Étude sur la prévision d'échecs de production

Garance Malnoë

Mai-Août 2023

Contents

Introduction

Ce document présente un projet de Machine Learning conçu dans le cadre du stage réalisé de Mai à Août 2023 à la DataFactory de Orange Rennes, encadré par Jérôme Pedro. J'ai travaillé sur le code et la production de ce dossier, Jérôme m'a aidée à trouver les sources de données et m'a conseillée tout au long du projet. Nous avons également rencontré de nombreuses personnes dans le cadre de ce projet pour mieux comprendre les attentes, savoir où trouver les données et comprendre l'organisation du réseau. Le projet propose un outil pour prévenir les échecs de production déjà en cours (i.e. où un technicien est déjà intervenu au moins une fois) à partir de l'analyse des commentaires des techniciens et de données techniques notamment la distance au point de branchement.

La première partie concerne le déroulement du stage, j'y ai pris en note les activités réalisées semaine par semaine. Cette partie n'a pas de lien direct avec le projet mais elle permet de voir plus clairement quelles ont été les plus grosses problématiques, ce qui a pris le plus de temps etc.

Dans la seconde partie, je présente les outils, les logiciels et les bibliothèques Python qui ont été utilisées dans ce projet.

Dans la troisième partie, on peut retrouver la présentation du choix des estimateurs.

La quatrième partie présente le travail effectué sur la prévision des échecs uniquement à partir des commentaires des techniciens. Cette partie comprend une description des données à notre disposition, des statistiques sur les commentaires qui permettent de mieux comprendre le dataset, une description étape par étape de la préparation du dataset, la modélisation avec 7 types de modèles et une analyse des différentes modélisations.

Dans la cinquième partie, j'expose l'intérêt de la prise en compte de la distance au point de branchement. Puis, je présente la deuxième vague de modèles qui prennent en compte ce nouveau paramètre.

Enfin, dans la dernière partie on s'intéresse également à d'autres données techniques (opérateur immeuble, le type de point de branchement et adductabilité) et on présente une troisième vague de modèles.

1 Suivi des semaines

Semaine 1

Durant les premiers jours, j'ai découvert les locaux et l'entreprise et Jérôme m'a présenté le sujet. Le reste de la semaine, j'ai travaillé sur le livre fourni par Jérôme : "Implémentation en Python avec Scikit-Learn" de Virginie Mathivet. J'ai essayé de rendre ma lecture la plus active possible en faisant des recherches supplémentaires (notamment pour l'aspect mathématique) et de synthétiser les idées dans un document LaTeX.

Semaine 2

J'ai continué à travailler sur le livre en prenant des notes sur LateX pour avoir une trace écrite de mon apprentissage et pouvoir m'y référer par le futur lorsque je travaillerai sur le sujet de stage. J'ai fini le livre le Mardi soir. Le Mercredi j'ai installé Jupyter et des package, cherché des livres sur le NLP et découvert le sujet avec une vidéo exemple de deux heures. Les jours suivants, j'ai retranscrit le contenu de la vidéo sur le document LaTeX. Enfin, le Vendredi je me suis formée sur la bibliothèque Pandas avec Orange Learning.

Semaine 3

J'ai continué à me former à la bibliothèque Pandas. J'ai appris à me servir de gitHub pour pouvoir sauvegarder mes fichiers de stage. Avec Jérôme, Alan Pillais et Phillipe Ménard nous avons fait plusieurs points pour bien délimiter le sujet du stage. Nous avons notamment mis de côté l'analyse de texte qui semble périlleuse au vu des textes, nous avons gardé le sujet principal qui est de prévenir les échecs de production mais en s'appuyant sur d'autres données. Mercredi et Jeudi, je me suis entraînée à travailler sur des cas d'arbre de décision et de régression linéaire.

Semaine 4

Absente cette semaine là, stage à Amiens.

Semaine 5

Finalement, nous sommes revenus à l'analyse de texte après que Jérôme en ait discuté avec d'autres personnes. En début de semaine j'ai essentiellement travaillé au nettoyage et à la compréhension des données du fichier ETI31. Puis j'ai travaillé toute la semaine sur la préparation des données (commentaires) sur Jupyter avec la bibliothèque pandas, sur la création d'un modèle performant et à la visualisation des résultats. J'ai également commencé à prendre en notes l'avancement sur ce fichier LaTeX, fait une réunion avec Sandrine Labarre, prévu une réunion avec Hervé Martiniere et remis en place le GitHub.

Semaine 6

J'ai commencé la semaine par l'amélioration des modèles en codant un algorithme permettant de trouver les meilleurs hyperparamètres et donc mis à jour la fiche des résultats et le rapport.

Le lundi j'ai eu plusieurs réunions avec des personnes du PPC d'Angers pour essayer de trouver des données intéressantes pour l'algorithme de prévisions. L'idée de faire le lien entre les échecs et la distance aux points de branchement (PB) est beaucoup revenue et donc j'ai travaillé sur ça tout le reste de la semaine. Les mardi, mercredi et jeudi j'ai travaillé sur comment trouver les coordonnées GPS des adresses des clients et les exploiter. Il a été difficile de trouver une API compatible avec la taille des données et la simplicité d'utilisation souhaitée. Finalement j'ai opté pour Nominatim pour le calcul de la distance et Jérôme m'a proposé PositionStack pour le calcul des coordonnées GPS. J'ai passé beaucoup de temps sur le nettoyage des données d'adresses pour les rendre utilisables.

Le vendredi je me suis occupé de la deuxième partie de cette tâche : essayer de faire le lien entre les numéros de désignation et les PB pour pouvoir calculer la distance. J'ai donc travaillé avec les tables disponibles sur GCP et fait du SQL avec BigQuery.

Semaine 7

Pendant le début de semaine j'étais bloquée pour avancer sur l'étude de la distance aux PB car je n'arrivait pas à faire le lien entre le numéro de désignation et l'adresse de PB. J'ai continué à me former à Python sur France IOI et j'ai réussi à faire fonctionner le proxy sur PythonCharm (IDE Python) donc j'ai transféré les 2 projets.

Puis j'ai implémenté de nouvelles fonctionnalités :

- Sélection de fichier et du dossier de sauvegarde des fichiers pickle.
- Fonction de préparation globale permettant de préparer à la prédiction n'importe quel fichier qui a les colonnes de commentaire de ETI31 et un numéro de désignation.
- Système d'erreurs pour éviter les plantages.
- Sauvegarde des modèles et de leurs statistiques.
- Fonction de prédiction (avec probabilités ou seuil et choix du type de modèle).
- Documentation pour chaque fonction.

Semaine 8

En début de semaine, Jérôme m'a indiqué un moyen de croiser les données pour obtenir les adresses des points de branchement. J'ai donc travaillé sur ces données (croisement dans GCP, vérification...) et refais une modélisation sur Python.

Nous avons fait plusieurs réunions avec Katia Louis et Jérome Le Hénaff qui nous ont indiqué des pistes pour récupérer encore d'avantage de données sur les adresses des PB.

Le mardi après-midi nous sommes allés à Saint-Jacques-de-la-Lande pour visiter un centre de formation technicien avec Romane, Maya et Jérôme.

Pendant le reste de la semaine j'ai travaillé sur le code pour faire le nettoyage, la préparation et la modélisation des données sur Python. J'ai également avancé sur le document LateX en m'inspirant d'un mémoire fait sur un autre projet de Machine Learning.

Semaine 9

J'ai poursuivi la modélisation pour la deuxième vague de modèles (prise en compte de la distance au point de branchement) ainsi que les fonctions de prédiction.

Puis j'ai fait le croisement des données fournies par Katia Louis (informations sur les PB) et Jérôme (archives ETI31). Dans l'optique de la création de modèle, j'ai fait des statistiques sur les données récupérées.

Durant cette semaine j'ai également travaillé sur le document LaTeX pour corriger le texte, ajouter les sections pour les nouveaux modèles et reprendre les sections précédentes.

Semaine 10

Durant le début de semaine, j'ai continué à travailler sur le document LaTeX.

Code sur Python pour traiter les informations sur le PB et pouvoir faire la modélisation. Création de nouveaux modèles. Statistiques.

Semaine 11

Semaine 12

Semaine 13

2 Outils, logiciels et bibliothèques Python

La plupart des projets de Machine Learning sont fait avec le langage Python, ce projet n'y fait pas exception. Au début du projet, j'ai fait les choix de travailler sur des notebooks Jupyter notamment à cause des soucis de proxy sur ma machine mais aussi par convention et la praticité en terme de visualisation des dataframes. Après avoir réglé les problèmes de proxy, le projet a migré sur l'IDE PyCharm (Licence gratuite de JetBrains) pour plus de simplicité d'écriture de code et de navigation au sein du projet.

Pour ce projet, de nombreuses bibliothèques Python ont été utilisées :

- Visualisation : Seaborn, Matplotlib.pyplot
- Manipulation des données : Numpy, Pandas, Textblob (analyse de texte)
- Création des modèles et prédictions : Scikit-learn (sklearn)
- Interaction avec l'ordinateur : warnings, re, pickle, tkinter, os
- Interaction avec internet : requests, socket
- Coordonnées GPS et calcul de distance : Geopy

Pour retrouver et manipuler les données d'Orange le projet utilise principalement Google Cloud Platform (GCP) avec BigQuery pour faire des requêtes en SQL sur des données massives. Les données manipulées sont toutes dans le répertoire ofr-bli-lab-prd/SPML0410 qui contient des imports de la base de données IPON.

