Advanced ML
Practical work 1

Authors

Name	UPC e-mail
Katrine Bukh Villesen	katrine.bukh.villesen@estudiantat.upc.edu
Daniel Reverter Condal	daniel.reverter@estudiantat.upc.edu
Marc Parcerisa Conesa	marc.parcerisa@estudiantat.upc.edu

November 16, 2025

Abstract

This project examines machine learning models for classifying events in the ATLAS Higgs Boson Machine Learning dataset. We compare L2-regularized logistic regression, Bayesian logistic regression, and several Naive Bayes variants designed to handle different feature distributions and structured missingness. The models are evaluated using Approximate Median Significance (AMS) together with standard classification metrics. Logistic regression and the Bayesian version of it shows similar predictive performances, while Bayesian inference provides deeper uncertainty quantification. Several of the Naive Bayes models achieve competitive AMS scores. The results highlight the trade-off between simplicity, interpretability, and predictive accuracy in large-scale event clarification.

1 Introduction

The discovery of the Higgs Boson at CERN's Large Hadron Collider (LHC) in 2012 [2] was one the most significant achievements in modern physics. Detecting it required distinguishing a few true signal events from a vast number of background interactions, motivating the use of advanced statistical and machine learning methods.

The Higgs Boson Machine Learning Challenge dataset was released by CERN and Kaggle and it simulates this classification task. Each observation represents a particle collision described by kinematic and engineered features derived from detector data, with the goal of classifying events as signal or background. Its high dimensionality and complex dependencies make it a strong benchmark for statistical modeling.

While complex models such as neural networks and boosted trees achieve high accuracy, they often lack interpretability and uncertainty quantification which is key in scientific contexts. Generalized linear models (GLMs) and Bayesian methods offer interpretable, principled alternatives that incorporate prior knowledge and provide uncertainty estimates. This project applies these approaches to the Higgs Boson dataset to assess their performance, interpretability, and robustness in large-scale event classification.

2 Problem statement

The objective of this project is to develop and evaluate statistical models for classifying events in the Higgs Boson dataset as either signal (Higgs Boson event) or background (non-Higgs event). The main goals are:

- To assess the ability of logistic regression to discriminate between signal and background events based on kinematic variables.
- To compare the performance of frequentist and Bayesian formulations in terms of predictive accuracy and calibration.

3 Related work

The search for the Higgs Boson has driven research in both particle physics and machine learning. Early analyses at the LHC relied on likelihood-based statistical models and multivariate techniques to separate signal from background events, forming the foundation of the 2012 discovery papers.

The dataset used in this project was released in 2014 as part of a Kaggle competition [1], which sparked interest in applying modern machine learning methods to this classification task. The top solutions employed ensemble models such as Gradient Boosted Decision Trees and XGBoost, achieving high accuracy through non-linear modeling and feature interactions. Deep learning approaches, including fully connected and convolutional networks, have also been explored, though at the cost of interpretability and transparency.

4 Data and Preprocessing

Our dataset is from the ATLAS Higgs Boson Machine Learning Challenge 2014. Some description of the data have already been given in the introduction but in the following a deep dive into the technical aspects of it will be given.

4.1 Data description

Each simulated event corresponds to a proton–proton collision at the LHC, generated in two stages: first, the underlying particle interactions are simulated based on known physics, and second, the resulting particles are propagated through a virtual model of the ATLAS detector. This produces synthetic events with statistical properties closely resembling those of real experimental data. Each event is assigned a weight reflecting its relative frequency under the simulated luminosity conditions, ensuring unbiased statistical comparisons between regions of the data. Of the 30 input variables, there are 17 "low-level" (or primitive) quantities about the bunch collision as measured by the detector and 13 "high-level" variables that are quantities computed from the primitive features, which were selected by the physicists of ATLAS.
Table 1 shows a summary of some characteristics of the dataset.

Property	**Variable**	**Value**
Number of observations	n	818,238
Number of features	d	30
Feature types	continuous, integer	29 / 1
Number of classes	C	2
Class distribution	imbalanced	approx. 35% signal, 65% background
Missing values	percentage	72.68% of events have missing value(s)

Table 1: Datas et characteristics.

4.2 Exploratory data analysis

A central goal of this project, as outlined in the proposal, is to understand the probabilities of each data point of belonging to a class $P(C_k|x_j)$ through accurate and interpretable models. The generative models used in this study rely heavily on the marginal distributions of the features of the classes, $P(x_j|C_k)$. Our EDA is designed to explore said distributions, and preprocess the data to make their profiles more manageable for the models. To do this, a comprehensive visual analysis was performed by plotting the distributions for all features, which can be seen in the grid of density histograms from Figure 3 (in the Appendix). Some preprocessing steps were required before further analysis of the actual distributions.

4.3 Preprocessing steps

The models used throughout this study have different requirements for the data preprocessing, which is why the designed strategy was split into two different stages:

4.3.1 Unaltering preprocessing steps

Missing values, encoded with the value -999 in the original dataset, were transformed into NaN, irrelevant competition-related variables were discarded, and the only ordinal (categorical) variable, the number of jets, was encoded numerically. These transformations were common across all models, and a processed dataset was saved for use mainly on Naive Bayes models.

