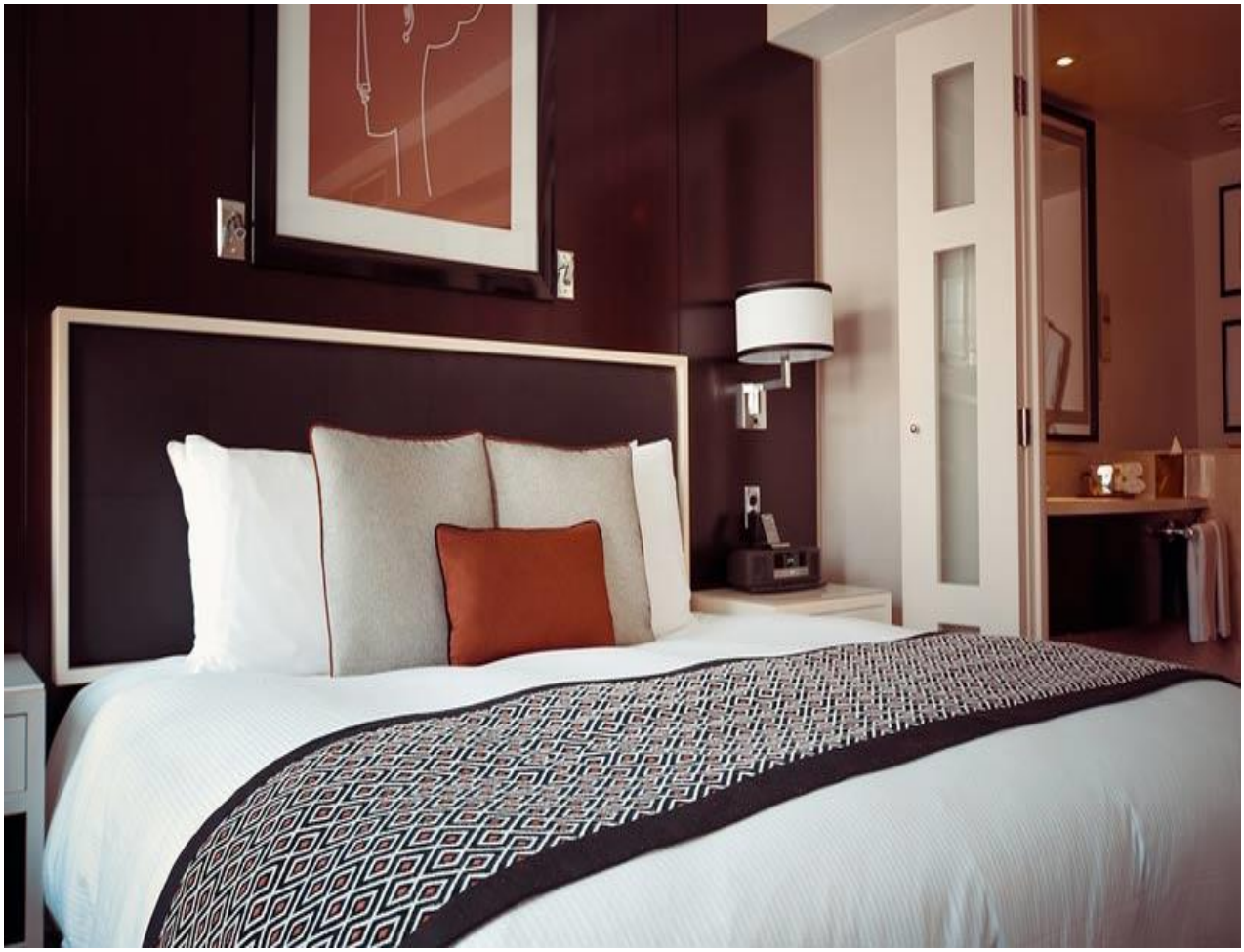
Gestion de Réservation de Chambres d'Hôtel grâce à l'Intelligence Artificielle



Oumar Niang : *Titre professionnel "Développeur en intelligence artificielle" de niveau 6 enregistré au RNCP sous le n°34757 · Passage par la voie de la formation*

- *parcours de 19 mois achevé le 20 octobre 2020*
- *Simplon-Microsoft-Orange 2020*

SOMMAIRE

1. Contexte et le projet :

Nous disposons d'un jeu de données composé de deux ensembles de données sur la demande d'hôtels. Nous avons deux types d'hôtels : (H1) est un hôtel de villégiature et l'autre est un hôtel de ville (H2).

Les deux ensembles de données partagent la même structure, avec **32** variables décrivant les **40 060** observations de H1 et **79330** observations de H2.

Chaque observation représente une réservation d'hôtel. Les deux ensembles de données comprennent les réservations qui doivent arriver entre le 1er juillet 2015 et le 31 août 2017, y compris les réservations effectivement arrivées et les réservations annulées.

Puisqu'il s'agit de données réelles de l'hôtel, tous les éléments de données concernant l'hôtel ou l'identification du client ont été supprimés.

Notre dataset contient des informations de réservation pour un hôtel de ville et un hôtel de villégiature, ainsi que des informations telles que la date de la réservation, la durée du séjour, le nombre d'adultes, d'enfants et / ou de bébés et le nombre de places de parking disponibles, entre autres.

a) Le client & Besoin :

Au regard de nombre de réservations et annulations reçues sur une période donnée pour un établissement hôtelier, et d'autres informations disponibles sur la base de données du client.

Le client souhaite mettre en place un système automatisé permettant de prédire les réservations et les annulations 24h en avance afin de mieux gérer ses employés et d'autres dépenses.

Autrement dit il veut savoir si une réservation sera aboutie ou annulée.

Le client souhaite aussi savoir davantage sur les profils de clients afin d'affiner ses offres pour les années à avenir.

b) La solution :

Pour répondre à son besoin nous allons d'abord récupérer ses données et construire une base de données adaptée, après cette étape nous procédons à l'exploration et l'analyse de données.

Puis nous construisons un modèle de prédiction, ce modèle sera déployé sous forme d'une application web.

c) L'organisation :

Dans ce projet nous allons dans un premier temps récupérer le dataset, vérifier son intégrité et nettoyer afin d'analyser à l'aide des outils adaptés. Une fois cette étape terminée nous allons insérer ces données dans une base de données MySQL.

Puis nous construisons un modèle d'intelligence artificielle qui sera déployé en ligne, ce sera dans une application interactive.

Pour s'organiser et suivre l'avancement du projet, nous utilisons la méthode agile à l'aide de l'outil Trello.

2. Les données & Traitement

a) La source de données :

Ces données proviennent d'une base de données de deux hôtels situés au Portugal: H1 dans la région balnéaire de l'Algarve et H2 dans la ville de Lisbonne.

