

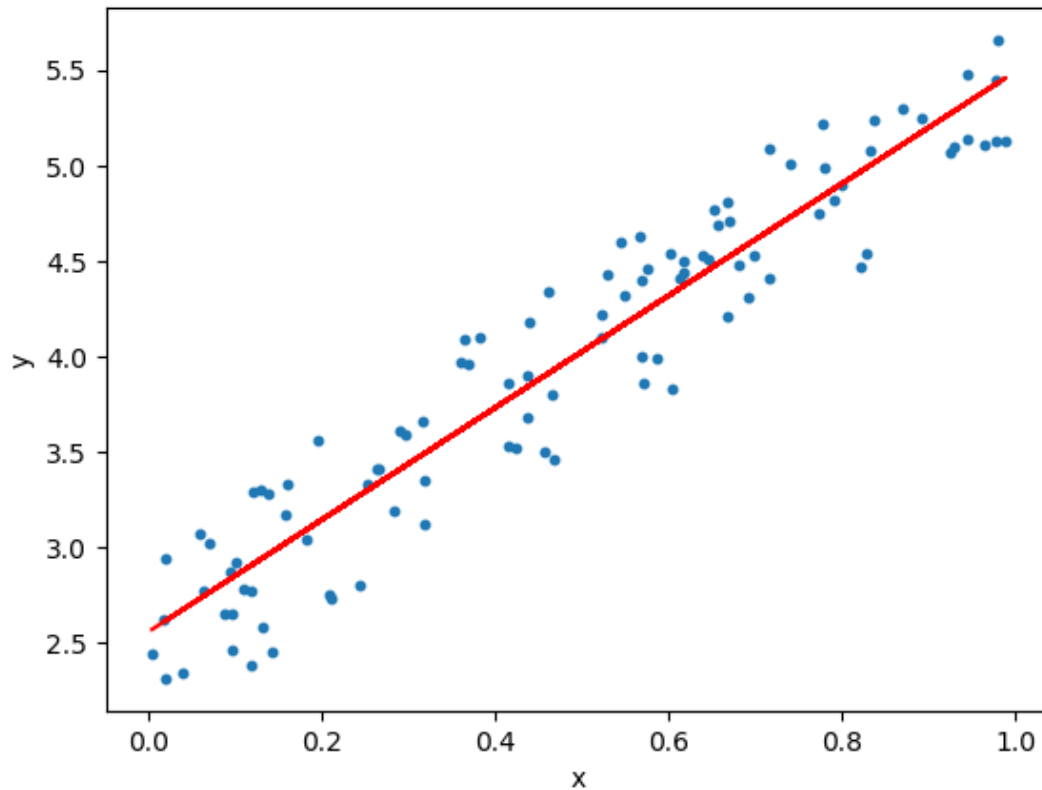
CURSO 2 – CD, AM E DM

REGRESSÃO UNIVARIADA E MULTIVARIADA MÉTRICAS DE REGRESSÃO

PROFA. ROSELI AP. FRANCELIN ROMERO



REGRESSÃO LINEAR UNIVARIADA



$$Y = W X + B$$

$W = ?$
 $B = ?$

X	Y
0.2	3.0
0.25	3.5
0.3	4.1
0.33	4.2
.	.
.	.