4.3.2 Altering preprocessing steps

For models which required further processing, missingness was handled by a combination of removing instances, removing entire columns, and imputing values where appropriate; highly correlated features were removed to reduce multicollinearity, and all continuous features were standardized to zero mean and unit variance to improve optimization stability and ensure fair regularization. Finally, the target variable was mapped from categorical labels (“b” and “s”) to binary form. A detailed description of each step and the

variables affected is provided in Appendix B. These transformations were only necessary for logistic regression models, as the Naive Bayes Classifier intrinsically handles all kinds of variables, is capable of handing missing data with some tweaks, and doesn't care about multicollinearity or normality.

Data Splitting

Finally, both transformed datasets were split into learning and testing datasets with 85% and 15% of the data respectively. The learning data was used throughout the study to fit models, tune hyper-parameters and select the best models, whereas the test data was only used once at the end of the study to generate the final, unbiased performance metrics reported in the paper.

Visualization of features

Figure 1 shows a histogram of each feature after applying all the pre-processing steps, which conveys a better approximation of their underlying marginal distributions. The resulting plots reveal a wide variety

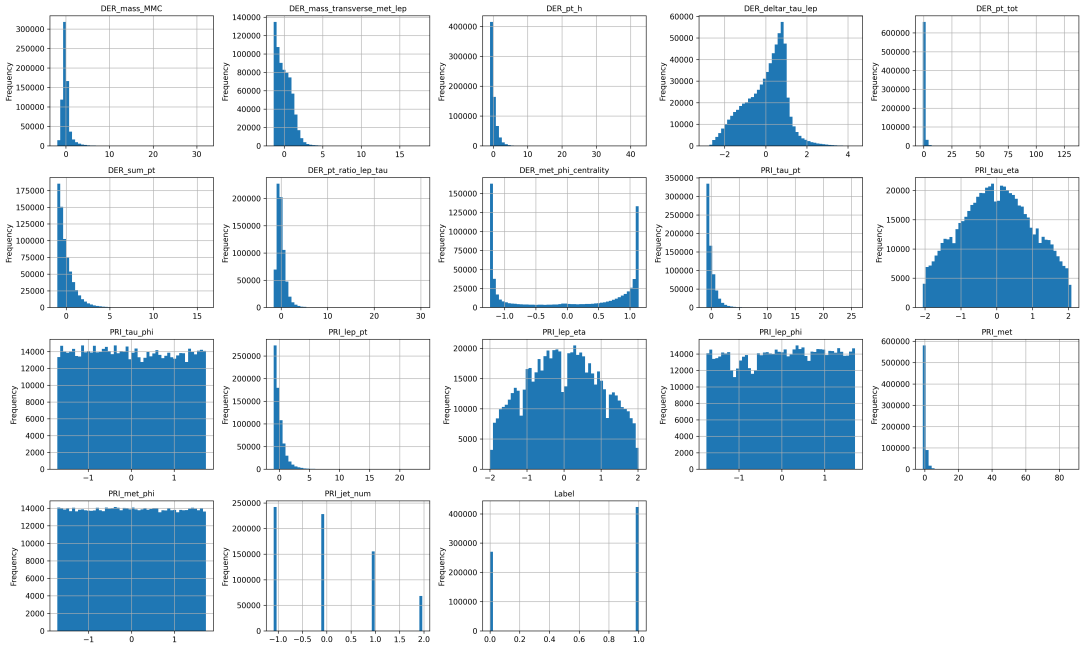

Figure 1: Histograms of each feature used in the training and testing of our models after being pre-processed as described in the section above.

of distributional shapes. The Gaussian assumption is clearly and decisively violated for a majority of the features. We observe:

- **Gaussian-like:** The two `*_eta` variables are symmetric and centered around 0, but they should have longer tails in order to be considered Gaussian. `DER_deltar_tau_lep` is somewhat bell shaped but skewed.
- **Skewed/Long-Tailed:** Most `DER_*` (Derived) and `PRI_*` (Primary) features are highly right-skewed, resembling log-normal or gamma distributions.
- **Bimodal:** `DER_met_phi_centrality` is U-Shaped with values concentrated around the extremes.
- **Uniform:** All `*_phi` variables look uniformly distributed, originally across $[-\pi, \pi]$ (uniform angle distributions).

5 Methodology

This section provides the detailed mathematical formulation, theoretical justification, and implementation plans for the three families of models under comparison: Logistic Regression (as a baseline), Bayesian Logistic Regression, and a proposed set of Non-Gaussian Naive Bayes classifiers.

5.1 Experimental protocol

Evaluation Metric: Approximate Median Significance (AMS)

In high-energy physics, the goal is not to maximize simple accuracy. Following the statement of the original challenge, the cost of a false positive (misidentifying background as signal) is different from a false negative (missing a true signal). The ultimate goal is statistical discovery. To align our project with the domain and to compare with previous works, we must adopt the evaluation metric used by the original ATLAS/Kaggle challenge: the Approximate Median Significance (AMS).

Formulation: The AMS is defined as

$$AMS = \sqrt{2\left((s+b+b_r)\log\left(1+\frac{s}{b+b_r}\right)-s\right)}, \tag{1}$$

where:

- s (signal) is the sum of weights of the true positives (signal events classified as signal).
- b (background) is the sum of weights of the false positives (background events classified as signal).
- b_r is a constant regularization term, $b_r = 10$, which penalizes solutions with low background statistics.

