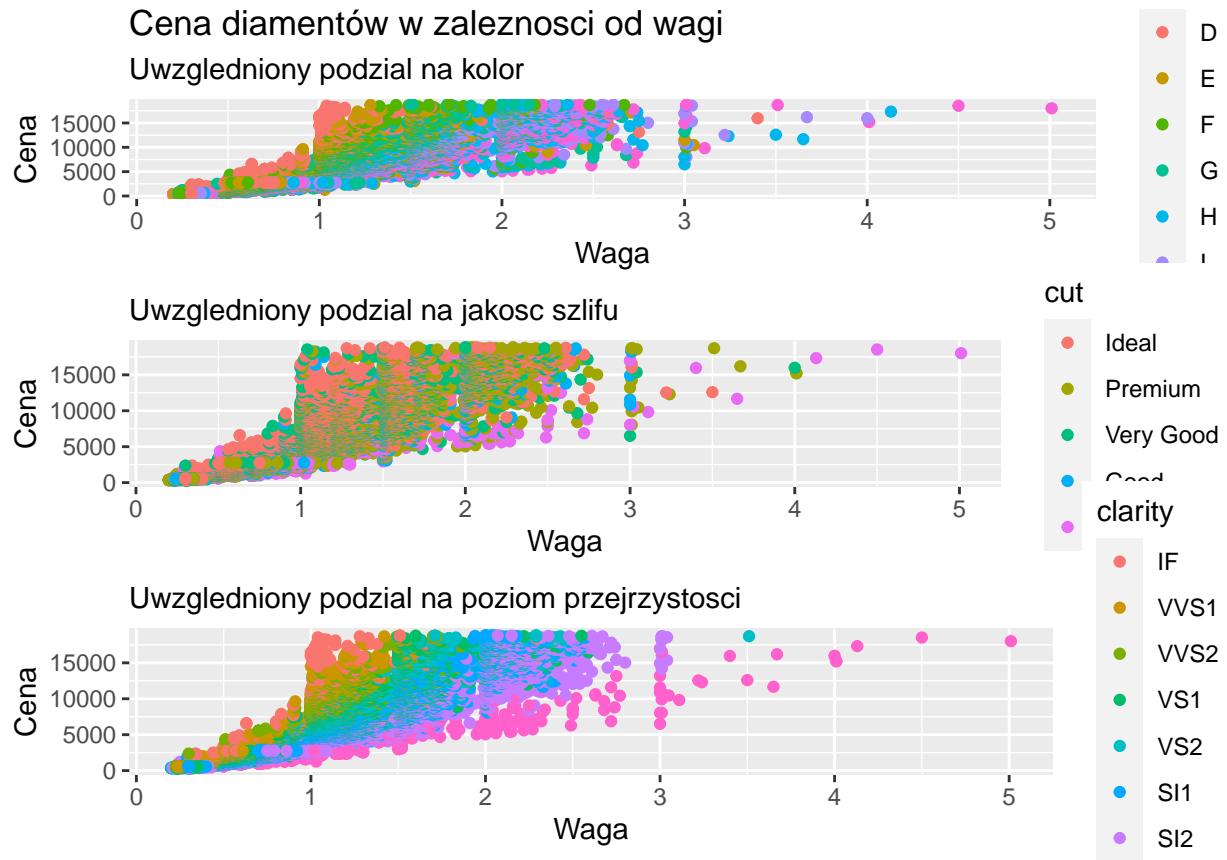
Model jest istotny statystycznie, a do tego objaśnia znaczną część wariancji. Wiemy zatem, że cena jest liniowo zależna od wagi diamentu. Teraz zobaczymy, jak wykres zależności ceny od wagi prezentuje się z podziałem na kolor, jakość szlifu oraz poziom przejrzystości.

```

plot_2_color <- ggplot(diamonds, aes(carat, price, color=color)) +
  labs(title='Cena diamentów w zależności od wagi', subtitle='Uwzględniony podział na kolor') +
  xlab('Waga') +
  ylab('Cena') +
  geom_point()
plot_2_cut <- ggplot(diamonds, aes(carat, price, color=cut)) +
  labs(subtitle='Uwzględniony podział na jakość szlifu') +
  xlab('Waga') +
  ylab('Cena') +
  geom_point()
plot_2_clarity <- ggplot(diamonds, aes(carat, price, color=clarity)) +
  labs(subtitle='Uwzględniony podział na poziom przejrzystości') +
  xlab('Waga') +
  ylab('Cena') +
  geom_point()

grid.arrange(plot_2_color, plot_2_cut, plot_2_clarity, nrow=3)

```



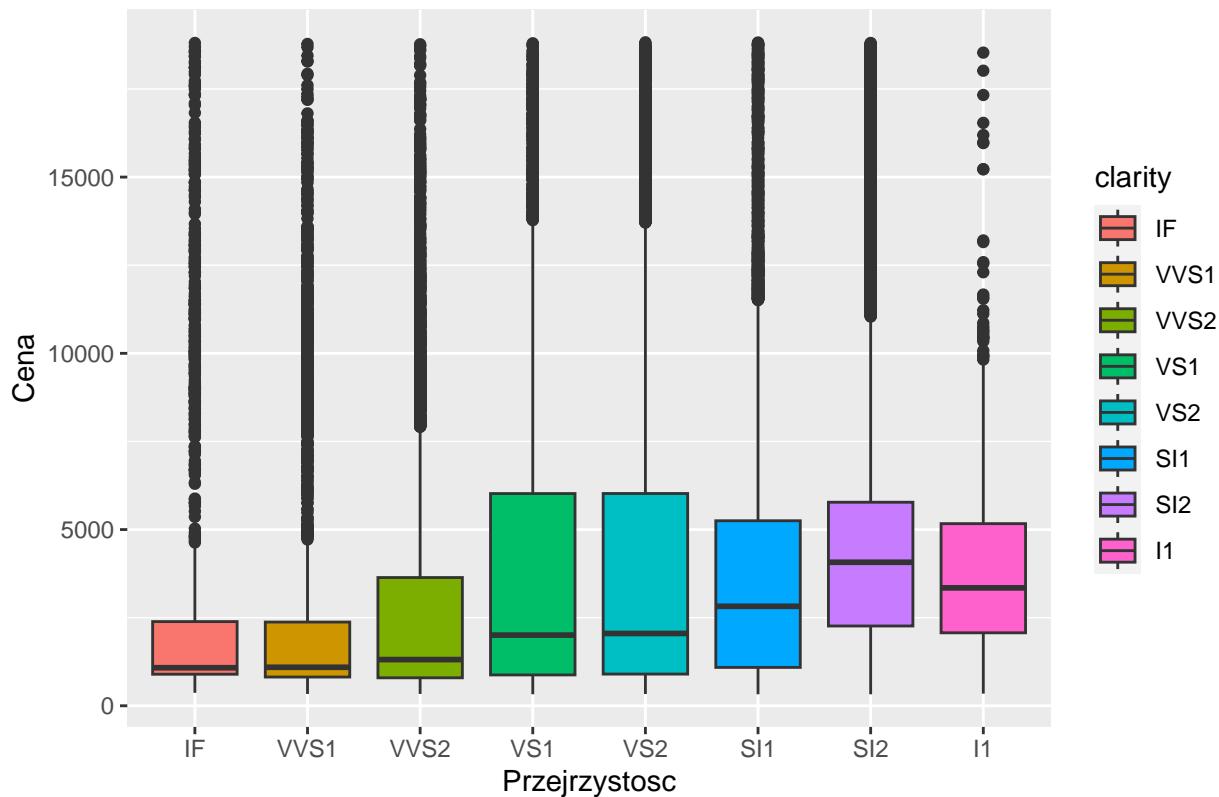
Widać dość wyraźny podział na klasy w zależności od przejrzystości oraz lekki podział ze względu na kolor. Zdaje się, że jakość szlifu nie powinna mieć wpływu na cenę.

Cena diamentu a kolor, poziom przejrzystości i jakość szlifu

Sprawdzimy cenę poszczególnych dla poziomów przejrzystości:

```
ggplot(diamonds, aes(clarity, price, fill=clarity)) +
  labs(title='Wykres pudełkowy ceny w zależności od przejrzystości') +
  xlab('Przejrzystość') +
  ylab('Cena') +
  geom_boxplot()
```