Ci-dessous le lien pour retrouver les données brutes.

https://www.sciencedirect.com/science/article/pii/S2352340918315191

https://www.kaggle.com/jessemostipak/hotel-booking-demand

Notre dataset est composé de 32 variables dont 10 sont qualitatives, la variable dépendante (à expliquer) ici est « **is_canceled** », ci-dessous la liste avec la définition de toutes variables :

Variable	Type	Description
1. *hotel*	*Catégorielle*	*Hôtel (H1 = Resort Hotel ou H2 = City Hotel)*
2. *is_canceled*	*Catégorielle*	*Valeur indiquant si la réservation a été annulée (1) ou non (0)*
3. *lead_time*	*Continue*	*Nombre de jours écoulés entre la date d'entrée de la réservation dans le PMS et la date d'arrivée*
4. *arrival_date_year*	*Date*	*Année de la date d'arrivée*
5. *arrival_date_month*	*Catégorielle*	*Mois de la date d'arrivée*
6. *arrival_date_week_number*	*Catégorielle*	*Numéro de semaine de l'année pour la date d'arrivée*
7. *arrival_date_day_of_month*	*Catégorielle*	*Jour de la date d'arrivée*
8. *stays_in_weekend_nights*	*Catégorielle*	*Nombre de nuits le week-end (samedi ou dimanche) que le client a séjourné ou réservé à l'hôtel*
9. *stays_in_week_nights*	*Catégorielle*	*Nombre de nuits en semaine (du lundi au vendredi) que le client a séjourné ou réservé à l'hôtel*
10. *adults*	*Discrète*	*Nombre d'adultes*
11. *children*	*Discrète*	*Nombre d'enfants*
12. *babies*	*Discrète*	*Nombre de bébés*
13. *meal*	*Catégorielle*	*Type de repas réservé. Les catégories sont présentées dans les forfaits repas standard: Indéfini / SC - pas de*
14. *country*	*Catégorielle*	*Pays d'origine.*
15. *market_segment*	*Catégorielle*	*Désignation du segment de marché. Dans les catégories, le terme «TA» signifie «Agents de voyage» et «TO» signifie*
16. *distribution_channel*	*Catégorielle*	*Canal de distribution des réservations. Le terme «TA» signifie «Agents de voyage» et «TO» signifie «Tour Operators»*

17. *is_repeated_guest*	*Continue*	*Valeur indiquant si le nom de la réservation provenait d'un invité répété (1) ou non (0)*
18. *previous_cancellations*	*Discrète*	*Nombre de réservations précédentes qui ont été annulées par le client avant la réservation en cours*
19. *previous_bookings_not_canceled*	*Discrète*	*Nombre de réservations précédentes non annulées par le client avant la réservation en cours*
20. *reserved_room_type*	*Catégorielle*	*Code du type de chambre réservé. Le code est présenté à la place de la désignation pour des raisons d'anonymat.*
21. *assigned_room_type*	*Catégorielle*	*Code du type de chambre affecté à la réservation. Parfois, le type de chambre attribué diffère du type de chambre*
22. *booking_changes*	*Discrète*	*Nombre de modifications / modifications apportées à la réservation à partir du moment où la réservation a été saisie sur le PMS*
23. *deposit_type*	*Continue*	*Indication si le client a effectué un acompte pour garantir la réservation. Cette variable peut prendre trois catégories: aucun dépôt - aucun dépôt n'a été effectué; Non remboursement - un acompte a été effectué pour la valeur du coût total du séjour; Remboursable - un dépôt a été effectué avec une valeur inférieure au coût total du séjour*
24. *agent*	*Catégorielle*	*ID de l'agence de voyage qui a effectué la réservation*
25. *company*	*Catégorielle*	*ID de la société / entité qui a effectué la réservation ou responsable du paiement de la réservation.*
26. *days_in_waiting_list*	*Discrète*	*Nombre de jours pendant lesquels la réservation était sur la liste d'attente avant d'être confirmée au client*
27. *customer_type*	*Catégorielle*	*Type de réservation, dans l'une des quatre catégories suivantes: Contrat - lorsque la réservation est associée à un lot ou à un autre type de contrat; Groupe - lorsque la réservation est associée à un groupe; Transitoire - lorsque la réservation ne fait pas partie d'un groupe ou d'un contrat et n'est pas associée à une autre réservation transitoire; Transient-party - lorsque la réservation est transitoire, mais est associée à au moins une autre réservation transitoire*
28. *adr*	*Continue*	*Tarif journalier moyen tel que défini en divisant la somme de toutes les transactions d'hébergement par le nombre total de nuitées*
29. *required_car_parking_spaces*	*Discrète*	*Nombre de places de parking requises par le client*
30. *total_of_special_requests*	*Discrète*	*Nombre de demandes spéciales faites par le client (par exemple lit double ou étage élevé)*
31. *reservation_status*	*Catégorielle*	*Dernier statut de la réservation, dans l'une des trois catégories suivantes: Annulée : la réservation a été annulée par le client; Check-Out : le client s'est enregistré mais est déjà parti; No-Show : le client ne s'est pas enregistré et a informé l'hôtel de la raison pour laquelle*
32. *reservation_status_date*	*Date*	*Date à laquelle le dernier statut a été défini. Cette variable peut être utilisée en conjonction avec le ReservationStatus pour comprendre quand la réservation a été annulée ou quand le client a-t-il quitté l'hôtel.*

