DataScientest • com

Rapport Technique d'évaluation

PyGraal

Data Jobs

Promotion :
Bootcamp Data Analyst Novembre 2021

Participants :

Baptiste GARNIER
Eric GAUTIER
Benoît MOREL

Mentor :

Laurène BOUSKILA

Table des matières

Objectif

L'objectif de ce projet est de comprendre à l'aide des données les différents profils techniques qui se sont créés dans l'industrie de la Data. Plus exactement, il faudra mener une analyse poussée des tâches effectuées ainsi que des outils utilisés par chaque poste afin d'établir des ensembles de compétences et outils correspondant à chaque poste du monde de la Data.

Ensuite, il sera possible de construire un système de recommandation de poste permettant aux autres apprenants de viser le poste correspondant le plus à leurs appétences.

C'est un sujet d'autant plus intéressant que nous le retrouvons régulièrement dans la presse spécialisée.

Data

2020 Kaggle Machine Learning & Data Science Survey
The most comprehensive dataset available on the state of ML and data science

Le jeu de données contient les réponses de plus de 20 000 participants au sondage lancé par le site Kaggle.com afin de présenter une vue la plus complète possible de l'état de la science des données et du machine learning dans l'industrie. Le lien du dataset est le suivant:

Main data
https://www.kaggle.com/c/kaggle-survey-2020/overview

Supplementary data
Annexe 1 | kaggle_survey_2020_answer_choices.pdf
Annexe 2 | kaggle_survey_2020_methodology.pdf

ANALYSE DE FORME:

variable cible	= poste actuel / métier exercé = Q5
lignes et colonnes	= 20037, 355
types de variables	= toutes qualitatives
analyse des valeurs manquantes	= beaucoup de valeurs manquantes (i.e. NaN), certaines colonnes avec 99.5% de Nan

I. Audit des données et DataViz'

1. Analyse des techniques et outils utilisés par les professionnels de la Data

Cette analyse repose sur un sondage contenant 20 036 réponses obtenues auprès de participants provenant de 171 pays différents.

Les participants ont dû répondre à une série d'une quarantaine de questions. Quelques unes sont d'ordre personnel (âge, genre...) ou bien en rapport avec leur poste et entreprise actuels. Cependant, l'essentiel concerne avant tout les outils utilisés dans le domaine de la Data Science (langage de programmation, bibliothèque de data visualisation...) ainsi que le rapport à l'utilisation de Machine Learning.

Ce sondage comprend à la fois des questions à choix unique et à choix multiples.

A noter également que certaines questions étaient accessibles en fonction de réponses précédentes.

A partir de ce sondage, nous souhaitons établir des ensembles de compétences et outils correspondant aux divers postes monde de la Data, afin de construire dans un second temps un système de recommandation de poste permettant au demandeur de viser exactement le poste correspondant le plus à ses appétences.

2. Analyse exploratoire

La visualisation du DataFrame nous permet de constater que pour les questions à choix multiples, chaque choix est une colonne avec deux valeurs possibles (Nan ou Valeur).

Ainsi nous pourrons traiter par la suite distinctement les questions à choix unique et les questions à choix multiples.

De plus, nous constatons que les différents postes en Data Science se trouvent dans la colonne Q5.

Q5 correspond donc à notre variable cible pour la suite de l'étude.

Il n'y a que des valeurs catégorielles, un encodage sera nécessaire pour la création d'un modèle de Machine Learning.

2.1. Analyse et Nettoyage des Doublons et Valeurs Manquantes

 # Visualisation / Suppression des doublons = 14 doublons
 # La première colonne (temps de réponse) n'est pas pertinente pour l'étude.
 # Recherche des NaNs dans la variable cible
 # Suppression des NaNs sur cette valeur = 745 NaNs

La première ligne du DataFrame n'est pas une entrée mais correspond à l'intitulé des différentes questions. Cette entrée n'est donc pas à exploiter dans le DataFrame, mais elle reste intéressante pour la compréhension du jeu de données.

 # Création d'une base de données : « Questions » à partir de la 1ère ligne de df
 # Suppression de la première ligne
 # Analyse des NaNs par % par colonne

Cela nous permet de constater des taux conséquents de NaNs.

Toutefois, pour rappel, certaines questions sont à choix multiples, par conséquent le phénomène s'explique en partie et pour ce type précis de questions les NaNs seront à interpréter mais surtout pas à supprimer ou à remplacer par un mode.

Il paraît donc nécessaire de distinguer les questions à choix multiples des questions à choix unique.

Un moyen simple de les distinguer est de s'appuyer sur le nombre de valeurs contenues dans chaque colonne. Les choix multiples ont été éclatés en colonnes, contenant ainsi soit le choix retenu, soit une valeur manquante.

Par conséquent, les colonnes issues de questions à choix multiples ne contiennent que deux valeurs.

2.2. Traitement des questions à choix unique.

 # Récupération des valeurs uniques pour chaque colonne
 # Recherche des colonnes avec plus de 2 valeurs
 # « quest_u » regroupe les colonnes des questions à choix unique
 # « quest_m » regroupe les colonnes des questions à choix multiple

Nous pouvons à présent observer la présence ou non de valeurs manquantes sur les colonnes de questions à choix unique.

Après consultation de la méthodologie du sondage, il apparait que les questions 30 et 32 n'ont été posées qu'à certaines personnes, en fonction des réponses aux questions précédentes, d'où le ratio conséquent de NaNs.

Nous décidons de ne pas garder ces deux colonnes.

Pour les autres colonnes, nous remplaçons les valeurs manquantes par le mode de chaque colonne.

 # Remplacement des NaNs des colonnes choix unique par le mode des colonnes

2.3. Traitement des questions à choix multiples

Les questions à choix multiples sont plus difficiles à interpréter de par la manière dont elles ont été intégrées dans la base de données.

Pour chaque question, un nombre de colonnes équivalent au nombre de réponses proposées a été créé.

Pour chaque réponse, chaque colonne est remplie de la valeur de la réponse si le répondant a sélectionné ce choix et de NaNs sinon.

Il ne faut pas supprimer les valeurs manquantes sur ces colonnes car elles ont une utilité de compréhension.

Afin de pouvoir les utiliser et les analyser ultérieurement, il est cependant nécessaire de les encoder de manière binaire : 1 en cas de réponse positive, 0 si la valeur est manquante.

Dans un souci de flexibilité et de lisibilité, les différentes questions à choix multiples seront regroupées au sein de mini dataframes contenant en colonnes chacune des options proposées pour répondre à la question.

```
# Création de sub dataframes répliquant les intervalles créés pour les questions
# dfQ7 à dfQ35B contiennent toutes les questions à réponses multiples
```

Une vérification rapide nous montre que dfQ7 intègre toutes les informations relatives à la question 7.

Toutefois le libellé des colonnes est peu parlant. Nous allons donc les renommer en utilisant chaque choix comme nouveau titre. L'encodage se fera ensuite à partir du nouveau titre. Ce procédé permet aussi de supprimer tout risque de labélisation incorrecte, vu qu'il est très difficile de repérer à l'œil nu le nombre d'espaces présents dans des cellules textes.

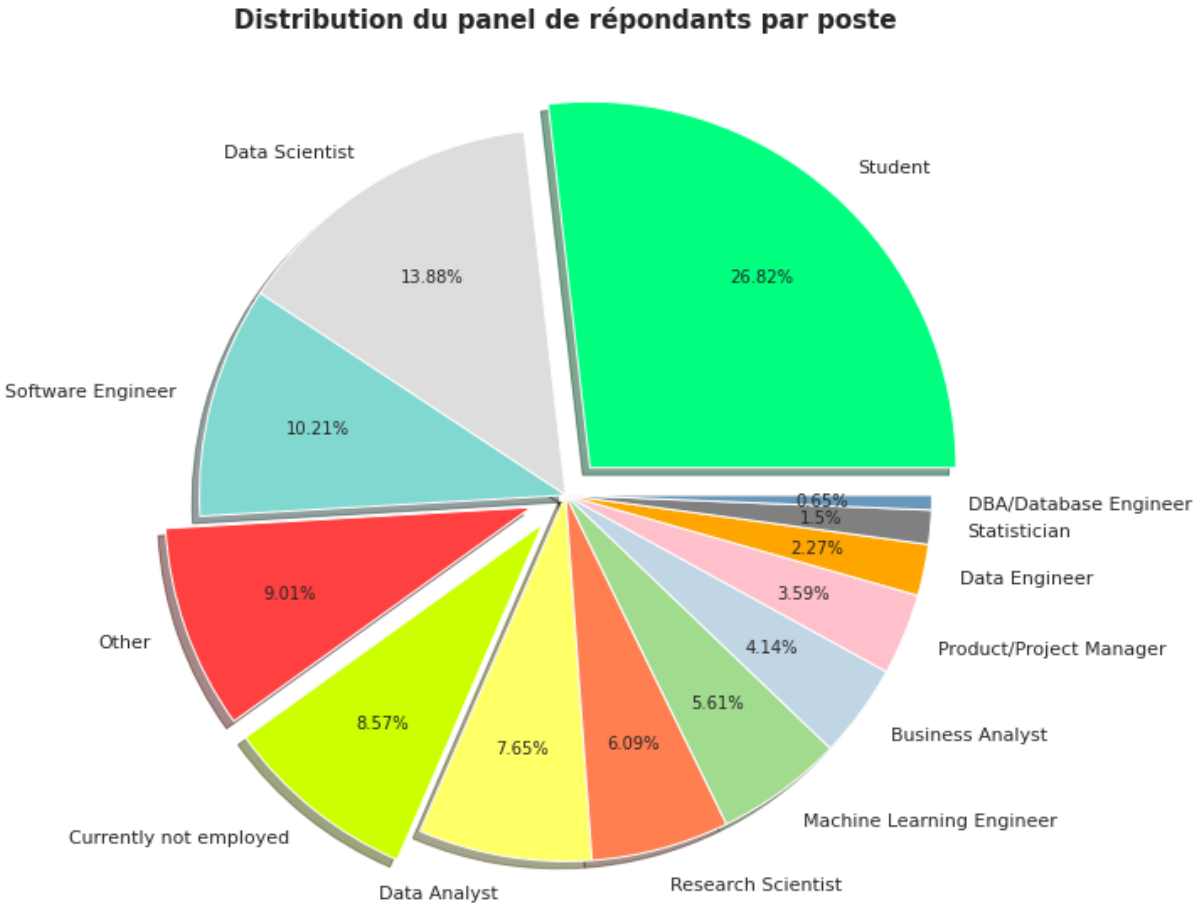
Le nettoyage ayant été réalisé, nous pouvons désormais émettre des hypothèses à rejeter ou non à l'aide de Data Visualization.

3. DATA VISUALIZATION

3.1. Hypothèse 1

Le domaine de la Data regroupe quatre ou cinq métiers ?

 # Visualisation des différents postes en DataScience (valeur cible)



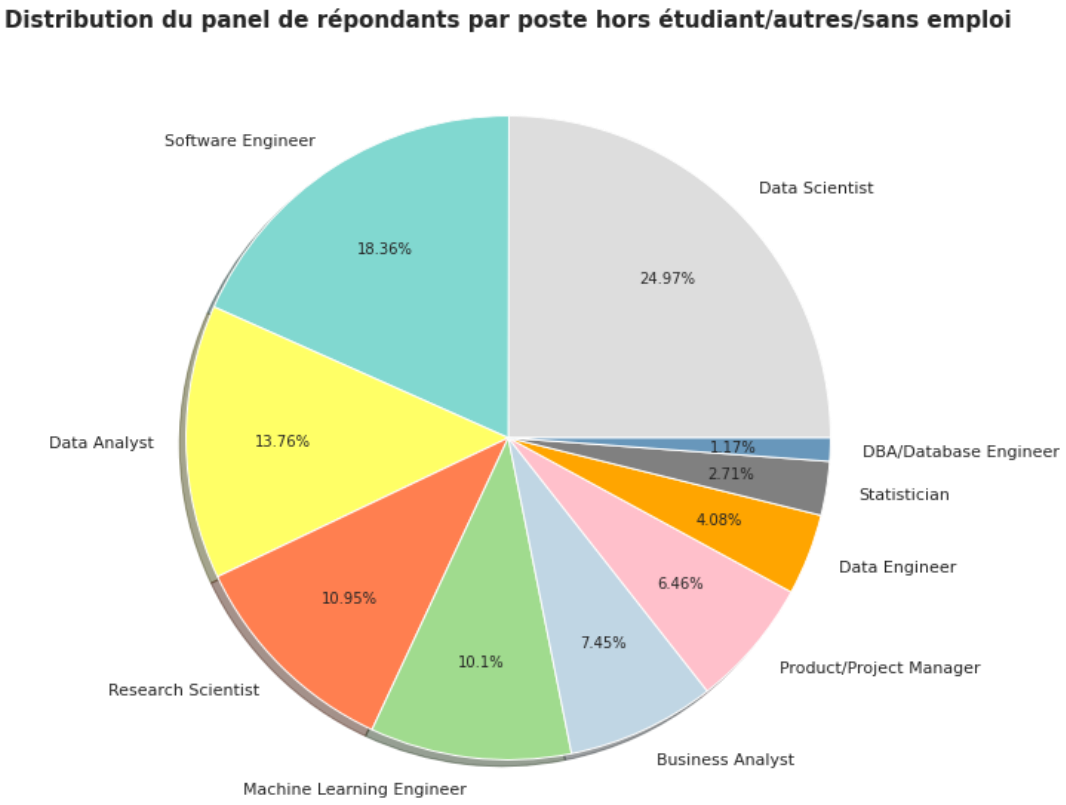
Cette première visualisation nous permet tout d'abord de rejeter notre hypothèse, car une dizaine de métiers sont représentés ici, dans le domaine de la Data.

Par ailleurs, nous constatons également trois catégories *('Student', 'Currently not employed', 'Other')* qui ne correspondent pas à des métiers et donc pas à des outils et compétences associés.

Il y a de fait un risque de biais conséquent sur la construction d'un modèle cherchant à prédire un métier sur une base d'outils et compétences techniques.

Afin de lever ce biais, nous excluons ces trois catégories "Non Professionnelles" pour la suite de l'analyse notre DataFrame.

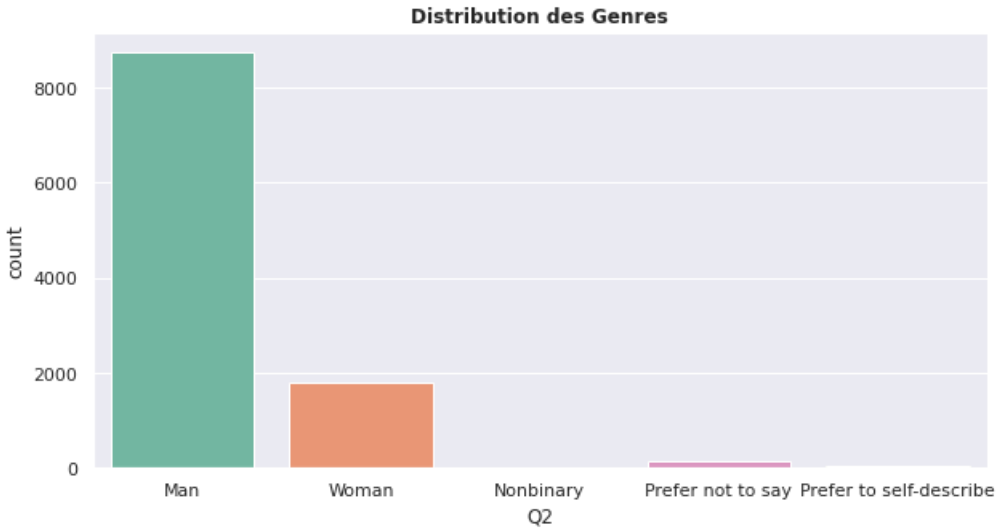
 # Validation du df "Professionnel" qui servira pour le reste de l'analyse



3.2. Hypothèse 2

Le domaine de la Data resterait masculinisé ?

