Análise de Dados Aplicada à Computação

REGRESSÃO

Prof. M.Sc. Howard Roatti

Sumário

1. Definição de Regressão
2. Regressão Linear Simples
3. Regressão Linear Múltipla
4. Avaliação de Regressão

Regressão

- A Regressão é um modelo de análise de dados que visa estudar o relacionamento da(s) variável(is) independente(s) com a variável dependente.
- Diferente do coeficiente de correlação de Pearson, que mostra o comportamento das variáveis medindo a força da relação, a Regressão quantifica a natureza do relacionamento.

Regressão

- Exemplos de Relacionamentos Analisados (1/2):

1. Variação de gastos familiares com alimentação em decorrência do quanto de renda a família ganha;
2. Variação da concessão de limite no cartão de crédito em decorrência do salário;
3. Crescimento da taxa de criminalidade, relacionado com o crescimento na taxa de desemprego.

Regressão

- Exemplos de Relacionamentos Analisados (2/2):

4. Renda semanal e despesas de consumo;
5. Variação dos salários e taxa de desemprego;
6. Demandas dos produtos e a publicidade do mesmo;

Regressão

- Se considerarmos que apenas uma variável independente (X) causa algum efeito sobre a variável dependente (Y), estaremos tratando de uma **Regressão Simples**

- Porém, se tivermos um conjunto de variáveis independentes (X_i, onde i = 1..n), então estaremos tratando de uma análise de **Regressão Múltipla**.

Regressão

- Um modelo de **Regressão Linear** é uma equação matemática que fornece uma relação linear, ou seja, uma reta entre duas variáveis.

$$y = a + bx$$

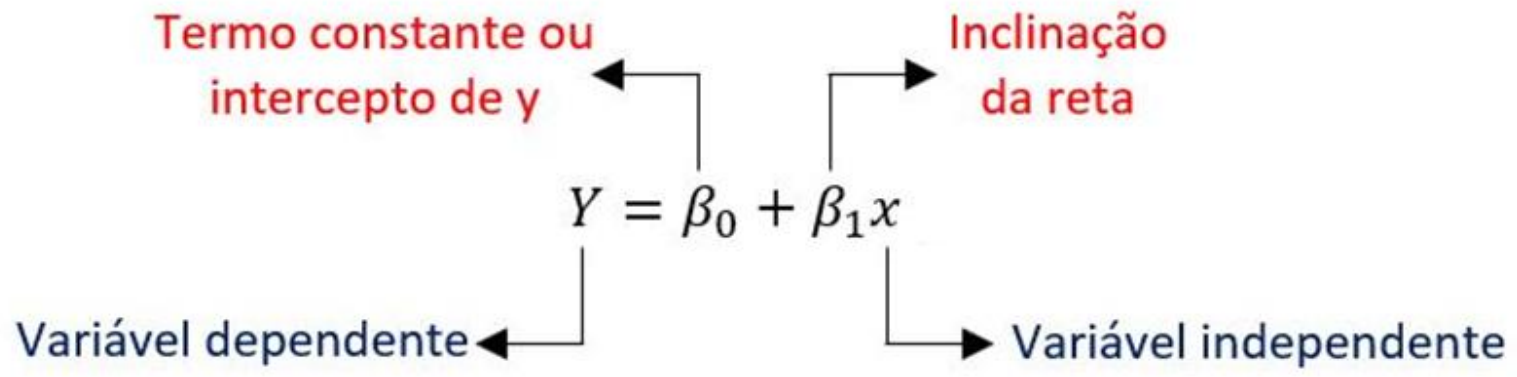
Matemática

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$$

Estatística

- y é igual à b vezes x, mais uma constante a
- y-hat é igual à Beta-hat$_1$ vezes x, mais uma constante Beta-hat$_0$

