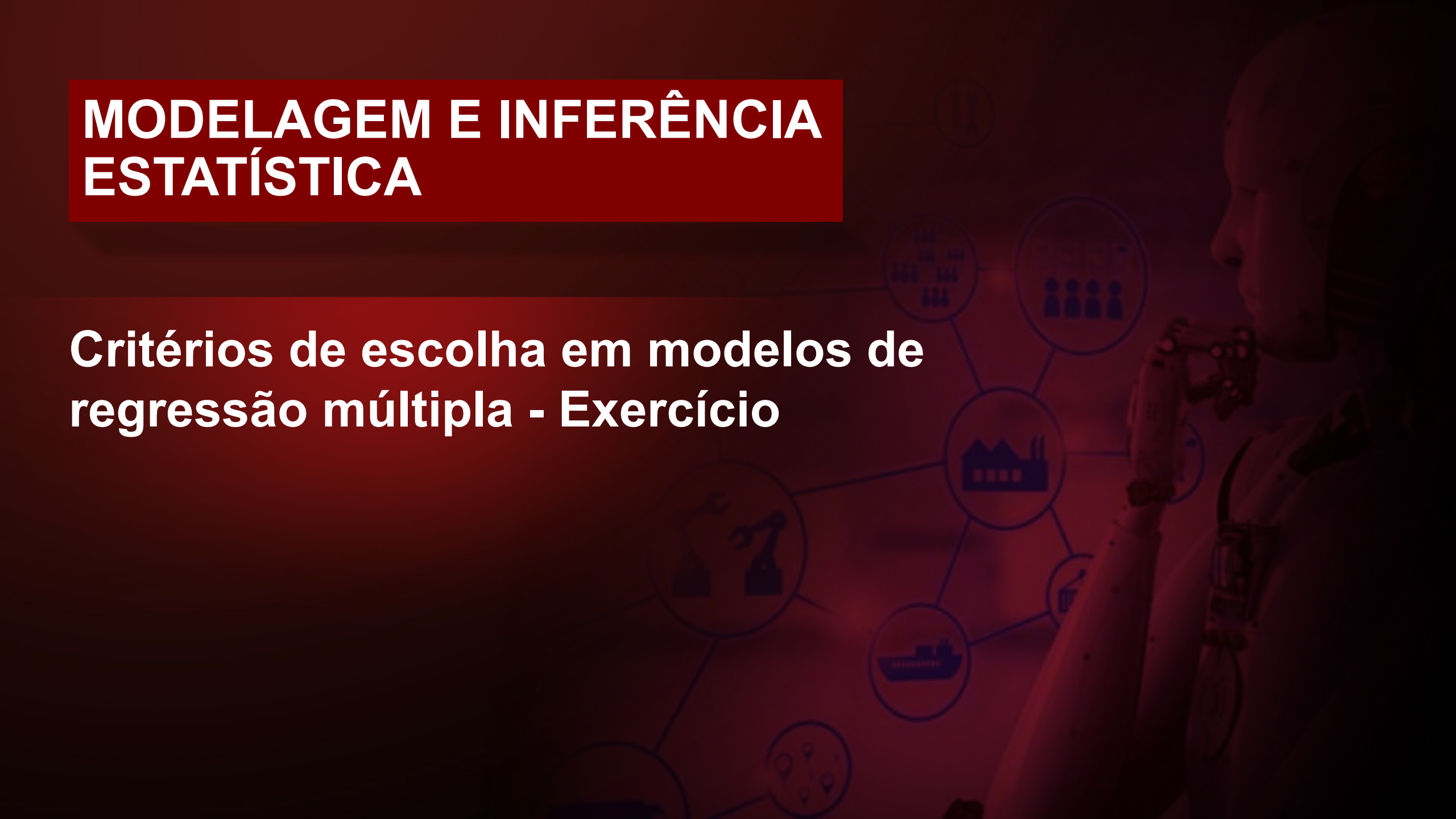
MODELAGEM E INFERÊNCIA ESTATÍSTICA

Critérios de escolha em modelos de regressão múltipla - Exercício



O QUE VOU ESTUDAR HOJE?

- Seleção de variáveis
- Análise e avaliação dos resultados de testes de hipótese
- Gráficos de diagnóstico
- Multicolinearidade

EXEMPLO

A loja "GT Auto" decidiu aprimorar o treinamento de seus vendedores inexperientes, e para tanto, criou uma base de dados dos veículos disponíveis, em documento de texto, com as seguintes informações:

- Marca/Modelo/Ano do carro: **brand_model_year**
- Capacidade volumétrica (cc) ou volume de deslocamento do motor: **cap_vol**
- Consumo de combustível (km/l): **consumo**
- Potência (cv): **power**
- Peso (kg): **weight**
- 0-100 (s) = tempo que o carro demora para atingir a velocidade de 100km/h: **cemm**
- Número de cilindros: **nu_cy**
- Tipo de motor, aspirado (0) ou turbo (1): **Etype**
- Número de cilindros: **nu_cy**

Após ter realizado a análise dos dados apresentados, obter o modelo de regressão múltipla e escolher as variáveis adequadas.

