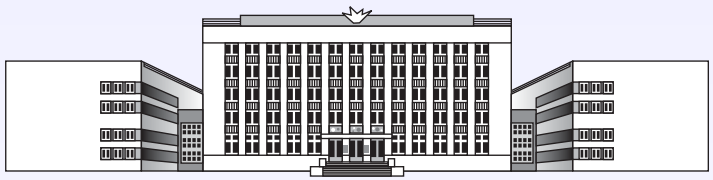
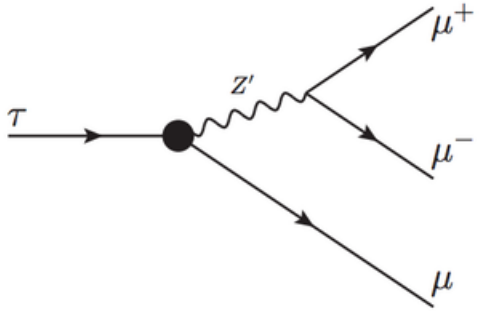
“Flavours of Physics” challenge: 2nd place solution

Alexander V. Gramolin
(Budker Institute of Nuclear Physics, Novosibirsk, Russia)



ALEPH Workshop @ NIPS 2015
Montreal, Canada
December 11, 2015

"Flavours of Physics" machine learning challenge



- Website: www.kaggle.com/c/flavours-of-physics
- Predict the probability that a given event is the $\tau^{\pm} \to \mu^{\pm}\mu^{+}\mu^{-}$ decay

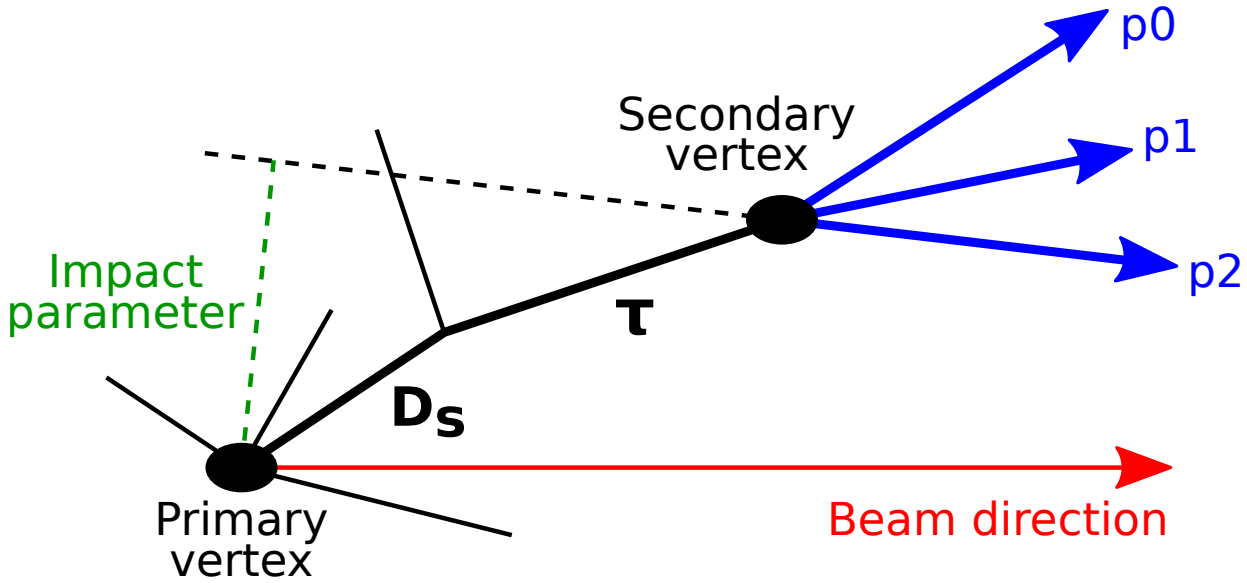
- Three types of events:

Signal (simulated):	**Control** (simulated + real):	**Background** (real):
$\tau^- \to \mu^-\mu^+\mu^-$	$D_s^- \to \phi(\mu^+\mu^-)\pi^-$	$D_s^- \to \eta(\mu^+\mu^-\gamma)\mu^-\bar{\nu}_\mu$
$m_\tau = 1776.82$ MeV	$m_{D_s} = 1968.30$ MeV	$D_{(s)}^- \to K^+\pi^-\pi^-$
		$D_{(s)}^- \to \pi^+\pi^-\pi^-$

- Four datasets were provided:
  - training.csv (67 553 labeled events, 41 674 signal + 25 879 background)
  - test.csv (855 819 unlabeled events, signal + control + background)
  - check_agreement.csv (331 147 control events)
    – to check that the classifier is not exploiting the imperfections of simulation
  - check_correlation.csv (5 514 background events)
    – to check that the prediction is not too correlated with the τ mass

Original features

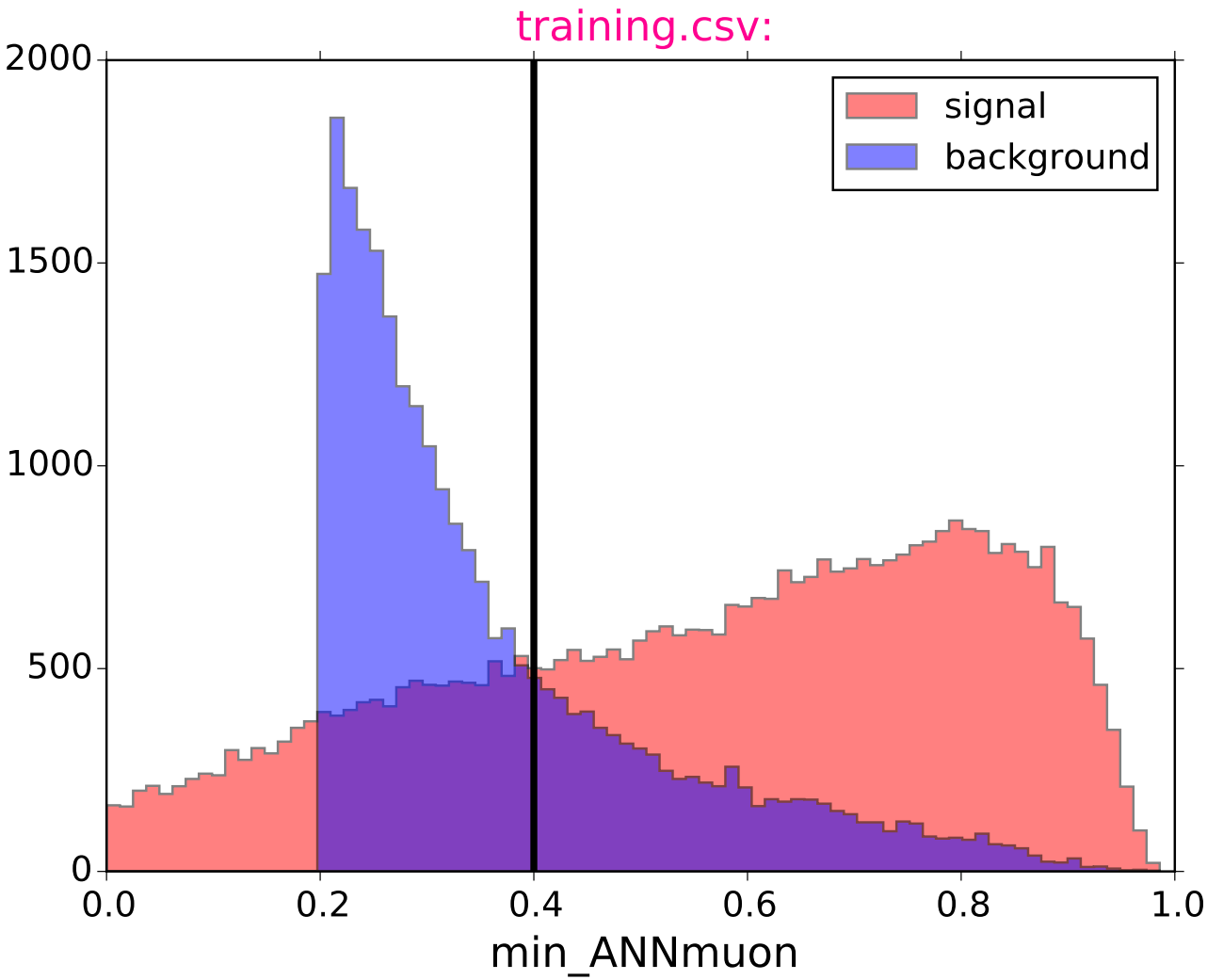
- 46 primary features:
  - 10 kinematic features (momenta, transverse momenta, and pseudorapidities of p0, p1, and p2; transverse momentum of the mother particle)
  - 35 "geometric" features (impact parameters, track isolation variables, flight distance and lifetime of the mother particle, etc.)
  - SPDhits (number of hits in the Scintillating Pad Detector, SPD)



