



Regresión Lineal con sk-learn



Scikit-learn

- Librería de software libre para Machine Learning sobre lenguaje Python.
- Incluye algoritmos para Regresión, Clasificación, clustering, preprocesamiento, reducción de dimensionalidad, entre otros.
- Prototipado rápido de modelos.
- Entiende formatos Numpy y Pandas.

Boston dataset

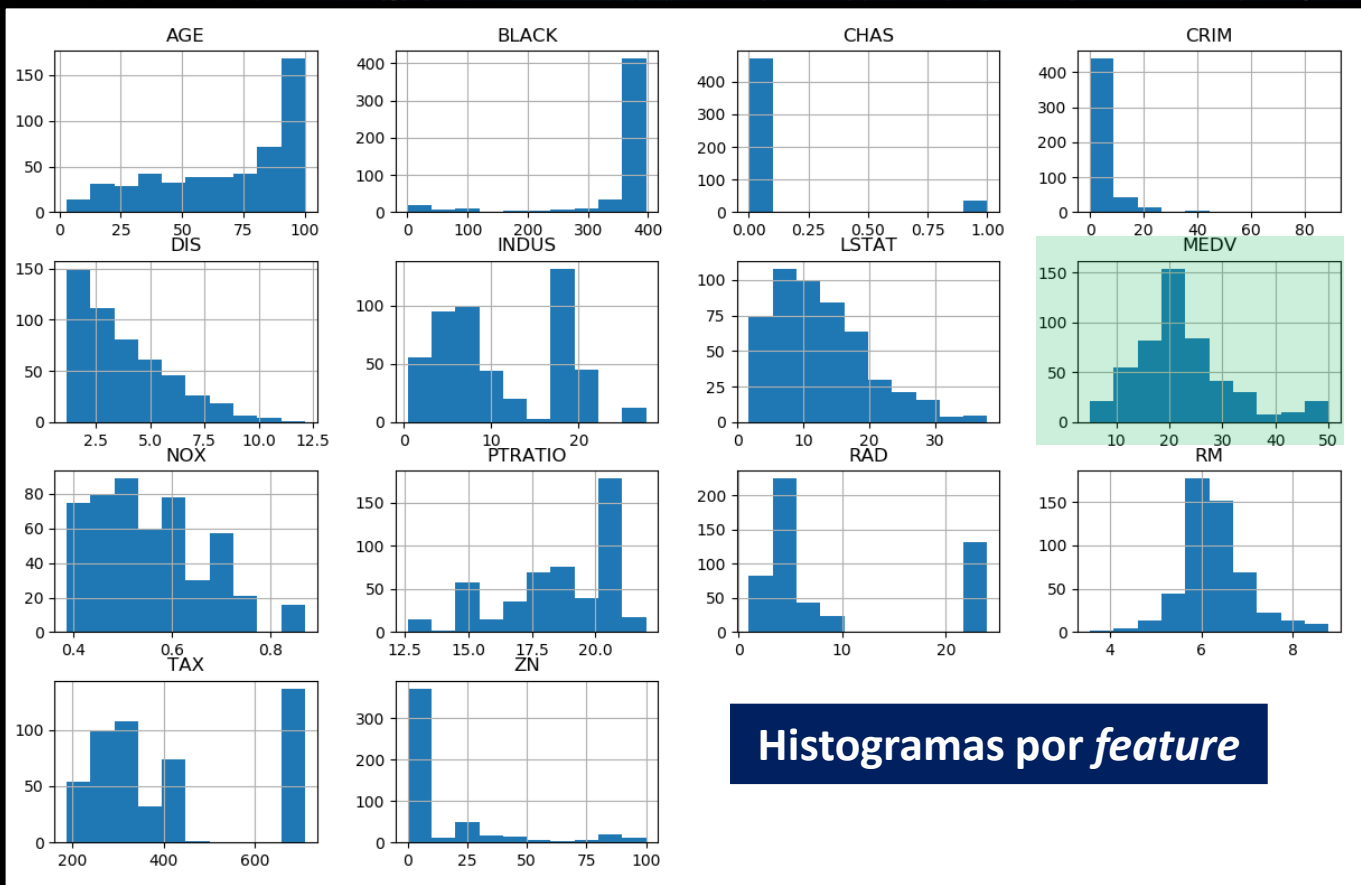
Regresión ND

Problema:

Predecir el precio de una casa en base a información existente en una base de datos.

