Z-Score

Nesse exercício vamos aprender uma técnica estatística para lidar com valores individuais.

Base Casas Vendidas em Ames entre 2006 e 2010

Vamos usar essa base com 2930 linhas com 82 colunas contendo informações de características de casas vendidas entre 2006 e 2010 na cidade Ames (estado de Iowa nos EUA).

Esse foi um trabalho feito pelo professor Dean DeCock, publicado neste artigo e os detalhes sobre as informações presentes na base estão neste link

O separador da base são tabs, é um arquivo do tipo TSV (tab-separated value), são basicamente espaços. Poderíamos usar a função read.csv e informar o parâmetro sep= "\t" que funcionaria da mesma forma.

```
base <- read_tsv("https://s3.amazonaws.com/dq-content/444/AmesHousing.txt")
```

```
## Rows: 2930 Columns: 82
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (45): PID, MS SubClass, MS Zoning, Street, Alley, Lot Shape, Land Contou...
## dbl (37): Order, Lot Frontage, Lot Area, Overall Qual, Overall Cond, Year Bu...
##
## 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.
```

```
glimpse(base)
```

```
## Rows: 2,930
## Columns: 82
## $ Order             <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 1~
## $ PID               <chr> "0526301100", "0526350040", "0526351010", "052635303~
```

```
## $ `MS SubClass`     <chr> "020", "020", "020", "020", "060", "060", "120", "12~
## $ `MS Zoning`       <chr> "RL", "RH", "RL", "RL", "RL", "RL", "RL", "RL", "RL"~
## $ `Lot Frontage`    <dbl> 141, 80, 81, 93, 74, 78, 41, 43, 39, 60, 75, NA, 63,~
## $ `Lot Area`        <dbl> 31770, 11622, 14267, 11160, 13830, 9978, 4920, 5005,~
## $ Street            <chr> "Pave", "Pave", "Pave", "Pave", "Pave", "Pave", "Pav~
## $ Alley             <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ `Lot Shape`       <chr> "IR1", "Reg", "IR1", "Reg", "IR1", "IR1", "Reg", "IR~
## $ `Land Contour`    <chr> "Lvl", "Lvl", "Lvl", "Lvl", "Lvl", "Lvl", "Lvl", "HL~
## $ Utilities         <chr> "AllPub", "AllPub", "AllPub", "AllPub", "AllPub", "A~
## $ `Lot Config`      <chr> "Corner", "Inside", "Corner", "Corner", "Inside", "I~
## $ `Land Slope`      <chr> "Gtl", "Gtl", "Gtl", "Gtl", "Gtl", "Gtl", "Gtl", "Gt~
## $ Neighborhood      <chr> "NAmes", "NAmes", "NAmes", "NAmes", "Gilbert", "Gilb~
## $ `Condition 1`     <chr> "Norm", "Feedr", "Norm", "Norm", "Norm", "Norm", "No~
## $ `Condition 2`     <chr> "Norm", "Norm", "Norm", "Norm", "Norm", "Norm", "Nor~
## $ `Bldg Type`       <chr> "1Fam", "1Fam", "1Fam", "1Fam", "1Fam", "1Fam", "Twn~
## $ `House Style`     <chr> "1Story", "1Story", "1Story", "1Story", "2Story", "2~
## $ `Overall Qual`    <dbl> 6, 5, 6, 7, 5, 6, 8, 8, 8, 7, 6, 6, 6, 7, 8, 8, 8, 9~
## $ `Overall Cond`    <dbl> 5, 6, 6, 5, 5, 6, 5, 5, 5, 5, 5, 7, 5, 5, 5, 5, 7, 2~
## $ `Year Built`      <dbl> 1960, 1961, 1958, 1968, 1997, 1998, 2001, 1992, 1995~
## $ `Year Remod/Add`  <dbl> 1960, 1961, 1958, 1968, 1998, 1998, 2001, 1992, 1996~
## $ `Roof Style`      <chr> "Hip", "Gable", "Hip", "Hip", "Gable", "Gable", "Gab~
## $ `Roof Matl`       <chr> "CompShg", "CompShg", "CompShg", "CompShg", "CompShg~
## $ `Exterior 1st`    <chr> "BrkFace", "VinylSd", "Wd Sdng", "BrkFace", "VinylSd~
## $ `Exterior 2nd`    <chr> "Plywood", "VinylSd", "Wd Sdng", "BrkFace", "VinylSd~
## $ `Mas Vnr Type`    <chr> "Stone", "None", "BrkFace", "None", "None", "BrkFace~
## $ `Mas Vnr Area`    <dbl> 112, 0, 108, 0, 0, 20, 0, 0, 0, 0, 0, 0, 0, 0, 0, 60~
## $ `Exter Qual`      <chr> "TA", "TA", "TA", "Gd", "TA", "TA", "Gd", "Gd", "Gd"~
## $ `Exter Cond`      <chr> "TA", "TA", "TA", "TA", "TA", "TA", "TA", "TA", "TA"~
## $ Foundation        <chr> "CBlock", "CBlock", "CBlock", "CBlock", "PConc", "PC~
## $ `Bsmt Qual`       <chr> "TA", "TA", "TA", "TA", "Gd", "TA", "Gd", "Gd", "Gd"~
## $ `Bsmt Cond`       <chr> "Gd", "TA", "TA", "TA", "TA", "TA", "TA", "TA", "TA"~
## $ `Bsmt Exposure`   <chr> "Gd", "No", "No", "No", "No", "No", "Mn", "No", "No"~
## $ `BsmtFin Type 1`  <chr> "BLQ", "Rec", "ALQ", "ALQ", "GLQ", "GLQ", "GLQ", "AL~
## $ `BsmtFin SF 1`    <dbl> 639, 468, 923, 1065, 791, 602, 616, 263, 1180, 0, 0,~
## $ `BsmtFin Type 2`  <chr> "Unf", "LwQ", "Unf", "Unf", "Unf", "Unf", "Unf", "Un~
## $ `BsmtFin SF 2`    <dbl> 0, 144, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1120, 0,~
## $ `Bsmt Unf SF`     <dbl> 441, 270, 406, 1045, 137, 324, 722, 1017, 415, 994, ~
## $ `Total Bsmt SF`   <dbl> 1080, 882, 1329, 2110, 928, 926, 1338, 1280, 1595, 9~
## $ Heating           <chr> "GasA", "GasA", "GasA", "GasA", "GasA", "GasA", "Gas~
## $ `Heating QC`      <chr> "Fa", "TA", "TA", "Ex", "Gd", "Ex", "Ex", "Ex", "Ex"~
## $ `Central Air`     <chr> "Y", "Y", "Y", "Y", "Y", "Y", "Y", "Y", "Y", "Y", "Y~
## $ Electrical        <chr> "SBrkr", "SBrkr", "SBrkr", "SBrkr", "SBrkr", "SBrkr"~
## $ `1st Flr SF`      <dbl> 1656, 896, 1329, 2110, 928, 926, 1338, 1280, 1616, 1~
## $ `2nd Flr SF`      <dbl> 0, 0, 0, 0, 701, 678, 0, 0, 0, 776, 892, 0, 676, 0, ~
```

```
## $ `Low Qual Fin SF` <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ `Gr Liv Area`     <dbl> 1656, 896, 1329, 2110, 1629, 1604, 1338, 1280, 1616,~
## $ `Bsmt Full Bath`  <dbl> 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1~
## $ `Bsmt Half Bath`  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ `Full Bath`       <dbl> 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 1, 3, 2, 1~
## $ `Half Bath`       <dbl> 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1~
## $ `Bedroom AbvGr`   <dbl> 3, 2, 3, 3, 3, 3, 2, 2, 2, 3, 3, 3, 3, 2, 1, 4, 4, 1~
## $ `Kitchen AbvGr`   <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
## $ `Kitchen Qual`    <chr> "TA", "TA", "Gd", "Ex", "TA", "Gd", "Gd", "Gd", "Gd"~
## $ `TotRms AbvGrd`   <dbl> 7, 5, 6, 8, 6, 7, 6, 5, 5, 7, 7, 6, 7, 5, 4, 12, 8, ~
## $ Functional        <chr> "Typ", "Typ", "Typ", "Typ", "Typ", "Typ", "Typ", "Ty~
## $ Fireplaces        <dbl> 2, 0, 0, 2, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1~
## $ `Fireplace Qu`    <chr> "Gd", NA, NA, "TA", "TA", "Gd", NA, NA, "TA", "TA", ~
## $ `Garage Type`     <chr> "Attchd", "Attchd", "Attchd", "Attchd", "Attchd", "A~
## $ `Garage Yr Blt`   <dbl> 1960, 1961, 1958, 1968, 1997, 1998, 2001, 1992, 1995~
## $ `Garage Finish`   <chr> "Fin", "Unf", "Unf", "Fin", "Fin", "Fin", "Fin", "RF~
## $ `Garage Cars`     <dbl> 2, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 2, 3~
## $ `Garage Area`     <dbl> 528, 730, 312, 522, 482, 470, 582, 506, 608, 442, 44~
## $ `Garage Qual`     <chr> "TA", "TA", "TA", "TA", "TA", "TA", "TA", "TA", "TA"~
## $ `Garage Cond`     <chr> "TA", "TA", "TA", "TA", "TA", "TA", "TA", "TA", "TA"~
## $ `Paved Drive`     <chr> "P", "Y", "Y", "Y", "Y", "Y", "Y", "Y", "Y", "Y", "Y~
## $ `Wood Deck SF`    <dbl> 210, 140, 393, 0, 212, 360, 0, 0, 237, 140, 157, 483~
## $ `Open Porch SF`   <dbl> 62, 0, 36, 0, 34, 36, 0, 82, 152, 60, 84, 21, 75, 0,~
## $ `Enclosed Porch`  <dbl> 0, 0, 0, 0, 0, 0, 170, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ `3Ssn Porch`      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ `Screen Porch`    <dbl> 0, 120, 0, 0, 0, 0, 0, 144, 0, 0, 0, 0, 0, 0, 140, 2~
## $ `Pool Area`       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ `Pool QC`         <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ Fence             <chr> NA, "MnPrv", NA, NA, "MnPrv", NA, NA, NA, NA, NA, NA~
## $ `Misc Feature`    <chr> NA, NA, "Gar2", NA, NA, NA, NA, NA, NA, NA, NA, "She~
## $ `Misc Val`        <dbl> 0, 0, 12500, 0, 0, 0, 0, 0, 0, 0, 0, 500, 0, 0, 0, 0~
## $ `Mo Sold`         <dbl> 5, 6, 6, 4, 3, 6, 4, 1, 3, 6, 4, 3, 5, 2, 6, 6, 6, 6~
## $ `Yr Sold`         <dbl> 2010, 2010, 2010, 2010, 2010, 2010, 2010, 2010, 2010~
## $ `Sale Type`       <chr> "WD", "WD", "WD", "WD", "WD", "WD", "WD", "WD", "WD"~
## $ `Sale Condition`  <chr> "Normal", "Normal", "Normal", "Normal", "Normal", "N~
## $ SalePrice         <dbl> 215000, 105000, 172000, 244000, 189900, 195500, 2135~
```

