El Panorama del Machine Learning Parte 2

Tipos de Sistemas de Machine Learning

- Es útil clasificarlos en categorías amplias, basándose en los siguientes criterios:
  - Cómo son supervisados durante el entrenamiento (supervisado, no supervisado, semi-supervisado, auto-supervisado y otros)
  - Si pueden o no aprender incrementalmente sobre la marcha (aprendizaje en línea versus aprendizaje por lotes)
  - Si funcionan simplemente comparando nuevos puntos de datos con puntos de datos conocidos, o en su lugar detectando patrones en los datos de entrenamiento y construyendo un modelo predictivo, como lo hacen los científicos (aprendizaje basado en instancias versus aprendizaje basado en modelos)

Supervisión del Entrenamiento

- Los sistemas de ML pueden clasificarse según la cantidad y tipo de supervisión que reciben durante el entrenamiento:

Aprendizaje supervisado

- En el aprendizaje supervisado, el conjunto de entrenamiento que alimentas al algoritmo incluye las soluciones deseadas, llamadas etiquetas

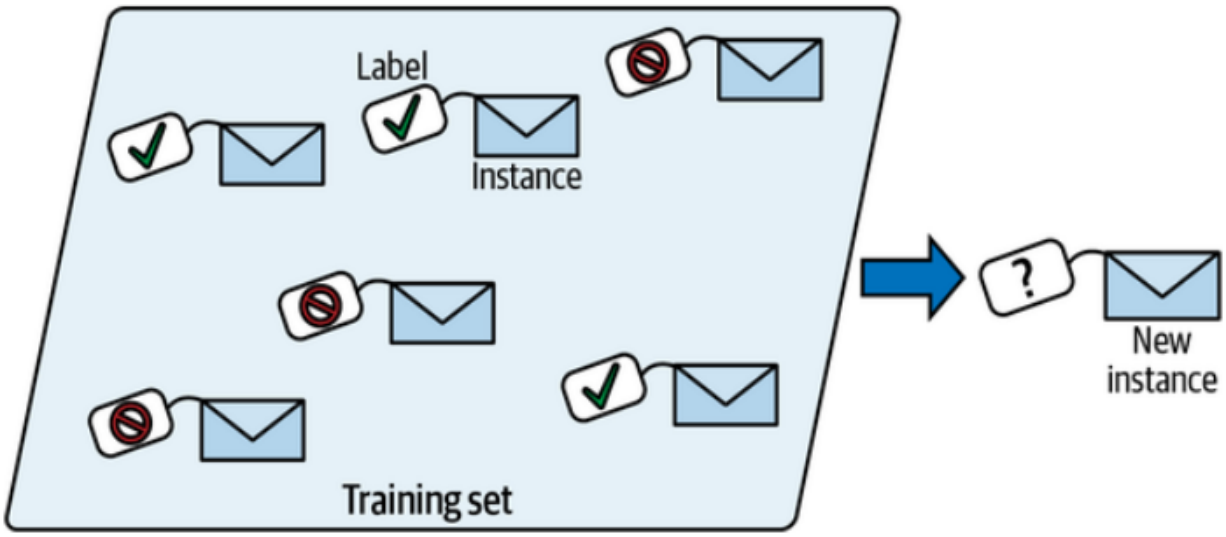


Figure 1-5. A labeled training set for spam classification (an example of supervised learning)

- Ej:
  - Clasificación: El filtro de spam, se entrena con muchos ejemplos de correos electrónicos junto con su clase (spam o ham).
  - Regresión: Predecir un valor numérico objetivo, como el precio de un coche, dadas un conjunto de características (kilometraje, edad, marca, etc.). Necesitas darle muchos ejemplos de coches, incluyendo características y objetivos.

Aprendizaje no supervisado

- Los datos de entrenamiento no están etiquetados. El sistema intenta aprender sin un maestro.
- Ej:
    - Digamos que tienes muchos datos sobre los visitantes de tu blog. Puede que quieras ejecutar un algoritmo de clustering para intentar detectar grupos de visitantes similares. En ningún momento le dices al algoritmo a qué grupo pertenece un visitante: encuentra esas conexiones sin tu ayuda.

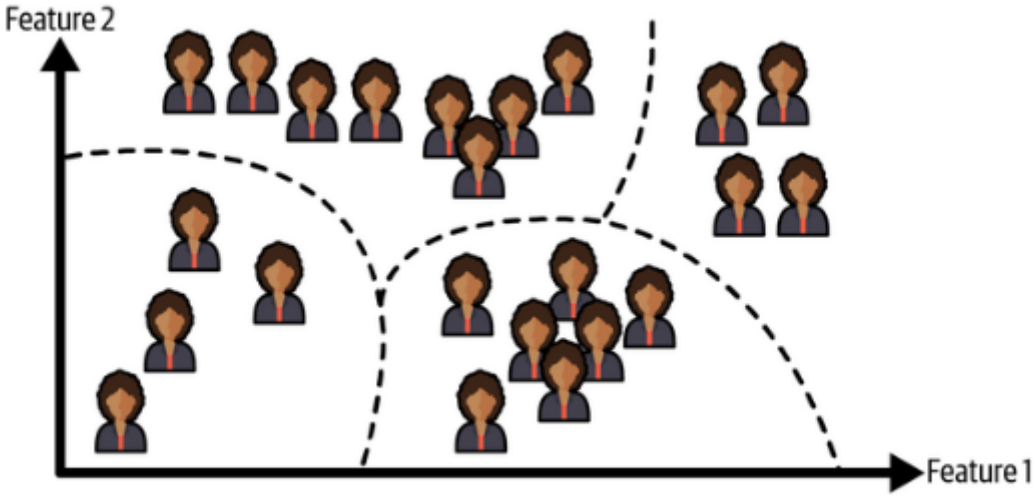


Figure 1-8. Clustering

    - Los algoritmos de visualización también son buenos ejemplos de aprendizaje no supervisado: les alimentas muchos datos complejos y no etiquetados, y generan una representación 2D o 3D de tus datos que puede ser fácilmente graficada