DEFINIR DADOS

BIBLIOTECA

```
#@title Bibliotecas
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
from statsmodels.formula.api import ols
import statsmodels.formula.api as smf #adicion
import seaborn as sns
import scipy.stats as stats #adicionada na sem
from scipy.stats import f #adicionada na seman
from statsmodels.graphics.gofplots import Probl
```

ESCOLHA DE DADOS

```
# Reduzir a base de dados para usar apenas
# o motor naturalmente aspirado
dfcars = cars.iloc[1:14,:]
dfcars.head(15)
```

UPLOAD BASE DE DADOS

```
#@title Upload da base de dados
from google.colab import files
uploaded = files.upload()
```

Escolher Ficheiros carst7.txt

- **carst7.txt**(text/plain) - 1808 bytes, last modified: 05/04/2022

```
Saving carst7.txt to carst7.txt
```

DEFINIR DADOS

RESULTADO

brand_model_year	cap_vol	consumo	power	weight	cemm	nu_cy	Etype
FIAT UNO Mille EP 1996	994	10.4	58	870	18.6	4	0
Hyundai HB20 Sense 2020	1000	12.8	80	989	14.5	3	0
FIAT Strada 1.4 2016	1368	10.3	86	1084	12.5	4	0
VolksWagen GOL 1.6 2015	1598	10.5	104	961	9.8	4	0
Chevrolet Cruze LTZ 1.8 2016	1796	8.5	144	1427	10.2	4	0
Honda Civic EXR 2016	1997	9.5	155	1294	10.9	4	0
Ford Focus 2.0 GLX 2012	1999	9.2	148	1347	10.4	4	0
BMW 325i 3.0 2012	2996	6.5	218	1460	7.1	6	0
AUDI A4 3.2 V6 Fsi 2011	3197	7.1	269	1610	6.4	6	0
Mercedes-Benz CLS 350 3.5 V6 2012	3498	6.6	306	1735	6.1	6	0
Mercedes-Benz CLS 500 5.5 V8 2007	5461	4.2	388	1760	5.4	8	0
Chevrolet Camaro SS 6.2 V8 2018	6162	6.4	461	1709	4.2	8	0
Pagani Zonda F 7.3 V12 2006	7291	3.0	602	1230	3.6	12	0

REALIZAR A REGRESSÃO

```
#@title Regressão Primeira opção
#Regressão com a fórmula import statsmodels.formula.api as smf
regmul = smf.ols('consumo ~ cap_vol + power + weight', data = dfcars)
#Realizar o processo de modelagem
res = regmul.fit()
```

- **Definição das variáveis**
  - Variável resposta y = consumo
  - Variáveis preditoras x_i= cap_vol, potência, peso

Resultados

```
#Resultado detalhado
print(res.summary())
```

ANALISAR OS RESULTADOS

OLS Regression Results

Dep. Variable:	consumo	R-squared:	0.858
Model:	OLS	Adj. R-squared:	0.811
Method:	Least Squares	F-statistic:	18.17
Date:	Wed, 06 Apr 2022	Prob (F-statistic):	0.000369
Time:	05:53:31	Log-Likelihood:	-18.408
No. Observations:	13	AIC:	44.82
Df Residuals:	9	BIC:	47.08
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	13.8973	1.640	8.475	0.000	10.188	17.607
cap_vol	-0.0005	0.001	-0.417	0.686	-0.003	0.002
power	-0.0061	0.016	-0.378	0.714	-0.043	0.030
weight	-0.0021	0.001	-1.464	0.177	-0.005	0.001

Omnibus:	4.645	Durbin-Watson:	2.711
Prob(Omnibus):	0.098	Jarque-Bera (JB):	2.326
Skew:	1.024	Prob(JB):	0.312
Kurtosis:	3.315	Cond. No.	1.89e+04

$$R^2_{3(adj)} = 0{,}811$$

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$$

$$y = 13{,}8973 - 0{,}0005 x_1 - 0{,}0061 x_2 - 0{,}0021 x_3$$

UTILIDADE DO MODELO

OLS Regression Results

Dep. Variable:	consumo	R-squared:	0.858
Model:	OLS	Adj. R-squared:	0.811
Method:	Least Squares	F-statistic:	18.17
Date:	Wed, 06 Apr 2022	Prob (F-statistic):	0.000369
Time:	05:53:31	Log-Likelihood:	-18.408
No. Observations:	13	AIC:	44.82
Df Residuals:	9	BIC:	47.08
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	13.8973	1.640	8.475	0.000	10.188	17.607
cap_vol	-0.0005	0.001	-0.417	0.686	-0.003	0.002
power	-0.0061	0.016	-0.378	0.714	-0.043	0.030
weight	-0.0021	0.001	-1.464	0.177	-0.005	0.001

Omnibus:	4.645	Durbin-Watson:	2.711
Prob(Omnibus):	0.098	Jarque-Bera (JB):	2.326
Skew:	1.024	Prob(JB):	0.312
Kurtosis:	3.315	Cond. No.	1.89e+04

$F_{k,n-(k+1)} \rightarrow F_{crit}$

$f \geq F_{crit}$ rejeitar H_0

$18{,}17 \geq 2{,}813$?? SIM

Portanto rejeitar H_0

Se α=0,1 (90% de significância)

```
#@title Cálculo do Fcrit (tabela)
import scipy.stats
F=res.fvalue
k=res.df_model # grau do modelo
n=res.nobs # num. amostras
dfn=k
dfd=n-(k+1)
alpha = 0.1 #nível de confiança.
F_critico=scipy.stats.f.ppf(1-alpha, dfn, dfd)
print("F_crit=",F_critico) #tabela F-dist
```

```
F_crit= 2.8128629971823895
```

TESTE DE HIPÓTESE DOS PARÂMETROS

OLS Regression Results

Dep. Variable:	consumo	R-squared:	0.858
Model:	OLS	Adj. R-squared:	0.811
Method:	Least Squares	F-statistic:	18.17
Date:	Wed, 06 Apr 2022	Prob (F-statistic):	0.000369
Time:	05:53:31	Log-Likelihood:	-18.408
No. Observations:	13	AIC:	44.82
Df Residuals:	9	BIC:	47.08
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	13.8973	1.640	8.475	0.000	10.188	17.607
cap_vol	-0.0005	0.001	-0.417	0.686	-0.003	0.002
power	-0.0061	0.016	-0.378	0.714	-0.043	0.030
weight	-0.0021	0.001	-1.464	0.177	-0.005	0.001

Omnibus:	4.645	Durbin-Watson:	2.711
Prob(Omnibus):	0.098	Jarque-Bera (JB):	2.326
Skew:	1.014	Prob(JB):	0.312
Kurtosis:	3.715	Cond. No.	1.89e+04