- 3 auxiliary features (absent in the test dataset):
  - production (τ production mechanism: $D_s^- \to \tau$, $D^- \to \tau$, or $X_b \to \tau$)
  - min_ANNmuon (muon identification variable)
  - mass (mass of the mother particle)

The “min_ANNmuon” feature

Muon identification variable obtained using an artificial neural network (ANN)

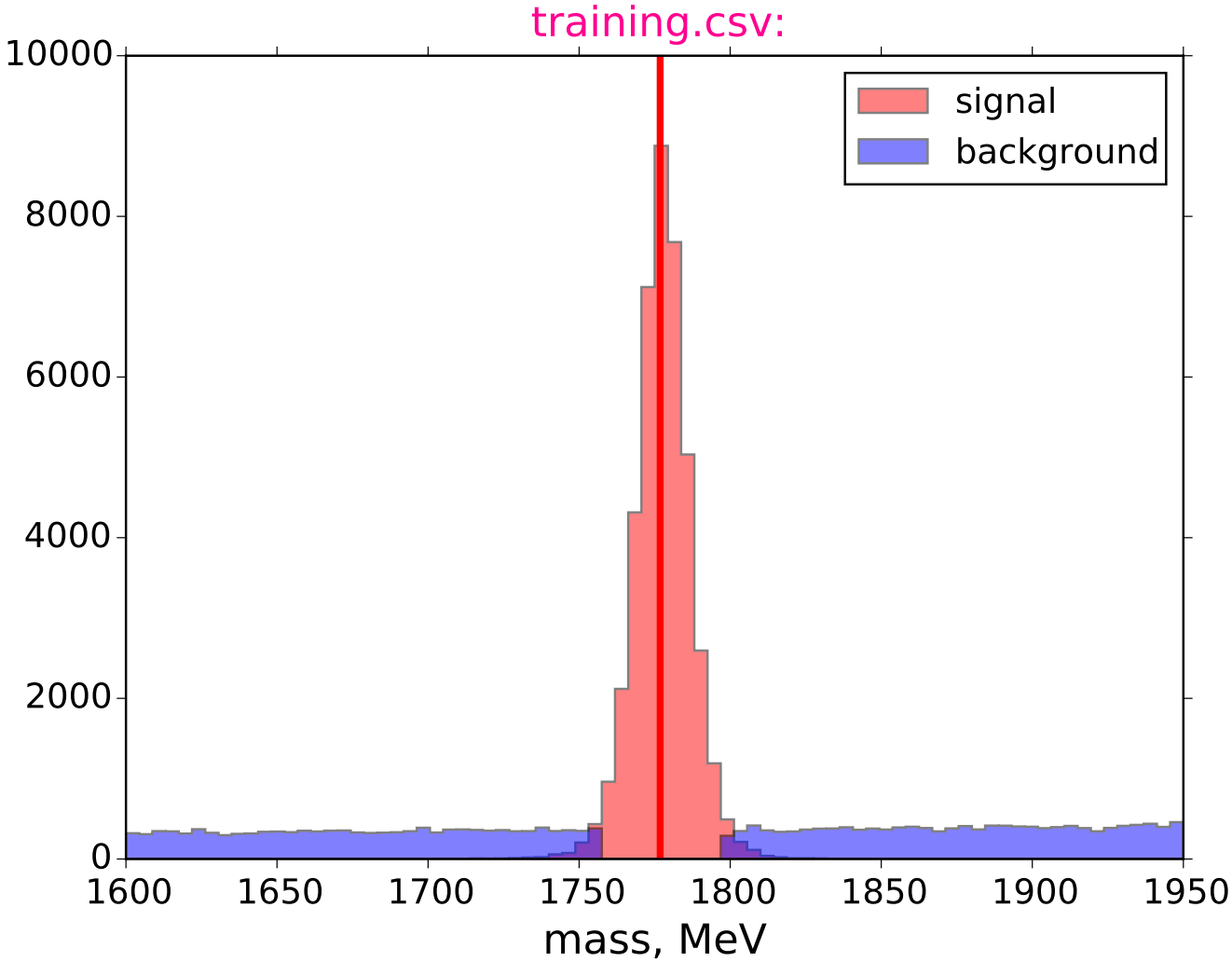


Only events with min_ANNmuon > 0.4 are used to score a model

$\Rightarrow$ use only these events for training (30 213 signal + 7 799 background events)

The "mass" feature

