Projet Machine Learning - Spaceship Titanic

EL GORGI Skander, BARNICHA Mohammed Yassine, AZZABI Mouhcin

École Nationale Supérieure d'Informatique pour l'Industrie et l'Entreprise

4 Avril 2024

Sommaire

Objectif et Description du Dataset

Objectif

- Analyser et prédire la variable cible *Transported* à partir de deux datasets .

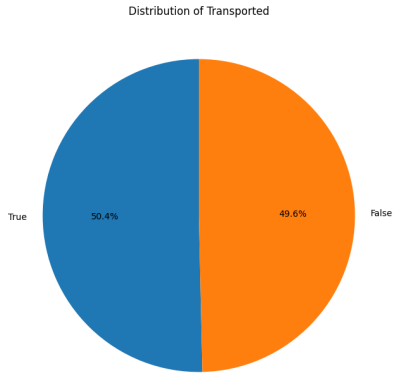
Description du Dataset

- La variable cible *Transported* indique l'état de transport du passager (True/False).
- Les datasets contiennent 13 features réparties en deux catégories :
  - **ID-like features** : Identifiants uniques par ligne, comme *PassengerId* et *Name*.
  - **Predictive features** : Variables prédictives incluant *Age*, *RoomService*, *CryoSleep*, etc.

Analyse des features et de la distribution du dataset

Table 1. Description of each attribute in our dataset

SL.No	Attribute	Description	Data Type
1	PassengerId	It represents the identification of the passenger	Object
2	HomePlanet	Passenger's planet of permanent residence	Object
3	CryoSleep	Indicates whether the passenger elected to be put into suspended animation for the duration of the voyage	Object
4	Cabin	Cabin number where passenger stay	Object
5	Destination	The planet the passenger will be debarking to.	Object
6	Age	Age of passengers on the ship	Float
7	VIP	Whether the passenger has paid for special VIP service during the voyage	Object
8	RoomService, FoodCourt, ShoppingMall, Spa and, VRDeck	The amount the passenger has billed at each of the Spaceship Titanic's many luxury amenities.	Float
13	Name	Name of the passenger	Object
14	Transported	Passengers whether transported or not during the voyage	Boolean



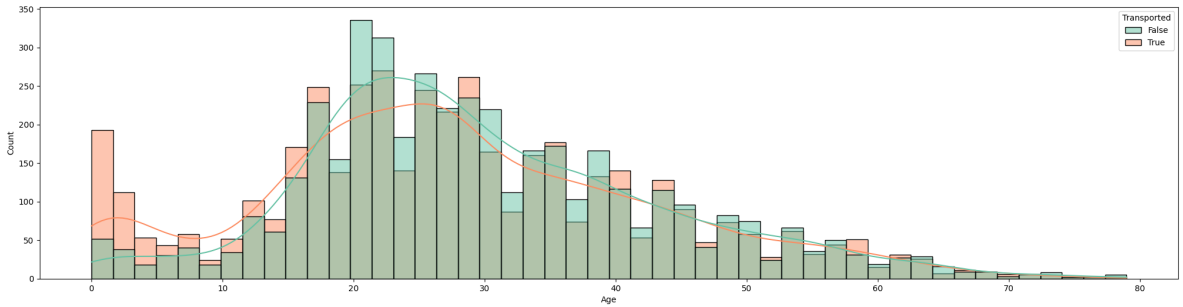
Prétraitement des Données

- le remplissage et l'encodage des variables.
- normalisation des variables .

Présentation des étapes du Feature Engineering.

- **NumRelatives** : basée sur les noms de famille pour détecter des liens familiaux et leur impact sur le transport.
- **Âge** : Identification des enfants de moins de 12 ans voyageant sans accompagnateurs comme un groupe systématiquement transporté. Creation de (Under12, Under12With0Relatives)
- **Cabin** : Découpage de la variable *Cabin* en pont, numéro et côté .
- **PassengerId** : Séparation de l'identifiant en groupes et numéros individuels pour examiner l'effet des structures sociales.
- **RoomService, FoodCourt, Spa, ShoppingMall** : Nous avons remarqué que la plupart des passagers ne dépensent rien ce qui mène à un fort skew en 0 de la distribution pour

Présentation des étapes du Feature Engineering.