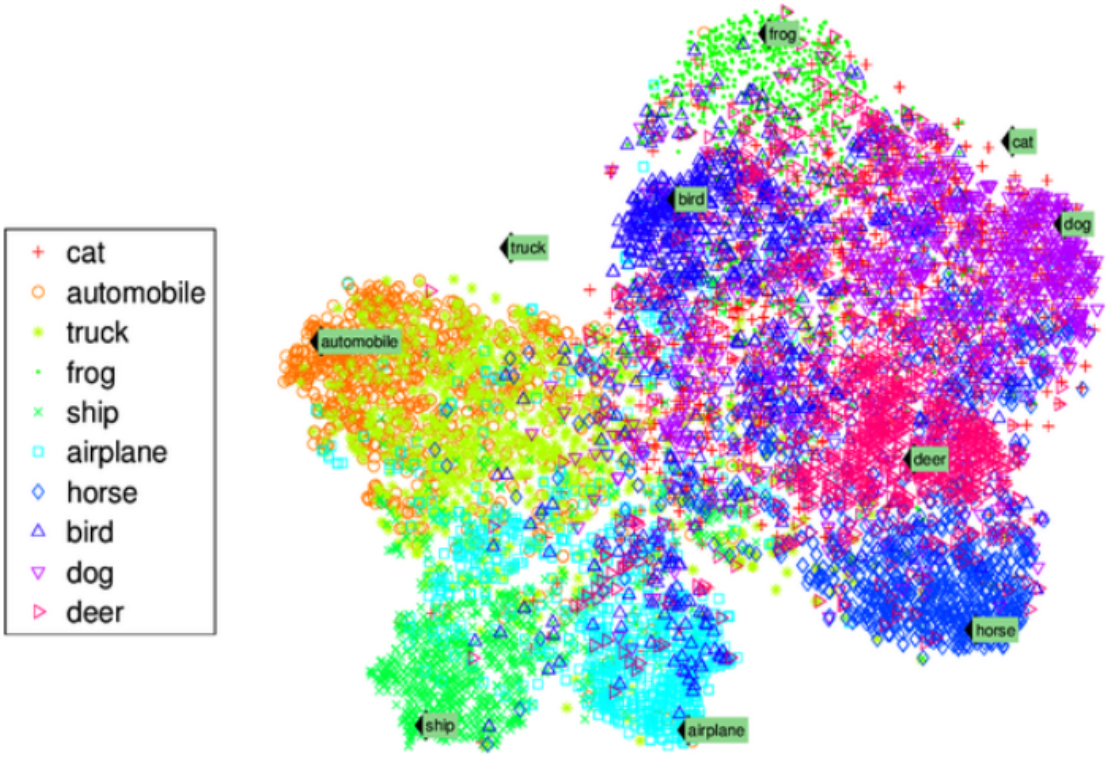


Figure 1-9. Example of a t-SNE visualization highlighting semantic clusters[2]

    - Una tarea relacionada es la reducción de dimensionalidad, en la cual el objetivo es simplificar los datos sin perder demasiada información. Una forma de hacer esto es fusionar varias características correlacionadas en una.

- Otra tarea no supervisada importante es la detección de anomalías—por ejemplo, detectar transacciones inusuales con tarjetas de crédito para prevenir fraude, capturar defectos de manufactura, o eliminar automáticamente valores atípicos de un conjunto de datos antes de alimentarlo a otro algoritmo de aprendizaje.

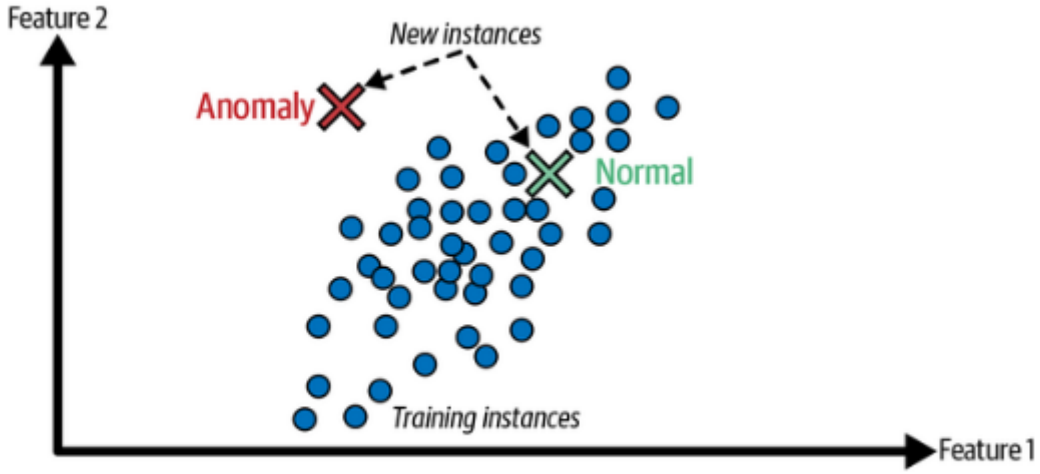


Figure 1-10. Anomaly detection

- Aprendizaje de reglas de asociación, en el cual el objetivo es explorar grandes cantidades de datos y descubrir relaciones interesantes entre atributos.

