Machine Learning

Introduction

“ Nous sommes une entreprise spécialisée en développement de solution IA et nous sommes invités à développer un Chatbot, intelligent, sous forme d'interrogatoire médical et permettant de poser un premier conseil médical à distance.”

Pour cela nous avons 3 axes de travail:

- Création et entraînement d’un **modèle**
- Création d’un **Chatbot**
- Création d’une **carte dynamique**

Pour cela, nous avons utilisé un fichier CSV nommé **CovidDataset.csv.**

Les données contiennent **divers symptômes** et le **résultat qui en résulte**, c'est-à-dire si la personne est atteinte ou non du COVID.
Elles ont été recueillies à partir de données issues d’un hôpital indien en mai 2020 selon les symptômes à surveiller décrits par l’OMS.

Il est composé des colonnes de divers symptômes et le résultat qui en résulte:
- Colonnes avec les différents symptômes: Yes/No
- COVID-19: Yes/No

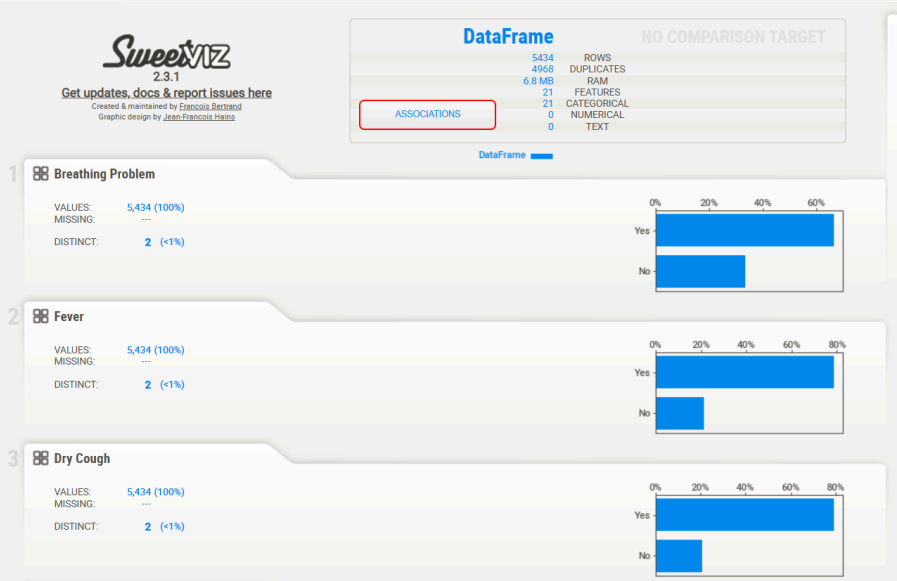
Préparation et visualisation des données

Dans un premier temps, nous avons voulu comprendre et analyser les données du fichier CSV.
Pour cela, nous avons dû:

- Transformation du fichier CSV en **DataFrame**
- **Analyse du DataFrame** et du contenu dans les différentes colonnes
- Détection de **valeurs nulles**

Après cette analyse, nous avons constaté que deux colonnes nommés “Wearing Masks” et “Sanitazation from Market” sont à supprimer car elles contiennent que des “NaN”
De plus, nous avons conclus que notre colonne cible se nomme “COVID-19”.

Voici un exemple de visualisation qui nous avons fait pour pouvoir nous aider à la compréhension des données:



Concernant le pré-processing des données:

- Utilisation de **LabelEncoder()** pour gérer les catégories au format texte.
- Création d'un dataframe avec **colonnes d'entraînement**
- Création d'un dataframe avec **colonne cible**
- Triage des colonnes par **corrélations** avec COVID-19.
  Cela nous a permis de détecter les colonnes qui ont le plus d'impact.

```
Colonnes et valeurs de corrélation avec la col covid :
COVID-19                                  1.000000
Sore throat                               0.502848
Dry Cough                                 0.464292
Abroad travel                             0.443875
Breathing Problem                         0.443764
Attended Large Gathering                  0.390145
Contact with COVID Patient                0.357122
Fever                                     0.352891
Family working in Public Exposed Places   0.160208
Visited Public Exposed Places             0.119755
Hyper Tension                             0.102575
Asthma                                    0.089930
Diabetes                                  0.040627
Heart Disease                             0.027072
Gastrointestinal                         -0.003367
Running Nose                             -0.005657
Headache                                 -0.027793
Fatigue                                  -0.044188
Chronic Lung Disease                     -0.056837
Wearing Masks                                  NaN
Sanitization from Market                       NaN
Name: COVID-19, dtype: float64
```

