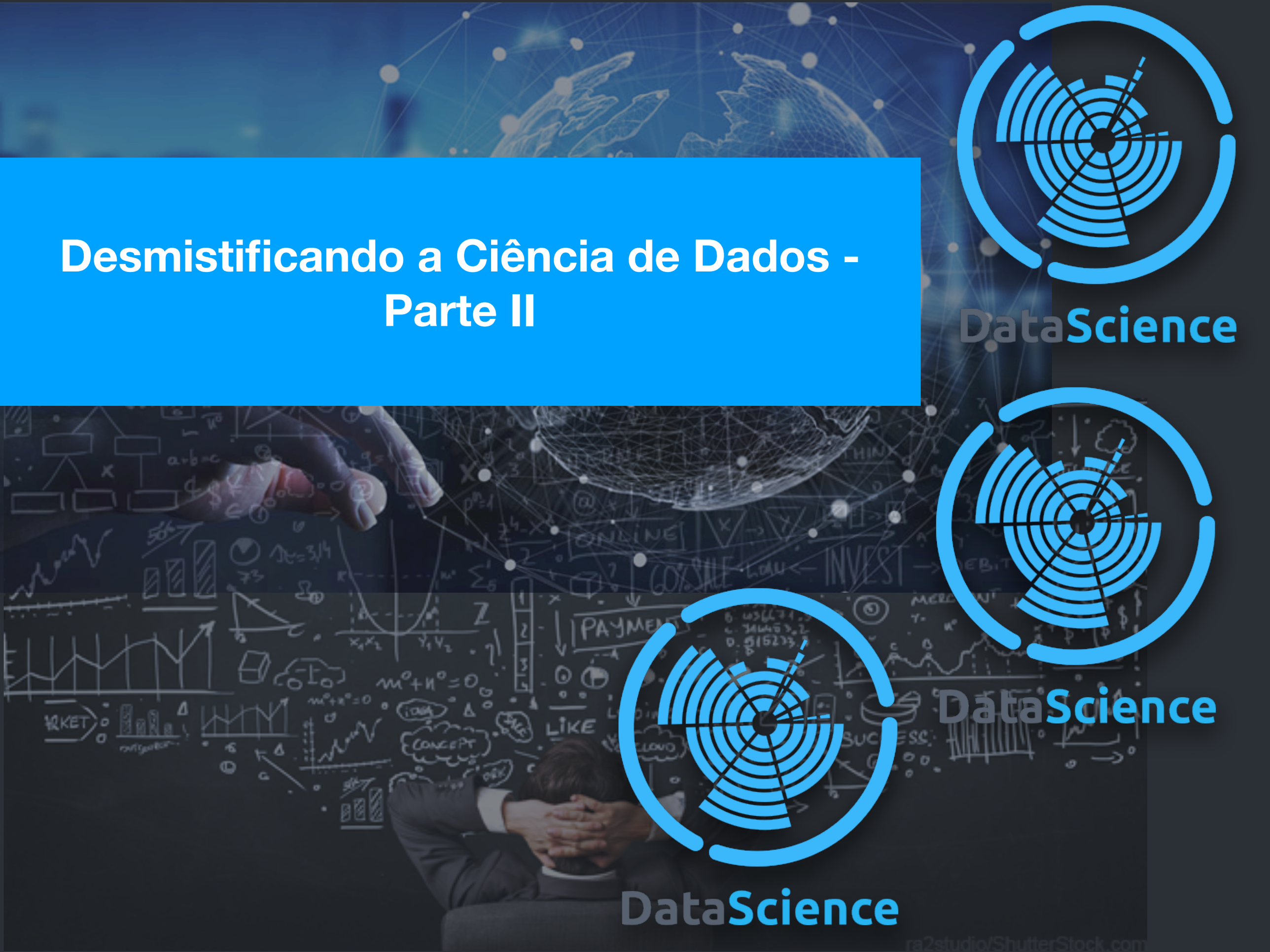
Desmistificando a Ciência de Dados - Parte II



Apresentação

Francisco Nauber Bernardo Gois

Engenheiro de aprendizado de máquina no Serviço Federal de Processamento de Dados

Doutor em Informática Aplicada

Mestre em Informática Aplicada

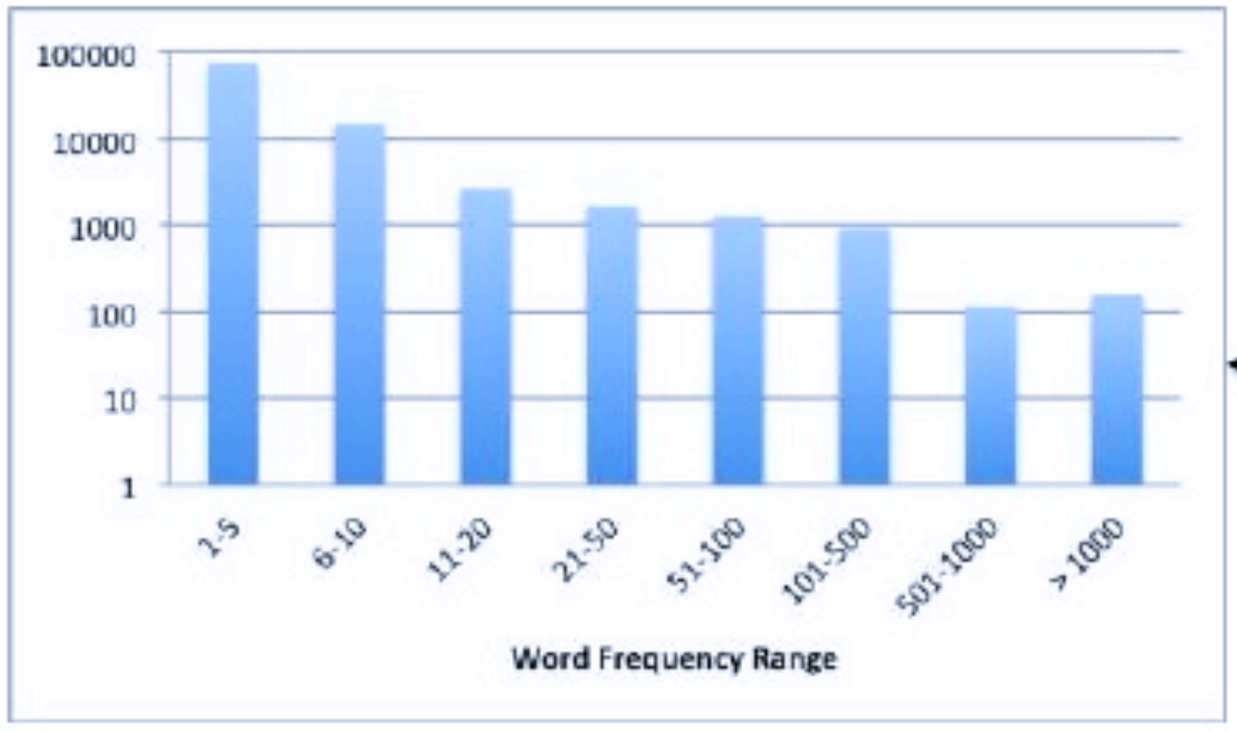
Especialista em desenvolvimento WEB

Dúvidas: naubergois@gmail.com

Esparsidade dos Dados

Esparsidade dos dados

What Does sparsity mean?



Training data contains many infrequent terms

Densidade vs Eparsidade

- Dense Vectors
  - Usually have a nonzero value for each variable
  - The “telecom churn” dataset we use in the labs is a dense dataset.
  - Use Vectors.dense

- Sparse Vectors
  - Most values are zero (or nonexistent)
  - Text Data yields sparse vectors
  - One-Hot, factor variables lead to sparse vectors
  - Use Vectors.sparse

Aprendizado Indutivo

Aprendizado Indutivo

Aprendizado indutivo *(inductive learning)*

6

- Em AM, o processo de determinar uma hipótese a partir de um conjunto de dados de treinamento é denominado aprendizado indutivo.
  - Indução se refere a aprender conceitos gerais a partir de exemplos.
- É o processo inverso do aprendizado dedutivo.

Aprendizado Indutivo

Generalização

- Em AM, o objetivo é induzir modelos que tenham um bom desempenho em dados do domínio do problema, não apenas no conjunto de treinamento.
- Diz-se que um modelo tem uma boa generalização se ele é capaz de realizar predições adequadas sobre dados não utilizados durante o treinamento.

Aprendizado Indutivo

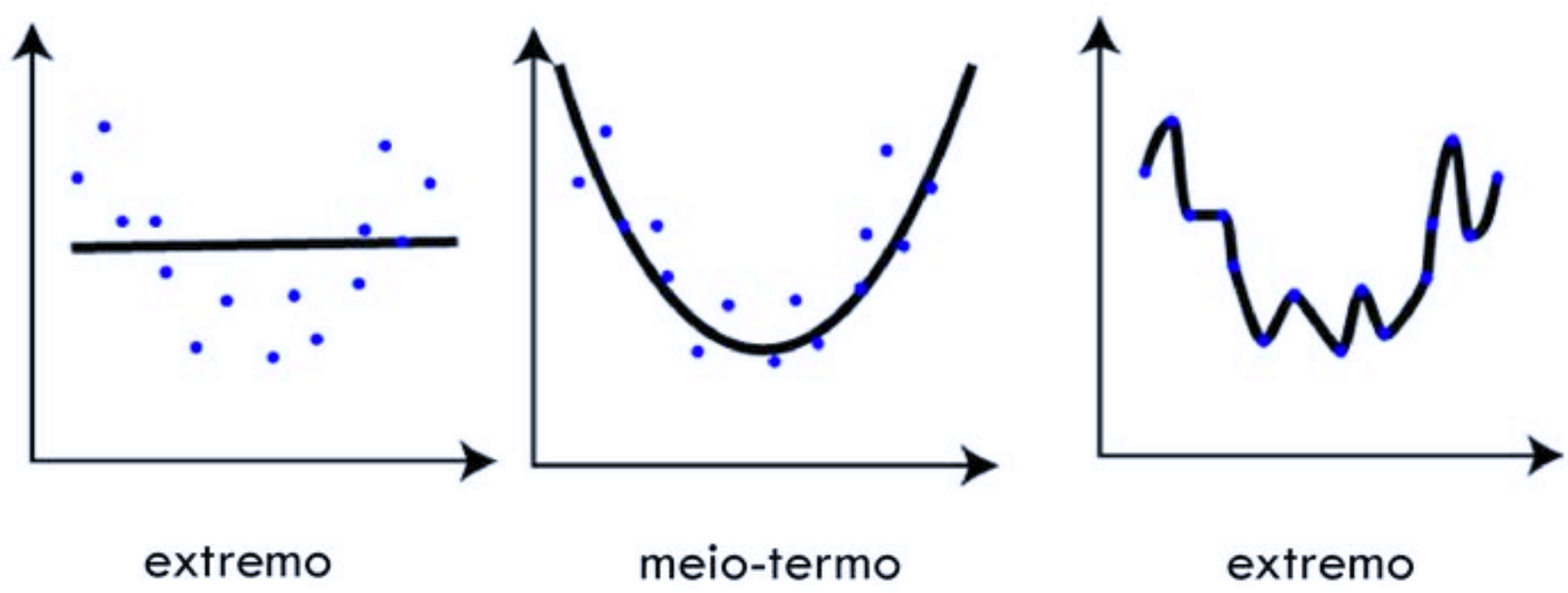
Qualidade do Ajuste *(goodness of fit)*

- Em AM, diz-se que uma função hipótese se ajusta aos dados de treinamento.
- A hipótese procura aproximar a função alvo (não conhecida) que mapeia variáveis de entrada nas de saída.
- Um aspecto importante a medir é a *qualidade* desse ajuste.

Aprendizado Indutivo

Qualidade do Ajuste *(goodness of fit)*

- Um algoritmo de AM deve exercitar a temperança para alcançar uma boa qualidade de ajuste.



Aprendizado Indutivo

Complexidade de um Modelo *(model complexity)*

- Em AM, a complexidade de um modelo de aprendizado diz respeito à quantidade de parâmetros livres que ele utilizada.
- Entretanto, podem ser definidas restrições sobre os parâmetros, o que diminui a complexidade do modelo (mais sobre isso adiante).

Aprendizado Indutivo

Complexidade de um Modelo *(model complexity)*

- Um modelo muito complexo quando é formado por tantos parâmetros que o modelo "memoriza" todos os dados de treinamento,
  - não apenas os sinais, mas também o ruído aleatório, os erros e todas as características específicas da amostra.
- Um modelo muito simples é formado por uma quantidade insuficiente de parâmetros que não permitem que ele apreenda a tendência subjacente nos dados.

Aprendizado Indutivo

Superajuste e subajuste

- A complexidade inadequada pode levar a um de dois fenômenos, ambos indesejados:
  - Subajuste (*underfitting*)
  - Superajuste (*overfitting*)

Aprendizado Indutivo

Superajuste (*overfitting*)

- O superajuste é causado por uma hipótese que se adapta aos dados disponíveis, mas não generaliza bem para (prever em) novos dados.
  - Geralmente é causado por uma função complicada que cria muitas curvas desnecessárias e ângulos não relacionados à população da qual os dados de treinamento vieram.

Aprendizado Indutivo

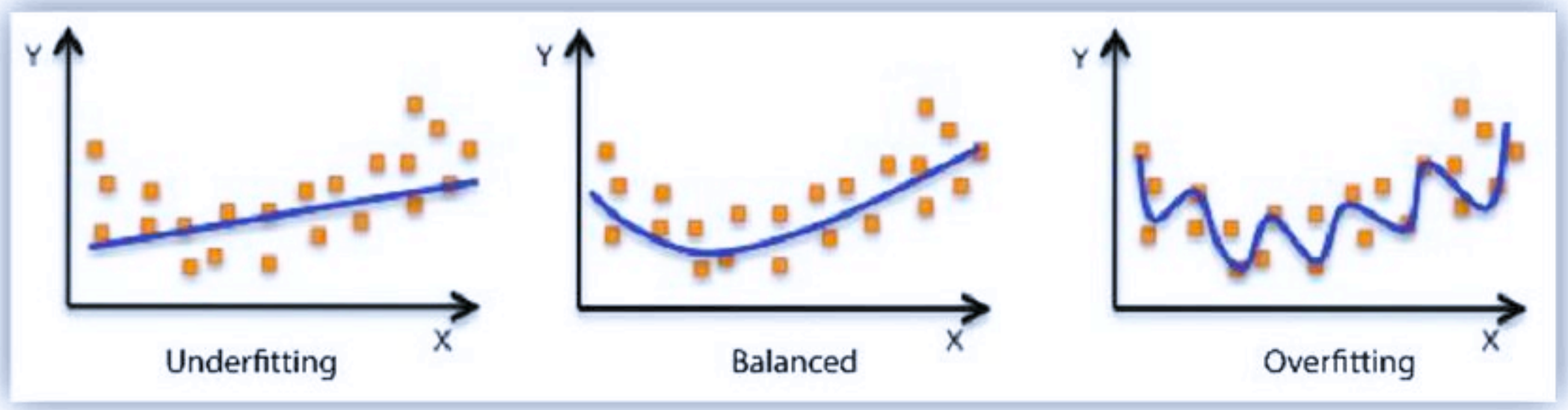
Superajuste vs subajuste

- A figura a seguir apresenta diferentes modelos, cada um com seu erro empírico correspondente.

Aprendizado Indutivo

Superajuste vs subajuste - exemplo

17



Fonte: http://docs.aws.amazon.com/machine-learning/latest/dg/model-fit-underfitting-vs-overfitting.html

Aprendizado Indutivo

Soluções para o superajuste

1) Reduzir o número de características
   - Selecionar manualmente quais características manter.
   - Usar um algoritmo de seleção de modelo (*model selection*); mais adiante no curso.
2) Regularização
   - Manter todas as características, mas reduzir a magnitude dos parâmetros $\theta_1,\ldots,\theta_n$.
   - A regularização funciona bem quando temos muitas características úteis.

Regularização

Regularização

Regularização

- A regularização penaliza explicações complexas ou extremas dos dados, mesmo que se encaixem no que melhor se observou (nos dados).
  - A ideia é que tais explicações não são susceptíveis de generalizar bem para o futuro;
  - Elas podem explicar alguns pontos de dados do passado, mas isso pode ser apenas por causa de particularidades da amostra.

Regularização

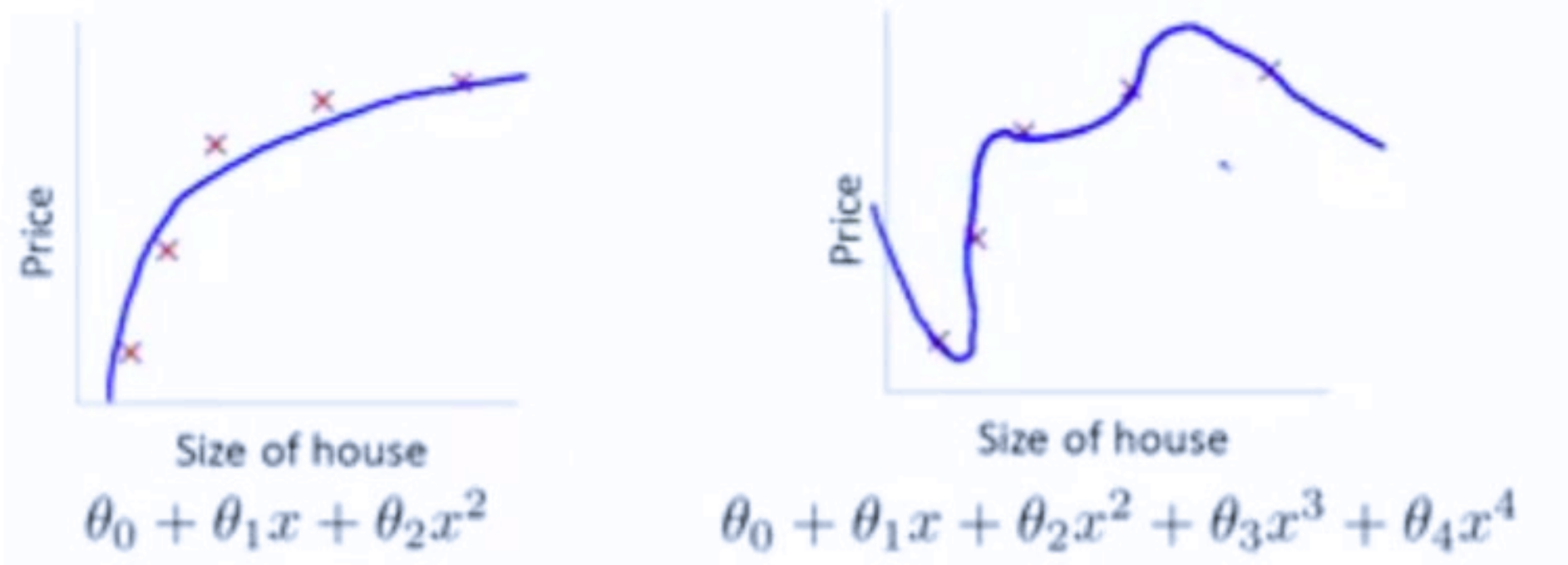
Regularização vs superajuste

- A regularização penaliza a complexidade, mesmo que isso signifique escolher uma hipótese menos precisa de acordo com os dados de treinamento.
- Útil para combater o superajuste.

Regularização

Regularização - intuição

22



- Suponha que desejemos fazer com que a hipótese à direita se aproxime mais de uma forma quadrática...

Regularização

Regularização - intuição (cont.)

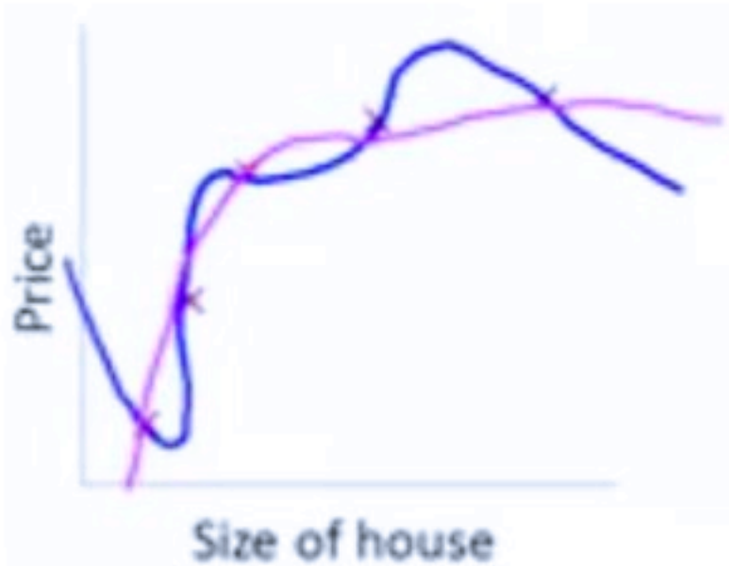
- Para isso, devemos reduzir a influência dos termos a seguir: $\theta_3 x^3$ e $\theta_4 x^4$.
- Isso pode ser feito por meio de uma modificação na função de custo:

$$min_\theta \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + 1000 \cdot \theta_3^2 + 1000 \cdot \theta_4^2$$

Regularização

Regularização - intuição (cont.)

- O efeito é uma suavização da hipótese original.

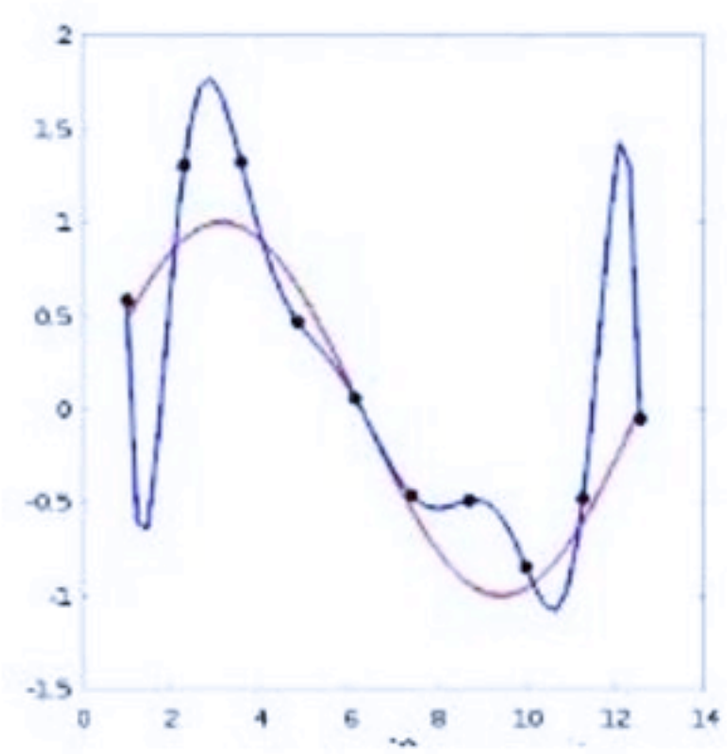


$$\theta_3 \approx \theta_4 \approx 0$$

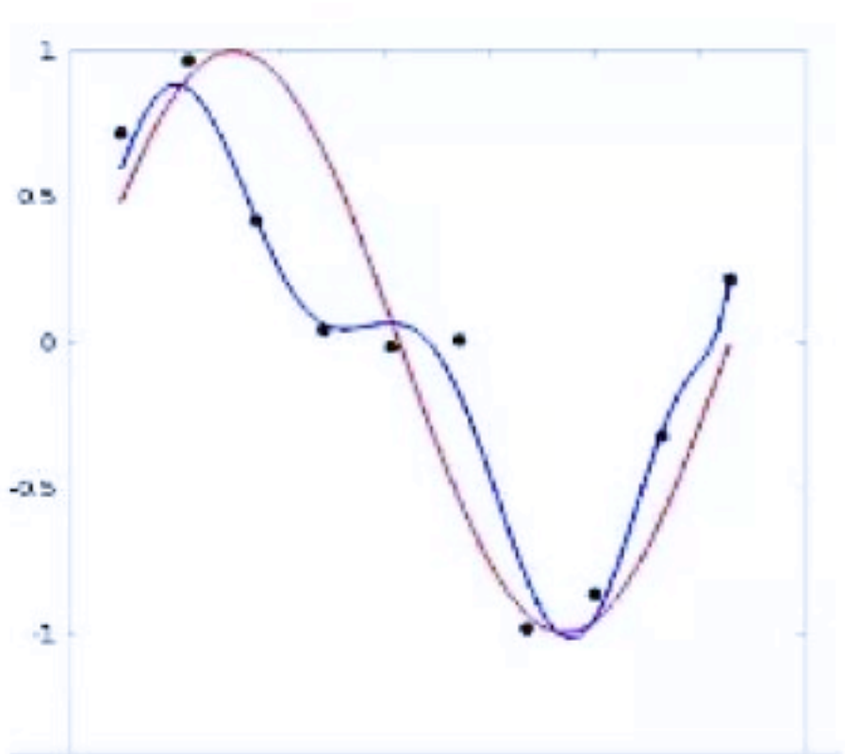
$$min_\theta \ \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + 1000 \cdot \theta_3^2 + 1000 \cdot \theta_4^2$$

Regularização

Regularização - exemplo



Sem regularização



Com regularização

Fonte: https://www.quora.com/profile/Quoc-Le-3

Curva em vermelho – função alvo
Curvas em azul - hipóteses

Regularização

Regularização – terminologia

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 + \lambda\sum_{j=1}^{n}\theta_j^2\right]$$

λ : parâmetro de regularização (*regularization parameter*)

$\lambda\sum_{j=1}^{n}\theta_j^2$: termo de regularização (*regularization term*)

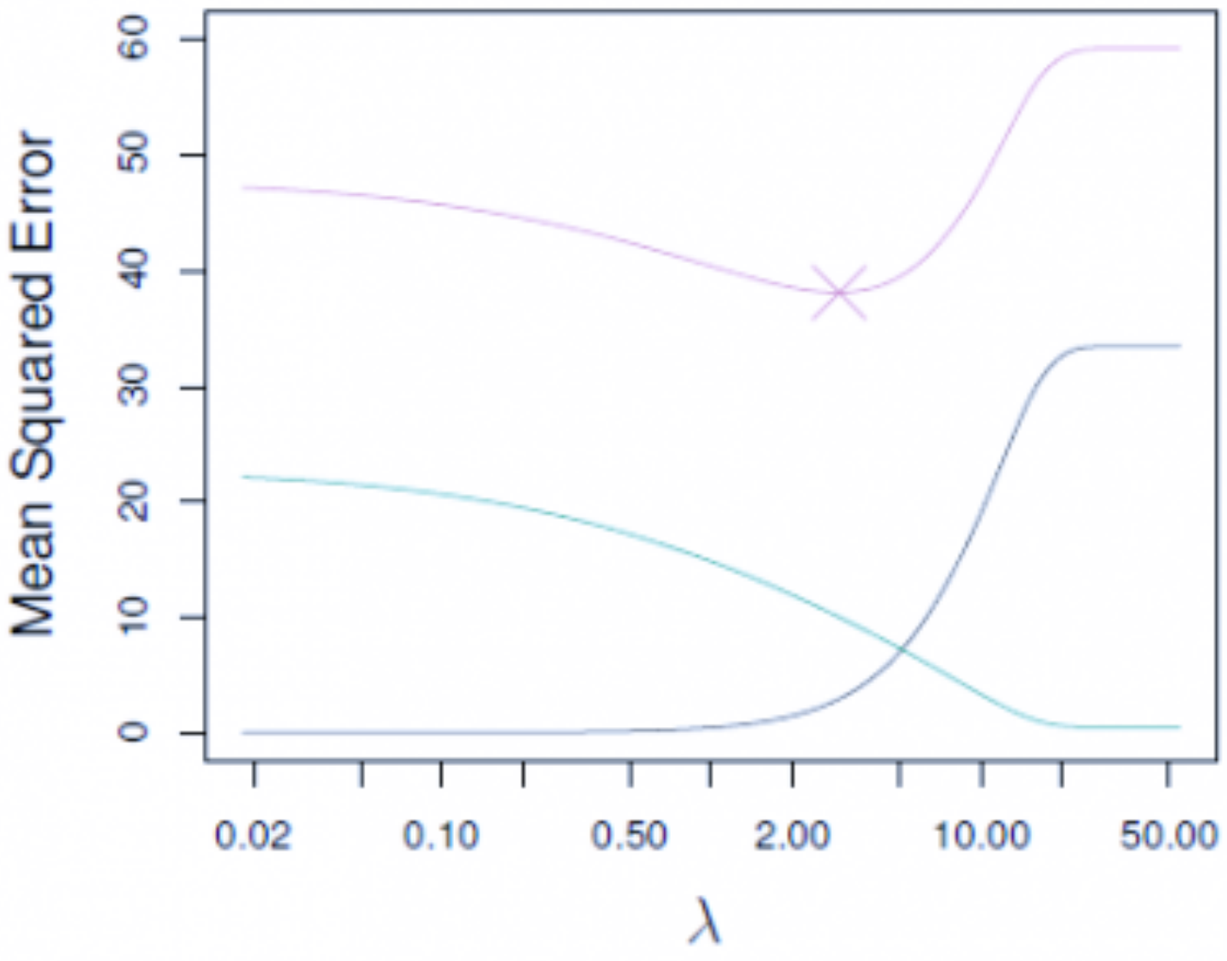
.

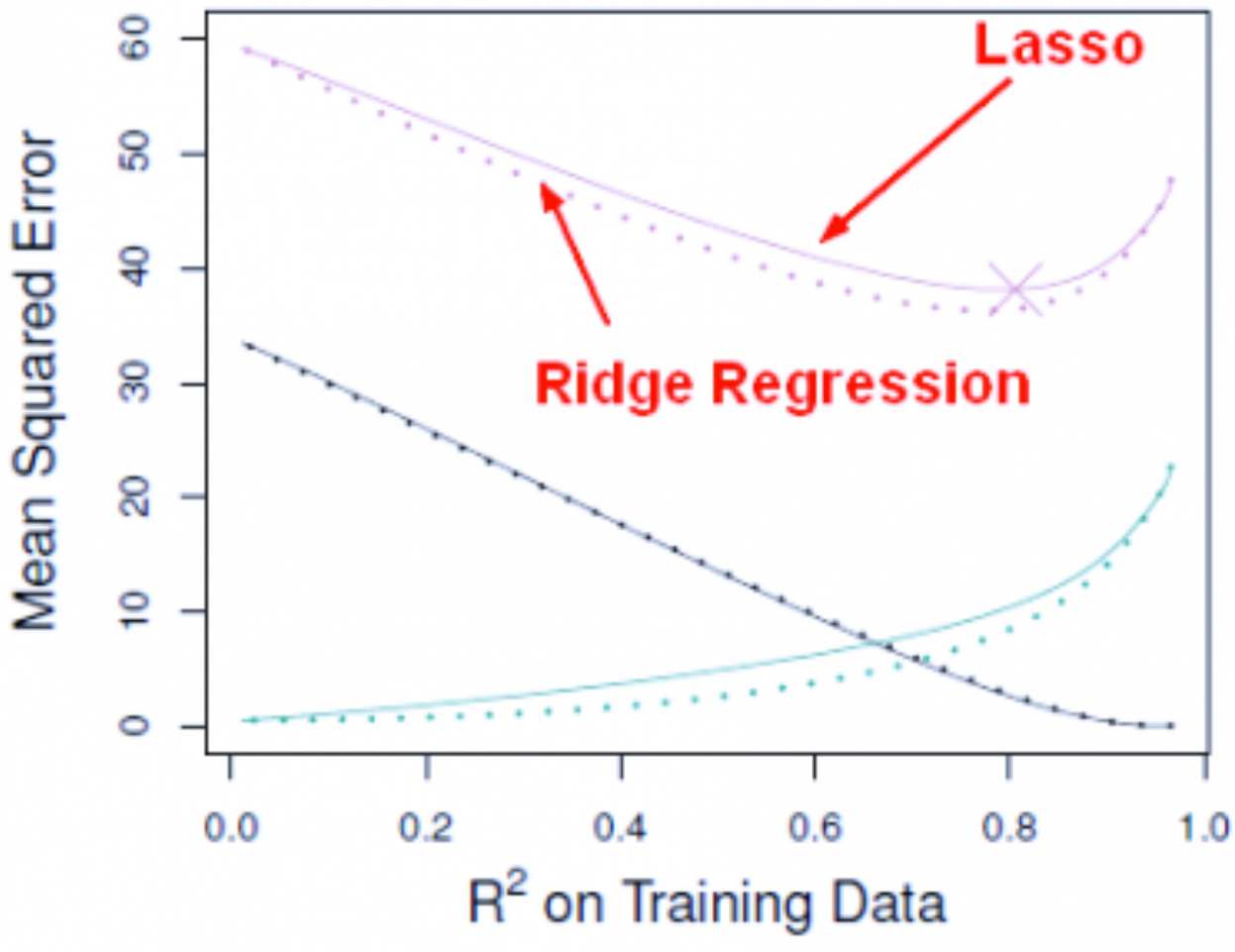
Regularização

Valor do termo de regularização

- Para que a técnica de regularização tenha sucesso, é preciso ter cuidado com a escolha do valor de λ
- O que pode acontecer se λ for:
  - Muito grande (e.g., $\lambda = 10^{10}$)?
  - Muito pequeno (e.g., $\lambda = 0$)?

