

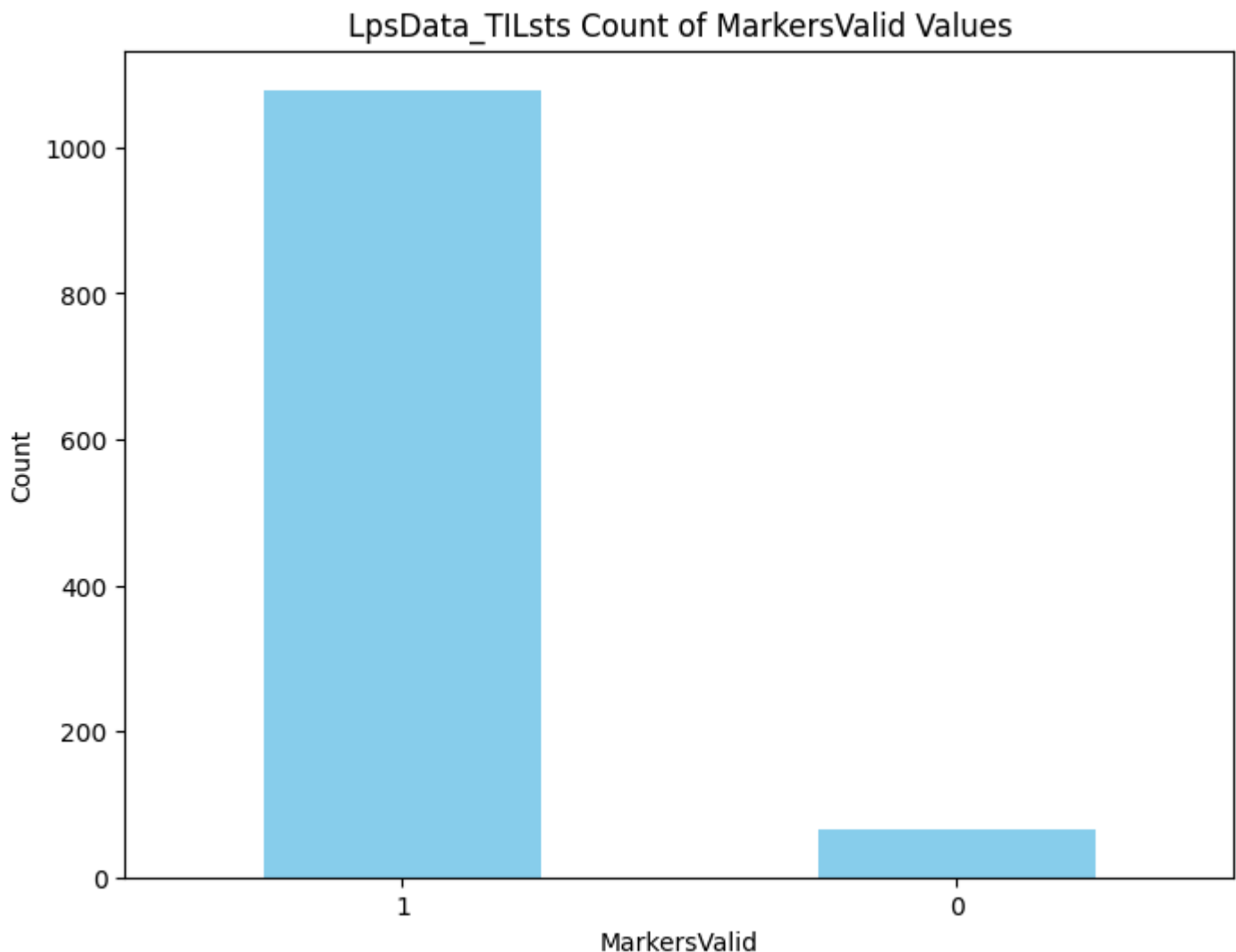
Rapport Test et Vérification

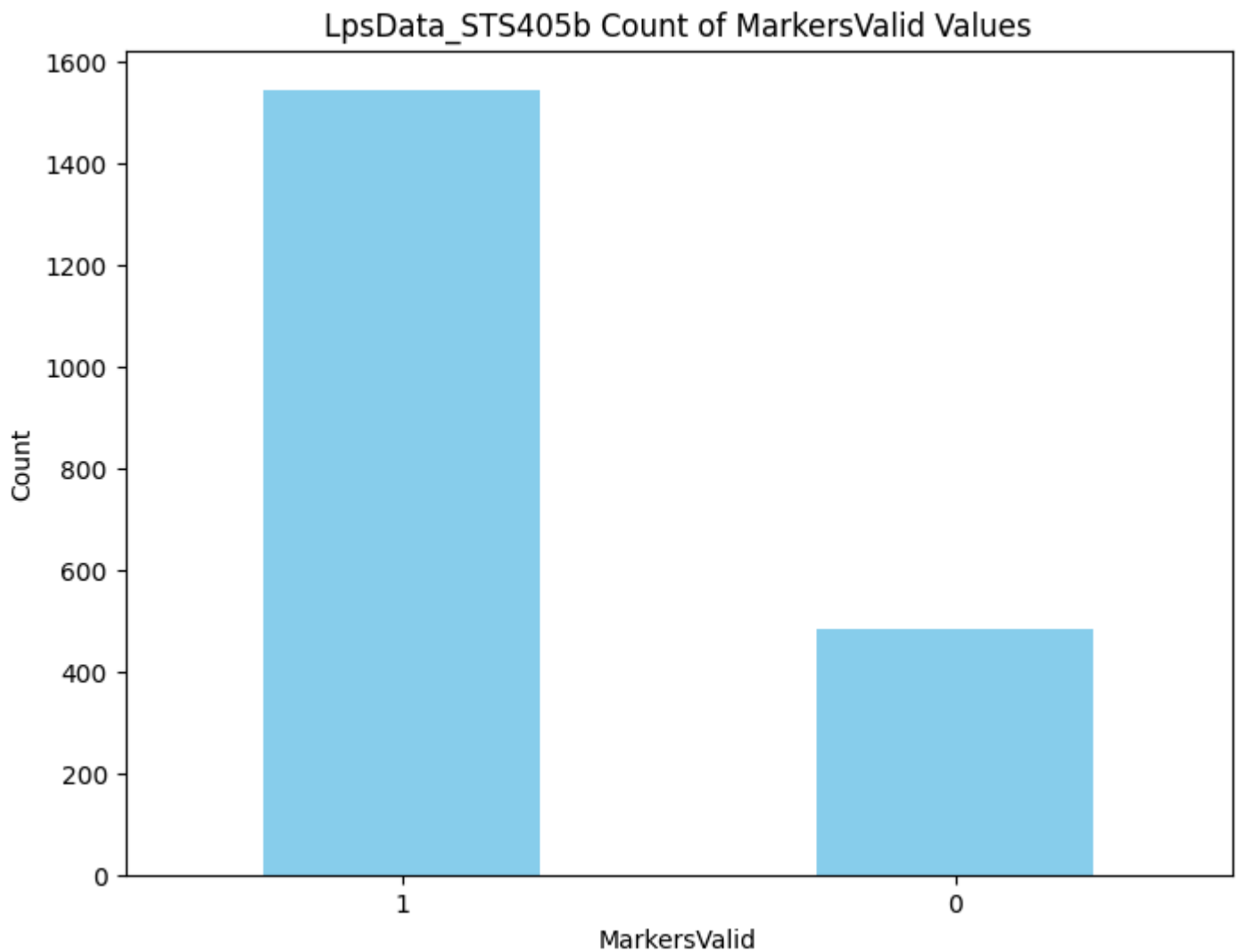
1. Crane

Durant cette exercice nous allons analysé des fichiers représentant la position de différents marquers.

EDA

On anylisant les deux fichiers on remarque que le nombre de valeur de MarkersValid a False est significativement plus petit que les valeurs a True.





Observation des marqueurs:

On visualisant les changement de position des 3 marqueurs, on remarque la présence d'une modification brusque et subite de la position d'un ou plusieurs marqueurs. la forme triangulaire ne persiste plus vraiment a certain moment.

la vidéo est disponible dans le fichier README.html

0:00



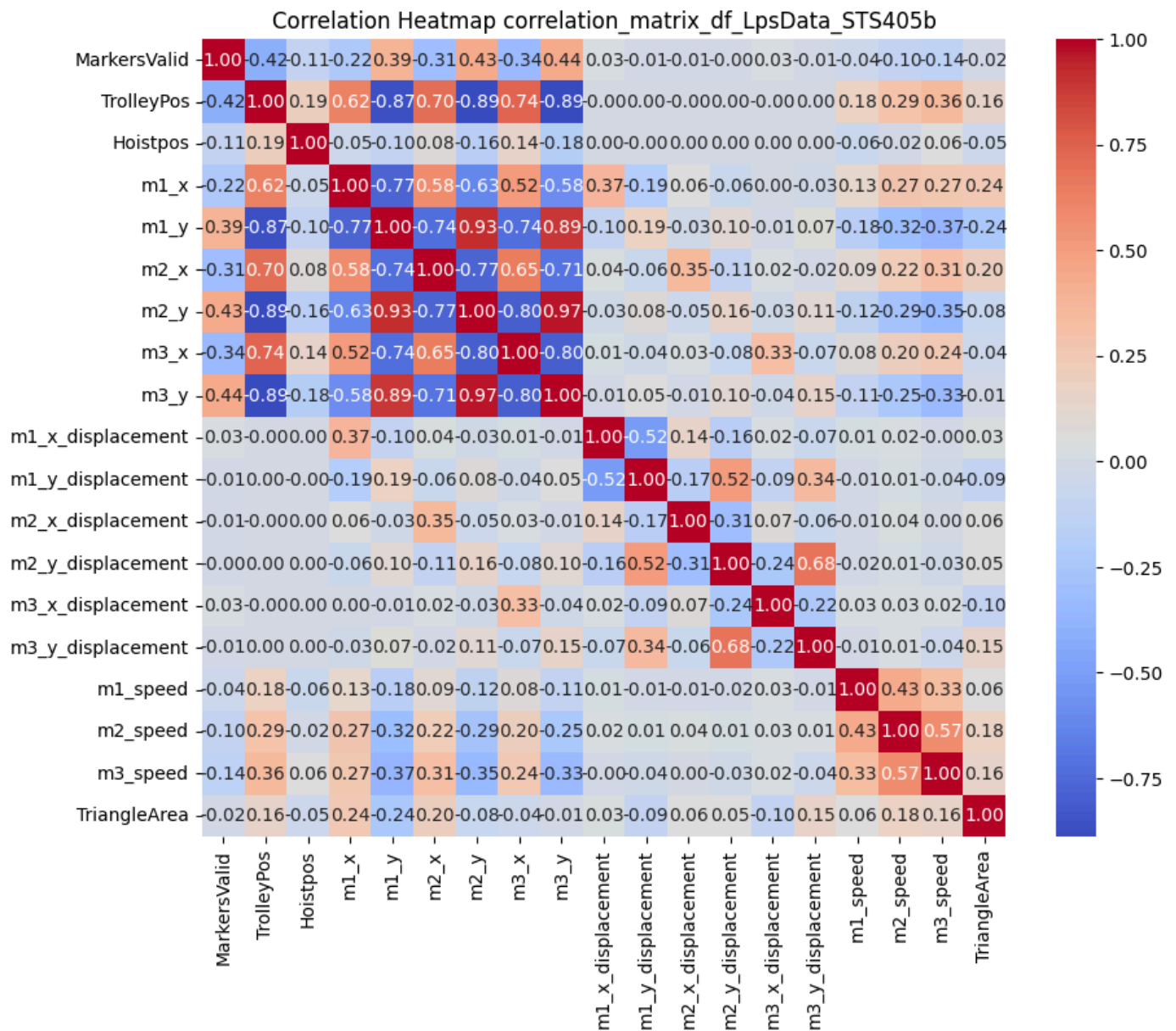
Features

Dans le but d'améliorer le modèles, plusieurs features sont mise en place:

- **Vitesse des marqueurs**: la vitesse est calculée selon la formule de la distance. On peut supposer que les déplacements à des vitesses différentes des différents marqueurs permettront d'identifier des logs erronés.
- **Déplacement des marqueurs** selon l'axe x et y : On réalise une soustraction entre $m1[i]$ et $m1[i+1]$ sur l'axe x et y afin d'avoir l'information sur le taux de déplacement.
- Calcule de la **surface du trinagle**: Variable ajouté sans observation précise, remarqué.

