Navegando los datos

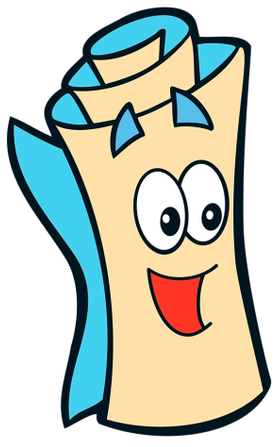
...sin naufragar en el intento

- Él es Da Wei
- Viaja en busca de verdades
- Cree que la ciencia de datos puede ayudar a encontrarlas

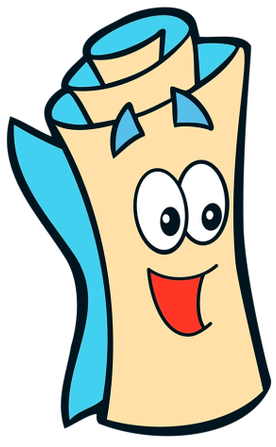
Quiero ser científico de datos

Pero no sé por donde partir :(

Pero no sé por donde partir :(



Soy el mapa, yo te ayudo!



Da Wei, decidido a recorrer el camino, debe pasar por aprender sobre:

- El lago de datos
- El Data Warehouse
- El análisis exploratorio de datos
- El procesamiento de datos
- Modelos de Machine Learning
- Cómo evaluar nuestros modelos
- La puesta en producción

En este doble click

- Recordar que son los términos *data lake* y *data warehouse*
- Aprender técnicas de análisis exploratorio de datos y de preparación de datos
- Una visión de alto nivel de los modelos de Machine Learning y cómo evaluar su desempeño

¿Por qué tener un Data Lake + Data Warehouse?

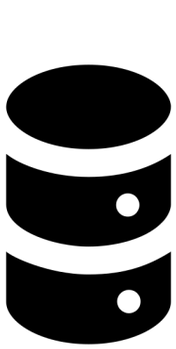
Una aplicación típica

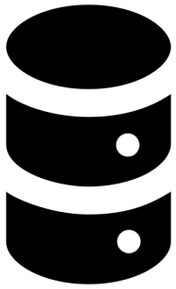
Típicamente vamos a tener una aplicación que resuelve un problema:

- Una aplicación para ahorrar dinero/invertir
- Una aplicación para hacer compras con envío a domicilio
- Una plataforma de aprendizaje en línea
- ...

Una aplicación típica

- Estas aplicaciones requieren interactuar con una base de datos
- Sin una base de datos, el producto no sería posible
- Pero en principio, queremos resolver el problema de negocio, no navegar en los datos!





Una base de datos de una aplicación suele ser transaccional

Una BD transaccional

Una BD transaccional está pensada para hacer muchas operaciones livianas al día:

- Inserción, actualización y eliminación de tuplas
- Búsquedas simples: traer el usuario **x**, buscar los productos de la compra **y**, ...
- Además queremos soporte para propiedades ACID

Análisis de datos

En algún momento vamos a querer **obtener valor** de los datos generados:

- Queremos hacer *clustering* de nuestros clientes
- Queremos predecir la demanda de un producto
- Queremos recomendar cursos a futuros alumnos
- ...

Análisis de datos

Vamos a querer hacer análisis de datos en sistemas especializados para esto: BigQuery, Redshift, Snowflake

Estos sistemas no son transaccionales, ya que están pensadas para hacer pocas consultas por día, pero cada una requiere hartos recursos computacionales

Data Lake

Un Data Lake es el lugar donde guardo todos los datos generados por mi producto:

- Datos de la base de datos del producto
- Datos del tracking de los usuarios
- Datos de nuestros experimentos (ej. AB testing)

En general, es **información cruda** sin estructura: archivos de distinto tipo, de distintos dominios, etc

Pero en algún momento vamos a procesar estos datos

Data Warehouse

Un Data Warehouse es el lugar en el que guardamos información procesada

En general, son datos puestos a disposición de los analistas para que obtengan valor de los datos

Ejemplo

Una compañía puede tener un repositorio de archivos (como S3 o Cloud Storage) con mucha información cruda

Podemos tener *jobs* que procesen estos archivos y le den estructura y sentido

La información procesada la guardamos en una base de datos de análisis (ej. BigQuery)

Buscando el valor de los datos

Navegando los datos

Una vez que tenemos un información procesada, tenemos que navegar esta información

