

Otros paradigmas de aprendizaje

MIAX11, noviembre 2023

Contenidos

- Aprendizaje semi-supervisado
- Aprendizaje activo
- Aprendizaje para series temporales
- Optimización de hiperparámetros
- Aprendizaje por refuerzo

Aprendizaje semi-supervisado

- En el conjunto de datos disponibles algunos ejemplos están etiquetados y otros no
- Razones frecuentes
 - ▶ Etiquetar todos los ejemplos es muy costoso o tedioso.
Ej.: indicar si hay una nebulosa en cada foto de un telescopio espacial
 - ▶ Existe un sesgo de muestreo y queremos considerar la distribución representativa de una población
Ej. en la aplicación de scoring de crédito, algunos clientes son pre-rechazados, otros pidieron el crédito y no firmaron, etc.

Idea General

Utilizar los ejemplos no etiquetados para mejorar el rendimiento de un modelo que se genera inicialmente con los ejemplos etiquetados.

- Se entrena un primer modelo con los datos etiquetados
- Se predicen los datos no etiquetados
- Se añaden las predicciones como etiquetas de los ejemplos no etiquetados
- Opcionalmente se pueden elegir sólo las predicciones que superen un umbral de confianza

- Se entrena un primer modelo con los datos etiquetados
- Se predicen los datos no etiquetados
- Se añaden las predicciones como etiquetas de los ejemplos no etiquetados
- Opcionalmente se pueden elegir sólo las predicciones que superen un umbral de confianza

Desventaja:

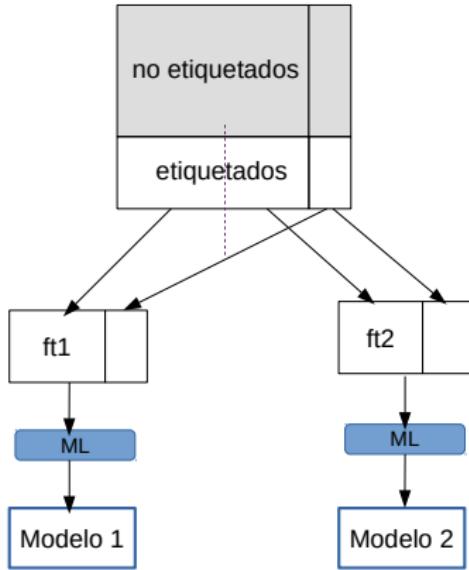
Los errores de clasificación se amplifican:

- El modelo ya tiene el conocimiento de los aciertos
- Los mal etiquetados no aportarán conocimiento nuevo

- Se tiene el dataset etiquetado con dos sub-conjuntos de características $\{X_1, Y\}$, $\{X_2, Y\}$ y un conjunto no etiquetado X'
- Idealmente las características en X_1 y X_2 deben ser independientes

Intuición

Al entrenar dos modelos, el primero va adquiriendo nuevo conocimiento de las etiquetas generadas por el segundo y viceversa



- Ambos modelos hacen su predicción sobre ejemplos no etiquetados
- Al asignar la clase con la mayor predicción probabilística asumimos que ese modelo tiene más confianza
- La nueva etiqueta completa un ejemplo que ayudará a discriminar al otro modelo en siguientes iteraciones

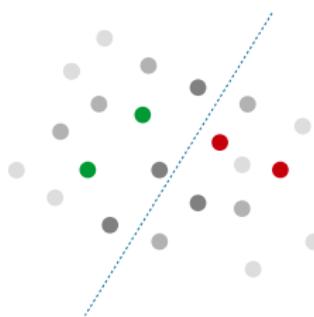
Modelo 1		Modelo 1		
Prob (SI)	Prob (NO)	Prob (SI)	Prob(NO)	Etiqueta
0,7	0,3	0,8	0,2	SI
0,1	0,9	0,2	0,8	NO
0,6	0,4	0,3	0,7	NO
0,6	0,4	0,5	0,5	SI

Algoritmo

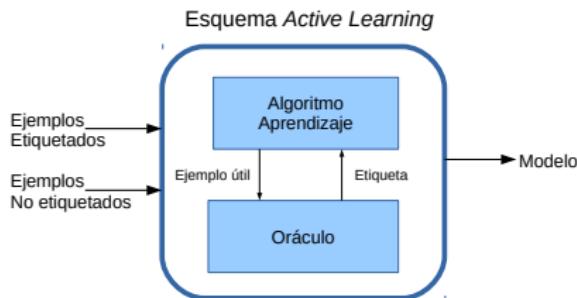
- Repetir k iteraciones o hasta agotar no-etiquetadas en X' :
 - 1 Entrenar modelo h_1 con $\{X_1, Y\}$
 - 2 Entrenar modelo h_2 con $\{X_2, Y\}$
 - 3 Seleccionar una muestra U del conjunto X'
 - 4 Predecir U con h_1 y con h_2
 - 5 Añadir al conjunto etiquetado (con sus respectivas características) la predicción de mayor confianza

Aprendizaje activo

- Escenario:
 - ▶ Algunos ejemplos etiquetados
 - ▶ Muchos ejemplos no etiquetados
 - ▶ Capacidad de saber la etiqueta, pero con cierto coste
- Idea: Preguntar las etiquetas de los ejemplos que más me ayuden a aprender



- Un algoritmo de ML no tendría por qué perder tiempo en ver repetidas veces ejemplos que le aporten poca información
- El oráculo puede participar de forma **activa** en la tarea de enseñar a discriminar entre categorías
- Muestreo por votación
 - ▶ Entrenar varios clasificadores
 - ▶ Predecir el ejemplo no etiquetado
 - ▶ Preguntar al oráculo cuando no exista una mayoría cualificada

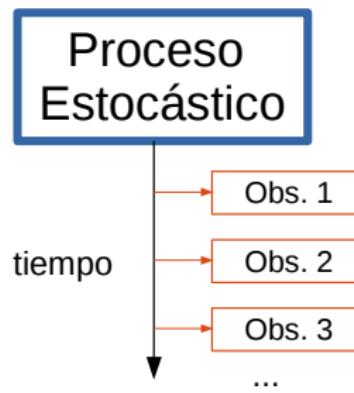
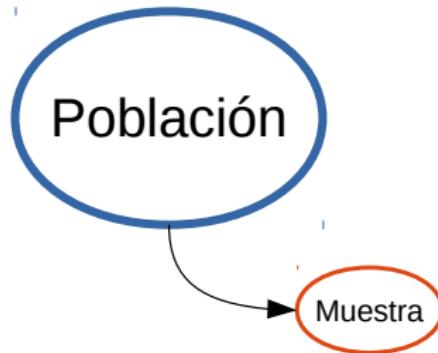


Series temporales

Definición

Conjunto de observaciones de una variable que están ordenadas en el tiempo

Datos Transversales vs. Series Temporales

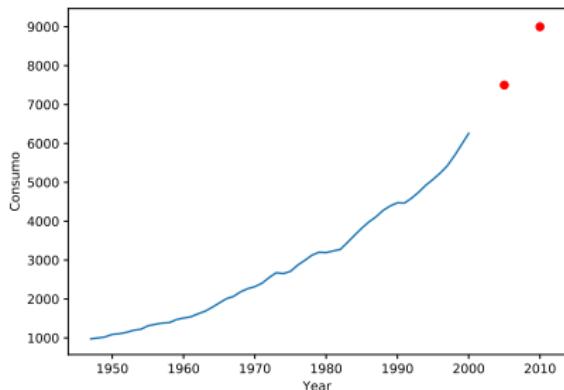


- **Modelado:** Dada una serie, construir un modelo que la aproxime para tener una mejor **comprensión** del proceso estocástico que la generó
- **Predicción:** Dada una serie histórica construir un modelo para dar una **previsión** a futuro de dicha serie
- **Clasificación:** Dada una serie no etiquetada, asignar una categoría a dicha serie
- **Clustering:** Dado un conjunto de series, agrupar en conjuntos las más similares

- Tendencia: Cambio a largo plazo del valor medio de la serie
- Estacionalidad: Variación cíclica, generalmente fruto de factores relacionados con momentos del año
- Residual: Variación irregular que queda al remover las otras fuentes de variación y que puede explicada o no por modelos probabilísticos