Choix et entraînement du modèle

RandomForestClassifier

- Capacité à gérer des données complexes avec une précision élevée : Le Random Forest est reconnu pour sa grande efficacité à **gérer des données complexes**, on ne compte plus le nombre d'articles scientifiques qui le prennent en compte.

- Interprétabilité des résultats : L'arbre de décision permet des **outils intéressant d'interprétation** comme la features importance.

- Adaptabilité d'utilisation pour notre dataset : Le Random Forest est connu pour **ne pas gérer les données manquantes** et justement notre dataset n'en contient pratiquement pas.

Adaptation du nombre de colonne

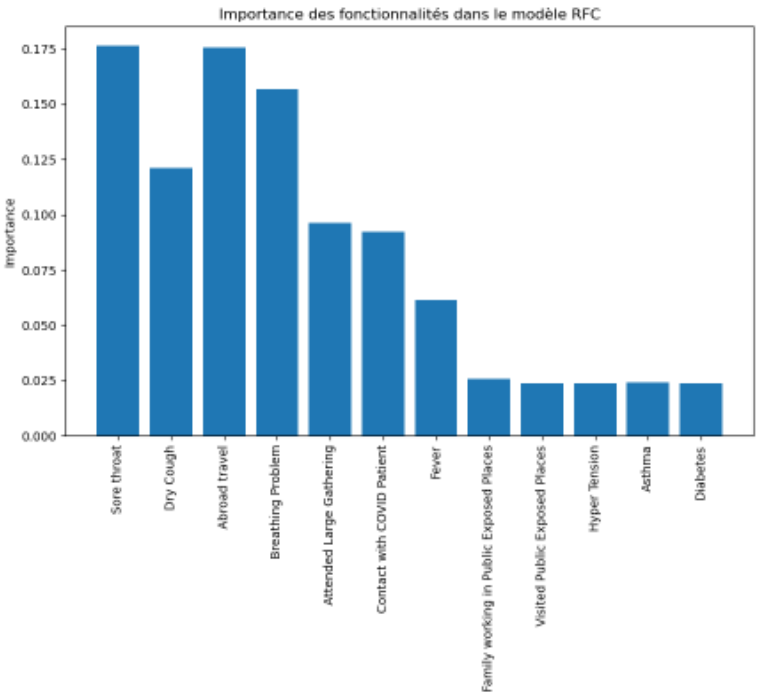
Pour notre modèle, on a donc gardé les **12 caractéristiques les plus corrélées** à la cible pour l'entraînement car en faisant des tests on s'est aperçu que notre score de prédiction n'évoluait plus avec ou sans les trois caractéristiques les moins corrélées.

Amélioration des hyperparamètres

Pour le choix des hyperparamètres, on a effectué un **GridSearchCV** en faisant varier n_estimators, criterion et max_features. On arrive au résultat suivant pour le meilleur paramétrage :

```
Best parameters found: {'criterion': 'gini', 'max_features': 'sqrt', 'n_estimators': 20}
```

Dans ce schémas les variables (features) sont **mis dans l'ordre de corrélation avec la variable cible**. On voit ainsi la **différence entre la corrélation et l'importance** que le modèle va accorder aux différentes variables dans la prédiction. Il est intéressant de noter qu'il y a un lien clair mais des différences peuvent cependant apparaître.



Il convient de rappeler que ce dataset date de Mai 2020. On retrouve l'importance dans la contamination lors d'un voyage à l'étranger.

