

MLOps



Multimodal Product Data Classification

Equipe projet:

- Fadimatou Abdoulaye
- Olivier Renouard

Agenda

1. Contexte & Objectifs
2. Data sets et Modèles
3. Composants et fonctionnalités
4. Architecture de la plateforme
5. Démonstration
6. Améliorations futures et Conclusion

1 - Contexte et Objectifs

Contexte

- Rakuten propose une **place de marché** où les **particulier et les professionnel** peuvent vendre de nombreux produits.
- Rakuten ne gère aucun stock

Rakuten en chiffres

- **12 millions** de membres inscrits dont 10 000 vendeurs professionnels et partenaires.
- Environ **200 millions** de produits référencés / Environ **50 millions** en seconde main
- **30 000 à 50 000** transactions / jour : **30 %** « C to C » **70 %** « B to B to C »
- **15 millions** de visiteurs uniques / mois sur le site
- **300** salariés de Rakuten France

Objectifs

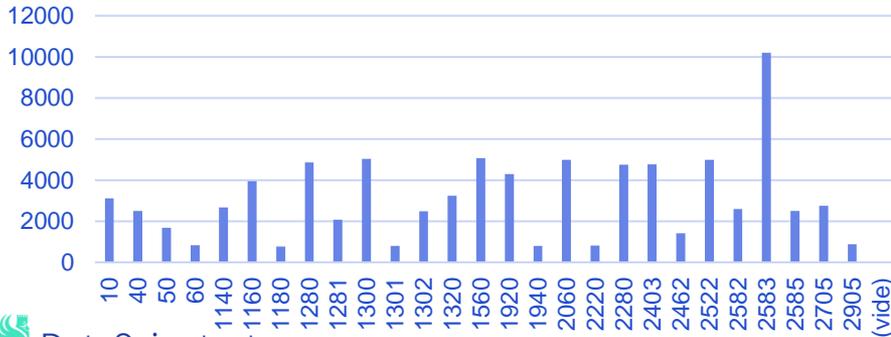
- Pour que la plateforme soit gérée avec aussi peu de personnes (300), il est nécessaire d'automatiser un grand nombre de processus.
- Un produit en vente mal classé peut devenir redondant et ses chances d'être vendu diminuent
- La plateforme met en œuvre 2 grandes fonctionnalités:
 - La prédiction de la catégorie d'un produit
 - Le réentraînement du modèle de prédiction en fonction des nouveaux produits et de ses performances

2 - Datasets et Modèles

Datasets

- Le Dataset **Train** comporte **84916 produits** avec leur désignation et description (Texte) ainsi que leur photo (Image)
- Le Dataset **Test** comporte **13810 produits**
- Seuls les produits du Dataset Train ont été classé en **27 catégories**
- Avec un tel déséquilibre dans les catégories, il est souhaitable de mesurer et surveiller le **F1 score**

Train: Nombre de produits par catégorie



Texte

input_1	input:	[(None, 50)]
InputLayer	output:	[(None, 50)]

embedding	input:	(None, 50)
Embedding	output:	(None, 50, 512)

bidirectional(gru)	input:	(None, 50, 512)
Bidirectional(GRU)	output:	(None, 1024)

dropout	input:	(None, 1024)
Dropout	output:	(None, 1024)

dense	input:	(None, 1024)
Dense softmax	output:	(None, 27)

Couche dense supplémentaire pour le fine-tuning

Image

VGG16

flatten	input:	(None, 7, 7, 512)
Flatten	output:	(None, 25088)

dense_1	input:	(None, 25088)
Dense relu	output:	(None, 512)

dense_2	input:	(None, 512)
Dense relu	output:	(None, 512)

dense_3	input:	(None, 512)
Dense softmax	output:	(None, 27)

Concaténation

(addition des proba pondérées)

