le cnam

ARTICLE

Titanic : Apprentissage Statistique - STA201

Kaouther MAKHLOUF & Wajd MESKINI*

Conservatoire National des Arts et Métiers (CNAM)
*Emails: makhlouf_kaouther@yahoo.fr-+33695009721,wajd.meskini@gmail.com-+33698267253

Abstract

L'objectif de ce travail est de tester des méthodes de régressions robustes (Régression Logistique et Régression Lasso) dans un contexte de Classement binaire avec **Python**.

- La **Régression Logistique** sera le modèle principal.
- La **Régression Lasso** sera utilisée pour prédire une variable quantitative influente dans le jeu de données, afin d'étudier l'impact de l'**imputation de valeurs manquantes avec des modèles** sur l'évaluation finale.
- Toutes les évaluations intermédiaires se feront par validation croisée avec pour objectif de maximiser le **score F1**. L'évaluation finale se fait sur un jeu d'évaluation dédié.
- Nous testerons également l'impact de l'**optimisation automatique des seuils** (à partir desquels une prédiction est 1), et **l'optimisation des hyperparamètres** des modèles.
- Le projet est sur ce lien github https://github.com/atracordis/projet-stats-cnam-titanic et est exécutable sur RStudio ou Google Colab. Il possible d'ouvrir le notebook sur github et d'appuyer sur "lancer avec Colab" pour pouvoir le tester en live.

Mots-clés: Régression Logistique, Lasso, Imputation des valeurs manquantes, Optimisation automatique des hyperparamètres, optimisation automatique des seuils de décision

Table des matières

1. Bases Théoriques

1.1 Modèles linéaires généralisés

1.1.1 Formulation générale des GLM

Un modèle linéaire généralisé est défini par (McCullagh et Nelder 1989) :

1. Une distribution de la famille exponentielle pour la cible Y.
2. Un prédicteur linéaire $\eta = X\beta$.
3. Une fonction de lien g telle que

$$\mathbb{E}[Y \mid X] = \mu = g^{-1}(\eta), \quad \eta = X\beta. \tag{1}$$

4. Une fonction de variance $\mathrm{Var}[Y \mid X] = V(\mu)$.

Les paramètres β s'estiment par maximum de vraisemblance via l'algorithme itératif des moindres carrés re-pesés (IRLS) :

$$\beta^{(t+1)} = \beta^{(t)} - \left[\nabla^2 \ell(\beta^{(t)})\right]^{-1} \nabla \ell(\beta^{(t)}), \tag{2}$$

où $\ell(\beta)$ est la log-vraisemblance.

1.1.2 Classement avec Régression logistique

La régression logistique est un cas particulier du GLM avec une fonction de lien logit pour classifer une variable binaire $Y \in \{0, 1\}$ (Cox 1958) :

$$\text{logit}(p_i) = \ln \frac{p_i}{1 - p_i} = \beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}, \quad p_i = P(Y_i = 1 \mid X_i). \tag{3}$$

La log-vraisemblance négative à minimiser s'écrit :

$$L(\beta) = -\sum_{i=1}^{n} \Big[y_i \ln p_i + (1 - y_i) \ln(1 - p_i) \Big]. \tag{4}$$

Ce qui donne le problème de minimisation suivant (Cox 1958), on minimise la log-vraisemblance négative :

$$\min_{(\beta_0, \beta)} -\sum_{i=1}^{n} \left[y_i \ln p_i + (1 - y_i) \ln(1 - p_i) \right], \quad \text{où } p_i = \frac{1}{1 + \exp(-\beta_0 - x_i^T \beta)}. \tag{5}$$

Hypothèses requises

— Relation linéaire entre les logs des probas (variable cible) et les variables explicatives
— Absence de multicolinéarité forte
— Indépendance des observations et erreurs non corrélées entre elles
— Pas besoin d'homoscedasticité ni de normalité des résidus

1.1.3 Solveurs en régression logistique - LBFGS

Le solveur LBFGS (Limited-memory BFGS) est un algorithme quasi-Newton à mémoire limitée (Liu et Nocedal 1989).

Méthode Newton classique Méthode d'optimisation utilisée en descente du gradient, capable d'atteindre un minimum en peu d'itérations, mais nécessitant un calcul complexe en gardant toute la donnée en mémoire.

Caractéristiques

— Convergence très lente. Pas parallélisable (contrairement à la descente du gradient avec des solveurs plus classiques)
— Nécessité de mettre toute la donnée en mémoire, ce qui peut être un frein pour de gros jeux de données

— Sensible au point initial de la descente
— Parfois en descente du gradient, on peut être amené à faire plusieurs pas de convergence avec la descente du gradient classique, suivi d'une ou plsieurs étapes de convergence Newton, pour atteindre la convergence plus vite (Methode One-Step) (Meskini, Brouste et Dugué 2024).

Approximation L-BFGS

— L-BFGS construit une approximation de Newton (Liu et Nocedal 1989).

Caractéristiques

— Convergence plus rapide que Newton
— Coût en mémoire contrôlé
— Supporte la régularisation L2 (Ridge) ou pas de pénalité.
— Bon compromis vitesse / utilisation mémoire pour problèmes de taille modérée.