Se $|t| \geq t_{crit}$ rejeitar H_0

$|t| \geq 1{,}7959$ **?? Se cumpre apenas para $\boldsymbol{\beta_0}$**

Portanto aceitar a Hipótese nula $H_0: \beta_1 = \beta_2 = \beta_3 = 0$

Se α=0,1 (90% de significância)

```
#@title Cálculo do Fcrit (tabela)
import scipy.stats
F=res.fvalue
k=res.df_model # grau do modelo
n=res.nobs # num. amostras
dfn=k
dfd=n-(k+1)
alpha = 0.1 #nível de confiança.
F_critico=scipy.stats.f.ppf(1-alpha, dfn, dfd)
print("F_crit=",F_critico) #tabela F-dist
```

```
F_crit= 2.8128629971823895
```

Valor-p <α=0,1 ?
Se cumpreapenas para $\boldsymbol{\beta_0}$
Hipótese nula
$H_0: \beta_1 = \beta_2 = \beta_3 = 0$

REFAZER A REGRESSÃO

```
1 #@title Regressão Segunda opção
2 regmul2 = smf.ols('consumo ~ cap_vol + power + weight + nu_cy', data = dfcars)
3 #Realizar o processo de modelagem
4 res2 = regmul2.fit()
5 #Resultado detalhado
6 print(res2.summary())
```

- **Definição das variáveis**
  - Variável resposta y = consumo
  - Variáveis preditoras x_i= cap_vol, potência, peso, número de cilindros.

REFAZER A REGRESSÃO: INCREMENTO DE VARIÁVEL

OLS Regression Results

Dep. Variable:	consumo	R-squared:	0.977
Model:	OLS	Adj. R-squared:	0.966
Method:	Least Squares	F-statistic:	85.91
Date:	Wed, 06 Apr 2022	Prob (F-statistic):	1.32e-06
Time:	06:06:10	Log-Likelihood:	-6.5200
No. Observations:	13	AIC:	23.04
Df Residuals:	8	BIC:	25.86
Df Model:	4		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	22.0213	1.437	15.329	0.000	18.708	25.334
cap_vol	0.0003	0.001	0.498	0.632	-0.001	0.002
power	0.0174	0.008	2.247	0.055	-0.000	0.035
weight	-0.0054	0.001	-6.850	0.000	-0.007	-0.004
nu_cy	-2.0584	0.318	-6.467	0.000	-2.792	-1.324

Omnibus:	1.730	Durbin-Watson:	1.192
Prob(Omnibus):	0.421	Jarque-Bera (JB):	0.890
Skew:	0.169	Prob(JB):	0.641
Kurtosis:	1.764	Cond. No.	3.97e+04

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4$$
$$y = 22{,}02 - 0{,}0003 x_1 - 0{,}0174 x_2 - 0{,}0054 x_3 - 2.0584 x_4$$

$$R^2_{3(adj)} = 0{,}811$$
$$R^2_{4(adj)} = 0{,}966$$
$$R^2_{4(adj)} > R^2_{3(adj)}$$

TESTE DE HIPÓTESE DOS PARÂMETROS

OLS Regression Results

Dep. Variable:	consumo	R-squared:	0.977
Model:	OLS	Adj. R-squared:	0.966
Method:	Least Squares	F-statistic:	85.91
Date:	Wed, 06 Apr 2022	Prob (F-statistic):	1.32e-06
Time:	06:06:10	Log-Likelihood:	-6.5200
No. Observations:	13	AIC:	23.04
Df Residuals:	8	BIC:	25.86
Df Model:	4		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	22.0213	1.437	15.329	0.000	18.708	25.334
cap_vol	0.0003	0.001	0.498	0.632	-0.001	0.002
power	0.0174	0.008	2.247	0.055	-0.000	0.035
weight	-0.0054	0.001	-6.850	0.000	-0.007	-0.004
nu_cy	-2.0584	0.318	-6.467	0.000	-2.792	-1.324

Omnibus:	1.730	Durbin-Watson:	1.192
Prob(Omnibus):	0.421	Jarque-Bera (JB):	0.890
Skew:	0.169	Prob(JB):	0.641
Kurtosis:	1.764	Cond. No.	3.97e+04

Se α=0,1 (90% de significância)

```
#@title Cálculo t_crit (tabela t-student)
from scipy.stats import t
alpha = 0.1  # significia = 5%
df = 11 # graus de liberdade
v = t.ppf(1 - alpha/2, df)
tt=v
print(f't_crit=: {v}')
```

```
t_crit=: 1.7958848187036691
```

Valor-p <α=0,1 ?

Se cumpre para todos os β_i menos para β_2

Portanto retirar β_2

Se $|t| \geq t_{crit}$ rejeitar H_0

$|t| \geq 1{,}7959$ **?? Se cumpre para todos os β_i menos para β_2**

Portanto aceitar a Hipótese nula $H_0: \beta_2 = 0$ mas manter os outros β_i

REFAZER A REGRESSÃO: RETIRAR VARIÁVEL NÃO SIGNIFICATIVA

```
#@title Regressão Terceira opção
regmul3 = smf.ols('consumo ~ power + weight + nu_cy', data = dfcars)
#Realizar o processo de modelagem
res3 = regmul3.fit()
#Resultado detalhado
print(res3.summary())
```

- **Definição das variáveis**
  - Variável resposta y = consumo
  - Variáveis preditoras x_i= ~~cap_vol~~, potência, peso, número de cilindros.

