Classification of Higgs Boson Decay Events

Kieran Morris, Cecina Babich Morrow, and Daniella Montgomery

January 19, 2024

Abstract

In 2012, the Large Hadron Collider at CERN discovered the existence of the Higgs Boson, a new particle which gives mass to all other elementary particles [1, 2]. In light of the fundamental nature of the Higgs Boson, it is evident that we should care about detecting them when new ones appear. In this project we attempt to create a classifier which can reliably categorise a particle collision as either including a Higgs Boson or not.

1 Introduction

In 2012, the Large Hadron Collider at CERN confirmed the existence of the Higgs Boson, a fundamental particle in the Standard Model of physics which gives mass to other elementary particles [1, 2]. Following the discovery of the Higgs boson, CERN began investigating its decay into other particles. The ATLAS experiment provided evidence of a specific decay channel in which the Higgs boson decays into two tau particles.

In order to leverage advances in the fields of statistics and machine learning to assist research on the particle, CERN launched a machine learning challenge in 2014 to improve the accuracy of identifying events associated with this specific decay pathway of the Higgs boson. The goal of the challenge was to discriminate between genuine decay occurrences and background noise events, using feature data extracted from particle collisions. Initially conducted on the Kaggle platform using a subset of the total data, the challenge culminated with CERN releasing the complete data set, opening avenues for ongoing investigation and analysis [3].

Using this data, our goal was to develop a classifier which could effectively distinguish between signal and background events. We began with feature selection by examining the distribution of predictor variables and calculating their mutual information. Then, we conducted Principle Component Analysis to assess the impact of dimensionality reduction on the dataset. Subsequently, we performed Fisher Discriminant Analysis to evaluate the linear separability of the classes.

Following these investigations of the data and their structure, we used support vector machines to classify signal vs. background events. We trained these models using a range of feature sets and hyperparameters to identify models with the best performance. Our investigation shows how differences in the distributions of the various features can be leveraged to yield more accurate classifiers.

2 Methods

2.1 Objective

Our data set [4] consists of observations from the Large Hadron Collider at CERN over the year of 2012. It is ordered data of the form $\{(\mathbf{x}_i, y_i, w_i)\}_{i\in D}$ where $\mathbf{x} \in \mathbb{R}^{30}$; $y \in \{\mathbf{b}, \mathbf{s}\}$ and $w \in [0, 1]$ is a weight which measures the intensity of each data point [3].

Each $\mathbf{x}_i$ is the collection of observables about each event and each y_i is a classification of the event as either 'Background' (= **b**) and 'Signal' = (**s**). We define

$$\mathcal{S} = \{i : y_i = \mathbf{s}\} \text{ and } \mathcal{B} = \{i : y_i = \mathbf{b}\}$$

with $n_s =\mid \mathcal{S} \mid$ and $n_b =\mid \mathcal{B} \mid$ respectively. The weight variable is not to be used to train the classifier f in any way, it is only to compute the AMS, which is our measure of the accuracy of our classifier f. Speaking of f, we define $\hat{f} = \{i : f(\mathbf{x}_i) = \mathbf{s}\}$, i.e the points labelled as signals by f and using this we define

$$s = \sum_{i \in \mathcal{S} \cap \hat{f}} w_i \text{ and } b = \sum_{i \in \mathcal{B} \cap \hat{f}} w_i$$

i.e s and b are the weighted *true positives* and *false positives* respectively. Finally the AMS function is defined as follows [3]:

$$\text{AMS} = \sqrt{2\left((s + b + b_{reg}) \ln\left(1 + \frac{s}{b + b_{reg}}\right) - s\right)}$$

Where b and s are as above and b_{reg} is a normalizing constant set to $b_{reg} = 10$ based on the advice of CERN [3]. Notice that the AMS is computed over a data set D, so if we wish to compute this over some training or test set we must re-normalize the weights.

2.2 Feature Selection

2.2.1 Missing Data

The dataset was provided with -999 in entries where no data was present. There was a logistical reason for 10/11 of the variables which contained missing data. On each detection of a collision, there could be 0,1,2 or 3 photon jets ejected from the collision. The number of jets was described in the variable jet `PRI_Jet_Num`, however 10 variables (see Hadron_WriteUp.pdf) described properties like momentum or angle of the photon jet, which clearly don't exist when there isn't a beam. The final variable which missed data was `DER_mass_MMC`, which is a little more mysterious - other attempts at the Kaggle challenge suggested it was not a result of measurement [5].

2.2.2 Co-Dependencies

Some of our variables were in fact codependent on each-other, meaning that analysis into whether some variables contributed negligible information to classification was necessary.

2.2.3 Mutual Information Tests

We employed a common technique using the mutual information to measure the impact of each variable on classification [6, 7]. Recall that the mutual information between two variables X and Y with sample space $\mathcal{X}$ is defined as

$$I(X, Y) = H(X) - H(X \mid Y)$$

where H is the entropy operator defined as $H(X) = -\sum_{x \in \mathcal{X}} p(x) \log(p(x))$ [8]. Since the entropy is a measure of uncertainty of a random variable, the mutual information describes how much uncertainty lies between the two variables. For example, $I(X, Y) = 0$ if and only if X and Y are independent, and the higher the mutual information is, the more dependent the two variables are [7]. Meaning the mutual information is a (symmetric) measure of how much information the variables share with each-other. A more useful formulation of I is the following:

$$I(X, Y) = D_{KL}(p_{XY} \mid\mid p_X p_Y)$$

where p_{XY} is the joint distribution,$p_X p_Y$ is the product of the two single distributions and D_{KL} is the KL-divergence operator [8]. In our context we can compute the mutual information between each variable V and our classification variable y. Computationally this is easy to implement as our data table contains all of p_{Vy}, p_V and p_y.