Pour le suivi du projet, un Notion dédié a été mis en place pour regrouper les nots prises durant les réunions, les pistes d'exploration et les todo-listes. Quant à la sauvegarde du projet, un dépot git a été créé à cette adresse avec une seule branche. La méthodologie CRISP-DM de suivi et de mise en place du projet est celle présentée dans le livre "Machine Learning Implémentation en Python avec Scikit-learn" écrit par Virginie MATHIVET et lu en début de stage. J'ai essayé d'appliquer au mieux les étapes de Business Understanding, Data Understanding, Data Preparation, Modelling, Evaluation et Déploiement .

Il est clair que le code pourrait grandement être amélioré : il est très probablement bourré de mauvais réflexes et d'erreur de débutant même si des efforts ont été fait pour le documenter, le rendre lisible et éviter la redondance de code. Si le projet est repris, il faudra sans doute remettre au propre le code.

Enfin, pour les coordonnées GPS et le calcul de la distance plusieurs API ont été envisagées. Finalement Geopy a été gardée pour le calcul de la distance et PositionStack pour retrouver les coordonnées GPS car limitée à 25,000 requêtes par mois et 1requête/seconde et simple d'utilisation.

3 Choix des estimateurs

Les principaux estimateurs pour le machine learning et la création de modèles sont les suivants :

- L'accuracy : proportion de prédictions correctes (VP + VN /(VP+VN+FN+FP)).
- Le rappel : proportion de réussites correctement prédites sur l'ensemble des réussites (VP/(VP+FN))
- La précision : proportion de véritables réussites parmi les réussites prédites (VP/(VP+FP))
- La spécificité : proportion d'échecs correctement prédits sur l'ensemble des échecs (VN/(VN+FP))
- La valeur prédictive négative : proportion de véritables échecs parmi les échecs prédits (VN/(VN+FN))

Ici, on cherche à prédire des échecs de production qui sont en plus faible proportion donc il faut surtout vérifier que la spécificité et la valeur prédictive négative sont proches de 1.

Si la valeur prédictive négative n'est pas bonne alors trop de données seraient considérées comme des échecs.

Si la spécificité n'est pas bonne alors à l'inverse pas assez d'échecs seraient prédits comme tels.

Après discussions avec des personnes du PPC et d'après les retours du premier projet de prédiction des erreurs, il est plus pertinent de créer un outil qui ne prédit pas toutes les erreurs mais qui ne n'a que très peu de faux échecs. Il faut donc plutôt maximiser la valeur prédictive négative dans un premier temps puis la spécificité dans un second temps.

4 Prévision échecs de production avec les commentaires

Dans cette section est décrit le premier essai de modélisation se basant uniquement sur l'analyse des commentaires des techniciens. On y présente la préparation des données, la modélisation, les choix effectués et les résultats des modèles créés.

4.1 Les données

Le code pour cette partie se trouve dans le dossier "Ancien" et notamment dans les fichier Jupyter.

Les données à notre disposition se trouvent dans le fichier ETI31 qui contient 6 colonnes qui nous intéressent. Deux servent pour l'identification et la labélisation : No DESIGNATION et CODE RELEVE. Quatre contenant des commentaires de techniciens : COMMENTAIRE DE SIG, COMMENTAIRE, BLOC NOTE, COMMLITIGEPIDI.

4.2 Préparation des commentaires et statistiques

Première préparation des données

La table ETI31 étant très volumineuse (32695 lignes et 175 colonnes), un premier traitement a été effectué sur Excel directement plutôt qu'avec Jupyter et Pandas car la tableau est beaucoup trop lourd pour être chargé en un temps acceptable. J'ai gardé uniquement les lignes concernant les production sur de la fibre (IQ*FTH). La suite du traitement se fait ensuite sur le notebook Jupyter avec la bibliothèque pandas et consiste à enlever les lignes qui ne sont pas exploitables : pas de code de relève ou aucun commentaire.

L'étape suivante est la labélisation des données dans une colonne "Reussite": passer des codes de relève à l'indication d'un échec ou d'une réussite. On peut alors enlever la colonne CODE DE RELEVE qui ne nous servira plus.

Réussite : RRC, DMS, TVC
Échec: ANC, ANN, ETU, MAJ, ORT, PAD, PBC, REO, RMC, RMF, RRC

En suite, on nettoie les textes :

- Enlever les codes de relève.
- Enlever la ponctuation.
- Remplacement des abréviations et des acronymes par le.s mot.s correspondant.s.
- Tout mettre en minuscule.
- Enlever les nombre.

Puis on termine la préparation avec la transformation des colonnes de commentaire en une seule colonne contenant une liste de commentaires. Sur une cellule de commentaire, il peut y avoir plusieurs commentaires qui sont séparés par un anti-slash. On utilise la fonction split de Python pour séparer les textes en fonction de ce séparateur et on l'applique à toutes les colonnes contenant du texte. Il est important de noter que pour certaines interventions, il n'y a pas de commentaires ou seulement des slashs.

Après cette première phase de nettoyage on se retrouve donc avec un dataframe composé de 3 colonnes : le numéro de désignation, le label de réussite et la liste des commentaires.

	No DESIGNATION	Reussite	COMMENTAIRES
0	299568158	False	[obtenu marchand christophe le a le clien...
1	297892193	True	[point de terminaison optique existante malf...
2	297897767	True	[point de terminaison optique non existante ...
3	223370510	True	[point de terminaison optique non existante ...
4	299556603	True	[point de terminaison optique existante malf...
...	...	...	...
4183	223275005	True	[merci de clrer le rendez vous du client car i...
4184	296611510	True	[point de terminaison optique existante malf...
4185	223309738	True	[point de terminaison optique non existante ...
4186	298170718	True	[point de terminaison optique non existante ...
4187	298170734	True	[point de terminaison optique non existante ...

4188 rows × 3 columns

Associations des mots à une réussite ou à un échec

Dans cette partie, on cherche à déterminer si certains mots sont plus souvent utilisés dans les échecs ou les réussites. Pour cela on crée une base de données avec tous les mots qui apparaissent dans l'ensemble des commentaires et le nombre de fois qu'ils apparaissent dans des réussites et le nombre de fois où ils apparaissent dans des échecs. Cependant, les réussites étant bien plus nombreuses que les échecs, on s'est plutôt intéressé à la proportion d'apparition du mot parmi l'ensemble des réussites ou des échecs plutôt qu'au nombre d'apparition.

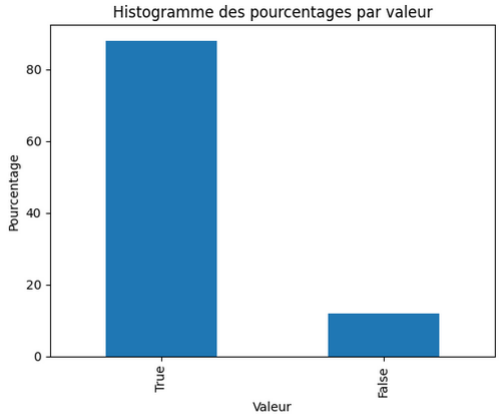
On a fait le choix de ne garder que ceux où la différence d'apparition est supérieure à 30% mais peut-être qu'un autre seuil serait plus pertinent. Un seuil à 10% a été testé mais cela n'apportait pas de résultats significatifs. Une partie du tableau que l'on obtient :

	Reussite	clt	existante	laisse	message	na	ok	optique	plp	point	pose	rdv	terminaison	vocal	éligible
0	True	0.1069	0.8741	0.2035	0.2016	0.1696	1.7784	1.7352	0.3394	1.8268	0.3264	0.6305	1.6877	0.1959	0.0963
1	False	0.3705	0.0000	0.5697	0.5757	0.6474	0.0259	0.5976	0.0239	0.8805	0.0378	1.1235	0.5677	0.5359	0.3845

En suite, à partir de ce tableau on peut produire une fonction qui donnera à chaque liste de commentaires un score de mots d'échec et un score de mots de réussite. Plus les commentaires contient de mots qui apparaissent souvent dans les échecs ou les réussites plus ce score sera haut. Il pourra ensuite être utilisé comme variable pour le Machine Learning.

Statistiques et nouvelles données

Cette partie porte sur les statistiques des commentaires pour utiliser ces informations comme variable pour le Machine Learning et mieux comprendre le jeu de données.

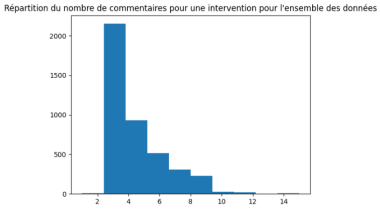


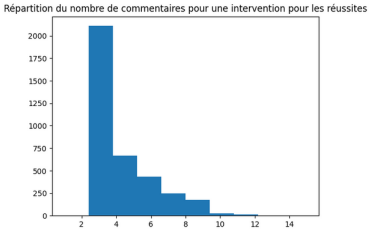

```
Reussite
True     88.013372
False    11.986628
```

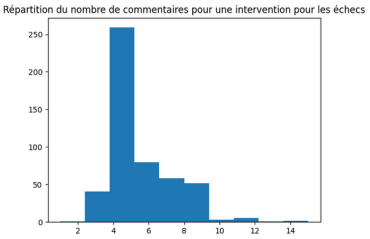
Dans cette partie j'ai fait la distinction entre le dataframe de l'ensemble, le dataframe des réussites et le dataframe des échecs.

On regarder 2 statistiques différentes : le nombre de commentaires et la longueur moyenne de chaque commentaire pour une intervention.

<u>Nombre de commentaires</u>
Pour l'ensemble la moyenne est de 4.38, pour les échecs elle est de 5.38 et pour les réussites elle est de 4.25.