1.1.4 Solveurs en régression logistique - liblinear

Le solveur Liblinear utilise la descente en coordonnées (décomposition d'un problème d'optimisation à plusieurs variables en une suite de sous-problèmes à une seule variable) pour minimiser l'erreur de la régression (Fan et al. 2008). Cette méthode est efficace lorsque le nombre d'échantillons n et de caractéristiques p est modéré, et elle supporte les pénalités Ridge et Lasso.

Caractéristiques

— Lorsque ni le nombre d'exemples n ni le nombre de variables p n'est excessivement grand ; la descente en coordonnées converge très rapidement.
— Support des régularisations L1 et L2, contrairement aux solveurs basés sur LBFGS qui ne gèrent que les pénalités lisses (L2).
— Faible empreinte mémoire, rapide d'entraînement, idéal pour le prototypage

1.2 Régularisation L1 (Lasso)

1.2.1 Formulation

En régression linéaire, le Lasso résout (Tibshirani 1996) :

$$\min_{\beta} \ \frac{1}{2}\sum_{i=1}^{n}(y_i - x_i^T\beta)^2 \ + \ \lambda\sum_{j=1}^{p}|\beta_j|. \tag{6}$$

Cet ajout de la somme des $\lambda|\beta|_j$ force certains β_j à zéro, puisque le but est de minimiser la fonction de coût. Ceci produit un modèle *sparse*, donc avec beaucoup de paramètres à zéro (Tibshirani 1996). L'intensité de cette mise à zéro dépend de λ, qui est représenté par $C = 1/\lambda$ en Régression Logistique. C est l'un des hyperparamètres à optimiser, où sa diminution augmente la force de la régularisation, et son augmentation la diminue.

En régression logistique pénalisée (Logistic Lasso) (Tibshirani 1996) :

$$\min_{\beta} \ -\sum_{i=1}^{n}\left[y_i \ln p_i + (1-y_i)\ln(1-p_i)\right] \ + \ \lambda\|\beta\|_1. \tag{7}$$

1.2.2 Caractéristiques

— facilite la sélection automatique de variables : L'annulation de certains coefficients à faible impact fait que les coefficients restants sont faciles à interpréter
— Modèle linéaire : Ne capture pas les relations non-linéaires sans transformations
— En éliminant les variables, la régularisation a tendance à en éliminer les plus redondantes, ce qui stabilise le modèle et le rend plus robuste à la multicolinéarité comparé à la régression linéaire sans régularisation

1.3 Imputation des valeurs manquantes

1.3.1 Méthodes naïves

La méthode la plus simple pour remplacer les valeurs manquantes consiste à les remplacer par la moyenne ou la médiane de la façon suivante (Little et Rubin 1987) :

$$x_i \leftarrow \begin{cases} \text{mean}(X_{\cdot j}) & \text{ou médiane}(X_{\cdot j}) \quad \text{si } x_{ij} = \text{NA}, \\ x_{ij} & \text{sinon.} \end{cases} \tag{8}$$

Cependant, la moyenne est sensible aux valeurs extrêmes. De plus, la médiane ne tient pas compte des corrélations multivariées.

1.3.2 Imputation supervisée par Régression Linéaire Lasso

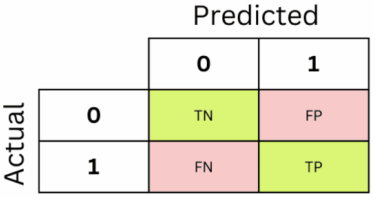
Pour tenir compte des relations multivariées et potentiellement produire de meilleures estimations si la variable explique bien la variable cible, nous pouvons entraîner manuellement une régression linéaire Lasso sur les parties du dataset où la valeur existe, pour ensuite prédire les parties où elle est manquante (Buuren et Groothuis-Oudshoorn 2011).

1.4 Optimisation et Évaluation des modèles

1.4.1 Métriques de performance

Définissons les quantités :

— TP : "*True Positives*", vrais positifs, où valeur prédite = vraie valeur = 1
— TN : "*True Negatives*", vrais négatifs, où valeur prédite = vraie valeur = 0
— FP : "*False Positives*", faux positifs, où valeur prédite = 1, mais vraie valeur = 0
— FN : "*False Negatives*", faux négatifs, où valeur prédite = 0, mais vraie valeur = 1



$$\text{Justesse (accuracy)} = \frac{TP + TN}{TP + TN + FP + FN}, \tag{9}$$

$$\text{Rappel (recall)} = \frac{TP}{TP + FN}, \tag{10}$$

$$\text{Précision (precision)} = \frac{TP}{TP + FP}, \tag{11}$$

$$\text{F1-score} = 2\,\frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \tag{12}$$

Sur un jeu déséquilibré, l'accuracy est trompeuse (un modèle prédisant toujours la classe majoritaire atteint une accuracy élevée sans détecter la minoritaire), on privilégiera le F1-Score, une combinaison de la précision et du rappel (Powers 2011).

1.4.2 Data Leakage

La fuite de données (data leakage) en apprentissage automatique se produit lorsqu'un modèle incorpore, lors de son entraînement, des informations qui ne seraient pas disponibles au moment de la prédiction réelle. Ce phénomène conduit à des estimations de performance optimistes et non généralisables. On distingue principalement deux grandes catégories : la Divulgation de la cible, et la contamination des exemples.

