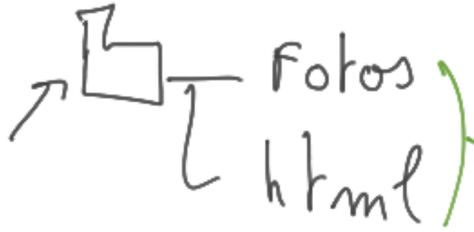
idealista

FICHEROS



kaggle

└ CSV

Fotocasa ~ IDEALISTA

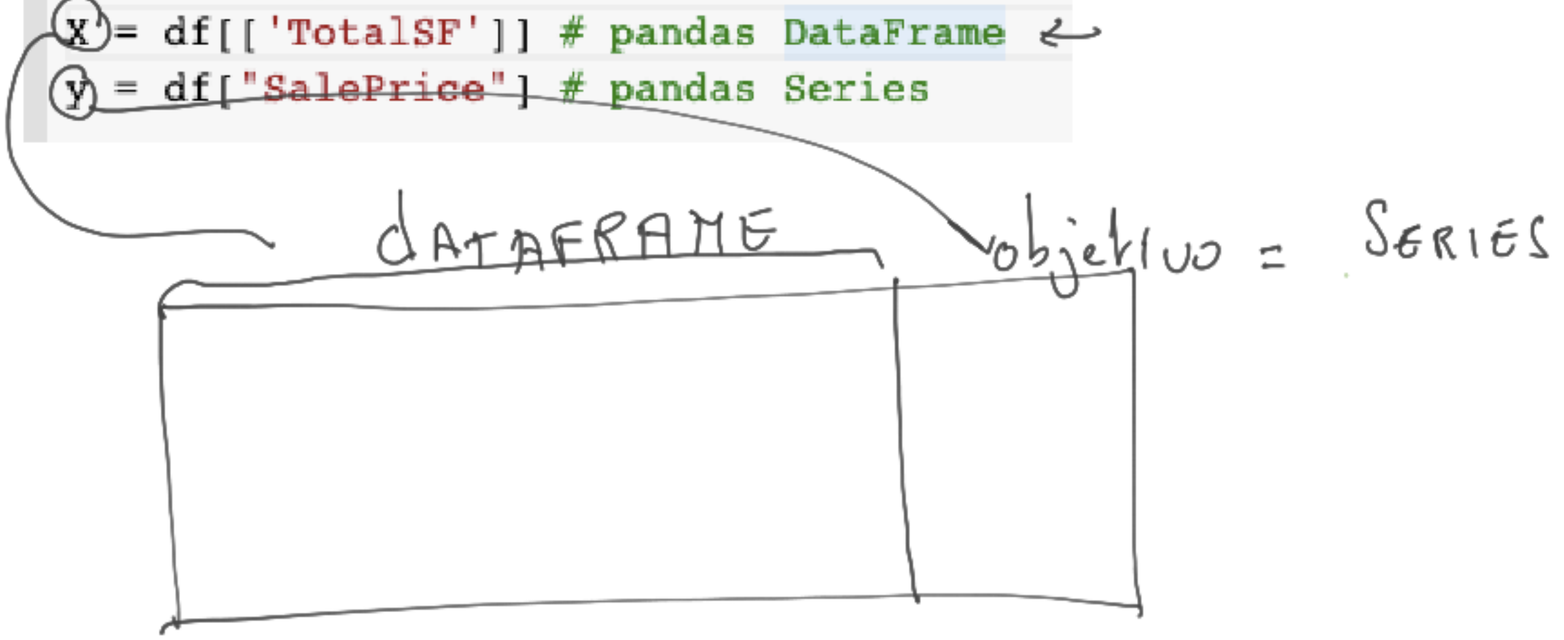
CATASTRO

html

Id	Area	BARRIO	Exterior ...	Obj
Casa 1	80	MORALEJA	✓	1000000
1.				200 000
1.				
1.		Vacío		

UNSTRUCTURED DATA → STRUCTURED DATA

```python
# Data Preparation
X = df[['TotalSF']] # pandas DataFrame
y = df["SalePrice"] # pandas Series
```