2.2.4 Principle Component Analysis

Principle component analysis (PCA) is a dimensionality reduction technique which works by projecting a dataset along its axis of greatest variance and naming this Principle Component (PC) 1. Then, the dataset is projected along an orthogonal vector along the axis of next greatest variance PC2 and so on. This is an eigenvalue decomposition of the covariance matrix. If the covariance matrix of X is denoted by C, then its eigenvalue decomposition is given by:

$$C = VDV^T, \tag{1}$$

where V is the matrix of eigenvectors and D is a diagonal matrix containing the eigenvalues.

The PCs can be obtained by selecting the top k eigenvectors corresponding to the largest eigenvalues. These eigenvectors form the matrix V_k, and the reduced-dimensional representation of X is given by:

$$X_{\text{reduced}} = XV_k. \tag{2}$$

We chose k using a scree plot, this described how much variance is explained by each PC, we can then visualise how many PCs are of significance either by setting a total percentage of variance we wish to be explained by the dataset or a threshold for the minimum variance to be explained by a PC for us to consider it. This dimensionality reduction allows us to visualise a summary our data in two or three dimensions and as such gain insight into how separable our data is when using different subsections of our data. Within this project this is used to confirm if retaining the missing data is valuable and to verify our choices for variable selection.

2.3 Fisher Discriminant Analysis

Fisher Discriminant Analysis (FDA) [9] is a method for dimensionality reduction with the goal of maximising the linear separability of a dataset. FDA finds a vector $\mathbf{w}$ such that the embedding $\mathbf{w}^\top \mathbf{x}$ maximises between-class scatterness while minimising within-class scatterness. For a given class k, the within-class scatterness is defined by

$$s_{w,k} = \sum_{i,y_i=k} (\mathbf{w}^\top \mathbf{x}_i - \hat{\mu}_k)^2$$

where $\hat{\mu}_k = \frac{1}{n_k} \sum_{i,y_i=k} \mathbf{w}^\top \mathbf{x}_i$ is the embedded center for class k. Similarly, the between-class scatterness for class k is defined by

$$s_{b,k} = n_k(\hat{\mu}_k - \hat{\mu})^2$$

where $\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{w}^\top \mathbf{x}_i$ is the overall embedded dataset center. FDA yields the vector

$$\mathbf{w} = \arg\max_{\mathbf{w}} \frac{\sum_k s_{b,k}}{\sum_k s_{w,k}}$$

.

In our case of binary classification

$$\mathbf{w} = \mathbf{S}_{\mathbf{w}}^{-1}(\mu_{\text{signal}} - \mu_{\text{background}})$$
$$\mathbf{S}_{\mathbf{w}} := n_s \mathbf{S}_s + n_b \mathbf{S}_b$$

where $\mathbf{S}_s$ and $\mathbf{S}_b$ are the sample covariances of the signal and background events, respectively [10].

We performed FDA to investigate the level of linear separability of our dataset using four sets of predictor variables: 1) Including all variables; 2) Removing the four variables that represent algebraic co-dependencies and any uniformly distributed variables; 3) Removing the 10 variables with the lowest mutual information; 4) Using only the 10 variables with the highest mutual information.

2.4 Implementing Support Vector Machines

We used support vector machine (SVM) classifiers to predict signal vs. background events. In the case of a soft-margin classifier, i.e. one where it is not possible to ensure perfect classification accuracy on the training data, SVMs aim to find a decision boundary $f(\mathbf{x}; \mathbf{w}) = 0$ that minimises $\|\mathbf{w}'\|^2 + C\sum_i \epsilon_i$, where C is the hyperparameter representing cost, subject to the constraints that for all i, $y_i(\langle \mathbf{w}', \mathbf{x}_i \rangle + w_0) + \epsilon_i \geq 1$, $\epsilon_i \geq 0$. This optimisation is known as the primal problem. Equivalently, it is possible to solve for the dual problem, constraining $0 \leq \lambda_i \leq C$. Higher values of C yield a higher penalty for misclassifying a training point and thus a more complex prediction function, with a potential for overfitting [11].

We created SVMs using the `e1071` R package [12]. For all SVMs generated, we scaled all input variables to zero mean and unit variance.

Based on the results of the dimensionality reduction techniques FDA and PCA, we conducted a preliminary analysis comparing SVMs using two kernels: a linear kernel vs. a Gaussian radial basis function (RBF) kernel. A linear kernel implements the kernel function

$$k(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x_j} \rangle$$

A Gaussian RBF kernel has the kernel function

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma |\mathbf{x}_i - \mathbf{x}_j|^2)$$

where $\gamma > 0$ is a second hyperparameter indicating the bandwidth. The Gaussian RBF is a general-purpose kernel particularly well-suited to situations like this where we do not have prior knowledge of the relationship between variables. We created SVMs using these two kernels on a training set consisting of 20% of the entire dataset and compared the AMS values on the 80% testing data, using all predictor variables. We used the default hyperparameter values from the `e1071` package: $C = 1$ and $\gamma = 0.033$ ($\gamma = \frac{1}{\text{data dimension}}$).