Divulgation de la cible Proxy de la cible : inclure un attribut qui explique la cible, qui ne serait pas disponible en situation réelle. Par exemple prédire le taux de survie en prenant en compte si le corps a été retrouvé. (duplicata déguisé de la cible)

contamination des exemples Prétraitement avant séparation : Par exempleappliquer une normalisation (standardisation, ou min-max) sur l'intégralité du dataset, puis découper en train/test, entraîne un partage d'informations entre les deux.

1.4.3 Normalisation

La **normalisation** (ou standardisation) est une étape cruciale dans le prétraitement des données en régression si le modèle est sensible aux variations d'échelle (ce qui est le cas pour nos modèles). Elle vise à mettre toutes les variables sur une même échelle afin de garantir que chacune contribue équitablement au modèle. En son absence, les variables avec une grosse échelle auront un impact disproportionné sur l'erreur.

Standardisation (Z-score) La standardisation transforme les données pour qu'elles aient une moyenne nulle et un écart-type égal à un. La formule est la suivante :

$$x_i' = \frac{x_i - \bar{x}}{\sigma_x},$$

où :

— $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ est la moyenne,

— $\sigma_x = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}$ est l'écart-type.

Cette méthode est utile pour satisfaire certaines hypothèses de certaines régressions (qui s'attendent à des lois normales).

Normalisation Min–Max

Cette normalisation transforme les valeurs pour les placer dans l'intervalle $[0, 1]$:

$$x'_i = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}},$$

où $x_{\min}$ et $x_{\max}$ sont respectivement les minimum et maximum de la variable.

Cette méthode est utilisée si on l'on souhaite respecter des bornes strictes. Puisque nous avons des dummies (indicatrices 0 ou 1), cette méthode permettra de normaliser nos variables quantitatives pour qu'elles soient dans la même échelle que les qualitatives.

Pourquoi la normalisation est-elle importante ?

— Les algorithmes de descente de gradient (ex. Gradient Descent, LBFGS) convergent plus rapidement lorsque la surface de la fonction de coût est bien conditionnée (sphérique). Des variables sur des échelles très différentes génèrent des vallées étroites qui ralentissent l'optimisation.
— Dans les modèles régularisés comme la régression Lasso (L1) ou Ridge (L2), une normalisation correcte permet à la pénalisation de s'appliquer équitablement à tous les coefficients. Sans normalisation, les variables avec des variances élevées auront des coefficients artificiellement réduits pour compenser leur plus grande amplitude, ce qui biaise la sélection de variables.
— La normalisation assure que chaque variable contribue de façon équitable au coût, évitant qu'une variable domine simplement en raison de son ordre de grandeur.

1.4.4 Validation croisée

Le k-fold stratifié partitionne en k plis équilibrés en classes, valide k fois et moyenne les scores pour réduire biais et variance (Kohavi 1995).

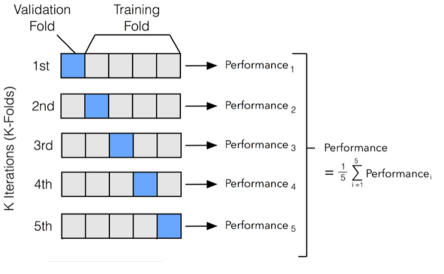


Figure 1. Validation croisée

1.4.5 Grid Search

`Grid Search` réalise une recherche sur une grille d'hyperparamètres, évaluant chaque combinaison par validation croisée (Kuhn et Johnson 2013).

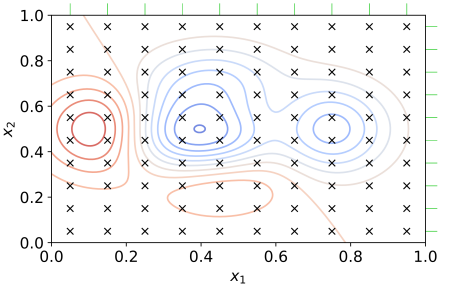


FIGURE 2. Exemple de Grid Search superposé avec l'erreur en fonction de deux hyperparamètres

1.4.6 Optimisation de seuil

L'Optimisation des seuils ajuste le seuil de décision des probabilités issues du modèle pour maximiser une métrique (le F1-score) via validation croisée interne (Fawcett 2006).

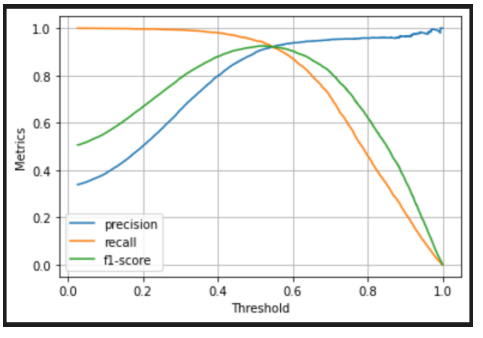


FIGURE 3. Exemple d'identification du meilleur seuil avec les courbes précision, rappel et F1-Score

2. Présentation de la base de données

Le jeu de données, appelé "Titanic", renferme les caractéristiques des passagers du Titanic, avec une variable cible à prédire nommée "Survived", l'état de survie (0 ou 1) du passager. Il contient les informations de 1309 passagers (OpenML 2014).