$$y = f(x)$$

REG. LINEAL

Características

$$y = a x_1 + b x_2 + c x_3$$

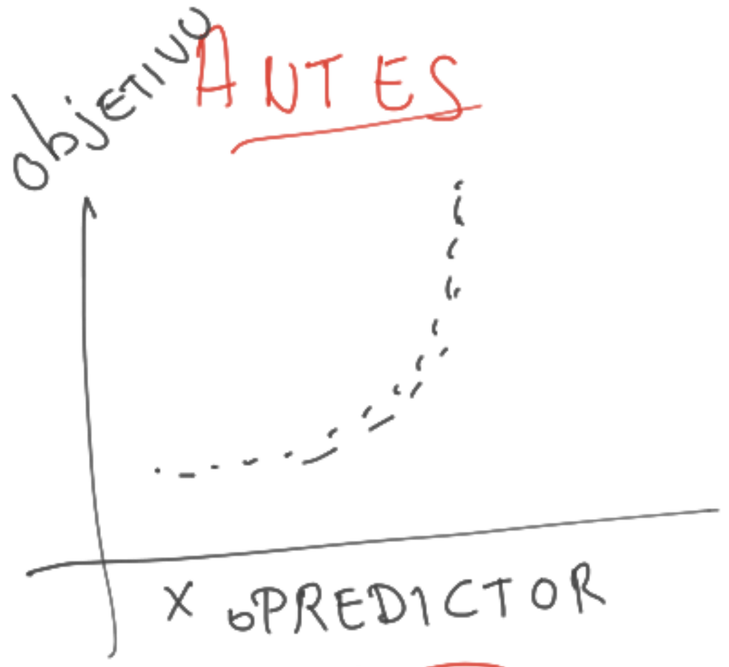
a, b, c

reg.fit(X, y) → best a, b, c

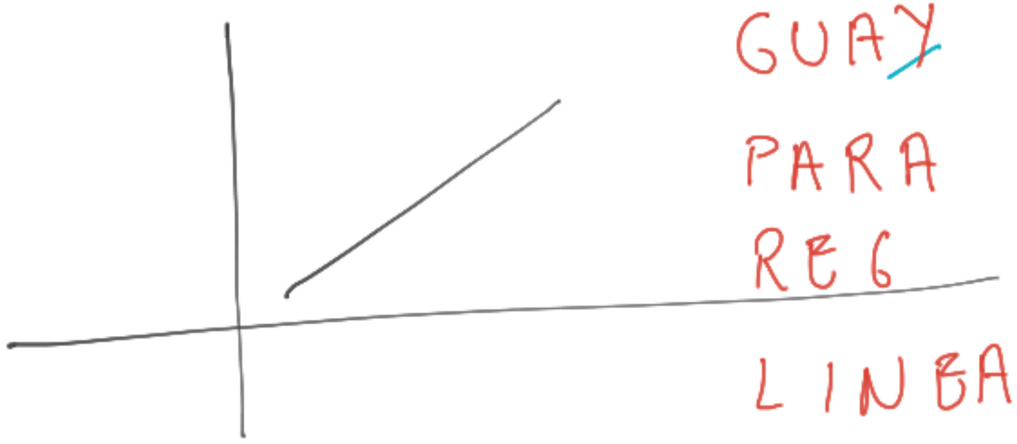
min OLS

$$J = \min \Sigma (y - \hat{y})^2$$

$$J_2 = \min \Sigma |y - \hat{y}|$$



log



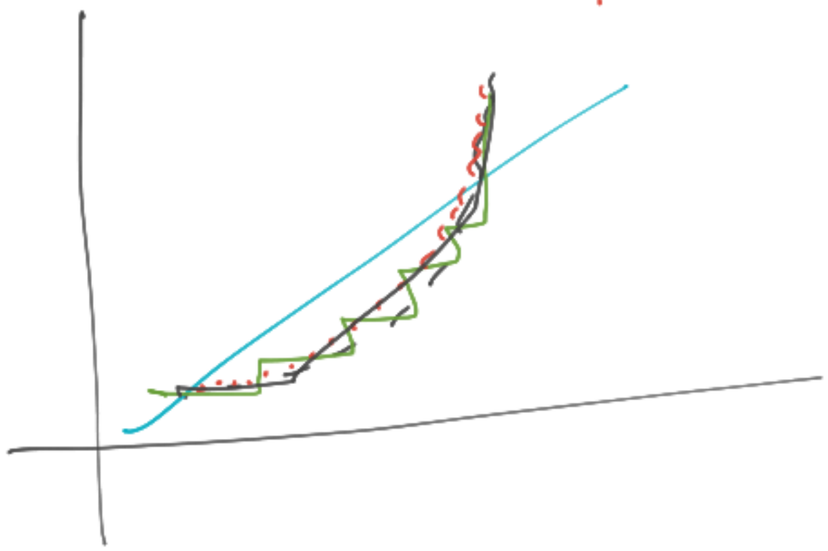
GUAY

PARA