Affichage de la matrice de correlation

- LpsData_STS405b :
 - Il y a une forte corrélation entre TrolleyPos et les positions sur l'axe y de m1, m2 et m3 avec une valeur de -0.89
 - La position des marqueurs selon l'axe x et y est fortement corrélé ($m1_x$ foretemnt corrélé avec $m1_y$)
 - TrolleyPos est moyennement corrélé avec MarkersValid
 - Les vitesses des marqueurs présentent une correlation moyenne entre eux avec des valeurs avoisinent 0.5



- LpsData_TILsts :

- TrolleyPos est moyennement corrélé avec Hoistpos, et ne présente pas de corrélation avec MarkersValid
- Les vitesses des marqueurs présente une corrélation forte entre eux avec des valeurs avoisinent 0.9
- La superficie du triangle formé par les marqueurs est moyennement corré avec les vitesses
- La position des marqueurs selon l'axe x et y est moyennement corrélé (m1_x moyennement corrélé avec m1_y)

- Gradient Boosting Classifier: Accuracy - 0.930
- All feature
 - Logistic Regression: Accuracy - 0.948
 - Random Forest: Accuracy - 0.939
 - Support Vector Machine: Accuracy - 0.948
 - Gradient Boosting Classifier: Accuracy - 0.939
- **LpsData_STS405b :**
 - without speed
 - Logistic Regression: Accuracy - 0.744
 - Random Forest: Accuracy - 0.889
 - Support Vector Machine: Accuracy - 0.749
 - Gradient Boosting Classifier: Accuracy - 0.815
 - with speed
 - Logistic Regression: Accuracy - 0.749
 - Random Forest: Accuracy - 0.739
 - Support Vector Machine: Accuracy - 0.749
 - Gradient Boosting Classifier: Accuracy - 0.736
 - All feature
 - Logistic Regression: Accuracy - 0.761
 - Random Forest: Accuracy - 0.815
 - Support Vector Machine: Accuracy - 0.749
 - Gradient Boosting Classifier: Accuracy - 0.786

Analyse des resultats:

En examinant les rapports de classification, nous pouvons observer les performances des divers modèles d'apprentissage automatique entraînés sur nos ensembles de données, avec et sans l'inclusion des différentes caractéristiques vitesse, déplacement et surface. Voici ce que nous avons remarqué :

Régression logistique :

La régression logistique présente des performances similaires pour les deux ensembles de données, qu'elles contiennent ou non les caractéristiques de vitesse.

Les résultats montrent une difficulté du modèle à bien classifier les instances de la classe 0 (MarkersValid : False), ce qui se traduit par des scores de précision, de rappel et de F1 relativement faibles pour cette classe.

En revanche, le modèle se comporte bien dans la classification de la classe 1 (MarkersValid : True), avec des scores de précision, de rappel et de F1 élevés.

Random Forest :

Sans les caractéristiques de vitesse, le modèle Random Forest obtient une précision plus élevée que la régression logistique pour les deux ensembles de données.

L'introduction des caractéristiques de vitesse entraîne une légère baisse de la précision, surtout pour l'ensemble de données LpsData_STS405b.

Globalement, le modèle Random Forest donne de bons résultats dans la classification des deux classes, avec des scores de précision, de rappel et de F1 relativement élevés pour chacune.

Machine à vecteurs de support (SVM) :

Les performances de SVM sont comparables à celles de la régression logistique, montrant des résultats similaires en termes de précision et de mesure de performance.

Toutefois, comme la régression logistique, SVM éprouve des difficultés à classer correctement les instances de la classe 0 (MarkersValid : False), ce qui entraîne des scores de précision, de rappel et de F1 plus bas pour cette classe.

Gradient Boosting Classifier :

Le Gradient Boosting Classifier se comporte bien en termes de précision, en particulier pour l'ensemble de données LpsData_STS405b.

Ce modèle affiche une performance équilibrée dans la classification des deux classes, avec des scores de précision, de rappel et de F1 relativement élevés pour chacune.

Conclusion

Les modèles Random Forest et Gradient Boosting ont tendance à mieux performer que la régression logistique et SVM en termes de précision et de performance équilibrée sur les deux classes.

Cependant, l'ajout des caractéristiques de vitesse n'améliore pas systématiquement les performances de ces modèles, et dans certains cas, cela conduit même à une légère diminution de la précision.

Cela suggère que les caractéristiques de vitesse ne fournissent pas nécessairement d'informations supplémentaires significatives pour la tâche de classification.

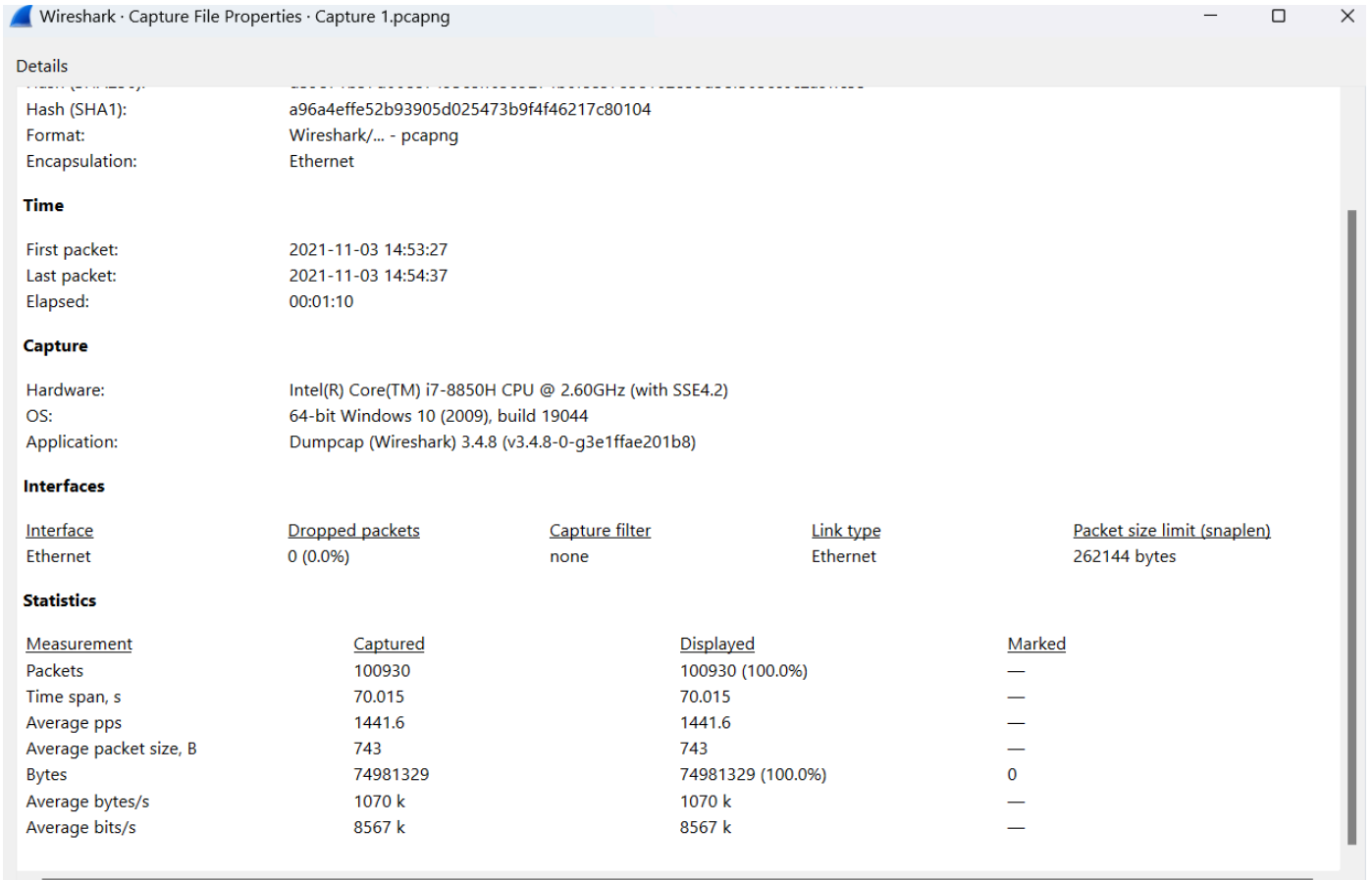
2. Trafic

Durant cette exercice nous allons effectué une analyse des packets réseau. Dans l'objectif est d'identifier les sessions en erreurs.

Carectéristic du trafic

En premier lieu nous allons analyser les packets dans WireShark afin d'identifier les caractéristique des deux fichiers.

- capture 1



Wireshark · Capture File Properties · Capture 1.pcapng

Details

Hash (SHA1): a96a4effe52b93905d025473b9f4f46217c80104
Format: Wireshark/... - pcapng
Encapsulation: Ethernet

Time

First packet: 2021-11-03 14:53:27
Last packet: 2021-11-03 14:54:37
Elapsed: 00:01:10

Capture

Hardware: Intel(R) Core(TM) i7-8850H CPU @ 2.60GHz (with SSE4.2)
OS: 64-bit Windows 10 (2009), build 19044
Application: Dumpcap (Wireshark) 3.4.8 (v3.4.8-0-g3e1ffae201b8)

Interfaces

<u>Interface</u>	<u>Dropped packets</u>	<u>Capture filter</u>	<u>Link type</u>	<u>Packet size limit (snaplen)</u>
Ethernet	0 (0.0%)	none	Ethernet	262144 bytes

Statistics

<u>Measurement</u>	<u>Captured</u>	<u>Displayed</u>	<u>Marked</u>
Packets	100930	100930 (100.0%)	—
Time span, s	70.015	70.015	—
Average pps	1441.6	1441.6	—
Average packet size, B	743	743	—
Bytes	74981329	74981329 (100.0%)	0
Average bytes/s	1070 k	1070 k	—
Average bits/s	8567 k	8567 k	—

- capture 2

Wireshark · Capture File Properties · Capture 2.pcapng

Details

Hash (SHA1):4d3b1c3a09de159133aaa0b5514a2b4a4d4a674c
Format:Wireshark/... - pcapng
Encapsulation:Ethernet

Time

First packet:2021-11-03 15:20:38
Last packet:2021-11-03 15:24:01
Elapsed:00:03:22

Capture

Hardware:Intel(R) Core(TM) i7-8850H CPU @ 2.60GHz (with SSE4.2)
OS:64-bit Windows 10 (2009), build 19044
Application:Dumpcap (Wireshark) 3.4.8 (v3.4.8-0-g3e1ffae201b8)

Interfaces

Interface	Dropped packets	Capture filter	Link type	Packet size limit (snaplen)
Ethernet	0 (0.0%)	none	Ethernet	262144 bytes

Statistics

Measurement	Captured	Displayed	Marked
Packets	317593	317593 (100.0%)	—
Time span, s	202.667	202.667	—
Average pps	1567.1	1567.1	—
Average packet size, B	175	175	—
Bytes	55545692	55545692 (100.0%)	0
Average bytes/s	274 k	274 k	—
Average bits/s	2192 k	2192 k	—

