18 mai 2020 - 31 mai 2020

RAPPORT BRIEF PROJET

Indeed Salary Range

Clément Brunet
Thibaut Fiolka
Karima Haret
Simon Lamontagne
Damien Mayer
Emilie Viard

Ecole IA Microsoft
powered by Simplon

Introduction

Aujourd'hui, pas une seule entreprise n'innove ou ne se positionne sur un marché sans avoir minutieusement étudié les chiffres concernant son secteur.
Il est difficile — voire impossible — de prendre une décision avec des données floues. Donc prendre la bonne décision c'est faire la bonne prévision.

Dans ce contexte, Le CEO d'une entreprise qui s'occupe de faire des statistiques sur l'emploi dans le secteur du développement informatique et de la data en France nous a mandatés pour une mission : faire une étude de marché des différences de salaires entre les différents métiers du numérique et les villes.

Pour mener à bien notre projet, nous avons planifié une marche à suivre dans les termes suivants :

- Définir précisément le problème et cerner les outils à mettre en place.
- Choisir une perspective de travail.
- Choisir les critères de jugement.
- Décrire les stratégies retenues.
- Mesurer et estimer les modèles décisionnels.
- Analyser et interpréter les résultats.
- Présenter notre travail à notre client.

Les pages qui suivent sont une restitution de ces différentes étapes.

I. Réalisation du projet

1) Collecte et stockage de données :

En premier temps il nous fallait scraper le site Indeed.fr afin de récupérer les offres d'emplois dans une base de données MongoDB que nous pourrions exploiter. Pour cela il convenait tout d'abord d'explorer le site. Nous avons identifié dans l'url du site les informations que nous devions modifier afin de récupérer différents métiers et ville. Il fallait ensuite récupérer les offres d'emplois sur les différentes pages. La structure étant la même sur chaque page, nous avons tout d'abord récupéré dans notre base de données les informations formulées lors de la requête au site, à savoir la ville, le métier ainsi que le type de contrat. Par la suite les informations que nous voulions étaient le titre, l'id (unique pour chaque offre), l'url de l'offre qui nous donnait accès à d'autres informations, telles que l'entreprise qui recrute et la ville. Toutes ces informations pouvaient être récupérées sur la page de recherche Indeed. Pour avoir plus d'informations il nous fallait utiliser l'url récupéré auparavant pour des données telles que l'ancienneté de l'offre sur le site, la description de poste, les différents contrats renseignés, et le salaire s'il était affiché. Toutes ces informations étaient directement envoyées dans mongoDB qui les ajoutait une à une.

Afin de parer à toute erreur, nous avons tout d'abord vérifié l'existence d'un contenu de la page du site pour toute requête formulée avant d'envisager la collecte des informations qui s'y trouvaient. Ensuite, nous devions vérifier si l'offre trouvée était déjà dans la base de données afin d'éviter les doublons. C'est pourquoi nous avions récupéré l'id : nous comparions les id déjà présents dans la base de données de mongoDB à celle que nous voulions ajouter. Si celle-ci était déjà présente, nous passions à la suivante sans scraper les autres informations. Une fois cette vérification faite, nous avons collecté toutes les offres où l'id n'était pas existant dans la base de données, et nous avons rempli de vide si certaines informations comme le salaire n'étaient pas renseignées. Une fois la page entièrement parcourue, le script sélectionnait la page suivante et exécutait la même opération jusqu'à la dernière offre présente sur les pages suivantes (ou bien au bout de 100 pages).

La plus grande difficulté dans cette partie a été d'ajouter en temps réelle les informations dans la base mongoDB sans passer par un fichier 'csv'. Le recherche de solutions et de script a mobilisé 2 journées de travail. Ce temps aurait pu être investi par exemple dans la collecte d'informations complémentaires ou dans la préparation d'un script sélectionnant de nouvelles offres toutes les semaines afin de remplir notre base de données. La difficulté fut surmontée dans la compréhension du fonctionnement de mongoDB, sur lequel les données doivent être chargées sous la forme d'un dictionnaire avec une seule valeur. Il était nécessaire d'ajouter les

informations collectées ligne par ligne dans notre base de données (et non pas d'ajouter d'un seul trait toutes les offres de la page qui venait d'être scrapée).

Après plusieurs itérations sur le script, une base conséquente a pu être constituée pour envisager l'entraînement d'un modèle. Il convenait à présent de traiter ces données.

2) Traitement des données

Après avoir récupéré notre base de données, il a fallu la traiter et la nettoyer pour qu'elle devienne exploitable. La procédure peut être résumée comme suit :

- Corriger les noms des postes de sorte à avoir les métiers "Data Analyste", "Data Scientiste", "Développeur", "Business Intelligence" et tous les autres métiers regroupés dans le nom "Autre".
- Recherche des données manquantes : nous avons remarqué que 80% des salaires n'étaient pas renseignés.
- Créer les salaires minimum, maximum et moyen pour chaque offre, comme les salaires renseignés par des tranches.
- Calibrer tous les salaires : modifier les salaires renseignés sur une base mensuelle, hebdomadaire, journalière et horaire pour en faire des salaires annuels.
- Sauvegarder la ville de l'annonce sous la forme de code postal.
- Créer la variable "Catégorie" qui est les tranches de salaire (par tranche de 10000€), nous avons ainsi obtenu 16 classes.
- Créer la variable "Région" qui regroupe les villes selon la région.
- Diviser notre base en deux, la première contenant les offres avec des salaires renseignés — que nous avons ensuite utilisées pour faire du Machine Learning — et la deuxième partie (avec les salaires manquants) pour faire des prédictions une fois que l'un modèle est validé.

Note sur le déroulé opérationnel : le processus décrit ci-avant, a en réalité été mis en œuvre sur la base de la recherche spontanée. La catégorisation de la variable salaire et la création de la variable "Région" ont répondu à la volonté d'améliorer les scores obtenus sur les premiers tests des modèles de Machine-Learning. Chaque itération sur le traitement des variables fut suivie d'un essai sur les modèles, équivalents ainsi en termes "agile" à 3 micro-sprints.