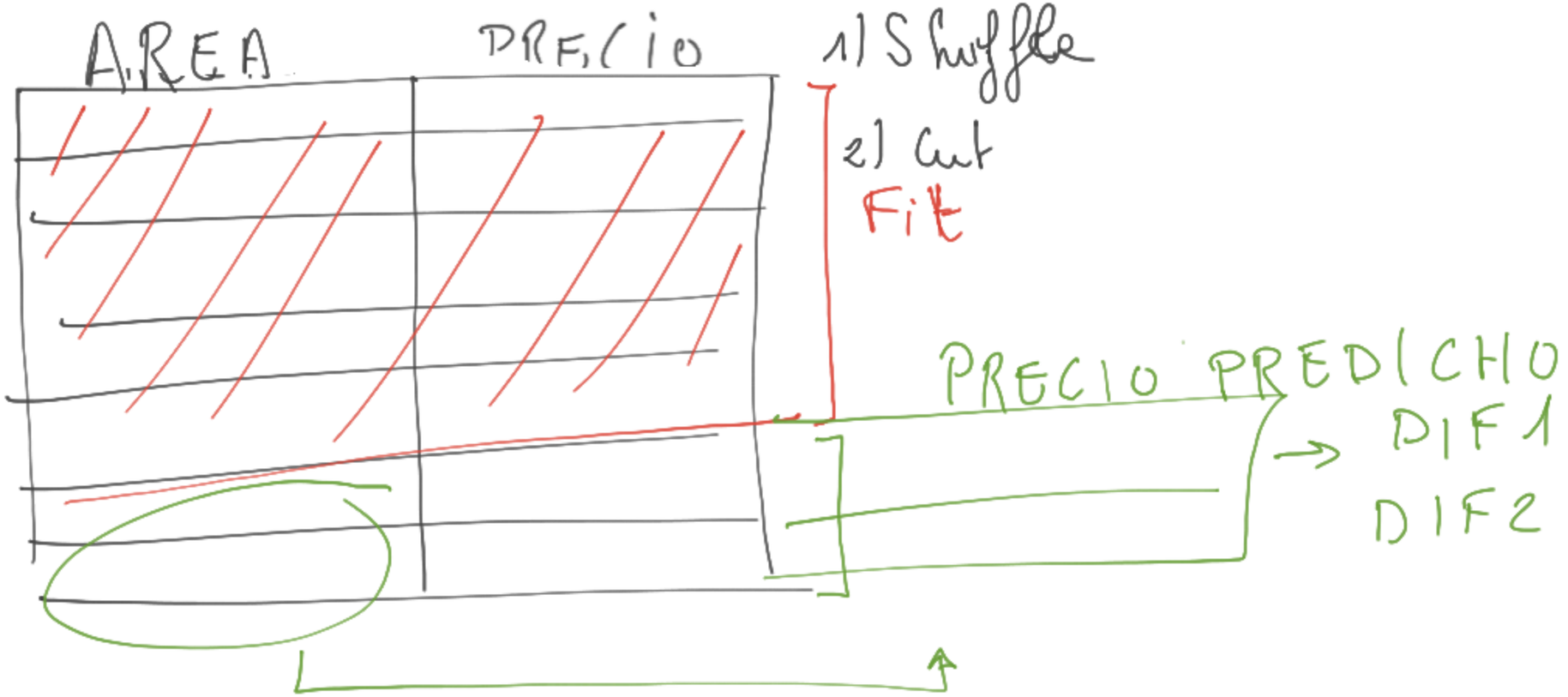
REG

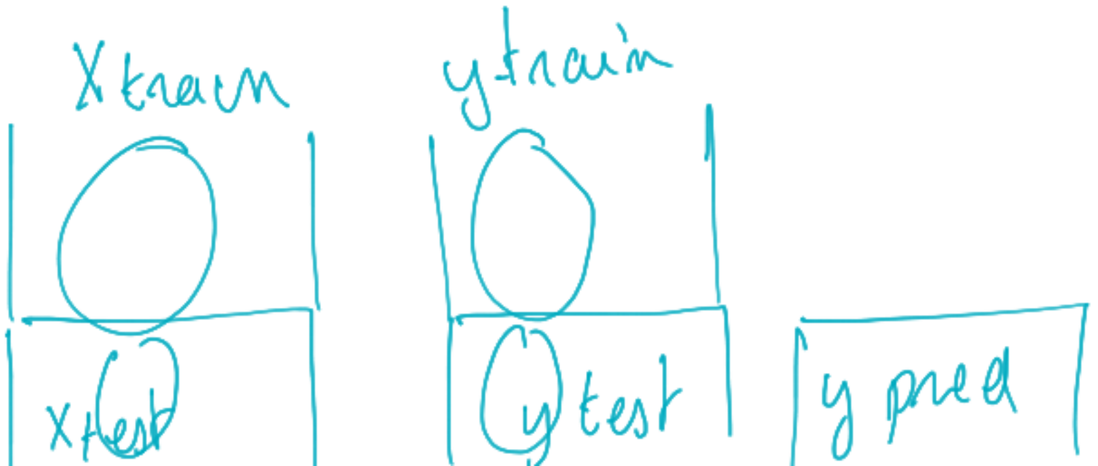
LINEAL

DESPUES

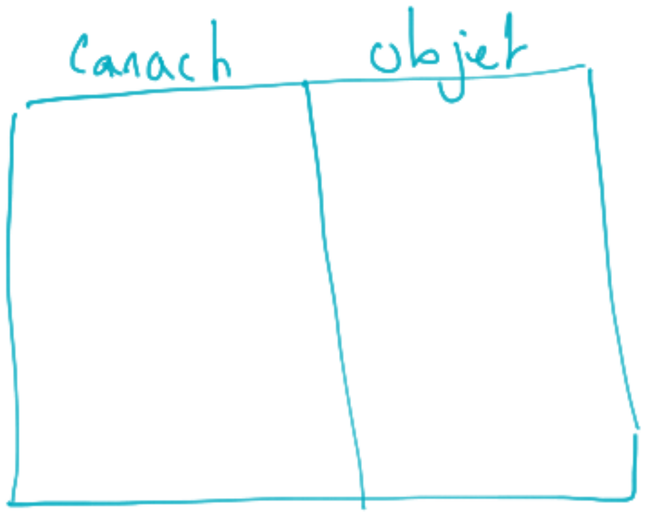


cat.		cat=A	cat=B	cat=C
A		1	0	0
B	→	0	1	0
A		1	0	0
B		0	1	0
C		0	0	1


AREA
PRECIO
1) Shuffle
2) Cut
Fit
PRECIO PREDICHO
DIF1
DIF2


Xtrain
ytrain
Xtest
ytest
ypred

K Nearest NEIGHBORS