Aprendizaje semi-supervisado

- Algunos algoritmos pueden lidiar con datos que están parcialmente etiquetados.
- Algunos servicios de alojamiento de fotos, como Google Photos, son buenos ejemplos de esto. Una vez que subes todas tus fotos familiares al servicio, automáticamente reconoce que la misma persona A aparece en las fotos 1, 5 y 11, mientras que otra persona B aparece en las fotos 2, 5 y 7.
- La mayoría de los algoritmos de aprendizaje semi-supervisado son combinaciones de algoritmos no supervisados y supervisados. Por ejemplo, un algoritmo de clustering puede ser usado para agrupar instancias similares juntas, y luego cada instancia no etiquetada puede ser etiquetada con la etiqueta más común en su cluster.
- Ej:
  - Dos clases (triángulos y cuadrados): los ejemplos no etiquetados (círculos) ayudan a clasificar una nueva instancia (la cruz) en la clase de triángulos en lugar de la clase de cuadrados, aunque está más cerca de los cuadrados etiquetados

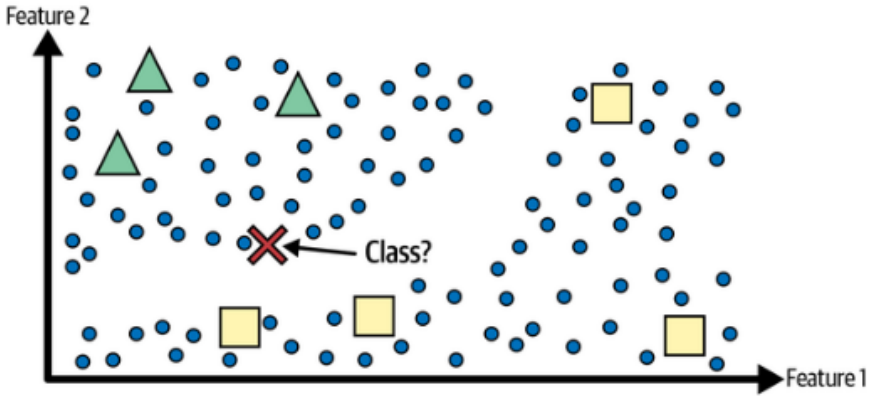


Figure 1-11. Semi-supervised learning with two classes (triangles and squares): the unlabeled examples (circles) help classify a new instance (the cross) into the triangle class rather than the square class, even though it is closer to the labeled squares

Aprendizaje auto-supervisado

- Generar un conjunto de datos completamente etiquetado a partir de uno completamente no etiquetado. Una vez que todo el conjunto de datos está etiquetado, cualquier algoritmo de aprendizaje supervisado puede ser usado.
- Ej:
  - Si tienes un gran conjunto de datos de imágenes no etiquetadas, puedes enmascarar aleatoriamente una pequeña parte de cada imagen y luego entrenar un modelo para recuperar la imagen original (Figura 1-12). Durante el entrenamiento, las imágenes enmascaradas se usan como entradas al modelo, y las imágenes originales se usan como etiquetas.

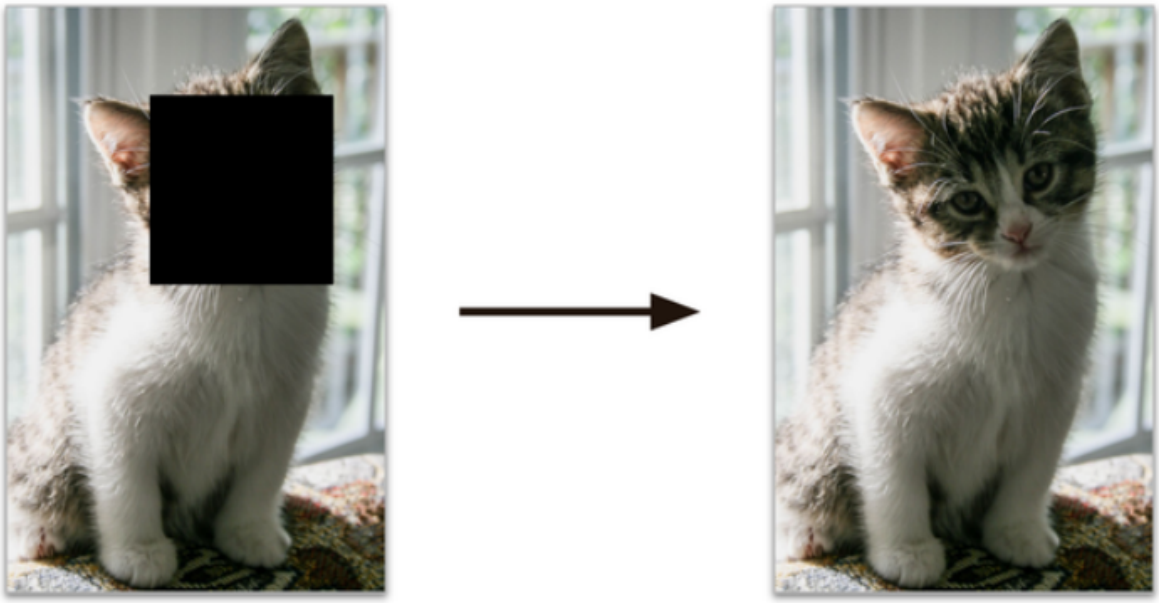


Figure 1-12. Self-supervised learning example: input (left) and target (right)

- Algunas personas consideran el aprendizaje auto-supervisado como parte del aprendizaje no supervisado, ya que trata con conjuntos de datos completamente no etiquetados. Pero el aprendizaje auto-supervisado usa etiquetas (generadas) durante el entrenamiento, así que en ese sentido está más cerca del aprendizaje supervisado.