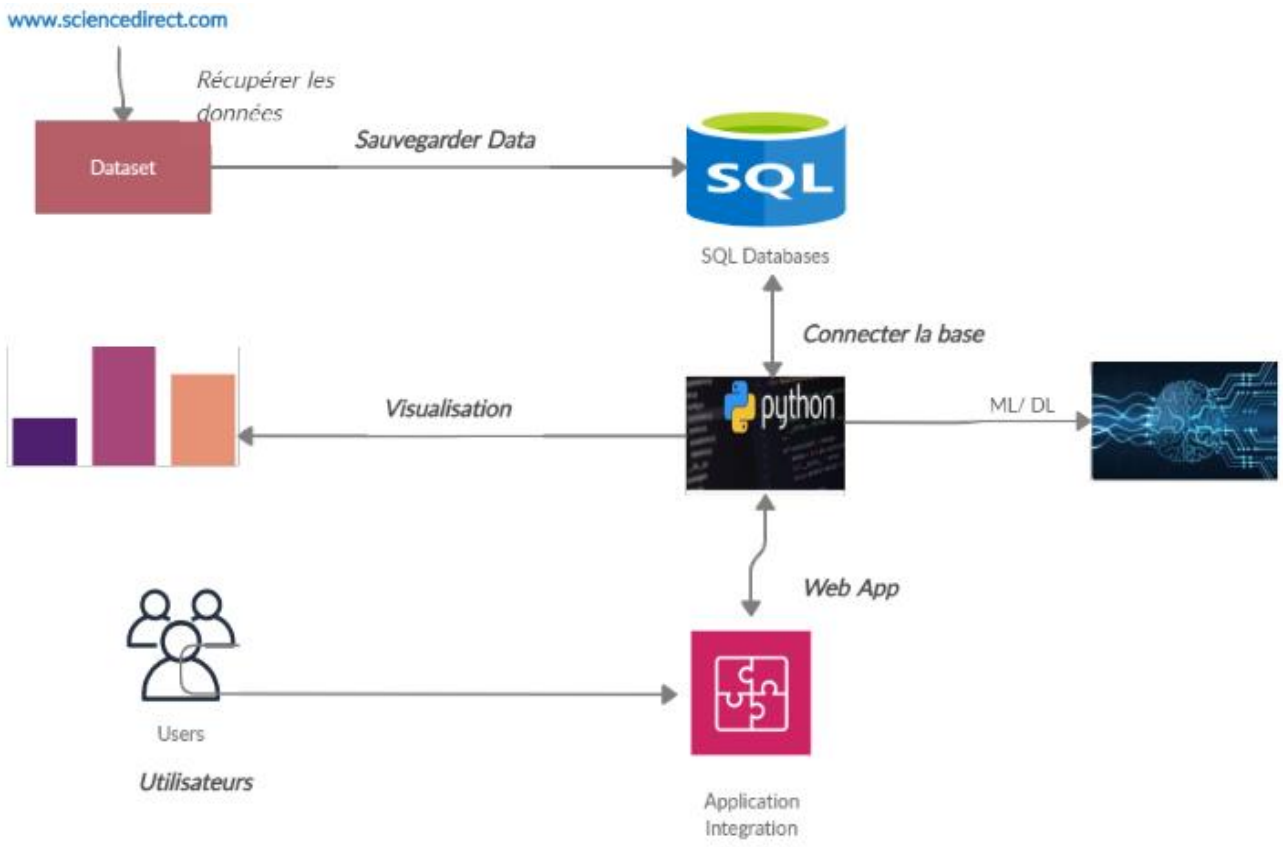
b) L'exploration de données

Ci-dessus la liste non exhaustive des outils qui nous ont permis à la réalisation du projet.

MySQL workbench : Est un outil graphique intégré, dédié aux développeurs et aux administrateurs. Il permet de concevoir et de modéliser une base de données.

WebAPP & Heroku : Heroku est une plateforme en tant que service (PaaS) permettant de déployer des applications sur le Cloud très facilement

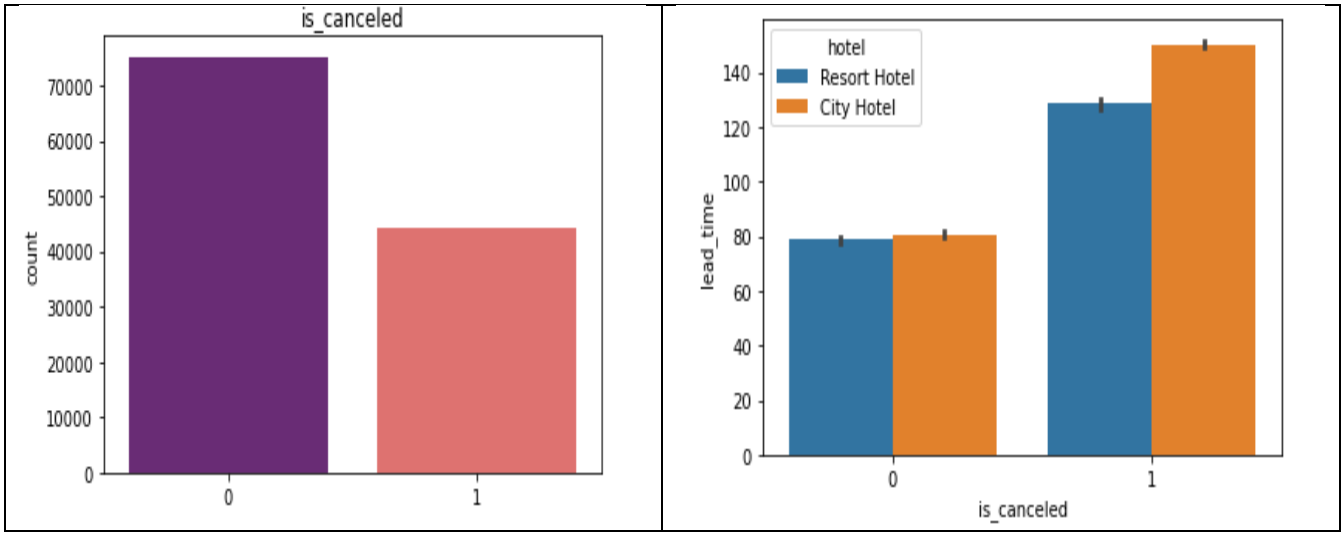
Pyton & ML & DL : Dans ce projet nous utilisons le langage de programmation Python.



Infrastructure du système

- **Description de notre variable d'intérêt**

Dans ce projet on s'intéresse en particulier à la variable « is_canceled », on construira notre modèle pour la prédire. Mais avant nous allons faire une analyse uni-variée pour cette variable.



D'après le diagramme à gauche on voit qu'il y a plus de réservations qui ont abouties que celles annulées. En la croisant avec la variable « lead_time » autrement ' Nombre de jours écoulés entre la date d'entrée de la réservation dans le PMS et la date d'arrivée' on remarque que plus le nombre de jours entre la date de réservation et le jour d'arrivée augmente, plus il y a le risque que la réservation soit annulée.

- **Statistiques élémentaires :**

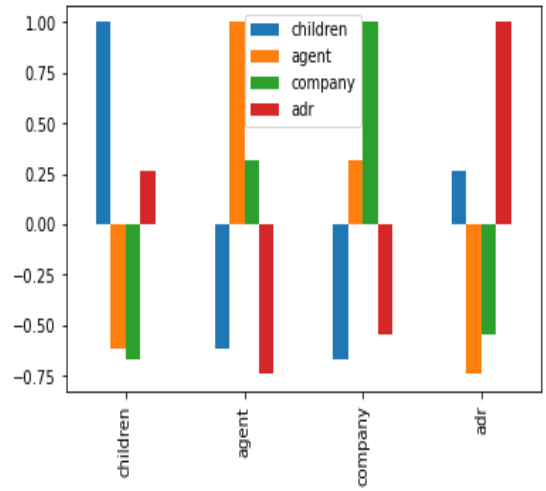
Dans la section ci-dessous, on effectue quelques statistiques élémentaires telles que la moyenne, les min, max et l'écart-type.

	count	mean	std	min	25%	50%	75%	max
id	20.0	10.500000	5.916080	1.00	5.75	10.500	15.25	20.0
children	119386.0	0.103890	0.398561	0.00	0.00	0.000	0.00	10.0
agent	103050.0	86.693382	110.774548	1.00	9.00	14.000	229.00	535.0
company	6797.0	189.266735	131.655015	6.00	62.00	179.000	270.00	543.0
adr	119390.0	101.831122	50.535790	-6.38	69.29	94.575	126.00	5400.0

Le nombre d'enfants par réservation peut aller de 0 à 10 enfants, mais analysant de plus prêt on voit que la plupart des clients n'incluent pas des enfants pendant la réservation.

