Prédiction des passagers du Titanic téléportés

Soutenance de projet – MOST

Colin Coërchon, Maël Daunas,
Mohamed Koudiraty, Théo Taillefumier

ENSIIE

4 avril 2024

Informations clefs

- Professeurs référents : **Blaise Hanczar** et **Jean-christophe Janodet**
- Période de travail : du **5 février** 2024 au **4 avril** 2024

Sommaire

Un peu d'histoire

Figure – Le *Spaceship Titanic*

Nos objectifs dans ce projet

1. Analyser et comprendre les données fournies.
2. Transformer et compléter le jeu de données.

Nos objectifs dans ce projet

1. Analyser et comprendre les données fournies.
2. Transformer et compléter le jeu de données.
3. Élaborer **des modèles d'apprentissage automatique** afin de prédire les voyageurs téléportés.
4. Essayer de grimper dans **le classement Kaggle** !

Nos objectifs dans ce projet

1. Analyser et comprendre les données fournies.
2. Transformer et compléter le jeu de données.
3. Élaborer **des modèles d'apprentissage automatique** afin de prédire les voyageurs téléportés.
4. Essayer de grimper dans **le classement Kaggle** !
5. Interpréter nos résultats.

Exploration des données

PassengerId	HomePlanet	CryoSleep	Cabin	Destination	Age	VIP
0001_01	Europa	False	B/0/P	TRAPPIST-1e	39.0	False
0002_01	Earth	False	F/0/S	TRAPPIST-1e	24.0	False
0003_01	Europa	False	A/0/S	TRAPPIST-1e	58.0	True
0003_02	Europa	False	A/0/S	TRAPPIST-1e	33.0	False
0004_01	Earth	False	F/1/S	TRAPPIST-1e	16.0	False

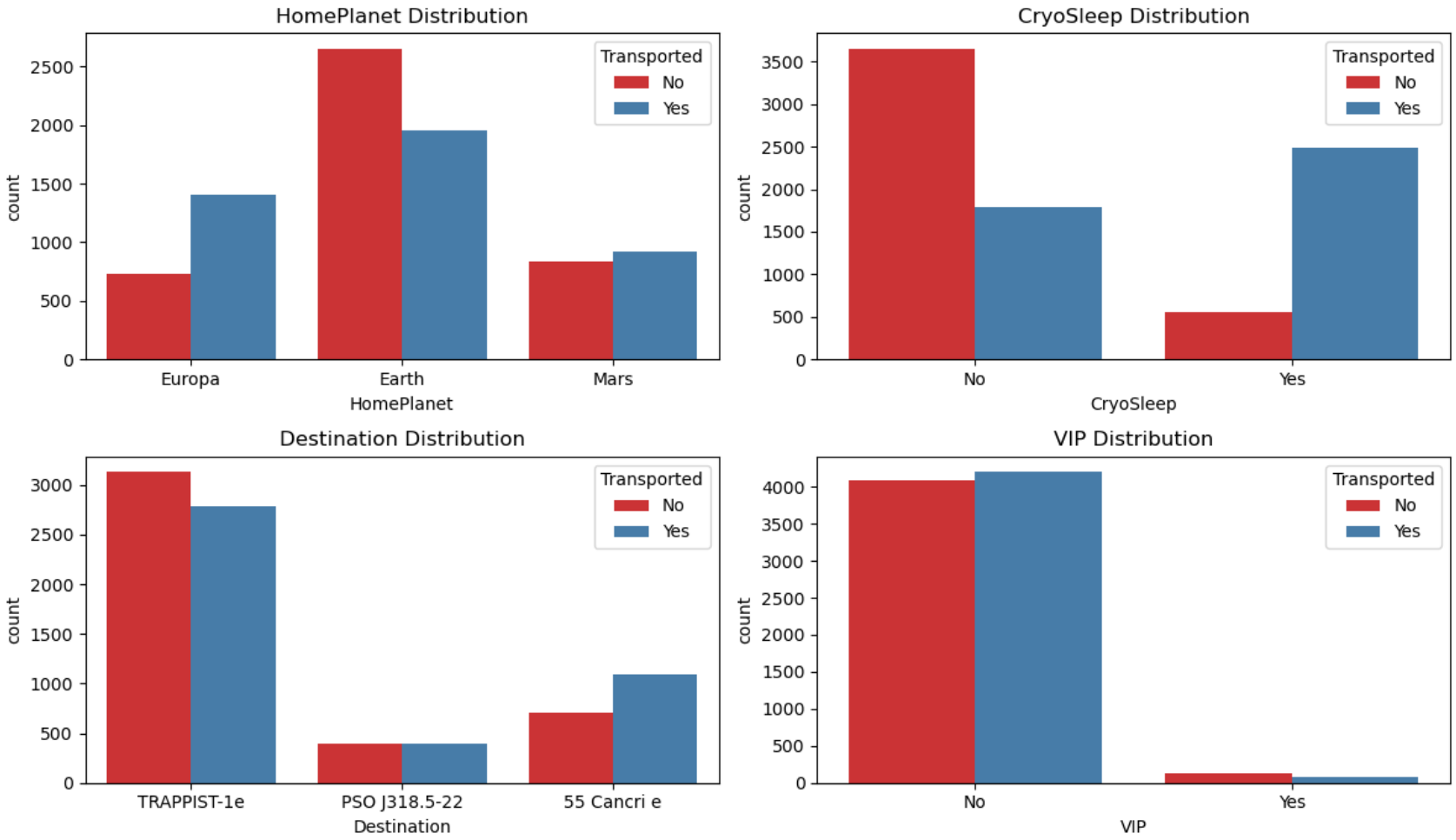
RoomService	FoodCourt	ShoppingMall	Spa	VRDeck	Name
0.0	0.0	0.0	0.0	0.0	Maham Ofracculy
109.0	9.0	25.0	549.0	44.0	Juanna Vines
43.0	3576.0	0.0	6715.0	49.0	Altark Susent
0.0	1283.0	371.0	3329.0	193.0	Solam Susent
303.0	70.0	151.0	565.0	2.0	Willy Santantines

Transported
False
True
False
False
True

Caractéristique	Cardinalité
PassengerId	8693
HomePlanet	3
CryoSleep	2
Cabin	6560
Destination	3
VIP	2
Name	8473

Table – Cardinalité des caractéristiques catégorielles dans les ensembles de données d'entraînement.