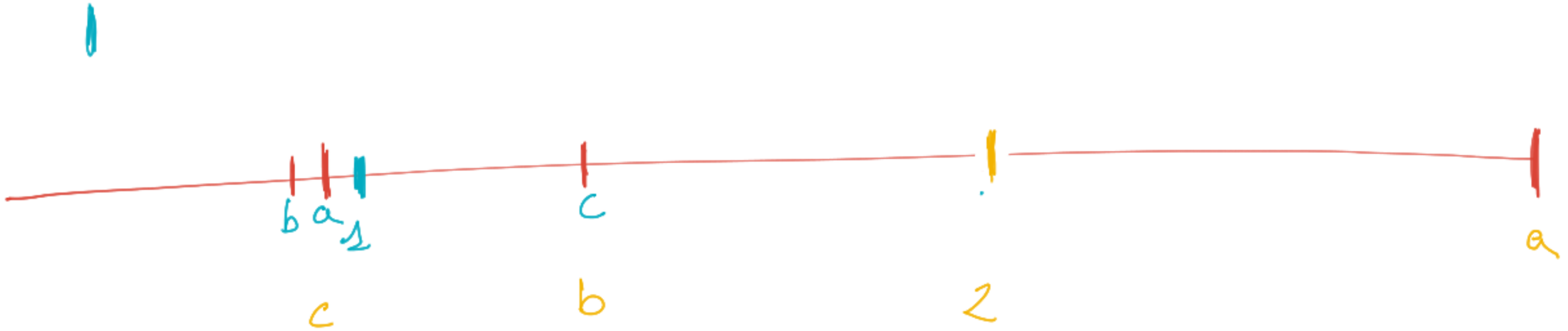


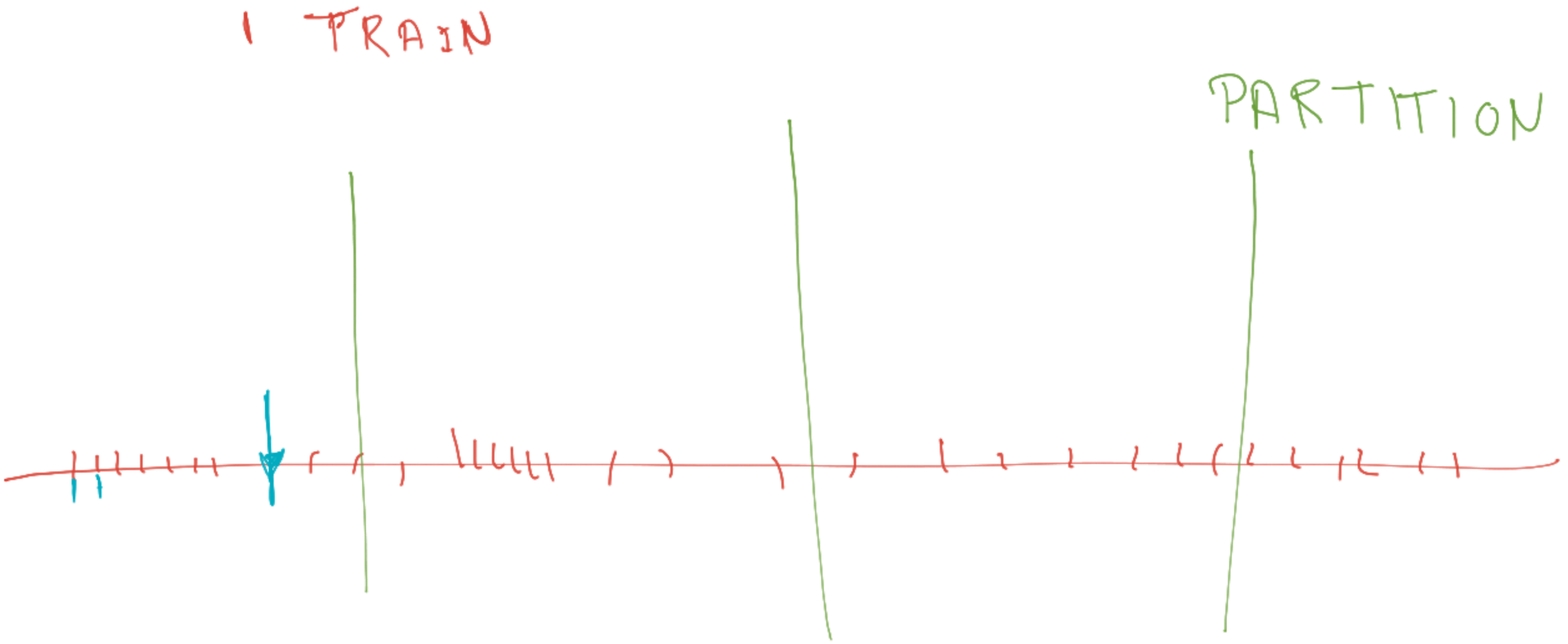
No Fit X

1) Buscar K elementos + próximos + parecidos

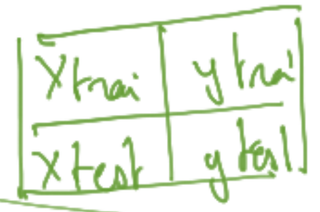
3 vecinos

(TRAIN → Se precio)




1 TRAIN
PARTITION

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=42)
```



LIN REG

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