5.2 Method 1: Logistic regression (Baseline)

The baseline model is a standard L2-regularized Logistic Regression

5.2.1 Model formulation

The probability of a signal event ($y_i = 1$) given an input vector $\mathbf{x}_i$ is modeled via the logistic sigmoid function

$$P(y_i = 1|\mathbf{x}_i, \boldsymbol{\beta}) = p_i = \sigma(\eta_i) = \frac{1}{1+e^{-\eta_i}},$$

where η_i is the linear predictor, or log-odds

$$\eta_i = \log\left(\frac{p_i}{1-p_i}\right) = \boldsymbol{\beta}^T\mathbf{x}_i = \beta_0 + \sum_{j=1}^{D}\beta_j x_{ij}.$$

The model outcome y_i is assumed to follow a Bernoulli distribution with parameter p_i: $y_i|\mathbf{x}_i, \boldsymbol{\beta} \sim \text{Bernoulli}(p_i)$.

5.2.2 Theoretical properties and justification

The parameters $\boldsymbol{\beta}$ are found by Maximum Likelihood Estimation (MLE), which maximizes the log-likelihood of the observed data. To prevent overfitting and address quasi-separation, an L2 regularization (Ridge) penalty is added to the objective function. This leads to the penalized log-likelihood that the algorithm minimizes

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\text{argmin}}\left[-\sum_{i=1}^{N}\left(y_i\log(p_i)+(1-y_i)\log(1-p_i)\right)+\lambda||\boldsymbol{\beta}||_2^2\right].$$

5.2.3 Implementation details

The implementation is handled by the `sklearn.linear_model.LogisticRegression` class. The regularization hyperparameter $C = 1/\lambda$ is tuned via `GridSearchCV`. And we find that the optimal solution is $\lambda = 10^{-6}$.

5.3 Method 2: Bayesian logistic regression

This model extends the standard LR by adopting a fully Bayesian approach. Instead of finding a single point estimate $\hat{\boldsymbol{\beta}}$, we seek to determine the full posterior probability distribution of the parameters $P(\boldsymbol{\beta}|\mathbf{X}, \mathbf{y})$, which allows for a principled quantification of uncertainty.

5.3.1 Model formulation

The Bayesian model is constructed from three components: the likelihood, the priors, and the resulting posterior.

1. **Likelihood**: The likelihood function is identical to the standard LR model. It is the product of N independent Bernoulli trials, which defines the probability of the data given a specific set of parameters $\boldsymbol{\beta}$ as
$$P(\mathbf{y}|\mathbf{X}, \boldsymbol{\beta}) = \prod_{i=1}^{N} \text{Bernoulli}(y_i|p_i) = \prod_{i=1}^{N} p_i^{y_i}(1-p_i)^{1-y_i},$$
where $p_i = \sigma(\boldsymbol{\beta}^T \mathbf{x}_i)$.

2. **Prior Distributions**: The key Bayesian step is to specify a prior distribution $P(\boldsymbol{\beta})$ for the parameters, representing our beliefs before observing the data. A common, weakly informative prior is a Gaussian distribution. However, to satisfy the requirement for high theoretical rigor, we will select a more robust and theoretically justified prior.

3. **Posterior Distribution**: Via Bayes' theorem, the posterior distribution is proportional to the likelihood times the prior,
$$\underbrace{P(\boldsymbol{\beta}|\mathbf{X}, \mathbf{y})}_{\text{Posterior}} \propto \underbrace{P(\mathbf{y}|\mathbf{X}, \boldsymbol{\beta})}_{\text{Likelihood}} \cdot \underbrace{P(\boldsymbol{\beta})}_{\text{Prior}}.$$
This posterior distribution $P(\boldsymbol{\beta}|\mathbf{X}, \mathbf{y})$ encapsulates all available information about the parameters and is the central object for inference.

5.3.2 Theoretical properties and justification

Intractability and Inference: The posterior distribution defined above is intractable. The product of a Bernoulli likelihood (with its sigmoid link) and any common prior (like a Gaussian or Student-t) does not result in a closed-form analytical solution. Therefore, we must use approximation techniques. We will use Markov Chain Monte Carlo (MCMC) methods, specifically the No-U-Turn Sampler (NUTS), which is a highly efficient Hamiltonian Monte Carlo algorithm that, given enough time, is guaranteed to converge to the true posterior distribution.

Justification for a Robust Prior (Student-t): While a Gaussian prior is common (and corresponds to L2 regularization in a MAP-estimation framework), it has theoretical drawbacks. A Gaussian prior's light tails (decaying as e^{-x^2}) can be too restrictive, shrinking genuinely large, important coefficients excessively. In our feature-wise large problem, it is highly likely that most features have a small effect (coefficients near 0) while a few (especially the high-level features) have a very large, significant effect. A more robust prior is needed, one that applies strong shrinkage to the noise coefficients but has heavy tails that allow the signal coefficients to remain large if the data provides sufficient evidence. The Student-t distribution is the ideal

candidate. As recommended by Gelman et al. (2008) for logistic regression, we will use a Cauchy distribution (a Student-t distribution with $\nu = 1$ degree of freedom) as a default, weakly informative prior. This prior is more conservative and robust, automatically applying more shrinkage to higher-order interactions and providing stable inferences even in cases of complete data separation, a common issue in logistic regression. Our chosen prior model is:

- Intercept: $\beta_0 \sim \mathcal{T}(\nu = 1, \mu = 0, s = 10)$ (A weakly informative prior with a wide scale, where $\mathcal{T}$ refers to the Student's T distribution).
- Coefficients: $\beta_{j\neq 0} \sim \mathcal{T}(\nu = 1, \mu = 0, s = 2.5)$ (The default Cauchy prior recommended by Gelman et al.).