REGRESSÃO LINEAR UNIVARIADA

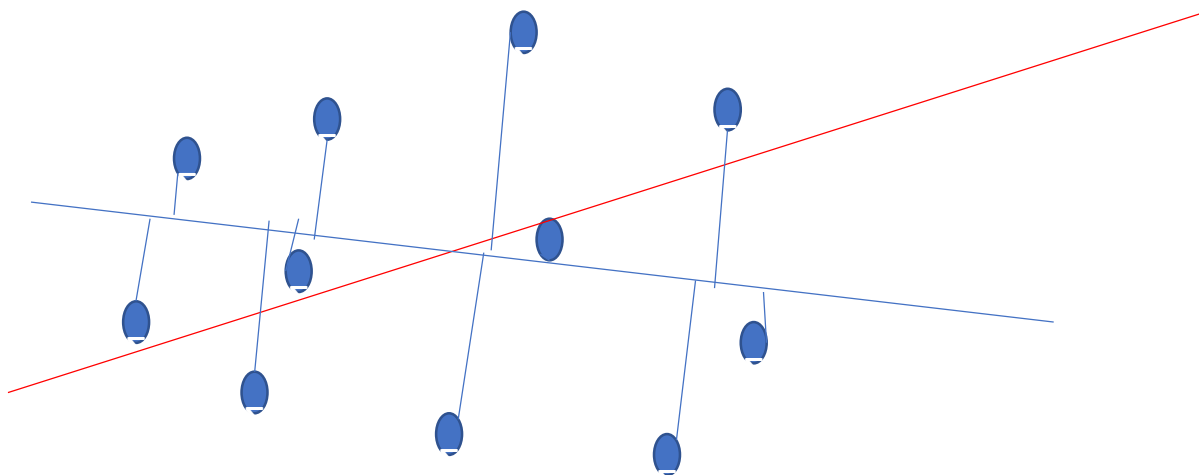
MÉTODO DOS MÍNIMOS QUADRADOS



RETA QUE **MELHOR** SE APROXIMA
DOS DADOS



Regressão Linear (mínimos quadrados)



$$\sum (Y_i - \hat{y}_i)^2 \quad \hat{y}_i = WX_i$$

$$\frac{1}{2} \sum (Y_i - WX_i)^2 ; W = ?$$

Derivar em rel. a W e igualar a zero

$$\frac{1}{2} (\sum Y_i^2 - 2 \sum Y_i WX_i + W^2 \sum X_i^2)$$

$$\sum Y_i X_i + W \sum X_i X_i = 0$$

$$W = (\sum Y_i X_i) / \sum X_i X_i$$

Forma matricial:

$$W = Y^t X / X^t X$$



REGRESSÃO LINEAR UNIVARIADA

- $Y = W X + B$
- $\hat{y} = W X$ (SUPOR $B = 0$)

X	Y	\hat{y}
X_0	Y_0	$W X_0$
X_1	Y_1	$W X_1$
....
X_i	Y_i	$W X_i$
...
X_m	Y_m	$W X_m$

$$\sum_{i=0}^m (Y_i - \hat{y}_i)^2 = \sum_{i=0}^m (Y_i - W X_i)^2$$


Minimizar esta soma

Forma matricial: $(Y - XW)^T (Y - XW)$
Derivando e igualando a zero

$$W = (X^T X)^{-1} X^T Y$$



REGRESSÃO LINEAR MULTIVARIADA

- $X = (X_0, X_1, X_2, \dots, X_n)$;  Y
- A SOLUÇÃO É A MESMA:

$$W = (X^T X)^{-1} X^T Y$$

inversa precisa existir



Métricas de Regressão

- MSE
- MAE
- R^2



MÉTRICAS DE REGRESSÃO

MSE – MEAN SQUARE ERROR

$$MSE = 1/n \sum_{I=0}^{n-1} (y_i - \hat{y}_i)^2$$

\hat{y} é vetor previsto
 y é o vetor dados
 n é no. de exemplos



MÉTRICAS DE REGRESSÃO

MAE – MEAN ABSOLUTE VALUE

$$MAE = \frac{1}{n} \sum_{i=0}^{n-1} (|y_i - \hat{y}_i|)$$

\hat{y} é vetor previsto
 y é o vetor dados
 n – no. de exemplos



MÉTRICAS DE REGRESSÃO

Medida R^2

$$R^2 = 1 - \sum_{i=0}^n (y_i - \hat{y}_i)^2 / (y_i - \bar{y})^2$$

\bar{y} é media aritmética (do vetor de dados)

\hat{y} é vetor previsto

y é o vetor dados

$$e_i = (y_i - \hat{y}_i)$$

Quanto mais próximo de 1 melhor é o ajuste



Regressão Linear

`sklearn.linear_model.LinearRegression`

```
class sklearn.linear_model.LinearRegression(*, fit_intercept=True, normalize=False, copy_X=True, n_jobs=None, positive=False)
```

[\[source\]](#)

X_0	X_1	Y
1	1	6
1	2	8
2	2	9
2	3	11

$$Y = 1 * X_0 + 2 * X_1 + 3$$

```
>>> import numpy as np
>>> from sklearn.linear_model import LinearRegression
>>> X = np.array([[1, 1], [1, 2], [2, 2], [2, 3]])
>>> # y = 1 * x_0 + 2 * x_1 + 3
>>> y = np.dot(X, np.array([1, 2])) + 3
>>> reg = LinearRegression().fit(X, y)
>>> reg.score(X, y)
1.0
>>> reg.coef_
array([1., 2.])
>>> reg.intercept_
3.0...
>>> reg.predict(np.array([[3, 5]]))
array([16.])
```