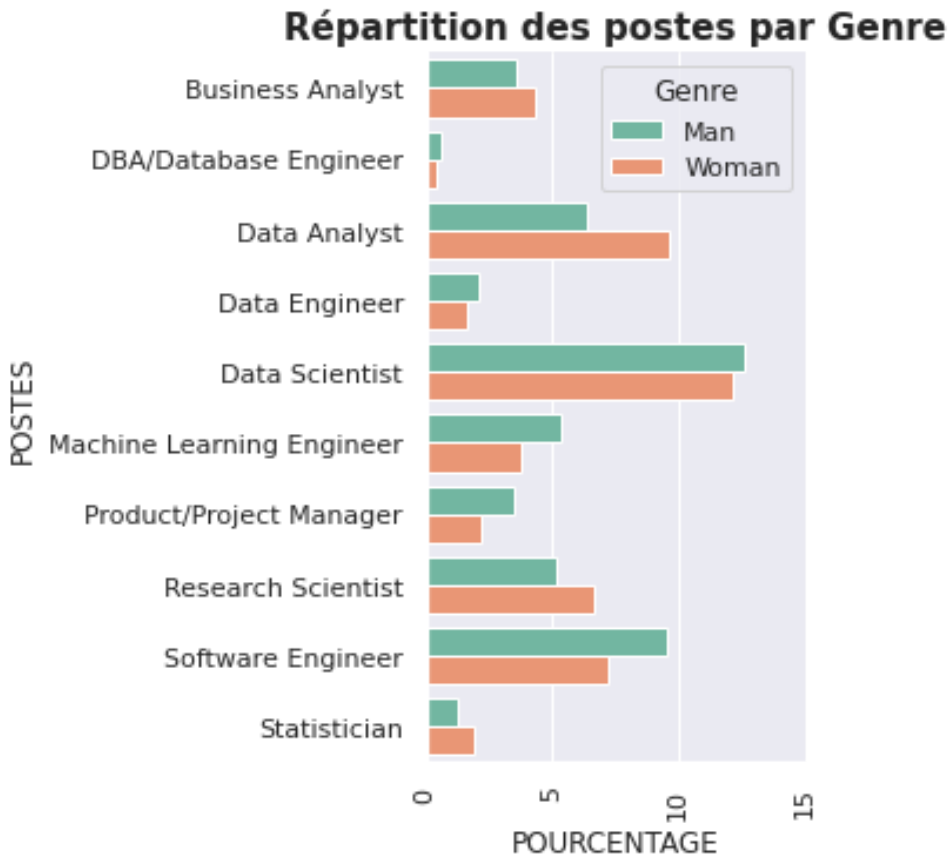
 # Visualisation de la répartition des Genres (Question 2)



Nous constatons un déséquilibre flagrant, notre hypothèse ne peut donc être rejetée. Il apparaît toutefois intéressant de pousser l'analyse sur la répartition au sein des différents types de poste.

Pour simplifier la visualisation, nous garderons seulement les valeurs 'Woman' et 'Man'.

 # Filtre du df sur les valeurs 'Woman' et 'Man'
 # Création d'une colonne pour comptage



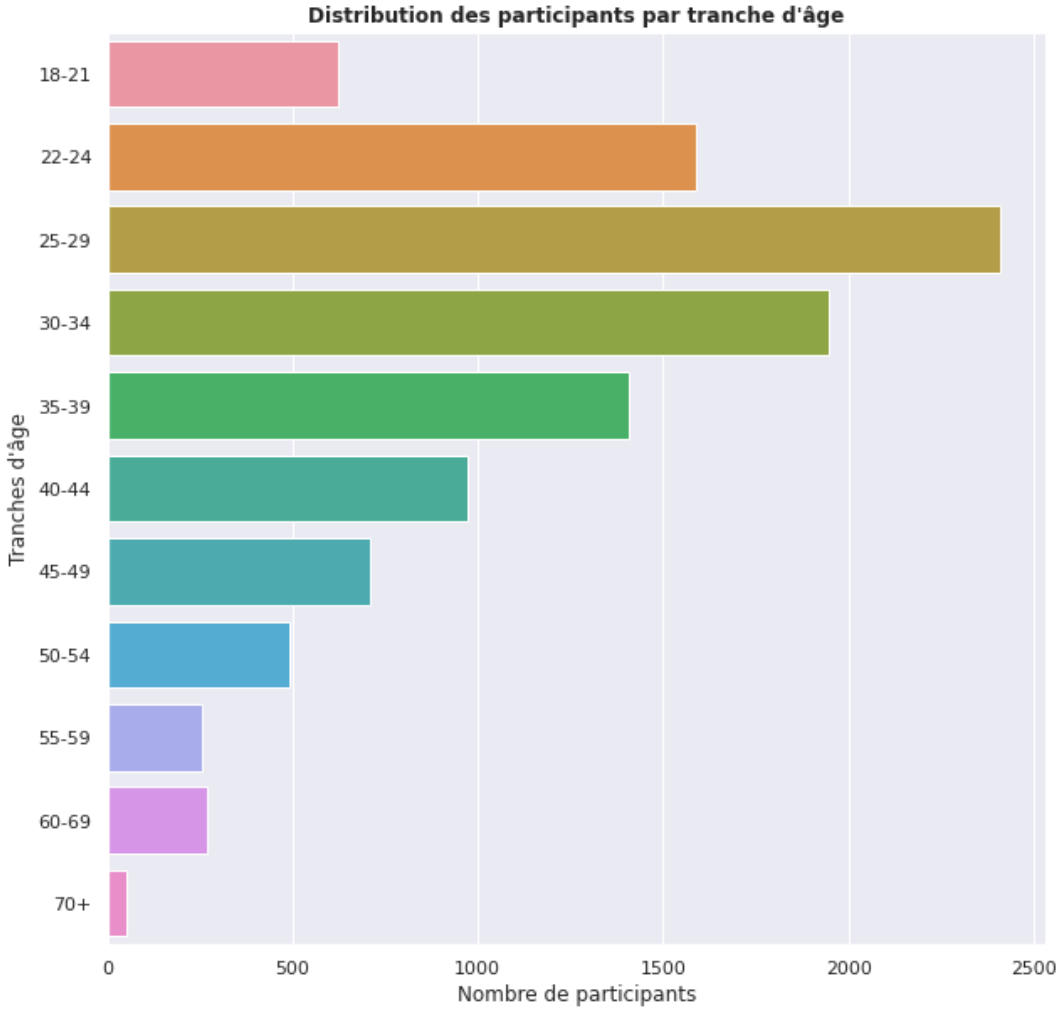
Nous constatons que les femmes sont proportionnellement plus nombreuses en Data Analysts et au même niveau en Data Scientists.

Par contre elles sont sous-représentées en Machine Learning Engineers et en Product Managers.

3.3. Hypothèse 3

Le domaine de la Data semble concerner un public de moins de 40 ans ?

 # Représentation de la distribution du panel de répondants par tranche d'âge



Nous constatons que la majorité des postes en Data est occupée par des moins de 40 ans, ce qui paraît naturel considérant l'émergence récente de ces métiers. Ces nouveaux intitulés de poste sont à même d'attirer une population jeune, les professionnels déjà établis conservant leurs titres. Les postes à pourvoir se réfèrent essentiellement à ces nouvelles appellations (Data Analyst vs. Statistician).

3.4. Hypothèse 4

Les "outils" les plus utilisés dans les métiers de la Data sont-ils ceux enseignés par DataScientest ?

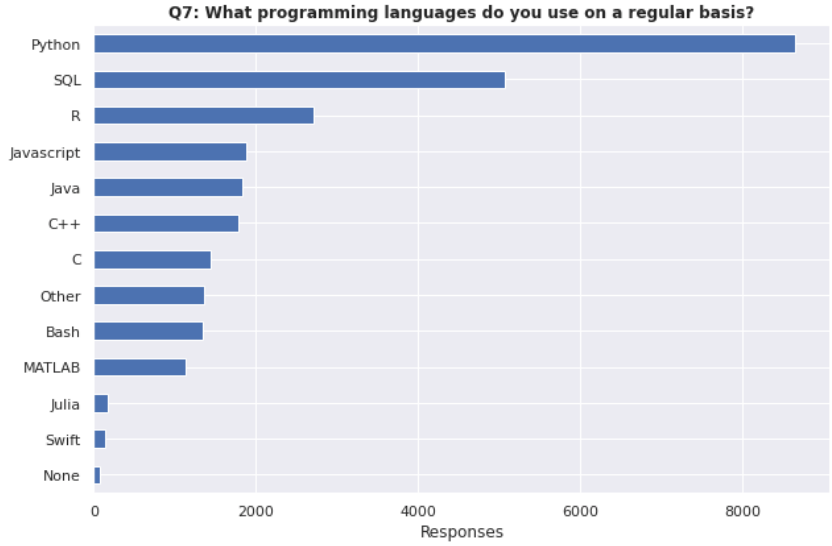
Afin d'illustrer de la manière la plus synthétique notre hypothèse, nous n'avons retenu que quatre questions portant respectivement sur l'utilisation quotidienne des outils suivants:

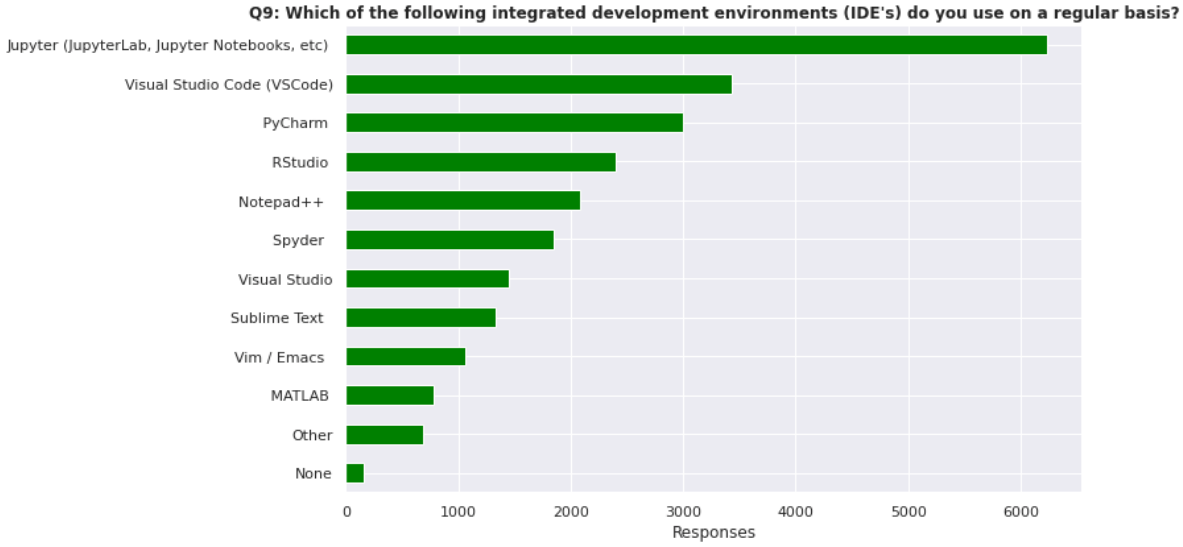
- *Langages de programmation (Q7)*
- *IDE (Q9)*
- *DataViz (Q14)*
- *Outils de Machine Learning (Q16)*

Tous ces éléments sont abordés via des questions à choix multiples, où plusieurs réponses sont possibles. Les catégories représentées ci-dessous ne seront donc pas nécessairement disjointes.

De plus, notre base d'étude se restreignant désormais uniquement aux professionnels, il est nécessaire d'ajuster les bases de données des questions 7, 9, 14 et 16 aux individus concernés.

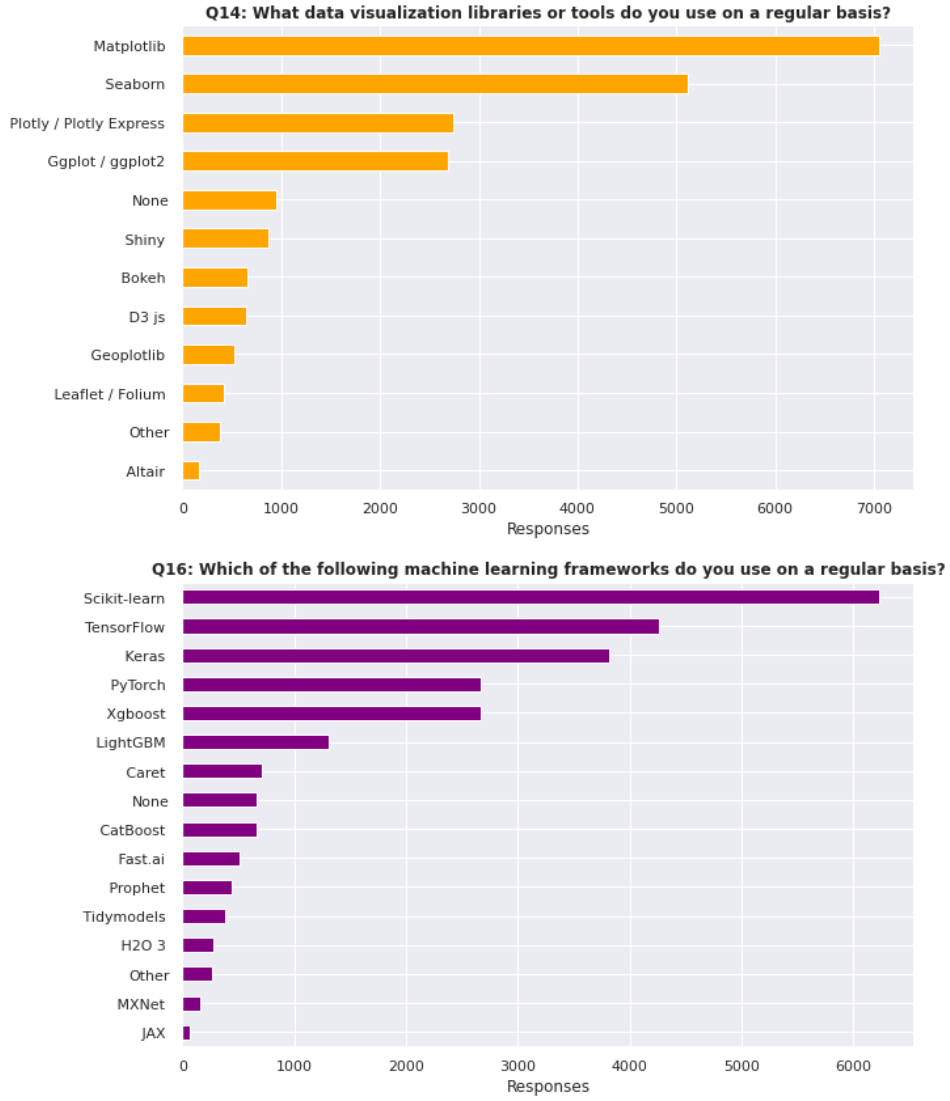
Ceci fait, nous travaillons bien désormais sur un groupe de 10,717 individus.





Python est de loin le langage de programmation le plus populaire dans les métiers de la data.

Jupyter Notebook est l'IDE privilégié, bien que d'autres IDE soient aussi utilisés.



Quand on parle de data visualization, Matplotlib et Seaborn se détachent clairement des autres outils à disposition.

Enfin le Machine Learning se fait majoritairement via Scikit-learn, même si TensorFlow et Keras ont aussi un certain succès.

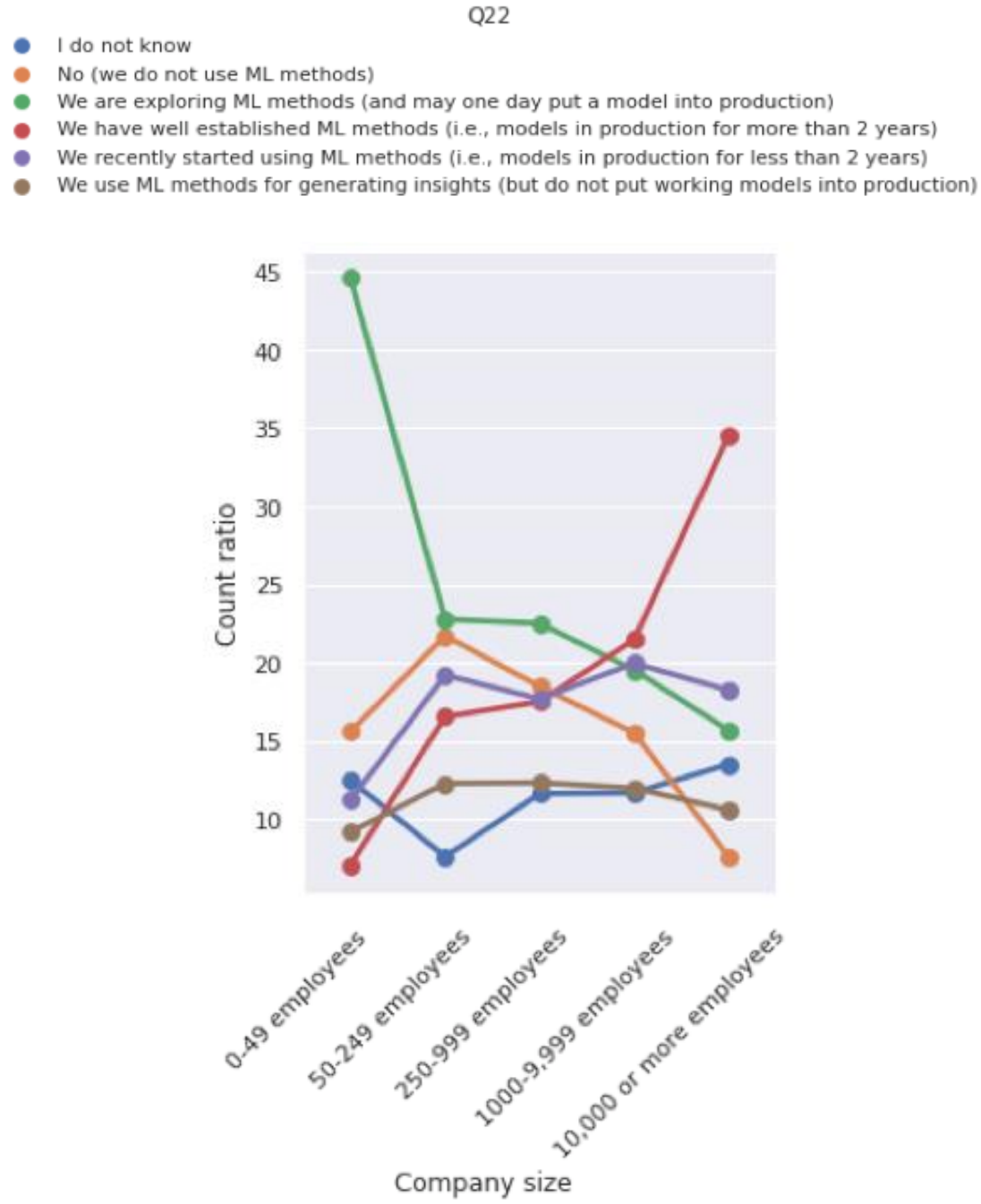
Le contenu de formation de DataScientest semble refléter de façon assez correcte, les outils actuellement utilisés par les professionnels de la data.

3.5. Hypothèse 5

Big Data = Big Company ?

Le Machine Learning n'est-il utilisé que dans les grandes entreprises ?

Répartition de l'utilisation de ML en fonction de la taille de l'entreprise



On voit clairement que plus la taille de l'entreprise est grande, plus les modèles de ML sont non seulement en production depuis plus de 2 ans, mais aussi ancrés dans la culture d'entreprise.

On constate toutefois que le ML est en cours de développement quelle que soit la taille de l'entreprise.

Donc nous rejetons notre hypothèse.

II. Élaboration de modèles et premières itérations

1. Synthèse des étapes précédentes

Pour réaliser cette première itération, nous repartons du dataset auquel ont été apportées les modifications suivantes :

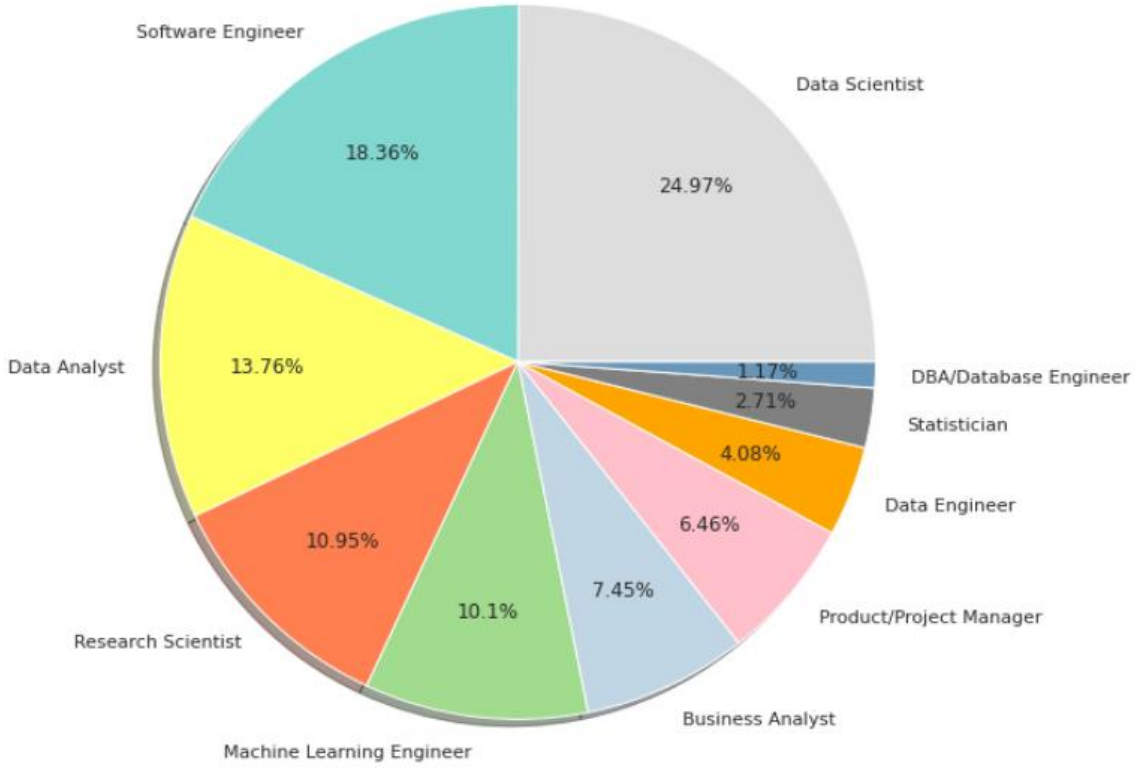
- Suppression des NaNs sur la variable cible ('Q5')
- Remplacement des NaNs par le mode pour chaque variable correspondant à des questions à choix unique
- Encodage des colonnes des questions à choix multiples par 0/1 selon NaN ou valeur
- Réduction du Dataset aux entrées des participants professionnels (ayant une profession précise, hors 'étudiant', 'sans emploi' et 'autre')

2. Fin du preprocessing

2.1. Analyse de la variable cible

Comme vu précédemment, la valeur cible est 'Q5' (poste actuellement occupé) et après un premier traitement, elle contient 10 valeurs uniques.

Distribution du panel de répondants par poste hors étudiant/autres/sans emploi



Le domaine de la Data et ses métiers sont en constante évolution. Ainsi nous décidons de restreindre notre analyse aux cinq principaux métiers, représentant près de 80% du panel.

 # Filtre du data set sur les 5 principaux métiers
Notre dataset contient à présent 8375 entrées et 344 colonnes.

2.2. Traitement des features

Pour rappel, notre objectif est de créer un modèle capable de proposer un poste en fonction de compétences et d'outils associés. Par conséquent, en analysant les questions posées, nous pouvons supprimer une partie des colonnes.

Voici la liste des colonnes concernées et notre raisonnement:

Q#	***Thème de la question***	***Raisonnement***
Q1, Q2, Q3, Q4	Age, Gender, Country, Education Level	Dans un souci d'éthique, cette variable ne peut être un élément différenciant pour une suggestion de poste
Q8	Code writing experience	Il s'agit d'une recommandation, donc point de vue subjectif, et non d'une compétence liée à un poste précis
Q11, Q12, Q13	Computing platform, Hardware, TPU used	Il s'agit d'une habitude, donc point de vue subjectif, et non d'une compétence liée à un poste précis
Q18, Q19, Q27A, Q28A, Q30, Q32, Q34A + Q26B à Q35B	Computer vision, NLP, Cloud computing, ML products, Big Data, BI, Automated ML tools + B Questions	Question dépendant d'une réponse précédente et n'ayant pas été posée à tout le panel
Q20, Q21, Q22, Q24, Q25	Company size, Data staff, ML implemented, Salary, ML Budget	Question liée à l'entreprise et non au poste du répondant
Q36, Q37, Q39	Sharing channels, Training platform, Favorite media sources	Il ne s'agit pas d'une compétence technique ou d'un outil lié au poste
Q38	Primary tools	Il s'agit d'une réponse par texte libre

Après suppression des colonnes, notre dataset contient 8375 entrées et 154 colonnes.

2.3. Encodage des colonnes restantes

L'ensemble des questions à choix multiple a déjà été traité précédemment. Il nous reste à encoder les colonnes Q6 et Q15.

Aperçu du dataset final:

	Q5	Q6	Q15	Q7:Python	Q7:R	Q7:SQL	Q7:C	Q7:C++	Q7:Java	Q7:Javascript	Q7:Julia	Q7:Swift	Q7:Bash
18600	Machine Learning Engineer	4	6	1	0	1	0	0	0	0	0	0	0
11086	Machine Learning Engineer	5	6	1	1	0	0	0	0	0	0	0	0

2 rows × 154 columns

3. Itération 1: Choix du modèle d'apprentissage

Création d'un vecteur 'target' contenant la variable cible 'Q5' et d'un Data Frame 'feats' contenant les différentes features.

 # Séparation du dataset en train set et test set

Notre variable cible étant une variable catégorielle composée de classes, nous utiliserons par la suite des modèles de classification.

3.1. Méthode de l'Arbre de Décision

Score Train set du DecisionTree : 97.5 %
Score Test set du DecisionTree : 39.701 %

Rapport de Classification Arbre de Décision

	precision	recall	f1-score	support
Data Analyst	0.31	0.33	0.32	294
Data Scientist	0.51	0.50	0.51	549
Machine Learning Engineer	0.21	0.23	0.22	207
Research Scientist	0.29	0.31	0.30	235
Software Engineer	0.51	0.44	0.47	390
accuracy			0.40	1675
macro avg	0.36	0.36	0.36	1675
weighted avg	0.41	0.40	0.40	1675

Matrice de confusion de l'Arbre de Décision

Classe prédite / Classe réelle	Data Analyst	Data Scientist	Machine Learning Engineer	Research Scientist	Software Engineer
Data Analyst	106	94	30	22	42
Data Scientist	94	266	69	63	57
Machine Learning Engineer	21	59	53	41	33
Research Scientist	41	41	32	79	42
Software Engineer	53	62	49	47	179

3.2. Méthode de Régression Logistique

Score Train set de la Régression Logistique 58.25 %
Score Test set de la Régression Logistique 54.30 %

Rapport de Classification Regression Logistique

	precision	recall	f1-score	support
Data Analyst	0.50	0.54	0.52	294
Data Scientist	0.57	0.66	0.61	549
Machine Learning Engineer	0.41	0.29	0.34	207
Research Scientist	0.48	0.36	0.41	235
Software Engineer	0.60	0.64	0.62	390
accuracy			0.54	1675
macro avg	0.51	0.49	0.50	1675
weighted avg	0.53	0.54	0.53	1675

Matrice de confusion de la Regression Logistique

Classe prédite / Classe réelle	Data Analyst	Data Scientist	Machine Learning Engineer	Research Scientist	Software Engineer
Data Analyst	158	84	5	10	37
Data Scientist	78	361	35	30	45
Machine Learning Engineer	17	69	59	22	40
Research Scientist	26	58	24	84	43
Software Engineer	34	57	21	30	248

3.3. Méthode des plus proches voisins (Minkowski, n=3)

Score Train set de la méthode des k plus proches voisins 63.13 %
Score Test set de la méthode des k plus proches voisins 38.20 %

Rapport de Classification Méthode des k plus proches voisins

	precision	recall	f1-score	support
Data Analyst	0.27	0.59	0.37	294
Data Scientist	0.44	0.46	0.45	549
Machine Learning Engineer	0.26	0.17	0.20	207
Research Scientist	0.41	0.20	0.27	235
Software Engineer	0.59	0.34	0.44	390
accuracy			0.38	1675
macro avg	0.40	0.35	0.35	1675
weighted avg	0.42	0.38	0.38	1675

Matrice de confusion des k plus proches voisins

Classe prédite / Classe réelle	Data Analyst	Data Scientist	Machine Learning Engineer	Research Scientist	Software Engineer
Data Analyst	172	84	6	13	19
Data Scientist	206	252	36	26	29
Machine Learning Engineer	52	82	35	12	26
Research Scientist	73	70	27	47	18
Software Engineer	123	84	33	16	134

3.4. Méthode SVM - Support Vector Machine

Score Train set du modèle SVM 74.25 %
Score Test set du modèle SVM 54.33 %

Rapport de Classification du modèle SVM

	precision	recall	f1-score	support
Data Analyst	0.46	0.50	0.48	294
Data Scientist	0.54	0.72	0.62	549
Machine Learning Engineer	0.48	0.26	0.33	207
Research Scientist	0.57	0.34	0.43	235
Software Engineer	0.64	0.59	0.61	390
accuracy			0.54	1675
macro avg	0.54	0.48	0.49	1675
weighted avg	0.54	0.54	0.53	1675

Matrice de confusion du modèle SVM

Classe prédite / Classe réelle	Data Analyst	Data Scientist	Machine Learning Engineer	Research Scientist	Software Engineer
Data Analyst	146	110	1	9	28
Data Scientist	75	398	22	20	34
Machine Learning Engineer	17	89	53	13	35
Research Scientist	32	67	19	81	36
Software Engineer	47	75	16	20	232

3.5. Méthode - Random Forest

Score Train set du modèle Random Forest 97.51 %
Score Test set du modèle Random Forest 53.37 %

Rapport de Classification du modèle Random Forest

	precision	recall	f1-score	support
Data Analyst	0.52	0.45	0.48	294
Data Scientist	0.55	0.75	0.63	549
Machine Learning Engineer	0.40	0.21	0.27	207
Research Scientist	0.59	0.34	0.43	235
Software Engineer	0.60	0.64	0.62	390
accuracy			0.55	1675
macro avg	0.53	0.48	0.49	1675
weighted avg	0.54	0.55	0.53	1675

Matrice de confusion du modèle Random Forest

Classe prédite / Classe réelle	Data Analyst	Data Scientist	Machine Learning Engineer	Research Scientist	Software Engineer
Data Analyst	132	114	2	8	38
Data Scientist	56	413	26	15	39
Machine Learning Engineer	14	84	43	17	49
Research Scientist	24	71	17	79	44
Software Engineer	29	75	19	16	251

3.6. Tableau de synthèse des résultats

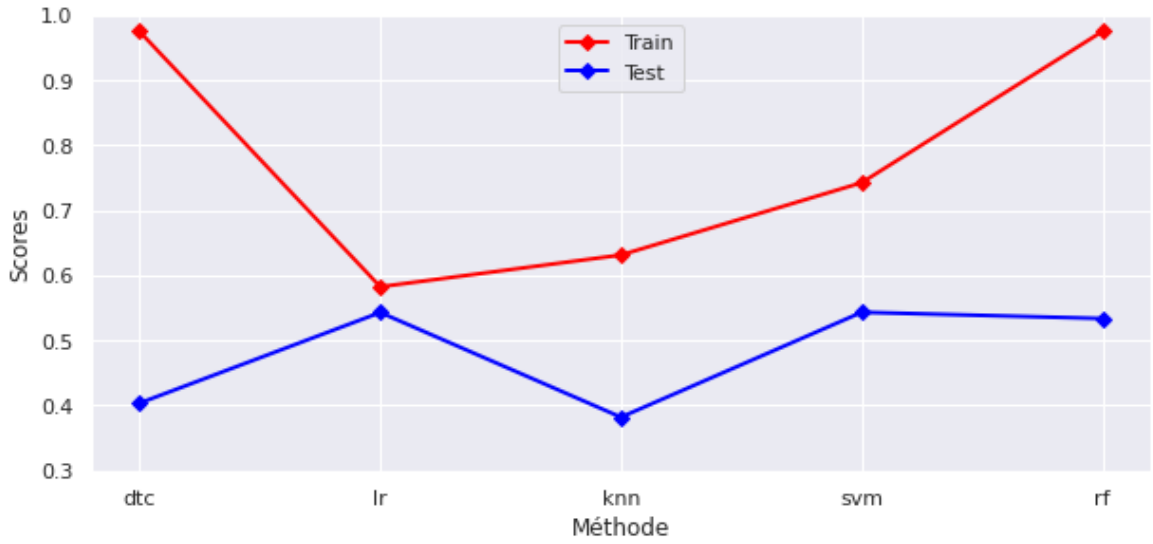
	Method	Method_Name	Score_Train	Score_Test	precision(macro_avg)	recall(macro_avg)	f1_score(macro_avg)	precision(weighted_avg)	recall(weighted_avg)	f1_score(weighted_avg)
0	dtc	DecisionTreeClassifier	0.9750	0.4036	0.37	0.37	0.37	0.41	0.40	0.40
1	lr	LogisticRegression	0.5825	0.5430	0.51	0.49	0.50	0.53	0.54	0.53
2	knn	KNeighborsClassifier	0.6313	0.3820	0.40	0.35	0.35	0.42	0.38	0.38
3	svm	svm.SVC	0.7425	0.5433	0.54	0.48	0.49	0.54	0.54	0.53
4	rf	RandomForestClassifier	0.9751	0.5337	0.52	0.46	0.47	0.53	0.53	0.52

Nous observons grâce aux rapports de classification qu'il est difficile, quels que soient les modèles, de distinguer précisément le Data Scientist du Data Analyst.

A contrario, le poste de Software Engineer semble présenter des spécificités qui aident à son identification. Tandis que le Machine Learning Engineer obtient les scores de précision et de rappel les plus bas.

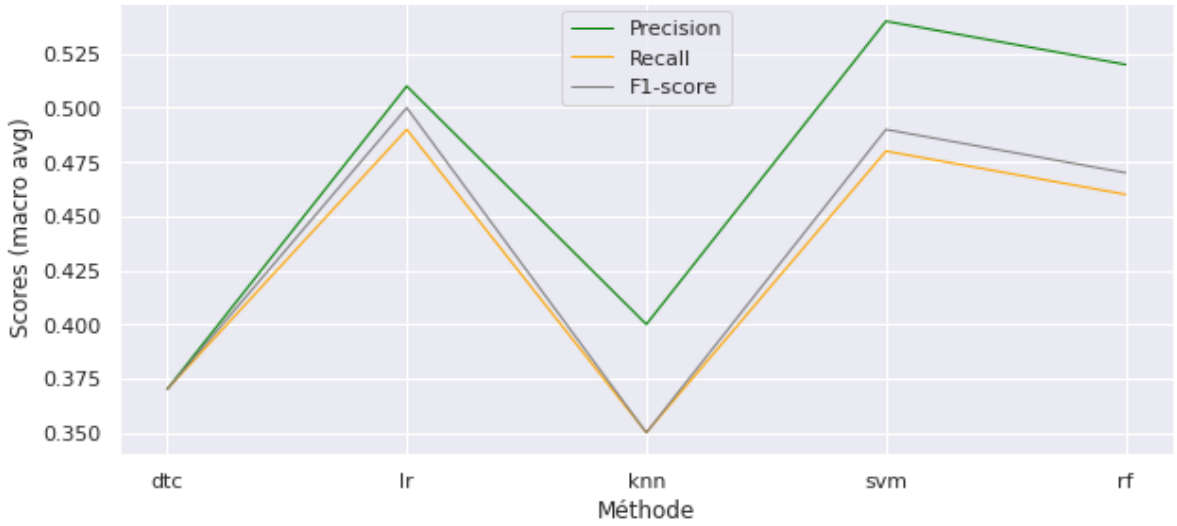
Ceci pourrait être lié à la nature même des métiers et aux tâches qui leur sont associées.

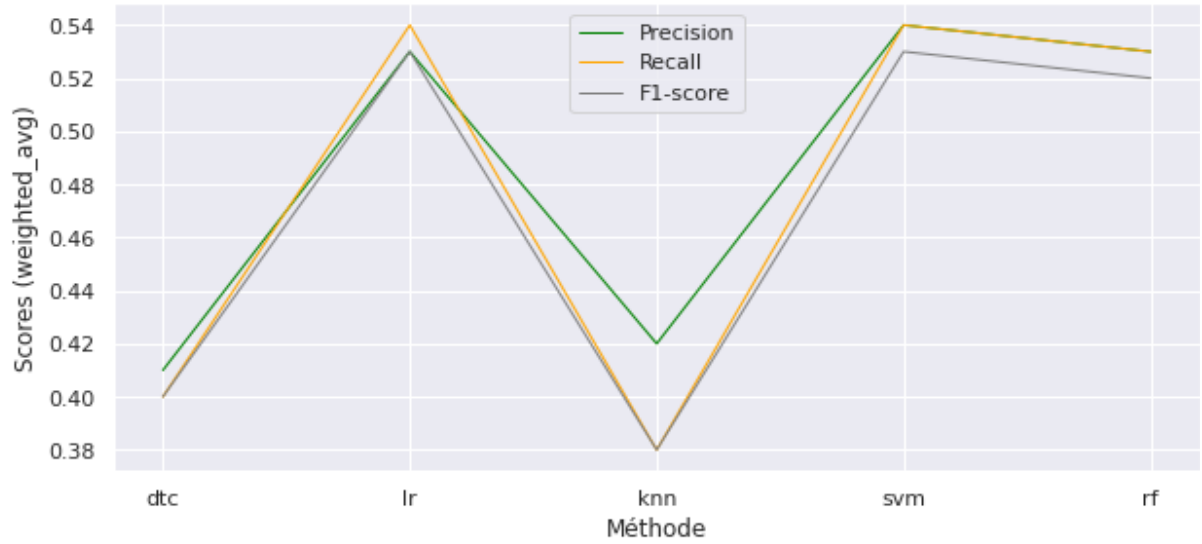
4. Comparaison des scores de chaque modèle entraîné



Les résultats des différents modèles entraînés présentent des disparités flagrantes.

- Les scores de test restent cantonnés entre 38 et 54%, une performance relativement faible.
- Les scores d'entraînement mettent en évidence du sur apprentissage sur 4 des modèles (en particulier Arbre de Décision et Forêts Aléatoires, environ 97%).
- On distingue 3 groupes de modèles : DTC et KNN à très faibles scores tests et sur apprentissage important, SVM et RF à sur apprentissage important, et LR.





Les moyennes pondérées ou non pondérées présentent des résultats similaires.

DTC et KNN ont 10 points de moins en précision et rappel que les autres modèles.

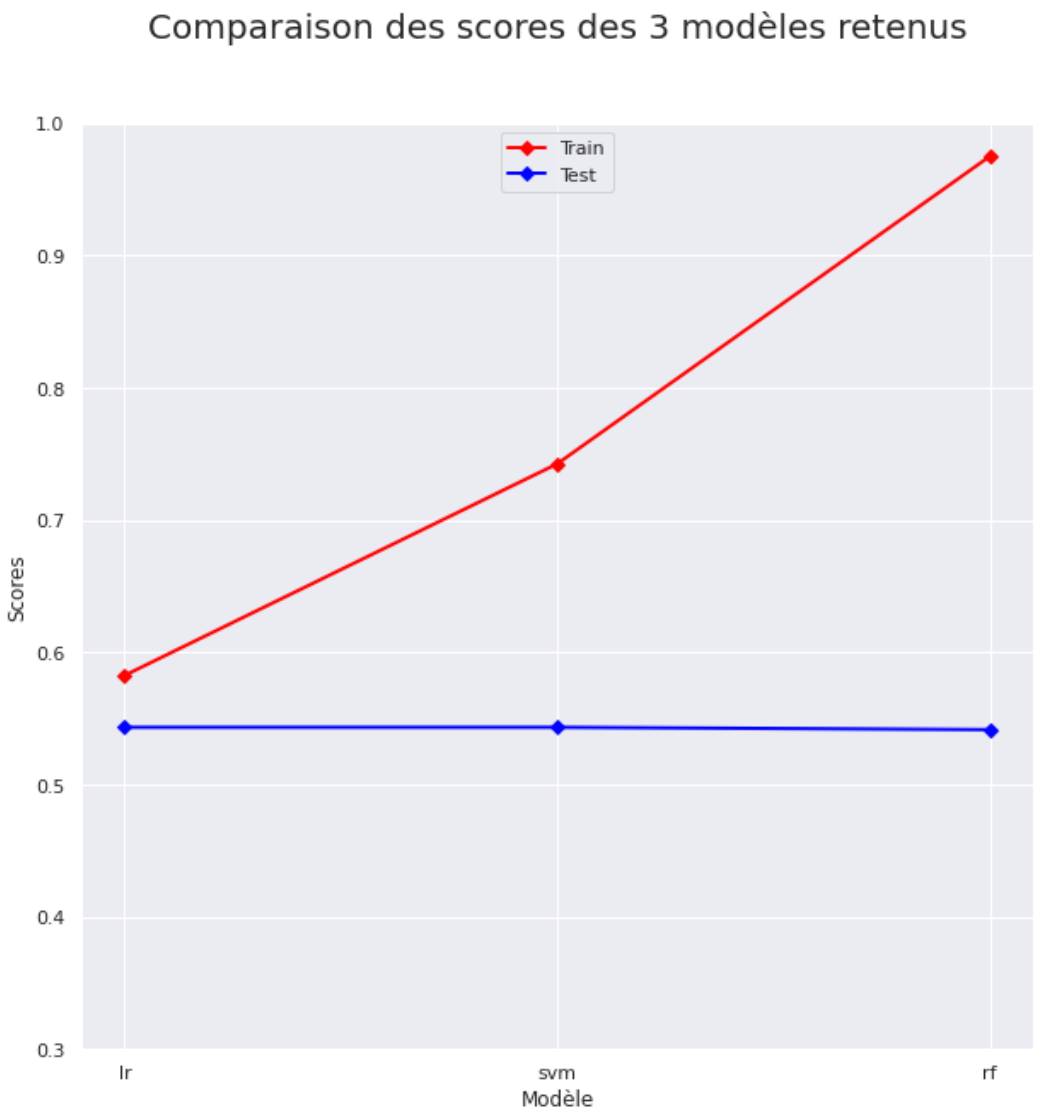
Nous choisissons donc d'écarter ces 2 modèles et de nous concentrer sur les 3 autres pour l'itération suivante, à savoir: LR, SVM, et RF.

III. Interprétation et Itération 2

1. Introduction

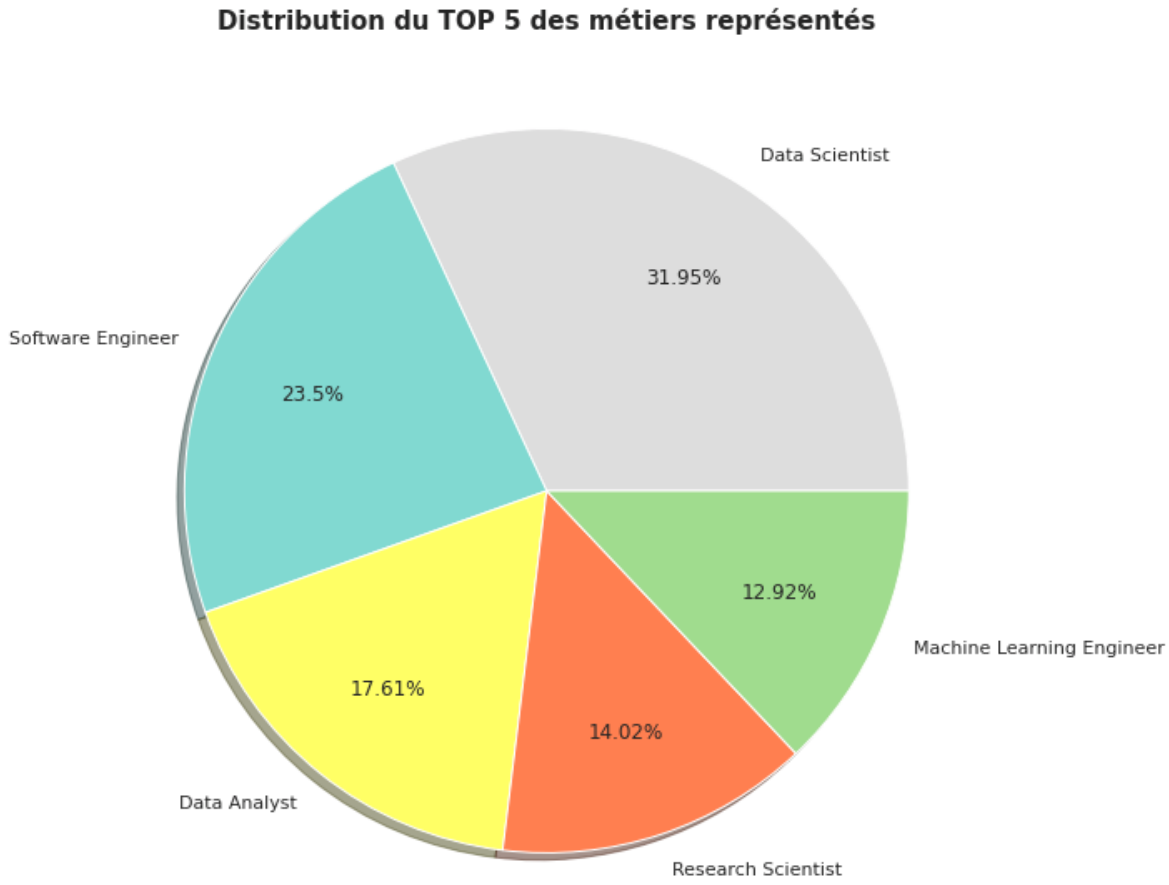
Nous reprenons donc les résultats des 3 modèles choisis.

	Modele	Modele_Name	Score_Train	Score_Test
0	lr	LogisticRegression	0.582537	0.543284
1	svm	SVM	0.742537	0.543284
2	rf	RandomForestClassifier	0.975075	0.541493



La visualisation des résultats avait permis de mettre en évidence un phénomène d'overfitting ainsi que des performances à améliorer.

Le phénomène d'overfitting peut s'expliquer en partie par le déséquilibre de la variable cible, que l'on peut voir au travers de la répartition des 5 classes.



2. Matrices de corrélation

Nous pouvons nous intéresser à la corrélation des variables, qui peut être source d'overfitting, en cas de corrélation forte.

2.1. Encodage de la variable cible pour établir une heatmap

La heatmap ne met pas en évidence des couleurs tranchées quant aux corrélations de la variable cible avec les features, regardons de plus près ses coefficients de corrélation.

NB: Avec 153 features, la heatmap n'est pas très lisible, elle n'est donc pas affichée dans ce rapport.

2.2. Corrélations principales entre la variable cible et les feats.

	Q5
Q5	1.000000
Q23:None of these activities are an important part of my role at work	0.285103
Q7:Javascript	0.277479
Q7:Java	0.262264
Q6	0.236462
...	...
Q14: Ggplot / ggplot2	-0.151714
Q16: Xgboost	-0.152320
Q9: RStudio	-0.183081
Q7:R	-0.219783
Q23:Analyze and understand data to influence product or business decisions	-0.339449

154 rows × 1 columns

2.3. Analyse statistique rapide des corrélations de la variable cible

```
count    154.000000
mean      -0.005192
std        0.117428
min       -0.339449
25%       -0.050201
50%       -0.013285
75%        0.021769
max        1.000000
Name: Q5, dtype: float64
```

Les corrélations observées entre la variable cible (Q5: métiers de la data) et les variables features ne sont pas très élevées. Leur écart-type est de 11,7%. De plus aucune variable n'est fortement corrélée positivement (maximum 28.5%) ou négativement (minimum -33.9%) à la variable cible.

Nous allons chercher à corriger cela au travers de trois étapes :

- **Réduction de dimensions**
- **Ré-échantillonnage**
- **Optimisation des hyperparamètres**

3. Réduction de dimensions

Avant de tester des sélecteurs, il peut être intéressant de voir l'importance des variables au sein des modèles. Voici en exemple pour le Random Forest :

	Feature importances
Q6	0.046383
Q15	0.040503
Q23:Analyze and understand data to influence product or business decisions	0.025927
Q7:Javascript	0.023957
Q23:None of these activities are an important part of my role at work	0.020384
Q7:SQL	0.019392
Q7:Java	0.018825
Q23:Build prototypes to explore applying machine learning to new areas	0.017486
Q7:R	0.016046
Q23:Do research that advances the state of the art of machine learning	0.015802
Q9:Visual Studio Code (VSCode)	0.015124
Q23:Experimentation and iteration to improve existing ML models	0.015087
Q9:Jupyter (JupyterLab, Jupyter Notebooks, etc)	0.014268

Nous constatons que les importances sont faibles, 0.046 pour la plus élevée.
Les réponses relatives aux questions 7 (langages de programmation) et 23 (tâches quotidiennes) reviennent parmi les variables les plus importantes.

3.1. Utilisation de SelectKBest

3.1.1.Paramètre score_func = mutual_info_classif

```
Les 10 features retenues sont :
 Index(['Q6', 'Q15', 'Q7:SQL', 'Q7:Java', 'Q7:Javascript',
        'Q17:Gradient Boosting Machines (xgboost, lightgbm, etc)',
        'Q23:Analyze and understand data to influence product or business decisions',
        'Q23:Build prototypes to explore applying machine learning to new areas',
        'Q23:Experimentation and iteration to improve existing ML models',
        'Q23:None of these activities are an important part of my role at work'],
       dtype='object')
```

LR	SVM	RF
Score train set: 50.04 %	Score train set: 51.58 %	Score train set: 65.12 %
Score du test set: 50.03 %	Score test set: 48.9 %	Score test set: 45.67 %

3.1.2.Paramètre score_func = f_classif

```
Les 10 features retenues sont :
 Index(['Q15', 'Q7:R', 'Q7:Java', 'Q7:Javascript',
        'Q17:Gradient Boosting Machines (xgboost, lightgbm, etc)',
        'Q23:Analyze and understand data to influence product or business decisions',
        'Q23:Build prototypes to explore applying machine learning to new areas',
        'Q23:Build and/or run a machine learning service that operationally improves my product or workflows',
        'Q23:Experimentation and iteration to improve existing ML models',
        'Q23:None of these activities are an important part of my role at work'],
       dtype='object')
```

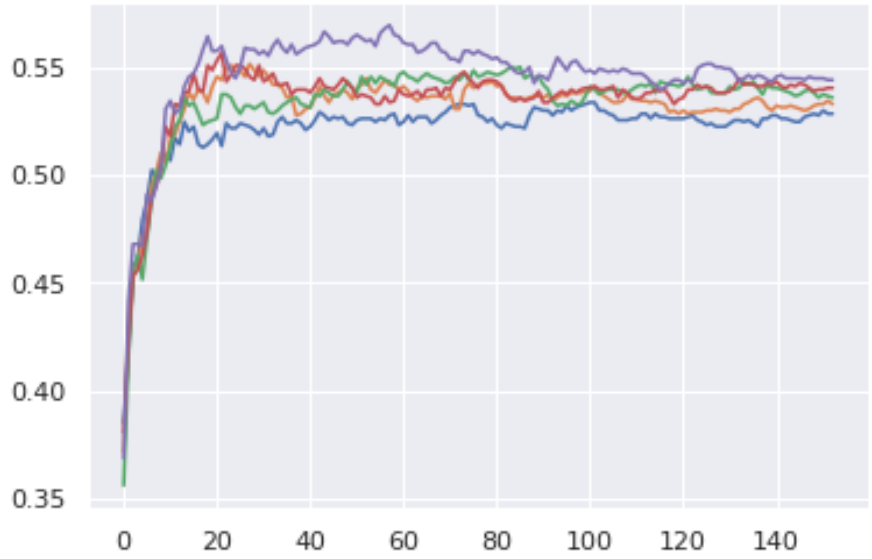
LR	SVM	RF
Score train set: 49.88 %	Score train set: 51.51 %	Score train set: 58.6 %
Score test set: 50.21 %	Score test set: 49.25 %	Score test set: 46.87 %

3.2. Utilisation de SelectFromModel

LR	SVM	RF
Nombre de features : 44	Nombre de features : 45	Nombre de features : 57
Score train set: 56.27 %	Score train set: 56.37 %	Score train set: 97.39 %
Score test set: 55.28 %	Score test set: 55.22 %	Score test set: 53.19 %

3.3. Utilisation de RFECV sur LR

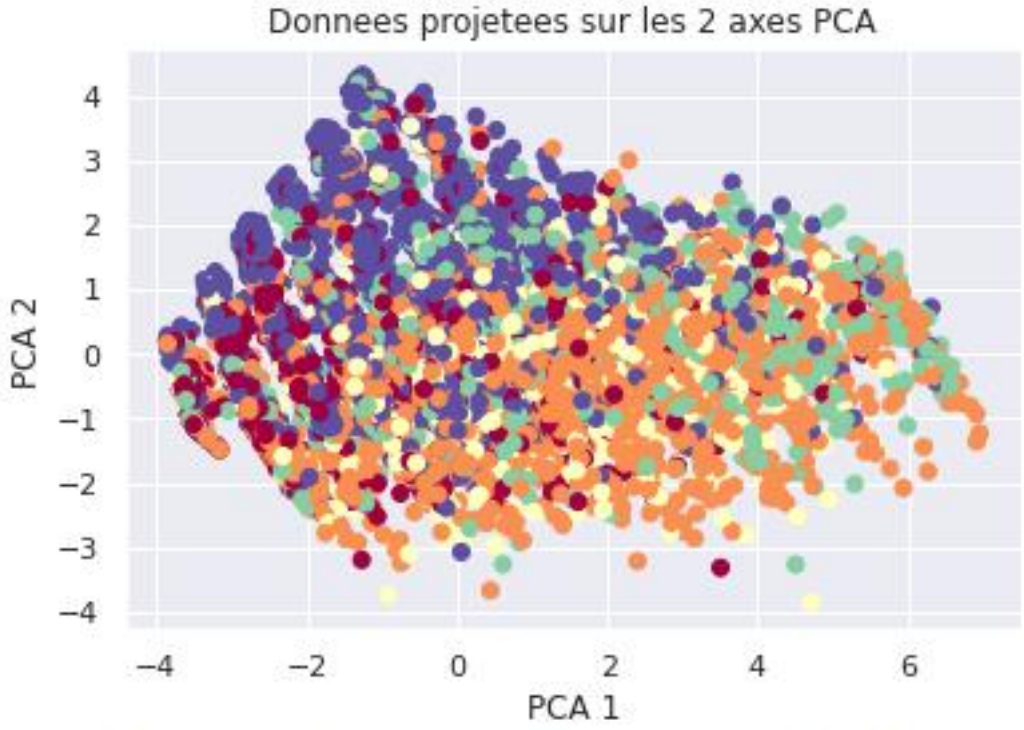
```
La rfecv de la Regression Logistique retient 74  features.
```



Cette méthode appliquée au modèle SVM n'a donné aucun résultat.
Quant au modèle Random Forest, les résultats variaient beaucoup à chaque exécution.

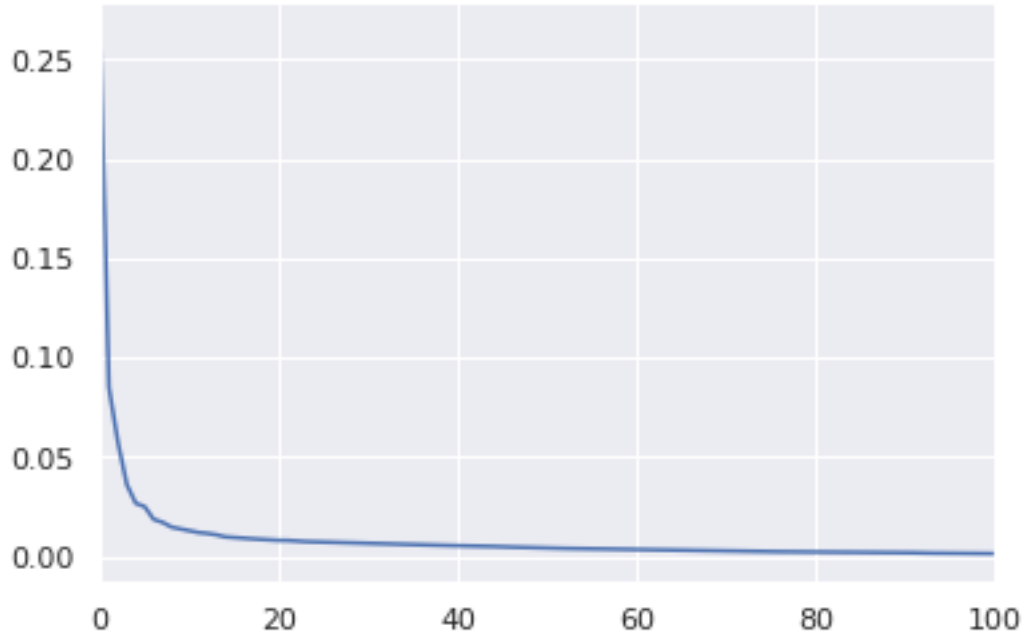
3.4. Utilisation de PCA

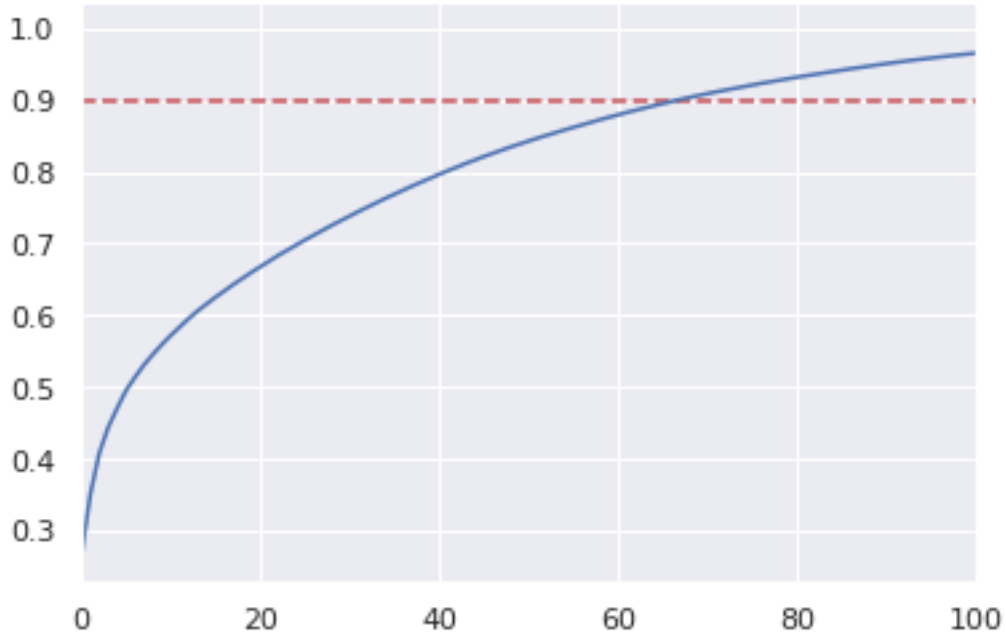
 # Initialisation de PCA avec la variable cible encodée
 # Premier test avec 2 composantes principales




```
La part de variance expliquee est : 0.35
```

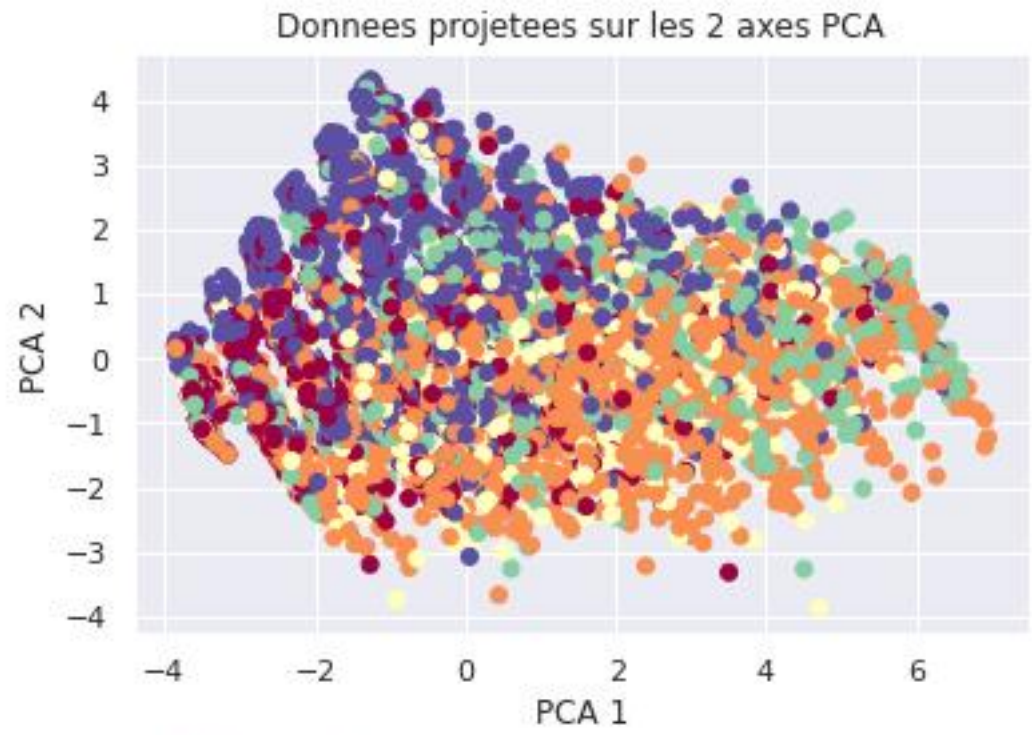
 # Courbes Explained Variance Ratio et EVR cumsum





On a pour habitude de considérer que conserver 90% de la variance d'un jeu de données est un seuil acceptable pour la PCA. Sur le deuxième graphe, on peut lire qu'il faut environ 70 Composantes Principales pour obtenir 90% de variance expliquée.

Nous réinitialisons un PCA, en gardant cette fois 90% de la variance expliquée:




```
La part de variance expliquee est : 0.9
```

Utilisation de PCA sur chaque modèle

LR
Score du train set: 57.19 %
Score du test set: 55.04 %

SVM
Score du train set: 69.04 %
Score du test set: 54.75 %

RF
Score du train set: 97.51 %
Score du test set: 52.78 %

Ces divers tests n'ont pas permis d'améliorer significativement les scores test et train initialement observés pour chacun des modèles.

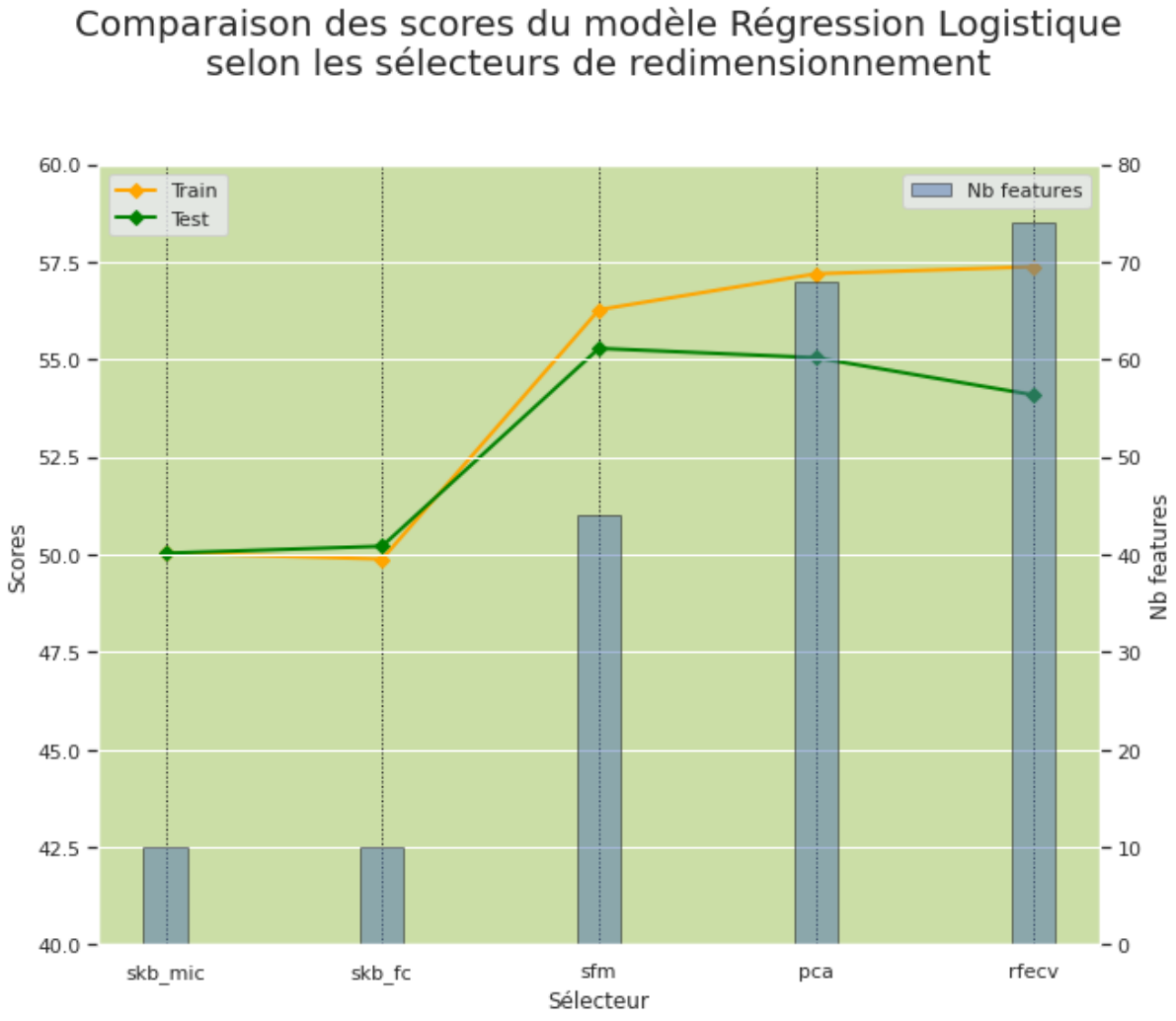
Le modèle de Régression Logistique semble se distinguer.

3.5. Comparaison des résultats obtenus

Nota Bene : La visualisation de ces comparaisons a été effectuée a posteriori de nos recherches d'optimisation et n'a donc pas été prise en compte pour ces dernières

Logistic Regression

	Selector	Selector_Name	Nb_features	Score_Train	Score_Test
0	skb_mic	SelectKBest_score_func_mutual_info_classif	10	50.04	50.03
1	skb_fc	SelectKBest_score_func_f_classif	10	49.88	50.21
2	sfm	SelectFromModel	44	56.27	55.28
3	pca	PCA	68	57.19	55.04
4	rfecv	RFECV	74	57.37	54.09

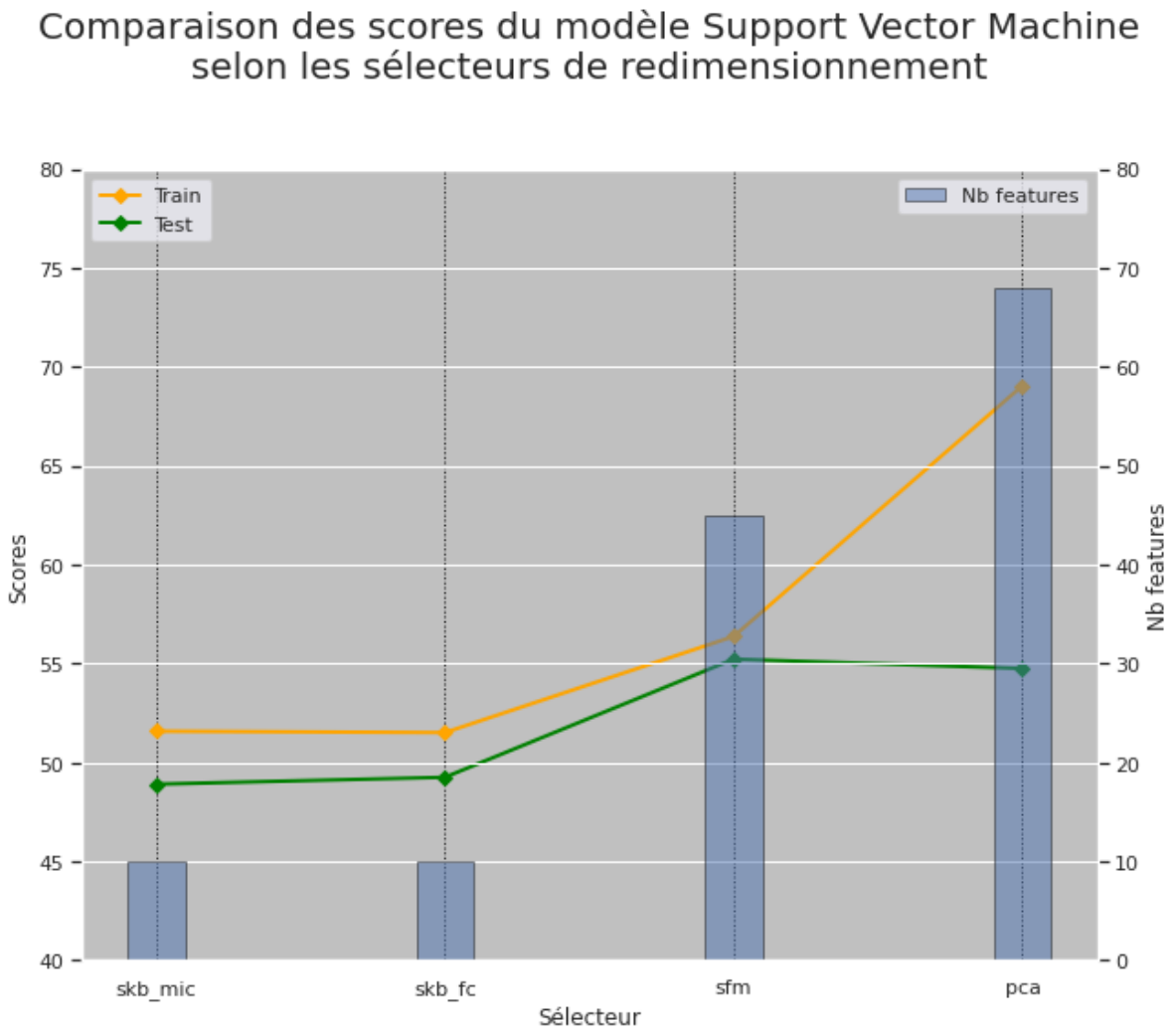


SelectFrom Model offre le meilleur compromis entre le nombre de variables retenues et les scores obtenus.

Une recherche du k optimum pour le SelectKBest aurait peut-être pu obtenir des résultats comparables ou meilleurs que le SelectFromModel.

Support Vector Machines

	Selector	Selector_Modele	Nb_features	Score_Train	Score_Test
0	skb_mic	SelectKBest_score_func_mutual_info_classif	10	51.58	48.90
1	skb_fc	SelectKBest_score_func_f_classif	10	51.51	49.25
2	sfm	SelectFromModel	45	56.37	55.22
3	pca	PCA	68	69.04	54.75

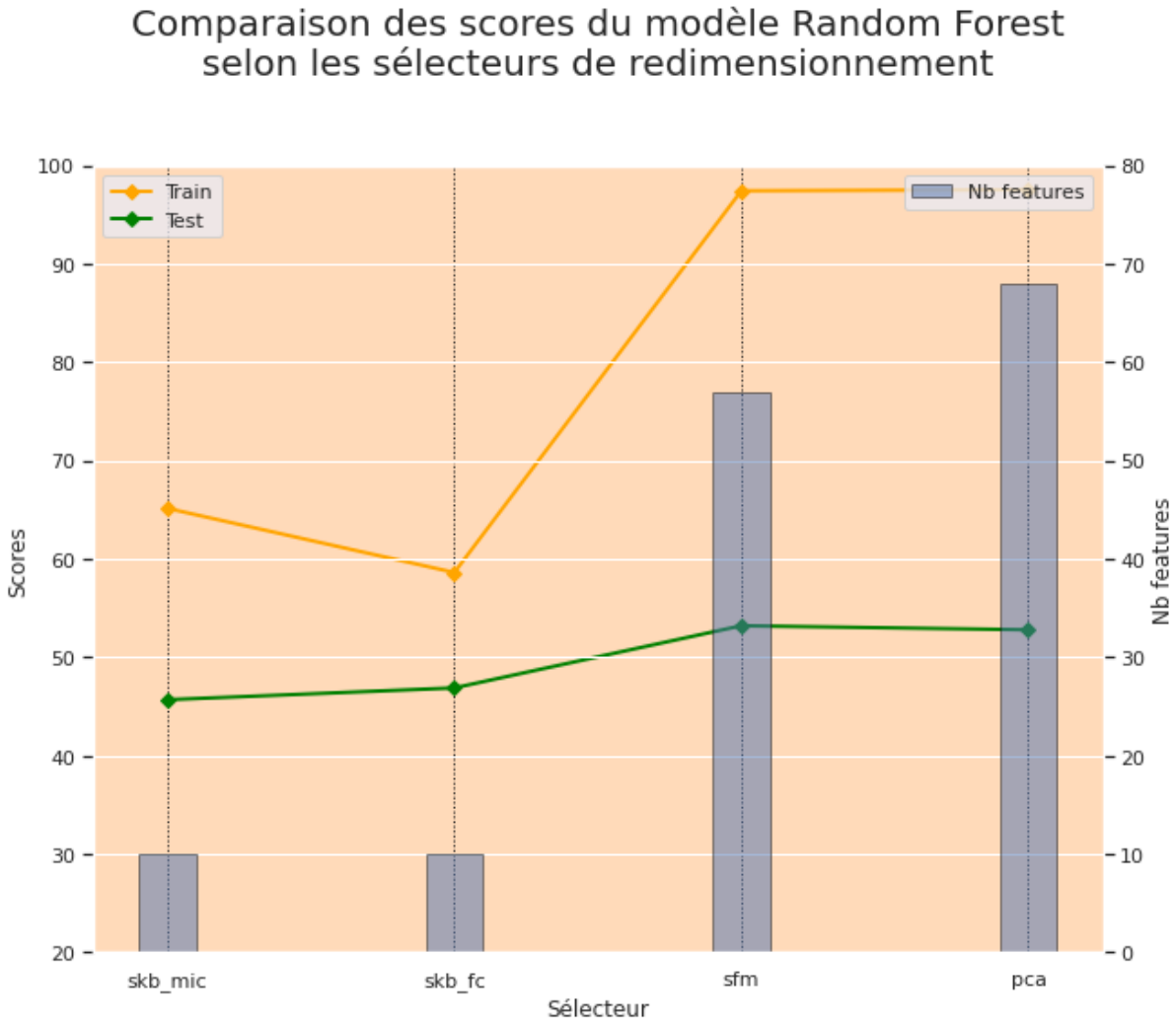


SelectFromModel offre le meilleur compromis entre le nombre de variables retenues et les scores obtenus, tout en réduisant le sur apprentissage.

Une recherche du k optimum pour le SelectKBest aurait peut-être pu obtenir des résultats comparables ou meilleurs que le SelectFromModel.

Random Forest

	Selector	Selector_Modele	Nb_features	Score_Train	Score_Test
0	skb_mic	SelectKBest_score_func_mutual_info_classif	10	65.12	45.67
1	skb_fc	SelectKBest_score_func_f_classif	10	58.60	46.87
2	sfm	SelectFromModel	57	97.39	53.19
3	pca	PCA	68	97.51	52.78



Nous n'avons pas réussi à trouver de compromis sans détériorer le score test, qui a souffert de nos efforts de réduction du sur apprentissage.

Des méthodes plus adaptées aux classifieurs de type arbre existent sûrement, nous n'avons pas su les identifier.

4. Ré-échantillonnage

4.1. Utilisation de RandomUnderSampler

Classes échantillon undersampled : {'Data Analyst': 875, 'Software Engineer': 875, 'Machine Learning Engineer': 875, 'Research Scientist': 875, 'Data Scientist': 875}

LR	SVM	RF
Score Train set: 54.96 %	Score Train set: 66.91 %	Score Train set: 82.01 %
Score Test set: 50.51 %	Score Test set: 50.33 %	Score Test set: 49.97 %

4.2. Utilisation de ClusterCentroids

Classes échantillon ClusterCentroids : {'Data Analyst': 875, 'Software Engineer': 875, 'Machine Learning Engineer': 875, 'Research Scientist': 875, 'Data Scientist': 875}

LR	SVM	RF
Score Train set: 52.84 %	Score Train set: 64.81 %	Score Train set: 73.27 %
Score Test set: 49.01 %	Score Test set: 49.19 %	Score Test set: 44.84 %

4.3. Utilisation de class_weight

4.3.1.Class_weight = dictionnaire manuel

A partir de la répartition des cinq classes de la variable cible, nous pouvons rééquilibrer les classes :

```
class_weight =      {'Data Scientist' : 0.10,
                     'Software Engineer' : 0.15,
                     'Data Analyst' : 0.25,
                     'Research Scientist' : 0.25,
                     'Machine Learning Engineer' : 0.25})
```

LR	SVM	RF
Score Train set: 54.76 %	Score Train set: 57.03 %	Score Train set: 97.46 %
Score Test set: 49.07 %	Score Test set: 49.79 %	Score Test set: 53.07 %

4.3.2.Class_weight = 'balanced' (uniquement sur SVM)

Score Train set: 74.03 %	Score Test set: 52.06 %

4.4. Utilisation de RandomOverSampling (uniquement sur SVM)

Score Train set: 77.52 %	Score Test set: 53.07 %

4.5. Utilisation de SMOTE (uniquement sur SVM)

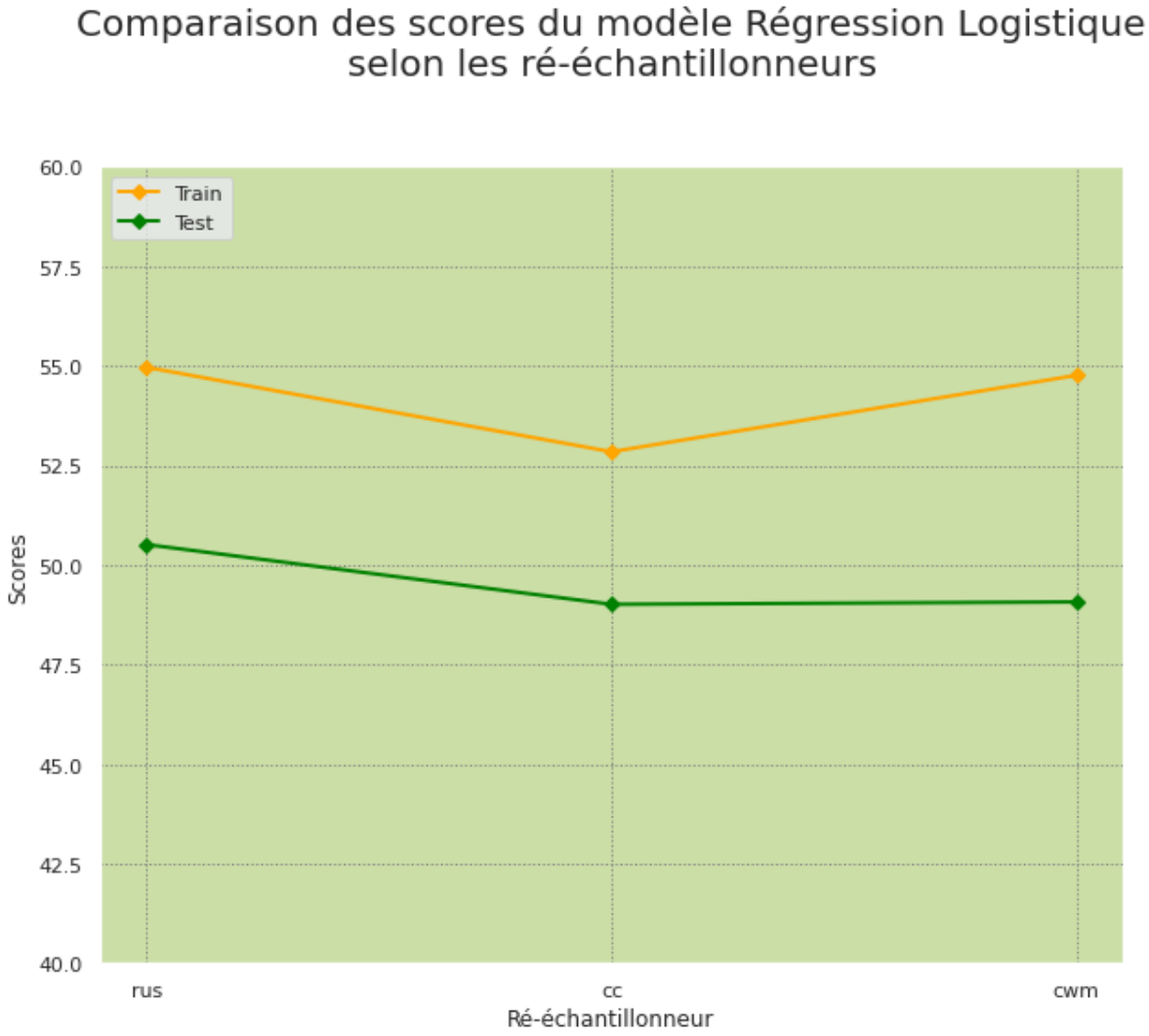
Score Train set: 78.73 %	Score Test set: 52.06 %

4.6. Comparaison des résultats obtenus

Nota Bene : La visualisation de ces comparaisons a été effectuée a posteriori de nos recherches d'optimisation et n'a donc pas été prise en compte pour ces dernières

Logistic Regression

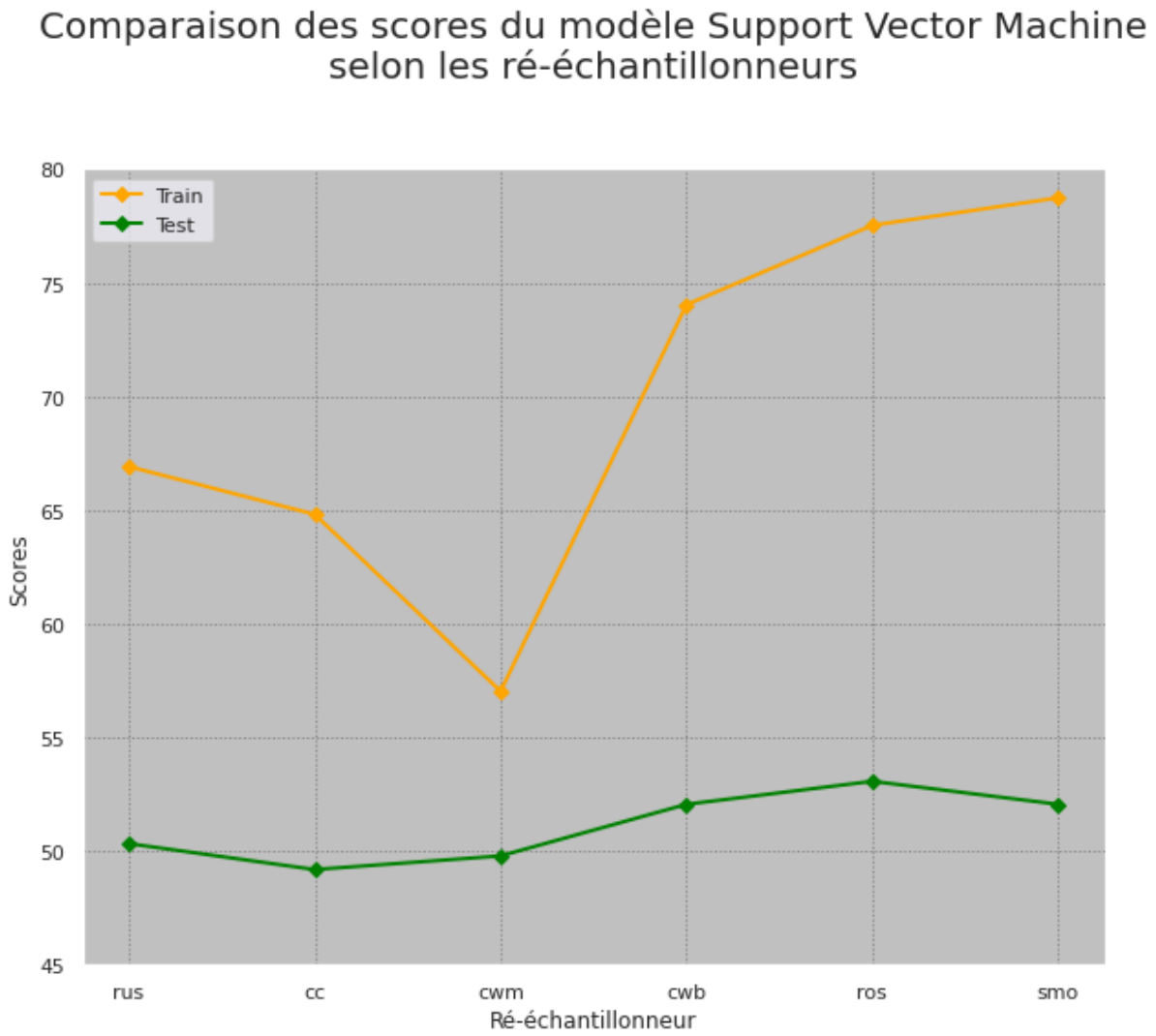
	Sampler	Sampler_Name	Score_Train	Score_Test
0	rus	RandomUnderSampler	54.96	50.51
1	cc	ClusterCentroids	52.84	49.01
2	cwm	class_weight_manual	54.76	49.07



Depuis le début de nos tests, le modèle de régression logistique présente un faible écart entre les scores train et test.

Support Vector Machines

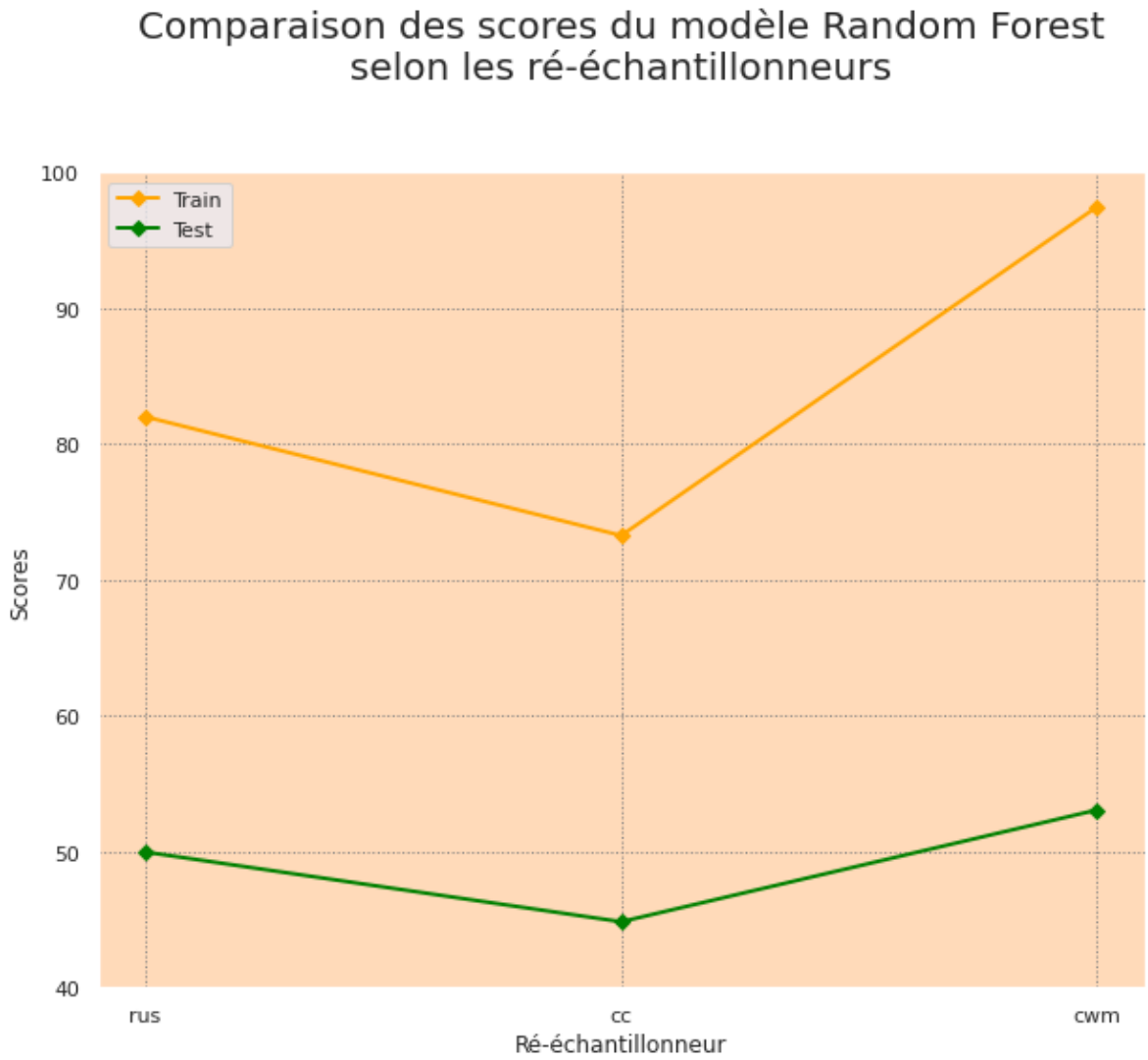
	Sampler	Sampler_Name	Score_Train	Score_Test
0	rus	RandomUnderSampler	66.91	50.33
1	cc	ClusterCentroids	64.81	49.19
2	cwm	class_weight_manual	57.03	49.79
3	cwb	class_weight_balanced	74.03	52.06
4	ros	RandomOverSampler	77.52	53.07
5	smo	SMOTE	78.73	52.06



On remarque l'amélioration des scores avec l'application d'un ré échantillonneur par pondération manuelle.