3) Visualisation des données

a) Data processing

Une première visualisation des données a été effectuée lors du *data processing* : une première répartition des annonces par catégorie (“métier” puis “type de contrat”) donne une représentation de la structure des données comme suit :

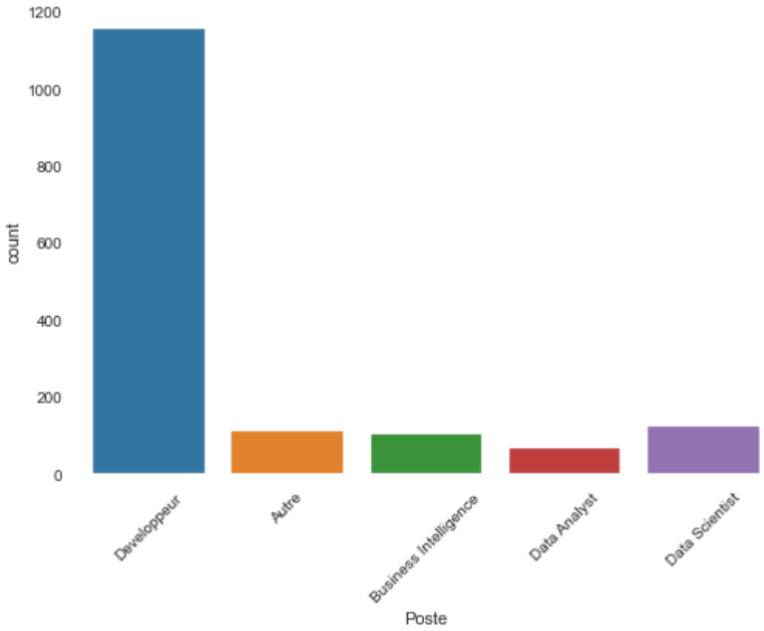


Figure [1] - Répartition des annonces par type de poste

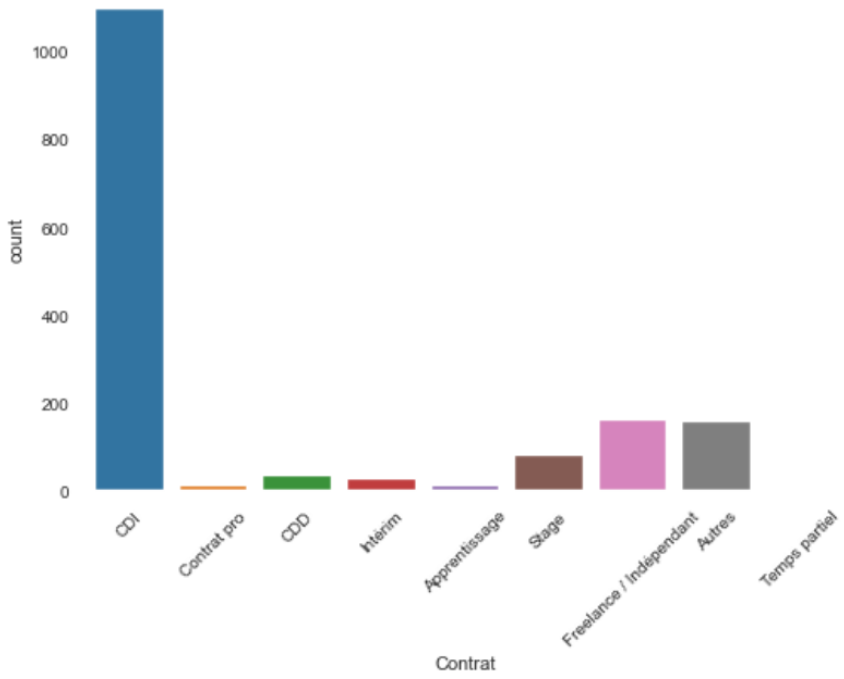


Figure [2] - Répartition des annonces par type de contrat

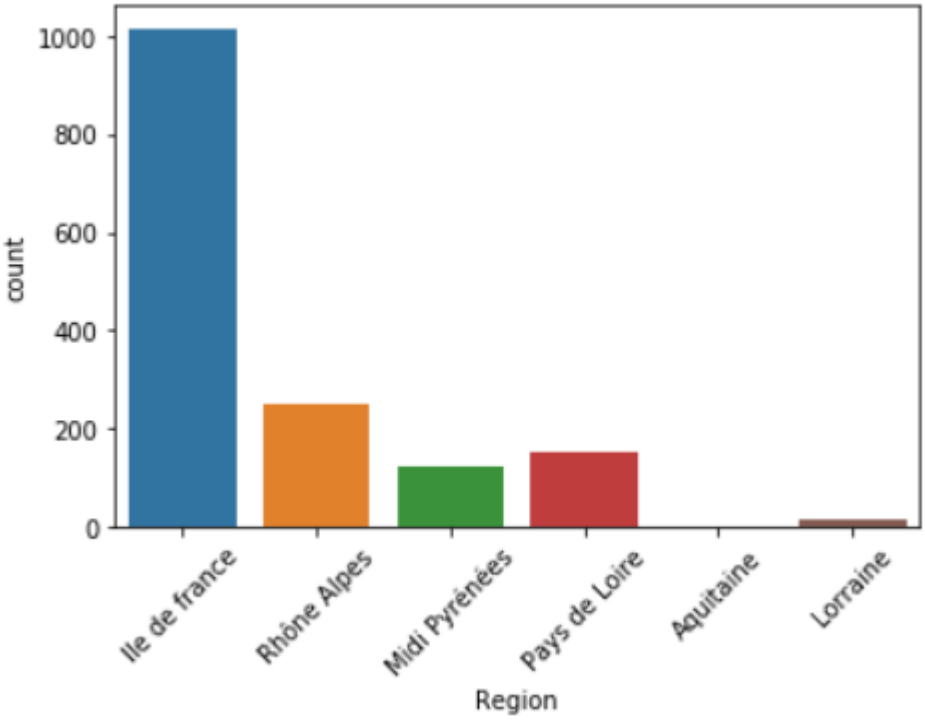


Figure [3] - Répartition des annonces par région

Les trois diagrammes ci-dessus montrent une nette rupture entre une catégorie “leader” et les autres. Parmi les différents profils recherchés dans le secteur de la data, les recruteurs montrent une nette préférence pour les développeurs. En termes de types de postes, ce sont les contrats à durée indéterminée qui dominent le volume des annonces. Enfin, la région parisienne s’affirme comme le pôle incontournable de l’IT en France.