- Estimar la tendencia de una serie para predecir valores
- Se modela la variable dependiente en función del tiempo

$$y_t = \alpha + \beta t + \epsilon$$



- Modelos de tendencia lineal
 - ▶ La variable dependiente cambia a una tasa constante por unidad de tiempo
- Modelos de tendencia log-lineal
 - ▶ El cambio en la variable dependiente crece (o decrece) a una tasa particular. Esto implica cambio exponencial

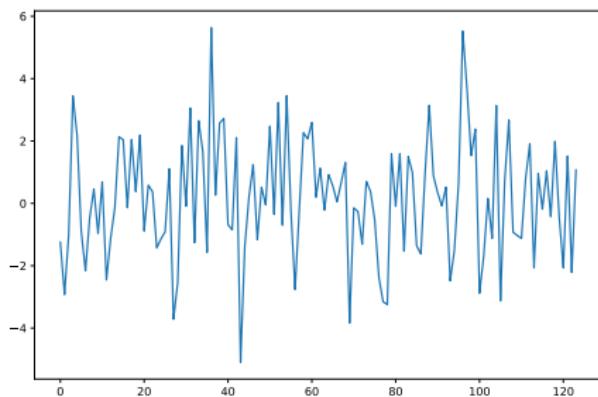
$$\ln y_t = \alpha + \beta t + \epsilon$$

- Los supuestos de la regresión lineal para encontrar buenos estimadores normalmente no se cumplen, en especial la ausencia de correlación serial de residuos
- Dos series con tendencia pueden mostrar una correlación alta cuando no tienen un vínculo causal
- Modelar una serie no-estacionaria en función de otra nos puede llevar al fenómeno conocido como regresión espuria
- En las particiones de conjuntos de train/test podríamos estar viendo valores en escala distinta provocado por la tendencia o incremento de la varianza

- Podemos calcular la correlación de una serie temporal con una copia de sí misma desplazada en el tiempo
- Para identificar posibles patrones periódicos que están enmascarados en la variación residual
- Para una serie $X = x_1, x_2, \dots$ se define como:

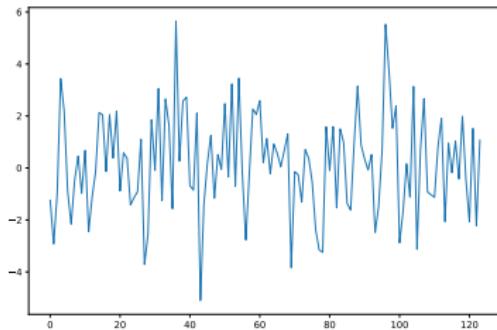
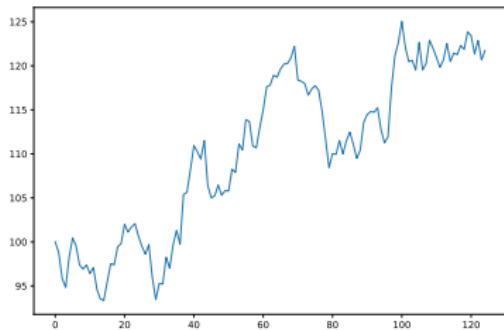
$$\rho(k) = \frac{[E(x_i - \mu)(x_{i-k} - \mu)]}{\text{var}(X)}$$

- sin tendencia
- de varianza constante
- las variaciones periódicas han sido eliminadas



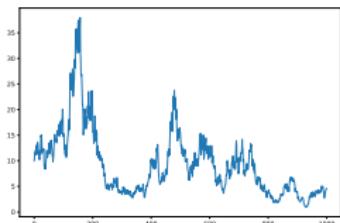
- **Diferenciación:** La serie se expresa en términos de la diferencia entre sus valores, para **remover la tendencia**.

$$\Delta x_t = x_t - x_{t-1}$$

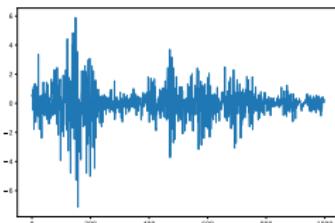


- **Transformación logarítmica:** la serie se representa con el logaritmo de sus valores, por ejemplo para remover la heterocedasticidad

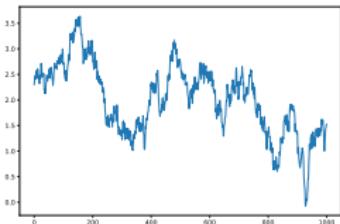
Original



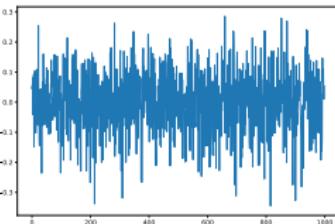
Diferenciada



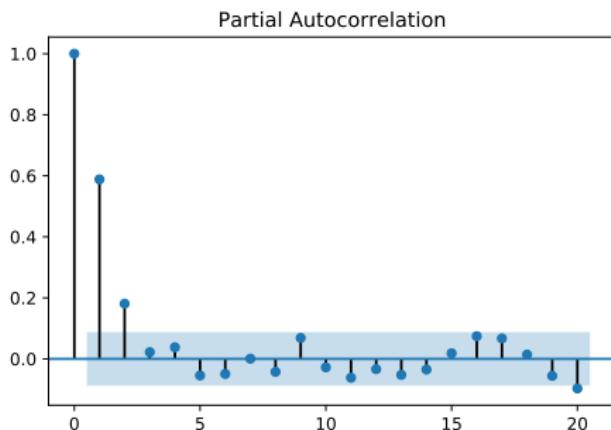
Log



Log + Diff



- **PACF** (Función de auto-correlación parcial): Calcula la auto-correlación de una serie estacionaria respecto a sus retardos de distancia k , eliminando el retardo de los retardos anteriores
- Un correograma de PACF muestra los retardos linealmente relevantes de la serie



Proceso Auto-regresivo

- El valor de la serie depende linealmente de los v valores anteriores de la propia serie
- v determina el orden del proceso
- un proceso AR(1):

$$y_t = \phi y_{t-1} + \varepsilon$$

Proceso de Media Móvil

- El valor de la serie depende linealmente de los v términos de error anteriores de la propia serie
- v determina el orden del proceso
- un proceso MA(1):

$$x_t = \varepsilon_t + \theta \varepsilon_{t-i}$$

Extracción de Ejemplos de Entrenamiento

- Horizonte de predicción
 - ▶ Fijo: $t + 1, t + 2, t + n$
 - ▶ Evento en el futuro dentro de un intervalo $[t, t + n]$
- Ventanas de observaciones $t, t - 1, t - m$
- Desplazamiento para ejemplos de entrenamiento $+n$

Date	Open	High	Low	Close	Volume	Adj.Close
2014-06-30	10.00	10.05	9.65	9.65	5412400	9.54
2014-07-01	9.77	10.30	9.75	10.08	2428600	9.96
2014-07-02	10.09	10.39	10.02	10.34	3400800	10.22
2014-07-03	10.35	10.65	10.29	10.45	7611100	10.33
2014-07-04	10.65	10.65	10.31	10.56	7480100	10.44
2014-07-07	10.50	10.52	10.40	10.49	1753200	10.37
2014-07-08	10.49	10.49	10.23	10.30	811700	10.18
2014-07-09	10.32	10.32	10.16	10.20	2118300	10.08
2014-07-10	10.23	10.25	10.00	10.00	271300	9.88
2014-07-11	10.33	10.33	10.00	10.00	122900	9.88
2014-07-14	10.14	10.14	9.89	10.05	1124200	9.93
2014-07-15	10.10	10.10	9.92	10.00	210600	9.88
2014-07-16	9.99	10.05	9.99	10.05	259700	9.93
2014-07-17	10.00	10.05	10.00	10.00	133300	9.88