Pour une prédiction sur des cas actuels, **un dataset d'entraînement plus récent semble indispensable**.

Evaluation du modèle

Interprétation des résultats du modèle

L'accuracy

L'accuracy mesure l'**efficacité d'un modèle à prédire correctement à la fois des individus classés positivement et négativement de façon symétrique.**

Elle est calculée à partir du rapport entre le nombre d'individus bien classé et le nombre total d'individus dans l'échantillon.

Son résultat est un nombre décimal compris entre 0 et 1, souvent exprimé en pourcentage, où 0 signifie que le modèle ne sait pas classer les individus et 1 où le modèle classe parfaitement les individus.

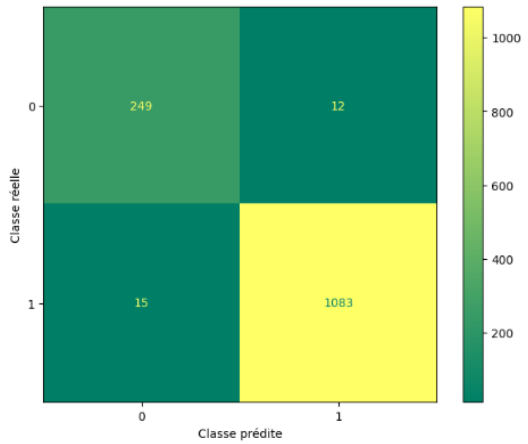
```
[ ]  from sklearn.metrics import accuracy_score
     accuracy_score(y_test, y_pred)

     0.9801324503311258
```

Notre modèle obtient **98%** de bonnes prédictions.

La matrice de confusion

La matrice de confusion est une matrice qui mesure la qualité d'un système de classification. Chaque ligne correspond à une classe réelle, chaque colonne correspond à une classe estimée.




```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)

[[ 249   12]
 [  15 1083]]
```

249 vrai négatif : la prédiction est négative et c'est la réalité (le modèle prédit que le patient n'a pas le covid et c'est vrai)

12 faux positif : la prédiction est positive mais ce n'est pas la réalité (le modèle prédit que le patient a le covid alors qu'il ne l'a pas)

15 faux négatif : la prédiction est négative mais ce n'est pas la réalité (le modèle prédit que le patient n'a pas le covid alors qu'il l'a)

1083 vrai positif : la prédiction est positive et c'est la réalité (le modèle prédit que le patient a le Covid et il l'a vraiment)

Le rapport de classification

Le rapport de classification nous permet d'**obtenir plusieurs mesures** :

La **précision** : c'est la proportion de prédictions correctes parmi les individus que l'on prédit positif.

Le **recall** : ou sensitivity correspond au nombre d'individus correctement classés

le **f1-score** : c'est la moyenne harmonique de la précision et du recall

support : c'est le nombre total d'échantillons appartenant à cette classe dans l'ensemble de données.

```
from sklearn.metrics import classification_report
classe = classification_report(y_test, y_pred)
print(classe)
```

```
              precision    recall  f1-score   support

           0       0.94      0.95      0.95       261
           1       0.99      0.99      0.99      1098

    accuracy                           0.98      1359
   macro avg       0.97      0.97      0.97      1359
weighted avg       0.98      0.98      0.98      1359
```

Le modèle a un f1-score de 0.95 sur la classe 0, ce qui signifie qu'il classe les individus non malades avec une précision **95%**.

Le modèle a un f1-score de 0.99 sur la classe 1, cela montre qu'il classe les individus non malades avec une précision de **99%**.

En conclusion, notre modèle est **très performant pour détecter les personnes malades**.

