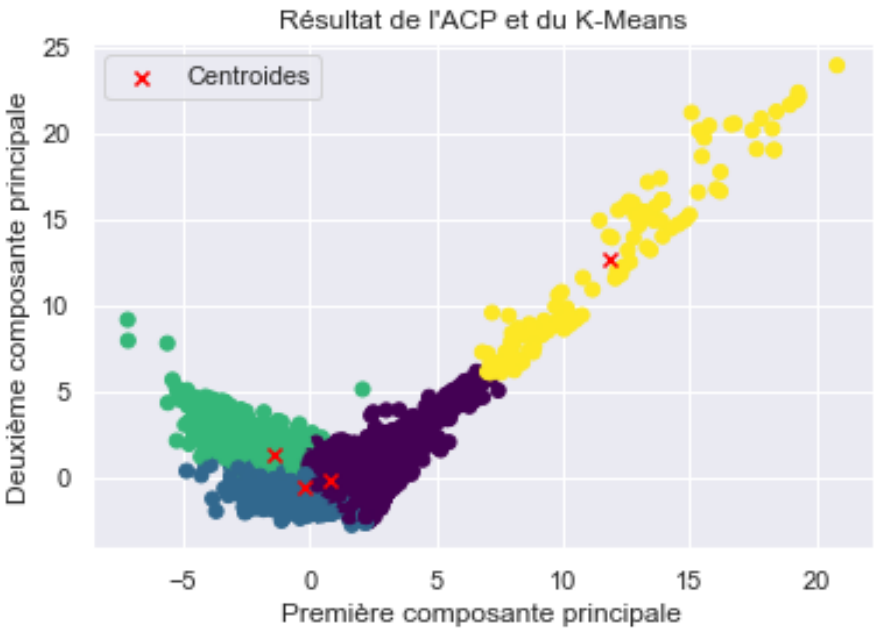
POLYTECH LILLE

Université de Lille

Projet Machine Learning



Année universitaire 2022-2023

Loucas Cubeddu & Emilian Loric
IS4

Partie I:

I.1 - Description brève

Dans le jeu de données Hotel Reservation.csv, nous avons 36274 observations de réservations à disposition, ainsi que 18 caractéristiques pour chacune d'elles.
Il se trouve qu'il n'y a pas de données manquantes pour aucune des variables.

Pour chaque réservation, nous avons les informations suivantes :

-**Booking_ID** : identifiant unique
-**no_of_adults** : nombre d'adultes
-**no_of_children** : nombre d'enfants
-**no_of_weekend_nights** : nombre de nuits réservées le week-end
-**no_of_week_nights** : nombre de nuits réservées la semaine
-**type_of_meal_plan** : type de menu réservé
-**required_car_parking_space** : réservation d'une place de parking (Non - 0, Yes = 1)
-**room_type_reserved** : type de chambre réservé
-**lead_time** : nombre de jour entre la réservation et la date d'arrivée
-**arrival_year** : année d'arrivée
-**arrival_month** : mois d'arrivée
-**arrival_date** : jour d'arrivée dans le mois (1-31)
-**market_segment_type** : type de client (particulier, entreprise etc)
-**repeated_guest** : 1 - client régulier , 0 sinon
-**no_of_previous_cancellations** : nombre de fois que le client a annulé ses réservations précédentes
-**no_of_previous_bookings_not_canceled** : nombre de fois que le client n'a pas annulé ses réservations précédentes
-**avg_price_per_room** : prix moyen de la réservation par jour
-**no_of_special_requests** : nombre de demandes spécifiques du client
-**booking_status** : statut de la réservation (annulé ou non annulé)

I.2 - Nettoyage de la base

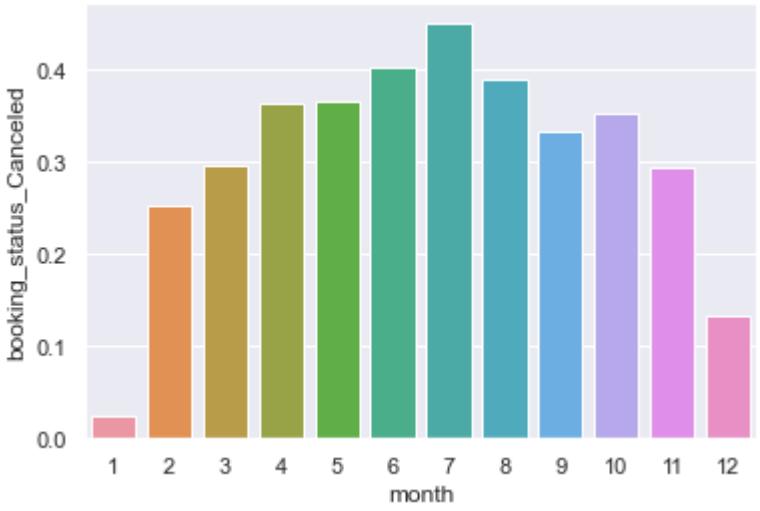
a) La variable Booking_ID

Cette variable n'apporte aucune information, c'est un simple identifiant de chaque réservation issue de la base. Nous ne gardons pas cette colonne.

b) Etude de l'influence de la date sur l'annulation des réservations

Tout d'abord, le jour dans le mois ne nous aidera pas dans la prédiction. On supprime donc cette variable.
Les dates de notre jeu de données allant de janvier 2017 à janvier 2019, l'année n'apporte pas une information pertinente. Ce qui nous intéresse est de savoir si selon la période de l'année, il y a plus ou moins d'annulations.