Ventana $[t-3, t]$

t

Horizonte $[t+5]$

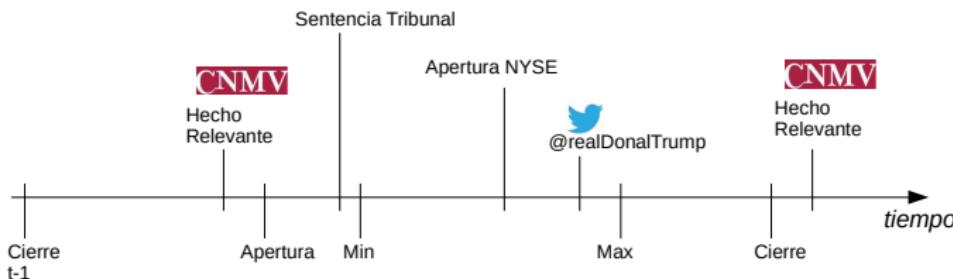
Extracción de Ejemplos de Entrenamiento

- Horizonte de predicción
 - ▶ Fijo: $t + 1, t + 2, t + n$
 - ▶ Evento en el futuro dentro de un intervalo $[t, t + n]$
- Ventanas de observaciones $t, t - 1, t - m$
- Desplazamiento para ejemplos de entrenamiento $+n$

Date	Open	High	Low	Close	Volume	Adj.Close
2014-06-30	10.00	10.05	9.65	9.65	5412400	9.54
2014-07-01	9.77	10.30	9.75	10.08	2428600	9.96
2014-07-02	10.09	10.39	10.02	10.34	3400800	10.22
2014-07-03	10.35	10.65	10.29	10.45	7611100	10.33
2014-07-04	10.65	10.65	10.31	10.56	7480100	10.44
2014-07-07	10.50	10.52	10.40	10.49	1753200	10.37
2014-07-08	10.49	10.49	10.23	10.30	811700	10.18
2014-07-09	10.32	10.32	10.16	10.20	2118300	10.08
2014-07-10	10.23	10.25	10.00	10.00	271300	9.88
2014-07-11	10.33	10.33	10.00	10.00	122900	9.88
2014-07-14	10.14	10.14	9.89	10.05	1124200	9.93
2014-07-15	10.10	10.10	9.92	10.00	210600	9.88
2014-07-16	9.99	10.05	9.99	10.05	259700	9.93
2014-07-17	10.00	10.05	10.00	10.00	133300	9.88

Desplazamiento [+2] → Ventana [t-3, t] → t → Horizonte [t+5]

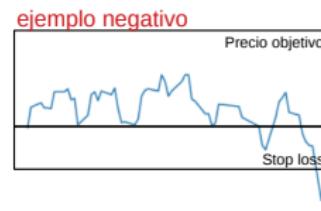
- La construcción de ejemplos implica simular la información que tendríamos disponible en el momento de haber tomado la decisión
- Utilizar información que en realidad corresponde con el futuro de la decisión se conoce como sesgo *look-ahead*
- Hay que considerar además el retardo en la distribución y las correcciones retrospectivas



- Predecir la rentabilidades es difícil porque en general tenemos un ratio bajo de señales reales respecto a ruido
- Simplificar la tarea y convertirla a un problema de clasificación ordinal

$$FDir_n(t) = \begin{cases} UP & FRet_n(t) \geq 0.06 \\ NEUTRAL & -0.06 \leq FRet_n(t) < 0.06 \\ DOWN & FRet_n(t) < -0.06 \end{cases}$$

- La predicción a un horizonte temporal fijo no considera la evolución de la serie hasta ese punto
 - Un etiquetado alternativo puede ser considerar si un *trade* será exitoso o no
-
- Etiquetado de 3 barreras:
 - ▶ un umbral superior (ej. precio objetivo)
 - ▶ un umbral inferior (ej. stop-loss)
 - ▶ y un horizonte temporal
 - La etiqueta la determina por cuál de las barreras ha salido la serie.
 - La barrera temporal podemos ignorarla o asignarla alguna de las otras clases



Motivación

- La evaluación debe considerar el aspecto temporal de los ejemplos
- Evaluar “a futuro”, tal como lo harían nuestras predicciones

Motivación

- La evaluación debe considerar el aspecto temporal de los ejemplos
- Evaluar “a futuro”, tal como lo harían nuestras predicciones

Procedimiento

- Se eligen ventanas temporales para entrenamiento y prueba
- Se realizan iteraciones sucesivas moviendo las ventanas hacia adelante
- Se estima el error real haciendo el promedio de los errores en cada iteración

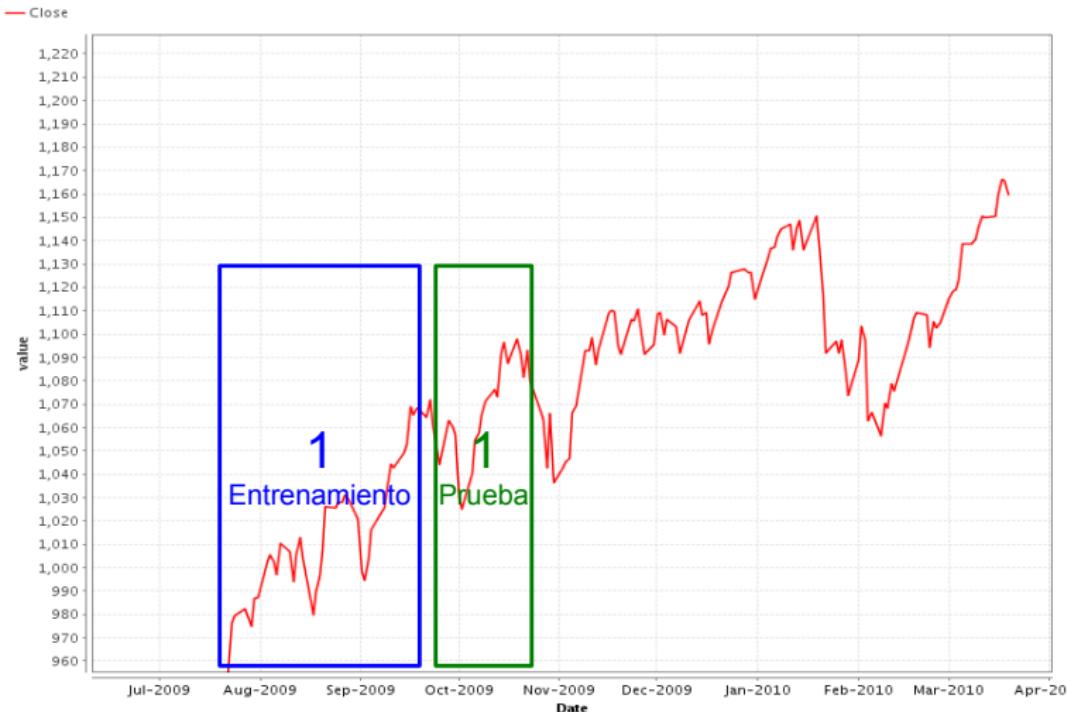
Validación de Ventanas Deslizantes



Validación de Ventanas Deslizantes



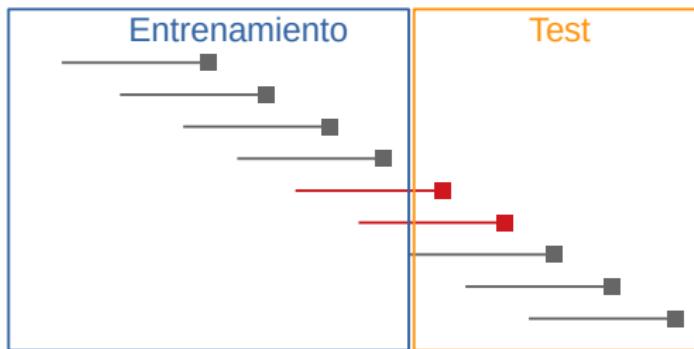
Aprendizaje Acumulativo



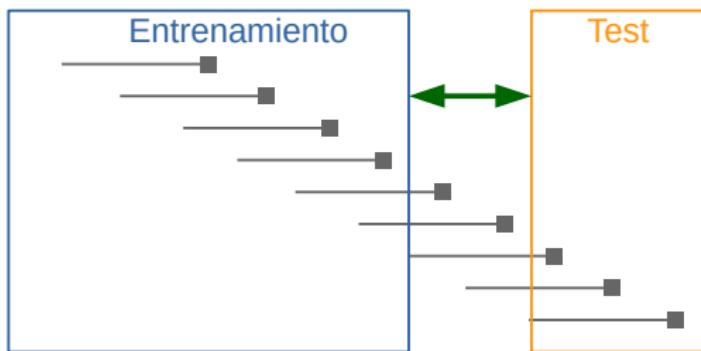
Aprendizaje Acumulativo



- Las características y las etiquetas se construyen en función de una ventana de información solapada en el tiempo. Podemos observar $Y_t \approx Y_{t+1}$
- En las particiones train/test del dataset podemos filtrar información que puede hacer mejorar artificialmente la evaluación