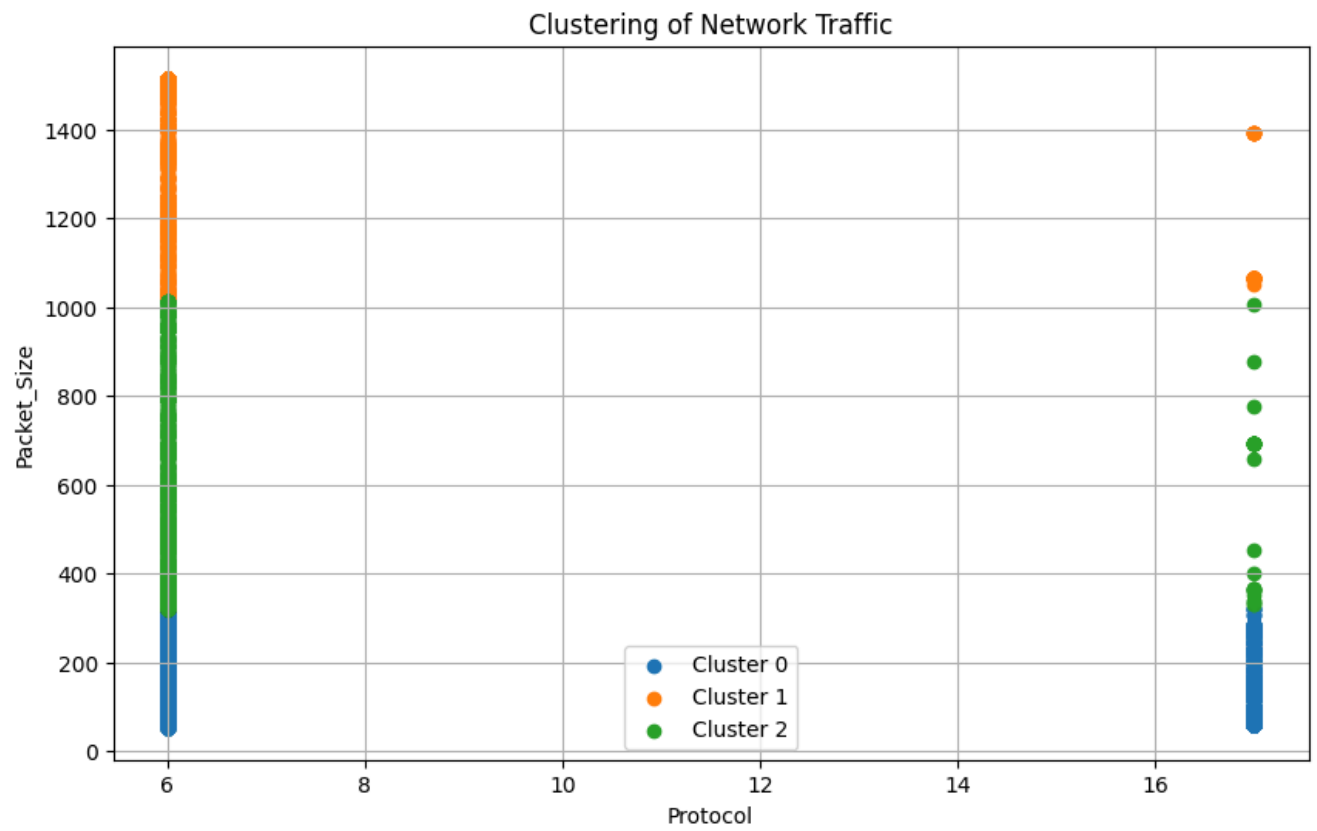
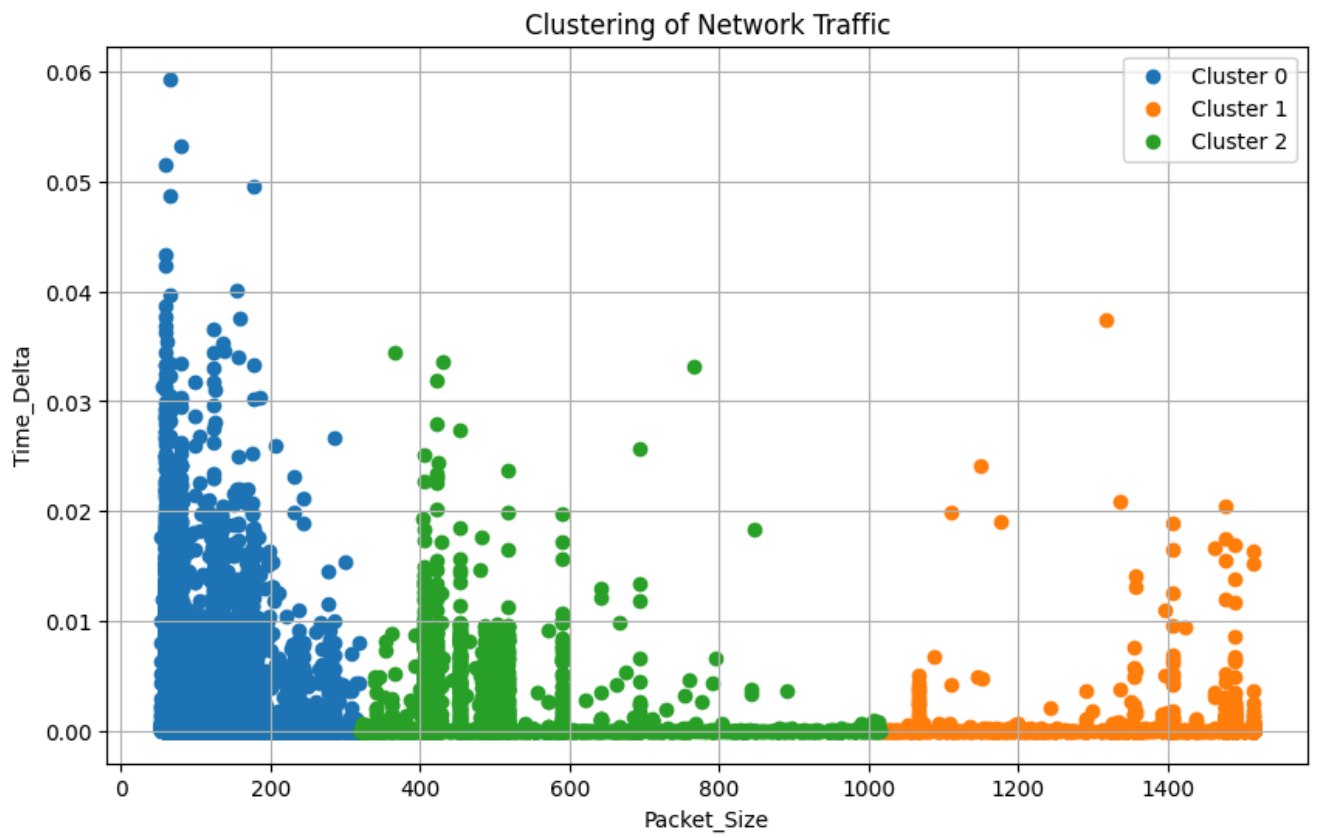
On remarque que le nombre de packets dans le fichier de capture 2 est supérieurs au packets de la capture 1. Par contre en moyenne les packets de la capture 1 sont supérieurs en terme de taille.

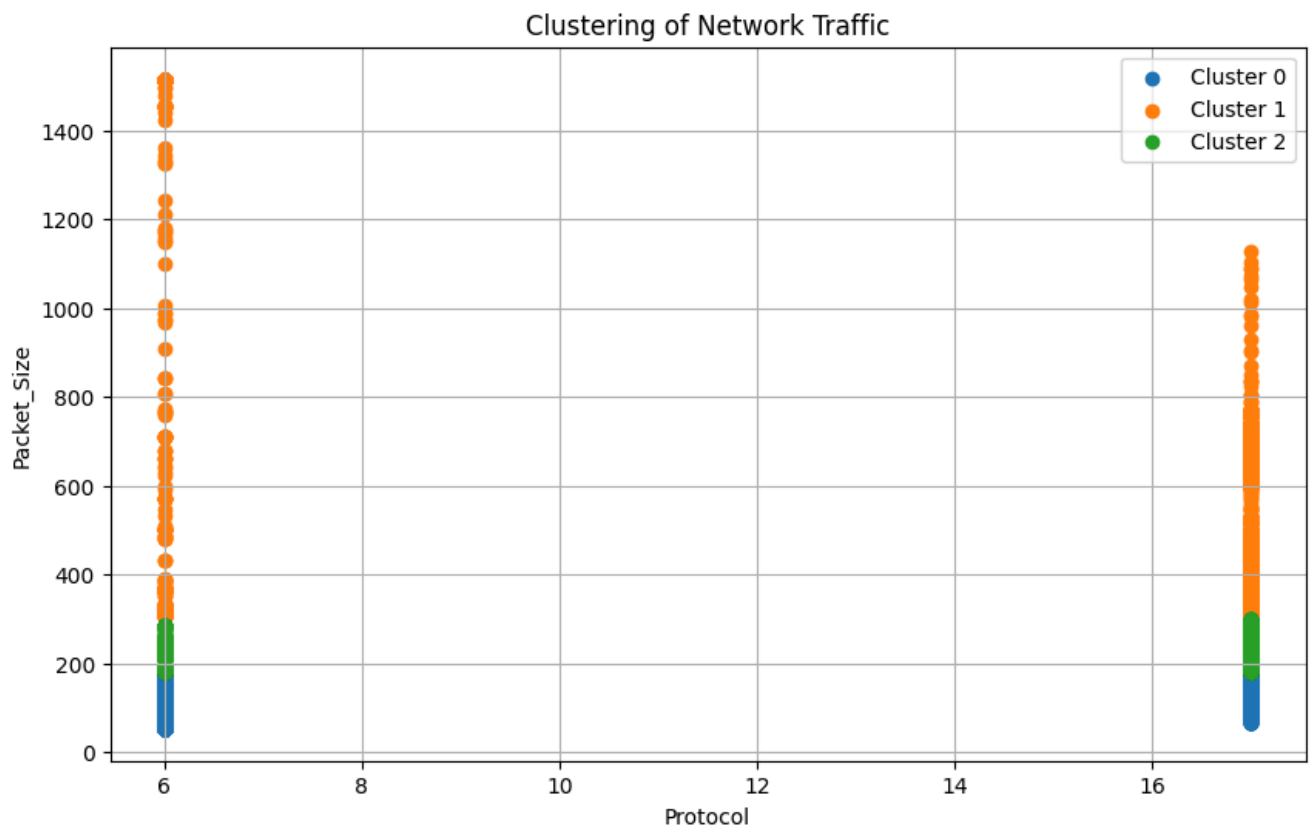
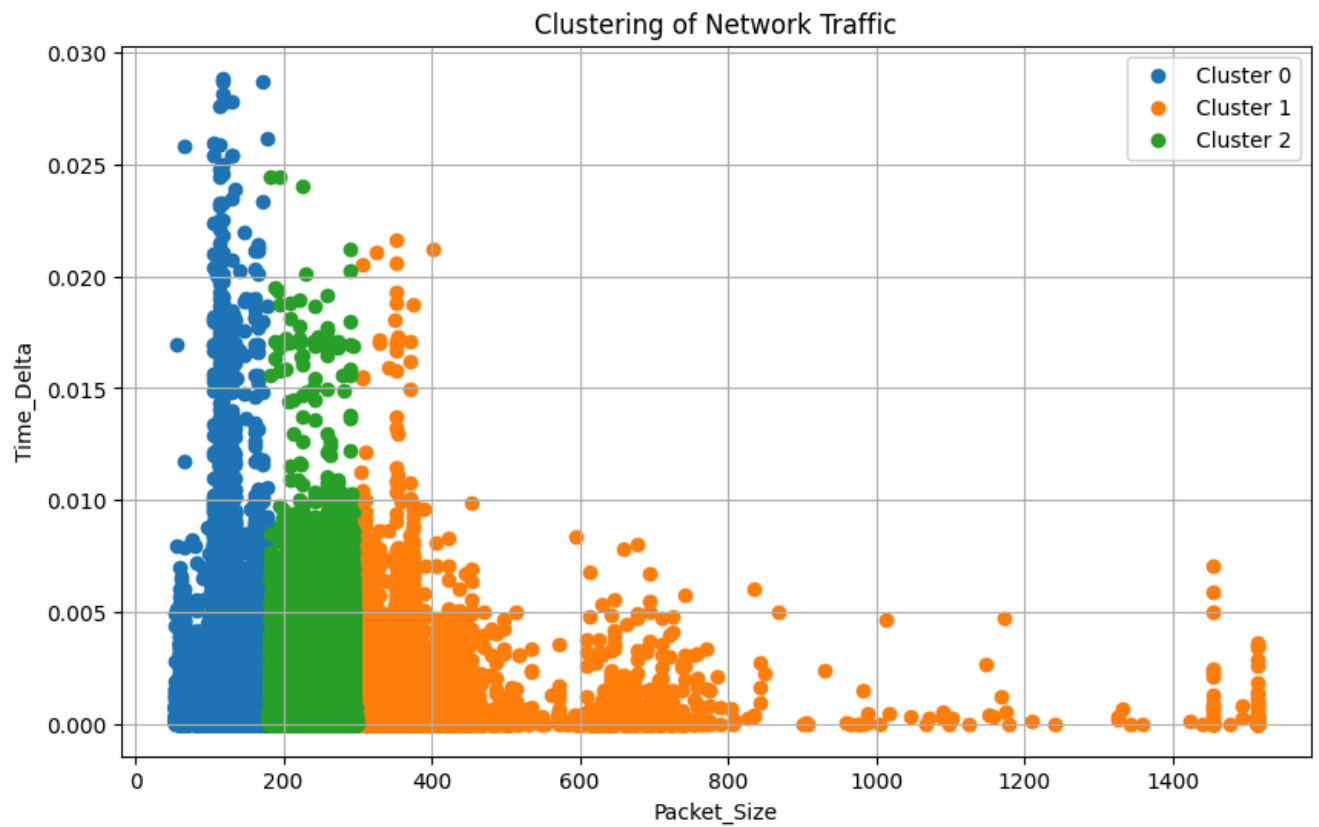
Extraction des caractéristique avec python & clustering

- packets
 - Informations

	Source_IP	Destination_IP	Source_Port	Destination_Port	Protocol	Packet_Size	Time_Delta	Packet_Type
0	172.18.226.121	192.168.229.51	52255.0	10443.0	6.0	60	None	Control
1	172.18.226.121	192.168.229.51	52255.0	10443.0	6.0	124	0.000332	Data
2	192.168.229.51	172.18.226.121	4500.0	64916.0	17.0	146	0.000002	Control
3	172.18.226.121	192.168.229.51	52255.0	10443.0	6.0	124	0.000159	Data
4	172.18.226.121	192.168.229.51	52255.0	10443.0	6.0	134	0.000002	Data