Wykres pudelkowy ceny w zależności od przejrzystości



Jak widać, cena ma pewną zależność od przejrzystości. Problemem może być jednak waga samego diamentu (z wykresu możemy odczytać, że najmniej przejrzyste diamenty mają średnio wyższą cenę), która może być nieproporcjonalna dla różnych klas przejrzystości, a tym samym być główną przyczyną powyższego rozkładu. Zdefiniujmy zatem nową zmienną, która objaśniać będzie cenę diamentu przypadającą na jeden karat jego wagi.

```

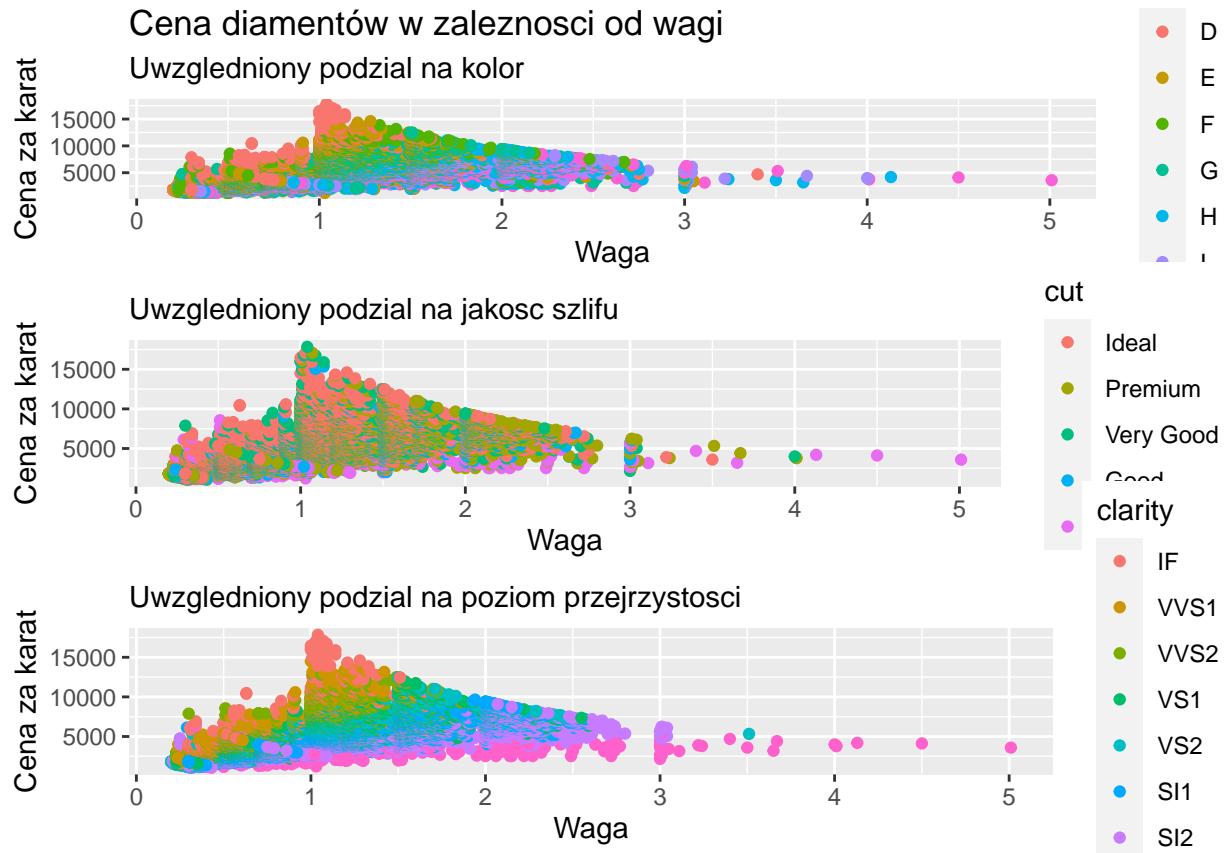
diamonds$price_carat <- diamonds$price / diamonds$carat
y_lab <- 'Cena za karat'

plot_3_color <- ggplot(diamonds, aes(carat, price_carat, color=color)) +
  labs(title='Cena diamentów w zależności od wagi', subtitle='Uwzględniony podział na kolor') +
  xlab('Waga') +
  ylab(y_lab) +
  geom_point()

plot_3_cut <- ggplot(diamonds, aes(carat, price_carat, color=cut)) +
  labs(subtitle='Uwzględniony podział na jakość szlifu') +
  xlab('Waga') +
  ylab(y_lab) +
  geom_point()

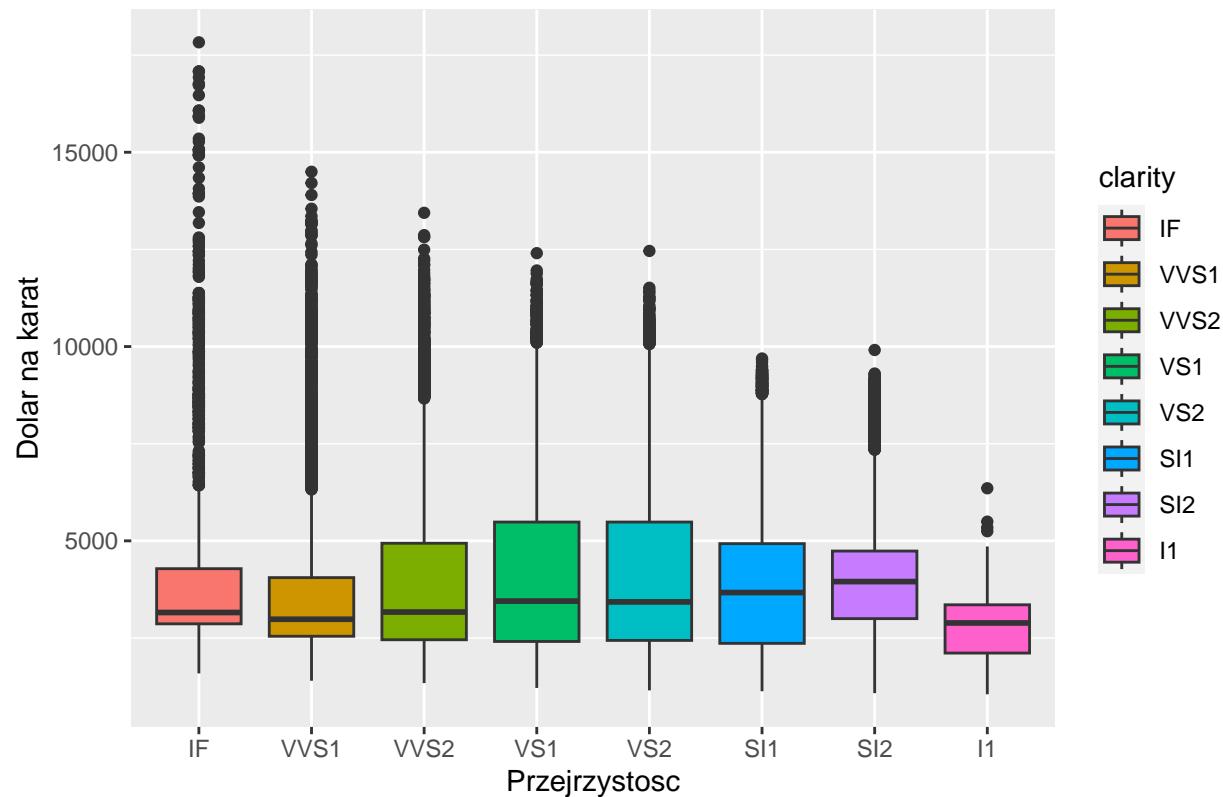
plot_3_clarity <- ggplot(diamonds, aes(carat, price_carat, color=clarity)) +
  labs(subtitle='Uwzględniony podział na poziom przejrzystości') +
  xlab('Waga') +
  ylab(y_lab) +
  geom_point()
  
```

```
grid.arrange(plot_3_color, plot_3_cut, plot_3_clarity, nrow=3)
```



```
diamonds$price_carat <- diamonds$price / diamonds$carat
ggplot(diamonds, aes(clarity, price_carat, fill=clarity)) +
  labs(title='Wykres pudełkowy ceny za karat w zależności od przejrzystości') +
  xlab('Przejrzystość') +
  ylab('Dolar na karat') +
  geom_boxplot()
```

Wykres pudelkowy ceny za karat w zależności od przejrzystości



Cena na karat jednak nie różni się na tyle znacząco, by stwierdzić, iż zależy od przejrzystości. Widać jednak spadek ceny diamentów o najgorszej przejrzystości w porównaniu do poprzedniego wykresu pudelkowego, co wskazuje na ich większą wagę.

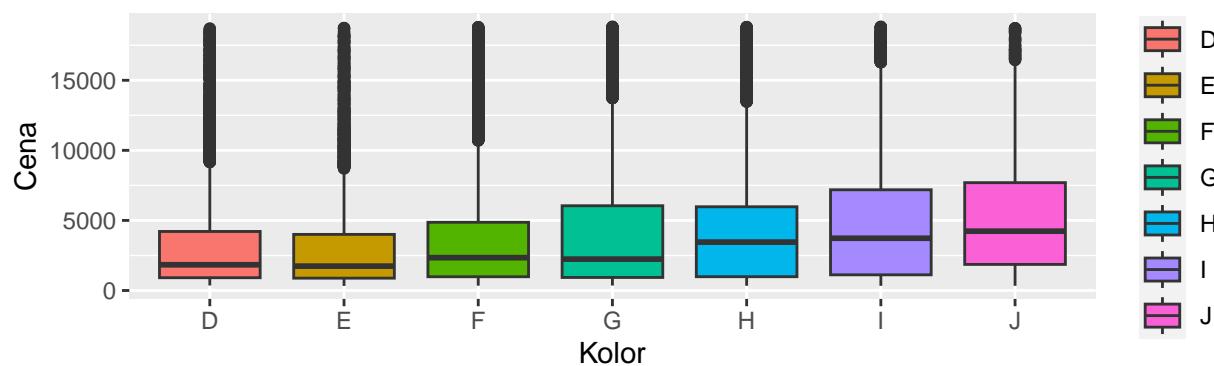
Sprawdźmy, jak wygląda sytuacja dla zmiennej 'color'.

```
boxplot_color_price <- ggplot(diamonds, aes(color, price, fill=color)) +
  labs(title='Wykres pudelkowy ceny w zależności od koloru') +
  xlab('Kolor') +
  ylab('Cena') +
  geom_boxplot()

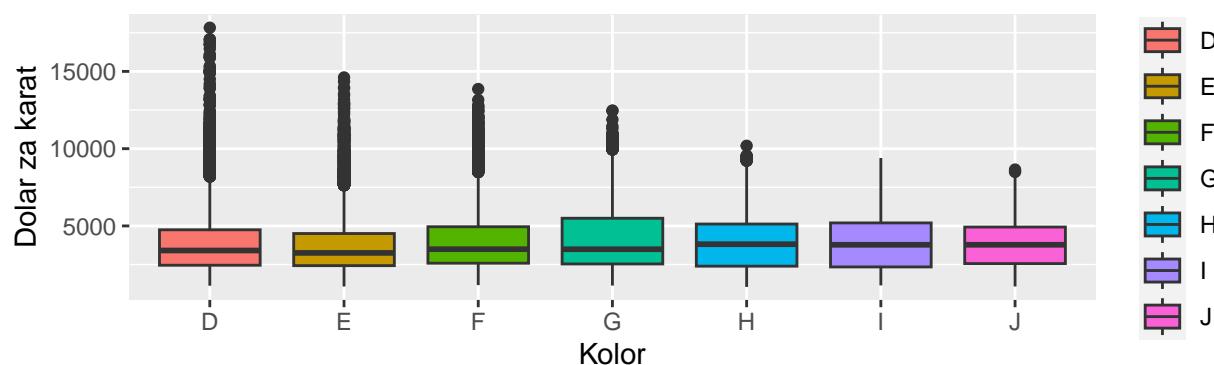
boxplot_color_cprice <- ggplot(diamonds,aes(color, price_carat, fill=color)) +
  labs(title='Wykres pudelkowy ceny za karat w zależności od koloru') +
  xlab('Kolor') +
  ylab('Dolar za karat') +
  geom_boxplot()

grid.arrange(boxplot_color_price, boxplot_color_cprice, nrow=2)
```

Wykres pudelkowy ceny w zależności od koloru



Wykres pudelkowy ceny za karat w zależności od koloru



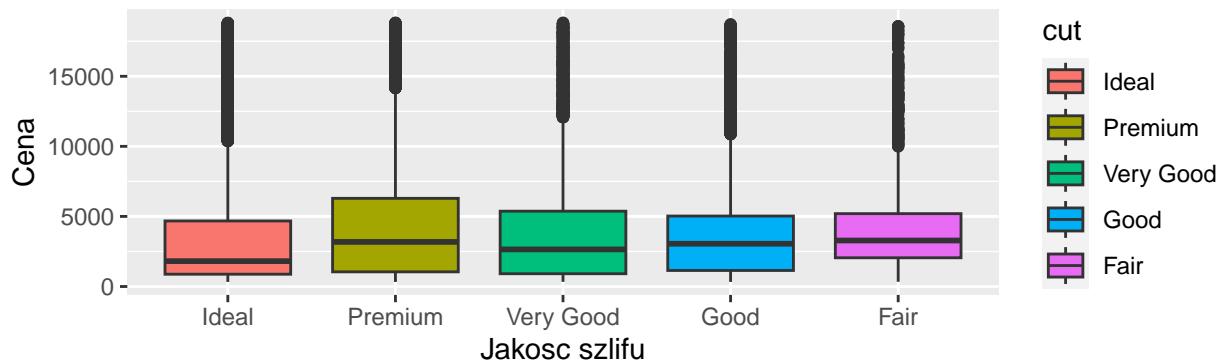
Widać pewną zależność ceny od koloru, jednak brak jakiekolwiek zależności dla ceny za karat od koloru. Ponownie, pewne diamenty zdają się mieć większą wagę, tym razem w zależności od koloru. Sprawdźmy jak wygląda cena przy poszczególnych jakościach szlifu.

```
boxplot_cut_price <- ggplot(diamonds, aes(cut, price, fill=cut)) +
  labs(title='Wykres pudelkowy ceny w zależności od jakości szlifu') +
  xlab('Jakość szlifu') +
  ylab('Cena') +
  geom_boxplot()

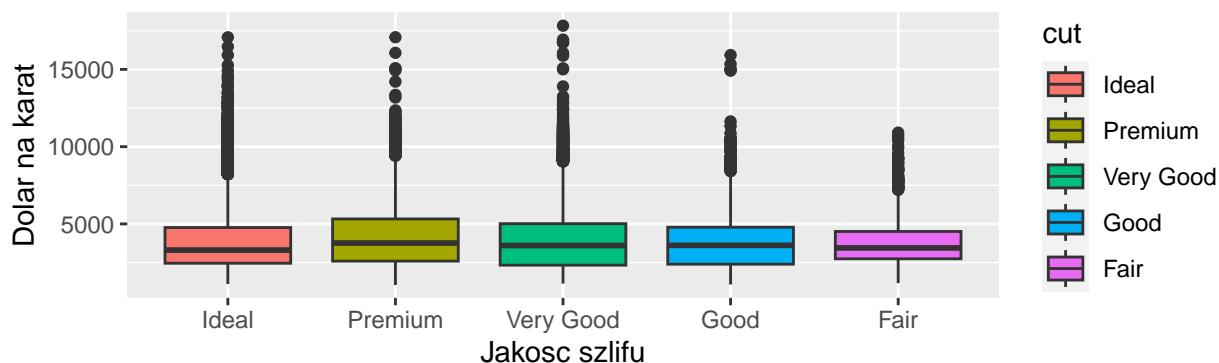
boxplot_cut_cprice <- ggplot(diamonds, aes(cut, price_carat, fill=cut)) +
  labs(title='Wykres pudelkowy ceny za karat w zależności od jakości szlifu') +
  xlab('Jakość szlifu') +
  ylab('Dolar na karat') +
  geom_boxplot()

grid.arrange(boxplot_cut_price,boxplot_cut_cprice,nrow=2)
```

Wykres pudełkowy ceny w zależności od jakości szlifu



Wykres pudełkowy ceny za karat w zależności od jakości szlifu



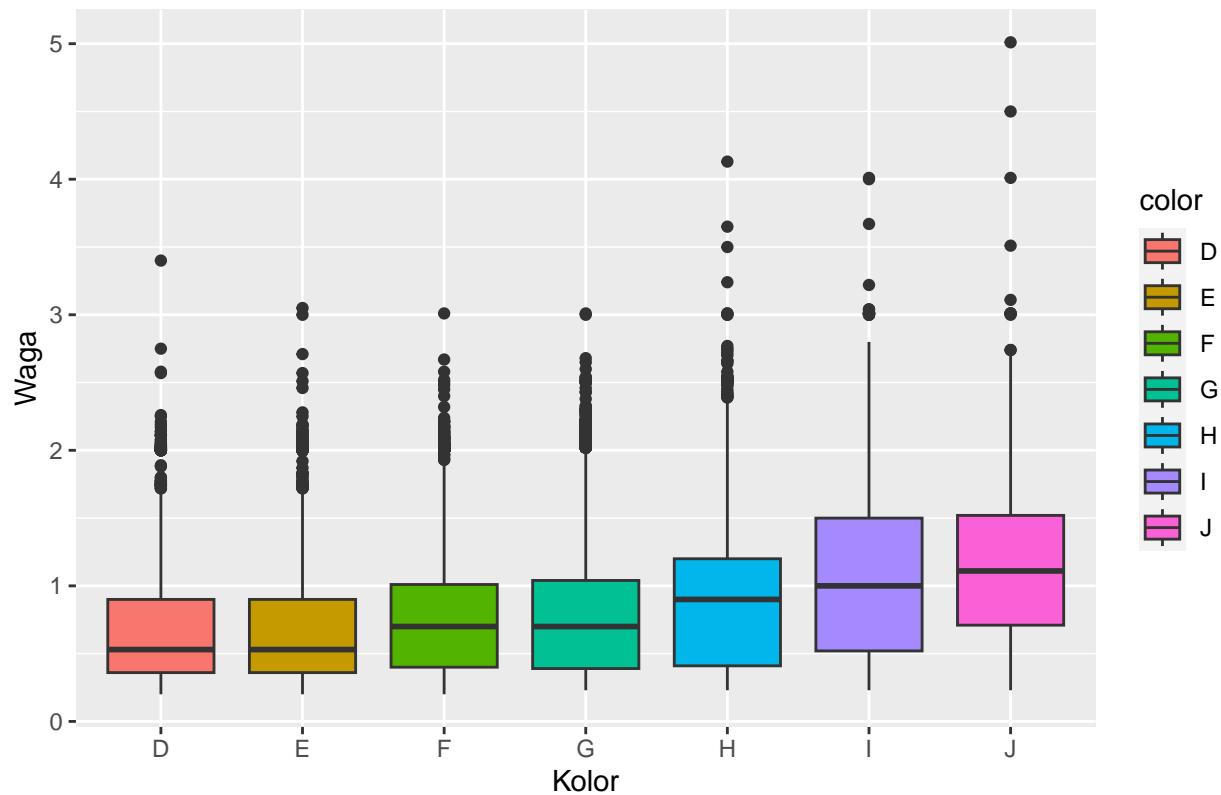
Widać lekkie odchylenie ceny w zależności od jakości szlifu, ale brak widocznej zależności dla ceny za karat.

Waga diamentu a kolor, poziom przejrzystości i jakość szlifu

Powyższe rozważania sugerują nam, że kolor, poziom przejrzystości i jakość szlifu nie mają wpływu na cenę, ale na liniowo od niej zależną wagę diamentu.

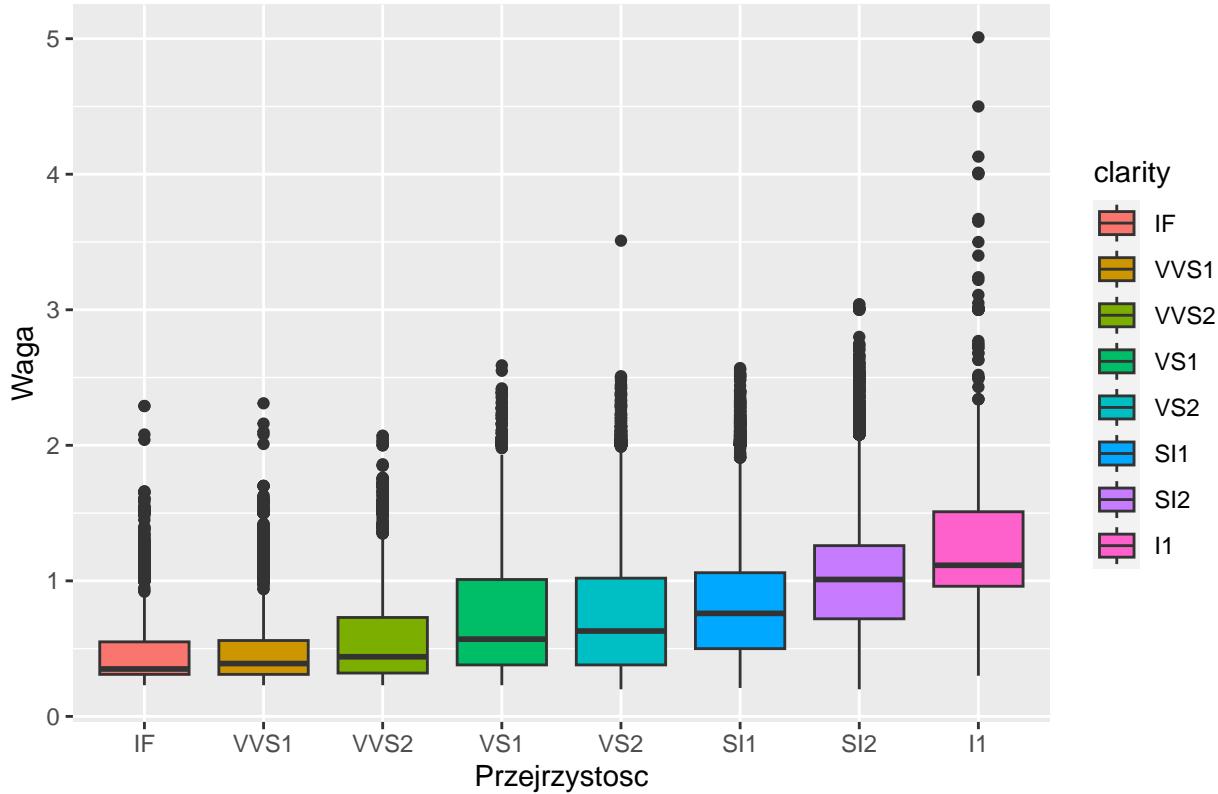
```
ggplot(diamonds, aes(color, carat, fill=color)) +
  labs(title='Wykres pudełkowy wagi diamentu w zależności od koloru') +
  xlab('Kolor') +
  ylab('Waga') +
  geom_boxplot()
```