Valores Individuais

Vamos imaginar que pegamos aleatoriamente uma casa no valor de 220 mil, como sabemos se esse é um valor caro, barato, na média?

```r
mean(base$SalePrice)
```

```
## [1] 180796.1
```

Calculando a média entendemos que 220 mil está da média, mas é muito acima, ou apenas ligeiramente acima?

Isso depende do desvio padrão.

Por exemplo se o desvio padrão fosse de 40 mil, uma casa de 220 mil ainda está dentro do desvio padrão (está a "um" desvio padrão de distância), e apesar de estar acima da média, está num dos valores comuns da base.

Mas se o desvio padrão fosse de 10 mil, os valores mais comuns iriam até 190 mil, o que indicaria 220 mil como um valor muito acima da média e muito caro, muito incomum (está a "quatro" desvio padrão de distância da média de 180 mil).

Como fica esse cálculo então? Entendemos a distância entre o valor selecionado e a média, e dividimos essa distância pelo desvio padrão

```r
(220000 - mean(base$SalePrice)) / sd(base$SalePrice)
```

```
## [1] 0.4907443
```

Z-Score

Representando isso numa fórmula, dado um valor x para uma população com média u e um desvio padrão o, qual o número de desvios padrão de distância da média ele tem?

$$z = \frac{x - \mu}{\sigma}$$

A fórmula acima é mais conhecida como "the standard score" ou **z-score**. Aplicando a fórmula para amostras teríamos esse resultado:

$$z = \frac{x - \bar{x}}{s}$$

Escrito em uma função:

```r
z_score <- function(valor, vetor){
  media_vetor <- mean(vetor)
  dp <- std_dev_bessel(vetor)

  (valor - media) / dp
}
```

O resultado desse cálculo pode ser:

- positivo, indicando que o valor está acima da média (ficando à direita da curva no gráfico) ou,
- negativo, indicando que está abaixo da média (ficando à esquerda da curva no gráfico)

O valor indica a quantos desvios padrão de distância da média ele está.