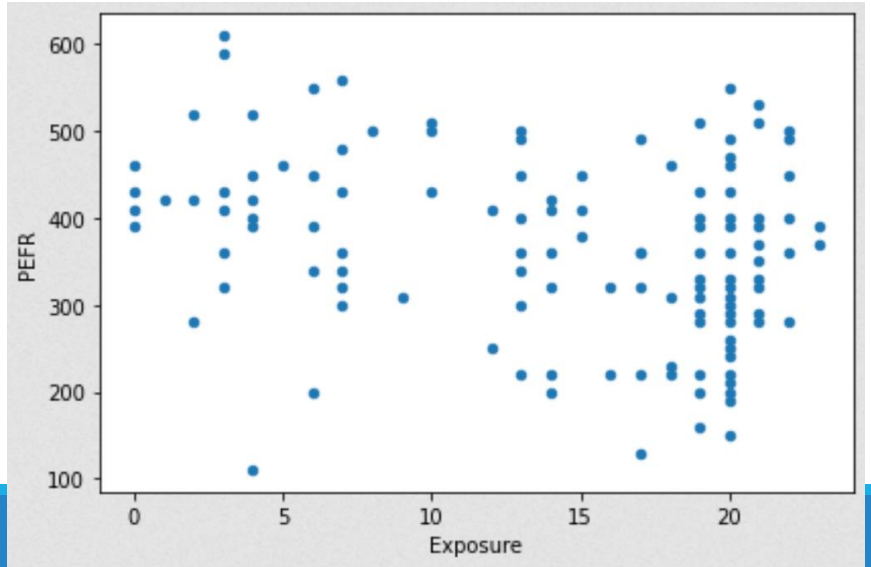
Regressão



- Cada conjunto de valores do estimador beta zero e beta 1 fornece uma linha reta diferente
- O coeficiente de x (inclinação) fornece a quantidade de variação em y

Regressão

```
lung = pd.read_csv('LungDisease.csv')
lung.plot.scatter(x='Exposure', y='PEFR')

plt.tight_layout()
plt.show()
```



Dataset : Doença Respiratória
PEFR : Taxa do Pico de Fluxo Expiratório
[Peak Expiratory Flow Rate (L/min)]
Exposure: Tempo de exposição

$$\widehat{PEFR} = \hat{\beta}_0 + \hat{\beta}_1 Exposure$$

A regressão linear simples tenta encontrar a melhor linha para prever a resposta, nesse caso a variável PEFR como uma função da variável preditora Exposure.

Regressão

- Como o modelo é ajustado aos dados?
- Inclinação → $\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x})}{\sum_{i=1}^{n}(x_i - \bar{x})}$
- Intercepto → $\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$

Regressão

```
predictors = ['Exposure']
outcome = 'PEFR'
model = LinearRegression()
model.fit(lung[predictors], lung[outcome])

print(f'Intercept: {model.intercept_:.3f}')
print(f'Coefficient Exposure: {model.coef_[0]:.3f}')
```

```
Intercept(PEFR): 424.583
Coefficient (Exposure): -4.185
```

$$\widehat{PEFR} = \hat{\beta}_0 + \hat{\beta}_1 Exposure$$

A interpretação que podemos ter em relação a esse resultado é que, como o intercepto vale 424.583, quando a exposição do trabalhador for igual a zero, a expiração será equivalente à 424.583 e a cada ano que o trabalhador se expõe a expiração diminui em -4.185 unidades.