Cet aperçu pourrait poser la question des données déséquilibrées (*imbalanced data*).
Nous avons gardé à l’esprit la possibilité de biais engendré par ce déséquilibre lors de l’application du Machine Learning.

Afin d'identifier les variables pertinentes pour prédire les salaires, nous avons établi une matrice de corrélation:

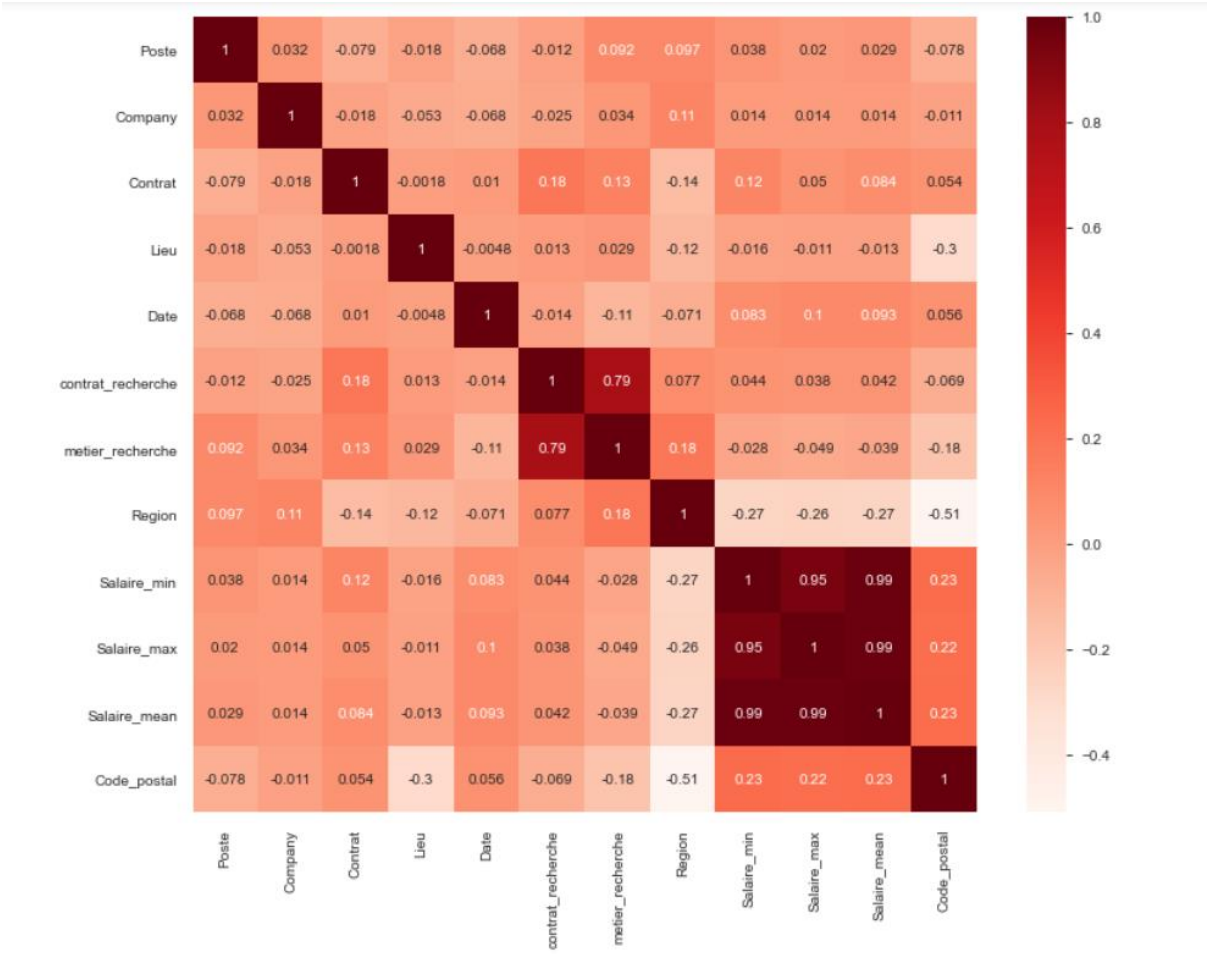
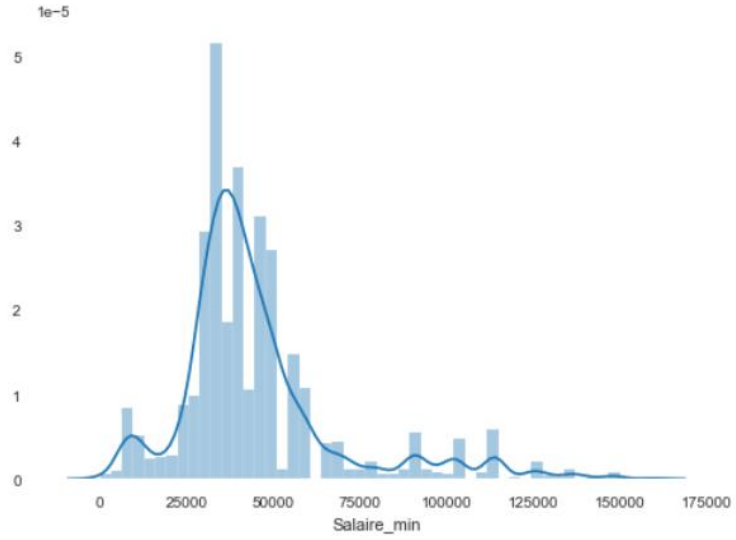
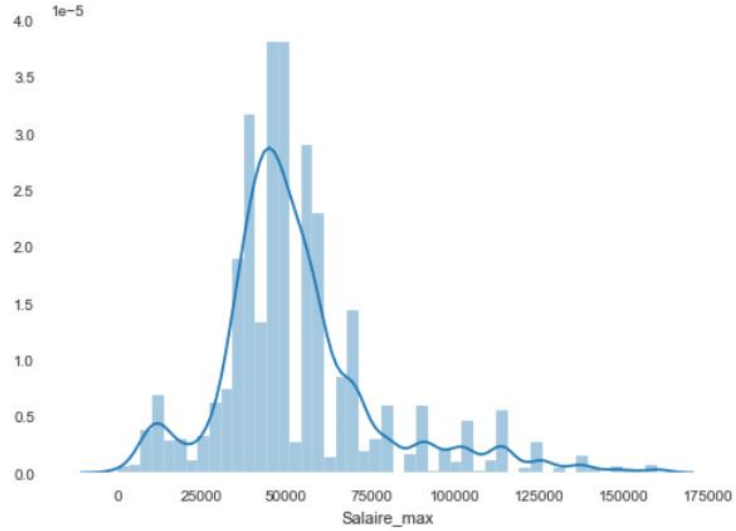