Regularização





Regularização

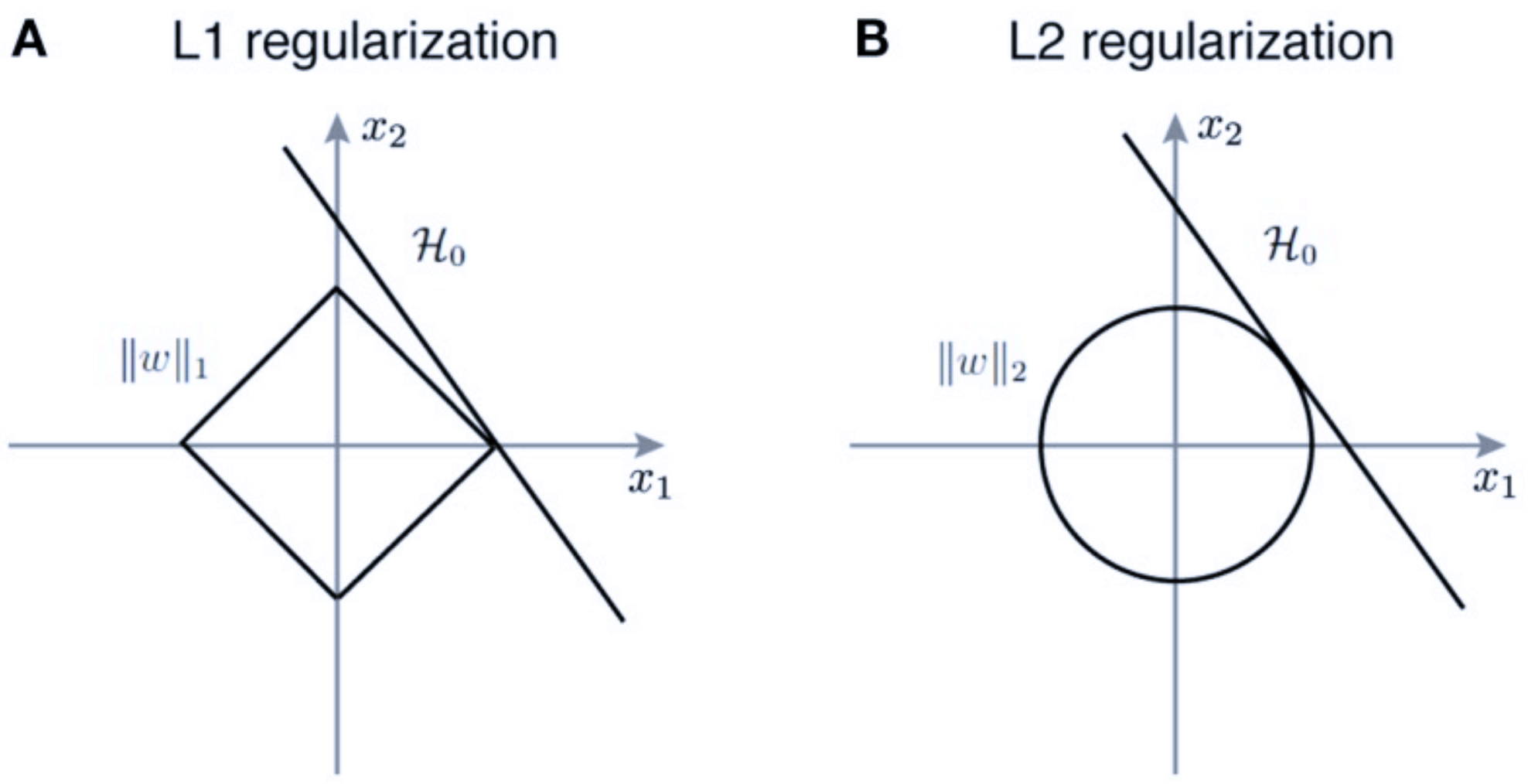
L1 regularization on least squares:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \sum_j \left(t(\mathbf{x}_j) - \sum_i w_i h_i(\mathbf{x}_j) \right)^2 + \lambda \sum_{i=1}^{k} |w_i|$$

L2 regularization on least squares:

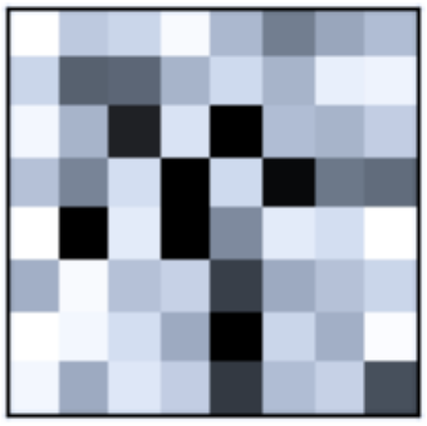
$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \sum_j \left(t(\mathbf{x}_j) - \sum_i w_i h_i(\mathbf{x}_j) \right)^2 + \lambda \sum_{i=1}^{k} w_i^2$$

L2 regularization	L1 regularization
Computational efficient due to having analytical solutions	Computational inefficient on non-sparse cases
Non-sparse outputs	Sparse outputs
No feature selection	Built-in feature selection

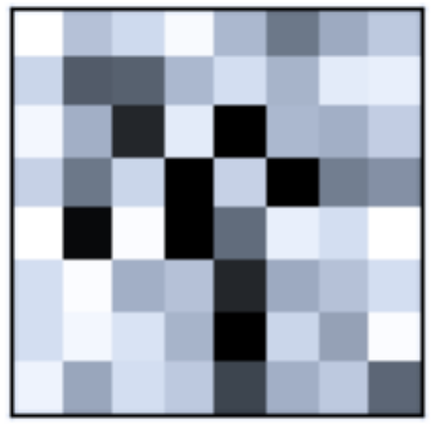


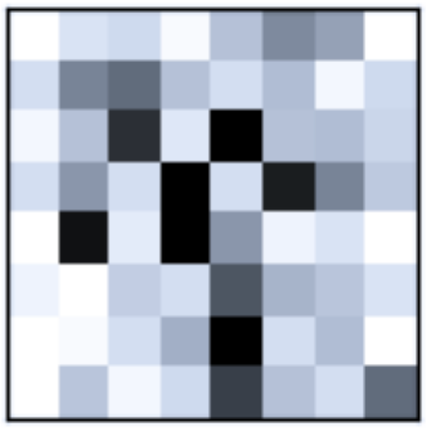
L1 penalty

L2 penalty

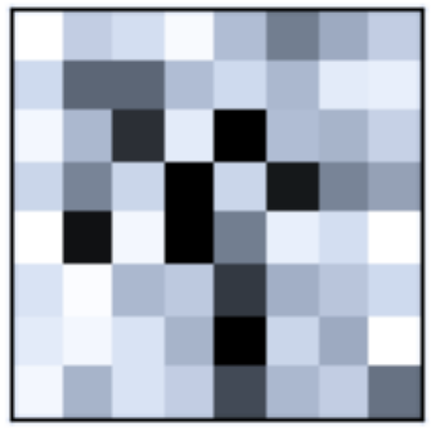


C = 100.00





C = 1.00



C = 0.01

Regularização


```
>>> from sklearn import linear_model
>>> clf = linear_model.Lasso(alpha=0.1)
>>> clf.fit([[0,0], [1, 1], [2, 2]], [0, 1, 2])
Lasso(alpha=0.1, copy_X=True, fit_intercept=True, max_iter=1000,
   normalize=False, positive=False, precompute=False, random_state=None,
   selection='cyclic', tol=0.0001, warm_start=False)
>>> print(clf.coef_)
[ 0.85  0.  ]
>>> print(clf.intercept_)
0.15
```

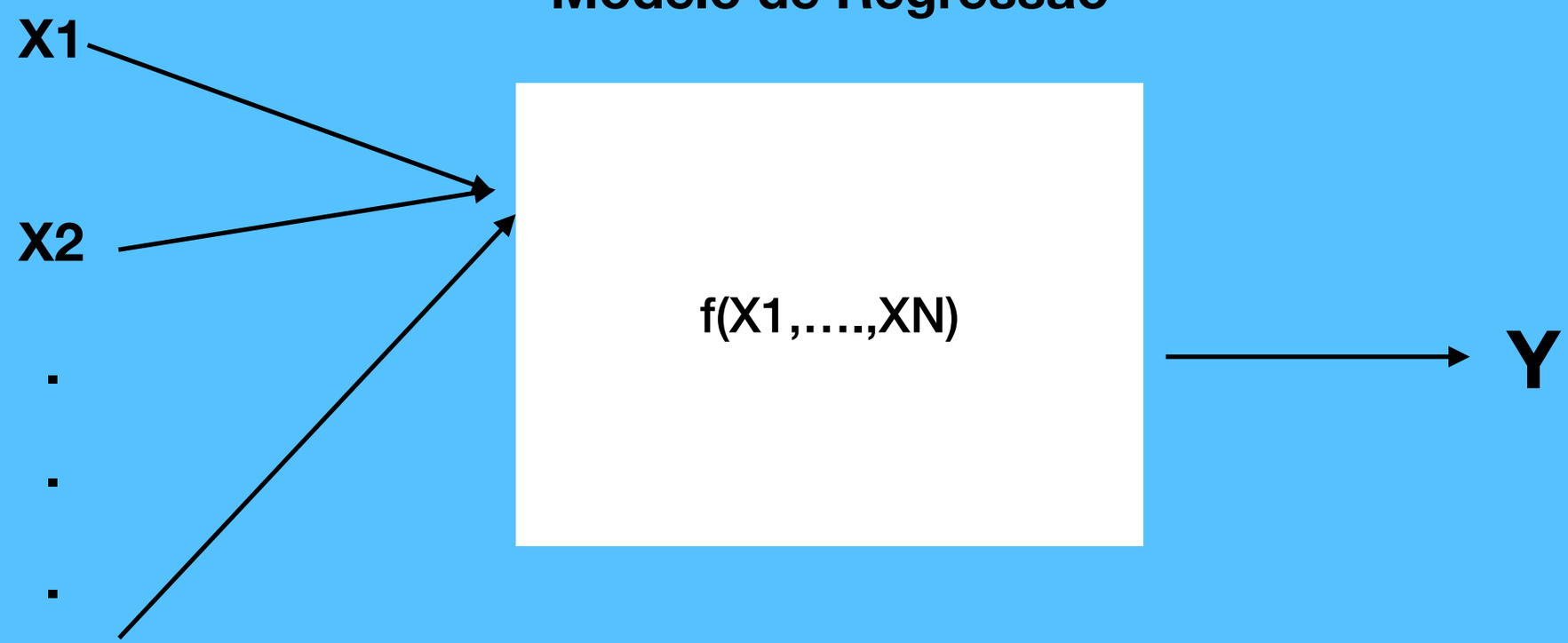
Regularização


```
>>> from sklearn.linear_model import Ridge
>>> import numpy as np
>>> n_samples, n_features = 10, 5
>>> np.random.seed(0)
>>> y = np.random.randn(n_samples)
>>> X = np.random.randn(n_samples, n_features)
>>> clf = Ridge(alpha=1.0)
>>> clf.fit(X, y)
Ridge(alpha=1.0, copy_X=True, fit_intercept=True, max_iter=None,
      normalize=False, random_state=None, solver='auto', tol=0.001)
```

Avaliação da Regressão

Análise de regressão é uma metodologia estatística que utiliza a relação entre duas ou mais variáveis quantitativas de tal forma que uma variável possa ser predita a partir de outra.

Modelo de Regressão



Y= a + Bx (Simples)

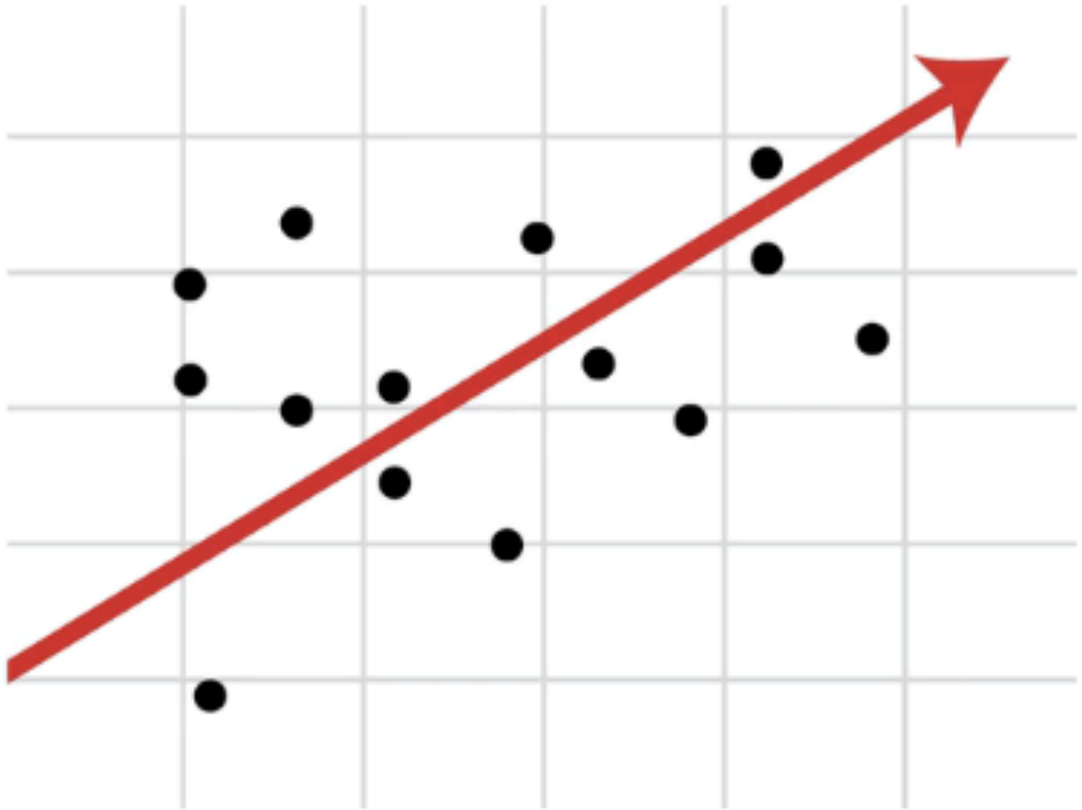
Y= a + B1x1+B2x2+B3X3 …. (Multipla)

Analise de Regressão

A análise de regressão compreende quatro tipos básicos de modelos

- Linear Simples
- Linear Múltiplo
- Não Linear Simples
- Não Linear Múltiplo

Analise de Regressão



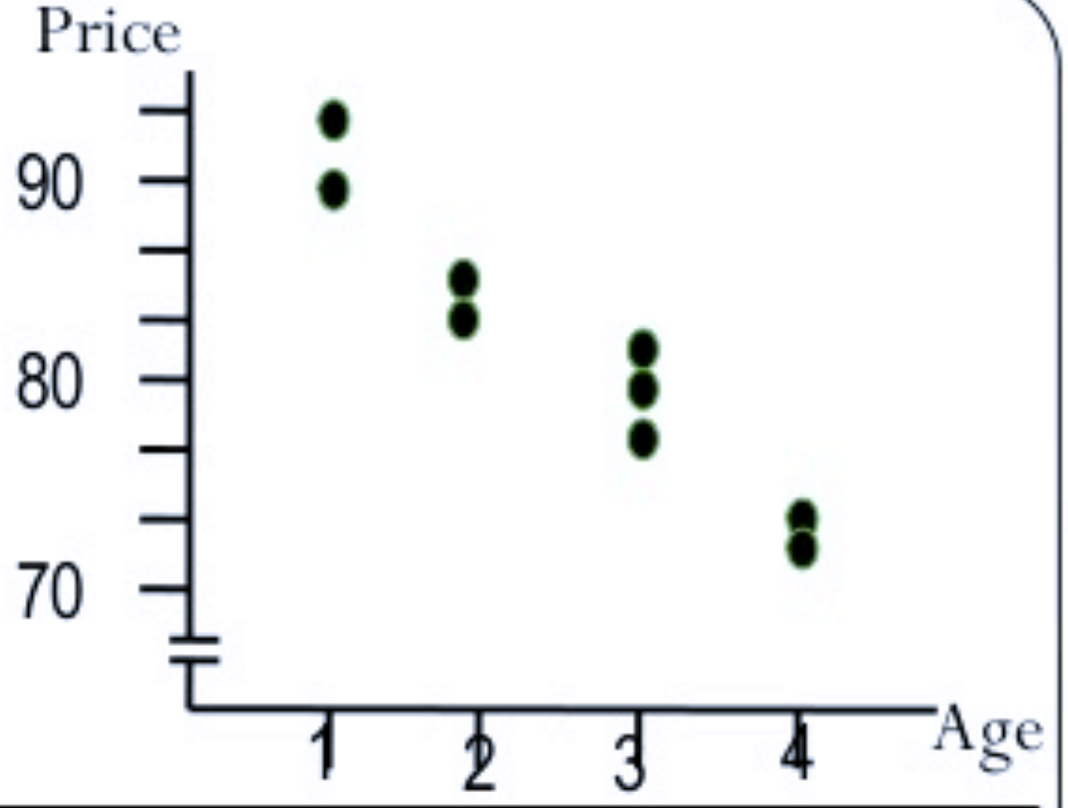
- Simple Linear Regression
- Multiple Linear Regression
- Ridge Regression
- **Lasso Regression**
- **Logistic Regression**
- Polynomial Regression
- Stepwise Regression
- Elastic Net Regression

How to Show it Statistically

$$Y(E) = b_0 + b_1 X$$

$$Y(E) = 97 - 5X$$

$$Y = 97 - 5X + E$$

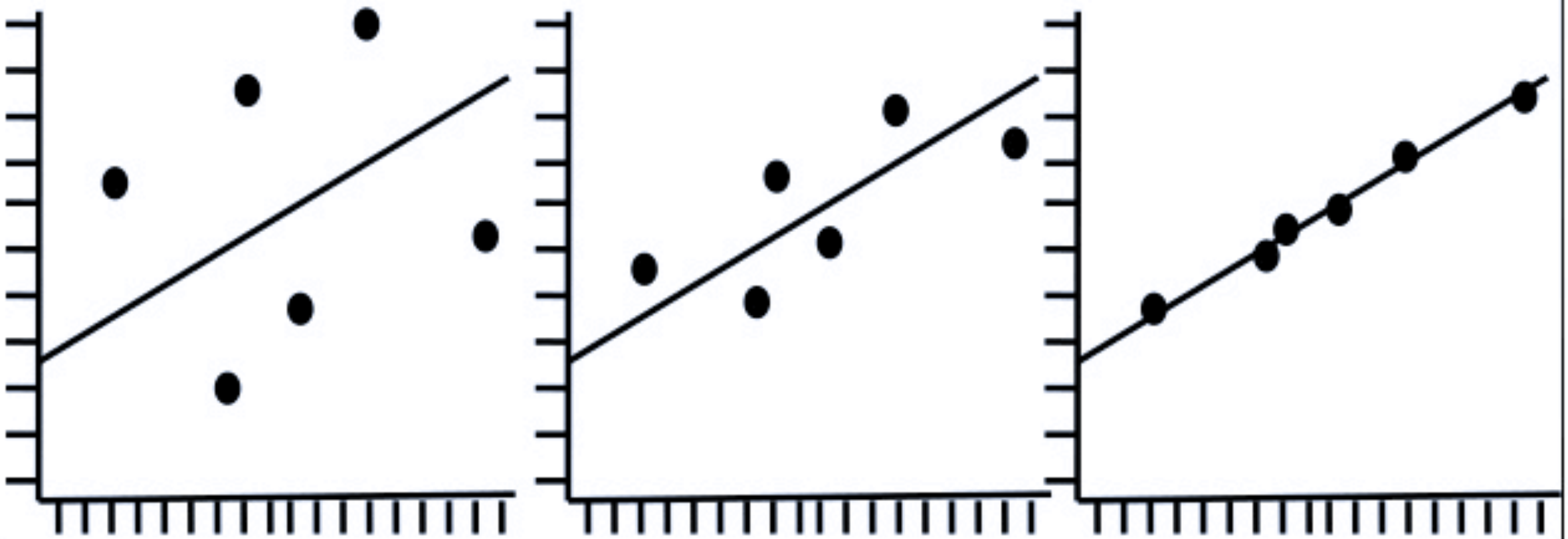


Term	What it is!
Y (E)	Dependent Variable whose behavior is to be determined
X	Independent Variable whose effect to be determined
b_0	Intercept: Value of Y(E) when X = 0
b_1	Estimated Change in Y in response to unit Change in X
E	Difference between the actual and estimated

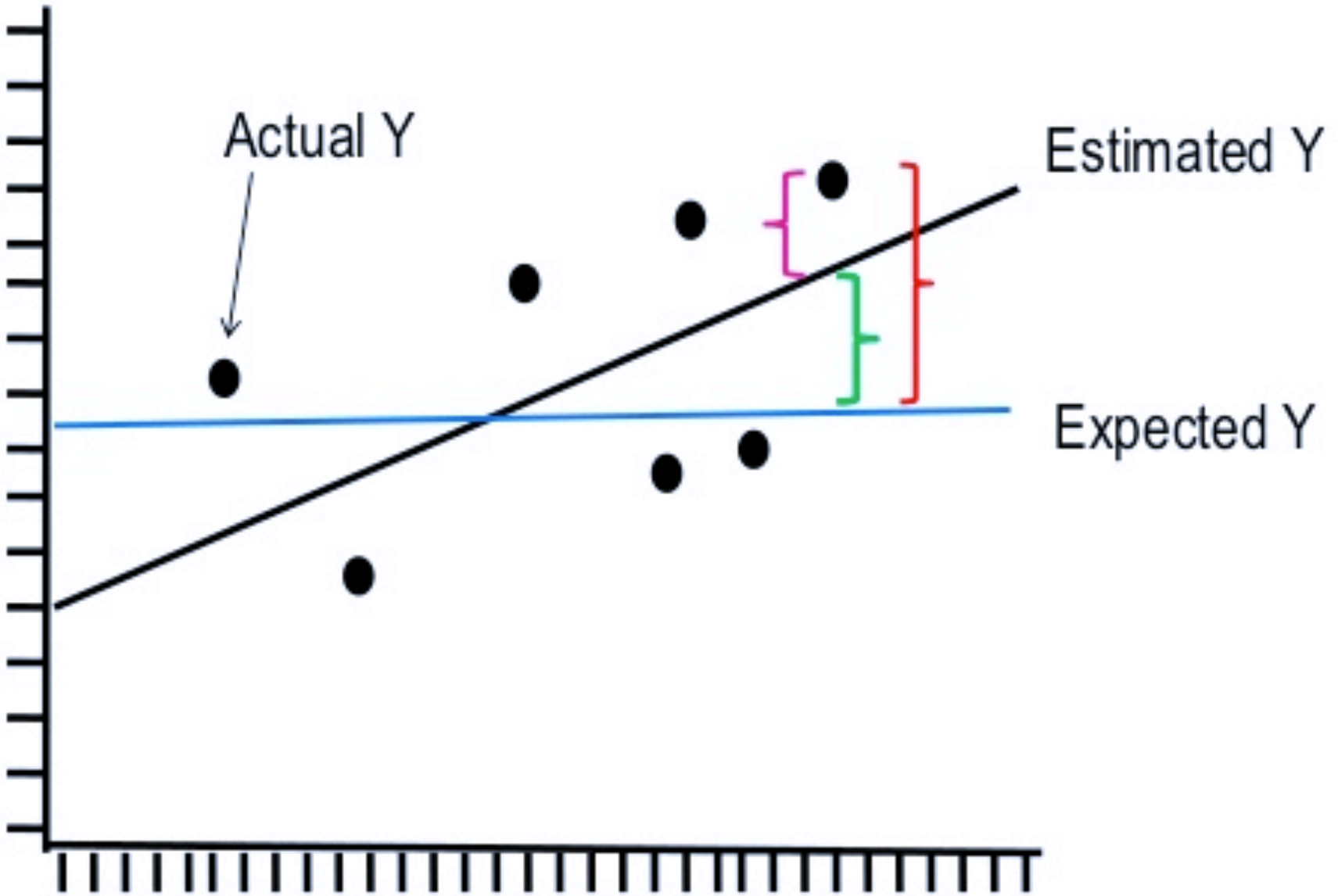
Assessing the Goodness of Fit: Graphical Way

Goodness of Fit Means	How well the model fits the actual data. Less residual means a good bit, more residual means bad Fit

Bad Fit	Good Fit	Perfect Fit



Assessing the Goodness of Fit: Statistical Way



Analise de Regressão

Mean Absolute Error (MAE)

$$\mathrm{MAE} = \frac{1}{n}\sum_{j=1}^{n} |y_j - \hat{y}_j|$$

Analise de Regressão

Root mean squared error (RMSE)

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

Analise de Regressão

RMSE penaliza maiores erros , no entanto o indicador possui maior dificuldade de interpretação.

Analise de Regressão

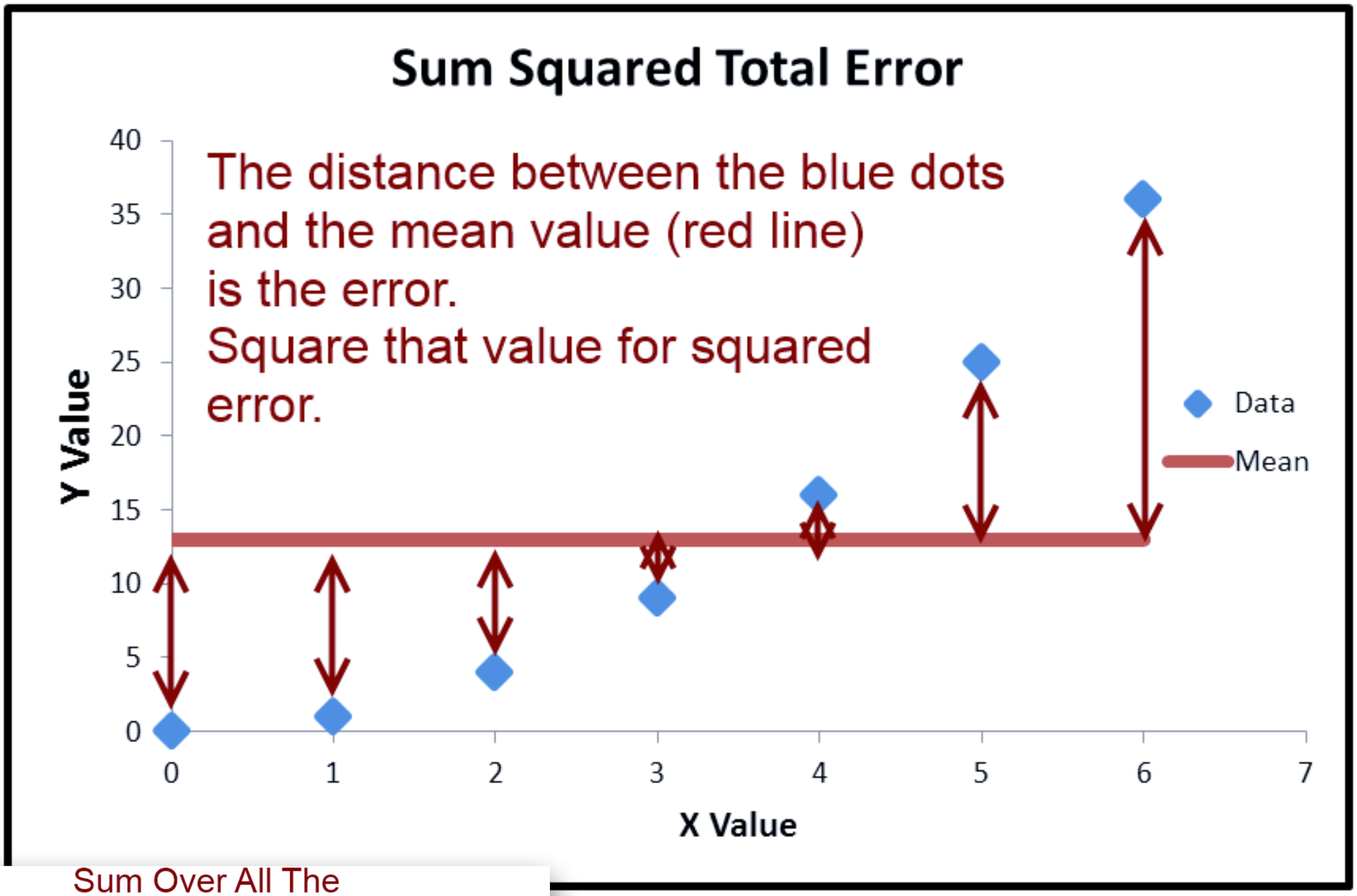
Coefficient of Determination → $$R^2 = \frac{SSR}{SST} = 1 - \frac{SSE}{SST}$$

Sum of Squares Total → $$SST = \sum (y - \bar{y})^2$$

Sum of Squares Regression → $$SSR = \sum (y' - \bar{y'})^2$$

Sum of Squares Error → $$SSE = \sum (y - y')^2$$

SST



$$SS_{Total} = \sum (y_i - \bar{y})^2$$

Sum Squared Total Error

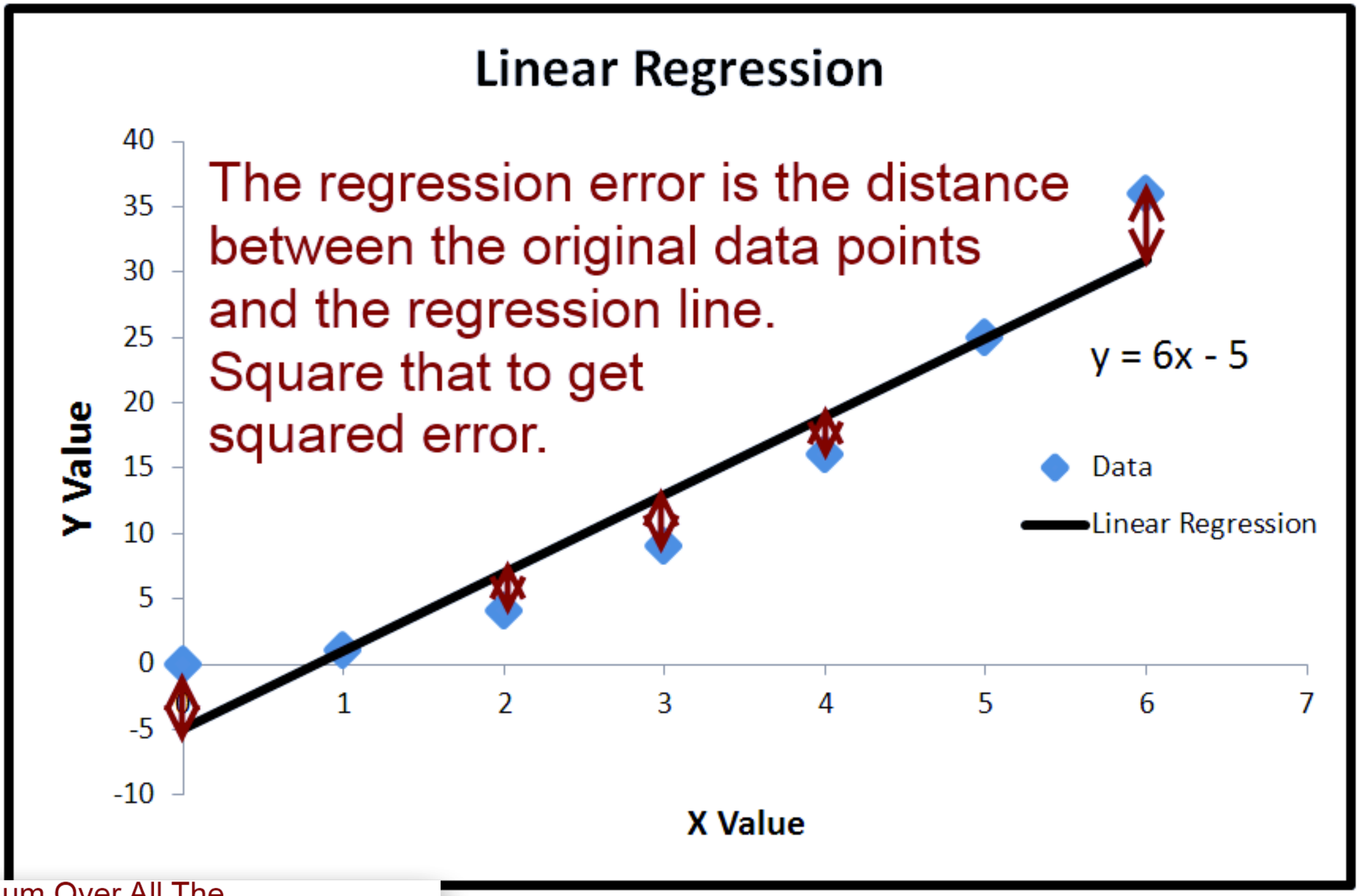
Sum Over All The Data Points

Each Data Point

Square The Result

Mean Value

SSR



Sum Over All The Data Points

Square The Result

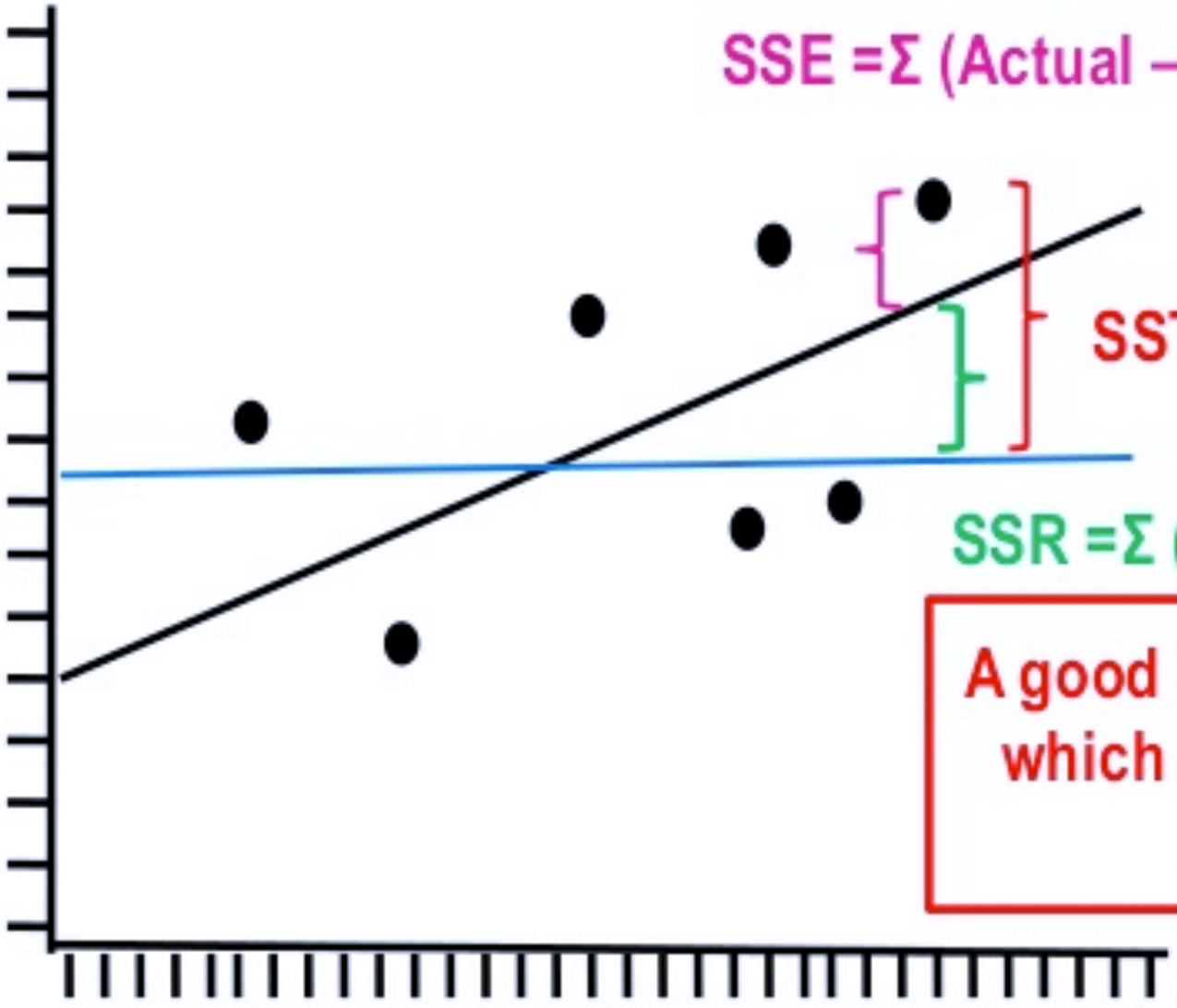
$$SS_{Regression} = \sum (y_i - y_{Regression})^2$$

