Machine Learning

Modelos Lineales

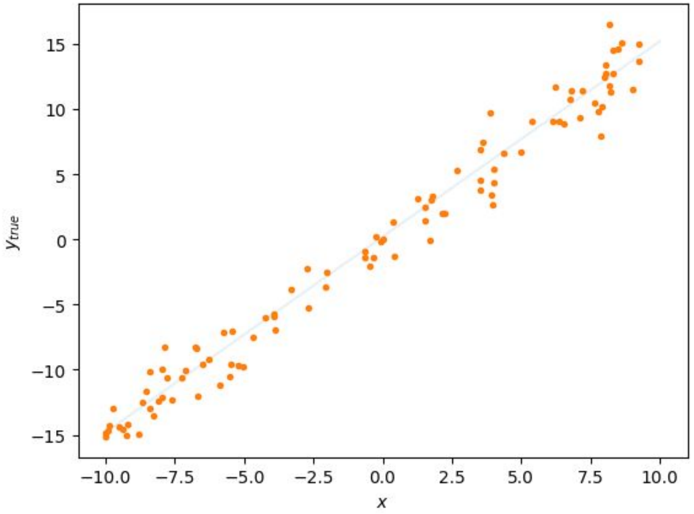
Christian Oliva Moya
Pedro Ramón Ventura Gómez

Regresión Lineal - 1D

- Quiero hacer una **regresión**

  Es decir, predecir valor real

- Problema: $\{(x_1, y_1), (x_2, y_2), \ldots, (x_N, t_N)\}$
  - x_i es el atributo de entrada
  - y_i es la variable objetivo
  - N es el número de ejemplos/datos
- Objetivo: predecir y a partir de x



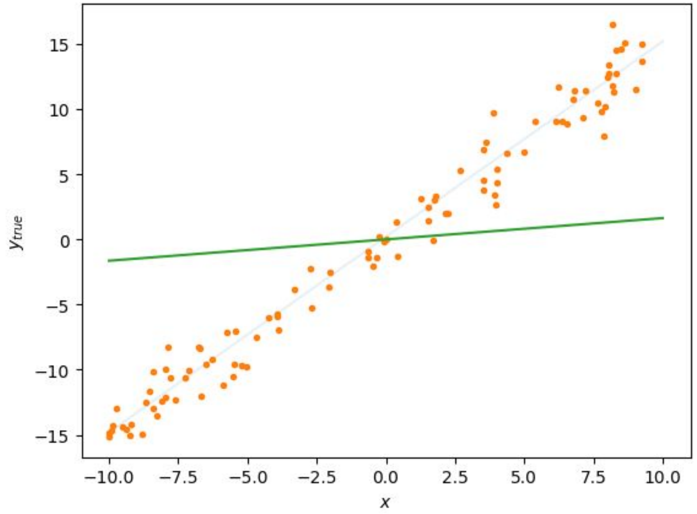
Regresión Lineal - 1D

- ¿Qué forma tiene una recta?