Les premières analyses



L'âge des passagers

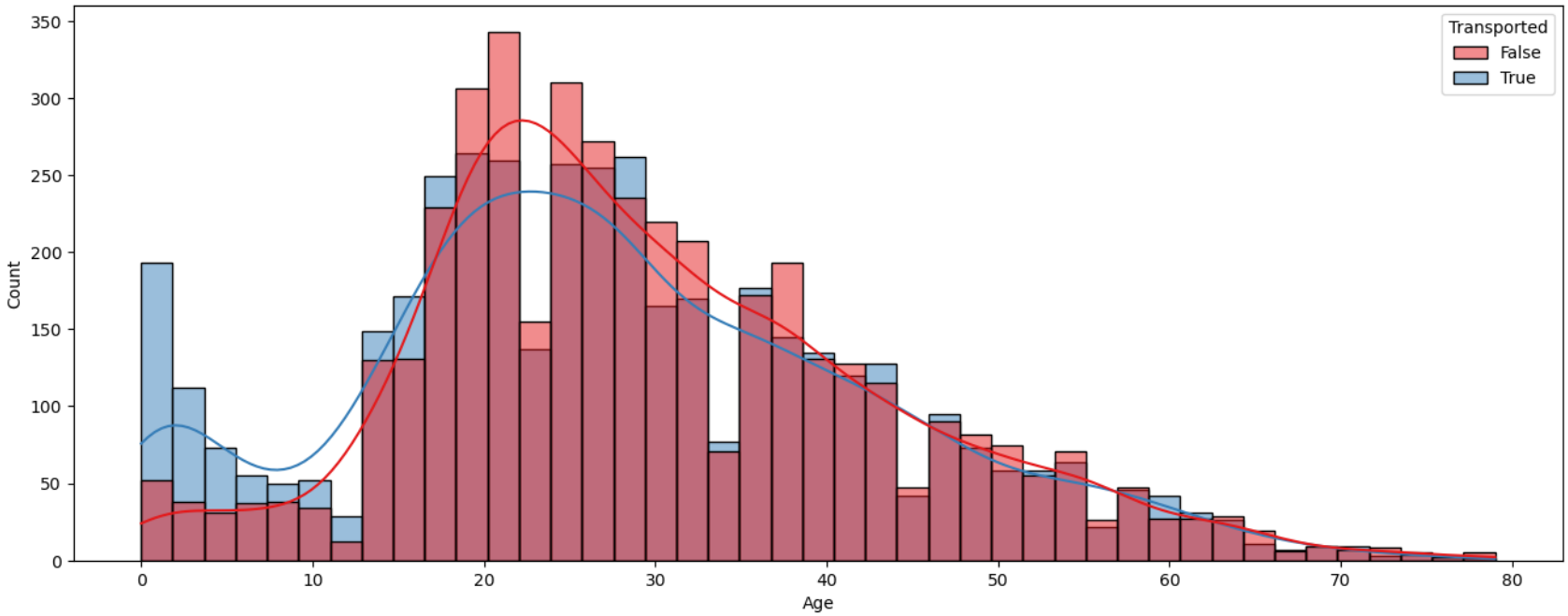
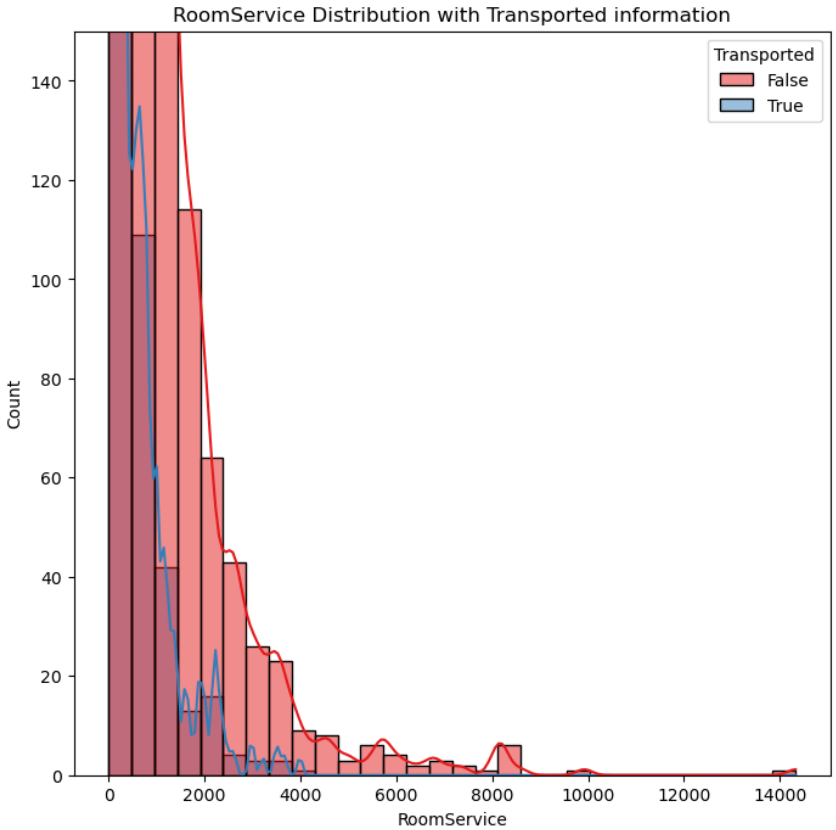
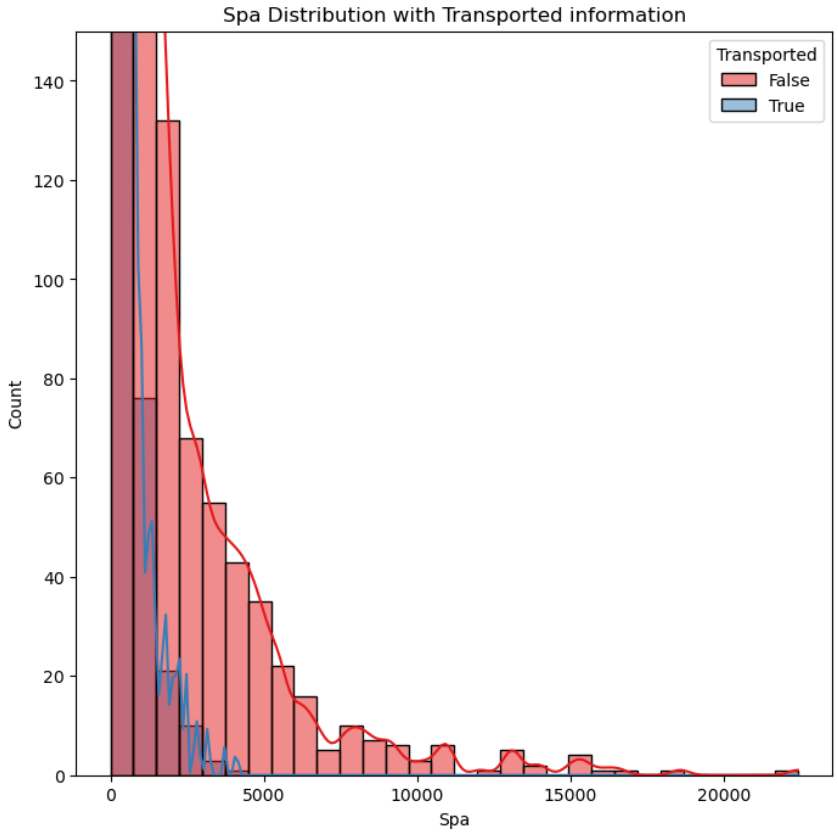


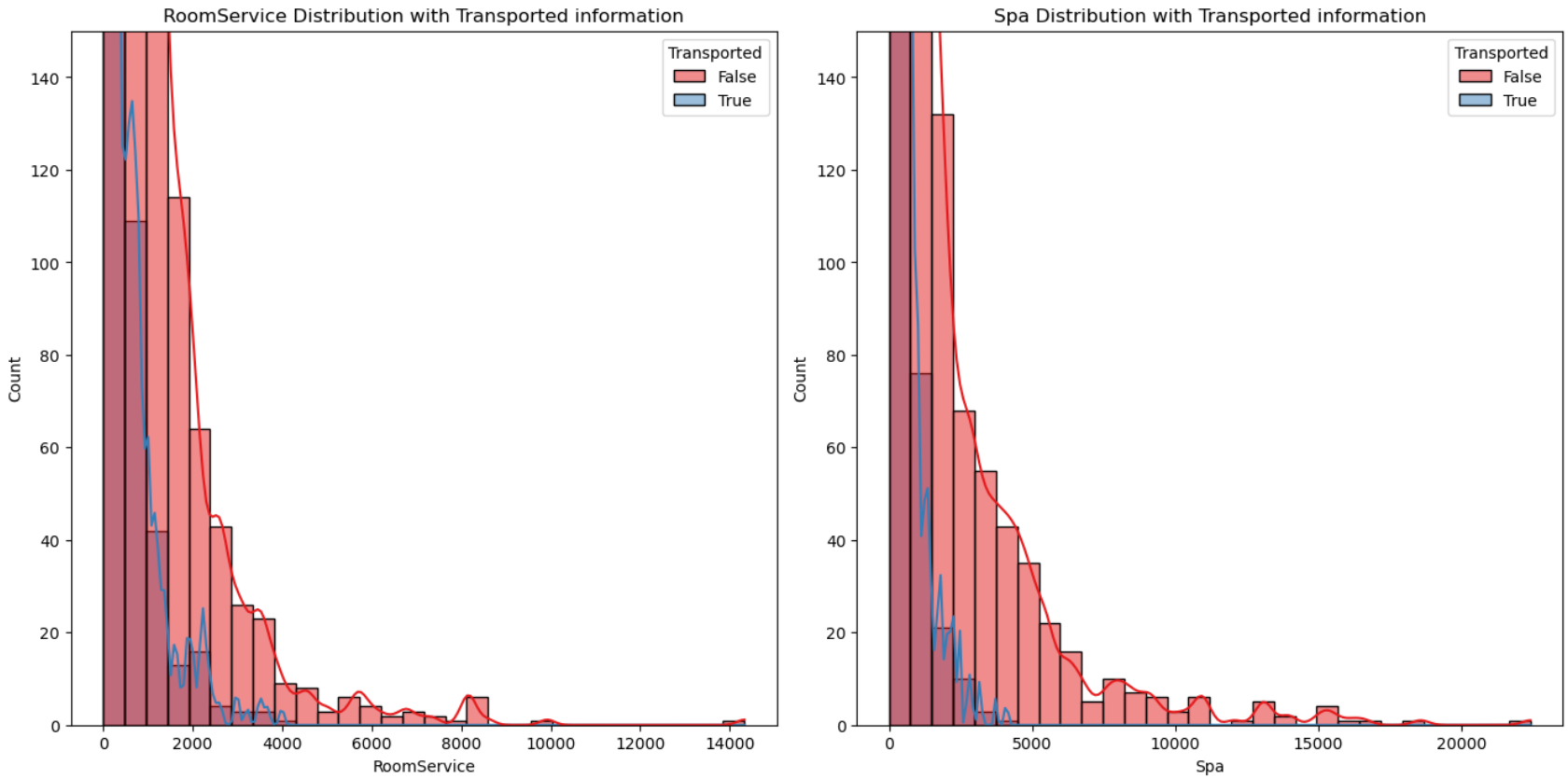
Figure – Distribution en fonction de l'âge des passagers

