**Slide 1**

Le MLOPs, c’est quoi ? C’est tout ce qui va permettre d’aider et d’accompagner le fait de mettre en production des algorithmes d'intelligence artificielles, ce ne sont pas des outils ni des technologies ; c’est majoritairement une question de de culture et de bonne pratique.

Bonjour, je m’appelle Guillaume chervet je suis ML ingénieur chez axa France alors je ne sais pas exactement ce que ça veut dire mais en gros je définis mon rôle comme le fait d’accompagner les les data scientist, mettre en production les algorithmes d intelligence artificielle.

Aujourd’hui je travaille dans une équipe qui travaille essentiellement sur des projets de lecture automatique de document en temps réel. Généralement cela va mettre en œuvre principalement un ou plusieurs algorithmes de Deep learning.

Aujourd’hui 95% des projets échoue à l’étape mise en production. Ce chiffre commence à diminuer. Sur ce type de projet toutes les clés sont réunies pour que ça échoué.

**Slide 2 : trio intro**

La première problématique et une problématique humaine alors aujourd’hui la plupart des applications on a des devs et des Ops et ce n’est déjà pas facile de faire travailler tout le monde dans une équipe unie. Là on va rajouter encore des spécialités dans les métiers de la data. Des experts de la data.

Aujourd’hui, il est quasiment impossible d’être expert dans les 3 domaines. Ce type de projet est au top du top de la complexité dans tous les domaines d’expertise technique. Cela nécessite par conséquent d’accepter les contraintes des autres et de leur faire confiances.

**Slide 3 : coûte chère intro**

Ajouter au fait qu’il y a plus d’acteurs : on est sur des projets qui généralement coûtent extrêmement chers.

Alors là c’est un projet de classification d’email qui a très bien fonctionné chez nous qui est un peu un projet exemple. Aujourd’hui le coût de fonctionnement du projet par mois est environ de 9000€, ce qui fait par an un coup de projet à un peu plus de 100000€. Je pense que l’on peut encore diviser en les coûts par 2. Aujourd’hui sur ce projet on classifie environ 10000 mails par jour et ce qui fait un coût de classification d'un email a 3 centimes d’euros. Si par exemple, on réalise un réentraînement par an, par exemple on va avoir des coûts de projet qui font qu’aujourd’hui le cout de classification d’un mail sera environ de 4 centimes d’euros et là encore je ne parle pas de la mise en place du projet qui elle a coûté un peu plus d’1000000 d’euros.

**Slide 4: coûte chère intro**

Ici un exemple fonctionnel de classification d’un mail.

Pour information, toutes les images que vous voyez là où il y a un petit texte en dessous en style python ce sont des images générées avec des les algorithmes d'intelligence artificielle.

**Slide 5 : sommaire**

Comme vous pouvez le constater dans ces projets au top du top de la complexité technique, toutes les clés pour que ça échoue sont réuni.

Cette présentation est découpée en 4 parties.

Dans la première partie on va introduire le Deep learning et c’est problématique.

Dans la 2nde partie on va expliquer ce qu’on attend d’un un projet en production production chez nous chez AXA France.

Ensuite dans la 3ème partie on va s'amuser à faire une recette de ce qu’il faut faire pour faire échouer un projet. L’idée c'est que on a eu des expériences qui nous aide à nous améliorer aujourd’hui.

Dans la 4e partie, Nous avons tirer des leçons de ces expériences qui vont nous permettre d’être sur le chemin pour faire qu’un projet se déroule bien.

**Slide 7**

Une des sous-ensembles de l'intelligence artificielle est le machine learning et en un des sous-ensembles du machine learning est le deep learning.

**Slide 8**

Vous entendrez parler souvent de données structurées en général être représenté par un fichier Excel. En général pour ce type de de de de problématique où il y a un nombre d'entrées sorties limitée. On va utiliser des algorithmes de machine learning classique.

**Slide 9**

Quand on a des données de type son, image, vidéo, le nombre d’entrées sortie possible sont en général beaucoup plus élevé, du coup on va généralement utiliser des algorithmes de Deep learning qui vont vraiment utiliser un véritable réseau de neurones plus complexes.

**Slide 10**

Alors ça c’est un schéma qui est sorti du livre de François Chollet qui est le créateur de Keras que je vous conseille vraiment de de de lire.