	Age	Transported	NumRelatives
102	0.0	True	0
383	3.0	True	0
554	3.0	True	0
687	2.0	False	0
773	0.0	True	0
774	0.0	True	0
966	8.0	True	0
969	3.0	True	0
1034	5.0	True	0
1206	1.0	True	0
2150	1.0	True	0
2359	6.0	True	0
3421	0.0	True	0
3852	0.0	True	0
3937	12.0	True	0
4305	9.0	True	0
4409	2.0	True	0
4461	6.0	True	0
4486	0.0	True	0
4622	3.0	True	0
4631	4.0	True	0
4634	1.0	True	0
4703	5.0	True	0
4795	11.0	True	0
4823	4.0	True	0
4901	8.0	True	0
4930	8.0	True	0
5206	11.0	True	0
5535	6.0	True	0

Régression Logistique avec Régularisation

- Modèle de classification basé sur la probabilité:
- $P(y = 1|\mathbf{x}; \boldsymbol{\theta}) = \frac{1}{1+e^{-\mathbf{x}^\top \boldsymbol{\theta}}}$
- Ridge (L2) ajoute: $\lambda \sum_{j=1}^{n} \theta_j^2$ à la fonction de coût.
- Lasso (L1) ajoute: $\lambda \sum_{j=1}^{n} |\theta_j|$ à la fonction de coût.
- **Fonction de coût**

$$J(\boldsymbol{\theta}) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log(P(y = 1|\mathbf{x}^{(i)}; \boldsymbol{\theta})) + (1 - y^{(i)}) \log(1 - P(y = 1|\mathbf{x}^{(i)}; \boldsymbol{\theta}))\right] \quad (1)$$

- Optimisation de α (inverse de λ) par Grid Search pour contrôler la complexité du modèle.

Machine à Vecteurs de Support (SVM)

Objectif : Trouver l'hyperplan qui sépare de manière optimale les différentes classes dans l'espace des caractéristiques, maximisant ainsi la marge entre les classes.

Formulation mathématique de l'hyperplan :

Décision :

$$f(\mathbf{x}) = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b)$$

qui détermine la classe de $\mathbf{x}$ en fonction du signe de la fonction de décision. où $\mathbf{w}$ est un vecteur normal à l'hyperplan, $\mathbf{x}$ est un vecteur de caractéristiques, et b est un biais.

Extension aux cas non linéaires : Utilisation de fonctions de noyau pour transformer l'espace des caractéristiques.

Fonction de noyau RBF :

$$K(\mathbf{x_i}, \mathbf{x_j}) = \exp\left(-\gamma \|\mathbf{x_i} - \mathbf{x_j}\|^2\right)$$

Machine à Gradient Boosting (GBM)

Objectif

Minimiser une fonction de perte en ajoutant des arbres de décision faibles pour corriger les erreurs précédentes.

Fonction de perte à l'itération m

$$L(y, F_m(\mathbf{x})) = \sum_{i=1}^{n} L(y_i, F_{m-1}(\mathbf{x}_i) + \gamma_m h_m(\mathbf{x}_i))$$

Où L est la fonction de perte, y est la réponse, F_{m-1} est le modèle à l'itération précédente, h_m est l'arbre ajouté, et γ_m est le taux d'apprentissage.

Machine à Gradient Boosting (GBM)

Optimisation

À chaque étape, le nouvel arbre $h_m(\mathbf{x})$ est ajusté pour minimiser la fonction de perte.

Mise à jour du modèle

$$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \gamma_m h_m(\mathbf{x})$$

Avec F_m étant la prédiction mise à jour du modèle.

Random Forest et Arbres de Décision

Arbre de Décision

Un modèle de prédiction basé sur une série de questions binaires, aboutissant à une décision finale. Tout en minimisant les fonctions:

$$\text{Entropie}(t) = -\sum_{i=1}^{c} p(i|t) \log_2 p(i|t)$$

$$\text{Indice de Gini}(t) = 1 - \sum_{i=1}^{c} p(i|t)^2$$

Forêt Aléatoire

Un ensemble de nombreux arbres de décision pour améliorer la stabilité et la précision des prédictions.