Le graphique suivant représente la proportion de réservation annulées selon le mois de l'année :



On observe que la proportion de réservations annulées est plus importante lors de la saison estivale, atteignant environ 47% d'annulation sur le mois de juillet.

Ainsi, nous gardons la variable du mois d'arrivée.

c) Types des variables

Nous avons redéfini les types de chaque variable.

Les variables **numériques continues** sont : 'lead_time', 'avg_price_per_room'

Les variables **numériques discètes** sont : 'no_of_adults', 'no_of_children', 'no_of_weekend_nights', 'no_of_week_nights', 'arrival_year', 'arrival_month', 'arrival_date', 'no_of_previous_cancellations', 'no_of_previous_bookings_not_canceled', 'no_of_special_requests'

Les variables **catégorielles** sont : 'type_of_meal_plan', 'room_type_reserved', 'market_segment_type', 'repeated_guest', 'required_car_parking_space'

d) Analyse de la distribution des variables numériques discrètes, continues et des variables catégorielles

Nous avons représenté de nombreux graphiques afin de mieux comprendre les spécificités de chaque variable du jeu de données. Nous n'avons pas estimé nécessaire de supprimer davantage de variables.

e) Informations sur la nouvelle base de données

Nous avons maintenant supprimer les variables que nous ne souhaitons pas garder dans notre nouvelle base.

Nous avons désormais 16 variables au lieu de 19.

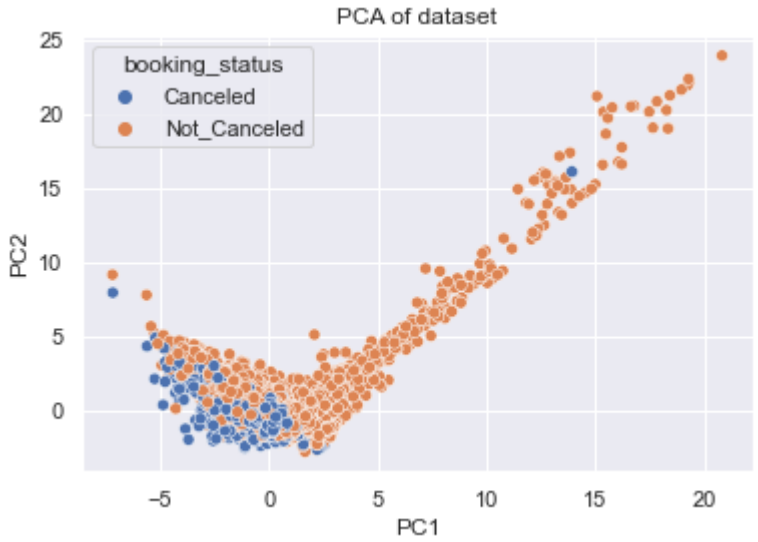
Voici un récapitulatif des variables de notre nouvelle base:

-**no_of_adults** : nombre d'adultes
-**no_of_children** : nombre d'enfants
-**no_of_weekend_nights** : nombre de nuits réservées le week-end
-**no_of_week_nights** : nombre de nuits réservées la semaine
-**type_of_meal_plan** : type de menu réservé
-**required_car_parking_space** : réservation d'une place de parking (Non - 0, Yes = 1)
-**room_type_reserved** : type de chambre réservé
-**lead_time** : nombre de jour entre la réservation et la date d'arrivée
-**month** : mois d'arrivée
-**market_segment_type** : type de client (particulier, entreprise etc)
-**repeated_guest** : 1 - client régulier , 0 sinon
-**no_of_previous_cancellations** : nombre de fois que le client a annulé ses réservations précédentes
-**no_of_previous_bookings_not_canceled** : nombre de fois que le client n'a pas annulé ses réservations précédentes
-**avg_price_per_room** : prix moyen de la réservation par jour
-**no_of_special_requests** : nombre de demandes spécifiques du client
-**booking_status** : statut de la réservation (annulé ou non annulé)