- Clustering

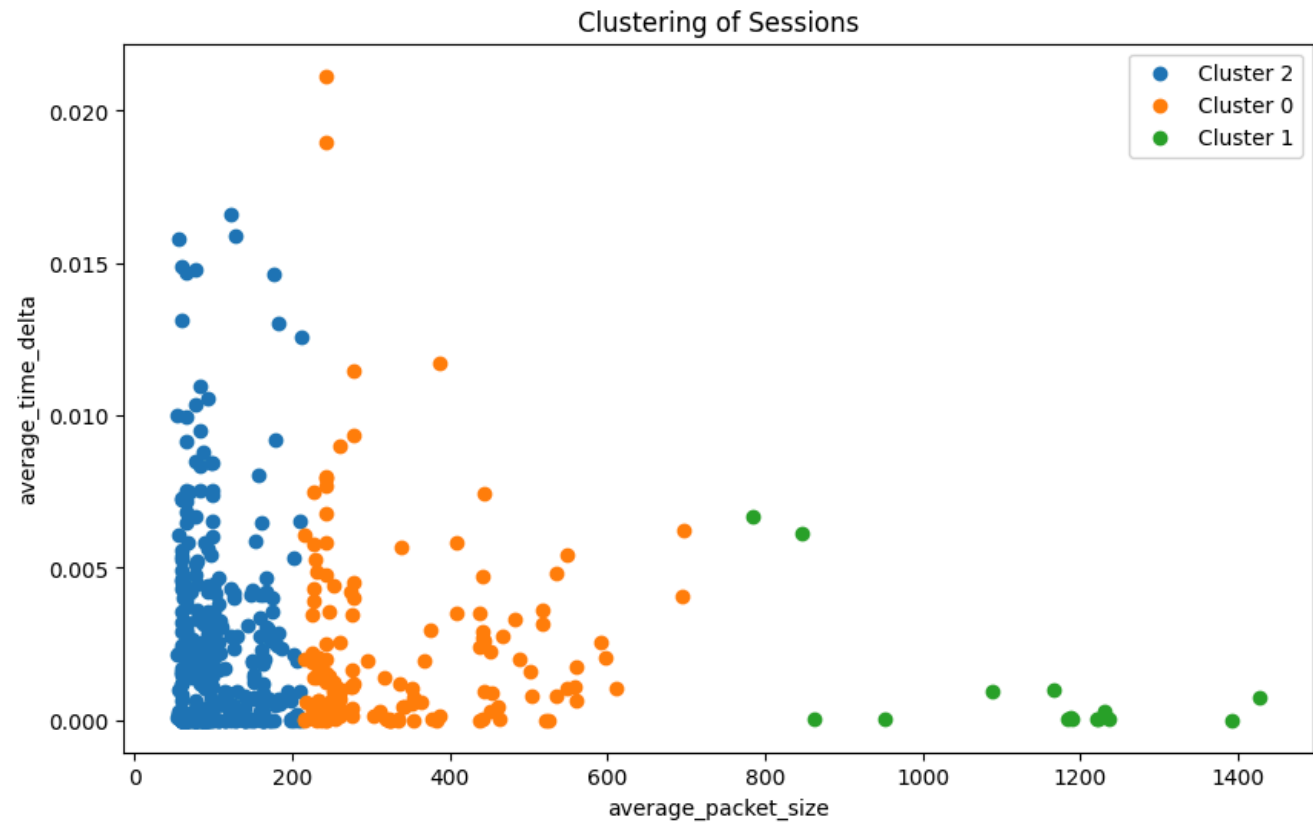
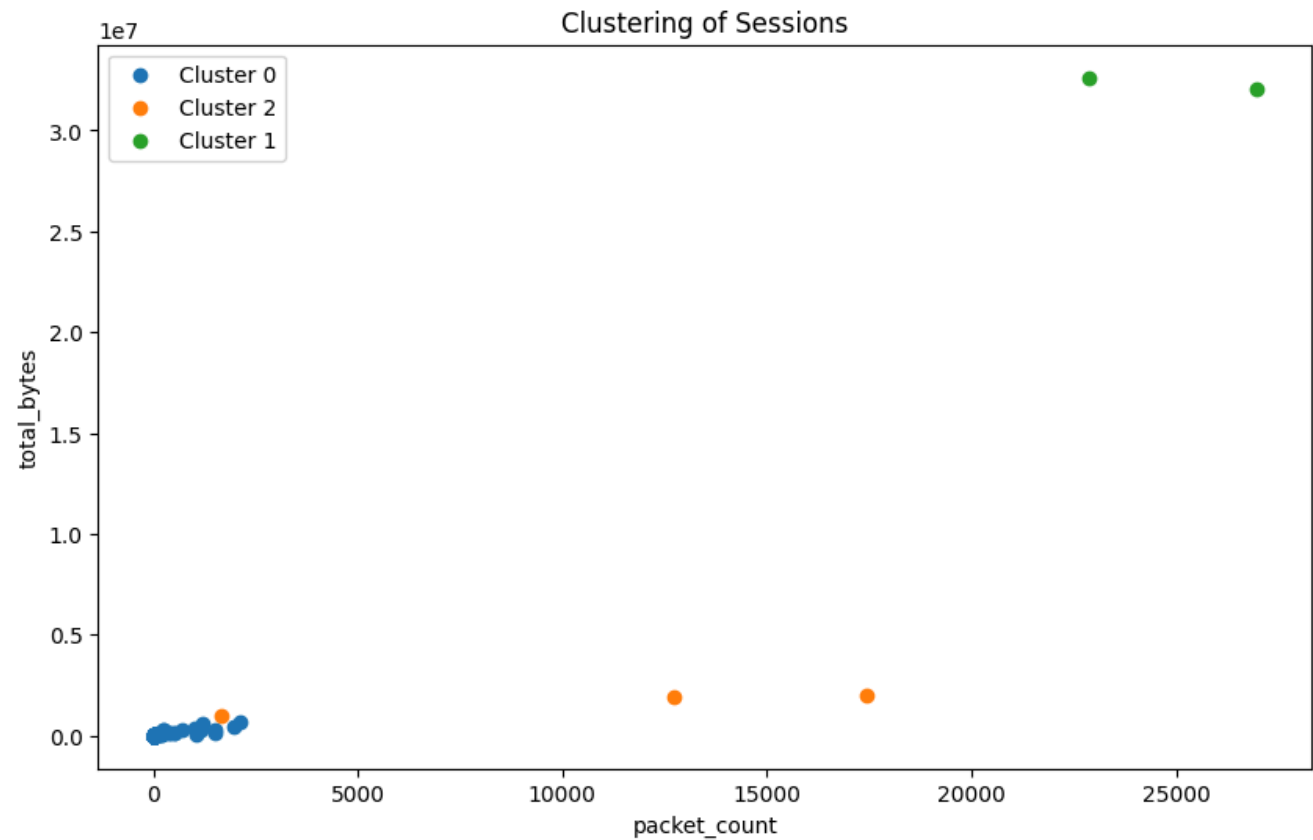


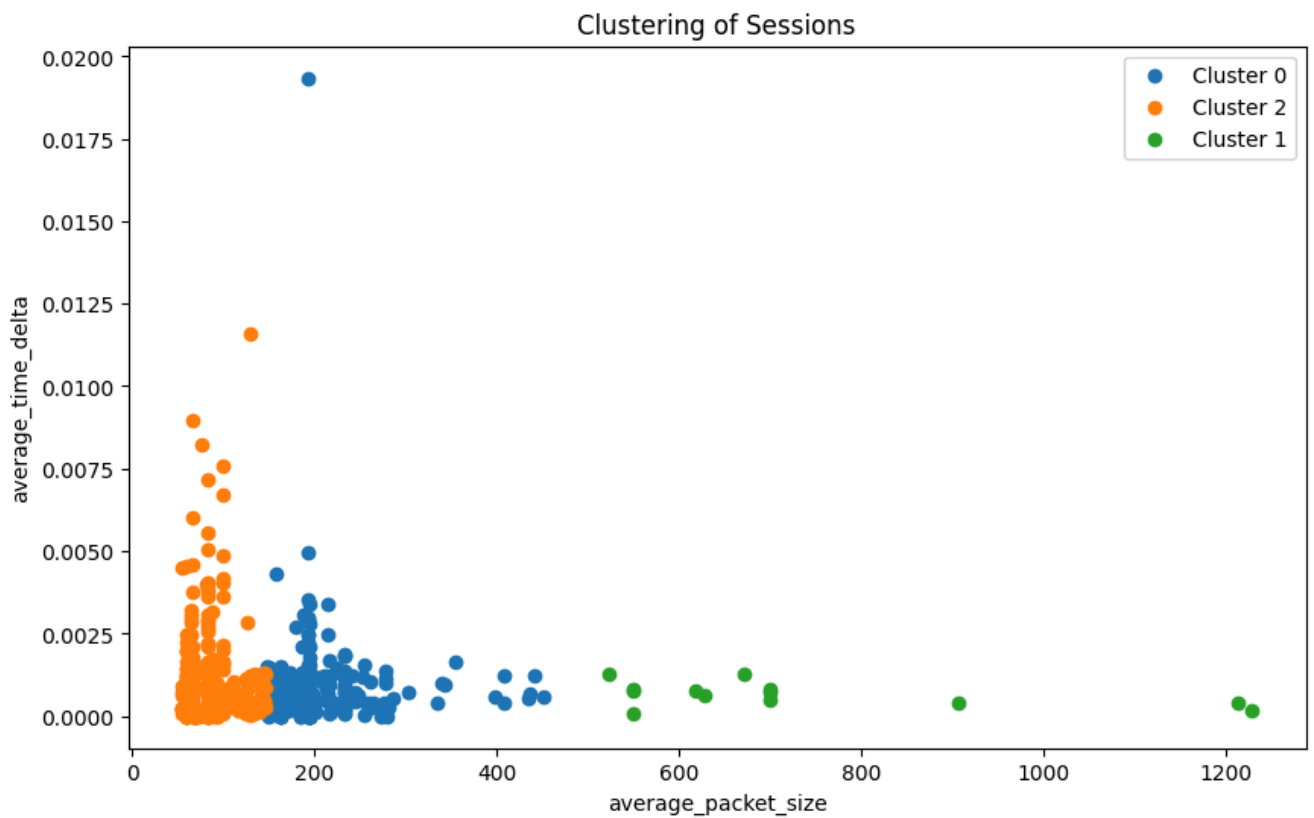
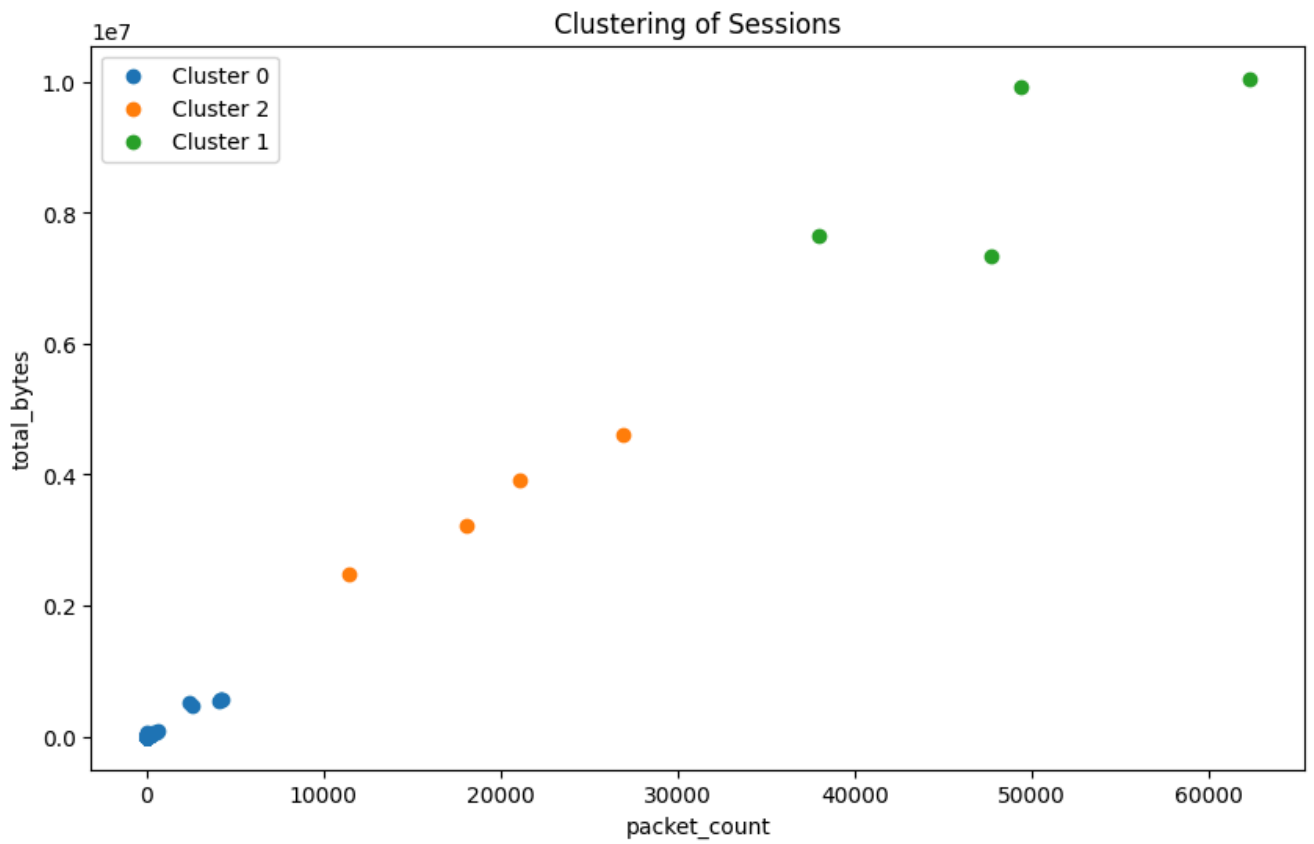


- sessions
 - Informations

packet_count	average_packet_size	average_time_delta	min_packet_size	max_packet_size	total_bytes	unique_source_ips	unique_destination_ips	unique_source_ports	unique_destination_ports	most_common_protocol
6	342.000000	0.000957	342	342	2052	1	1	1	1	17.0
205	144.341463	0.000330	102	290	29590	1	1	1	1	17.0
172	124.232558	0.000304	106	158	21368	1	1	1	1	17.0
225	138.124444	0.000243	98	270	31078	1	1	1	1	17.0
123	125.869919	0.000213	118	158	15482	1	1	1	1	17.0

o Clustering





Avec le Clustering, on remarque la separation des différentes classe pour l'ensemble des variables utilisées.