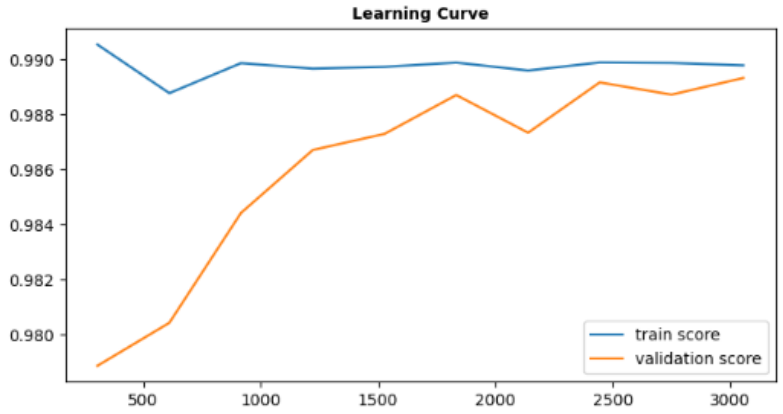
La learning curve

Elle **trace l'évolution des performances du modèle** sur l'ensemble d'apprentissage et sur l'ensemble de validation en fonction du nombre d'échantillons utilisés pour l'apprentissage.

- Axe des x (abscisse) : Il représente le nombre d'échantillons utilisés pour l'apprentissage. Plus précisément, il s'agit de la taille de l'ensemble d'apprentissage
- Axe des y (ordonnées) : Il représente la performance du modèle.
- Courbe d'apprentissage (train score) : Cette courbe montre la performance du modèle sur l'ensemble d'apprentissage
- Courbe de validation (validation score) : Cette courbe montre la performance du modèle sur l'ensemble de validation en fonction de la taille de l'ensemble d'apprentissage.

Dans notre cas, on arrive à une sorte de seuil à partir de 2500 itérations mais le score de validation continue à augmenter légèrement. Une augmentation du nombre de données pourrait être utile.

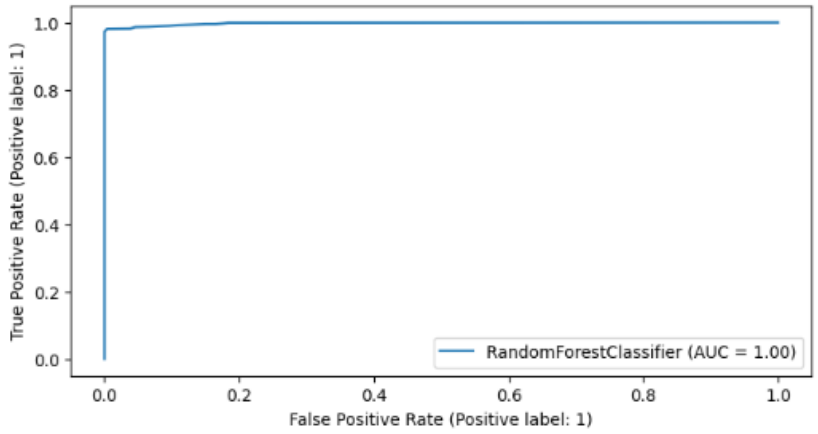
On peut voir qu'il y **peu de surapprentissage**, la courbe monte presque uniformément.



La Roc Curve ou courbe ROC

Elle **illustre l'aire sous la courbe** (Area Under the Curve). L'aire sous la courbe représente la **mesure de séparabilité du modèle**. Elle nous indique à quel point le modèle peut faire la distinction entre deux classes.

Plus l'AUC est élevé, plus le modèle est en **mesure de détecter correctement les classes.**



Nous obtenons une aire sous la courbe de 1, ce qui signifie que le modèle est parfaitement capable de faire la distinction entre une personne atteinte du Covid-19 et une personne en bonne santé.