I.3 - Définitions des profils clients

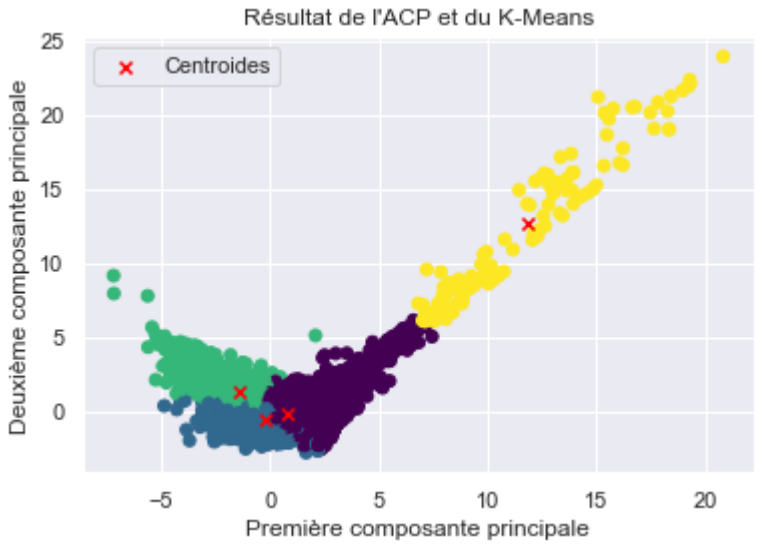
Afin d'établir des profils de clients, nous effectuons tout d'abord de l'analyse exploratoire avec une Analyse en Composante Principale dite ACP.
Nous avons choisis l'ACP car elle peut être réalisée à partir de données numériques continues et discrètes, contrairement à l'AFC et ACM. Or ce sont bien ces types de données qui sont majoritaires dans notre base. Cette représentation des données qu'est l'ACP permet de réduire la dimensionnalité des données tout en préservant la variance maximale.

Voici notre ACP en 2 dimensions :



Ensuite, nous avons décidé d'appliquer un algorithme de clustering sur un certain nombre de composantes principales (méthode du coude) issues de l'ACP. Nous avons choisis l'algorithme K-means puisque celui-ci permet de segmenter les clients en groupe homogènes sur la base des caractéristiques les plus importantes de nos données.

Voici le résultat de notre K-means pour 4 clusters :



Nous utilisons ces clusters pour définir des profils clients.

I.4 - Analyse des variables explicatives des profils

Pour déterminer les variables importantes pour chaque cluster, nous examinons les variables qui ont le plus grand écart entre chaque clusters. Pour ce faire, nous calculons la moyenne et l'écart-type de chaque variable pour chaque cluster, puis comparons ces valeurs pour trouver les variables les plus distinctes entre les clusters.

Ainsi, nous pouvons dresser un profil de clients pour chaque cluster.

Voici une analyse des profils clients pour chaque cluster:

Cluster 1 - Les clients occasionnels

-Séjours plutôt courts, principalement en semaine
-Peu de demandes spéciales
-Faible taux d'annulations précédentes
-Tarif moyen par chambre de 91,29 € (modéré par rapport aux autres clusters)
-Réservations planifiées avec un délai moyen

Profil: Ces clients pourraient être des voyageurs d'affaires ou des personnes en visite pour de courtes périodes, principalement en semaine. Ils ont généralement peu de demandes spéciales et un historique d'annulation faible.

Taux d'annulation : 23,11%
Nombre d'individus : 15 119

Cluster 2 - Les planificateurs

-Séjours plus longs avec un mélange de nuits en semaine et le week-end
-Faible taux d'annulations précédentes
-Tarif moyen par chambre de 95,64 € (légèrement supérieur au cluster 1)
-Réservations planifiées longtemps à l'avance

Profil: Ces clients pourraient être des familles ou des couples en vacances, qui planifient leurs séjours longtemps à l'avance et passent plus de temps à l'hôtel. Ils sont moins susceptibles d'annuler leurs réservations.

Taux d'annulation : 47,41%
Nombre d'individus : 13 962

Cluster 3 - Les clients exigeants

-Plus d'adultes et d'enfants par réservation (par rapport aux autres clusters)
-Plus de demandes spéciales (par rapport aux autres clusters)
-Très faible taux d'annulations précédentes
-Tarif moyen par chambre de 145,40 € (le plus élevé parmi les clusters)
-Réservations planifiées avec un délai moyen

Profil: Ces clients pourraient être des familles ou des groupes avec des attentes plus élevées en matière de service et de qualité. Ils sont prêts à payer un tarif plus élevé par chambre et ont tendance à formuler davantage de demandes spéciales.

Taux d'annulation : 24,94%
Nombre d'individus : 7081

Cluster 4 - Les clients fidèles mais indécis

-Peu d'enfants et moins d'adultes par réservation par rapport aux autres clusters
-Séjours principalement en semaine
-Taux d'annulations précédentes élevé
-Nombre élevé de réservations précédentes non annulées
-Tarif moyen par chambre de 57,66 € (le plus bas parmi les clusters)

-Réservations planifiées avec un court délai

Profil: Ces clients pourraient être des voyageurs d'affaires réguliers ou des clients fidèles à l'hôtel, mais qui ont tendance à annuler leurs réservations plus souvent. Ils réservent généralement à court terme et paient le tarif le plus bas par chambre.