Para esto necesitamos de varios *skills*, los más importantes:

- Manejo de datos tabulares (con SQL, Pandas, ...)
- Técnicas para hacer *Exploratory Data Analysis* (EDA)
- Uso de herramientas de visualización

Ejemplo práctico: datos de Bad Bunny

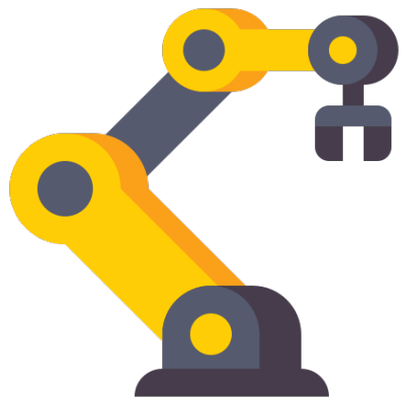
Más allá de la exploración

Explorar los datos nos sirve para encontrar patrones que no son explícitos

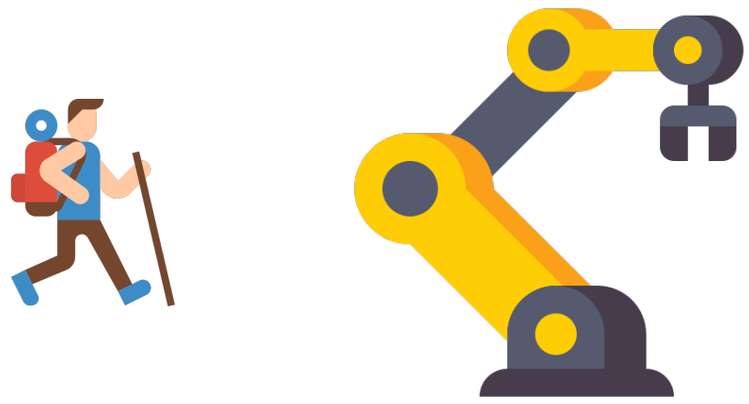
Por ejemplo, podemos encontrar *clusters* de canciones para separar las canciones exitosas de las no exitosas

En el ejemplo, podíamos visualizar *clusters*; ahora vamos a ir más allá, haciendo programas que aprendan a separar canciones automáticamente