Les dépenses des passagers



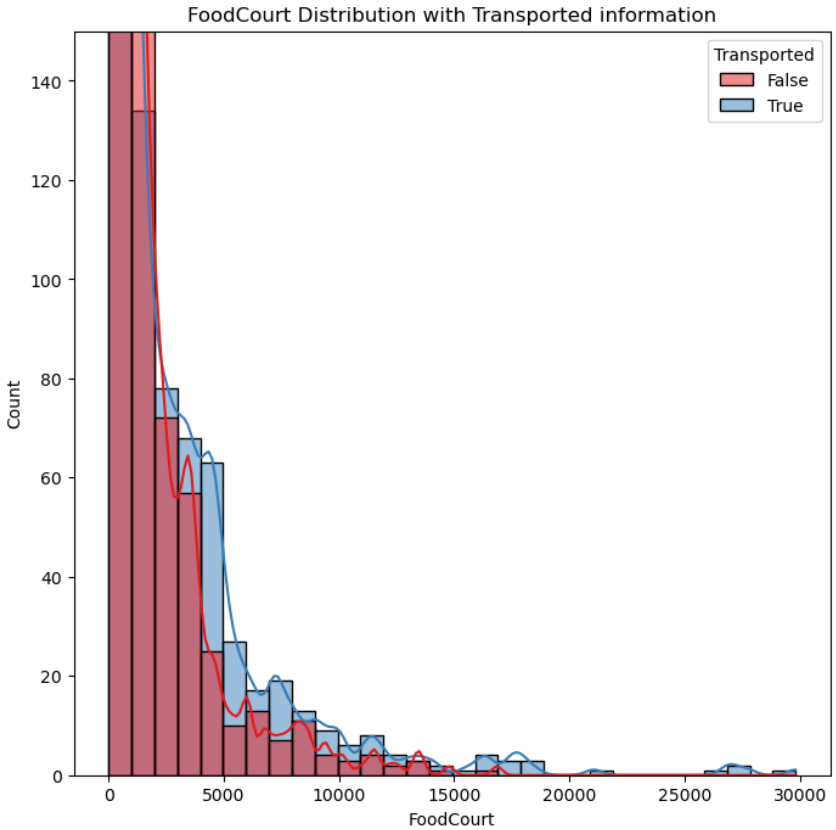


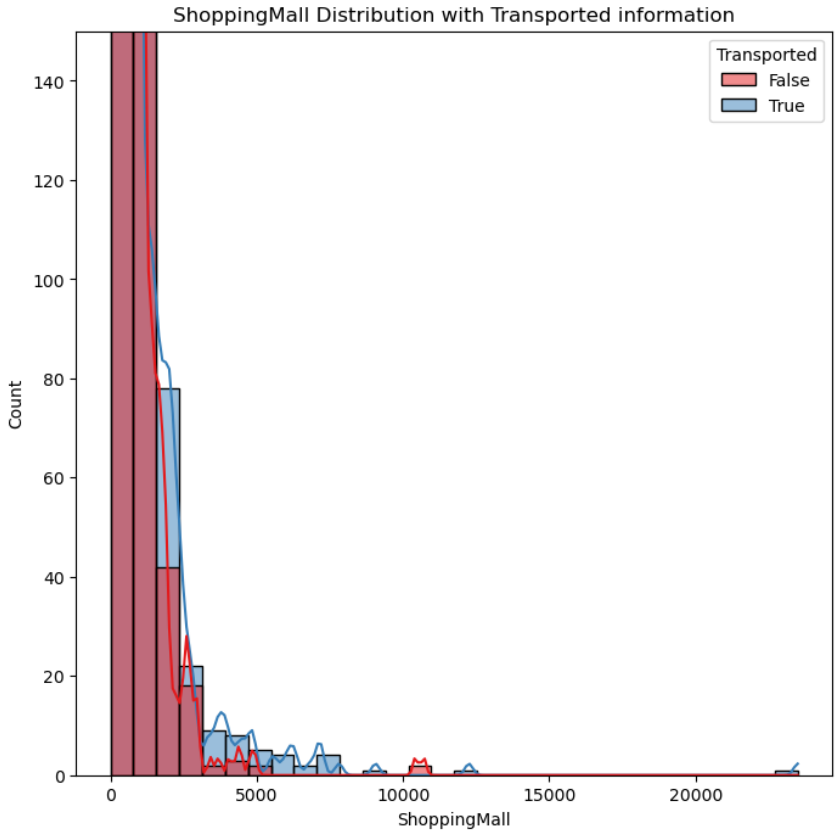
Les dépenses des passagers



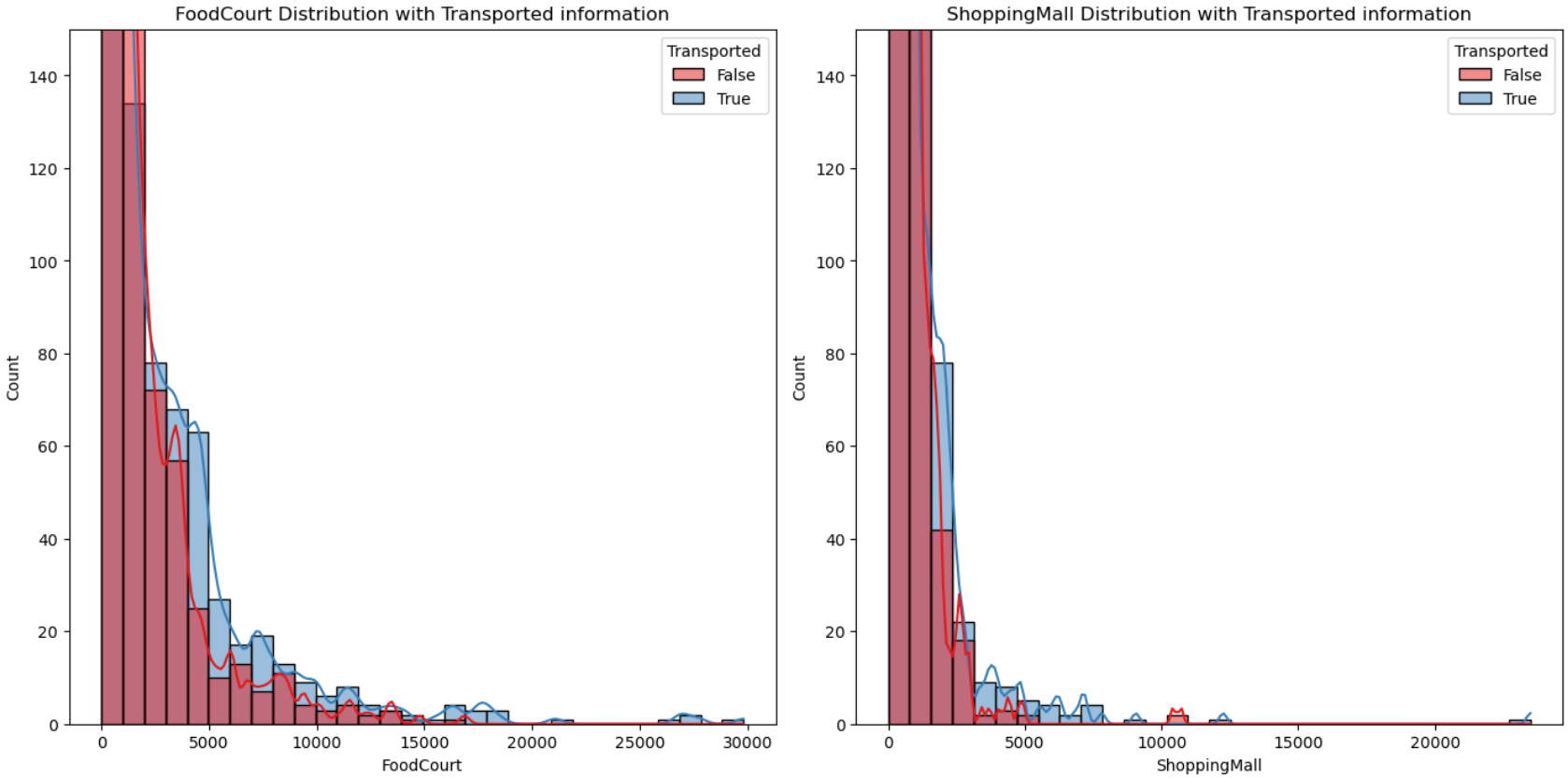
Ici, **les moins dépensiers** sont davantage transportés.

Les dépenses des passagers





Les dépenses des passagers



À l'inverse, ici, **les plus dépensiers** sont davantage transportés.