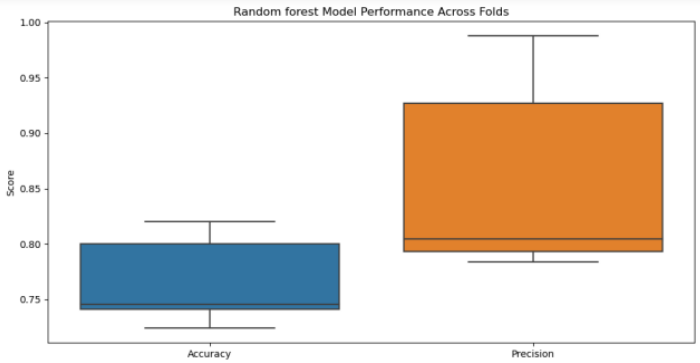
$$f(\mathbf{x}) = \frac{1}{B} \sum_{k=1}^{B} h(\mathbf{x}, \Theta_k)$$

- Les arbres sont construits sur des sous-ensembles de données aléatoires.
- Chaque arbre donne un vote pour la classification d'une observation.
- La classification finale est déterminée par un vote majoritaire.

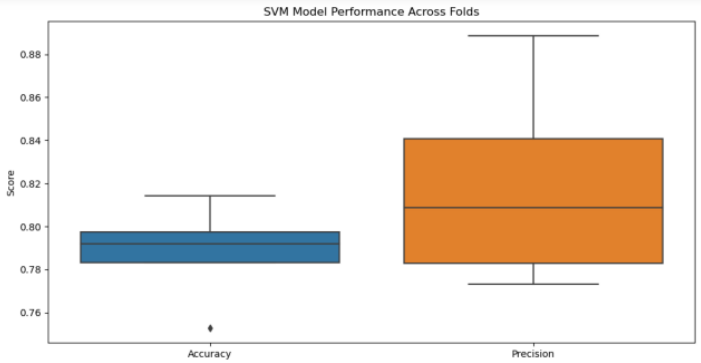
Comparaison des Modèles

- Afin de déterminer le modèle le plus performant, une recherche exhaustive, ou Grid Search, a été mise en œuvre pour chaque algorithme. Cette technique a permis d'identifier les hyperparamètres optimaux.
- Pour évaluer la robustesse des modèles, nous avons procédé à une validation croisée à 5 plis sur la précision et de l'exactitude .
- Les résultats sont synthétisés sous forme de boxplots.

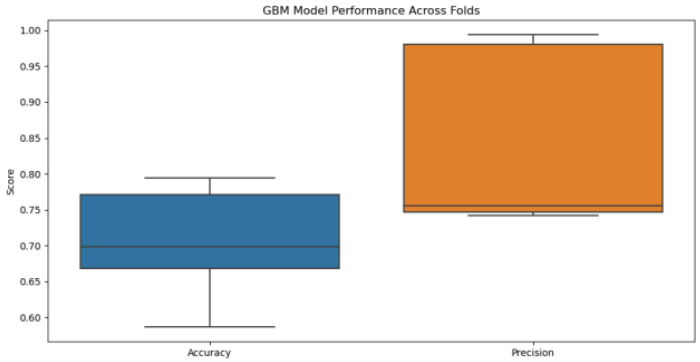
Comparaison des Performances des Modèles