Sum Squared Regression Error

Each Data Point

Regression Value

Assessing the Goodness of Fit: Statistical Way R^2



$SSE = \Sigma (Actual - Expected)^2$

$SST = \Sigma (Real - Expected)^2$

$SSR = \Sigma (Estimated - Expected)^2$

A good Model is the one in which SSE is the lowest

SSE = 0

$SST = SSR + SSE$	$R^2 = SSR/SST$	$R^2 = 1 - SSE/SST$

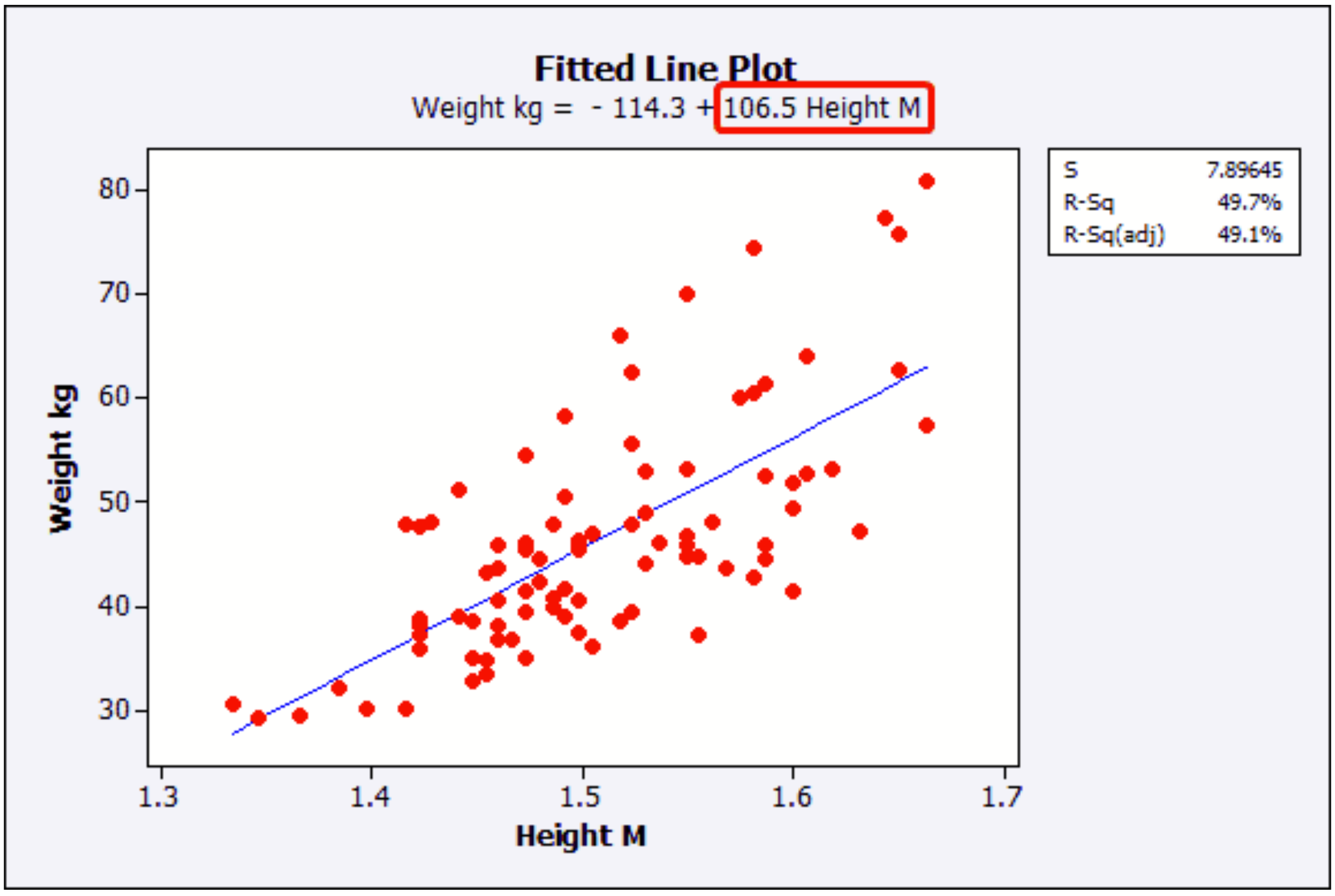
Analise de Regressão

Mean squared error	$\mathrm{MSE} = \frac{1}{n}\sum_{t=1}^{n} e_t^2$
Root mean squared error	$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n} e_t^2}$
Mean absolute error	$\mathrm{MAE} = \frac{1}{n}\sum_{t=1}^{n} \lvert e_t \rvert$
Mean absolute percentage error	$\mathrm{MAPE} = \frac{100\%}{n}\sum_{t=1}^{n} \left\lvert \frac{e_t}{y_t} \right\rvert$

```
import matplotlib.pyplot as plt
import numpy as np
from sklearn import datasets, linear_model
from sklearn.metrics import mean_squared_error, r2_score

# Load the diabetes dataset
diabetes = datasets.load_diabetes()


# Use only one feature
diabetes_X = diabetes.data[:, np.newaxis, 2]

# Split the data into training/testing sets
diabetes_X_train = diabetes_X[:-20]
diabetes_X_test = diabetes_X[-20:]

# Split the targets into training/testing sets
diabetes_y_train = diabetes.target[:-20]
diabetes_y_test = diabetes.target[-20:]

# Create linear regression object
regr = linear_model.LinearRegression()

# Train the model using the training sets
regr.fit(diabetes_X_train, diabetes_y_train)

# Make predictions using the testing set
diabetes_y_pred = regr.predict(diabetes_X_test)

# The coefficients
print('Coefficients: \n', regr.coef_)
# The mean squared error
print("Mean squared error: %.2f"
      % mean_squared_error(diabetes_y_test, diabetes_y_pred))
# Explained variance score: 1 is perfect prediction
print('Variance score: %.2f' % r2_score(diabetes_y_test, diabetes_y_pred))
```

Analise de Regressão

Análise de Coeficientes



Analise de Regressão

Sum Squared Regression Error

$$R^2 = 1 - \frac{SS_{Regression}}{SS_{Total}}$$

Sum Squared Total Error

Analise de Regressão

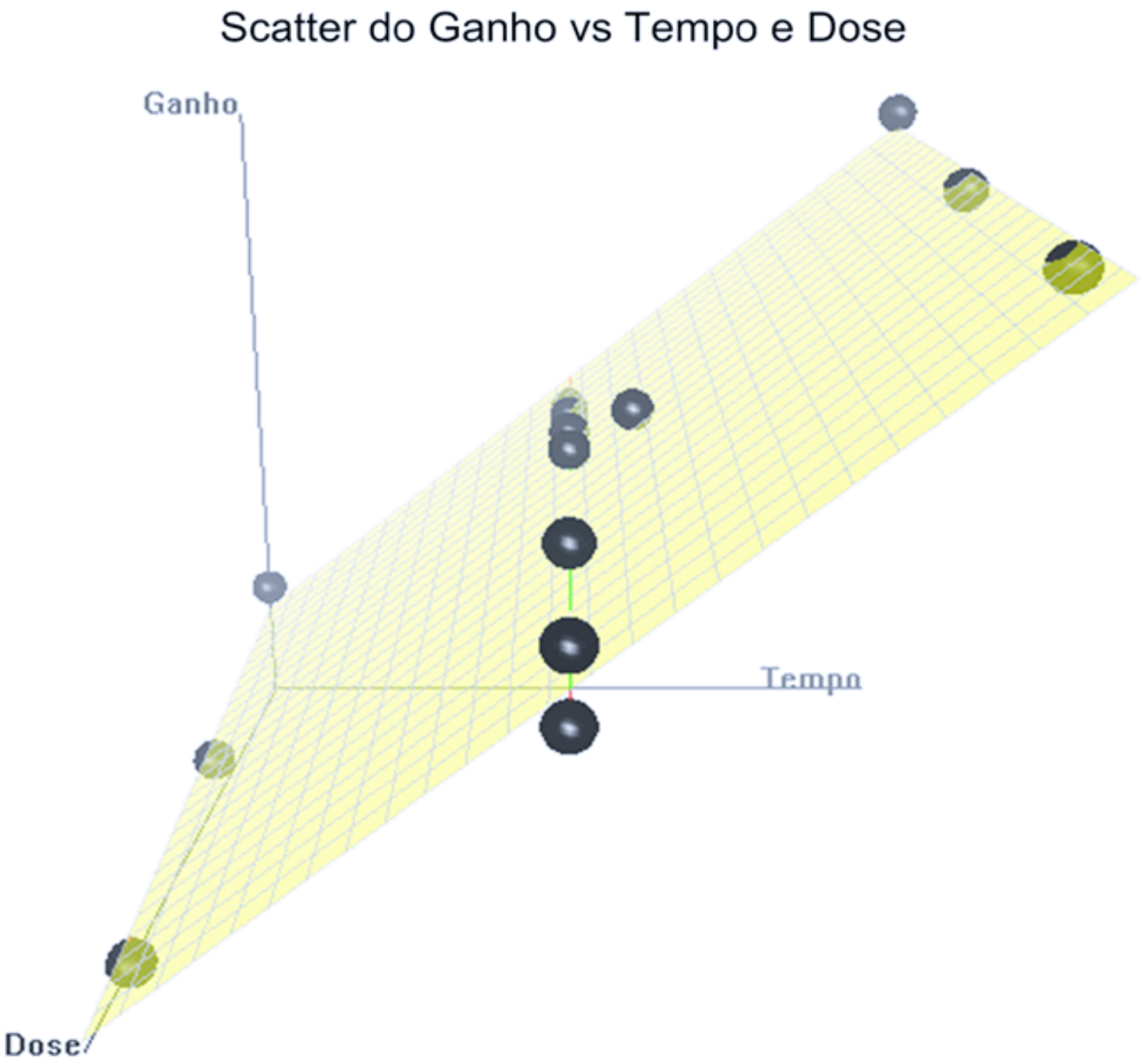
Zero Regression Error

$$R^2 = 1 - \frac{0}{SS_{Total}} \rightarrow 1.0$$

```
>>> from sklearn.metrics import r2_score
>>> y_true = [3, -0.5, 2, 7]
>>> y_pred = [2.5, 0.0, 2, 8]
>>> r2_score(y_true, y_pred)
```

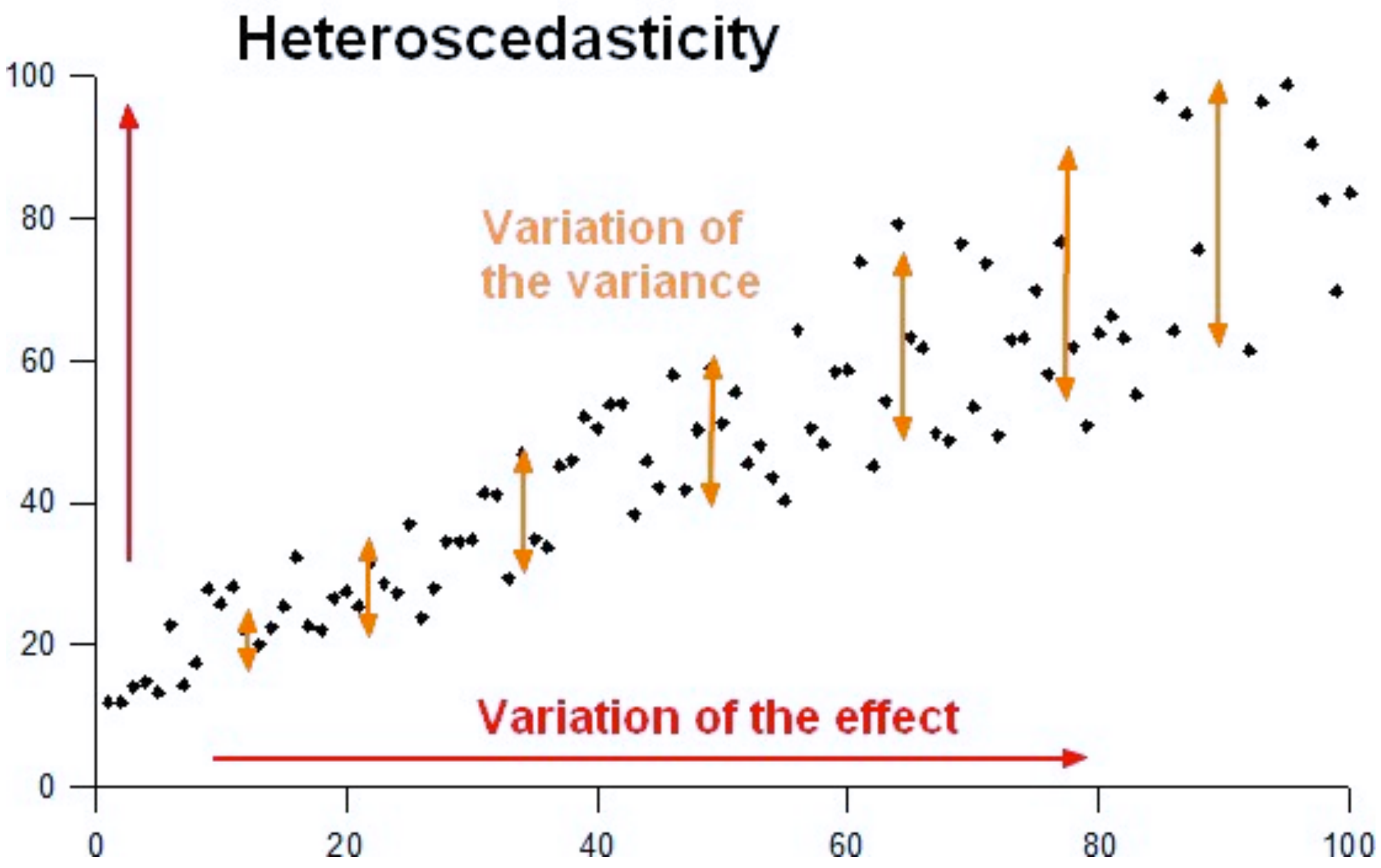
Analise de Regressão

Regressão Linear Multipla



Observação	Tempo (min)	Ganho (hFe)	Dose de íons ($\times 10^{14}$)
1	195	1004	4
2	255	1636	4
3	195	852	4,6
4	255	1506	4,6
5	225	1272	4,2
6	225	1270	4,1
7	225	1269	4,6
8	195	903	4,3
9	255	1555	4,3
10	225	1260	4
11	225	1146	4,7
12	225	1276	4,3
13	225	1225	4,72
14	230	1321	4,3

Analise de Regressão



Método dos Mínimos Quadrados

Método dos Mínimos Quadrados

Caso discreto ou Aproximação Linear

$$\begin{bmatrix} \sum_{i=1}^{qtd} x_i^2 & \sum_{i=1}^{qtd} x_i \\ \sum_{i=1}^{qtd} x_i & qtd \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{qtd} f(x_i).x_i \\ \sum_{i=1}^{qtd} f(x_i) \end{bmatrix}$$

Onde *qtd* é o número de valores da amostra, *F(x)* a função "desconhecida", e a função de aproximação será

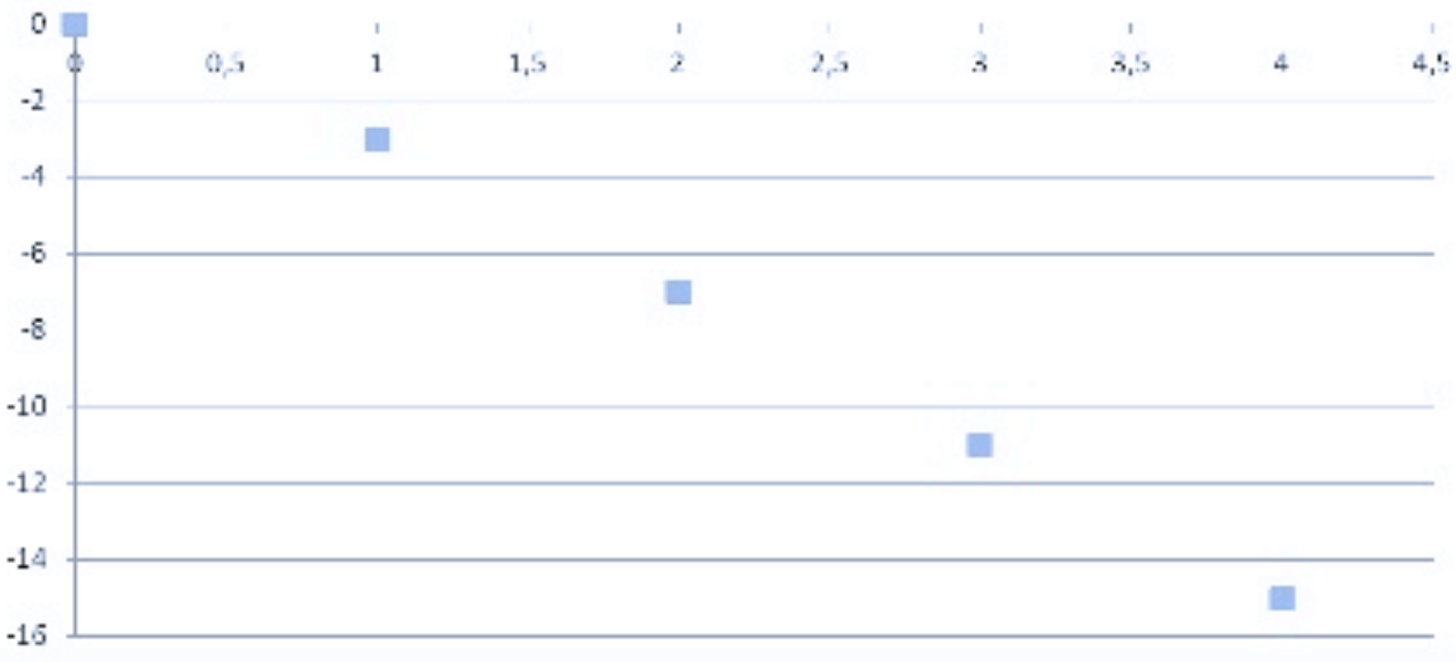
$$g(x) = a_1.x + a_2$$

Método dos Mínimos Quadrados

Caso discreto ou Aproximação Linear

Exemplo 1: Num experimento foram feitas 5 amostras com os respectivos valores. Qual a função linear que melhor representa os dados?

x	0	1	2	3	4
F(x)	0.98	-3,01	-6,99	-11,01	-15



Método dos Mínimos Quadrados

Exemplo 1 – Resolução:

Qtd é igual a cinco pois temos cinco amostras. Devemos obter os valores dos somatórios.

$$\begin{bmatrix} \sum_{i=1}^{5} x_i^2 & \sum_{i=1}^{5} x_i \\ \sum_{i=1}^{5} x_i & 5 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{5} f(x_i).x_i \\ \sum_{i=1}^{5} f(x_i) \end{bmatrix}$$

Método dos Mínimos Quadrados

Caso discreto ou Aproximação Linear

Exemplo 1 – Resolução:

	x_i	$F(x_i)$	x_i^2	$F(x_i).\ x_i$
	0	0,98	0	0,00
	1	-3,01	1	-3,01
	2	-6,99	4	-13,98
	3	-11,01	9	-33,03
	4	-15,00	16	-60,00
Σ	**10**	**35,03**	**30**	**-110,02**

$$\begin{bmatrix} 30 & 10 \\ 10 & 5 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} -110{,}02 \\ -35{,}03 \end{bmatrix}$$

Método dos Mínimos Quadrados

Caso discreto ou Aproximação Linear

Exemplo 1 – Resolução:

Solucionando o sistema:

a1=-3,9960 e a2=0,9860

Então:

$g(x) = -3,9960.x + 0,9860$

Se desejássemos prever o valor da décima amostra, teríamos:

$g(10) = -3,9960.10+0,0960 =$ **-39,864**

Método dos Mínimos Quadrados

Aproximação por polinômio de 2º Grau

$$\begin{bmatrix} \sum_{i=1}^{qtd} x_i^4 & \sum_{i=1}^{qtd} x_i^3 & \sum_{i=1}^{qtd} x_i^2 \\ \sum_{i=1}^{qtd} x_i^3 & \sum_{i=1}^{qtd} x_i^2 & \sum_{i=1}^{qtd} x_i \\ \sum_{i=1}^{qtd} x_i^2 & \sum_{i=1}^{qtd} x_i & qtd \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{qtd} f(x_i).x_i^2 \\ \sum_{i=1}^{qtd} f(x_i).x_i \\ \sum_{i=1}^{qtd} f(x_i) \end{bmatrix}$$

Onde *qtd* é o número de valores da amostra, *F(x)* a função "desconhecida", e a função de aproximação será

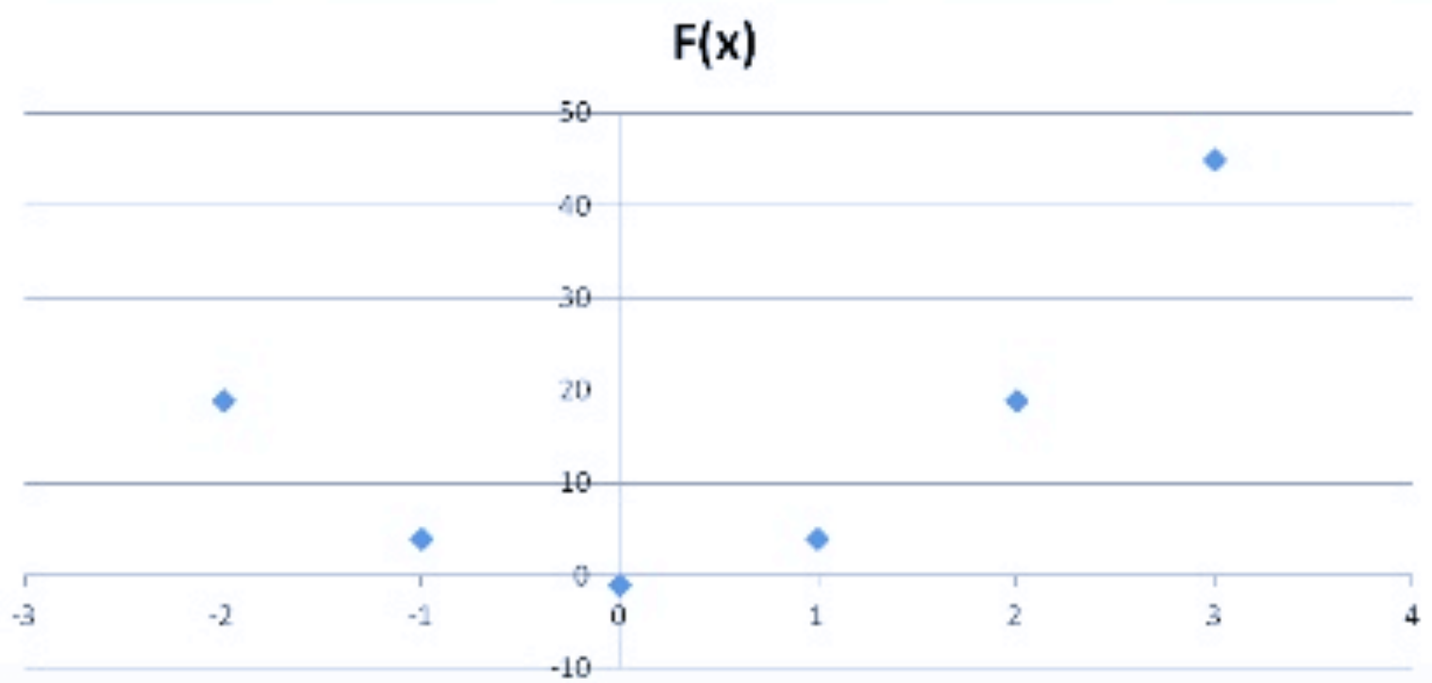
$$g(x) = a_1.x^2 + a_2.x + a_2$$

Método dos Mínimos Quadrados

Caso discreto ou Aproximação Linear

Exemplo 2: Num experimento foram feitas 6 amostras com os respectivos valores. Qual o polinômio de segundo grau que melhor representa os dados?

x	-2	-1	0	1	2	3
F(x)	19,01	3,99	-1,00	4,01	18,99	45,00



Método dos Mínimos Quadrados

Caso discreto ou Aproximação Linear

Exemplo 2 – Resolução:

Qtd é igual a 6 pois temos seis amostras. Devemos obter os valores dos somatórios.

$$\begin{bmatrix} \sum_{i=1}^{6} x_i^4 & \sum_{i=1}^{6} x_i^3 & \sum_{i=1}^{6} x_i^2 \\ \sum_{i=1}^{6} x_i^3 & \sum_{i=1}^{6} x_i^2 & \sum_{i=1}^{6} x_i \\ \sum_{i=1}^{6} x_i^2 & \sum_{i=1}^{6} x_i & 6 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{6} f(x_i).x_i^2 \\ \sum_{i=1}^{6} f(x_i).x_i \\ \sum_{i=1}^{6} f(x_i) \end{bmatrix}$$

Método dos Mínimos Quadrados

Exemplo 1 – Resolução:

	x_i	x_i^2	x_i^3	x_i^4	$F(x_i)$	$F(x_i).x_i$	$F(x_i).x_i^2$
	-2	4	-8	16	19,01	-38,02	76,04
	-1	1	-1	1	3,99	-3,99	3,99
	0	0	0	0	-1,00	0,00	0,00
	1	1	1	1	4,01	4,01	4,01
	2	4	8	16	18,99	37,98	75,96
	3	9	27	81	45,00	135,00	405,00
Σ	**3**	**19**	**27**	**115**	**90,00**	**134,98**	**565,00**

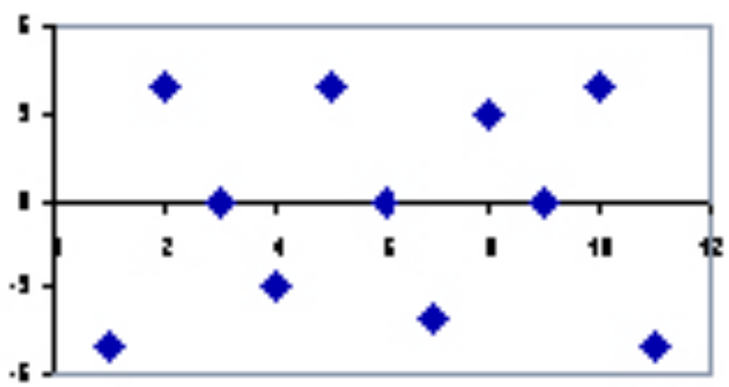
$$\begin{bmatrix} 115 & 27 & 19 \\ 27 & 19 & 3 \\ 19 & 3 & 6 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} 565 \\ 134{,}98 \\ 90 \end{bmatrix}$$

Análise de Resíduos

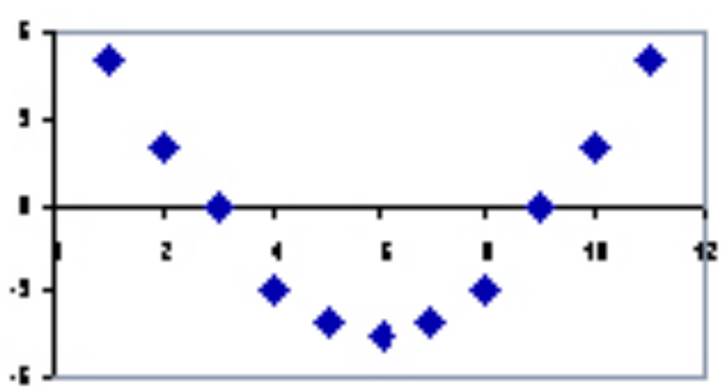
Análise de Resíduos

Residual Plots

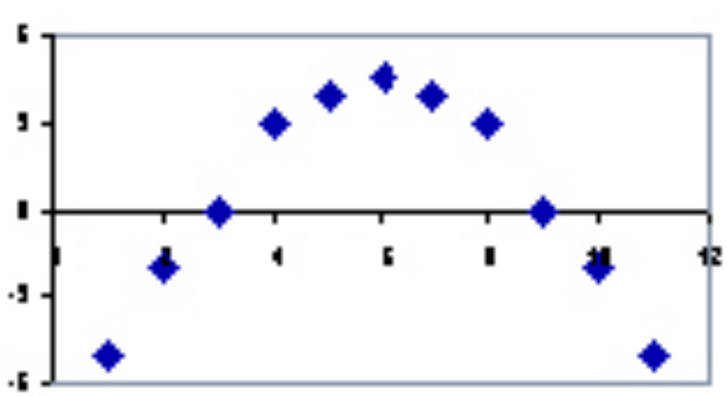
A **residual plot** is a graph that shows the residuals on the vertical axis and the independent variable on the horizontal axis. If the points in a residual plot are randomly dispersed around the horizontal axis, a linear regression model is appropriate for the data; otherwise, a non-linear model is more appropriate.



