Estatística Aplicada I

Aluno: Victor Lima

Data: 18/06/2023

Terceira Lista de Exercícios

Com a base de dados “imoveiscwbav” fazer uma regressão linear e obter os resultados em um modelo de previsão.

a. Tentar alterar as variáveis ou implementar um outro método de estimação que minimize o problema da heterocedasticidade. Explique como você procedeu, apresente justificativa(s) para o(s) seu(s) procedimento(s) e também os códigos utilizados
b. Fazer todos os testes necessários e apresente os resultados, assim como sua interpretação dos testes;
c. Fazer uma simulação, estabeleça os valores da simulação e apresente-os, assim como o resultado da predição e o intervalo de confiança.

1 - Carregar conjunto de dados

```
load("imoveiscwbav.RData")
imoveis <- imoveiscwbav
```

```
head(imoveis)
```

	price <dbl>	age <dbl>	parea <dbl>	tarea <dbl>	bath <dbl>	ensuit <dbl>	garag <dbl>	plaz <dbl>	park <dbl>
1	1100000	15	150	190	4	1	2	0.08058169	0.7132806
2	895000	11	165	210	4	1	2	0.16635098	0.6983694
3	2513600	2	146	275	4	3	3	0.05607530	1.3129824
4	755000	25	163	238	3	1	2	0.32159391	2.1099578
5	1099000	1	107	189	3	1	2	0.14663511	1.0175299
6	475000	31	96	124	2	1	1	0.12615199	1.9700244

6 rows | 1-10 of 21 columns

2 - Estimar um modelo preliminar

Gerando um Regressão Linear tomando como referência o preço dos imóveis.

```
resultados <- lm (price~., data=imoveis)
summary (resultados)
```

```
##
## Call:
## lm(formula = price ~ ., data = imoveis)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -495718 -134211   -2632  104528 2419265
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -420453.5   130052.5  -3.233   0.0013 **
## age           -7839.1     1025.3  -7.645 1.01e-13 ***
## parea          2592.2      624.0   4.154 3.82e-05 ***
## tarea          1975.8      333.9   5.918 5.91e-09 ***
## bath          13452.6    14832.9   0.907   0.3649
## ensuit       125949.6    18560.7   6.786 3.15e-11 ***
## garag        169687.5    21756.1   7.800 3.41e-14 ***
## plaz         224393.0    94219.1   2.382   0.0176 *
## park         -63439.6    27154.0  -2.336   0.0199 *
## trans         26642.3    22718.5   1.173   0.2414
## kidca         10452.8    34899.8   0.300   0.7647
## school        -7975.8    56635.7  -0.141   0.8881
## health         1217.4    56216.5   0.022   0.9827
## bike         -85864.4    56073.0  -1.531   0.1263
## barb         -43925.7    22602.3  -1.943   0.0525 .
## balc          65144.8    25242.3   2.581   0.0101 *
## elev        -111743.4    25295.0  -4.418 1.21e-05 ***
## fitg         123052.7    28456.0   4.324 1.83e-05 ***
## party         36463.1    28481.1   1.280   0.2010
## categ        283061.5    55653.0   5.086 5.11e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 229400 on 521 degrees of freedom
## Multiple R-squared:  0.8099, Adjusted R-squared:  0.803
## F-statistic: 116.8 on 19 and 521 DF,  p-value: < 2.2e-16
```

3 - Manipulando alguns parâmetros para melhorar a significância das mesmas

a. Substituindo o “price” e “tarea” por uma variável que representa o log natural do preço do metro quadrado do imóvel.

```
imoveis$lnPricePerMeter <- with(imoveis, log(price/tarea))
```

4 - Reestimar o modelo com a variável “lnPricePerMeter” ao invés de “price” e “tarea”

```
resultados <- lm (lnPricePerMeter~.-price-tarea, data=imoveis)
summary (resultados)
```

```
##
## Call:
## lm(formula = lnPricePerMeter ~ . - price - tarea, data = imoveis)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.6644 -0.1374 -0.0040  0.1342  0.7934
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  8.1072230  0.1176307  68.921  < 2e-16 ***
## age         -0.0105255  0.0009147 -11.507  < 2e-16 ***
## parea       -0.0003820  0.0004370  -0.874  0.38243
## bath         0.0158442  0.0133514   1.187  0.23588
## ensuit       0.0690920  0.0167785   4.118 4.44e-05 ***
## garag        0.0999483  0.0192758   5.185 3.09e-07 ***
## plaz         0.1794591  0.0852099   2.106  0.03567 *
## park        -0.0444557  0.0244936  -1.815  0.07010 .
## trans        0.0278778  0.0205167   1.359  0.17480
## kidca        0.0056863  0.0314702   0.181  0.85668
## school       0.0263432  0.0511635   0.515  0.60685
## health       0.0471362  0.0508308   0.927  0.35419
## bike        -0.1105924  0.0506873  -2.182  0.02957 *
## barb         0.0061324  0.0204439   0.300  0.76432
## balc         0.0532105  0.0228292   2.331  0.02014 *
## elev        -0.0638321  0.0228457  -2.794  0.00540 **
## fitg         0.0709512  0.0257382   2.757  0.00604 **
## party        0.0416514  0.0257252   1.619  0.10603
## categ        0.1621735  0.0495843   3.271  0.00114 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2075 on 522 degrees of freedom
## Multiple R-squared:  0.6162, Adjusted R-squared:  0.603
## F-statistic: 46.56 on 18 and 522 DF,  p-value: < 2.2e-16
```

5 - Verificar os outliers pelo teste de Bonferroni

```
library (carData)
library(car)
outliers <- outlierTest(resultados)
print(outliers)
```

```
## No Studentized residuals with Bonferroni p < 0.05
## Largest |rstudent|:
##     rstudent unadjusted p-value Bonferroni p
## 393  3.93913         9.2912e-05     0.050265
```

Com a manipulação que foi realizada com as parâmentros, vemos que não foram observados nenhum outlier para ser descartado.

6 - Realizar um teste Stepwise, para verificar quais variáveis devem efetivamente entrar no modelo

```
library(RcmdrMisc)
```

```
## Carregando pacotes exigidos: sandwich
```

```
stepwise <- stepwise(resultados, direction= 'backward/forward', criterion ='AIC', trace = FAL
SE)
```

```
##
## Direction:  backward/forward
## Criterion:  AIC
```

Verificando a variável call do teste Stepwise, que sugere o melhor modelo de parâmetros a serem utilizados, supondo a melhor siginificância.

```
print(stepwise$call)
```

```
## lm(formula = lnPricePerMeter ~ age + ensuit + garag + plaz +
##     park + trans + bike + balc + elev + fitg + party + categ,
##     data = imoveis)
```

Adicionar os parâmentros que o modelo sugeriu em uma variável para facilitar a utilização

```
parametersStepwise <- lnPricePerMeter ~ age + ensuit + garag + plaz + park + trans + bike + b
alc + elev + fitg + party + categ
```

Atualizado o modelo atual para utilizar a sugestão do teste Stepwise

```
resultados <- lm(formula = parametersStepwise, data=imoveis)
summary (resultados)
```

```
##
## Call:
## lm(formula = parametersStepwise, data = imoveis)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.67201 -0.13662 -0.00441  0.14088  0.76320
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  8.1423442  0.0906466  89.825  < 2e-16 ***
## age         -0.0105241  0.0008678 -12.127  < 2e-16 ***
## ensuit       0.0744857  0.0124824   5.967 4.43e-09 ***
## garag        0.1011571  0.0170680   5.927 5.58e-09 ***
## plaz         0.1860202  0.0796146   2.337 0.019838 *
## park        -0.0463787  0.0204125  -2.272 0.023483 *
## trans        0.0268011  0.0180755   1.483 0.138742
## bike        -0.1104707  0.0467025  -2.365 0.018370 *
## balc         0.0535542  0.0222088   2.411 0.016232 *
## elev        -0.0622101  0.0226944  -2.741 0.006329 **
## fitg         0.0721503  0.0252520   2.857 0.004442 **
## party        0.0452975  0.0254059   1.783 0.075168 .
## categ        0.1633865  0.0465153   3.513 0.000482 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2068 on 528 degrees of freedom
## Multiple R-squared:  0.6142, Adjusted R-squared:  0.6055
## F-statistic: 70.06 on 12 and 528 DF,  p-value: < 2.2e-16
```

7 - Verificar a existência de multicolinearidade

```
car::vif(resultados)
```

```
##      age   ensuit    garag     plaz     park    trans     bike     balc
## 1.532377 1.617533 1.667997 1.163733 1.731994 1.483340 1.221906 1.541971
##     elev     fitg    party    categ
## 1.389840 1.720755 2.023789 1.159932
```

Vamos usar como base que o fator de inflação da variância deve ser maior do que 5 para ter uma correlação siginificativa. Desta forma, nenhum dado deve ser desconsiderado por multicolinearidade.

8 - Teste de especificação do modelo (Teste RESET)

```
library (zoo)
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
library (lmtest)
resettest(parametersStepwise, power=2:3, type="regressor", data=imoveis)
```

```
##
##  RESET test
##
## data:  parametersStepwise
## RESET = 1.0909, df1 = 24, df2 = 504, p-value = 0.3493
```

Verificando o valor tabelado para considerar uma especificação correta do modelo.

```
qf(.95,df1=24,df2=504)
```

```
## [1] 1.538985
```

Desta forma como o valor apresentado do RESET é `1.0909` , que é menor do que o valor tabelado de `1.538985` , podemos afirmar que o modelo foi corretamente especificado.