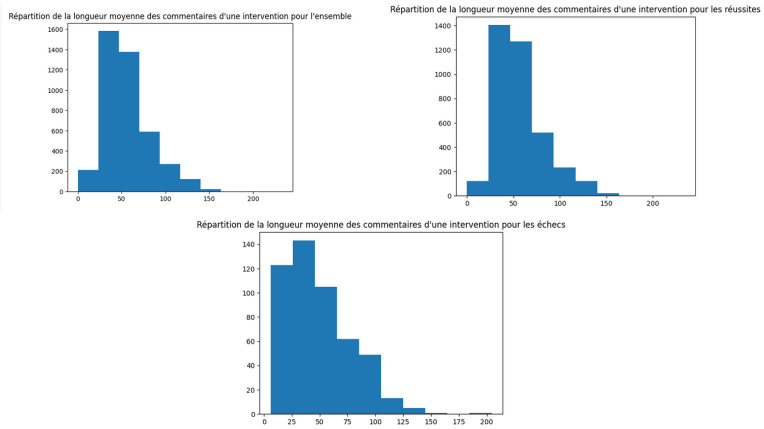






Longueur moyenne

Pour l'ensemble la longueur moyenne d'un commentaire est de 56.1 caractères, contre 49.6 pour les échecs et 57 pour les réussites.



On peut observer quelques différences entre les échecs et les réussites. Les échecs ont souvent plus de commentaires mais ils sont plus courts. Ces 2 paramètres pourront sans doute être utile pour la création d'un modèle : on ajoute une colonne longueur moyenne commentaire et une colonne nombre_commentaire.

Normalisation des données

A fin d'éviter que la longueur moyenne (réel autour de 56 caractères) n'ait artificiellement plus d'influence, on applique une normalisation aux données.

	nombre_commentaires	Longueur moyenne commentaire	No DESIGNATION	Reussite	COMMENTAIRES
0	-0.755824	-0.328797	299568158	False	[obtenu marchand christophe le a le clien...
1	0.333062	-0.468411	297892193	True	[point de terminaison optique existante malf...
2	1.966390	1.752941	297897767	True	[point de terminaison optique non existante ...
3	-0.755824	-0.627969	223370510	True	[point de terminaison optique non existante ...
4	-0.755824	-0.877279	299556603	True	[point de terminaison optique existante malf...
...	...	...	...	...	...
4183	1.966390	-0.359961	223275005	True	[merci de clrer le rendez vous du client car i...
4184	-0.755824	-0.665366	296611510	True	[point de terminaison optique existante malf...
4185	-0.755824	-0.627969	223309738	True	[point de terminaison optique non existante ...
4186	0.333062	-0.722707	298170718	True	[point de terminaison optique non existante ...
4187	-0.755824	-0.927141	298170734	True	[point de terminaison optique non existante ...

Polarité et subjectivité des commentaires

Cette partie du tratiement permet de déterminer si les mots utilisés dans les commentaires ont une polarité négative ou positive et s'ils sont subjectifs. On peut supposer que les échecs seront plus polarisés vers le négatif que les réussites et que cela permettra de les différencier dans notre modèle. On crée donc 2 nouvelles colonnes : polarite et subjectivite qui sont remplies en appliquant les fonction polarity et subjectivity de la bibliothèque TextBlob qui est spécialisée dans l'analyse de texte.

On trouve finalement les résultats suivants :

	nombre_commentaires	Longueur moyenne commentaire	Reussite	polarite	subjectivite
0	-0.755824	-0.328797	False	0.000000	0.000000
1	0.333062	-0.468411	True	0.100000	0.100000
2	1.966390	1.752941	True	0.062500	0.062500
3	-0.755824	-0.627969	True	0.133333	0.158333
4	-0.755824	-0.877279	True	0.166667	0.166667
...	...	...	...	...	...
4183	1.966390	-0.359961	True	0.000000	0.000000
4184	-0.755824	-0.665366	True	0.166667	0.166667
4185	-0.755824	-0.627969	True	0.166667	0.166667
4186	0.333062	-0.722707	True	0.100000	0.100000
4187	-0.755824	-0.927141	True	0.166667	0.166667

Dataframe	Polarité	Subjectivité
Ensemble	0,1	0,13
Réussites	0,12	0,14
Echecs	-0,002	0,04

Maintenant que nos données sont mises au propre et normalisées, on a tout ce qu'il faut pour pouvoir passer à la modélisation.

4.3 Modélisation

Arbre de décision classiques

Pour une première approche de modélisation, on fait une modélisation par un arbre de décision. On utilise le module DecisionTreeClassifier du module scikit-learn.

On a commencé par une premier arbre de profondeur maximale 5 et voici les résultats que l'on obtient :
Accuracy : proportion de prédictions correctes (VP + VN /(VP+VN+FN+FP)) : 92.124%
Rappel : proportion de réussites correctement prédites sur l'ensemble des réussites (VP/(VP+FN)) : 96.685%
Précision : proportion de véritables réussites parmi les réussites prédites (VP/(VP+FP)) : 94.34%
Spécificité : proportion d'échecs correctement prédits sur l'ensemble des échecs (VN/(VN+FP)) : 63.158%
Valeur prédictive négative: proportion de véritables échecs parmi les échecs prédits (VN/(VN+FN)) : 75.0%

Ce qui n'est pas si mauvais pour un premier essai mais la spécificité et la VPN sont assez bas. On peut probablement faire mieux.

Pour essayer d'améliorer l'arbre de décision, on applique un algorithme qui permet de trouver les meilleurs hyperparamètres selon la VPN. Il consiste a créer x fois un nouveau découpage du dataframe (entrainement et test) et à chaque découpage trouver les meilleurs paramètres parmi une liste donnée en argument (en gardant ceux donnant la meilleur VPN). Finalement après les x itérations, on retient les hyper-paramètres qui apparaissent le plus souvent.

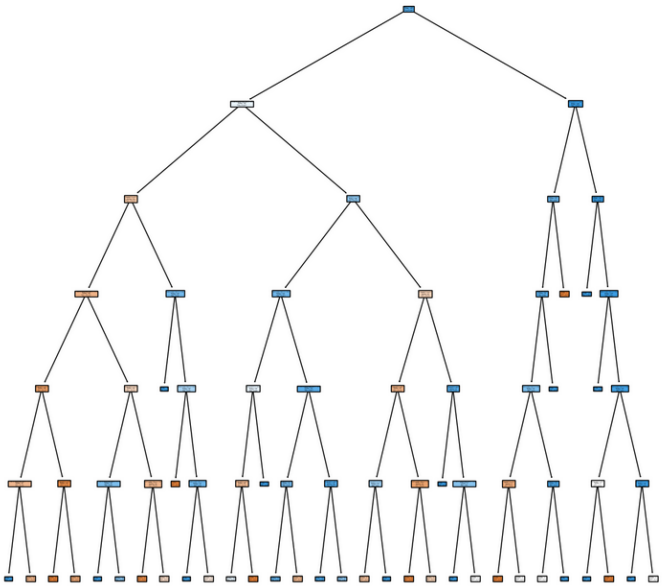
```
def BestHyperparameters_vpn
    (trainX,trainY,testX,testY,min_example,depth):
    best_vpn = 0
    best_min_example = 0
    best_depth = 0
    best_criterion = ""
    #On va tester toutes les combinaisons possibles :
    for criterion in ["gini","entropy"]:
        for example in min_example:
            for profondeur in depth:
                #On creer l'arbre
                arbre = DecisionTreeClassifier(criterion=criterion,
                min_samples_leaf=example,
                max_depth=profondeur)
                arbre.fit(trainX, trainY)
                #On calcule la prediction et la vpn associee
                predictionY = arbre.predict(testX)
                vpn =
                    round(sk.metrics.precision_score(testY,predictionY,pos_label=0),5)*100
                #On compare
                if vpn>best_vpn :
                    best_vpn = vpn
                    best_min_example = example
```

```
                best_depth = profondeur
                best_criterion= criterion

    return (best_vpn,best_min_example,best_depth,best_criterion)

def Finetune(nombre_essai,min_example,depth):
    l_vpn = []
    l_best_min_example = []
    l_best_depth = []
    l_best_criterion = []

    for i in range (0,nombre_essai) :
        if i%10==0 :
            print(i)
        trainX,trainY,testX,testY = creation_test_train(df2,"Reussite")
        best_vpn,best_min_example,best_depth,best_criterion =
            BestHyperparameters_vpn(trainX,trainY,testX,testY,min_example,depth)
        l_vpn.append(best_vpn)
        l_best_min_example.append(best_min_example)
        l_best_depth.append(best_depth)
        l_best_criterion.append(best_criterion)
    return mean(l_vpn),
    Counter(l_best_min_example).most_common(1)[0][0],
    Counter(l_best_depth).most_common(1)[0][0],
    Counter(l_best_criterion).most_common(1)[0][0]

Finetune(500,[2,3,4,5],[2,3,4,5,6])
```

En choisissant d'itérer 500 fois comme sur la dernière ligne on trouve finalement une valeure prédictive négative moyenne de 72.6 et que les meilleurs paramètres sont :

- min_example = 2
- max_depth = 6
- criterion = "entropy"

En appliquant le même algorithme mais cette fois la spécificité comme moyen de comparaison on trouve les mêmes paramètres.