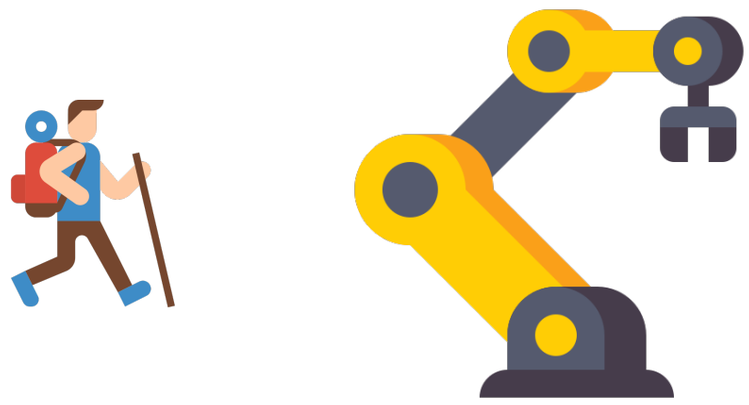
Programas que aprenden por nosotros



AKA Aprendizaje Automático



AKA Machine Learning

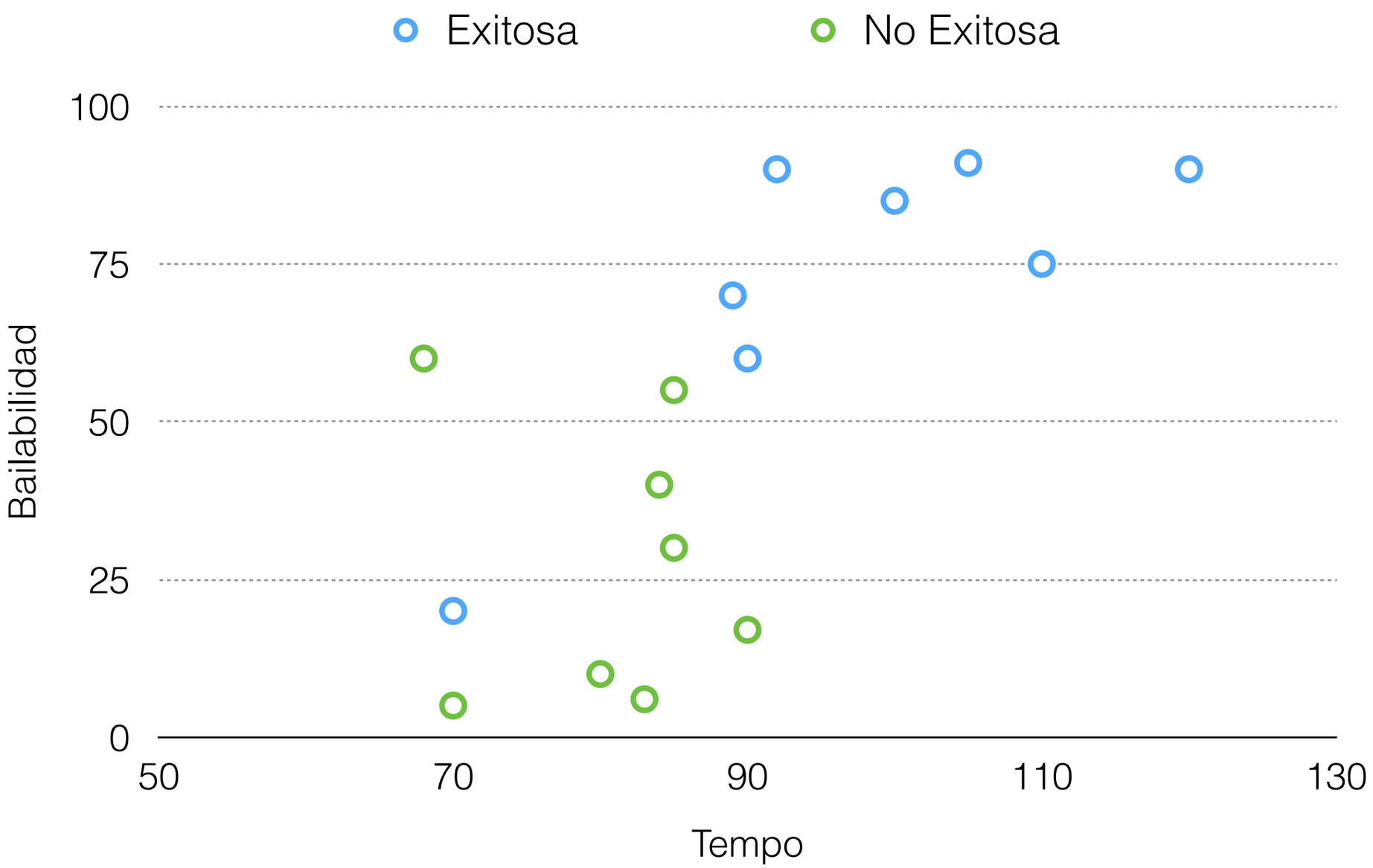