- Debemos asegurarnos que el conjunto de entrenamiento y test son realmente independientes en los que se evita filtración por solapamiento o por correlación serial
- Los ejemplos que podrían ocasionar la filtración se descartan



- Algoritmos de Bagging pueden preferirse sobre los de Boosting
 - ▶ En finanzas el problema relevante es el overfitting. Con Bagging nos centramos en reducir la varianza de los modelos
 - ▶ El bagging se paraleliza fácilmente. Con gran cantidad de datos tardaremos menos en la construcción del modelo
- Evitar el sesgo de selección al encontrar una estrategia rentable en un conjunto elevado de simulaciones
- Los datos históricos son sólo una realización posible de un conjunto de procesos con alto componente estocástico

Optimización de hiperparámetros

- Muchas técnicas de ML incluyen parámetros iniciales que podemos elegir y que afectan el rendimiento final del predictor
 - ▶ El número de k vecinos más cercanos en el KNN
 - ▶ El número de árboles en el Random Forest
 - ▶ Incluso un Auto-ML (elegir entre varios modelos de diferentes técnicas)
- Debemos elegir el mejor parámetro o algoritmo en función de alguna métrica que se evalúa fuera de la muestra de entrenamiento

- Muchas técnicas de ML incluyen parámetros iniciales que podemos elegir y que afectan el rendimiento final del predictor
 - ▶ El número de k vecinos más cercanos en el KNN
 - ▶ El número de árboles en el Random Forest
 - ▶ Incluso un Auto-ML (elegir entre varios modelos de diferentes técnicas)
- Debemos elegir el mejor parámetro o algoritmo en función de alguna métrica que se evalúa fuera de la muestra de entrenamiento

Advertencia

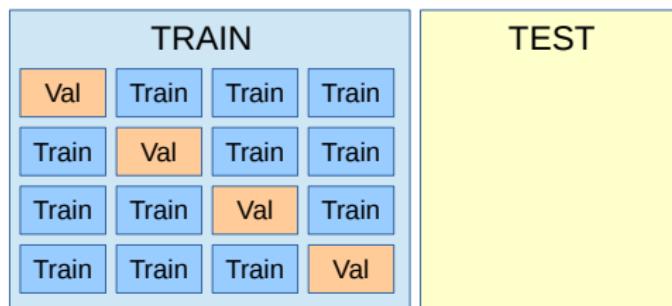
- Una validación de parámetros debe realizarse sobre una muestra distinta de la utilizada para estimar el error de un modelo para evitar sesgo de selección

Diseño Experimental

- Dataset dividido en 3 conjuntos
 - 1 Conjunto de entrenamiento: Para entrenar el modelo
 - 2 Conjunto de validación: Para optimizar/buscar los hiper-parámetros
 - 3 Conjunto de prueba: Para estimar el error del modelo

Diseño Experimental

- Dataset dividido en 3 conjuntos
 - 1 Conjunto de entrenamiento: Para entrenar el modelo
 - 2 Conjunto de validación: Para optimizar/buscar los hiper-parámetros
 - 3 Conjunto de prueba: Para estimar el error del modelo
- Esquema anidado de validación cruzada en train/test para validar hiper-parámetros dentro del conjunto de train

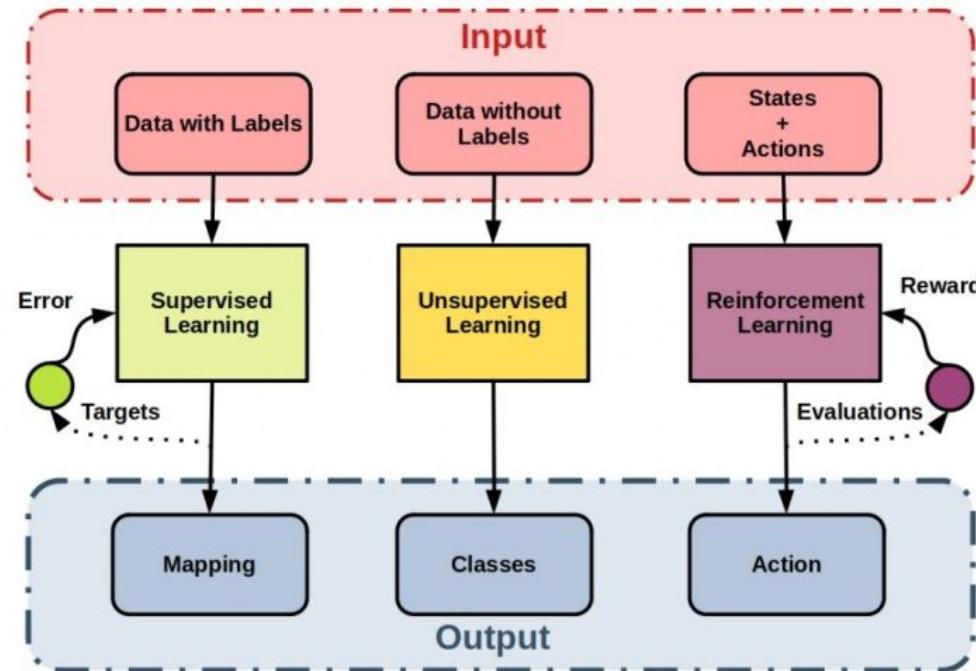


- La optimización de hiper-parámetros es un proceso de búsqueda que consta de:
 - 1 un algoritmo de ML o un estimador
 - 2 un espacio de parámetros
 - 3 un algoritmo de búsqueda
 - 4 un esquema de validación (train/valid) o validación cruzada
 - 5 una métrica de evaluación

- La optimización de hiper-parámetros es un proceso de búsqueda que consta de:
 - 1 un algoritmo de ML o un estimador
 - 2 un espacio de parámetros
 - 3 un algoritmo de búsqueda
 - 4 un esquema de validación (train/valid) o validación cruzada
 - 5 una métrica de evaluación
- Algoritmos de búsqueda
 - ▶ Búsqueda exhaustiva (*Grid Search*): Probar todas las posibles combinaciones de parámetros
 - ▶ Búsqueda avariciosa: Se decide un orden de los parámetros. En cada paso se elige el mejor modelo fijando los parámetros previos
 - ▶ Búsqueda Estocástica: Se busca aleatoriamente sobre el espacio de parámetros que sigue una distribución de posibles valores. En cada iteración se almacena la mejor solución hasta el momento

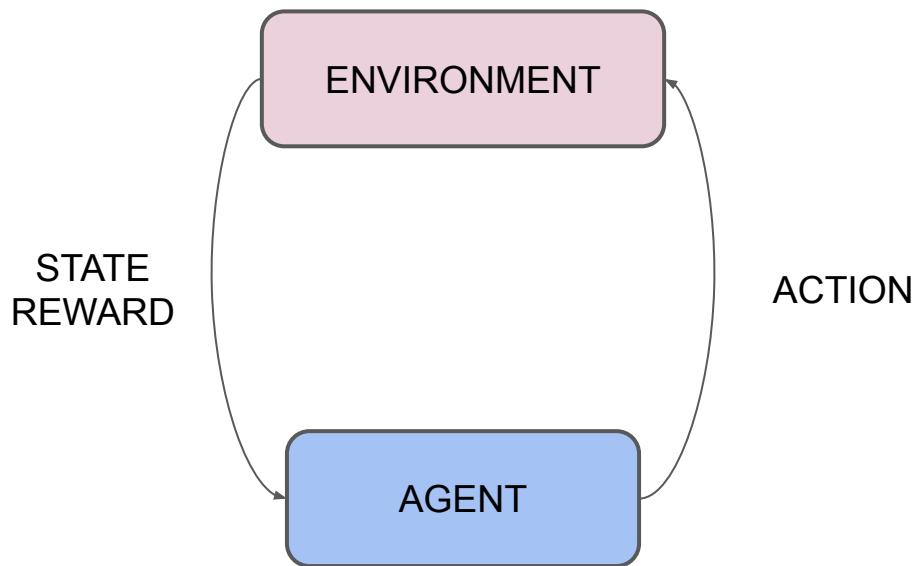
Aprendizaje por refuerzo

Machine Learning and Reinforcement Learning



[<https://starship-knowledge.com/supervised-vs-unsupervised-vs-reinforcement>]