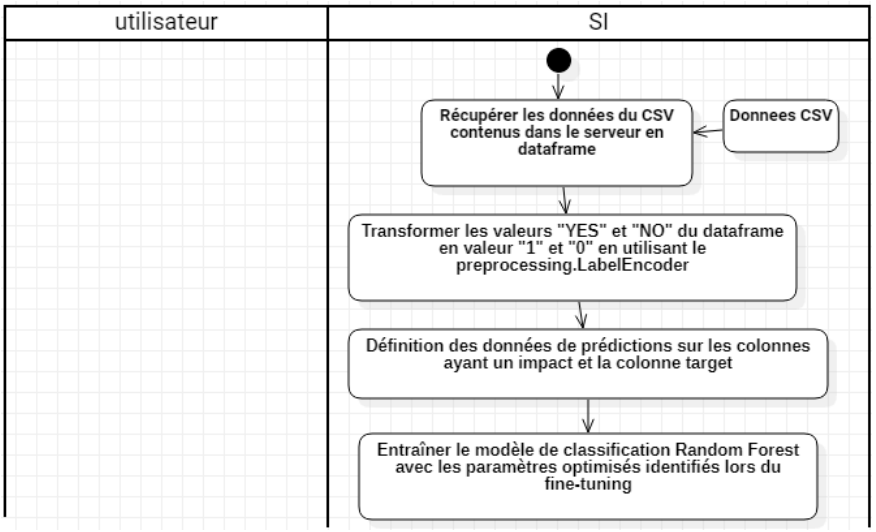
Chatbot

Nous avons pris la décision de déployer le Chatbot sur un serveur Flask, offrant une **plateforme flexible et facile à utiliser**, afin de créer nos points d'APIs permettant la communication entre l'utilisateur et le système d'information.

Le fonctionnement du serveur fonctionne en deux temps.

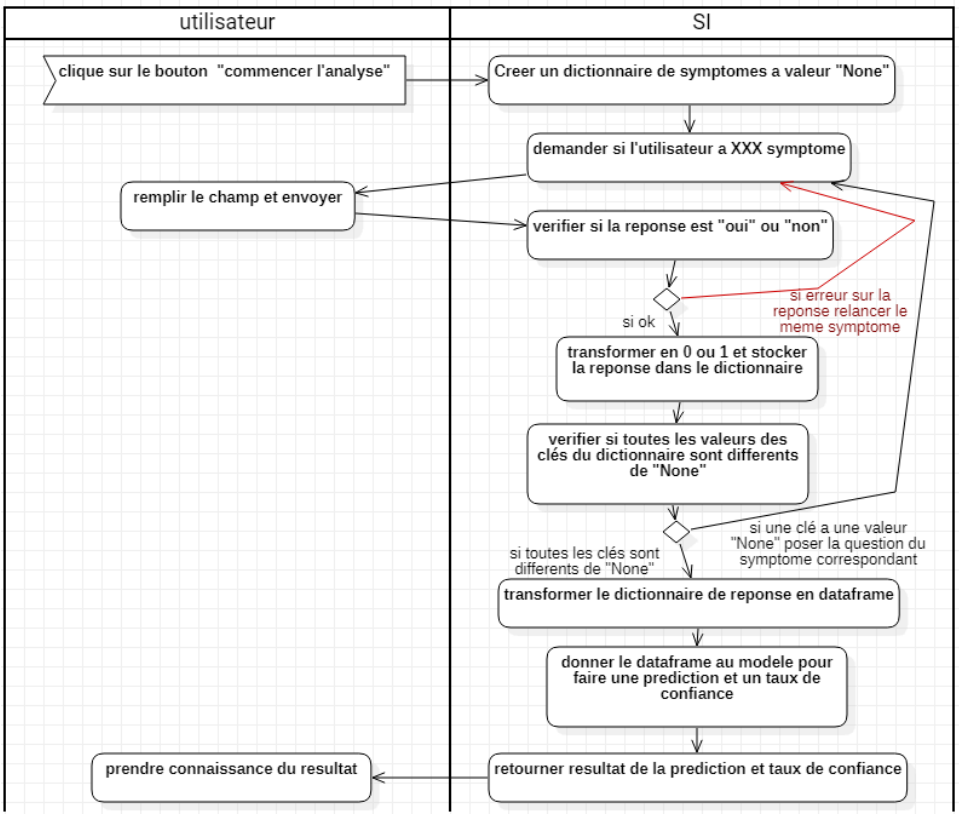
- Le premier temps est réalisé lors du lancement du serveur Flask.
- Le second temps est réalisé lorsque l'utilisateur décide d'interagir avec le chatbot.