$$y_{pred} = f(x) = xw + b$$

- Si inicializamos de forma aleatoria:
  - w = N(0, 1)   b = 0

En el ejemplo: w = 0.1635

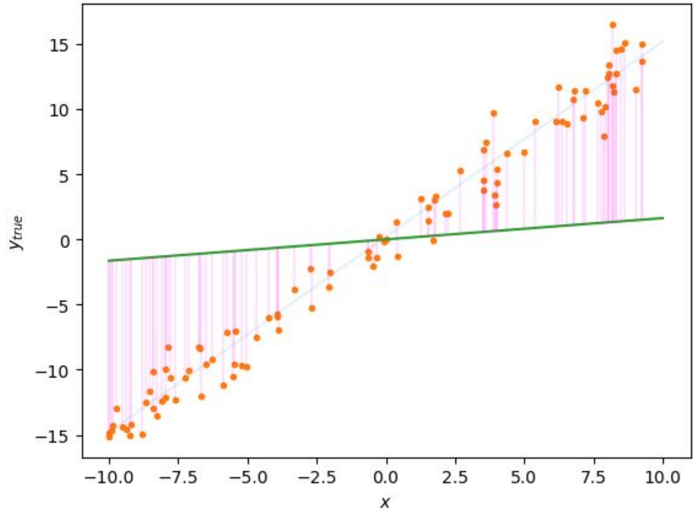


Regresión Lineal - 1D

- ¿Cómo evaluamos el error?
- **Error Cuadrático Medio (MSE)**

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_{true} - y_{pred})^2$$

En el ejemplo: MSE = 74.3825



Regresión Lineal - 1D

- ¿Cómo estimamos el valor óptimo de w y b?

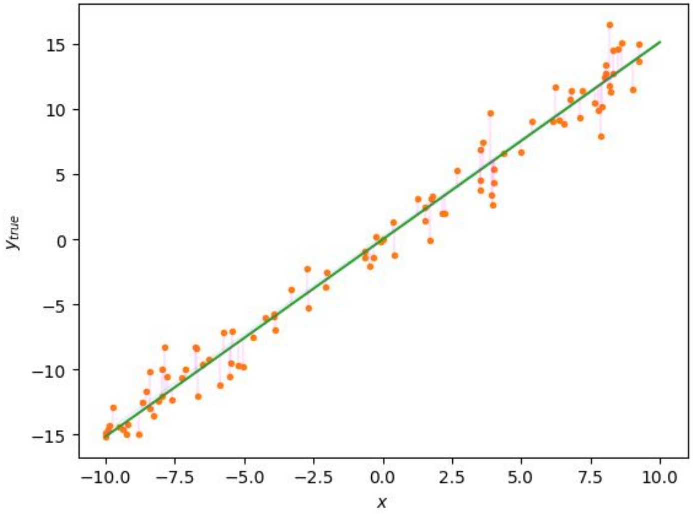
- **Mínimos cuadrados ordinarios (OLS)**

$$w = \frac{cov(\mathbf{x}, \mathbf{y})}{var(\mathbf{x})} = \frac{\sum_{i=1}^{N}(x_i - \bar{\mathbf{x}})(y_i - \bar{\mathbf{y}})}{\sum_{i=1}^{N}(x_i - \bar{\mathbf{x}})^2}$$

$$b = \bar{\mathbf{y}} - \bar{\mathbf{x}}w$$

En el ejemplo: w = 1.51   b = - 0.01

MSE = 2.2145



Regresión Lineal

- En general, para un espacio N-dimensional

$$y_{pred} = f(\mathbf{x}) = \sum_{i=1}^{D} x_i w_i + b = \mathbf{x}\mathbf{w} + b$$

Para resolverlo, hacemos la transformación:

$$\mathbf{x}' = (1, x_1, x_2, \ldots, x_d)$$

$$y_{pred} = f(\mathbf{x}') = \sum_{i=0}^{D} x'_i w_i = \mathbf{x}'\mathbf{w}$$ *donde $w_0 = b$*

Regresión Lineal

- Resolvemos por OLS:

$$y_{pred} = f(\mathbf{x}') = \sum_{i=0}^{D} x'_i w_i = \mathbf{x}'\mathbf{w}$$

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Regresión Lineal

sklearn.linear_model.LinearRegression

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

Ordinary least squares Linear Regression.

LinearRegression fits a linear model with coefficients w = (w1, ..., wp) to minimize the residual sum of squares between the observed targets in the dataset, and the targets predicted by the linear approximation.

https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html

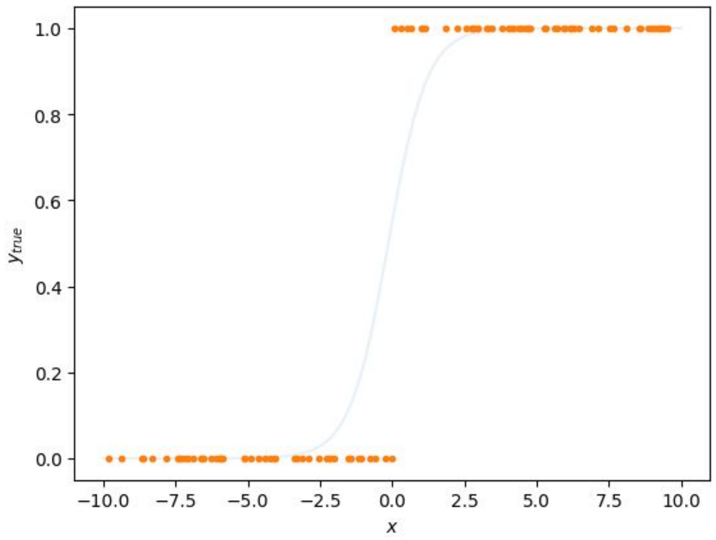
Vamos al notebook: ejemplos_de_regresion_lineal.ipynb

Regresión Logística - 1D

- Quiero hacer una **clasificación**

  Es decir, predecir una probabilidad

- Problema: $\{(x_1, y_1), (x_2, y_2), \ldots, (x_N, t_N)\}$
  - x_i es el atributo de entrada
  - y_i es la variable objetivo
  - N es el número de ejemplos/datos
- Objetivo: predecir la clase y a partir de x



Regresión Logística - 1D

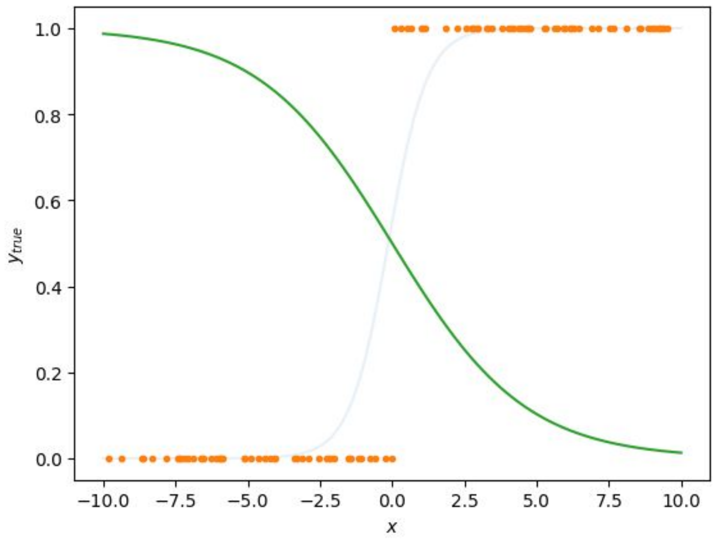
- El modelo es un separador lineal al que se le aplica una transformación no lineal

$$z = xw + b$$

$$y_{pred} = \sigma(z) = \frac{1}{1 + e^{-z}}$$

- Si inicializamos de forma aleatoria:
    - w = N(0, 1)   b = 0

En el ejemplo: w = - 0.4336



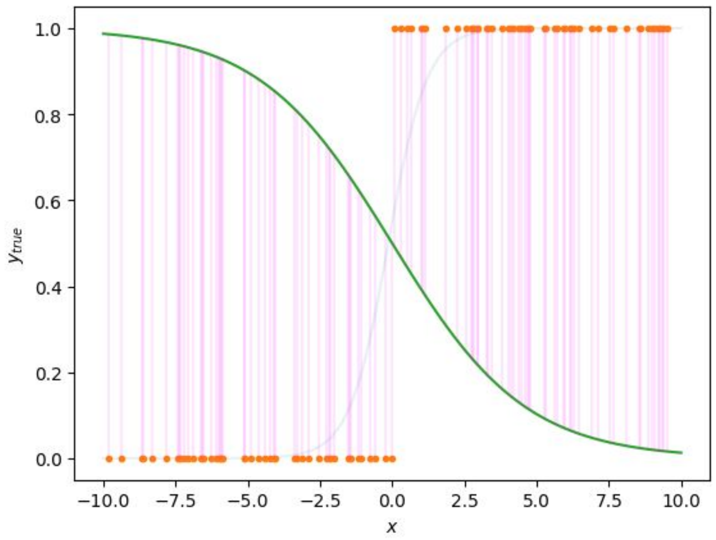
Regresión Logística - 1D

- ¿Cómo evaluamos el error?

- **Cross-Entropy**

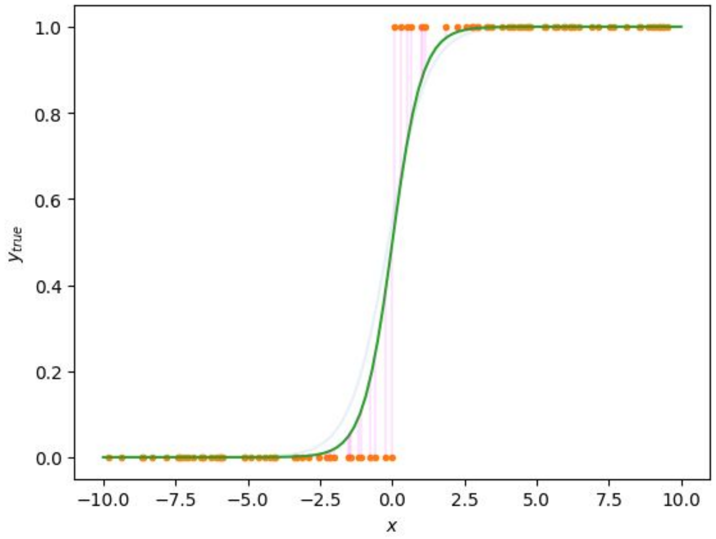
$$XE = -\frac{1}{N}\sum_{i=1}^{N} t_i\, log(y_i) + (1 - t_i)\, log(1 - y_i)$$

En el ejemplo, XE = 2.2978



Regresión Logística - 1D

- ¿Cómo estimamos el valor óptimo de w y b?
- **¡No tiene solución exacta!**
- Utilizamos optimizadores iterativos:
    - SKLearn tiene varios implementados
    - Más adelante veremos cómo optimizar mediante descenso por gradiente



En el ejemplo: w = 1.96 b = - 0.01

XE = 0.0415

Regresión Logística

sklearn.linear_model.LogisticRegression

class `sklearn.linear_model.LogisticRegression`(*penalty='l2', *, dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='lbfgs', max_iter=100, multi_class='auto', verbose=0, warm_start=False, n_jobs=None, l1_ratio=None*) [source]

Logistic Regression (aka logit, MaxEnt) classifier.

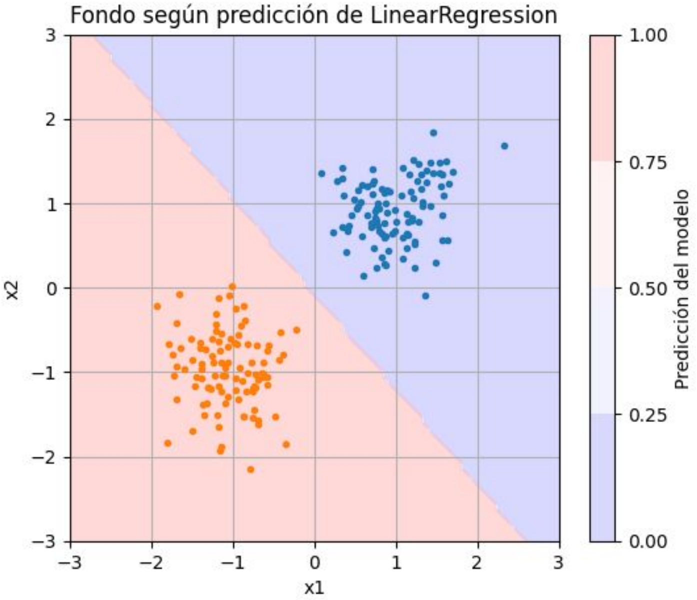
In the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme if the 'multi_class' option is set to 'ovr', and uses the cross-entropy loss if the 'multi_class' option is set to 'multinomial'. (Currently the 'multinomial' option is supported only by the 'lbfgs', 'sag', 'saga' and 'newton-cg' solvers.)

This class implements regularized logistic regression using the 'liblinear' library, 'newton-cg', 'sag', 'saga' and 'lbfgs' solvers. **Note that regularization is applied by default**. It can handle both dense and sparse input. Use C-ordered arrays or CSR matrices containing 64-bit floats for optimal performance; any other input format will be converted (and copied).

The 'newton-cg', 'sag', and 'lbfgs' solvers support only L2 regularization with primal formulation, or no regularization. The 'liblinear' solver supports both L1 and L2 regularization, with a dual formulation only for the L2 penalty. The Elastic-Net regularization is only supported by the 'saga' solver.

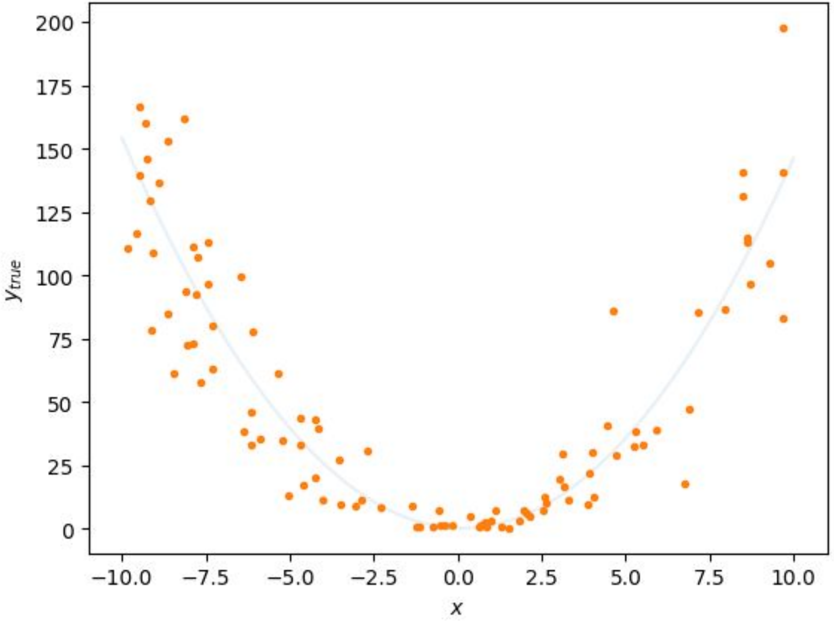
Regresión Logística - Ejemplo 2D

- La clasificación en 2 dimensiones se ve más clara
- El espacio se divide según el separador lineal de la regresión logística



Problemas no lineales

- ¿Qué podemos hacer cuando los problemas son no lineales?



Problemas no lineales

- ¿Qué podemos hacer cuando los problemas son no lineales?
- **Hacer una transformación** no lineal de los atributos del problema

  Entonces podemos resolver el problema linealmente en el nuevo espacio de atributos

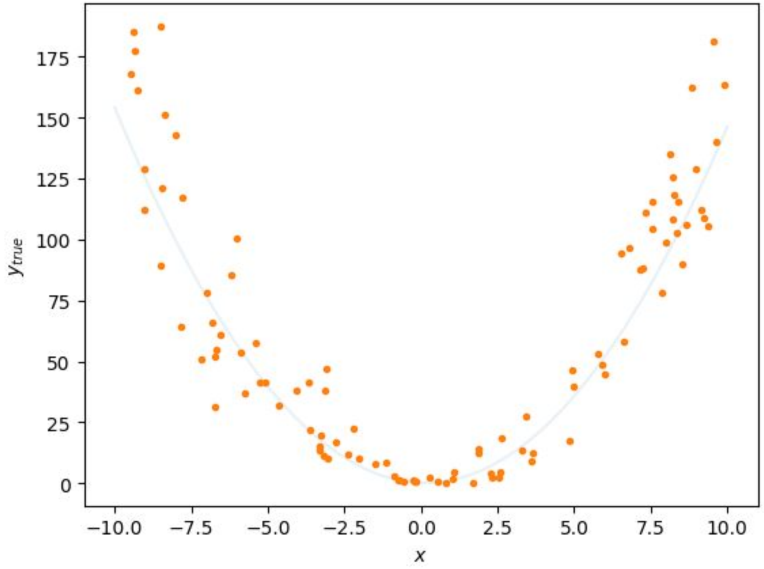
- Utilizar heurísticas con **conjuntos de clasificadores**
- Dejar que el modelo aprenda a hacer la transformación no lineal
  - **SVMs**
  - **Redes neuronales**

Problemas no lineales - Transformación no lineal

- Podemos **hacer una transformación** no lineal de los atributos del problema

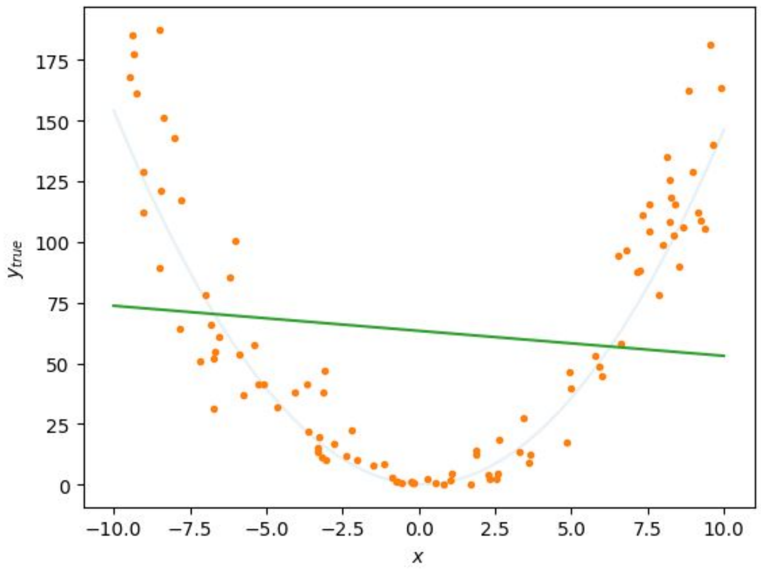
  Entonces podemos resolver el problema linealmente en el nuevo espacio de atributos

- Veámoslo con un ejemplo sencillo



Problemas no lineales - Transformación no lineal

- La regresión lineal se adapta a los datos lo mejor que puede

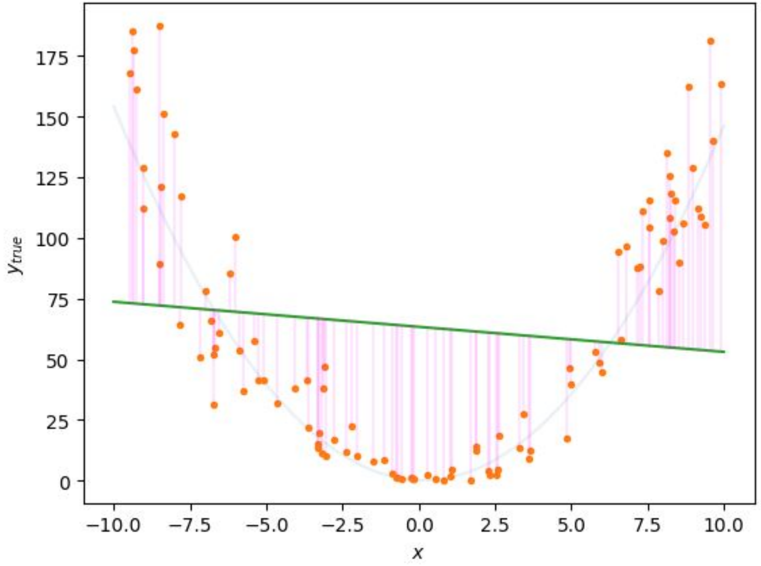


Problemas no lineales - Transformación no lineal

- La regresión lineal se adapta a los datos lo mejor que puede

  En el ejemplo: MSE = 3047.42

- ¿Qué transformación podemos hacer?

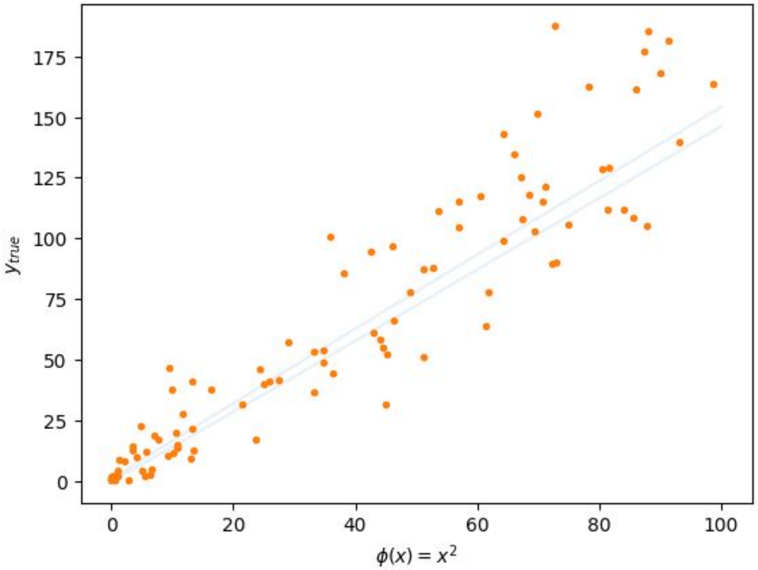


Problemas no lineales - Transformación no lineal

- ¿Qué transformación podemos hacer?

$$\phi(x) = x^2$$

- Ahora sí, aplicamos la regresión lineal



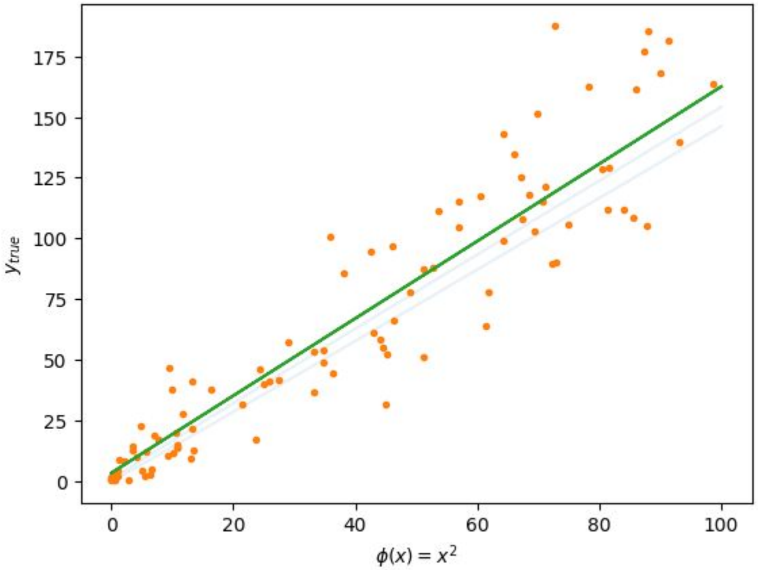
Problemas no lineales - Transformación no lineal

- ¿Qué transformación podemos hacer?

$$\phi(x) = x^2$$

- Ahora sí, aplicamos la regresión lineal

En el ejemplo w = 1.59   b = 3.22



Problemas no lineales - Transformación no lineal

- ¿Qué transformación podemos hacer?

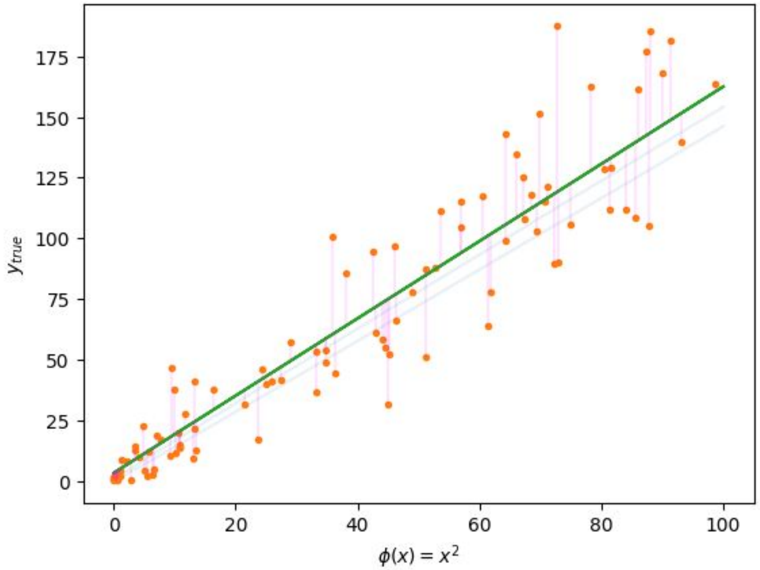
$$\phi(x) = x^2$$

- Ahora sí, aplicamos la regresión lineal

En el ejemplo w = 1.59   b = 3.22

- ¿Qué error cometemos?

MSE = 354.12

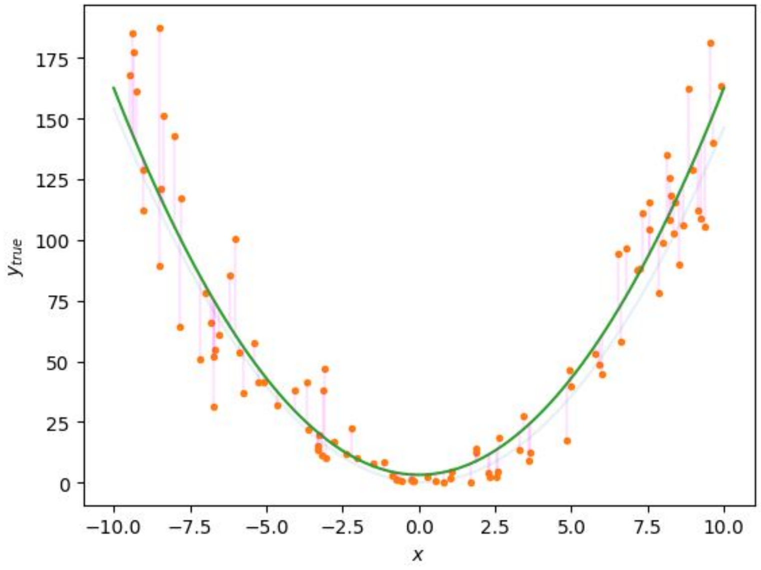


Problemas no lineales - Transformación no lineal

- Volviendo al espacio original…

  En el ejemplo w = 1.59   b = 3.22

  MSE = 354.12



Problemas no lineales - Transformación no lineal

- Podemos **hacer una transformación** no lineal de los atributos del problema
- Veámoslo con un ejemplo sencillo
- **NO te recomiendo** buscar a mano la transformación no lineal. ¿Por qué?
  - Depende del problema a resolver
  - Necesitas una idea feliz en la mayoría de los casos
  - Si tenemos alternativas, ¿por qué no usarlas?
    - **SVMs**
    - **Conjuntos de clasificadores**
    - **Redes neuronales**