Taux d'annulation : 4,42%
Nombre d'individus : 113

Ainsi parmi ces 4 clusters et profils de clients dressés, le cluster 2 qui représente à priori des familles et des groupes sont les plus à mêmes d'annuler leurs réservations puisque près d'une réservation sur 2 est annulée (47,41%). Les taux d'annulations des clusters 1 et 3 sont à peu près similaires. La taille du cluster 4 est très faible (113 réservations) mais on y observe particulièrement peu de d'annulations (4,42%).

Partie II:

II.1 - Modèle de prédiction d'annulation des réservations

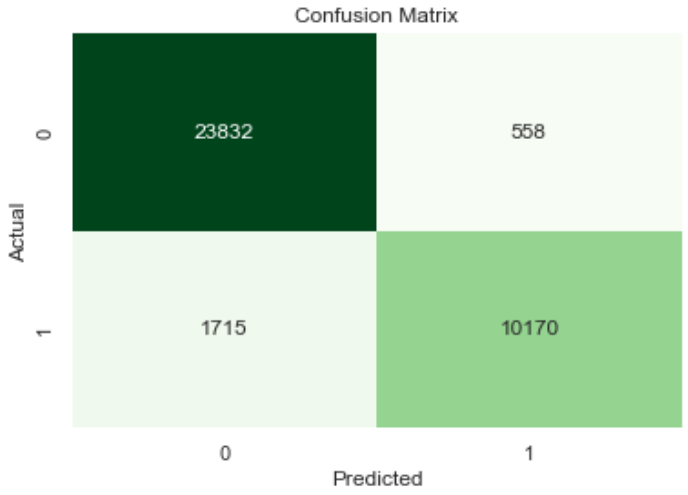
Nous avons créé un pipeline avec 2 préprocesseurs différents, un qui normalise les données, et un autre qui effectue une ACP normée.

Dans ce pipeline, nous entraînons les méthodes de régression logistique, de forêt aléatoire et de gradient boosting. Pour chacun d'entre eux, nous avons réalisé au sein du pipeline un GridSearchCV avec plusieurs hyper-paramètres pour chaque.

Le jeu étant déséquilibré comme nous l'avions dans la partie une, le critère sur lequel nous baserons la performance de notre modèle sera le F1-score. Par conséquent, nous précisions comme paramètre au GridSearch que c'est le F1-score que nous souhaitons maximiser et non l'accuracy défini par défaut.

Ainsi, nous avons obtenu un meilleur modèle qui est issu du Gradient Boosting avec les hyper-paramètres.

Voici la matrice de confusion de notre modèle sur ses données d'entrainement:



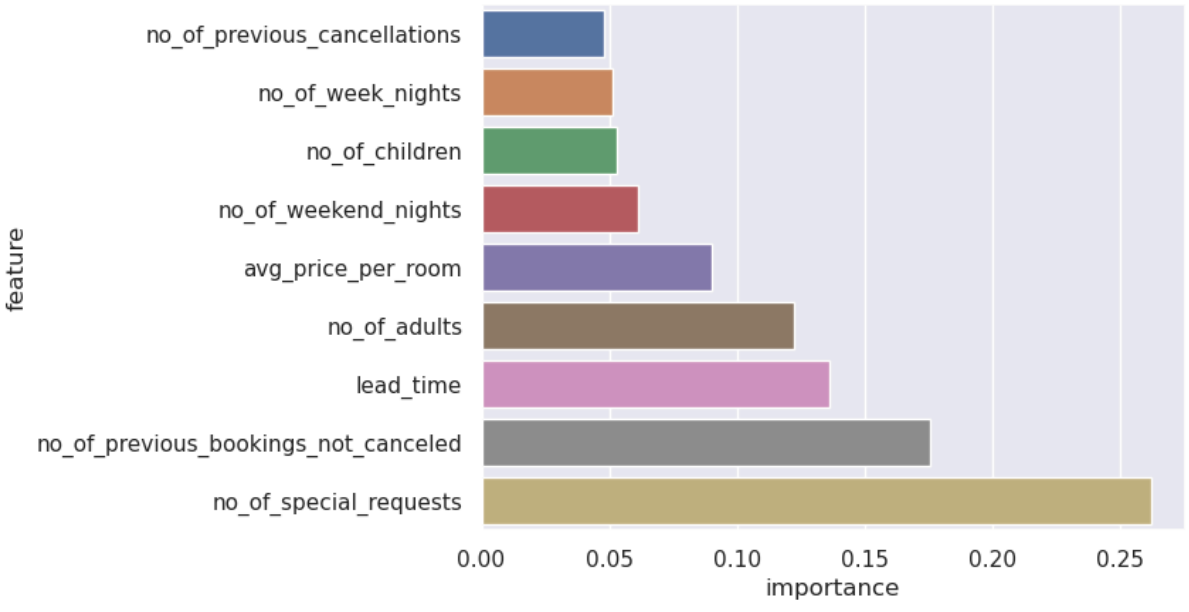
Voici les performances de ce modèle sur ses données d'entrainement:

Accuracy	F1-score	Recall	Précision
93,73%	89,94%	85,57%	94,79%

N'oublions pas que la vraie performance a été donnée par le f1 avec cross validation c'est à dire 76,7% de score f1.

II.2 - Analyse des variables explicatives de ce modèle

Voici le graphique qui représente l'importance de chaque variable de notre modèle :



Les 5 variables qui expliquent le plus le modèle sont le “no_of_special_requests”, “no_of_previous_bookings_not_canceled”, “lead_time”, “no_of_adults”, et “avg_price_per_room”.

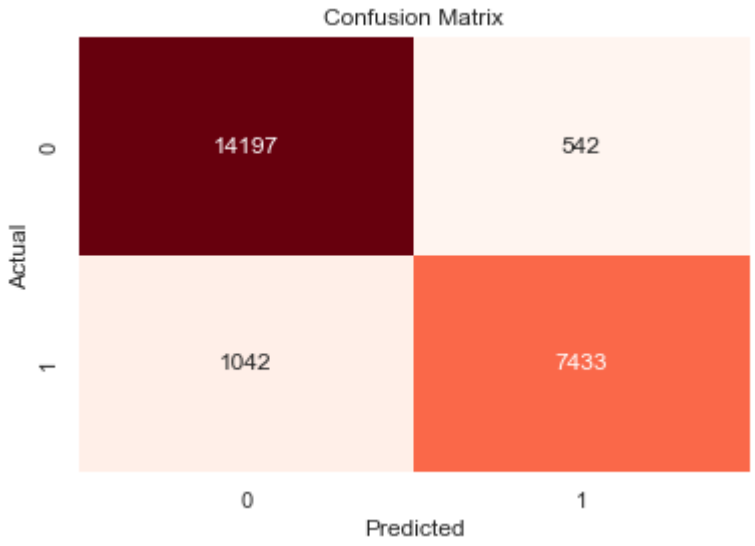
Ainsi, ce sont ces variables qui viennent principalement contribuer à ce qu’une réservation sera prédite comme future réservation annulée ou non.

II.3 - Modèle de prédiction d’annulation des réservation “online”

Nous avons repris le même pipeline que précédemment, nous avons simplement créé un nouveau jeu de données avec uniquement les réservations réalisées “online”.

Nous avons obtenu un meilleur modèle qui est issu du Gradient Boosting avec les hyper-paramètres.

Voici la matrice de confusion de notre modèle sur les données d'entraînement :



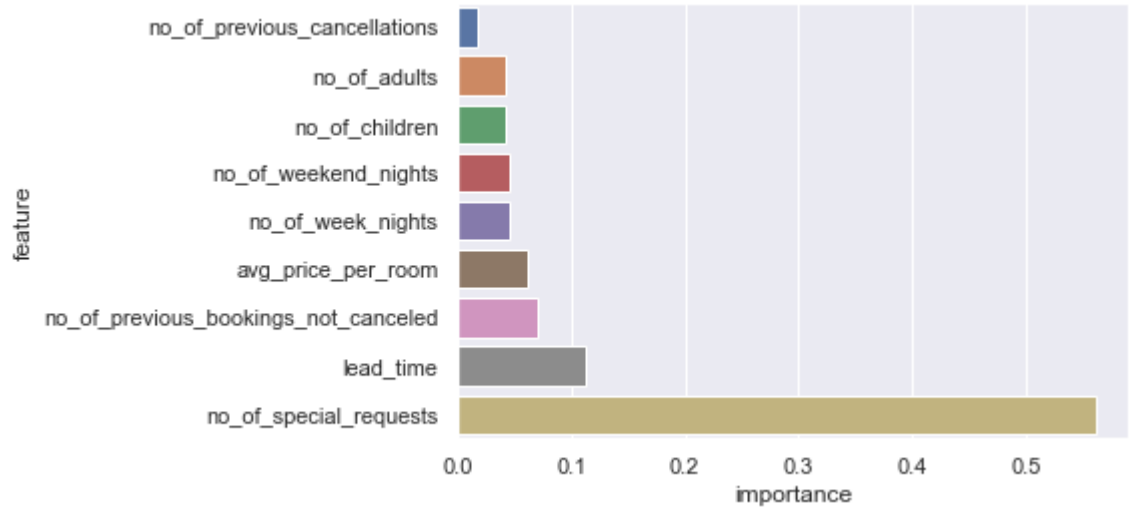
Voici les performances de ce modèle sur les données d'entraînement:

Accuracy	F1-score	Recall	Précision
93,17%	90,37%	87,70%	93,20%

Le critère recall est également un critère qu'il est intéressant de regarder en plus du F1-score. Il indique la proportion de vrais positifs. Dans notre cas, cela correspond à la proportion de réservation prédit comme annulée et qui le sont vraiment. Nous avons 87,70% au recall, ce qui nous paraît être une bonne performance. Mais attention ces mesures ont été faites sur les données d'entraînement. Une vraie mesure du modèle résulte ici de la cross validation où l'on trouve un score f1 de 77,2%.