REFAZER A REGRESSÃO: RETIRAR VARIÁVEL NÃO SIGNIFICATIVA

OLS Regression Results

Dep. Variable:	consumo	R-squared:	0.977
Model:	OLS	Adj. R-squared:	0.969
Method:	Least Squares	F-statistic:	124.9
Date:	Wed, 06 Apr 2022	Prob (F-statistic):	1.19e-07
Time:	06:09:22	Log-Likelihood:	-6.7184
No. Observations:	13	AIC:	21.44
Df Residuals:	9	BIC:	23.70
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	21.9086	1.358	16.132	0.000	18.836	24.981
power	0.0203	0.005	4.092	0.003	0.009	0.031
weight	-0.0053	0.001	-7.193	0.000	-0.007	-0.004
nu_cy	-2.0227	0.297	-6.813	0.000	-2.694	-1.351

Omnibus:	2.118	Durbin-Watson:	1.228
Prob(Omnibus):	0.347	Jarque-Bera (JB):	0.936
Skew:	0.089	Prob(JB):	0.626
Kurtosis:	1.698	Cond. No.	1.43e+04

$$R^2_{3(adj)} = 0{,}811$$
$$R^2_{4(adj)} = 0{,}966$$
$$R^2_{3a(adj)} = 0{,}969$$
$$R^2_{3a(adj)} > R^2_{4(adj)} > R^2_{3(adj)}$$

$$F_{crit-3} = 18{,}17$$
$$F_{crit-4} = 85{,}91$$
$$F_{crit-3a} = 124{,}9$$

valores-p <α =0,05

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$$
$$y = 21{,}9086 - 0{,}0203 x_1 - 0{,}053 x_2 - 2{,}0227 x_3$$

RESÍDUOS E RESÍDUOS PADRONIZADOS

Obter:

- Valores previstos (y_pred)
- Resíduos (resi)
- Resíduos padronizados (stdresid)
- Proporção entre resíduos e resíduos padronizados (prop)

```
#@title Gráficos dispersão e resíduos
#Valores previstos e residuos
y_pred=list(res3.predict())
resi=list(res3.resid)
#crear instancia influence
influence = res3.get_influence()
#obter residuos standardizados
stdresid = list(influence.resid_studentized_internal)
prop=np.divide(resi,stdresid) #e/e*
```

RESÍDUOS E RESÍDUOS PADRONIZADOS

Obter:

- Valores previstos (y_pred)
- Resíduos (resi)
- Resíduos padronizados (stdresid)
- Proporção entre resíduos e resíduos padronizados (prop)

```
1 y=list(dfcars['consumo'])
2 dftab = pd.DataFrame(list(zip(y,y_pred,resi,stdresid,prop)),
3                       columns =["y","y_p","e","e*","e/e*"])
4 dftab.head(10)
```

RESÍDUOS E RESÍDUOS PADRONIZADOS

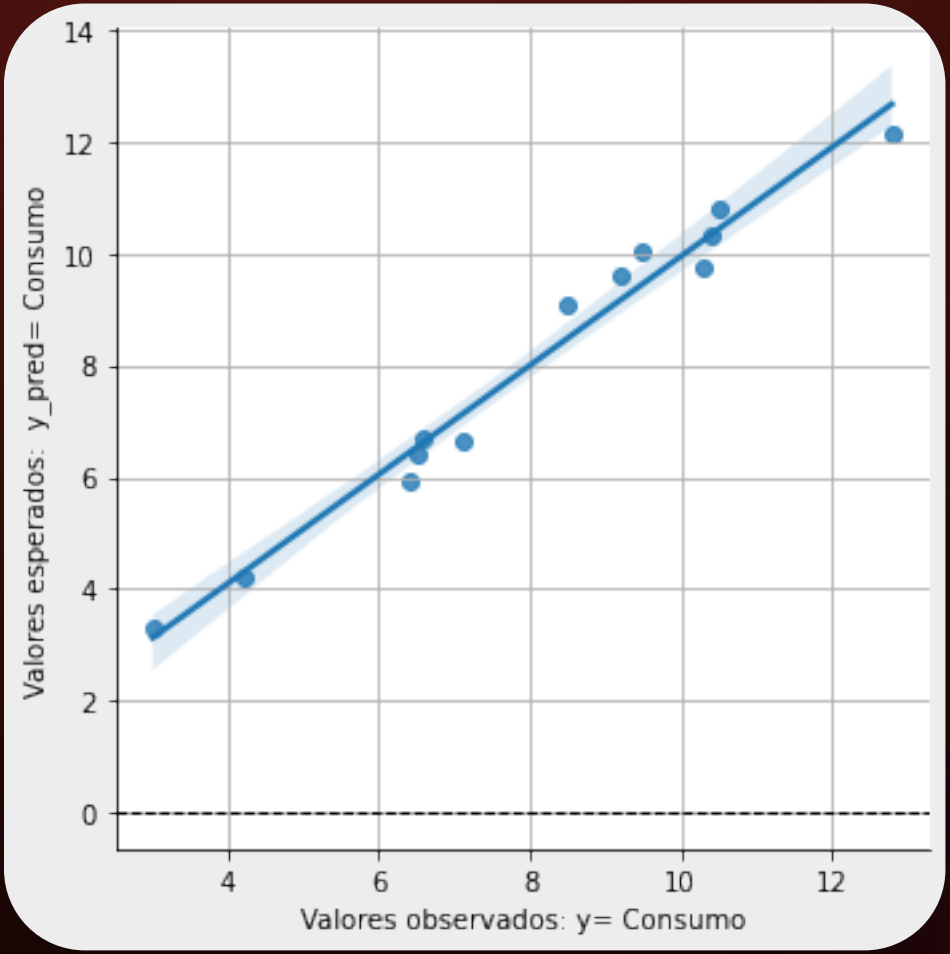
Obter:

- Valores previstos (y_pred)
- Resíduos (resi)
- Resíduos padronizados (stdresid)
- Proporção entre resíduos e resíduos padronizados (prop)

```
1 y=list(dfcars['consumo'])
2 dftab = pd.DataFrame(list(zip(y,y_pred,resi,stdresid,prop)),
3                       columns =["y","y_p","e","e*","e/e*"])
4 dftab.head(10)
```

	y	y_p	e	e*	e/e*
0	10.4	10.345134	0.054866	0.140443	0.390663
1	12.8	12.178190	0.621810	1.617789	0.384358
2	10.3	9.769516	0.530484	1.217066	0.435871
3	10.5	10.791847	-0.291847	-0.675226	0.432221
4	8.5	9.113044	-0.613044	-1.359483	0.450939
5	9.5	10.046835	-0.546835	-1.228797	0.445017
6	9.2	9.621654	-0.421654	-0.927807	0.454463
7	6.5	6.392115	0.107885	0.272351	0.396125
8	7.1	6.624970	0.475030	1.067406	0.445033
9	6.6	6.707464	-0.107464	-0.252444	0.425695

GRÁFICOS DE DIAGNÓSTICO



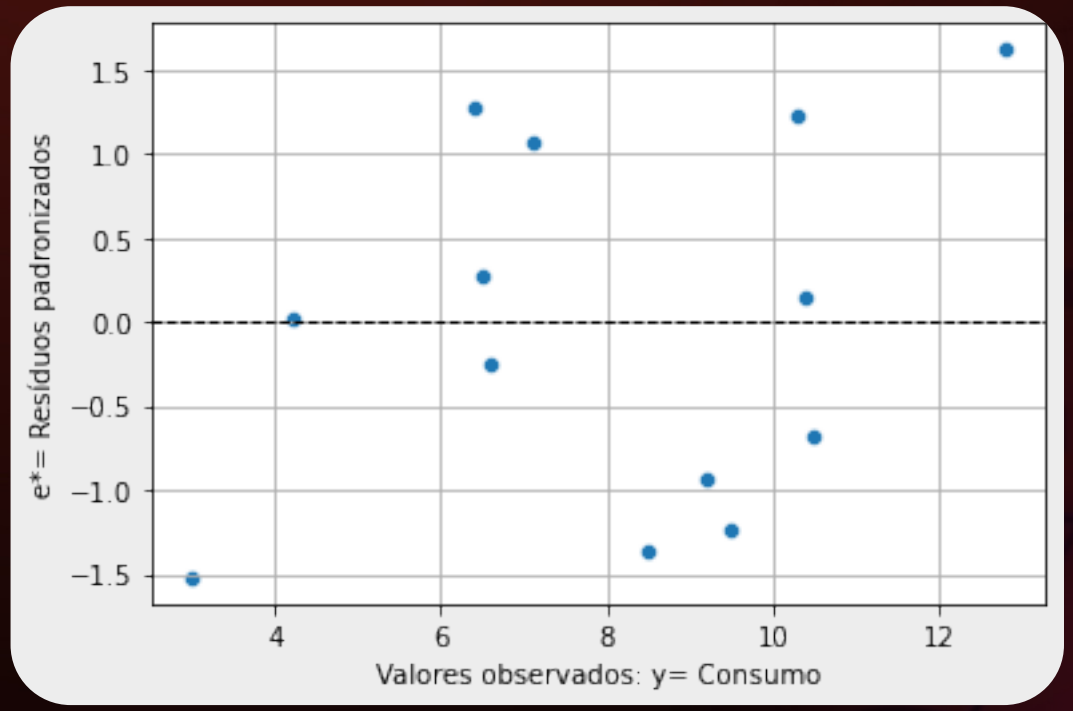
y vs. $\hat{y}$

```
#@title Gráficos de diagnóstico 2 (y vs. yc)
sns.lmplot(x='y', y='y_p', data=dftab);plt.grid(True)
plt.xlabel('Valores observados: y= Consumo')
plt.ylabel('Valores esperados:  y_pred= Consumo')
plt.axhline(y=0, color='black', linestyle='--', linewidth=1)
plt.show()
```

GRÁFICOS DE DIAGNÓSTICO

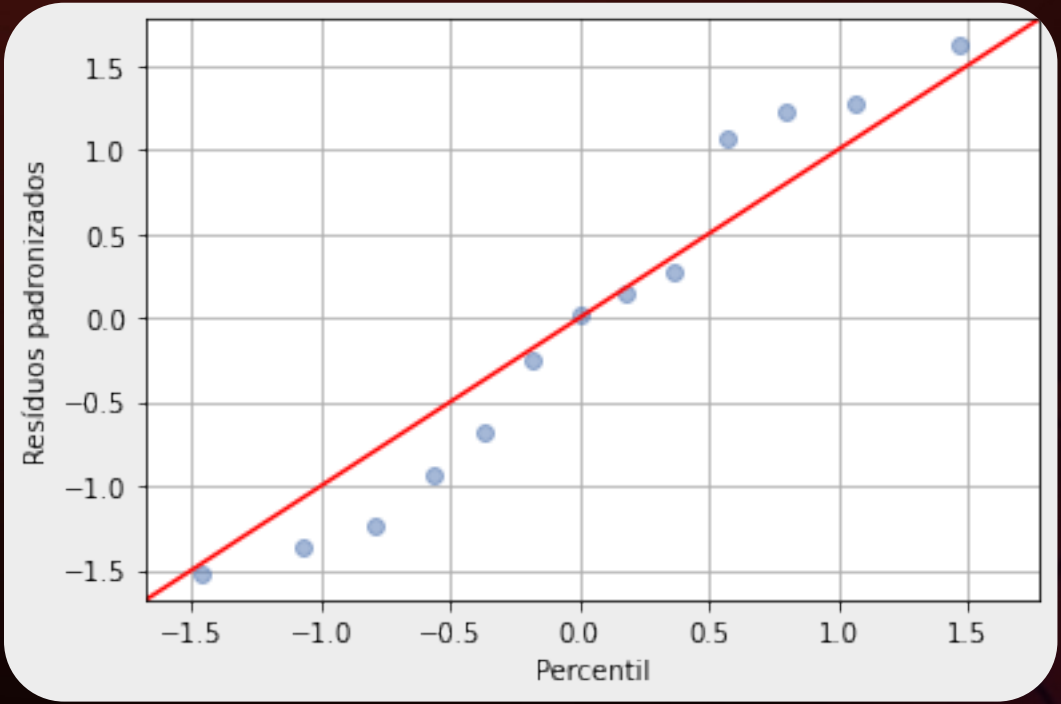
```
#@title Gráficos de diagnóstico 3 (resíduos padronizados vs. x)
sns.scatterplot(x='y', y='e*', data=dftab);plt.grid(True)
plt.xlabel('Valores observados: y= Consumo')
plt.ylabel('e*= Resíduos padronizados')
plt.axhline(y=0, color='black', linestyle='--', linewidth=1)
plt.show()
```