Détection d'anomalie

- Afin de reconnaître les paquets en erreurs nous allons utiliser un filtre "tcp.analysis.flags". Ce dernier permet de récupérer les packets TCP présentant des anomalies.
- **Filtre:** ((((((((((((((tls.record.length.invalid) || (dns.retransmit_request)) || (quic.decryption_failed)) || (dns.retransmit_response)) || (dns.retransmit_request)) || (tls.ignored_unknown_record)) || (tcp.analysis.ack_lost_segment)) || (esp.sequence-analysis.wrong-sequence-number)) || (quic.coalesced_padding_data)) || (quic.retransmission)) || (tcp.options.sack_perm.absent)) || (tcp.analysis.duplicate_ack)) || (tcp.connection.syn)) || (http.chat)) || (http.chat)
- Nous procédons ensuite à la détection des anomalies dans l'ensemble des sessions.

Résultats

Logistic Regression:

	precision	recall	f1-score	support
0	0.97	1.00	0.98	536
1	1.00	0.05	0.10	19
-----	-----	-----	-----	-----
accuracy			0.97	555
macro avg	0.98	0.53	0.54	555
weighted avg	0.97	0.97	0.95	555

SVM:

	precision	recall	f1-score	support
0	0.97	1.00	0.98	536
1	0.00	0.00	0.00	19
-----	-----	-----	-----	-----
accuracy			0.97	555
macro avg	0.48	0.50	0.49	555

	precision	recall	f1-score	support
weighted avg	0.93	0.97	0.95	555

Random Forest:

	precision	recall	f1-score	support
0	0.98	0.97	0.98	536
1	0.38	0.47	0.42	19
-----	-----	-----	-----	-----
accuracy			0.95	555
macro avg	0.68	0.72	0.70	555
weighted avg	0.96	0.95	0.96	555

Analyse

Basé sur le rapport de classification fourni, analysons les résultats pour chaque modèle :

Régression logistique :

- La précision pour la classe 0 est de 0.97, ce qui indique que lorsque le modèle prédit la classe 0, il a raison 97 % du temps.
- Le rappel pour la classe 0 est de 1.00, ce qui signifie que le modèle identifie correctement toutes les instances de la classe 0.
- La précision pour la classe 1 est de 1.00, ce qui indique que lorsque le modèle prédit la classe 1, il a toujours raison. Cependant, le rappel pour la classe 1 n'est que de 0.05, ce qui signifie que le modèle n'est capable d'identifier que 5 % des instances réelles de la classe 1.
- La précision globale du modèle est de 0.97.

SVM :

- La précision pour la classe 0 est de 0.97, ce qui indique que lorsque le modèle prédit la classe 0, il a raison 97 % du temps.
- Le rappel pour la classe 0 est de 1.00, ce qui signifie que le modèle identifie correctement toutes les instances de la classe 0.

- La précision et le rappel pour la classe 1 sont tous deux de 0.00, ce qui indique que le modèle échoue à prédire correctement les instances de la classe 1.
- La précision globale du modèle est de 0.97.

Random Forest :

- La précision pour la classe 0 est de 0.98, ce qui indique que lorsque le modèle prédit la classe 0, il a raison 98 % du temps.
- Le rappel pour la classe 0 est de 0.97, ce qui signifie que le modèle identifie correctement 97 % des instances de la classe 0.
- La précision pour la classe 1 est de 0.38, ce qui indique que lorsque le modèle prédit la classe 1, il a raison 38 % du temps. Le rappel pour la classe 1 est de 0.47, ce qui signifie que le modèle est capable d'identifier 47 % des instances réelles de la classe 1.
- La précision globale du modèle est de 0.95.

En résumé, les trois modèles fonctionnent bien pour identifier les instances de la classe 0, avec des scores de précision et de rappel élevés. Cependant, ils éprouvent des difficultés à identifier les instances de la classe 1, le modèle de régression logistique ayant le score de rappel le plus élevé de 0.05 parmi eux. Le modèle Random Forest a la meilleure équilibre entre précision et rappel pour la classe 1, avec des scores respectifs de 0.38 et 0.47. La précision globale est la plus élevée pour le modèle de régression logistique (0.97), suivi du modèle Random Forest (0.95) et du modèle SVM (0.97). Cependant, les scores de précision élevés sont principalement dus au jeu de données déséquilibré, où la classe 0 a significativement plus d'instances que la classe 1.

3. Analyse de Sentiment

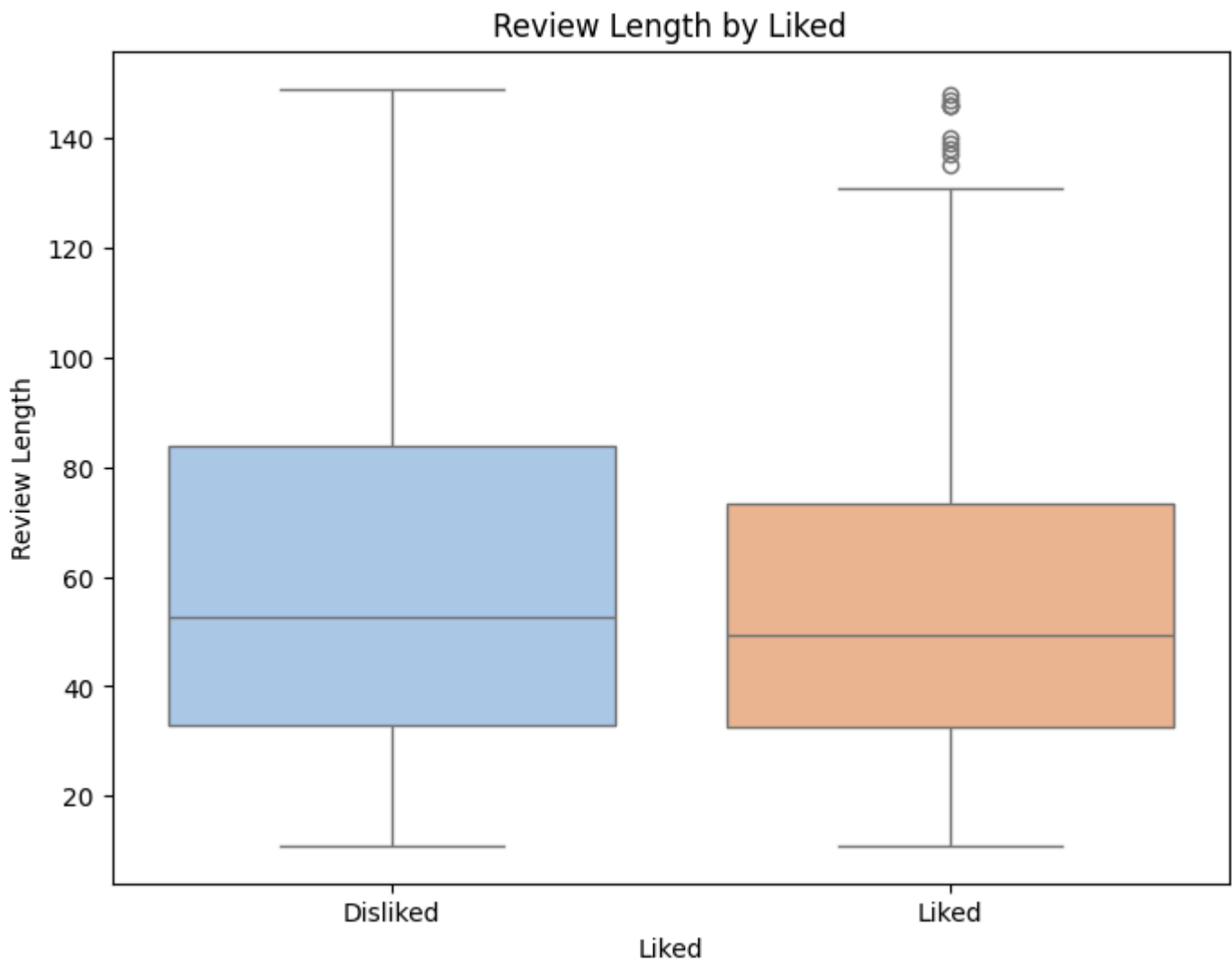
Nous allons réaliser une étude d'identification des sentiments des revues sur des restaurants.

EDA

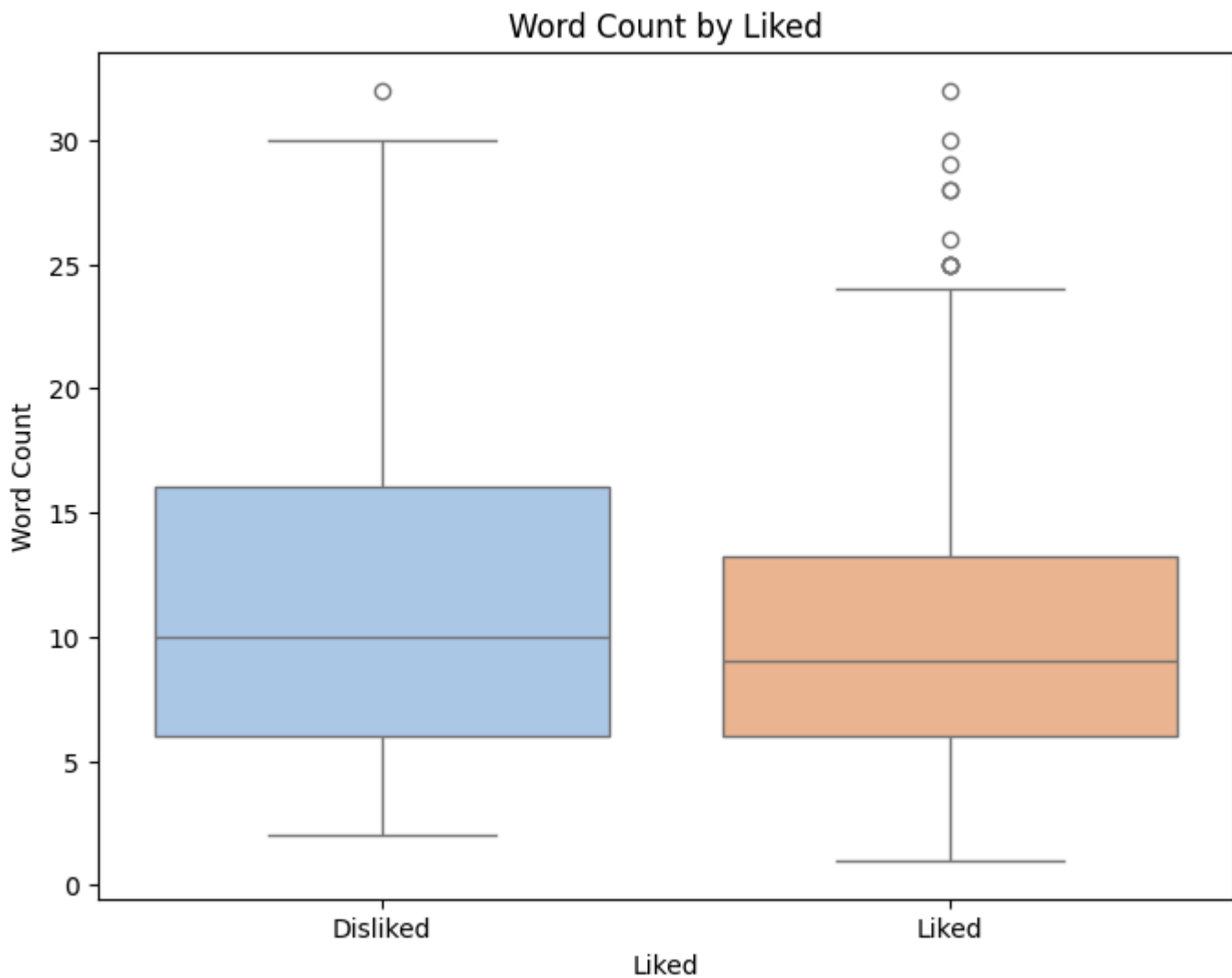
En analysant le fichier le nombre de review positive et négative est le meme 500 pour chaque.

Plusieurs features seront ajoutés afin d'améliorer les predictions des modèles.

- i. **Taille des reviews** : D'après le diagramme en boîte, la longueur moyenne des commentaires Disliked se situe entre 30 et 80 caractères, pour les commentaire liked entre 30 et 70. Il y a plus de variabilité dans la longueur des commentaires liked que dans les commentaires Disliked. On peut également constater que les commentaires Disliked semble plus long.