REGRESSÃO LOGISTICA

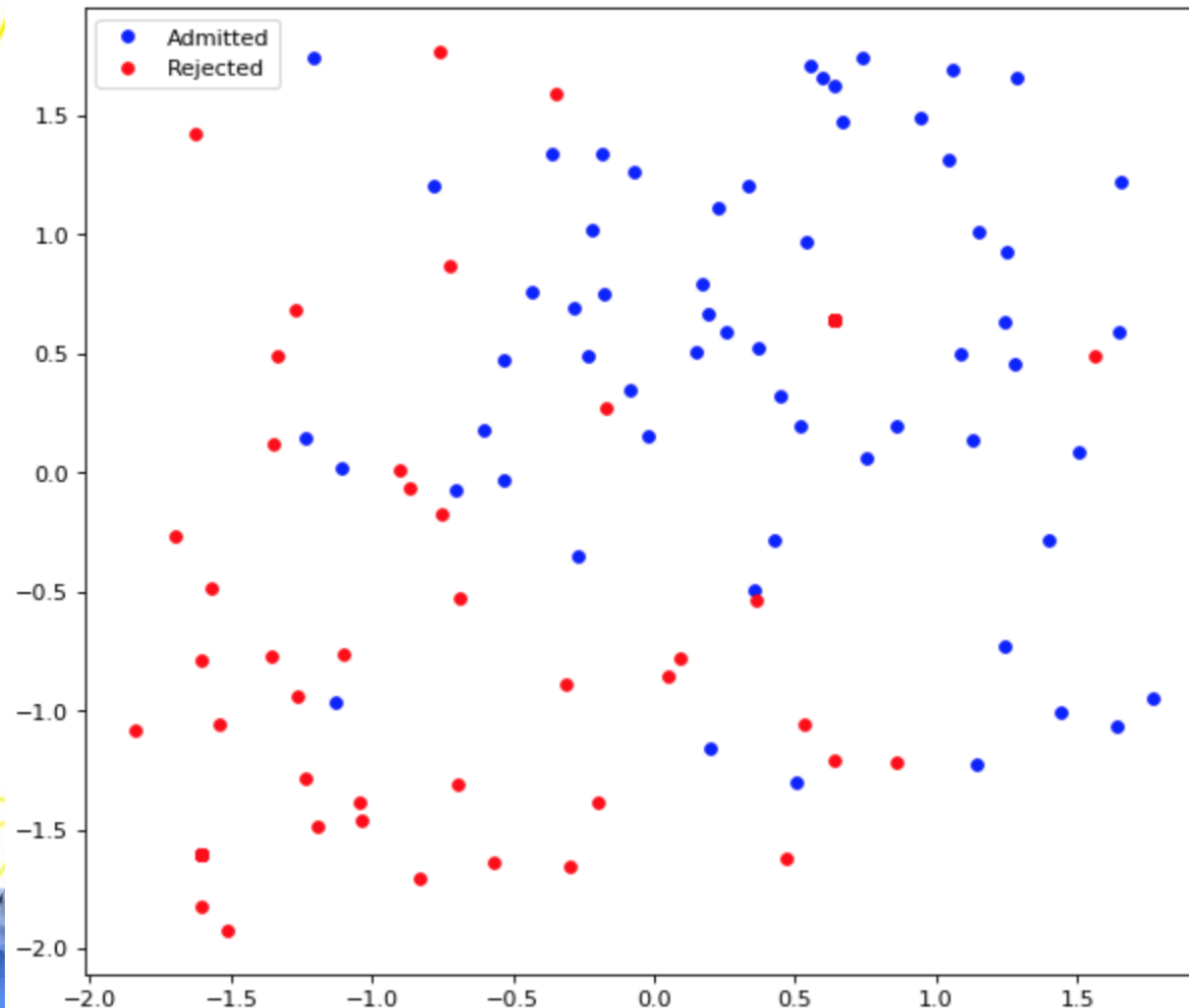
$$h(x) = w_0 + w_1.x_1 + w_2.x_2 ;$$

W ótimo na regressão linear pois temos que prever um valor contínuo

Na classificação temos que classificar ou categorizar em SIM=1 ou NÃO=0 (valores binários).