- **Corrélation entre les variables quantitatives**

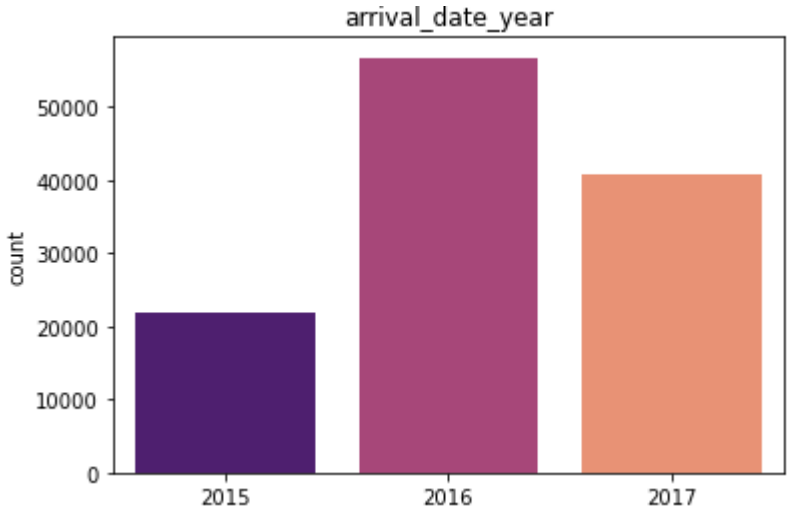
Ci-dessous on s'intéresse à savoir s'il y a un lien entre quelques variables.



	children	agent	company	adr
children	1.000000	0.041066	0.030931	0.324854
agent	0.041066	1.000000	0.350746	-0.024695
company	0.030931	0.350746	1.000000	0.086376
adr	0.324854	-0.024695	0.086376	1.000000

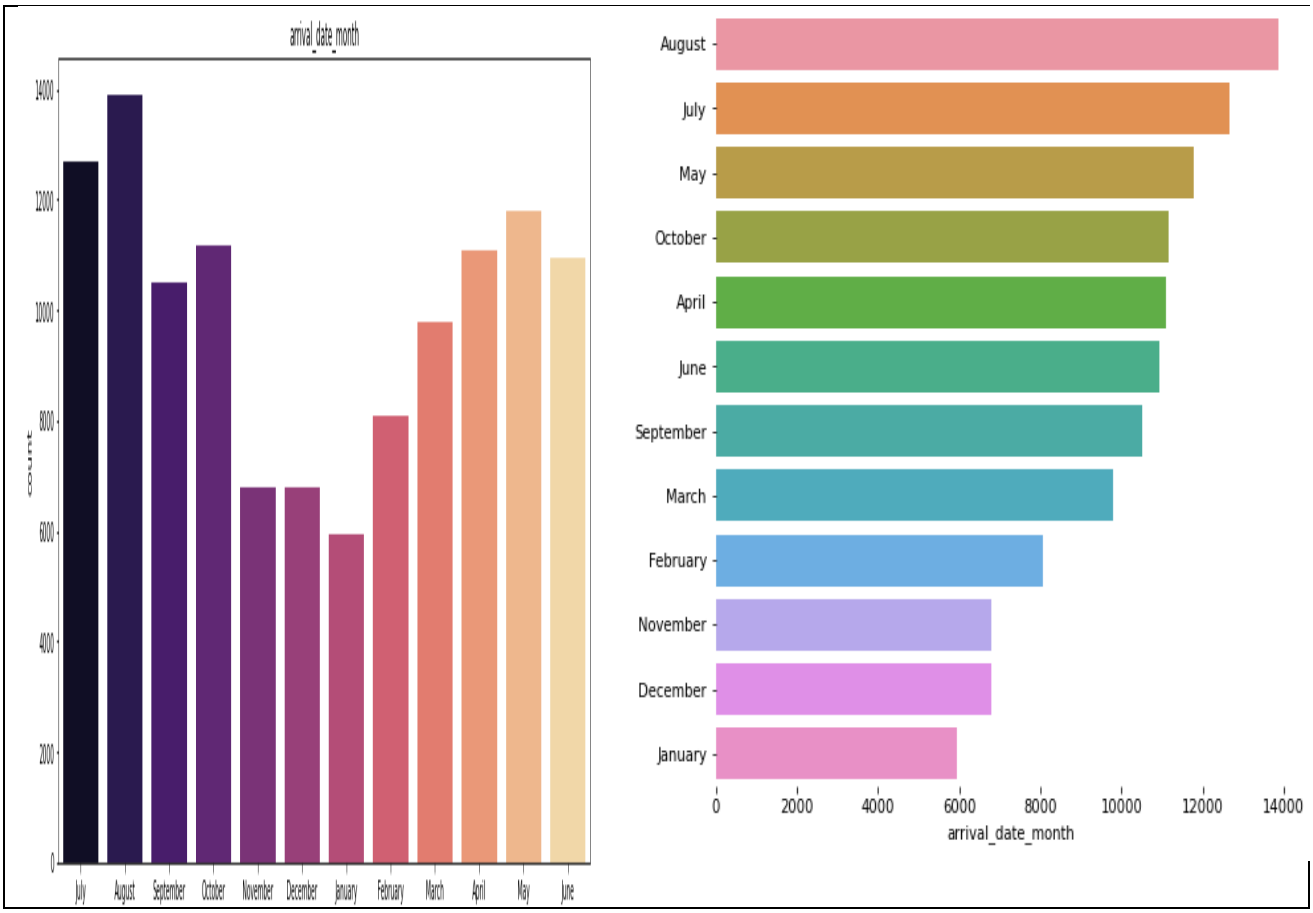
On a un coefficient de corrélation entre les deux variables « **adr** » et « **children** » vaut 0,32, ce qui signifie que ces variables sont corrélées, autrement dit plus le nombre d'enfants augmente plus le tarif journalier augmente aussi.

Maintenant nous voulons savoir davantage sur la réparation des réservations au cours de trois années d'études : de 2015 à 2017



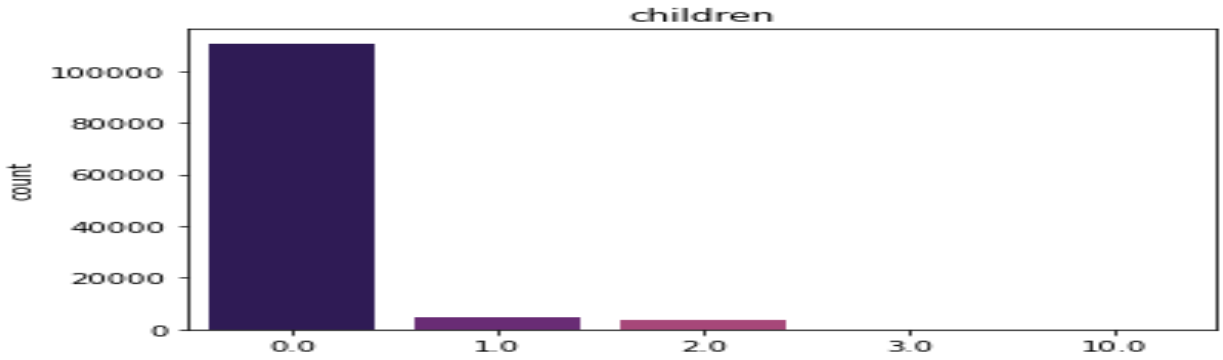
On constate que la plupart des réservations était en 2016.

On veut aussi voir de plus près la répartition des réservations pendant les trois d'année selon les mois. Pour cela on regarde la date d'arrivée.



On remarque que le mois ou il y a plus de réservations est le mois d'aout **(14000 réservations)** suivi de juillet (12000) en suite le mai. Ceci peut s'expliquer évidemment par le fait qu'il s'agit de la période d'été.

Il faut aussi noter que les mois ou il y a moins de réservations sont : Novembre, Décembre et janvier



Il est clair que la grande majorité de clients réservent les chambres d'hôtels sans enfants comme le montre le graphique ci-dessus.