II.4 - Analyse des variables du modèle d'annulation des réservation "online"

Voici le graphique qui représente l'importance de chaque variable de notre modèle de prédiction :



Dans ce modèle de prédiction des annulations des réservations faites "online", la variable "no_of_special_requests" est celle qui explique le mieux le modèle très loin devant les autres qui sont "lead_time", "no_of_previous_not_canceled" et "avg_price_per_room".

II.5 - Grid Search pour rechercher un meilleur modèle

Nous avons effectué un premier Grid Search avec Cross Validation sur un ensemble de méthodes (SVM, Régression Logistique, Random Forest, XGBoost ~ Gradient

Boosting régularisé), avec différents preprocessing (réduction de dimension par ACP, Normalisation standard).

Nous avons choisi comme métrique à maximiser le score f1 car nous souhaitons un compromis égal entre la précision et la sensibilité de notre modèle de classification (le score f1 étant la moyenne harmonique entre ces deux métriques).
On trouve que les méthodes de régression logistique et de SVM ne fournissent pas un résultat satisfaisant, on les abandonne donc pour la suite de cette recherche.

Cette analyse se trouve dans le notebook principal ML Projet

On trouve que le modèle le plus performant dans cette recherche est XGBoost avec 100 estimateurs, et une profondeur maximale de 10. On trouve après cross-validation un f1 moyen de 78,9%.
Les Random Forest donnent aussi des assez bons résultats avec un f1 max de 75,6%.

Cette analyse se trouve dans le notebook secondaire Projet ML - Random Search Long

Nous avons quand même essayé de rendre la régression logistique non linéaire en utilisant une méthode d'estimation du kernel polynomial (avec Nystroem) ce qui a seulement permis d'obtenir un f1 70%, mieux qu'avant, mais pas suffisant. On abandonne donc cette méthode de classification.

Comme nous restons avec uniquement des méthodes utilisant des arbres de classification, nous pouvons réinclure dans le dataset toutes les variables catégoriques. Cela augmente drastiquement notre score f1.

Nous essayons les méthodes de Random Forest et XGBoost avec 100, 500, 1000 et 1300 estimateurs et trouvons un f1 avec cross validation 5-fold de 81,9% pour XGBoost avec 500 estimateurs et aussi de 81,9% avec la Random Forest de 1000 estimateurs. Les différences ne sont cependant pas grandes pour la random forest entre 100 et 500

Nous choisissons donc de finir la recherche avec un Random Search sur un plus grand ensemble d'hyperparamètres.

Nous découvrons que malgré un score f1 atteignant les 88% sur les cross validations 5-fold, c'est un score absolument biaisé, et qui ne se généralise pas, ni sur un train/test (où l'on trouve un f1 de 82% environ), ni sur une RepeatedStratifiedKFold en 10 fold répété 10 fois avec des seeds différents, où l'on trouve des f1 de 82%

On n’arrive donc pas à dépasser la barre des 82% de f1, et tous les modèles semblent se valoir avec des auc aux alentours de 85%.

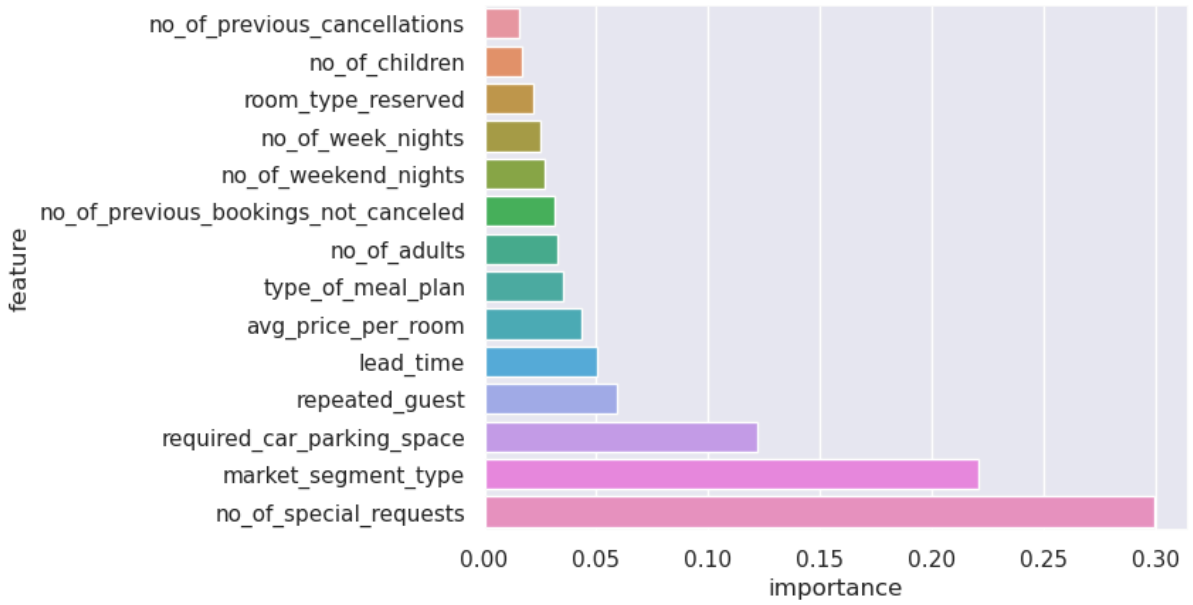
Réalisons à nouveaux les analyses précédentes avec les nouveaux hyperparamètres. Nous utiliserons le XGBoost avec ces hyperparamètres:
{'clf__n_estimators': 500,
 'clf__min_child_weight': 1,
 'clf__max_depth': 12,
 'clf__learning_rate': 0.05,
 'clf__gamma': 0.3,
 'clf__colsample_bytree': 0.5}

Trouvés grâce à un random search (bien qu’avec cross validation de seulement 5 folds).

II.5 - Les réservations générales avec la nouvelle méthode et hyperparamètres

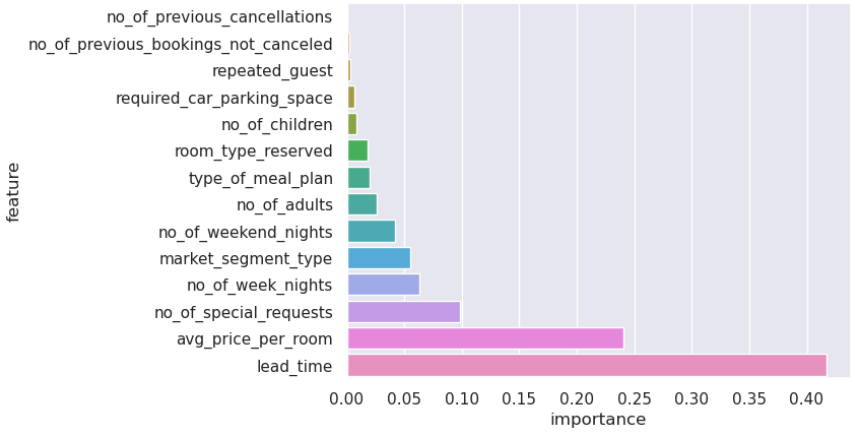
On effectue un RepeatedStratifiedKFold en 10 fold répété 5 fois avec des seeds différents, où l’on trouve un f1 de 82.6% sur le dataset complet.

Ce modèle XGB donne une forte importance à des features aussi bien numériques que catégoriques.

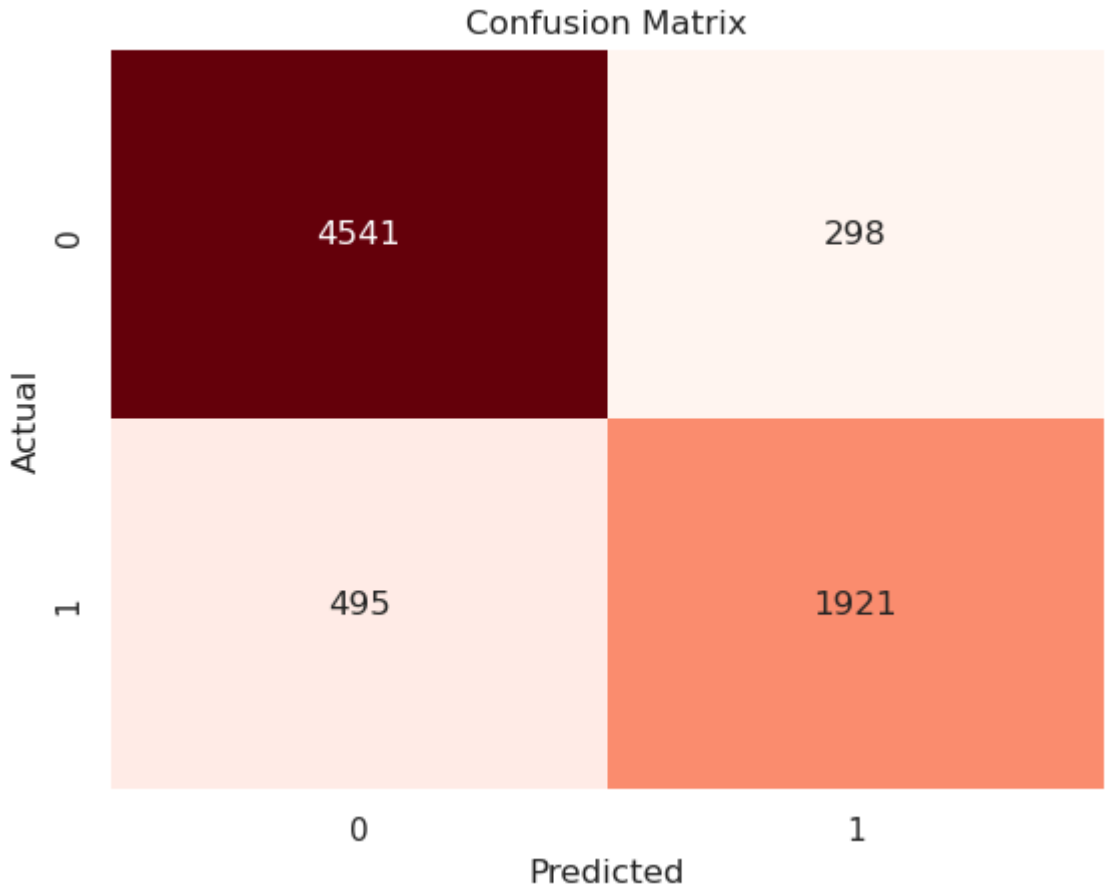


Ce modèle prend comme feature la plus importante dans sa classification le no_of_special_requests et le market_segment_type.

