Clasificación desbalanceada con Python: modificando datos y algoritmos

Raquel Pezoa Rivera

raquel.pezoa@uv.cl

Temas

- Clasificación
- Desbalance de Clases
- Ejemplos
- Python y desbalance

Mini Bio:

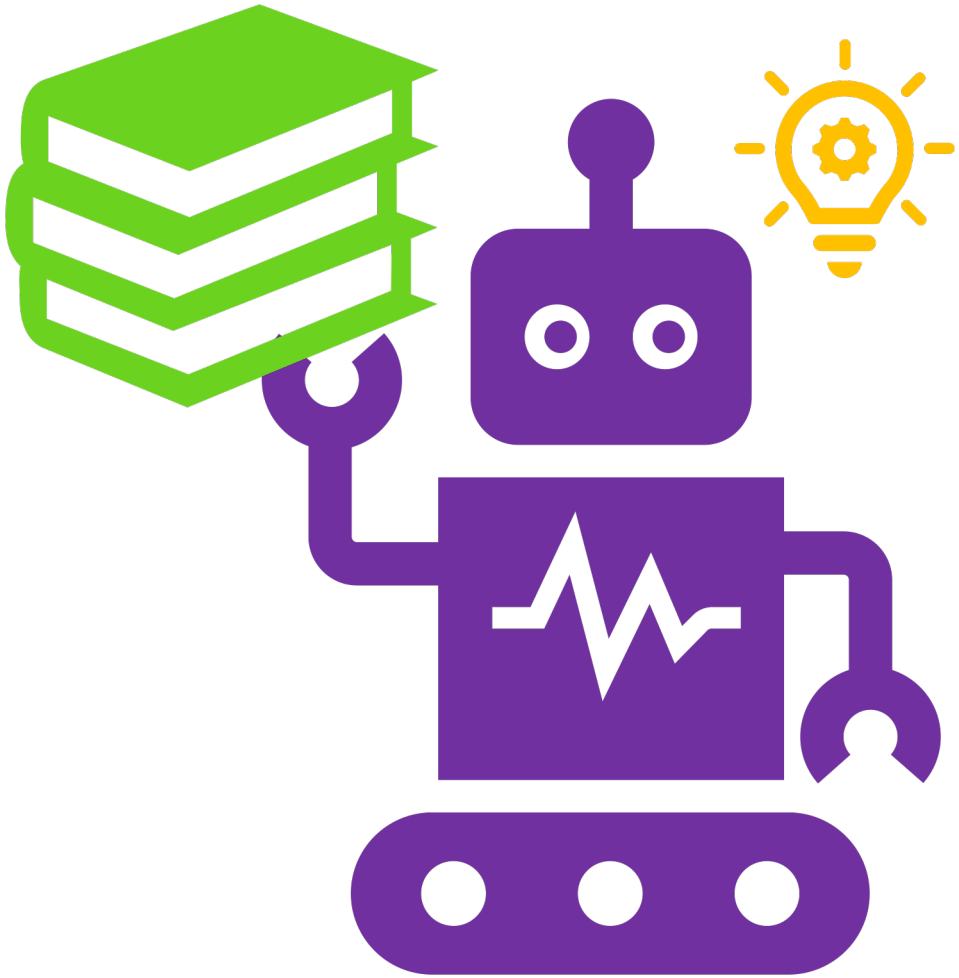
Raquel Pezoa Rivera

- Soy mamá: dos niños ❤️ ❤️
- Informática: Ingeniera, Magíster, Doctora
- Académica: Profesora Adjunta Escuela Ing. Informática UV
- Investigadora Asociada del CCTVal-UTFSM
- Trabajo en aprendizaje automático, usando 

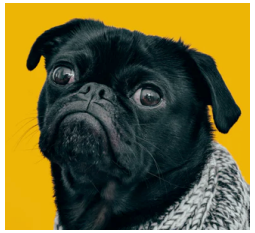
Aprendizaje Automático

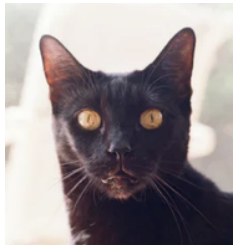
Generación de algoritmos computacionales que mejoran automáticamente desde la experiencia.