Aprendizaje por refuerzo

- El sistema de aprendizaje, llamado agente en este contexto, puede observar el entorno, seleccionar y realizar acciones, y obtener recompensas a cambio (o penalizaciones en forma de recompensas negativas).

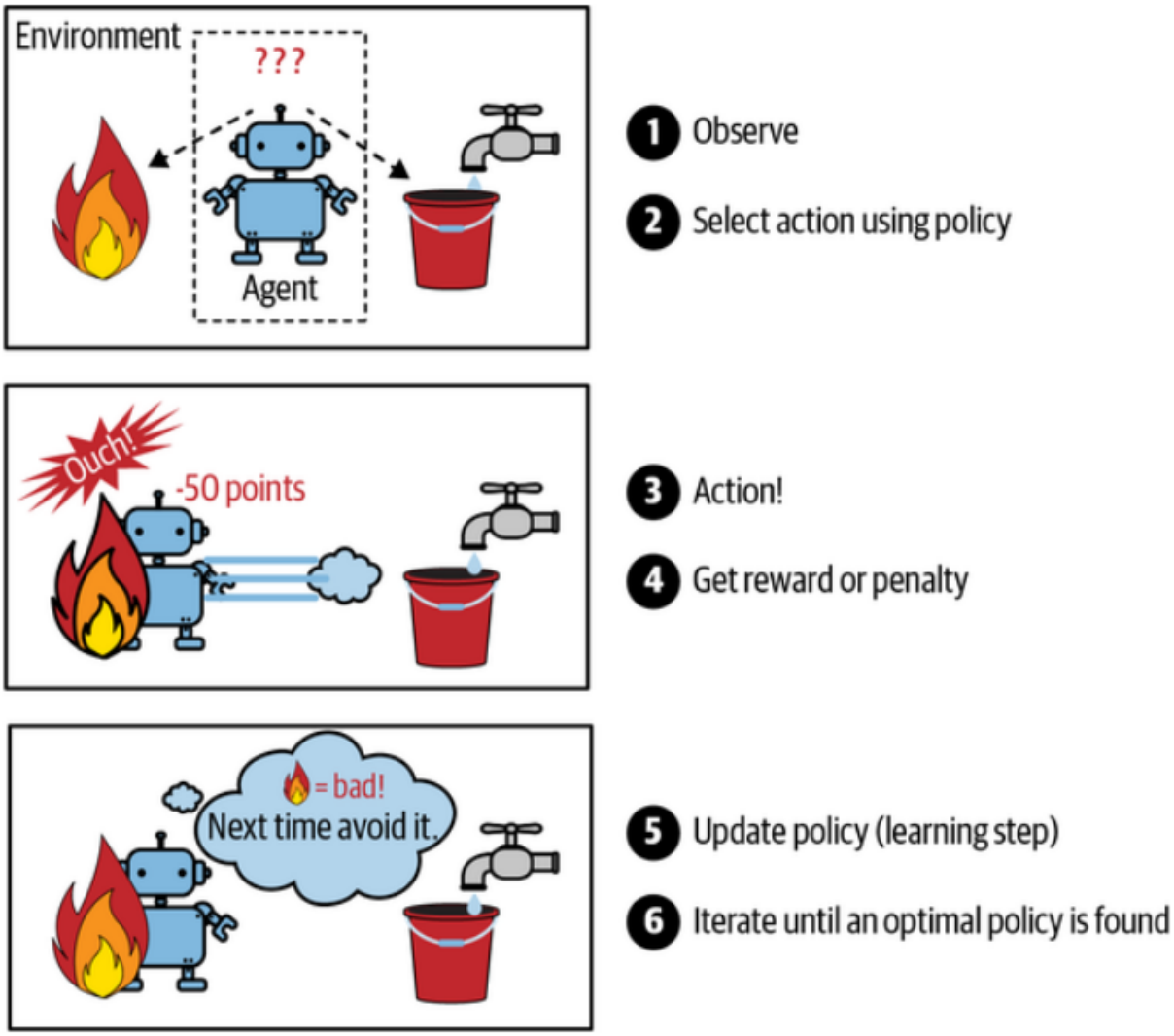


Figure 1-13. Reinforcement learning

- Ej:
  - Muchos robots implementan algoritmos de aprendizaje por refuerzo para aprender cómo caminar.

Aprendizaje por Lotes versus Aprendizaje en Línea

- Otro criterio usado para clasificar sistemas de machine learning es si el sistema puede o no aprender incrementalmente desde un flujo de datos entrantes.

Aprendizaje por lotes

- El sistema es incapaz de aprender incrementalmente: debe ser entrenado usando todos los datos disponibles.
- Típicamente se hace fuera de línea
- Primero el sistema es entrenado, y luego es lanzado a producción y funciona sin aprender más.
- El rendimiento de un modelo tiende a decaer lentamente con el tiempo, simplemente porque el mundo continúa evolucionando mientras el modelo permanece sin cambios. La solución es reentrenar regularmente el modelo con datos actualizados.