RNN TXT

CNN IMG

input_10	input:	[(None, 27)]
InputLayer	output:	[(None, 27)]

input_11	input:	[(None, 27)]
InputLayer	output:	[(None, 27)]

lambda_2	input:	[(None, 27), (None, 27)]
Lambda	output:	(None, 27)

3 – Composants et Fonctionnalités

- Un repo [Github](#) organisé et documenté
- **4 API** fastAPI sécurisées avec des tokens OAuth2 comportant **13 endpoints**
- Une **base de données Utilisateurs MySQL**
- Un ensemble de **16 tests** qui testent les [autorisations et les principales API](#) avec **Github Actions** lors des pushes ou des pull request dans tous les branches
- Une conteneurisation de toute la plateforme avec **15 conteneurs Docker** construite et lancée grâce à une seule commande (setup.sh)
- Une fonctionnalité **Tensorboard** pour analyser l'entraînement des modèle Texte et Image
- Une fonctionnalité **Airflow** pour alerter l'administrateur en cas de baisse de performance du modèle, mettre à jour automatiquement le modèle, tester certaines API dans l'environnement de PréProd
- Une application **Streamlit** qui donne accès à (presque) toutes ces fonctionnalités

POST FastAPI OAuth2 token generate

GET FastAPI OAuth2 secured

GET New Products Proposal

GET Predict initialization

POST Predict (secured or not)

GET Add_new_products

GET Compute metrics for new prod...

GET Save model & start train

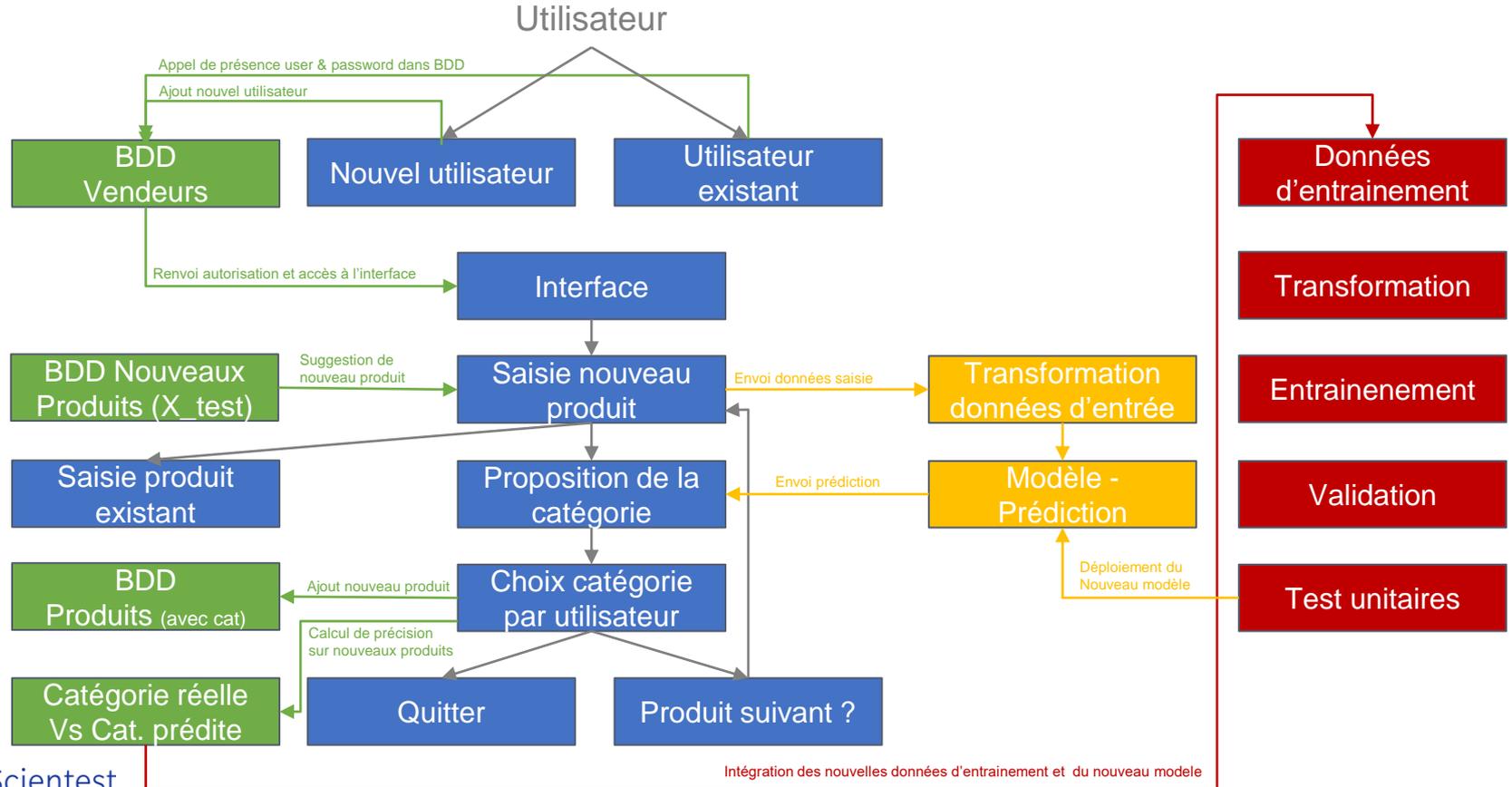
POST Train (secured or not)

