INTRO TO REGRESION

Para una regresion simple con statsmodels →

```
from statsmodels.formula.api import ols
mod = ols('y ~ x', data=df).fit()
```
↳ Para ver parámetros → `mod.params`

Para hacer una prediccion → `mod.predict(nueva)`

Para ver los residuales → `mod.resid`

Para ver un resumen del modelo → `mod.summary()`

Suele ser necesario aplicar tranformaciones para mejora el acoplamiento lineal frente a Y.

Coeficiente de determinacion (R^2)

$1 \to$ Perfect fit
$0 \to$ Worst fit

Para velo, se ve en summary
↳ ó → `mod.rsquared`

Error residual estandar (RSE)

Es el error "típico" entre una prediccion y una observación.

$MSE = RSE^2$

Para velo → `np.sqrt(mod.mse_resid)`

`sns.residplot(x='...', y='...', data=..., lowess=True)`

Y(Residuals) - X(fitted Values) → Deben estar cercanos a 0 y normalmente distribuidos

Gráficos de posible interes
→ Q-Q plot (Veras si están normalmente distribuidos).
→ Scale-location plot

↳ `from statsmodels.api import qqplot`
`qqplot(data=mod.resid, fit=True, line='45')`

```
mod_norm = mod.get_influence().resid_studentized_internal
mod_norm = np.sqrt(np.abs(mod_norm))
sns.regplot(x=mod.fittedvalues, y=mod_norm, ci=None, lowess=True)
```

Leverage y Influence

Para velos →
```
sum = mod.get_influence().summary_frame()
leverage = sum['hat_diag']
cooks_di = sum['cooks_d']
```
→ Valores altos son peores actuan como outliers

INTRO TO REGRESION

Para una regresion logistica →

```
from statsmodels.formula.api import logit
mod = logit ('y ~ x', data:.. ).fit()
```

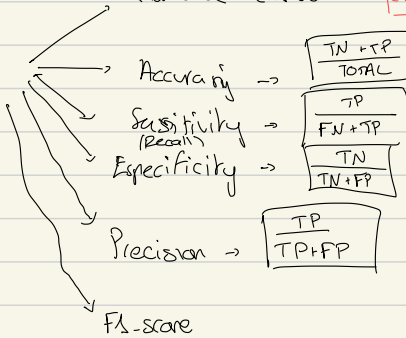Para ver hia un modelo logístico en un regplot → `logistic = True`

True negative	False positive
False negative	true positive

Odds_ratio →
$$\frac{\text{Probabilidad (real)}}{(1 - \text{Probabilidad})}$$

Matriz de confusion → `conf_mat = mod.pred_table()`

Cuantificado el fit

→ Accuracy → $\dfrac{TN + TP}{TOTAL}$

→ Susitivity → $\dfrac{TP}{FN + TP}$
   (Recall)

→ Especificity → $\dfrac{TN}{TN + FP}$

→ Cuanto mas alto mejor

Precision → $\dfrac{TP}{TP + FP}$

F1_score