Clasificando Canciones

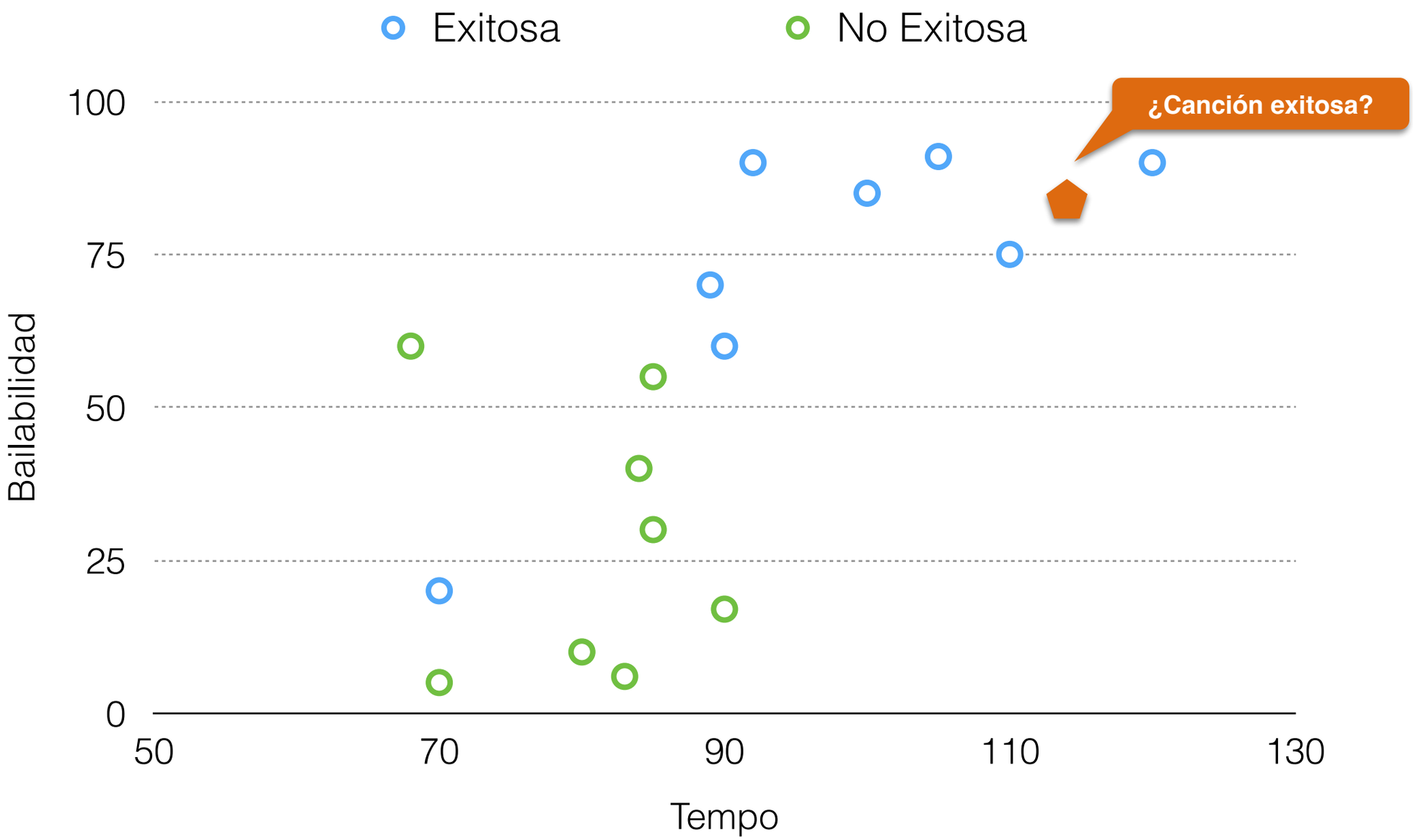
Imaginemos que tenemos datos de canciones: su tempo, su bailabilidad y si fue exitosa o no