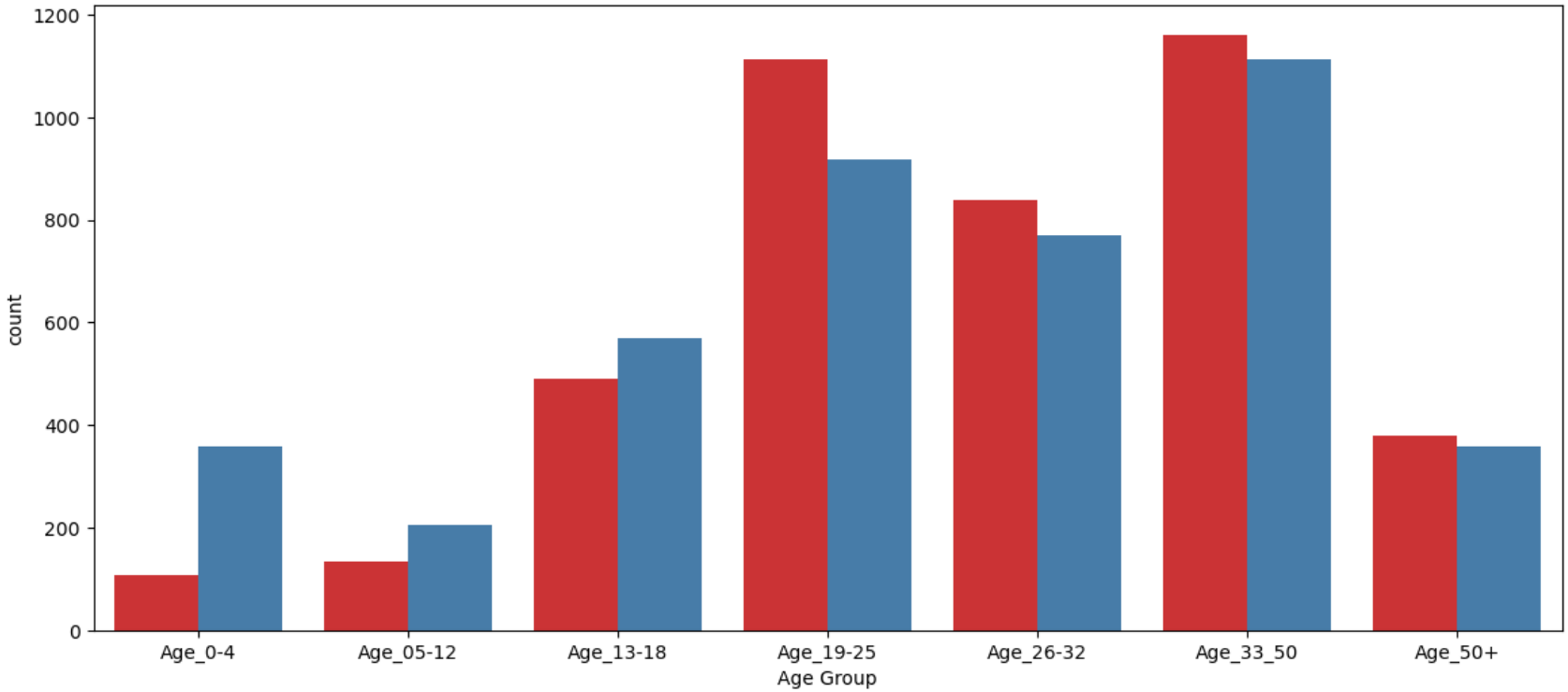
Transformations des données

Problème en machine learning

Variables inutiles ? Données manquantes ? Données numériques trop variées ?

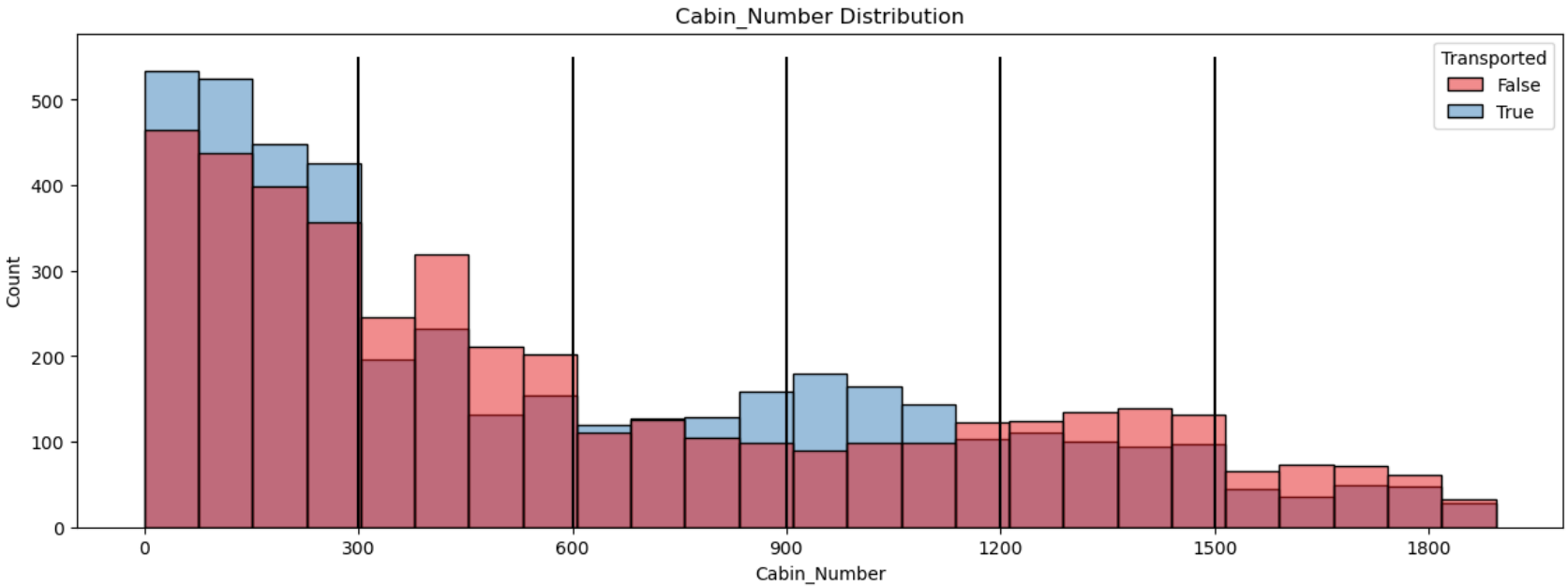
Des catégories d'âge

Regroupement par catégories d'âge $\Longrightarrow$ création de "`Age Group`"



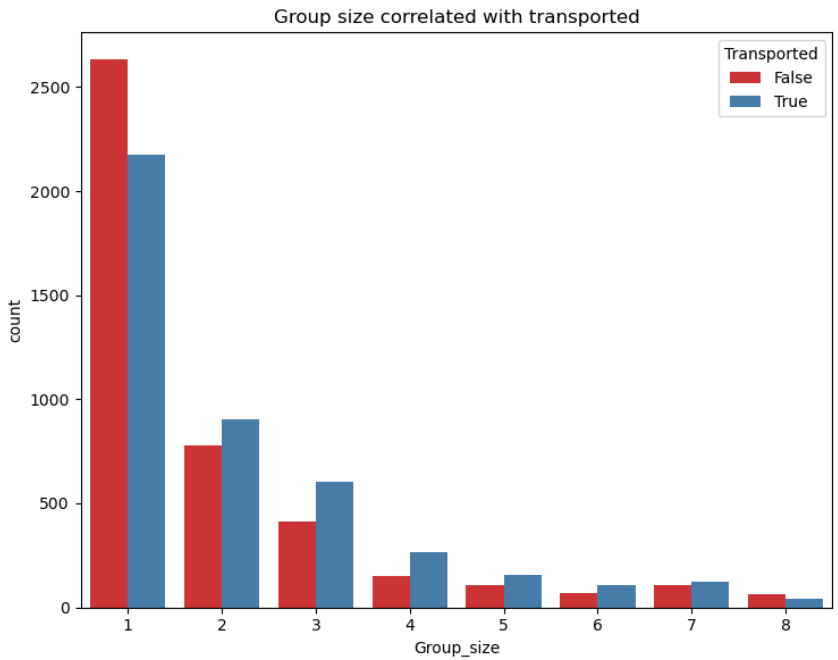
Transformation de la variable Cabin

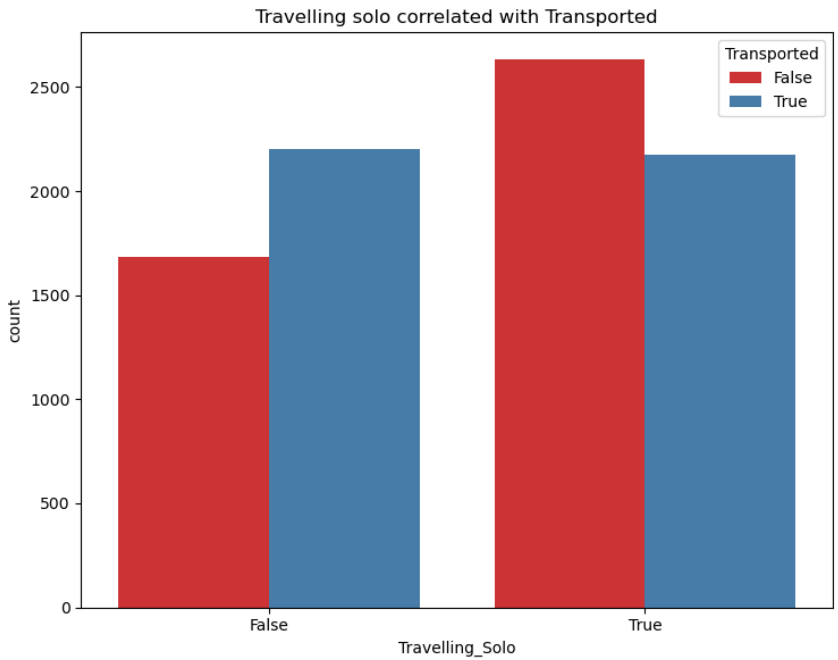
Exemple d'instance de Cabin : (F/1/S).
On crée alors les variables : Cabin_Deck, Cabin_Number et Cabin_Size.