Variable	Detalle
crim	crímenes per cápita
zn	proporción de áreas residenciales
indus	proporción de negocios no retail
chas 1	si está cerca del río, 0 sino
nox	concentración de óxidos de nitrógeno
rm	habitaciones promedio
age	proporción de inmuebles anteriores a 1940
dis	distancia promedio a centros de empleo
rad	accesibilidad por autopistas
tax	impuestos por \$10.000
ptratio	relación alumno-docente en escuelas
black	coeficiente de personas negras (dataset del 1980!)
lstat	porcentaje de personas de bajo status
Medv	Valor medio de viviendas, en miles de \$.

Boston dataset – histogramas por variable



Regresión lineal con Scikit-learn

```
data= pd.read_csv('datasets/boston.csv')  
x= data.drop('medv', axis=1)  
y= data['medv']
```

Cargar data set y
separar datos (x) de
target (y)

```
from sklearn.linear_model import LinearRegression  
modelo= LinearRegression()
```

Crear modelo de
regresión lineal

```
modelo.fit(x, y)
```

Entrenamiento de un
modelo

```
y_predict= modelo.predict(x)
```

Utilizar el modelo
entrenado para
predecir nuevos valores

Calculo del error del modelo

```
from sklearn.metrics import mean_squared_error, mean_absolute_error

mse_error= mean_squared_error(y_predict, y)
print("Error cuad. medio: %.2f" % mse_error)

mae_error= mean_absolute_error(y_predict, y)
print("Error abs. medio: %.2f" % mae_error)
```




Calculamos el error cuadrático medio y el error absoluto medio para el modelo entrenado