Random Forest

	Sampler	Sampler_Name	Score_Train	Score_Test
0	rus	RandomUnderSampler	82.01	49.97
1	cc	ClusterCentroids	73.27	44.84
2	cwm	class_weight_manual	97.46	53.07



Nous ne sommes pas parvenus à corriger le sur apprentissage. De la même manière que pour le redimensionnement, il existe peut-être des outils adaptés à ce type de classifieur.

5. Optimisation des hyperparamètres

Suite à ces différents tests, nous optons pour une optimisation personnalisée de chaque modèle.

5.1. Optimisation du modèle de Régression Logistique

Réduction de dimension	RFECV avec cv = 5
Ré échantillonnage	class_weight manuel
Optimisation des hyperparamètres	gridSearchCV avec cv = 5

Train score du Pipeline_lr : 54.25
Test score du Pipeline_lr : 52.54

Nombre de features retenus pour la Logistique Régression : 33
Meilleurs paramètres retenus pour la Logistique Régression :
{'C': 0.01, 'solver': 'liblinear'}

5.2. Optimisation du modèle de Support Vector Machine

Malgré de nombreuses tentatives, nous n'avons pas réussi à exécuter la méthode du pipeline sur ce modèle. Toutefois, nous tentons une optimisation à l'aide du sélecteur 'SelectFromModel' et d'un kernel 'linear'.

Réduction de dimension	SelectFromModel
Ré échantillonnage	NaN
Optimisation des hyperparamètres	kernel = ‘linear’

Train score : 56.37 %
Test score : 55.22 %
Nombre de features retenues : 45

5.3. Optimisation du modèle de Random Forest

Réduction de dimension	RFECV avec cv calculé sur KFold
Ré échantillonnage	NaN
Optimisation des hyperparamètres	GridSearchCV avec cv calculé sur KFold

Pour ce modèle, les tests s'exécutent entre 15min et 30min, en fonction des cv (3 à10) avec ou sans class_weight.

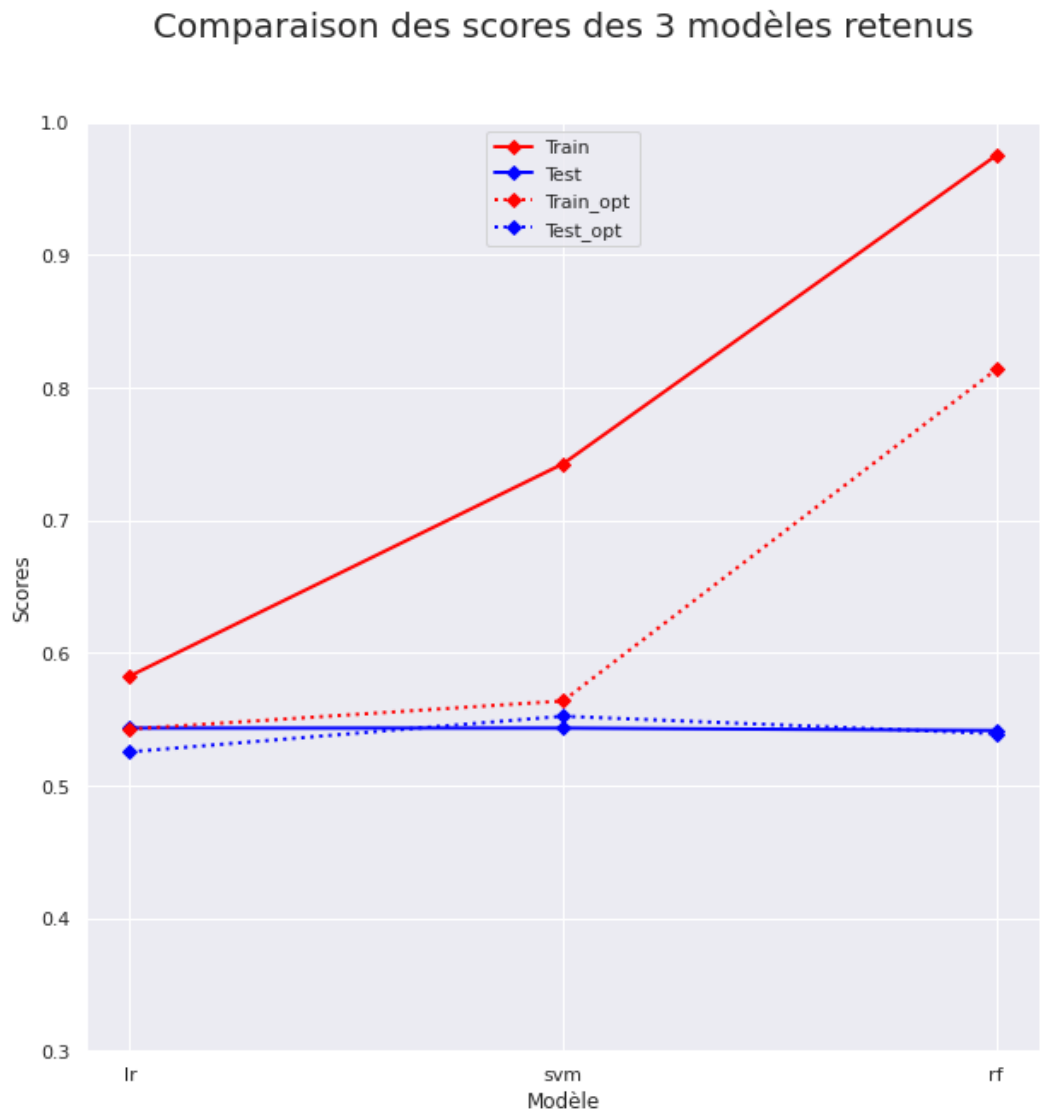
Après regroupement sur ce notebook, ce modèle n'a pas ressorti les mêmes résultats, à savoir:
score train 0.8137 / score test 0.5391

Train score du Pipeline_rf : 97.51 %
Test score du Pipeline_rf : 54.45 %

Nombre de features retenues pour le modèle Random Forest : 136
Meilleurs paramètres retenus pour le modèle Random Forest :
{'max_features': 'sqrt', 'min_samples_leaf': 1, 'n_estimators': 500}

5.4. Comparaison des performances des modèles optimisés

	Modele	Modele_Name	Score_Train	Score_Test	Score_Train_opt	Score_Test_opt
0	lr	LogisticRegression	0.582537	0.543284	0.542537	0.525373
1	svm	SVM	0.742537	0.543284	0.563731	0.552239
2	rf	RandomForestClassifier	0.975075	0.541493	0.813700	0.539100



Commentaires :

LR → Nous avons pu réduire l'overfitting au détriment d'une légère détérioration de performance.

SVM → L'overfitting a été corrigé, et nous avons obtenu une petite amélioration de la performance.

RF → Bien que considérablement réduit, l'overfitting reste important. La performance est restée la même.

Il est possible que nous n'ayons pas su trouver les sélecteurs et transformeurs les plus adaptés au modèle Random Forest pour obtenir de meilleurs résultats.

IV. PyGraal – Conclusion et perspectives

Notre étude des différents métiers de la data nous a conduit à analyser un jeu de données représentant l'ensemble des réponses de plus de 20 000 individus à une quarantaine de questions, ceci dans le but d'élaborer un modèle de prédiction de postes data en fonction des compétences des individus.

Après exclusion des non-professionnels (étudiants, sans-emploi et autres), le panel de professions classifiées a été réduit de 10 à 5 (soit 80% du panel de répondants) afin de ne pas surcharger la modélisation.

Cinq modèles ont été retenus : Logistic Regression, SVM, Random Forest, Decision Tree et KNN , produisant des prédictions dont la précision (score test) variait entre 38 et 54%.

Une faible corrélation des variables entre elles ainsi qu'un sur apprentissage des modèles nous ont conduit à explorer différentes techniques (réduction de dimensions, ré échantillonnage, optimisation des hyper paramètres) dans le but d'améliorer la performance globale des modèles.

Cependant nos différents tests ne nous ont pas permis d'observer une amélioration notable des scores aussi bien dans le sur apprentissage que dans la précision globale.

Plusieurs hypothèses peuvent rentrer en ligne de compte pour expliquer ces scores :

- Le domaine de la Data Science est encore récent et en évolution, ne permettant pas une distinction nette de ses postes au vu des compétences et tâches quotidiennes.

- Des besoins de polyvalence et d'interchangeabilité peuvent aussi être recherchés par les entreprises, soit dans un but évolutif ou tout simplement car la taille de l'entreprise ne permet pas de recruter des spécialistes de chaque métier.

- On ne peut aussi exclure que le questionnaire, bien que fourni, aurait pu être enrichi de questions complémentaires qui auraient accentué les distinctions potentielles entre chaque métier.

Toutefois, nous retenons le modèle **Support Vector Machine** qui présente les meilleures performances relatives, avec pour paramètre kernel = 'linear' et une réduction de dimensions via SelectFromModel.

Il serait intéressant d'inclure les features sélectionnées par SVM dans un nouveau questionnaire que l'on soumettrait à l'échantillon de non professionnels, non retenus durant la modélisation. On ne retiendrait que les individus ayant répondu à toutes ces questions / features. Quels résultats obtiendrions-nous alors ?

V. Bibliographie

Pour appuyer notre réflexion, nous nous sommes inspirés des articles et vidéos suivants :

2020 Kaggle Machine Learning & Data Science Survey - kaggle.com

The most comprehensive dataset available on the state of ML and data science
https://www.kaggle.com/c/kaggle-survey-2020/overview

De l'avenir du métier de Data Scientist(e) - journaldunet.com
https://www.journaldunet.com/solutions/dsi/1506951-de-l-avenir-du-metier-de-data-scientist-e/

EXPLORATORY DATA ANALYSIS - CORRIGÉ (27/30) - YouTube | Machine Learnia
https://youtu.be/u64sWJEP4S0

Overfitting and Underfitting Principles - Medium
Understand basic principles of underfitting and overfitting and why you should use particular techniques to deal with them
https://towardsdatascience.com/overfitting-and-underfitting-principles-ea8964d9c45c

Feature Selection in Scikit-learn - Medium
Simple ways to filter out features for simpler and faster model
https://towardsdatascience.com/feature-selection-in-scikit-learn-dc005dcf38b7

Machine learning pour la classification automatique de musiques - La revue IA
Ilyes TALBI | Samir JEETO | Valentin DORE Mon attrait pour le machine learning vient du fait qu'il n'a quasiment aucune limites en termes d'applications. Dès que vous avez des données, vous pourrez faire des choses intéressantes avec. Et ceux quelque soit la nature de vos données. Images, vidéos, textes, sons, laissez vous guider par […]
Écrit par Ilyes Talbi
https://larevueia.fr/machine-learning-pour-la-classification-automatique-de-musiques-avec-python/

Weighted Logistic Regression for Imbalanced Dataset - Medium
Define custom weights in logistic regression to handle class imbalance in dataset.
https://towardsdatascience.com/weighted-logistic-regression-for-imbalanced-dataset-9a5cd88e68b

I Analyzed 2k Data Scientist and Data Engineer Jobs and This is What I Found - Medium
Leverage your Python Skills to Understand the Skills and Requirements of your Dream Job
https://pub.towardsai.net/i-analyzed-2k-data-scientist-and-data-engineer-jobs-and-this-is-what-i-found-1ed37f98a704

python — Comment effectuer la sélection des fonctionnalités avec gridsearchcv dans sklearn dans python - it-swarm-fr.com

https://www.it-swarm-fr.com/fr/python/comment-effectuer-la-selection-des-fonctionnalites-avec-gridsearchcv-dans-sklearn-dans-python/810367731/

FEATURE SELECTION avec SKLEARN (23/30) - YouTube | Machine Learnia
https://www.youtube.com/watch?v=T4nZDuakYlU

What is the difference between a Data Analyst and a Data Scientist? - Medium
There's a bit of a confusion as what the difference is between a data analyst and a data scientist.
https://georgefirican.medium.com/what-is-the-difference-between-a-data-analyst-and-a-data-scientist-946cd3aede76

SHAP: A reliable way to analyze your model interpretability - Medium
I had started this series of blogs on Explainable AI with 1st understanding what's the balance between accuracy vs interpretability, then…
https://towardsdatascience.com/shap-a-reliable-way-to-analyze-your-model-interpretability-874294d30af6

Explain Your Model with the SHAP Values - Medium
Use the SHAP Values to Explain Any Complex ML Model
https://towardsdatascience.com/explain-your-model-with-the-shap-values-bc36aac4de3d

https://shap.readthedocs.io/en/latest/index.html

Beware Default Random Forest Importances - explained.ai
Training a model that accurately predicts outcomes is great, but most of the time you don't just need predictions, you want to be able to interpret your model. The problem is that the scikit-learn Random Forest feature importance and R's default Random Forest feature importance strategies are biased. To get reliable results in Python, use permutation importance, provided here and in our rfpimp package (via pip). For R, use importance=T in the Random Forest constructor then type=1 in R's importance() function. (125 kB)
https://explained.ai/rf-importance/index.html

Understanding model predictions with LIME - Medium
In my previous post on model interpretability, I provided an overview of common techniques used to investigate machine learning models. In…
https://towardsdatascience.com/understanding-model-predictions-with-lime-a582fdff3a3b