```
reg = LinearRegression()
```

```
reg.fit(X_train,y_train)
```

PRED TEST

```
y_pred = reg.predict(X_test)
```

```
mean_absolute_error(y_test,y_pred)
```

```
# RMSE: Lin Reg
np.sqrt(mean_squared_error(y_test,y_pred))
```

K Nei

```
from sklearn.neighbors import KNeighborsRegressor
```

```
regk = KNeighborsRegressor(n_neighbors=5)
```

```
regk.fit(X_train,y_train)
```

↓ PRED TEST

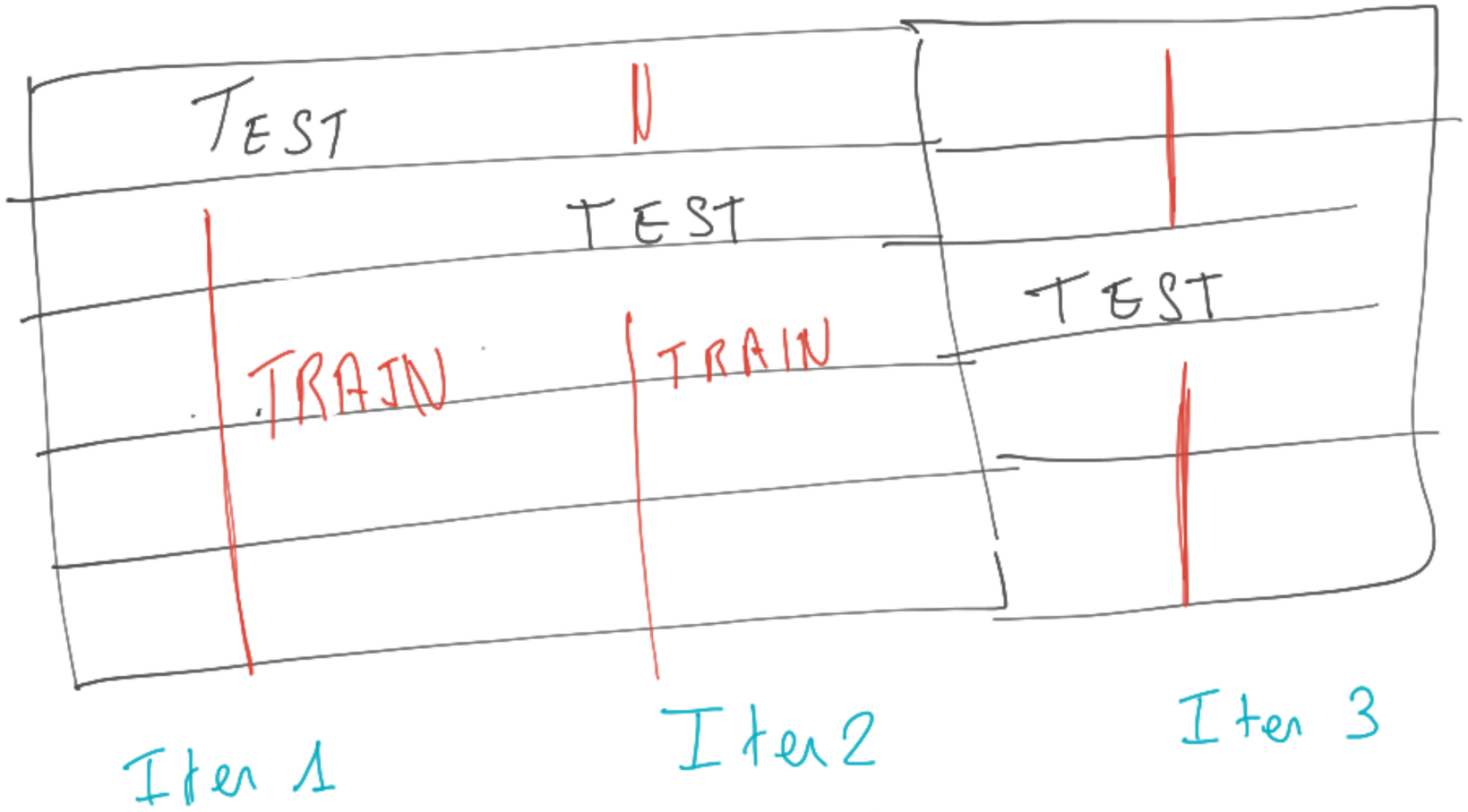
```
] y_predk = regk.predict(X_test)
```

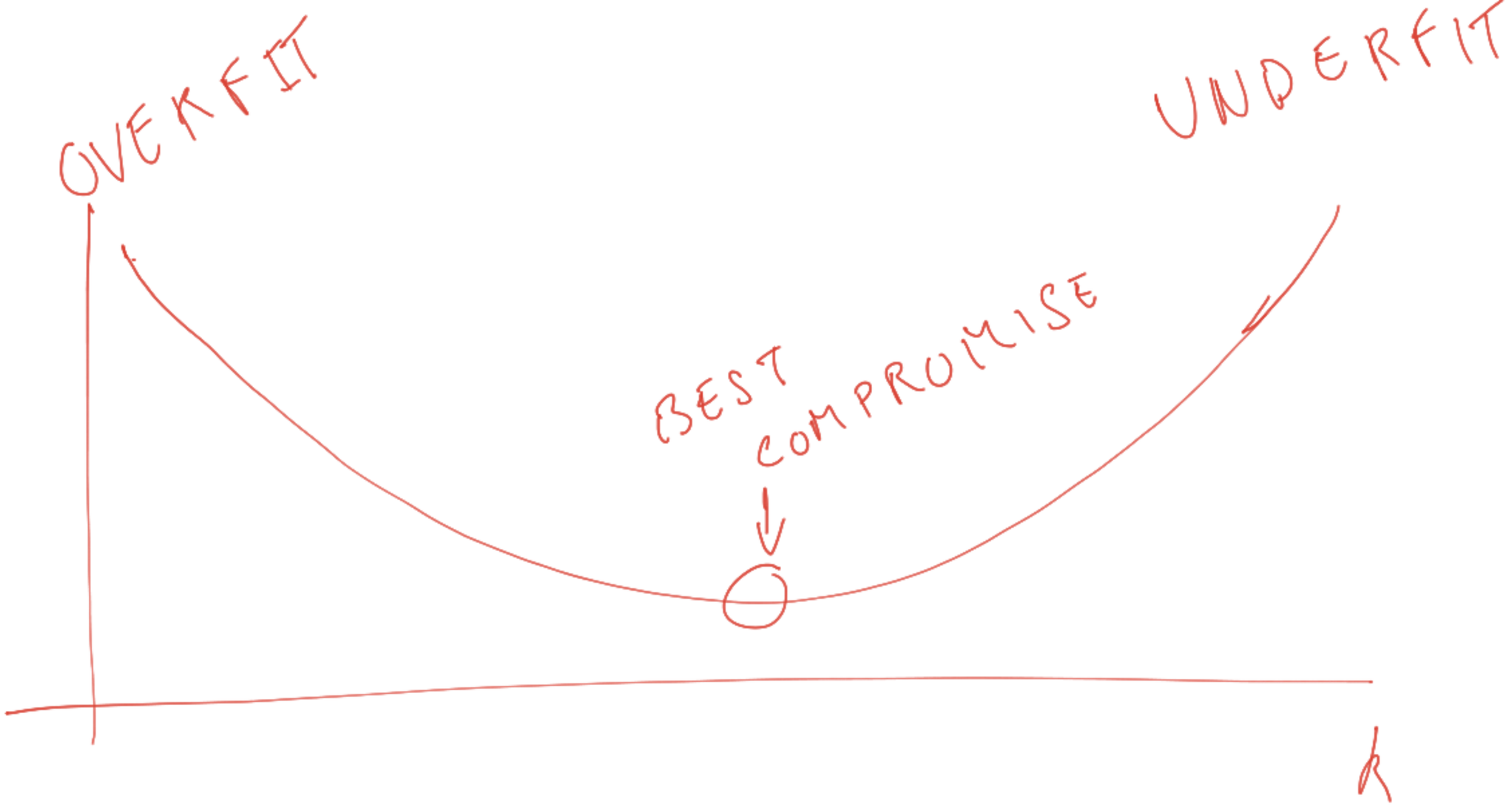
```
mean_absolute_error(y_test,y_predk)
```