Transformation de la variable PassengerId

Exemple d'instance de PassengerId : 0003_02.
On crée alors les variables : Group et Member. Ainsi que les variables Group_size et Travelling_Solo





Ajout des dépenses totales

On crée ici la variable "`Total Expenditure`" (et "`No Spending`") qui correspond à la somme des variables de dépenses : `RoomService`, `FoodCourt`, `ShoppingMall`, `Spa` et `VRDeck`.

Remplacement des données manquantes

Les données manquantes :

Variable	Nombre de valeurs manquantes	% de valeurs manquantes
HomePlanet	201	2.31%
CryoSleep	217	2.50%
Cabin	199	2.29%
Destination	182	2.09%
Age	179	2.06%
VIP	203	2.34%
RoomService	181	2.08%
FoodCourt	183	2.11%
ShoppingMall	208	2.39%
Spa	183	2.11%
VRDeck	188	2.16%
Name	200	2.30%

Table – Résumé des valeurs manquantes dans les données du *Spaceship Titanic* (sur un total de 8693 valeurs)

Gestion des données manquantes

Stratégie simplifiée :

1. **Pour les passagers en groupe** :
   - Attributs numériques : Utiliser la *moyenne* du groupe.
   - Attributs catégoriels : Utiliser la *valeur la plus fréquente* dans le groupe.

Gestion des données manquantes

Stratégie simplifiée :

1. **Pour les passagers en groupe** :
   - Attributs numériques : Utiliser la *moyenne* du groupe.
   - Attributs catégoriels : Utiliser la *valeur la plus fréquente* dans le groupe.
2. **Pour les passagers seuls** :
   - Attributs numériques : Se baser sur la *moyenne* de passagers dans des conditions de cabine similaires.
   - Attributs catégoriels : Prendre la *valeur la plus commune* parmi ces passagers.

Note : Les "conditions de cabine similaires" incluent la région de la cabine, le pont et le côté du vaisseau.

Dans notre étude, la variable cible Y est une variable **binaire**.
Nous passons donc par une **régression logistique**

Modèle de régression logistique

Dans ce modèle, on part du principe que les observations y_i sont des réalisations de variables aléatoires Y_i indépendantes, de loi de Bernoulli de paramètre $p_\beta(x_i)$ tel que :

$$\text{logit}\, p_\beta(x_i) = \ln\left(\frac{p_\beta(x_i)}{1 - p_\beta(x_i)}\right) = \beta_1 x_{i,1} + \cdots + \beta_p x_{i,p} = \beta^T x_i$$

Où la fonction de transfert est donnée par **la fonction sigmoïde** :

$$p_\beta(x_i) = \frac{e^{\beta^T x_i}}{1 + e^{\beta^T x_i}}$$

Premiers résultats

Régression Logistique
$\begin{pmatrix} 643 & 220 \\ 174 & 702 \end{pmatrix}$
0.7734330

Table – Résultats du modèle de Régression Logistique

Méthode de régression pénalisée

Présentation de la méthode de régression régularisée Elastic Net

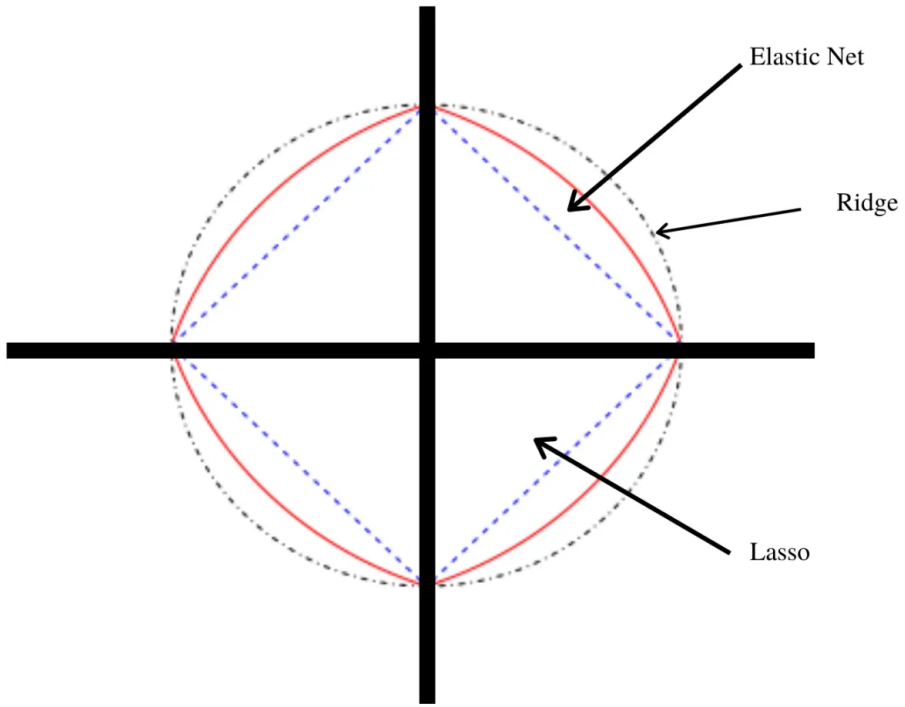


Figure – Explication visuelle derrière "Elastic Net"

Méthode de régression pénalisée

Cela permet de bénéficier d'un équilibre entre sélection de variables et régularisation, qui est contrôlé par deux hyperparamètres principaux :

- `C` : Inverse de la force de régularisation. Plus C est petit, plus la régularisation est forte.
- `L1_ratio` : Le ratio de mélange entre la régularisation Lasso et Ridge.

Résultats :

Hyperparamètre	**Valeur Optimale**
`L1_Ratio`	0.7431
`C`	0.0336

Comparaison des modèles

Modèle L1	**Modèle L2**	**Elastic Net**
$\begin{pmatrix} 643 & 220 \\ 176 & 700 \end{pmatrix}$	$\begin{pmatrix} 643 & 220 \\ 174 & 702 \end{pmatrix}$	$\begin{pmatrix} 649 & 214 \\ 181 & 695 \end{pmatrix}$
0.7723	0.7734	0.7737

Table – Résultats des modèles de régression régularisée

Principe des SVM

Les SVM cherchent **le meilleur hyperplan** qui maximise la **marge** entre les deux classes de données.
Dans le modèle des SVM, le **choix du noyau** est vraiment important.