Based on the results of the preliminary analysis comparing kernels, we proceeded using RBF kernel SVMs only. We divided the original dataset into training and test sets, with the training set being 20% of the total data. Since `e1071` implements SVMs using the Lagrangian dual, which allows for the use of kernel functions, the optimisation is computationally intensive when the sample size n is large. Our dataset has a very large sample size, so we selected a relatively small training set. We tuned the hyperparameters C and γ using 5-fold cross-validation on the training set, with a grid search over $C \in \{0.5, 1, 2\}$ and $\gamma \in \{0.01, 0.1, 0.5\}$. After cross-validation, we selected the model with the highest AMS value and calculated the AMS value of this model when used to predict on the testing dataset. We did this using the following sets of predictor variables: 1) Including all variables; 2) Removing the algebraic co-dependencies and uniformly distributed variables; 3) Removing the variables with the 10 lowest mutual information; 4) Using only the 10 variables with the highest mutual information.

3 Code

Code for these analyses can be found in the following GitHub repository: `sm1_large_hadron_collider`. We have created an R package containing a variety of useful functions for the analysis, which can be found here: `higgiesmalls`.

4 Results

4.1 Feature Selection

4.1.1 Missing Data

To find out whether the missing data had an impact on classification we proceeded to condition each variable V_i on whether it was classified 'b' or 's' which we denote by $V_i^{(b)}$ and $V_i^{(s)}$ and computed the proportion of missing data in each. We discovered that in both the `DER_mass_MMC`

and jet-variable case, $V_i^{(s)}$ had $13 - 18\%$ more missing data than $V_i^{(b)}$. This led us to conclude that using -999 values would help with classification and proceeded, leaving them as they were presented to us.

4.1.2 Co-Dependencies

The specific dependencies which we discovered were the following:

- `PRI_jet_all_pt` = `PRI_jet_leading_pt` + `PRI_jet_subleading_pt`
- `PRI_lep_pt` = `DER_pt_ratio_lep_tau` ∗ `PRI_tau_pt`

This gave us the option to either eliminate: none, `PRI_jet_all_pt` and `PRI_lep_pt` (the variables which were a result of the others) or we remove `PRI_jet_leading_pt`, `PRI_jet_subleading_pt`, `DER_pt_ratio_lep_tau` and `PRI_tau_pt`. The final condition reducing the dimension by four - but hoping that the remaining two variables contain enough information about the other four to not see detrimental affects.

4.1.3 Mutual Information Tests

Computing the mutual information between each variable V and the classifier variable y gave us a hierarchy of variables which matched up with our intuition from our initial plots. Table 1 contains a sample of these results, for a full table and chart of the mutual information please see the appendix.

Variables	Mutual Information
DER_mass_MMC	0.158
DER_mass_transverse_met_lep	0.099
PRI_met_sumet	0.031
PRI_met	0.028
PRI_lep_phi	0.004
PRI_met_phi	0.004

Table 1: Top, middle and lowest ranking variables by mutual information.

`DER_mass_MMC` ranked highest with `DER_mass_transverse_met_lep` and `DER_mass_vis` as second and third respectively, indicating that these variables are top contenders for classification - which agrees with the intuition from the density plots. Additionally all the uniform angle-type variables ranked lowest, which was to be expected, meaning they are clear contenders for elimination.

4.1.4 Principle Component Analysis

From the principle component analysis of the dataset we obtained an intuition of how keeping the missing variables in effected the separability of the data. We found by including the missing variables with -999 values we increase the separability of the dataset. See Figure 1 reference. Additionally by projecting variables onto the principle components we achieved a ranking for the impact of the variables on classification, see Figure 2.

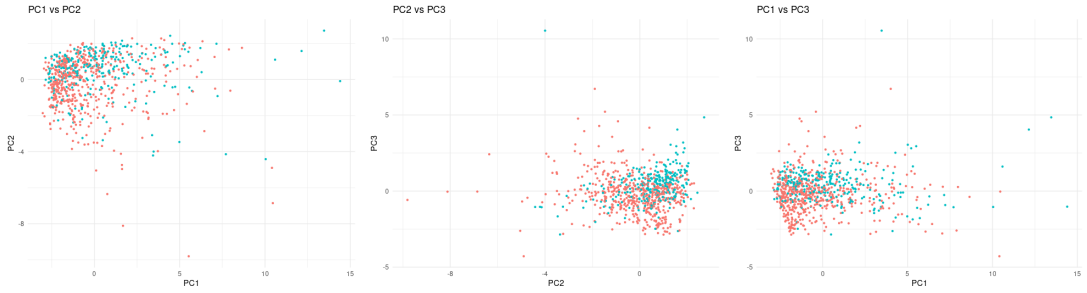

Figure 1: Data in the dimensions of the first 3 principle components, only using complete variables.

Figure 2: Data in the dimensions of the first 3 principle components, with all the variables used and missing values replaced by -999.