```
# RMSE: k Nei
np.sqrt(mean_squared_error(y_test,y_predk))
```

```
from sklearn.metrics import mean_absolute_error
```

```
from sklearn.metrics import mean_squared_error
```


TEST
TRAIN
TEST
TRAIN
TEST
Iter 1
Iter 2
Iter 3

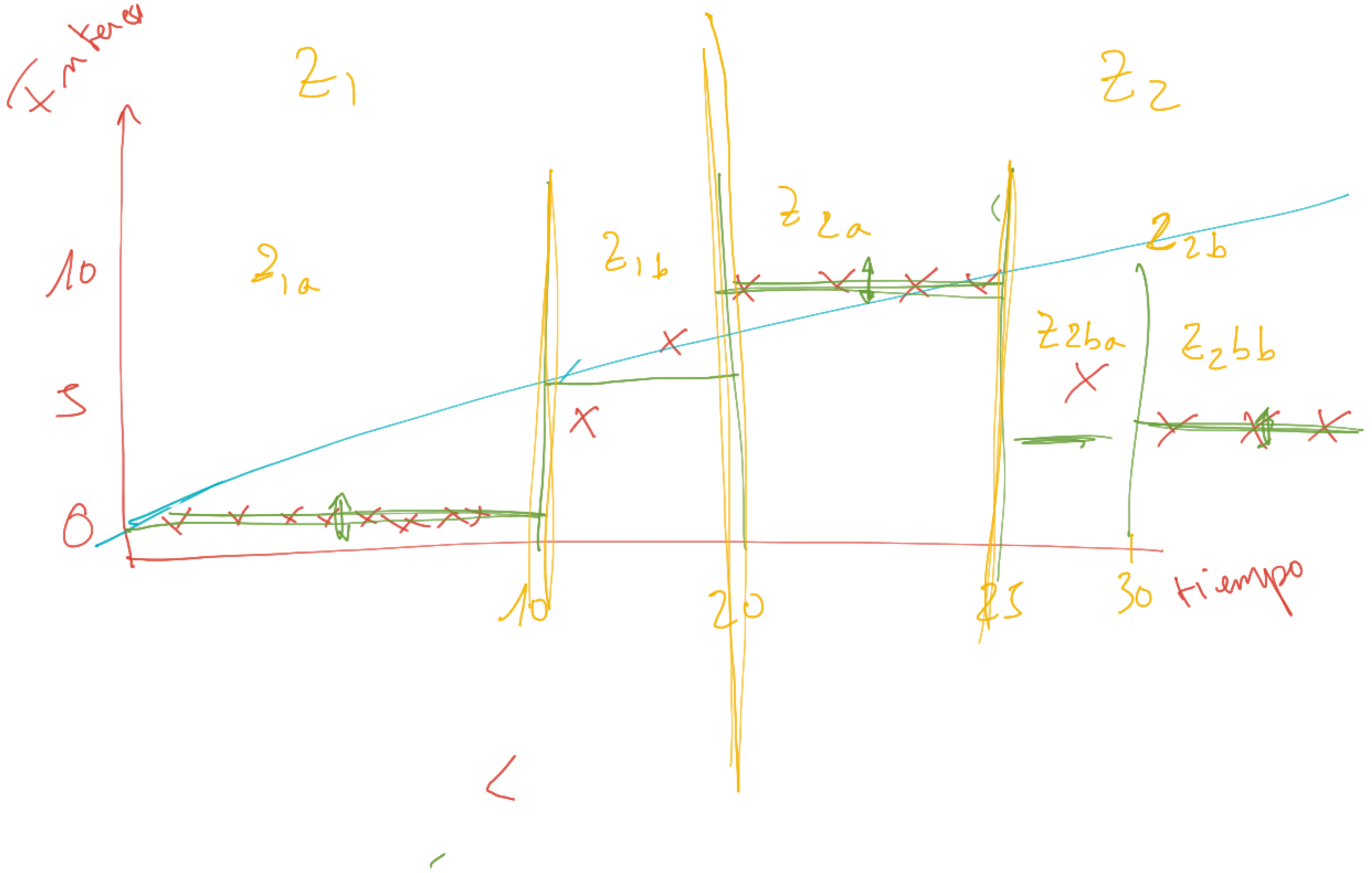

OVERFIT
UNDERFIT
BEST
COMPROMISE
k

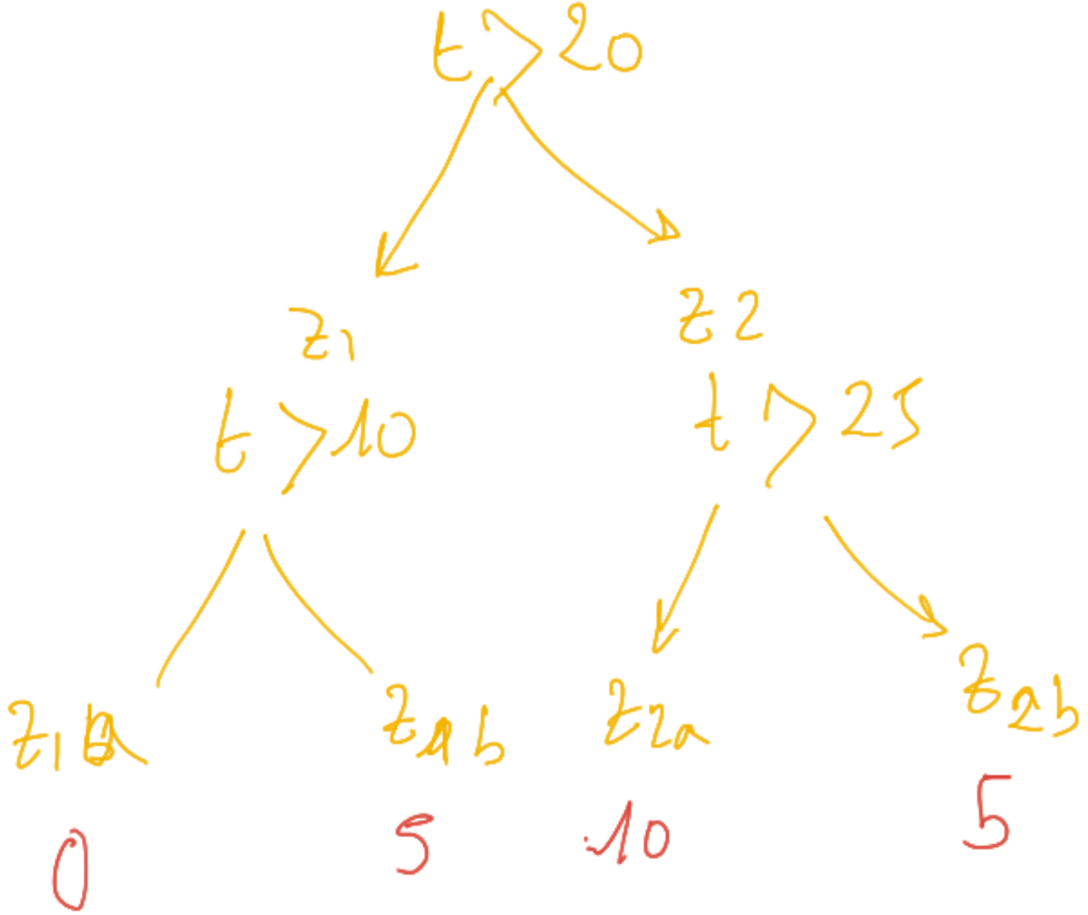
```python
from sklearn.model_selection import cross_val_score