Figure [4] - Matrice de corrélation des variables collectées sur Indeed

Cette matrice montre globalement une faible corrélation entre les variables, exception faite des deux vocables utilisés lors de la requête pour rechercher le contrat et le métier (indice de 0.79).
La région semble cependant influer significativement sur le niveau de salaire. Cette variable sera donc particulièrement considérée dans le déploiement des algorithmes de machine learning.

Des histogrammes ont ensuite été réalisés sur les variables "salaire minimum", "salaire maximum", "salaire moyen" ainsi que "date de publication" et "code postal". Nous avons obtenu les répartitions suivantes :





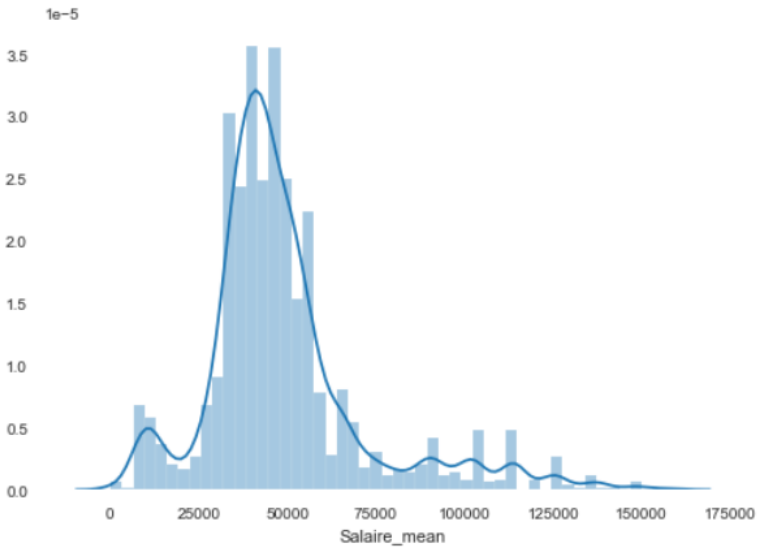


Figure [5] - Répartition des salaires minimum (en haut), des salaires maximum (au centre) et de la moyenne des deux (en bas) (en euro par année)

La répartition des salaires suit *grosso modo* une loi normale, contrairement à la répartition temporelle et géographique des annonces :

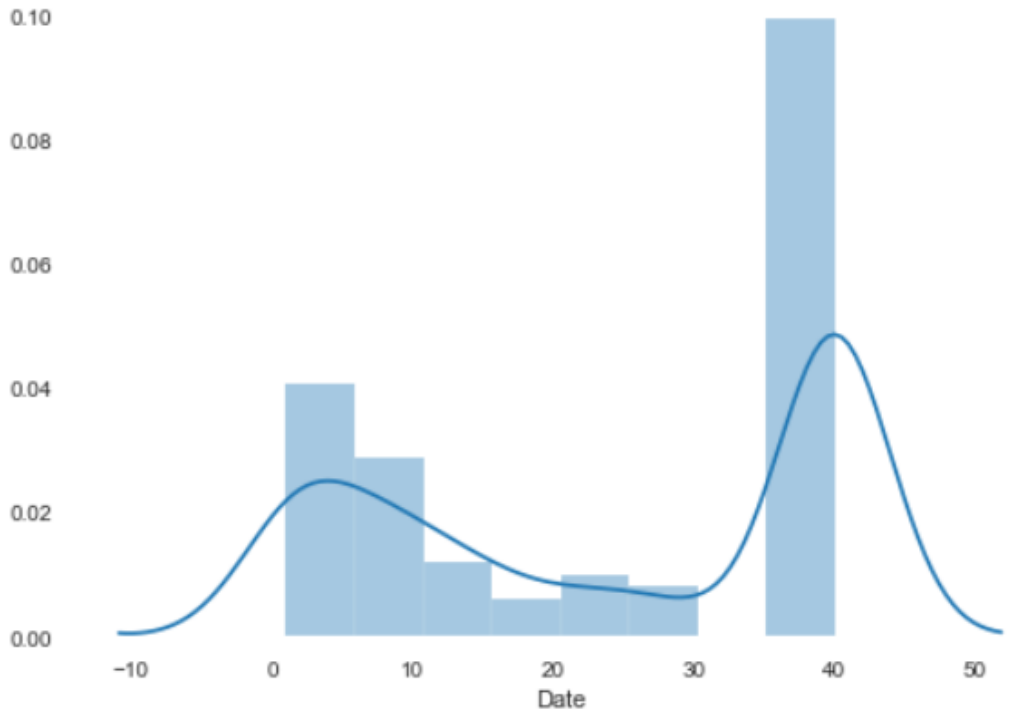


Figure [6] - Répartition de l'ancienneté des annonces Indeed (en jours)

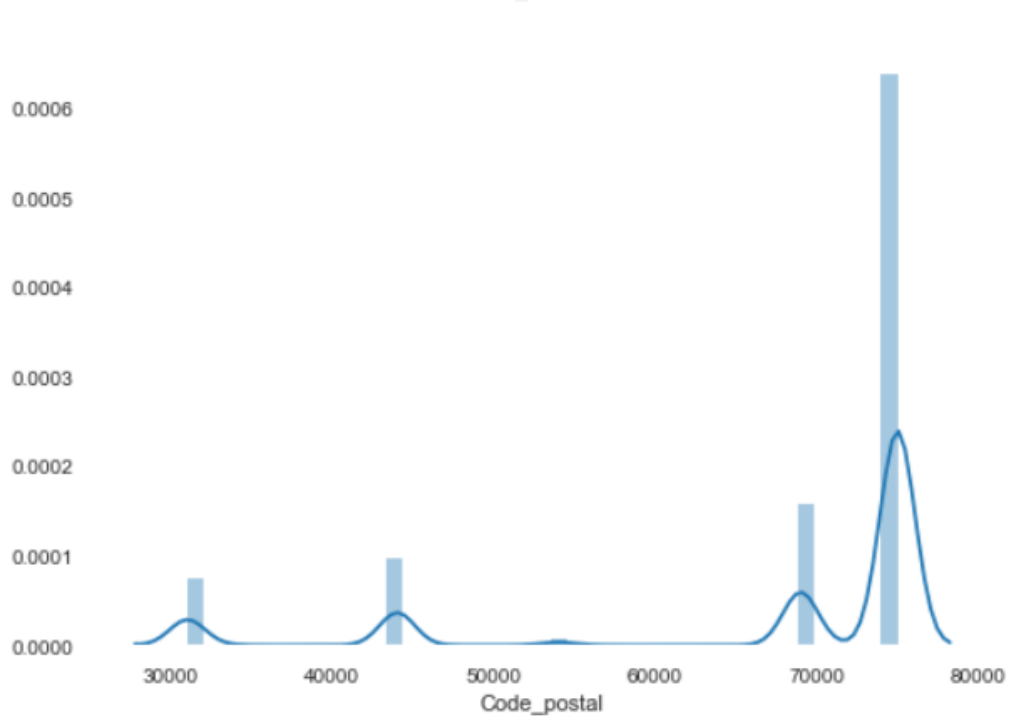


Figure [7] - Répartition des annonces par code postal

b) Présentation Client

Pour la partie “restitution au client”, nous avons décidé de faire un visuel Power BI qui nous permet de compléter les différents graphes et tableaux déjà présents dans nos différents scripts.

Power BI permet de rendre ces graphes interactifs. En d’autres termes, le contenu peut être modifié en direct, et cela en cochant les postes, régions et contrats. Le premier graphe “en jauge” en haut à gauche représente les offres où les salaires étaient renseignés sur Indeed. Nous avons donc le salaire minimum et le salaire maximum d’une fourchette salariale indiquée dans l’annonce, ainsi que la moyenne de ces deux bornes au centre. Pour celui en bas à droite, les mêmes principes sont appliqués aux salaires que nous avons prédits pour les offres d’emplois n’ayant aucun salaire d’affiché.

Les deux graphes en jauges montrent des écarts de valeurs, en grande partie expliqués par les performances non optimales de notre modèle. Le dernier graphe représente le nombre d’offres présentes pour les deux bases de données (avec salaire renseigné et avec salaire prédit). Nous pouvons par exemple observer le nombre d’offres en fonction des différentes régions ou encore comparer les salaires en fonction des postes.

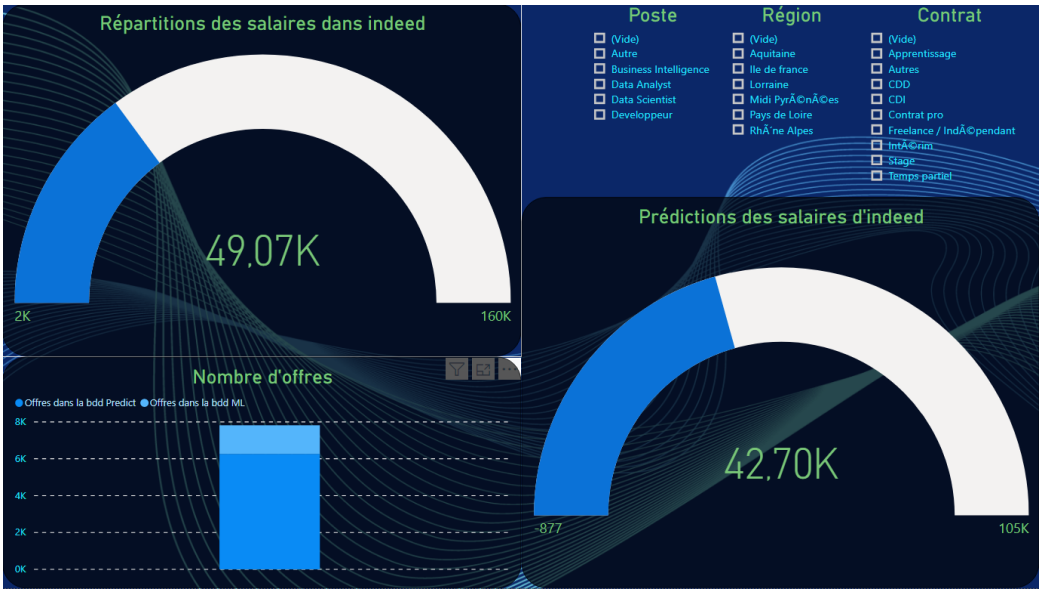


Figure [8] - Visuel du graphique interactif représentant les salaires en fonction des types de postes, des régions et des types de contrats

Un autre visuel a été réalisé avec l'outil flask, afin d'offrir un visuel cartographié des offres :