Agent-environment interaction



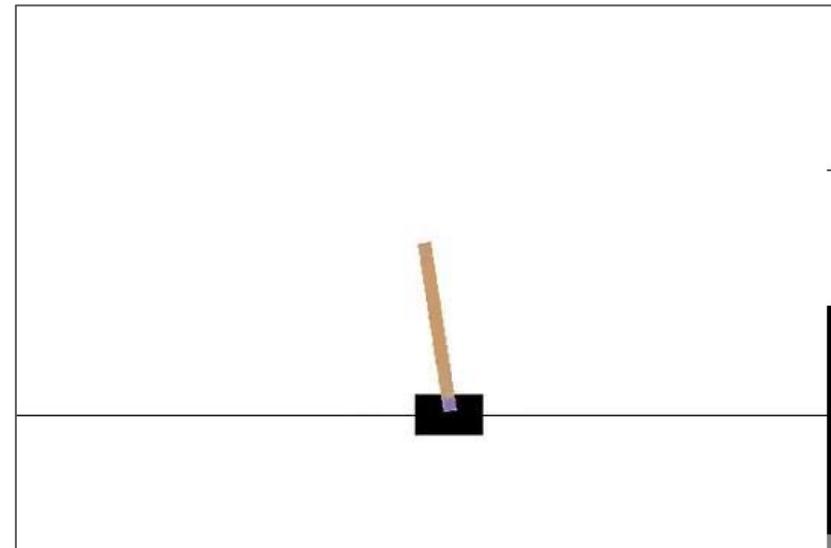
Cart-pole

Keep the pole in upright position

State: angle, angular speed, position, horizontal velocity

Action: horizontal force (+1, -1) applied on the cart

Reward: +1 for every timestep that the pole remains upright



[<https://gym.openai.com/videos/2019-10-21--mqt8Qj1mwo/CartPole-v1/original.mp4>]

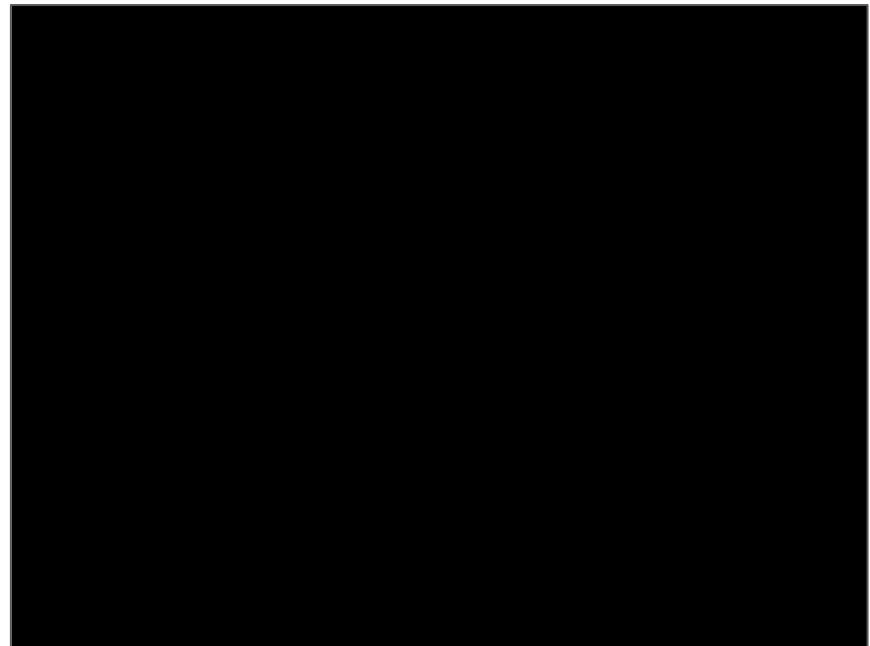
Atari Breakout

Maximize score in the game

State: RGB image of the screen, which is an array of shape (210, 160, 3)

Action: move right/left

Reward: score



[<https://gym.openai.com/videos/2019-10-21--mqt8Qj1mwo/Breakout-v0/original.mp4>]

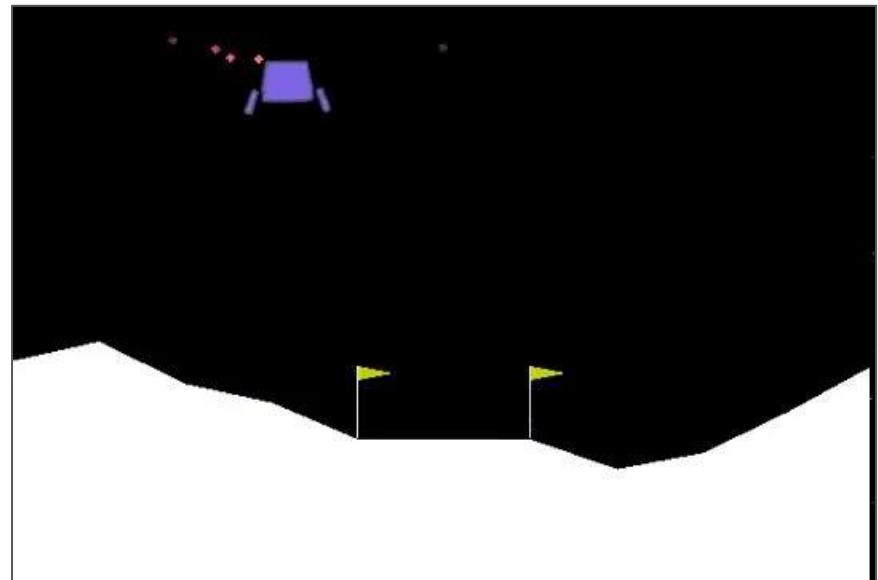
Lunar lander

Land the capsule between flags

State: position, angle, velocity...

Action: fire different engines

Reward: score that depends on performance



[<https://gym.openai.com/videos/2019-10-21--mqt8Qj1mwo/LunarLander-v2/original.mp4>]

Model, value, policy

- **Model:** the agent's representation of the environment
- **Value function:** function that quantifies how good is each state or state-action pair
- **Policy:** function that determines the agent's behavior (which action to take for a given state)

Discounted future reward

The value associated to a state s is the expected reward that will be obtained in the future when starting from s

$$\tau = (s_1, a_1, s_2, \dots), \quad X_t(\tau) = s_t, \quad Y_t(\tau) = a_t,$$

$$G(\tau) = \sum_{t=1}^{\infty} \gamma^{t-1} R(X_t(\tau), Y_t(\tau)),$$

Value

Value function (for a policy π):

$$V^\pi(s) = \mathbb{E}^\pi[G|X_1 = s]$$

Action-value function (for a policy π):