- Si quieres que un sistema de aprendizaje por lotes conozca sobre nuevos datos (como un nuevo tipo de spam), necesitas entrenar una nueva versión del sistema desde cero con el conjunto completo de datos (no solo los datos nuevos, sino también los datos antiguos).
- Esta solución es simple y a menudo funciona bien, pero entrenar usando el conjunto completo de datos puede tomar muchas horas.
- Entrenar con el conjunto completo de datos requiere muchos recursos computacionales (CPU, espacio de memoria, espacio en disco, E/S de disco, E/S de red, etc.), costando mucho dinero

Aprendizaje en línea

- En el aprendizaje en línea, entrenas el sistema incrementalmente alimentándolo con instancias de datos secuencialmente, ya sea individualmente o en pequeños grupos llamados mini-lotes.
- Cada paso de aprendizaje es rápido y económico, así que el sistema puede aprender sobre nuevos datos sobre la marcha, a medida que llegan

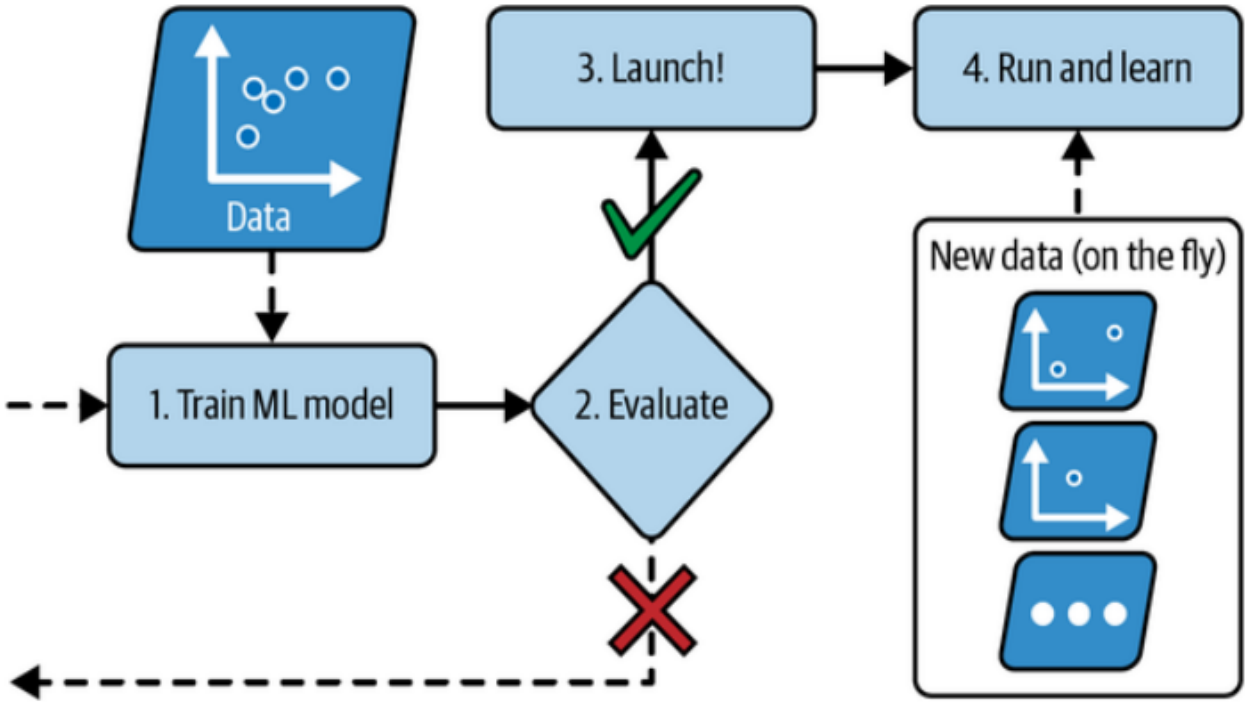


Figure 1-14. In online learning, a model is trained and launched into production, and then it keeps learning as new data comes in

- El aprendizaje en línea es útil para sistemas que necesitan adaptarse al cambio extremadamente rápido.
- Los algoritmos de aprendizaje en línea pueden ser usados para entrenar modelos en conjuntos de datos enormes que no caben en la memoria principal de una máquina (esto se llama aprendizaje fuera del núcleo).