Au vu du descriptif , nous devons supprimer les variables suivantes :

— **boat** car elle correspond à une information obtenue après le naufrage (donc inutile d'un point de vue prédictif)
— **body** car elle décrit le numéro d'identification du cadavre si applicable - pour les mêmes raisons citées ci-dessus
— **ticket**, le numéro du billet, qui ne devrait pas comporter d'information pertinente
— **home.dest**, qui ramène trop de complexité de traitement dans un premier temps

Table 1. Description des colonnes du jeu de données

Colonne	Description
name	Nom du passager
gender	Sexe du passager (male ou female)
age	Âge du passager au moment du décès. Peut contenir des valeurs manquantes.
survived	Statut de survie du passager (0 : Perdu, 1 : Sauvé)
pclass	Classe du passager (1re/2e/3e)
cabin	Cabine du passager
embarked	Port d'embarquement (C : Cherbourg, Q : Queenstown, S : Southampton, ? : Inconnu). Peut contenir des valeurs manquantes.
fare	Tarif du billet (livre sterling). Peut contenir des valeurs manquantes.
home.dest	Lieu de résidence et destination. Peut être séparé par des virgules (','), des slash ('/'), des espaces (' '), ou n'avoir que le lieu de résidence ou la destination. Peut être le nom d'une ville, d'un état des États-Unis ou d'un pays.
ticket	Numéro du billet, peut contenir une lettre
parch	Nombre de parents/enfants à bord
sibsp	Nombre de frères/sœurs/époux.ses à bord
boat	Canot de sauvetage utilisé par le passager
body	Numéro d'identification du corps

3. Analyse descriptive et pré-traitements

3.1 Création de jeu d'évaluation

Nous allons séparer le jeu de données que nous avons reçu en deux jeux, *train* (entraînement) et *eval* (évaluation) - sachant que le jeu de train va être séparé par la suite en train et test par validation croisée. Nous ferons nos analyses, visualisations et modélisations sur le jeu de train. Nous utiliserons le jeu eval pour évaluer nos modèles et nos approches.

La variable cible est légèrement déséquilibrée **(38% de taux de survie)**, donc il faut séparer les jeux de données de façon stratifiée (donc de façon à avoir 38% de taux de survie dans chaque segment).

3.2 Valeurs manquantes

- **age** a 20% de valeurs manquantes.
- **cabin** a 77% de valeurs manquantes, mais nous pouvons interpréter l'absence de valeur comme une absence de cabine, ce qui correspond à un indicateur de catégorie socio-professionnelle supplémentaire
- Très peu de valeurs manquantes dans d'autres colonnes (**fare** et **embarked**), qu'on pourra gérer par médiane et mode respectivement

3.3 Identification des variables Qualitatives et Quantitatives

- Nous distinguons les variables quantitatives, qualitatives et qualitatives à forte cardinalité. Nous retiendrons les variables **age** et **fare** comme quantitatives.
- Au vu du faible nombre de modalités, nous allons interpréter les variables quantitatives suivantes comme qualitatives : **pclass, sibsp, parch**
- Au vu du très fort nombre de modalités, nous allons traiter les variables qualitatives suivantes différemment :
  - **name** : Nous allons extraire les titres à partir des noms avec des expressions régulières **(Mr.,Mrs.,Miss., Ms...)**. Notre attente est que certains titres plus

fortunés ont plus de chances de survivre que d'autres. Puisqu'il y a plusieurs titres, nous pourrons les regrouper selon leur effet moyen sur la cible.

— **cabin** : Nous allons uniquement voir la présence (not na) ou l'absence (na) de cabine. Notre attente est que la présence d'une cabine améliore les chances de survie.

3.4 Visualisation et traitement des variables Qualitatives

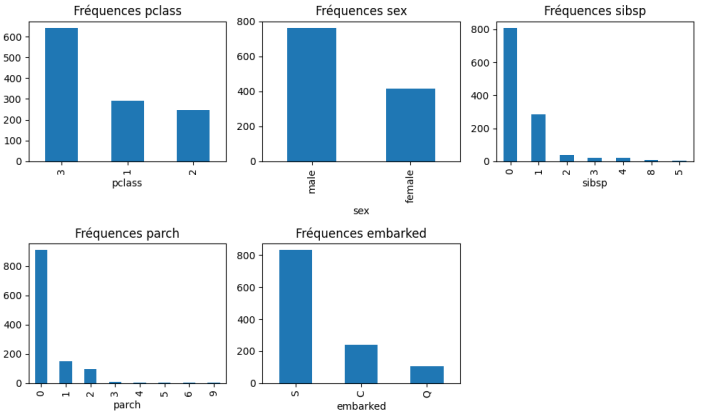


FIGURE 4. Analyse univariée des variables qualitatives

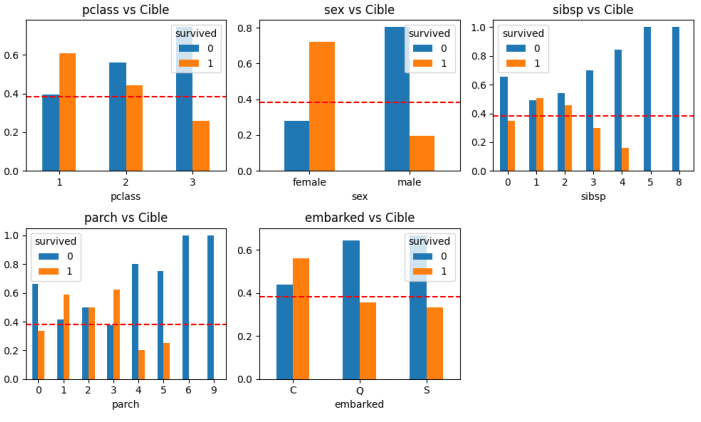


FIGURE 5. Analyse bivariée des variables qualitatives vs la cible

Ces graphiques univariés et bivariés nous donnent les conclusions suivantes :