Quand on fait du code classique en général on va avoir des règles des données d’entrées et on va écrire un algorithme qui va nous permettre de des réponses.

En machine learning c'est un peu différent ce qui va se passer c’est qu’il va nous falloir de la donnée d’entrée et aussi beaucoup de réponses. Pas qu’un peu, mais vraiment beaucoup, des milliers d'exemples je 10000 10000, ce qui va permettre d’écrire du code qui va pouvoir générer les règles une intelligence artificielle, on va parler en techniquement de modèle.

**Slide 11**

En production le code classique ou l’ia s’utilise de la même façon.

On reçoit une entrée, on exécute, puis on récupère le résultat.

**Slide 12**

Dans le passé on parlait de projet : modèle centrique, parce que on n’était pas sûr d’arriver à faire une ia adapté. Aujourd’hui la plupart des points côté ia : en tout cas pour nos besoins chez AXA France déjà cracké. On sait qu'on va arriver à avoir une ia pour nos cas d’usage. Par contre le cœur de la problématique, vraiment le truc qui fait qu’un un projet est faisable ou pas : c’est la donnée. Si on a de la bonne donnée de bonne qualité, c’est vraiment ça qui va faire la clé du succès d’un projet. La donnée, c’est votre trésor qui fera la différence.

**Slide13**

Alors maintenant on va expliquer comment se déroule un peu un projet on si on reprend notre projet de de classification des emails si on veut commencer le projet. La première chose à faire c'est récupérer de la donnée. Sur ce projet cela tombe bien on avait déjà des emails des annotations car en production on a des humains qui classifient les emails manuellement. On a déjà la donnée et les réponses c'est parfait ça pour constituer un set de données.

**Slide14 et 15 et 16**

Ensuite avec cette donnée on va pouvoir Itérer pour créer une IA et construire une API que l’on pourra mettre en production. Notre application en production pourra être consommer par des application cliente.

**Slide 17**

En production il est extrêmement important de mettre en place ce que l'on appelle une boucle de feedback. Une boucle vertueuse. En cas d’erreur les application cliente doivent permettre de corriger l’erreur manuellement. On va pouvoir au fil de l’eau constituer de nouveau set de données de meilleure qualité et ainsi améliorer nos Intelligence artificielle.

**Slide 18**

On va faire une petite démo de comment se déroule un projet en se basant un dataset chats et chiens bien connu de Kaggle.

**Slide 19**

Avant de de commencer la petite démo. Imaginons que l’on a 20 000 fichiers à disposition. On va utiliser 17000 fichiers pour l’entrainement du modèle. Puis on va garder 3000 fichiers de côté pour la validation, pour tester sur notre modèle sur des données qu’il n’a jamais vues. C’est donné de validation nous serviront aussi à tester les environnements de production.

**Demo Time**

**slide 20**

Si on donne à manger à nos models des données qui différents des données sur lesquels elles ont apprises. Les prédictions risque se dégrader.

Reprenons la demo avec nos chats et chiens. Maintenant imaginons que notre utilisateur commence à utiliser des photos de peluche de chien et chat, les prédictions ne fonctionne plus correctement. Autre exemple avec des dessins de chiens et chats.

**slide 21**

Le Graal pour nous ; ce serait de pouvoir relivrer en production sans quasiment aucune action manuelle

**slide 22**

Projet ça c’est un slide Google, un projet de Deep learning c’est un peu moins de 5% data science, par contre c’est 100% de code !

**Slide 23**

Maintenant on va voire qu’est-ce qu’on attend d’un projet en production chez AXA France

**Slide 24**

Le besoin le plus important de répondre ce sont les besoins\*\* utilisateurs et en général ce que veut l'utilisateur. Il veut une bonne qualité de prédiction, q/ue la production ne soit pas chère et que ça soit très rapide nous chez AXA on se donne des contrainte de temps de réponses qui doivent être inférieures à 10 secondes pour la lecture automatique des documents. La sécurité ça n’est pas négociable. On veut que les projets soit monitoré. L’idée c’est que si on a un drift de données cela risque d’entrainer un incident de production. Si on est sur un projet majeur qui bloque la boite, on ne va pas attendre l’incident et bloquer la boîte pendant des semaines. On arrive au dernier point, il faut que l’on puisse être en mesure de réentraîner et redéployer rapidement. Si on a un incident majeur qui bloque la boite, il faut pouvoir relivrer le plus vite possible et ne pas attendre des mois pour débloquer la boite.

**slide 25**

Alors là, c'est le schéma fonctionnel du workflow de comment on lit un permis de conduire. On lit aussi les anciens permis de conduire français, mais on ne le pas affiché ici. L'idée c'est que on va recevoir une image ou 1 PDF en entrée et on va d'abord réaliser une première étape on va découper ce PDF en une ou plusieurs pages. Sur chaque page on va exécuter le même algorithme de zoning. Après ça on va rentrer dans des algorithmes spécifiques au recto et au verso. Par exemple on va d'abord remettre droite l’image recto avec algorithme spécifique recto. Puis ensuite exécuter un autre algorithme pour le zoner les champs pour finalement exécuter un OCR pour extraire finement le texte des champs.

Si on hébergeait une API qui expose ce service en mode monolithe. On aurait plusieurs problèmes, Le premier c’est le temps de réponse c’est-à-dire que quand on va appeler cette API en monolithe ce qui va se passer c’est que chaque algorithme va consommer toutes les ressources CPU de la machine ce qui fait que le pipeline va s’exécuter en séquentiel. Par conséquent le temps de réponse va être supérieure à 30 secondes. Autre problématique c'est que pendant ces 30 secondes si notre appel API arrive à cet à API ; cet appel va devoir attendre la fin du premier traitement pour commencer à traiter le nouveau traitement. Alors vous me direz que face à cela on peut rajouter des machines ; mais autre problématique et si on héberge cela en mode monolithe, il nous faut des machines d’environ 32 giga de RAM qui coute chère et qui être lon\*gue à démarrer. Cela va très compliqué de de pouvoir scaler rapidement et s’adapter à la charge.

La solution c’est de mettre en place des microservices, on va héberger chaque algorithme sur des ressources différentes ce qui nous permettent de finement pouvoir configurer la RAM et le CPU nécessaire. On va pouvoir jouer les traitements en parallèle et scaler rapidement.

**slide 26**

Comment ça fonctionne techniquement ? On utilise ce qu'on appelle des fonctions. Chaque algorithme est hébergé sur une fonction. Cette fonction écoute une Queue et dès qu’elle a du temps libre, elle va pouvoir dépiler les messages de Queue et les traiter. par exemple prenons le SPLITTER qui extrait les page d’un PDF, la fonction va récupérer un message de la Queue, les fichiers ne sont pas stocké dans le Queue mais elles sont stockées dans un REDIS on va recevoir uniquement l’identifiant du fichier en entrée ; on va aller chercher l’image dans REDIS pour exécuter nos algorithmes et par exemple si on a un PDF avec 2 pages on va extraire les 2 pages puis Générer 2 identifiant puis pousser les images dans REDIS et finalement faire 2 appels dans la QUEUE en aval pour pouvoir appeler les services suivants.

**slide 27**

L’avantage de fonctionner ainsi est l’on peut scaler rapidement et supporter les piques de charge. Imaginer que tout d’un coup on va rereçois 100 permis, le nombre de fonction va augmenter pour pouvoir dépiler plus vite les messages. Il y a aussi un avantage financier car on ne consomme que ce que l’on a besoin.

**Slide 28**

Alors ça c'est le schéma d'architecture. Si vous le trouver compliqué, c’est normal car c’est compliqué. Je ne vais pas décrire tout le schéma en entier mais, en en résumé on utilise une architecture asynchrone. On va recevoir en haut à gauche une image en PDF le premier POD va le recevoir le fichier le pousser directement dans redis puis avec l’identifiant du fichier notifier la QUEUE du splitter. Ensuite le Splitter va récupérer le fichier le traiter, etc. etc.

**slide 29**

**Ces projets ne sont pas sans difficultés.**

Je vais vous présenter les 2 plus importantes que l'on rencontre

**slide 30**

Alors tout d'abord il y a un vrai triptyque de choix, un curseur à positionner entre la qualité de prédiction, le temps de réponse et le coût des infrastructures en production. C'est à dire que si on veut avoir de la meilleure qualité de prédiction on peut accepter de payer plus chère ou de perdre du temps de réponse. A l'inverse si on veut que on puisse répondre plus rapidement, il faut pouvoir accepter soit de payer plus chère ou de perdre de la qualité de prédiction. Et à l'inverse si on veut que le projet coûte moins cher en production, il faut soit accepté de de rallonger le temps de traitement des réponses ou alors de diminuer la qualité de prédiction.