Wykres pudełkowy wagi diamentu w zależności od koloru



```
ggplot(diamonds, aes(clarity, carat, fill=clarity)) +  
  labs(title='Wykres pudełkowy wagi diamentu w zależności od przejrzystości') +  
  xlab('Przejrzystość') +  
  ylab('Waga') +  
  geom_boxplot()
```

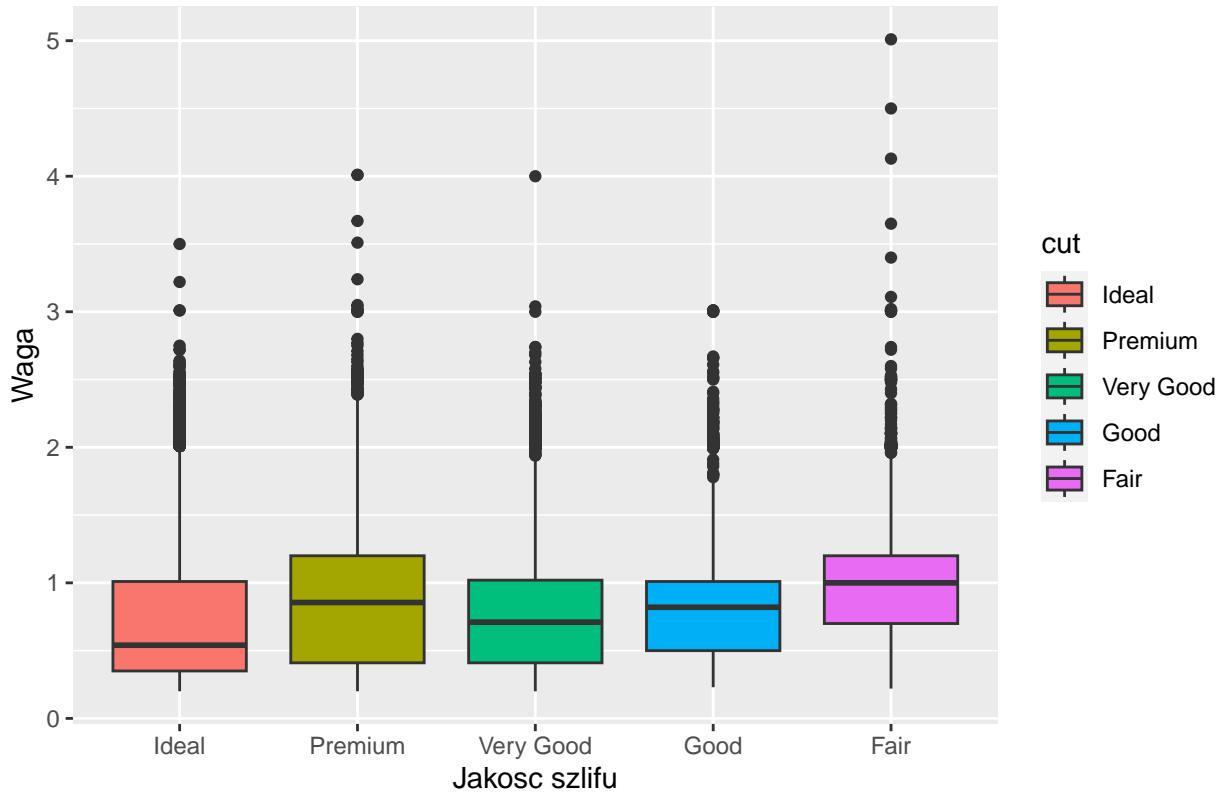
Wykres pudelkowy wagi diamentu w zależności od przejrzystości



Z powyższych wykresów można wywnioskować, że waga diamentów może mieć zależność od koloru i poziomu przejrzystości. Co ciekawe, to diamenty, których cechy (zmienne kategoryczne) oceniono “najgorzej”, na wykresach występują jako najczęstsze. To będzie pierwsza z naszych hipotez badawczych. Modele je weryfikujące zbudujemy w dalszej części pracy.

```
ggplot(diamonds, aes(cut, carat, fill=cut)) +
  labs(title='Wykres pudelkowy wagi diamentu w zależności od jakości szlifu') +
  xlab('Jakość szlifu') +
  ylab('Waga') +
  geom_boxplot()
```

Wykres pudelkowy wagi diamentu w zależności od jakości szlifu



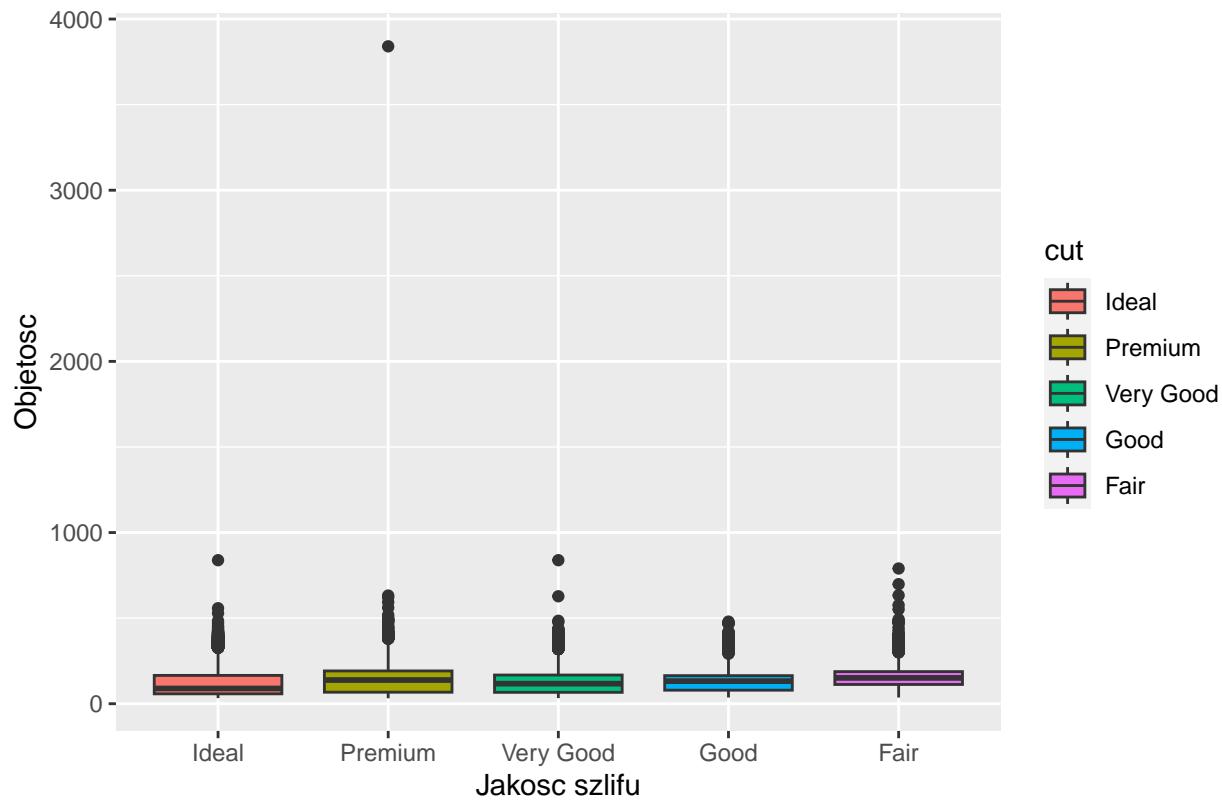
Z wykresu można odczytać, że waga nie zmienia się znacznie dla poszczególnych poziomów jakości szlifu. Widać pewną dysproporcję wagi jedynie dla diamentów o jakości szlifu "Ideal" oraz "Fair", jednak rozkład wagi diamentów ze szlifem o jakości "Premium", "Very good" oraz "Good" raczej przeczy, iż wraz z polepszeniem się szlifu waga maleje.

Jakość szlifu a rozmiar diamentu

Dotychczas jakość szlifu nie wydawała się być istotną cechą wpływającą na cenę czy wagę diamentu. Zobaczmy, jak sprawia się ma w kontekście rozmiaru

```
ggplot(diamonds, aes(cut, volume, fill=cut)) +
  labs(title='Wykres pudelkowy objętości w zależności od jakości szlifu') +
  xlab('Jakość szlifu') +
  ylab('Objętość') +
  geom_boxplot()
```

Wykres pudełkowy objętości w zależności od jakości szlifu



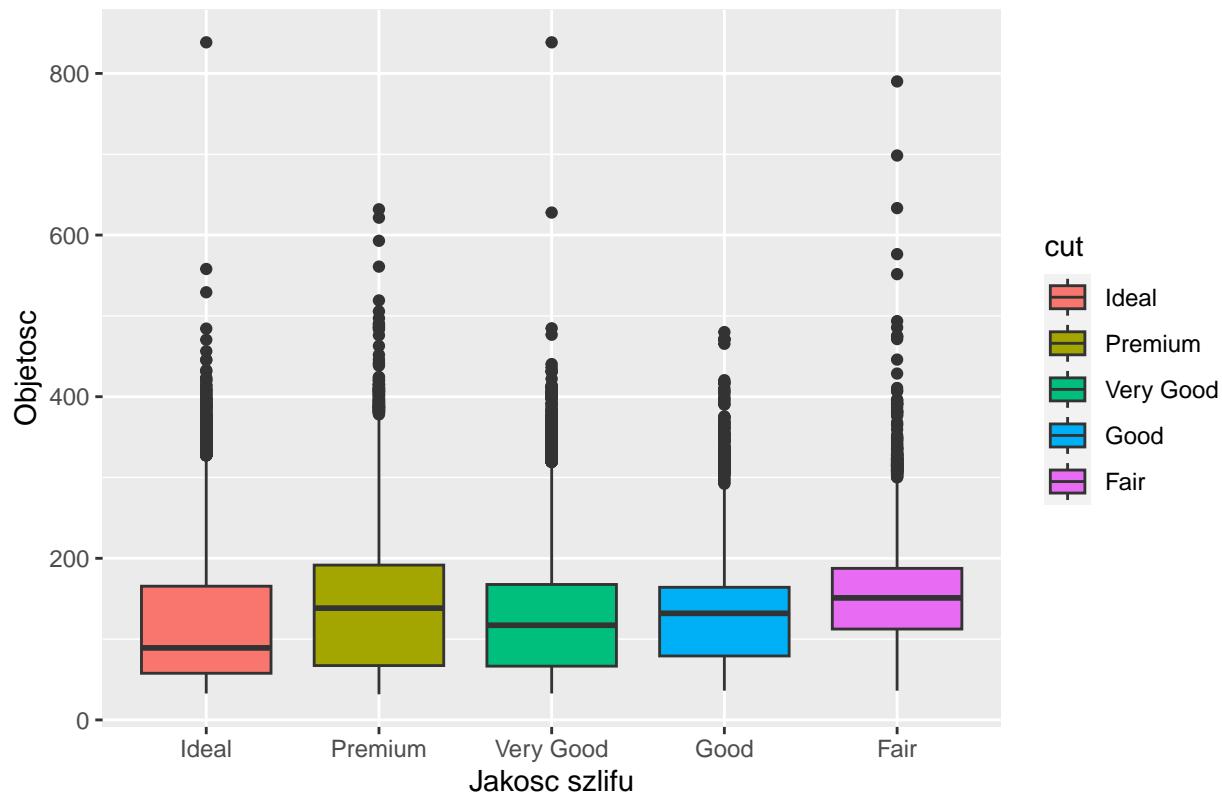
Jedna obserwacja znacznie odstaje od reszty i utrudnia odczytywanie danych z wykresu. W naszych dalszych badaniach ją pominiemy.

```
diamonds <- diamonds[-which(diamonds$volume==max(diamonds$volume)),]
```