Error cuad. medio: 21.89
Error abs. medio: 3.27

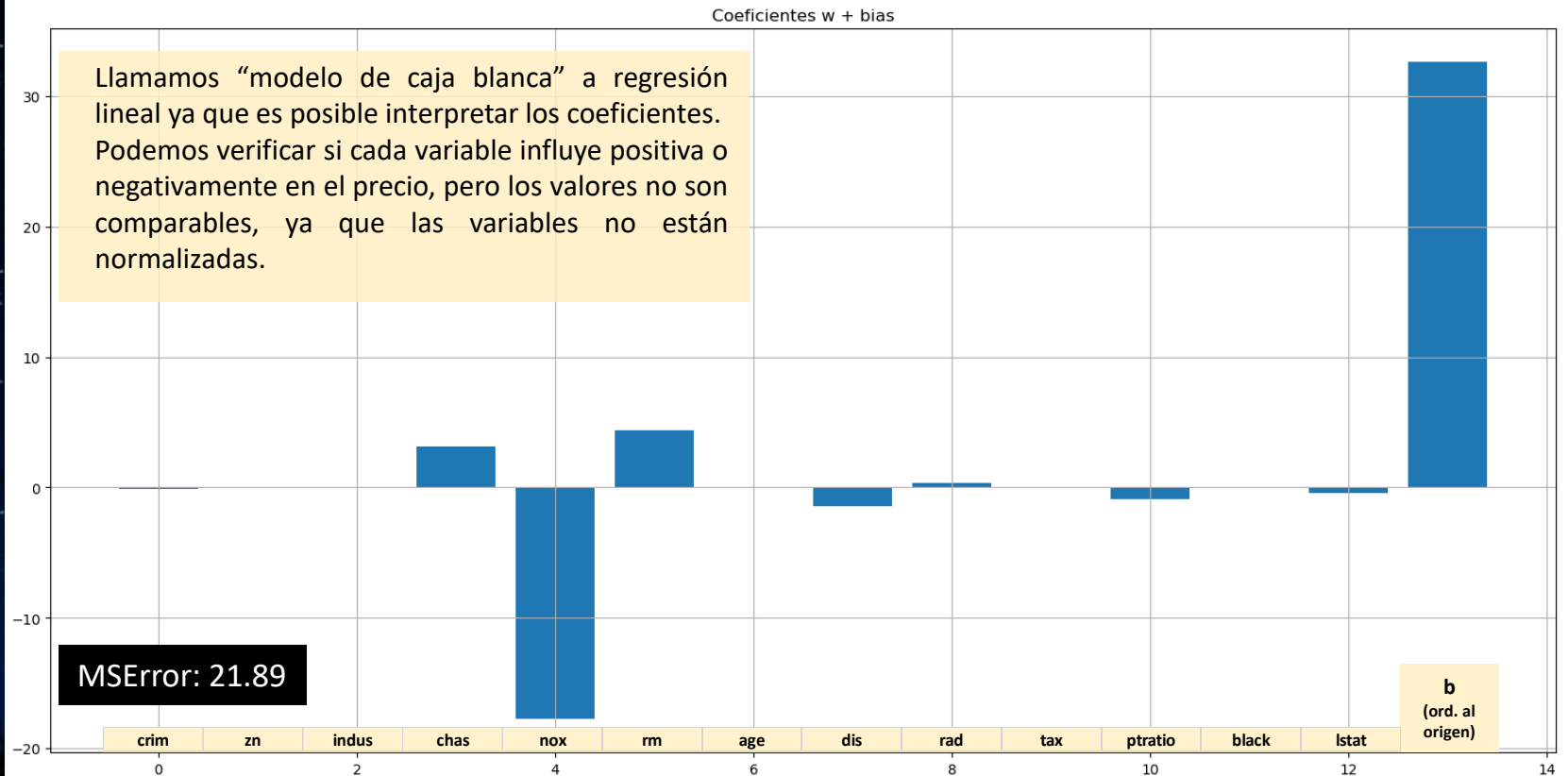
Coeficientes del modelo

Sklearn pone un guión bajo a todas las variables que derivan de los datos de entrenamiento (para no confundir con parámetros del usuario).



```
w= modelo.coef_  
b= modelo.intercept_  
  
plt.bar(range(d+1), np.concatenate((w,[b])))
```


Coeficientes del modelo



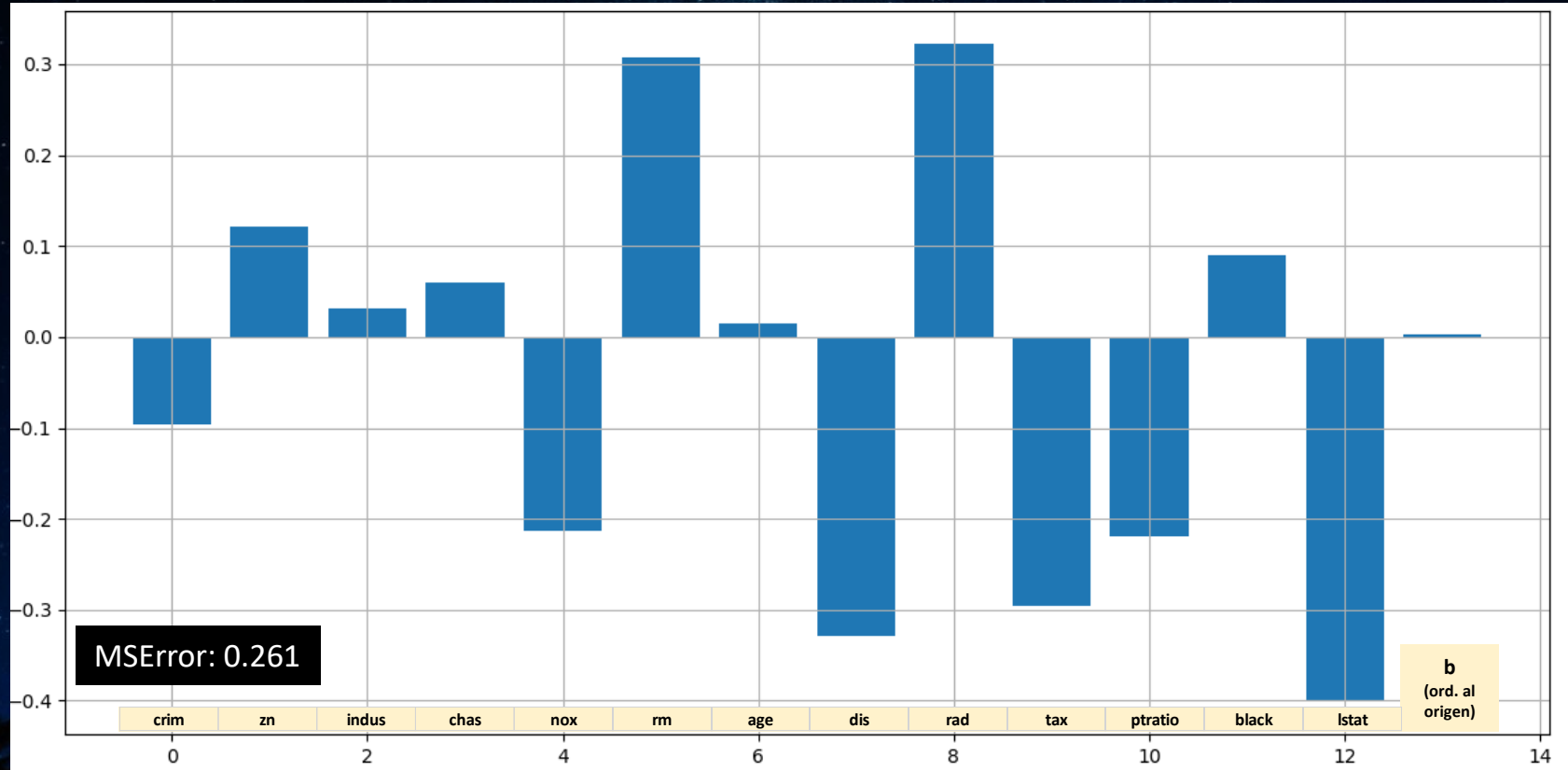
Coeficientes del modelo – Variables normalizadas

```
from sklearn.preprocessing import StandardScaler  
scaler = StandardScaler()  
scaler.fit(x)  
x= scaler.transform(x)
```