datos



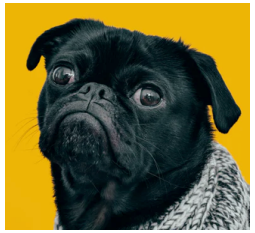
Clasificación

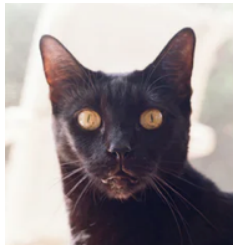




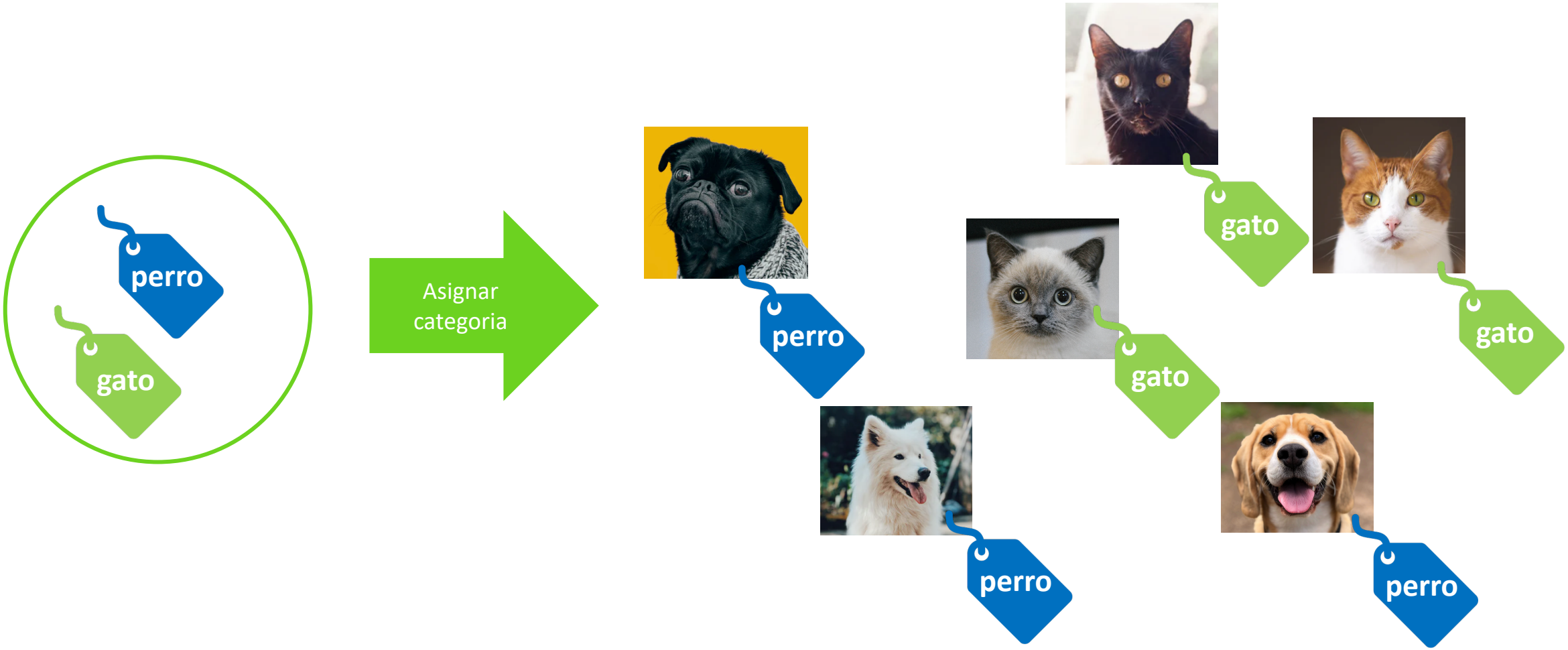
Clasificación

Asignar categoria



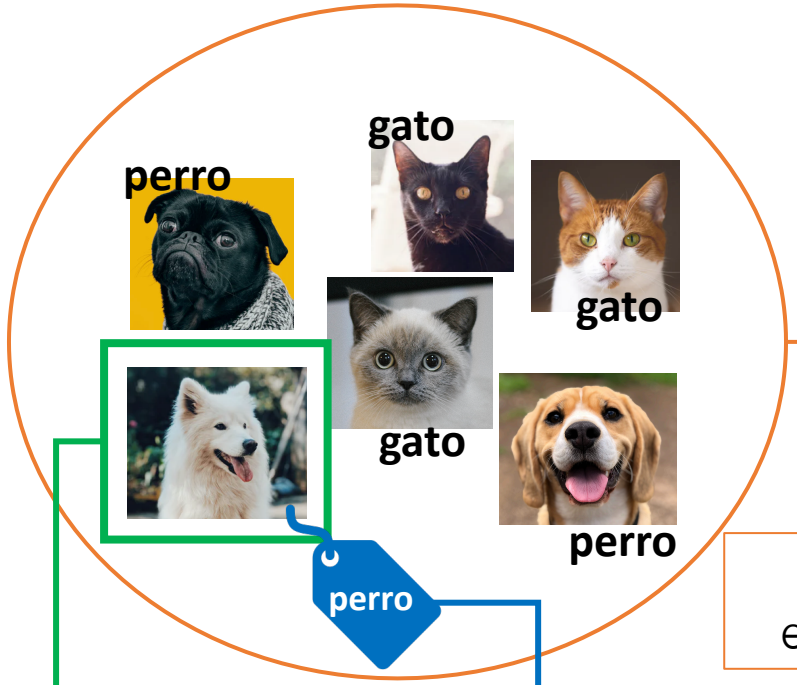


Clasificación



Clasificación

enfoque supervisado



Datos de entrenamiento

$x \in \mathbb{R}^n$

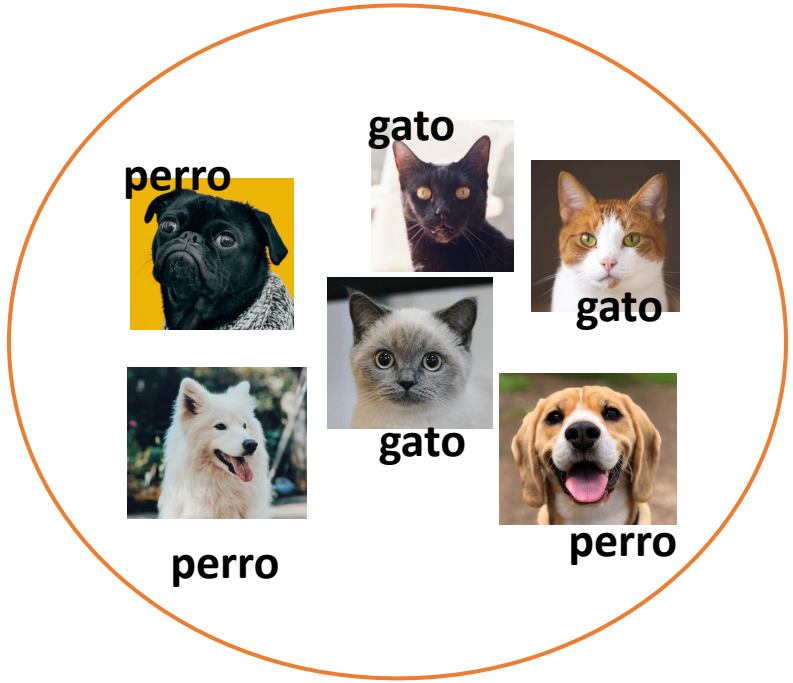
$y \in \{0, 1\}$

Entrenamiento

Modelo de clasificación

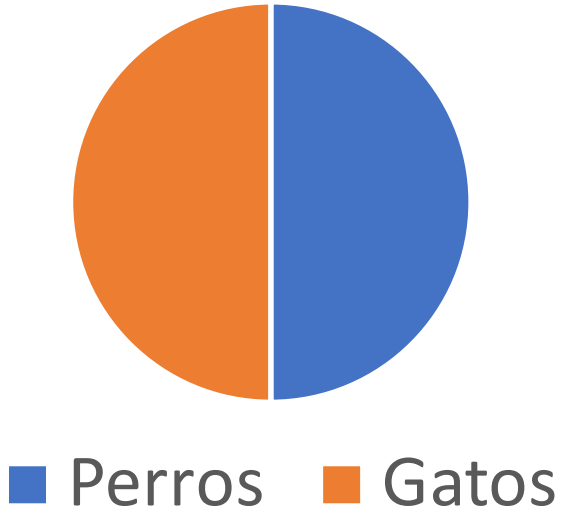
$f(x) = y$

Clasificación

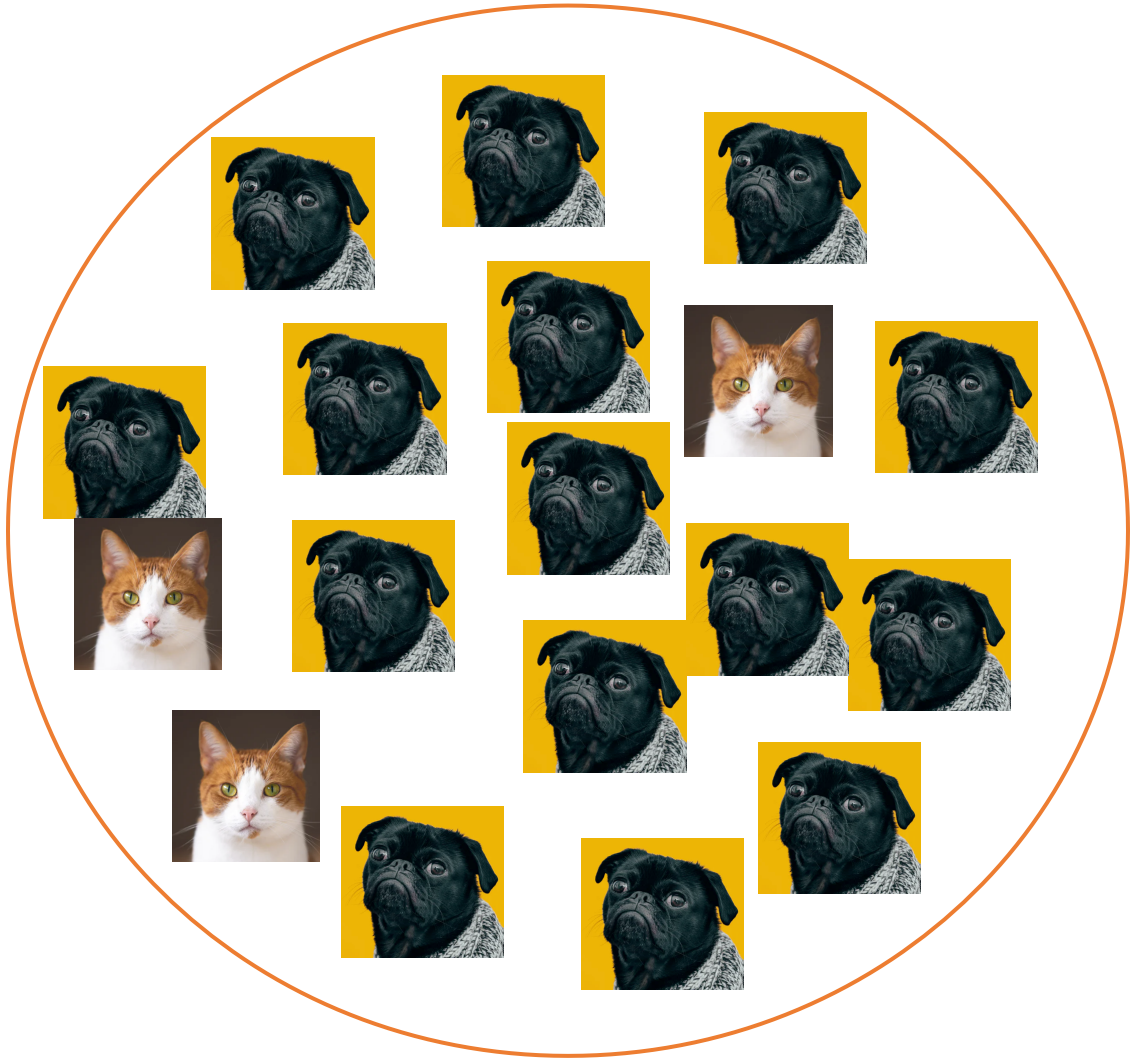


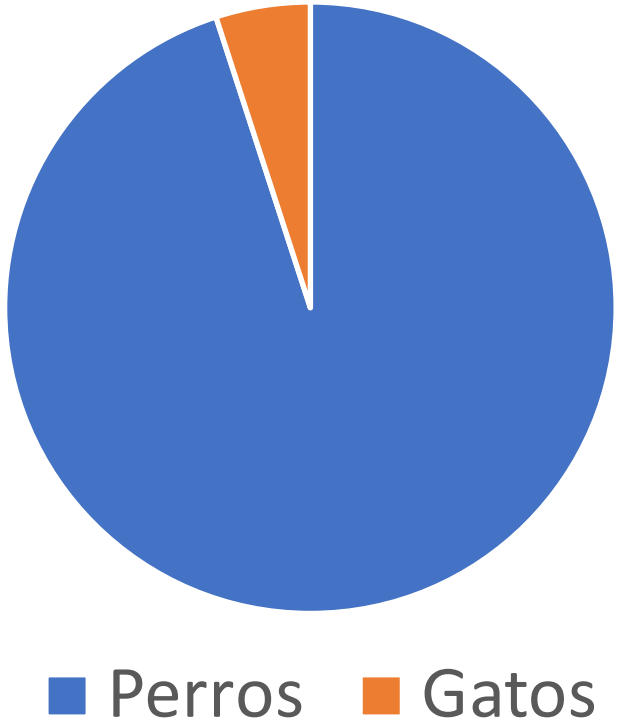
3 datos → clase perro

3 datos → clase gato



Clasificación





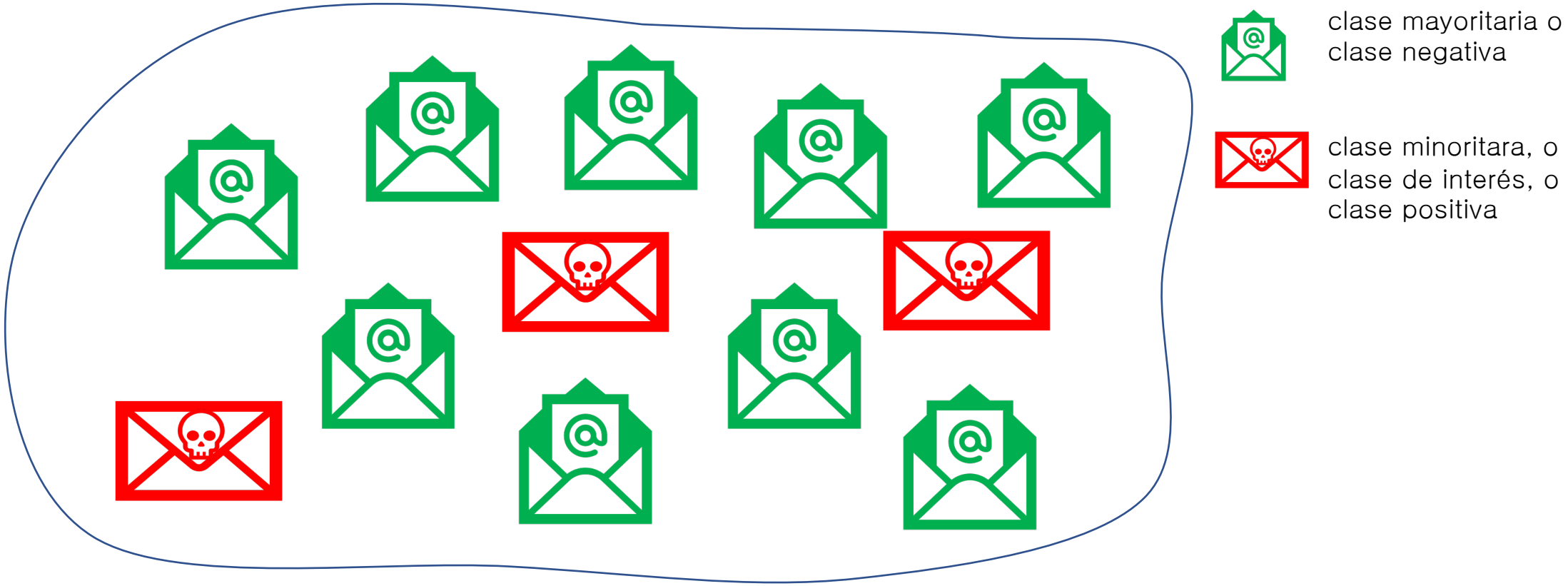
Desbalance de Clases

En clasificación (binaria), el desbalance de clases ocurre cuando los datos una clase están subrepresentados en relación a la otra clase.

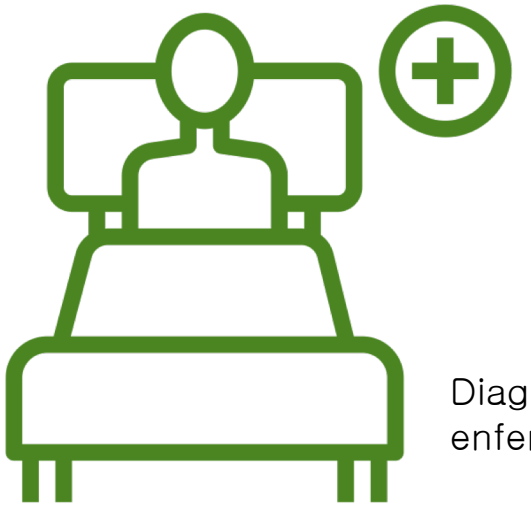
clase minoritara = clase positiva
clase mayoritaria = clase negativa

Ejemplo

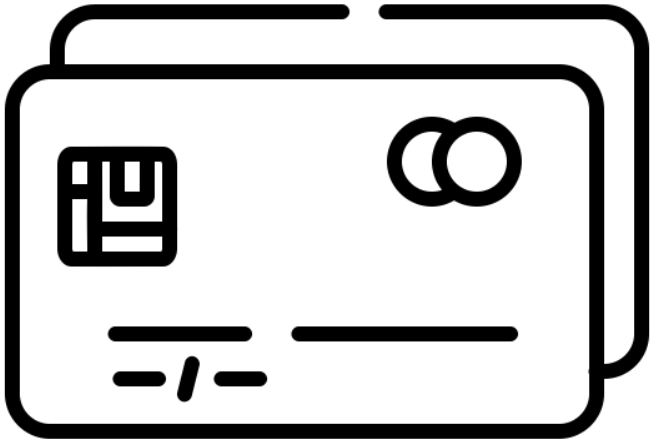
- Desbalance de clases en clasificación de emails: *spam* o *no-spam*