— Les personnes aux classes supérieures **(pclass 1 et 2)** ont sensiblement plus de chances de survie que la 3ème classe (séparateur CSP)

— Les femmes ont sensiblement plus de chances de survie que les hommes (séparateur hommes/femmes)

— Avoir un ou deux frères/sœurs/époux.se donne plus de chances de survie que l'alternative
— Avoir de 1 à 3 enfants augmente ses chances de survie par rapport à l'alternative
— Ceux qui ont embarqué à Cherbourg ont plus de chances de survie que ceux qui ont embarqué aux autres ports.

3.5 Visualisation et traitement des variables Quantitatives

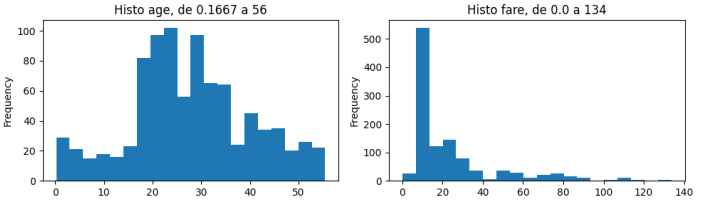


FIGURE 6. Analyse univariée des variables quantitatives

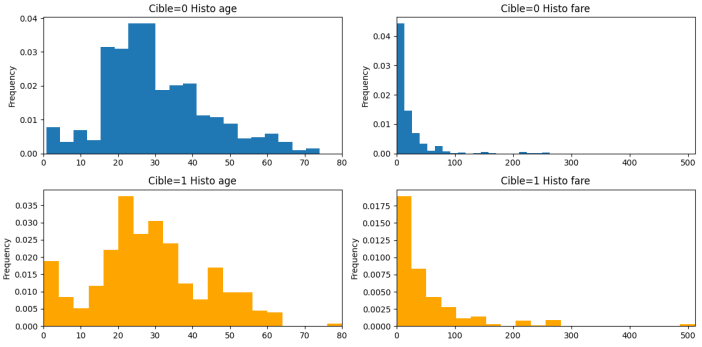


FIGURE 7. Analyse bivariée des variables quantitatives vs la cible

— Il y a une différence d'échelle entre les 2 variables. Il faudra donc normaliser
— A priori, il n'y a pas de valeurs aberrantes dans les âges
— Il y a des valeurs extrêmes dans **fare** à priori, même si elle n'est pas problématique
— A priori pas de problème avec les valeurs extrêmes d'un point de vue modélisation, tant qu'on utilise des modèles non-paramétriques, sans hypothèse sur les variables d'entrée (style arbres, ensembles). Mais ce serait un problème pour les modèles que nous utilisons et nous aurons donc besoin de normaliser
— On remarque que la variable âge a le potentiel d'être très influente, puisque la distribution change et devient décalée vers la gauche, semblant indiquer que les enfants ont eu plus de chances de survie.
— Le prix du ticket est également influent, la distribution se décale vers la droite.

3.6 Visualisation et traitement des variables qualitatives à forte cardinalité

Il convient de traiter ces variables à part car d'un côté leur pouvoir prédictif est probablement plus faible si traitées modalité par modalité (car l'information est éparpillée

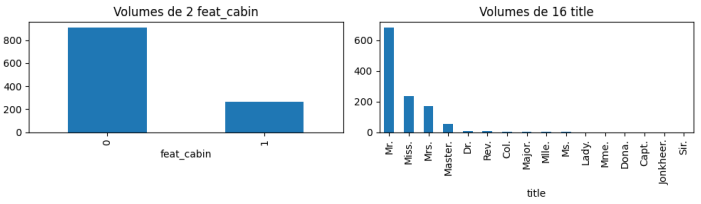


FIGURE 8. Analyse univariée des variables qualitatives à forte cardinalité

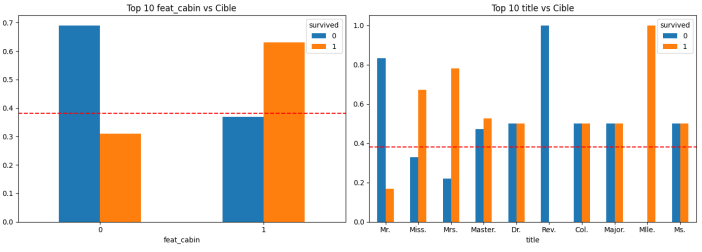


FIGURE 9. Analyse bivariée des variables qualitatives à forte cardinalité vs la cible

sur plein de modalités) et de l'autre, les modèles (en particulier ceux basés sur les arbres) auront tendance à surévaluer leur pouvoir prédictif et les utiliser plus souvent.

— Pour **cabin**, on va considérer uniquement la présence ou l'absence de cabine
— Pour **name**, on va extraire les titres avec des expressions régulières, puis regrouper les titres selon leur pouvoir prédictif

4. Démarche choisie et objectifs

L'objectif est de prédire la variable binaire "survie".