5.3.3 Implementation details

The model will be implemented using the PyMC probabilistic programming library. PyMC allows for a declarative definition of the model and provides access to advanced MCMC samplers like NUTS.

5.3.4 Inference and Interpretability:

The Bayesian model provides a significant upgrade in inference. Instead of a single point-estimate $\hat{\beta}_j$, we have the entire posterior distribution $P(\beta_j|\mathbf{X}, \mathbf{y})$ for each feature, which is plotted together with the convergence plot in Figure 4 in Annex C and contained in the trace object. Also, a numerical summary of the parameters inference is found in Table 3, containing the 95% Highest Density Interval (HDI). This is the narrowest interval that contains 95% of the posterior probability mass. If the 95% HDI does not contain 0.0, we conclude the feature has a statistically credible effect. This is a more intuitive and powerful statement than a p-value. Basing our results on this intervals, we would say that signal events tend to exhibit *higher Higgs–mass estimators and overall transverse activity*, whereas background events show *larger τ–lepton separation, stronger lepton and τ kinematics, specific MET directions, and increased jet multiplicity.*

5.4 Method 3: Non-Gaussian Naive Bayes

Naive Bayes Classifiers rely on the conditional probabilities of the features conditioned to each category of the target variable to calculate the probability of each data point to belong to each class,

$$P(C_k|\mathbf{x}_i) \propto P(\mathbf{x}_i|C_k) \cdot P(C_k), \tag{2}$$

by assuming that the features themselves are completely independent from one another,

$$P(\mathbf{x}_i|C_k) = \prod_{j=1}^{D} P(x_i^j|C_j).$$

Thanks to the property of Equation 2, this family of classifiers skips computing probabilities altogether, and rather relies on a *discriminant* function to choose the class that will have the *maximum* $P(C_k|\mathbf{x}_i)$, or rather, to simplify, its logarithm, as follows:

$$f(\mathbf{x}_i) = \underset{C_k \in C}{\operatorname{argmax}} \left\{ \log P(C_k) + \sum_{j=1}^{D} \log P(x_i^j|C_k) \right\}.$$

If one further assumes that all features are gaussian distributed, the model is the well-known *Gaussian Naive Bayes Classifier.*

This family of models is one of the most well studied, as they tend to be really computationally inexpensive, have easy interpretability, and sometimes can yield good enough results. However, due to its overly-simplifying assumptions, whenever possible, more powerful Bayesian models should be considered.

5.4.1 Motivation

Figure 2 shows the comparison of the distributions of the features in the dataset between the two target classes. In it, one can see that some of the features follow slightly different, although strongly overlapping, distributions. For this reason, we expected Naive Bayes Classifiers to perform good enough[1].

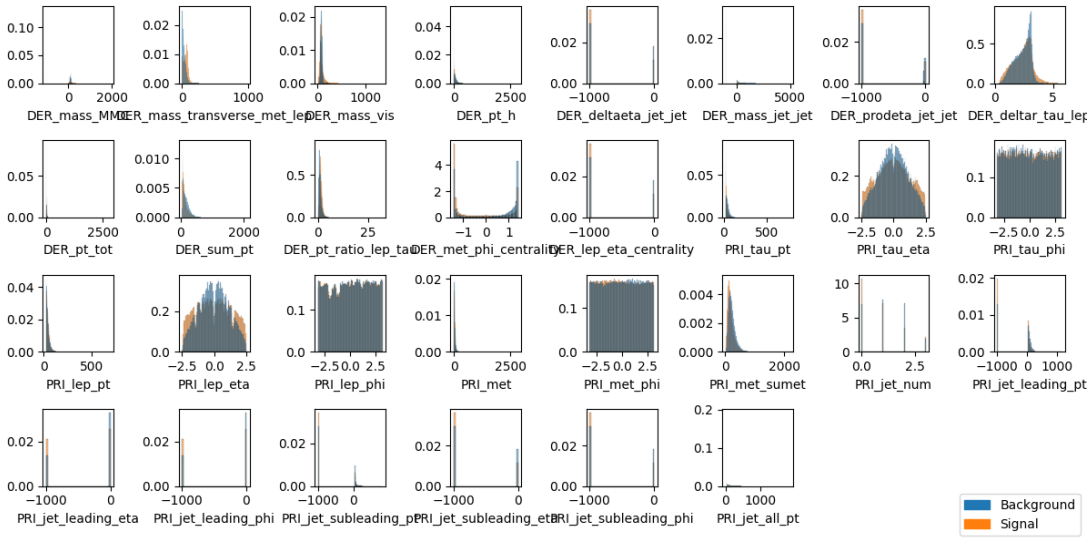

Figure 2: Comparison of the distributions of all the features in the dataset between the two target classes.