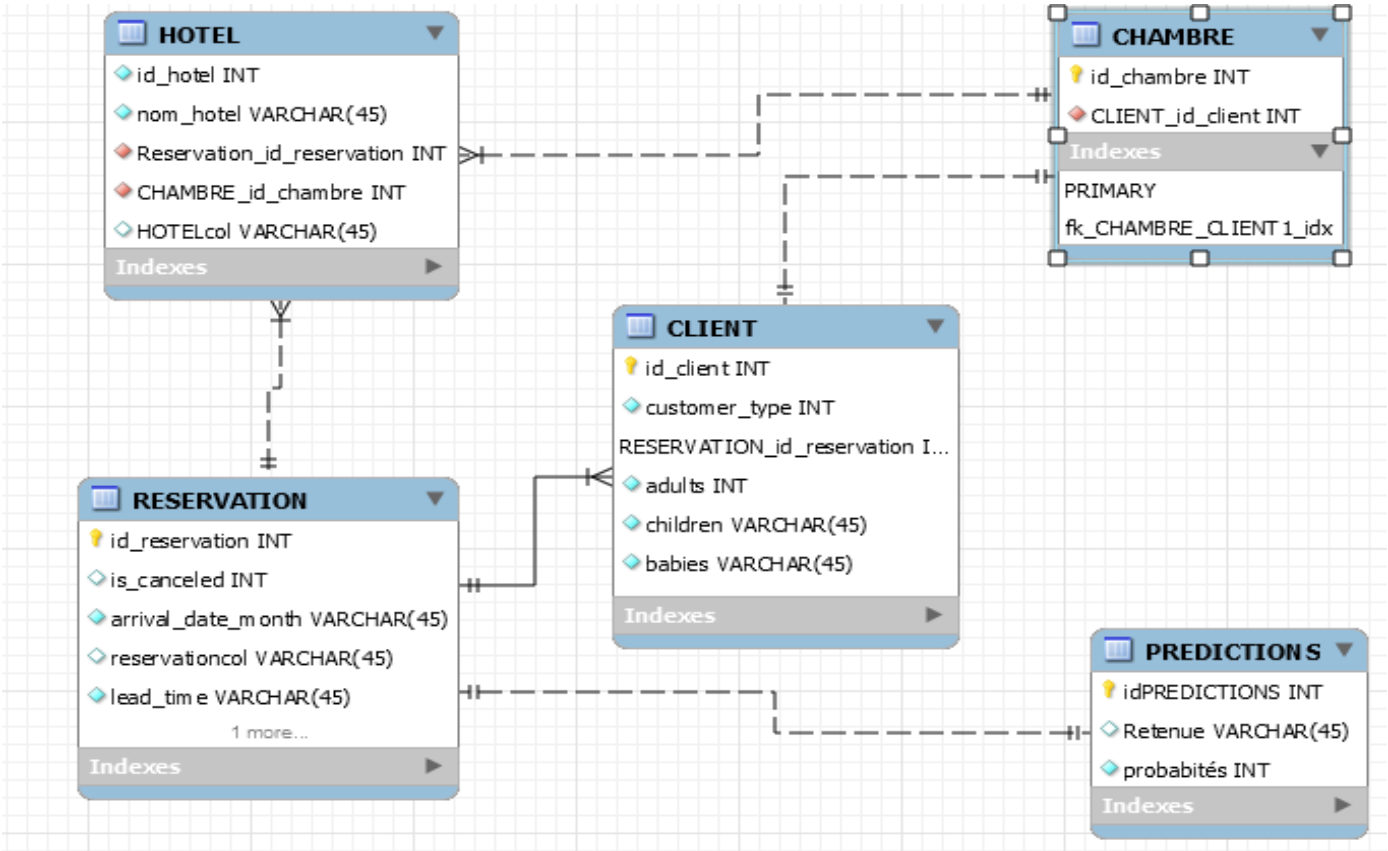
c) La base de données :

Cette partie est consacrée à la gestion de la base de données, pour cela nous procéderons en plusieurs étapes. D'abord créer un modèle de notre BDD sur mysql workbench, puis connecter mysql avec notebook python et suite remplir les tables à partir de notre jeu de données, à l'aide des requêtes Python.

- **Conception du modèle :**

Pour exploiter et analyser nos données nous allons récupérer des données analytiques et les préparer. Pour cela on commence par modéliser la base de données relationnelle sur MySQL workbbench.

Nous avons ajouté deux tables UTILISATEURS et PREDICTIONS, la première va contenir des données saisies par les utilisateurs et la seconde les sorties du modèle.



- **Ci-dessous le code pour créer des tables**

```
-- Schema mydb
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;
USE `mydb` ;

-- -----------------------------------------------------
-- Table `mydb`.`chambre`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`chambre` (
  `id_chambre` INT NOT NULL,
  PRIMARY KEY (`id_chambre`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;


-- -----------------------------------------------------
-- Table `mydb`.`client`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`client` (
  `id_client` INT NOT NULL,
  `customer_type` INT NOT NULL,
  PRIMARY KEY (`id_client`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;
```

```
-- Table `mydb`.`reservation`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`reservation` (
  `id_reservation` INT NOT NULL,
  `is_canceled` INT NULL DEFAULT NULL,
  `arrival_date_month` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id_reservation`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;


-- -----------------------------------------------------
-- Table `mydb`.`hotel`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`hotel` (
  `id_hotel` INT NOT NULL,
  `nom_hotel` VARCHAR(45) NOT NULL,
  `Client_id_client` INT NOT NULL,
  `Reservation_id_reservation` INT NOT NULL,
  INDEX `fk_Hotel_Client_idx` (`Client_id_client` ASC) VISIBLE,
  INDEX `fk_Hotel_Reservation1_idx` (`Reservation_id_reservation` ASC) VISIBLE,
  CONSTRAINT `fk_Hotel_Client`
    FOREIGN KEY (`Client_id_client`)
    REFERENCES `mydb`.`client` (`id_client`),
  CONSTRAINT `fk_Hotel_Reservation1`
    FOREIGN KEY (`Reservation_id_reservation`)
    REFERENCES `mydb`.`reservation` (`id_reservation`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;
```

```
-- Table `mydb`.`PREDICTIONS`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`PREDICTIONS` (
  `idPREDICTIONS` INT NOT NULL AUTO_INCREMENT,
  `Retenue` VARCHAR(45) NULL,
  `probabités` INT NOT NULL,
  PRIMARY KEY (`idPREDICTIONS`))
ENGINE = InnoDB;


SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

- **Connexion entre MySQL Workbench et notebook Python :**

```
import pymysql
host = 'localhost'
password = 'boulo1986'
database = 'bdd_titre_pro'

conn = pymysql.connect(
    host=host,
    port=int(3306),
    user="root",
    passwd=password,
    db=database,
    charset='utf8mb4')
df = pd.read_csv("hotel_bookings.csv", sep=";")
```

- **Insérer des données à partir de notebook**

Le code ci-dessous nous permet d'insérer les tables crées au paravent à partir de notre dataset, cette base est alimenté par des données des utilisateurs. Des temps en temps on fait des mises à jour.