(Les modèles RandomForest ont plus tendance à donner de l'importance à nos variables numériques:)



On effectue un train/test split pour pouvoir afficher un rapport de classification plus en détail (même si potentiellement plus biaisé)



La matrice de confusion montre de plutôt bons résultats avec

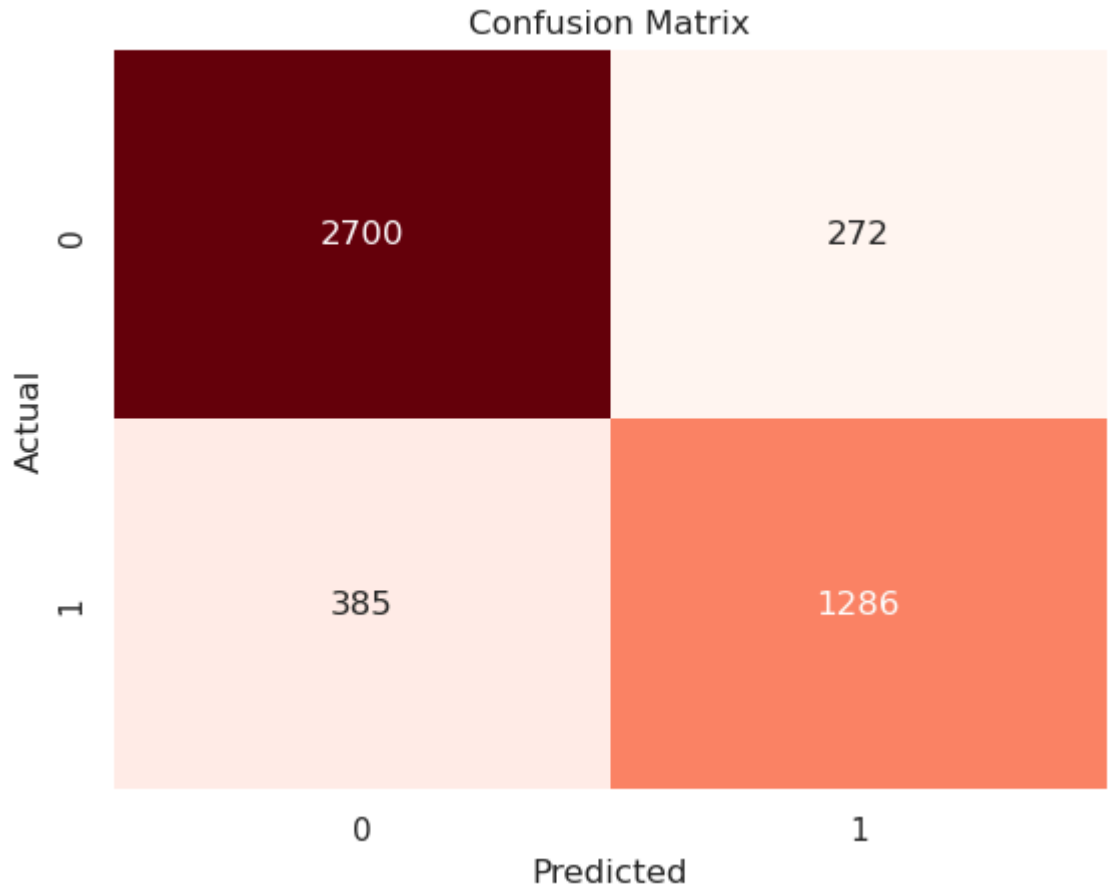
- f1-CV = 82,627%

sur la CV et sur le train/test:

- f1-train/test = 82,891%
- accuracy = 89,0696 %
- recall/sensibility = 79,512 %
- precision = 86,571 %
- auc = 86,6766 %

II.6 - Les réservations online avec la nouvelle méthode et hyperparamètres

Maintenant, regardons les performances du modèle entraîné sur des online uniquement, avec la même méthode et hyperparamètres



On trouve en effet après un train/test split que le modèle est légèrement moins performant sur les online uniquement même après avoir été réentrainé dessus:

- f1 = 0.7965314338804584,
- accuracy = 0.8584966616411803,
- recall = 0.7695990424895273,
- precision = 0.8254172015404364,
- auc = 0.8390390905583572

Mais ces métriques restent respectables.

Il serait intéressant d’effectuer un random search sur ces données restreintes au Online afin de trouver un modèle plus efficace dessus, mais nos essais étaient malheureusement sans succès.