Department of Physics

Master's degree in Advanced Methods in Particle Physics

Multivariate classification of LHCb data

Selection of $B_S^0 \to \psi(2S)K_S^0$ *events*

Supervisors:
Dr. Luca Balzani
Dr. Marco Colonna
Dr. Luca Toscano

Lab report of:
Giulio Giamello
Cem Kurt

Summer Semester 2025

Contents

Introduction

Particle physics is at a point of a very statistical and gradual unravelling of the cosmic mystery that is our reality. Most of the processes that are of interest in our discipline occur within time and length scales that are practically impossible to record or observe. The strong and electroweak interactions that we are trying to understand can only be probed by the rigorous analysis of the final and initial states of the particles involved in these interactions. The strategy scientists decided to employ in order to study these fundamental interactions has evolved ever so slightly since the 50's, when CERN was founded. The true evolution occurred in our detector and computational capacity. Today, in Large Hadron Collider (LHC), we are colliding protons with 13 TeV center-of-mass energies at 40×10^{16} times per second. Such high energies and high rate of collisions unlock the possibility of analyzing rarer processes or those that are suppressed at lower energies. This increased capacity comes with a caviat; the perfect physics breeding ground we have at our collision point results in huge amounts of physical processes to occur per collision and accordingly, massive amount of data is registered for each collision event.

Our purpose built detectors manage to record the aftermath or the final states of the fundamental particle interactions very well; however, the flipside of the coin, the massive data that we collect needs to be analyzed with many layers of similarly purposefully designed hardware systems, software systems and algorithms. Usually, we are interested in figuring out the details of a specific process. The detector measurement for any process in LHC is buried under every other process that happens at the same event, therefore we need to come up with a strategy to select only the signal for the process we want to study. In this report, we are going to use the aid of the relatively high technological advance of computers to build a machine learning algorithm (Multi-Variate Classifier) that will be able extract the signal for the process we are looking for from the dataset we have from one of the four detectors at the LHC, the LHC-beauty detector.

Chapter 1

Theoretical Background

1.1 LHCb

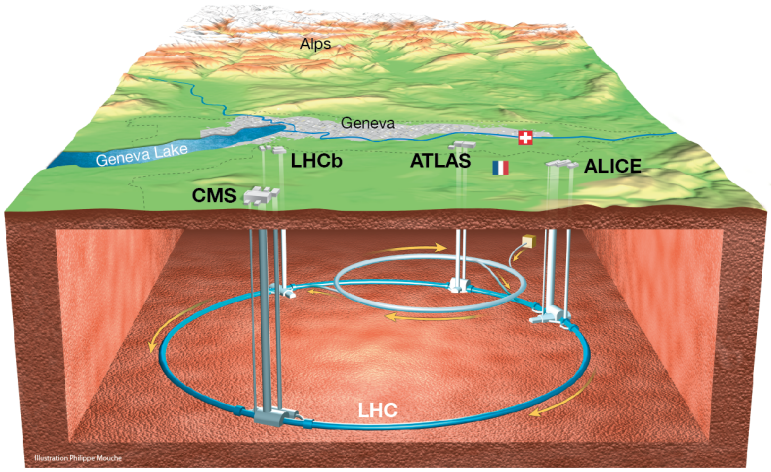


Figure 1.1: LHC schema with ALICE, Atlas, CMS and LHCb at CERN.

The LHCb Detector is part of one of the four large experiments set up on different points of the circular accelerator design of LHC, at CERN. Our detectors are constructed around collision points where the accelerated beams of protons are crossed with each other 40×10^{16} times per second. Unlike rest of the detectors at LHC, the LHCb detector is built to detect particles from a singular direction, with the official name for the design being, "Single arm forward spectrometer". It can detect particles coming from the interaction point in the pseudorapidity

range between $2 < \eta < 5$ and it is design philosophy is geared towards measuring b (bottom) and c (charm) hadron properties alongside CP violation. [**?**]

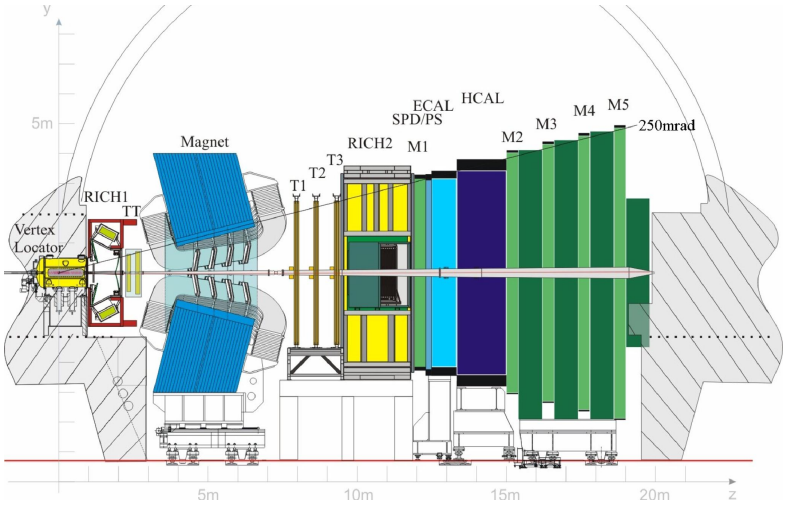


Figure 1.2: The LHCb Detector.