GET Reset Datasets

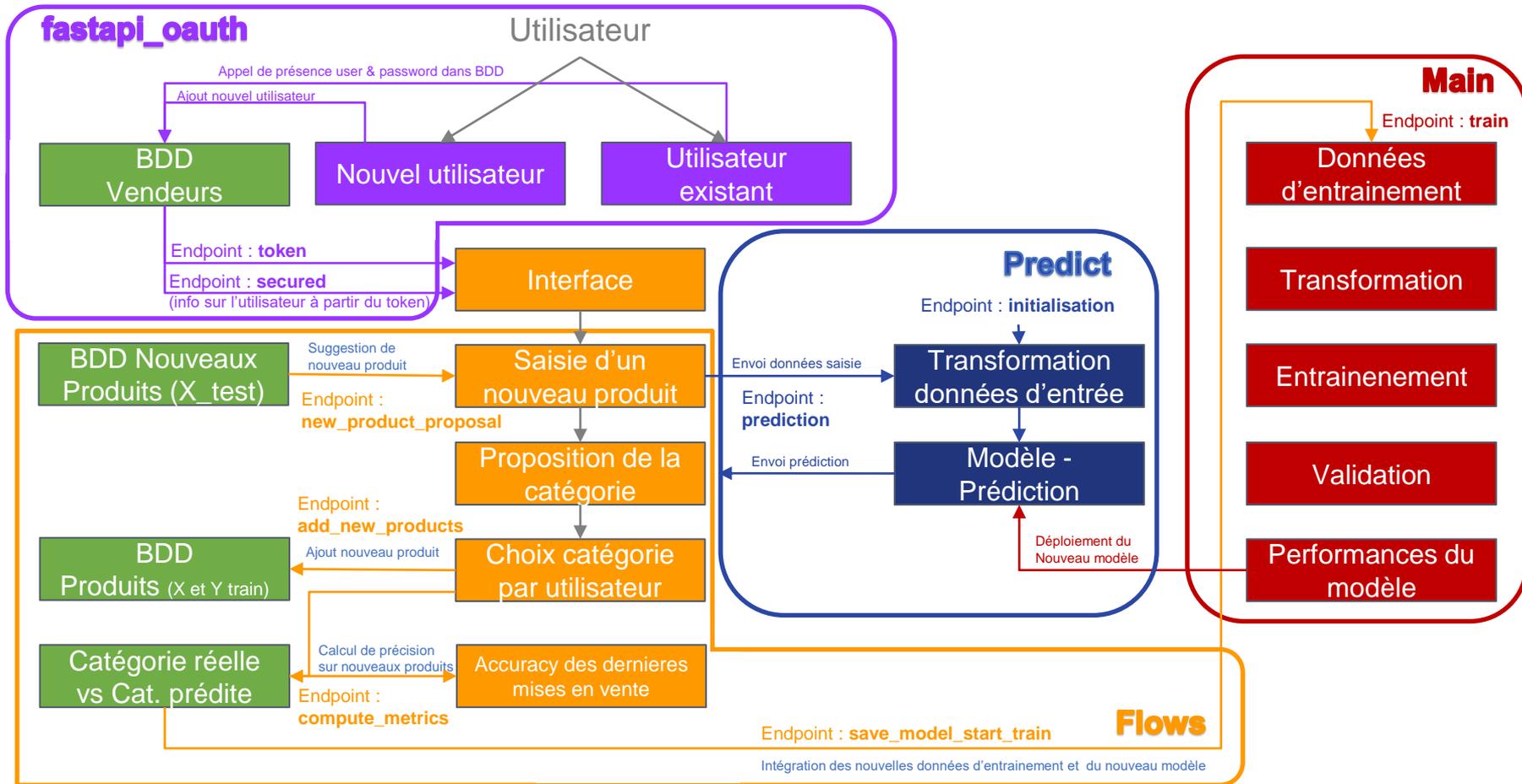
POST FastAPI OAuth2 Users

4 – Architecture de la plateforme

