

# Entrenament d'un Model amb Docker i Kubernetes [↗](#)

---

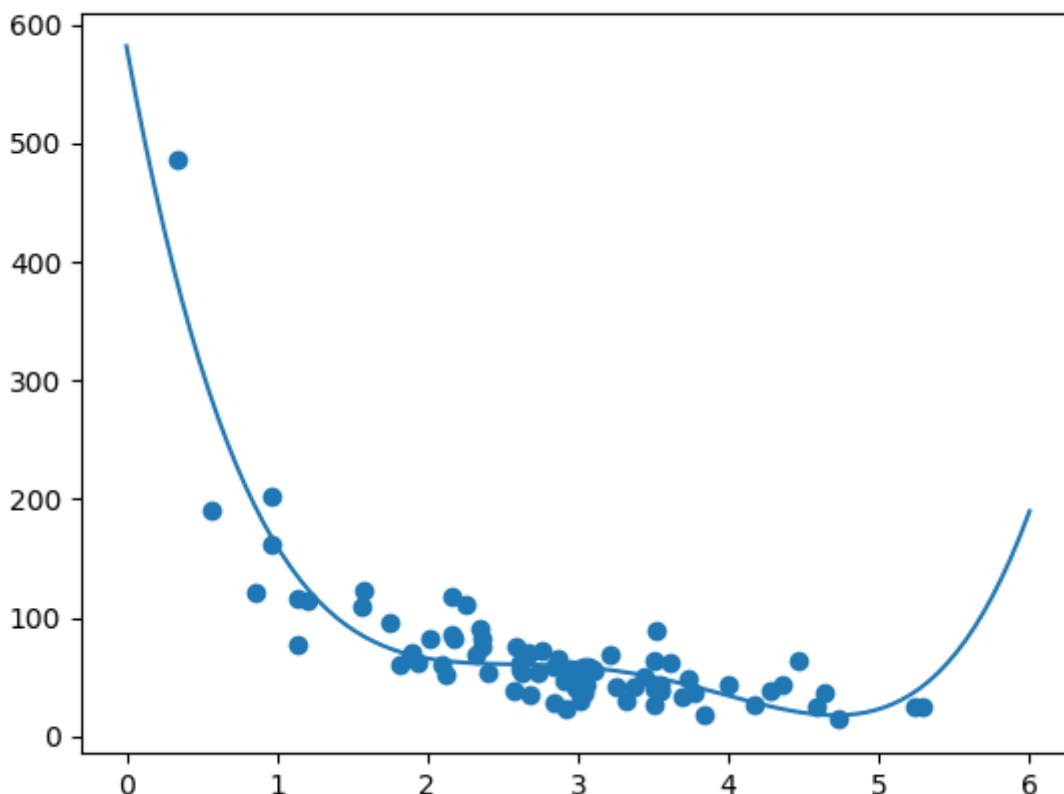
## El Model [↗](#)

---

El nostre conjunt de dades il·lustra 100 clients en una botiga i els seus hàbits de compra.

- L'eix x representa el nombre de minuts abans de fer una compra.
- L'eix y representa la quantitat de diners gastats en la compra.

Un cop el model estigui entrenat, es veurà així:



**Nota:** Aquest és l'exemple de model per a aquest exercici. No obstant això, podeu utilitzar el vostre propi model. Fer això comptarà per aconseguir 10 punts en aquest exercici.

Font del model: [https://www.w3schools.com/python/python\\_ml\\_train\\_test.asp](https://www.w3schools.com/python/python_ml_train_test.asp)

## Tasques

---

L'objectiu d'aquest exercici serà crear un servei per exposar un model entrenat en ciència de dades. Això constarà de 3 tasques.

### 1. Configuració i Instal·lació de Dependències

En primer lloc, necessitarem instal·lar les dependències. Aquestes són les dependències que cal instal·lar:

1. Docker
  - Instal·lació: <https://docs.docker.com/engine/install/ubuntu/>
  - Afegir al grup sudo: <https://docs.docker.com/engine/install/linux-postinstall/>
2. Kubectl:
  - Instal·lació: <https://kubernetes.io/docs/tasks/tools/install-kubectl-linux/>
  - Afegir autocompletament: <https://kubernetes.io/docs/tasks/tools/install-kubectl-linux/#enable-shell-autocompletion>
3. Minikube:
  - Instal·lació: <https://minikube.sigs.k8s.io/docs/start/>
  - Iniciar: `minikube start` (Això podria frenar el vostre VM, podeu fer-ho només quan desplegueu l'aplicació a la fase 3)

### 2. Crear Aplicació i Imatges Docker

Per a aquest exercici, necessitarem crear 2 imatges Docker.

1. Una aplicació de feina o "script" que bàsicament entreni un model i el desa en un disc. La imatge hauria d'estar etiquetada com `model-train:default`.
2. Una aplicació de servei amb Flask que farà el següent:
  - A `/` carregarà una pàgina HTML senzilla que expliqui com funciona el servei.
  - A `/model` carregarà el model i rebrà els paràmetres del model per a retornar la sortida del model com a JSON.
  - La imatge hauria d'estar etiquetada com `model-server:default`.

