Tuning of logistic regression for Higgsboson ML challenge

Ettore Fincato, Hannah Sansford, Harry Tata

January 15, 2022

Abstract

Write abstract here.

1 Introduction

1.1 Background

The Higgs boson can *decay* through various different processes, producing other particles in the process. In physics, one calls a decay into specific particles a *channel*. Until fairly recently, the Higgs boson had been seen only in boson pair decay channels. It is now of importance to seek evidence on the decay into *fermion* pairs, specifically *tau-leptons* or *b-quarks*, and to measure their characteristics [2]. The ATLAS experiment [1] was the first to report evidence of the H to tau-tau channel and the goal of this report is to improve on this analysis.

1.2 Overview

The Atlas experiment at CERN provided simulated data that was used to optimise the analysis of the Higgs boson. In the Large Hadron Collider (LHC), proton bunches are accelerated in both directions on a circular trajectory. This results in some of the protons colliding as the bunches cross the ATLAS detector (called an *event*), which produces hundreds of millions of proton-proton collisions per second. The particles resulting from each event are detected by sensors and, from this raw data, certain real-valued features are estimated [2].

Most of the uninteresting events (called the *background*) are discarded using a real-time multi-stage cascade classifier. However, many of the remaining events represent known processes that are also known as *background*. Our aim is to find the region of the feature space in which there is a significant excess of events compared to what known background processes can explain (called *signal*).

Once the region has been fixed, the significance of the excess is determined using a statistical test. If the probability that the excess has been produced by background processes falls below a pre-determined limit, the new particle is deemed to be discovered.

2 Problem Formulation

Let $\mathcal{D} = \{(\boldsymbol{x}_1, y_1, w_1), ..., (\boldsymbol{x}_n, y_n, w_n)\}$ the the training set, where $\boldsymbol{x}_i \in \mathbb{R}^d$ is a d-dimensional feature vector, $y_i \in \{\text{b,s}\}$ is the label, and $w_i \in \mathbb{R}^+$ is a non-negative weight. Let $\mathcal{S} = \{i : y_i = \text{s}\}$ and $\mathcal{B} = \{i : y_i = \text{b}\}$ be the index sets of signal and background events respectively, and let $n_\text{s} = |\mathcal{S}|$ and $n_\text{b} = |\mathcal{B}|$ be the number of simulated signal and background events.

The simulated dataset also includes importance weights for each event. Since the objective function (5) depends on the *unnormalised sum* of weights, in order to make the setup invariant to the *numbers* of simulated events n_s and n_b, the sum across each set (test/training) and each class (signal/background) is set to be fixed, i.e.,

$$\sum_{i \in \mathcal{S}} w_i = N_\text{s} \quad \text{and} \quad \sum_{i \in \mathcal{B}} w_i = N_\text{b}. \tag{1}$$

These normalisation constants N_{s} and N_{b} are simply the *expected total number* of signal and background events, respectively, during the time interval of the data taking. The individual weights are then proportional to the conditional densities,

$$p_{\mathrm{s}}(\boldsymbol{x}_i) = p(\boldsymbol{x}_i|y = \mathrm{s}) \quad \text{and} \quad p_{\mathrm{b}}(\boldsymbol{x}_i) = p(\boldsymbol{x}_i|y = \mathrm{b}),$$

divided by the instrumental densities $q_{\mathrm{s}}(\boldsymbol{x}_i)$ and $q_{\mathrm{b}}(\boldsymbol{x}_i)$, i.e.,

$$w_i \propto \begin{cases} p_{\mathrm{s}}(\boldsymbol{x}_i)/q_{\mathrm{s}}(\boldsymbol{x}_i), & \text{if} \quad y_i = \mathrm{s}. \\ p_{\mathrm{b}}(\boldsymbol{x}_i)/q_{\mathrm{b}}(\boldsymbol{x}_i), & \text{if} \quad y_i = \mathrm{b}. \end{cases} \tag{2}$$

Now, let $g : \mathbb{R}^d \to \{\mathrm{b},\mathrm{s}\}$ be a classifier. Let the *selection region* $\mathcal{G} = \{\boldsymbol{x} : g(\boldsymbol{x}) = \mathrm{s}\}$ be the set of points classified as signal, and let $\hat{\mathcal{G}}$ denote the *index set* of points that g classifies as signal, i.e.,

$$\hat{\mathcal{G}} = \{i : f(\boldsymbol{x}_i) \in \mathcal{G}\} = \{i : g(\boldsymbol{x}_i) = \mathrm{s}\}.$$

Then we have that

$$s = \sum_{i \in \mathcal{S} \cap \hat{\mathcal{G}}} w_i \tag{3}$$

is an unbiased estimator of the expected number of signal events selected by g, and similarly,

$$b = \sum_{i \in \mathcal{B} \cap \hat{\mathcal{G}}} w_i \tag{4}$$

is an unbiased estimator of the expected number of background events selected by g. Alternatively, s and b are the *unnormalised* true and false positive rates, respectively.

High-energy physicists suggest the use of the *approximate median significance* (AMS) objective function defined by

$$\mathrm{AMS} = \sqrt{2\left[(s + b + b_{\mathrm{reg}}) \ln\left(1 + \frac{s}{b + b_{\mathrm{reg}}}\right) - s\right]} \tag{5}$$

to optimise the selection region for discovery significance, where b_{reg} is a regularisation term suggested to be set to $b_{\mathrm{reg}} = 10$. Hence, our aim is simply to train a classifier g based on the training data $\mathcal{D}$ with the goal of maximising the AMS on some unseen test data.

3 Maximising the AMS

Having decided a statistical model for classification, its parameters will be tuned by maximising the AMS. This can be done in two ways.

- **Direct maximisation**: One can consider the AMS as an error/objective measure in a cross validation procedure. So the model will be tuned by a cross validation (cv) procedure by looking at the coefficients which directly maximise the AMS.
- **Two-stage maximisation**: Given a real-valued function f (e.g. a linear $f(\boldsymbol{w}, \boldsymbol{x}) = \boldsymbol{w}'\boldsymbol{x}$), one can always get a classification model by setting a threshold $\theta \in A \subset \mathbb{R}$ and a classifier

  $$h(\theta) := \mathrm{sgn}(f - \theta) \in \{-1, 1\} \tag{6}$$

  One can therefore train and tune f "independently" of the AMS, and then find the threshold θ for which the classifier $h(\theta)$ maximises the AMS. For example, one could train f in a logistic regression (i.e. train f s.t. it minimises a logistic loss), and then, on a second stage, choose the classification threshold based on AMS. This approach is presented in [3]

We will see that the two-stage maximisation is much more performant (in terms of AMS) than the direct one.

3.1 Two-stage maximisation

While the direct maximisation with cv can be seen as a usual tuning procedure by cross validation, the two-stage maximisation should be justified. This section is dedicated to the explanation of the two stage procedure for a classifier $h : \mathcal{X} \to \{-1, 1\}$, as defined in (6) , where $\mathcal{X}$ denotes the features space, and f **is trained by minimising a logistic loss**.

By incorporating the weights to the probabilities, without loss of generality one may write s and b from (3) and (4) as

$$s = s(h) := \mathbb{P}(h(\boldsymbol{X}) = 1, Y = 1) \ , \quad b = b(h) := \mathbb{P}(h(\boldsymbol{X}) = -1, Y = 1)$$

where $\boldsymbol{X}$ denotes a random imput vector and Y a random binary response. The paper justifies the two stage procedure explained above for logistic loss transformation of f, as follows. Given a classifier h obtained from f as in (6), one can define the"AMS regret" as

$$R_{AMS}(h) := \text{AMS}^2(h^*) - \text{AMS}^2(h) \tag{7}$$

where

$$h^* := \underset{h}{\text{argmax}} \ \text{AMS}^2(h).$$

Similarly, the"logistic regret" of f is defined as

$$R_{log}(f) := L(f) - L(f^*) \tag{8}$$

for the expected logistic loss transformation of f

$$L(f) := \mathbb{E}\Big[\log\Big(1 + e^{-Yf(\boldsymbol{X})}\Big)\Big] \tag{9}$$

and

$$f^* := \underset{f}{\text{argmin}} \ L(f)$$

The formal justification of the idea of

- training a logistic loss transformation of f, and
- finding the threshold θ in 6 by maximising the AMS on a second stage

comes from the following theorem.

Theorem 3.1. *Given a real-valued function f and the related classifier $h(\theta) := \text{sgn}(f - \theta)$, let $\hat{\theta} := \underset{\theta}{\text{argmax}} \ \text{AMS}(h)$. Then,*

$$R_{AMS}(h(\hat{\theta})) \leq \frac{s(h^*)}{b(h^*)} \sqrt{\frac{1}{2} R_{log}(f)} \tag{10}$$

Proof. See [3]-Theorem 2. □

That is, the AMS regret of classifier $h(\theta)$ is upper bounded by the logistic regret of the underlying f. It is possible to prove that it is sufficient to optimize θ on the empirical counterpart of AMS calculated on a separate validation sample.

Remark 3.2. *Training f by minimising a logistic loss is particularly suited for a classification task. For example, it allows us to train (on the first stage) models such as a logistic regression.*

Remark 3.3. *As far as logistic regression is concerned, recall that after model's coefficients $\boldsymbol{w}$ have been estimated, classification for a new observation y_0 can be done by setting $\hat{\boldsymbol{w}}'\boldsymbol{x}_0 > \theta$ for some suitable $\theta \in \mathbb{R}$. In a "proper" logistic regression, θ is directly set to 0. Here we choose it during a second stage via maximising AMS.*

Our aim is to tune a logistic regression. The theoretical set-up of logistic regression can be found below.

3.2 Logistic regression

Suppose $y \in \{-1, 1\}$. Note that

$$p(y=1|\mathbf{x}) = \frac{p(\mathbf{x}|y=1)p(y=1)}{p(\mathbf{x}|y=1)p(y=1) + p(\mathbf{x}|y=-1)p(y=-1))} = \frac{1}{1 + \frac{p(\mathbf{x}|y=-1)p(y=-1)}{p(\mathbf{x}|y=1)p(y=1)}} \tag{11}$$

Logistic regression is a discriminative learning model which aims at minimising an expected loss by predicting

$$\hat{y} = \text{argmin}_{y_0} \mathbb{E}_{p(y|\mathbf{x})}(L(y, y_0)|\mathbf{x})$$

and by *inferring* $p(y|\mathbf{x})$ *directly.*

Specifically, it models the log density ratio as

$$\log\Big(\frac{p(\mathbf{x}|y=-1)p(y=-1)}{p(\mathbf{x}|y=1)p(y=1)}\Big) \overset{model}{=} f(\mathbf{x}, \boldsymbol{\beta}) := \langle \boldsymbol{\beta}, \mathbf{x} \rangle + \beta_0$$

i.e. models the "log-odds" by a linear model (the term on the left of the proportionality symbol is added for intuition). In this way we can write

$$p(y|\mathbf{x}; \boldsymbol{\beta}) = \sigma(f(\mathbf{x}, \boldsymbol{\beta})y) \tag{12}$$

for a so-called "activation/logistic function" $\sigma(t) := \frac{1}{1+\exp(-t)}$.
ML estimates of $\boldsymbol{\beta}$ are computed from the log-likelihood as

$$\boldsymbol{\beta}_{ML} = \text{argmax}_{\boldsymbol{\beta}} \sum_{i \in D} \log \sigma(f(\mathbf{x}_i, \boldsymbol{\beta})y_i)$$

Remark 3.4. *Notice that*

$$\begin{aligned} argmax \sum_{i \in D} \log \sigma(f(\mathbf{x}_i, \boldsymbol{\beta})y_i) &= argmax \sum_{i \in D} -\log(1 + \exp\{-y_i \boldsymbol{\beta}' \mathbf{x}\}) \\ &= argmin \sum_{i \in D} \log(1 + \exp\{-y_i \boldsymbol{\beta}' \mathbf{x}\}) \end{aligned}$$

so that we are consistent with Kotlowsky paper by minimising a log loss as in equation 9 (of course, we are minimising its sample counterpart).

3.2.1 Weighted logistic regression

Remark 3.5. *In this dataset weights have an essential role in rebalancing the likelihood of signals and background events, as data have been artificially simulated. We will therefore fit the model with a weighted logistic regression.*

While unweighted logistic regression maximises

$$\hat{\boldsymbol{\beta}} = \text{argmax} \sum_{i \in D} -\log(1 + \exp\{-y_i \boldsymbol{\beta}' \mathbf{x}\})$$

weighted logistic regression takes into account weights w_i by inserting them in the log likelihood, as follows

$$\hat{\boldsymbol{\beta}} = \text{argmax} \sum_{i \in D} -\log(1 + \exp\{-w_i y_i \boldsymbol{\beta}' \mathbf{x}\}).$$

3.3 Generalising logistic regression's decision function

After ML estimates $\hat{\boldsymbol{\beta}}$ have been computed, classification of an observation $\mathbf{x}_0$ can be based on whether $p(y = 1|\mathbf{x}; \boldsymbol{\beta})$ is bigger than $p(y = -1|\mathbf{x}; \boldsymbol{\beta})$. That is,

$$\hat{y}_{\mathbf{x}_0} = 1 \text{ if } \sigma(f(\mathbf{x}_0, \hat{\boldsymbol{\beta}})) > \sigma(-f(\mathbf{x}_0, \hat{\boldsymbol{\beta}}))$$

$$\text{or, equivalently, } \hat{y}_{\mathbf{x}_0} = 1 \text{ if } \log\Big(\frac{\sigma(f(\mathbf{x}_0, \hat{\boldsymbol{\beta}}))}{\sigma(-f(\mathbf{x}_0, \hat{\boldsymbol{\beta}}))}\Big) > 0$$

We know that

$$\log\Big(\frac{\sigma(f(\mathbf{x}_0, \hat{\boldsymbol{\beta}}))}{\sigma(-f(\mathbf{x}_0, \hat{\boldsymbol{\beta}}))}\Big) \propto \langle \hat{\boldsymbol{\beta}}, \mathbf{x}_0 \rangle + \hat{\beta}_0$$

Therefore, classification from standard logistic regression can also be made as follows

$$\hat{y}_{\mathbf{x}_0} = 1 \text{ (resp. -1) if } \langle \hat{\boldsymbol{\beta}}, \mathbf{x}_0 \rangle + \hat{\beta}_0 > 0 \text{ (resp. } < 0).$$

Remark 3.6. *The two-stage procedure "generalises" this classification method by admitting that there is a $\theta \in \mathbb{R}$, possibly different from 0, such that the classification based on*

$$\hat{y}_{\mathbf{x}_0} = 1 \textit{ (resp. -1) if } \langle \hat{\boldsymbol{\beta}}, \mathbf{x}_0 \rangle + \hat{\beta}_0 > \theta \textit{ (resp. } < \theta).$$

could be "better" (for some purposes) than the standard logistic regression one.

4 Model fitting

4.1 AMS as direct measure - Cross Validation

As first attempt, a standard (i.e., $\theta = 0.5$) weighted logistic regression (WLR) has been fitted on the train set. Cross validation suggested that we can expect to get an AMS in the range [0.15, 0.41] on a test set. This is a relatively small value compared to results in following sections. Many attempts result in a AMS value larger than the upper bound found in cross-validation on test sets.

Using the entire set of training data to fit the WLR model, we get an AMS value of 0.18 on the test data set.

4.2 Two-stage maximisation

4.2.1 First stage

The training set $\mathcal{D} = \{(\boldsymbol{x}_1, y_1, w_1), ..., (\boldsymbol{x}_n, y_n, w_n)\}$ has been splitted into D_t (containing 80% of observations) and a validation set D_v, containing 20% of the observations. A weighted logistic regression has been fitted on D_t, thus obtaining its estimated coefficients $\hat{\boldsymbol{\beta}}$.

Remark 4.1. *Every time we subset the original training set, we must re-calibrate the weights such that the sum of selected background weights be equal to N_b and the sum of selected signal weights be equal to N_s . A specific function has been created for it.*

4.2.2 Tuning of threshold theta

The validation set D_v has been used to select the value of the threshold θ for which the AMS is maximised (on D_v). Uncertainty on AMS and on the best θ have been obtained by a cross validation procedure.

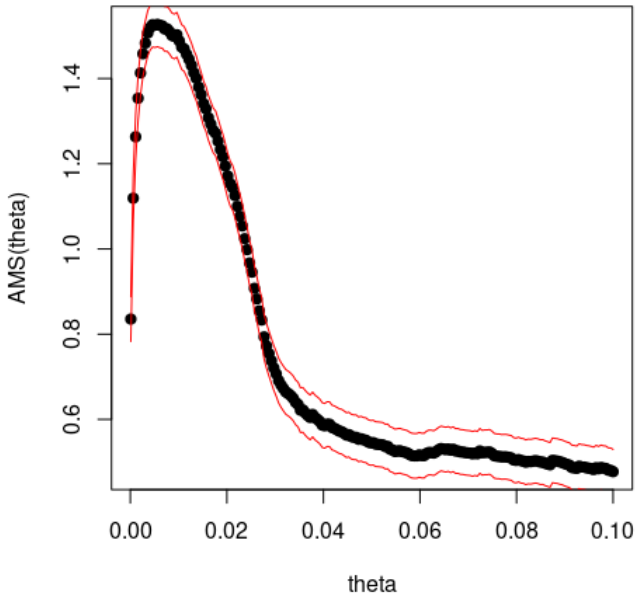

Figure 1: AMS values from CV procedure for different thresholds

Figure ?? shows that AMS is maximised for small values of θ (smaller than 0.02). The maximum value of AMS is around 1.45 - much higher than the one found with a standard logistic regression. A simple Monte carlo maximisation of AMS(θ) leads the optimal threshold $\hat{\theta}$, which is generally found to be between around 0.004.

The same procedure has beed tried after PCA. Analysis of variance explained by PC's suggested to try to use a small number of PC's 9see figure ??.

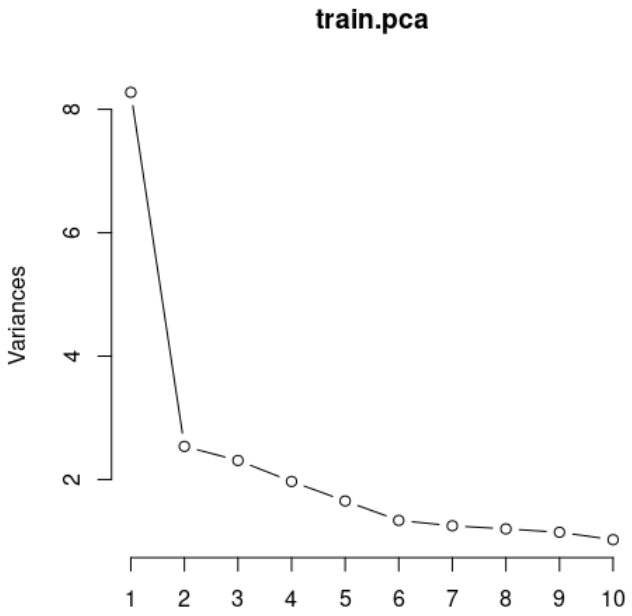

Figure 2: Screeplot

We decided to use the first 3 PC's as a new dataset, and repate the procedure below. Cross-validated AMS values are plotted in figure ??.

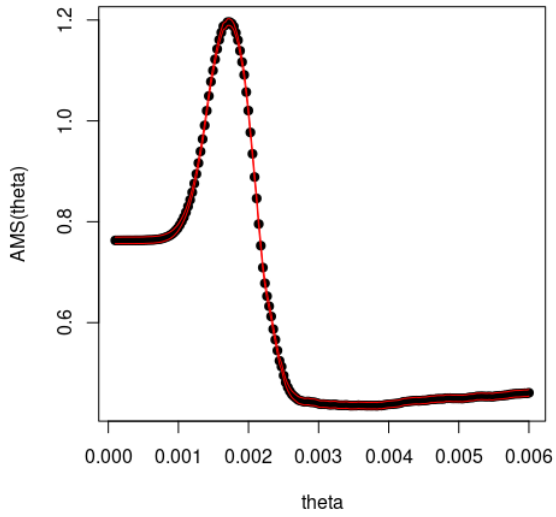

Figure 3: AMS values from CV procedure for different thresholds - on the first three pca's

One can see that the AMS on validation set on the PC's is much more "peaked". In particular, AMS variance is lower than before, and the range of suitable thetas is much shorter too. This might be a sign of overfitting. $\hat{\theta}_{pc}$ is almost always 0.0017.

4.3 Results on test set

It has been already said that a standard logistic regression on test set reaches an AMS of about 0.18, which is a very low performance.

On the contrary, the two stage procedure has a much better performance on test set. With the threshold theta suggested from the second stage procedure on all the features, AMS on test set is about 1.5, which is a big improvement. With the threshold suggested after PCA, we get an AMS of about 1.37. This is still much better than using a standard logistic regression classification rule, but smaller than for the threshold suggested by the original training set. This, as said before, may be connected to some overfitting - the range of suggested theta's was much shorter than before dimensional reduction.

There is something else suggesting that the two-stage procedure allows us to pick a threshold which is suited for the test set too. In figure ??, AMS values from predictions on test set are plotted for different thetas. It is clear that few values of theta lead to a high value of AMS, and among them we find the ones suggested from the second stage of our procedure.

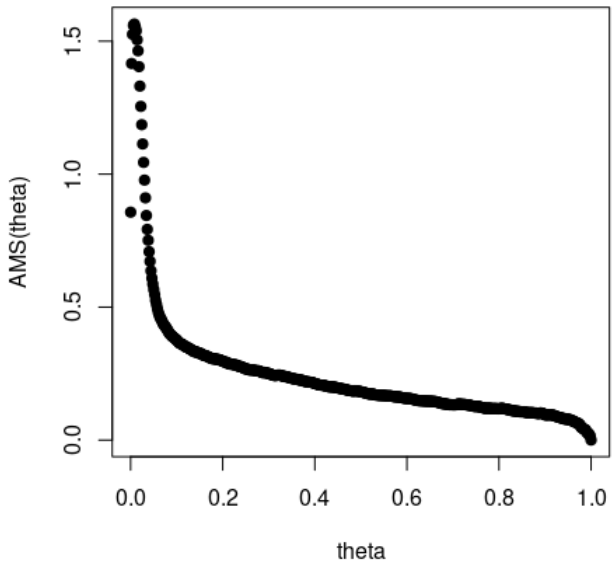

Figure 4: AMS values after predictions on test set - for different thetas.

4.4 Comments

References

[1] Evidence for Higgs Boson Decays to the $\tau^+\tau^-$ Final State with the ATLAS Detector. Technical report, CERN, Geneva, Nov 2013. All figures including auxiliary figures are available at https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/CONFNOTES/ATLAS-CONF-2013-108.

[2] Claire Adam-Bourdarios, Glen Cowan, Cecile Germain, Isabelle Guyon, Balazs Kegl, and David Rousseau. Learning to discover: the higgs boson machine learning challenge.

[3] Wojciech Kotlowski. Consistent optimization of ams by logistic loss minimization, 2014.