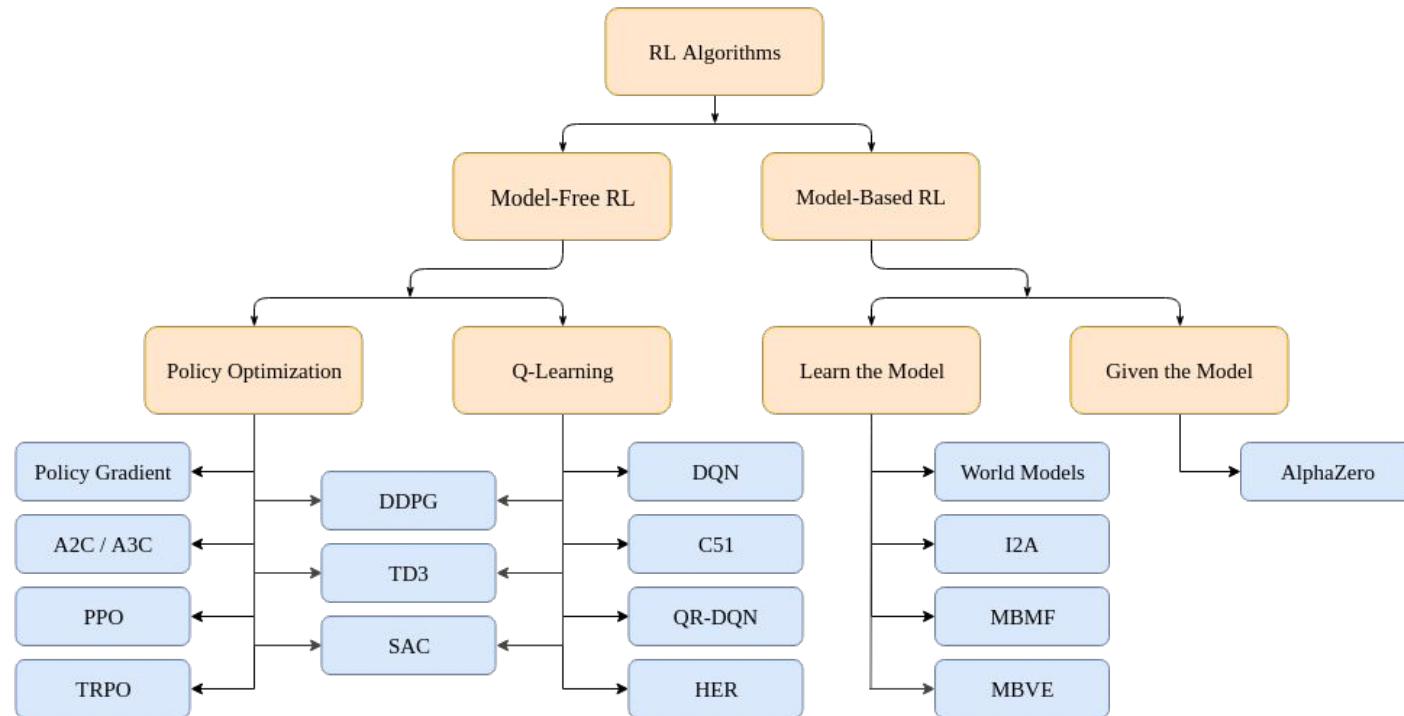
$$Q^\pi(s, a) = \mathbb{E}^\pi[G|X_1 = s, Y_1 = a].$$

Exploration vs exploitation

ϵ -greedy strategy:

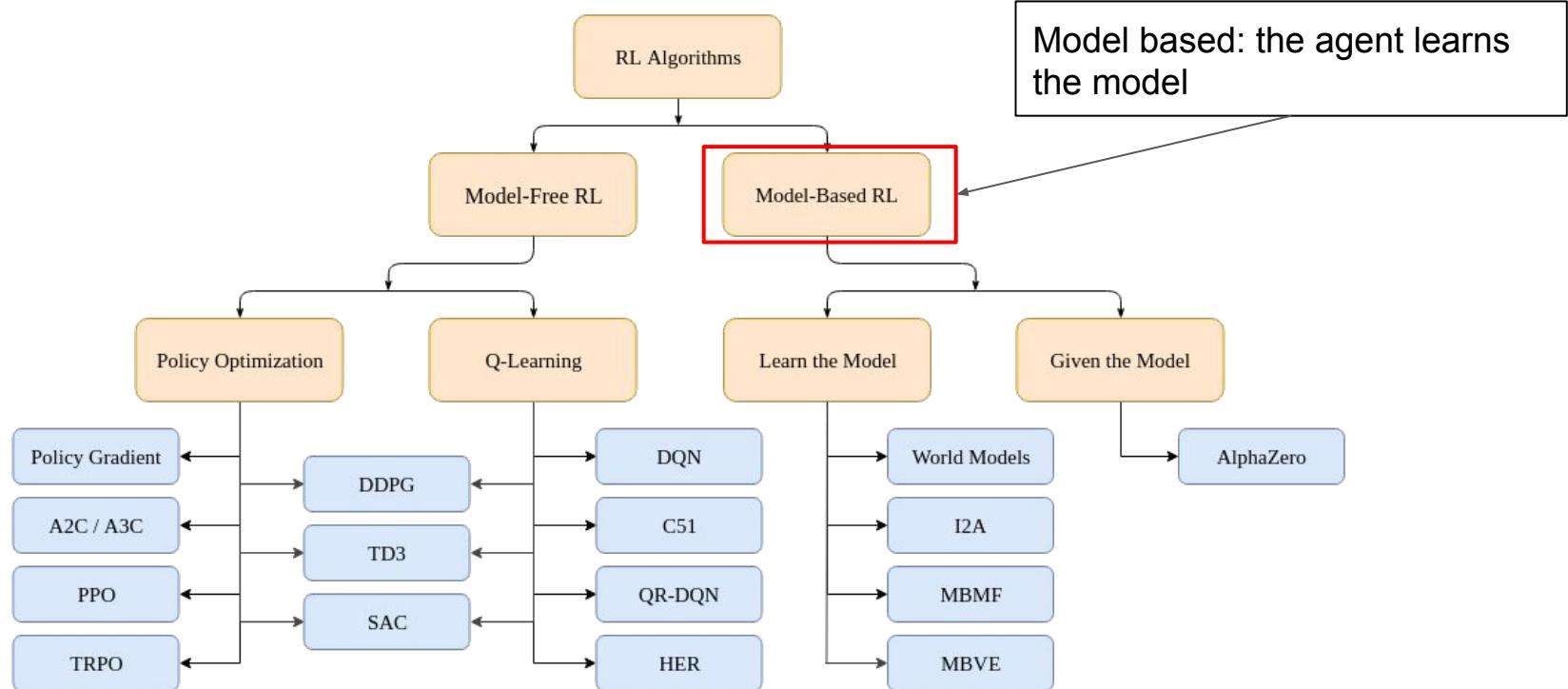
- Take a random action with probability $\epsilon \in (0, 1)$
- Take the optimal action (according to the policy) with probability $1 - \epsilon$

Taxonomy of RL algorithms



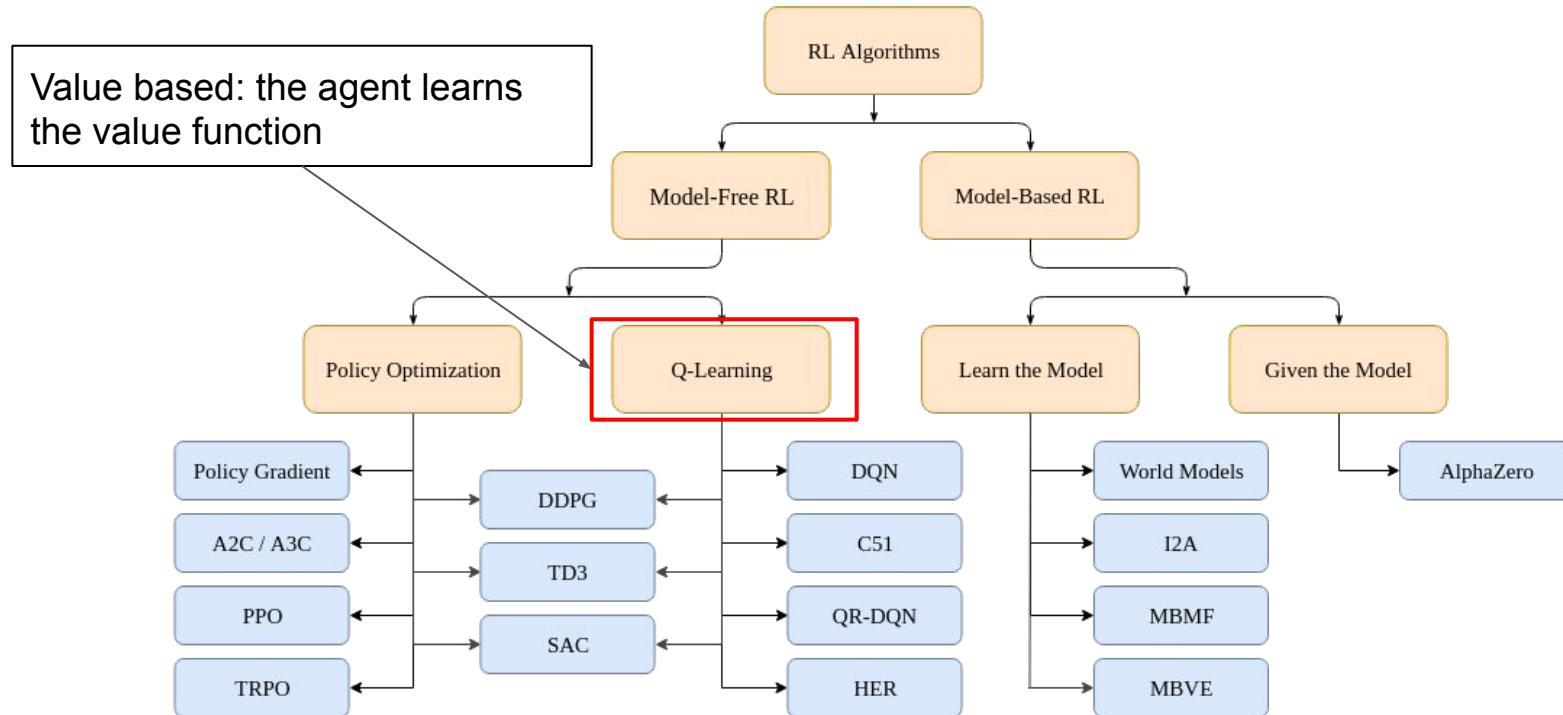
[Image from https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html]

Taxonomy of RL algorithms



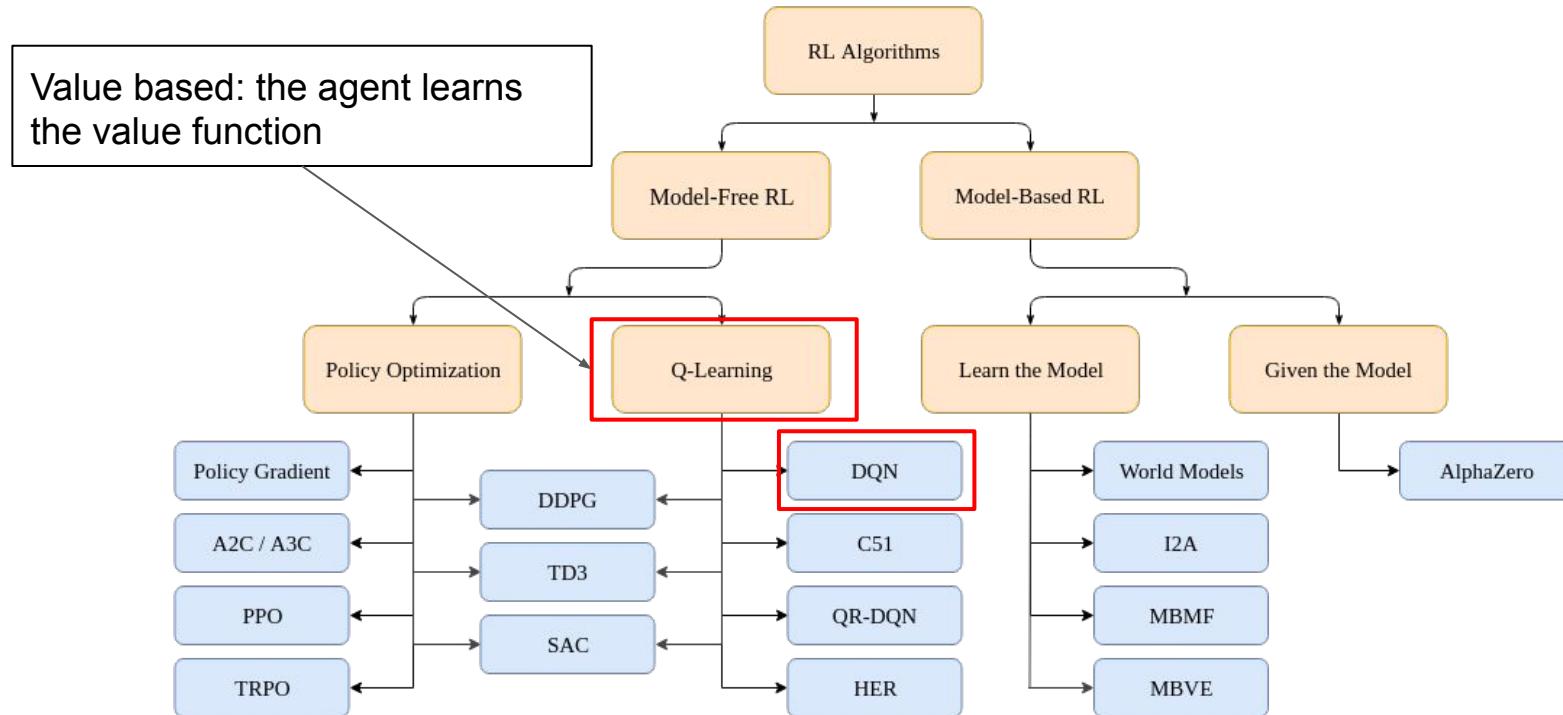
[Image from https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html]

Taxonomy of RL algorithms



[Image from https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html]

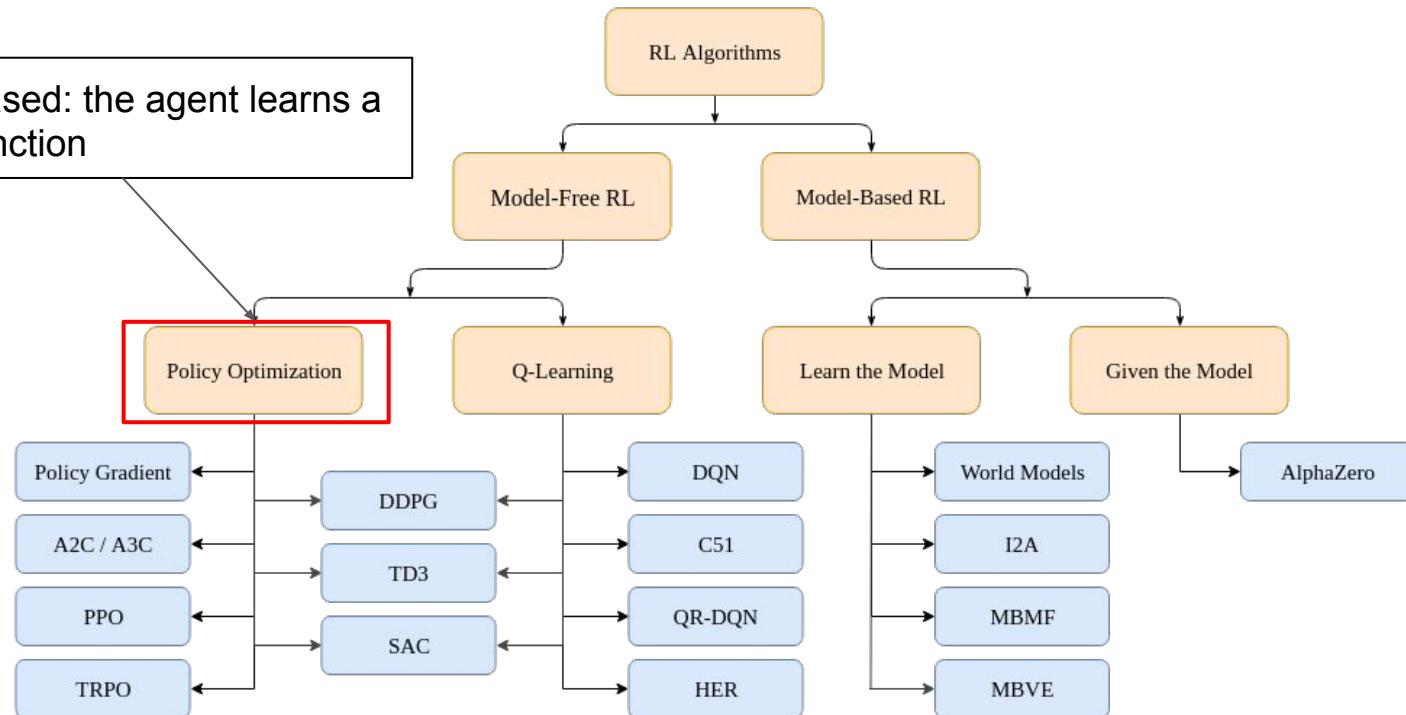
Taxonomy of RL algorithms



[Image from https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html]