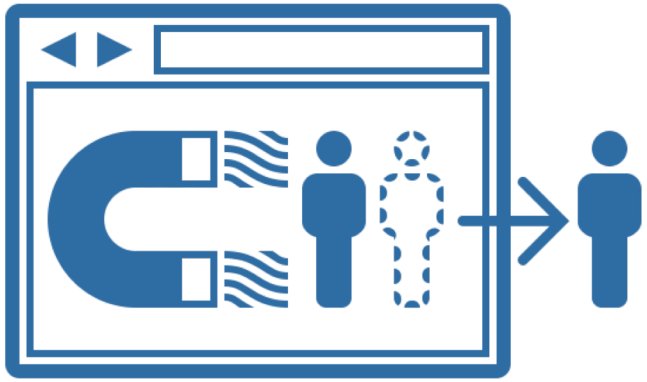
Problemas reales



Diagnóstico de enfermedades

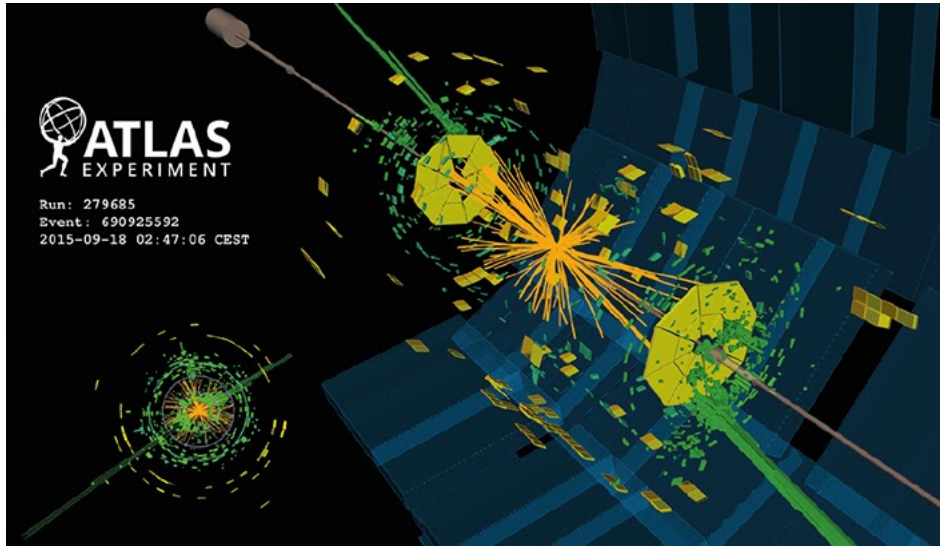


Transacciones bancarias fraudulentas



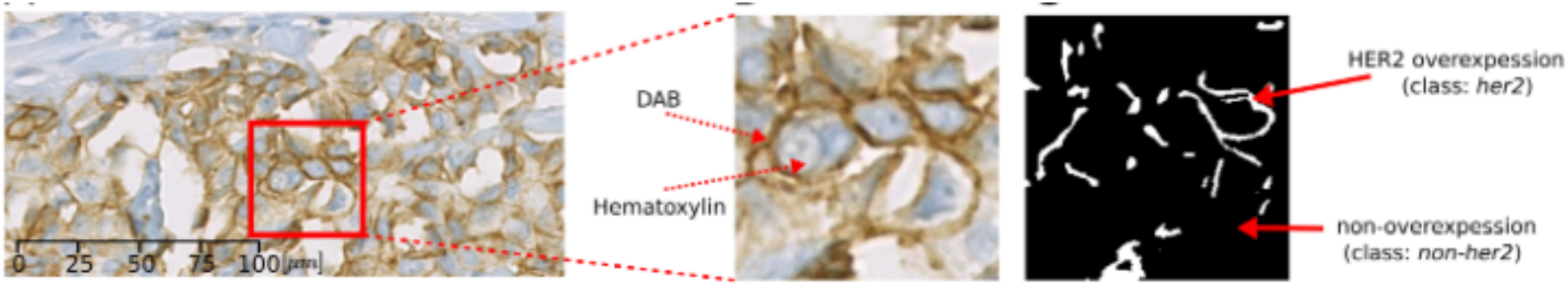
Predicción de abandono de los clientes

Problemas reales



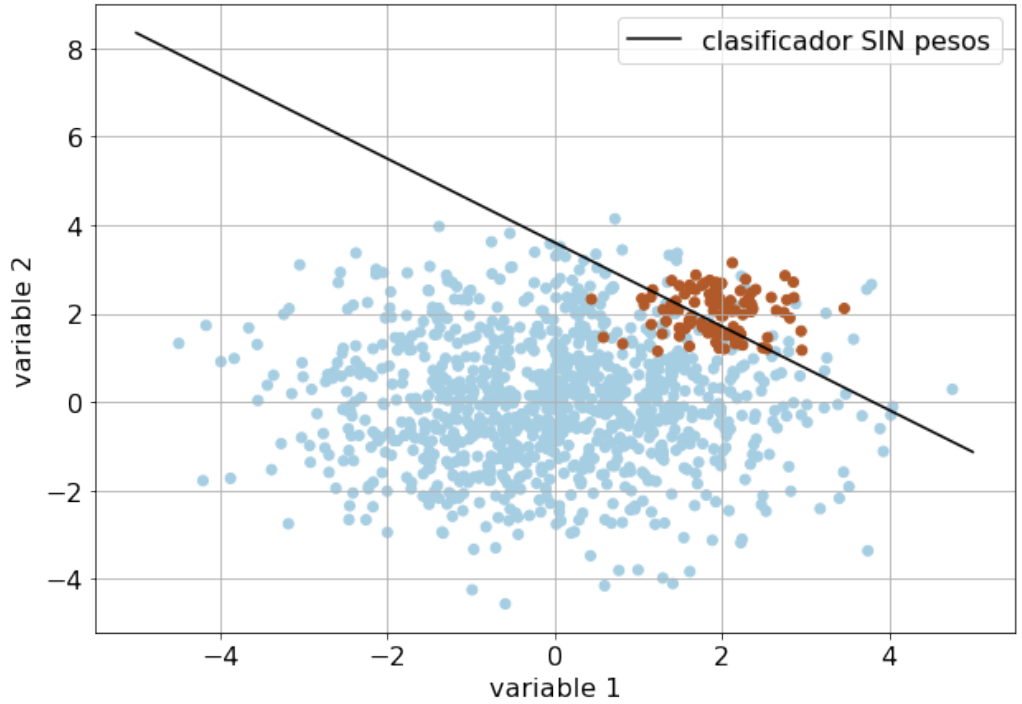
CLASIFICACIÓN DE EVENTOS DE FÍSICA DE PARTÍCULAS

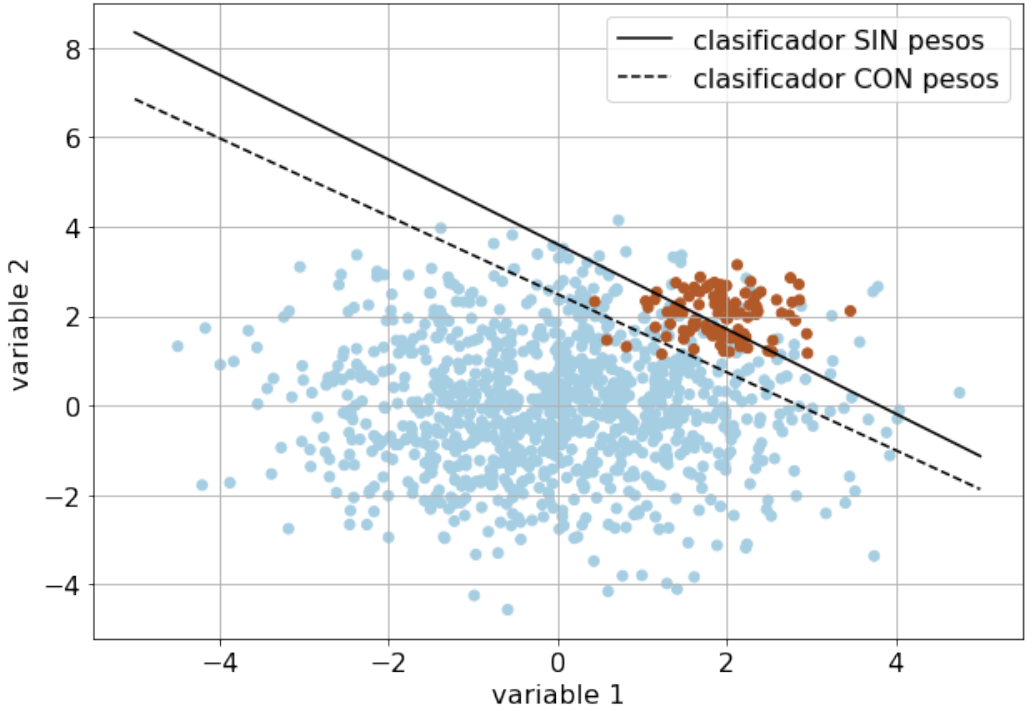
Boson de Higgs → 1 en 10^{13} colisiones



CLASIFICACIÓN DE PIXELES HER2

¿Clasificador con desbalance?