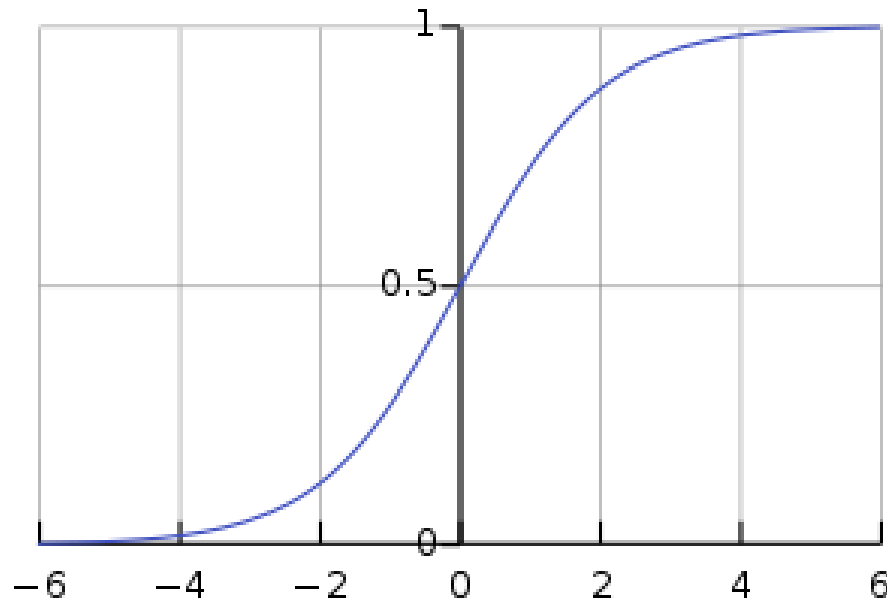
REGRESSÃO LOGÍSTICA



Estudantes foram ou não aprovados no VESTIBULAR

REGRESSÃO LOGISTICA

FUNÇÃO SIGMOID



$$g(x) = \frac{1}{1 + \exp(-x)} = \frac{1}{1 + e^{-x}}$$



REGRESSÃO LOGISTICA

$$g(h(x)) = 1/(1 + \exp(-h(x))) = \frac{1}{1 + e^{-h(x)}}$$

$$g(h(x)) = \frac{1}{1 + e^{-(w_0x_0 + w_1x_1 + w_2x_2)}}$$



Regressão Logística

```
from sklearn.datasets import load_iris  
from sklearn.linear_model import LogisticRegression  
X, y = load_iris(return_X_y=True)  
clf = LogisticRegression(random_state=0).fit(X, y)  
clf.predict(X[:2, :])
```

R.: array([0, 0])



Regressão Logística

```
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression
X, y = load_iris(return_X_y=True)
clf = LogisticRegression(random_state=0).fit(X, y)
clf.predict(X[50:54, :])
```