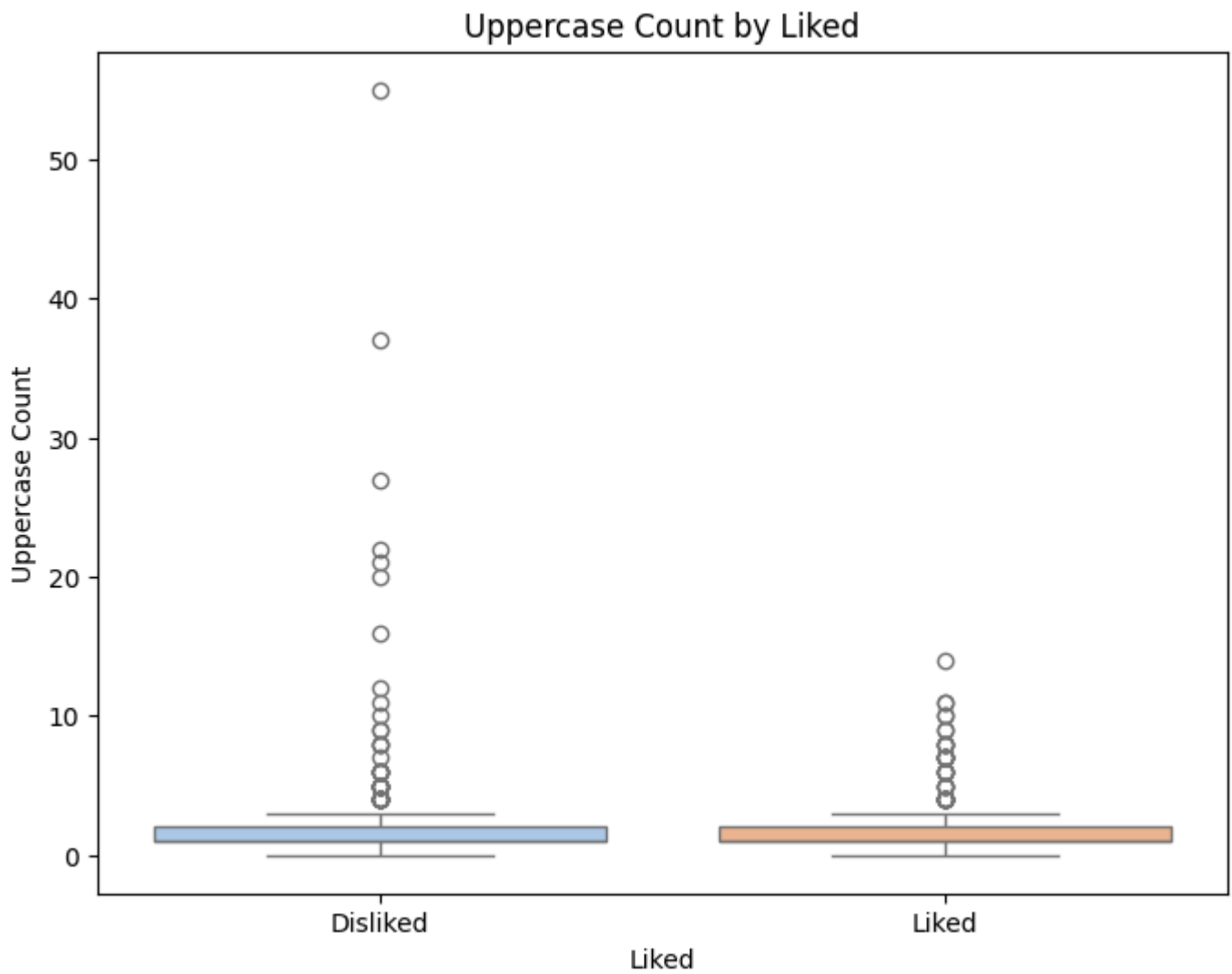
- ii. **nombre de mots** : La médiane des commentaires Disliked est plus levée que celle des commentaires aimé. Ce qui signifie que le nombre de mots est généralement plus grand lors des commentaire Disliked.
Il y a aussi quelques commentaires Liked qui sont beaucoup plus longs que la plupart des autres commentaires.



- iii. **Nombre de lettre en majuscule**

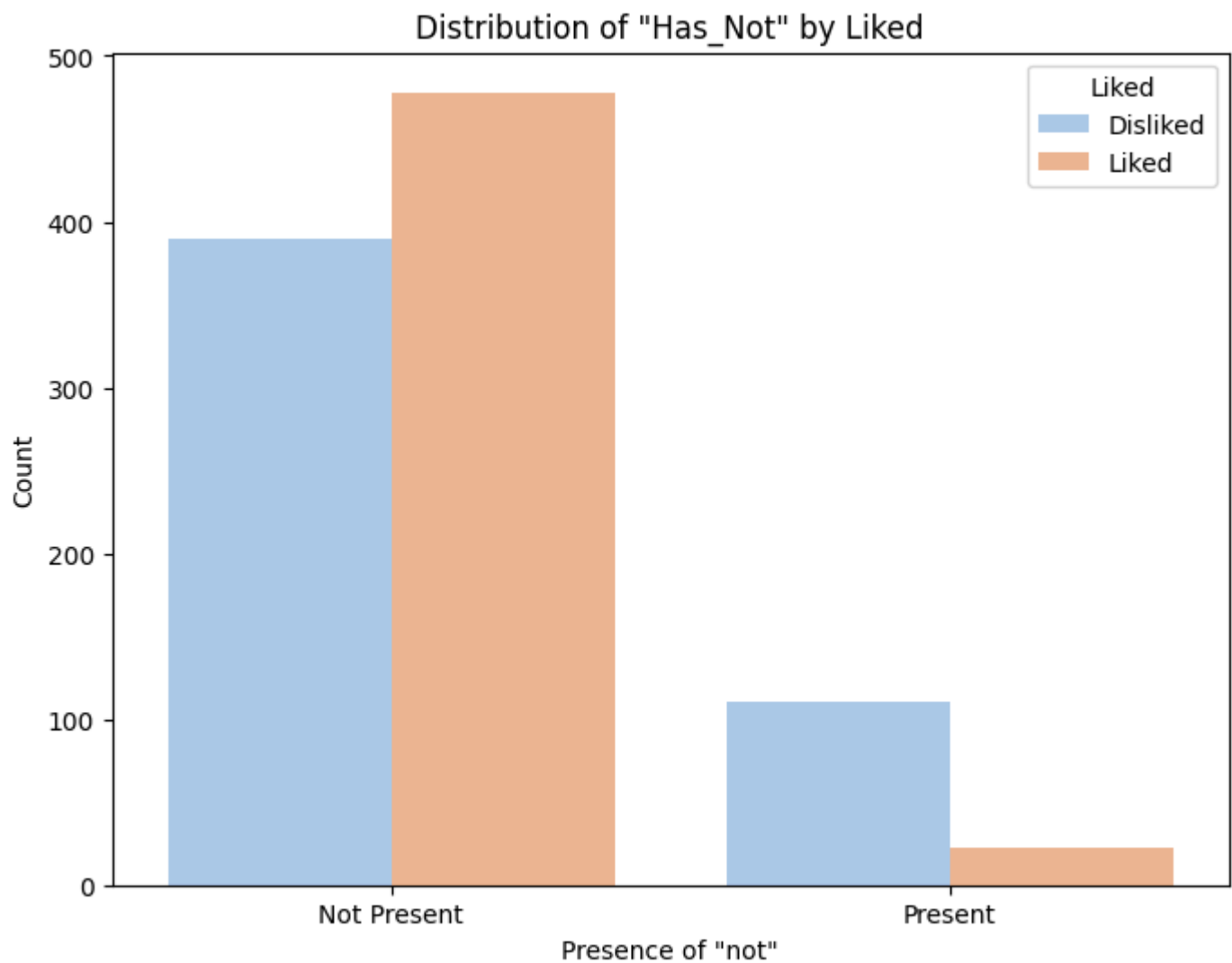
L'utilisation des majuscules est plus fréquente dans les commentaires "Disliked" que dans les commentaires "Liked".

Il y a quelques commentaires "Disliked" qui ont beaucoup plus de majuscules que la plupart des autres commentaires. Les majuscules sont souvent utilisées au début des phrases et des noms propres, et peuvent également être utilisées pour mettre l'accent sur certains mots ou phrases.



- iv. **La présence du mot not**

l'utilisation du mot "not" est plus élevée dans les commentaires "Disliked". L'utilisation de la négation est souvent utilisé dans ces cas la.



Modelisation

- Only Reviews

Model	Vectorizer	Accuracy
Logistic Regression	vectorizer_BOW	0.81
Logistic Regression	vectorizer_TF_IDF	0.80
Logistic Regression	vectorizer_BOW_bigram	0.81
Logistic Regression	vectorizer_TF_IDF_bigram	0.79
SVM	vectorizer_BOW	0.74
SVM	vectorizer_TF_IDF	0.81
SVM	vectorizer_BOW_bigram	0.74

Model	Vectorizer	Accuracy
SVM	vectorizer_TF_IDF_bigram	0.81
Random Forest	vectorizer_BOW	0.77
Random Forest	vectorizer_TF_IDF	0.73
Random Forest	vectorizer_BOW_bigram	0.76
Random Forest	vectorizer_TF_IDF_bigram	0.73

D'après les scores de précision fournis, le meilleur modèle parmi ceux testés semble être le modèle de régression logistique avec les vectoriseurs `vectorizer_BOW` (Bag-of-Words) et `vectorizer_BOW_bigram` (Bag-of-Words avec bigramme), atteignant tous deux une précision de 0,81. Ces modèles ont été constamment performants par rapport aux autres à travers différentes techniques de vectorisation.

Les modèles SVM avec `vectorizer_TF_IDF` et `vectorizer_TF_IDF_bigram` ont également bien performé, atteignant une précision de 0,81, légèrement inférieure aux modèles de régression logistique les plus performants.

Les modèles de forêt aléatoire ont montré des précisions plus faibles par rapport aux modèles de régression logistique et SVM avec toutes les techniques de vectorisation.

Dans l'ensemble, basé uniquement sur la précision, les modèles de régression logistique avec la vectorisation Bag-of-Words et les caractéristiques de bigramme semblent être le meilleur choix parmi les modèles testés.

- **Features**

- ['Word_Count', 'Review_Length']

Model	Accuracy
Logistic Regression	0.55
SVM	0.51
Random Forest	0.47

À partir des résultats fournis, il est évident que le modèle de régression logistique a mieux performé parmi les modèles testés, avec une précision de 0.55. Le modèle SVM a atteint une précision de 0.51, tandis que le modèle de Forêt Aléatoire a eu la plus faible précision de 0.47.

- ['Word_Count', 'Review_Length', 'Uppercase_Count']

Modèle	Précision
Régression Logistique	0.55
SVM	0.51
Forêt Aléatoire	0.47

Les résultats restent similaires à ceux précédemment analysés, avec la régression logistique ayant la meilleure performance, suivie du SVM, puis de la forêt aléatoire.

- ['Word_Count', 'Review_Length', 'Has_Not']

Modèle	Précision
Régression Logistique	0.58
SVM	0.50
Forêt Aléatoire	0.50

Encore une fois, la régression logistique obtient la meilleure précision, suivie du SVM et de la forêt aléatoire, mais les différences de performances entre les modèles sont moins prononcées dans ce cas.

- Review + Word_Count

Modèle	Précision
Régression Logistique (BOW)	0.81
Régression Logistique (TF-IDF)	0.77
Régression Logistique (BOW bigram)	0.81
Régression Logistique (TF-IDF bigram)	0.72
SVM (BOW)	0.59
SVM (TF-IDF)	0.51
SVM (BOW bigram)	0.61
SVM (TF-IDF bigram)	0.51
Forêt Aléatoire (BOW)	0.74
Forêt Aléatoire (TF-IDF)	0.70

Modèle	Précision
Forêt Aléatoire (BOW bigram)	0.77
Forêt Aléatoire (TF-IDF bigram)	0.70

Une fois de plus, la régression logistique avec la vectorisation BOW bigram donne la meilleure précision, suivie de près par la régression logistique avec la vectorisation BOW et la forêt aléatoire avec la vectorisation BOW bigram. Les performances des modèles varient en fonction du type de vectorisation utilisé.

- Usage of multiple Review
 - Bow for word and char
 - Régression Logistique avec vectoriseur de mots et caractère : 0.77
 - SVM avec vectoriseur de mots et caractère: 0.65
 - Forêt Aléatoire avec vectoriseur de mots et caractère : 0.71
 - Bow for word and char ngram(1,2) and ngram (1,5)
 - Régression Logistique avec vectoriseur de mots : 0.85
 - SVM avec vectoriseur de mots : 0.72
 - Forêt Aléatoire avec vectoriseur de mots : 0.77
 - Bow for word and char ngram(1,2) and ngram (2,5)
 - Régression Logistique avec vectoriseur de mots : 0.84
 - SVM avec vectoriseur de mots : 0.71
 - Forêt Aléatoire avec vectoriseur de mots : 0.80

Les performances des modèles avec des n-grammes de taille plus grande (par exemple, (1,5)) sont meilleures que celles avec des n-grammes plus petits (par exemple, (1,2)). Cela indique que la prise en compte de séquences de mots plus longues peut capturer des informations plus riches et conduire à de meilleures prédictions.

La régression logistique semble être le modèle le plus performant dans la plupart des cas, suivie de la forêt aléatoire. Le SVM obtient généralement les performances les plus faibles, bien qu'il puisse être sensible aux réglages des hyperparamètres.

Conclusion:

La régression logistique s'est avérée être le choix le plus fiable parmi les modèles testés, offrant des performances solides et cohérentes sur une variété de techniques de vectorisation et de caractéristiques. L'utilisation de bigrammes a amélioré la précision des prédictions. L'ensemble des feature ajoutés n'a fait que rajouté du bruit et complexifier le modèle.