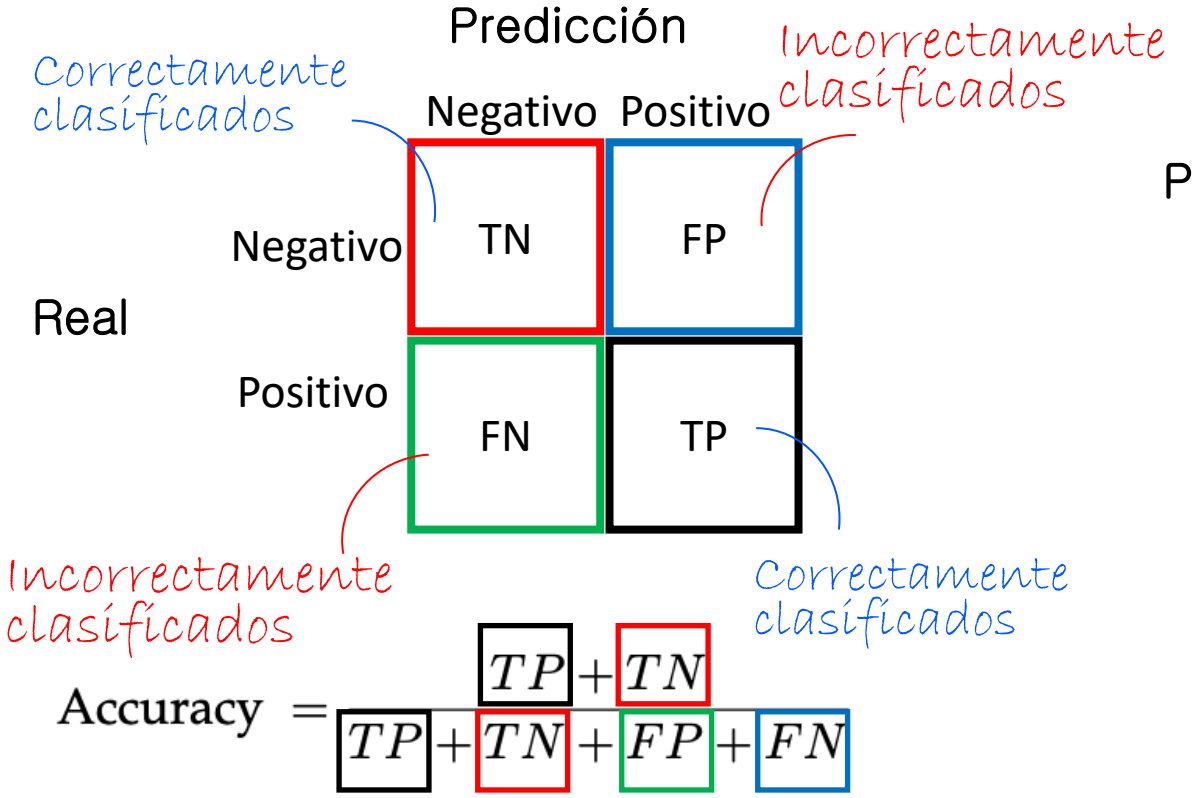




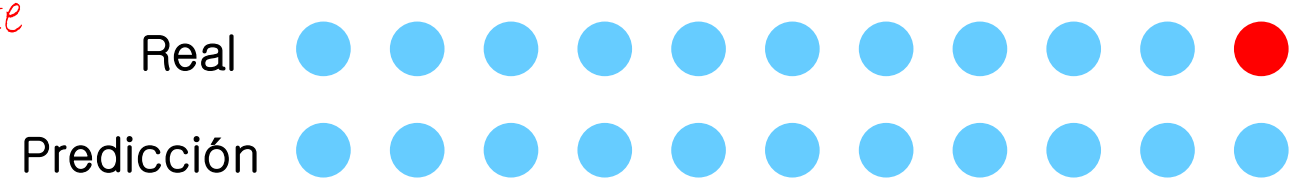
Ejemplo de:
https://scikit-learn.org/0.18/auto_examples/svm/plot_separating_hyperplane_unbalanced.html

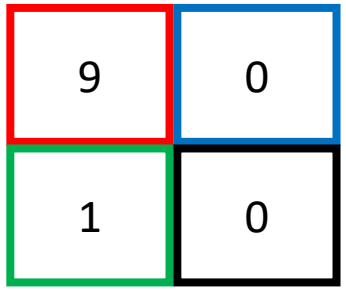
¿Métricas de desempeño?

MATRIZ DE CONFUSIÓN



$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$



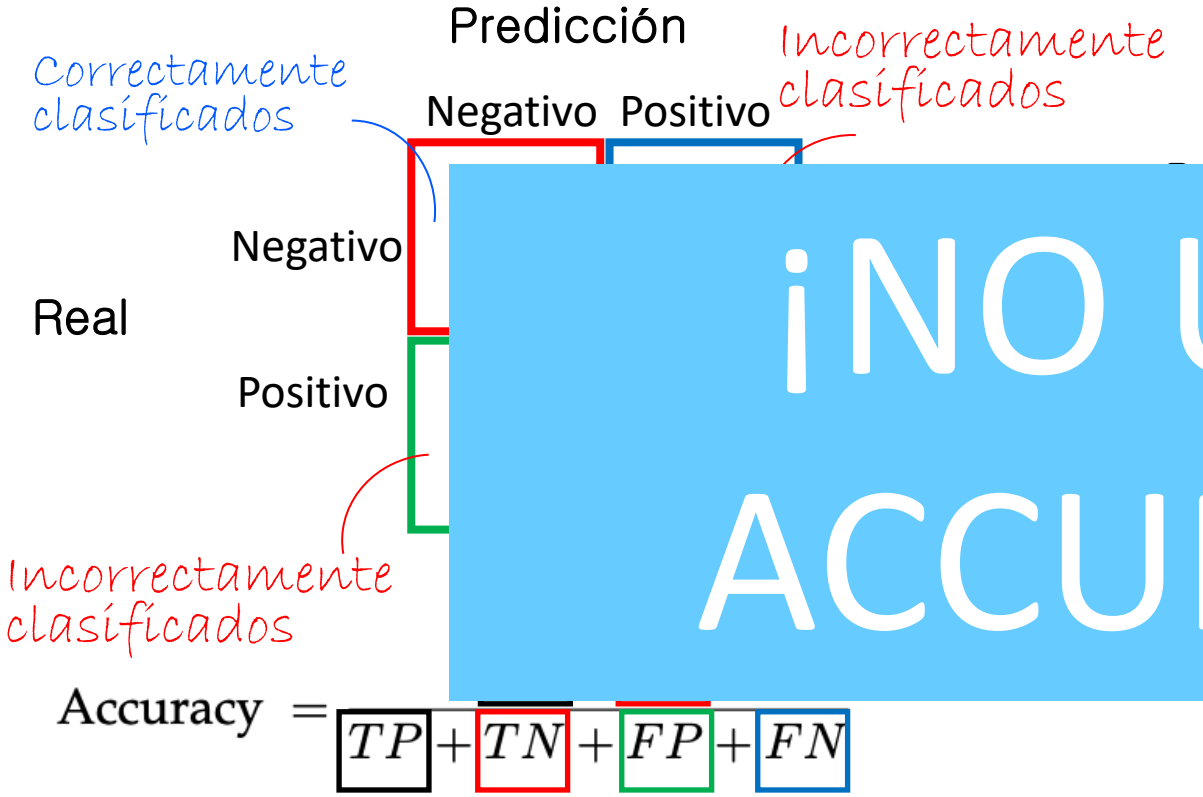


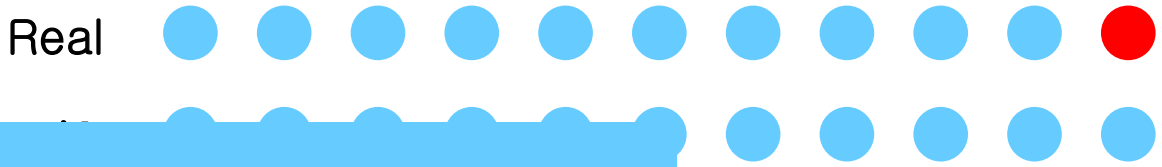
$$\text{Accuracy} = \frac{0 + 9}{0 + 9 + 0 + 1} = 0.9$$

¡MÉTRICAS NO REPRESENTATIVAS!

¿Métricas de desempeño?

MATRIZ DE CONFUSIÓN





$$\frac{}{0 + 9 + 0 + 1} = 0.9$$

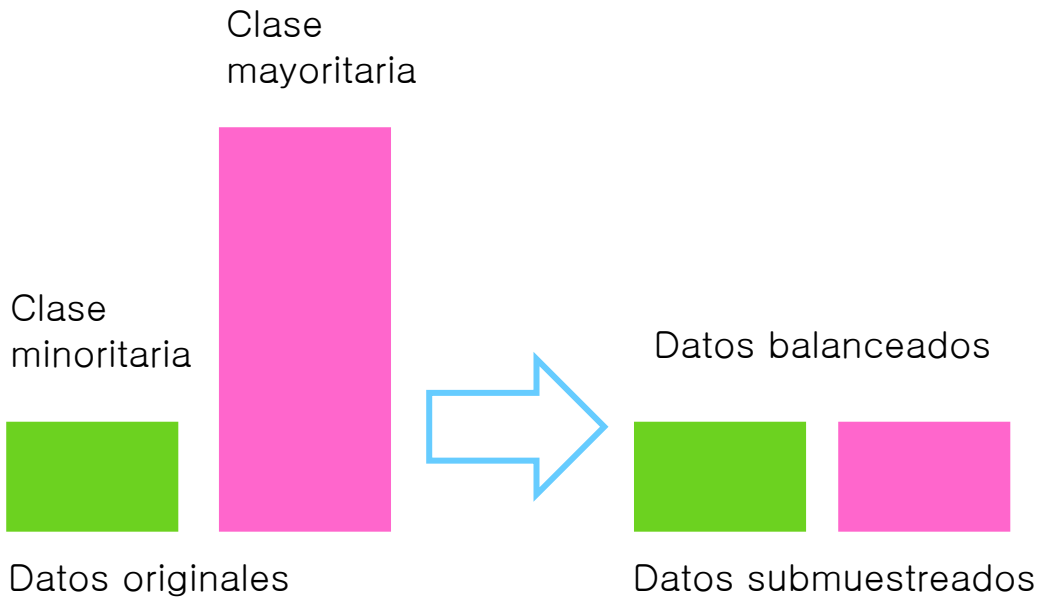
¡NO USAR ACCURACY!

¡MÉTRICAS NO REPRESENTATIVAS!

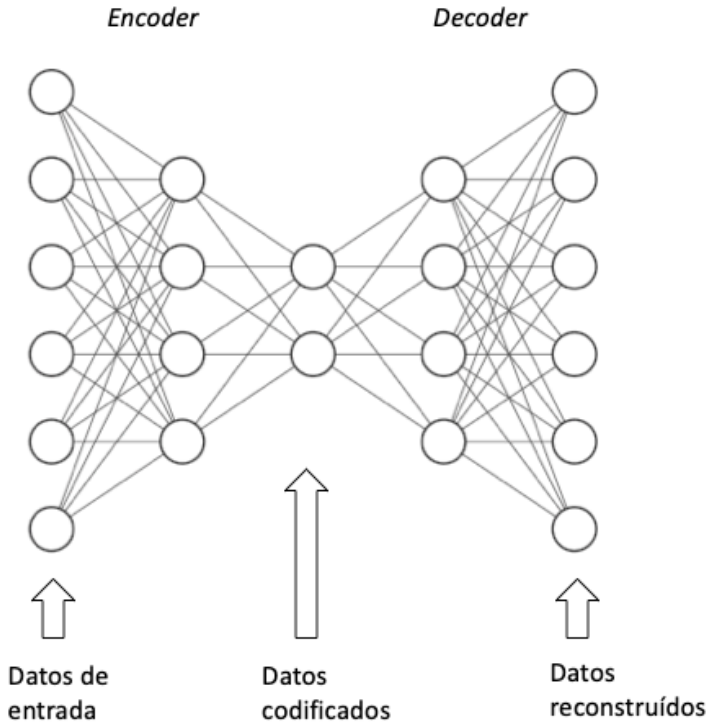
Enfoques para abordar desbalance

2 enfoques principales:

- Nivel de datos



- Nivel del algoritmo

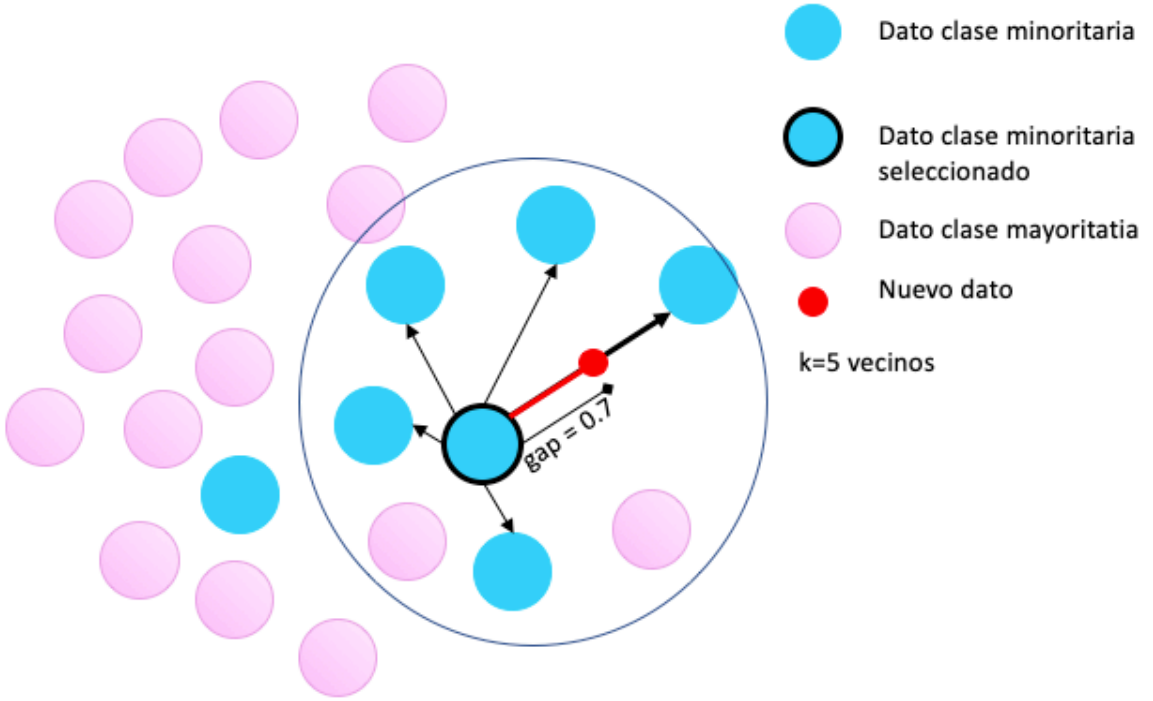


Nivel de datos

- Modificación de la distribución de los datos de entrenamiento:

(i) Oversampling

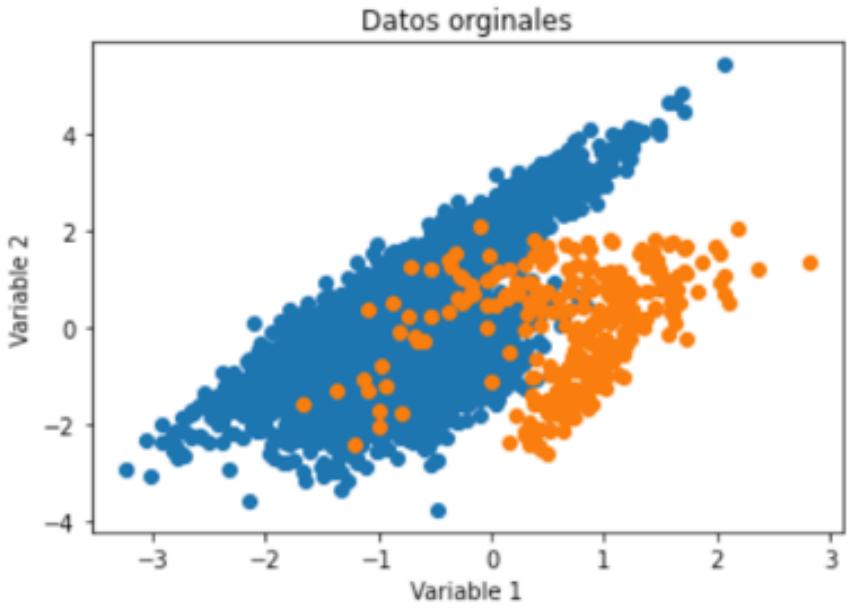
Random Oversampling ,SMOTE, ADASYN, etc.

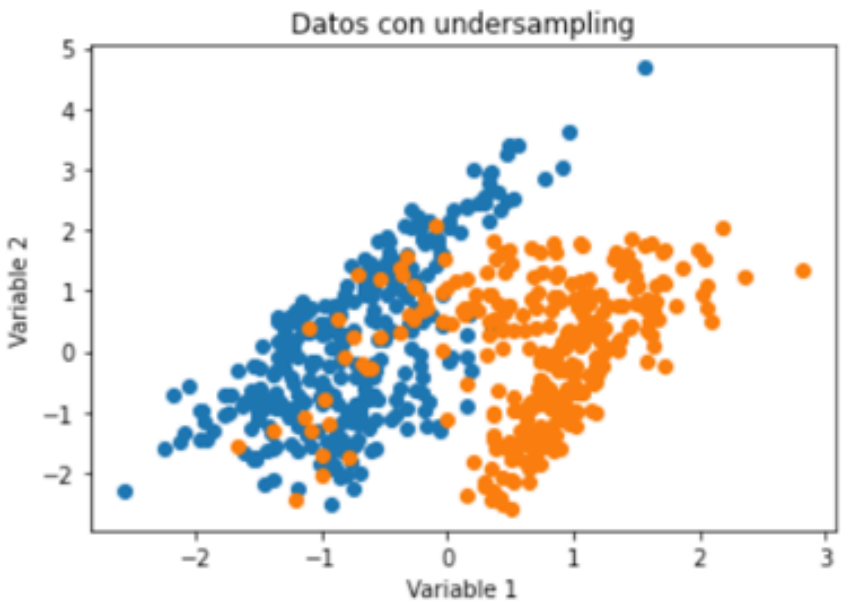


SMOTE: Proceso iterativo que crea datos sintéticos, usando enfoque de k-vecinos más cercanos, y selección aletoria

Nivel de datos

(ii) Undersampling: Random undersampling

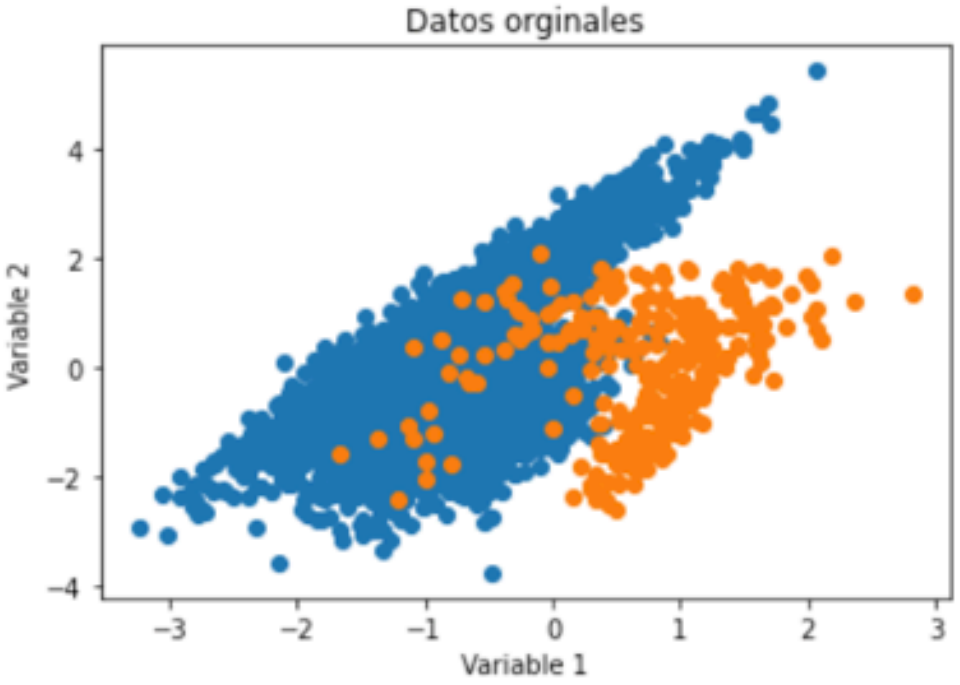


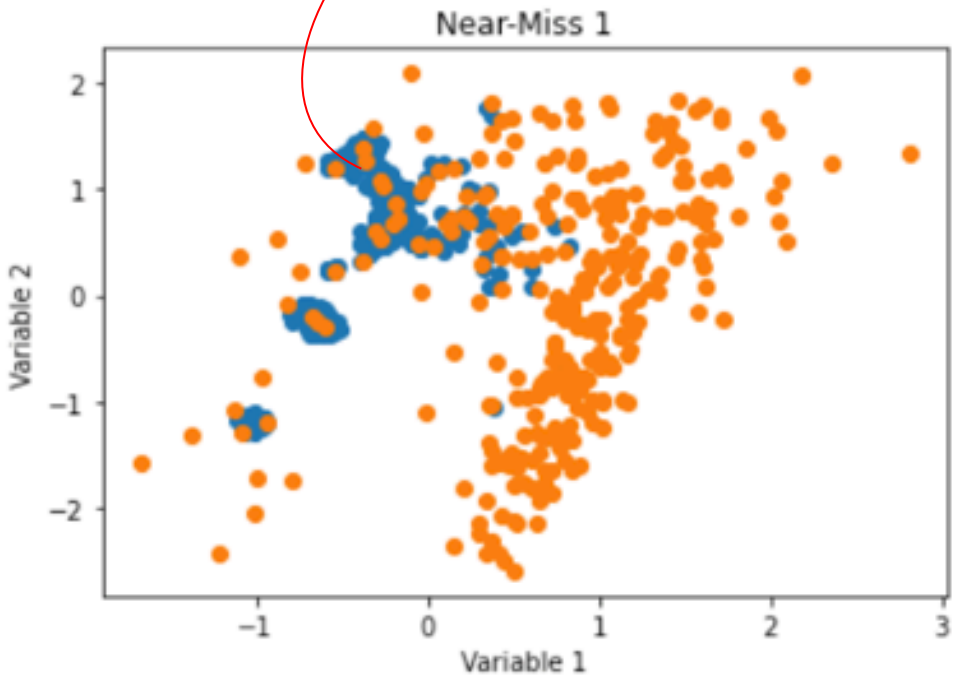


Nivel de datos:

se mantienen los más cercanos a datos positivos

Near-Miss 1



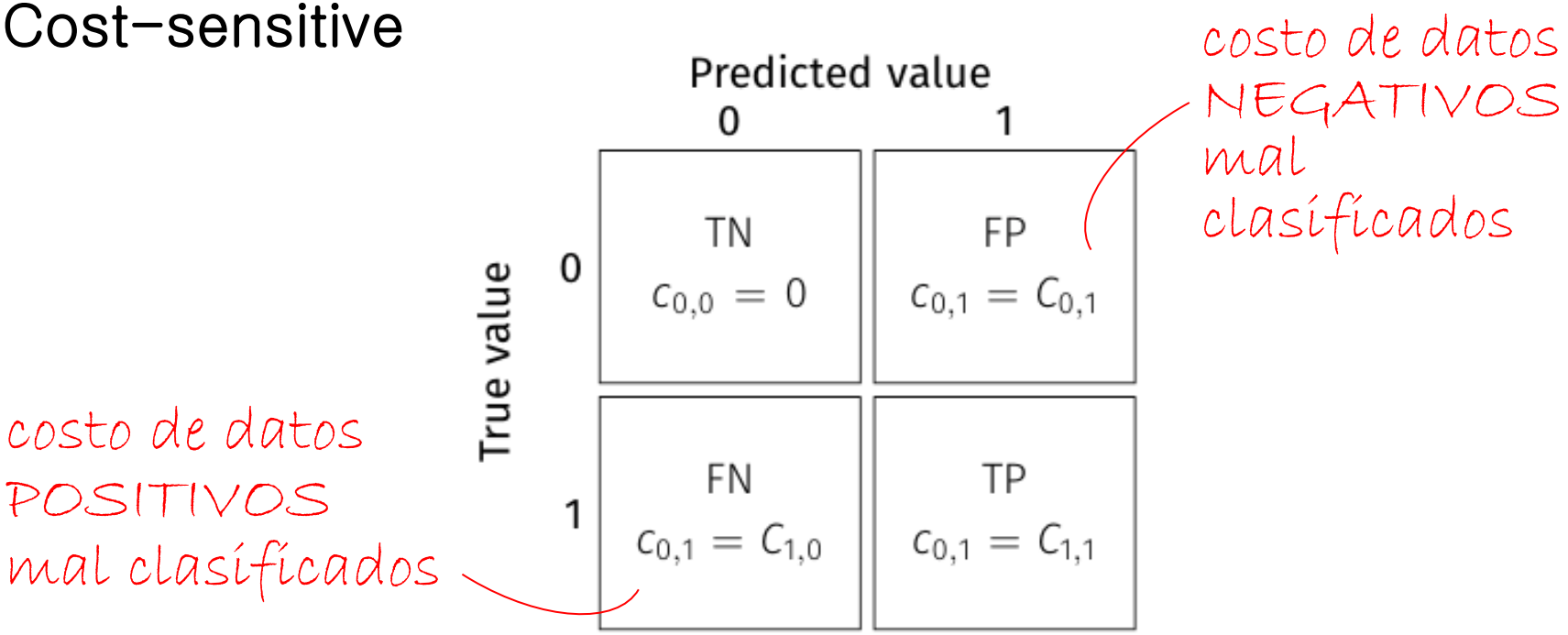


Revisar:

- Learning from imbalanced data. H He, EA Garcia – IEEE Transactions on knowledge and data engineering, 2009
- Fernández, Alberto, et al. Learning from imbalanced data sets. Vol. 10. Berlin: Springer, 2018.
- https://imbalanced-learn.org/stable/user_guide.html

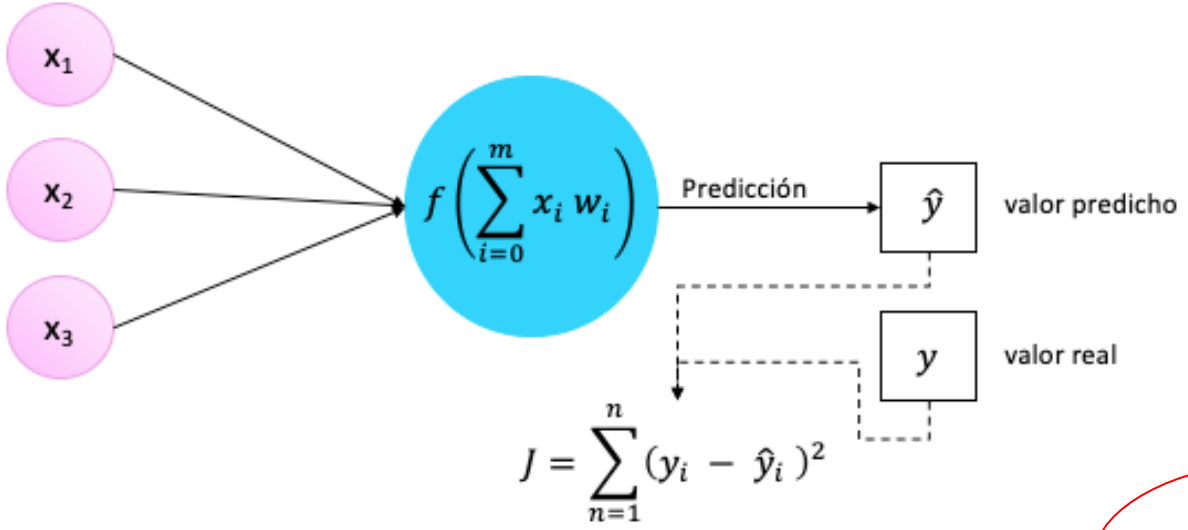
Nivel del algoritmo

- El **proceso entrenamiento** se ajusta de manera de dar más importancia a la clase minoritaria, se modifican algoritmos existentes o creación de nuevas arquitecturas .
  1. Cost-sensitive



Nivel del algoritmo

2. Modificación de funciones de pérdida



predicción del modelo

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{otherwise,} \end{cases}$$

cross entropy

$$\mathrm{CE}(p, y) = \mathrm{CE}(p_t) = -\log(p_t)$$

balanced cross entropy

$$\mathrm{CE}(p_t) = -\alpha_t \log(p_t)$$

Funciones de pérdida

$$\mathrm{FL}(p_t) = -\alpha_t (1 - p_t)^{\gamma} \log(p_t)$$

focal loss

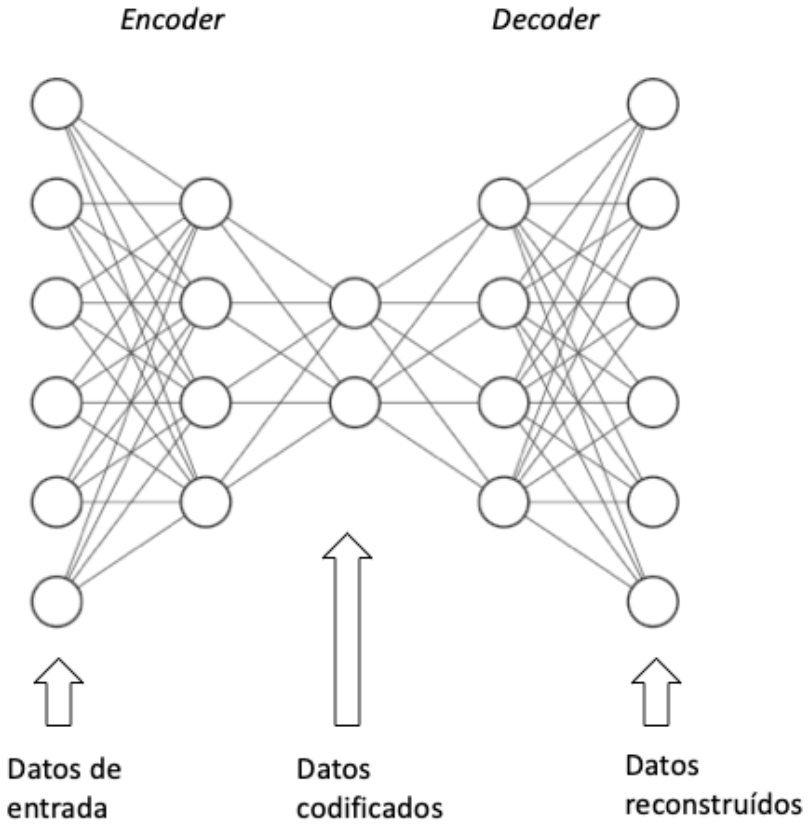
Lin, Tsung-Yi, et al. "Focal loss for dense object detection." *Proceedings of the IEEE international conference on computer vision*. 2017.

Nivel del algoritmo

3. Arquitecturas específicas:

por ejemplo, *Autoencoders,*

un tipo de red neuronal que copia los valores de entrada a los valores de salida.



Python y Desbalance de clases: imbalanced-learn

- Instalación simple

con pip

```
pip install -U imbalanced-learn
```

https://imbalanced-learn.org

o conda

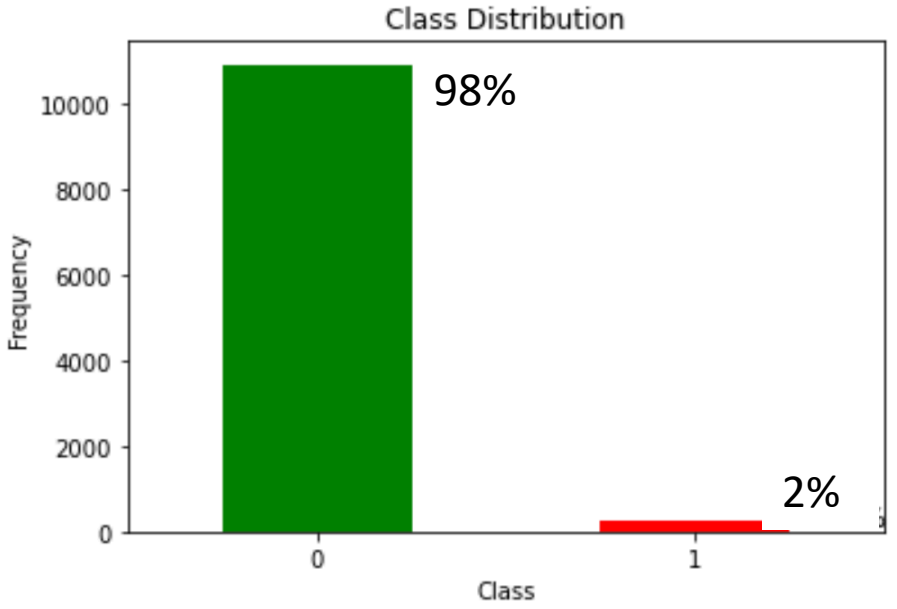
```
conda install -c conda-forge imbalanced-learn
```

Ejemplo 1: Mamography dataset (1/4)

```
import pandas as pd
```

```
df = pd.read_csv("../data/mammography.csv", header=None)
```

	Area_Obj	Avg_Gray	Grad	RMS	Cont	Momentum	Class
0	0.230020	5.072578	-0.276061	0.832444	-0.377866	0.480322	0
1	0.155491	-0.169390	0.670652	-0.859553	-0.377866	-0.945723	0
2	-0.784415	-0.443654	5.674705	-0.859553	-0.377866	-0.945723	0
3	0.546088	0.131415	-0.456387	-0.859553	-0.377866	-0.945723	0
4	-0.102987	-0.394994	-0.140816	0.979703	-0.377866	1.013566	0
...	...	...	...	...	...	...	...



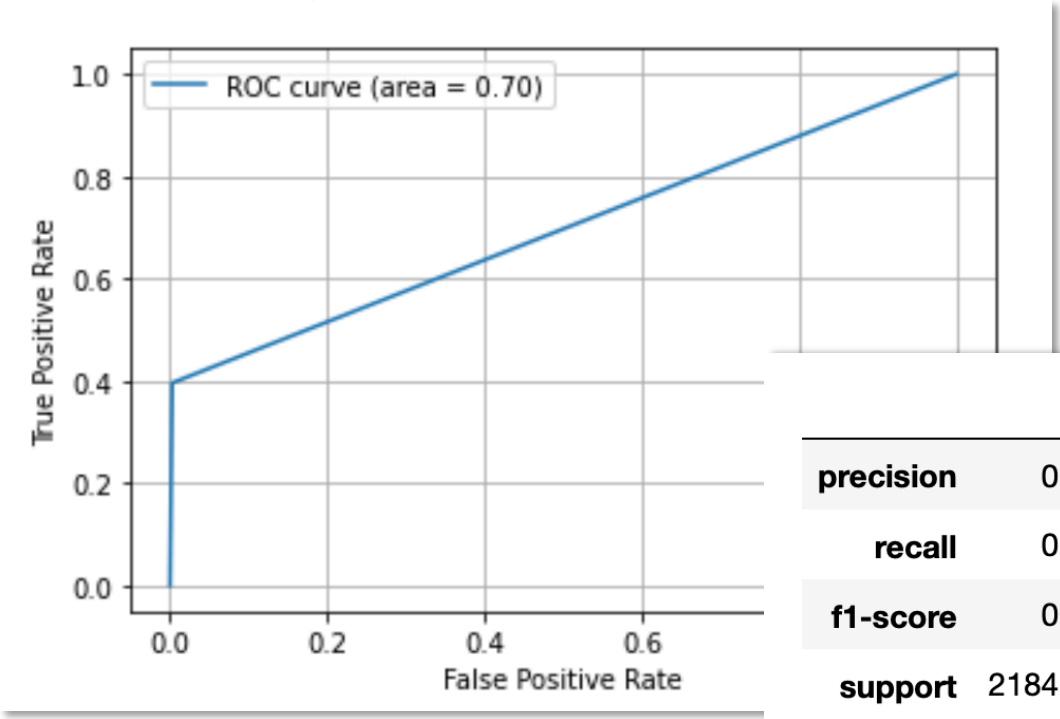

```
0    10923
1      260
Name: Class, dtype: int64
```