Voici les mesures pour cet arbre :
Accuracy : proportion de prédictions correctes (VP + VN /(VP+VN+FN+FP)) : 93%
Rappel : proportion de réussites correctement prédites sur l'ensemble des réussites (VP/(VP+FN)) : 95%
Précision : proportion de véritables réussites parmi les réussites prédites (VP/(VP+FP)) : 97%
Spécificité : proportion d'échecs correctement prédits sur l'ensemble des échecs (VN/(VN+FP)) : 81%
Valeur prédictive négative: proportion de véritables échecs parmi les échecs prédits (VN/(VN+FN)) : 69%

Dans le but de maximiser la valeur prédictive négative, on a mis en place un seuil plus haut pour qu'une potentielle erreur soit labélisée comme telle. On utilise la fonction predict_proba pour obtenir la probabilité qu'une donnée soit de chaque label.

Le soucis c'est qu'en fonction du dataset d'entraînement, les pourcentages peuvent énormément varier même si on garde le même seuil. Avec à chaque fois un seuil à 80% au début puis la meilleure valeur prédictive négative possible à obtenir sans avoir une spécificité à 0%

- 0.8 → Spécificité : 47% et VPN à 90%
  1 → Spécificité : 17% et VPN à 94%
- 0.8 → Spécificité : 58% et VPN à 87%

1 → Spécificité : 15% et VPN à 93%

- 0.8 → Spécificité : 44% et VPN à 96%
  1 → Spécificité : 26% et VPN à 96.5%

Il faudra donc bien veiller à choisir le bon dataset d'entraînement et de choisir le bon seuil à appliquer.

Random Forests

Ce modèle est similaire à celui des arbres de décisions mais applique une couche supplémentaire qui permet d'obtenir de meilleurs résultats mais rend le programme est plus complexe donc plus lent. De plus, il est possible qu'il y ait un sur-apprentissage qui sera peut-être à tester sur d'autres données. Là aussi on utilise la bibliothèque scikit-learn pour créer le modèle.

Le premier modèle donne de meilleurs résultats qu'avec les arbres de décisions simples :
- Accuracy (VP + VN /(VP+VN+FN+FP)) : 95%
- Rappel (VP/(VP+FN)) : 98%
- Précision (VP/(VP+FP)) : 96%
- Spécificité (VN/(VN+FP)) : 80%
- Valeur prédictive négative (VN/(VN+FN)) : 86%

En ajoutant un seuil à 0.8, les résultats deviennent vraiment bons au prix de la spécificité et sont surtout beaucoup plus "stables" qu'avec les arbres normaux. La VPN reste quasiment toujours entre 95% et 100%:

- Spécificité : 25% et VPN à 95%
- Spécificité : 28% et VPN à 96%
- Spécificité : 24% et VPN à 100%
- Spécificité : 23% et VPN à 95%
- Spécificité : 25% et VPN à 100%
  ...

En fin, le test des hyper-paramètres nous indique des résultats égaux à ceux des arbres de décision normaux.

K-nearest neighbors

On s'est ensuite intéressé à la méthode des k plus proches voisins.

Le premier modèle donne des résultats moins bons que les modèles précédents :
- Accuracy (VP + VN /(VP+VN+FN+FP)) : 94%
- Rappel (VP/(VP+FN)) : 97%

- Précision (VP/(VP+FP)) : 96%
- Spécificité (VN/(VN+FP)) : 70%
- Valeur prédictive négative (VN/(VN+FN)) : 76%

En ajoutant un seuil à 0.9, les résultats deviennent vraiment bons. La VPN est un petit peu moins bonne mais reste très bonne et la spécificité est meilleurs. C'est peut-être un entre-deux qui pourrait convenir :

- Spécificité : 28% et VPN à 96%
- Spécificité : 33% et VPN à 94%
- Spécificité : 32% et VPN à 96%
- Spécificité : 39% et VPN à 97%
- Spécificité : 42% et VPN à 93%
  ...

Enfin, comme pour la modélisation avec des arbres de décision on applique un algorithme permettant de trouver les meilleurs hyperparamètres pour la modélisation avec seuil : le nombre de voisins utilisés pour la classification, le type de poids (distance ou uniforme) et la mesure utilisée (manhattan ou euclidienne). On trouve finalement que les meilleurs hyperparamètres sont : 5 voisins, poids uniforme et distance de Manhattan avec une valeur prédictive négative de 97.2% en moyenne.

Régression logistique

Le type de modèle étudié en suite a été celui de régression logistique qui sont également proposés par la bibliothèque scikit-learn.

Il n'y a pas de probabilités pour ce type de modèle. Voici les résultats obtenus pour différents dataset d'entraînement :

- Spécificité : 39% et VPN à 73%
- Spécificité : 42% et VPN à 61%
- Spécificité : 36% et VPN à 62%
- Spécificité : 40% et VPN à 55%
- Spécificité : 36% et VPN à 59%
  ...

La VPN et la spécificité sont assez faibles, ce type de modèle n'est pas adapté pour notre projet.

SVM

Le type de modèle suivant a été celui de Support Vector Machine (SVM) qui sont également proposés par la bibliothèque scikit-learn.

Il n'y a pas de probabilités pour ce type de modèle. Voici les résultats obtenus pour différents dataset d'entraînement :

- Spécificité : 45% et VPN à 64%
- Spécificité : 49% et VPN à 67%
- Spécificité : 51% et VPN à 64%
- Spécificité : 54% et VPN à 66%
- Spécificité : 55% et VPN à 64%
  ...

La VPN et la spécificité sont assez faibles, ce type de modèle ne semble pas très adapté pour notre projet.

Naive Bayes

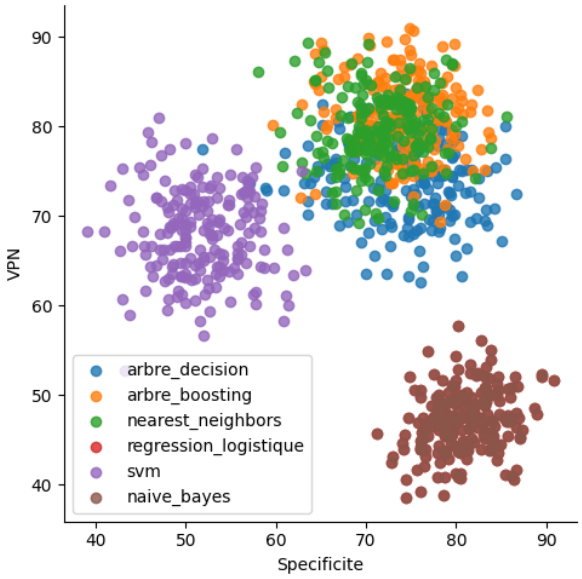
Pour ce dernier type de modèle on utilise l'approche Naive Bayes Gaussiennes. Voici un échantillon des résultats obtenus :

- Spécificité : 82% et VPN à 46%
- Spécificité : 78% et VPN à 46%
- Spécificité : 76% et VPN à 43%
- Spécificité : 80% et VPN à 42%
- Spécificité : 79% et VPN à 53%
  ...

4.4 Résultats de modélisation

L'objectif est de déterminer quels sont les modèles les plus pertinents mais aussi les risques qui leur sont associés. Pour visualiser facilement la spécificité et la valeur prédictive négative on utilise un scatter plot où un point représente un modèle.

VPN en fonction de la spécificité pour des modèles de différents types



On recherche le type de modèle donnant les meilleurs spécificités et VPN : les modèles basés sur des régressions logistiques et de de naive bayes ont une VPN notablement plus faible que les autres types de modèles, ils ne seront pas retenus. Les modèles de SVM ne sont pas retenus non plus car ils ont une spécificité plus faible et une VPN légèrement moins bonne. Enfin, les modèles basés sur les arbre de décision et les algortihmes de nearest neighbors sont un peu moins bons en moyenne que les forêts aléatoires. On retiendra donc ce dernier type de modèle.

Les statistiques des modèles qui ont été sauvegardés dans les fichiers joblib :

Type de modèle	Arbre de décision	Forêt aléatoire	Nearest neighbors	Régression logistique	SVM	Naive Bayes
Accuracy	95,1	96,2	95	89,9	92,2	88,7
Rappel	97	97,9	97,6	90,5	96,9	89,1
Spécéficité	82,89	83	75,2	85	57,8	85,4
Valeur prédéctive négative	81,5	83	80,2	54,5	71,9	52

La modélisation basée uniquement sur les commentaires s'est arrêtée à cette étape car nous nous sommes intéressés à une nouvelle variable. Mais cette première analyse a été très utile pour bien me former aux outils de ML et beaucoup de lignes de codes ont été réutilisées dans la suite.

5 Prévisions des échecs de production avec la distance aux points de branchement

Après plusieurs réunions avec des personnes issues des équipes PPC entre autres, il est ressorti que la distance au point de branchement et le type de branchement était souvent source d'erreur ou de rendez-vous multiples inutiles (ex : une ligne aérienne pour laquelle il faut venir en nacelle mais où aucune indication n'est faite donc plusieurs rendez-vous ont lieu avant que des techniciens viennent avec l'équipement nécessaire). Il a donc été décidé de s'intéresser à la distance entre le PTO et le PB.

5.1 La préparation des données

Obtenir les données

Pour s'intéresser à la distance aux points de branchement, nous avions besoin de l'adresse du client et de l'adresse du PB qui lui est associé.
L'adresse du client a été simple à obtenir car directement présente dans le fichier ETI31 mais les adresses aux points de branchement ont été beaucoup plus difficiles à trouver avec souvent des données manquantes ou une absence de pivot pour pouvoir faire le lien avec nos données labélisées.

Néanmoins, à partir de de données se trouvant dans IPON ensuite importées sur GCP (adresses edrponoc, adresses edrponoi, adresses_pf_oi_t...) il a été possible de croiser ces données avec les données de la table ETI31. Cependant, sur les 5333 lignes de ETI31 seulement 501 donnaient un résultat lors du croisement, ce qui semble peu pour pouvoir produire une modélisation suffisamment générale.

Préparation des commentaires

Cette partie a été faite sur le fichier python *traitement_adresses.py* et est très similaire à la préparation des données qui a été faite dans la partie précédente. On réutilise les bibliothèques tkinter, textblob, matplotlib et pandas.

La préparation des données commence par la vérification des colonnes présentent dans les données sélectionnées par l'utilisateur ou passées en argument. Si certaines colonnes ne sont pas des colonnes de commentaires (COMMENTAIRE_DE_SIG, COMMENTAIRE, BLOC_NOTE, COMMLITIGEPIDI, COMMENTCONTACT) elles sont enlevées du dataset à l'exception de la colonne avec le numéro de désignation et les colonnes manquantes sont ajoutées avec pour donnée un string vide : "". En suite, le dataframe est nettoyé des lignes qui n'ont pas de données pour toutes les colonnes. Puis, les commentaires sont nettoyés pour enlever code de relève, tout mettre en minuscule etc (c'est le même nettoyage que précisé dans la section précédente). Enfin, on récupère les différentes statistiques sur les commentaires : longueur moyenne des mots, nombre de commentaires, polarité, subjectivité, score d'échec et score de réussite.

La préparation se termine avec la sauvegarde du dataframe dans un fichier Excel sous le nom *df_stats_commentaires* dans le dossier sélectionné par l'utilisateur.

Préparations des données d'adresse

Pour cette partie on commence également par ne garder que les colonnes nécessaires (adresses_client, adresse_pb) au traitement. Puis on calcule les coordonnées GPS de l'adresse client et du PB à partir de l'API Position Stack. puis on calcule la distance entre les 2 coordonnées avec l'API Nominatim de GeoPy. Les 2 API ont une limitation de requête à la seconde ce qui ralenti fortement le processus de préparation des données d'adresse.

Le résultats est finalement sauvegardé dans le dossier sélectionné par l'utilisateur dans le fichier "df_distance_adresse.xlsx"

Mutualisation des traitement

Un troisième fichier python permet de rassembler l'ensemble des traitement. La fonction principale permet de sélectionner via l'explorateur de fichier le dataframe à traiter et un dossier où sauvegarder les données. Le traitement des commentaires et des adresses est ensuite appliqué au dataframe sélectionné puis une normalisation (basées sur les données de ETI31) est appliquée pour éviter que certaines variables est une importance artificielle liée aux valeurs qu'elle peut prendre.

Finalement, le dataframe entièrement préparé est sauvegardé sous le nom "df_prepare.xlsx" dans le dossier sélectionné au préalable par l'utilisateur.

	iar_ndfictif	COMMLITIGEPIDI	COMMENTCONTACT	BLOC_NOTE	COMMENTAIRE	COMMENTAIRE_DE_SIG	adresse_client	adresse_pb	Reussite
0	296613031	\\Reprise rdv pour mercredi 8h manque de temps \	\GM Un rdv a ete repris par le technicien p...	NaN	/DMS/#PTO_non existante#/C cble passer en aeri...	NaN	Kerguen 22065 GOUDELIN	3, RUE DE KERNILIEN, 22065, GOUDELIN	True
1	296614596	\GM Obtenu client au 06a le 06/05 a r...	\GM Obtenu client au 06a le 06/05 a r...	NaN	/DMS/#PTO_non existante#/50m de cable en facad...	NaN	2 R . Forge 22065 GOUDELIN	3, RUE DE KERNILIEN, 22065, GOUDELIN	True
2	296506924	\$gm$obtenu client le 17/04 rdv repris le 22/...	\$gm$obtenu client le 17/04 rdv repris le 22/...	NaN	/DMS/#PTO_non existante#/PB 19 tube rouge fibr...	NaN	3 R . Abbé Maurice Barre 22093 LAMBALLE ARMOR	5, RUE DE LA CROIX BLANCHE, 22093, LAMBALLE	True
3	296941662	\gm obtenu cliente sur son tel le 03/05 rdv...	\gm obtenu cliente sur son tel le 03/05 rdv...	NaN	/DMS/#PTO_non existante#/Pose PTO Tire 100m de...	NaN	18 . Béchepée 22059 LE FOEIL	LIEU DIT KERGOMAU, 22059, LE FOEIL	True
4	296011281	\GM CP Laisse mevo pour reprendre nouveau R...	\\Laisse mevo pour reprendre nouveau RDV des T...	NaN	Apres plusieurs relances pas de retour client ...	NaN	5 . Le Beaudoué 22059 LE FOEIL	LIEU DIT KERGOMAU, 22059, LE FOEIL	True
...	...	...	...	...	...	...	...	...	...
495	297891012	\\VGEM/#LITIGE: CP-Travaux sur conduite privat...	\GM Laisse mevo au 0630535631le 22/4 a po...	NaN	/DMS/#PTO_non existante#/Pose jrt au pm + tira...	NaN	5 RTE. kerplat 56161 PLOEMEL	29, B, POUL HO, 56161, PLOEMEL	True
496	297241892	\GM TH Un rdv a ete repris par le technicie...	\\Un rdv a ete repris par le technicien pour l...	NaN	/DMS/#PTO_non existante#/Zahir ok 100m Aero st...	NaN	86 . Locmiquel 56161 PLOEMEL	29, B, POUL HO, 56161, PLOEMEL	True
497	297051994	\\Besoin d'un arreter de circulation pour effe...	\GM Laisse mevo au 0685211808 le 19/4 a ...	NaN	/DMS/#PTO_non existante#/Pose jarretiere pm+tl...	NaN	2 RTE. Pont Fol 56161 PLOEMEL	29, B, POUL HO, 56161, PLOEMEL	True
498	297894162	\gm cp Un rdv a ete repris par le technici...	\\ Un rdv a ete repris par le technicien pour ...	NaN	Suite a plusieurs relances aucun retour de la ...	NaN	4 ALL. Eau 56161 PLOEMEL	29, B, POUL HO, 56161, PLOEMEL	True
499	297687302	\GM Laisse mevo au 0633155590 le 26/4 a po...	\GM Laisse mevo au 0633155590 le 26/4 a po...	NaN	/MAJ/#PTO_non existante#/1 FO AERIEN TEST DELC...	NaN	0 . Croix de Kerno 56087 ILE AUX MOINES	0, CHEMIN DE LA CORNICHE, 56087, ÎLE-AUX-MOINES	True

Figure 1. ETI31 avant le traitement

	iar_ndfictif	score_reussite	score_echec	nombre_commentaires	longueur_moyenne	polarite	subjectivite	distance	Reussite
0	296613031	13.1875	9.3077	9	30.666667	0.066667	0.100000	221.82	True
1	296614596	21.3705	17.4615	9	38.000000	0.066667	0.100000	221.82	True
2	296506924	29.3238	25.7882	17	36.117647	0.029412	0.029412	4580.64	True
3	296941662	30.9935	24.1150	13	26.307692	0.038462	0.038462	0.00	True
4	296011281	25.5221	19.7118	13	47.076923	0.000000	0.000000	0.00	True
...	...	...	...	...	...	...	...	...	...
495	297891012	18.8862	12.6539	13	41.923077	0.038462	0.038462	2136.88	True
496	297241892	19.5804	17.5770	11	42.909091	0.030303	0.030303	506.28	True
497	297051994	22.5001	17.9807	13	43.692308	0.038462	0.038462	1227.42	True
498	297894162	25.3125	23.8272	13	66.076923	0.000000	0.000000	1227.42	True
499	297687302	32.8103	25.4229	13	56.692308	0.038462	0.038462	0.00	True

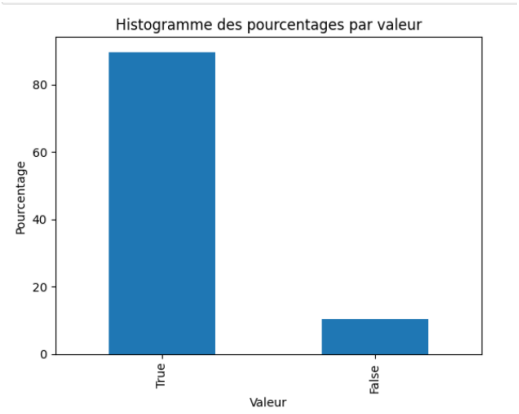
Figure 2. ETI31 avant la normalisation

	score_reussite	score_echec	nombre_commentaires	longueur_moyenne	polarite	subjectivite	distance	Reussite	iar_ndfictif
0	-0.816269	-0.916998	-0.945463	-1.072344	1.166285	1.086802	-0.792283	True	296613031
1	-0.063711	-0.027270	-0.945463	-0.550920	1.166285	1.086802	-0.792283	True	296614596
2	0.667723	0.881324	1.783146	-0.684762	0.166608	-0.372113	2.429000	True	296506924
3	0.821278	0.698747	0.418842	-1.382281	0.409445	-0.185073	-0.956214	True	296941662
4	0.318095	0.218278	0.418842	0.094478	-0.622610	-0.979994	-0.956214	True	296011281
...	...	...	...	...	...	...	...	...	...
495	-0.292182	-0.551867	0.418842	-0.271977	0.409445	-0.185073	0.622997	True	297891012
496	-0.228339	-0.014667	-0.263311	-0.201869	0.190524	-0.353692	-0.582060	True	297241892
497	0.040174	0.029384	0.418842	-0.146179	0.409445	-0.185073	-0.049118	True	297051994
498	0.298819	0.667343	0.418842	1.445438	-0.622610	-0.979994	-0.049118	True	297894162
499	0.988362	0.841463	0.418842	0.778162	0.409445	-0.185073	-0.956214	True	297687302