y vs. e^*



GRÁFICOS DE DIAGNÓSTICO

```
#@title Gráficos de diagnóstico 4
#Verificar a normalidade dos resíduos
#qqpolot vs. normal distribution
QQ = ProbPlot(influence.resid_studentized_internal)
plot_lm_2 = QQ.qqplot(line='45', alpha=0.5, color='#4C72B0', lw=1)
plot_lm_2.axes[0].set_xlabel('Percentil')
plot_lm_2.axes[0].set_ylabel('Resíduos padronizados')
plt.grid(True)
```



Normalidade dos resíduos padronizados

AMOSTRAS INFLUENTES

```
#Analisar amostras influentes
infl = res3.get_influence()
#leverage
print(infl.hat_matrix_diag)
```

$$h_{jj} > \frac{2(k+1)}{n}$$

```
[0.35806038 0.37861409 0.20088976 0.21422029 0.14468658 0.16700463
 0.13126423 0.33998544 0.16694511 0.23776741 0.32914283 0.46714838
 0.86427087]
```

```
#Valores dos resíduos (influencias internas)
#residus = res3.resid.as_matrix() #residuals
leviers = infl.hat_matrix_diag #leverage
sigma_err = np.sqrt(res3.scale) #regression standard error
res_stds = stdresid/(sigma_err*np.sqrt(1.0-leviers))
print(res_stds)
```

```
[ 0.35949995  4.20906954  2.79225934 -1.5622264  -3.01478277 -2.7612392
 -2.04154318  0.68753833  2.39848859 -0.59301501  0.05304166  3.59442848
 -8.46147643]
```

AMOSTRAS INFLUENTES

```
#Limiar
#threshold leverage
seuil_levier = 2*(pa+1)/na
print(seuil_levier)
#identification
atyp_levier = leviers > seuil_levier
print(atyp_levier)
```

$$s_h = 2 \times \frac{p+1}{n} \qquad h_i > s_h$$

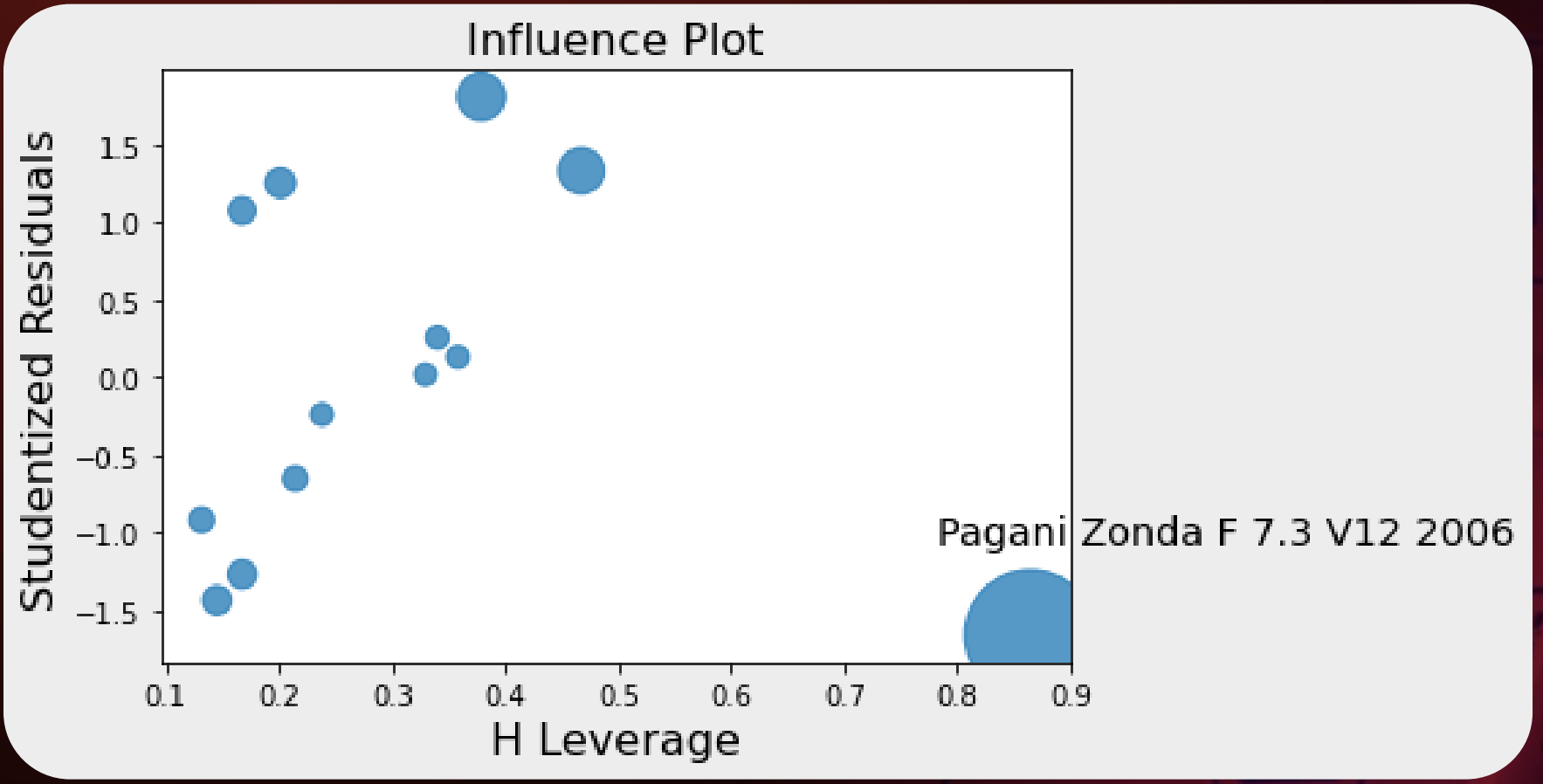
$$h_{jj} > \frac{2(k+1)}{n}$$

```
0.6153846153846154
[False False False False False False False False False False False False False
  True]
```

```
#Quais carros ultrapassam esse limiar
print(dfcars.index[atyp_levier],leviers[atyp_levier])
```

```
Index(['Pagani Zonda F 7.3 V12 2006'], dtype='object', name='brand_model_year') [0.86427087]
```