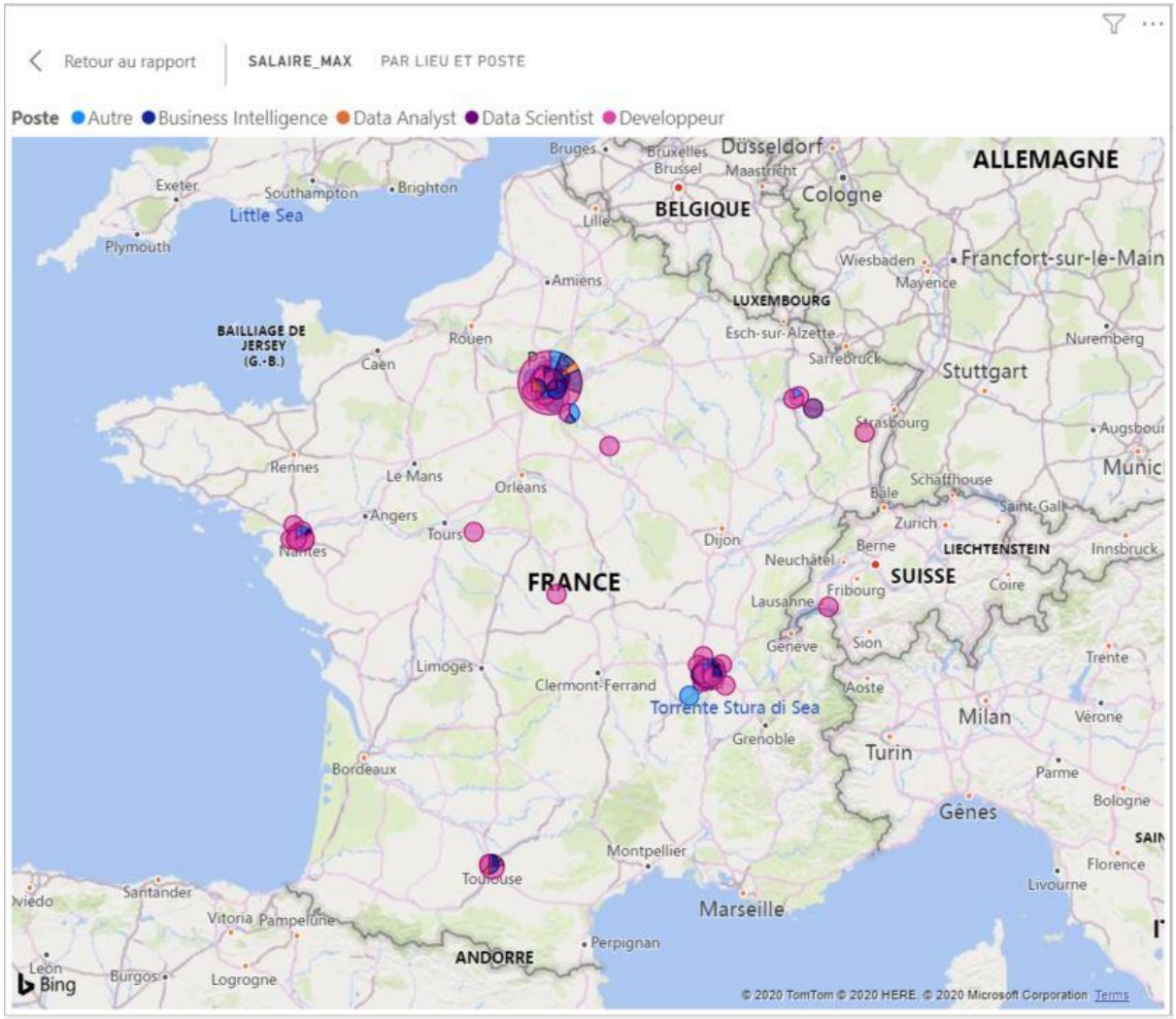


Figure [9] - Carte de France représentant les types de postes par région

La carte ci-dessus laisse clairement apparaître les disparités géographiques dans les offres d'emploi pour les professionnels de la data.

4) Machine Learning

Dans cette partie, nous avons décidé de tester plusieurs algorithmes de régression, comme nous sommes en train de prédire une variable continue, mais nous avons aussi testé la classification pour pouvoir conclure sur le modèle à retenir.

a) Rappels sur les métriques

MAE : MAE est simplement, comme son nom l'indique, la moyenne des erreurs absolues. L'erreur absolue est la valeur absolue de la différence entre la valeur prévue et la valeur réelle. MAE nous indique l'ampleur d'une erreur que nous pouvons attendre des prévisions en moyenne.

MAPE : Erreur absolue moyenne en pourcentage.

RMSE : Pour ajuster les grandes erreurs rares, nous calculons l'erreur quadratique moyenne (RMSE). En mettant les erreurs au carré avant de calculer leur moyenne, puis en prenant la racine carrée de la moyenne, nous arrivons à une mesure de la taille de l'erreur qui donne plus de poids aux erreurs importantes mais peu fréquentes que la moyenne.

Nous pouvons également comparer RMSE et MAE pour déterminer si la prévision contient des erreurs importantes mais peu fréquentes. Plus la différence entre RMSE et MAE est grande, plus la taille de l'erreur est incohérente.

R^2 : Plus le coefficient de détermination se rapproche de 0, plus le nuage de points se disperse autour de la droite de régression. Au contraire, plus le R^2 tend vers 1, plus le nuage de points se resserre autour de la droite de régression. Quand les points sont exactement alignés sur la droite de régression, alors $R^2 = 1$.

b) Régression Linéaire :

Dans la régression linéaire nous avons choisi “Poste”, “Entreprise”, “Contrat”, “Lieu” et “Date” comme variables explicatives, le salaire moyen comme variable à expliquer.

Pour pouvoir le mettre en pratique, il a fallu transformer les variables catégorielles en variables ordinales.

Poste:

- Autre : 0
- Business intelligence : 1
- Data analyste : 2
- Data scientiste : 3
- Développeur : 4

Contrat :

- Apprentissage : 0
- Autre : 1
- CDD : 2
- CDI : 3
- Contrat pro : 4
- Freelance : 5
- Intérim : 6
- Stage : 7
- Temps partiel : 8

	Poste	Entreprise	Contrat	Lieu	Date	Salaire_mean
0	4	48	2	62	8	47500.0
1	4	246	2	62	40	42500.0
2	4	362	2	62	15	52500.0
3	4	271	2	62	4	33600.0
4	0	294	2	62	8	24000.0

Les métriques à évaluer avec ce modèle sont : R^2, MSE, MAE. Plus le R^2 est proche de 1 et le MSE proche de zéro, plus notre modèle est performant. A l'inverse, un R^2 nul et un MSE conséquent dénotent un modèle peu performant.