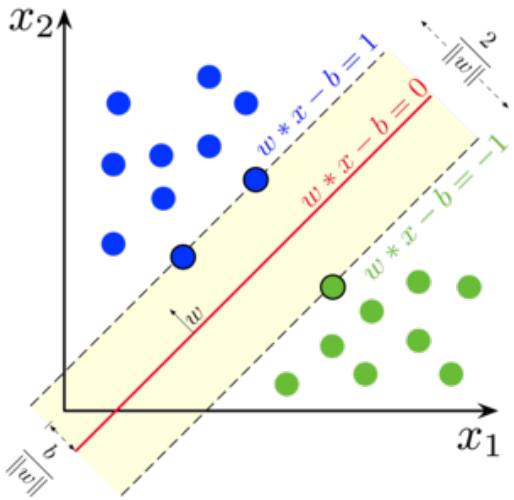


Figure – Principe de fonctionnement des SVM

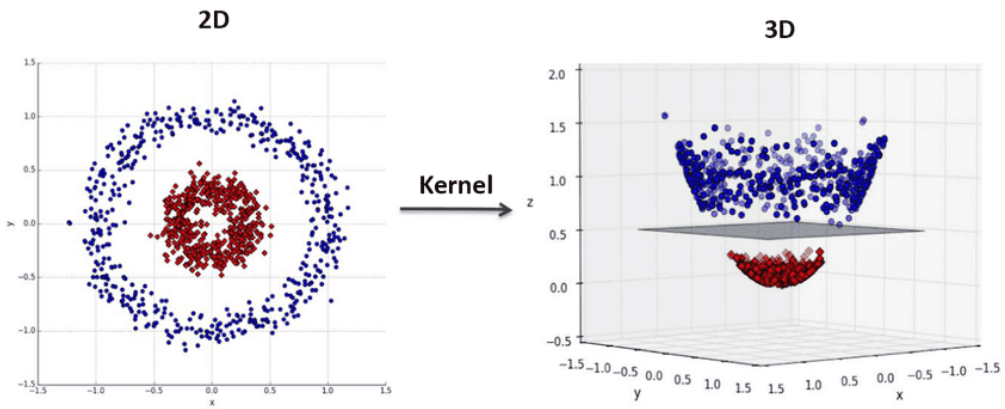


Figure – Exemple de l'utilisation d'un noyau gaussien

Brève explication mathématique des SVM

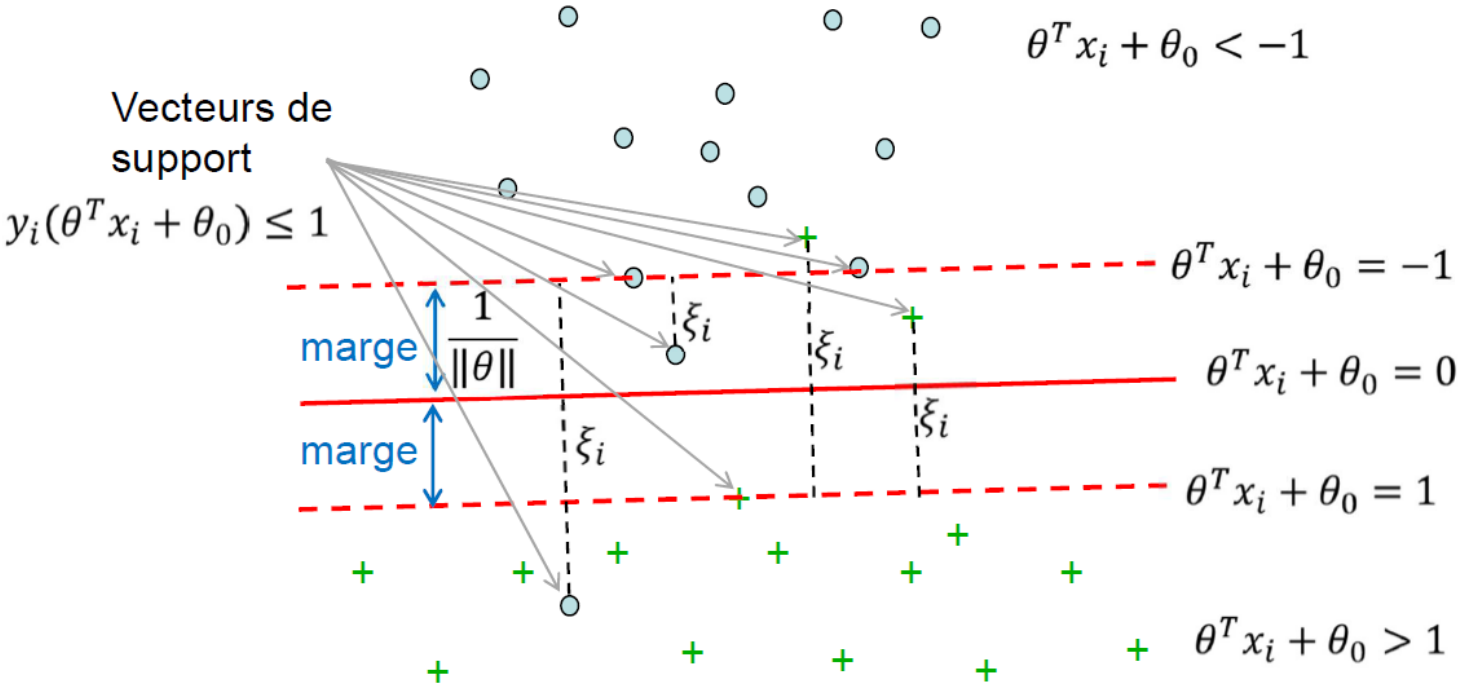
Le problème d'optimisation

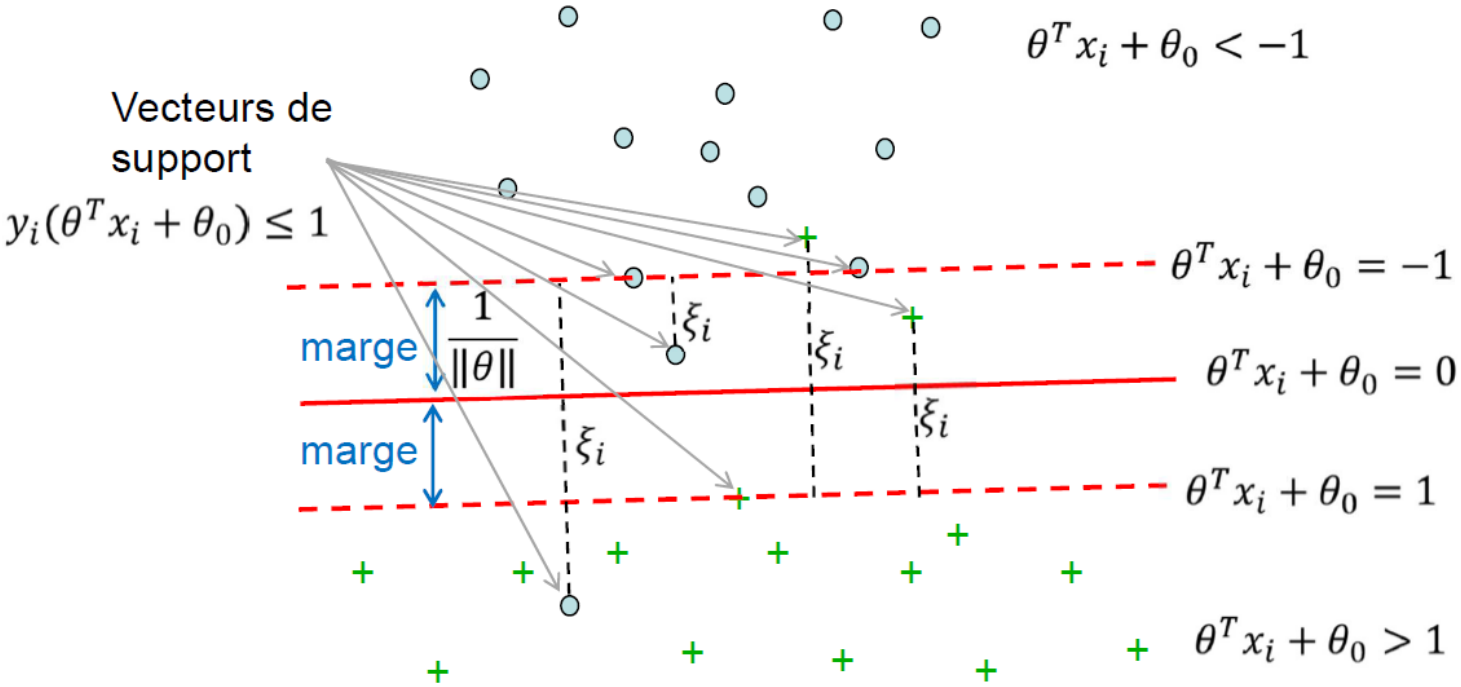
Le SVM cherche à résoudre ce problème d'optimisation :

$$\min_{\theta,\theta_0,\xi} \quad \frac{1}{2}\theta^T\theta + C\sum_{i=1}^{n}\xi_i$$

$$\text{sous contraintes} \quad \begin{cases} y_i(\theta^T x_i + \theta_0) \geq 1 - \xi_i, \\ \xi_i \geq 0. \end{cases}$$

avec : $\theta =$ poids, $\theta_0 =$ biais, $\xi =$ variables de relâchement


$\theta^T x_i + \theta_0 < -1$
Vecteurs de support
$y_i(\theta^T x_i + \theta_0) \leq 1$
$\theta^T x_i + \theta_0 = -1$
marge
$\frac{1}{\|\theta\|}$
ξ_i
ξ_i
ξ_i
$\theta^T x_i + \theta_0 = 0$
marge
ξ_i
$\theta^T x_i + \theta_0 = 1$
$\theta^T x_i + \theta_0 > 1$



Fonction de décision

$$\begin{cases} h(x) = \theta^T \phi(x) + \theta_0, \\ G(x) = \text{sign}(h(x)). \end{cases}$$

Les hyperparamètres (noyau gaussien)

- `C` : Inverse de la force de régularisation. Plus C est élevé, plus le modèle sera sensible aux données d'entraînement, au risque de surapprentissage.
- `gamma` : Paramètre du noyau gaussien. Un γ faible définit une gaussienne avec une grande variance.
- `kernel` : Type de noyau utilisé. Ici, `rbf` indique un noyau gaussien,

Paramètre	**Valeur Optimale**
`C`	15.840958
`gamma`	0.0086557
`kernel`	rbf

Table – Résultats de l'optimisation des hyperparamètres pour SVM avec Noyau Gaussien

Résultats du modèle SVM

Avec **le noyau gaussien**, on a :

Modèle SVM
$\begin{pmatrix} 679 & 184 \\ 172 & 704 \end{pmatrix}$
0.7953

Table – Matrice de confusion et précision du modèle SVM

Les autres noyaux ?