AMOSTRAS INFLUENTES



AMOSTRAS INFLUENTES

```python
#limiares dos resíduos (influencias externas)
import scipy
seuil_stud = scipy.stats.t.ppf(0.975,df=na-pa-2)
print(seuil_stud)
#detection - absolute value > threshold
atyp_stud = np.abs(res_studs) > seuil_stud
#which ones?
print(dfcars.index[atyp_stud],res_studs[atyp_stud])
```

```
2.3060041350333704
Index(['FIAT Strada 1.4 2016', 'Honda Civic EXR 2016',
       'Ford Focus 2.0 GLX 2012', 'AUDI A4 3.2 V6 Fsi 2011'
      dtype='object', name='brand_model_year') [ 7.1997264
```

```python
#Observações suspeitas considerando ambos critérios
pbm_infl = np.logical_or(atyp_levier,atyp_stud)
print(dfcars.index[pbm_infl])
```

```
Index(['FIAT Strada 1.4 2016', 'Honda Civic EXR 2016',
       'Ford Focus 2.0 GLX 2012', 'AUDI A4 3.2 V6 Fsi 2011'
       'Pagani Zonda F 7.3 V12 2006'],
      dtype='object', name='brand_model_year')
```

$$s_t = t_{1-0.05/2}(n-p-2)$$

$$\left|t_i^*\right| > s_t$$

AMOSTRAS INFLUENTES

```
1 #Outros critérios para determinar amostras influentes
2 print(infl.summary_frame().filter(["hat_diag","student_resid","dffits","cooks_d"]))
```

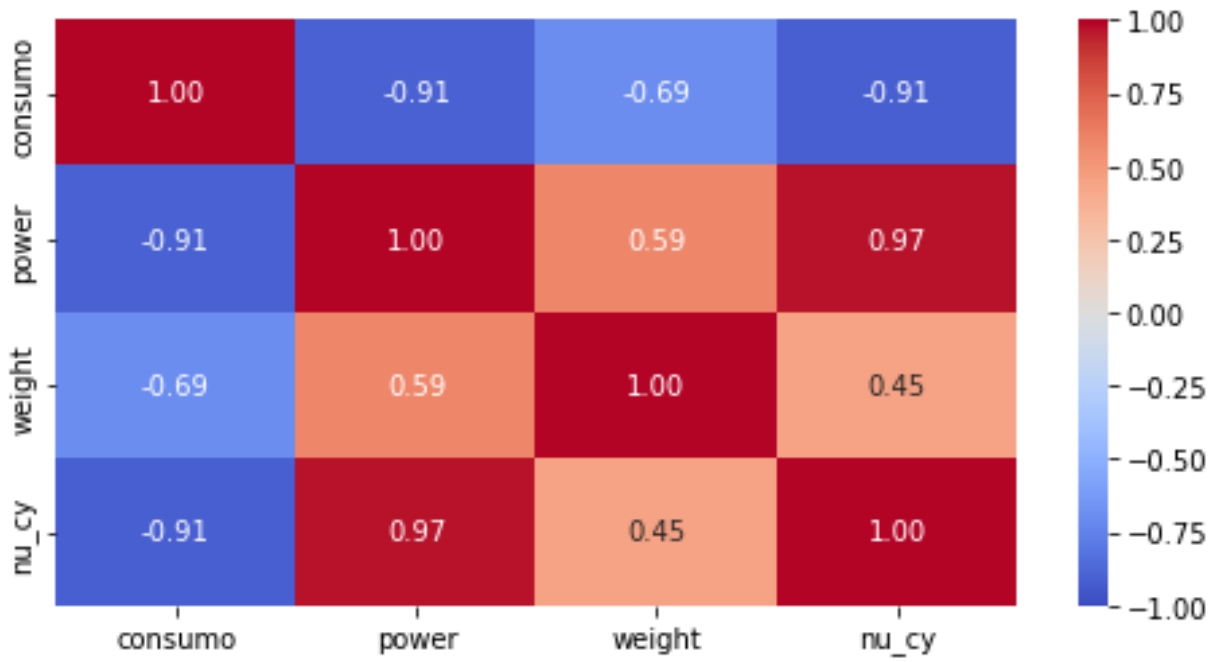
brand_model_year	hat_diag	student_resid	dffits	cooks_d
FIAT UNO Mille EP 1996	0.358060	0.132557	0.098999	0.002750
Hyundai HB20 Sense 2020	0.378614	1.811184	1.413775	0.398675
FIAT Strada 1.4 2016	0.200890	1.255412	0.629451	0.093093
VolksWagen GOL 1.6 2015	0.214220	-0.653374	-0.341147	0.031074
Chevrolet Cruze LTZ 1.8 2016	0.144687	-1.437841	-0.591374	0.078161
Honda Civic EXR 2016	0.167005	-1.269939	-0.568625	0.075681
Ford Focus 2.0 GLX 2012	0.131264	-0.919840	-0.357554	0.032517
BMW 325i 3.0 2012	0.339985	0.257840	0.185056	0.009552
AUDI A4 3.2 V6 Fsi 2011	0.166945	1.076826	0.482054	0.057082
Mercedes-Benz CLS 350 3.5 V6 2012	0.237767	-0.238853	-0.133402	0.004970
Mercedes-Benz CLS 500 5.5 V8 2007	0.329143	0.019972	0.013989	0.000055
Chevrolet Camaro SS 6.2 V8 2018	0.467148	1.333514	1.248596	0.358727
Pagani Zonda F 7.3 V12 2006	0.864271	-1.662190	-4.194391	3.677834