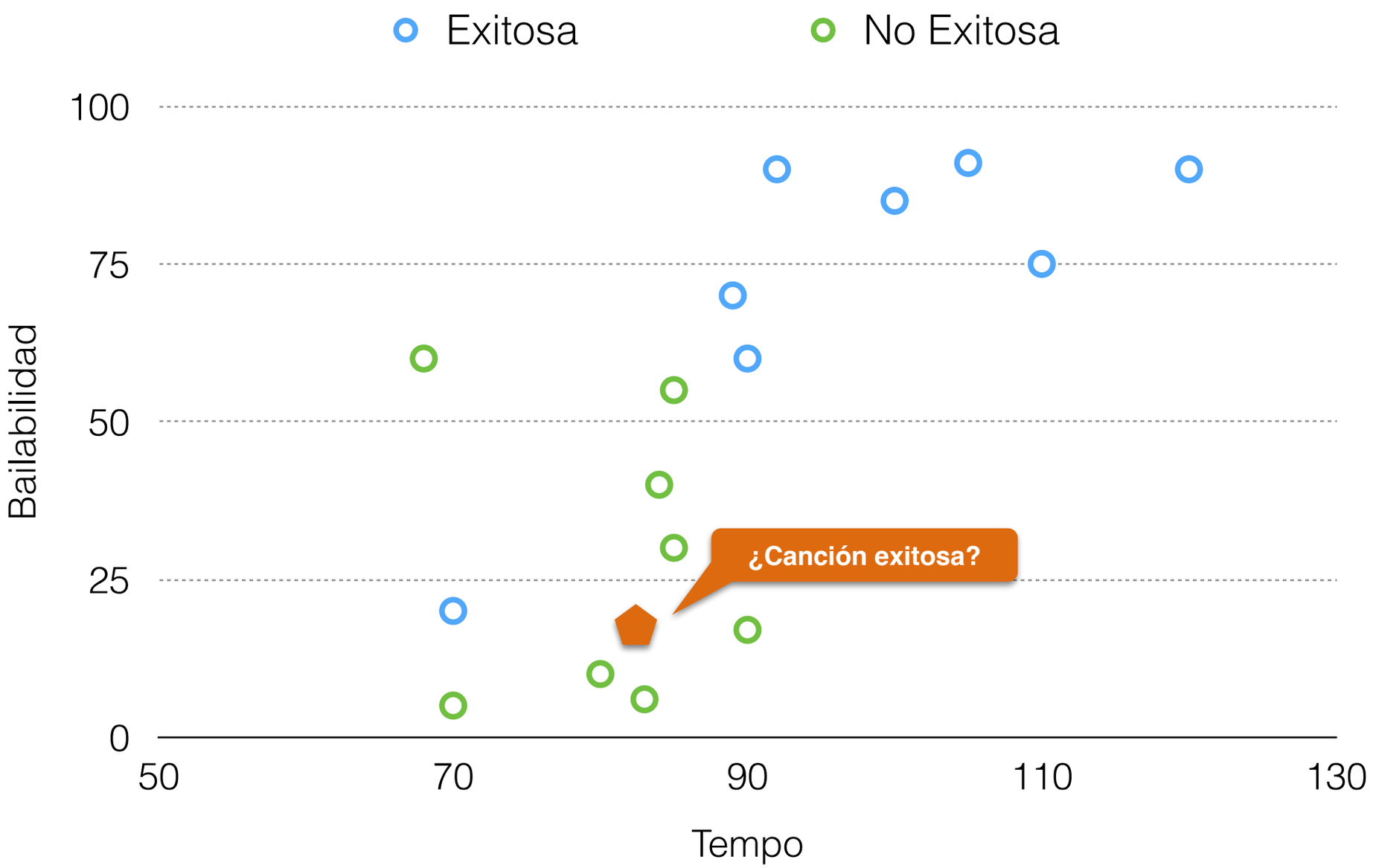
Si creamos una nueva canción, nos gustaría saber si será exitosa

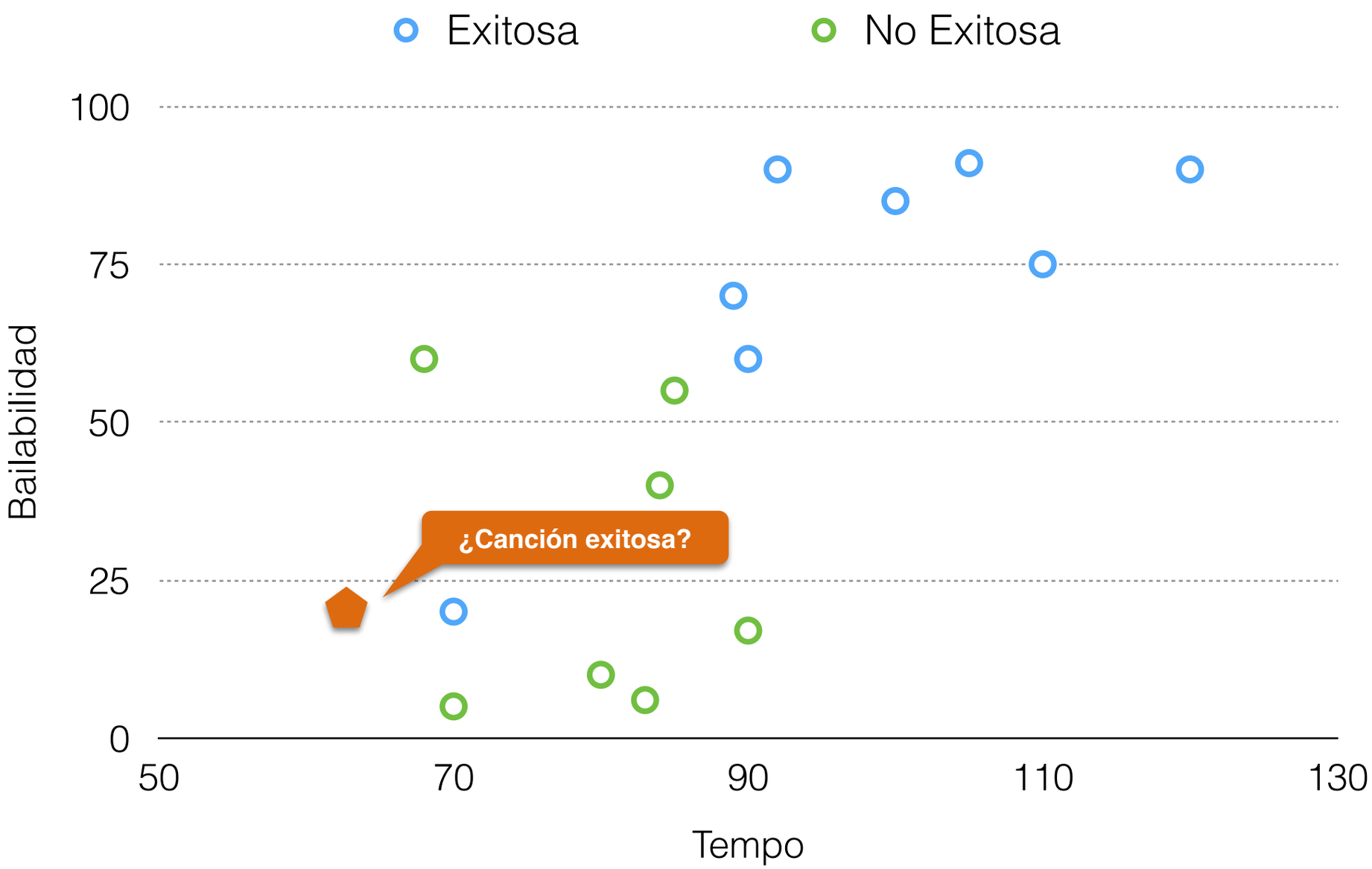
¿Cómo lo hacemos?