From further analysis of the contributions of each variable to each principle component (see appendix) along side the density plots we found that principle components one through 3 were largely describing the nature of this missing data and principle component four was largely due to the variables `DER_deltar_tau_lep`, and `DER_mass_vis` suggesting a strong correlation between them. Overall this analysis and these plots helped us visualise our data in a simpler way.

4.1.5 Feature Selection Conclusions

Consider the agreement of our PCA analysis with our mutual information tests, we decided on a hierarchical approach to feature selection. Opting for the following order:

1. Keep all variables.
2. Remove the (four) algebraic co-dependencies and uniform variables.
3. Remove the bottom ten ranking mutual information variables.
4. Only use the top ten ranking mutual information variables.

This would give us increasing strictness of variable elimination to get the highest AMS possible.

4.2 Fisher Discriminant Analysis

We found that the dataset had very poor linear separability across the different sets of predictor variables (Figure 3). Even after FDA, the embedding does not yield separability between classes. Removing the co-dependencies and then the uniformly distributed variables resulted in slightly lower ratios of between-class to within-class scatterness, but the values were comparable to the results of FDA on all predictor variables. Similarly, discarding the 10 variables with the lowest mutual information yielded an almost identical scatter ratio (Table 2). Using only the 10 variables with the highest mutual information, however, resulted in a lower scatter ratio (Table 2).

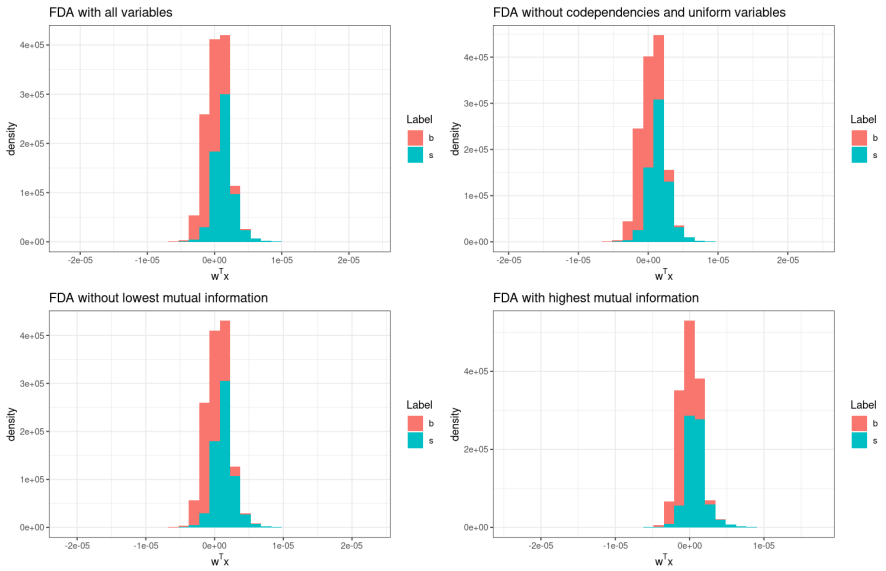

Figure 3: Fisher Discriminant Analysis embeddings for the following sets of predictor variables: 1) all variables; 2) removing the algebraic co-dependencies and uniformly distributed variables; 3) removing the 10 variables with the lowest mutual information; 4) using only the 10 variables with the highest mutual information.

Variables included	Scatter ratio
All variables	0.325
Without co-dependencies and uniform variables	0.324
Without lowest mutual information	0.325
Highest mutual information	0.254

Table 2: Scatter ratios resulting from FDA performed using different combinations of predictor variables.

4.3 Support Vector Machine Classification

Our preliminary SVM analysis showed that an RBF kernel SVM trained on 20% of the total dataset resulted in a 40% higher AMS value on the testing set than a linear kernel SVM trained on the same training set (linear kernel: AMS = 2.01; RBF kernel: AMS = 2.81). These results align with the FDA analyses, which indicated that the data was not linearly separable. Based on these results, we moved forward using only RBF kernel SVMs.

Cross-validation resulted in different hyperparameter values for each of the three sets of variables included (Table 3). The model built using all predictor variables resulted in a best value of $C = 1$, the default cost value for SVMs. The models built on the three other variable combinations, however, had better performance with $C = 2$. The bandwidth hyperparameter γ yielded highest AMS at a value of 0.1 for all models except the one using only the 10 variables with the highest mutual information, which had best performance at $\gamma = 0.5$.

The different sets of variables included resulted in varying AMS performance when applied to the testing data. The model built by excluding the 10 variables with the lowest mutual information values ($\gamma = 0.1, C = 2$) yielded the highest testing AMS out of the three models built (AMS = 2.866, Table 3), slightly higher than the model excluding the co-dependencies and uniformly distributed variables ($\gamma = 0.1, C = 2$, AMS = 2.857). The model using only the ten variables with the highest mutual information had the lowest testing AMS, with a value of 2.813, even lower than the model using all variables (Table 3).

Variables included	γ	C	Training AMS	Testing AMS
All variables	0.1	1	424.63	2.832
Without co-dependencies and uniform variables	0.1	2	433.68	2.857
Removing lowest mutual information	0.1	2	432.66	2.866
Highest mutual information	0.5	2	418.35	2.813

Table 3: Results of SVM-tuning. All SVMs used a radial kernel and were tuned using a grid search over $\gamma \in \{0.01, 0.1, 0.5\}$ and $C \in \{0.5, 1, 2\}$. The γ and C columns show the hyperparameter values yielding the highest AMS during 5-fold cross-validation. Training AMS shows the average AMS achieved during 5-fold cross-validation for the model using the best parameters. Testing AMS is the AMS achieved by the best model when applied to the 80% testing data.