Figure 3. ETI31 après normalisation

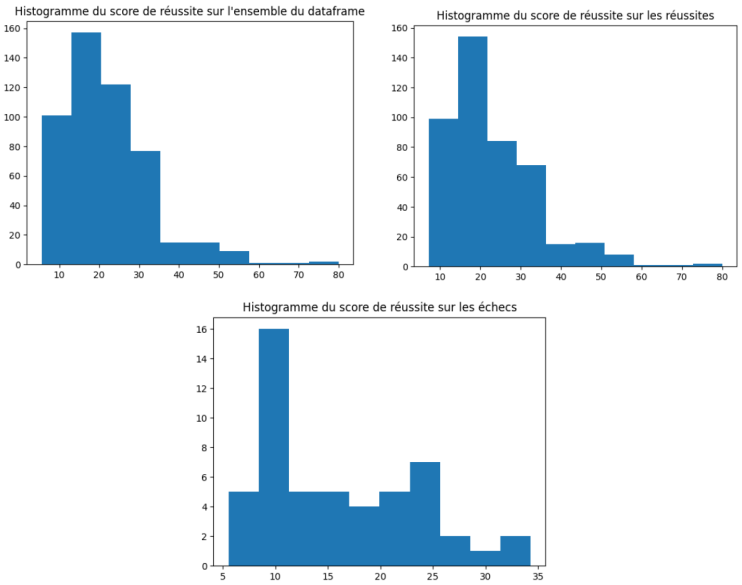
5.2 Statistiques

Les dataframes étant assez peu lisibles, faisons quelques statistiques pour mieux comprendre les résultats. Les histogrammes suivants sont basés sur les données de ETI31 avant la normalisation et sont séparés en 3 : l'ensemble, les réussites et les échecs. Tout d'abord, regarde la proportion d'échecs et de réussites :

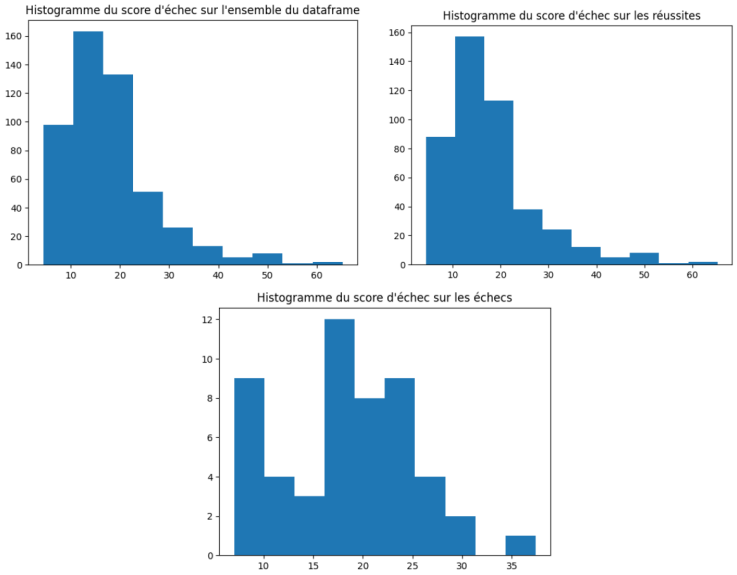



```
Reussite
True     89.6
False    10.4
```

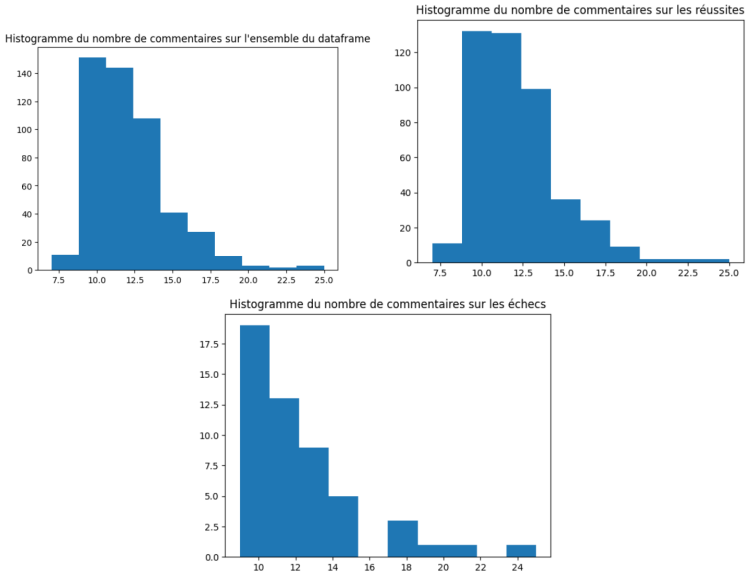
Score réussite



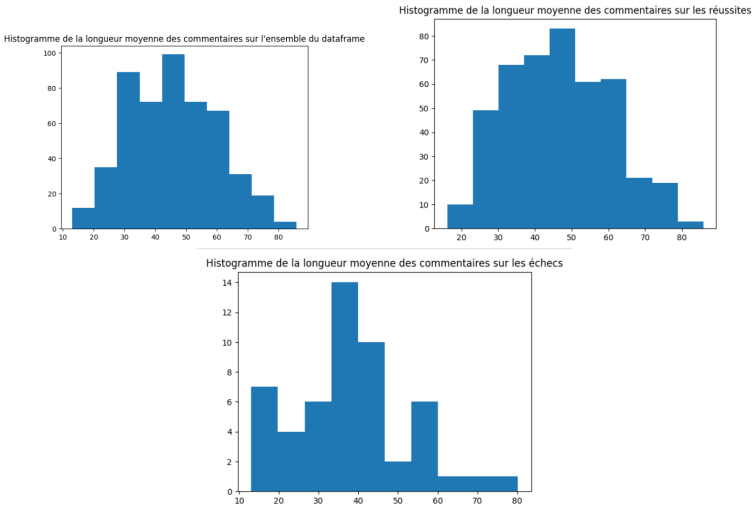
Score échec



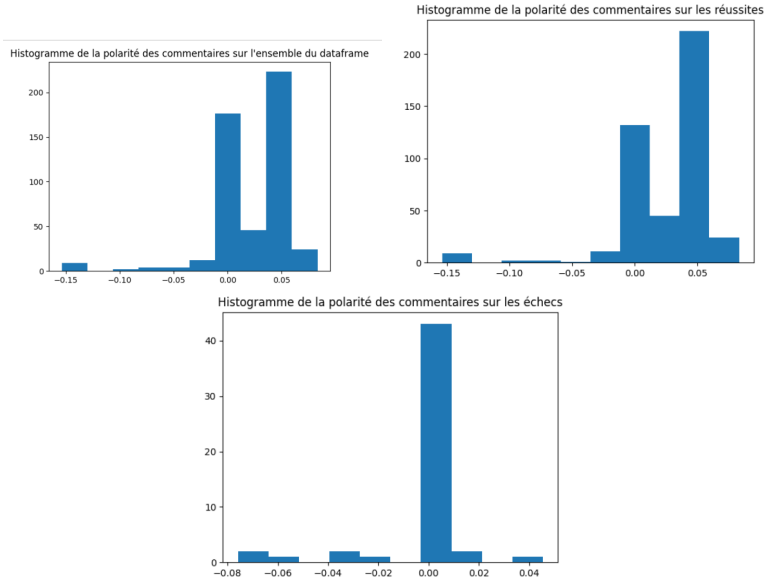
Nombre de commentaires



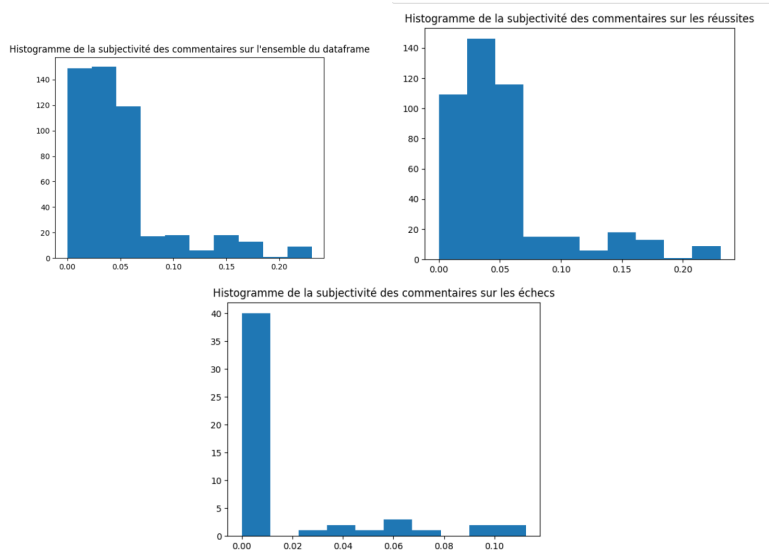
Longueur moyenne des commentaires



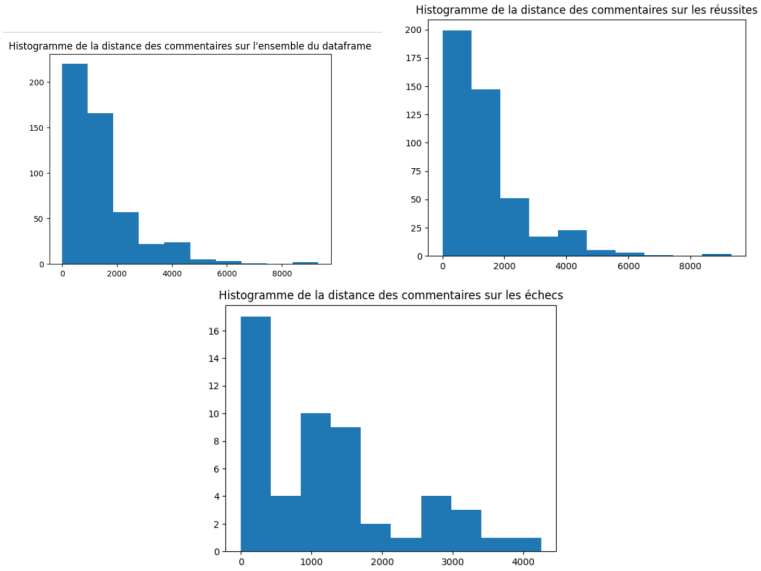
Polarité



Subjectivité



Distance au point de branchement



5.3 Création des modèles et résultats

Après la préparation des données, on s'occupe de la création de modèles à partir des données d'ETI31 traitées en suivant le processus précédemment décrit. De la même manière que pour la modélisation basée uniquement sur les commentaires nous regardons 6 types de modélisation (arbre de décision, forêts aléatoires, k-nearest neighbors, regression logstique, SVM et Naive Bayes). La méthode de comparaison entre les différents modèles sera toujours la même : maximiser la valeur prédictive négative dans un premier temps et dans un seconde temps maximiser la spécificité. Rappelons la signification de ses 2 indicateurs :