Voici les différentes intéractions qui se produisent lors du lancement du serveur Flask.



1. En premier, le système d'information récupère les données dans un csv contenu à l'intérieur du serveur flask. Ces données sont ensuite transformées en dataframe.
2. Les données "yes/no" sont ensuite transformés en valeur "1/0" en utilisant le LabelEncoder
3. Une séparation entre la target et les données d'inputs est réalisée afin de n'utiliser que les 12 colonnes ayant un impact sur le résultat.
4. Le modèle Random Forest Classifier est ensuite appliqué avec l'ajustement des hyperparamètres.
5. Le modèle est ensuite entraîné sur les données d'inputs et la target, afin de conserver le modèle entraîné durant toute la durée de vie du serveur.

Voici les différentes intéractions qui se produisent lorsqu' un utilisateur décide d'interagir avec le chatbot.



1. L'utilisateur signale au système d'information qu'il souhaite utiliser le chatbot en cliquant sur le bouton "commencer l'analyse".
2. Le système crée un dictionnaire contenant comme clés les différents symptômes, ayant pour chacun une valeur nulle.
3. Le système va ensuite demander à l'utilisateur si il possède ce symptôme.
4. L'utilisateur répond, puis envoie sa réponse au système d'information.
5. Le système va vérifier que le contenu de la réponse est "oui" ou "non". Si ce n'est pas le cas, il repose la même question avec le même symptôme. Si c'est le cas, la réponse sera transformer en valeur 1 ou 0, puis assigné à la valeur de la clé représentant ce symptôme.
6. Le système vérifie ensuite les différentes valeurs des clés du dictionnaire. Si un symptôme contient encore une valeur nulle, le système va demander à l'utilisateur si il possède ce symptôme. Si toutes les clés contiennent un résultat, le système transforme le dictionnaire en dataframe.
7. Le système fournit ensuite le dataframe au modèle Random Forest Classifier afin de réaliser une prédiction et un taux de confiance.
8. Le résultat est ensuite retourner à l'utilisateur

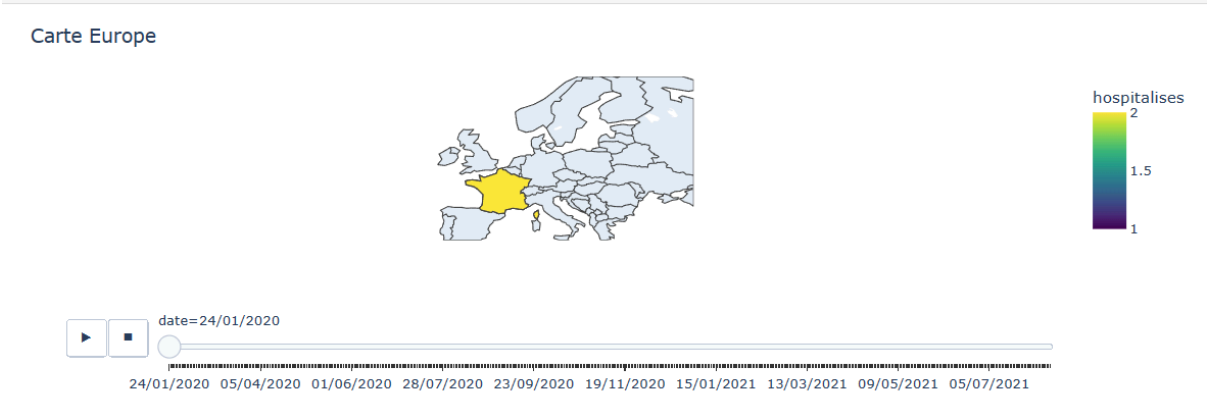
Carte dynamique (bonus)

Afin d'obtenir une meilleure visualisation des cas Covid, nous avons réalisé **différentes cartes afin de déceler des clusters**.

La première carte est réalisée en faisant un **merge d'un dataframe** contenant les cas Covid ayant une temporalité avec un autre dataframe contenant les différentes régions de france.
Ce nouveau dataframe est ensuite fourni à la librairie **Plotly express**.