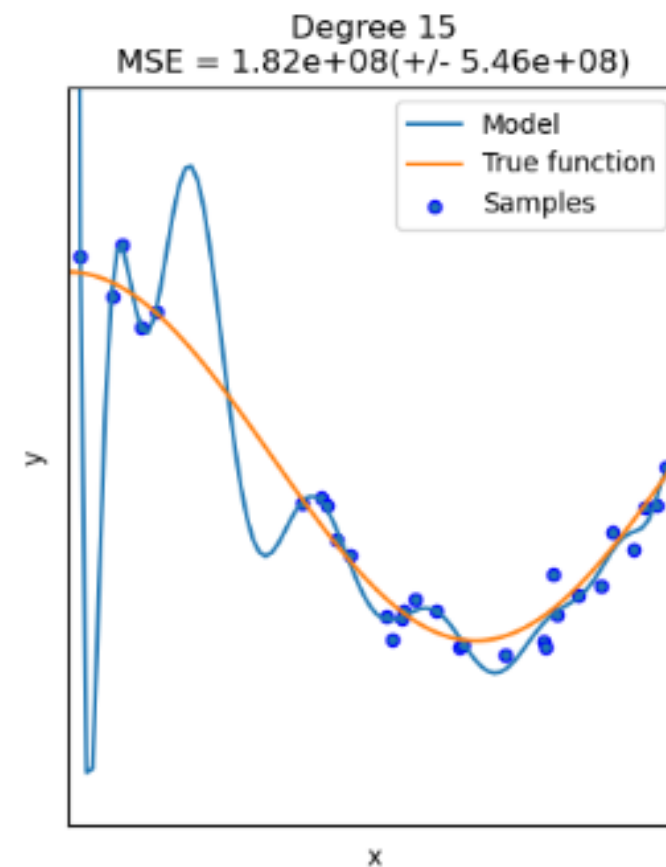
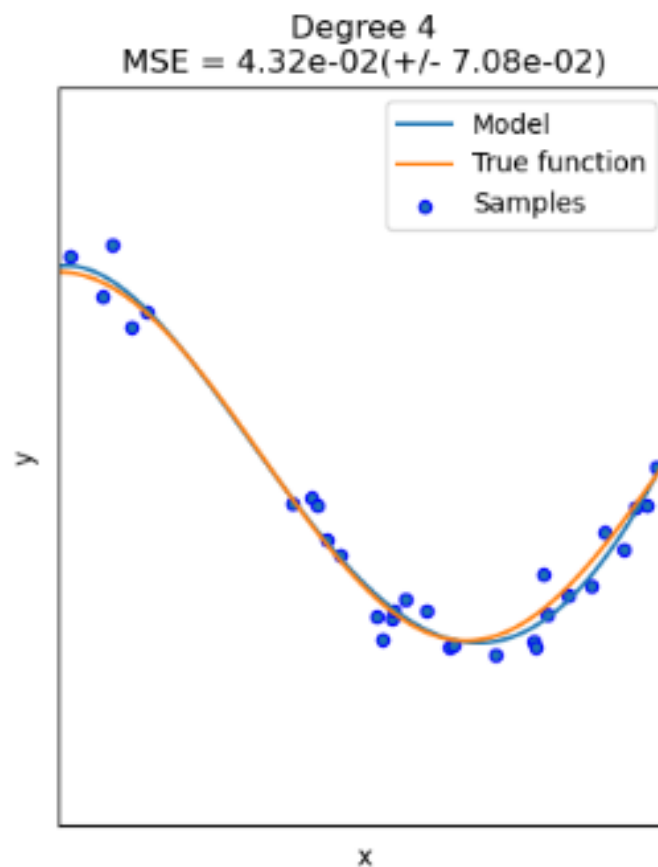
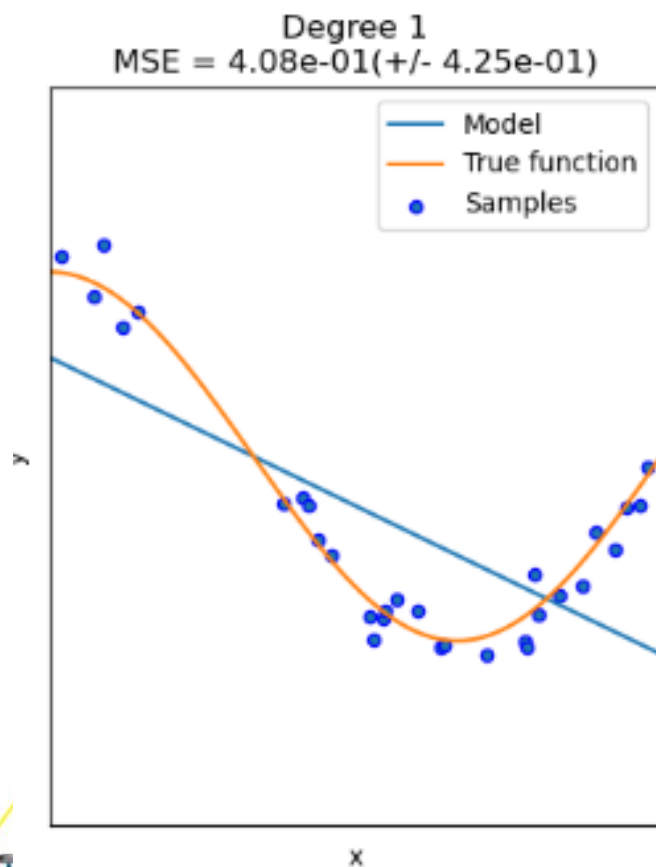
array([1, 1, 1, 1])

```
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression
X, y = load_iris(return_X_y=True)
clf = LogisticRegression(random_state=0).fit(X, y)
clf.predict(X[100:104, :])
```

array([2, 2, 2, 2])