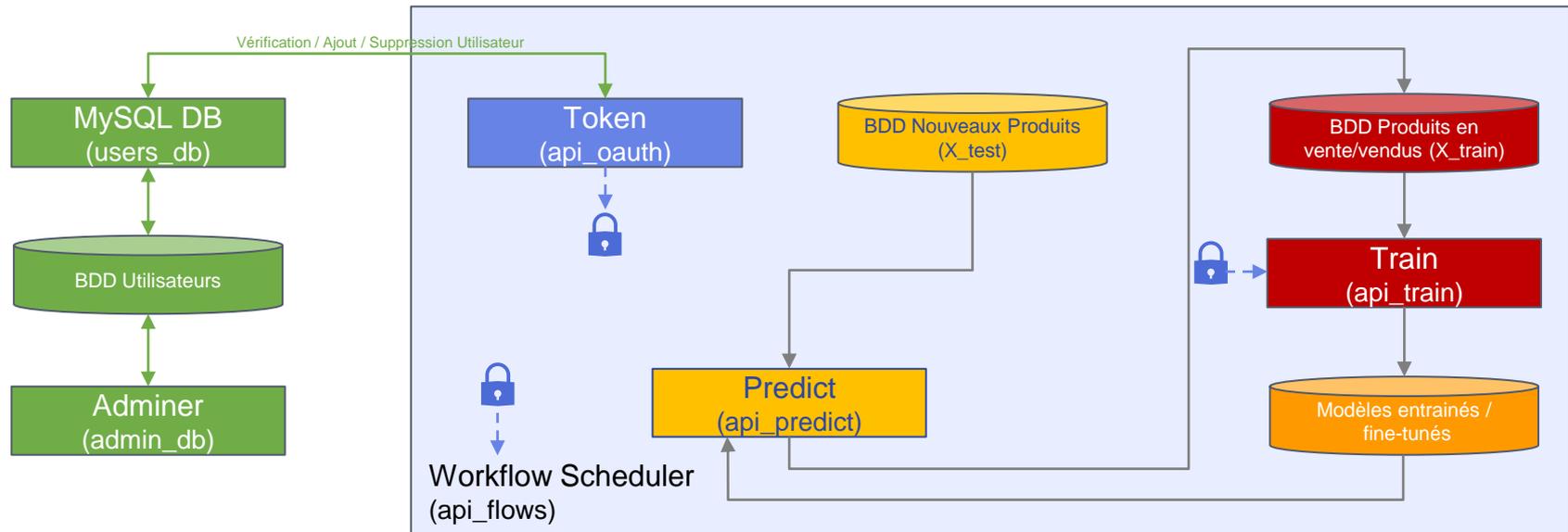
Processus « Rakuten Multimodal Product Data Classification »