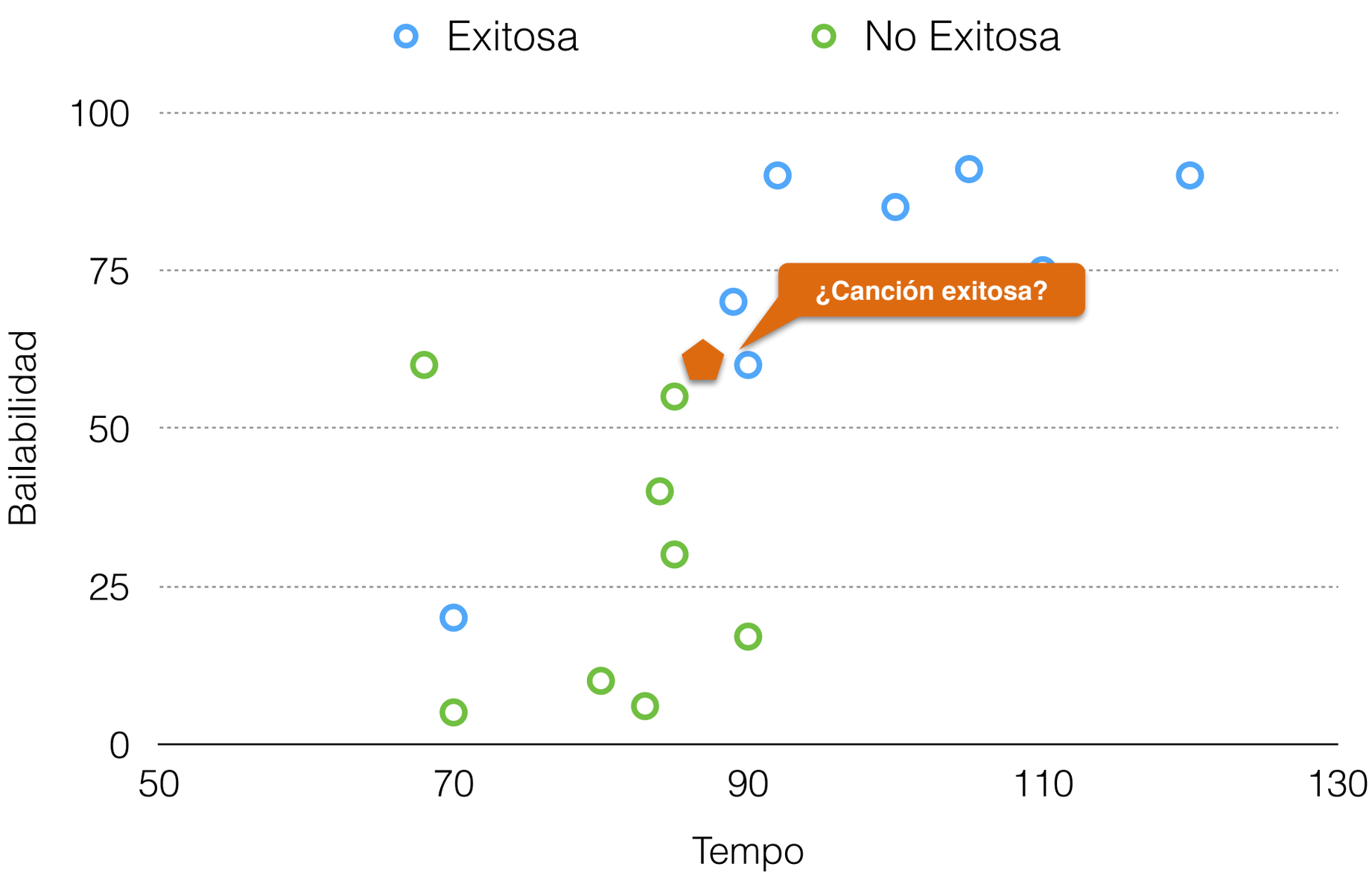

Exitosa
No Exitosa
100
75
50
25
0
Bailabilidad
50
70
90
110
130
Tempo

Agreguemos una nueva canción


Exitosa
No Exitosa
¿Canción exitosa?
100
75
50
25
0
Bailabilidad
50
70
90
110
130
Tempo


Exitosa
No Exitosa
100
75
50
25
0
Bailabilidad
50
70
90
110
130
Tempo
¿Canción exitosa?


Exitosa