- Valeur prédictive négative : proportion de véritables échecs parmi les échecs prédits (VN/(VN+FN))
- Spécificité : proportion d'échecs correctement prédits sur l'ensemble des échecs (VN/(VN+FP))

5.3.1 Résultats par modèle

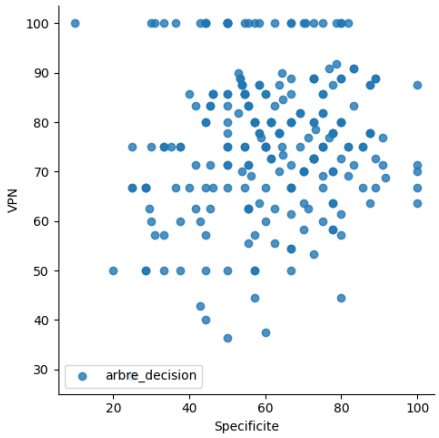
Arbre de décision

Comme pour la partie précédente un réglage des hyperparamètre a été fait. Voici les hyper-paramètres retenus :

- Criterion : gini
- min_samples_leaf = 1

- max_depth = 3

250 modèles ont été générés pour voir les résultats que l'on pouvait attendre de ce type de modèles :



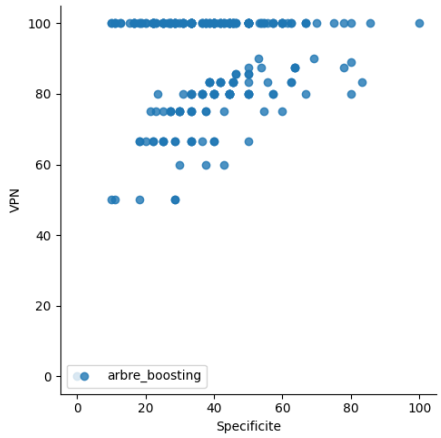
On remarque tout de suite que les modèles ont des statistiques très variables. On peut passer de moins de 40% de VPN et de spécificité à une modèle avec une spécificité parfaite et 80% de spécificité.

Random forests

Comme pour la partie précédente un réglage des hyperparamètre a été fait. Voici les hyper-paramètres retenus :

- Criterion : gini
- min_samples_leaf = 1
- max_depth = 3

250 modèles ont été générés pour voir les résultats que l'on pouvait attendre de ce type de modèles :



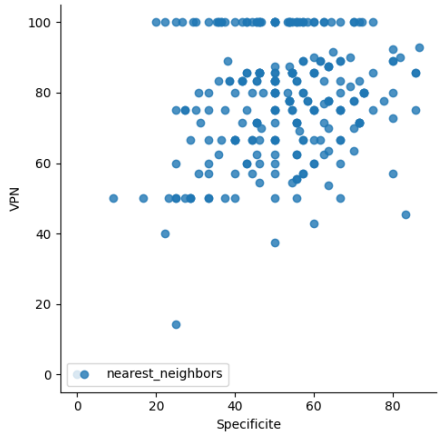
On remarque que les modèles de forêts aléatoires ont des résultats un peu moins variables et généralement meilleurs que les arbres de décision classiques. Beaucoup de modèles atteignent même une VPN de 100% négliger totalement la spécificité même si la plupart des modèles sont entre 20 et 60%.

Nearest Neighbors

Comme pour la partie précédente un réglage des hyperparamètre a été fait. Voici les hyper-paramètres retenus :

- n_neighbots = 3
- weights = uniform
- p = 1

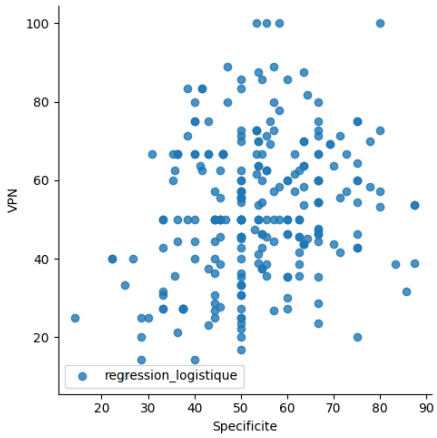
250 modèles ont été générés pour voir les résultats que l'on pouvait attendre de ce type de modèles :



On observe que les modèles ont des statistiques aussi assez variables, la VPN peut varier de 60% et la spécificité de 40% entre 2 modèles. Cependant, on note que une partie non négligeable des modèles atteint les 100% de valeur prédictive négative.

Régression logistique

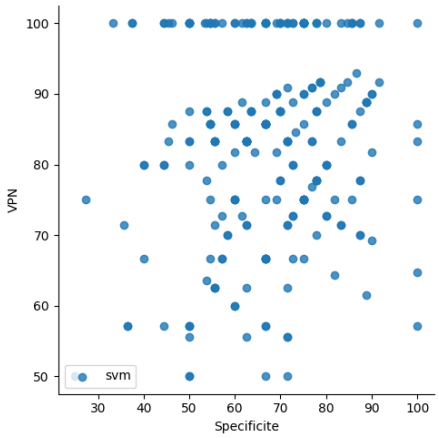
250 modèles ont été générés pour voir les résultats de que l'on peut obtenir avec un modèle de régression logistique. Voici le nuage de point représentant la vpn et la spécificité de ces modèles :



On observe une forte variabilité en terme de résultats. La VPN varie de - de 20% à 100% et la spécificité prend des valeurs comprises entre 10% et 90%.

SVM

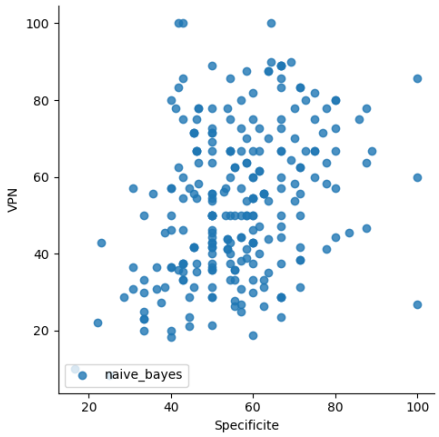
A nouveau, on a généré 250 modèles pour se faire une idée des résultats que peut produire un modèle de type SVM. Voici les résultats obtenus :



Comme pour beaucoup d'autres modèles on peut tout de suite voir que les modèles ont des statistiques très variées. Néanmoins, beaucoup d'entre eux atteignent une valeur prédictive négative de 100% et une spécificité relativement satisfaisante bien que rarement au dessus des 80%.

Naive Bayes

Enfin, nous avons également créé 250 modèles du type Naive Bayes. Voici les résultats obtenus :



Les résultats sont très similaires aux modèles de régression logistique (comme dans le cas de l'analyse uniquement basée sur les commentaires) avec des valeurs très variables en fonction des modèles vis de la spécificité comme de la valeur prédictive négative.

5.3.2 Résultats globaux et comparaison

Ci-dessous, le graphique regroupant l'ensemble des types de modèles avec en ordonnées la valeur prédictive négative et en abscisses la spécificité. Attention lors de la lecture des graphiques, les axes ne sont pas gradués de la même façon.