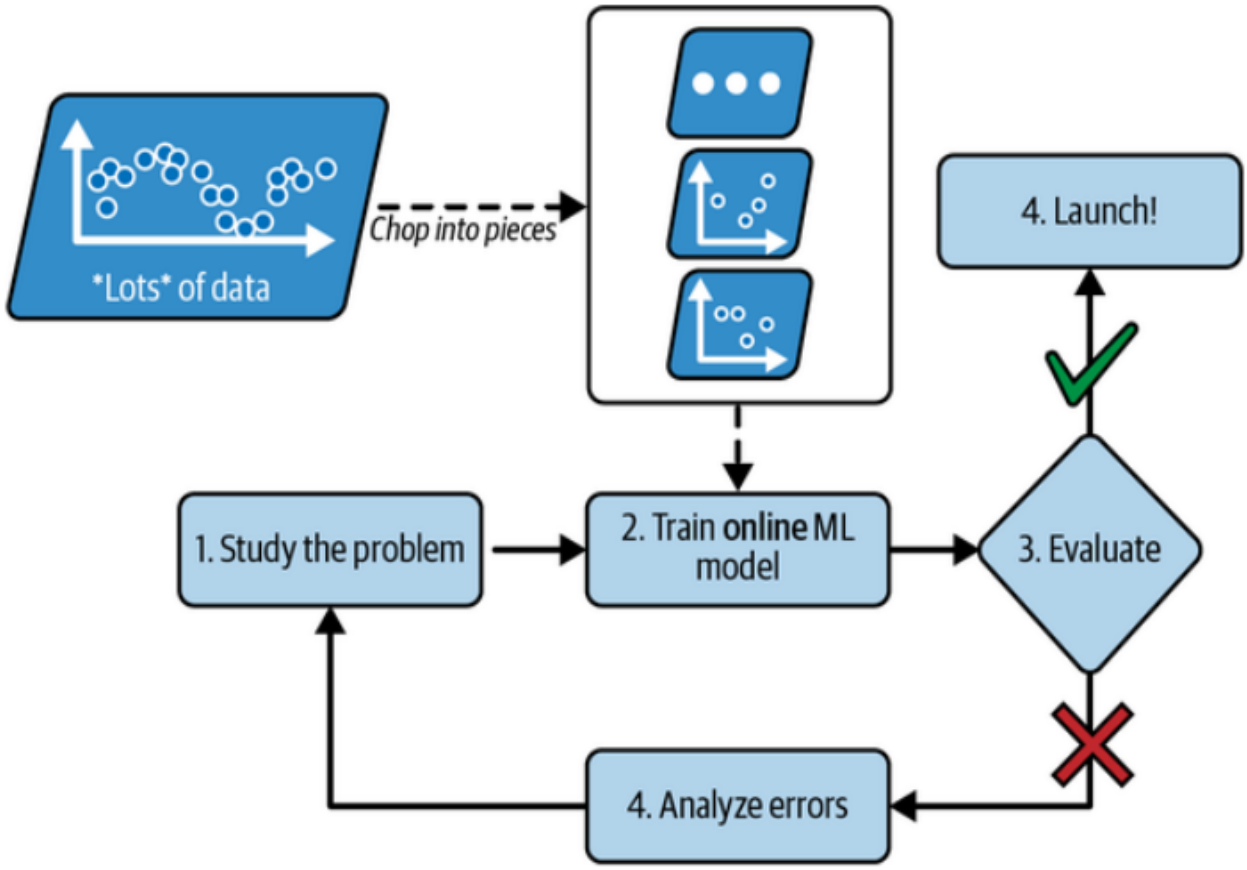


Figure 1-15. Using online learning to handle huge datasets

- Tasa de aprendizaje: Qué tan rápido deben adaptarse a los datos cambiantes.
  - Tasa de aprendizaje alta: Tu sistema se adaptará rápidamente a nuevos datos, pero también tenderá a olvidar rápidamente los datos antiguos.
  - Tasa de aprendizaje baja: El sistema tendrá más inercia; es decir, aprenderá más lentamente, pero también será menos sensible al ruido en los nuevos datos o a secuencias de puntos de datos no representativos (valores atípicos).
- Un gran desafío con el aprendizaje en línea es que si se alimentan datos malos al sistema.
  - Si es un sistema en vivo, tus clientes lo notarán.
  - Por ejemplo, los datos malos podrían provenir de un error.
  - Para reducir este riesgo:
    - Monitorea tu sistema de cerca y desactiva rápidamente el aprendizaje (y posiblemente revierte a un estado previamente funcional).
    - Monitorea los datos de entrada y reacciona a datos anormales.

Aprendizaje Basado en Instancias versus Aprendizaje Basado en Modelos

- Una forma más de categorizar sistemas de machine learning es por cómo generalizan.
- La mayoría de las tareas de machine learning son sobre hacer predicciones.
- Tener un buen rendimiento en los datos de entrenamiento es bueno, pero insuficiente; el verdadero objetivo es tener un buen rendimiento en nuevas instancias.

Aprendizaje basado en instancias

- Posiblemente la forma más trivial de aprendizaje es simplemente aprender de memoria.
- Ej:

    - Simplemente marcaría todos los correos electrónicos que son idénticos a correos electrónicos que ya han sido marcados por usuarios.
    - El sistema marcaría un correo electrónico como spam si tiene muchas palabras en común con un correo de spam conocido.
- Esto se llama aprendizaje basado en instancias: el sistema aprende los ejemplos de memoria, luego generaliza a nuevos casos usando una medida de similitud para compararlos con los ejemplos aprendidos (o un subconjunto de ellos).
- Por ejemplo, la nueva instancia sería clasificada como un triángulo porque la mayoría de las instancias más similares pertenecen a esa clase.

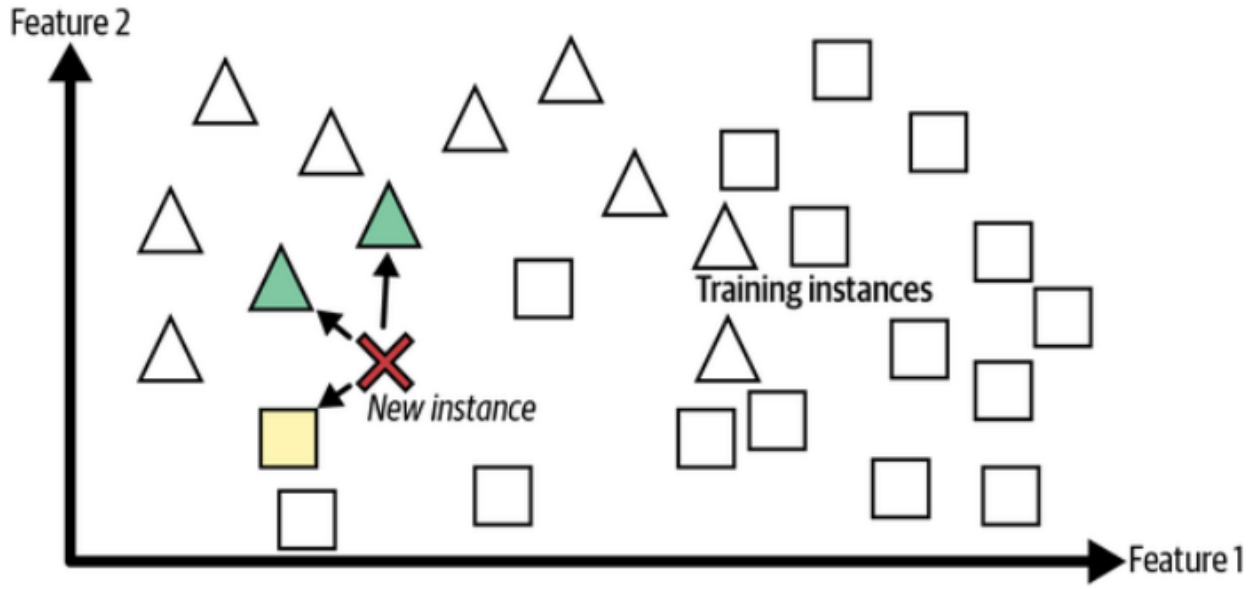


Figure 1-16. Instance-based learning

Aprendizaje basado en modelos y un flujo de trabajo típico de machine learning

- Otra forma de generalizar a partir de un conjunto de ejemplos es construir un modelo de estos ejemplos y luego usar ese modelo para hacer predicciones.

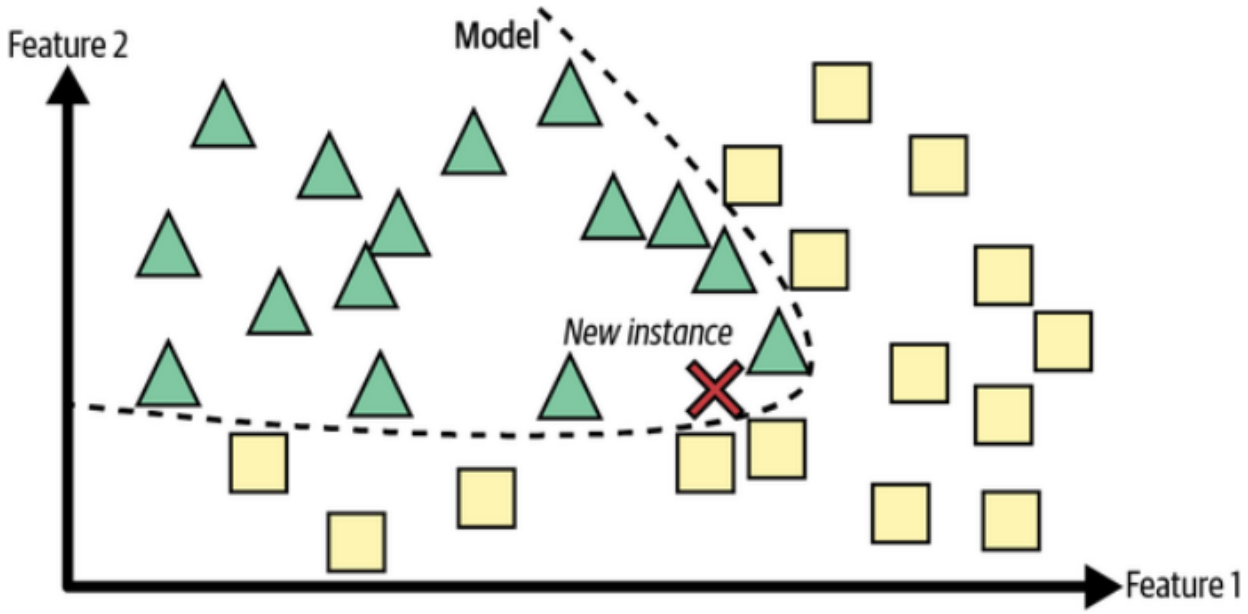
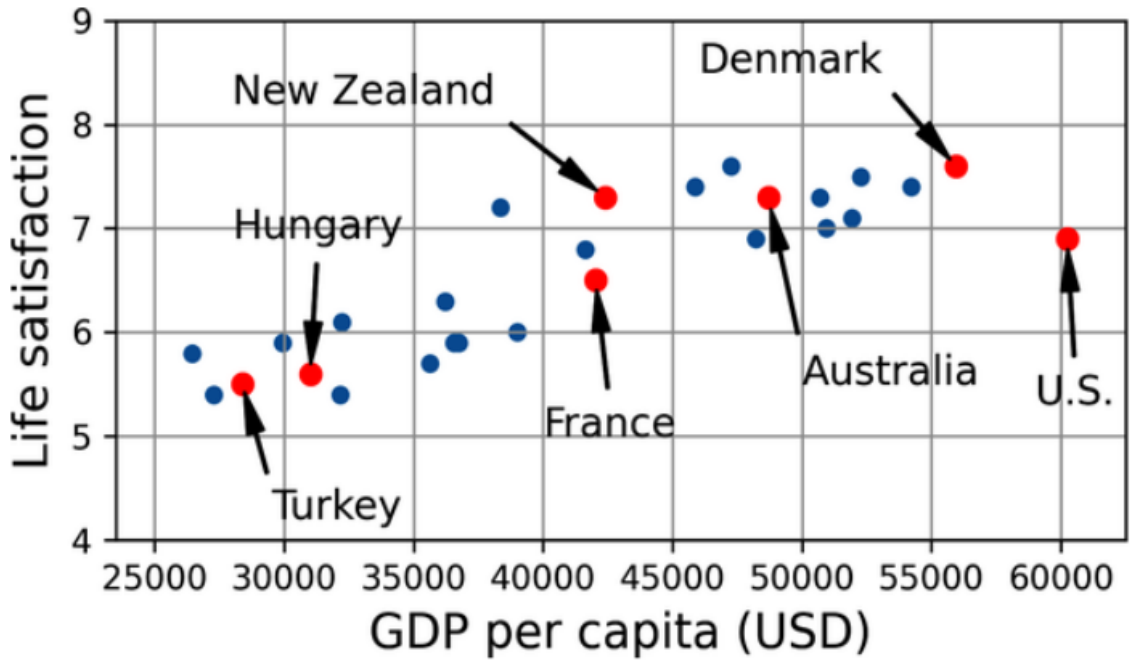


Figure 1-17. Model-based learning

- Ej:
  - Supón que quieres saber si el dinero hace a la gente feliz

Country	GDP per capita (USD)	Life satisfaction
Turkey	28,384	5.5
Hungary	31,008	5.6
France	42,026	6.5
United States	60,236	6.9
New Zealand	42,404	7.3
Australia	48,698	7.3
Denmark	55,938	7.6

  - 

  - *Figure 1-18. Do you see a trend here?*
  - Selección del modelo: Parece que la satisfacción con la vida aumenta más o menos linealmente a medida que el PIB per cápita del país aumenta, así que decides modelar la satisfacción con la vida como una función lineal del PIB per cápita.