5 Discussion

In this project our objective was to create a classifier which maximises the AMS value, naturally we used a variety of techniques and assumptions to attempt a scattershot style approach to the problem. Since we had varying reasons for rejection or inclusion of certain variables, we decided on having increasing levels of strictness with our variable inclusion.

This had mostly positive results, our initial rejection of co-dependencies marginally increased the AMS, and rejection of bottom 10 ranking mutual information variables even more so however the largest increase in AMS resulted from the use of an RBF kernel. Unfortunately considering only the top ten ranking mutual information variables led to diminishing returns, even less than including all variables. This is not particularly surprising as this involved rejecting 2/3 of the variables. There are clearly other families of variable selection we could have chosen - of which we are sure there are much higher AMS associated.

The choice of RBF kernel was a result of the substantial (40%) increase in AMS. Further study could have gone into finding alternative kernels (possibly polynomial or sigmoid) which may have produced higher AMS values; however the simplicity and multipurpose nature of RBF made it an ideal candidate for our attempt. Additionally we could have attempted logistic regression - which has the benefit of providing a probability distribution for our classes. Although unfortunately the use of an RBF kernel is not possible in that context.

In the actual Kaggle challenge, the winning AMS value was 3.38, achieved via a Gradient Boosting Classifier. Resulting in 16% more than our highest value. Not too bad.

A Density Plots

A.1 Density Plots with missing data

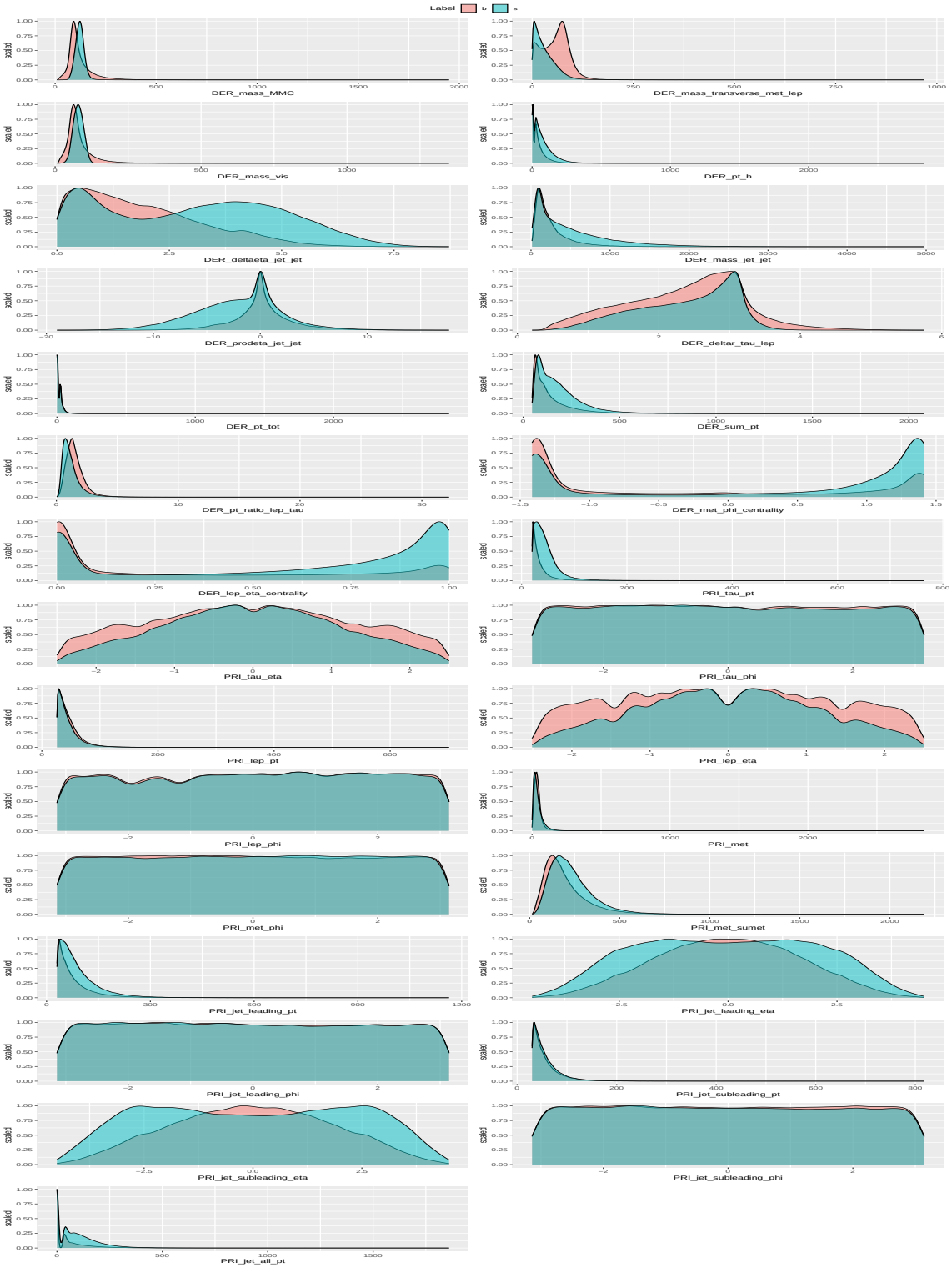

Figure 4:

A.2 Density Plots with -999 values

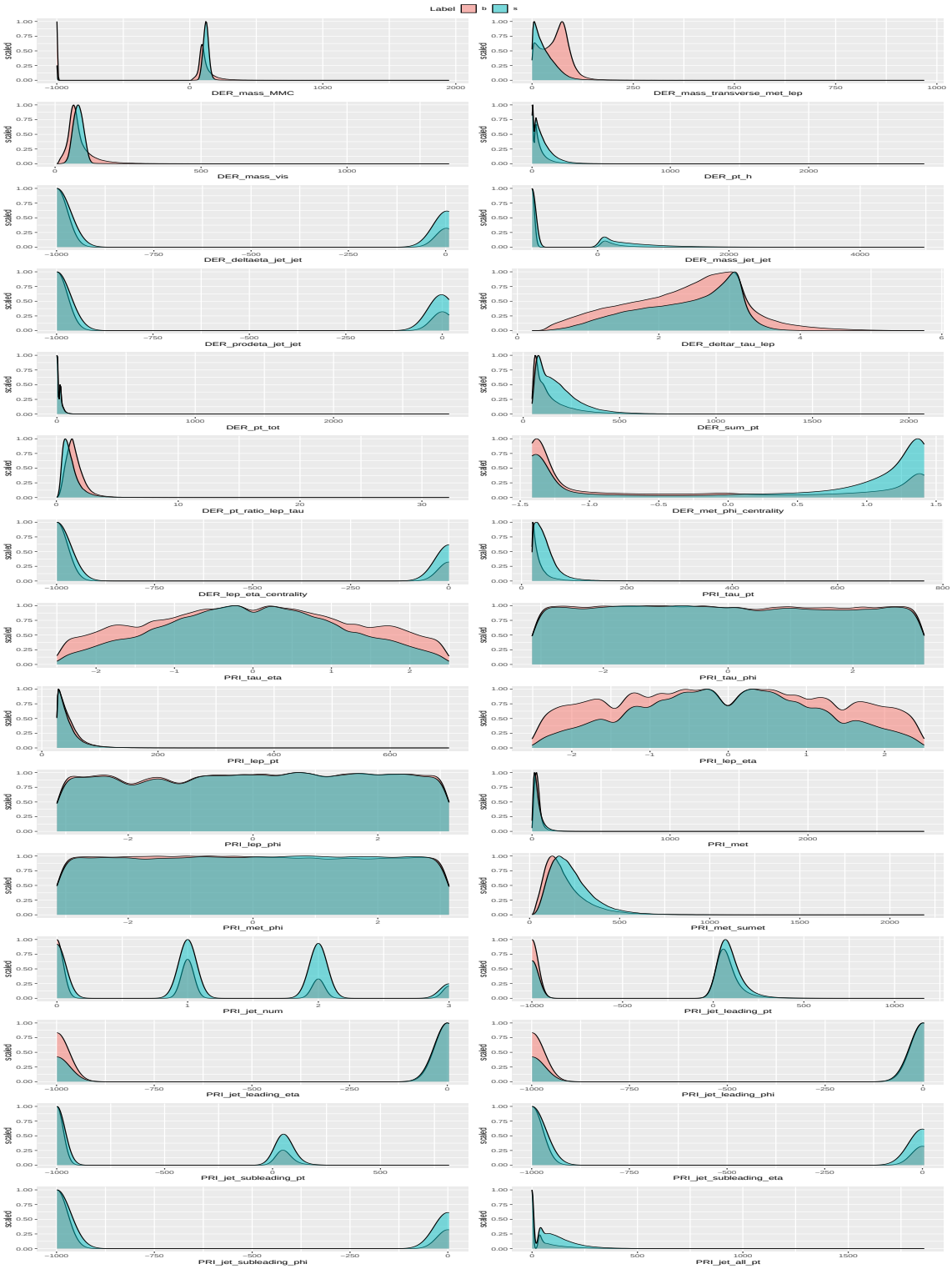

Figure 5:

B Principle Component Analysis

C Mutual Information Test

Table 4:

Variables	Mutual Information
DER_mass_MMC	0.158
DER_mass_transverse_met_lep	0.099
DER_mass_vis	0.091
PRI_tau_pt	0.065
DER_met_phi_centrality	0.048
DER_pt_ratio_lep_tau	0.044
DER_sum_pt	0.040
DER_pt_h	0.034
DER_deltaeta_jet_jet	0.033
DER_mass_jet_jet	0.033
DER_prodeta_jet_jet	0.032
PRI_met_sumet	0.031
PRI_met	0.028
PRI_jet_leading_pt	0.027
PRI_jet_leading_eta	0.026
DER_lep_eta_centrality	0.025
PRI_jet_subleading_eta	0.021
PRI_jet_num	0.020
DER_deltar_tau_lep	0.019
PRI_jet_leading_phi	0.016
PRI_lep_eta	0.014
PRI_jet_subleading_pt	0.014
PRI_lep_pt	0.013
PRI_jet_subleading_phi	0.013
DER_pt_tot	0.012
PRI_tau_eta	0.009
PRI_tau_phi	0.005
PRI_lep_phi	0.004
PRI_met_phi	0.004

Table 4: Top, middle and lowest ranking variables by mutual information.