4.1 Traitement des variables qualitatives

Visualiser les variables qualitatives nous permet de prendre les décisions suivantes :

— On convertit nos variables binaires en 0 et 1, avec 1, valant la modalité qui a le plus de chance de survivre. Cela inclut **sibsp, parch, et sex**
— On convertit nos variables multicatégorielles similairement. Cela inclut **pclass et embarked**. Ca permet également de gérer les valeurs manquantes de **embarked**

4.2 Traitement des variables quantitatives

Visualiser les variables quantitatives nous permet de prendre les décisions suivantes :

— Remplacement par la médiane : Plus robuste que la moyenne, nous l'utiliserons pour la variable **fare** par défaut, et pour la variable **âge** dans l'un des deux scénarios
— Remplacement par régression linéaire : Tire partie des multicolinéarités qui existent entre toutes les variables et la variable âge, et permet de la prédire avec plus de précision. Nous l'utiliserons pour la variable **âge** dans le deuxième scénario.
— On normalise les variables quantitatives à cause de leur différences d'échelle, effectuant une transformation vers des valeurs entre 0 et 1.

4.3 Traitement des variables qualitatives à forte cardinalité

Visualiser les variables à forte cardinalité nous permet de prendre ces décisions :

— Les titres connus seront encodés selon leur effect supérieur à la moyenne ou inférieur à la moyenne sur la variable cible, et regroupés dans ces deux catégories
— Si un titre n'existe pas dans le train, il ne générera pas une nouvelle colonne.
— Pour éviter le data leakage, nous allons identifier ces classes dites "majoritaires" et "minoritaires" sur le train, mais les appliquer sur le test

4.4 Analyse Multivariée - Corrélations des variables transformées

Après traitement des variables, toutes les variables restantes correspondent soit à des indicatrices (0 ou 1) soit à des variables quantitatives transformées pour être entre 0 et 1.

Ceci nous permet de faire une analyse multivariée en utilisant les corrélations de Spearman. Celle-ci mesure le degré d'association monotone entre deux variables en utilisant leurs rangs plutôt que leurs valeurs. Elle ne suppose ni linéarité, ni normalité des données, et est robuste aux valeurs extrêmes - contrairement à la corrélation de Pearson.

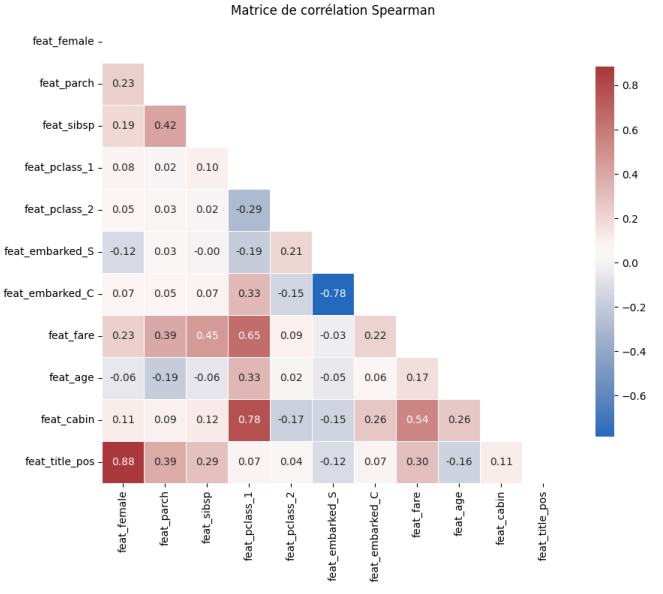


Figure 10. Analyse des corrélations des variables transformées

On peut donc avoir les interprétations suivantes :

— **feat_title_pos** correspond aux titres ayant un effet positif sur la survie, principalement des titres féminins, ce qui génère la corrélation avec **feat_female**
— **feat_cabin** est fortement corrélé avec **feat_pclass_1** - indicateurs socio-économiques
— Ceux qui ont embarqué au port S n'ont pas embarqué au port C
— Il y a des corrélations intéressantes entre **feat_fare** et **feat_sibsp**, indiquant peut-être des tickets groupés (un prix élevé mais pour toute la famille)
— Grâce au Lasso, la multicolinéarité faible ne nous inquiète pas, le modèle devrait mettre à zéro les variables qui ne lui ramènent pas d'info.

5. Modélisation et résultats

5.1 Modèle plancher

Nous avons commencé par construire un modèle dit "plancher", c'est à dire un modèle avec les paramètres par défaut (donc régression logistique avec optimiseur liblinear sans régularisation). Nous utilisons ce modèle pour avoir une performance "plancher" pour notre modèle, à laquelle les techniques suivantes vont se mesurer.

Pour l'évaluer, nous utilisons la validation croisée stratifiée à cause du déséquilibre dans notre cible. De plus, pour éviter le data leakage, nous avons construit une pipeline de pré-traitement qui s'assure de ne pas utiliser de données du jeu de test pour transformer le jeu de train (par exemple, en prenant le maximum du train et pas du test). Pour chaque sous-ensemble de validation croisée, la pipeline s'exécute sur le sous-ensemble train puis s'applique sur le test - sur lequel le modèle s'évalue. Après validation croisée, les modèles sont évalués sur le jeu d'évaluation avec la matrice de confusion et la courbe ROC.

L'entraînement du modèle plancher nous permet d'avoir ce résultat initial, contre lequel les autres approches vont être comparées par la suite.

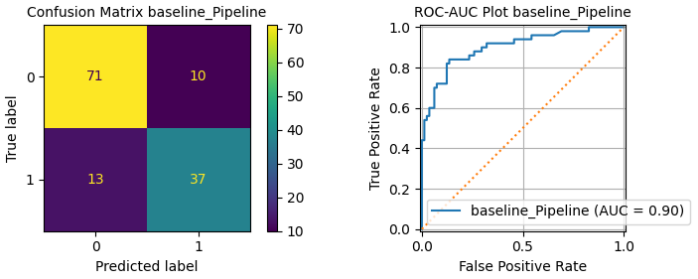


Figure 11. Evaluation du modèle plancher - Matrice de confusion et courbe ROC

5.2 Modèle plancher avec imputation Lasso des âges manquants

L'approche suivante consiste à voir l'impact de l'imputation Lasso des âges manquants par rapport à l'imputation par la médiane. La pipeline de pré-traitement est donc modifiée pour imputer les âges manquants sur le train et le test en entraînant un modèle de régression Lasso sur les variables explicatives le train et en l'appliquant sur le test.

On voit une légère amélioration par rapport au modèle plancher.

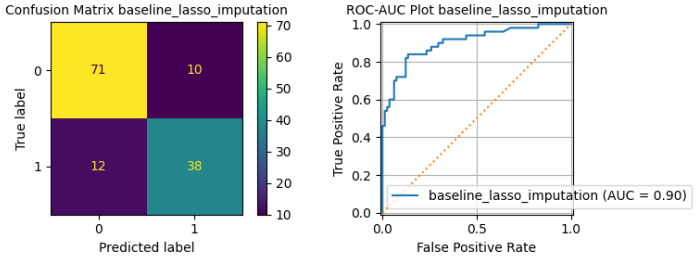


Figure 12. Evaluation du modèle plancher avec imputation Lasso - Matrice de confusion et courbe ROC

5.3 Optimisation automatique des hyperparamètres

L'approche suivante vise à tester toutes les combinaisons des hyperparamètres possibles. Dans notre cas, nous avons testé les hyperparamètres suivants :

— penalty : présence ou absence de L1 (Lasso)
— solver : Optimiseur utilisé (liblinear ou LBFGS)
— C : Intensité de régularisation ($1/\lambda$), un C faible force plus de paramètres à zéro
— class_weight : Rééquilibre (ou pas) les classes à 50/50 en dupliquant la classe minoritaire

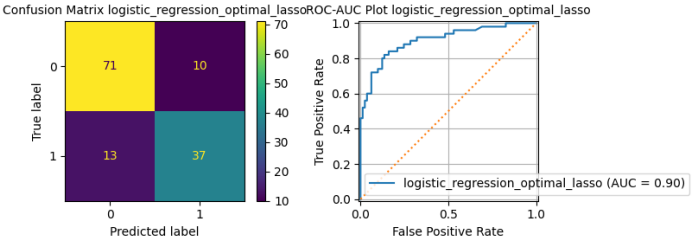


FIGURE 13. Evaluation de l'optimisation automatique des hyperparamètres

La diminution (légère) des performances par rapport au modèle plancher avec Lasso montre que cette méthode seule n'augmente pas les performances.

5.4 Optimisation des seuils

Nous couplons l'optimisation des hyperparamètres avec l'optimisation automatique des seuils pour maximiser le F1-Score. La méthode prend les modèles précédemment entraînés, puis teste tous les seuils entre 0 et 1 (avec un pas de 0.005) pour voir quel seuil de décision maximise le F1-Score. Ceci a pour effet d'augmenter légèrement le F1-Score.

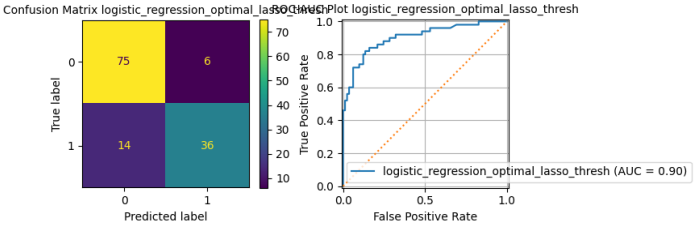


FIGURE 14. Evaluation de l'optimisation automatique du seuil de décision

6. Résultats Finaux

6.1 Tableau de résultats comparatif

En tête du classement, Nous avons le modèle dit **logistic_regression_optimal_lasso_thresh**, pour "régression logistique à hyperparamètres optimaux avec imputation lasso de l'âge et optimisation automatique du seuil", suivi du même modèle sans imputation de l'âge, ce qui démontre malgré tout la pertinence des différentes techniques appliquées pour gagner jusqu'à 1% supplémentaire de performance.

Table 2. Tableau des résultats finaux

Modèle	Accuracy	F1-score	AUC	Précision	Rappel
logistic_regression_optimal_lasso_thresh	85%	78%	90%	86%	72%
logistic_regression_optimal_thresh	85%	78%	90%	86%	72%
baseline_lasso_imputation	83%	78%	90%	79%	76%
baseline	82%	76%	90%	79%	74%
logistic_regression_optimal_lasso	82%	76%	90%	79%	74%
logistic_regression_optimal	82%	76%	90%	79%	74%
baseline_thresh	82%	76%	90%	80%	72%
baseline_lasso_imputation_thresh	82%	76%	90%	80%	72%

6.2 Importance des variables dans le meilleur modèle

Le meilleur modèle est un modèle de régression logistique avec régularisation Lasso, avec imputation de l'âge avec régression lasso, suivie d'une optimisation automatique des hyperparamètres et d'une optimisation du seuil de décision. Visualiser ses coefficients nous permet de voir quelles classes le modèle a choisi de rapprocher de 0.

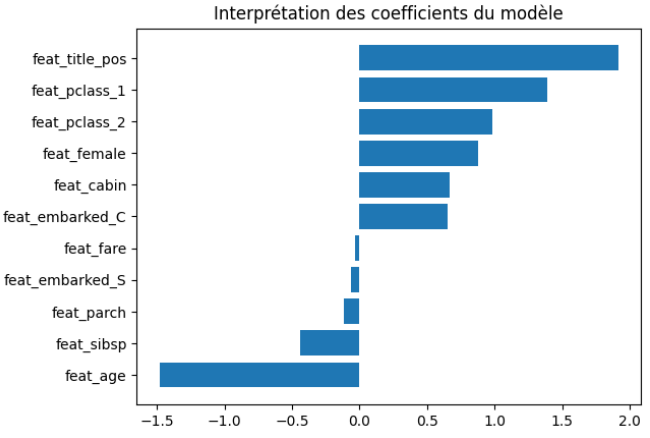


Figure 15. Coefficients du modèle final

Sans surprise, il s'agit de variables corrélées avec les variables dont les coefficients sont grands, par exemple **feat_embarked_S** vs **feat_embarked_C**, et **feat_fare** vs **feat_pclass_1**. Il est intéressant de noter que **feat_title_pos** et **feat_female**, fortement corrélées, ont toutes les deux des coefficients élevés - augmentant la probabilité de survie.

7. Conclusion

Le projet montre l'efficacité initiale de la régression logistique pour le classement binaire, et sa robustesse face à des variables qualitatives et quantitatives avec des lois différentes après un pré-traitement personnalisé. Il démontre également l'efficacité de la méthode Lasso associée à la régression logistique face à un colinéarité forte.

Il nous permet aussi d'explorer différentes approches pour améliorer les performances d'un modèle : l'imputation de valeurs manquantes avec une régression Lasso, l'optimisation automatique des hyperparamètres et des seuils de décision, ainsi que leur impact sur la performance.

Références

Buuren, Stef van et Karin Groothuis-Oudshoorn. 2011. mice: Multivariate Imputation by Chained Equations in R. *Journal of Statistical Software* 45 (3). http://www.jstatsoft.org/v45/i03.

Cox, David R. 1958. The Regression Analysis of Binary Sequences (with discussion). *Journal of the Royal Statistical Society: Series B (Methodological)* 20 (2) : 215-242. https://doi.org/10.1111/j.2517-6161.1958.tb00292.x.

Fan, Rong-En, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang et Chih-Jen Lin. 2008. LIBLINEAR: A Library for Large Linear Classification. In *Journal of Machine Learning Research,* 9 : 1871-1874.

Fawcett, Tom. 2006. An Introduction to ROC Analysis. *Pattern Recognition Letters* 27 (8) : 861-874. https://doi.org/10.1016/j.patrec.2005.10.010.

Kohavi, Ron. 1995. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI),* 1137-1145. Morgan Kaufmann.

Kuhn, Max et Kjell Johnson. 2013. Applied Predictive Modeling. Chap. 4. New York : Springer. ISBN : 978-1-4614-6848-6. https://doi.org/10.1007/978-1-4614-6849-3.

Little, Roderick J. A. et Donald B. Rubin. 1987. *Statistical Analysis with Missing Data.* 1st. Chap. 4. See Chapter 4 ("Single-Imputation Methods": mean imputation, regression imputation, hot-deck, etc.) New York : Wiley. ISBN : 0-471-80254-9.

Liu, Dong C. et Jorge Nocedal. 1989. On the Limited Memory BFGS Method for Large Scale Optimization. *Mathematical Programming* 45, n^{os} 1–3 (août) : 503-528. https://doi.org/10.1007/BF01589116. https://doi.org/10.1007/BF01589116.

McCullagh, Peter et John A. Nelder. 1989. *Generalized Linear Models.* 1st. London : Chapman & Hall. ISBN : 978-0412317606.

Meskini, Wajd, Alexandre Brouste et Nicolas Dugué. 2024. Speeding up the Training of Neural Networks with the One-Step Procedure. *Neural Processing Letters* 56 (1) : 178. https://doi.org/10.1007/s11063-024-11637-6. https://link.springer.com/article/10.1007/s11063-024-11637-6.

OpenML. 2014. *Titanic (OpenML dataset).* https://www.openml.org/d/40945. Accessed: 2025-05-07.

Powers, David M. W. 2011. Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation. *Journal of Machine Learning Technologies* 2 (1) : 37-63.

Tibshirani, Robert. 1996. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society: Series B (Methodological)* 58 (1) : 267-288. https://doi.org/10.1111/j.2517-6161.1996.tb02080.x.