Equation 1-1. A simple linear model

$$\text{life\_satisfaction} = \theta_0 + \theta_1 \times \text{GDP\_per\_capita}$$

  - Este modelo tiene dos parámetros del modelo, θ0 y θ1. Ajustando estos parámetros, puedes hacer que tu modelo represente cualquier función lineal

- ¿Cómo puedes saber qué valores harán que tu modelo tenga el mejor rendimiento?
  - Puedes definir una función de utilidad (o función de aptitud) que mida qué tan bueno es tu modelo.
  - O puedes definir una función de costo que mida qué tan malo es.
  - Para problemas de regresión lineal, las personas típicamente usan una función de costo que mide la distancia entre las predicciones del modelo lineal y los ejemplos de entrenamiento; el objetivo es minimizar esta distancia.
- Le alimentas tus ejemplos de entrenamiento, y encuentra los parámetros que hacen que el modelo lineal se ajuste mejor a tus datos. Esto se llama entrenar el modelo. En nuestro caso, el algoritmo encuentra que los valores óptimos de los parámetros son θ0 = 3.75 y θ1 = 6.78 × 10–5.

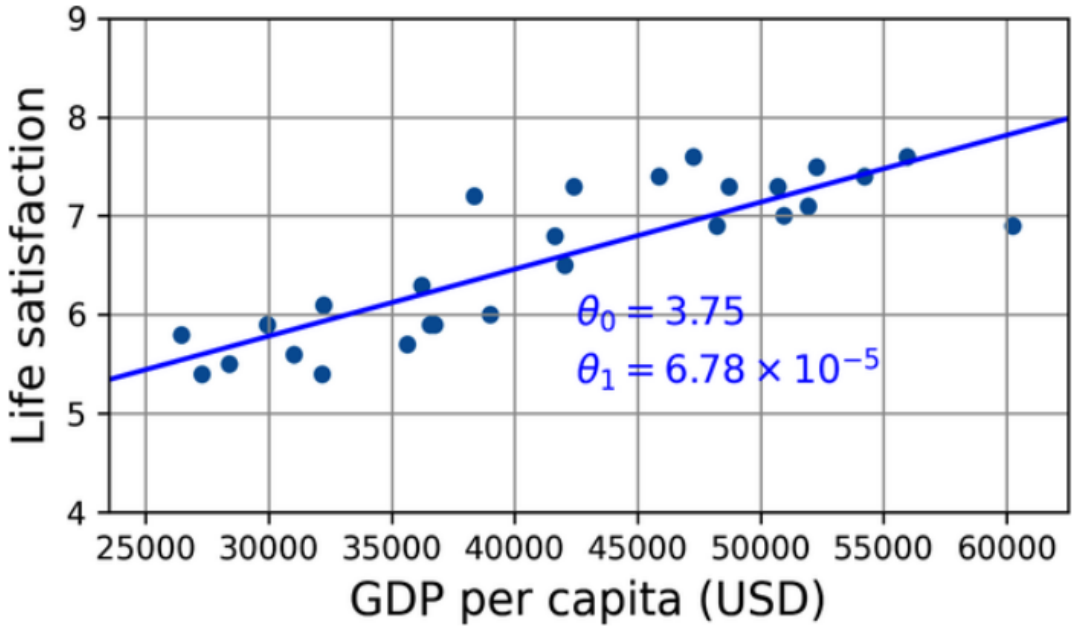


Figure 1-20. The linear model that fits the training data best

-
- Finalmente estás listo para ejecutar el modelo para hacer predicciones.
  - Ej: ¿Qué tan felices son los chipriotas?
    - Buscas el PIB per cápita de Chipre, encuentras $37,655, y luego aplicas tu modelo y encuentras que la satisfacción con la vida probablemente estará alrededor de 3.75 + 37,655 × 6.78 × 10–5 = 6.30.

◦ Código Python:

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression
# Descargar y preparar los datos
data_root = "https://github.com/ageron/data/raw/main/"
lifesat = pd.read_csv(data_root + "lifesat/lifesat.csv")
X = lifesat[["GDP per capita (USD)"]].values
y = lifesat[["Life satisfaction"]].values
# Visualizar los datos
lifesat.plot(kind='scatter', grid=True,
x="GDP per capita (USD)", y="Life satisfaction")
plt.axis([23_500, 62_500, 4, 9])
```

```
plt.show()
# Seleccionar un modelo lineal
model = LinearRegression()
# Entrenar el modelo
model.fit(X, y)
# Hacer una predicción para Chipre
X_new = [[37_655.2]] # PIB per cápita de Chipre en 2020
print(model.predict(X_new)) # salida: [[6.30165767]]
```

- Si todo salió bien, tu modelo hará buenas predicciones. Si no, puede que necesites usar más atributos, obtener más o mejor calidad de datos de entrenamiento, o quizás seleccionar un modelo más poderoso.
- Resumen:
  - Estudiaste los datos.
  - Seleccionaste un modelo.
  - Lo entrenaste con los datos de entrenamiento.
  - Finalmente, aplicaste el modelo para hacer predicciones en nuevos casos.