Ponownie spójrzmy na ten wykres, teraz powinniśmy móc lepiej odczytać wyniki

```
ggplot(diamonds, aes(cut, volume, fill=cut)) +
  labs(title='Wykres pudełkowy objętości w zależności od jakości szlifu ') +
  xlab('Jakość szlifu') +
  ylab('Objętość') +
  geom_boxplot()
```

Wykres pudelkowy objetosci w zaleznosci od jakosci szlifu



Również rozmiar zdaje się nie wykazywać liniowej zależności względem jakości szlifu. Ponownie, największa dysproporcja wyników jest między najlepszą a najgorszą jakością szlifu.

Hipoteza 1: Im gorszy kolor i poziom przejrzystości diamentu, tym większa jego waga

```
model_1_color <- aov(carat ~ color, diamonds)
print(summary.lm(model_1_color))

##
## Call:
## aov(formula = carat ~ color, data = diamonds)
##
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -0.9321 -0.3477 -0.0879  0.2735  3.8479 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 0.6577148  0.0054808 120.003 <2e-16 ***
## colorE      0.0001519  0.0071281   0.021   0.983    
## colorF      0.0787656  0.0071675  10.989 <2e-16 ***
## colorG      0.1130146  0.0069334  16.300 <2e-16 ***
## colorH      0.2533278  0.0073868  34.295 <2e-16 ***
## colorI      0.3689869  0.0082204  44.886 <2e-16 ***
```

```

## colorJ      0.5044220  0.0101245  49.822   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4511 on 53912 degrees of freedom
## Multiple R-squared:  0.09354, Adjusted R-squared:  0.09344
## F-statistic: 927.2 on 6 and 53912 DF, p-value: < 2.2e-16
model_1_clarity <- aov(carat ~ clarity, diamonds)
print(summary.lm(model_1_clarity))

```

```

##
## Call:
## aov(formula = carat ~ clarity, data = diamonds)
##
## Residuals:
##     Min      1Q  Median      3Q      Max
## -0.9841 -0.3271 -0.0962  0.2363  3.7259
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.505123  0.010403 48.556   < 2e-16 ***
## clarityVVS1 -0.001992  0.012698 -0.157    0.875
## clarityVVS2  0.091079  0.012102  7.526 5.31e-14 ***
## clarityVS1   0.221985  0.011486 19.326   < 2e-16 ***
## clarityVS2   0.258593  0.011137 23.220   < 2e-16 ***
## claritySI1   0.345149  0.011093 31.115   < 2e-16 ***
## claritySI2   0.572084  0.011372 50.308   < 2e-16 ***
## clarityII1   0.779023  0.019254 40.461   < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4401 on 53911 degrees of freedom
## Multiple R-squared:  0.1371, Adjusted R-squared:  0.137
## F-statistic: 1224 on 7 and 53911 DF, p-value: < 2.2e-16

```

Modele zdają się wskazywać na pewne powiązanie wagi z kolorem oraz przejrzystością. Ponadto widzimy stopniowy wzrost współczynnika przy odpowiednio coraz gorszej przejrzystości. Sam kolor jest w stanie wyjaśnić ~ 9 % całkowitej wariancji, natomiast przejrzystość ~ 14 %. Sprawdźmy, jaki ułamek wariancji potrafi przewidzieć model uwzględniający obie zmienne:

```

model_1_both <- aov(carat ~ color + clarity, diamonds)
print(summary.lm(model_1_both))

```

```

##
## Call:
## aov(formula = carat ~ color + clarity, data = diamonds)
##
## Residuals:
##     Min      1Q  Median      3Q      Max
## -1.1213 -0.2937 -0.0808  0.2237  3.4196
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.317849  0.011172 28.451   < 2e-16 ***
## colorE      0.025615  0.006589  3.888  0.000101 ***

```

```

## colorF      0.116416  0.006641  17.529  < 2e-16 ***
## colorG      0.180825  0.006474  27.931  < 2e-16 ***
## colorH      0.271051  0.006835  39.656  < 2e-16 ***
## colorI      0.392257  0.007605  51.581  < 2e-16 ***
## colorJ      0.511251  0.009359  54.625  < 2e-16 ***
## clarityVVS1 0.016003  0.012025   1.331  0.183282
## clarityVVS2 0.125418  0.011470  10.935  < 2e-16 ***
## clarityVS1  0.219400  0.010889  20.148  < 2e-16 ***
## clarityVS2  0.280974  0.010589  26.535  < 2e-16 ***
## claritySI1  0.362054  0.010563  34.274  < 2e-16 ***
## claritySI2  0.592158  0.010815  54.755  < 2e-16 ***
## clarityII   0.761268  0.018232  41.753  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4163 on 53905 degrees of freedom
## Multiple R-squared:  0.2281, Adjusted R-squared:  0.2279
## F-statistic:  1225 on 13 and 53905 DF,  p-value: < 2.2e-16

```

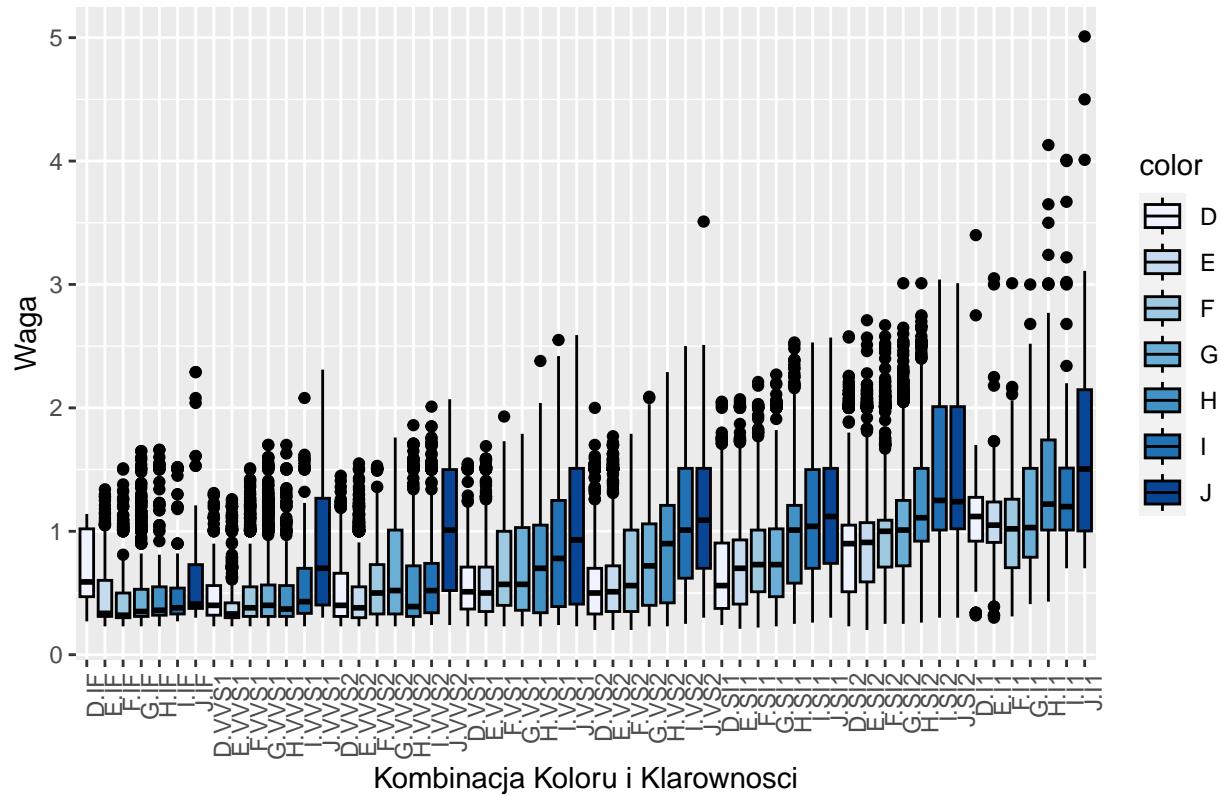
Model jest w stanie wyznaczyć ~23% wariancji

```

ggplot(diamonds, aes(interaction(color, clarity), carat, fill=color)) +
  geom_boxplot(color = 'black') +
  xlab('Kombinacja Koloru i Klarownosci') +
  ylab('Waga') +
  labs(title='Wykres pudełkowy z interakcjami dla wagi diamentu') +
  theme(axis.text.x = element_text(angle=90, hjust=1)) +
  scale_fill_brewer(palette='Blues')

```

Wykres pudelkowy z interakcjami dla wagi diamentu



Jak widać na powyższym wykresie, zarówno kolor, jak i przejrzystość mają dość istotny wpływ na wagę. Co więcej, stopniowe pogarszanie klarowności wskazuje jednoznacznie na większą wagę, a pewne kolory diamentów wykazują większą wagę od pozostałych.

Hipotezę przyjmujemy za prawdziwą.