4. Covid

Dans ce dernier exercice nous allons travailler sur un DataSet du covid 19.

Le jeu de données brut contient un grand nombre d'informations relatives aux patients anonymisés, y compris les préconditions. Le jeu de données brut comprend 21 caractéristiques différentes et 1 048 576 patients uniques. Dans les caractéristiques booléennes, 1 signifie "oui" et 2 signifie "non". Les valeurs 97 et 99 sont des données manquantes.

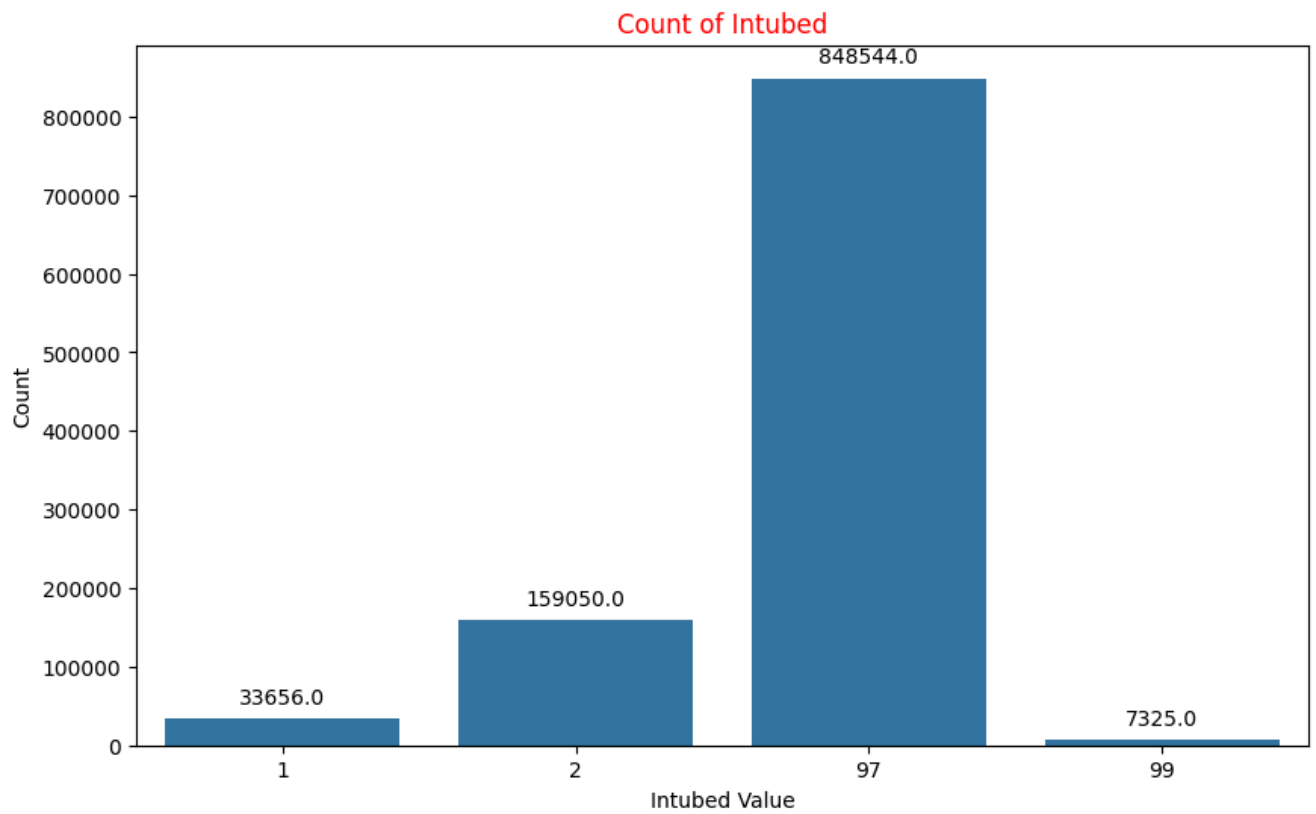
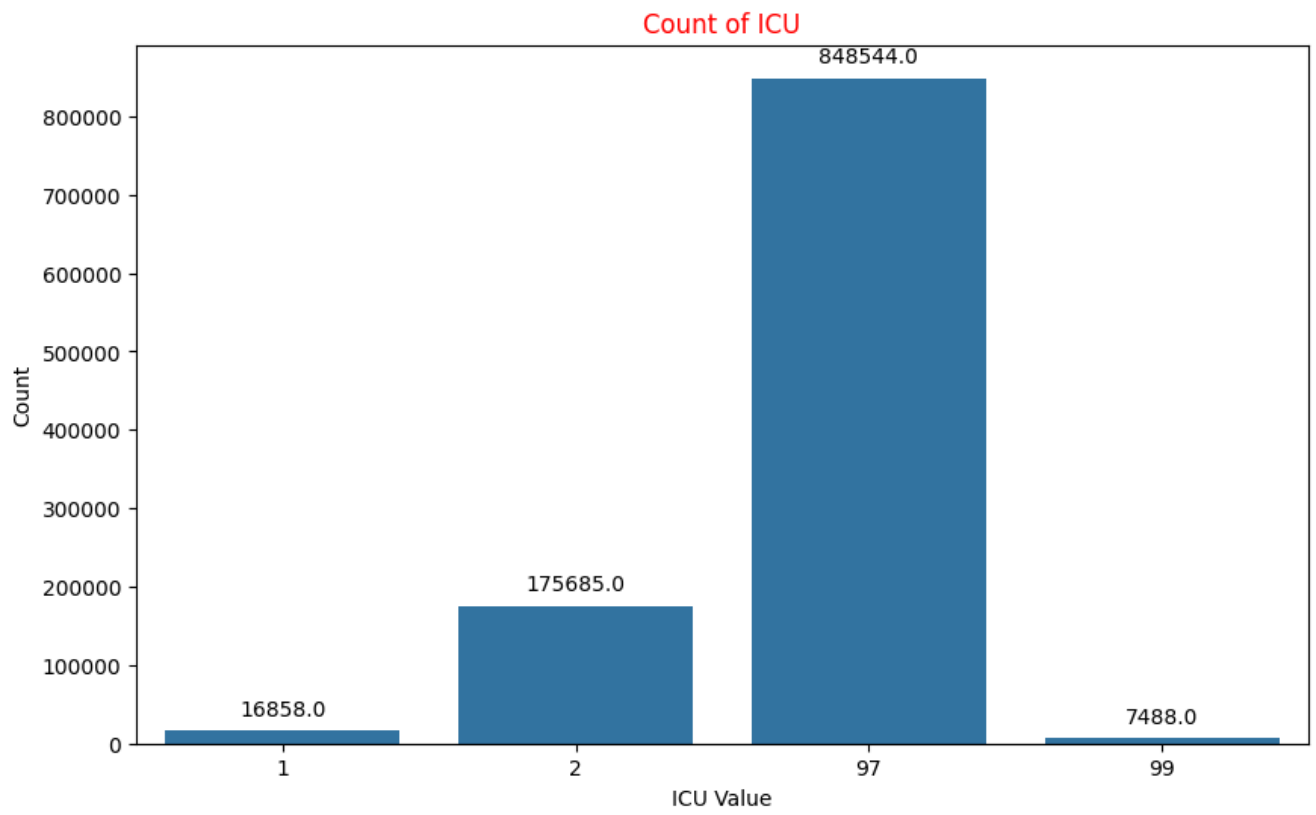
Feature	Description (French)
MEDICAL_UNIT	Type d'institution du système national de santé ayant fourni les soins.
SEX	1 - féminin. 2 - masculin
PATIENT_TYPE	Type de soin reçu par le patient dans l'unité. 1 pour un retour à domicile et 2 pour une hospitalisation.
DATE_DIED	Si le patient est décédé, indique la date de décès, et 9999-99-99 sinon.
INTUBED	Indique si le patient était relié au ventilateur.
PNEUMONIA	Indique si le patient a déjà une inflammation des sacs aériens ou non.
AGE	Âge du patient.
PREGNANT	Indique si le patient est enceinte ou non.
DIABETES	Indique si le patient est diabétique ou non.
COPD	Indique si le patient est atteint de bronchopneumopathie chronique obstructive ou non.
ASTHMA	Indique si le patient est asthmatique ou non.
INMSUPR	Indique si le patient est immunodéprimé ou non.
HIPERTENSION	Indique si le patient est hypertendu ou non.
OTHER_DISEASE	Indique si le patient a une autre maladie ou non.
CARDIOVASCULAR	Indique si le patient a une maladie cardiaque ou vasculaire.

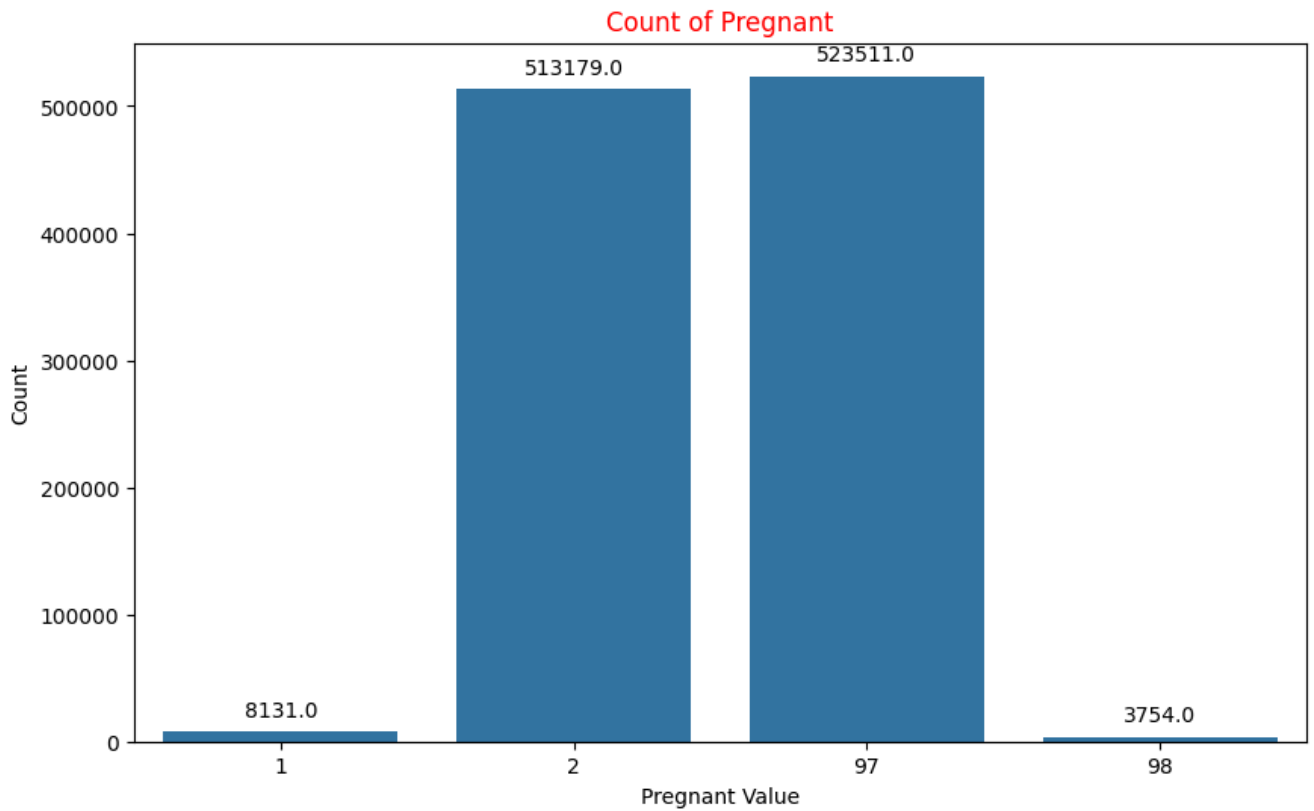
Feature	Description (French)
OBESITY	Indique si le patient est obèse ou non.
RENAL_CHRONIC	Indique si le patient souffre d'insuffisance rénale chronique ou non.
TOBACCO	Indique si le patient est un utilisateur de tabac.
CLASIFFICATION_FINAL	Résultats du test Covid. Voir la description des données.
ICU	Indique si le patient a été admis en unité de soins intensifs.

Traitement des données

Nous allons commencé par traiter les données manquantes ainsi que la création d'une colonne a prédire "DEAD".

- **Données Manquantes**
 - On va commencé par voir les valeurs uniques de chaque colonne.
On peut on déduire que pour cetrtaine colonne dans les quelles on expecte avoir uniquement 2 valeurs, on retrouve 3 à 4 comme par exemple PNEUMONIA, TABACO. Pour palier a cela nous allons garder uniquement les lignes avec les valeurs 1 et 2.
 - On analysant le compte de chaque valeur unique, on remarque que les variables ICU, PREGNANT et INTUBED présente plusieurs valeurs unique. On décidras de ce passer de ces dérnieres.