```
for row in df.iterrows():
    print("INSERT INTO mydb(id, followers, authors_gender)
          VALUES ({iden},{followers},{authors_gender})".format(iden=row[1]['id'],
          followers=row[1]['followers'],
          is_canceled=row[1]['is_canceled']))

#Update an Attribute Value in the Table
arrival_date_month = "UPDATE  df SET arrival_date_month= 'July'  WHERE is_canceled = '1' ;"
cursor.execute(arrival_date_month)
```

3. Création d'un modèle & Intelligence artificielle

a) Les outils :

Dans cette partie nous souhaitons construire un modèle d'intelligence artificielle pour répondre à notre problématique : prédire si une réservation sera maintenue ou annulée. Pour cela nous allons utiliser plusieurs algorithmes tél que « Random Forest », « Régression logistique » et « librairie Keras ».

Puis nous allons choisir le meilleur algorithme qui sera déployé en ligne. Nous avons choisi ces algos car nous disposons d'une variable binaire.

b) La préparation :

Pour construire notre modèle nous allons en premier temps partitionner notre dataset en trois parties reparties comme suit:

- 70% du data set pour entrainer le modèle (data training)
- 20% pour valider le modèle
- Et 10% pour tester le modèle (à faire une seule fois pour éviter le sur-apprentissage)

```
X = data_table.drop(['is_canceled'], axis=1)
y = data_table["is_canceled"]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=0)
```

c) L'état de l'art

Dans cette sections nous avons choisi les trois algorithmes d'apprentissage automatique : Régression logistique, Random Forest et Deep learning.

- **Random Forest :**

Cet algorithme appartient à la famille des agrégations de modèles, c'est en fait un cas particulier de bagging (bootstrap aggregating) appliqué aux arbres de décision de type CART. La base du calcul repose sur l'apprentissage par arbre de décision. Ci-dessous le code qui nous permet d'entrainer et d'évaluer le modèle.

```
# Random Forest Classifier
forest = RandomForestClassifier(n_estimators=100, random_state=0)
forest.fit(X_train, y_train)

print("Random forest model accuracy :")
print("Accuracy on training set : {:.3f}".format(forest.score(X_train, y_train)))
print("Accuracy on test set : {:.3f}".format(forest.score(X_test, y_test)))
# cross validation, optimisation de paramtres et grid search
```

```
Random forest model accuracy :
Accuracy on training set : 0.991
Accuracy on test set : 0.862
```

On constate qu'il y a un problème de sur-apprentissage, l'Accuracy sur les données d'entrainement est très élevé. Pour pallier à cette problématique nous intervenons aux niveaux de paramètres tels que grid search et cross validation.

```
rfc1=RandomForestClassifier(random_state=42, max_features='auto', n_estimators= 200, max_depth=8, criterion='gini')
rfc1.fit(X_train, y_train)
pred=rfc1.predict(X_test)
print("Accuracy for Random Forest on CV data: ",accuracy_score(y_test,pred))
```

```
print("Random forest model accuracy : ")
print("Accuracy on training set : {:.3f}".format(forest.score(X_train, y_train)))
print("Accuracy on test set : {:.3f}".format(forest.score(X_test, y_test)))
# cross validation, optimisation de paramtres et grid search
```

```
Random forest model accuracy :
Accuracy on training set : 0.631
Accuracy on test set : 0.626
```

On obtient des résultats plus raisonnables.

Dans la suite nous utilisons la librairie Keras qui va permettre d'interagir avec les algorithmes de réseaux de neurones profonds et d'apprentissage automatique, notamment Tensorflow

- **Création du Modèle Keras :**

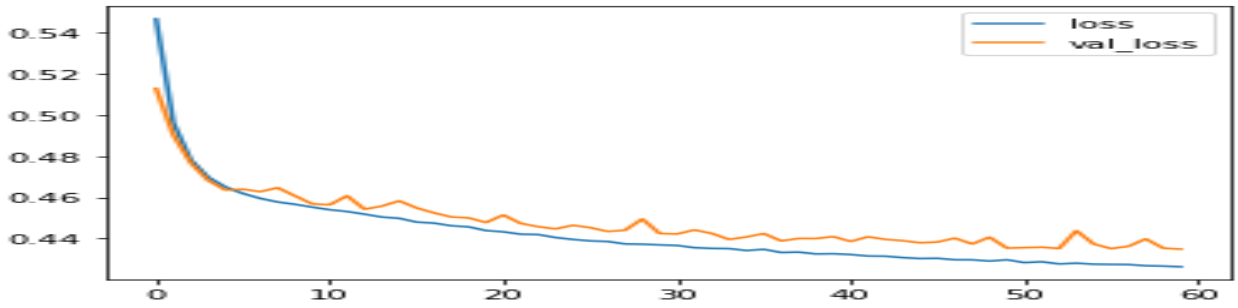
Avant de créer le modèle il faut partitionner le dataset, en plus il est vivement conseillé standariser (normaliser) les données.

Choisir trop d'Epochs et Overfitting :
Pour les contraintes de timings nous choisissons seulement 60 époques

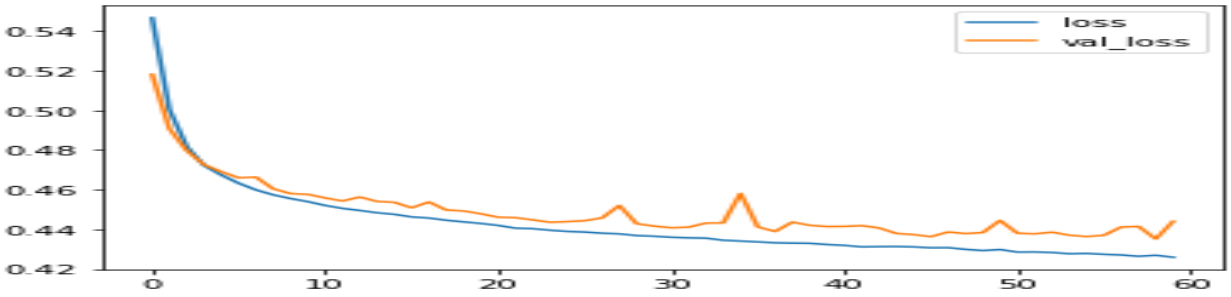
```
model = Sequential()
model.add(Dense(units=30,activation='relu'))
model.add(Dense(units=15,activation='relu'))
model.add(Dense(units=1,activation='sigmoid'))
# For a binary classification problem
model.compile(loss='binary_crossentropy', optimizer='adam')
model.fit(x=X_train,
          y=y_train,
          epochs=60,
          validation_data=(X_test, y_test), verbose=1
          )
```

On constate que le modèle s'arrête de s'améliorer en moment donné, donc nous allons utiliser quelques mesures pour améliorer notre modèle.

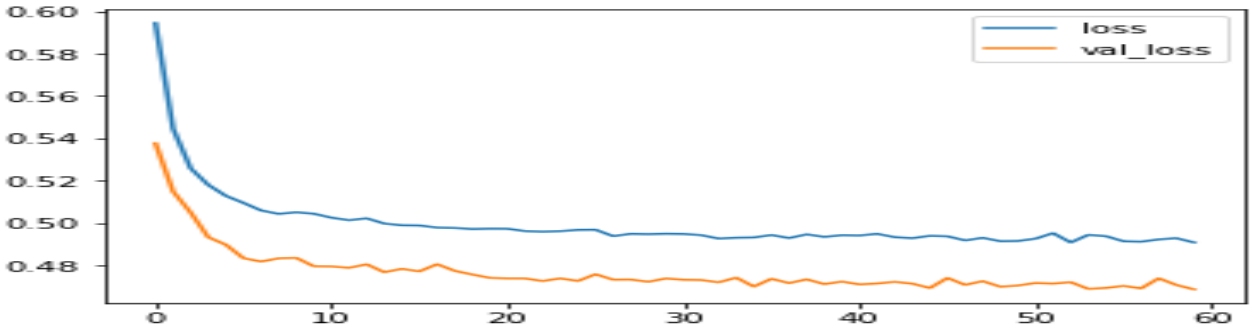
model.history.history



Early Stop



Ajout de couches DropOut



Évaluation du modèle : Ci-dessous nous avons quelques paramètres pour évaluer le modèle tels que : La précision, le rappel et f1-score

	precision	recall	f1-score	support
0	0.77	0.91	0.84	24679
1	0.79	0.55	0.65	14720
accuracy			0.78	39399
macro avg	0.78	0.73	0.74	39399
weighted avg	0.78	0.78	0.76	39399

Comparaison des modèles : Sur le tableau ci-dessous nous avons les accuracys de nos trois algorithmes.

	Train	Validation
Régression logistique	0.737	0.734
Random Forest	0.631	0.626
Keras	0.773	0.772

d) Le choix du modèle :

Après avoir comparé les trois algorithmes, nous avons choisi celui obtenu à l'aide de Keras, il sera intégré dans l'application WEB puis déployer en ligne sur cloud Heroku.

4. L'application web

Dans cette partie nous allons construire une application web. Via cet outil les utilisateurs peuvent se connecter pour réserver les chambres d'un côté, de l'autre côté le gérant du site peut désormais suivre les réservations à temps réel, et surtout il va recevoir les notifications lui indiquant quelle est la probabilité qu'une réservation sera maintenue ou annulée.

a) Les tests unitaires : pytest

Pour vérifier le bon fonctionnement du code, nous appliquons des tests unitaires, pour cela on utilise le package pytest.

Pytest est un Framework permettant de faire des tests et de vérifier si les différentes conditions sont justes ou fausses. Il permet de tester les éléments un à un mais on peut aussi lui demander de faire une série de tests