$$|DFFITS_i| > 2\sqrt{\frac{p+1}{n}}$$ DFFITS

$$D_i > \frac{4}{n-p-1}$$

MULTICOLINEARIDADE

```python
# Calcula a correlação entre os atributos numéricos
corr = dfex.corr()
# Resultado
plt.figure(figsize=(8, 4))
sns.heatmap(corr, cmap='coolwarm', annot=True, fmt=".2f", vmin=-1, vmax=1)
plt.show()
```



MULTICOLINEARIDADE

Existe multicolinearidade se > 0.8

```
1 print(corr)
```

	consumo	power	weight	nu_cy
consumo	1.000000	-0.905766	-0.687542	-0.913702
power	-0.905766	1.000000	0.592254	0.968908
weight	-0.687542	0.592254	1.000000	0.445695
nu_cy	-0.913702	0.968908	0.445695	1.000000

Existe multicolinearidade se a relação >R^2

$$R^2_{3a(adj)} = 0{,}969$$

```
1 #Klein's rule of thumb
2 mc2 = corr**2
3 print(mc2)
```

	consumo	power	weight	nu_cy
consumo	1.000000	0.820412	0.472714	0.834851
power	0.820412	1.000000	0.350764	0.938783
weight	0.472714	0.350764	1.000000	0.198644
nu_cy	0.834851	0.938783	0.198644	1.000000

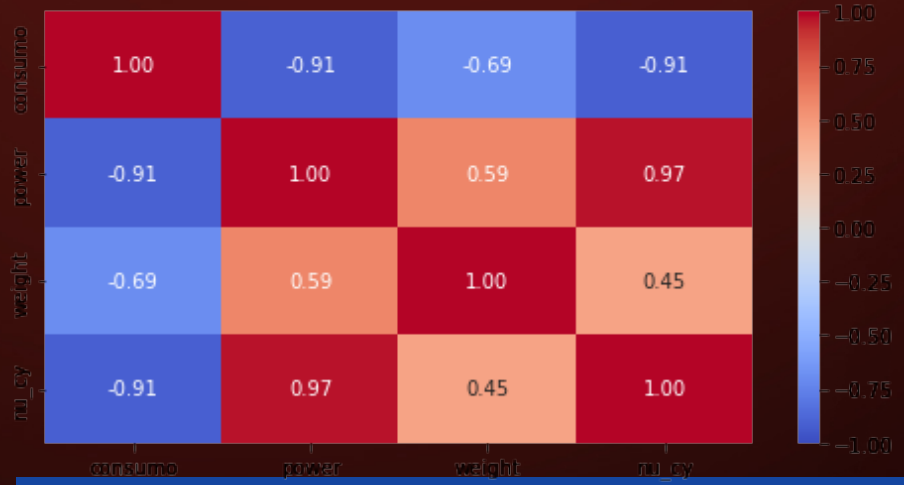
Existe multicolinearidade se VIF > 4

```
1 #VIF criterion
2 vif = np.linalg.inv(mc2)
3 print(vif)
```

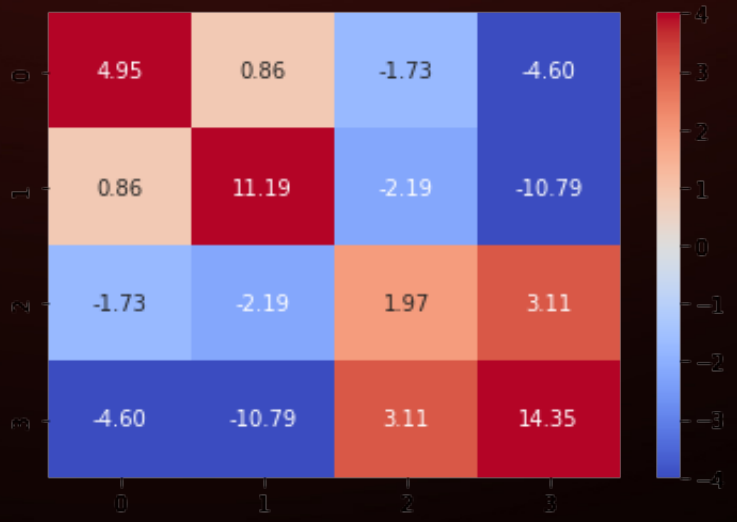
```
[[  4.94510596   0.86342415  -1.727545    -4.59582941]
 [  0.86342415  11.18956754  -2.18959009 -10.79045883]
 [ -1.727545    -2.18959009   1.96748437   3.10696401]
 [ -4.59582941 -10.79045883   3.10696401  14.34955472]]
```

MULTICOLINEARIDADE

Existe multicolinearidade se > 0.8



Existe multicolinearidade se VIF > 4



Existe multicolinearidade se a relação $>R^2$

$$R^2_{3a(adj)} = 0{,}969$$

```
sns.heatmap(vif, cmap='coolwarm', annot=True, fmt=".2f", vmin=-4, vmax=4)
```

MODELAGEM E INFERÊNCIA ESTATÍSTICA

Critérios de escolha em modelos de regressão múltipla - Exercício