5.4.2 Model formulations

As explained above, the Gaussian formulation of the model is what is usually implemented in most statistical libraries. However, as stated before, the profiles of most variables clearly diverge from gaussian distributions, so several different implementations were considered for exploration, which can be categorized based on the following criteria:

- Based on whether they consider missingness of a value as actionable information: As showed before, some columns are missing lots of data points and, most importantly, they aren't missing completely at random. For that reason, the models we call "robust" consider that

$$\mathcal{P}_{X^j}(x|C_k) = \begin{cases} p^j_{\text{NA}} \text{ if } x = \text{NaN} \\ (1 - p^j_{\text{NA}}) P_{X^j}(x|C_k; x \neq \text{NaN}) \text{ if } x \neq \text{NaN} \end{cases} .$$

- Based on how they estimate each variable's distribution: We considered the following distribution estimations:
  - Gaussian: The baseline, it assumes each feature is gaussian-distributed, and computes the mean and average:
$$X^j|C_k \sim N(\mu^j, (\sigma^j)^2)$$
  - Yeo-Johnson-transformed Gaussian: The Yeo-Johnson is an approximate *generalization* of the Box-Cox transformation that allows variables to be negative [3]. Using it, it optimizes a λ^j value such that
$$\psi^{\text{YJ}}(X^j; \lambda^j)|C_k \sim N(\mu^j, (\sigma^j)^2).$$
    Note that this is an *assumption* of the model, but it is usually impossible to know the *actual* distribution of $\psi^{\text{YJ}}(X^j; \lambda^j)$, as we don't know the real distribution of X^j.

[1] We won't lie, we didn't have very high expectations of them.

- Histogram: This method makes little to no assumptions on the distribution of the data, as it does not attempt to model it further than with a histogram-based profile of the feature in the training dataset. It then uses said histogram, normalized to 1, during inference to approximate each point's likelihood.
- Kernel Density Estimation (KDE): This method also makes very few assumptions, as it uses the data itself to predict the probability at each point. It does so by approximating each feature's probability density function to

$$p_{X^j}(x|C_k) = \frac{1}{nh} \sum_{\substack{y_i = C_k \\ x_i^j \neq \text{NaN}}} K\left(\frac{x - x_i^j}{h}\right),$$

where n is the number of non-missing data points in feature X^j for class C_k, h is a hyperparameter known as "bandwidth", and K is the *kernel function* which gives name to the method. The most typical kernel choice is the gaussian,

$$K(u) = \phi(u) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-u^2}{2\sigma^2}\right),$$

where one can easily see the parallelism between h^2 and its variance σ^2, as they are sometimes joined together as $\tilde{h} = h\sigma$. Note that KDE estimators are, by design, lazily evaluated, but in the code we implemented an eager version which pre-computes the estimation on a large-enough grid such that inference is as fast as a lookup table.

- Based on whether they assume full independence or not: Because the dataset has only one categorical variable (number of jets), which, as outlined before, seems to have a strong effect on the behavior of some features (e.g their missingness), a deviation of the theoretical Naive Bayes framework was also considered. In this version, all variables are assumed independent from one another, except for a dependency with both the label and the categorical variable. Note that this is equivalent to fitting a classifier for each value of the number of jets, and adding at the end the log probability of said value (conditioned to the class),

$$P_{X^j}(x|C_k) = P_{X^j}(x|\text{num\_jets}; C_k) \cdot P_{X^{\text{num\_jets}}}(\text{num\_jets}|C_k).$$

5.4.3 Implementation details

A custom *Python* module was developed to allow for full control over the definition of all these kinds of models while reusing most of the code across classes. To do so, two interchangeable pieces were developed: the model kind, which defines whether complete independence is assumed or not (`BespokeNB` and `CategoricalAwareBespokeNB` respectively); and a set of estimators, which control the rest.

5.5 Comparison framework

All three models were trained on the same dataset (with some differences in preprocessing, as described above). Model performance is primarily measured by the Approximate Median Significance (AMS) as described in Equation 1, and with ROC-AUC, accuracy and F1-score as complementary metrics. In addition to predictive performance we assessed interpretability, uncertainty estimation and computational efficiency. For model selection and hyperparameter tuning, especially in the case of the Naive Bayes classifiers, a low-footprint testing framework was developed to perform experiments with all types of combinations of model kind, estimator and other hyperparameters. With it, we used cross-validation strategies whenever the model was computationally inexpensive (e.g. Gaussian), and simple train/validation splitting otherwise (e.g. KDE). Finally, once a set of hyperparameters was selected for each model, a final training was performed with the

entire learning split, and the models results evaluated with the testing one. At the end, logistic regression provided a baseline for linear separability, Bayesian logistic regression added uncertainty quantification through posterior inference and the non-Gaussian naive Bayes model served as a probabilistic benchmark with surprisingly good results.

6 Discussion of results

6.1 Overall performance

To sum up the performance of each model, Table 2 presents four different performance metrics evaluated over the testing split, for the most relevant models trained, as well as two benchmark models for reference, among which is the winner of the Kaggle competition. For more details on the actual hyperparameters used, and the results obtained, please refer to the code attached.