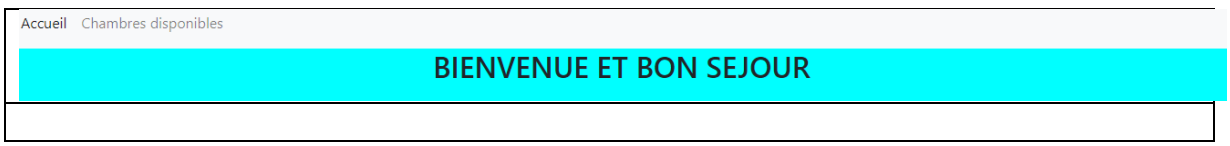
b) Le monitoring

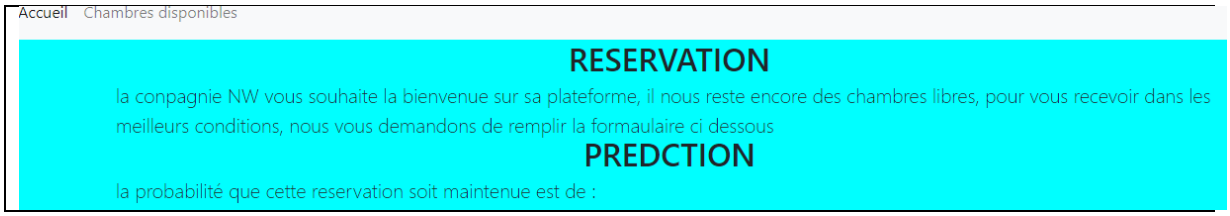
Le monitoring va nous permettre de détecter et corriger les éventuels dysfonctionnent de notre application.

c) La maquette de l'APPLI :

Pour rendre notre application présentable, nous la mettons dans une maquette, dans notre cas on utilise FIGMA.

Capture d'écran de la page d'accueil du site :



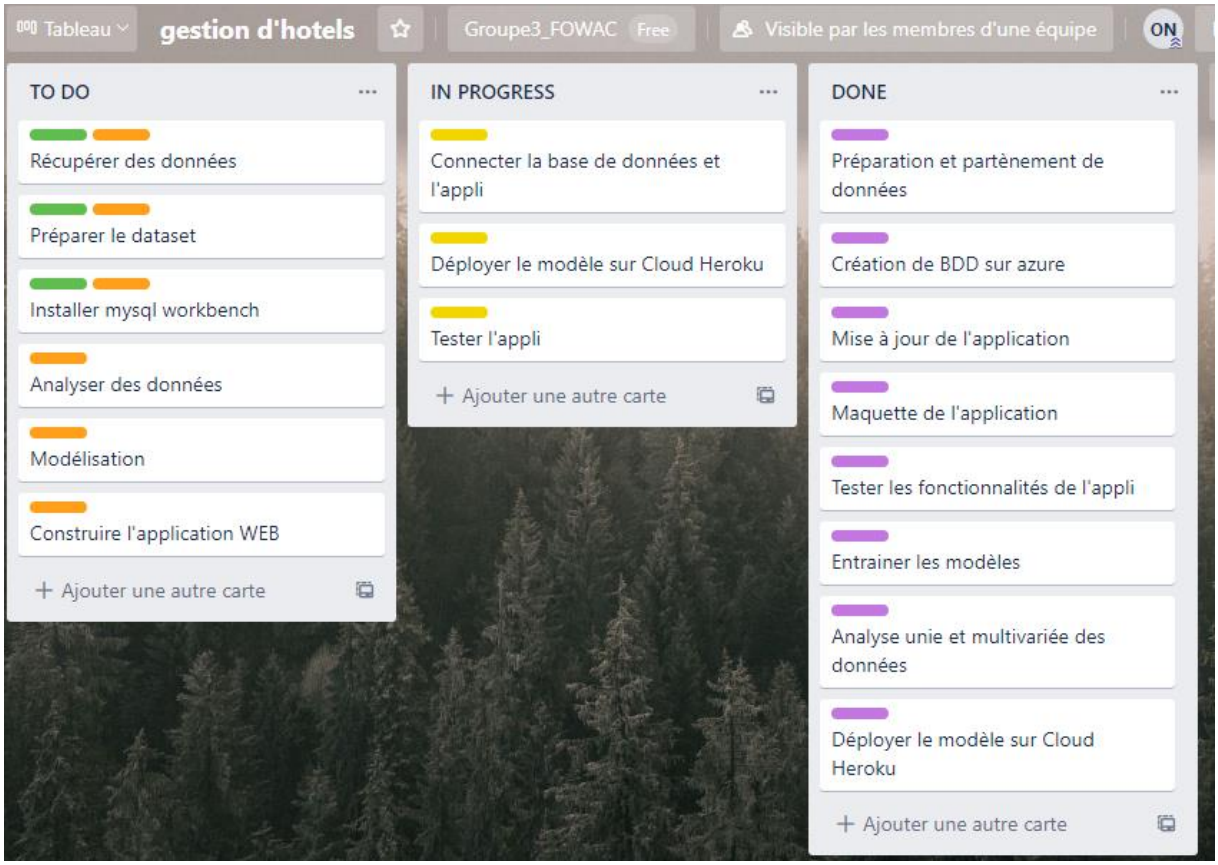


5. Gestion du projet par méthode SCRUM

Pour suivre l'avancement de notre projet on s'appuie sur la méthode SCRUM. En effet cette méthode va nous aider à rester concentrer sur nos objectifs et gagner du temps, en plus être plus productif. Nous construisons une équipe SCRUM qui sera composé comme suit :

- Un Scrum Master
- un Product Owner
- une équipe de développement

Nous utilisons le trello pour suivre l'évolution de notre projet



6. Conclusion

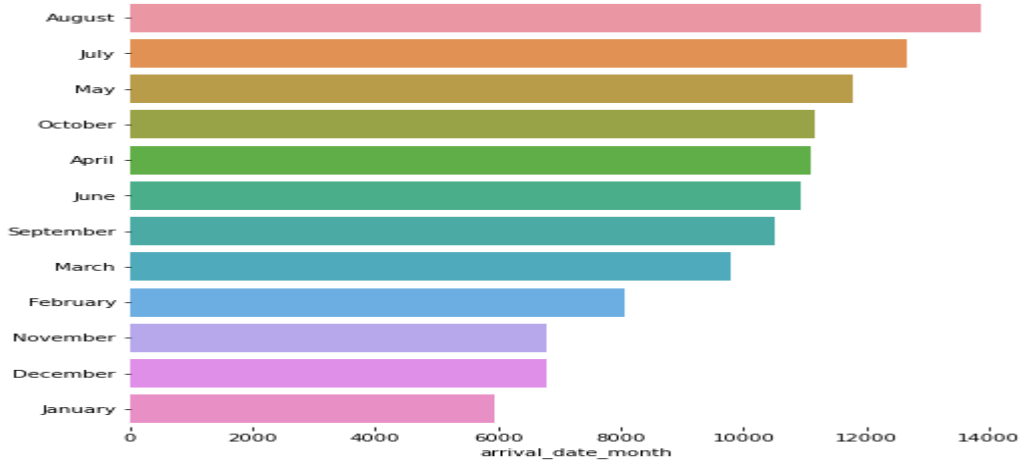
A travers ce projet j'ai pu appliquer les connaissances acquises durant la période de formation. D'abord la récupération et nettoyage de notre dataset, puis concevoir une base de données, en plus relier cette base avec mon notebook et en fin construction d'une application web.

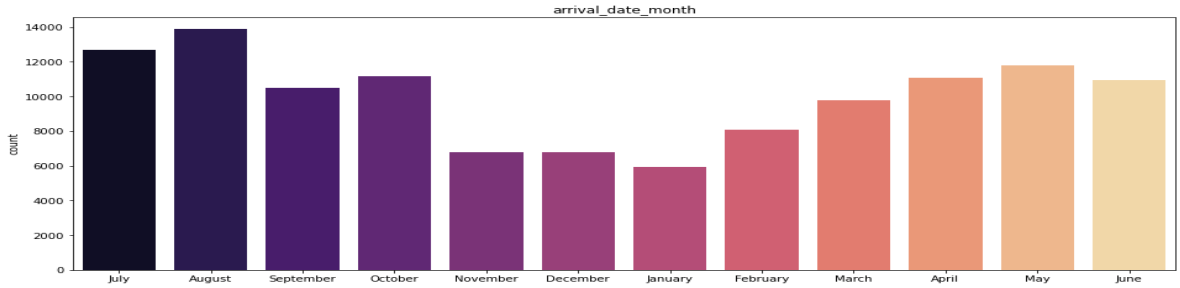
J'ai puis constaté qu'il est tout à fait possible de prédire l'état d'une réservation en se basant sur certains caractéristiques tels que : « le nombre de jour entre la date réservation et la date d'arrivée » et « Le tarif journalier moyen ». D'autres paramètres aussi sont à prendre en compte : « le mois » et « le nombre d'adultes ».

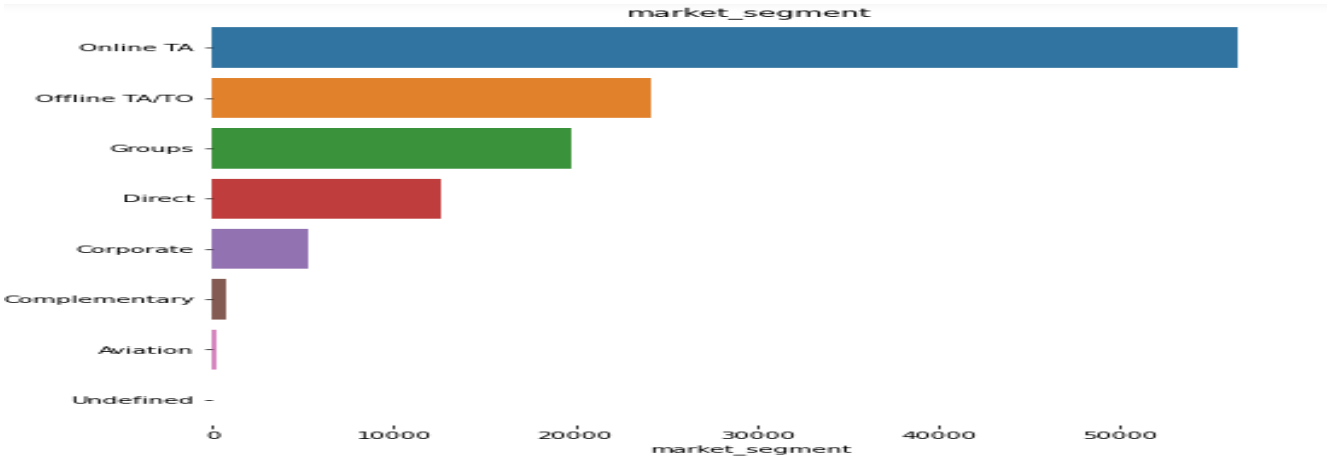
Ce projet que j'ai effectué m'a permis de chercher puis mettre en place de plusieurs technologies pour atteindre un objectif. Ceci m'a permis d'apprendre et consolider un certain savoir-faire en programmation et en gestion de projet. Ça a fut aussi une opportunité pour moi d'élargir mon champ de compétences et d'en découvrir d'autres.

Annexe :

- Graphiques :







- Code :
- Le notebook
- Appli

- Tables

	Numeric_var
0	id
1	is_canceled
2	lead_time
3	arrival_date_year
4	arrival_date_week_number
5	arrival_date_day_of_month
6	stays_in_weekend_nights
7	stays_in_week_nights
8	adults
9	children
10	babies
11	is_repeated_guest
12	previous_cancellations
13	previous_bookings_not_canceled
14	booking_changes
15	agent
16	company
17	days_in_waiting_list
18	adr
19	required_car_parking_spaces

	Categ_vars
0	hotel
1	arrival_date_month
2	meal
3	country
4	market_segment
5	distribution_channel
6	reserved_room_type
7	assigned_room_type
8	deposit_type
9	customer_type
10	reservation_status

```
[[22516  2163]
 [ 6683  8037]]
```

Matrice de confusion