**Important:** En construir les imatges Docker, tingueu en compte les bones pràctiques de Docker: [https://docs.docker.com/develop/develop-images/dockerfile\\_best-practices/](https://docs.docker.com/develop/develop-images/dockerfile_best-practices/) Això afectarà la vostra puntuació! Consulteu la secció de puntuació.

### 3. Desplegar l'Aplicació a Kubernetes 🔗

Finalment, desplegarem l'aplicació al nostre entorn de "Producció" a Kubernetes. Heu de proporcionar els següents recursos de Kubernetes:

1. Feina: <https://kubernetes.io/docs/concepts/workloads/controllers/job/>
  - Entrenarà el model i el desarà en un volum.
  - Necessita tenir una variable d'entorn `MODEL_PATH` en la qual s'emmagatzemarà el model (<https://kubernetes.io/docs/tasks/configure-pod-container/configure-pod-configmap/>)
  - Cal que tingui sol·licituds i límits de recursos definits (<https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>)
2. Desplegament: <https://kubernetes.io/docs/concepts/workloads/controllers/deployment/>
  - Aplicació que servirà el model entrenat.
  - Necessita tenir 3 rèpliques.
  - Necessita tenir una variable d'entorn `MODEL_PATH` des de la qual es llegirà el model (<https://kubernetes.io/docs/tasks/configure-pod-container/configure-pod-configmap/>)
  - Cal que tingui sol·licituds i límits de recursos definits (<https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>)
  - Cal que tingui probes de salut i preparació (<https://kubernetes.io/docs/tasks/configure-pod-container/configure-liveness-readiness-startup-probes/>)
3. Servei: <https://kubernetes.io/docs/concepts/services-networking/service/>
  - Es crearà per servir les 3 rèpliques diferents del desplegament.
4. ConfigMap: <https://kubernetes.io/docs/concepts/configuration/configmap/>
  - Emmagatzemarà un valor tant per a la Feina com per al Desplegament per saber on desar o llegir el model.

**Nota:** Per accedir al servei podeu fer `kubectl port-forward service/model-server 5000:5000` Més informació: <https://kubernetes.io/docs/tasks/access-application-cluster/port-forward-access-application-cluster/>

# Camí Recomanat

---

## Sessió 1:

- Acabar d'instal·lar totes les dependències (Tasca 1)
- Crear el Dockerfile `model-train:default` (Tasca 2.1)

## Sessió 2:

- Crear el Dockerfile `model-server:default` (Tasca 2.2)
- Crear la Feina de Kubernetes (Tasca 3.1)
- Crear el ConfigMap de Kubernetes (3.4)

## Sessió 3:

- Crear el Desplegament de Kubernetes (Tasca 3.2)
- Crear el Servei de Kubernetes (Tasca 3.2)
- Assegurar-vos que tot funcioni bé

**Nota:** Per "moure" les imatges Docker del vostre ordinador local a Minikube, heu de fer el següent: `minikube image load model-train:default` i `minikube image load model-server:default`

## Puntuació

---

### Lliurament (2 Punts)

L'exercici lliurat ha de contenir tot això per obtenir els 2 punts.

- Documentació per a l'exercici:
  - Expliqueu com desplegar la vostra solució: (Com construir imatges, desplegar el codi a Kubernetes, etc.)
  - Captures de pantalla del vostre aplicatiu funcionant.
  - Quines dificultats heu enfrontat durant aquest exercici?
  - Què heu après d'aquest exercici?
  - Retroalimentació per al professor (No s'avaluarà en la puntuació).
- Crear fitxers Docker per a les dues aplicacions
- Seguir les bones pràctiques de Docker
- Construir amb èxit les dues aplicacions localment amb les 2 imatges Docker
- Crear amb èxit la Feina de Kubernetes escrivint el model en un volum

- Crear amb èxit el Desplegament de Kubernetes que serveixi l'API i llegeixi el model
- Crear amb èxit el Servei de Kubernetes i accedir-hi, podent arribar al desplegament
- Llegir/escriure amb èxit el model utilitzant la variable d'entorn ConfigMap

### **Examen (6 Punts):** 🔗

Per a l'examen, crearem exercicis basats en el lliurament, si heu fet el lliurament no haureu de tenir problemes per aprovar l'examen.

### **Contingut/Excel·lència (2 Punts)** 🔗

Es donaran 2 punts addicionals basats en el contingut del vostre exercici. Fer una o moltes de les següents accions us donarà l'opció d'obtenir la puntuació més alta per a aquest exercici.

- Canviar el model proporcionat i mostrar com després d'executar la feina el servidor API retorna un valor diferent
- Lliurar tot en un repositori Git (amb un historial de commits, un sol commit amb totes les versions "finals" no compta)
- Construir la mateixa aplicació amb el vostre model o un altre model diferent de l'aproposat