# We calculate the metric for several subsets (determine by cv)
# With cv=5, we will have 5 results from 5 training/test
cross_val_score(reg,X,y,cv=5,scoring="neg_mean_squared_error")
```

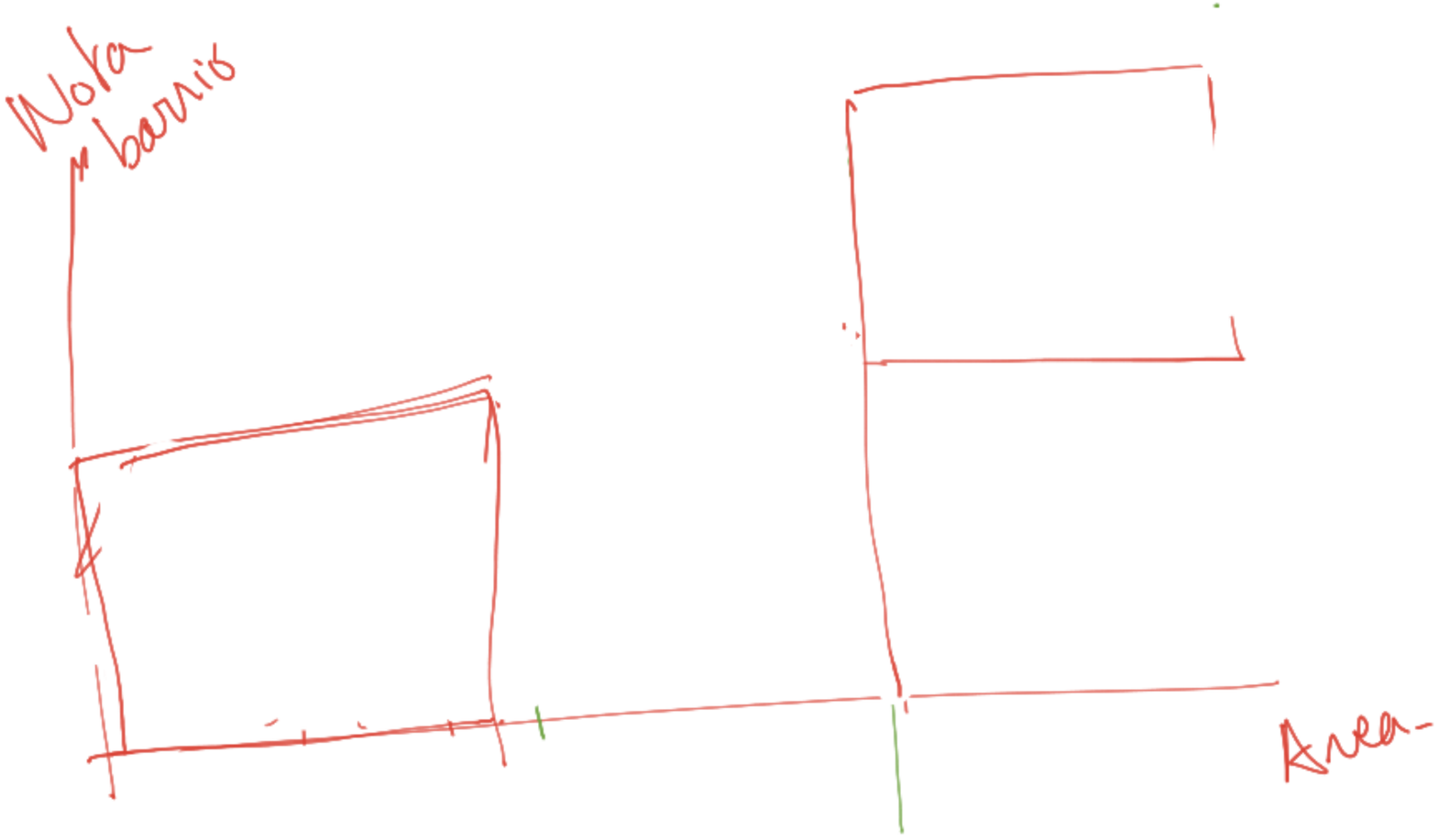
Evalua 1 modelo

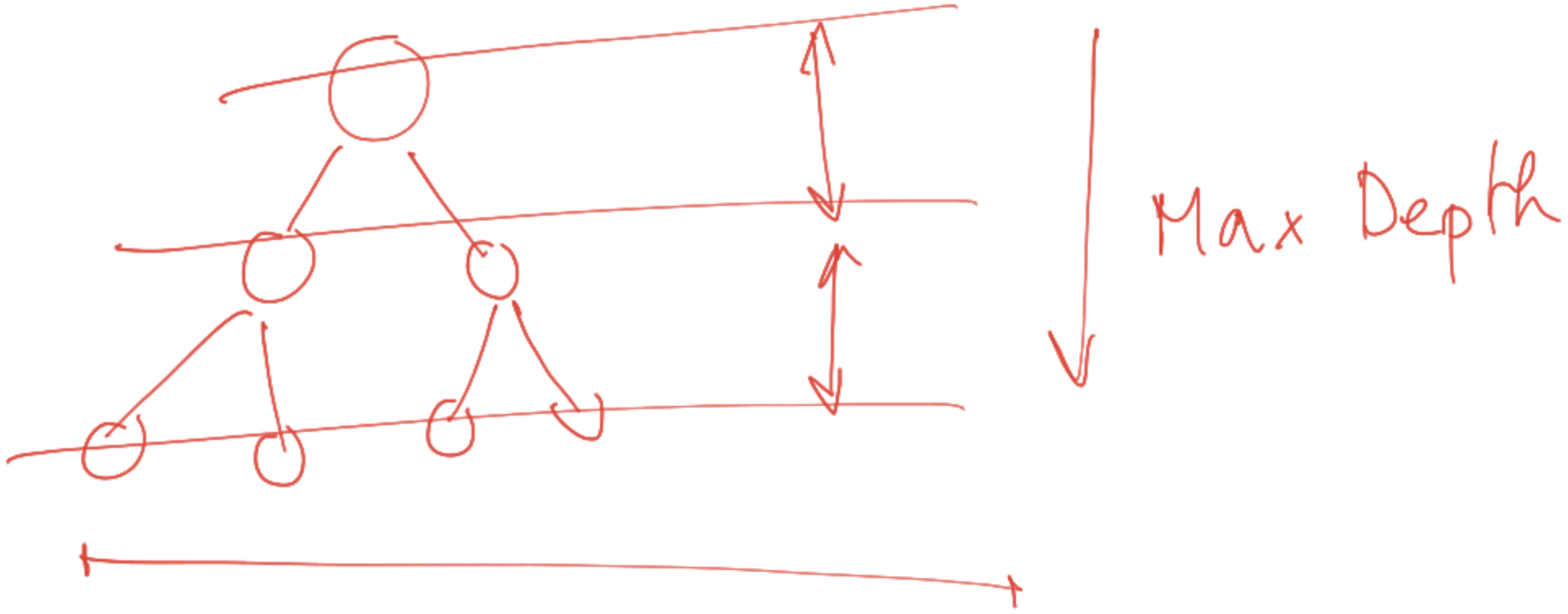
```python
from sklearn.model_selection import GridSearchCV
from sklearn.neighbors import KNeighborsRegressor