Random pattern



Non-random: U-shaped



Non-random: Inverted U

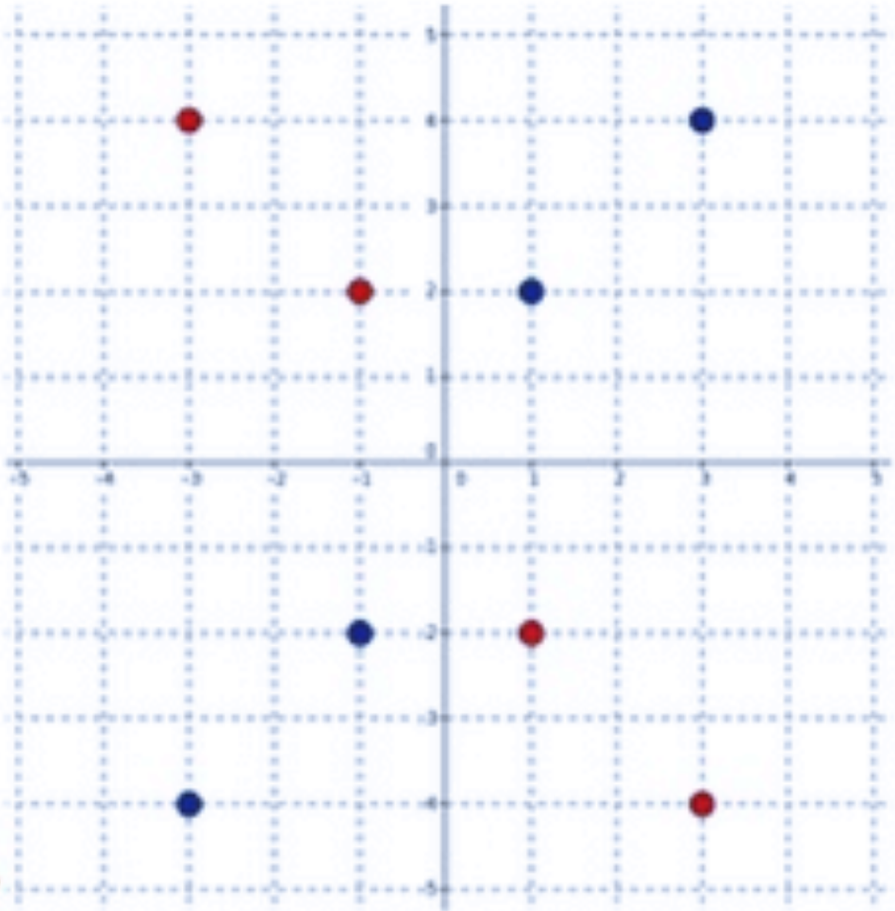
Análise de Resíduos

```
import numpy as np
import seaborn as sns
sns.set(style="whitegrid")

# Make an example dataset with y ~ x
rs = np.random.RandomState(7)
x = rs.normal(2, 1, 75)
y = 2 + 1.5 * x + rs.normal(0, 2, 75)

# Plot the residuals after fitting a linear model
sns.residplot(x, y, lowess=True, color="g")
```

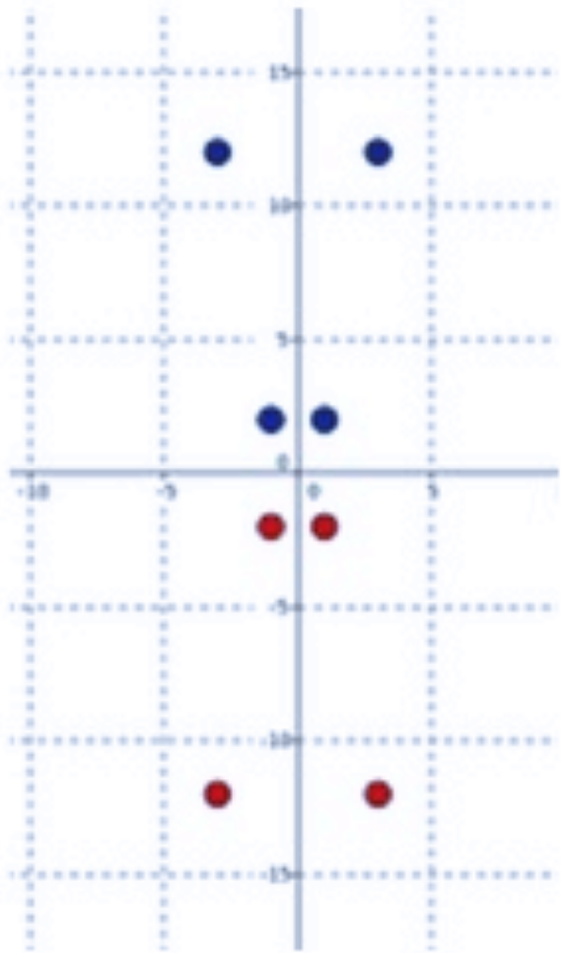
Engenharia de Features

Engenharia de Features

Feature Engineering



$$(x_1', x_2') = (x_1, x_1 x_2)$$



Decision Boundary in Transformed Space

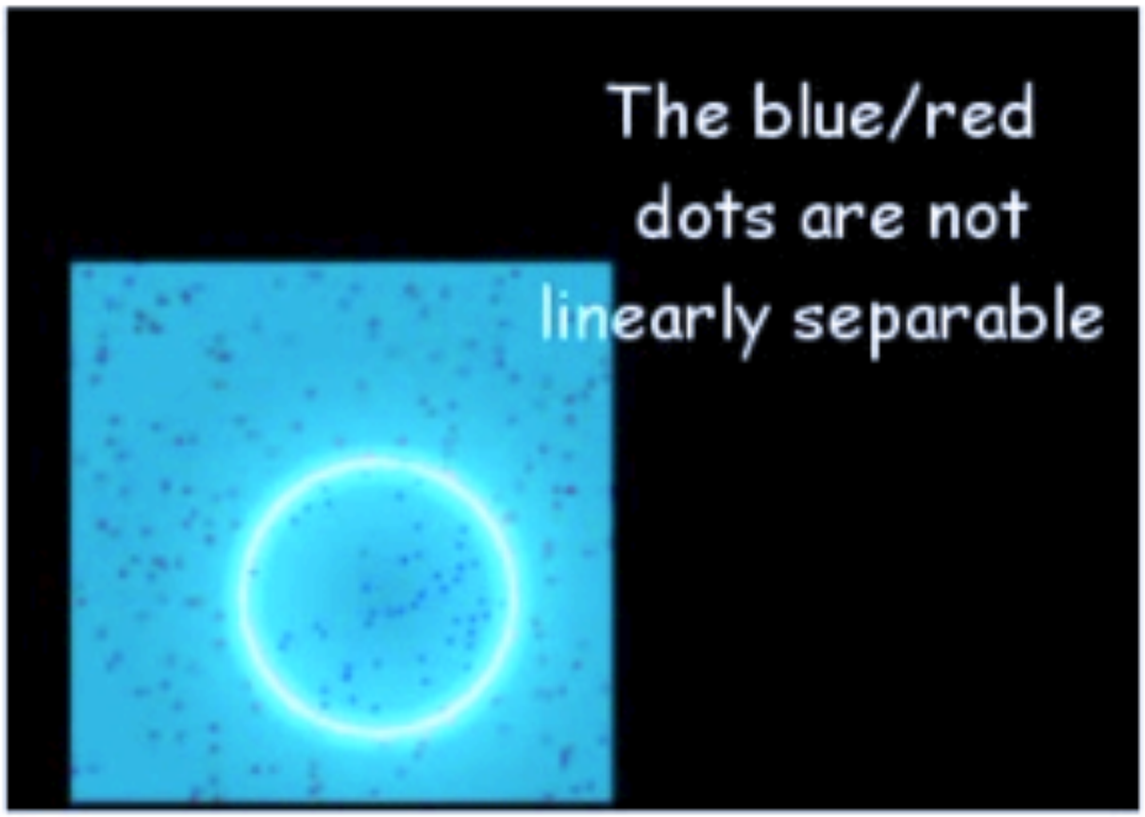
$$x_2' = 0$$

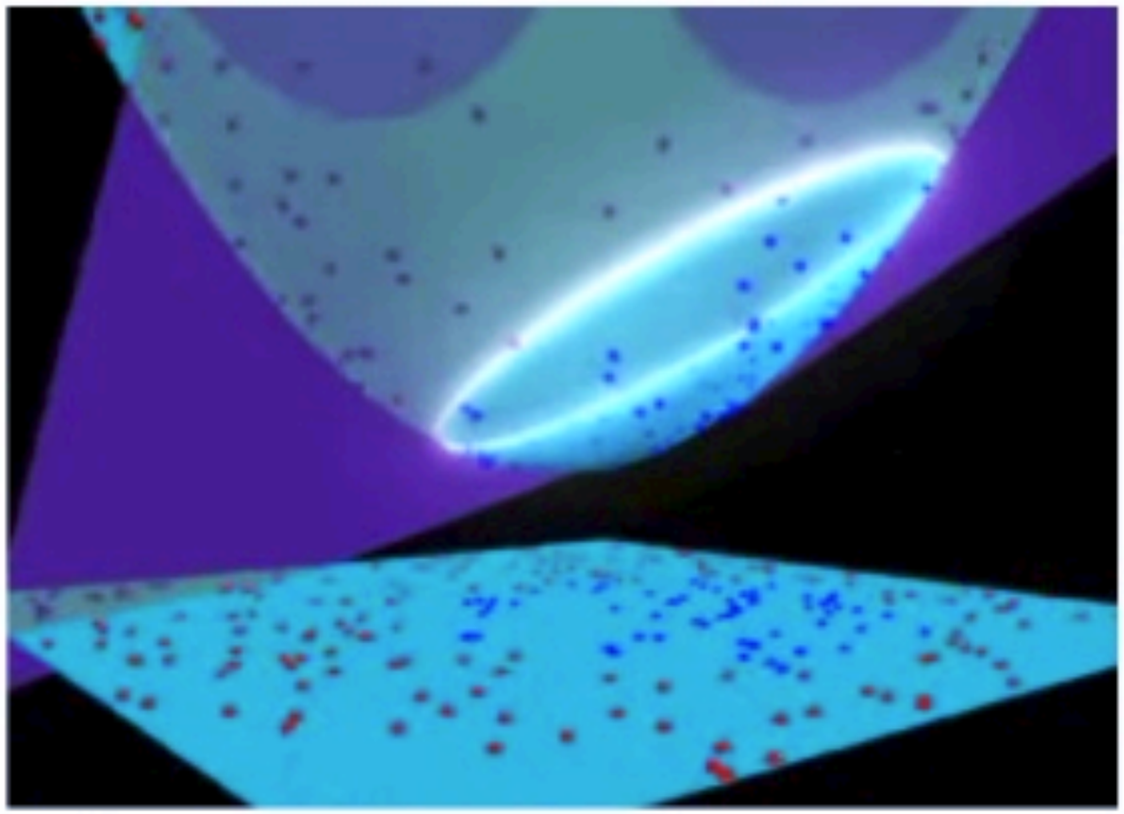
Decision Boundary in Original Space

$$x_1' x_2' = 0$$

Engenharia de Features

Feature Engineering





Ref: https://youtu.be/3liCbRZPrZA

Engenharia de Features

One Hot Encoding

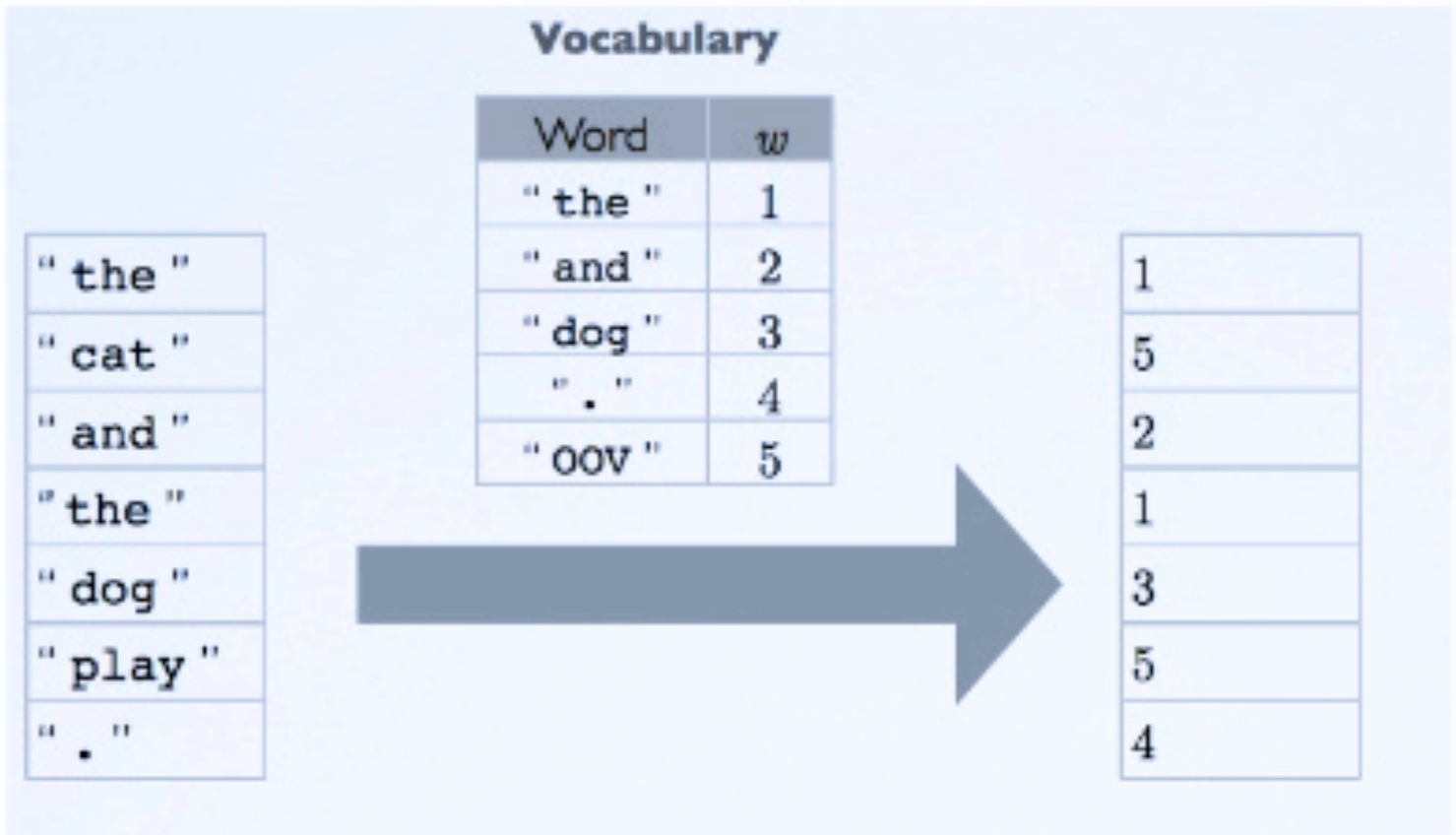
Color
Red
Red
Yellow
Green
Yellow

Red	Yellow	Green
1	0	0
1	0	0
0	1	0
0	0	1

Engenharia de Features

One-hot encoding

- Form vocabulary of words that maps lemmatized words to a unique ID (position of word in vocabulary)
- Typical vocabulary sizes will vary between 10 000 and 250 000



Engenharia de Features

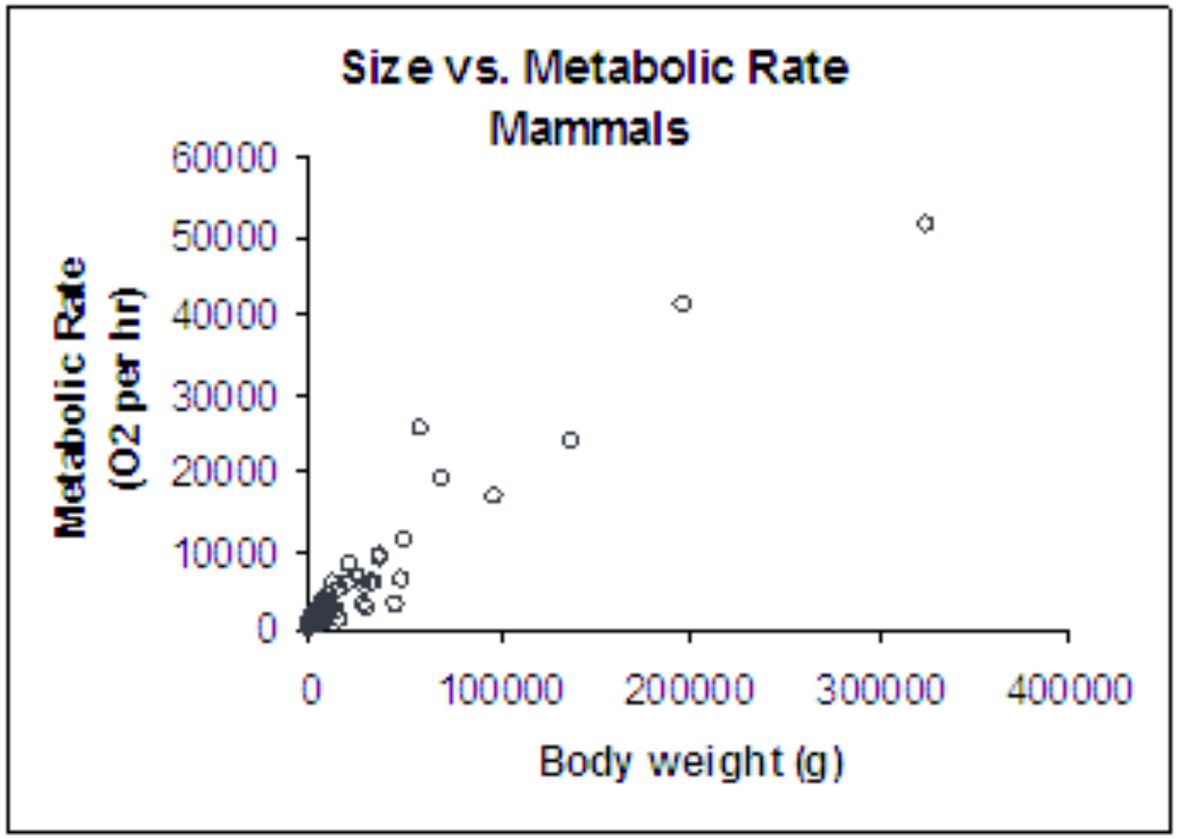
```
>>> from sklearn.preprocessing import OneHotEncoder
>>> enc = OneHotEncoder()
>>> enc.fit([[0, 0, 3], [1, 1, 0], [0, 2, 1], [1, 0, 2]])
OneHotEncoder(categorical_features='all', dtype=<... 'numpy.float64'>
       handle_unknown='error', n_values='auto', sparse=True)
>>> enc.n_values_
array([2, 3, 4])
>>> enc.feature_indices_
array([0, 2, 5, 9])
>>> enc.transform([[0, 1, 1]]).toarray()
array([[ 1.,  0.,  0.,  1.,  0.,  0.,  1.,  0.,  0.]])
```

Engenharia de Features

Transformations of Variables

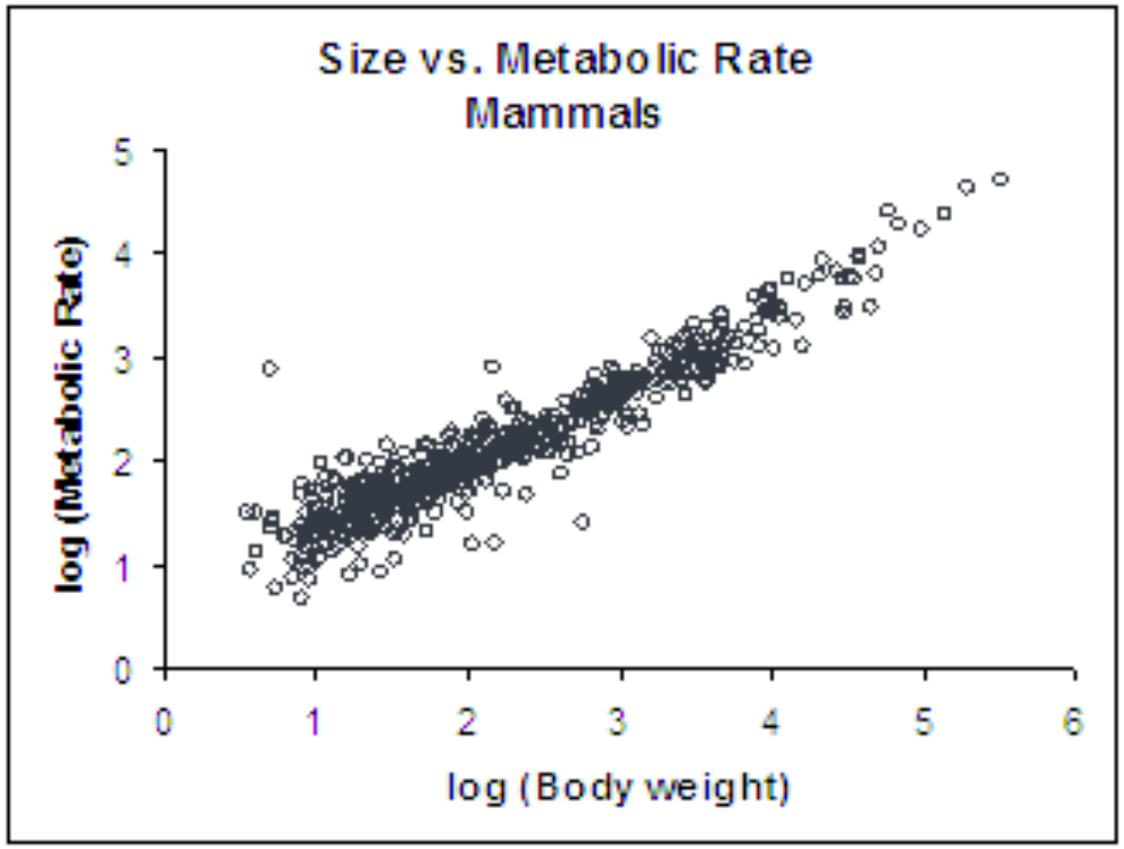
Linear transformation. A linear transformation preserves linear relationships between variables. Therefore, the correlation between *x* and *y* would be unchanged after a linear transformation. Examples of a linear transformation to variable *x* would be multiplying *x* by a constant, dividing *x* by a constant, or adding a constant to *x*.
Nonlinear tranformation. A nonlinear transformation changes (increases or decreases) linear relationships between variables and, thus, changes the correlation between variables. Examples of a nonlinear transformation of variable *x* would be taking the square root of *x* or the reciprocal of *x*.

Engenharia de Features

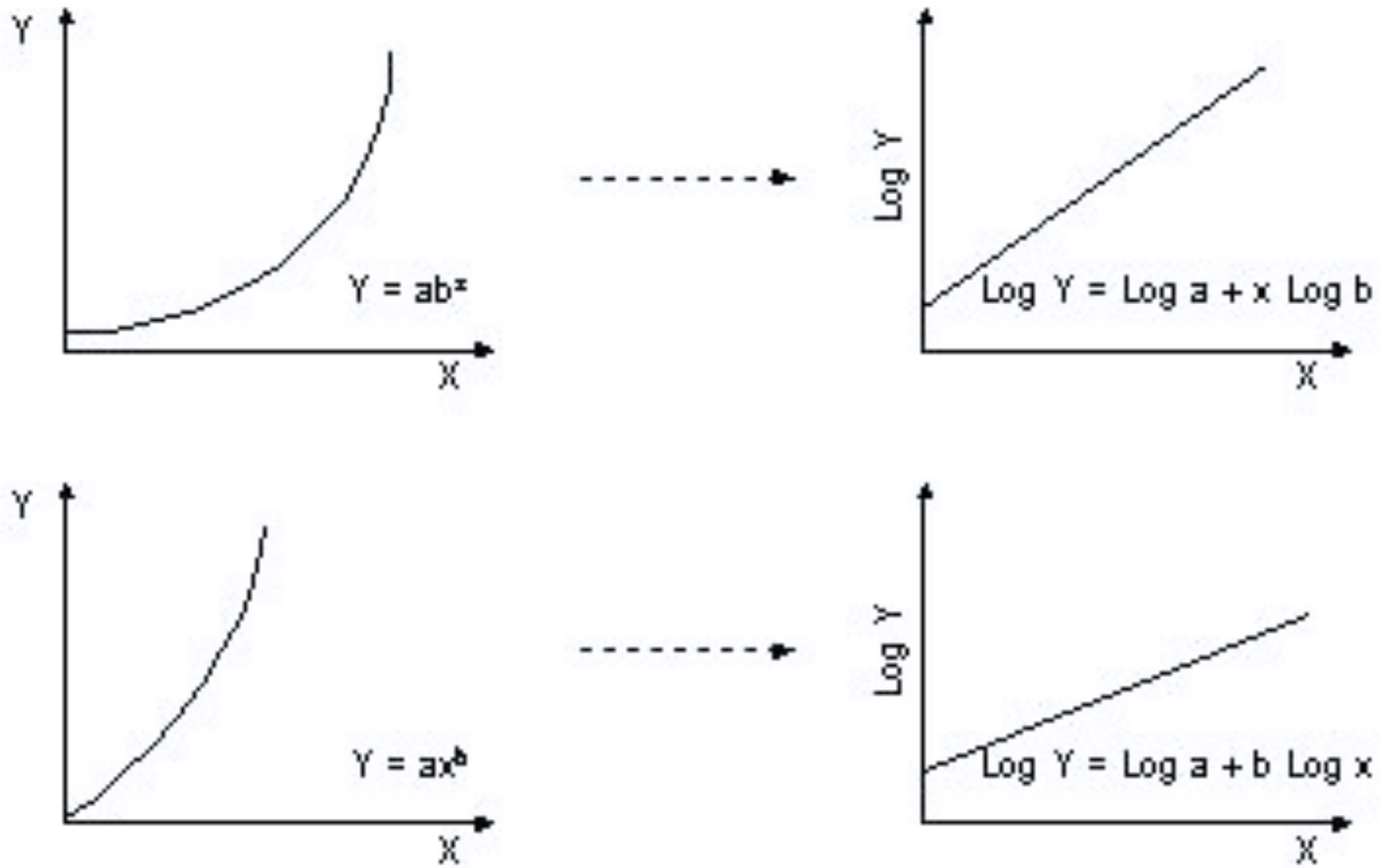


Log

Trans-
form



Engenharia de Features



Engenharia de Features

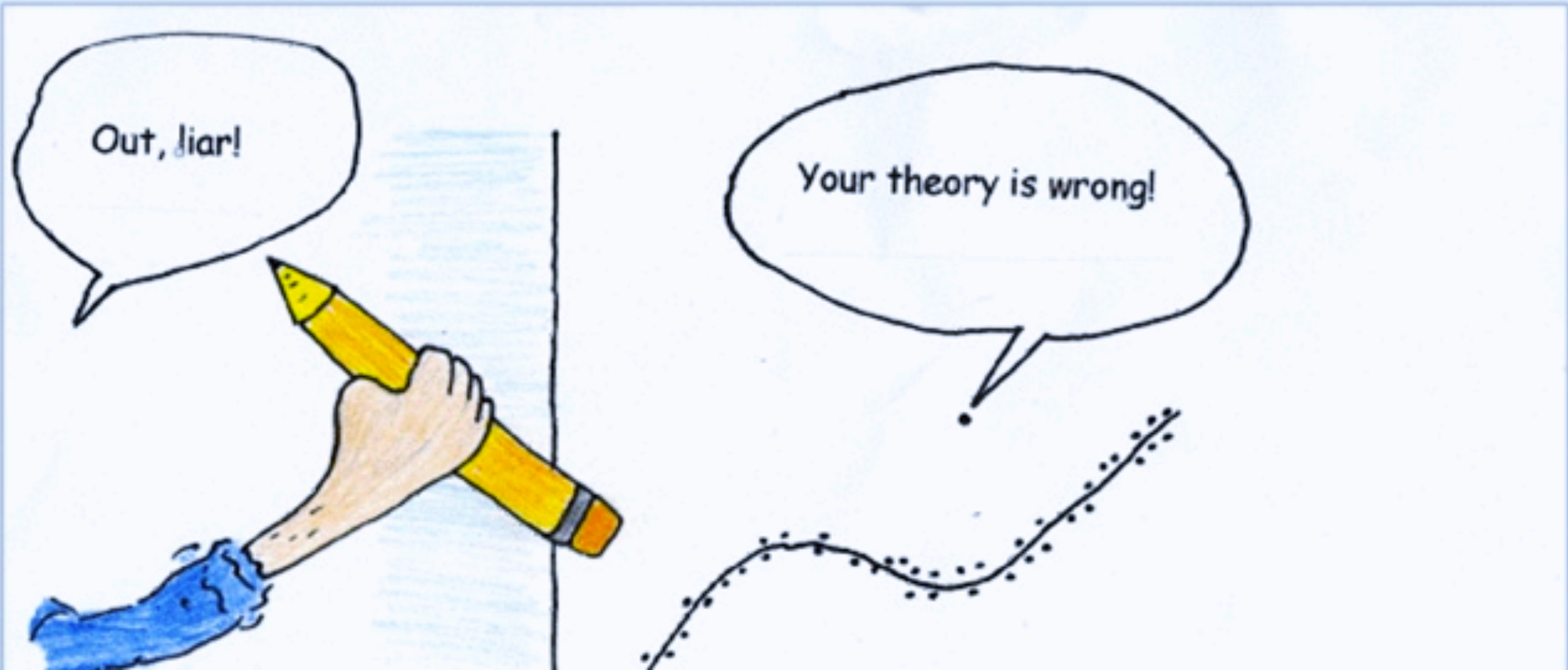
```
>>> from sklearn import preprocessing
>>> import numpy as np
>>> X_train = np.array([[ 1., -1.,  2.],
...                     [ 2.,  0.,  0.],
...                     [ 0.,  1., -1.]])
>>> X_scaled = preprocessing.scale(X_train)
```

```
>>> X_scaled
array([[ 0.  ..., -1.22...,  1.33...],
       [ 1.22...,  0.  ..., -0.26...],
       [-1.22...,  1.22..., -1.06...]])
```

Engenharia de Features

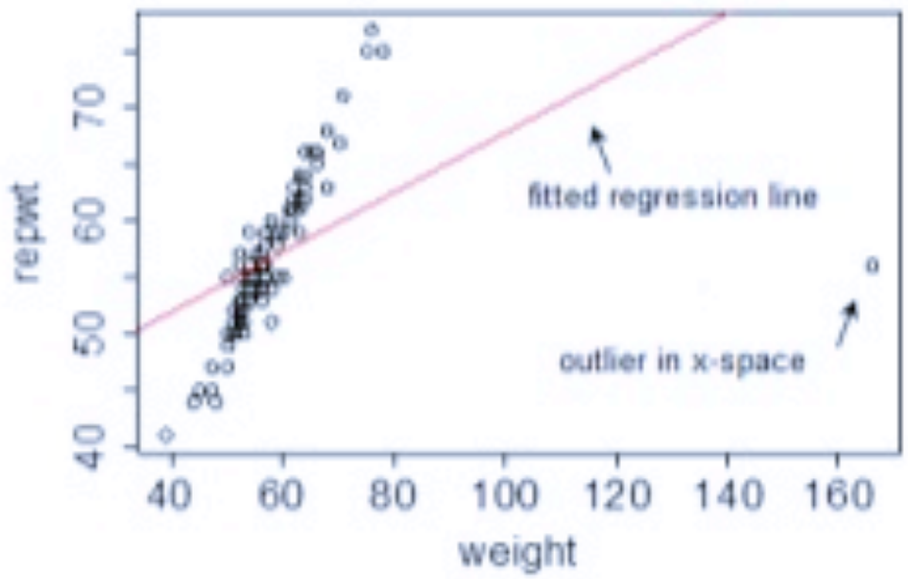
Method	Transform	Regression equation	Predicted value ($\hat{y}$)
Standard linear regression	None	$y = b_0 + b_1 x$	$\hat{y} = b_0 + b_1 x$
Exponential model	DV = log(y)	$\log(y) = b_0 + b_1 x$	$\hat{y} = 10^{b_0 + b_1 x}$
Quadratic model	DV = sqrt(y)	$\text{sqrt}(y) = b_0 + b_1 x$	$\hat{y} = (b_0 + b_1 x)^2$
Reciprocal model	DV = 1/y	$1/y = b_0 + b_1 x$	$\hat{y} = 1 / (b_0 + b_1 x)$
Logarithmic model	IV = log(x)	$y = b_0 + b_1 \log(x)$	$\hat{y} = b_0 + b_1 \log(x)$
Power model	DV = log(y) IV = log(x)	$\log(y) = b_0 + b_1 \log(x)$	$\hat{y} = 10^{b_0 + b_1 \log(x)}$

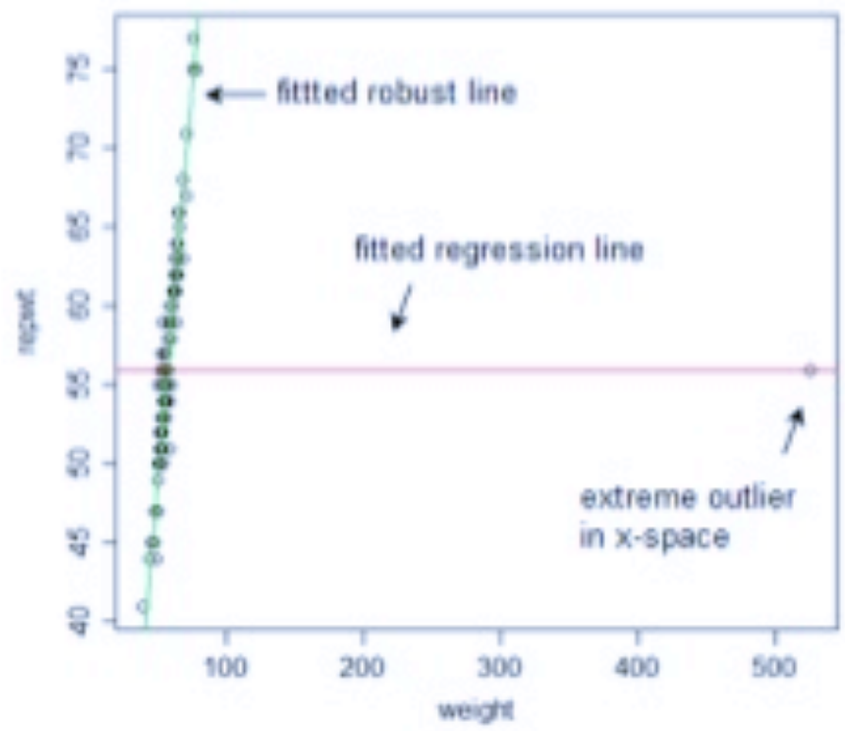
Engenharia de Features



Engenharia de Features

How outliers impact ordinary regression

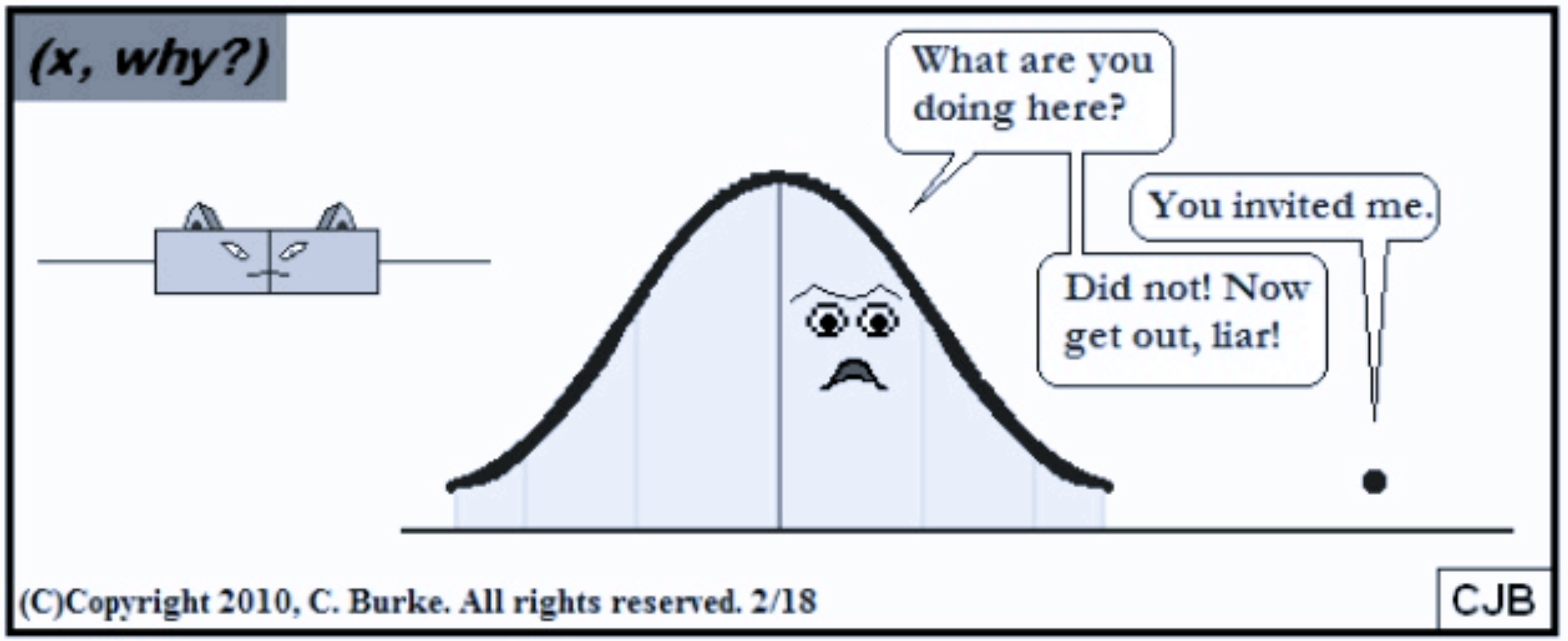




Even worse, with multiple regression, an outlier in x-space may not look particularly unusual for any single x-variable. If there's a possibility of such a point, it's potentially a very risky thing to use least squares regression on.