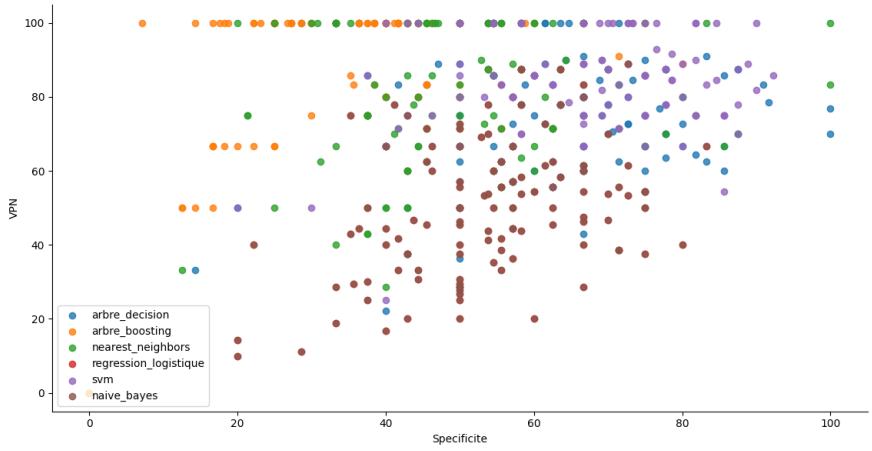


Figure 1: Avec la distance au PB

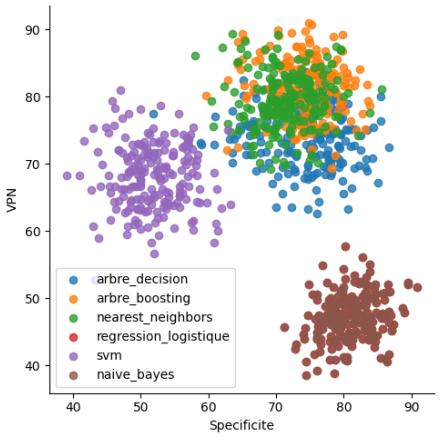


Figure 2: Sans la distance au PB

On observe tout d'abord que les résultats sont assez différents de ceux que l'on avait avec l'analyse uniquement basée sur les commentaires. On ne retrouve

plus les 3 groupes de points : les modèles ont des performances très variables. Et pour certains modèles, les résultats sont meilleurs. Il semblerait donc que la distance au PB a une réelle incidence dans la construction des modèles.

Cette fois ce sont les modèles basés sur les SVM et les arbres de décisions qui ont les meilleurs résultats. Les forêts aléatoires et nearest neighbors ont également des bons résultats mais il y a d'avantage de variabilité sur ces modèles. Les modèles de naive bayes et de régression logistiques ont des résultats nettement moins bons (notamment vis à vis de la VPN) que la plupart des modèles des autres types, ils sont mêmes moins bons que pour les modèles fait sur les données sans la distance au point de branchement.

Pour créer les modèles finaux, une boucle est utilisée pour créer un nouveau modèle jusqu'à ce qu'il atteigne au moins un certain pourcentage de spécificité et de valeur prédictive négative. Voici les statistiques des modèles qui ont été enregistrés dans les fichiers Joblib.

Modèle	Arbre de décision	Arbre boosting	Nearest neighbors	Regression logistique	SVM	Naive Bayes
VPN	100%	90%	81,80%	75%	100%	88%
Spécificité	100%	90%	100%	60%	100%	88%

Il semblerait que les modèles à privilégier soient les arbres de décision et les SVM mais il faut cependant rester prudent pour éviter de tomber dans le cas du sur apprentissage.

5.4 Prédictions

La dernière étape est celle de la prédiction, pour cela nous proposons 2 approches : 2 retours avec la probabilité d'échec et la probabilité de réussite ou un seul retour avec un label "Réussite" ou "Echec" avec la possibilité d'ajouter un seuil de confiance (compris entre 0 et 1).

Voici un exemple avec une petit dataframe :

	iar_ndfictif	COMMLITIGEPIDI	COMMENTCONTACT	BLOC_NOTE	COMMENTAIRE	COMMENTAIRE_DE_SIG	adresse_client	adresse_pb
0	12	\\Rdv reprit avec le client pour le mercredi 1...	\GM Un rdv a ete repris par le technicien p...	Oui	/DMS/#PTO_non existante#/Poser 30 metres de c ...	NaN	4 impasse de l'if Montfort-sur-Meu 35160	6 Impasse de l'If Montfort-sur-Meu 35160 Ille ...
1	788	\\VGEM/#LITIGE: AB-Client absent#/Client absen...	\GM Laisse MEVO au 0779578758.. le 25/04 a ...	NaN	Suite a plusieurs relances aucun retour de la ...	NaN	106 Boulevard George Clemenceau 35200 Rennes B...	108 Boulevard Georges Clemenceau Rennes 35200
2	782	\\Envoye SMS au 06 22/03 a 18h30 pour repren...	NaN	NaN	Inter OK- Pose PTO Choix clt	NaN	4 Rue Pierre Kerneis, 29270 Carhaix-Plouguer	13 Rue Pierre Kerneis, 29270 Carhaix-Plouguer

Figure 1. Dataframe avant traitement.

	score_reussite	score_echec	nombre_commentaires	longueur_moyenne	polarite	subjectivite	distance	Proba échec	Proba réussite	iar_ndfictif
0	-0.861019	-1.105859	-0.604387	-1.439710	0.719061	0.053404	-0.956214	0.203571	0.796429	12
1	-1.193577	-0.117489	-0.945463	-0.803732	-0.622610	-0.979994	0.563121	0.776461	0.223539	788
2	-0.914387	-0.719734	-1.286539	-1.270840	1.054479	0.311753	-0.855817	0.288116	0.711884	782

Figure 2. Dataframe avec résultats probabilités.

	score_reussite	score_echec	nombre_commentaires	longueur_moyenne	polarite	subjectivite	distance	Reussite	iar_ndfictif
0	-0.861019	-1.105859	-0.604387	-1.439710	0.719061	0.053404	-0.956214	True	12
1	-1.193577	-0.117489	-0.945463	-0.803732	-0.622610	-0.979994	0.563121	False	788
2	-0.914387	-0.719734	-1.286539	-1.270840	1.054479	0.311753	-0.855817	True	782

Figure 3. Dataframe avec résultats labélisés (seuil=0.5)

	score_reussite	score_echec	nombre_commentaires	longueur_moyenne	polarite	subjectivite	distance	Reussite	iar_ndfictif
0	-0.861019	-1.105859	-0.604387	-1.439710	0.719061	0.053404	-0.956214	True	12
1	-1.193577	-0.117489	-0.945463	-0.803732	-0.622610	-0.979994	0.563121	True	788
2	-0.914387	-0.719734	-1.286539	-1.270840	1.054479	0.311753	-0.855817	True	782

Figure 4. Dataframe avec résultats labélisés (seuil=0.9)

On voit que lorsque le seuil choisi est plus grand (de 0.5 à 0.9), la donnée à l'index 1 passe de labélisée comme un échec à labélisée comme une réussite puisque la probabilité est seulement d'environ 0.77 d'après la figure 2.

6 Prévisions des échecs de production à partir du type de branchement entre le point de branchement et le point de terminaison optique

Après plusieurs recherches, de nouvelles données ont été mises à notre disposition par le biais de Katia Louis. Ces données contiennent notamment des informations sur le type de point de branchement, le type de raccordement entre le PB et le PTO, l'adductabilité, l'opérateur immeuble, le type d'habitation et si c'est un raccordement long.

Pour essayer d'utiliser un maximum ces données nous avons téléchargé les archives de ETI31 des 2-3 dernières années (2,5 Go !) et que nous avons croisées à l'aide de Google Cloud Storage et BigQuery avec le numéro de désignation comme pivot.

Malheureusement, le croisement n'est pas très fructueux et nous laisse avec seulement 383 lignes utilisables. De plus, quasiment toutes les nouvelles informations nous intéressant se trouvent dans une même colonne "info_pb" sous la forme d'un long string peu pratique à traiter, par exemple :

'PT 000165 // 1, R . Perray, 44119, GRANDCHAMPS DES FONTAINES — Amiante présence PM : AMX (Absence de DTA) - LocalisationPBO: 2, R . Eglantines, 44119, GRANDCHAMPS DES FONTAINES - TypePBO: PBO interieur – TypeMaterielPBO: Black Box – AutresInformations: Nombre d'appartements déclarés : 4 – RaccordementLong: Non – Pmaccessible: NA – CodeAccesSous-Sol: NA – CodeLocalPM: NA')

Certaines données comme RaccordementLong et Adduction sont respectivement présentes dans 312 données et 318 données mais sont présentent toutes les 2 dans seulement 250 cas. On a donc fait le choix de ne garder que l'information sur le type de PB et l'adductabilité car nous avions déjà très peu de données.

L'essentiel de la nouvelle préparation se passera donc sur le traitement de ce string pour récupérer le type de PB et l'adductabilité.

6.1 Préparation des données

Pour cette partie la préparation des données reste la même pour les adresses et les commentaires. Le travail se concentre sur le traitement de la colonne "info_pb" et de la colonne "operateur_immeuble".

La colonne operateur_immeuble

Pour cette colonne on opère ce que l'on appelle une "dummification". Au lieu d'avoir une seule variable avec plusieurs catégories on crée un variable (et donc une nouvelle colonne) pour chacune des catégorie. On met la valeur 1 dans la colonne de la catégorie originelle et 0 dans toutes les autres. Voici le nombre d'apparition de chaque opérateur :

operateur_immeuble	
Loire Atlantique Numerique	130
MAYENNE FIBRE	120
THDB	71
LTHD	31
GIP VENDEE NUMERIQUE (VENU)	10
FT	8
LAVAL TRES HAUT DEBIT (LTHD)	7
Vendee Numerique	3
VANNES AGGLO NUMERIQUE (REVA)	2
info	1

Pour les variables apparaissant moins de 30 fois, on les classe dans une colonne "Autres". On se retrouve donc finalement avec 5 colonnes : Loire Atlantique Numerique, MAYENNE FIBRE, THDB, LTHD et Autres.

La colonne info_pb

Pour cette colonne, on effectue également une dummification en créant 6 colonnes : aerien, 3m, chambre, blackbox, tyco et autres. Il est important de noter que

dans certaines lignes on trouve par exemple à la fois "TypeMaterialPBO" et "TypePBO" suivi du type. Pour bien prendre en compte les deux informations, on ajoute 1 dans la colonne correspondante au type pour TypeMaterialPBO et TypePBO (cela peut parfois être la même colonne).

6.2 Statistiques

6.3 Création des modèles et résultats

6.3.1 Résultats par modèle

6.3.2 Résultats généraux et comparaison