Notre modèle de régression a obtenu les résultats comme suit :

```
Coefficients :  [2204.66532034  -12.3432607  1546.0754726   -30.62858681   72.76211913]
MAE :  16366.289886264445
MSE :  613480040.4564378
R² :  -0.0038631366346151275
Adjusted R² :  -0.03254494053846124
```

Conclusion :

D'après ces résultats, on conclut que la régression linéaire n'est pas adaptée à nos données car R^2 = - 0.003, le signe moins n'est pas une erreur mais pour indiquer que le modèle est mauvais.

d) RandomForestRegressor

```
Coefficients :  [[-0.0181879  -0.00114952 -0.02281022  0.00033349  0.0145003 ]]
MAE :  905.5672722454608
MSE :  2345294.3179286094
R² :  0.8223303482134363
Adjusted R² :  0.8200109010621757
```

e) GradientBoostingRegressor

C'est avec cet algorithme que nous avons obtenu le meilleur score, après utilisation du GridSearch:

```
MAE :  0.8449680599127726
MAPE :  inf %
MSE :  1.2160619903200307
RMSE :  1.102752007624575
R² :  0.7777482610706752
Adjusted R² :  0.7726137324485456
```

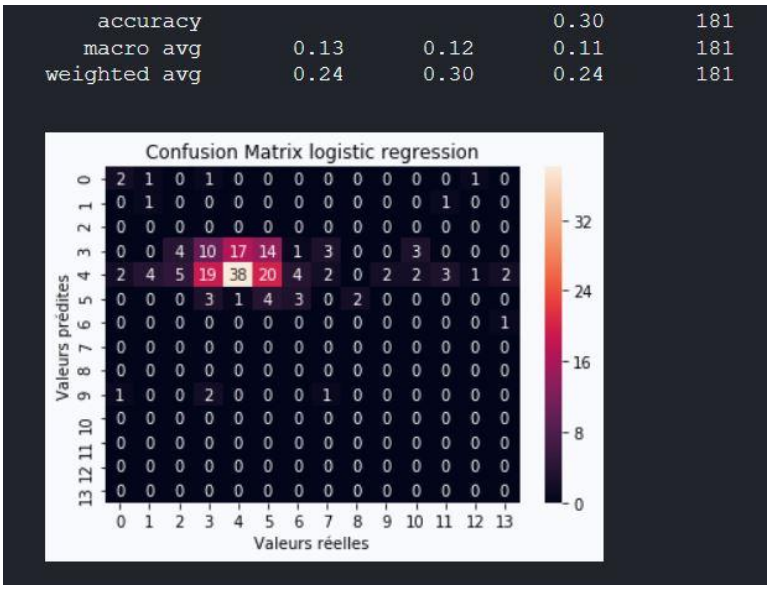
Ce score fut obtenu toutefois après nouveau traitement des données pour appliquer la variable catégorielle Salaire ainsi que l'ajout de la variable explicative "Région".

f) Régression Logistique :

Comme précédemment, on a choisi les mêmes variables, mais pour la variable à expliquer on a pris "catégorie".

	Poste	Entreprise	Contrat	Lieu	Date	Categorie
0	4	48	2	62	8	4
1	4	246	2	62	40	4
2	4	362	2	62	15	5
3	4	271	2	62	4	3
4	0	294	2	62	8	2

Les scores obtenus sont les suivants :

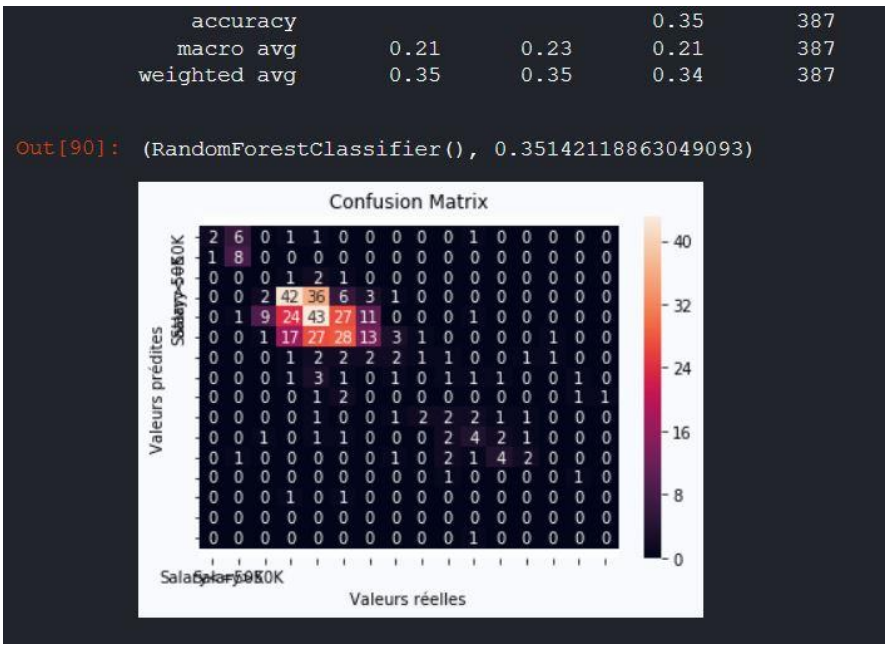


Conclusion :

L'accuracy étant égal à 0.3 implique que la régression logistique n'est pas le modèle performant pour répondre à nos besoins.

g) Random Forest Classifier :

Sur les mêmes variables que dans la régression logistique nous avons testé le *RandomForestClassifier, et les résultats sont les suivants :*



Conclusion :

L'accuracy est un peu meilleure que celle de la régression logistique mais ça reste toujours non performant pour notre objectif.

h) GradientBoostingClassifier

Le modèle a été testé sans toutefois apporter de meilleurs scores que les précédents.