Normalizamos las
variables con z-core



Coeficientes del modelo – Variables normalizadas



Wine Quality Dataset

- El dataset contiene información sobre diferentes vinos juntos con su calidad establecida por un experto (de 1 a 10).
- No confundir con un problema de clasificación!

Wine quality (2009)

Input variables:

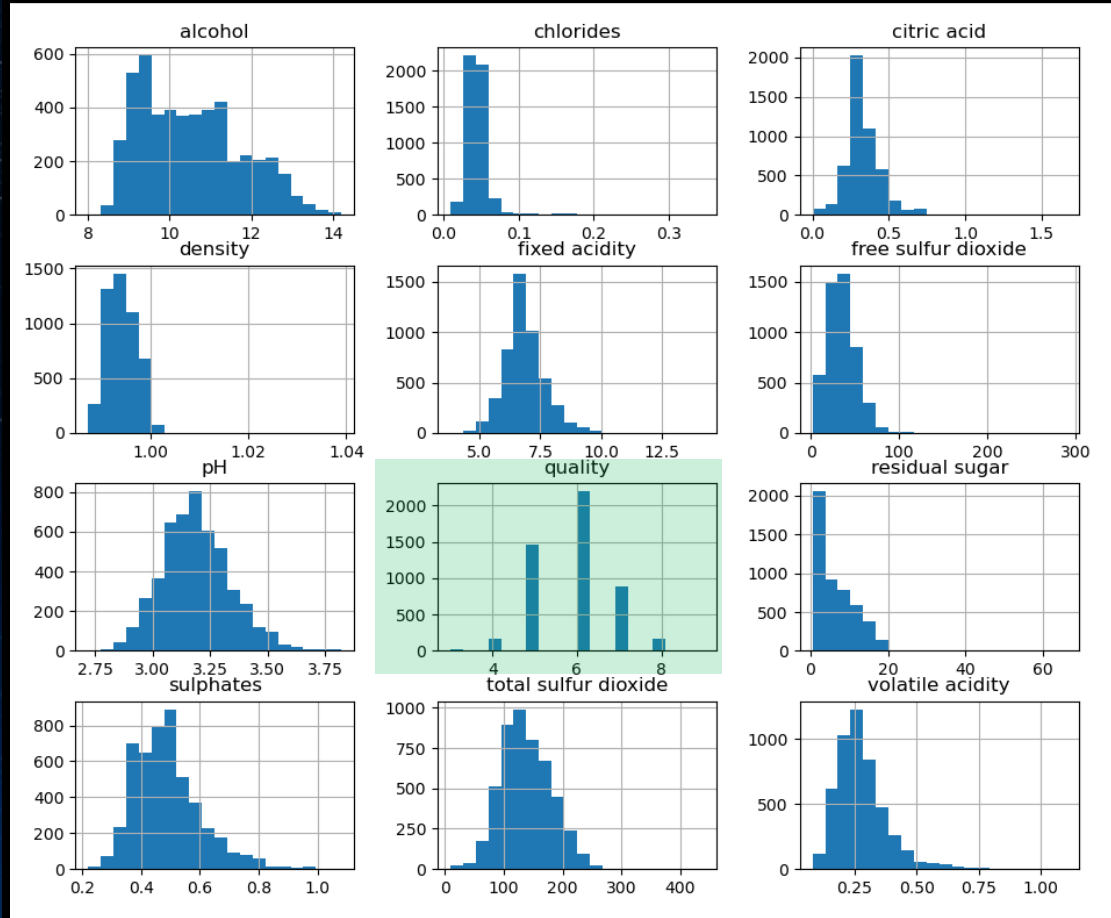
- 1 - fixed acidity
- 2 - volatile acidity
- 3 - citric acid
- 4 - residual sugar
- 5 - chlorides
- 6 - free sulfur dioxide
- 7 - total sulfur dioxide
- 8 - density
- 9 - pH
- 10 - sulphates
- 11 - alcohol