The real events collected by LHCb were used as background. To make sure that there are no $\tau \to 3\mu$ decays among them, events were removed from the τ mass window.



$\Rightarrow$ mass is a "golden feature" that can separate signal and background almost perfectly!

How to reconstruct the mass: the first way

1. Calculate the longitudinal momenta of p0, p1, and p2:

$$\mathbf{p0\_z} = \mathrm{p0\_pt} \cdot \sinh(\mathrm{p0\_eta}), \quad \mathbf{p1\_z} = \mathrm{p1\_pt} \cdot \sinh(\mathrm{p1\_eta}), \quad \mathbf{p2\_z} = \mathrm{p2\_pt} \cdot \sinh(\mathrm{p2\_eta}).$$

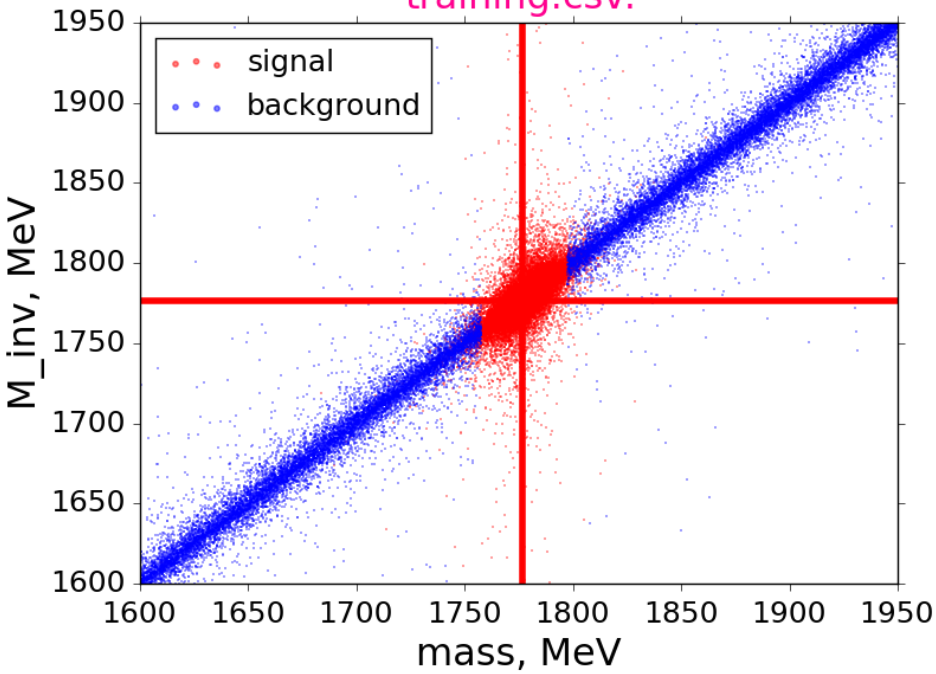
2. Assume that p0, p1, and p2 are muons and calculate their full energies:

$$\mathbf{E0} = \sqrt{m_\mu^2 + \mathrm{p0\_p}^2}, \quad \mathbf{E1} = \sqrt{m_\mu^2 + \mathrm{p1\_p}^2}, \quad \mathbf{E2} = \sqrt{m_\mu^2 + \mathrm{p2\_p}^2}, \quad m_\mu = 105.66\ \mathrm{MeV}.$$

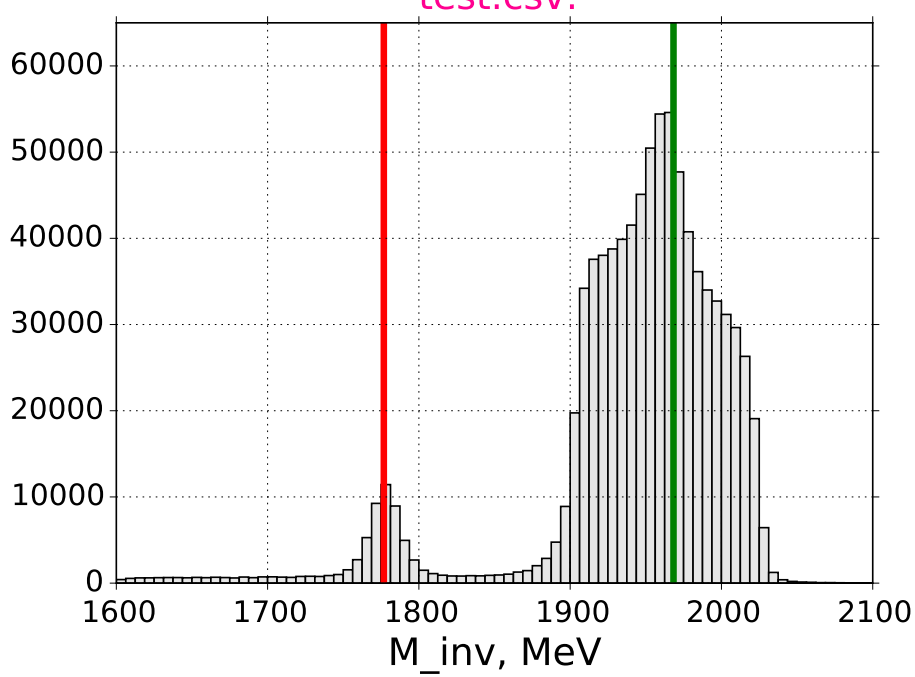
3. Calculate the kinematic parameters of the mother particle:

$$\mathbf{E} = \mathrm{E0} + \mathrm{E1} + \mathrm{E2}, \quad \mathbf{pz} = \mathrm{p0\_z} + \mathrm{p1\_z} + \mathrm{p2\_z}, \quad \boxed{\mathbf{M\_inv} = \sqrt{\mathrm{E}^2 - (\mathrm{pt}^2 + \mathrm{pz}^2)}}.$$

training.csv:



test.csv:



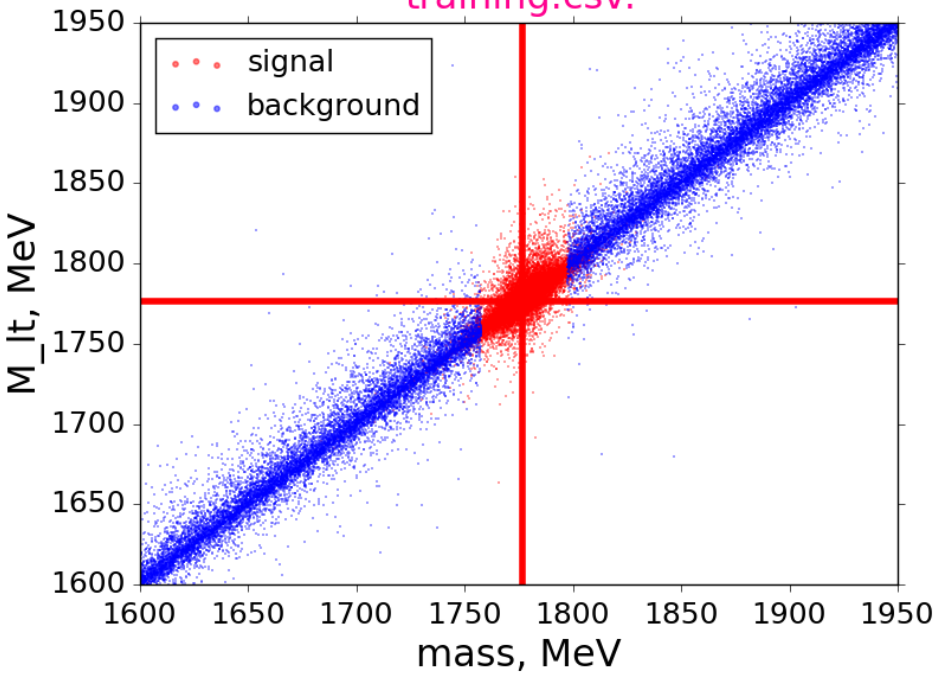
How to reconstruct the mass: the second way

Alternatively, the mass of the mother particle can be calculated from LifeTime and FlightDistance using special relativity:

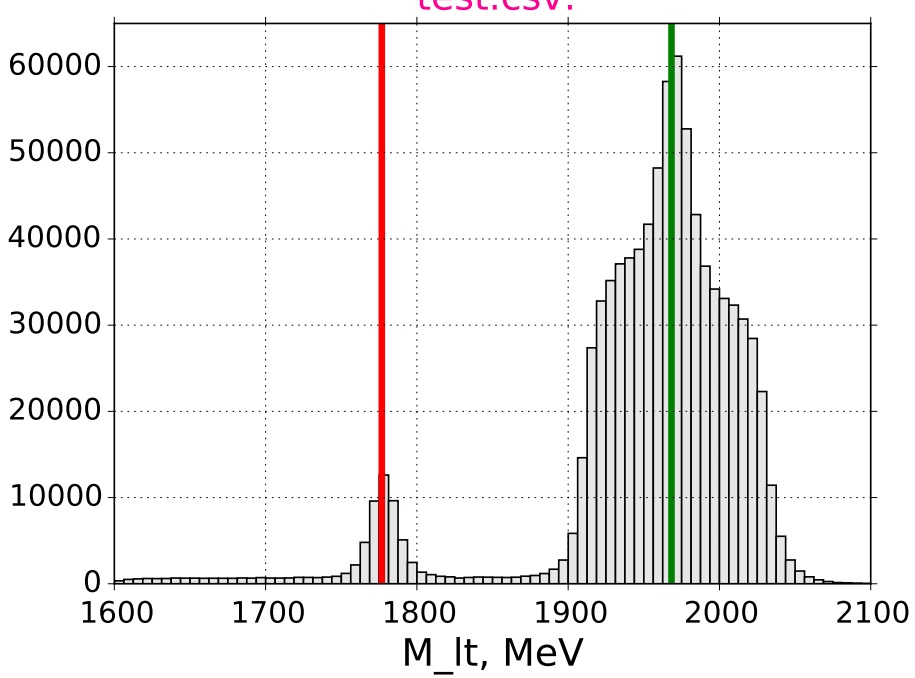
$$\beta\gamma \approx \frac{\text{FlightDistance}}{\text{LifeTime} \cdot c}, \quad \text{where } \beta = \frac{v}{c}, \quad \gamma = \frac{1}{\sqrt{1-\beta^2}},$$

and $c = 299.792458 \ \frac{\text{m}}{\mu\text{s}}$ is the speed of light. Then $\boxed{\textbf{M\_lt} = \frac{\sqrt{\text{pt}^2 + \text{pz}^2}}{\beta\gamma}}$.