5) Prédiction des salaires non renseignés

Le modèle ayant obtenu le meilleur score a été appliqué au dataset sans salaires.
Nous avons considéré que le temps partiel était 80% du temps pleins afin de faire les calculs des différents salaires.

Le résultat obtenu fut le suivant :

	Poste	Salaires_predit
3	Data Scientist	44098.630042
2	Data Analyst	43423.923077
1	Business Intelligence	43179.090323
4	Developpeur	42526.496734
0	Autre	41090.884384

Figure [10] - salaires prédits par types de postes (en euros)

	Contrat	Salaires_predit
5	Freelance / Indépendant	93705.440000
0	Apprentissage	45078.981132
1	Autres	44975.355208
3	CDI	40821.382757
2	CDD	37127.375000
7	Stage	35305.246281
6	Intérim	34960.089286
4	Contrat pro	24076.168467
8	Temps partiel	6024.857143

Figure [11] - salaires prédits par types de contrats (en euros)

Nous avons constaté les particularités suivantes :

- Il n'y a globalement pas de grandes différences entre les différents postes, toutefois le profil de développeur a un revenu globalement moindre que les autres profils dans notre base de données.
- Le contrat de Freelance / Indépendant a un salaire bien plus élevé que les autres.
- Le faible salaire du temps partiel est expliqué par le fait que nous l'avons considéré comme étant 80% du temps plein. Cependant son salaire reste bien moins élevé que les autres contrats.
- Le machine learning nous a permis d'avoir des résultats convenables, cependant le deep learning nous permettrait probablement d'avoir de meilleures prédictions.
- Avec un plus grand jeu de données nous pourrions probablement avoir de meilleures prédictions, c'est pourquoi avec une actualisation hebdomadaire des données, nous serions en mesure de fournir une meilleure précision tout en restant sur des bases connues (telles que le machine learning).

II. Gestion de Projet

1) Répartition des tâches

La première étape de la collecte de données s'est faite en individuel, suivi d'une mise en commun pour partager les difficultés rencontrées. Les tentatives d'optimisation du script pour le scraping ont débouché sur une augmentation du volume de données ainsi récoltées. Les membres du groupe ont comparé leurs résultats respectifs et la base de données la plus développée a été conservée pour la suite de la mission.

Sur une base de données identique, les apprenants ont fait des choix quant à l'exploration des modèles testés. Un membre s'est concentré sur la régression logistique (salaire inférieur ou supérieur à la moyenne), un autre membre a fait plusieurs itérations sur un même modèle optimisé à mesure des améliorations opérées sur le dataset lui-même.

La data visualisation à destination des clients (power BI et dashboard) a été confiée à deux apprenants, la préparation de la préparation à un autre apprenant et la supervision du présent rapport à un quatrième membre.

En rétrospective, la répartition des tâches s'est réalisée d'abord spontanément puis de manière concertée afin que chacun apporte sa contribution.

2) Lean Management

Le lean est une méthode de management qui vise l'amélioration des performances de l'entreprise par le développement de tous les employés. La méthode permet de rechercher les conditions idéales de fonctionnement en faisant travailler ensemble personnel, équipements et sites de manière à ajouter de la valeur avec le moins de gaspillage possible.[1]

Dans la réalisation du présent projet, chaque membre du groupe a contribué selon ses capacités et a apporté une plus-value au résultat final. Les tâches réalisées d'abord à l'identique, puis de manière complémentaire a permis de bénéficier d'une synergie de groupe, que l'on peut apparenter à du lean management.

[1] https://www.operaepartners.fr/la-definition-du-lean-management/

3) Méthode Agile

Afin de tenter une mise en situation de la méthode agile, les membres du groupe ont appliqué certaines particularités “agiles”. Thibaut fut désigné scrum master.

Dans le déroulé opérationnel, plusieurs micro-sprint ont été exécutés dans la mesure où des modifications sur la database ont permis d’améliorer les scores des modèles (fourchettes, nouvelle colonne “région”, etc.). En ce qui concerne la régularité des échanges, le groupe s’est réuni régulièrement sur Discord dans le salon vocal, de manière informelle et en moyenne deux fois par jour. Des échanges ont eu lieu avec les formateurs considérés comme le *product-owner*.

Dans l’ensemble, la méthode fut mise en œuvre en mode découverte, dans un respect de la dynamique spontanée de groupe (le groupe se contactait en cas de besoin).

Conclusion

Ce projet synthétisant plusieurs savoir-faire de la formation a permis, non seulement la comparaison de plusieurs modèles de machine learning mais aussi de mettre en œuvre les techniques de collecte de la donnée.

Des difficultés particulières ont été rencontrées dans le traitement de la variable à expliquer (le salaire). Pour les cas où cette donnée était disponible, elle n'apparaissait pas de manière uniforme : tantôt sur base annuelle, tantôt sur base mensuelle, sinon journalière ou horaire… Le plus grand défi concerne toutefois le traitement des fourchettes de salaires, d'abord considéré comme deux variables distinctes (bornes inférieur et supérieure), puis comme une catégorie.

D'autres difficultés sont apparues en lien avec la gestion de l'équipe. La coordination en distanciel d'une équipe de 6 personnes s'est révélée ardue et la communication a pu d'en trouver accidentée. La souplesse pour intégrer la spontanéité dans la dynamique a pu pallier cet aspect.

En dépit des points mentionnés ci-dessus, certains acquis sont à noter. En premier lieu le scraping implémenté sous mongoDB a élargi nos champs de compétences. Ensuite la considération de variables numériques en fourchettes considérées comme une variable catégorielle a ajouté à notre horizon des types de variables. La découverte s'est ainsi ajoutée à la connaissance.

Enfin, les perspectives d'amélioration sont toutes trouvées dans l'exploration du deep learning et l'implémentation prochaine des méthodes agiles, afin notamment de faire l'expérience de restitutions régulières avec le product owner. Et pour faire des sprints, même en activité physique adaptée, rien ne sert de courir...