Model	**AMS**	**ROC-AUC**	**Accuracy**	**F1-Score**
Our Models				
Logistic Regression (L2)	0.74	0.77	0.71	0.71
Bayesian LR (Student-t)	0.75	0.70	0.77	0.70
Gaussian NB (Drop Missing Rows)	0.43	-	0.74	0.74
Gaussian NB (Drop Missing Columns)	0.61	-	0.72	0.68
Histogram NB (Drop Missing Rows)	0.54	-	0.74	0.72
Histogram NB (Drop Missing Columns)	0.72	-	0.67	0.67
Eager Gaussian KDE NB (Drop Missing Rows)	0.50	-	0.79	0.79
Eager Gaussian KDE NB (Drop Missing Columns)	0.75	-	0.74	0.71
Yeo-Johnson Gaussian NB (Drop Missing Rows)	0.25	-	0.58	0.46
Yeo-Johnson Gaussian NB (Drop Missing Columns)	0.03	-	0.66	0.40
Robust Gaussian NB	0.60	-	0.74	0.68
Robust Histogram NB	0.76	-	0.72	0.71
Robust Eager Gaussian KDE NB	0.60	-	0.72	0.69
Robust Yeo-Johnson Gaussian NB	0.20	-	0.67	0.43
Robust Gaussian NB with Categorical Dependency	0.59	-	0.74	0.69
Robust Histogram NB with Categorical Dependency	0.68	-	0.74	0.72
Robust Eager Gaussian KDE NB with Categorical Dependency	0.86	-	0.79	0.77
Robust Yeo-Johnson Gaussian NB with Categorical Dependency	0.48	-	0.65	0.55
Benchmarks				
Baldi et al. (DNN)	$\sim$(3.5–3.8)	$\sim$0.88	$\sim$0.81	-
Kaggle Winner (XGBoost)	$\sim$3.8+	$\sim$0.90	-	-

Table 2: Summary of each models performance compared to two benchmark models. All hyperparameters have been tuned with cross-validation or other feasible resampling techniques, and only the best combination of them is shown. ROC-AUC for Naive Bayes Classifiers is not shown because by the time the decision to add this metric was taken, 40K NB experiments had already been performed. Note that models in which rows are dropped may have its accuracy inflated, as some testing instances had to be ignored due to missingness.

6.2 Detailed analysis

The results of our experiments indicate that the Bayesian classifiers achieved competitive performance compared to classical baselines. Overall, models incorporating probabilistic reasoning were able to better capture the uncertainty in event classification. We see in Table 2 that logistic regression and the Bayesian version of

it have very similar performance, which is espected, because when using a very large sample, the variance of the parameters is really low. The parameter density converges to the true value when dataset size tends to infinity, and also to the MLE from logistic regression (if also applied to infinite data). That is why the performance of Bayesian logistic regression is the same as the logistic regression. Any further improvements should come from changing the model and not from more data. But where they do differ is in interpretability. Bayesian inference gives us a more nuanced view of our predictive results as we are looking at a predictive distribution and not just a single point estimate.

Naive Bayes Classifiers, in turn, showed surprisingly good results, especially those models that approximated distributions numerically, rather than parametrically, as they were capable of better capturing the nuances of each feature's distribution, without assuming Gaussianity. Unsurprisingly, models with more complex probability estimations were capable of better differentiating the classes, even getting close to the Baldi et. al. result with a *Robust Eager Gaussian KDE Naive Bayes with Categorical Dependency*, which was the most complex model of all. This last result was somewhat expected, as KDE is capable of approximating a variable's distribution similarly to how a histogram does it, but because it relies on a continuous function, it is able to capture finer differences between neighbor data points.

7 Conclusions

In this project three modeling approaches, logistic regression, Bayesian logistic regression, and non-Gaussian Naive Bayes, was compared on the task of distinguishing Higss Boson signal events from background collisions. Logistic regression served as a strong baseline across all experiments, achieving stable performance with minimal computational cost. The Bayesian formulation produced nearly identical predictive accuracy and AMS, which was expected given the size of the dataset: when using hundreds of thousands of observations, the posterior concentrates tightly and approaches the MLE solution. However, Bayesian inference provides clear advantages in interpretability by yielding full posterior distributions and credibility intervals for each coefficient, which enable a more nuanced assessment of feature importance.

The non-Gaussian Naive Bayes models showed highly variable performance, but some of the estimators, in particular histogram-based and KDE-based models that incorporated missingness patterns, performed surprisingly well. Their best variants achieved AMS scores comparable to the discriminative models.

Overall, our findings shows that even though more complex and expensive models and methods can outperform these interpretable models on this Higgs Boson challenge, statistical models that are carefully designed will still offer competitive performance and some valuable scientific insights. Future work could explore more complex models and more sophisticated uncertainty-aware approaches.

References

[1] Claire Adam-Bourdarios, Glen Cowan, Cécile Germain, Isabelle Guyon, Balàzs Kégl, and David Rousseau. The Higgs boson machine learning challenge. In Glen Cowan, Cécile Germain, Isabelle Guyon, Balázs Kégl, and David Rousseau, editors, *Proceedings of the NIPS 2014 Workshop on High-energy Physics and Machine Learning*, volume 42 of *Proceedings of Machine Learning Research*, pages 19–55, Montreal, Canada, 13 Dec 2015. PMLR.

[2] CERN. Observation of a new particle in the search for the standard model higgs boson with the atlas detector at the lhc. *Physics Letters B*, 716, 2012.

[3] The SciPy community. *SciPy Documentation: Yeo-Johnson Transformation.* Package version 1.16.2. Accessed: 2025-11-15.

A Exploratory data analysis

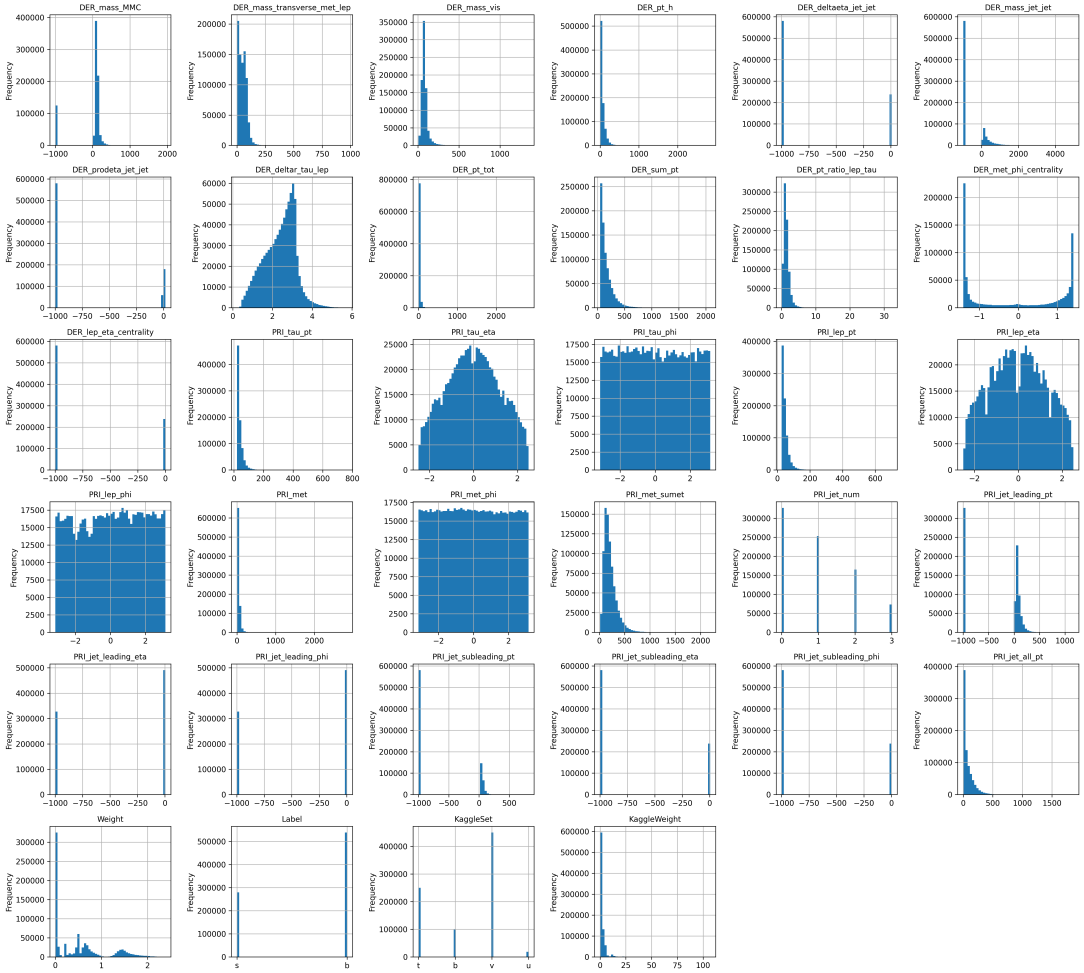

Figure 3: Marginal densities of each feature presented as histograms before being pre-processed.

B Preprocessing of data

B.1 Treatment of Outliers:

Some feature values fall outside the typical interquartile range and could be considered outliers by standard statistical definitions. However, these values originate from the true simulated physics distribution as seen in Figure 1 and represent rare but valid events. Therefore, they are not treated as outliers and are retained in the dataset.

B.2 Treatment of Missing and Incoherent or Incorrect Values

In the dataset the value -999 corresponds to a missing value. As seen in Table 1, there are missing values in more than 71% of the events; therefore, we looked deeper into these. The missing values in columns with `'jet'` in their name are from particles with `PRI_jet_num` equal to 0 or 1, so instead of dropping the rows which would skew the data set and make it 71% smaller we will drop those columns. They only give information about particles with 2 or 3 jets. There is another column, `DER_mass_MCC` with 15.23% of missing

values. In this case we prefer to delete those rows since the variable is important and we don't want to lose it, and 15.23% of the data is an acceptable price to keep it.

B.3 Elimination of Irrelevant Variables

Two variables related to the Kaggle competition: `KaggleSet` and `KaggleWeight` are irrelevant to our models and therefore discarded. Variable `Weight` is separated from the features, but saved since it is used in the evaluation metric.

B.4 Elimination of Redundant Variables

Highly correlated variables can cause multi-co-linearity and slow down Bayesian sampling. We detect features with $|\text{correlation}| > 0.85$ and drop one of each pair.

- Dropped `DER_mass_vis` (correlated with `DER_mass_MMC` at 0.91)
- Dropped `PRI_met_sumet` (correlated with `DER_sum_pt` at 0.91 and `PRI_jet_all_pt` at 0.89)
- Dropped `PRI_jet_all_pt` (correlated with `DER_sum_pt` at 0.97)

B.5 Coding of Non-Continuous Variables

The categorical variable `PRI_jet_num` represents the number of jets (0, 1, 2, or 3). Even though it is technically discrete, the numeric values have an inherent ordering. Treating `PRI_jet_num` as numeric implicitly assumes that the log-odds of being the response being 'signal' changes monotonically with jet count.