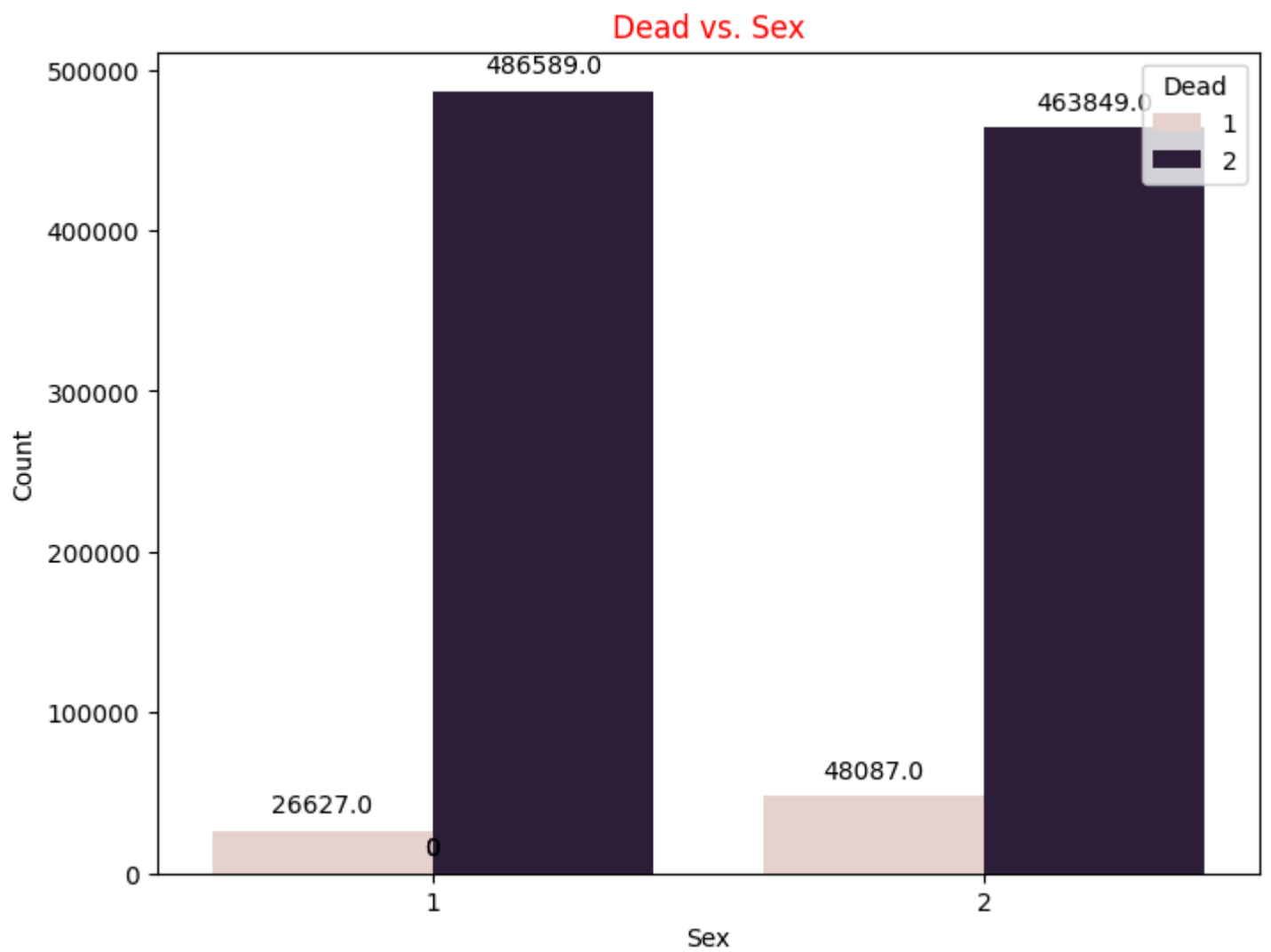


- La colonne "DATE_DIED" possède des dates réel ou la personne est morte, ainsi que la valeur 9999-99-99, dans ce cas la la personne a survécu. Une colonne 2 DEAD est créée, 2 pour les personne ayant survécu et 1 une personne morte.

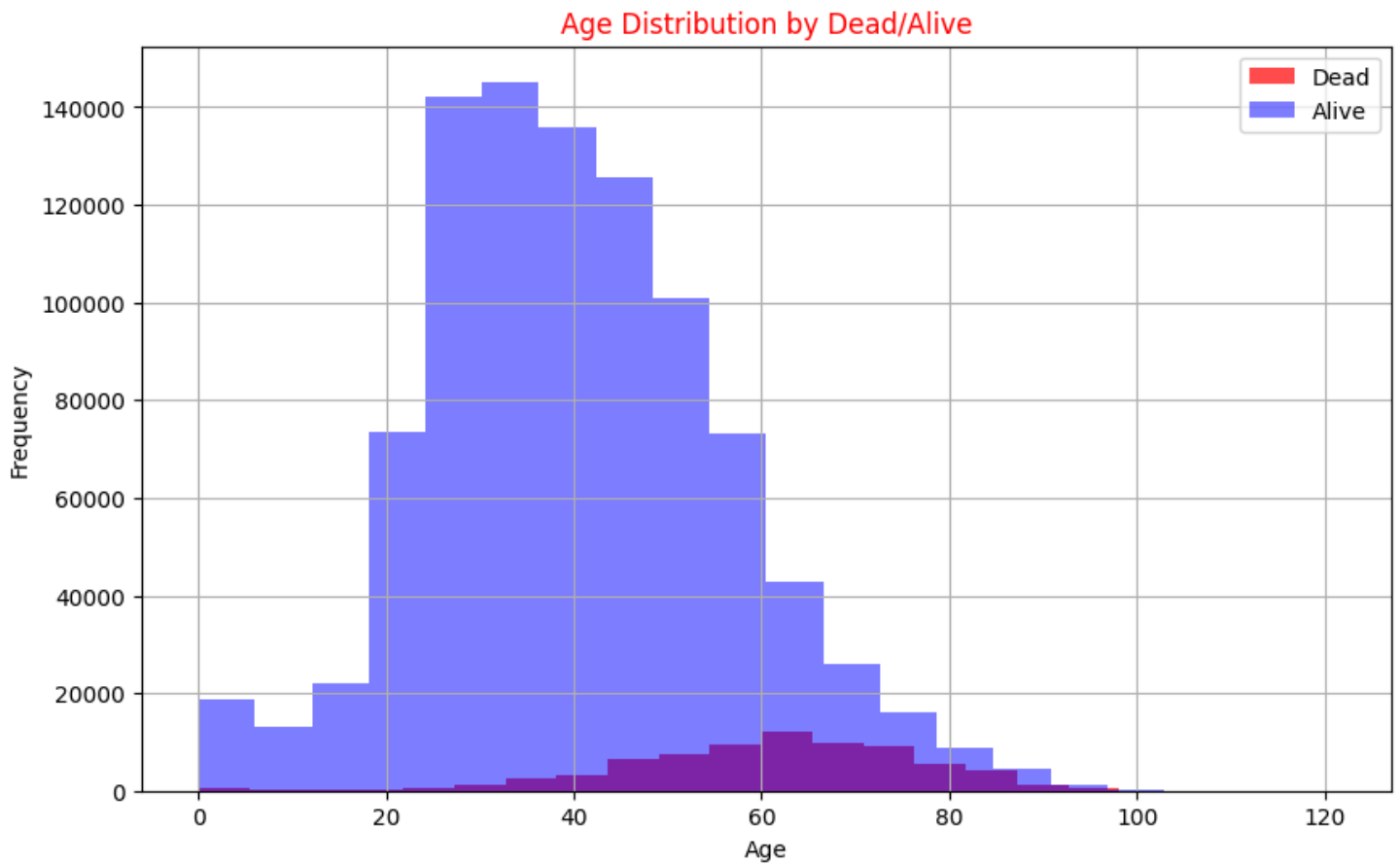
Data Visualisation

Nous allons analyser nos données grace des graphe de distribution.

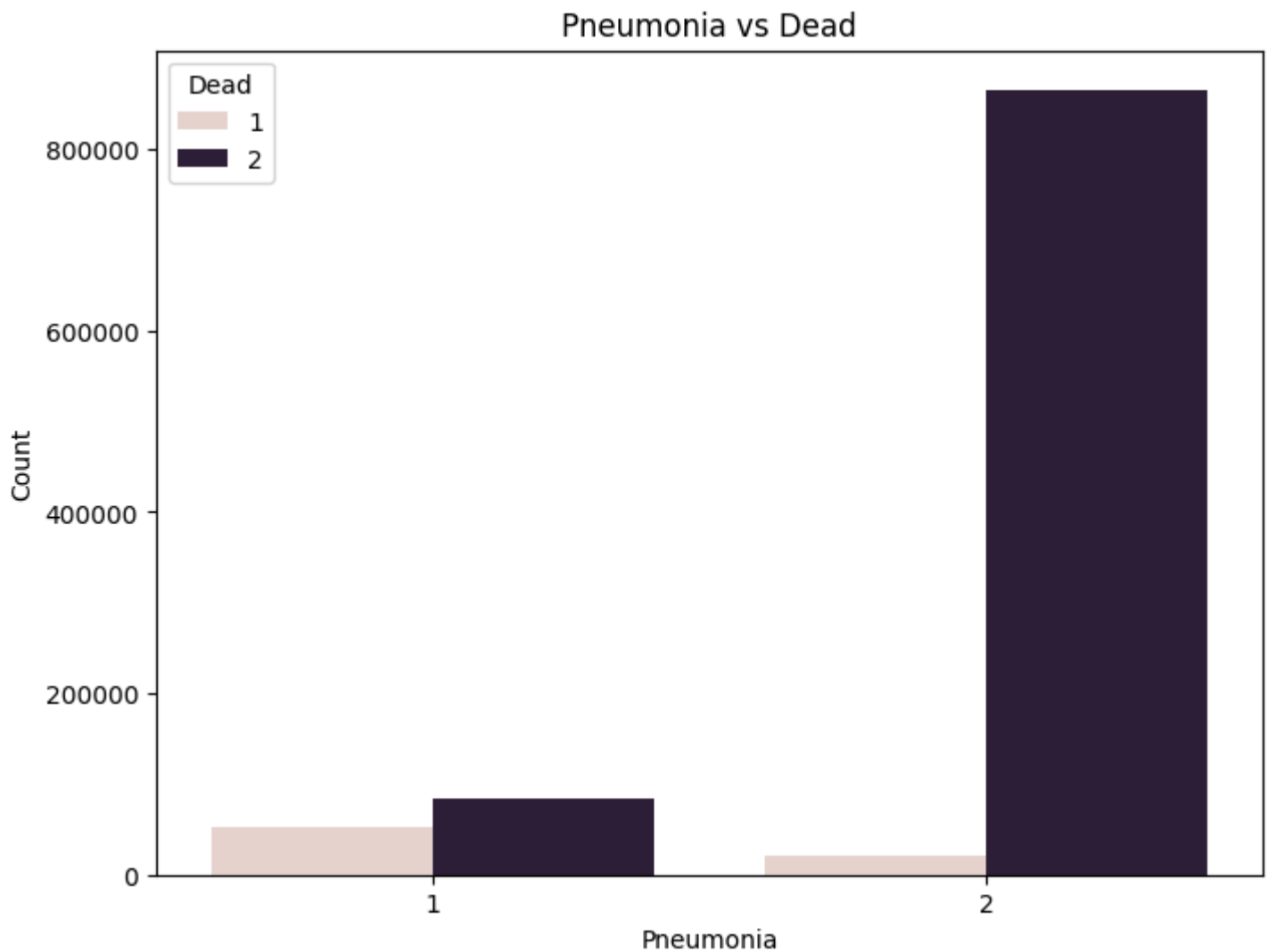
- Distribution de la variable DEAD et le sex de la personne : on remarque que plus d'homme sont mort que de femme.



- Distribution de la variable DEAD et l'age: Avec l'augmentation de l'age, on remarque plus de personne mourante.



- Distribution de la variable DEAD et PNEUMONIA : on remarque que les personnes ayant une pneumonie sont plus susceptibles de mourir.



Modelisation.

On commencera par analyser la matrice de confusion, afin d'obtenir les variables les plus significatif.

Les variable qui n'aposte rien sont : "SEX","COPD","ASTHMA","INMSUPR",
"OTHER_DISEASE","CARDIOVASCULAR","OBESITY","TOBACCO".

- Used models : Logistic Regression, Decision Tree and Random Forest
 - All features
 - Logistic Regression: 0.9346
 - Decision Tree: 0.9323
 - Random Forest: 0.9326
 - Feature Set 1: ["AGE", "PATIENT_TYPE", "PNEUMONIA"]
 - Logistic Regression: 0.9337
 - Decision Tree: 0.9338
 - Random Forest: 0.9338
 - Feature Set 2: ["AGE", "PATIENT_TYPE", "HIPERTENSION", "PNEUMONIA", "DIABETES"]

- Logistic Regression: 0.9338
- Decision Tree: 0.9337
- Random Forest: 0.9339

Dans l'ensemble, tous les modèles ont obtenu des précisions similaires, allant d'environ 93,37 % à 93,39 %. Cela suggère que le choix de l'ensemble de fonctionnalités n'a pas d'impact significatif sur les performances des modèles. Cependant, il est important de noter que les précisions sont relativement élevées, ce qui indique que les modèles se comportent bien dans la prédiction de la variable cible.