Hipoteza 2: Jakość szlifu nie wpływa na rozmiar diamentów

```
model_2_cut <- aov(volume ~ cut, diamonds)
print(summary.lm(model_2_cut))

##
## Call:
## aov(formula = volume ~ cut, data = diamonds)
##
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -128.89  -61.15  -18.45   42.71  723.09 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 115.4110    0.5135 224.77 <2e-16 ***
## cutPremium  29.4887    0.8222  35.87 <2e-16 ***
## cutVery Good 15.5996    0.8567  18.21 <2e-16 ***
## cutGood     20.9575    1.1927  17.57 <2e-16 ***
## cutFair     49.6421    1.9479  25.48 <2e-16 ***
```

```

## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 75.37 on 53914 degrees of freedom
## Multiple R-squared:  0.0311, Adjusted R-squared:  0.03103
## F-statistic: 432.7 on 4 and 53914 DF,  p-value: < 2.2e-16

```

Model wykazuje istotne różnice między poszczególnymi jakościąmi cięcia. Wartość R^2 jest natomiast bardzo mała - wynosi około 3%, co może sugerować, że inne zmienne wpływają na zmienną rozmiar diamentu.

Posortujmy szlif według współczynnika malejaco:

Fair

Premium

Good

Very Good

Ideal

Sprawdzmy, czy któraś zmienna wpływa na jakość szlifu, tzn. czy dana jakość szlifu jest powiązana z innymi zmiennymi, które w konsekwencji mogą mieć wpływ na rozmiar.

```

ggplot(diamonds,aes(carat, cut, fill=cut)) +
  labs(title='Wykres pudełkowy wagi w zależności od jakości szlifu') +
  xlab('Waga') +
  ylab('Jakość szlifu') +
  geom_boxplot()

```



```

model_2_carat_cut <- aov(carat ~ cut, diamonds)
print(summary.lm(model_2_carat_cut))

##
## Call:
## aov(formula = carat ~ cut, data = diamonds)
##
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -0.8261 -0.3764 -0.1064  0.2587  3.9639 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 0.702759  0.003170 221.69   <2e-16 ***
## cutPremium  0.188569  0.005076  37.15   <2e-16 ***
## cutVery Good 0.103606  0.005289  19.59   <2e-16 ***
## cutGood     0.145951  0.007364  19.82   <2e-16 ***
## cutFair     0.343319  0.012026  28.55   <2e-16 ***
## ---      
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4653 on 53914 degrees of freedom
## Multiple R-squared:  0.03536,    Adjusted R-squared:  0.03529 
## F-statistic: 494.1 on 4 and 53914 DF,  p-value: < 2.2e-16

```

Mamy tę samą kolejność:

Fair

Premium

Good

Very good

Ideal

Waga wydaje się być skorelowana z jakością szlifu, sprawdźmy, czy jest skorelowana modelem ANOVA.

```

carat_cut_ANOVA <- aov(carat ~ cut, diamonds)
summary.lm(carat_cut_ANOVA)

```

```

##
## Call:
## aov(formula = carat ~ cut, data = diamonds)
##
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -0.8261 -0.3764 -0.1064  0.2587  3.9639 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 0.702759  0.003170 221.69   <2e-16 ***
## cutPremium  0.188569  0.005076  37.15   <2e-16 ***
## cutVery Good 0.103606  0.005289  19.59   <2e-16 ***
## cutGood     0.145951  0.007364  19.82   <2e-16 ***
## cutFair     0.343319  0.012026  28.55   <2e-16 ***
## ---      

```

```

## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4653 on 53914 degrees of freedom
## Multiple R-squared: 0.03536, Adjusted R-squared: 0.03529
## F-statistic: 494.1 on 4 and 53914 DF, p-value: < 2.2e-16

```

R^2 na poziomie ~0.035 wskazuje, że cięcie może nie być zbyt istotnym czynnikiem wpływającym na wagę diamentu.

Być może kolor jest powiązany z jakością cięcia. Sprawdźmy powiązanie zmiennych przez test Chi-kwadrat Pearsona.

```
table(diamonds$cut, diamonds$color)
```

```

##
##          D      E      F      G      H      I      J
## Ideal     2834  3903  3825  4882  3115  2093   896
## Premium   1602  2337  2330  2921  2354  1427   808
## Very Good 1513  2400  2164  2299  1823  1204   678
## Good      662   933   907   869   702   522   307
## Fair       163   224   312   313   303   175   119

```

```
chisq.test(diamonds$cut, diamonds$color)
```

```

##
## Pearson's Chi-squared test
##
## data: diamonds$cut and diamonds$color
## X-squared = 309.49, df = 24, p-value < 2.2e-16

```

P-value < 0.05. Jest pewne powiązanie, sprawdźmy więc model.

```
color_volume_ANOVA <- aov(volume ~ color, diamonds)
summary.lm(color_volume_ANOVA)
```

```

##
## Call:
## aov(formula = volume ~ color, data = diamonds)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -150.78 -55.78 -14.61  44.37 731.06
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 107.2067    0.8863 120.97 <2e-16 ***
## colorE       0.2311    1.1526   0.20   0.841
## colorF      12.7667    1.1590  11.02 <2e-16 ***
## colorG      18.3333    1.1212  16.35 <2e-16 ***
## colorH      40.7352    1.1945  34.10 <2e-16 ***
## colorI      59.5194    1.3293  44.78 <2e-16 ***
## colorJ      81.2730    1.6372  49.64 <2e-16 ***
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 72.94 on 53912 degrees of freedom
## Multiple R-squared: 0.0926, Adjusted R-squared: 0.0925
## F-statistic: 917 on 6 and 53912 DF, p-value: < 2.2e-16

```

Kolor wyjaśnia ponad trzykrotnie więcej wariancji, niż jakość szlifu.

```
table(diamonds$cut, diamonds$color)

##
##          D      E      F      G      H      I      J
## Ideal     2834  3903  3825  4882  3115  2093  896
## Premium   1602  2337  2330  2921  2354  1427  808
## Very Good 1513  2400  2164  2299  1823  1204  678
## Good      662   933   907   869   702   522   307
## Fair       163   224   312   313   303   175   119

chisq.test(diamonds$cut, diamonds$color)

##
## Pearson's Chi-squared test
##
## data: diamonds$cut and diamonds$color
## X-squared = 309.49, df = 24, p-value < 2.2e-16

clarity_volume_ANOVA <- aov(volume ~ clarity, diamonds)
summary.lm(clarity_volume_ANOVA)

##
## Call:
## aov(formula = volume ~ clarity, data = diamonds)
##
## Residuals:
##      Min      1Q Median      3Q      Max
## -155.02  -52.68 -16.40   37.70  719.64
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 83.2592    1.6847  49.420 < 2e-16 ***
## clarityVVS1 -0.6873    2.0564  -0.334   0.738
## clarityVVS2 14.3584    1.9599   7.326  2.4e-13 ***
## clarityVS1  35.5995    1.8602  19.138 < 2e-16 ***
## clarityVS2  41.1068    1.8036  22.792 < 2e-16 ***
## claritySI1  54.8497    1.7965  30.532 < 2e-16 ***
## claritySI2  91.4171    1.8416  49.640 < 2e-16 ***
## clarityI1   122.0986   3.1181  39.158 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 71.28 on 53911 degrees of freedom
## Multiple R-squared:  0.1336, Adjusted R-squared:  0.1335
## F-statistic: 1187 on 7 and 53911 DF,  p-value: < 2.2e-16
```

Jakość szlifu jest powiązana z przejrzystością. Przejrzystość wyjaśnia ponad 13% wariancji.

Sprawdźmy, czy dodając jakość szlifu do modelu istotnie zmieni się R^2 :

```
ANOVA_1 <- aov(volume ~ color + cut, diamonds)
ANOVA_2 <- aov(volume ~ clarity + cut, diamonds)
ANOVA_3 <- aov(volume ~ color + clarity, diamonds)
ANOVA_4 <- aov(volume ~ color + clarity + cut, diamonds)

summary.lm(ANOVA_1)
```

```

##
## Call:
## aov(formula = volume ~ color + cut, data = diamonds)
##
## Residuals:
##      Min     1Q Median     3Q    Max 
## -170.18 -53.58 -12.89  41.79 744.03 
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 94.50896   0.94799 99.694 <2e-16 ***
## colorE      -0.03941   1.13610 -0.035   0.972    
## colorF      12.16528   1.14240 10.649 <2e-16 ***
## colorG      18.25598   1.10535 16.516 <2e-16 ***
## colorH      39.20414   1.17803 33.280 <2e-16 ***
## colorI      58.47041   1.31033 44.623 <2e-16 ***
## colorJ      78.56382   1.61495 48.648 <2e-16 ***
## cutPremium  27.11026   0.78504 34.534 <2e-16 ***
## cutVery Good 14.70413   0.81782 17.980 <2e-16 ***
## cutGood     19.61887   1.13850 17.232 <2e-16 ***
## cutFair     45.08660   1.85936 24.248 <2e-16 ***
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 71.89 on 53908 degrees of freedom
## Multiple R-squared:  0.1187, Adjusted R-squared:  0.1186 
## F-statistic: 726.2 on 10 and 53908 DF, p-value: < 2.2e-16
summary.lm(ANOVA_2)

##
## Call:
## aov(formula = volume ~ clarity + cut, data = diamonds)
##
## Residuals:
##      Min     1Q Median     3Q    Max 
## -159.59 -51.96 -16.95  36.85 727.31 
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 79.0530   1.6876 46.843 < 2e-16 ***
## clarityVVS1 -2.1165   2.0458 -1.035   0.301    
## clarityVVS2 12.3501   1.9514  6.329 2.49e-10 ***
## clarityVS1  32.1403   1.8556 17.321 < 2e-16 ***
## clarityVS2  37.1159   1.8012 20.606 < 2e-16 ***
## claritySI1  49.9680   1.8002 27.757 < 2e-16 ***
## claritySI2  85.3855   1.8495 46.168 < 2e-16 ***
## clarityI1   112.2461   3.1530 35.600 < 2e-16 ***
## cutPremium  18.6341   0.7852 23.732 < 2e-16 ***
## cutVery Good 9.1303   0.8110 11.258 < 2e-16 ***
## cutGood     8.1674   1.1337  7.204 5.92e-13 ***
## cutFair     24.0464   1.8737 12.834 < 2e-16 ***
##
## ---
```

```

## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 70.86 on 53907 degrees of freedom
## Multiple R-squared: 0.1437, Adjusted R-squared: 0.1436
## F-statistic: 822.7 on 11 and 53907 DF, p-value: < 2.2e-16
summary.lm(ANOVA_3)

##
## Call:
## aov(formula = volume ~ color + clarity, data = diamonds)
##
## Residuals:
##      Min     1Q   Median     3Q    Max 
## -179.43 -47.38 -13.23  36.28 745.96 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 53.066    1.811   29.307 < 2e-16 ***
## colorE       4.304    1.068   4.031 5.57e-05 ***
## colorF       18.790   1.076  17.456 < 2e-16 ***
## colorG       29.156   1.049  27.787 < 2e-16 ***
## colorH       43.585   1.108  39.343 < 2e-16 ***
## colorI       63.244   1.233  51.311 < 2e-16 ***
## colorJ       82.363   1.517  54.296 < 2e-16 ***
## clarityVVS1  2.197   1.949   1.127   0.26    
## clarityVVS2 19.871   1.859  10.689 < 2e-16 ***
## clarityVS1  35.172   1.765  19.929 < 2e-16 ***
## clarityVS2  44.695   1.716  26.043 < 2e-16 ***
## claritySI1  57.562   1.712  33.621 < 2e-16 ***
## claritySI2  94.639   1.753  53.993 < 2e-16 ***
## clarityII1 119.236   2.955  40.350 < 2e-16 ***
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 67.47 on 53905 degrees of freedom
## Multiple R-squared: 0.2237, Adjusted R-squared: 0.2236
## F-statistic: 1195 on 13 and 53905 DF, p-value: < 2.2e-16
summary.lm(ANOVA_4)

##
## Call:
## aov(formula = volume ~ color + clarity + cut, data = diamonds)
##
## Residuals:
##      Min     1Q   Median     3Q    Max 
## -185.90 -46.22 -12.41  35.33 752.37 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 50.0823   1.8103  27.665 < 2e-16 ***
## colorE       3.9252   1.0630   3.693 0.000222 ***
## colorF       18.1462  1.0716  16.934 < 2e-16 ***
## colorG       28.4281  1.0448  27.210 < 2e-16 ***