No Exitosa
100
75
50
25
0
Bailabilidad
¿Canción exitosa?
50
70
90
110
130
Tempo


Exitosa
No Exitosa
100
75
50
25
0
Bailabilidad
¿Canción exitosa?
50
70
90
110
130
Tempo

Clasificando Canciones

Hasta ahora, por inspección visual podemos hacer un buen trabajo

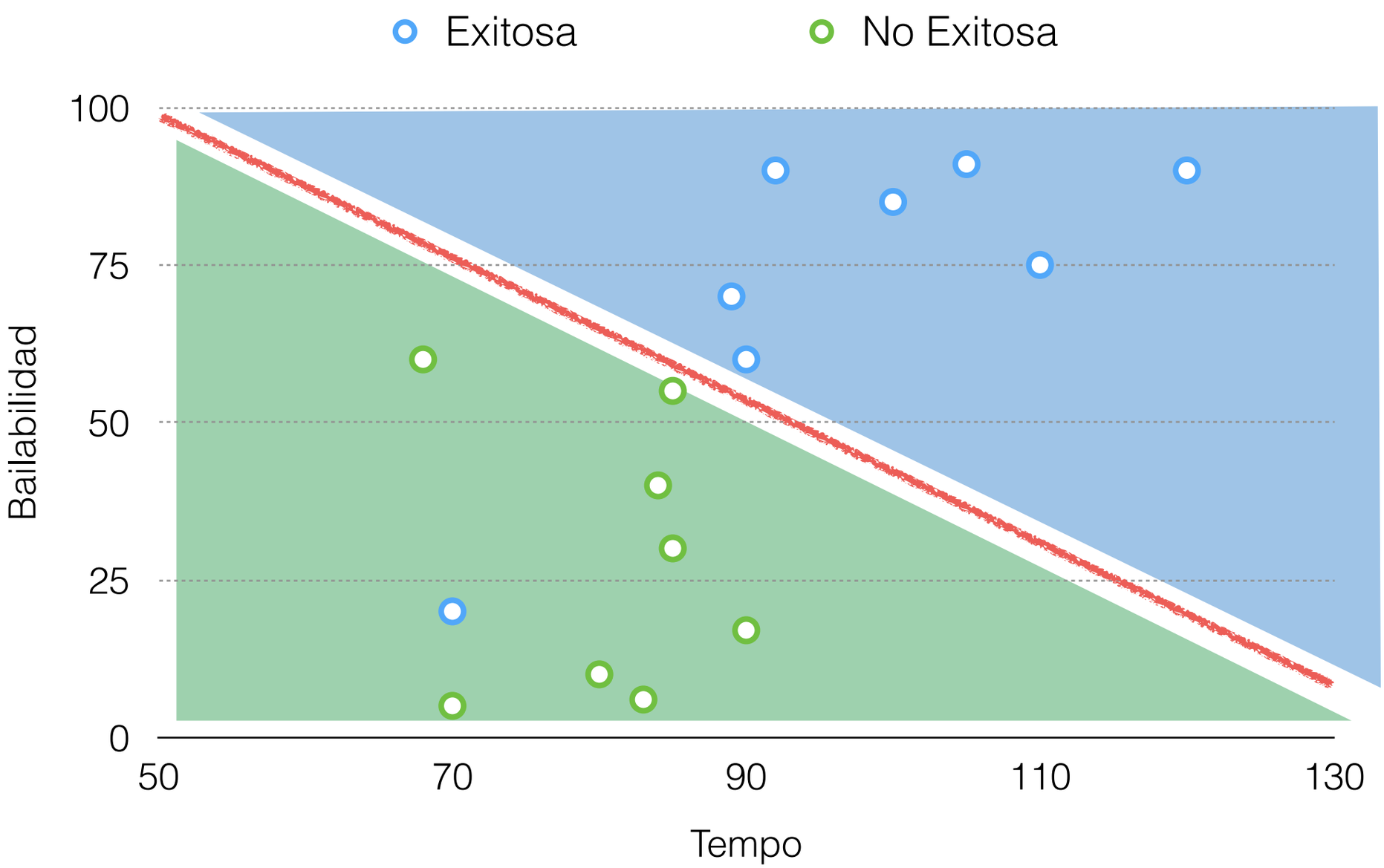
¿Hay una forma de automatizar esto?