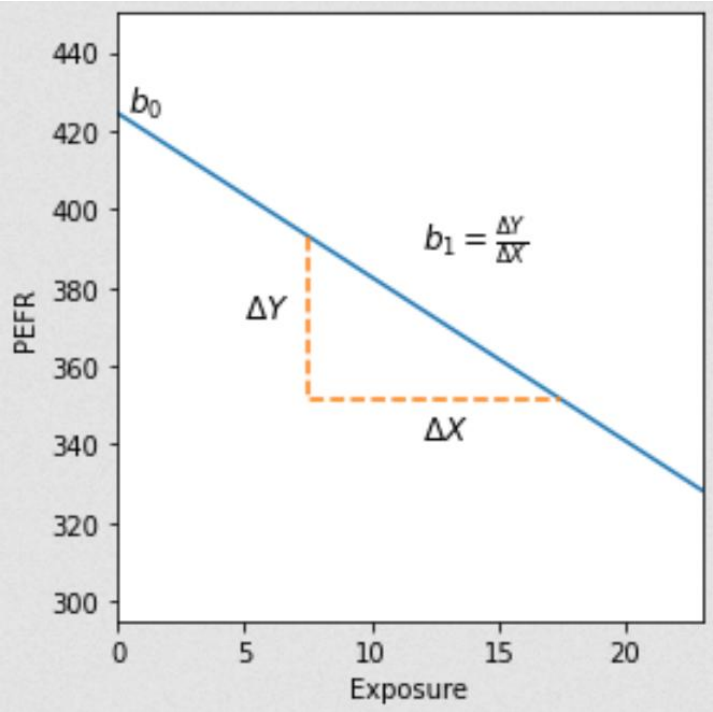
Regressão

- Python

```
fig, ax = plt.subplots(figsize=(4, 4))
ax.set_xlim(0, 23)
ax.set_ylim(295, 450)
ax.set_xlabel('Exposure')
ax.set_ylabel('PEFR')

y1 = model.predict(pd.DataFrame({"Exposure": [0, 23]}))

ax.plot((0, 23), (y1[0], y1[1]))
ax.text(0.4, model.intercept_, r'$b_0$', size='larger')

y2 = model.predict(pd.DataFrame({"Exposure": [7.5, 17.5]}))
ax.plot((7.5, 7.5, 17.5), (y2[0], y2[1], y2[1]), '--')

ax.text(5, np.mean(y), r'$\Delta Y$', size='larger')
ax.text(12, y[1] - 10, r'$\Delta X$', size='larger')
ax.text(12, 390, r'$b_1 = \frac{\Delta Y}{\Delta X}$', size='larger')

plt.tight_layout()
plt.show()
```

Regressão

- Python



Regressão

- A equação incluindo o erro residual ficaria assim:

$$y_i = \beta_0 + \beta_1 x_1 + e_i$$

- Onde e_i pode ser estimado por:

$$\hat{e}_i = y_i - \hat{y}_i$$

- E y-hat será estimado por cada par de valores (y_i, x_i):

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$$

- y_i é o valor real e y-hat$_i$ é o valor calculado pelo modelo de regressão

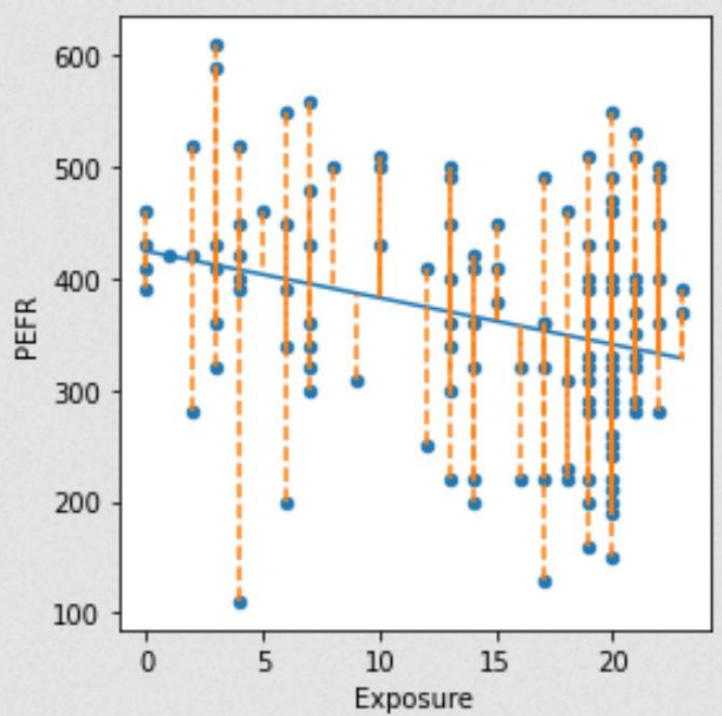
Regressão

- Python

```
predictors = ['Exposure']
outcome = 'PEFR‘

fitted = model.predict(lung[predictors])
residuals = lung[outcome] – fitted

ax = lung.plot.scatter(x='Exposure', y='PEFR', figsize=(4, 4))
ax.plot(lung.Exposure, fitted)
for x, yactual, yfitted in zip(lung.Exposure, lung.PEFR, fitted):
    ax.plot((x, x), (yactual, yfitted), '--', color='C1')

plt.tight_layout()
plt.show()
```