Vamos entender na prática com o preço das casas, o valor médio do preço das casa é 180.796,10

```r
media <- mean(base$SalePrice)
media
```

```
## [1] 180796.1
```

E se calcularmos a quantos desvios padrão da média esse valor está? Ele será zero pois é a própria média

```r
z_score(media, base$SalePrice)
```

```
## [1] 0
```

E qual é o desvio padrão mesmo? É de 79.886,69

```r
std_dev_bessel(base$SalePrice)
```

```
## [1] 79886.69
```

Vamos olhar o maior valor presente na base

```r
max_valor <- max(base$SalePrice)
max_valor
```

```
## [1] 755000
```

E qual o seu desvio padrão? é de mais de 7 desvios padrão acima da média

```r
z_score(max_valor, base$SalePrice)
```

```
## [1] 7.18773
```

- 1 desvio padrão acima da média abrange valores até 260.682,79 (média 180796.1 + 1 desvio padrão 79.886,69)
- 2 desvios padrão acima da média abrange valores até 340.569,48
- 3 desvios padrão acima da média abrange valores até 420.456,17
- ...
- 7 desvios padrão acima da média abrange valores até 740.002,93
- 8 desvios padrão acima da média abrange valores até 819.889,62

Então de fato faz sentido esse valor estar entre 7 e 8 desvios padrão acima da média.

Aplicação do Z-Score

Vamos deixar mais claro numa situação prática, por que calcular o z-score pode ser útil. Pense que uma empresa quer encontrar o melhor bairro para investir em casas por com valor de 200.000,00, a ideia é comprar casas num preço médio para revender num preço maior. O preço médio faz parte da estratégia da empresa, por acreditar que atrai mais compradores interessados. Esse valor de 200 mil pode ser abaixo da média num bairro, enquanto no outro acima da média, e o z-score vai nos ajudar a entender isso, quanto mais próximo de 0 melhor o bairro (está dentro da proposta)

Nesse exemplo vamos focar em 5 bairros:

- 'NAmes' for North Ames
- 'CollgCr' for College Creek
- 'OldTown' for Old Town
- 'Edwards' for Edwards
- 'Somerst' for Somerset

```r
target_neighborhoods <- c('NAmes', 'CollgCr', 'OldTown', 'Edwards', 'Somerst')

result <-
  base %>%
  filter(Neighborhood %in% target_neighborhoods) %>%
  group_by(Neighborhood) %>%
  summarise(z_score = abs(z_score(200000,SalePrice))) %>%
  arrange(z_score)

result
```

```
## # A tibble: 5 x 2
##   Neighborhood z_score
##   <chr>           <dbl>
## 1 Somerst         0.334
## 2 CollgCr         0.354
## 3 Edwards         0.400
## 4 OldTown         0.433
## 5 NAmes           0.602
```

De acordo com esse resultado o melhor bairro seria Somerset, pois tem o menor z-score o que significa que está mais próximo do preço médio dentre todas as opções.

Distribuição do Z-Score

E se a gente transformasse a base toda em z-score? como a base fica distribuída?

```r
base <- base %>%
    mutate(z_prices = (SalePrice - mean(SalePrice)) / std_dev_bessel(SalePrice) )

grafico1 <-
  ggplot(data = base,
    aes(x = SalePrice)) +
    geom_density(alpha = 0.1,
                 color='blue',
                 fill='blue') +
    scale_y_continuous(labels = scales::comma) +
    scale_x_continuous(labels = scales::comma,
                       lim = c(min(base$SalePrice),
                               max(base$SalePrice))) +
    theme_bw() +
```