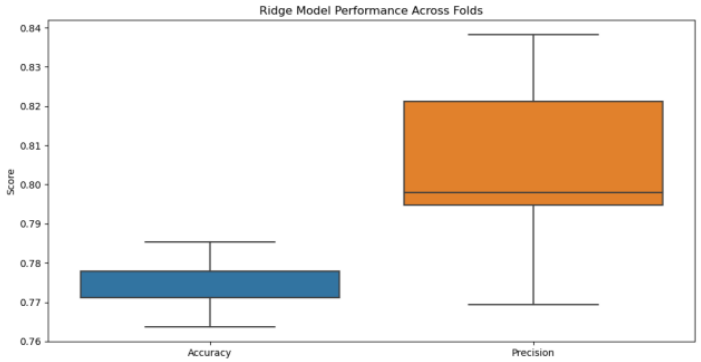
(a) Forêt aléatoire



(b) SVM



(c) GBM

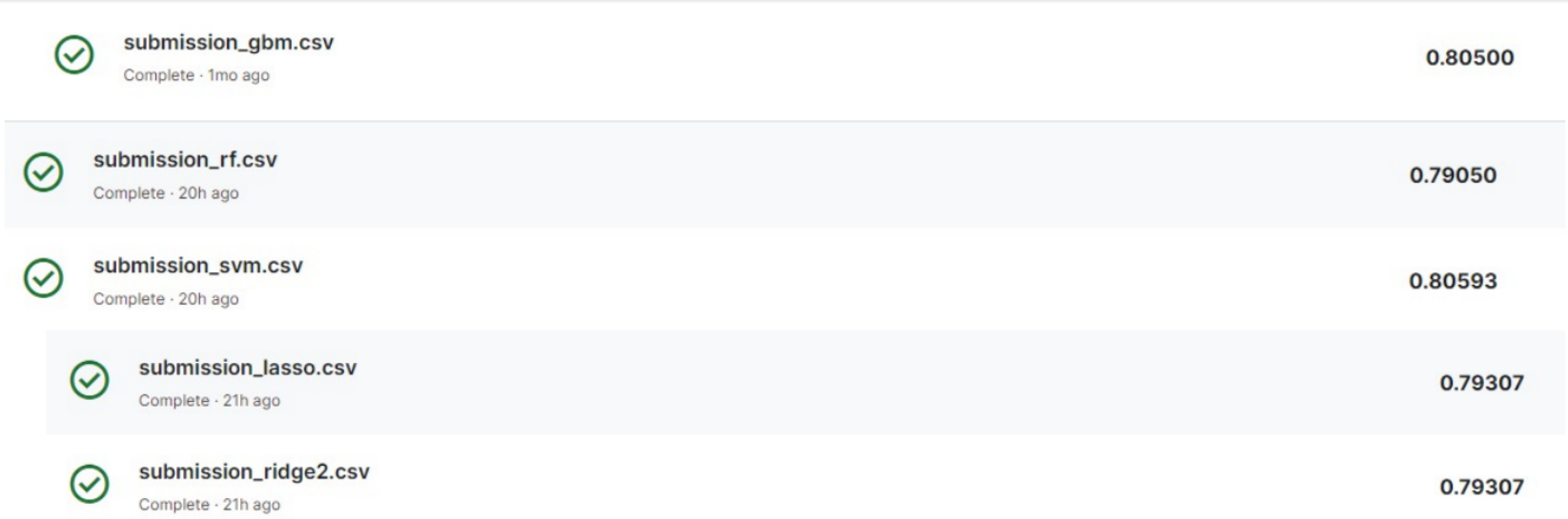


(d) Ridge

Comparaison des Performances des Modèles

La forêt aléatoire montre une précision constante avec un écart interquartile (EIQ) plus serré par rapport à son exactitude. Le SVM affiche un EIQ plus large en précision, ce qui suggère une variabilité plus importante dans ses performances. L'exactitude du GBM s'étend d'environ 0,65 à 0,85, indiquant une variabilité significative à travers les différents plis. Enfin, le modèle Ridge présente un écart d'exactitude restreint mais une large gamme de précision, ce qui indique une capacité stable à étiqueter correctement la classe positive mais avec une confiance variable à travers les différents plis.

Performance sous Kaggle



Scores de performance sur Kaggle:

- Forêt Aléatoire: 0.79682
- SVM: 0.80593
- GBM: 0.80453
- Lasso et Ridge: 0.79307

Classement Kaggle

430	**Yassine Barnicha**		0.80593

Conclusion

- le modèle SVM s'est avéré être le plus performant. Son score plus élevé sur la plateforme suggère qu'il a mieux généralisé sur les données de test, ce qui en fait notre choix préféré pour des prédictions précises et fiables dans des conditions réelles.
- il reste encore des choses a ameliorer et d'autres modeles a exploiter.

GRID SEARCH

Considérons un modèle M caractérisé par un ensemble de paramètres $\Theta = \{\theta_1, \theta_2, \ldots, \theta_n\}$. L'objectif de la recherche exhaustive est de trouver la combinaison des valeurs des paramètres à partir de leurs plages respectives qui résulte dans les meilleures performances du modèle.

- Soit V_i l'ensemble de toutes les valeurs possibles pour le paramètre θ_i.
- La grille $G = V_1 \times V_2 \times \ldots \times V_n$ représente toutes les combinaisons possibles des valeurs des paramètres.

L'algorithme de recherche exhaustive itère sur chaque vecteur $v \in G$ de valeurs des paramètres, évalue la performance du modèle M configuré avec v, et sélectionne la combinaison v^* qui maximise la mesure de performance :

$$v^* = \underset{v \in G}{\operatorname{argmax}}\, f(v)$$

où $f(v)$ est la métrique de performance du modèle M avec des paramètres fixés aux valeurs dans v.