**slide 31**

Alors est-ce que vous voyez les différences ? Les différences dans ces images, nous on ne peut pas les voir, par contre l’IA si ! J'ai eu une première expérience de ce problème sur notre tout premier projet sur lequel j'ai travaillé. En bon développeur, il y avait une librairie Pillow qui sert à redimensionner les images qui n’était plus à jour. Comme on doit maintenir toujours les librairies. Je j’ai mis à jour. Ce qui s'est passé ensuite c’est que les prédictions ne marchent quasiment plus. Alors j'ai regardé les images. Images identiques à mes yeux, impossible à comprendre. Et en fait, la particularité c'est que c’est vraiment un problème pour l’IA qui elle ne reconnait plus les images. La conséquence de ce problème est que si en production on n'utilise pas les mêmes versions de python, ni les mêmes versions de chaque librairie et ni même le même OS on n’est quasiment sûr que la production ne va pas fonctionner comme attendu.

Revenons à la démo. Si on revient avec mes chats et mes chiens l'algorithme IA utilisé ici reçoit en entrer des images redimensionnées en 200\*200 pixels via la librairie Pillow. Elle a aussi été entrainé avec en entrer des images redimensionnées avec Pillow en 200\*200 pixels. Sur le projet permis de conduire, on a remplacé les redimensionnements d’images par la librairie OpenCV ce qui nous a permis de gagner un peu moins de 2 seconds totaux sur le traitement complet du pipeline. Dans cette démo j’ai aussi fait une version qui ou le images sont en production redimensionné avec OpenCV. Par conséquent, ce qui se passe c’est que si je réalise des prédictions avec un redimensionnement via Pillow puis avec via OpenCV on n’obtient pas les mêmes résultats. J'ai fait un test avec 12500 images et sur 12500 images et on a environ 500 images qui ne prédisent pas la même chose. Cela ne fait que 3% de différences de prédiction, mais si vous prenez un pipeline complexe on va enchaîner des algorithmes avec différents IA et bien ce qui va se passer c’est que les 3% qui vont se propager tout le long de du pipeline, ce qui fait qu’à la fin le taux de prédiction devient très faible.

**Slide 31**

Alors, l’idée, c’est que maintenant pour cette partie on va faire un petit retour d’expérience pour expliquer nos expériences qui nous ont permis de d’apprendre. Pour le faire de façon rigolote on va ouvrir une pizzeria et on va s'organiser pour servir des pizzas à nos clients. On va mettre en place ce qu’il ne faut faire pour faire couler notre pizzeria.

**Slide 34**

Pour commencer on va organiser notre équipe. Dans un projet ce qui se passe c'est qu'en général il y a 3 phases principales. Dans la première phase d’exploration on va tester si le projet est faisable, Ensuite on vient une phase de de d’industrialisation puis une phase de Déploiement et de Run.

Alors là je vous présente Élisabeth qui est data scientist et que l’on va mettre toute seule à travailler sur la phase d’exploration.

**Slide 35**

Alors Élisabeth est une excellente cuisinière on va lui demander de de pétrir la pâte à pizza et de bien l'étaler les ingrédients et c'est ce qu'elle fait avec tout son cœur et elle le fait très bien.

**Slide 36**

Voilà maintenant Élisabeth a fait sa pâte à pizza elle l’a étalé.

**Slider 37**

Je vous présente Hicham, son rôle c’est de répartir les ingrédients sur la pâte à pizza.

**Slide 38**

Maintenant Élisabeth refourgue la pâte à pizza et les ingrédients à Hicham. Comme vous pouvez voir là à gauche, Élisabeth a oublié de de donner le jambon à Hicham.

**Slide 39**

Alors maintenant, Hicham, c’est à son tour de jouer et c’est à lui d’étaler les ingrédients sur la pâte à pizza, cependant il se rend compte que y a le jambon a été oublié et il pensait aussi qu’Elisabeth aurais coupé les ingrédients, et malheureusement, il n’a pas de couteau à disposition.