Cette librairie permet de **créer une carte** afin de visualiser l'évolution des cas Covid à travers le temps en utilisant la **barre de navigation latérale**. Cependant, il n'est pas possible d'obtenir de carte plus détaillée que celle de l'europe sur cette librairie.

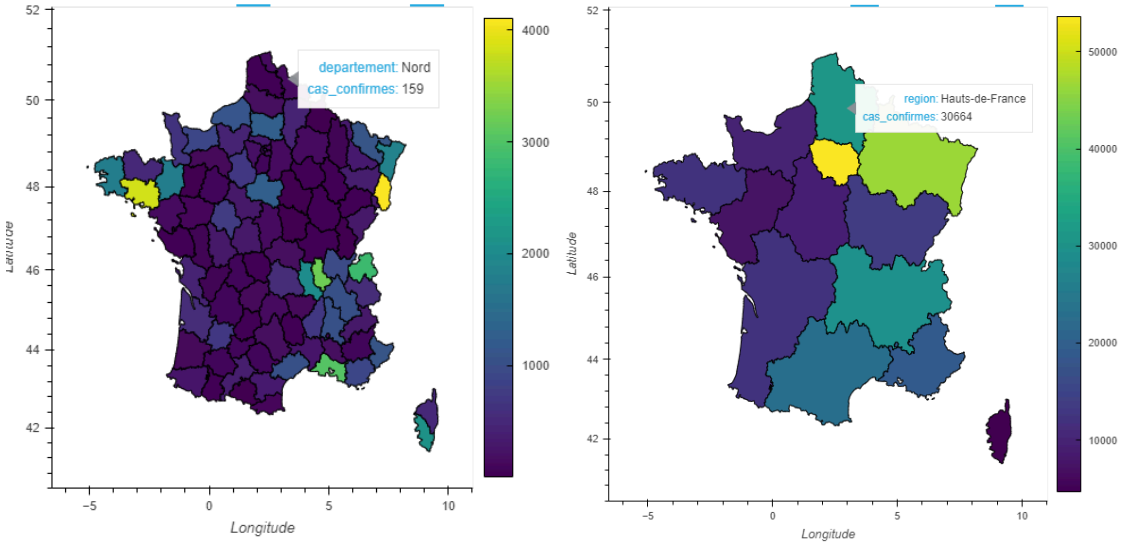
L'ensemble des informations pour chaque région sont donc rassemblés en tant que pays "france", ce qui crée **une perte d'information importante**.



La seconde carte est réalisée en utilisant un dataframe contenant les **différents cas Covid par régions et par départements**.

Il est ensuite mergé sur un second dataframe, contenant des **données de type “Polygon”** pour chaque région et département, ce qui permet de dessiner chaque région sur une carte.

La librairie **geoview.polygon** est ensuite utilisée afin de faire apparaître la carte de France par département et par région en utilisant ces données polygon.



Conclusion

Le modèle de prédiction semble **très efficace** mais on a noté quelques points d'amélioration:

- Il nous faudrait un **dataframe plus récent**. L'importance dans la prédiction de la maladie du voyage à l'étranger est une parfaite illustration de ce problème.

- Une autre **limite du dataframe est qu'il provient d'un seul hôpital** en Inde. Il conviendrait d'étendre le questionnaire à d'autres zones géographiques afin d'éliminer d'éventuels biais créés par l'environnement.

- Un **débat métier semble nécessaire** sur l'efficacité du dépistage COVID en fonction de la méthode utilisée (ici non précisée) qui pourrait introduire un biais sur la variable cible.
  Une méthode pour contourner ce problème pourrait être d'augmenter le nombre de données en diversifiant les sources de dépistages tout en créant une variable "méthode de dépistage".
  Une autre stratégie serait d'être très conservateur dans nos sources de dépistage et de ne garder que les méthodes identifiées comme les plus fiables par les experts métier.

- Le Chatbot est **spécifiquement développé à notre cas d'usage**. On pourrait imaginer une nouvelle utilisation dans des domaines différents.