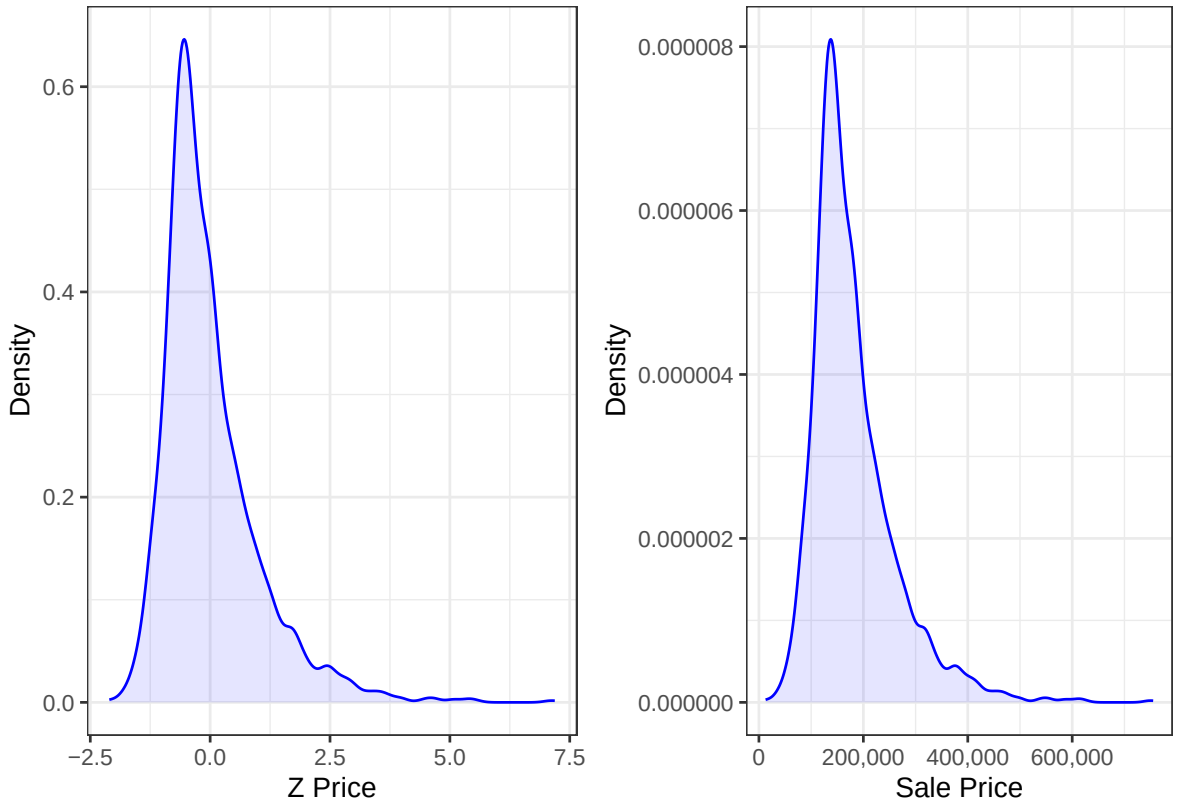
```
    xlab("Sale Price") +
    ylab("Density")

grafico2 <-
  ggplot(data = base,
    aes(x = z_prices)) +
    geom_density(alpha = 0.1,
                 color='blue',
                 fill='blue') +
    scale_y_continuous(labels = scales::comma) +
    scale_x_continuous(labels = scales::comma,
                       lim = c(min(base$z_prices),
                               max(base$z_prices))) +
    theme_bw() +
    xlab("Z Price") +
    ylab("Density")

# Show the plots side-by-side
library(ggpubr)
```

```
## Warning: package 'ggpubr' was built under R version 4.3.3
```

```
ggarrange(grafico2, grafico1, ncol = 2, nrow = 1)
```



Podemos ver que a distribuição dos dados se mantém, apenas trocamos a escala do valor original para o z-score.

E se calculássemos a média e desvio padrão do z-score como fica? A média fica extramamente próximo à zero, e o desvio padrão é 1. Isso se dá pois o zero representa a média de 180k do SalePrice original, e 1 desvio padrão acima ou abaixo da média representa ~79k do SalePrice original. Na conversão do SalePrice para Z-score os valores se mantiveram nas mesmas posições, então o acúmulo dos dados ao redor da média se manteve, então todos registros que compunham 1 desvio padrão de distância da média continuam no mesmo lugar, portanto o z-score vai ter o valor 1 como desvio padrão, pois representa o 79k.

```
mean(base$z_prices)
```

```
## [1] -0.00000000000000001003927
```

```
std_dev_bessel((base$z_prices))
```

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

Convertendo do Z-Score para o Valor Original

O Z-Score demonstra ser muito útil para análises onde precisamos comparar valores em escalas diferentes, ou tomar decisões que dependem de avaliar o quão próximo da média um valor está. No entanto para comunicar esses resultados, o z-score não é a melhor opção por ser muito técnico. Nesse caso faz sentido recuperar o valor original.

Basta pegarmos a fórmula do Z-score e fazer algumas transformações até chegar no seguinte resultado:

$$x = z\sigma + \mu$$

Vamos relembrar o exemplo feito acima, o bairro de Somerset foi o escolhido por ter o valor de 200 mil mais próximo da média.

```
valor_original <- function(zscore, vetor){
  media_vetor <- mean(vetor)
  dp <- std_dev_bessel(vetor)

  zscore*dp+media_vetor
}
```

```
somerst <-
  base %>% filter(Neighborhood == "Somerst") %>% pull(SalePrice)

valor_original(0.3343456, somerst)
```

```
## [1] 248911.3
```

```
mean(somerst)
```

```
## [1] 229707.3
```