LDA Advisory

10 Projets pour un Portfolio Data Science Réussi

Natacha Njongwa Yepnga

LeCoinStat

Introduction

Dans ce document, je vais vous présenter 10 projets essentiels pour décrocher un emploi, avec des exemples concrets dans les secteurs de la banque, du marketing, de la santé, de l'industrie et de l'audiovisuel. Nous allons également voir comment intégrer la mise en production pour chaque projet, un aspect crucial dans les entreprises.

Vous voulez en savoir plus ? Rejoignez-moi sur Youtube, LinkedIn, Instagram et TikTok

Projet 1 : Analyse Exploratoire des Données (EDA)

Contexte

L'analyse exploratoire des données (EDA) est une étape clé dans tout projet de data science. Ce projet permet de montrer votre capacité à comprendre des données brutes, à les nettoyer, et à formuler des hypothèses avant d'appliquer des modèles plus complexes.

Objectifs

- Comprendre la structure des données et leur distribution.
- Identifier les variables importantes et les relations entre elles.
- Produire des visualisations claires pour présenter les résultats.

Problématiques

Banque : Comment analyser les transactions bancaires pour identifier des schémas de fraude ?

- Dataset : https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

Marketing : Quels sont les segments de clients les plus rentables en fonction de leurs interactions avec des campagnes ?

- Dataset : https://www.kaggle.com/datasets/rodsaldanha/arketing-campaign/data

Technologies Utilisées

- **Python**

P**andas :** Pour la manipulation et la transformation des données.

Matplotlib/Seaborn : Pour visualiser les tendances à travers des graphiques comme les histogrammes, scatter plots, et heatmaps**.**

NumPy : Pour les calculs mathématiques et statistiques.

Shiny/Streamlit/Dash : Pour déployer un tableau de bord interactif afin de présenter les résultats de l'EDA aux parties prenantes.

- **Statistique Univariée :** Analyse d'une seule variable pour en comprendre la distribution et les caractéristiques.
- **Statistique Multivariée :** Analyse des relations entre plusieurs variables afin d'explorer des corrélations ou des interactions.

Compétences Acquises

- Manipulation des données : Capacité à manipuler de larges volumes de données avec Pandas et NumPy.
- Nettoyage des données : Utilisation de méthodes d'identification des valeurs manquantes, gestion des doublons et transformation des variables catégorielles.
- Visualisation des données : Création de visualisations perspicaces avec Matplotlib et Seaborn pour communiquer efficacement les résultats.
- Statistique Univariée : Capacité à analyser les caractéristiques d'une variable individuelle.
- Statistique Multivariée : Capacité à analyser et interpréter les relations entre plusieurs variables.
- Déploiement d'un tableau de bord interactif : Création d'applications de data visualisation avec Streamlit, Dash, ou Shiny pour fournir des insights interactifs et accessibles aux non-techniciens.

Ces projets vous inspirent ? Suivez-moi sur mes différents réseaux pour ne pas rater les prochains.

Projet 2 : A/B Testing

Contexte

L'A/B testing est une méthode utilisée pour comparer deux versions d'un produit ou d'un service afin de déterminer laquelle est la plus performante. Ce projet est très populaire dans le marketing, les produits digitaux et l'e-commerce pour optimiser des campagnes ou des fonctionnalités.

Objectifs

- Mettre en place une expérimentation contrôlée pour comparer deux groupes
- Analyser les résultats avec des statistiques pour choisir la version gagnante

Problématiques

Marketing : Quelle version d'une landing page convertit le mieux les utilisateurs ?

- D**ataset :** https://www.kaggle.com/datasets/zhangluyuan/ab-testing

Technologies Utilisées

- **Python : Pandas, Scipy (tests statistiques)**
- **Matplotlib/Seaborn : Pour visualiser les résultats**

Compétences Acquises

- Tests statistiques
- Analyse statistique des résultats
- Visualisation des résultats pour la prise de décision

Projet 3 : Clustering et Réduction de Dimensions

Contexte

Le clustering permet de segmenter des données en groupes homogènes, utile pour la segmentation des clients.

Objectifs

- Appliquer des techniques de clustering
- Utiliser des techniques de réduction de dimension pour visualiser les clusters

Problématiques

Marketing : Comment segmenter les clients en fonction de leurs comportements d'achat ?
- Dataset **:**
https://www.kaggle.com/datasets/kaushiksuresh147/customer-segmentation

Technologies Utilisées

- **Python :** Scikit-learn (KMeans, PCA), t-SNE

Compétences Acquises

- Clustering et réduction de dimension : Application des techniques de clustering et de réduction de dimension avec des algorithmes comme KMeans, CAH, DBSCAN PCA, et t-SNE.Analyse statistique des résultats

- Visualisation des clusters : Utilisation de visualisations interactives pour mieux comprendre les résultats des clusters.

Projet 4 : Régression

Contexte

La régression permet de prédire des valeurs continues.

Objectifs

- Construire des modèles de régression.
- Comparer et optimiser les performances.

Problématiques

Marketing : Comment prédire les ventes futures ?
- Dataset **:** **https://www.kaggle.com/datasets/yasserh/advertising-sales-dataset/data**

Immobilier : Comment prédire le prix de vente d'une maison en fonction de ses caractéristiques ?
- Dataset **:** **https://www.kaggle.com/c/house-prices-advanced-regression-techniques**

Technologies Utilisées

- **Python :** Scikit-learn, Flask/FastAPI, Docker

Compétences Acquises

- **Modélisation de régression**

- Régression Linéaire : Simple et multiple, pour des relations linéaires entre les variables
- Ridge Regression : Régression linéaire avec régularisation pour éviter le surapprentissage

- Lasso Regression : Régression linéaire avec réduction des coefficients non pertinents
- ElasticNet : Combinaison de Ridge et Lasso pour optimiser les modèles avec de nombreuses variables

- Modèles basés sur les arbres de décision :

- Arbre de décision pour la régression : Simple, explique facilement les relations complexes.
- Random Forest : Combinaison d'arbres de décision pour réduire le surapprentissage et améliorer la précision
- AdaBoost : Méthode de boosting qui améliore les performances des modèles faibles
- XGBoost : Algorithme de boosting populaire et efficace pour la régression
- LightGBM : Version optimisée pour de très grands ensembles de données
- CatBoost : Spécialisé dans les données catégorielles et performant dans les situations complexes

- **Déploiement d'une API de régression : Mise en place d'une API pour fournir des prédictions en temps réel**
- **Docker : Containerisation des modèles pour faciliter leur mise en production sur le cloud (AWS, GCP, Azure)**

Projet 5 : Modélisation Prédictive avec un Classificateur

Contexte

Les entreprises utilisent les données pour prédire des événements futurs tels que le défaut de paiement.

Objectifs

- Créer un modèle prédictif.
- Comparer les performances de différents modèles de classification.

Problématiques

Banque : Quels clients risquent de faire défaut sur leurs prêts ?

Contexte : Dans le secteur bancaire, il est crucial d'identifier les clients à risque pour minimiser les pertes liées aux prêts non remboursés. Ce projet vise à créer un modèle de classification qui prédit la probabilité qu'un client fasse défaut

- **Dataset : https://www.kaggle.com/datasets/itsmesunil/bank-loan-modelling**

Banque : Quels clients présentent un risque de crédit élevé ?

Contexte : Évaluer le risque de crédit est fondamental pour les institutions financières afin de décider si un prêt doit être accordé. Ce projet permet de créer un modèle de classification qui prédit la probabilité qu'un client représente un risque de crédit élevé

- Dataset**: https://www.kaggle.com/datasets/laotse/credit-risk-dataset**

Santé : Quels patients risquent d'être réadmis à l'hôpital ?
Contexte : Réduire les réadmissions hospitalières est une priorité pour les systèmes de santé, car elles entraînent des coûts supplémentaires et peuvent indiquer une mauvaise prise en charge initiale. Ce projet utilise des algorithmes de classification pour prédire les risques de réadmission

- Dataset **:**
https://www.kaggle.com/datasets/redwankarimsony/heart-disease-data

Télécommunications : Quels clients risquent de se désabonner (churn) ?
Contexte : Dans le secteur des télécommunications, la rétention des clients est un enjeu majeur. Ce projet a pour objectif de prédire quels clients risquent de se désabonner de leurs services en fonction de leurs comportements et caractéristiques.

- Dataset **: https://www.kaggle.com/datasets/blastchar/telco-customer-churn**

Technologies Utilisées

- **Python :** Scikit-learn, Flask/FastAPI, Docker

Compétences Acquises

- **Modélisation prédictive**

- Régression Logistique : Pour la classification binaire ou multiclass
- Support Vector Machines (SVM) : Pour les séparations linéaires et non-linéaires
- K-Nearest Neighbors (KNN) : Algorithme simple basé sur la distance
- Naive Bayes : Modèle probabiliste, efficace pour la classification de texte

- **Modèles d'ensemble (Ensemble Learning)**

- Random Forest : Un ensemble d'arbres de décision pour améliorer la précision
- Gradient Boosting Machines (GBM) : Modèles successifs corrigeant les erreurs des précédents
- XGBoost : Variante rapide et efficace du GBM, couramment utilisée en production
- CatBoost et LightGBM : Modèles basés sur le boosting, optimisés pour la vitesse et la performance

- **Déploiement d'une API prédictive**

- **Docker :** Containerisation des modèles pour faciliter leur mise en production sur le cloud (AWS, GCP, Azure)

Projet 6 : Analyse des Séries Temporelles

Contexte

L'analyse des séries temporelles permet de modéliser des données dépendantes du temps.

Objectifs

- Prédire les tendances futures

Problématiques

Banque : Comment prédire le volume des transactions bancaires ?
- Dataset **:** **https://www.kaggle.com/datasets/mczielinski/bitcoin-historical-data**

Technologies Utilisées

- **Python :**

- Statsmodels : Implémentation des modèles ARIMA et SARIMA.
- Prophet (Facebook) : Pour la prévision des tendances saisonnières et des événements irréguliers.
- XGBoost : Modèle de boosting pour les séries temporelles, efficace pour capturer des relations complexes dans les données.
- LSTM (Long Short-Term Memory) : Réseau de neurones récurrent utilisé pour modéliser des séries temporelles à long terme et des dépendances complexes.
- Flask/FastAPI : Déploiement d'une API REST qui fournit des prévisions basées sur les données historiques.

- **Docker et Kubernetes : Containerisation et orchestration des modèles pour les déployer sur des plateformes cloud.**

Compétences Acquises

- **Modélisation des séries temporelles :** Utilisation de modèles ARIMA, SARIMA et Prophet pour capturer les tendances temporelles et saisonnières.
- **Déploiement d'une API de prévision :** Construction d'une API pour fournir des prévisions continues
- **Orchestration avec Kubernetes :** Gestion des déploiements à grande échelle avec Docker et Kubernetes

Projet 7 : Deep Learning – Classification Cat vs Dog

Contexte

Ce projet montre comment implémenter un modèle de deep learning pour classer des images de chats et de chiens

Objectifs

- Implémenter un modèle de deep learning.
- Optimiser les hyperparamètres.

Problématiques

Industrie : Comment classifier des images de chats et de chiens ?
- Dataset **:** **https://www.kaggle.com/c/dogs-vs-cats/data**

Technologies Utilisées

- **Python :** TensorFlow, Keras, OpenCV
- **Modèles :**

- CNN (Convolutional Neural Networks) : Pour la classification d'images
- VGG16, ResNet, InceptionV3 : Modèles préentraînés utilisés pour le transfert learning

- **Flask/FastAPI :** Déploiement du modèle sous forme d'API pour prédictions en temps réel.
- **Docker :** Containerisation pour déploiement en production
- **Kubernetes :** Orchestration des conteneurs pour une gestion à grande échelle

Transfert Learning et Fine-Tuning

- **Transfert Learning** : Utilisation de modèles préentraînés sur de grands ensembles de données (comme ImageNet) pour initialiser les poids du modèle, ce qui permet d'accélérer l'entraînement et d'améliorer la précision avec peu de données.
- **Fine-Tuning** : Réentraînement des couches supérieures du modèle préentraîné pour ajuster les paramètres aux nouvelles données spécifiques au projet (exemple : classifier des images de chats et de chiens).

Compétences Acquises

- **Implémentation de réseaux de neurones convolutionnels (CNN) :** Capacité à construire et entraîner des CNNs pour la classification d'images.
- **Transfert Learning et Fine-Tuning :** Application de techniques avancées pour améliorer les performances du modèle en utilisant des modèles préentraînés.
- **Optimisation des hyperparamètres :** Utilisation de Grid Search et Random Search pour ajuster les paramètres du modèle.
- **Déploiement d'un modèle de deep learning sous forme d'API :** Capacité à créer une API REST qui fournit des prédictions en temps réel.
- **Mise en production avec Docker et Kubernetes :** Containerisation des modèles pour une gestion et une mise en production évolutive

Projet 8 : IA Générative

Contexte

L'IA générative est une technologie révolutionnaire qui permet de créer des images, du texte, ou même de la musique à partir de données d'entraînement. Ce projet vise à implémenter un GAN (Generative Adversarial Network) pour générer des images réalistes, en particulier des vêtements à partir du dataset Fashion MNIST. Le projet couvre les aspects fondamentaux de l'entraînement de modèles génératifs et leur optimisation, ainsi que leur mise en production pour des applications en temps réel

Objectifs

- Implémenter un modèle Generative Adversarial Network (GAN) pour générer des images à partir de données existantes.
- Optimiser les hyperparamètres pour améliorer la qualité des images générées.
- Utiliser des techniques avancées comme le Deep Convolutional GAN (DCGAN) et le CycleGAN pour générer des images plus réalistes.
- Déployer le modèle génératif sous forme d'API pour des prévisions ou générer du contenu sur demande.

Problématiques

Mode : Comment générer automatiquement des images de vêtements ou de nouveaux designs pour inspirer les créateurs de mode ?

- Dataset **: https://www.kaggle.com/datasets/zalando-research/fashionmnist**

Art et Design : Comment générer des œuvres d'art à partir d'images existantes en utilisant des réseaux GAN ?

- Dataset **: https://www.kaggle.com/datasets/ikarus777/best-artworks-of-all-time**

Technologies Utilisées

- **Python : TensorFlow, Keras, OpenCV**
- **Modèles :**

- GAN (Generative Adversarial Networks) : Modèle de base utilisé pour générer des images réalistes.
- DCGAN (Deep Convolutional GAN) : Utilisé pour les tâches de génération d'images avec des convolutions profondes, idéal pour améliorer la qualité des images générées.
- CycleGAN : Utilisé pour des transformations d'image à image, par exemple pour générer des variations stylistiques d'images.

- **Flask/FastAPI :** Pour déployer une API REST qui permet de générer des images à la demande.
- **Docker et Kubernetes :** Pour déployer le modèle dans un environnement scalable

Transfert Learning et Fine-Tuning

- Transfert Learning : Utilisation de modèles préentraînés pour accélérer la génération d'images en partant de concepts appris sur d'autres ensembles de données d'images.
- Fine-Tuning : Réentraînement des couches du modèle pour adapter les générateurs à des styles spécifiques ou des types d'images particuliers (exemple : générer des vêtements ou des œuvres d'art).

Compétences Acquises

- **Création de modèles d'IA générative :** Conception et entraînement de réseaux GAN pour générer des images réalistes à partir de données d'entrée.
- **Utilisation de modèles avancés :** Implémentation de DCGAN et CycleGAN pour améliorer la qualité des images générées.
- **Optimisation des modèles :** Ajustement des hyperparamètres pour améliorer les résultats du générateur et du discriminateur.
- **Transfert Learning et Fine-Tuning :** Capacité à utiliser des modèles préentraînés et à les ajuster pour des tâches spécifiques.
- **Déploiement d'une API de génération d'images :** Création d'une API pour générer des images en temps réel, intégrée dans des applications web ou mobiles.
- **Mise en production avec Docker et Kubernetes :** Déploiement des modèles de génération dans des environnements cloud pour assurer leur scalabilité et leur disponibilité.

Ces projets vous inspirent ? Suivez-moi sur mes différents réseaux pour ne pas rater les prochains.

Projet 9 : Utilisation des API GPT et LLaMA pour la Génération de Contenu Automatisée

Contexte

Avec l'essor des modèles de langage génératif comme GPT-4, LLaMA et d'autres modèles open source disponibles via Hugging Face, il devient possible de générer du texte automatiquement avec une grande précision. Ces modèles peuvent être utilisés dans diverses applications telles que les chatbots, la génération de contenu automatisée (articles, scripts, résumés), ou encore la personnalisation de réponses client. Ce projet consistera à intégrer l'un de ces modèles via les API, tout en optimisant les résultats pour répondre à des besoins spécifiques

Objectifs

- Intégrer les API de GPT, LLaMA ou autres modèles Hugging Face pour la génération de texte dans une application web ou mobile.
- Personnaliser les prompts et optimiser les résultats selon les besoins métiers spécifiques (marketing, support client, génération d'idées créatives).
- Évaluer la qualité des réponses générées et ajuster les paramètres pour obtenir un contenu de meilleure qualité.
- Déployer l'application qui utilise l'API sur le cloud et permettre un usage en temps réel avec une interface utilisateur simple.

Exemple de Dataset

· Dataset de conversation client **:**
https://www.kaggle.com/datasets/thoughtvector/customer-support-on-twitter

Technologies Utilisées

- API de GPT : Utilisation de l'API de GPT-4 pour la génération de texte de haute qualité. L'API de GPT permet de générer des conversations, des articles ou des réponses spécifiques basées sur des prompts donnés.

- API de LLaMA (Hugging Face) : Intégration de modèles open-source comme LLaMA pour des tâches similaires, avec des possibilités d'affiner le modèle sur des données spécifiques (fine-tuning).

- Hugging Face Transformers : Pour accéder à une large bibliothèque de modèles, tels que T5, BERT, et GPT-J, disponibles sur Hugging Face.

- Flask/FastAPI : Pour déployer l'application générative sous forme d'API REST.

- Docker et Kubernetes : Pour déployer l'application à grande échelle dans des environnements cloud comme AWS, Google Cloud ou Azure.

- Streamlit ou Gradio : Pour créer une interface web interactive où l'utilisateur peut tester et interagir avec l'API en temps réel.

Compétences Acquises

- **Utilisation des API de modèles de langage génératifs :** Maîtrise de l'intégration des API de GPT, LLaMA, ou autres modèles sur Hugging Face pour générer du texte.
- **Optimisation des prompts et fine-tuning des modèles :** Ajustement des prompts et personnalisation des modèles pour répondre à des besoins métiers spécifiques.
- **Déploiement d'une application basée sur un modèle génératif :** Utilisation de Flask/FastAPI pour déployer une API capable de générer du texte en temps réel.
- **Mise en production et scalabilité :** Utilisation de Docker et Kubernetes pour la mise en production et la gestion d'une infrastructure scalable dans le cloud.

Étapes du projet

1. Exploration des API disponibles : S'approprier l'API de GPT-4 via OpenAI ou l'API Hugging Face pour LLaMA et autres modèles. Tester des exemples de génération de texte.
2. Conception d'un pipeline de prompts : Définir différents prompts et scénarios pour la génération automatique de contenu.
3. Fine-tuning et personnalisation : Si nécessaire, affiner le modèle sur un jeu de données spécifique pour améliorer la qualité des réponses.
4. Création d'une interface interactive : Utiliser Streamlit ou Gradio pour permettre aux utilisateurs de tester l'API en temps réel via une interface simple.
5. Déploiement en production : Déployer l'API et l'interface utilisateur à l'aide de Docker et Kubernetes pour permettre une utilisation à grande échelle

Projet 10 : Systèmes de Recommandation

Contexte

Les systèmes de recommandation sont au cœur de nombreuses plateformes modernes telles que Netflix, Amazon, ou Spotify. Ces systèmes permettent de personnaliser l'expérience utilisateur en proposant des produits ou des contenus en fonction de leurs préférences et comportements passés.

Le but de ce projet est de construire un système de recommandation basé sur des algorithmes de filtrage collaboratif et de filtrage de contenu, en exploitant des données utilisateurs pour générer des recommandations pertinentes et en temps réel.

Objectifs

- Construire un système de recommandation qui propose des films, séries ou produits en fonction des préférences des utilisateurs.
- Comparer différentes approches de recommandation, telles que le filtrage collaboratif et le filtrage de contenu.
- Mettre en production le système de recommandation pour des prédictions en temps réel via une API.

Problématiques

Audiovisuel : Comment recommander des films ou des séries à des utilisateurs en fonction de leurs goûts et de leurs historiques de visionnage ?

- Dataset **: https://www.kaggle.com/datasets/grouplens/movielens-20m-dataset**

E-commerce : Comment recommander des produits à des clients en fonction de leurs précédentes interactions avec la plateforme ?

- Dataset **: https://www.kaggle.com/datasets/snap/amazon-fine-food-reviews**

Approches de Systèmes de Recommandation

1. Filtrage Collaboratif (Collaborative Filtering) :
- Utilisation des interactions passées entre utilisateurs et items (films, produits) pour recommander des contenus similaires à ceux appréciés par d'autres utilisateurs avec des goûts similaires.
Exemple : Si plusieurs utilisateurs qui aiment un film X ont également aimé un film Y, alors on recommande Y aux utilisateurs qui n'ont pas encore vu ce film.
- Modèle : Matrix Factorization (SVD), K-Nearest Neighbors (KNN).

2. Filtrage Basé sur le Contenu (Content-Based Filtering) :

- Utilisation des attributs des items (genres de films, catégories de produits) et des utilisateurs (préférences explicites) pour recommander des éléments similaires à ceux déjà consommés.
Exemple : Si un utilisateur aime les films d'action, recommander d'autres films d'action basés sur les genres et descriptions.

3. Hybrid Systems :

- Combinaison des deux approches (collaborative et content-based) pour tirer parti à la fois des interactions utilisateurs-items et des attributs des items.

Technologies Utilisées

- **Python** :

- Scikit-learn : Implémentation des algorithmes de filtrage collaboratif (SVD) et filtrage de contenu.
- Surprise : Librairie dédiée aux systèmes de recommandation pour des implémentations avancées de filtrage collaboratif et de factorisation matricielle.

- **Flask/FastAPI :** Déploiement d'une API REST permettant de servir des recommandations en temps réel, en fonction des requêtes des utilisateurs.
- **Streamlit/Gradio :** Création d'une interface utilisateur pour tester le système de recommandation en temps réel.
- **Docker et Kubernetes :** Containerisation et orchestration pour le déploiement scalable de l'application dans le cloud (AWS, GCP, Azure).

Compétences Acquises

- **Implémentation de systèmes de recommandation :** Capacité à développer des systèmes de recommandation basés sur le filtrage collaboratif, le filtrage de contenu, ou des approches hybrides.
- **Optimisation des algorithmes :** Capacité à utiliser des techniques d'optimisation pour améliorer la précision et la performance des recommandations (Grid Search, Cross-validation).
- **Création d'une API pour la recommandation en temps réel :** Développement d'une API qui sert des recommandations personnalisées aux utilisateurs en fonction de leur historique et de leurs préférences.

- **Déploiement de systèmes à grande échelle :** Containérisation des systèmes avec Docker, et gestion de l'orchestration des conteneurs avec Kubernetes pour permettre un déploiement à grande échelle dans un environnement cloud.

- **Exploration des biais et explication des recommandations :** Développer des méthodes pour expliquer pourquoi un certain item est recommandé à un utilisateur (par exemple, via des systèmes d'explicabilité).

Étapes du projet

1. Exploration des Données : Analyse des données utilisateurs et items (films ou produits) pour mieux comprendre les comportements d'achat ou de visionnage
2. Construction des Modèles : Implémentation de plusieurs approches de recommandation, incluant le filtrage collaboratif (SVD) et le filtrage basé sur le contenu.
3. Évaluation des Modèles : Comparaison des performances des modèles à l'aide de métriques comme le Root Mean Squared Error (RMSE) et le Mean Absolute Error (MAE).
4. Déploiement du Système : Développement d'une API REST pour offrir des recommandations en temps réel aux utilisateurs
5. Mise en Production : Utilisation de Docker et Kubernetes pour déployer l'application dans le cloud et la rendre accessible à grande échelle