Underfitting e Overfitting



REGRESSÃO POLINOMIAL

CASO UNIDIMENSIONAL

$$Y = AX^2 + BX + C \quad ; A = ?, B = ?, C = ?$$

$$Y = AX^3 + BX^2 + CX + D \quad ; A = ?, B = ?, C = ?, D = ?$$

CASO BIDIMENSIONAL

$$Y = AX^2 + BY^2 + CX + DY + E XY + F$$

$$Y = AX^3 + BY^3 + CX^2 + DY^2 + EX + FY + GXY + H$$

```
import numpy as np
```

```
from sklearn.preprocessing import  
PolynomialFeatures
```

```
X = np.arange(6).reshape(3, 2)
```

```
X
```

```
array([[0, 1],  
       [2, 3],  
       [4, 5]])
```

```
poly = PolynomialFeatures(2)
```

```
poly.fit_transform(X)
```

```
array([[ 1., 0., 1., 0., 0., 1.],  
       [ 1., 2., 3., 4., 6., 9.],  
       [ 1., 4., 5., 16., 20., 25.]])
```

```
poly = PolynomialFeatures(interaction_only=True)
```

```
poly.fit_transform(X)
```

```
array([[ 1., 0., 1., 0.],  
       [ 1., 2., 3., 6.],  
       [ 1., 4., 5., 20.]])
```



CASOS DE COVID

DADOS DE MARÇO A ABRIL DE 2021

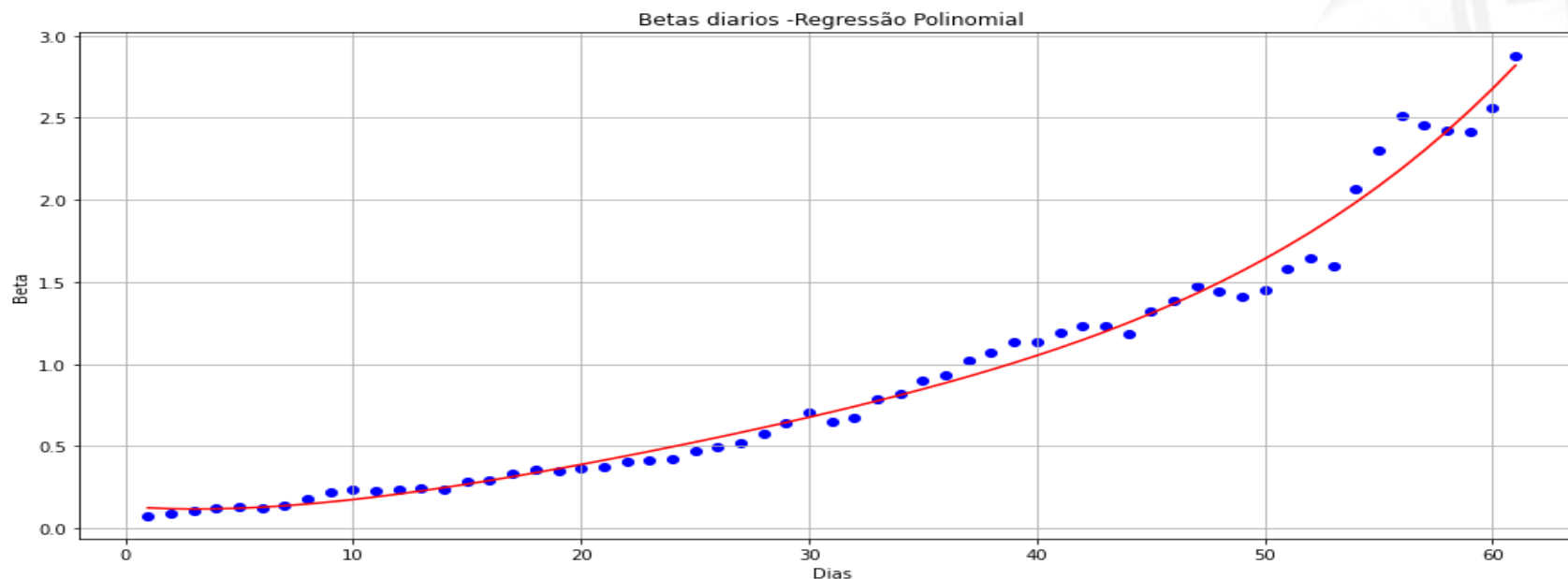
A primeira técnica investigada foi a regressão polinomial. Escolheu-se os dados da cidade de São Paulo, como estudo de caso.

Usando a técnica de regressão polinomial, foram testadas regressões de vários graus diferentes 1, 2, 3, 4 e 5.

A **regressão polinomial de grau 4** foi a que melhor se ajustou aos dados de São Paulo e pode ser visualizada na Figura 1, considerando dados coletados para os meses de março a abril.



PREVISÃO PARA DADOS DE COVID



Segundo esta regressão, a previsão para o dia 09 de junho, por exemplo, fornece um valor previsto de beta igual a 4.5. O erro quadrático médio obtido foi de aproximadamente 0.00869.



EXEMPLO 5: REGRESSÃO MULTIVARIADA