Taxonomy of RL algorithms

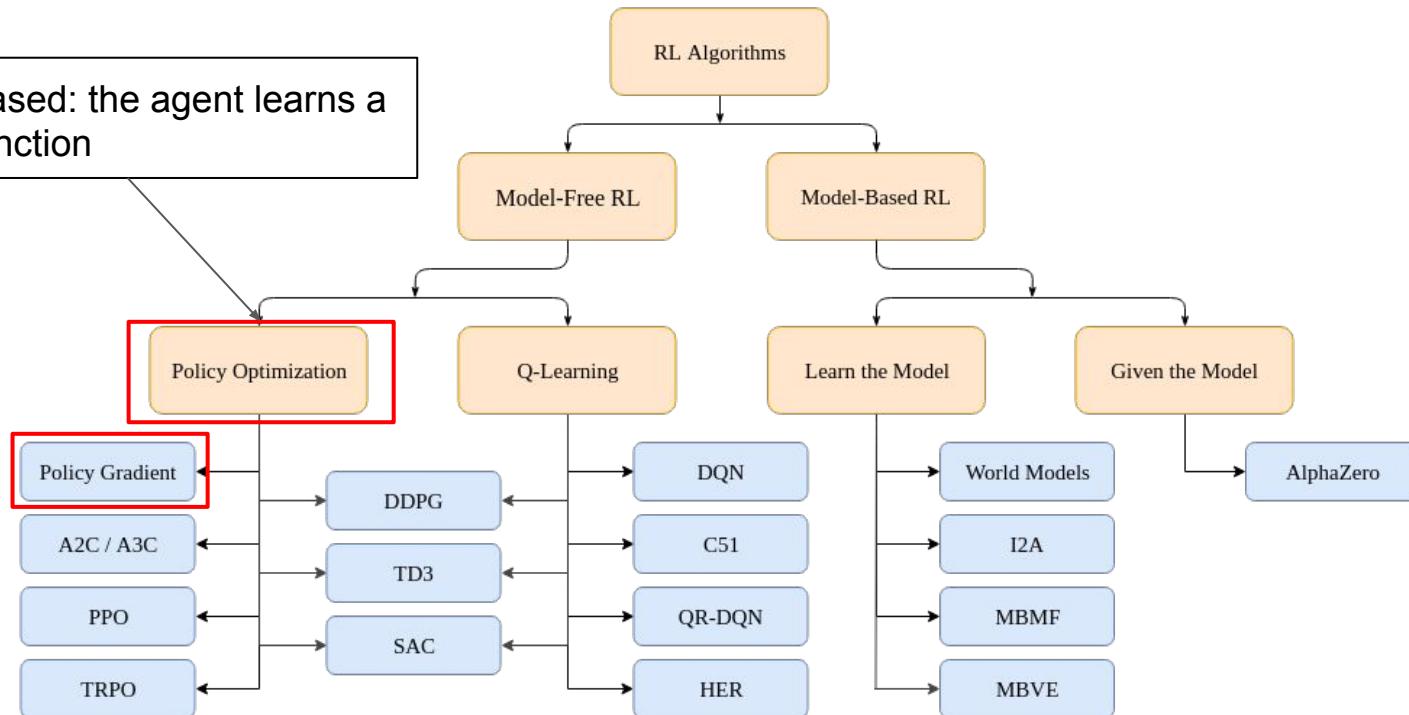
Policy based: the agent learns a policy function



[Image from https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html]

Taxonomy of RL algorithms

Policy based: the agent learns a policy function



[Image from https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html]

On-policy vs off-policy

On-policy: The policy that interacts with the environment (that used to make decisions) and the policy that we optimize are the same.

Off-policy: The policy that interacts with the environment (that used to make decisions) and the policy that we optimize are different.

Classic RL

Q-learning

Sarsa

- Both are value-based and model free.
- Sarsa is on-policy.
- Q-learning is off-policy.

Q-learning and Sarsa

TD-error: Transition (s, a, r, s')

$$\delta = r + \gamma V(s') - V(s),$$

$$\delta = r + \gamma Q(s', a') - Q(s, a).$$

Sarsa

Given (s, a, r, s') :

- Choose a' according to policy, with an ϵ -greedy strategy.
- Update table according to:

$$Q(s, a) \leftarrow Q(s, a) + \alpha(r + \gamma Q(s', a') - Q(s, a)),$$

- Execute action a' in environment.

Q-learning

Given (s, a, r, s') :

- Update table according to:

$$Q(s, a) \leftarrow Q(s, a) + \alpha(r + \gamma \max_{a' \in A} Q(s', a') - Q(s, a)),$$

- Choose a' according to policy, with an ϵ -greedy strategy.
- Execute action a' in environment.

Q-Learning

The state-action value function is a table:

$Q^\pi(s, a)$ = expected future reward when starting from state s , performing action a and following policy π

	s_1	s_2	s_3	...	s_k
a_1					
...					
a_n					

Q-Learning

The state-action value function is a table:

$Q^\pi(s, a)$ = expected future reward when starting from state s , performing action a and following policy π

	s_1	s_2	s_3	...	s_k
a_1					
...					
a_n					

Q-Learning (exploitation vs exploration)

Given a state s :

- The agent selects the action that maximizes the future reward with probability $1 - \epsilon$ (exploitation)
- The agent selects a random action with probability ϵ (exploration)

	s_1	s_2	s_3	...	s_k
a_1					
...					
a_n					

Q-Learning

The agent learns by updating the table according to

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t)(1 - \alpha) + \alpha[R_{t+1} + \gamma \max_a Q_t(s_{t+1}, a)]$$

	s_1	s_2	s_3	...	s_k
a_1					
...					
a_n					

Q-Learning

The agent learns by updating the table according to

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t)(1 - \alpha) + \alpha[R_{t+1} + \gamma \max_a Q_t(s_{t+1}, a)]$$

Learning rate

	s_1	s_2	s_3	...	s_k
a_1					
...					
a_n					

Q-Learning

The agent learns by updating the table according to

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t)(1 - \alpha) + \alpha[R_{t+1} + \gamma \max_a Q_t(s_{t+1}, a)]$$

Discount factor

	s_1	s_2	s_3	...	s_k
a_1					
...					
a_n					

Q-Learning algorithm

1. Init table $Q(s, a)$
2. Repeat:
 - a. Select action a
 - b. Get reward r and next state s'
 - c. Update $Q(s, a) = Q(s, a)(1 - \alpha) + \alpha[r + \gamma \max_a Q(s', a')]$
 - d. Set $s = s'$

Q-Learning algorithm

1. Init table $Q(s, a)$
2. Repeat:
 - a. Select action a
 - b. Get reward r and next state s'
 - c. Update $Q(s, a) = Q(s, a)(1 - \alpha) + \alpha[r + \gamma \max_a Q(s', a')]$
 - d. Set $s = s'$

Problem: impractical in real situations: the table is too big!!!

Q-learning example

EXERCISE 5 (1 point): The following table shows the transitions for a reinforcement learning model with 4 states and 2 possible actions. The table gives, for a given state-action pair, the next state and the associated reward. For example, starting in state s_0 and performing action a_1 the agent jumps to state s_1 and gets a reward of 100 units.

	s_0	s_1	s_2	s_3
a_1	$s_1 / 100$	$s_2 / 0$	$s_3 / 50$	$s_0 / -100$
a_2	$s_3 / 50$	$s_0 / -100$	$s_1 / 100$	$s_2 / 0$

Departing from the Q -table displayed below, where all the expected future rewards are initialized to 0, compute the Q -table values when the agent starts at state s_0 and performs the actions a_1, a_2, a_1 . The learning rate is $\alpha = 0.8$ and the discount factor is $\gamma = 0.8$.

Q	s_0	s_1	s_2	s_3
a_1	0	0	0	0
a_2	0	0	0	0

What if the Sarsa algorithm is used instead?