B.6 Extraction of New Variables

Trusting the criteria of the experts in CERN, the most explicative high-level features are already implemented in the original dataset.

B.7 Transformation of variables

We will use standard scaling on each variable, fitting with only training data to avoid data leakage. The Non-Gaussian Naive Bayes model does not require this scaling, as it models each feature's density independently. The justification for this transformation is the following:

1. **Optimization**: The convergence of the optimization algorithms used to find the maximum likelihood estimate is highly sensitive to the scale of the features. Features with large variances can dominate the objective function, slowing or preventing convergence.
2. **Regularization**: L1 and L2 regularization terms penalize the sum of coefficients. This penalty is only fair and meaningful if all features are on a comparable scale; otherwise, the penalty is biased.
3. **Prior Interpretation**: In our Bayesian model, we place priors on the β_j coefficients. These priors (e.g., $\beta_j \sim \mathcal{T}(0, 2.5)$) are defined on the coefficients and implicitly assume that the features have been standardized.

Also we will map our target variable `Label` from values b and s to 1 and 0 because some libraries used later are design to use these as labels.

C Bayesian Logistic Regression

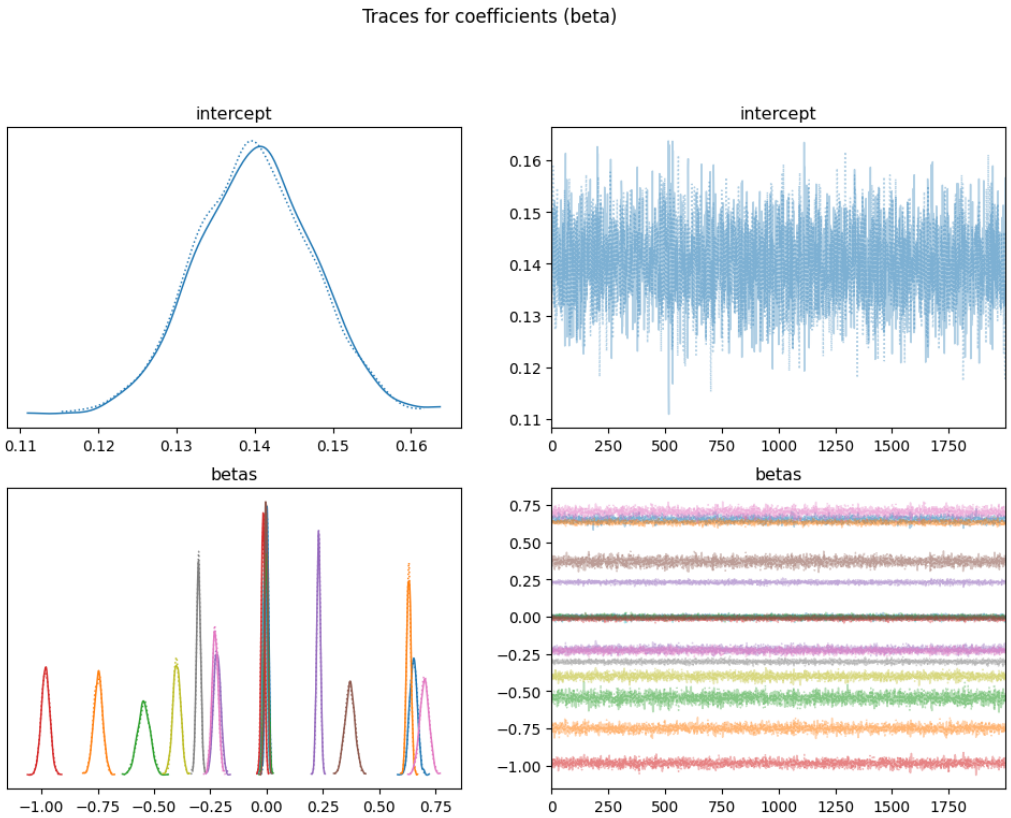

Figure 4: Probability distribution and convergence plot for bayesian logistic regression

Variable	**mean**	**sd**	**hdi_2.5%**	**hdi_97.5%**
intercept	0.140	0.007	0.126	0.155
DER_mass_MMC	0.653	0.017	0.622	0.685
DER_mass_transverse_met_lep	0.632	0.010	0.612	0.650
DER_pt_h	-0.545	0.028	-0.597	-0.487
DER_deltar_τ_*lep*	-0.982	0.019	-1.018	-0.945
DER_pt_tot	0.231	0.008	0.214	0.247
DER_sum_pt	0.370	0.022	0.327	0.412
DER_pt_ratio_lep_τ	0.702	0.021	0.661	0.744
DER_met_phi_centrality	-0.302	0.009	-0.321	-0.284
PRI_tau_pt	-0.399	0.017	-0.431	-0.365
PRI_tau_eta	-0.008	0.009	-0.026	0.008
PRI_tau_phi	0.002	0.007	-0.012	0.017
PRI_lep_pt	-0.748	0.020	-0.789	-0.709
PRI_lep_eta	-0.002	0.009	-0.020	0.016
PRI_lep_phi	-0.016	0.008	-0.030	-0.001
PRI_met	-0.220	0.016	-0.250	-0.188
PRI_met_phi	-0.002	0.007	-0.016	0.012
PRI_jet_num	-0.228	0.014	-0.255	-0.201

Table 3: Posterior summary of Bayesian logistic regression coefficients