Engenharia de Features

WHAT CAUSES OUTLIERS?



Engenharia de Features

Data errors

Name	Age	Job
Andrew	24	Software Engineer
Jenny	255	UI Developer
Sam	30	Project Manager
Jenny	28	Teacher

Outliers are often caused by human error, such as errors in data collection, recording, or entry.

Engenharia de Features

Intentional or motivated mis-reporting.

- Social desirability and self-presentation motives can be powerful

- This can also happen for obvious reasons when data are sensitive (e.g., teenagers under-reporting drug or alcohol use, mis-reporting of sexual behavior).

http://www.livescience.com/7038-men-report-sex-partners-women.html

Engenharia de Features

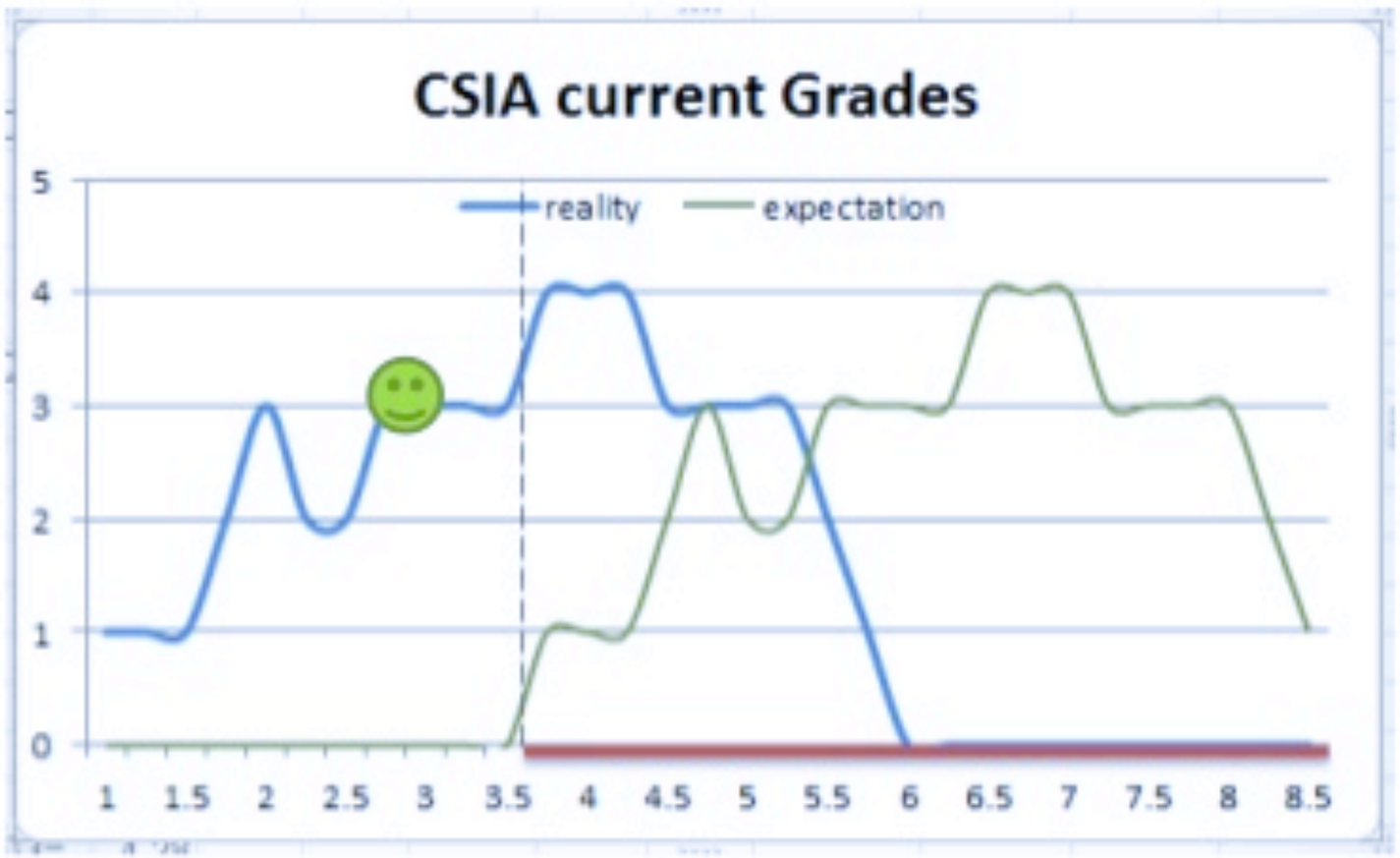
Standardization failure

If something anomalous happened during a particular subject's experience it will influence the study outcome.

Engenharia de Features

Faulty distributional assumptions

Incorrect assumptions about the distribution of the data can also lead to the presence of suspected outliers



Engenharia de Features

Legitimate cases sampled from the correct population

It is possible that an outlier can come from the population being sampled legitimately through random chance.

It is important to note that sample size plays a role in the probability of outlying values.

Engenharia de Features

Dealing with Outliers

- Removal

2,3,4,5,6,10,100 → 2,3,4,5,6,10,~~100~~

- Transformations (e.g.: taking the log)

2,3,4,5,6,10,100 →0.3, 0.47, 0.6, 0.69, 1 , 2

- Trunchiation

2,3,4,5,6,10,100 → 2,3,4,5,6,10,10

Engenharia de Features

Robust Methods – Trimmed mean

Is calculated by temporarily eliminating extreme observations at both ends of the sample (between 10%-25% of ends)

1, 6, 12, 14, 20, 24, 36, 100

Regular Mean: 26.62

~~1, 6~~, 12, 14, 20, 24, ~~36, 100~~

Trimmed Mean: 17.5

More Info

Engenharia de Features

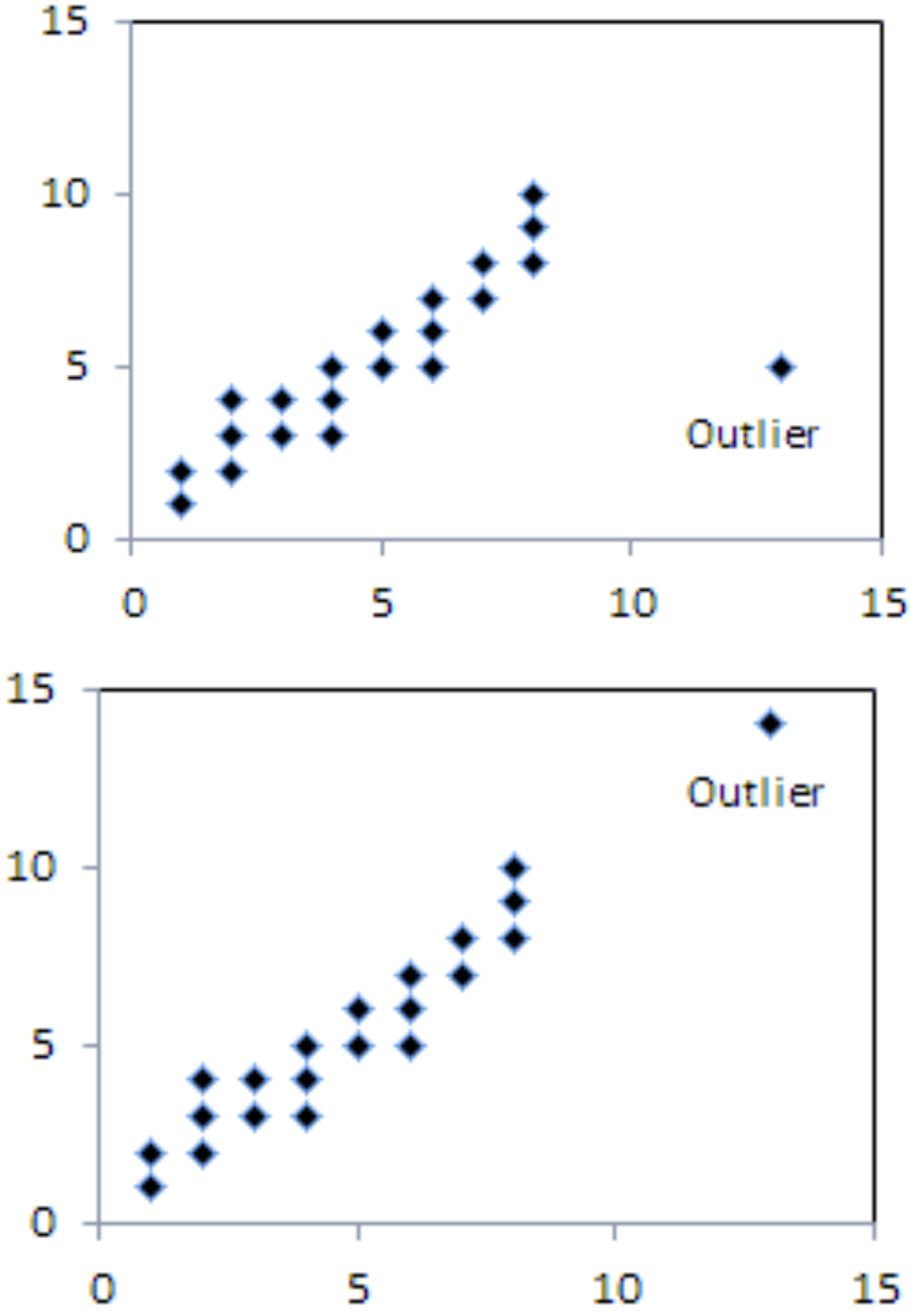
Robust Methods – **LMS**

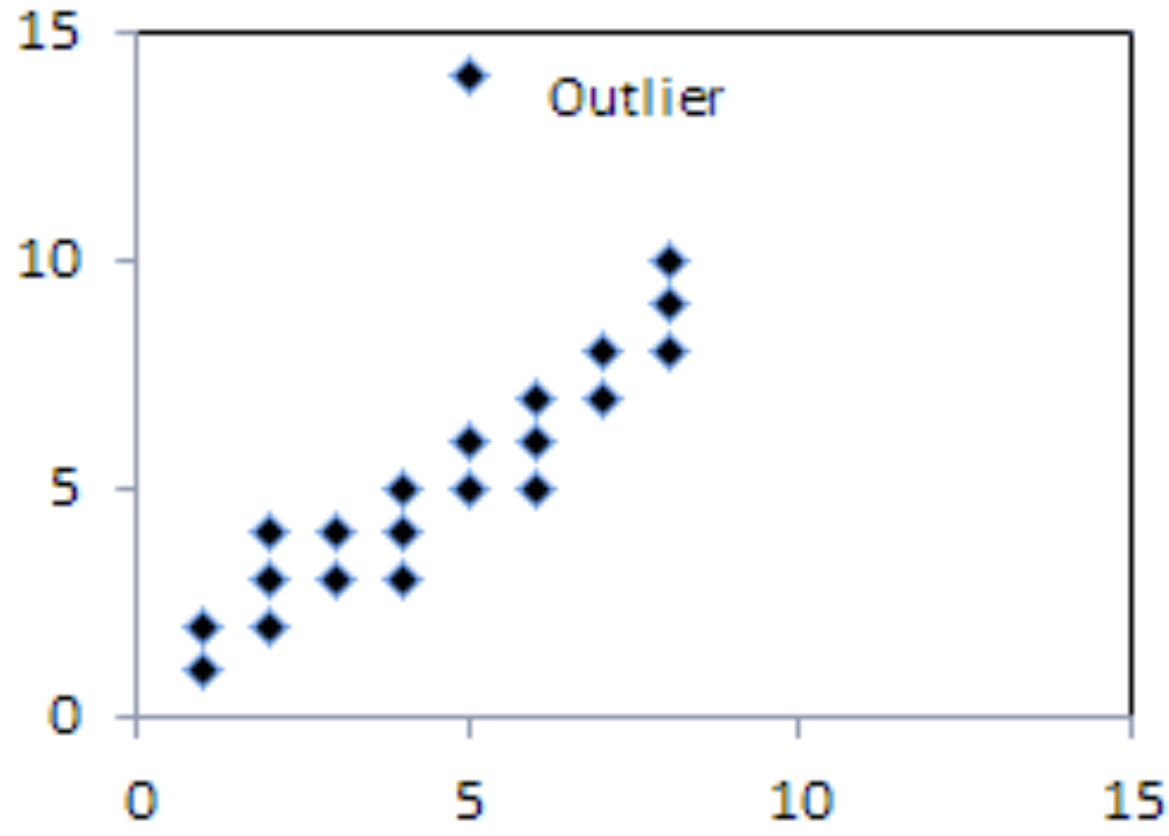
The **Least Median of Squares** (LMS) replaces the mean by the much less sensitive **median**, witch generates a more robust estimator.

1, 6, 12, 14, 20, 24, 36, 100
Regular Mean: 26.62

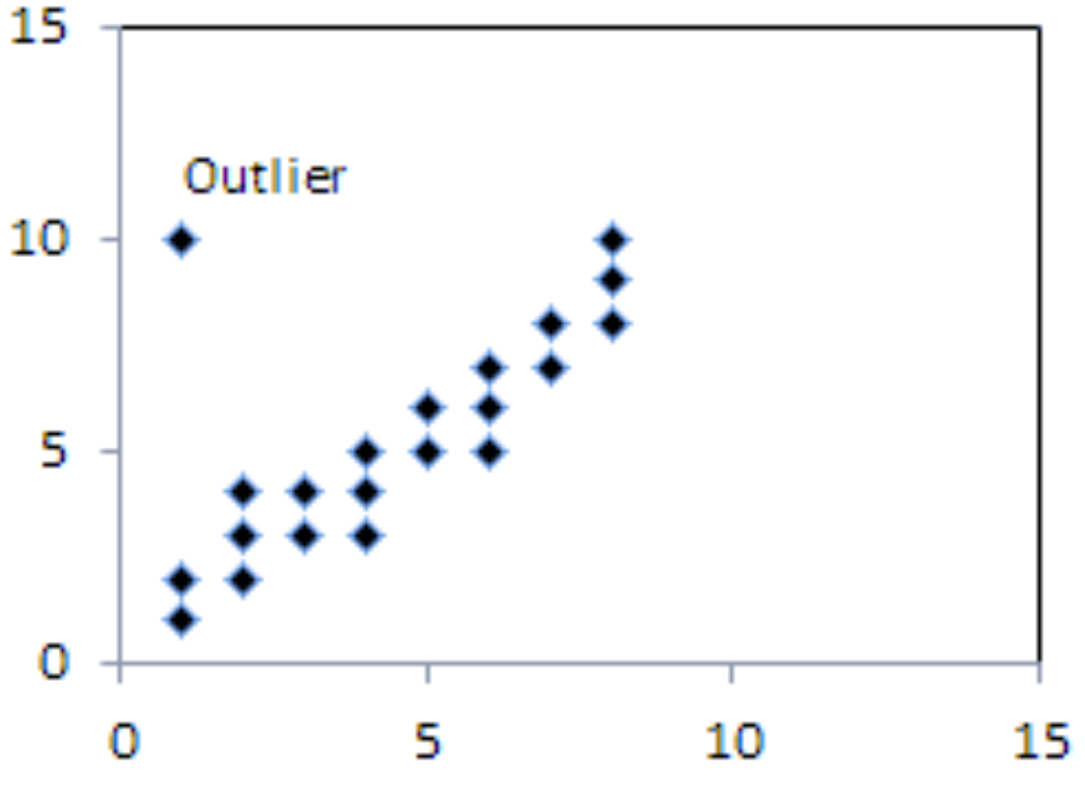
Median = (20+14)/2 = 17

OUTLIERS

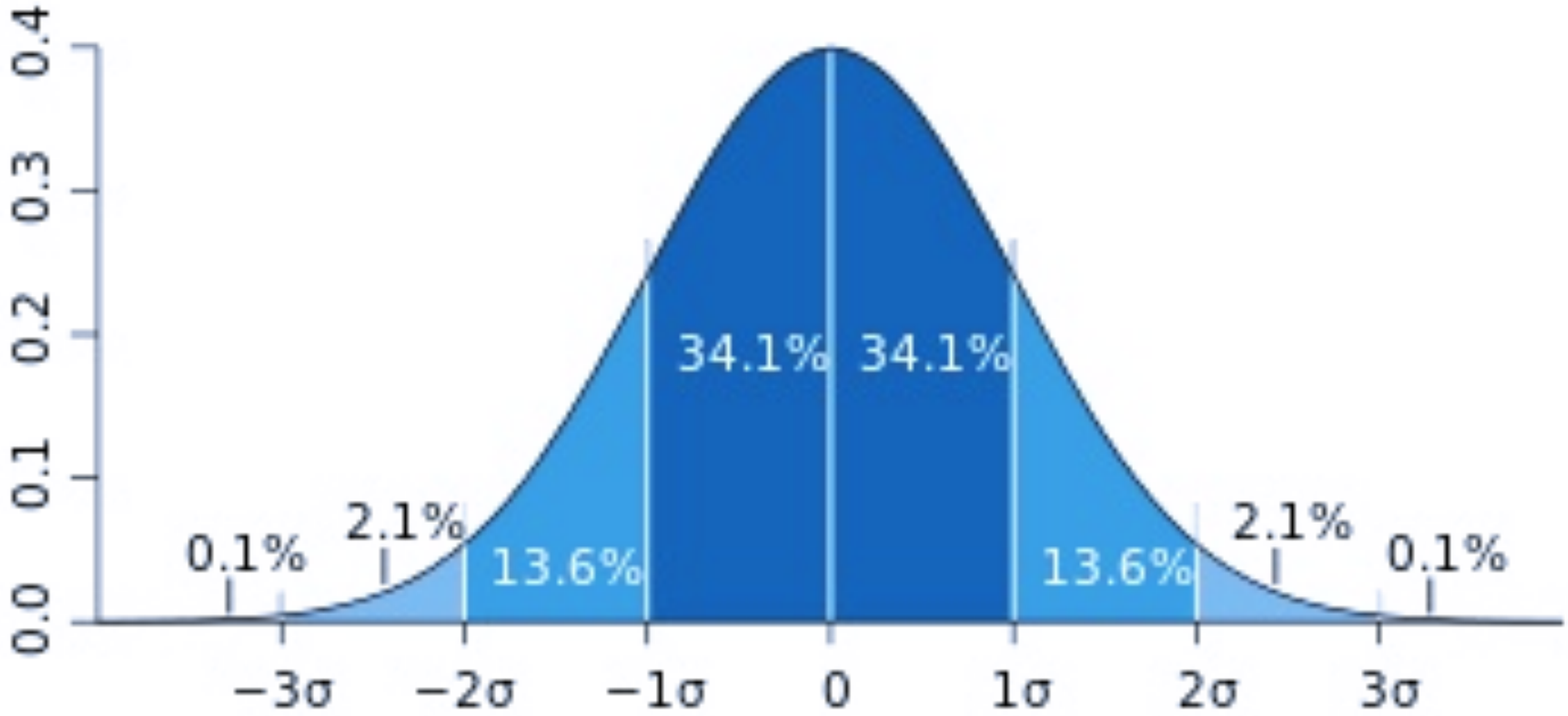

15
10
5
0
0
5
10
15
Outlier


15
10
5
0
0
5
10
15
Outlier

15
10
5
0
0
5
10
15
Outlier


15
10
5
0
0
5
10
15
Outlier

Engenharia de Features



Engenharia de Features

```
1   import numpy
2
3   arr = [10, 386, 479, 627, 20, 523, 482, 483, 542, 699, 535, 617
4
5   elements = numpy.array(arr)
6
7   mean = numpy.mean(elements, axis=0)
8   sd = numpy.std(elements, axis=0)
9
10  final_list = [x for x in arr if (x > mean - 2 * sd)]
11  final_list = [x for x in final_list if (x < mean + 2 * sd)]
12  print(final_list)
```

Testes de hipóteses individuais para os coeficientes da regressão são fundamentais para se determinar se cada variável explicativa é importante para o modelo de regressão. Por exemplo, o modelo pode ser mais eficaz com a inclusão ou com a exclusão de novas variáveis

How Do I Interpret the P-Values in Linear Regression Analysis?

The p-value for each term tests the null hypothesis that the coefficient is equal to zero (no effect). A low p-value (< 0.05) indicates that you can reject the null hypothesis. In other words, a predictor that has a low p-value is likely to be a meaningful addition to your model because changes in the predictor's value are related to changes in the response variable.

Coefficients

Term	Coef	SE Coef	T	P
Constant	389.166	66.0937	5.8881	0.000
East	2.125	1.2145	1.7495	0.092
South	5.318	0.9629	5.5232	0.000
North	-24.132	1.8685	-12.9153	0.000

Engenharia de Features

```
import pandas as pd
import numpy as np
from sklearn import datasets, linear_model
from sklearn.linear_model import LinearRegression
import statsmodels.api as sm
from scipy import stats

diabetes = datasets.load_diabetes()
X = diabetes.data
y = diabetes.target

X2 = sm.add_constant(X)
est = sm.OLS(y, X2)
est2 = est.fit()
print(est2.summary())
```

OLS Regression Results

Dep. Variable:	y	R-squared:	0.518
Model:	OLS	Adj. R-squared:	0.507
Method:	Least Squares	F-statistic:	46.27
Date:	Wed, 08 Mar 2017	Prob (F-statistic):	3.83e-62
Time:	10:08:24	Log-Likelihood:	-2386.0
No. Observations:	442	AIC:	4794.
Df Residuals:	431	BIC:	4839.
Df Model:	10		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	152.1335	2.576	59.061	0.000	147.071	157.196
x1	-10.0122	59.749	-0.168	0.867	-127.448	107.424
x2	-239.8191	61.222	-3.917	0.000	-360.151	-119.488
x3	519.8398	66.534	7.813	0.000	389.069	650.610
x4	324.3904	65.422	4.958	0.000	195.805	452.976
x5	-792.1842	416.684	-1.901	0.058	-1611.169	26.801
x6	476.7458	339.035	1.406	0.160	-189.621	1143.113
x7	101.0446	212.533	0.475	0.635	-316.685	518.774
x8	177.0642	161.476	1.097	0.273	-140.313	494.442
x9	751.2793	171.902	4.370	0.000	413.409	1089.150
x10	67.6254	65.984	1.025	0.306	-62.065	197.316

Omnibus:	1.506	Durbin-Watson:	2.029
Prob(Omnibus):	0.471	Jarque-Bera (JB):	1.404
Skew:	0.017	Prob(JB):	0.496
Kurtosis:	2.726	Cond. No.	227.

Engenharia de Features

Feature Selection

IMPORTANT!
(Noise, overfitting, curse of dimensionality, efficiency)

- Domain knowledge
- Variance threshold
- Exhaustive search
- Decision trees
- …

Simplest example:
Greedy Backward Selection

start: $\mathbf{X} = [\mathbf{x}_1, \cancel{\mathbf{x}_2}, \mathbf{x}_3, \mathbf{x}_4]$

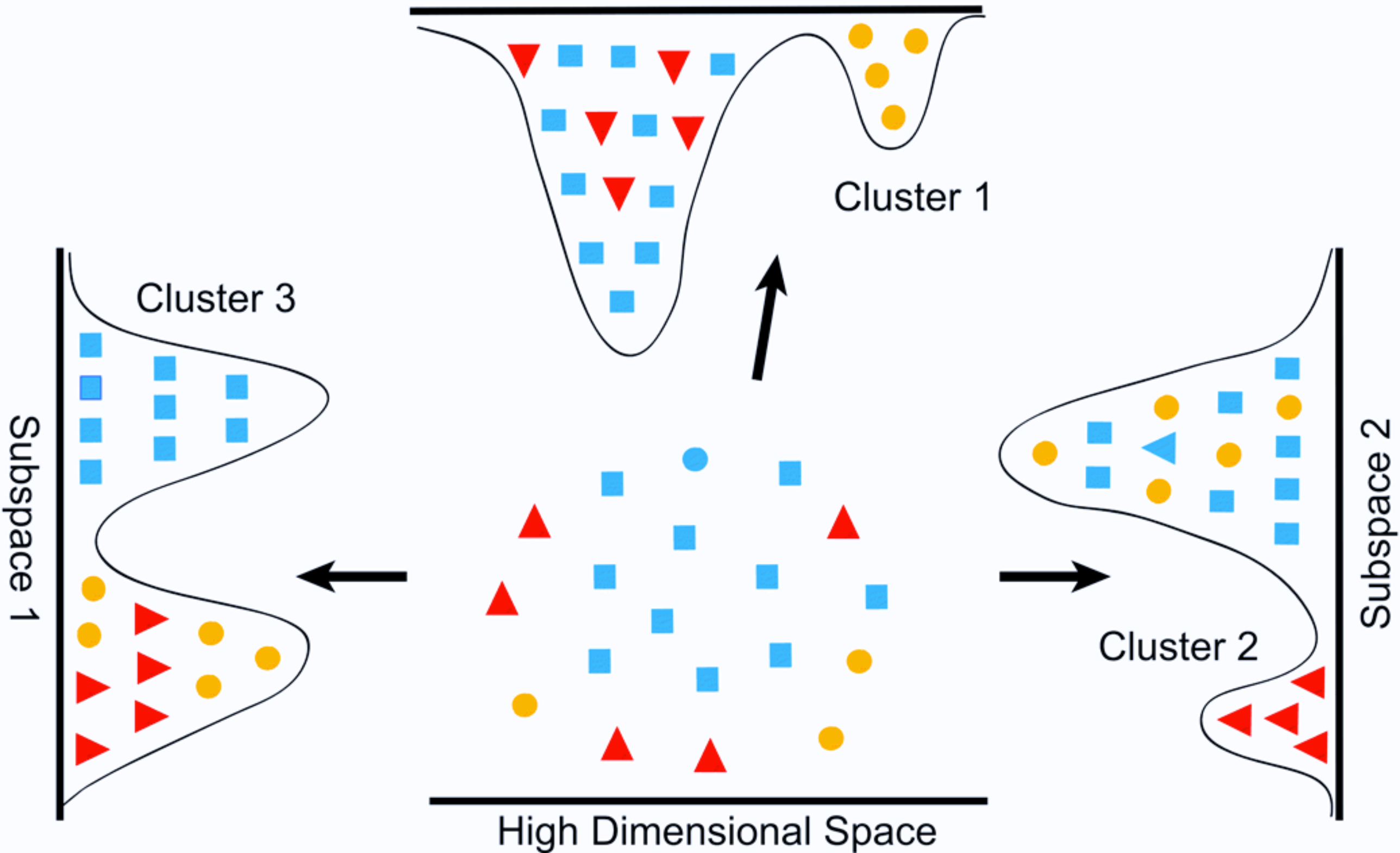
$\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_3, \cancel{\mathbf{x}_4}]$

stop: $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_3]$
(if d = k)

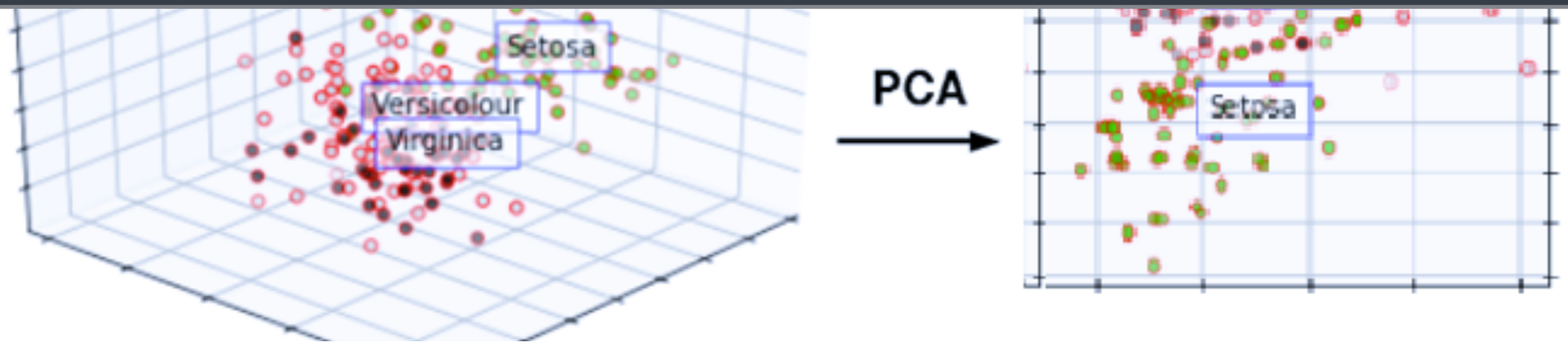
Engenharia de Features

- Seleção Manual de Variáveis
- Seleção Automática de Variáveis baseada em Árvores de Decisão
- Seleção Automática de Variáveis baseada em Regressão
- Redução de Dimensionalidade com PCA
- Redução de Dimensionalidade com Partial Least Squares (PLS)
- Estimativa de Parâmetros via Métodos de Encolhimento (Shrinkage)

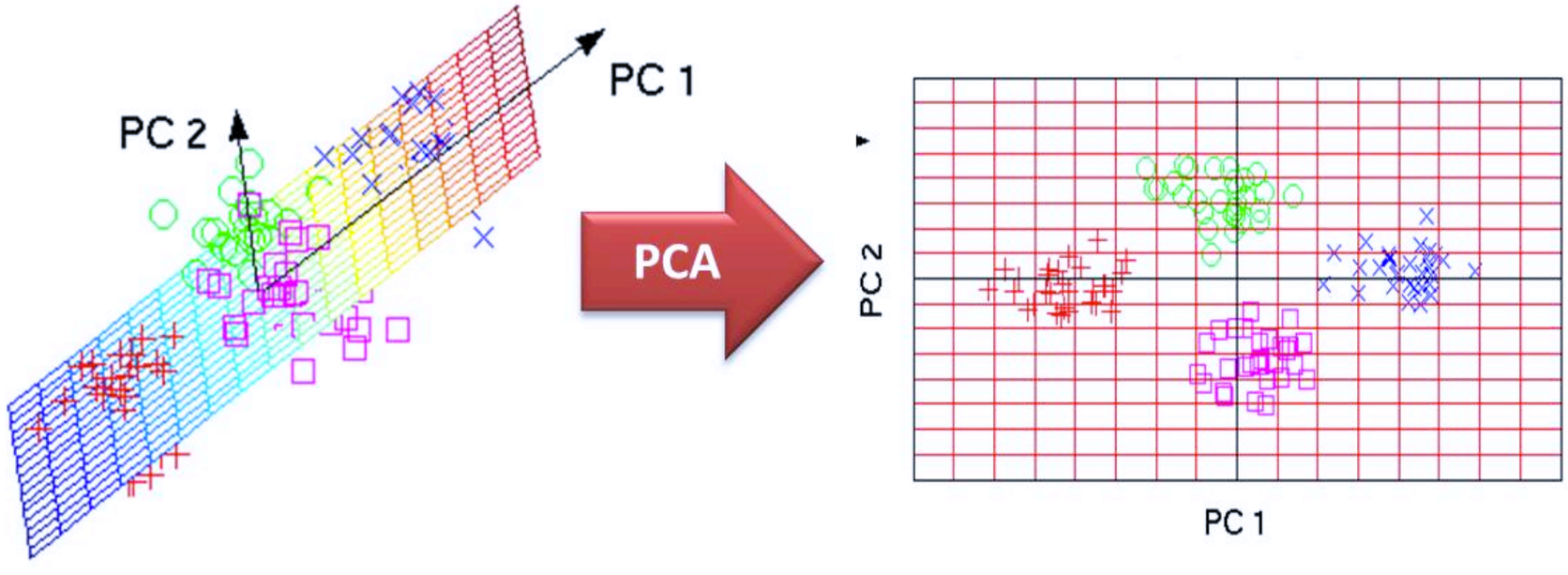
Engenharia de Features



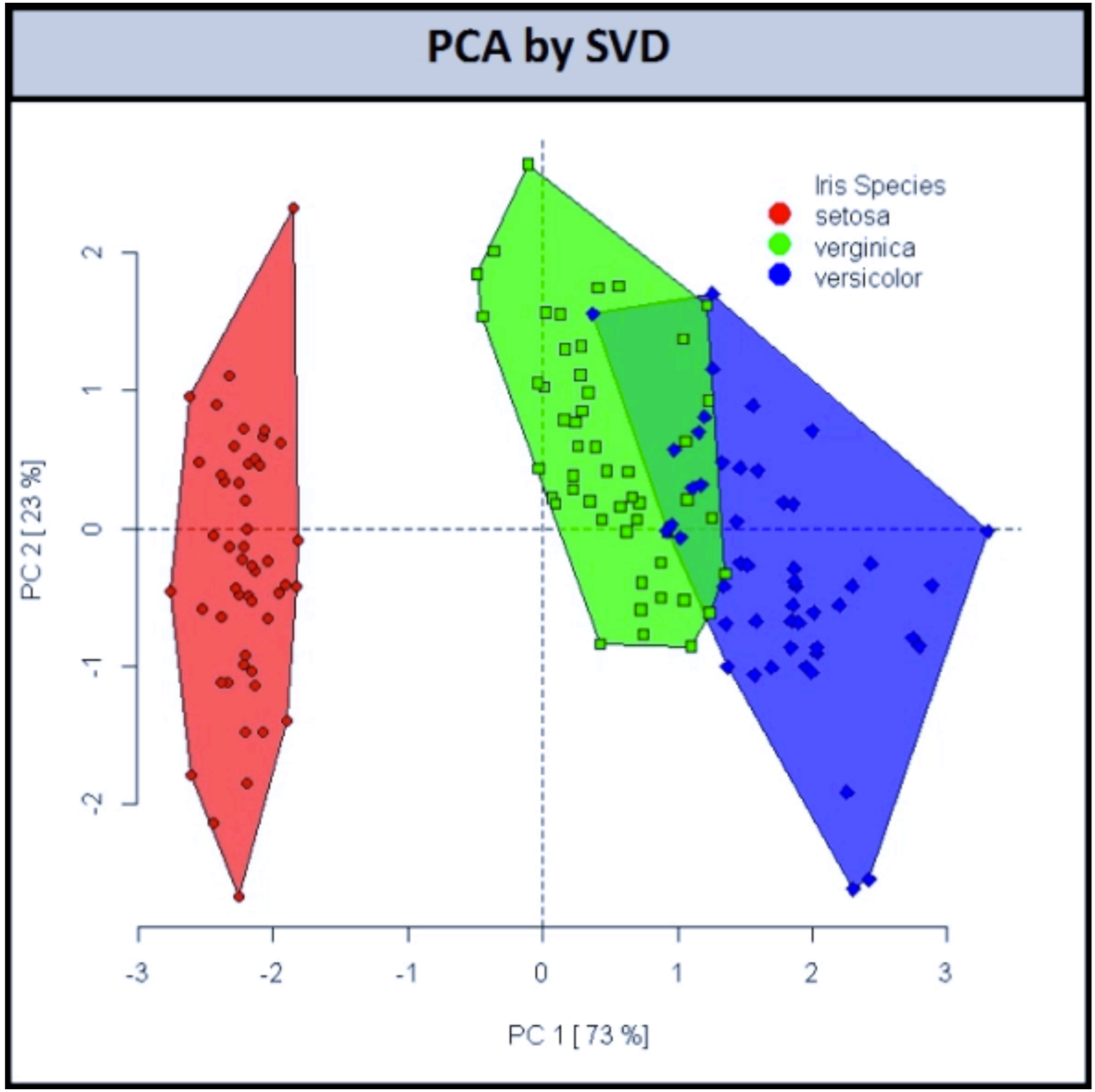
Engenharia de Features



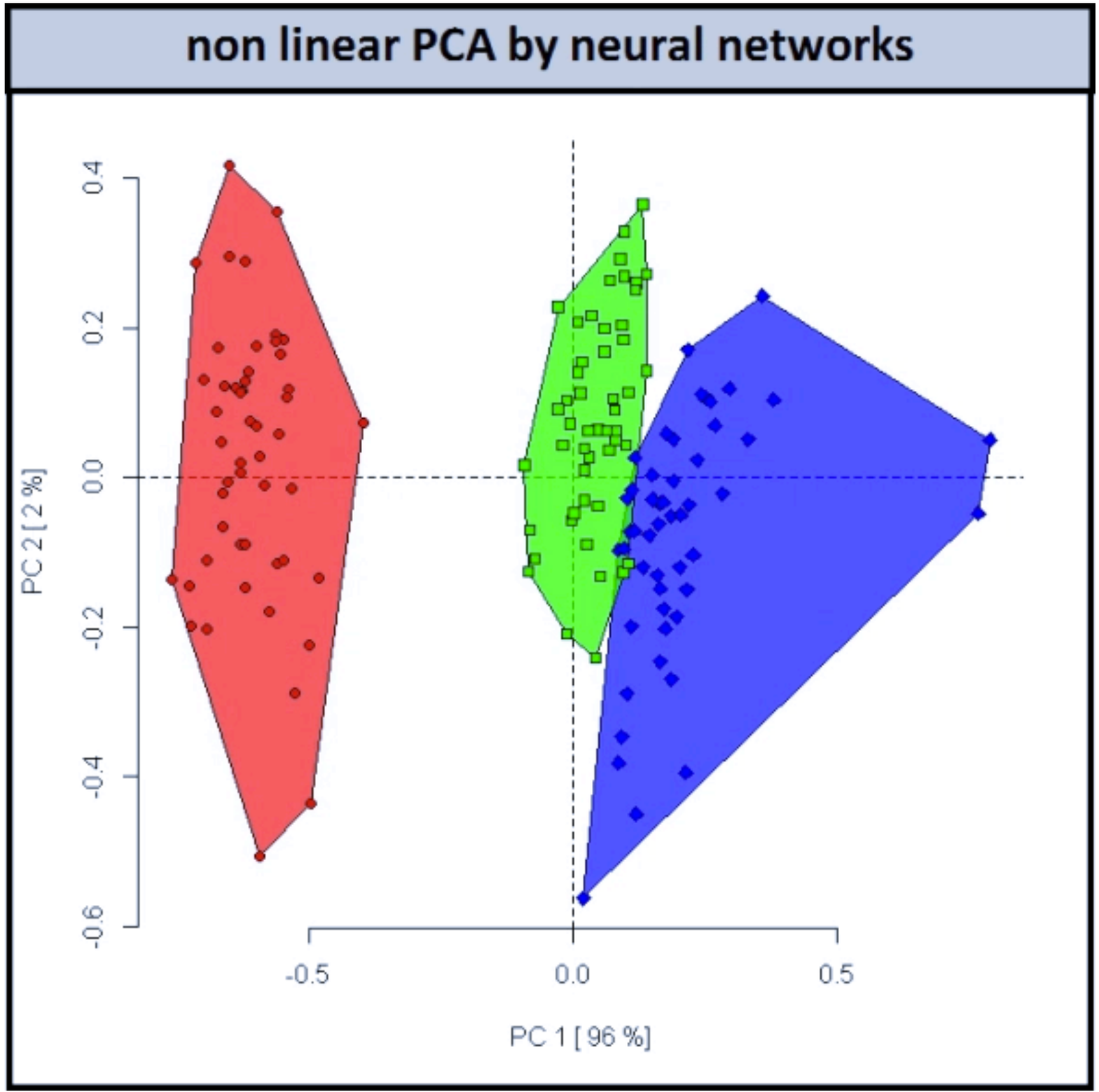
Dimensionality Reduction
&
Principal Component Analysis



Engenharia de Features



Engenharia de Features



Engenharia de Features

```
>>> import numpy as np
>>> from sklearn.decomposition import PCA
>>> X = np.array([[-1, -1], [-2, -1], [-3, -2], [1, 1], [2, 1], [3, 2]])
>>> pca = PCA(n_components=2)
>>> pca.fit(X)
PCA(copy=True, iterated_power='auto', n_components=2, random_state=None,
  svd_solver='auto', tol=0.0, whiten=False)
>>> print(pca.explained_variance_ratio_)
[ 0.99244...  0.00755...]
>>> print(pca.singular_values_)
[ 6.30061...  0.54980...]
```

```
>>> import numpy as np
>>> from sklearn.decomposition import IncrementalPCA
>>> X = np.array([[-1, -1], [-2, -1], [-3, -2], [1, 1], [2, 1], [3,
2]])
>>> ipca = IncrementalPCA(n_components=2, batch_size=3)
>>> ipca.fit(X)
IncrementalPCA(batch_size=3, copy=True, n_components=2, whiten=False)
>>> ipca.transform(X)
```

Engenharia de Features

Removing features with low variance

```
>>> from sklearn.feature_selection import VarianceThreshold
>>> X = [[0, 0, 1], [0, 1, 0], [1, 0, 0], [0, 1, 1], [0, 1, 0],
[0, 1, 1]]
>>> sel = VarianceThreshold(threshold=(.8 * (1 - .8)))
>>> sel.fit_transform(X)
array([[0, 1],
       [1, 0],
       [0, 0],
       [1, 1],
       [1, 0],
       [1, 1]])
```

Engenharia de Features

Tree-based feature selection

```
>>> from sklearn.ensemble import ExtraTreesClassifier
>>> from sklearn.datasets import load_iris
>>> from sklearn.feature_selection import SelectFromModel
>>> iris = load_iris()
>>> X, y = iris.data, iris.target
>>> X.shape
(150, 4)
>>> clf = ExtraTreesClassifier()
>>> clf = clf.fit(X, y)
>>> clf.feature_importances_
array([ 0.04...,  0.05...,  0.4...,  0.4...])
>>> model = SelectFromModel(clf, prefit=True)
>>> X_new = model.transform(X)
>>> X_new.shape
(150, 2)
```

Engenharia de Features

L1-based feature selection

```
>>> from sklearn.svm import LinearSVC
>>> from sklearn.datasets import load_iris
>>> from sklearn.feature_selection import SelectFromModel
>>> iris = load_iris()
>>> X, y = iris.data, iris.target
>>> X.shape
(150, 4)
>>> lsvc = LinearSVC(C=0.01, penalty="l1", dual=False).fit(X, y)
>>> model = SelectFromModel(lsvc, prefit=True)
>>> X_new = model.transform(X)
>>> X_new.shape
(150, 3)
```

Engenharia de Features

Univariate feature selection

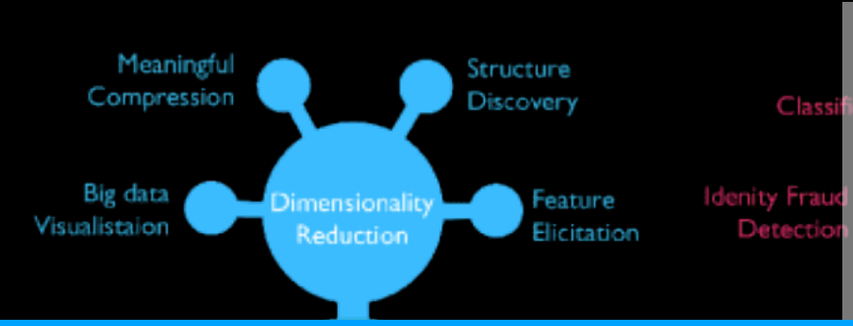
```
>>> from sklearn.datasets import load_iris
>>> from sklearn.feature_selection import SelectKBest
>>> from sklearn.feature_selection import chi2
>>> iris = load_iris()
>>> X, y = iris.data, iris.target
>>> X.shape
(150, 4)
>>> X_new = SelectKBest(chi2, k=2).fit_transform(X, y)
>>> X_new.shape
(150, 2)
```

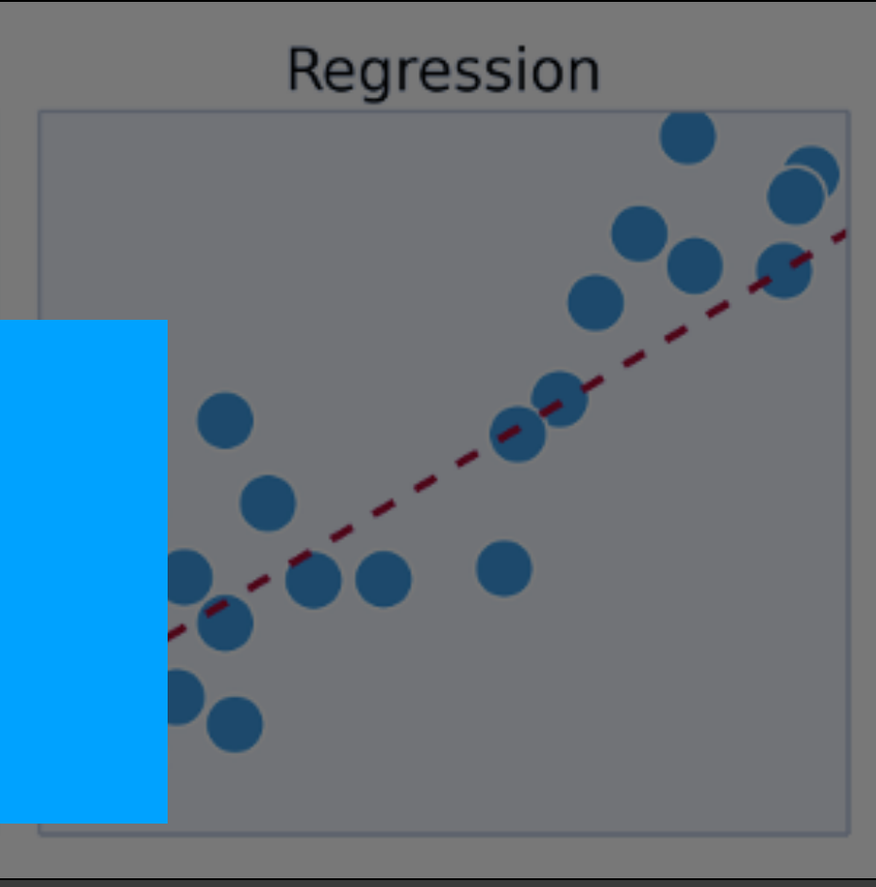
Engenharia de Features

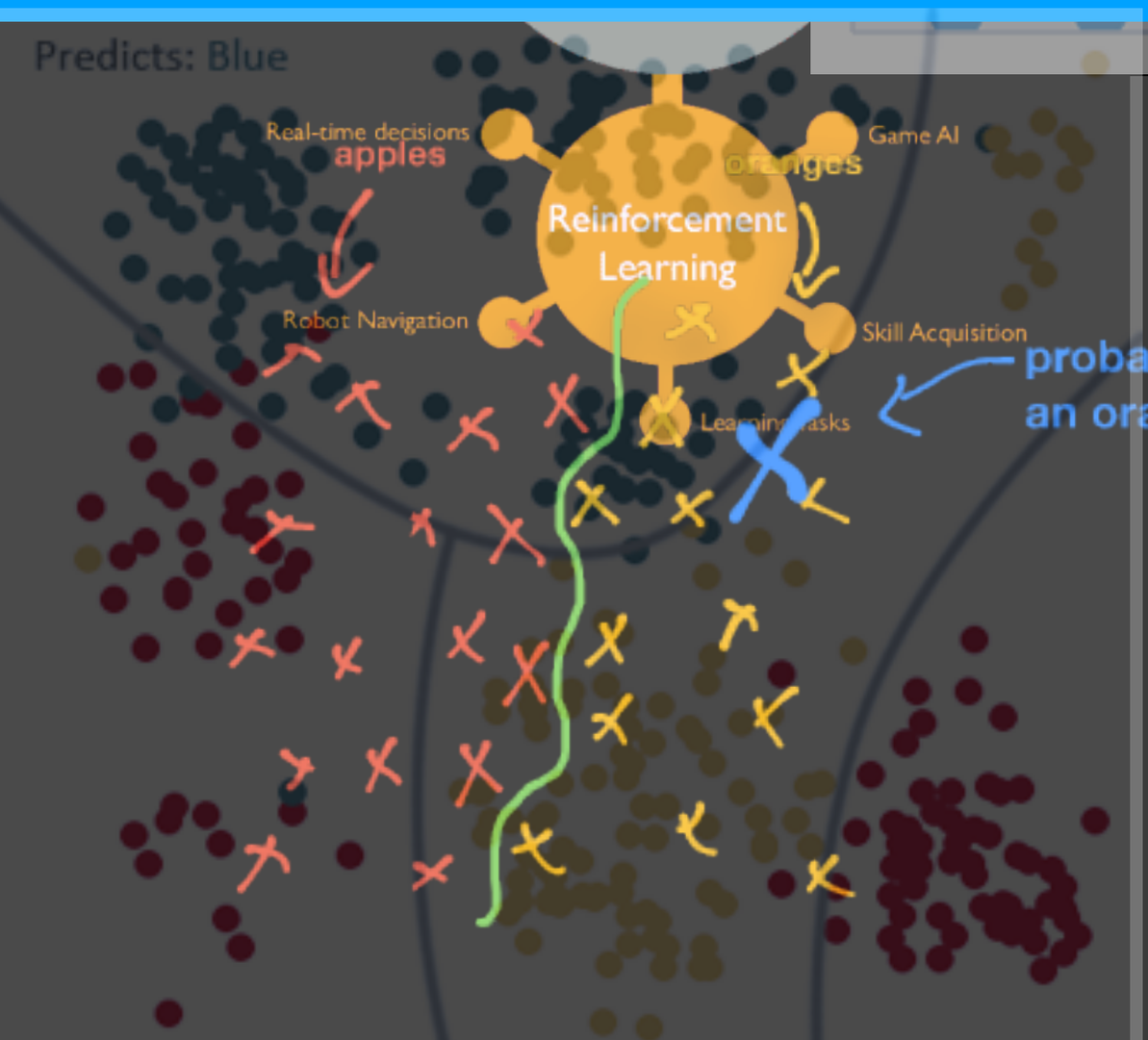
Feature selection as part of a pipeline

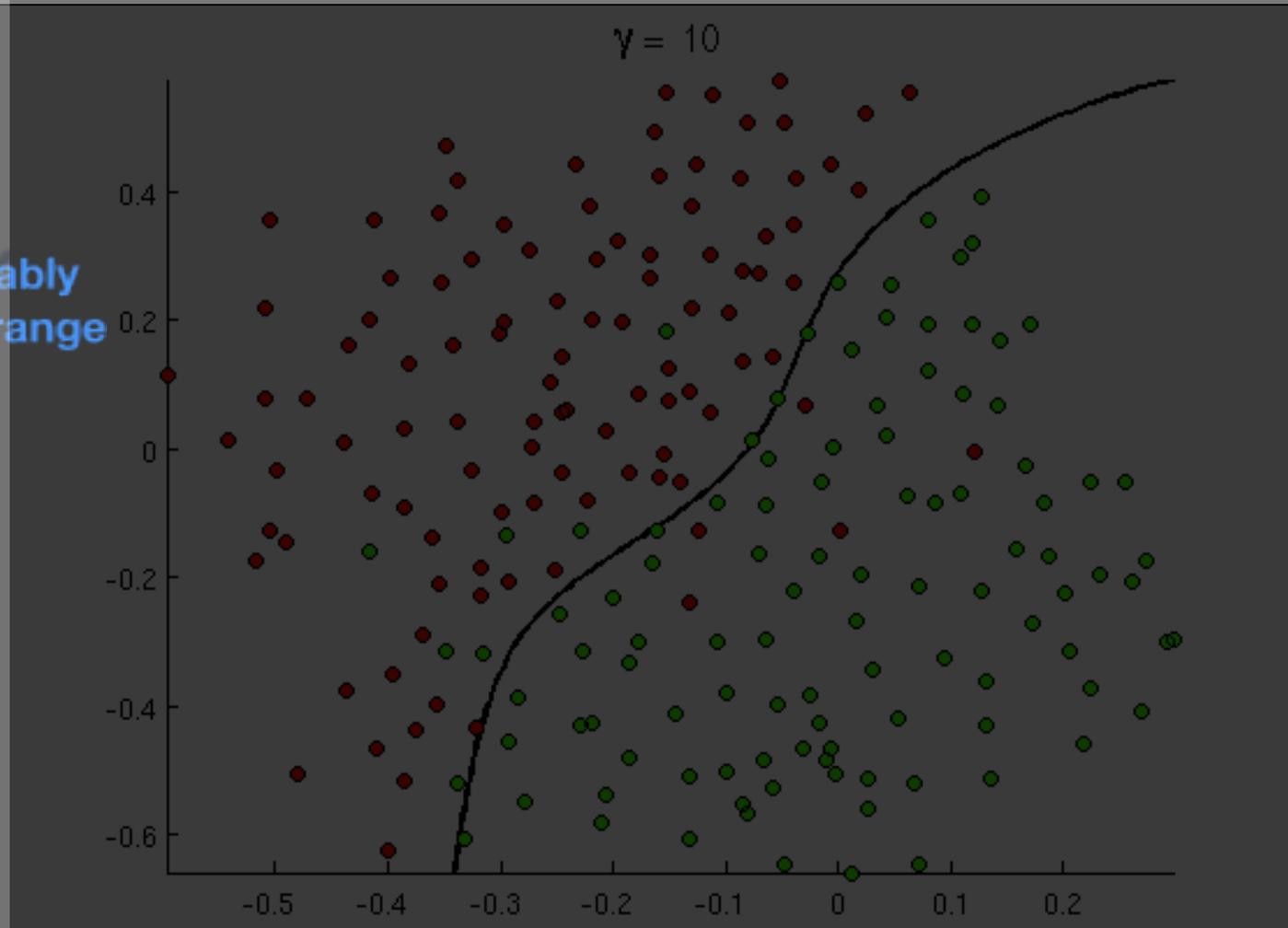
```
clf = Pipeline([
  ('feature_selection',
SelectFromModel(LinearSVC(penalty="l1"))),
  ('classification', RandomForestClassifier())
])
clf.fit(X, y)
```

Classificação









Classificação

Critérios diferentes para classificação

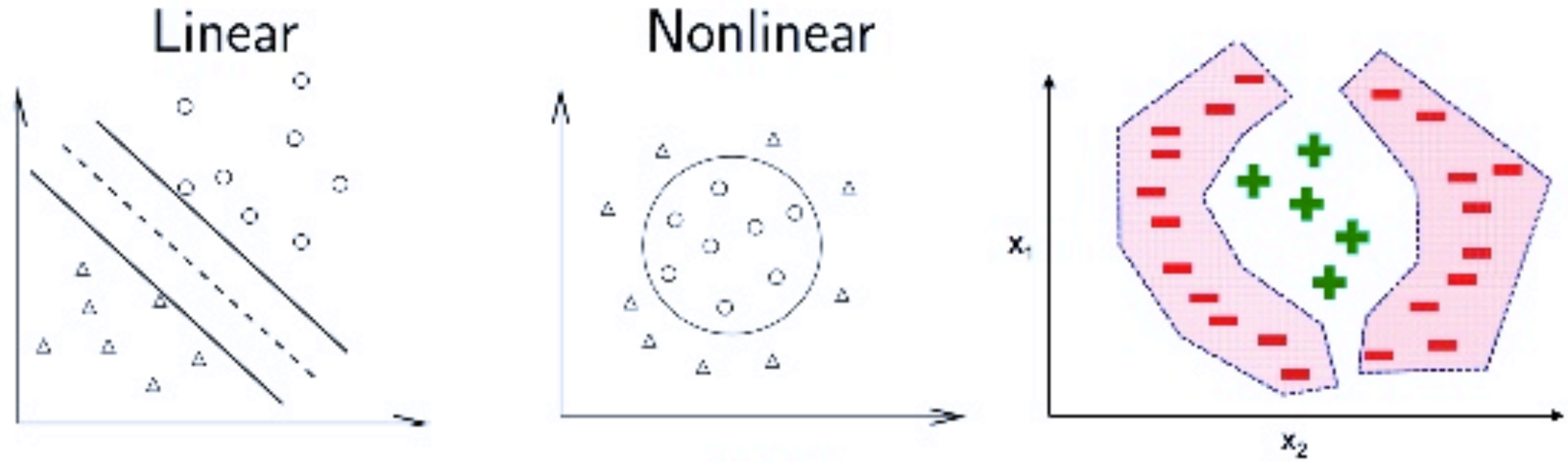
- Baseados em distâncias
  - K-NN
- Baseados em otimização
  - RNs

Geométricos

- Baseados em probabilidade
  - NB
- Baseados em procura (lógicos)
  - Indução de ADs

Classificação

Linear / Nonlinear Classification



- Linear : In the data's original input space, labels can be classified by a linear decision boundary.

- Nonlinear : The classifiers have nonlinear, and possibly discontinuous decision boundaries.

Classificação

Linear Classifier Examples

- Logistic Regression
- Support Vector Machine
- Naive Bayes Classifier
- Linear Discriminant Analysis

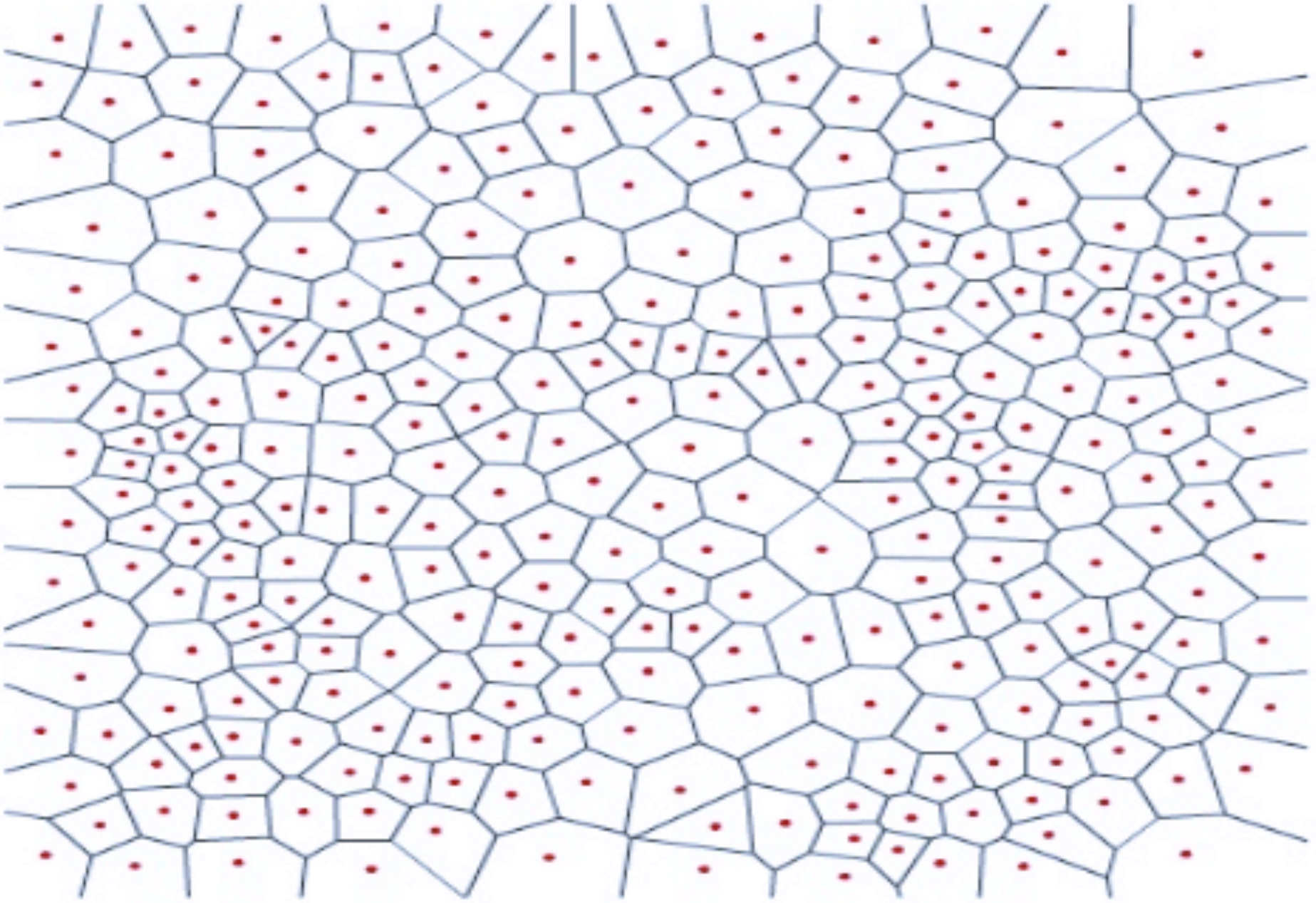
Classificação

Nonlinear Classifier Examples

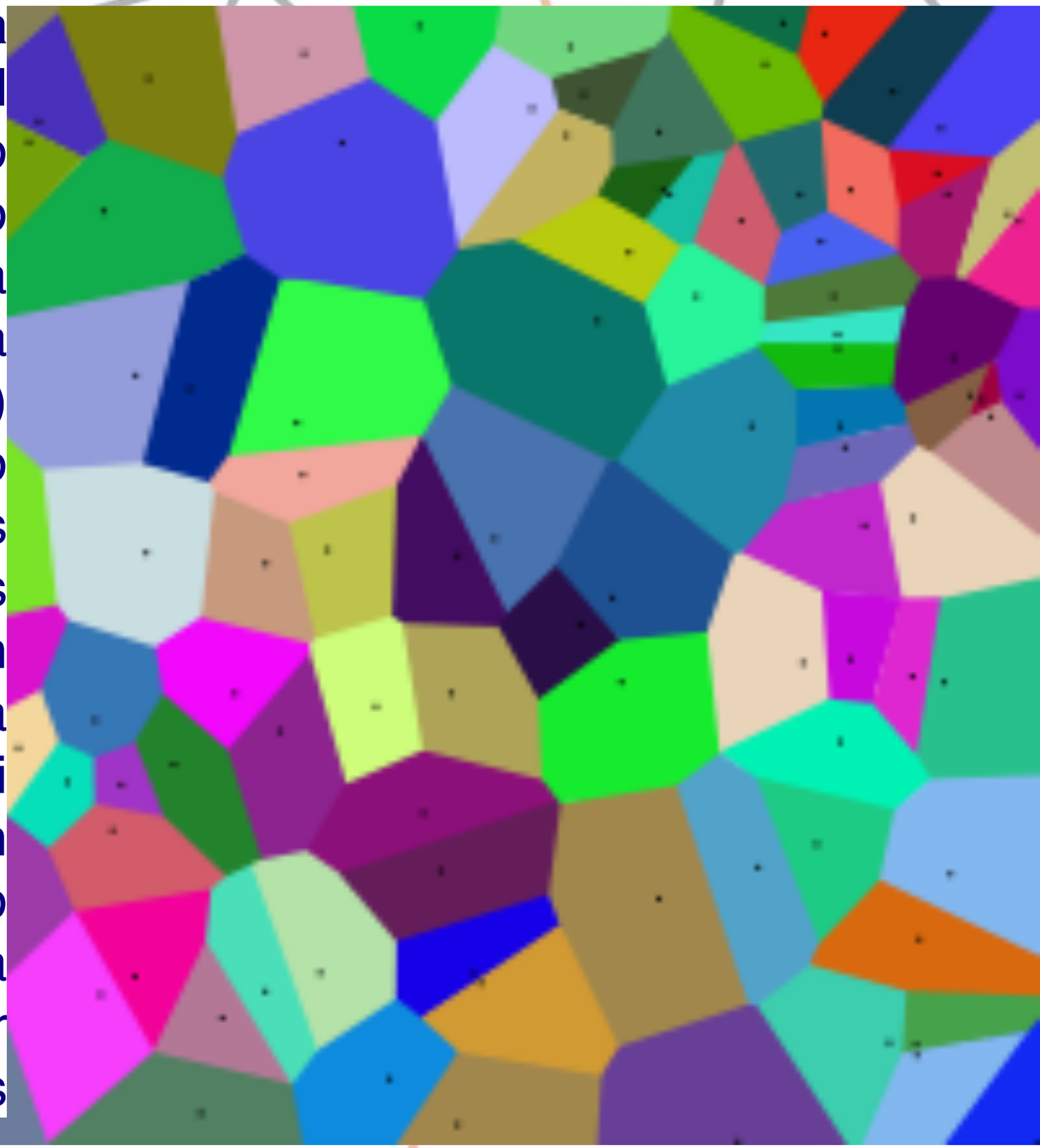
- Kernel Support Vector Machine
- Multi-Layer Neural Networks
- Decision Tree / Random Forest
- Gradient Boosted Decision Trees
- K-nearest Neighbors Algorithm

Classificação Baseada em Instâncias

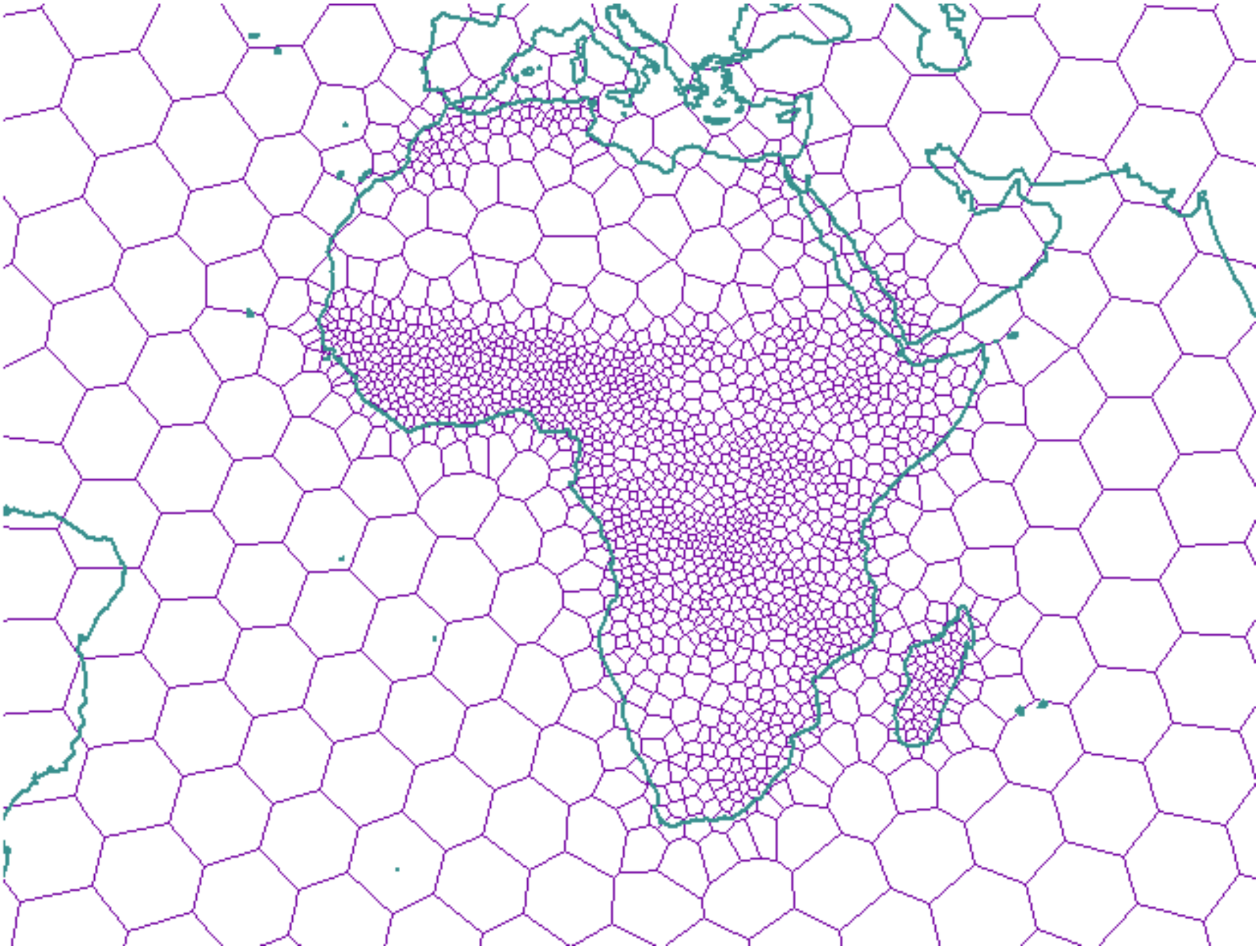
Matematica arquitectura y urbanism
Diagrama de Voronoi



Na matemática, um **Diagrama de Voronoi** é um tipo especial de decomposição de um dado espaço, por exemplo, um espaço métrico, determinado pela distância para uma determinada família de objetos (subconjuntos) no espaço. Estes objetos são normalmente chamados de sítios ou geradores (apesar de nomes como “sementes” estarem também em uso). Cada sítio está associado a célula de Voronoi correspondente, isto é um conjunto de todos os pontos no dado espaço o qual a distância para o dado sítio não é maior que sua distância para os outros objetos.



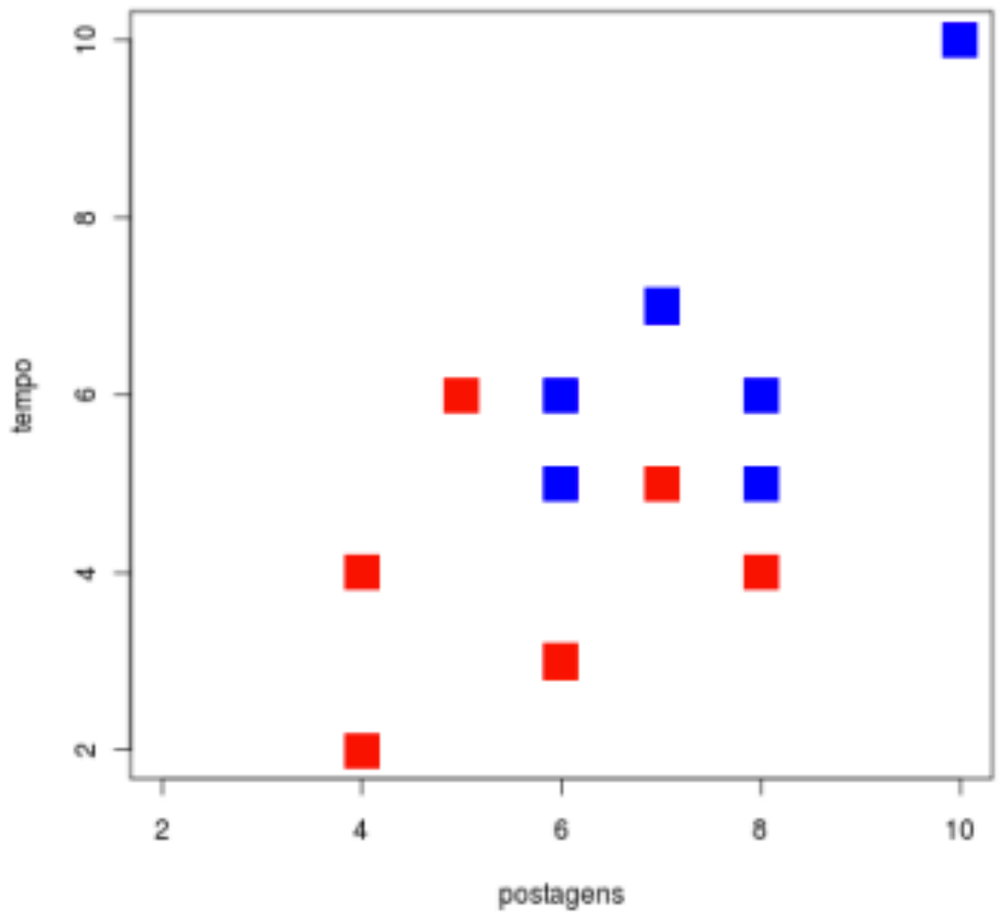
Esparsa - Polígonos Grandes



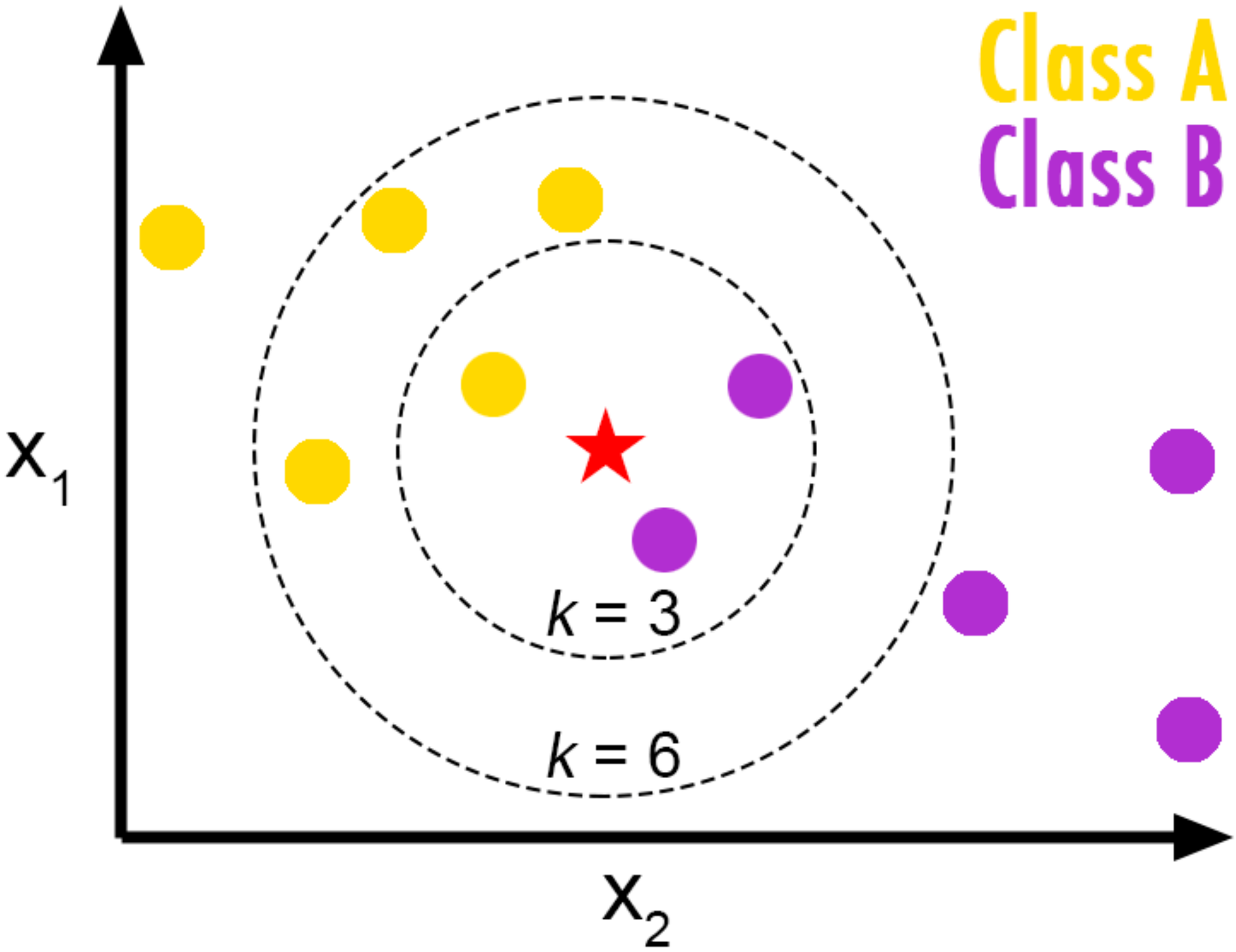
KNN

K-Vizinhos Mais Próximos

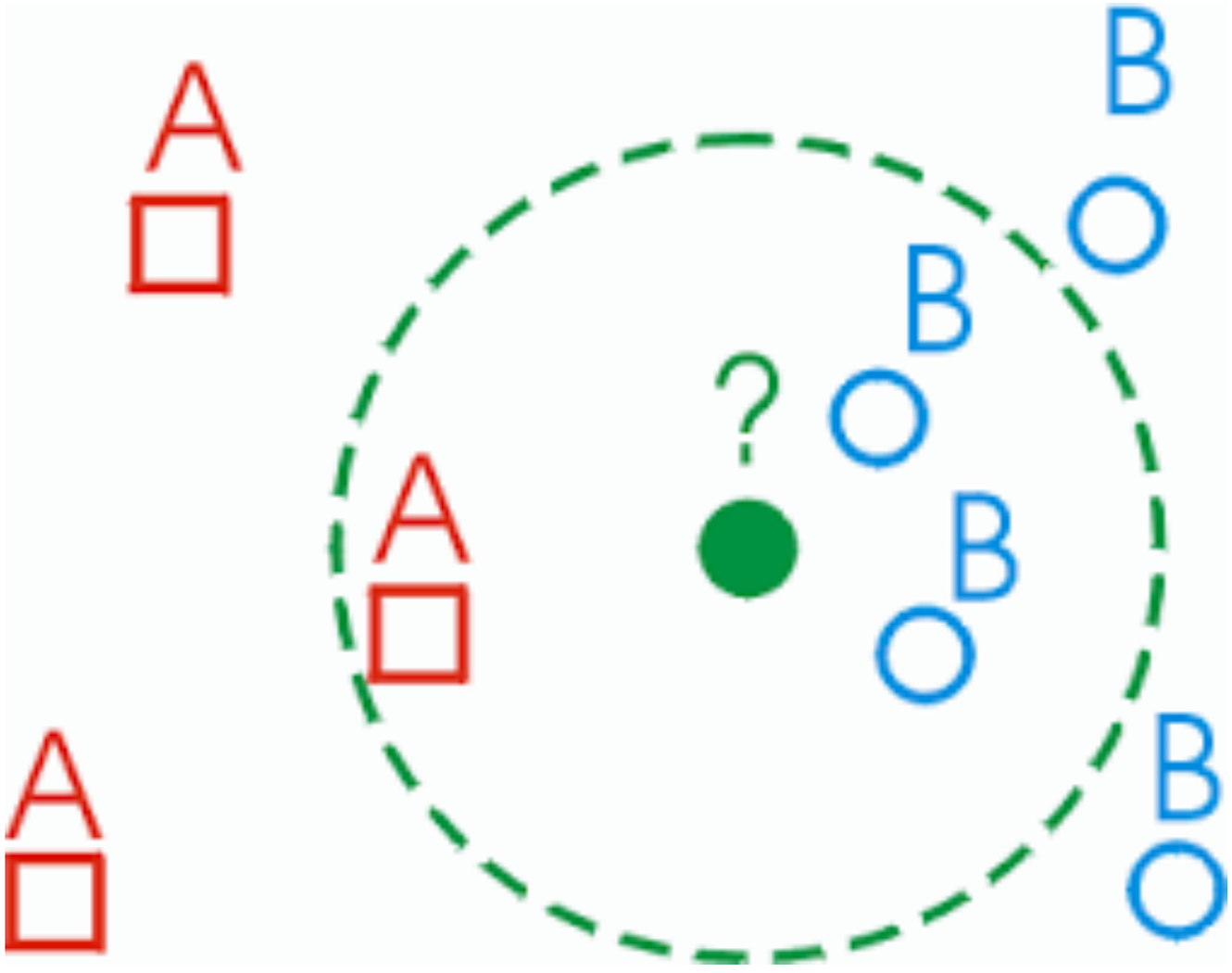
Dados dos alunos dispostos graficamente (em azul, os alunos que passaram, em vermelho, os alunos que não passaram).



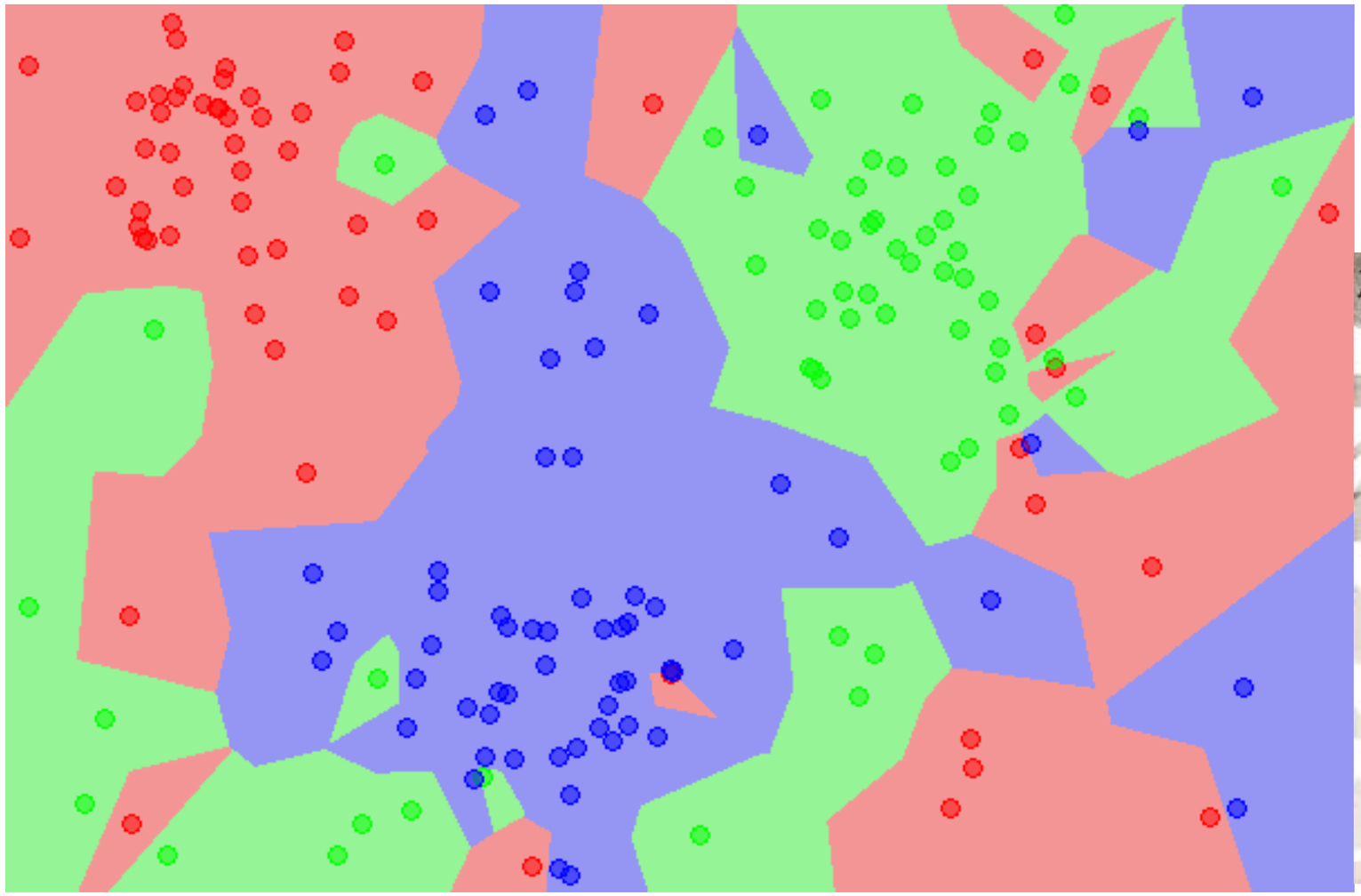
KNN



KNN

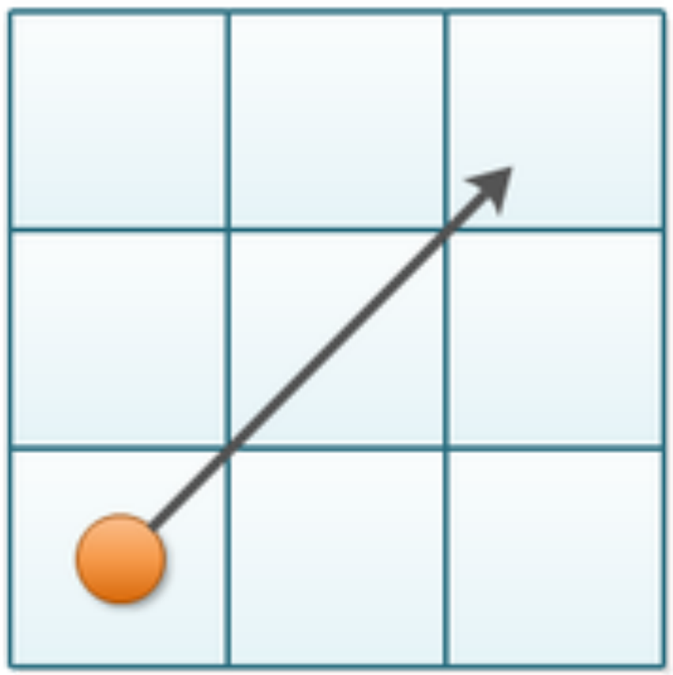


KNN



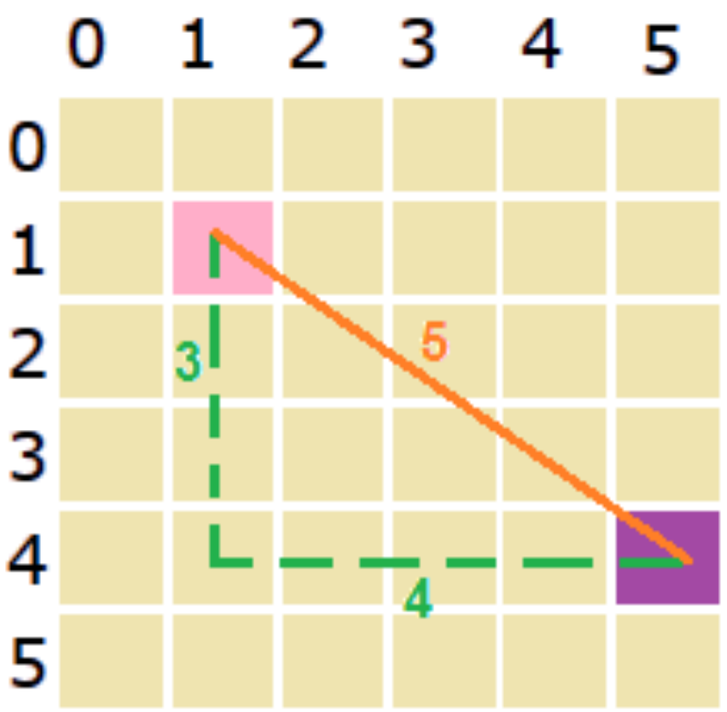
KNN

Euclidean Distance



$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

KNN



Point P1 = (1,1)

Point P2 = (5,4)

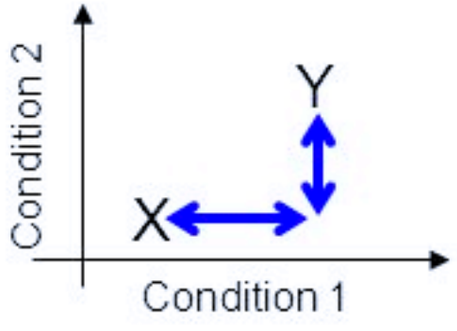
Euclidean distance = $\sqrt{(5-1)^2 + (4-1)^2} = 5$

Manhattan distance = $|5-1| + |4-1| = 7$

KNN

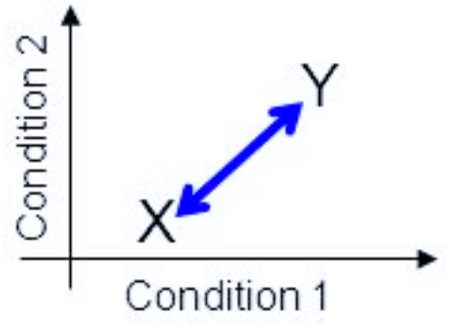
- City Block (Manhattan) distance:
  - Sum of differences across dimensions
  - Less sensitive to outliers
  - Diamond shaped clusters

$$d(X,Y) = \sum_i |x_i - y_i|$$



- Euclidean distance:
  - Most commonly used distance
  - Sphere shaped cluster
  - Corresponds to the geometric distance into the multidimensional space

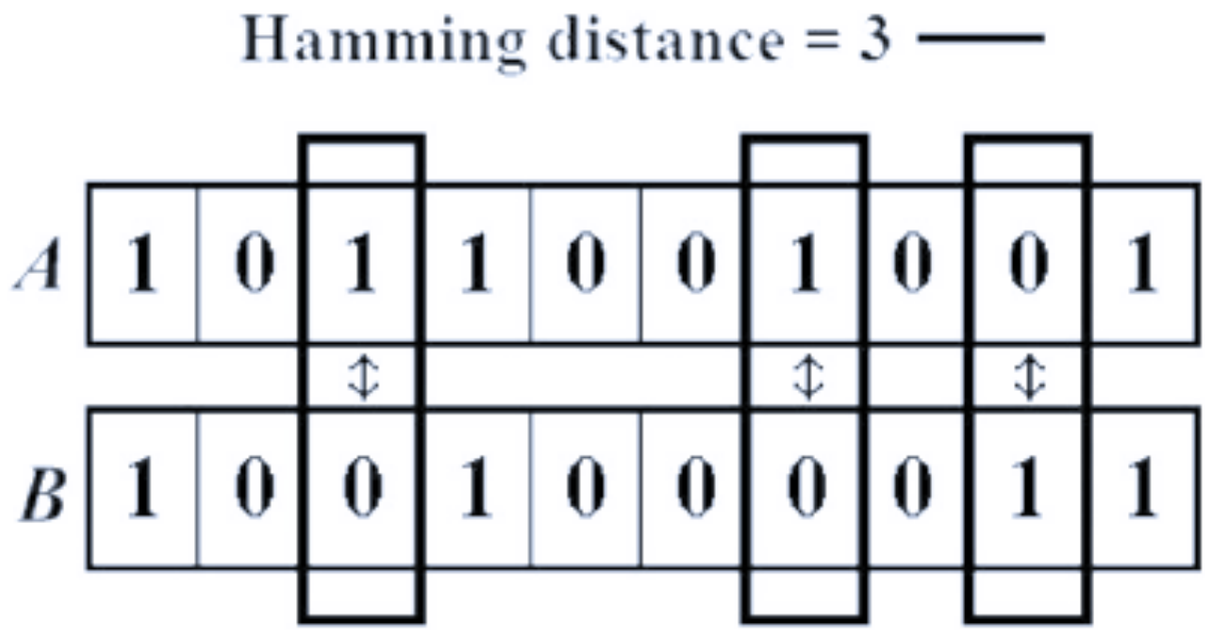
$$d(X,Y) = \sqrt{\sum_i (x_i - y_i)^2}$$



where gene $X = (x_1,\ldots,x_n)$ and gene $Y=(y_1,\ldots,y_n)$

KNN

Distância de Hamming para valores Categóricos



ATCGGTAGT

ATGGTTCCT

The hamming distance between these two sequences is **4**.