(Detector elements from left to right: I-Vertex locator, II-Ring Imaging Cherenkov Detector 1, III-Trigger Tracker(TT), IV-Dipole Magnet, V-Outer Trackers(T1,T2 and T3), VI-Ring Imaging Cherenkov Detector 2, VII-Muon Detector 1, VIII-Electromagnetic and Hadronic Calorimeters, IX-Muon Detector 2,3,4 and 5).

The LHCb detector is made up of the following components:

1. **VELO (Vertex Locator)**

   Purpose: Precise reconstruction of primary and secondary vertices coming from the collision.

   Features: Silicon-strip detectors close to the beam line, retractable during the beam injection phase.

2. **TT Station (Trigger Tracker)**

   Purpose: Tracking particles before entering the magnetic field.

   Features: Silicon microstrip detector upstream of the magnet.

3. **Dipole Magnet**

   Purpose: Bends charged particles, making it possible to measure their momentum later.

   Features: Approximately 4 T·m integrated Magnetic field special to the forward spectrometer design.

4. **Tracking Stations (T1,T2 and T3)**

   Purpose: Tracking particles after leaving the magnetic field (Tracking charged particles for momentum measurement.).

   Features: Straw tube drift chambers that cover a large radius.

5. **RICH Detectors (Ring-Imaging Cherenkov Detectors)**

   Purpose: Particle identification (separating pions, kaons, protons).

   Features: Based on the Cherenkov effect, can measure the velocity of the passing particles by recording the angle of which their Cherenkov radiation is received. The equation for the angle in terms of the velocity of the radiating particle is:

$$\cos(\theta_C) = \frac{1}{n\frac{v}{c}} = \frac{c}{nv}$$

   where the "n" is the refractive index of the detector material.

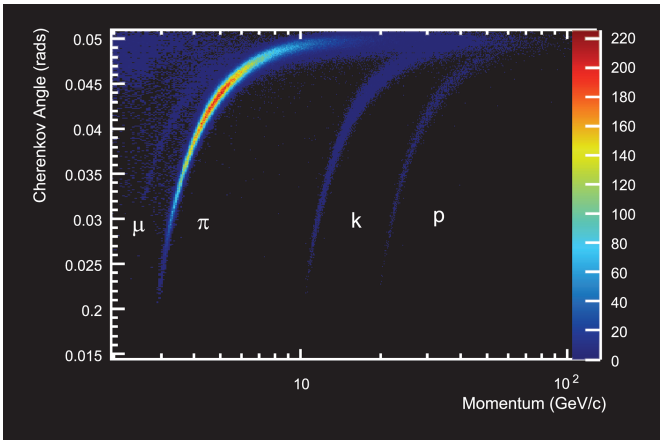


Figure 1.3: Possible particles and their momentum via a given Cherenkov angle.

6. **Calorimeters**

   Purpose: Measuring the energy of electrons, photons, and hadrons via particle shower.

   Features:

   - **Scintillating Pad Detector:** Electron/photon Identification.
   - **Preshower Detector:** Distinguishes electrons from hadrons.
   - **Electromagnetic Calorimeter:** Measures energy of electrons and photons.

- **Hadronic Calorimeter:** Measures energy of hadrons (neutral hadrons like neutrons are primarily stopped here).

7. Muon System

Purpose: Muon identification.

Features: Five stations (M1–M5) using Multi-wire proportional chambers and gas-electron multipliers. The M1 station is before the calorimeters and M2 to M5 after, with shielding in between.

1.2 Isolating the B_s^0 Decay

In this lab, we are interested in analysing the specific decay $B_s^0 \to \psi(2S)\, K_S^0$.

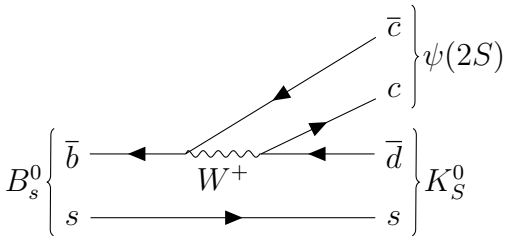


Figure 1.4: Leading order feynman diagram for the decay $B_s^0 \to \psi(2S) K_S^0$.

This decay is interesting because of some CP sensitive properties. However, we also have to contend with the signal of the similar, but more abundant decay of $B_d^0 \to \psi(2S)\, K_S^0$ in our dataset.

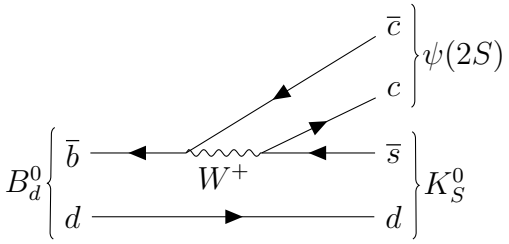


Figure 1.5: Leading order feynman diagram for the decay $B_d^0 \to \psi(2S) K_S^0$.

The data we receive, which has already been through a preliminary (however pretty intensive) selection process, is a reconstruction of the initial state of B_S^0 meson from the candidates of final state from the decay: $B_s^0 \to \psi(2S)\, K_S^0 \to (\mu^-\mu^+)(\pi^-\pi^+)$.

Our task is to isolate the signal of the B_s^0 decay from a recorded dataset and,in the following, we are going to explain how we trained a classifier to do that; so we need here to introduce and explain some theoretical concepts.

1.2.1 Figure of merit and significance

During the training procedure of the classifier we need to evaluate and optimize the performances (maximizing the number of signal candidates while keeping the background low). In order to increase the performances we need to minimize the loss function: a function that measure the distance between the target value and the one reconstructed by the classifier.

The inverse of such loss functions is called "Figure of merit" (FOM). In this analysis, we use the Punzi FOM test to calculate the significance of the signal that comes out of our multivariate classifier:

$$\mathrm{FOM} = \frac{\epsilon_{\mathrm{sig}}}{\frac{5}{2} + \sqrt{N_{\mathrm{bkg}}}} \tag{1.1}$$

where N_{bkg} is the number of background events in the signal region and ϵ_{sig} is the classification efficiency of the signal.

After the optimal cut has been found, we need to evaluate the number of signal events in the data sample. To this end, we model the invariant mass distribution of the $\psi(2S)K_S$ combination with two peaks (B_s^0 and B_d^0 events) and a decreasing exponential function for the remaining combinatorial background. By fitting this model to the data distribution, the number of signal candidates can be obtained from the integral of the signal peak model.

From here, we can also compute an estimated significance of the observation:

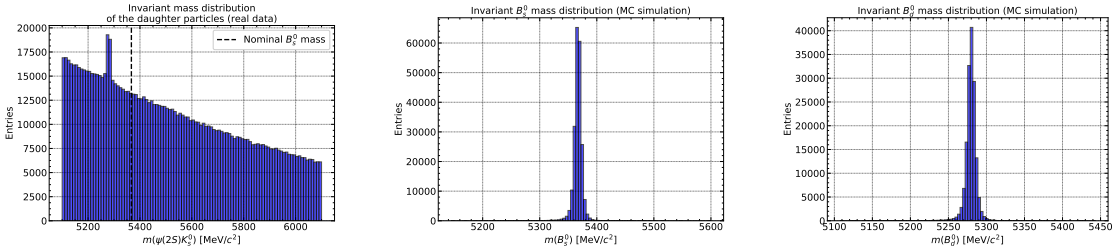
$$m = \frac{N_{\mathrm{sig}}}{\sqrt{N_{\mathrm{sig}} + N_{\mathrm{bkg}}}} \tag{1.2}$$

Chapter 2

Data Analysis

Before starting our analysis of the real LHCb data, it is good practice to test and improve our analysis strategy on simulated data. This preliminary procedure is actually a common practice. In a simulation, we work under the assumption of a perfect detector with no background (only signal events) while the analysis of real events is more challenging.

2.1 Real and simulated data



(a) LHCb data of reconstructed invariant mass.

(b) Signal simulation for B_s^0 decay.

(c) Signal simulation for B_d^0 decay.

Figure 2.1

For the analysis of $B_s^0 \to \psi(2S)K_S$, we are going to use three data samples:

- Real data collected from the LHCb;
- Monte-Carlo simulation of the signal decay $B_s^0 \to \psi(2S)K_S^0$;
- Monte-Carlo simulation of the signal decay $B_d^0 \to \psi(2S)K_S^0$, (kinematically similar decay).

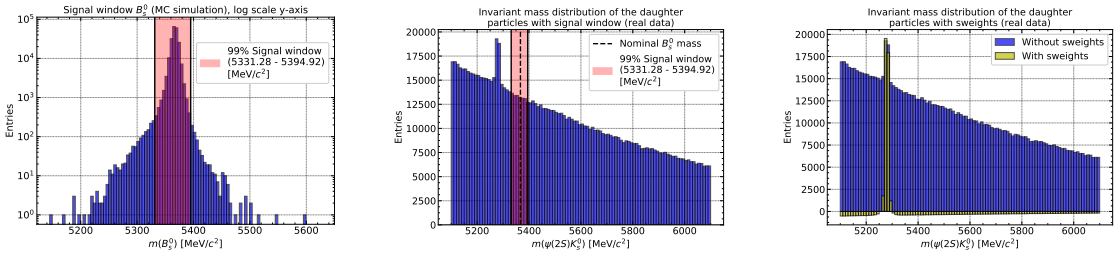
Of which the LHCb data has already went through heavy pre-selection via trigger, track reconstruction, etc. while the distributions and the peaks of B_s^0 and B_d^0 simulation are centered around the known meson masses: 5366.93 [MeV/c^2] for B_s^0 and 5279.72 [MeV/c^2] for B_d^0 (from the PDG [**?**]) as seen in **??**.

2.2 Strategy

The event datasets collected at the LHCb are put through a preliminary filter working synchronously with the data collection via hardware and software triggers. This helps us eliminate the majority of the unwanted events where analysis would have diminishing returns. After this elimination, we utilize layers of Particle Identification systems (PID) that may run both synchronously and asynchronously to the data taking. The PID systems are a vital analysis tool for reconstructing the actual process we are looking for within the amassed data. In this lab, we are going to only be interested in the steps after these procedures. We will construct a machine learning algorithm, a multi-variate classifier, that will aid us in the latter steps of isolating the signal for the decay process we are interested in.

In the dataset we receive from the LHCb, we have 863 variables logged per each event shown in **??**. These variables are also called "features". We are going to try to identify a few that will train a useful algorithm. The usefulness lies in the ability of discrimination of subtle differences between possible background and signal events as well as transferability from simulation to real data. The variables that are redundant, too dominatingly distant due to training bias or too indifferent between signal and background data will be discarded. We are going to train our multi-variate classifier on the purely background invariant mass range and the sPlot (statistical fit) isolated B_d^0 signal, so that we can apply the classification to the full range mass data and extract the mass of B_s^0 by doing a mass fit alongside it. Therefore completing the analysis of the B_s^0 decay.

2.3 Signal window and background upper sideband



(a) Signal window on the simulated B_s^0 data.

(b) Signal window on the LHCb real data.

(c) Invariant mass distribution of LHCb real data with sWeights.

Figure 2.2

We need to exhibit the problem step by step. First things first, we are going to extract some information from the invariant mass signal distributions for the simulations of B_s^0 as well as B_d^0 (**??** and **??**). The mass distribution of B_s^0 decay is only a short interval within the LHCb data. In order to isolate it, we define the signal window. We construct an interval centered around the peak with a lower and higher boundary on the invariant mass. We extend the boundaries until 99% of our simulated B_s^0 signal events fall inside as can be seen in **??** (Approximately 3.5 standart deviations of the distribution). As for the B_d^0 simulation data, we are going to complement it using sPlot to extract sWeights from the LHCb data and get a fit from real data as can be seen in **??**.

For the background events to be used as non-signal label in the training of the classifier algorithm, we are going to make a cut at the higher limit of the B_s^0 simulated signal band and use the upper side band interval where there are no resonances. We will call this interval the "upper side-band background" **??**.

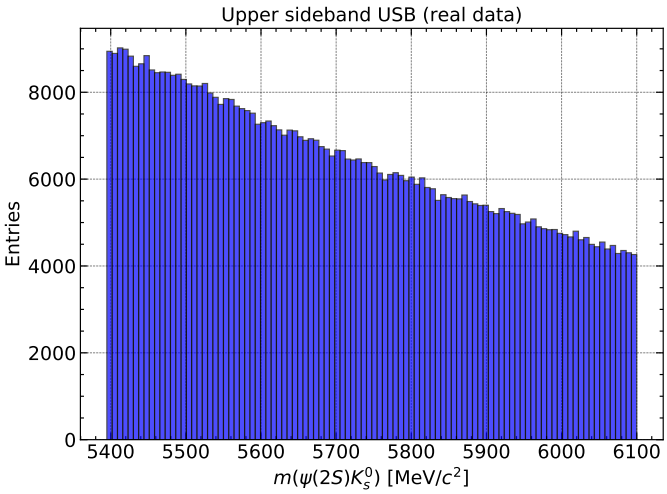


Figure 2.3: Upper Side-Band Background Region.

2.4 Feature Selection

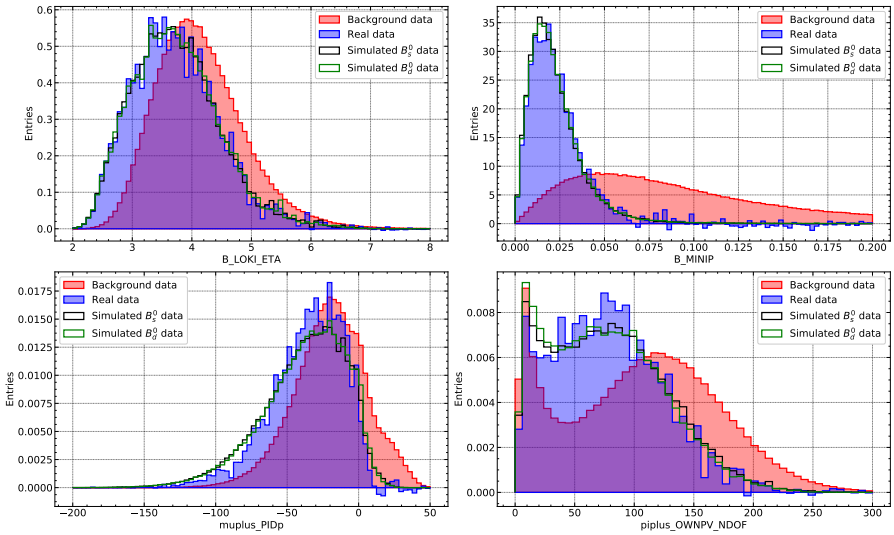


Figure 2.4: Comparison of 4 feature distributions over the different datasets we have.

In order to start the training of a multi-variate classifier, we need to pick a number of data *features* out of the 863 that will be used for the training of the machine learning algorithm. First of all, we identify the best features to use in the analysis in three steps.

First step, a selection is made by comparing each feature from the sWeights applied real signal of B_d^0 (blue colour in **??**) with the ones from the simulation of B_d^0 (green colour). We utilize the Kolmogorov–Smirnov test to measure the distance between the same features on different datasets and making a cut to keep only the features that fall below a certain distance threshold. Choosing features with minimal distance selects only those with the most similar feature distribution between simulation and real data, so it gives us an idea of how good the simulation is in reproducing the real processes.

The Kolmogorov-Smirnov test [**?**] keeps the largest distance between the cumulative probability distributions F^i:

$$D_n = \sup_x \left| F_n^1(x) - F_n^2(x) \right| \tag{2.1}$$

where n runs over all bins of the histograms.

Second step, in feature selection, we compare the simulation of B_d^0(green colour) and the background data (red colour) to identify this time the maximal distance between the features and do another selection. In this case we want our model to be able to distinguish and classify signal over background events.

Finally, we have a set of variables that is well simulated and provides good discrimination between signal and background. Last step is to check the correlation of the selected features and discard the ones with an high correlation coefficient in order to avoid biasing our model to the B_d^0 simulation.

In Figure **??**, we plotted the correlation value for each feature on a colour scale with high colour intensity that corresponds to high correlation.

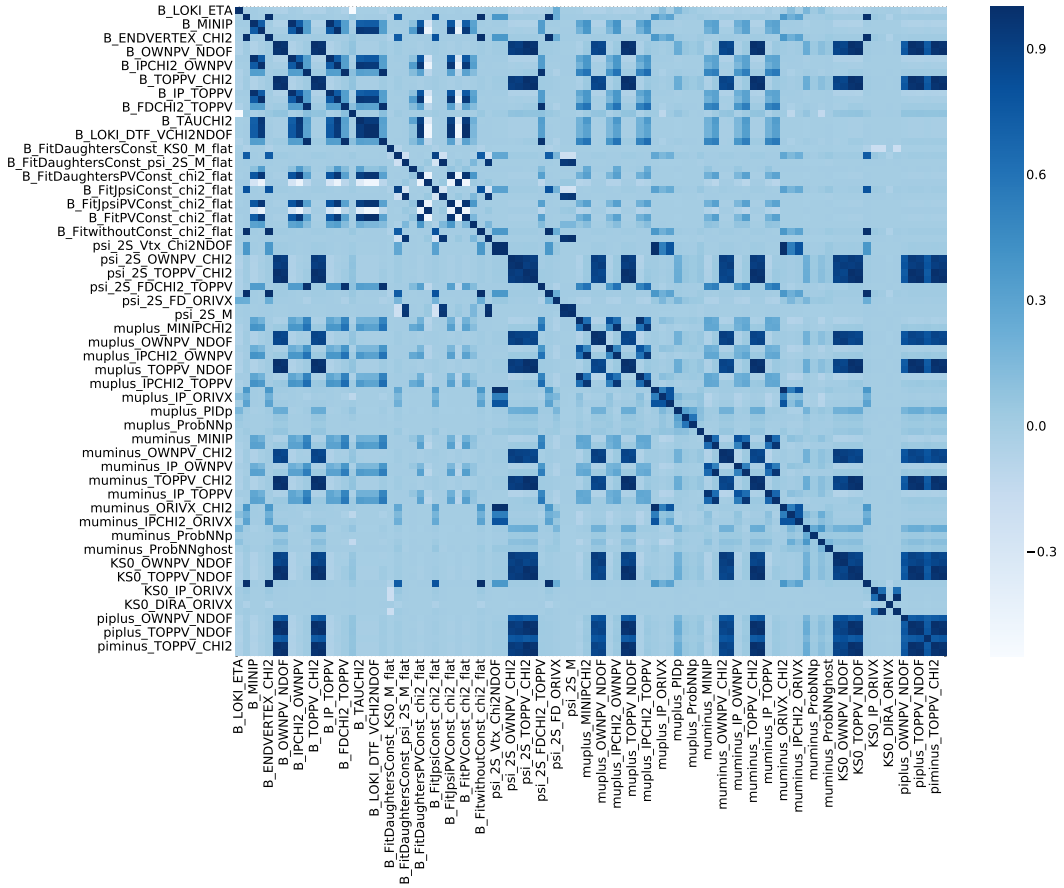


Figure 2.5: Correlation between different features

2.5 Model training and multivariate classification

The multivariate classifier learner we chose to use for this analysis is a gradient boosted decision tree in the `XGBoost` package [**?**]. A Boosted Decision Tree (BDT) is a powerful machine learning method that builds an ensemble of decision trees in a sequential manner, where each new tree is trained to correct the errors made by the previous ones. The `XGBoost` (Extreme Gradient Boosting) package is a widely-used implementation of gradient-boosted decision trees, especially in classification tasks.

Since the background training dataset is much larger than the signal training dataset, the background data will be weighted so that the classifier can be trained

equally on signal and background samples:

$$\text{weight} = \frac{\#\text{signal}}{\#\text{background}}$$

#signal	#background	weight
211314	643816	0.3282

2.5.1 Boosted Decision Tree

For the classification, we train five BDTs on a 5-fold cross validation: this procedure consists of splitting the data into k subsamples and use $k-1$ of those for the training ad the other one for the testing; and then repeating for k times. Cross validation is used to avoid *overfitting*, indeed no one specific subsample is used for final validation of the classifier.

To measure the performance of a BDT we look at quantities like the true positive rate (TPR) and the false positive rate (FPR), meaning those events that the model has failed to classify correctly in the testing procedure. Our aim is to train the model to be as precise as possible and so, to minimize this the FPR while keeping the TPR high.

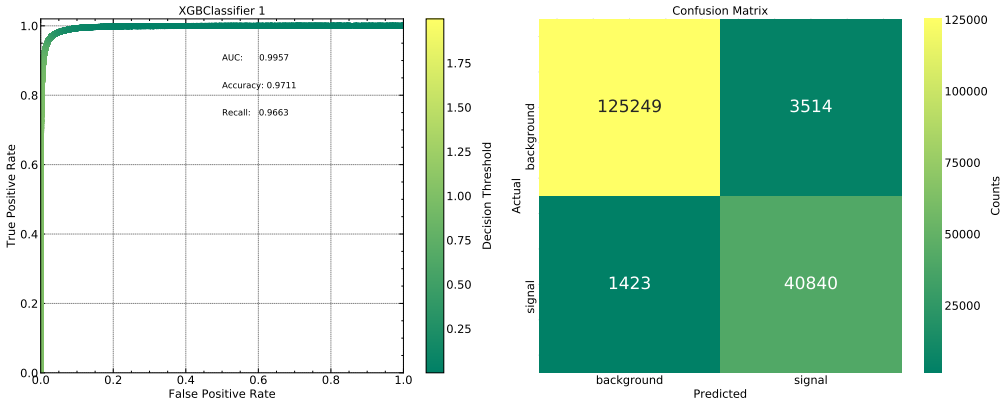


Figure 2.6: ROC curve of the first trained XBG Classifier (*left*); Confusion matrix of the first trained XBG Classifier (*right*).

When plotted together to a normalized value of 1, the TPR and FPR make up the "Receiver Operator Characteristics (ROC)" curve and the area under the curve (AUC) ranks the accuracy of the learner from 0 to 1; 1 being a perfect score [**?**]. An example of the ROC results from the first of 5 BDT's is shown in **??**.

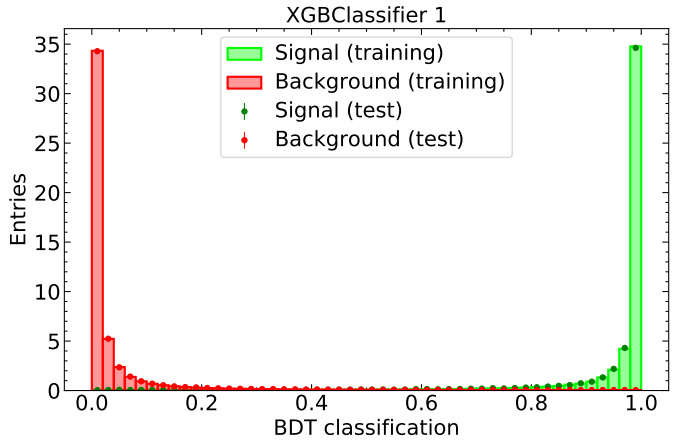


Figure 2.7: Accuracy test of the first iteration of BDT classifier.

To check for overtraining, the output of the first classifier is histogramed on training and test data **??**. The agreement of the histograms columns and the points is a good sign that no overtraining has occurred.

2.5.2 Classification threshold

Once we complete the training and testing procedure with the 5-fold cross validation, we have to find out a suitable *threshold* value (between 0 and 1).

We use the Punzi figure of merit (FOM), **??**: for varying threshold values, the signal efficiency and number of background events in the signal window is computed. In particular, the number of background events in the signal window is computed by fitting an exponential background model to the upper sideband and extrapolating the number of events in the desired signal region. We repeat this for all five BDTs and then we compute the best threshold cut, taking the average of the threshold of each BDT.

An entry beyond this best threshold will be considered as a signal in our classifier.

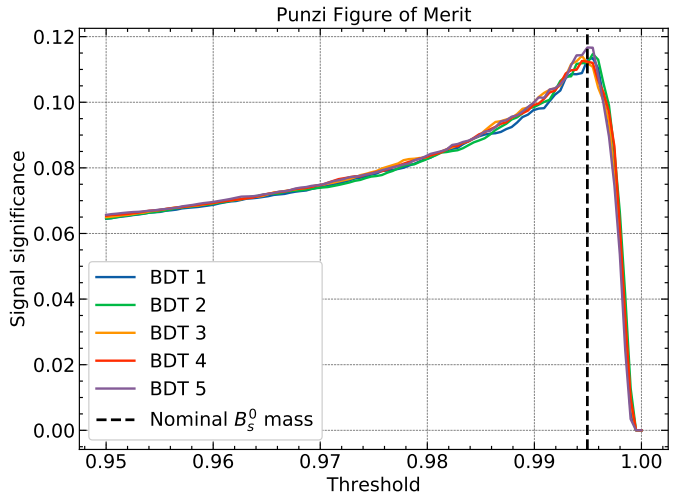


Figure 2.8: Punzi FOM results for all 5-fold BDT.

For our classifier, the best threshold happens to be 0.995 (**??**).

2.6 Invariant mass fit

We apply the best threshold cut to the data predicted by the BDT models, obtaining the distribution plotted in **??**:

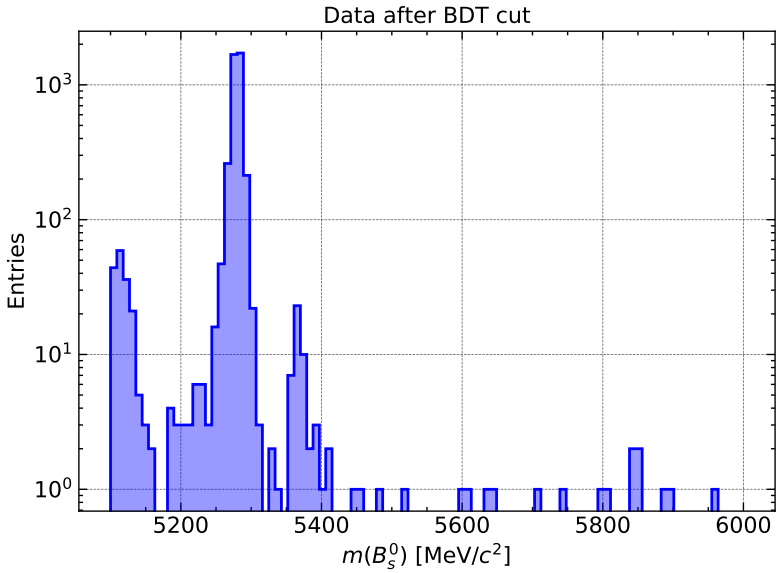


Figure 2.9: BDT real data invariant mass distribution after the threshold cut.

Here we have to notice that in the distribution, obtained from the classifier, we can (still) identify the B_d^0 peak and now the B_s^0 peak too.

Now, we have to find out what signal and background yield we are getting here. Therefore, we need to model the entire distribution: our model consist of two Gaussians (for each of the two mass peaks) and one exponential background component. The two Gaussian peaks are defined as:

$$G(m) = \frac{r}{\sqrt{2\pi}\sigma_1} \exp\left[-\frac{(m-\mu_1)^2}{2\sigma_1^2}\right] + \frac{(1-r)}{\sqrt{2\pi}\sigma_2} \exp\left[-\frac{(m-\mu_2)^2}{2\sigma_2^2}\right] \tag{2.2}$$

where μ_i and σ_i are the mean and width of the two Gaussians, r is the ratio of the two distributions in the model.

So the full-fit model is defined as

$$F_{\text{full}}(m) = \alpha \cdot G_{B_s^0}(m) + \beta \cdot G_{B_d^0}(m) + \gamma \cdot e^{-\delta \cdot m} \tag{2.3}$$

where $G_{B_s^0}(m)$ and $G_{B_d^0}(m)$ are the Gaussian model used to fit the two B_s^0 and B_d^0 distributions and α, β, γ, δ are free parameter of the model, to be determined.

In order to build a stable model, we firstly (pre-)fit the peaks to their respective simulation distribution so that we fix the mean and width of both signal samples, respectively. (To ensure that the peaks in the simulation match the peak shape in the data, we have classified both signal samples and applied the BDT cut before performing the fit).

After that, the only free parameters of the final fit are the background yield, the B_s^0 and B_d^0 signal yield, and the slope of the exponential combinatorial background. The plot of the full-fit is in **??**:

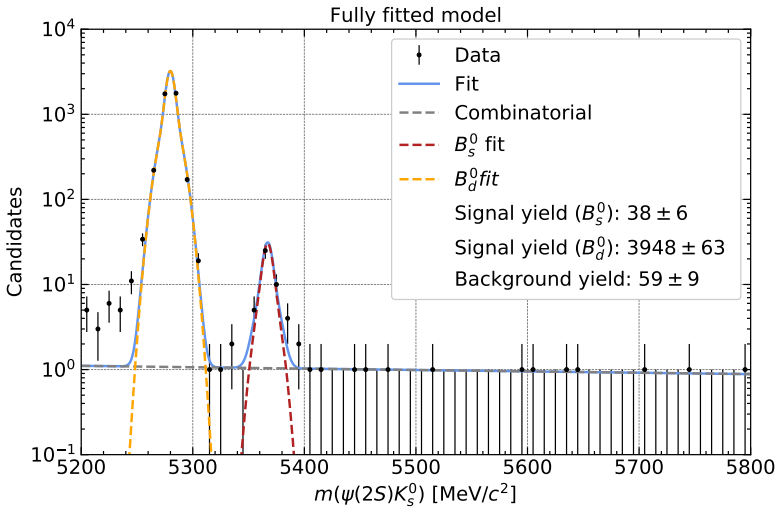


Figure 2.10: Fit of the BDT classified data on the invariant mass range for reconstructed final products of B_s^0 decay.

Finally, with the parameters obtained from the fit, we can obtain the number of signal and background event, needed to compute **??**, integrating over the B_s^0 peak and the exponential background.

The obtained significance is:

$$m = 5.68\sigma \tag{2.4}$$

Conclusions

In this lab, we investigated the rare decay $B_s^0 \to \psi(2S)K_S^0$, which is experimentally challenging due to its low yield in the presence of significant background, including a nearby peak from the more abundant $B_d^0 \to \psi(2S)K_S^0$ decay. To conduct the search, we employed a multivariate classifier based on a Boosted Decision Tree (BDT) that exploits the discriminating power of multiple kinematic and topological variables. We trained our classifier using select variables (also called features) specifically chosen to be unbiased, well simulated and well discriminating. The classifier effectively suppressed combinatorial and partially reconstructed background while retaining our signal. This approach enabled a clearer separation between the B_s^0 and B_d^0 components in the invariant mass distribution, massively improving the signal-to-background ratio. The use of machine learning proved crucial in isolating this suppressed decay. This result suggests that there is a good chance that the observed B_s^0 particle and not to a statistical fluctuation.