Regressão

- Python



Regressão Múltipla

- O que muda na regressão múltipla é o número de variáveis independentes: $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + e$
- Ao invés de uma linha, teremos um modelo linear.
- Todos os outros conceitos permanecem os mesmos, bem como estimar valores: $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{1,i} + +\hat{\beta}_2 x_{2,i} + \cdots + \hat{\beta}_p x_{p,i}$

Regressão Múltipla

- Python

```
subset = ['AdjSalePrice', 'SqFtTotLiving', 'SqFtLot', 'Bathrooms', 'Bedrooms',
'BldgGrade']

house = pd.read_csv('house_sales.csv', sep='\t')
print(house[subset].head())
```

	AdjSalePrice	SqFtTotLiving	SqFtLot	Bathrooms	Bedrooms	BldgGrade
1	300805.0	2400	9373	3.00	6	7
2	1076162.0	3764	20156	3.75	4	10
3	761805.0	2060	26036	1.75	4	8
4	442065.0	3200	8618	3.75	5	7
5	297065.0	1720	8620	1.75	4	7

Regressão Múltipla

- Python

```
predictors = ['SqFtTotLiving', 'SqFtLot', 'Bathrooms',
              'Bedrooms', 'BldgGrade']
outcome = 'AdjSalePrice'

house_lm = LinearRegression()
house_lm.fit(house[predictors], house[outcome])

print(f'Intercept: {house_lm.intercept_:.3f}')
print('Coefficients:')
for name, coef in zip(predictors, house_lm.coef_):
    print(f' {name}: {coef}')
```

```
Intercept: -521871.368
Coefficients:
 SqFtTotLiving: 228.8306036024083
 SqFtLot: -0.06046682065305298
 Bathrooms: -19442.840398320997
 Bedrooms: -47769.955185214465
 BldgGrade: 106106.96307898074
```

Avaliação de Regressão

- Avaliando o Modelo:
  - **Raiz Quadrada do Erro Quadrático Médio**

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}}$$

  - **Erro-Padrão Residual →** $RSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{(n-p-1)}}$
  - **R2 →** $R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2}$
  - **Erro Médio Absoluto(MAE) →** $\frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$

Outras Métricas:
- MSE (Mean Squared Error)
- MSLE (Mean Squared Logarithmic Error)
- RMSLE (Root Mean Squared Logarithmic Error)
- MAPE (Mean Absolute Percentage Error)
- RAE (Relative Absolute Error)
- RSE (Relative Squared Error)

Avaliação de Regressão

- Python – Avaliando o Modelo

```
fitted = house_lm.predict(house[predictors])

RMSE = np.sqrt(mean_squared_error(house[outcome], fitted))

r2 = r2_score(house[outcome], fitted)

mae = mean_absolute_error(house[outcome], fitted)

rse = RSE(house[outcome], fitted)

print(f'RMSE: {RMSE:.2f}')
print(f'R²: {r2:.4f}')
print(f'MAE: {mae:.4f}')
print(f'RSE: {rse:.4f}')
```

```
RMSE: 261220.20
R²: 0.5406
MAE: 150660.9141
RSE: 261231.7123
```

Referências

- Bruce, P.; Bruce, A.; **Estatística Prática para Cientista de Dados**: 50 Conceitos Essenciais; Rio de Janeiro; Alta Books; 2019.
- Morettin, P. A.; Bussab, W. O.; **Estatística Básica**. 8 ed. São Paulo: Saraiva, 2013.
- https://oestatistico.com.br/regressao-linear-simples/

FAESA
CENTRO UNIVERSITÁRIO

Análise de Dados Aplicada à Computação

PROF. M.SC HOWARD ROATTI