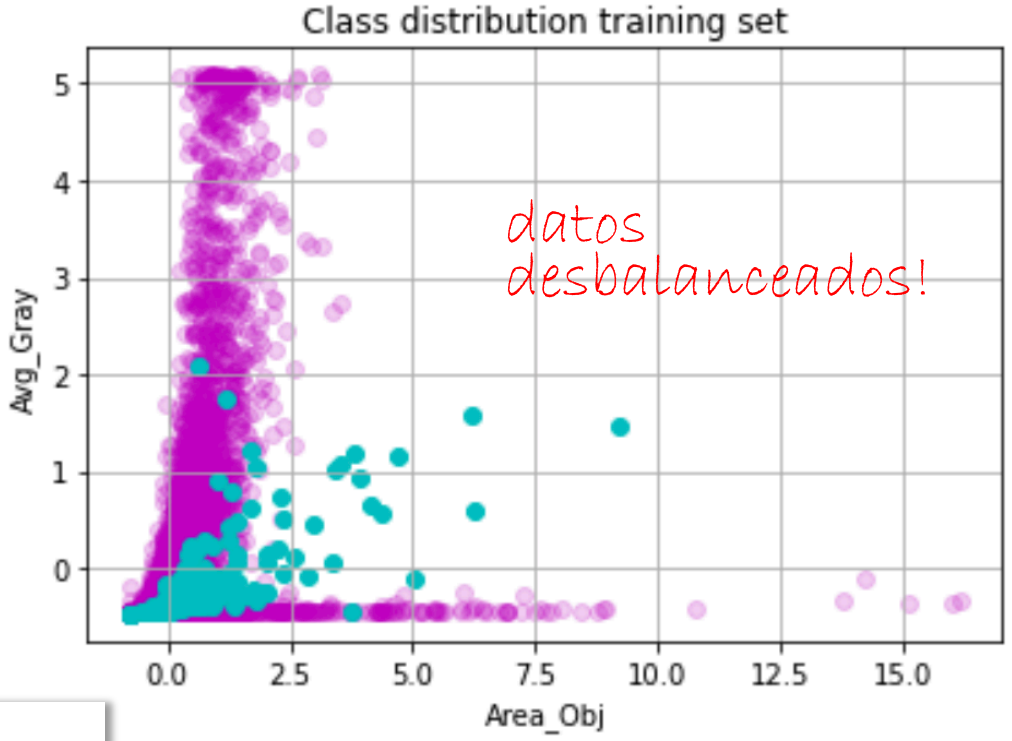
- Problema: detectar microcalcificaciones
- Clases:
  - con-microcalcificaciones → 1
  - sin-microcalcificaciones → 0

Ejemplo 1: Mamography dataset (2/4)

```
X_train, X_test, y_train, y_test = train_test_split(X,y,
                                   test_size=0.2, random_state=0)
```

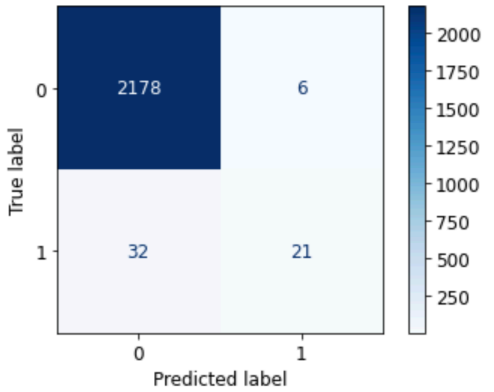
```
def classif_model(X_train, y_train):
    clf = GradientBoostingClassifier(n_estimators=100, max_depth=1)
    clf.fit(X_train, y_train)
    return clf
```





	0	1	accuracy
precision	0.985520	0.777778	0.983013
recall	0.997253	0.396226	0.983013
f1-score	0.991352	0.525000	0.983013
support	2184.000000	53.000000	0.983013

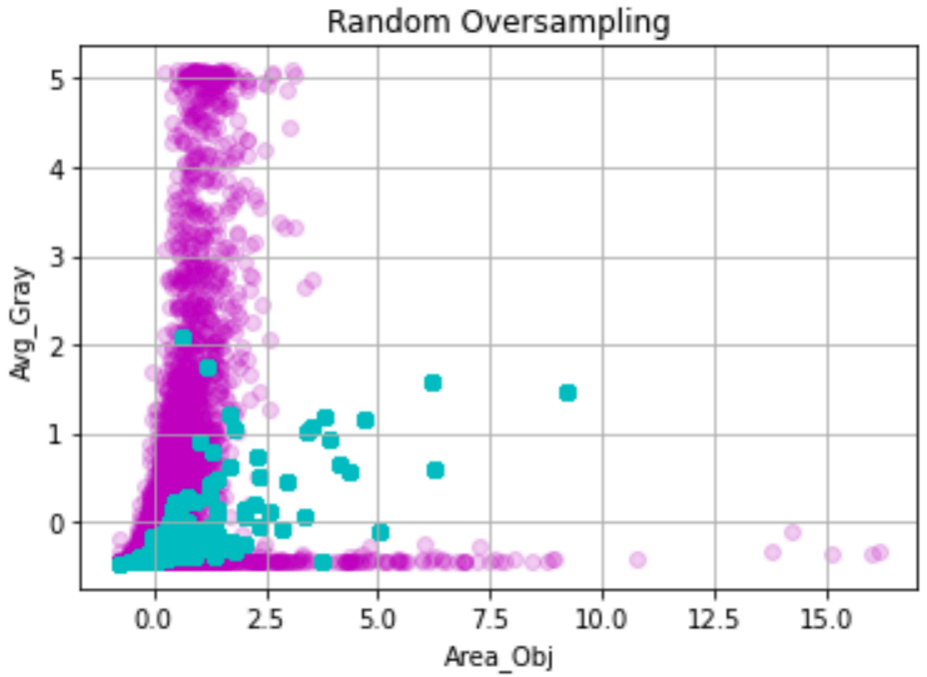
métricas clase de interés

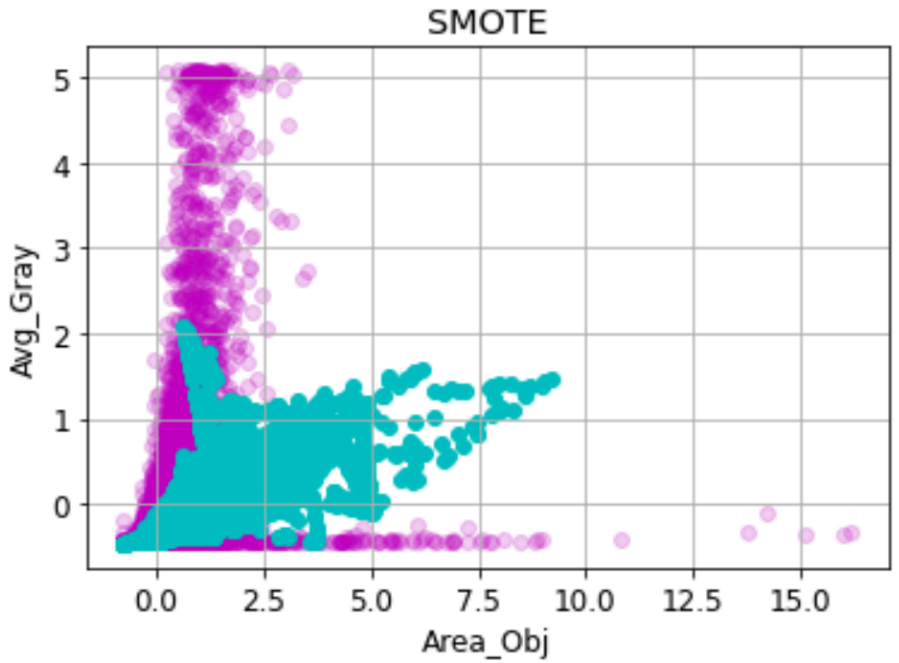


Ejemplo 1: Mamography dataset (3/4)

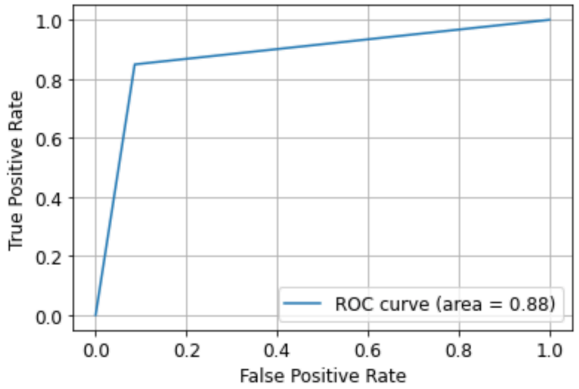
```python
from imblearn.over_sampling import RandomOverSampler
ros = RandomOverSampler(random_state=0)
X_ros, y_ros = ros.fit_resample(X_train, y_train)
```

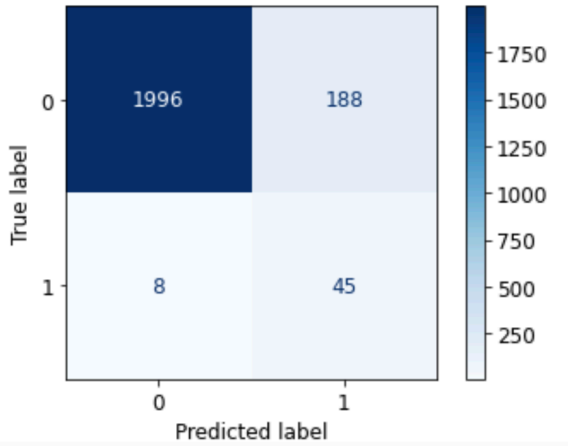
```python
from imblearn.over_sampling import SMOTE

X_smote, y_smote = SMOTE().fit_resample(X_train, y_train)
```





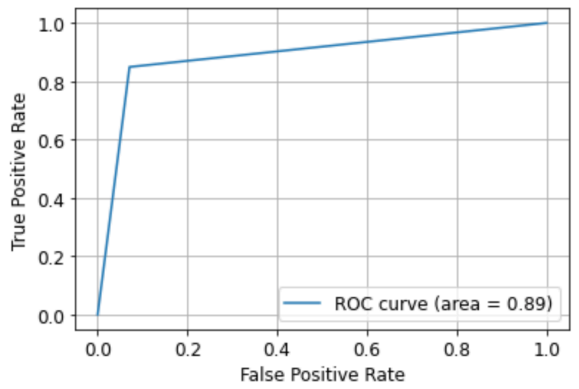
Ejemplo 1: Mamography dataset (4/4)

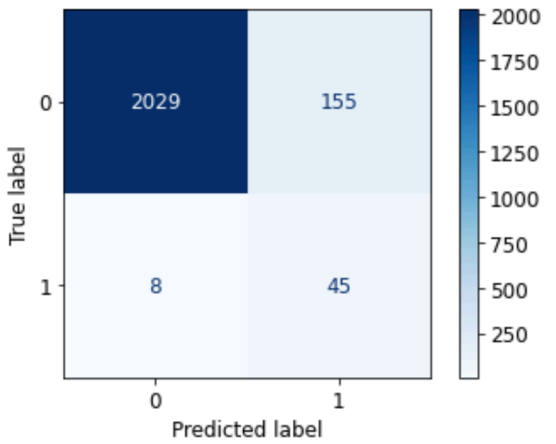




	0	1	accuracy	macro avg	weighted avg
precision	0.996008	0.193133	0.912383	0.594571	0.976986
recall	0.913919	0.849057	0.912383	0.881488	0.912383
f1-score	0.953200	0.314685	0.912383	0.633942	0.938072
support	2184.000000	53.000000	0.912383	2237.000000	2237.000000

RANDOM OVERSAMPLING





	0	1	accuracy	macro avg	weighted avg
precision	0.996073	0.225000	0.927135	0.610536	0.977804
recall	0.929029	0.849057	0.927135	0.889043	0.927135
f1-score	0.961384	0.355731	0.927135	0.658557	0.947034
support	2184.000000	53.000000	0.927135	2237.000000	2237.000000

SMOTE

Ejemplo 2: Higgs dataset (1/2)

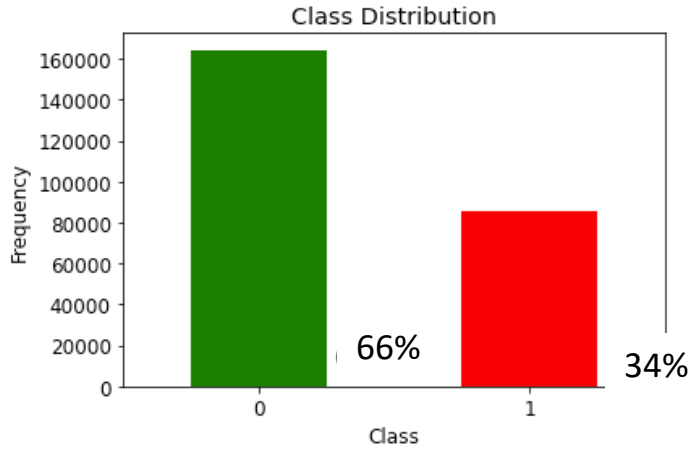
Datos simulados → señal: generación de un boson de Higgs decayendo en dos leptones ($\tau\tau$). Datos públicos: *https://www.kaggle.com/c/higgs-boson/data*

```
df_train = pd.read_csv("higgs-boson/training.csv")
```

```
df_train.shape
```

```
(250000, 33)
```

	EventId	DER_mass_MMC	DER_mass_transverse_met_lep	DER_mass_vis	DER_pt_h
0	100000	138.470	51.655	97.827	27.980
1	100001	160.937	68.768	103.235	48.146
2	100002	-999.000	162.172	125.953	35.635
3	100003	143.905	81.417	80.943	0.414
4	100004	175.864	16.915	134.805	16.405



- Problema: clasificacion de eventos:
- Clases:
  - señal (generación Higgs) → 1
  - background → 0

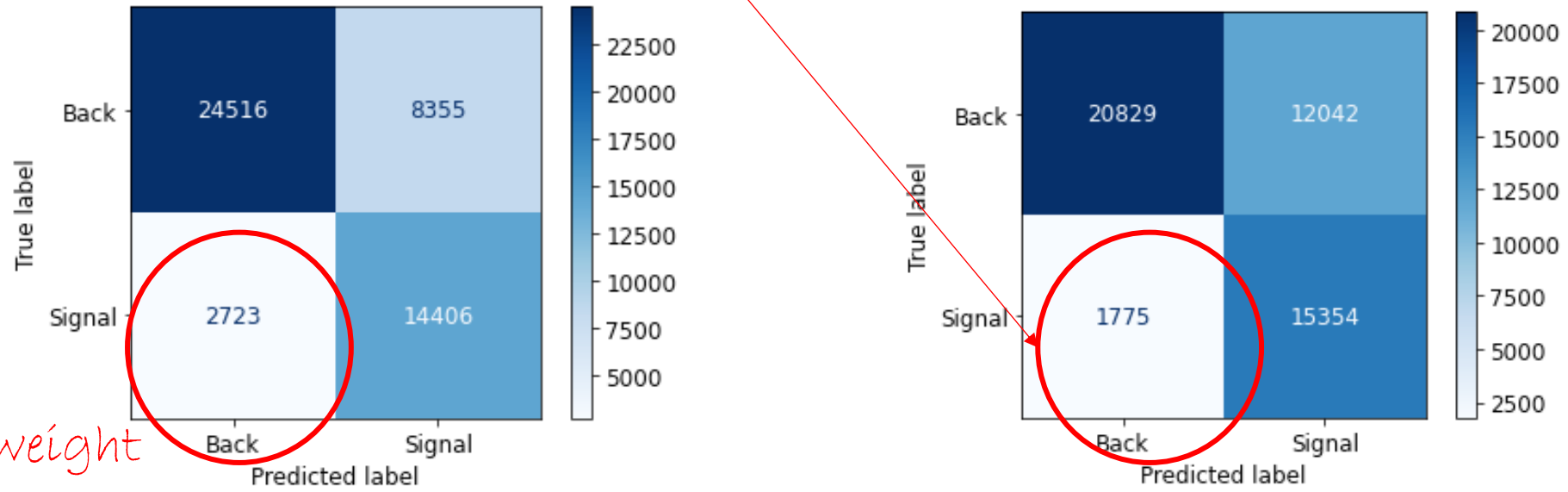
```
df_train["Label"].value_counts()
```

```
0    164333
1     85667
Name: Label, dtype: int64
```

Ejemplo 2: Higgs dataset (2/2)

```
model2 = Sequential()
model2.add(Dense(300, input_shape=(X_train.shape[1],), activation='relu'))
model2.add(Dense(150, activation = "relu"))
model2.add(Dense(1, activation='sigmoid'))
model2.summary()
model2.compile(optimizer='rmsprop', loss= "binary_crossentropy",
               metrics=['accuracy',tf.keras.metrics.Precision(),tf.keras.metrics.Recall()])
history2 = model2.fit(X_train, y_train, epochs=10,
                      verbose=1,
                      validation_data = (X_val, y_val),
                      class_weight={0:1, 1:2})
```

CON class_weight



SIN class_weight

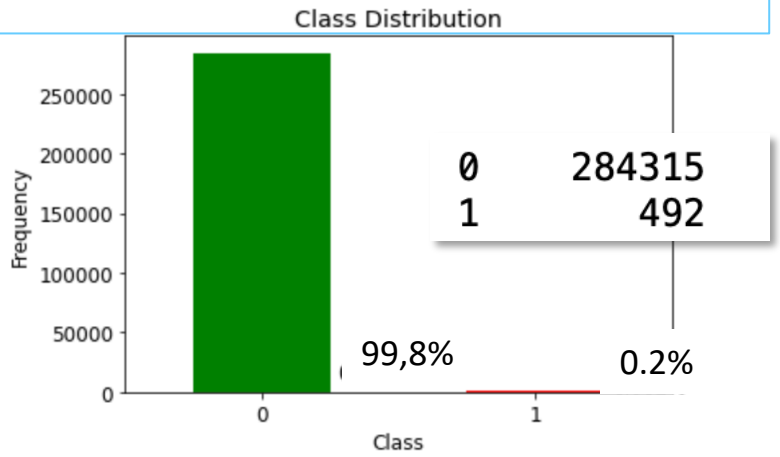
Ejemplo 3: Fraude tarjeta de crédito (1/1)

```
import tensorflow_addons as tfa
```

```
model2 = Sequential()
model2.add(Dense(300, input_shape=(X_train.shape[1],), activation='relu'))
model2.add(Dense(150, activation = "relu"))
model2.add(Dense(100, activation = "relu"))
model2.add(Dense(50, activation = "relu"))
model2.add(Dense(1, activation='sigmoid'))

model2.summary()

model2.compile(optimizer=Adam(),
               loss=tfa.losses.SigmoidFocalCrossEntropy(alpha=0.15, gamma=1.8, from_logits=False),
               metrics=['accuracy',tf.keras.metrics.Precision(),tf.keras.metrics.Recall()])

#loss=focal_loss_custom(alpha=0.2, gamma=5.0),
history2 = model2.fit(X_train, y_train, epochs=5,
                      verbose=1,
                      validation_data = (X_val, y_val))
```

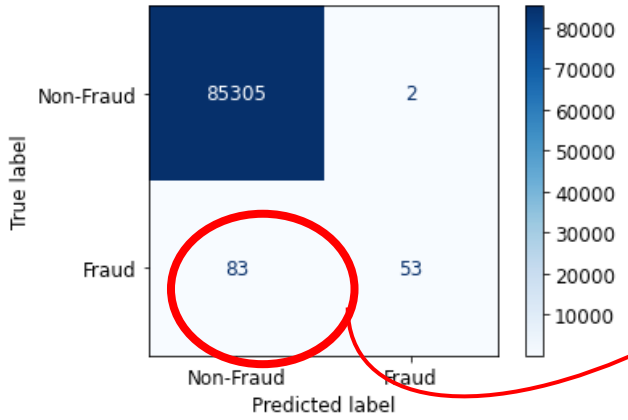
$$\mathrm{FL}(p_t) = -\alpha_t (1 - p_t)^{\gamma} \log(p_t)$$

- Problema: detectar transacción fraudulenta
- Clases: fraude→ 1 o no-fraude → 0



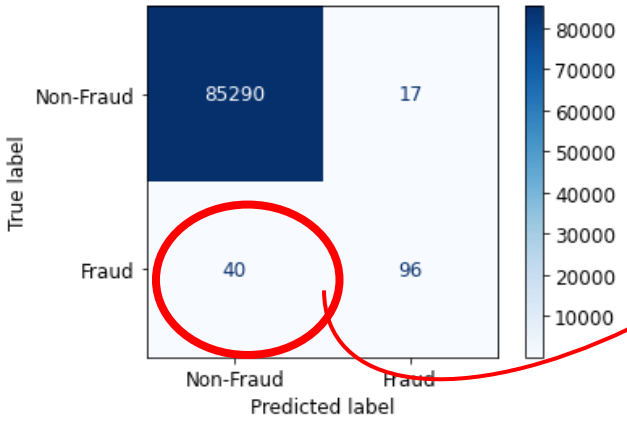
parámetro α_t : peso para la clase positiva y clase negativa

parámetro γ : asociado al los datos "difíciles" (los falsos negativos!!)



BINARY CROSSENTROPY

	0	1
precision	0.999028	0.963636
recall	0.999977	0.389706
f1-score	0.999502	0.554974
support	85307.000000	136.000000



FOCAL LOSS

	0	1
precision	0.999531	0.849558
recall	0.999801	0.705882
f1-score	0.999666	0.771084
support	85307.000000	136.000000

Finalizando ... en clasificación desbalanceada

- No hay una "receta" única para abordar el desbalance de clases
  - Enfoques a utilizar dependen del problema
  - También incluye optimización de parámetros (peso de cada clase, parámetros funciones de pérdida)
  - Se pueden utilizar enfoque híbrido: nivel de datos + nivel de algoritmo
- Desafíos incluyen:
  - Entender el problema/datos, generar nuevos algoritmos, nuevas funciones de pérdida, etc.

https://github.com/rpezoa/imbalanced-class

Clasificación desbalanceada con Python: modificando datos y algoritmos

PyCon Chile 2021

Raquel Pezoa Rivera

¡GRACIAS!

Agradecimientos: Proyecto FONDECYT POSTDOC Nº 3190740 y ANID PIA/APOYO AFB180002.