API « Rakuten Multimodal Product Data Classification »



15 Docker Containers



Légende:

Container

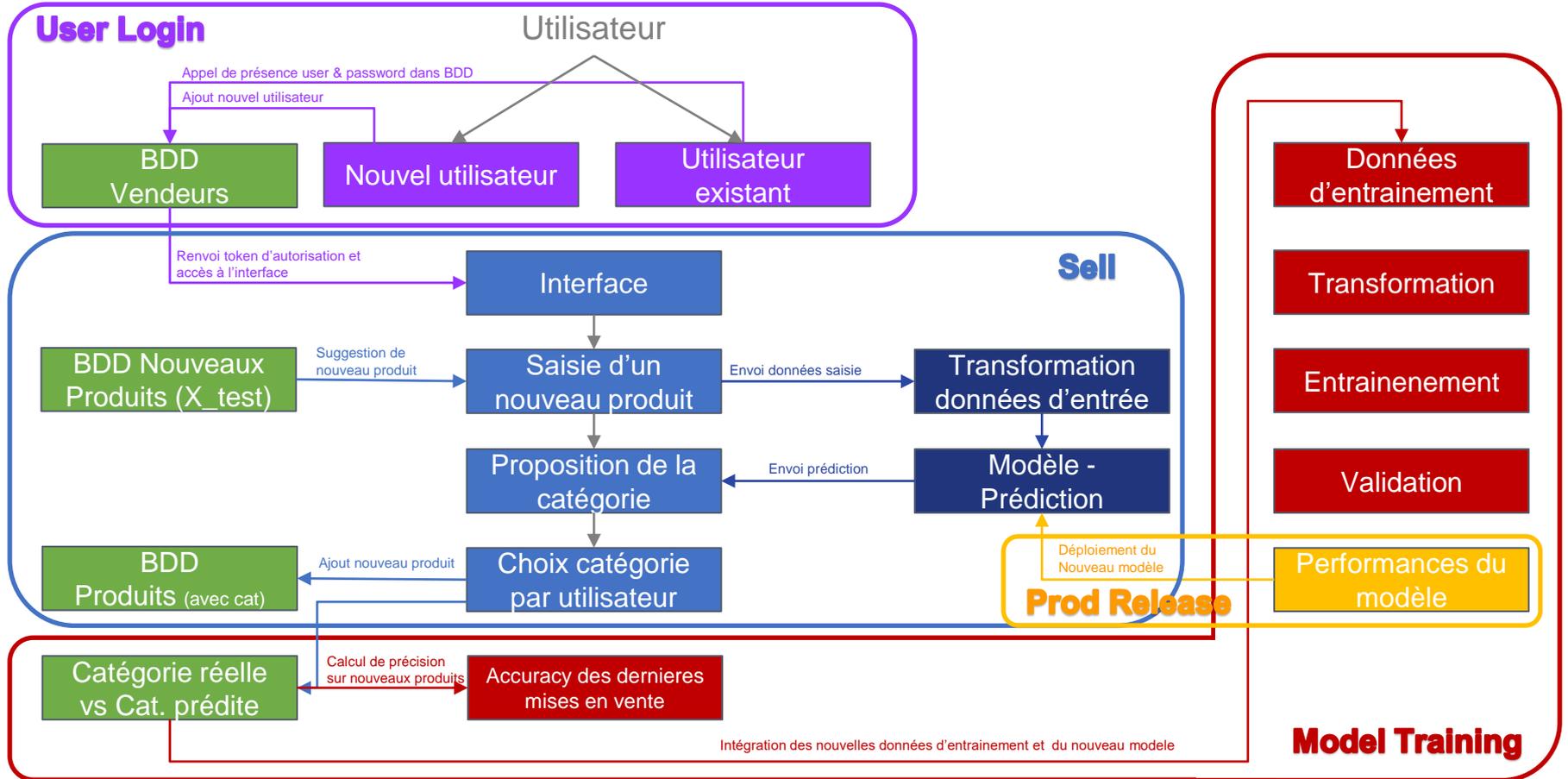
BDD

Streamlit
(streamlit)