Output variable :

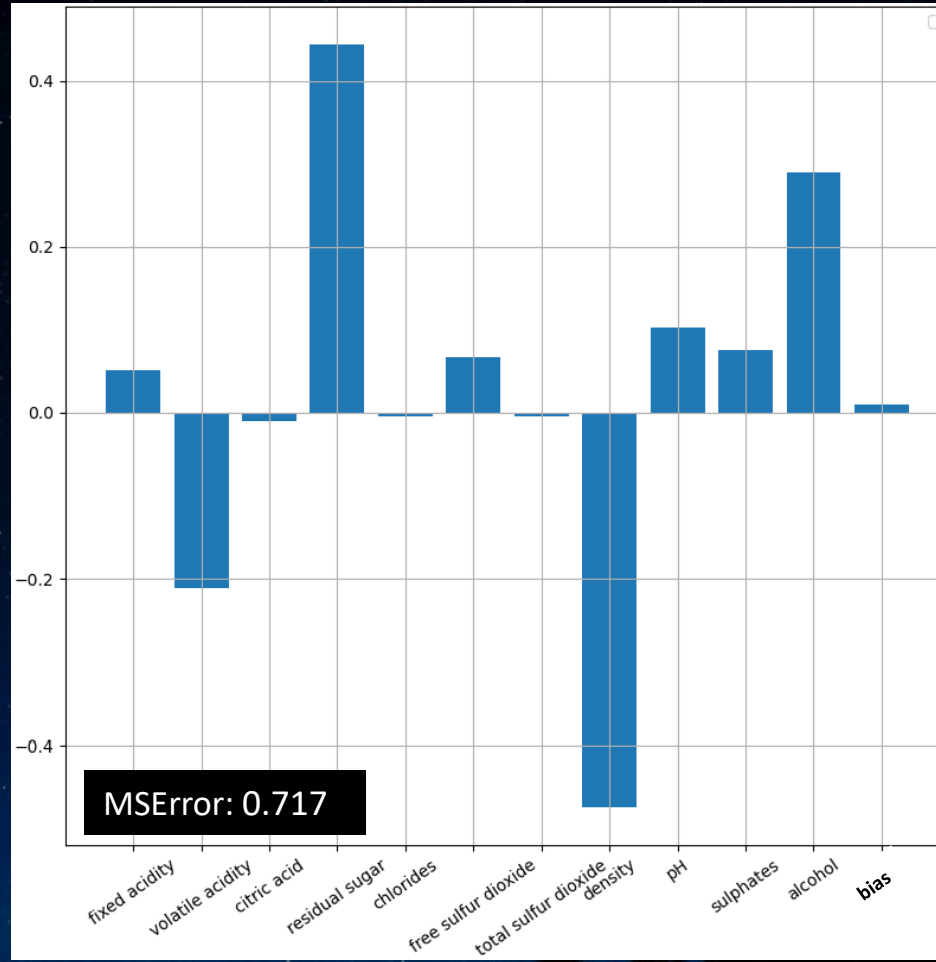
- 12 - quality (score between 0 and 10)

Wine Quality Dataset - histogramas

- Notar que la “calidad” es un valor discreto.
- No hay la misma cantidad de datos por cada tipo de vino.
- El dataset tiene un sesgo!



Wine Quality Dataset - Coeficientes



Sesgo del dataset

Error diferente por cada tipo de vino.