**Slide 40**

Alors ce qui se passe par conséquent, c’est que Hicham en bon développeur fait tout ce qu’il peut. Il prend les ingrédients et les étales comme il peut sur la pâte à pizza.

**Slide 40**

Voila

**Slide 41**

Je vous présente Khalid ; c’est notre Ops est responsable de cuire la pizza pour ensuite la livrer au clients et ça va être à son tour de jouer.

**Slide 42**

Hop, Hicham refourgue la pizza bien garnie avec son coeur a Khalid.

**Slide 43**

Khalid se retrouve avec la pizza bien garnie mais mince on a oublié de lui donner les instructions. Il ne sait pas comment la cuire.

**Slide 44**

Alors Khalid fait ce qu’il peut pour cuire la pizza mais mince c'est une pizza spéciale elle est brûlée

**Slide 45**

Pour résumer, ici on a organisé notre cuisine de façon à séparer les rôles et à ne pas les faire travailler ensemble en même temps en « équipes » Elisabeth, Hicham et Khalid. Si vous voulez faire échouer votre projet. Il faut savoir que le choix même des librairies dès le début du projet impacte directement sur la production donc ne pas faire le travail des acteurs ensemble, ne pas se soucier de la production dès le début du projet, c'est là une des clés pour faire échouer votre projet

**Slide 46**

J’allais oublier de parler de Lilian. Lilian est venu ce matin livrer du sel, c'est une des matières premières de la pizza mais il s'est trompé il a livré du sucre ! Elle va être bonne notre pizza.

**Slide 47**

Maintenant on va dresser la table pour nos clients. Alors si vous voulez faire échouer un projet cela peut être une très bonne idée de ne pas se focaliser sur le besoin.

Premier exemple sur le projet permis de conduire, on nous a demander de lire les permis en moins de 10 secondes. Quand on a commencé à travailler dessus et prendre le tout premier algorithme, le Splitter PDF pour le mettre en production. On s’est rendu vite compte que sur l’environnement de production que les temps d’extraction des document PDF prenaient environ 15 à 30 secondes ce qui étaient problématiques. Les datascientist s’en sont pas aperçue car ils travaillent sur des machines surpuissantes qui réalise l’opération très rapidement.

Deuxième exemple, quand on était un peu plus loin sur le projet. On s'est rendu compte que certaines lectures étaient très lentes et engorgeait les systèmes. En regardant un petit peu plus en détail les permis qui ne passait pas. On s'est rendu compte que c'est permis, en général, étaient des permis quasiment illisibles. Pour ce genre de cas, cela aurait été une bonne idée d'aller voir les utilisateurs et de leur demander s’il ne vaut pas mieux tout de suite remonter que le permis n'est pas assez propre et pas lisible plutôt que d’essayer de le lire « quoi qu'il en coûte » et qu’à la fin, après 40 secondes on ne remonte quasiment aucun champ lu.

**Slide 48**

Apéro

Alors là on va enfin servir l'apéro à nos premiers clients. Quand on commence un projet, surtout si on veut faire de l’agile et avoir un Time To Market rapide. Cela peut être une bonne idée de toujours tester toutes les approches de code classique. Plutôt que de tout de suite réaliser du Deep learning.

Mettre en place un algorithme de Deep Learning, c’est un peu une usine a gaz, c’est compliquer à mettre en place, cela coute très cher.

Tout d'abord, le code classique va plus vite à mettre en place. Cela permet d’itérer rapidement et en général le temps de traitements est plus rapide et demande moins de ressources CPU.

Le code classique ne me permettra cependant pas d’être aussi bon en prédiction que le Deep learning. Le Deep learning permettra de de résoudre des problématiques très complexe de façon bien meilleure.

De manière générale, si vous commencez avec du code classique et que vous n’arrivez pas à atteindre résultats désirez, ce n’est pas du temps perdu, parce que ce code classique va vous permettre de gagner du temps sur la partie annotation. Vous pouvez pré-annoter vos annotations et aussi vous aurez aussi un système de fallback dans le cas où le l’IA n’arrive pas bien à prédire.

**Slide 49**

Alors ça c’est un exemple qu’on a eu sur le projet permis de conduire. Pour redresser un recto ou un verso on utilisait à chaque fois 2 IA afin de remettre l’image droite. Le problème, c’est que ces algorithmes étaient très gourmands en ressources, donc coutaient très cher et prenaient trop de temps de traitement. On a pu les remplacer par des algorithmes plus classiques avec la même qualité de résultat.

**Slide 50**

On a eu la même problématique que sur le projet de classification d'emails. Ce qui se passait c’est qu’on avait un algorithme de Deep learning qui devait remettre droit les documents. Elle ne le faisait pas bien et ça prenait même plus de temps que la lecture du document en lui-même. Du coup on a on a insisté pour chercher des algorithmes et a réussi à mettre à l’image parfaitement droite, extrêmement rapidement en moins 0,3 secondes et cela avec très peu de ressources consommées.

**Slide 51**

Dernier exemple pour notre apéritif, on a essayé de remplacer les algorithmes écrits en Deep Learning sur mesure pour détecter les champs à lire par des algorithmes gratuits trouvés sur internet déjà tout fait. On s'est rendu compte que la qualité de prédiction était quasiment identique. Les seules différences, cet algorithme était moins bon sur les permis qui sont vraiment complètement illisibles. Si on avait exclu dès le début les permis illisibles, on aurait pu utiliser cet algorithme et ainsi gagner du Time To Market et gagner beaucoup d’argent.

**Slide 52**

Maintenant on a fini l’apéritif, on va passer à l'entrée. Cela peut-être une bonne idée de ne pas industrialiser son code d'entraînement. Si vous n’industrialiser pas votre code d’entrainement. Quand on doit partir sur la phase mise en production on est quasiment obligé de réentraîner les modèles parce qu’obligé de faire des modifications du code. Si vous n’avez pas industrialisé le code d’entraînement c’est-à-dire que vous êtes resté sur Jupyter Notebook sans versionner les données ni le code. On se rend compte que c’est quasiment le temps complet du projet de refaire l’entraînement des modèles.

Alors maintenant on va pouvoir passer au plat principal et servir notre pizza à nos clients

**Slide 53**

En fait non, on s’est trompé, ce n’est pas une pizza ; on a servi un plat de spaghetti. La qualité de code, les tests unitaires aujourd'hui le c'est extrêmement important pour la maintenance du projet. Ne le fait pas si vous ne voulez pas que votre projet réussisse. Par exemple sur le projet de classification d’emails, on avait une phase d’extraction des données de beaucoup de mails qui prenait 2 semaines. Il y avait 2 choix possible, attendre 2 semaines pour valider une modification de code, ou prendre 20 minutes pour réaliser un test unitaire. Les tests unitaires sont un gain de temps immédiat.

**Slide 54**

Vous voyez cet exemple de code. C’est un code classique que l’on peut généralement voire en python. Sans test unitaire, sans exemple d’entrées et sorties, je suis personnellement incapable de savoir ce que réalise cette fonction.

**Slide 55**

Une petite pincée de sel sur notre pizza.

Si vous voulez vraiment faire échouer votre projet ça peut être aussi une bonne idée de travailler sur des GIT séparés. Un unique git c’est simple et pragmatique, cela permet de travailler en mode équipe focaliser la production en utilisant les PullRequest. Cela permet de minimiser le nombre d’action manuel et ainsi simplifie les automatisations. Alors si vous ne voulez pas travailler en équipe et complexifier votre projet déjà complexe, séparer les GIT, c’est une bonne recette pour échouer.

Séparer les GIT c’est aussi super pour avoir des déphasages de code entre l’entrainement et la production et du coup que cela ne fonctionne pas.

**Slide 56**

Cerise sur le gâteau pour le dessert !

Ne pas monitorer sa production, ça c'est simple vous avez un projet majeur, vous ne l'avez pas monitoré. Le modèle commence à prédire n'importe quoi et vous n'êtes pas au courant. Vous allez vers l’incident majeur qui va bloquer toute votre boîte et potentiellement la couler.

**Slide 57**

Petit café

Cela peut être une bonne idée de se focaliser uniquement sur les statistiques. Les statistiques sont un indice bon indicateur. Cependant, d'expérience rien de tel que de regarder de visualiser la donnée. Avec la donnée, on voit exactement ce qui se passe dans un pipeline complexe et cela extrêmement rapidement. Quand il y a un souci il faut toujours revenir au cœur de du problème là donnée.

**Slide 58**

Passons à l'addition

Il ne faut pas attendre qu’on vienne vers nous pour commencer à faire attention si le projet coûte trop cher. La planète a des ressources limitées, autant tout de suite penser au fait de de consommer le moins possible de ressources et d'être le plus efficient possible.

SI votre projet coute trop chère, il ne risque pas de vivre longtemps.

**Slide 59**

Avec c’est expériences, nous avons appris et maintenant on peut essayer de se mettre sur le bon chemin et de mettre toutes les chances de succès de notre côté.

**Slide 60**

On va repartir de la phase d’explorations. Ce qui est important dès le début dans l’organisation du projet, c’est de faire travailler tous les acteurs ensemble avec un seul objectif : que cela fonctionne en production. Humainement, de bien comprendre que les problématiques des uns sont les problématiques des autres pour que cela fonctionne. C’est vraiment cela le cœur du succès d’un projet ou non. C’est pour cela que le titre de la présentation s’appelle le ML OPS est une aventure humaine, c'est parce que c'est vraiment une culture de partage associé au fait de se faire confiance.

Cette phase d'exploration va permettre d'estimer si le projet est faisable

**Slide 61**

Chez AXA France nous travaillons pour les problématiques IA directement avec les données de production. Pour sécuriser cela, on va demander à Khalid notre Ops de déployer un environnement de développement. Seulement quelques personnes auront accès sur cet environnement donc on travaille sur qui sur AzureML

**Slide 62**

Dès le début Élisabeth et Hicham vont travailler ensemble depuis GIT. Ce sera principalement Élisabeth sur la première phase mais accompagné par Hicham qui connais les problématiques de production.

**Slide 63**

Assez rapidement on va devoir commencer des phases d’annotation. C'est très important de travailler en collaboration avec l’équipe qui va à annoter les données. Pourquoi ? c’est vraiment la données le trésor de votre boite.

Ce que l’on va faire c’est qu’on va d’abord faire des petits lots de données. Qui vont nous servir à affiner les règles d’annotation. Il est extrêmement important que l’annotations soit réaliser de manière identique par tout le monde.

Imaginez que vous avez un chat qui est sur une table que ça queue passe sous la table coupée sur la photo et redeviens visible de l'autre côté. Il se peut qu’un annotateur va zoner le chat sans zoner la queue a l’autre bout de la table. Et qu’un autre qui va sélectionner le chat ainsi que les 2 morceaux de queue.

L’IA ne pourra pas bien apprendre, il est très important de travailler en équipe pour affiner les règles d’annotation en fonction de chaque cas spécifique que l’on rencontre.

Une fois les règles bien définies, le volume de données a annoté pourra être augmenter.

**Slide 64**

Une fois que on a les annotés les annotations on peut entraîner les modèles et là tout de suite dès le début on va mettre en place une une série d’intégration continue qui va permettre de de pouvoir rejouer l’entraîne en en automatique depuis la Dana et le guide afin de générer un modèle qui sera placé dans le modèle registre alors pourquoi c’est parce que ce n’est pas le modèle qui fait foi mais plutôt le code qui est dans git plus la donnée qui est dans le data Lake donc en fait on ne on ne le livrable ce n'est pas le modèle c'est le code et là donnée code qui permet de générer le modèle et la donner

On peut voir que là  »n »a déjà une pâte à pizza bien étalée tous nos ingrédients et un couteau prêt à l'emploi donc on a vraiment tout ce qu'il faut

slide 65

11 sortie de cette étape un on doit tous ensemble être capable d'estimer les métriques hein c'est à dire la qualité de prédiction qu'on va pouvoir avoir le coût de production le temps de réponse et on a déjà un code d'entraînement et les données

slide 66

on va pouvoir entrer dans la phase d'industrialisation ou le but va être de de développer et de porterporter en production nonous apèis

slide 68

On va mettre en place une une une série d’intégration continue qui va nous permettre de construire des dockers qui vont récupérer les modèles dans le modèle registre construire nos PI avec le code d’inférence et les pousser dans un docker registre

slide 69

à partir de là nous on travaille on déploie les les nos dockers sur open shift on va demander à notre Obs de nous construire les environnements de dev recette pré prod et production fait du gitops c’est-à-dire que on va avoir 4 branches dans git une de dev une des recettes une de problème de production ce qui va nous permettre n'importe qui va pouvoir travailler dans git pour pouvoir déployer les images sur l'environnement par exemple de développement. on déploie à partir de manipulations dit

slide 70

aujourd’hui on a des outils qui nous permettent de de de valider l’environnement une fois que la la développement est validé qu’on a validé que les résultats sont ISO en entraînement donc entraînement des modèles on peut déployer en recette

pour cela on fait une poule request sur la branche de recette

soixante-et-onze

ce qui se passe c'est qu'on on à ce moment-là on va rentrer dans une autre phase du projet on va demander à notre Ops de de créer un nouvel environnement azur et Mel toujours en production mais sur celui-là on n’aura pas d’accès en écriture on aura seulement accès en lecture seule

slide 72

On va mettre en place un pipeline de CD qui va permettre de réentraîner toute la chaîne en automatique afin d’être sûr que on on a aucune aucun déphasage sur chaque étape reconstruire des modèles et pousser les images dans le docker Registry des à p souligne

une fois que l'on a ça

**Slide 73**

on peut via de manipulation voilà une fois que on a ça on refait tout le pipeline on va développer déployer en développement valider l'environnement ensuite

**Slide 74**

il va déployer un recette valider la recette

**Slide 75**

à ce stade on a déjà une pizza quasiment précuites

**Slide 76**

alors nous n’avons pas les droits de livrer en préproduction seul l’obs là par sécurité et il va pouvoir livrer cette pizza précuite en préproduction

**Slide 77**

une fois que l'environnement de protection sera validé on partira l'obs déploiera en production

**Slide 78**

on rentre dans la dernière phase c’est-à-dire qu’on est dans une phase où la paye est déployée et maintenant il faut qu’elle vive

**Slide79**

pour le moment on a mis du du monitoring en utilisant Prometheus et grafana pour surtout pour mentorer le le data drift donc on monitore des des des informations fonctionnelles ça c’est un exemple pour permis mais aujourd’hui ça nous convient pas on peut pas avoir des des métriques Des metrica assez fines du coup on est en train de travailler sur le fait de mettre en place la boucle de feedback et on va plutôt se brancher sur cette boucle de feedback pour mettre en place un un alerting beaucoup plus fin on a des équipes différentes séparées en production aujourd’hui en on a déjà de la team branchée sur premier \*\*\*\* grafana mais on va mettre en place un autre système

**Slide 80**

aujourd’hui on a 2 façons de de partager du code je travaille dans le même repository soit aujourd’hui met un répertoire directement en production qui est ce ce code qui est qui partirait en production et aussi ce code qui est réutilisé pendant les entraînements pour les entraînements

**Slide 81**

autre technique qu'on est en train de tester pour partager du code et travailler ensemble ce le fait de faire des des des packages des packages locaux quand on réutilise en local

**Slide 82**

Melope n’est pas on n'est pas centré sur les outils mais sur le projet grâce à nos projets on a pu mettre en place des outils open source que c'est un plaisir de partager notamment dans le délit clean qui permet d’éteindre les environnements kubernetes bah la nuit comme on s’appelle automatiser l’extinction des environnements c’est open source et c’est compatible environnement kubernetise donc c’est utilisable par tout le monde éco tag c’est notre solution d’annotation pareil qui est open source et notamment on peut annoter des emails ou des des documents parce qu’on n’a pas trouvé sur les outils du marché EMC live c’est un autre outil de de test d’intégration ça nous permet de d’appeler nos PI pour valider que on obtient les mêmes résultats de prédictions en production que sur les environnements d’entraînement

16 de 83

donc j'espère que l'essentiel de la présentation je j’espère que le message a été compris que le le l Obs c’est avant tout une culture un partage aujourd’hui on a des projets qui sont au top du top de la complexité et que si on on met pas tout en place pour que les experts travaillent tous ensemble et comprennent que la problématique de l'un est la problématique de l'autre et Ben ça ne peut pas marcher et le succès la clé du succès vient dans ce partage et ce travail ensemble

c'est le 84

merci beaucoup