Airflow
(7 containers)

Tensorboard
(tensorboard)

Streamlit Pages : « Rakuten Multimodal Product Data Classification »





5 – Démonstration

6 – Améliorations futures et Conclusion

a. Amélioration futures

Il serait souhaitable de :

- « porter » (notamment) les **modèles entraînés sur le Cloud**
- Disposer d'**environnement de développement, de tests et de préproduction**
- Remplacer le front « Streamlit » par un **front web plus robuste**
- « porter » **le front et les API « Predict » et « Train » sur Kubernetes**
- Analyser le cycle de vies des modèles avec **MLflow**
- Surveiller l'ensemble avec **Prometheus et Grafana**

b. Conclusion et Remerciement

- Ce projet nous a permis de mettre en œuvre de nombreux concepts étudiés
- Nous remercions très chaleureusement **Maëlys** pour son soutien et lui souhaitons beaucoup de réussite dans ses futures aventures

Annexe

<p>PREDICTION TASK </p> <p>Type of task? Entity on which predictions are made? Possible outcomes? Wait time before observation?</p> <p>Category prediction on Test description and product picture. The category should be predicted within 5 secondes.</p>	<p>DECISIONS </p> <p>How are predictions turned into proposed value for the end-user? Mention parameters of the process / application that does that.</p> <p>The prediction is supposed to guaranty that the product to sell is in the right category, so it will be easily found by buyers.</p> <p>Furthermore, when the company is modifying its category structure, the ML model will help to find the new category for each product.</p>	<p>VALUE PROPOSITION </p> <p>Who is the end-user? What are their objectives? How will they benefit from the ML system? Mention workflow/interfaces.</p> <p>The end user will be assisted in the process of selling its product. The ML will support him in finding the right place to record the product.</p>	<p>DATA COLLECTION </p> <p>Strategy for initial train set & continuous update. Mention collection rate, holdout on production entities, cost/constraints to observe outcomes.</p> <p>Initial train set comes from existing product database. It will be enriched with "anomalies" prediction</p>	<p>DATA SOURCES </p> <p>Where can we get (raw) information on entities and observed outcomes? Mention database tables, API methods, websites to scrape, etc.</p> <p>Data comes from web platform:</p> <ul style="list-style-type: none"> - Users, - Product with text and Image - Predictions - Anomalies
<p>IMPACT SIMULATION </p> <p>Can models be deployed? Which test data to assess performance? Cost/gain values for (in)correct decisions? <u>Fairness constraint</u>?</p> <p>The modes can be deployed on the Cloud. The performance is assessed thanks to the weighted F1 score, (train set is imbalanced). The ML is processing to 2 models based on Text description and Image. Furthermore, there are millions of products.</p> <p>So the model is costly to train on full data set. It is wishable to train it on small dataset often, and on full dataset rarely.</p>	<p>MAKING PREDICTIONS </p> <p>When do we make real-time / batch pred.? Time available for this + featurization + post-processing? Compute target?</p>		<p>BUILDING MODELS </p> <p>How many prod models are needed? When would we update? Time available for this (including featurization and analysis)?</p> <p>1 prod model is needed, updated at least once a week</p>	<p>FEATURES </p> <p>Input representations available at prediction time, extracted from raw data sources.</p> <ul style="list-style-type: none"> - Product Id, - Designation - Description - Image Id - Category Id
<p>MONITORING </p> <p>Metrics to quantify value creation and measure the ML system's impact in production (on end-users & business)?</p> <p>Weighted F1 score and accuracy</p>				