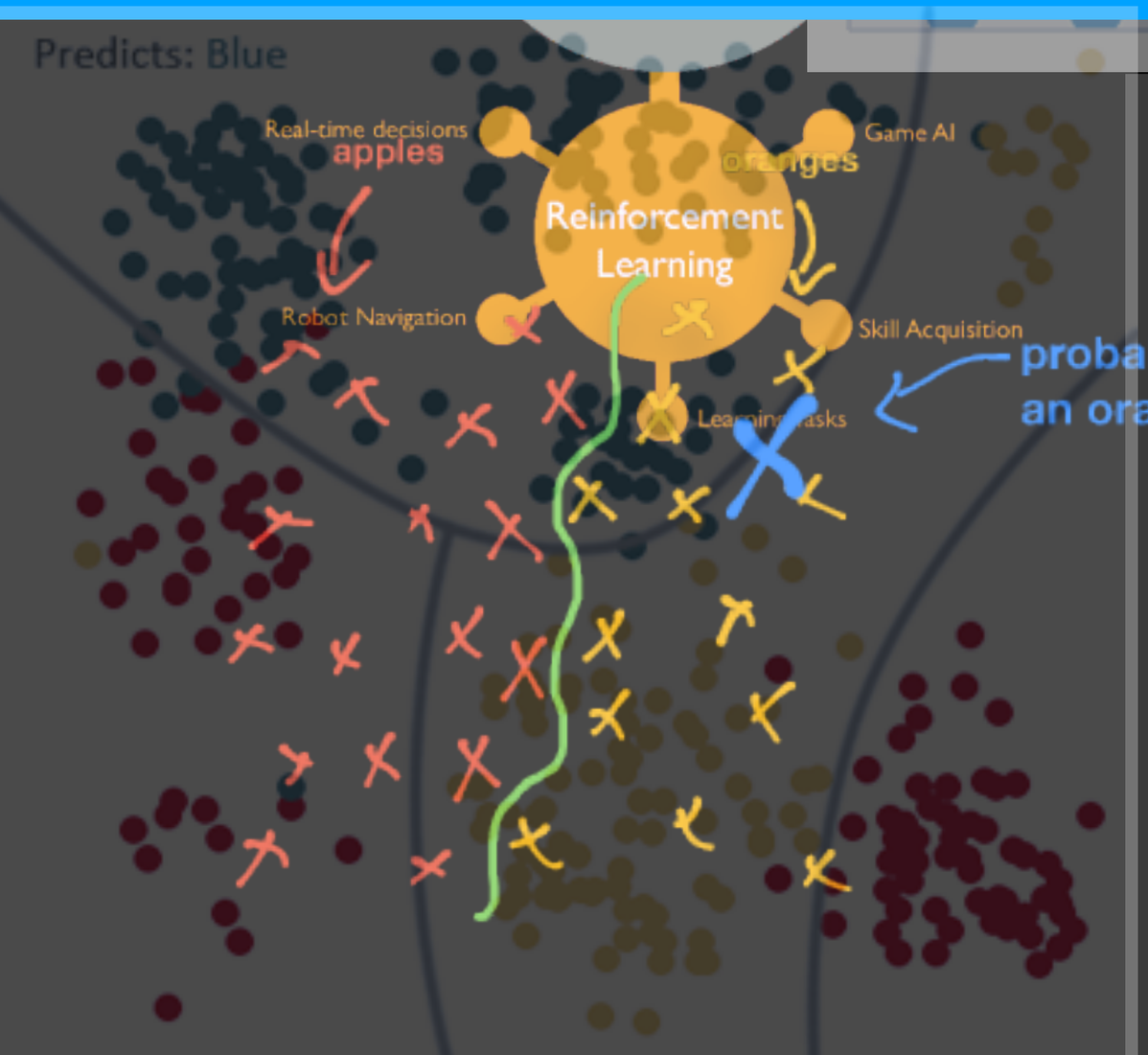
KNN

Euclidean

Manhattan

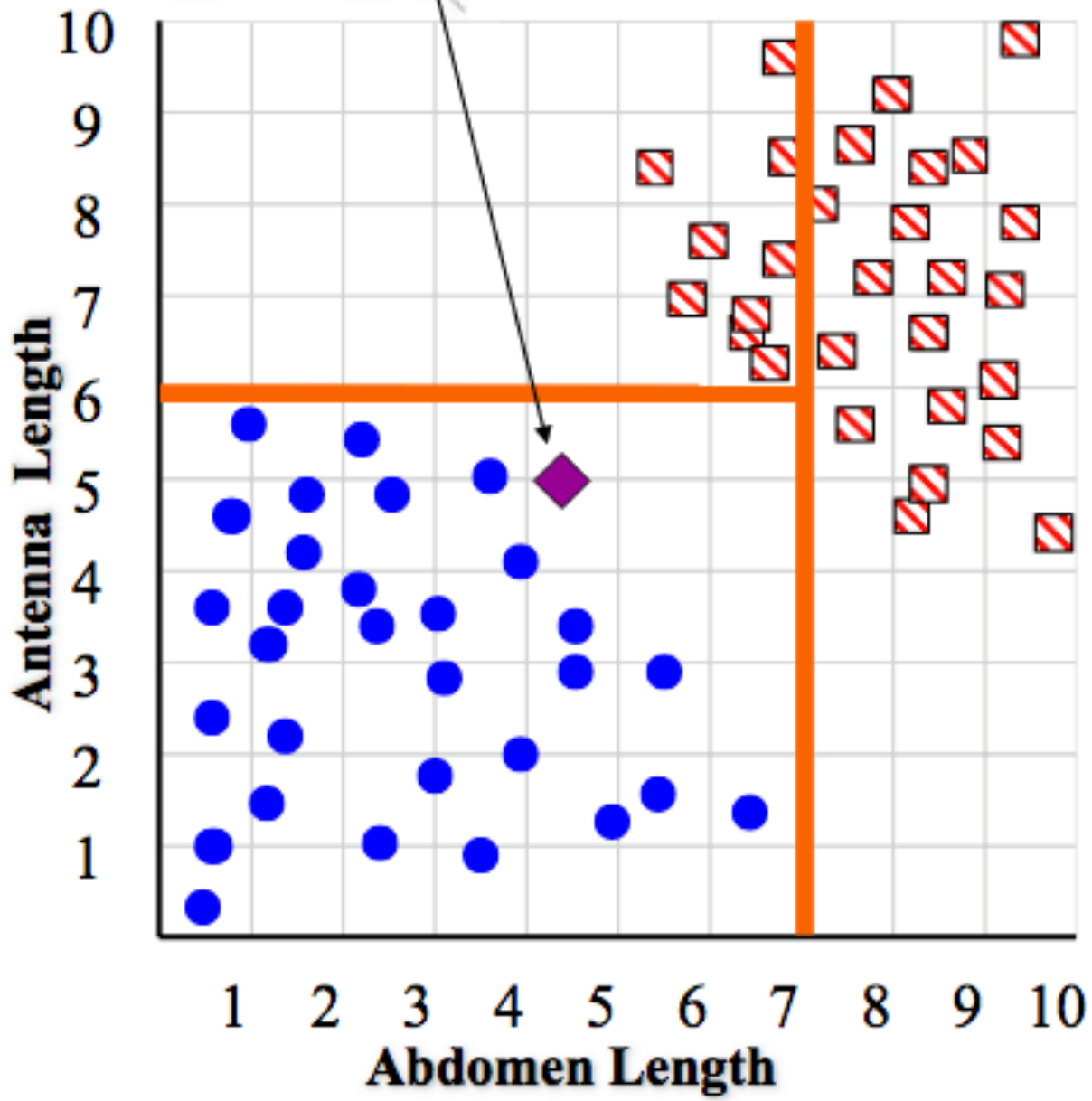
Dúvidas: naubergois@gmail.com

Árvores de Decisão

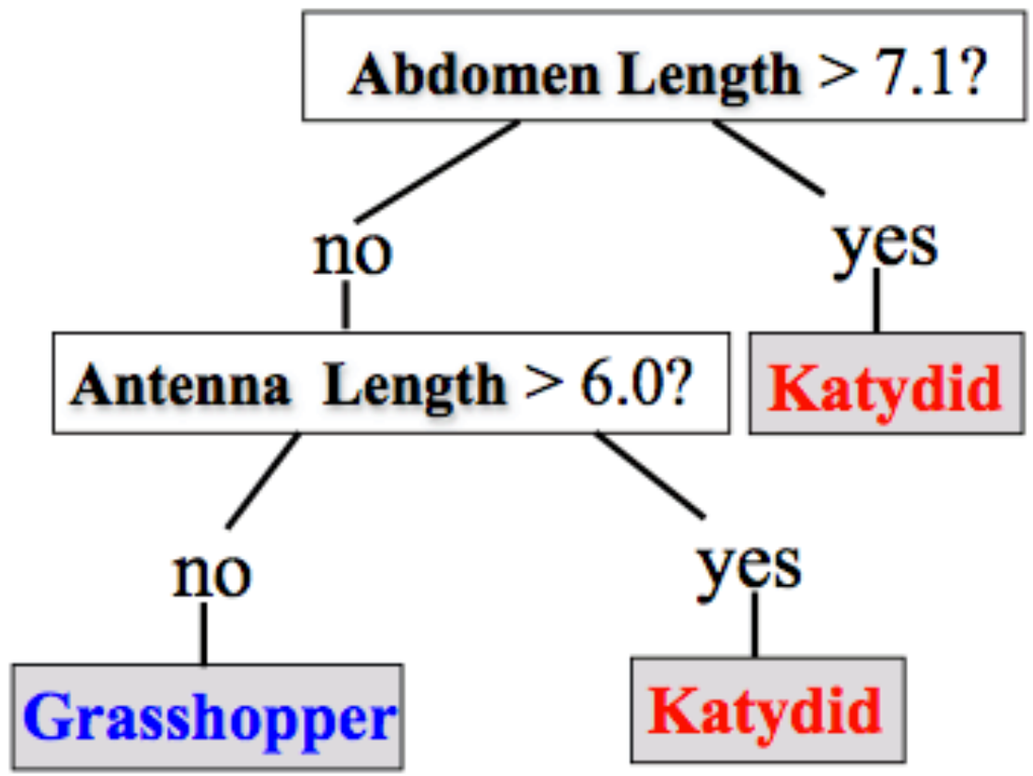


Árvores de Decisão

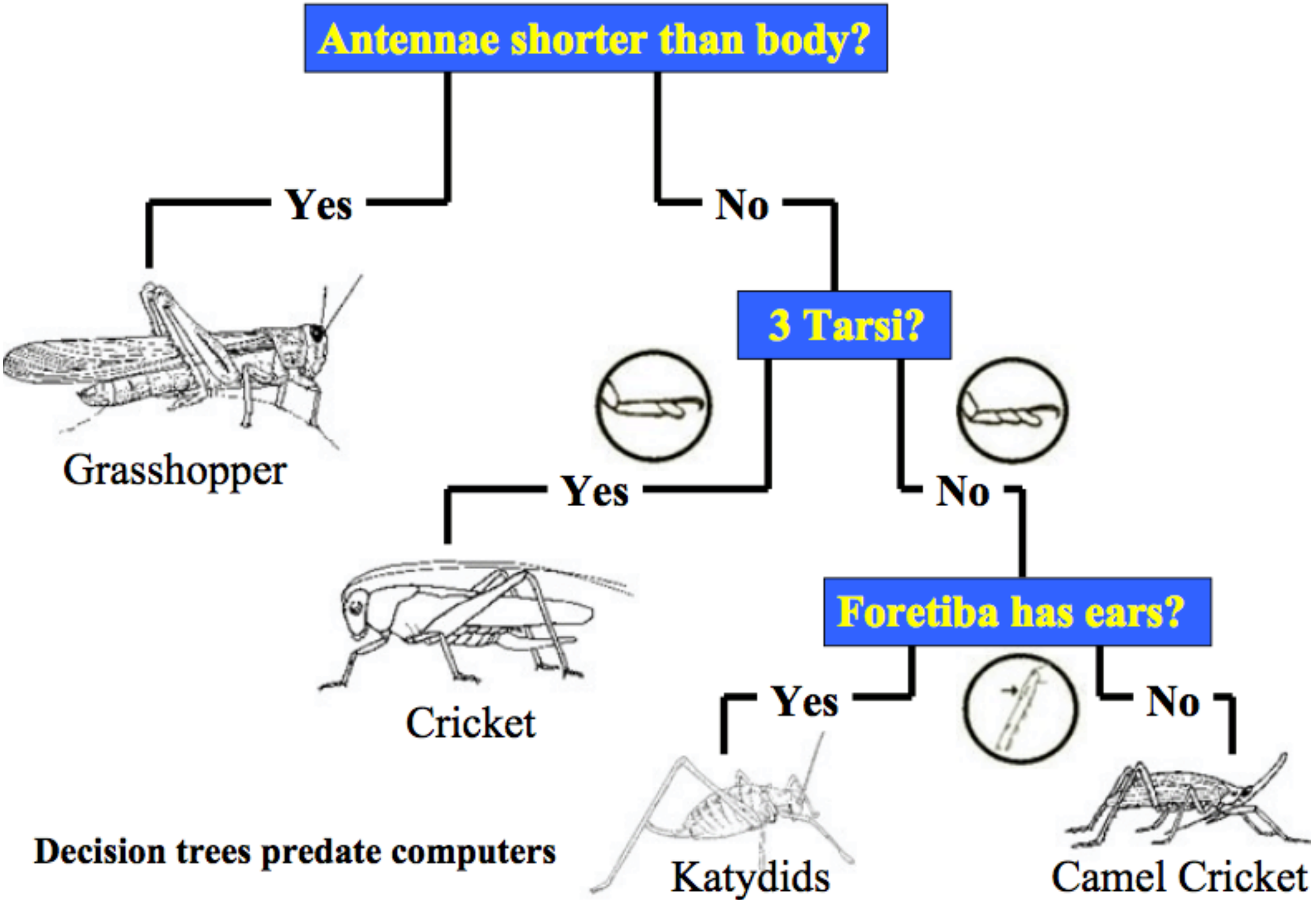
Decision Tree Classifier



Ross Quinlan



Árvores de Decisão



Árvores de Decisão

Decision Tree Classification

- Decision tree
  - A flow-chart-like tree structure
  - Internal node denotes a test on an attribute
  - Branch represents an outcome of the test
  - Leaf nodes represent class labels or class distribution
- Decision tree generation consists of two phases
  - Tree construction
    - At start, all the training examples are at the root
    - Partition examples recursively based on selected attributes
  - Tree pruning
    - Identify and remove branches that reflect noise or outliers
- Use of decision tree: Classifying an unknown sample
  - Test the attribute values of the sample against the decision tree

E. Keogh, UC Riverside

Árvores de Decisão

Split on Gender

Students =30
Play Cricket = 15 (50%)

Female

Students =10
Play Cricket = 2 (20%)

Male

Students = 20
Play Cricket = 13 (65%)

Árvores de Decisão

Split on Height

< 5.5 ft

Students = 12
Play Cricket = 5 (42%)

>= 5.5 ft

Students = 18
Play Cricket = 10 (56%)

Árvores de Decisão

Split on Class

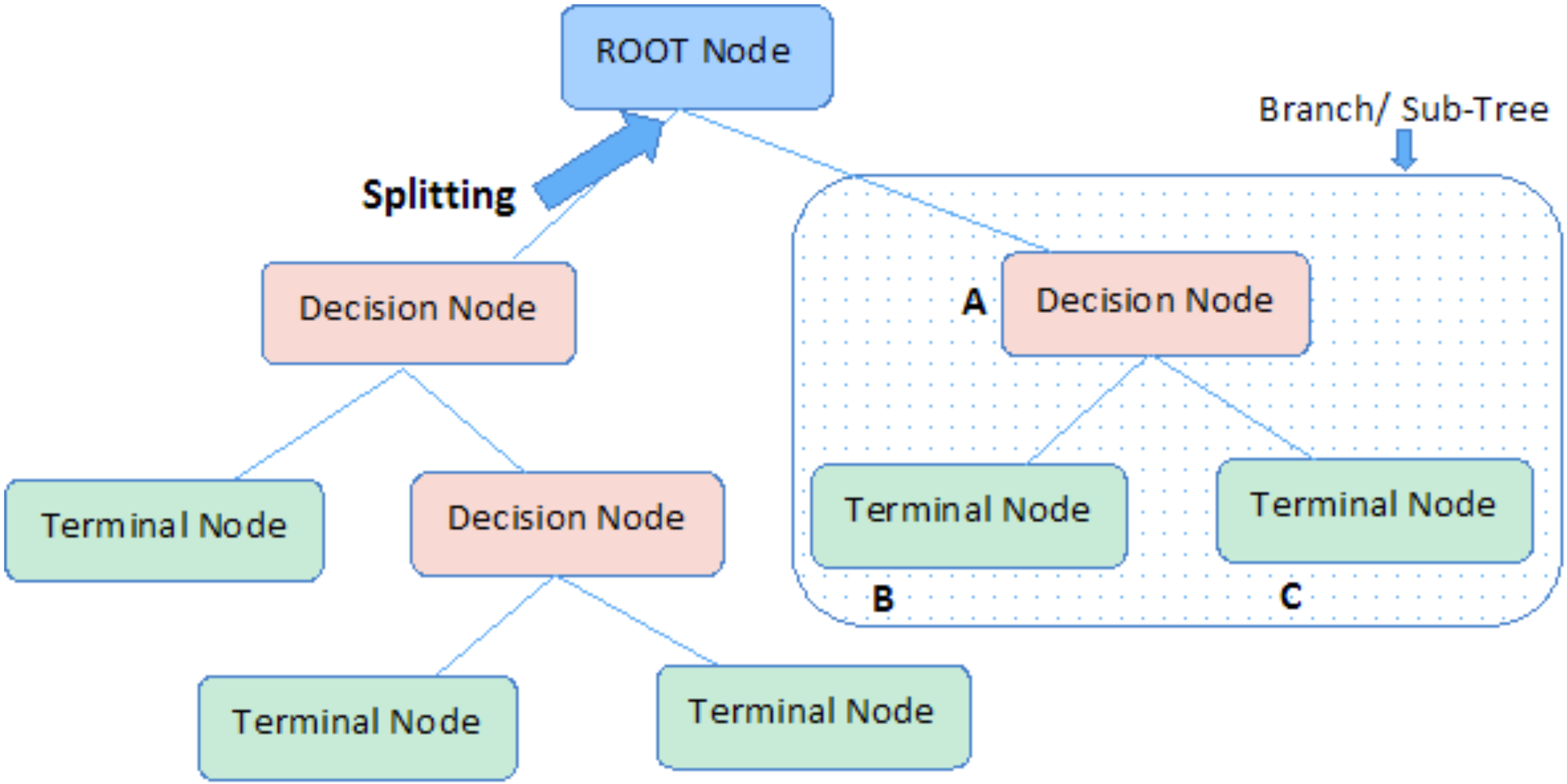
Class IX

Students = 14
Play Cricket = 6 (43%)

Class X

Students = 16
Play Cricket = 9 (56%)

Árvores de Decisão



Note:- A is parent node of B and C.

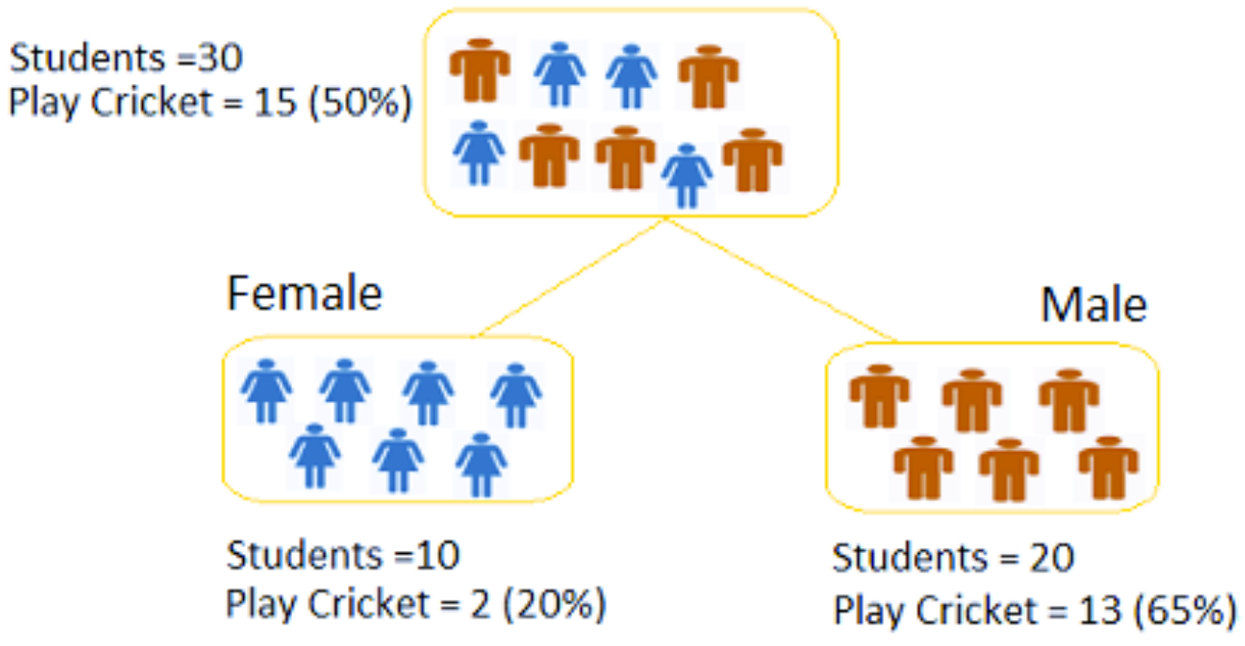
Árvores de Decisão

Gini Index

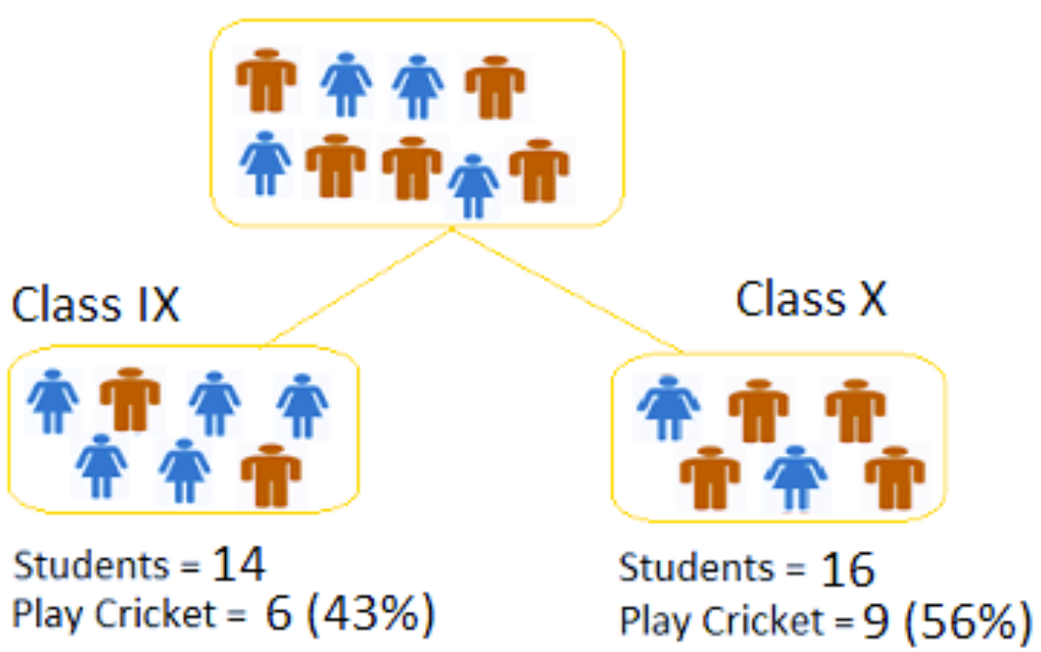
Gini index says, if we select two items from a population at random then they must be of same class and probability for this is 1 if population is pure.

- **It works with categorical target variable "Success" or "Failure".**
- **It performs only Binary splits**
- **Higher the value of Gini higher the homogeneity.**
- **CART (Classification and Regression Tree) uses Gini method to create binary splits.**

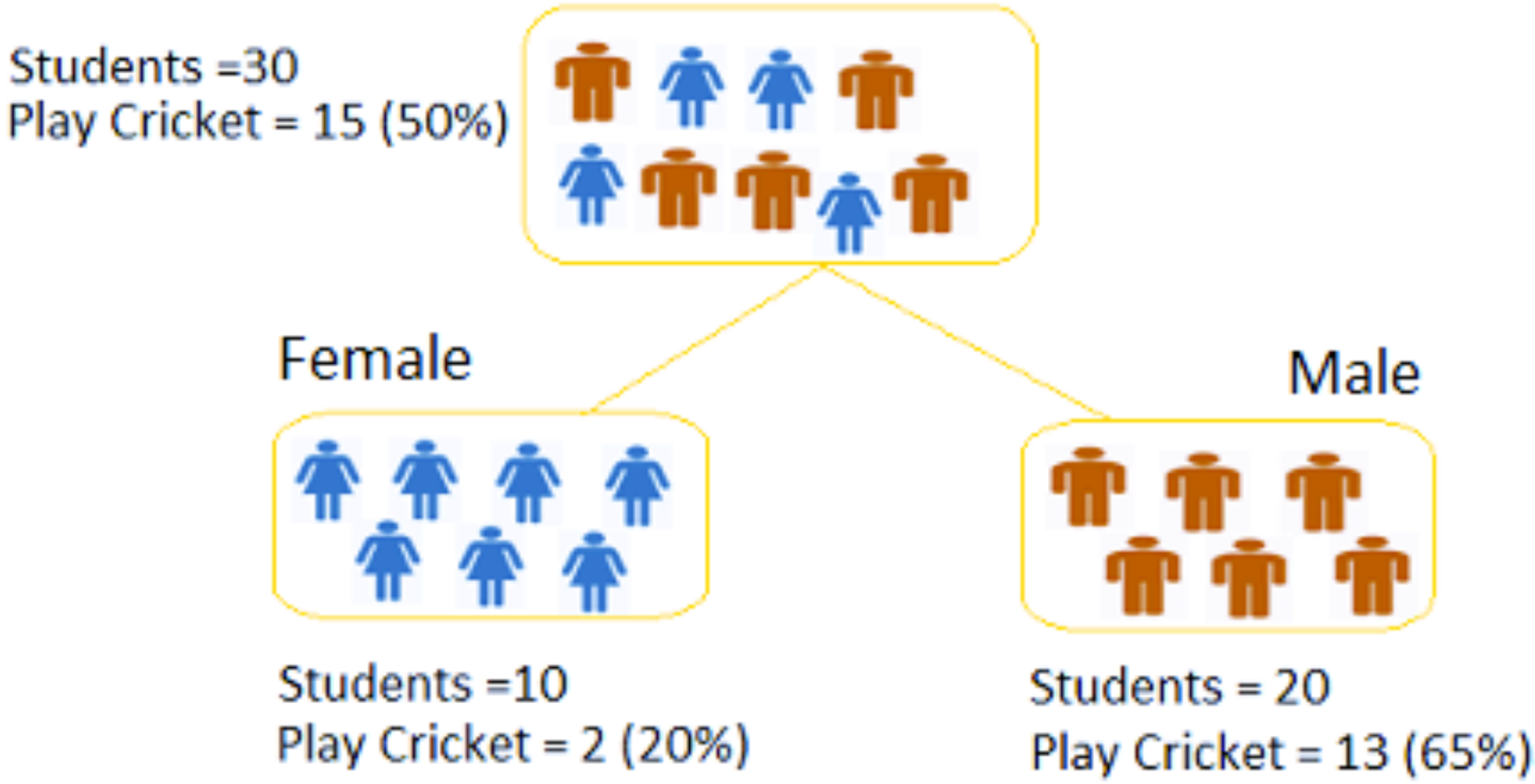
Split on Gender



Split on Class

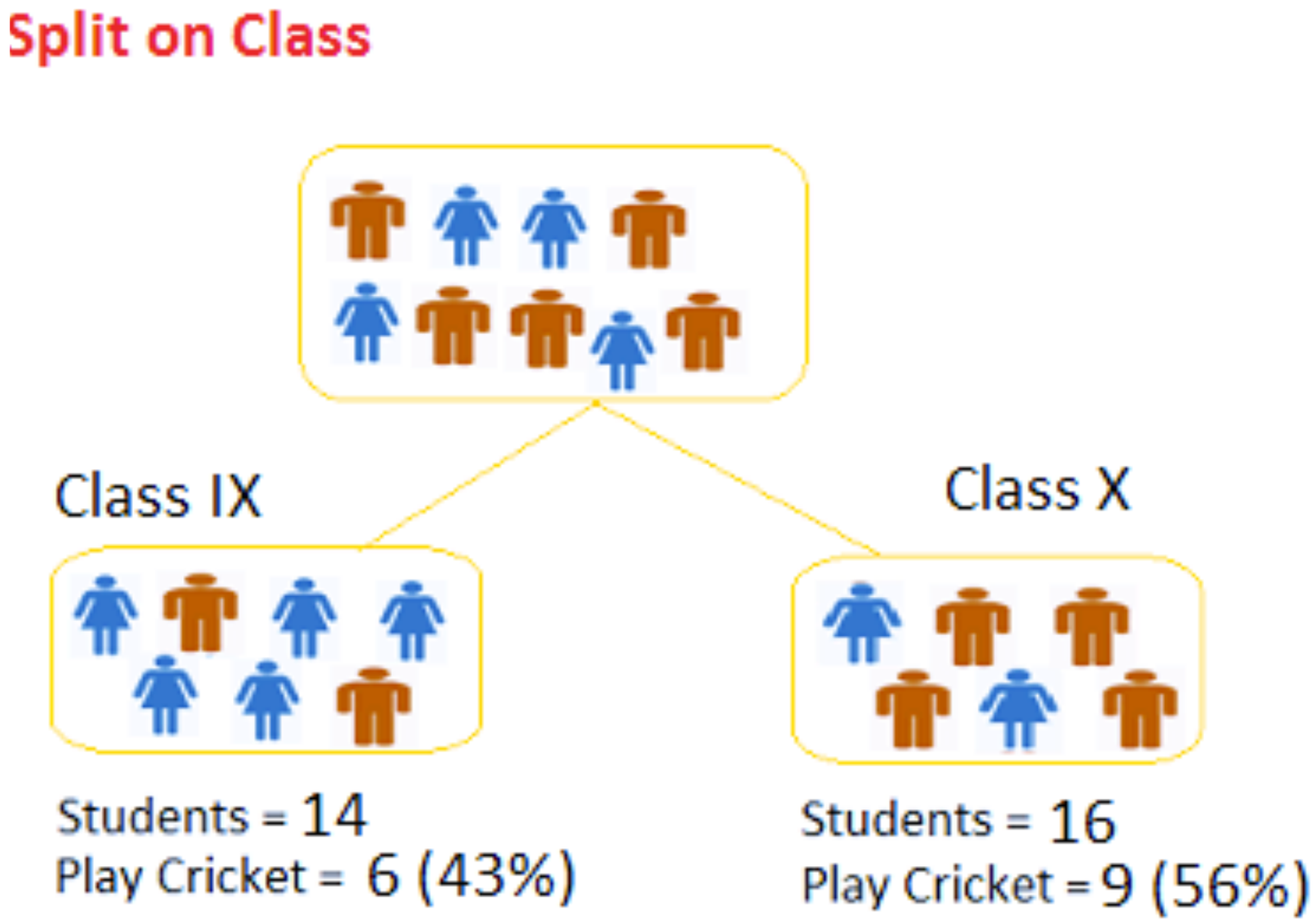


Árvores de Decisão



- **Calculate Gini for sub-nodes, using formula sum of square of probability for success and failure (p^2+q^2).**
  - **Calculate, Gini for sub-node Female = (0.2)*(0.2)+(0.8)*(0.8)=0.68**
  - **Gini for sub-node Male = (0.65)*(0.65)+(0.35)*(0.35)=0.55**
  - **Calculate weighted Gini for Split Gender = (10/30)*0.68+(20/30)*0.55 = 0.59**

Árvores de Decisão



. Gini for sub-node Class IX = (0.43)*(0.43)+(0.57)*(0.57)=0.51
. Gini for sub-node Class X = (0.56)*(0.56)+(0.44)*(0.44)=0.51
. Calculate weighted Gini for Split Class = (14/30)*0.51+(16/30)*0.51 = 0.51

Above, you can see that Gini score for *Split on Gender* is higher than *Split on Class*, hence, the node split will take place on Gender.

Árvores de Decisão

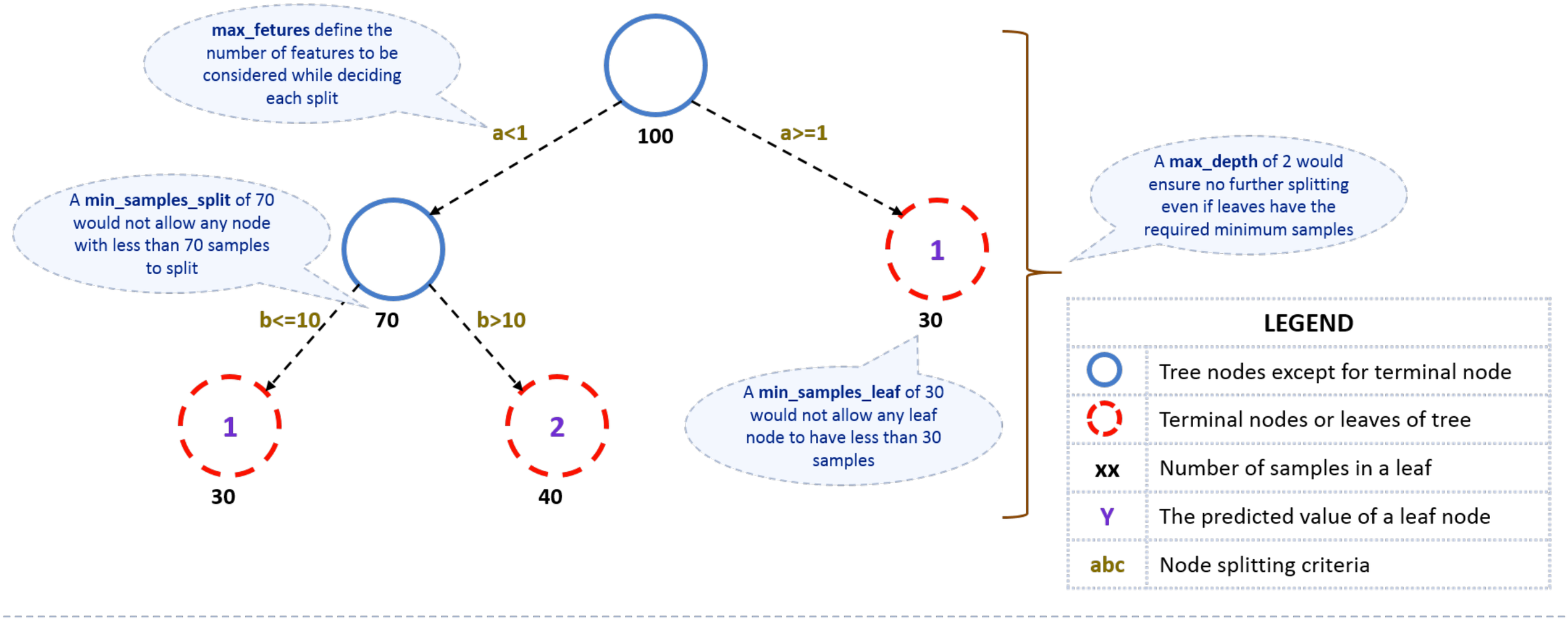
$$\text{Entropy} = -p \log_2 p - q \log_2 q$$

- Entropy for parent node = -(15/30) log2 (15/30) – (15/30) log2 (15/30) = **1**. Here 1 shows that it is a impure node.
- Entropy for Female node = -(2/10) log2 (2/10) – (8/10) log2 (8/10) = 0.72 and for male node, -(13/20) log2 (13/20) – (7/20) log2 (7/20) = **0.93**
- Entropy for split Gender = Weighted entropy of sub-nodes = (10/30)*0.72 + (20/30)*0.93 = **0.86**
- Entropy for Class IX node, -(6/14) log2 (6/14) – (8/14) log2 (8/14) = 0.99 and for Class X node, -(9/16) log2 (9/16) – (7/16) log2 (7/16) = 0.99.
- Entropy for split Class = (14/30)*0.99 + (16/30)*0.99 = **0.99**

Above, you can see that entropy for *Split on Gender* is the lowest among all, so the tree will split on *Gender*.

1- Entropy.

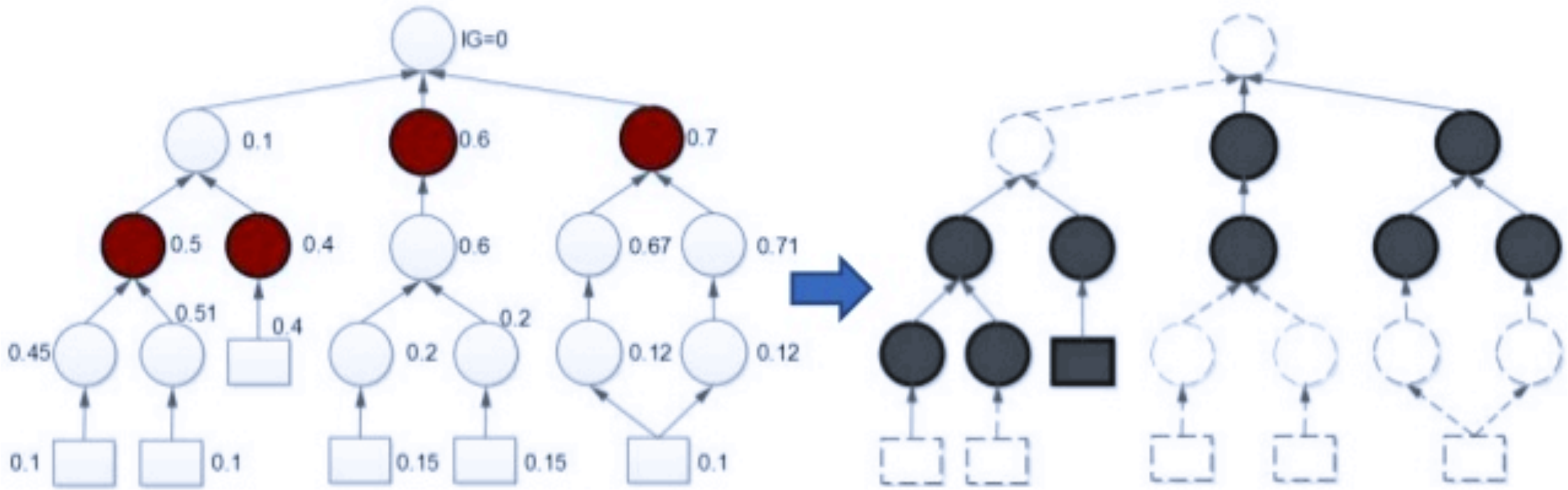
Árvores de Decisão



Árvores de Decisão

Árvores de Decisão

Standard Feature Selection: Information Gain



```python
# Import the necessary modules and libraries
import numpy as np
from sklearn.tree import DecisionTreeRegressor
import matplotlib.pyplot as plt

# Create a random dataset
rng = np.random.RandomState(1)
X = np.sort(5 * rng.rand(80, 1), axis=0)
y = np.sin(X).ravel()
y[::5] += 3 * (0.5 - rng.rand(16))

# Fit regression model
regr_1 = DecisionTreeRegressor(max_depth=2)
regr_2 = DecisionTreeRegressor(max_depth=5)
regr_1.fit(X, y)
regr_2.fit(X, y)

# Predict
X_test = np.arange(0.0, 5.0, 0.01)[:, np.newaxis]
y_1 = regr_1.predict(X_test)
y_2 = regr_2.predict(X_test)
```

Árvores de Decisão

```
>>> from sklearn import tree
>>> X = [[0, 0], [1, 1]]
>>> Y = [0, 1]
>>> clf = tree.DecisionTreeClassifier()
>>> clf = clf.fit(X, Y)
```

```
>>> clf.predict([[2., 2.]])
array([1])
```