Nous avons testé d'autres types de noyaux : linéaire, polynomial et sigmoïde.
Malgré des essais d'optimisation d'hyperparamètres, la précision restait en dessous de $0,795$.

Description du Modèle

- Algorithme de classification basé sur le choix des features pour prédire les passagers transportés
- Division des données en sous-groupes avec des caractéristiques significatives pour la décision

Mesure d'impureté pour optimiser la séparation des données

$$G(N) = G(p_N) = \sum_{k=0}^{1} p_{N,k}(1 - p_{N,k}) \qquad \text{(Indice de Gini)}$$

$$H(N) = H(p_N) = -\sum_{k=0}^{1} p_{N,k} \log_2(p_{N,k}) \qquad \text{(Indice d'Entropie)}$$

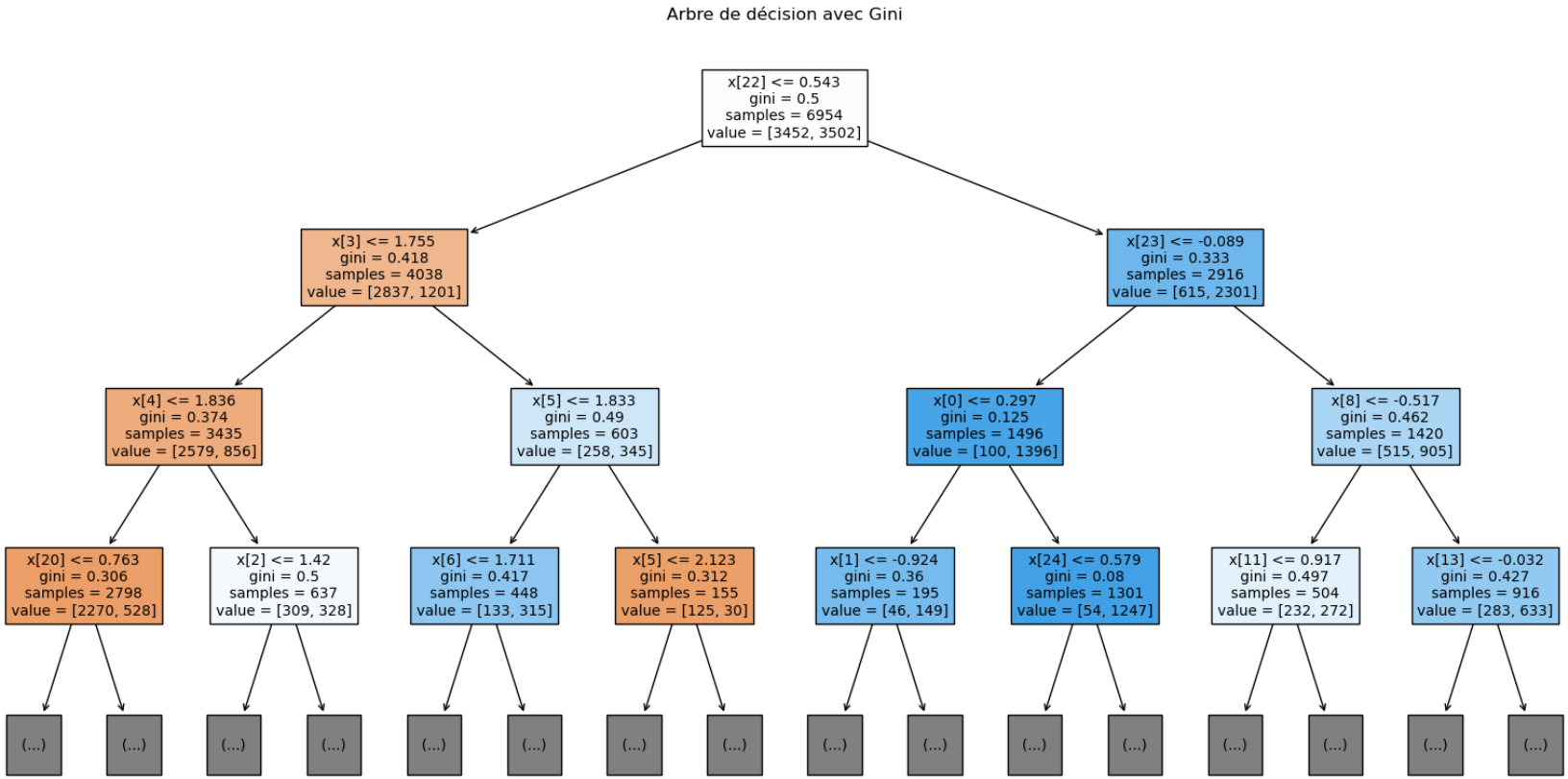
Mesure d'impureté pour optimiser la séparation des données

$$G(N) = G(p_N) = \sum_{k=0}^{1} p_{N,k}(1 - p_{N,k}) \qquad \text{(Indice de Gini)}$$

$$H(N) = H(p_N) = -\sum_{k=0}^{1} p_{N,k} \log_2(p_{N,k}) \qquad \text{(Indice d'Entropie)}$$

- **Indice de Gini** ($G(N)$) : Probabilité qu'un élément soit mal classé si on le choisit au hasard selon la distribution de classe dans le nœud N
- **Indice d'Entropie** ($H(N)$) : Degré de désordre ou d'incertitude dans les informations présentées par le nœud N

Visualisation de l'Arbre de Décision "gini"



Résultats du modèle des Arbres de décision

Critère	Max Depth	Min Samples Leaf	Min Samples Split	Meilleur Accuracy
Gini	7	9	19	0.78861106
Entropy	9	16	11	0.78817847

Table – Optimisation des hyperparamètres pour les critères "Gini" et "Entropy"

Résultats Finaux
$\begin{pmatrix} 656 & 207 \\ 169 & 707 \end{pmatrix}$
0.7886

Table – Précision et matrice de confusion du modèle final des Arbres de décision

Description du modèle

Le Boosting

C'est un algorithme qui vise à créer **un modèle prédictif fort** à partir d'une série de **modèles faibles**.
L'idée repose sur un système de pondération des données d'apprentissage. À chaque étape, on modifie les **poids** pour concentrer l'algorithme sur des exemples mal classés.