```

```

## colorH      42.5340   1.1034  38.547 < 2e-16 ***
## colorI      62.4886   1.2270  50.928 < 2e-16 ***
## colorJ      80.9121   1.5110  53.548 < 2e-16 ***
## clarityVVS1 0.8703    1.9409   0.448  0.653857
## clarityVVS2 17.9957   1.8528   9.713 < 2e-16 ***
## clarityVS1  32.1199   1.7623  18.227 < 2e-16 ***
## clarityVS2  41.1192   1.7157  23.966 < 2e-16 ***
## claritySI1  53.2137   1.7174  30.986 < 2e-16 ***
## claritySI2  89.3163   1.7622  50.686 < 2e-16 ***
## clarityI1   110.9770  2.9909  37.105 < 2e-16 ***
## cutPremium  16.2371   0.7448  21.800 < 2e-16 ***
## cutVery Good 8.3018    0.7689  10.797 < 2e-16 ***
## cutGood     7.0698    1.0746   6.579  4.79e-11 ***
## cutFair     19.9146   1.7765  11.210 < 2e-16 ***
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 67.14 on 53901 degrees of freedom
## Multiple R-squared:  0.2313, Adjusted R-squared:  0.2311
## F-statistic: 954.2 on 17 and 53901 DF,  p-value: < 2.2e-16

```

Okazuje się, że model wykorzystując jedynie kolor oraz przejrzystość diamentu jest w stanie uzyskać R^2 na poziomie ~22.3%, dodanie jakości szlifu do modelu polepsza model o jedyne ~0.8 %, co wskazuje na dość istotny wpływ omówionych zmiennych na zmienną ‘cut’.

Wniosek: Hipotezę przyjmujemy za prawdziwą.

Hipoteza 3: Zmienna *table* wpływa na *depth*

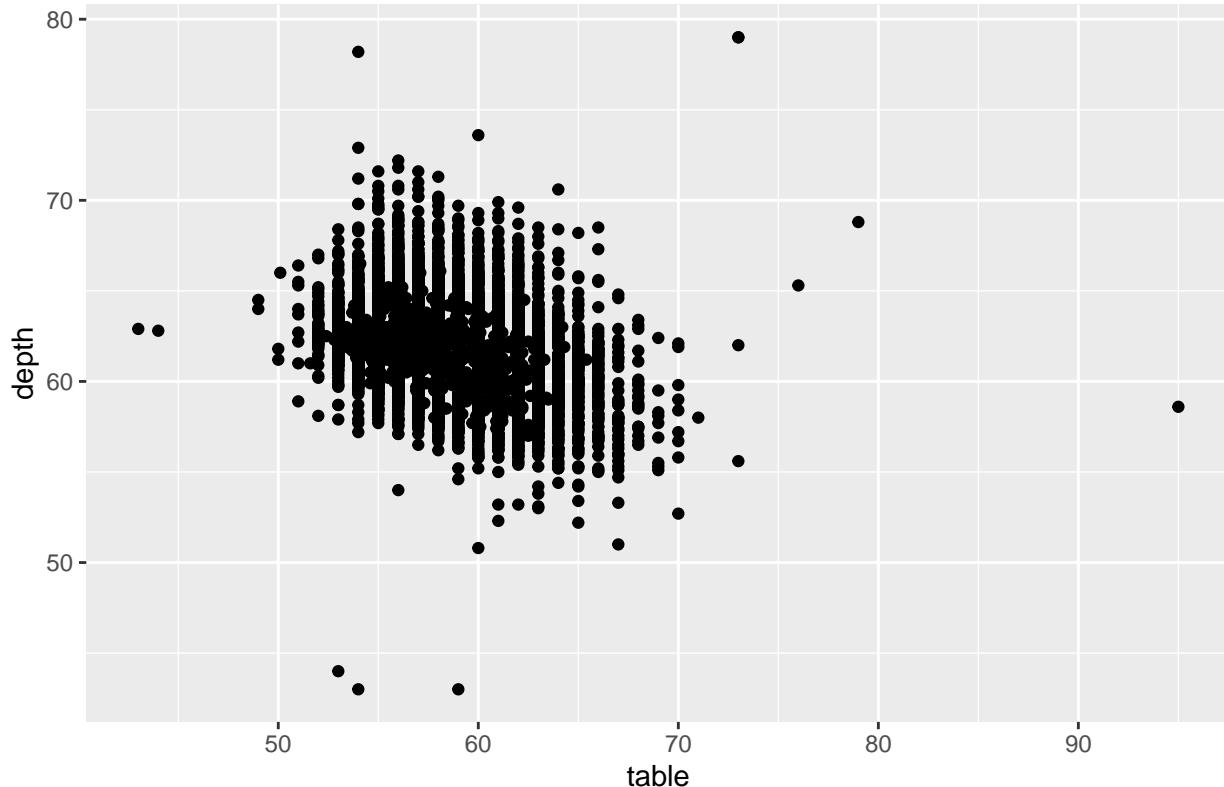
Bardzo duży wpływ na wygląd diamentu mają proporcje wielkości opisanych w naszych danych zmiennymi *table* oraz *depth*. Sprawdzimy, czy jedna wpływa na drugą.

```

ggplot(diamonds, aes(table, depth)) +
  labs(title='Zależność zmiennej depth od table') +
  geom_point()

```

Zależność zmiennej depth od table



```
model_3 <- lm(depth ~ table, diamonds)
summary.lm(model_3)
```

```
##
## Call:
## lm(formula = depth ~ table, data = diamonds)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -19.4050  -0.7362  -0.0570   0.7638  20.1976
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 72.643919   0.151657 479.00  <2e-16 ***
## table      -0.189609   0.002638 -71.89  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.368 on 53917 degrees of freedom
## Multiple R-squared:  0.08747,    Adjusted R-squared:  0.08745
## F-statistic: 5168 on 1 and 53917 DF,  p-value: < 2.2e-16
```

Dla takiego modelu zachodzi istotność statystyczna, jednakże R^2 wynosi zaledwie niecałe 9%. Spróbujmy znaleźć zmienną towarzyszącą, która mogłaby ten parametr poprawić.

```
plot_4_color <- ggplot(diamonds, aes(table, depth, color=color)) +
  labs(title='Zależność zmiennej depth od table', subtitle='Uwzględniony podział na kolor') +
```

```

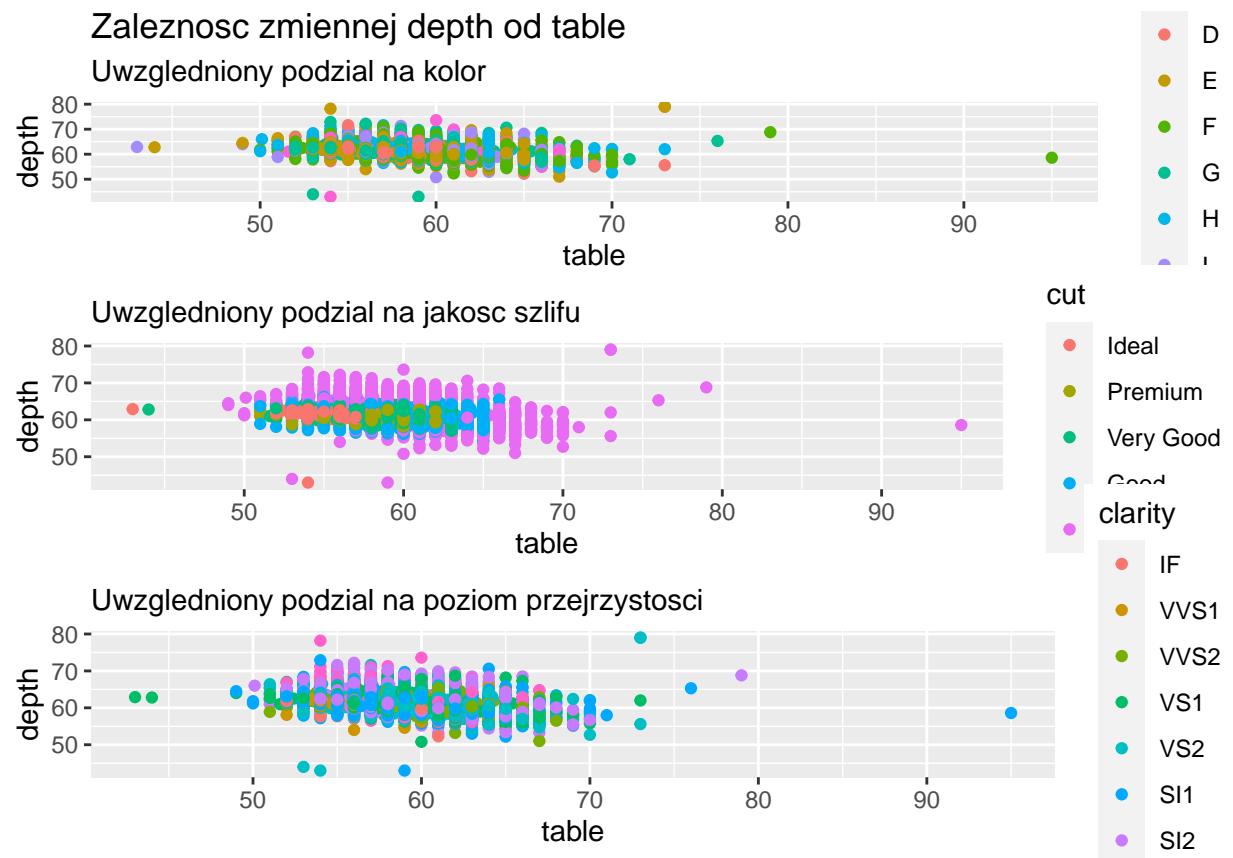
geom_point()

plot_4_cut <- ggplot(diamonds, aes(table, depth, color=cut)) +
  labs(subtitle='Uwzględniony podział na jakość szlifu') +
  geom_point()

plot_4_clarity <- ggplot(diamonds, aes(table, depth, color=clarity)) +
  labs(subtitle='Uwzględniony podział na poziom przejrzystości') +
  geom_point()

grid.arrange(plot_4_color, plot_4_cut, plot_4_clarity, nrow=3)