References

[1] G. Aad, T. Abajyan, B. Abbott *et al.*, "Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC," *Physics Letters B*, vol. 716, no. 1, pp. 1–29, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S037026931200857X

[2] S. Chatrchyan, V. Khachatryan, A. Sirunyan *et al.*, "Observation of a new boson at a mass of 125 gev with the cms experiment at the lhc," *Physics Letters B*, vol. 716, no. 1, pp. 30–61, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0370269312008581

[3] C. Adam-Bourdarios, G. Cowan, C. Germain, I. Guyon, B. Kégl, and D. Rousseau, "Learning to discover: the Higgs boson machine learning challenge - documentation," *CERN Open Data Portal*, 2015.

[4] ATLAS collaboration, "Dataset from the ATLAS Higgs boson machine learning challenge 2014," *CERN Open Data Portal*, 2014.

[5] R. C. Ho Fai Wong, Wanda Wang and Yannick, "Higgs boson kaggle machine learning competition," *NYC Data Science Academy*, 2016. [Online]. Available: https://nycdatascience.com/blog/student-works/centaurs-higgs-boson-kaggle-competition/

[6] Zhang and Hancock, "Mutual information criteria for feature selection," *Similarity-Based Pattern Recognition*, 2011.

[7] R. Battiti, "Using mutual information for selecting features in supervised neural net learning," *IEEE Transactions in on Neural Networks*, 1994.

[8] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2016.

[9] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of Eugenics*, 1936.

[10] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[11] A. Karatzoglou and D. Meyer, "Support vector machines in R," *Journal of Statistical Software*, 2006.

[12] D. Meyer, E. Dimitriadou, K. Hornik, A. Weingessel, and F. Leisch, *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien*, 2023, r package version 1.7-13. [Online]. Available: https://CRAN.R-project.org/package=e1071

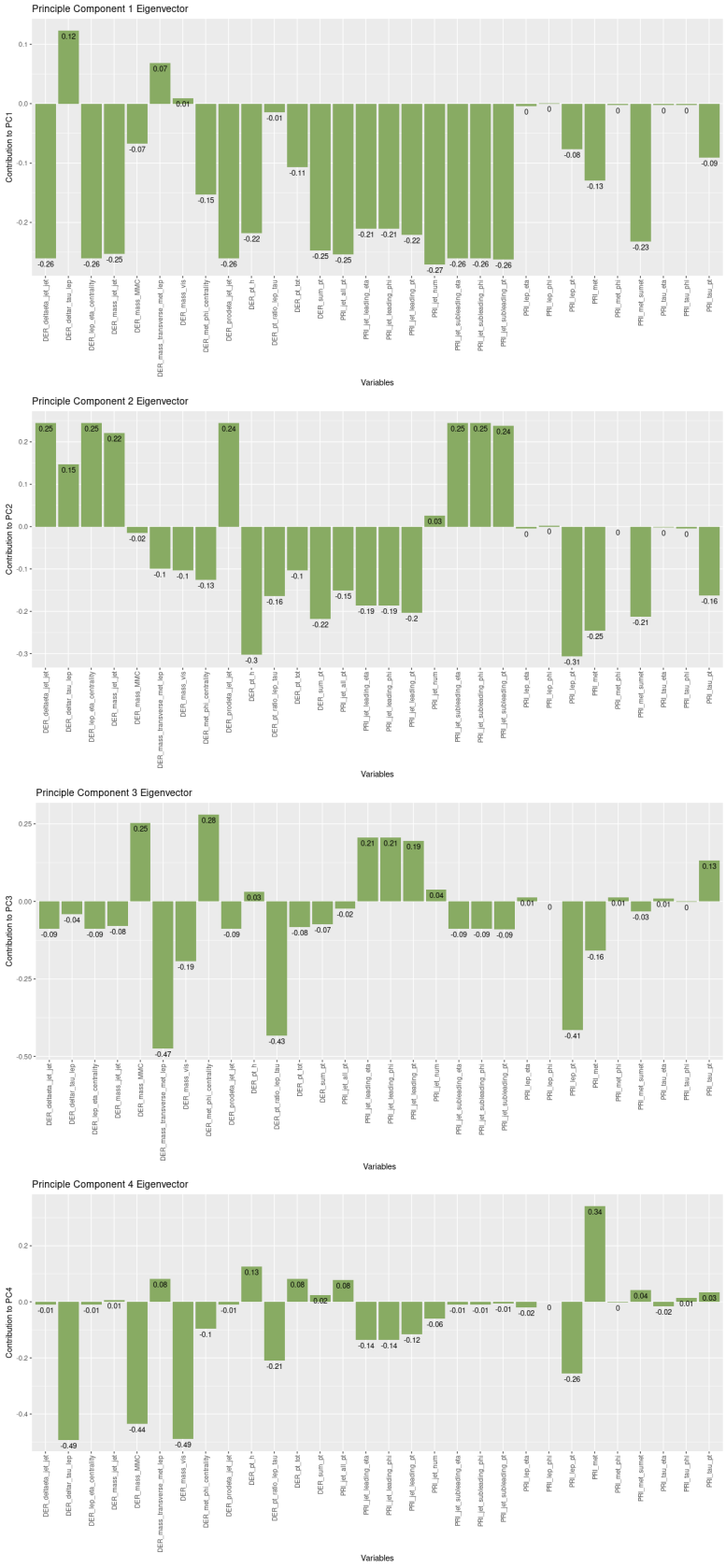

Figure 6: Plots of the contributions of each variable to each principle component.