9 - Verificação de heterocedasticidade

Teste utilizado: Teste de Breusch-Pagan

```
bptest(parametersStepwise, studentize=FALSE, data=imoveis)
```

```
##
##  Breusch-Pagan test
##
## data:  parametersStepwise
## BP = 30.548, df = 12, p-value = 0.002308
```

Verificando valor tabelado para avaliação de heterocedasticidade

```
qchisq(0.95, df=12)
```

```
## [1] 21.02607
```

Desta forma, observando o valor BP do teste que é de `30.548` , que é maior que o valor tabelado de `21.02607` , podemos afirmar que este modelo possui problemas de heterocedasticidade.

```
print(resultados)
```

```
##
## Call:
## lm(formula = parametersStepwise, data = imoveis)
##
## Coefficients:
## (Intercept)          age       ensuit        garag         plaz         park
##     8.14234     -0.01052      0.07449      0.10116      0.18602     -0.04638
##       trans         bike         balc         elev         fitg        party
##     0.02680     -0.11047      0.05355     -0.06221      0.07215      0.04530
##       categ
##     0.16339
```

10 - Correção da variância não constante por regressão robusta

```
library (sandwich)
coeftest(resultados, vcov=vcovHC(resultados, type="HC1"))
```

```
##
## t test of coefficients:
##
##                Estimate  Std. Error  t value  Pr(>|t|)
## (Intercept)  8.14234423  0.08955328  90.9218 < 2.2e-16 ***
## age         -0.01052406  0.00087809 -11.9852 < 2.2e-16 ***
## ensuit       0.07448572  0.01218288   6.1140 1.891e-09 ***
## garag        0.10115713  0.01776478   5.6943 2.059e-08 ***
## plaz         0.18602023  0.08890742   2.0923 0.0368894 *
## park        -0.04637871  0.02009018  -2.3085 0.0213553 *
## trans        0.02680114  0.01735857   1.5440 0.1231944
## bike        -0.11047067  0.04498544  -2.4557 0.0143823 *
## balc         0.05355424  0.02162596   2.4764 0.0135842 *
## elev        -0.06221007  0.02217448  -2.8055 0.0052095 **
## fitg         0.07215034  0.02505278   2.8799 0.0041391 **
## party        0.04529753  0.02535070   1.7868 0.0745377 .
## categ        0.16338653  0.04648733   3.5146 0.0004782 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Pelo teste T de coeficientes, vemos que as vairáveis `trans` e `party` não são siginificativas ao modelo, e podemos removê-las à fim de corrigir o problema de heterocedasticidade.

```
parametersStepwiseWithTTeste <- lnPricePerMeter ~ age + ensuit + garag + plaz + park + bike +
balc + elev + fitg + categ
resultados <- lm(formula = parametersStepwiseWithTTeste, data=imoveis)
summary (resultados)
```

```
##
## Call:
## lm(formula = parametersStepwiseWithTTeste, data = imoveis)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.7097 -0.1357 -0.0098  0.1386  0.7622
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  8.2168576  0.0719062 114.272  < 2e-16 ***
## age         -0.0105071  0.0008703 -12.073  < 2e-16 ***
## ensuit       0.0757395  0.0125097   6.054 2.67e-09 ***
## garag        0.1050199  0.0170264   6.168 1.37e-09 ***
## plaz         0.1904206  0.0797644   2.387 0.017323 *
## park        -0.0616568  0.0175458  -3.514 0.000479 ***
## bike        -0.0950288  0.0461634  -2.059 0.040027 *
## balc         0.0625512  0.0214461   2.917 0.003688 **
## elev        -0.0458250  0.0212405  -2.157 0.031421 *
## fitg         0.0856065  0.0229559   3.729 0.000213 ***
## categ        0.1628563  0.0457546   3.559 0.000405 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2075 on 530 degrees of freedom
## Multiple R-squared:  0.6103, Adjusted R-squared:  0.603
## F-statistic: 83.01 on 10 and 530 DF,  p-value: < 2.2e-16
```

Agora, iremos compilar novamente o teste de Breusch-Pagan, e verificar se a heterocedasticidade foi corrigida.

```
bptest(parametersStepwiseWithTTeste, studentize=FALSE, data=imoveis)
```

```
##
##  Breusch-Pagan test
##
## data:  parametersStepwiseWithTTeste
## BP = 26.779, df = 10, p-value = 0.002823
```

```
qchisq(0.95, df=10)
```

```
## [1] 18.30704
```

Como observado o valor de BP `26.779` , mesmo que seja menor que o valor anterior, ainda está acima do valor tabelado de `18.30704` para considerar um modelo com homocedasticidade.

11 - Calcular os intervalos de confiança para regressão linear robusta

```
confint <- confint (resultados, level = 0.95)
confint
```

```
##                   2.5 %       97.5 %
## (Intercept)  8.07560143  8.358113677
## age         -0.01221675 -0.008797437
## ensuit       0.05116480  0.100314264
## garag        0.07157231  0.138467435
## plaz         0.03372734  0.347113779
## park        -0.09612457 -0.027188955
## bike        -0.18571448 -0.004343214
## balc         0.02042146  0.104680924
## elev        -0.08755097 -0.004098985
## fitg         0.04051067  0.130702279
## categ        0.07297357  0.252738993
```

12 - Análise dos indicadores de performace do modelo final

```
library(performance)
model_performance(resultados)
```

	AIC	AICc	BIC	R2	R2_adjusted	RMSE	Sigma
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	-153.4365	-152.8456	-101.9155	0.6103199	0.6029674	0.2053729	0.2074932

1 row

Não será o caso desta atividade, mas podemos usar estes valores de performance para comparar com o resultado de outros modelo que podem serdesenvolvidos para a mesma base de dados.

13 - Realizar teste com dados fictícios

Vamos criar um dataframe com apenas uma linha onde teremos nossos dados fictícios para teste.

```
testeList <- list()
testeList["tarea"] <- 183
testeList["age"] <- 14
testeList["ensuit"] <- 1
testeList["garag"] <- 2
testeList["plaz"] <- 0
testeList["park"] <- 2
testeList["bike"] <- 0.5
testeList["balc"] <- 0
testeList["elev"] <- 0
testeList["fitg"] <- 0
testeList["categ"] <- 1
teste <- data.frame(testeList)
print(teste)
```

```
##   tarea age ensuit garag plaz park bike balc elev fitg categ
## 1   183  14      1     2    0    2  0.5    0    0    0     1
```

Realizando a predileção do valor do parâmentro `lnPricePerMeter`

```
predito <- predict(object = resultados, teste)
predito
```

```
##        1
## 8.347566
```

Como o valor do parâmento `lnPricePerMeter` não representa o valor do preço final, vamos fazer um calculo reverso de sua concepção para encontrar o valor predito do parâmentro `price`

```
preditoFinal <- (exp(predito))*teste$tarea
preditoFinal
```

```
##      1
## 772241
```

```
# Valor formatado: R$ 772.241,00
```

14 - Estimar o preço inferior do intervalo de confiança

```
Lestimate=confint[1,1] + confint[2,1]*teste$age + confint[3,1]*teste$ensuit + confint[4,1]*te
ste$garag +
  confint[5,1]*teste$plaz + confint[6,1]*teste$park + confint[7,1]*teste$bike + confint[8,1]*
teste$balc +
  confint[9,1]*teste$elev + confint[10,1]*teste$fitg + confint[11,1]*teste$categ
LestimateFormat <- (exp(Lestimate))*teste$tarea
LestimateFormat # R$ 487.102,40
```

```
## [1] 487102.4
```

15 - Estimar o preço superior do intervalo de confiança

```
Uestimate=confint[1,2] + confint[2,2]*teste$age + confint[3,2]*teste$ensuit + confint[4,2]*te
ste$garag +
  confint[5,2]*teste$plaz + confint[6,2]*teste$park + confint[7,2]*teste$bike + confint[8,2]*
teste$balc +
  confint[9,2]*teste$elev + confint[10,2]*teste$fitg + confint[11,2]*teste$categ
UestimateFormat <- (exp(Uestimate))*teste$tarea
UestimateFormat # R$ 1.224.293,00
```

```
## [1] 1224293
```

16 - Estimar o intervalo de confiança para a média

```
n <- nrow(imoveis)
m <- preditoFinal
s <- sd(imoveis$price)
dam <- s/sqrt(n)
CIlwr <- m + (qnorm(0.025))*dam
CIupr <- m - (qnorm(0.025))*dam
# Valor médio inferior: 728692.7 (R$ 728.692,70)
# Valor médio supeior: 815789.4 (R$ 815.789,40)
```

Conclusão

Como podemos observar nos dados de predileção, e comparando com os dados do intervalo de confiança para a média, temos uma distância ainda siginificativa do valor predito com a confiança do mesmo, pois vemos uma diferença de 36% entre eles. Esses valores na prática podem não ser efetivos, e podemos imaginar que este problema deve-se ao fato do que vimos nos testes de heterocedasticidade do modelo, que não conseguiu passar, mesmo após as correções realizadas com o teste de coeficiente T. Assim, implicando que temos um grupo de dados prejudicial para a configuração do modelo, e levando a uma conclusão que devemos reformular os dados e parâmentros utilizados ou substituir a base de dados.

Fim.