```

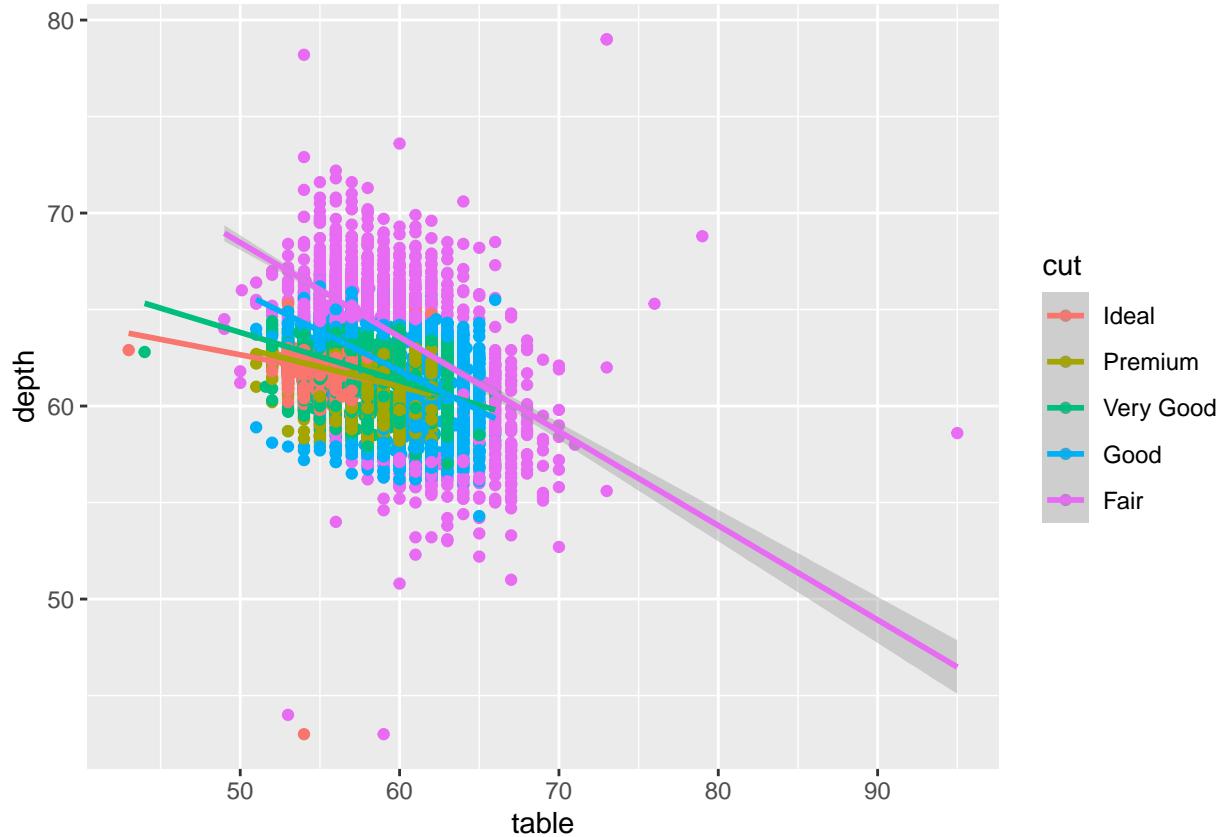


Zmienna jakości szlifu zdaje się układać na wykresie znacznie mniej chaotycznie od pozostałych. Przyjrzymy się jej na wykresie z liniami regresji dla poszczególnych poziomów jakości szlifu.

```

ggplot(diamonds, aes(table, depth, color=cut)) +
  geom_point() +
  geom_smooth(method='lm', formula='y~x')

```



Spróbujmy zbudować model z uwzględnieniem zmiennej `cut`.

```
model_3_cut <- lm(depth ~ table + cut, diamonds)
summary.lm(model_3_cut)

##
## Call:
## lm(formula = depth ~ table + cut, data = diamonds)
##
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -21.8242  -0.5846   0.1047   0.7459  19.0561 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 78.16025   0.16078 486.15 <2e-16 ***
## table      -0.29402   0.00287 -102.46 <2e-16 ***
## cutPremium  0.37737   0.01560   24.19 <2e-16 ***
## cutVery Good 0.69810   0.01508   46.28 <2e-16 ***
## cutGood     1.46125   0.02095   69.76 <2e-16 ***
## cutFair     3.24684   0.03293   98.60 <2e-16 ***  
## ---        
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 1.227 on 53913 degrees of freedom
## Multiple R-squared:  0.2664, Adjusted R-squared:  0.2663 
## F-statistic: 3915 on 5 and 53913 DF,  p-value: < 2.2e-16
```

Zmienna `cut` pozwala na objaśnienie ~26% wariancji. Sprawdźmy, czy zaimplementowanie interakcji będzie zasadne.

```
model_3_cut2 <- lm(depth ~ table * cut, diamonds)
summary.lm(model_3_cut2)

##
## Call:
## lm(formula = depth ~ table * cut, data = diamonds)
##
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -23.0017  -0.5612   0.0982   0.7416  21.7719 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 70.625412  0.369574 191.100 < 2e-16 ***
## table       -0.159350  0.006604 -24.131 < 2e-16 ***
## cutPremium   2.602448  0.551396  4.720 2.37e-06 ***
## cutVery Good 5.749463  0.476371 12.069 < 2e-16 ***
## cutGood      15.904268 0.512856 31.011 < 2e-16 ***
## cutFair      22.276427 0.584072 38.140 < 2e-16 ***
## table:cutPremium -0.044282 0.009597 -4.614 3.95e-06 ***
## table:cutVery Good -0.091816 0.008395 -10.938 < 2e-16 ***
## table:cutGood    -0.252365 0.008957 -28.176 < 2e-16 ***
## table:cutFair     -0.329332 0.010100 -32.607 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.208 on 53909 degrees of freedom
## Multiple R-squared:  0.2885, Adjusted R-squared:  0.2884 
## F-statistic: 2429 on 9 and 53909 DF, p-value: < 2.2e-16
```

Model z interakcją radzi sobie nieco lepiej. Interakcje między `table` a każdą przyjmowaną przez `cut` wartością są istotne statystycznie.

Sprawdźmy jeszcze, czy aby na pewno użycie zmiennych towarzyszących `clarity` i `color` nie da lepszego efektu.

```
model_3_color <- lm(depth ~ table + color, diamonds)
summary.lm(model_3_color)

##
## Call:
## lm(formula = depth ~ table + color, data = diamonds)
##
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -19.6156  -0.7503  -0.0649   0.7537  20.3009 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 72.665439  0.152244 477.296 < 2e-16 ***
## table       -0.191055  0.002636 -72.470 < 2e-16 ***
## colorE      -0.019383  0.021579  -0.898   0.369    
## colorF       0.001773  0.021697   0.082   0.935
```

```

## colorG      0.037117  0.020991   1.768    0.077 .
## colorH      0.161276  0.022363   7.212 5.60e-13 ***
## colorI      0.181443  0.024889   7.290 3.14e-13 ***
## colorJ      0.267107  0.030667   8.710 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.366 on 53911 degrees of freedom
## Multiple R-squared:  0.09117, Adjusted R-squared:  0.09105
## F-statistic: 772.6 on 7 and 53911 DF, p-value: < 2.2e-16
model_3_clarity <- lm(depth ~ table + clarity, diamonds)
summary.lm(model_3_clarity)

##
## Call:
## lm(formula = depth ~ table + clarity, data = diamonds)
##
## Residuals:
##       Min     1Q Median     3Q    Max 
## -19.4155 -0.7314 -0.0314  0.7447 20.4257 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 72.934434  0.152794 477.338 < 2e-16 ***
## table      -0.202166  0.002644 -76.461 < 2e-16 ***
## clarityVVS1 0.190057  0.039063   4.865 1.15e-06 ***
## clarityVVS2 0.257842  0.037243   6.923 4.46e-12 ***
## clarityVS1  0.320453  0.035388   9.055 < 2e-16 ***
## clarityVS2  0.397997  0.034334  11.592 < 2e-16 ***
## claritySI1  0.576063  0.034251  16.819 < 2e-16 ***
## claritySI2  0.548640  0.035172  15.599 < 2e-16 ***
## clarityII1  1.593476  0.059401  26.826 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.354 on 53910 degrees of freedom
## Multiple R-squared:  0.1071, Adjusted R-squared:  0.1069
## F-statistic: 807.9 on 8 and 53910 DF, p-value: < 2.2e-16

```

Modele nie radzą sobie tak dobrze, jak ten ze zmienną cut. Zatem szerokość górnej części diamentu wpływa na proporcję wysokości diamentu do jego obwodu, jednak dobrze jest uwzględnić zmienną towarzyszącą jakości szlifu, co pozwala na lepsze objaśnianie wariancji zmiennej celu.