In [1]:

```
%load_ext watermark
%watermark
```

```
2019-05-30T21:30:27+02:00

CPython 3.6.5
IPython 6.4.0

compiler   : GCC 7.2.0
system     : Linux
release    : 5.1.5-arch1-2-ARCH
machine    : x86_64
processor  :
CPU cores  : 4
interpreter: 64bit
```

Evaluacion de modelos de regresion

In [2]:

```
from sklearn import datasets
```

Cargamos el dataset. Para este ejemplo utilizaremos el dataset de viviendas de Boston (Boston Housing Dataset)

In [3]:

```
boston = datasets.load_boston()
```

Creamos un modelo de regresión lineal

In [4]:

```
from sklearn.linear_model import LinearRegression
```

In [5]:

```
model = LinearRegression()
```

Con el modelo de regresión lineal instanciado ajustamos el modelo teniendo como variable independiente la columna "data" del dataset y como variable dependiente los valores de la columna "target"

In [6]:

```
model.fit(X=boston["data"], y=boston["target"])
y_pred = model.predict(boston["data"])
```

In [7]:

```
y_objetivo = boston["target"]
y_pred = model.predict(boston["data"])
```

In [8]:

```
from sklearn import metrics
```

Error Absoluto medio

El Error absoluto medio (Mean Absolute Error o MAE) se define como:

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

Es decir, la media de las diferencias entre la variables objeto y las predicciones sin el signo. MAE es una métrica robusta, en cuanto a que no varia mucho si hay valores extremos en los datos. El error se puede interpretar como unidades de la variable objetivo (por ejemplo, si la variable objetivo es en dolares MAE estará tambien en dólares).

In [9]:

In [9]:

```
metrics.mean_absolute_error?
```

In [10]:

```
metrics.mean_absolute_error(y_objetivo, y_pred)
```

Out[10]:

```
3.2708628109003137
```

Error cuadrático medio

El Error cuadrático medio (Mean Squared Error o MSE)

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

Dado que el MSE se define en unidades al cuadrado, lo cual no es intuitivo, generalmente se usa su raíz.

Raíz del error cuadrático medio

La raiz del error cuadrático medio (Root Mean Squared o RMSE) se diferencia del MSE en que el resultadose medir en las mismas unidades que la variable objetivo.

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

Sin embargo, tiene un problema y es que da más importancia a los errores grandes. Por ejemplo en el Boston Housing DataSet, si tenemos dos onservaciones y sus predicciones:

```
observacion1: MEDV: 10   MEDV_pred: 15    RMSE: (10-15)²=25
observacion2: MEDV: 1000 MEDV_pred: 1010 RMSE: (1000-1010)²=100
```

El usar RMSE como medida de error significa que se le dará más peso de la observacion2 que al de la observacion1, pese a que un error de 5.000 dólares en una zona donde el valor media es 10.000 dólares es un error mucho más grave que un error de 10.000 dólares en una zona donde el valor medio de las casas es de 1 millon de dólares.

RMSE es mas sensible que MAE a variaciones en los errores de predicción aqui tenemos un ejemplo donde se ve esto:

En este ejemplo se ce como para los tres casos, el MAE no varía, mientras que en función de la distrubición de errores el RMSE pue ser igual al MAE, (todos los errores iguales) o muchos mayor (muchos errores nulos con un error muy grande)

Para aquellos casos donde lo que queramos es esto (evitar errores grandes) y cuando la distribución de la variable objetivo este bién distribuida se puede usar RMSE en de MAE.

In [11]:

```
import numpy as np
```

Se calcula haciendo la funcion `sqrt`

In [12]:

```
np.sqrt(metrics.mean_squared_error(y_objetivo, y_pred))
```

Out[12]:

```
4.679191295697281
```

Coeficiente de determinación

El coeficiente de determinación (R2 o R-squared)(Se usa sobre todo en regresion lineal) mide la porción de la varianza de la variable objetivo que se puede explicar por el modelo.

R2 tiene un valor máximo de 1 (cuando el modelo explica toda la varianza), aunque puede tener valores negativos.

Hay **varias formas de definir R2**, pero una de las más sencillas es simplemente la correlación (definida como la **Correlacion de Pearson**) entre la variable objetivo y la predicciones, elevada al cuadrado.

Un problema importante que tiene R2 es que no nos indica si el modelo explica la varianza debido a que está sobreajustado

Un problema importante que tiene R2 es que no nos indica si el modelo explica la varianza debido a que está sobreajustado consideración la complejidad del modelo.

$$1 - \frac{(1 - R^2)(n-1)}{(n-k-1)}$$

donde `n` es el número de observaciones y `k` es el número de coeficientes del modelo (sin contar el término independiente)

In [13]:

```
model_r2 = metrics.r2_score(y_objetivo,y_pred)
model_r2
```

Out[13]:

```
0.7406426641094095
```

In [14]:

```
import numpy
```

In [15]:

```
np.corrcoef(y_objetivo, y_pred)**2
```

Out[15]:

```
array([[1.        , 0.74064266],
       [0.74064266, 1.        ]])
```

In [16]:

```
len(model.coef_)
```

Out[16]:

```
13
```

In [17]:

```
model_r2_ajustado = 1 - (1-model_r2)*(len(boston["target"])-1)/(len(boston["target"])-boston["data"
].shape[1]-1)
model_r2_ajustado
```

Out[17]:

```
0.733789726372463
```