Description du modèle

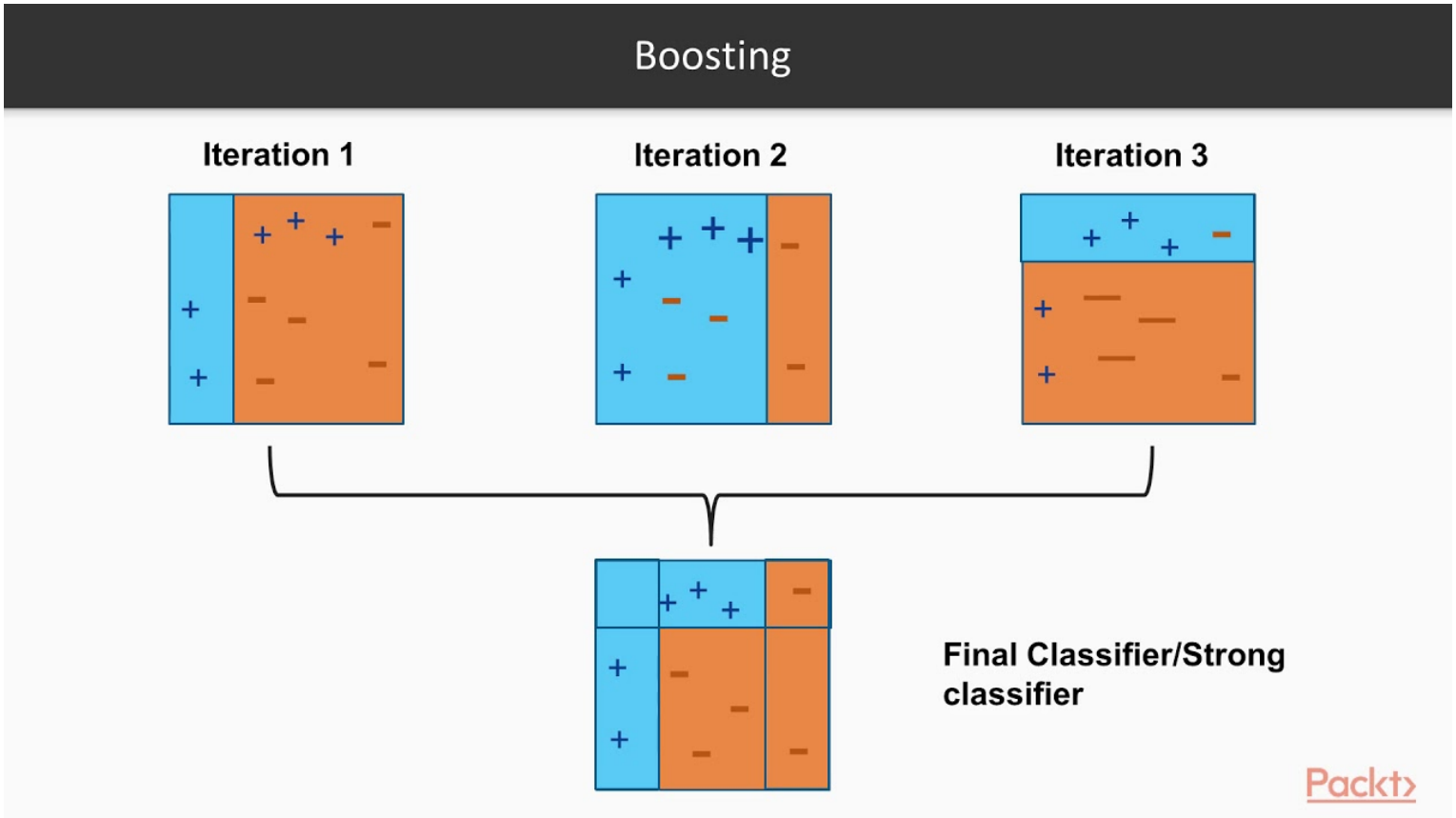
Le Boosting

C'est un algorithme qui vise à créer **un modèle prédictif fort** à partir d'une série de **modèles faibles**.
L'idée repose sur un système de pondération des données d'apprentissage. À chaque étape, on modifie les **poids** pour concentrer l'algorithme sur des exemples mal classés.

Le pseudo-code de l'algorithme (vu en cours) est disponible dans le rapport page 25.

Le calcul phare de la re-pondération

$$\forall t \in [\![1, T]\!], \forall i \in [\![1, n]\!], \quad \omega_{t+1}(x_i) = \frac{\omega_t(x_i)\, e^{-\alpha_t y_i h_i(x_i)}}{Z_t}$$



Les hyperparamètres

les hyperparamètres suivants doivent être soigneusement ajustés :

- `n_estimators` : Le nombre d'arbres à inclure dans le modèle. Attention au surapprentissage.
- `learning_rate` : Contrôle la contribution de chaque arbre au modèle final.
- `estimator` : Nous utilisons des arbres de décision.

Paramètre	**Valeur Optimale**
`estimator__max_depth`	2
`learning_rate`	0.02915
`n_estimators`	871

Table – Résultats de l'optimisation des hyperparamètres pour AdaBoost

Résultat de l'algorithme AdaBoost

Avec comme ensemble de modèles "faibles" des arbres de décision :

Modèle AdaBoost
$\begin{pmatrix} 652 & 211 \\ 138 & 738 \end{pmatrix}$
0.79988

Table – Matrice de confusion et précision de l'algorithme AdaBoost

Conclusion

Résumé de notre travail

Notre travail est ainsi passé par différentes étapes.

- Travail important d'analyse et de transformation des données, puis utilisation de **28 variables prédictrices** dans l'objectif de renforcer la précision des prédictions.

Conclusion

Résumé de notre travail

Notre travail est ainsi passé par différentes étapes.

- Travail important d'analyse et de transformation des données, puis utilisation de **28 variables prédictrices** dans l'objectif de renforcer la précision des prédictions.
- Évaluation de divers modèles de machine learning, y compris **la régression logistique**, **Elastic Net**, **SVM**, **arbres de décision**, et boosting avec **AdaBoost**.

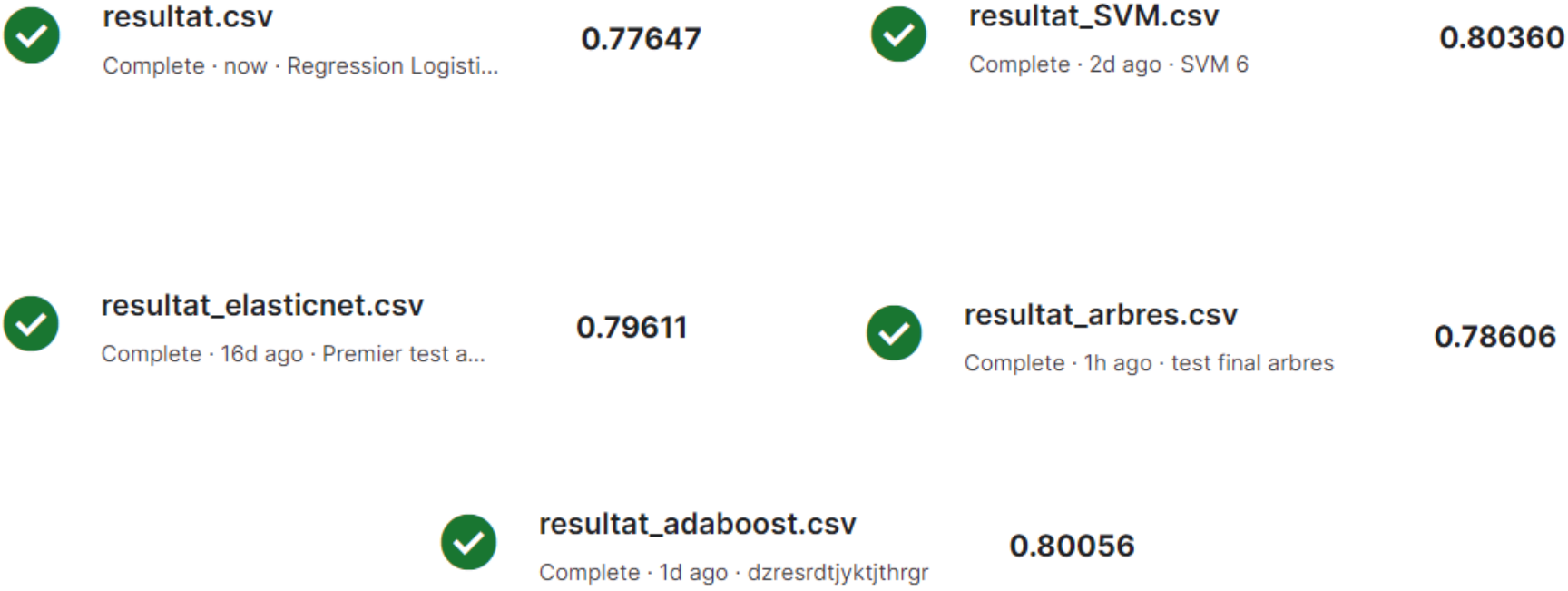
Conclusion

Résumé de notre travail

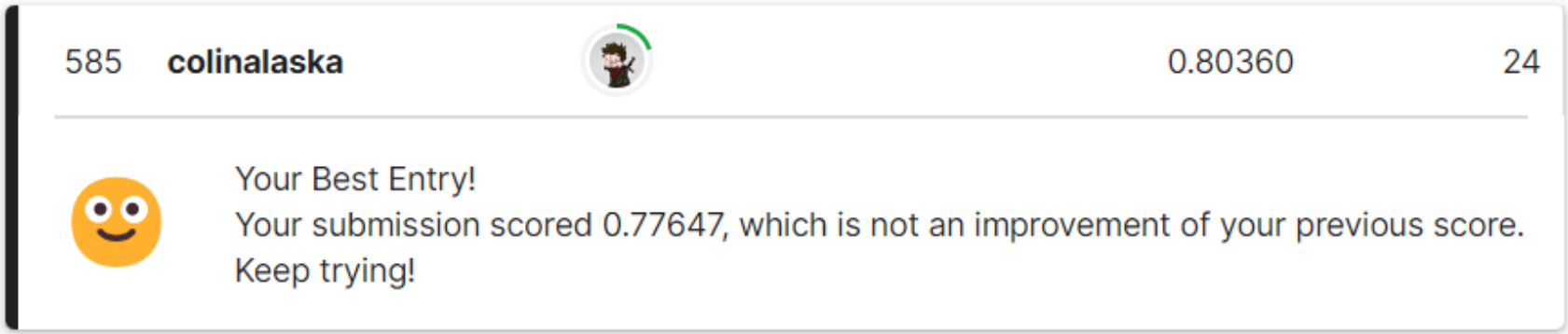
Notre travail est ainsi passé par différentes étapes.

- Travail important d'analyse et de transformation des données, puis utilisation de **28 variables prédictrices** dans l'objectif de renforcer la précision des prédictions.
- Évaluation de divers modèles de machine learning, y compris **la régression logistique**, **Elastic Net**, **SVM**, **arbres de décision**, et boosting avec **AdaBoost**.
- Les meilleurs résultats ont été obtenus par les SVM et le boosting des arbres de décision.

Les résultats sur Kaggle



Le classement final



Nous finissons donc $585^{\text{ème}}/2612$, ce qui n'est pas un si mauvais résultat en somme !