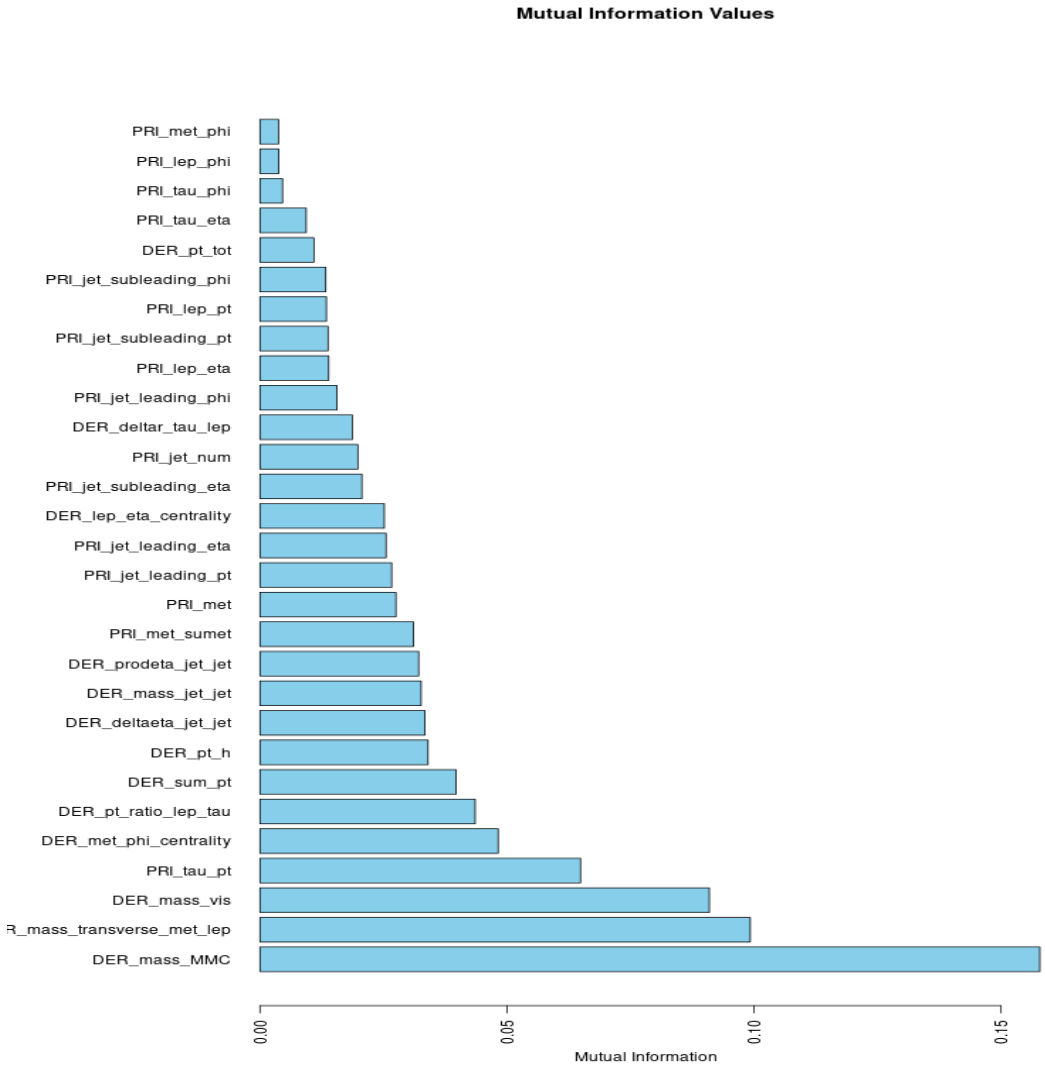
Mutual Information Values
PRI_met_phi
PRI_lep_phi
PRI_tau_phi
PRI_tau_eta
DER_pt_tot
PRI_jet_subleading_phi
PRI_lep_pt
PRI_jet_subleading_pt
PRI_lep_eta
PRI_jet_leading_phi
DER_deltar_tau_lep
PRI_jet_num
PRI_jet_subleading_eta
DER_lep_eta_centrality
PRI_jet_leading_eta
PRI_jet_leading_pt
PRI_met
PRI_met_sumet
DER_prodeta_jet_jet
DER_mass_jet_jet
DER_deltaeta_jet_jet
DER_pt_h
DER_sum_pt
DER_pt_ratio_lep_tau
DER_met_phi_centrality
PRI_tau_pt
DER_mass_vis
R_mass_transverse_met_lep
DER_mass_MMC
0.00
0.05
0.10
0.15
Mutual Information

Figure 7: