REDS - Projet Boson de Higgs

Rapport

Kim-Anh Laura Nguyen
Keyvan Beroukhim
Master 2 DAC
Promo 2019-2020

Enseignant : Olivier Schwander

1 Introduction

Nous souhaitons détecter la présence du boson de Higgs dans des données simulées dans le but de reproduire le comportement de l'expérience ATLAS. Il s'agit d'un problème de détection d'évènement, ou classification binaire, dans lequel les deux classes sont :

— *background*

— *tau tau decay of a Higgs boson*

Les données sont issues du projet Kaggle *ATLAS Higgs Boson Machine Learning Challenge 2014*.

2 Analyse préliminaire des données

La base de données est constituée de 818238 évènements simulés. Chaque évènement est défini par 30 attributs numériques et un label à prédire.

Les données sont composées à **34% de labels positifs** (les *signaux*) et à 66% de labels négatifs (le *background*). Les classes ne sont pas suffisamment déséquilibrées pour gêner l'entraînement. Par ailleurs, la métrique propre à cette tâche, nommée *AMS*, nous est imposée. Le choix habituel de la mesure d'évaluation à utiliser pour prendre en compte ce déséquilibre ne se pose donc pas.

2.1 Distribution des variables

La figure 1 contient les histogrammes de répartition des valeurs de quelques variables, en omettant les valeurs manquantes. Nous constatons que **toutes les variables ne suivent pas le même type de distribution**. Or, les algorithmes d'apprentissage se comportent généralement mieux avec une distribution des données équilibrée. Dans ce dataset, de nombreuses variables ont une distribution exponentielle décroissante (e.g `DER_MASS_VIS` sur la figure 1).

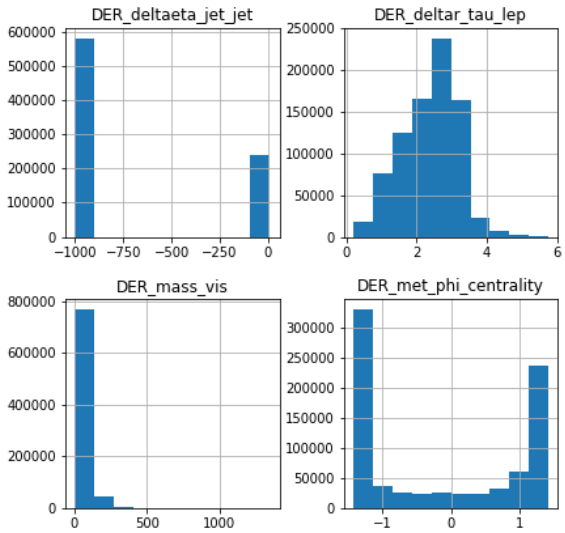
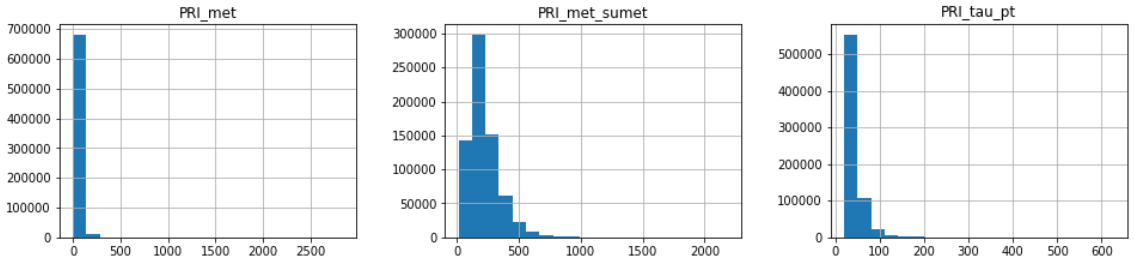
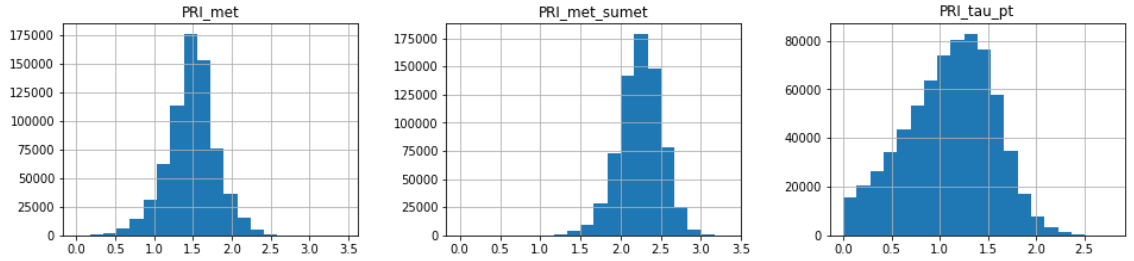


FIGURE 1 – Histogrammes de répartition des valeurs de quelques variables

Afin d'obtenir des **distributions normales** et de **stabiliser la variance**, nous effectuons une **log-transformation** de manière indépendante sur les colonnes. Nous commençons par ajouter une constante à chaque colonne à transformer pour que la valeur minimale de sur cette colonne soit 1 (comme le log ne prend que des valeurs strictement positives). Nous appliquons ensuite la fonction log sur les colonnes. Avec ce dataset et en utilisant cette méthode, nous obtenons toujours des distribution normales. Finalement nous **centrons et réduisons** chaque variable de manière indépendante car cela facilite généralement l'apprentissage des modèles.



(a) Avant log-transformation



(b) Après log-transformation

FIGURE 2 – Histogrammes de répartition des valeurs de quelques variables avant et après log-transformation

2.2 Valeurs manquantes

Le dataset contient **21% de valeurs manquantes** (indiquées par la valeur -999). D'une manière générale, les modèles d'apprentissage statistique ne gèrent pas automatiquement les valeurs manquantes. Il faut donc traiter ces données avant de les présenter à nos modèles. Nous considérons trois façons de pallier à ce problème.

2.2.1 Omission

En supprimant les évènements dont au moins un attribut n'est pas valide, nous retrouvons avec seulement 223574 exemples d'apprentissage : **restreindre le dataset aux données complètes fait perdre 75% des évènements**. Par ailleurs, la base restreinte contient 47% de labels positifs, soit 13% de plus que le dataset original. Cela met en évidence le fait que les données ne sont pas manquantes de manière indépendante : l'absence des données est liée à leur valeur (nous sommes dans un contexte *Missing Not At Random*). Un modèle entraîné sur le dataset restreint en faisant l'hypothèse "i.i.d." serait donc biaisé.

2.2.2 Conservation

Les ***Random Forest*** font partie des algorithmes permettant de **gérer les données manquantes**. En effet, les arbres de décision peuvent **reconnaître une donnée manquante par un simple test de valeur**.

2.2.3 Affectation (*Imputing*)

Une autre méthode consiste à **compléter les données manquantes** de la manière la plus "intelligente" possible. Une fois les données complétées, cela permet d'**utiliser tous les modèles de classification à notre disposition**.

Nous nous servons de l'algorithme `IterativeImputing` de scikit-learn, qui complète de manière itérative les données manquantes en utilisant des modèles de régression prédisant la valeur d'une variable à partir de la valeur des autres variables.

En pratique, nous appliquons d'abord les **log-transformations** puis nous réalisons l'**imputing** avant de finir par **scaler** les données. Ce choix se justifie premièrement par l'intuition que l'imputing fonctionnera mieux sur des données log-transformées et, deuxièmement, par le fait qu'estimer la moyenne et la variance en ignorant les données manquantes serait biaisé.

2.3 Sélection de features

Les données sont représentées par **30 variables explicatives**. Ce nombre n'étant **pas très élevé** (par exemple comparé au nombre de pixels dans une image), il n'y a donc **pas de grand risque de sur-apprentissage** pour les modèles. Ainsi, la sélection de caractéristique ne semble pas nécessaire. Cependant, comme le montre la figure 3, certaines variables sont **fortement corrélées** (e.g. `DER_mass_MMC` et `DER_mass_transverse_met_lep`), on comparera donc les performances avec ou sans l'utilisation d'une ***Analyse en Composantes Principales*** **(PCA)** afin de réduire le nombre de données explicatives.

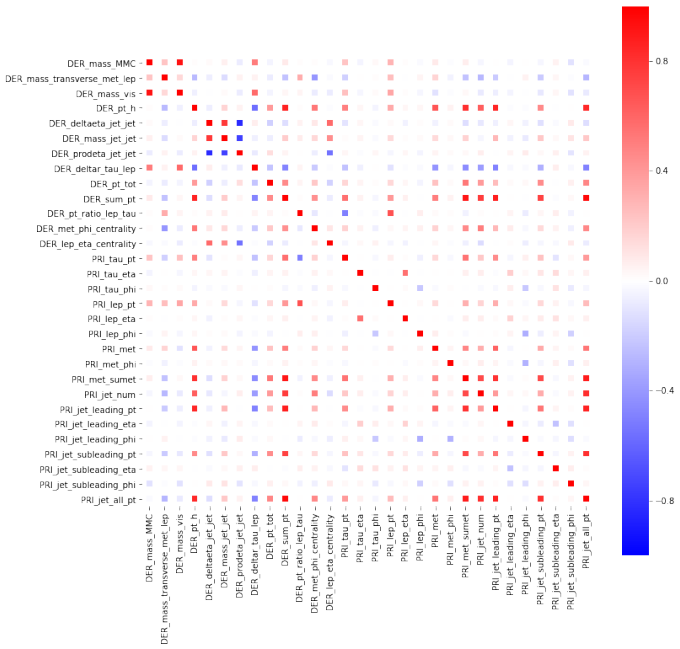


FIGURE 3 – Matrice de corrélation entre chaque attribut

3 Classification des évènements

Nous considérons plusieurs modèles d'apprentissage pour classifier les évènements :

— SVM

— méthodes ensemblistes : Bagging de Perceptron, Random Forest, AdaBoost

Par souci de temps de calcul, les expériences sont menées sur le dataset restreint à 10000 évènements. 70% de cette base est dédiée à l'entraînement et la validation des modèles, 30% au test.

Pour chaque algorithme, nous procédons par *grid search* pour trouver les hyperparamètres optimaux. Pour chaque combinaison d'hyperparamètres, le score de prédiction est estimé par *cross-validation* sur l'ensemble d'apprentissage. Nous récupérons le classifieur ayant produit le meilleur résultat et l'évaluons sur les données de test. La seule métrique utilisée est l'AMS.