Clasificación Lineal

Podemos encontrar una línea (recta) que divide el espacio en dos

Cuando creemos una canción, podemos ver de que lado de la recta está

Dependiendo del lado, podremos predecir si será exitosa o no exitosa


Exitosa
No Exitosa
100
75
50
25
0
Bailabilidad
50
70
90
110
130
Tempo

Encontrando la recta

Podemos encontrar esta recta resolviendo un problema de optimización

Por ejemplo, el modelo de regresión logística se basa en encontrar un estimador de máxima verosimilitud

En cambio, el modelo SVM, busca maximizar el margen entre instancias "en la frontera"

Encontrando la recta

Cada modelo tiene su receta, pero al final todos buscan lo mismo: encontrar una recta que separe lo mejor posible las observaciones

Más dimensiones

¿Qué pasa si además de tempo y bailabilidad, también tenemos la positividad de una canción?

¿Hasta qué punto puedo seguir agregando columnas a mi *dataset*?

Clasificación no lineal

Además, podemos hacer clasificación no lineal

Esto es, encontrar una función más compleja que separe las observaciones

Evaluando en modelo

Supongamos que tenemos un modelo que predice si una canción es exitosa o no, ¿cómo probarías si este modelo funciona bien?

Evaluando en modelo

Podemos empezar a coleccionar canciones que el modelo no ha visto, y ver si predice correctamente

En principio, podemos calcular el porcentaje de respuestas correctas (*accuracy*)

Pero también podemos analizar los Verdaderos-Falsos/ Positivos-Negativos

Canción	Predicción	Respuesta Real
Moscow Mule	Exitosa	Exitosa
Me Porto Bonito	Exitosa	Exitosa
Tarot	No Exitosa	Exitosa
El Efecto	Exitosa	Exitosa
La Corriente	Exitosa	No Exitosa
Neverita	Exitosa	Exitosa
Ojitos Lindos	No Exitosa	Exitosa
Otro Atardecer	No Exitosa	No Exitosa
Me fui de vacaciones	Exitosa	Exitosa

Canción	Predicción	Respuesta Real
Moscow Mule	Exitosa	Exitosa
Me Porto Bonito	Exitosa	Exitosa
Tarot	No Exitosa	Exitosa
El Efecto	Exitosa	Exitosa
La Corriente	Exitosa	No Exitosa
Neverita	Exitosa	Exitosa
Ojitos Lindos	No Exitosa	Exitosa
Otro Atardecer	No Exitosa	No Exitosa
Me fui de vacaciones	Exitosa	Exitosa

accuracy = 6/9

true positives = 5/9
false positives = 1/9

true negatives = 1/9
false negatives = 2/9

Ejemplo práctico: clasificando dígitos

Navegando los datos

...sin naufragar en el intento