reg_test = GridSearchCV(KNeighborsRegressor(),
                        param_grid={"n_neighbors":np.arange(3,50)})
# Fit will test all of the combinations
reg_test.fit(X,y)
```

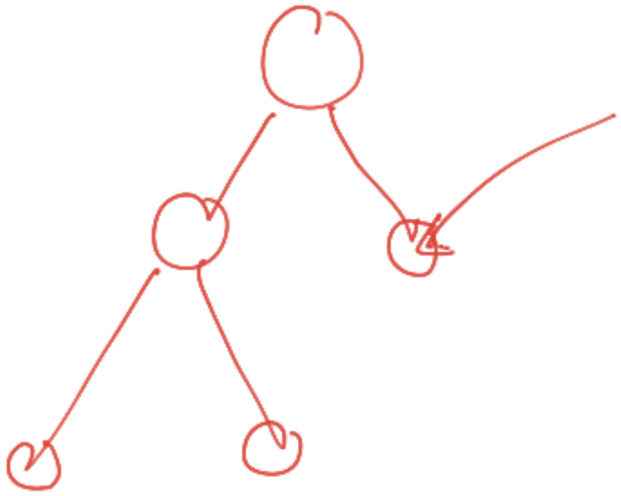
Evalua varios modelos y se queda el mejor.


Z1
Z2
10
5
0
Z1a
Z1b
Z2a
Z2b
Z2ba
Z2bb
10
20
25
30 tiempo


t > 20
z1
t > 10
z2
t > 25
z1a
z1b
z2a
z2b
0
5
10
5


Wola
barrio
Area-





num elem > min samples leaf.