training.csv:



test.csv:



⇒ M_lt is a more accurate estimate of the mass than M_inv!

Additional kinematic features

- Features related to the mother particle:

$$\textbf{flag\_M} = \begin{cases} 1 & \text{if } |\text{M\_It} - m_\tau - 1.44| < 17, \\ 0 & \text{otherwise,} \end{cases}$$

$$\textbf{Delta\_M} = \text{M\_It} - \text{M\_inv},$$

$$\textbf{Delta\_E} = \sqrt{\text{M\_It}^2 + (\text{pt}^2 + \text{pz}^2)} - \text{E},$$

$$\textbf{gamma} = \frac{\text{E}}{\text{M\_inv}}, \quad \textbf{beta} = \frac{\sqrt{\text{gamma}^2 - 1}}{\text{gamma}}.$$

- Features related to the final-state particles p0, p1, and p2:

$$\textbf{E0\_ratio} = \frac{\text{E0}}{\text{E}}, \quad \textbf{E1\_ratio} = \frac{\text{E1}}{\text{E}}, \quad \textbf{E2\_ratio} = \frac{\text{E2}}{\text{E}},$$

$$\textbf{p0\_pt\_ratio} = \frac{\text{p0\_pt}}{\text{pt}}, \quad \textbf{p1\_pt\_ratio} = \frac{\text{p1\_pt}}{\text{pt}}, \quad \textbf{p2\_pt\_ratio} = \frac{\text{p2\_pt}}{\text{pt}},$$

$$\textbf{t\_coll} = \frac{\text{p0\_pt} + \text{p1\_pt} + \text{p2\_pt}}{\text{pt}},$$

$$\textbf{eta\_01} = \text{p0\_eta} - \text{p1\_eta}, \quad \textbf{eta\_02} = \text{p0\_eta} - \text{p2\_eta},$$

$$\textbf{eta\_12} = \text{p1\_eta} - \text{p2\_eta}.$$

Additional "geometric" features

- Significance of the flight distance:

$$\textbf{FlightDistanceSig} = \frac{\text{FlightDistance}}{\text{FlightDistanceError}}.$$

- Sums of the original features:

$$\begin{aligned}
\textbf{DOCA_sum} &= \text{DOCAone} + \text{DOCAtwo} + \text{DOCAthree},\\
\textbf{isolation_sum} &= \text{isolationa} + \text{isolationb} + \text{isolationc}\\
&\quad + \text{isolationd} + \text{isolatione} + \text{isolationf},\\
\textbf{IsoBDT_sum} &= \text{p0_IsoBDT} + \text{p1_IsoBDT} + \text{p2_IsoBDT},\\
\textbf{IP_sum} &= \text{p0_IP} + \text{p1_IP} + \text{p2_IP},\\
\textbf{IPSig_sum} &= \text{p0_IPSig} + \text{p1_IPSig} + \text{p2_IPSig},\\
\textbf{CDF_sum} &= \text{CDF1} + \text{CDF2} + \text{CDF3}.
\end{aligned}$$

- Quality of the p0, p1, and p2 tracks:

$$\begin{aligned}
\textbf{track_Chi2Dof} = \big[&(\text{p0_track_Chi2Dof} - 1)^2 + (\text{p1_track_Chi2Dof} - 1)^2\\
&+ (\text{p2_track_Chi2Dof} - 1)^2\big]^{1/2}.
\end{aligned}$$

Description of the solution

- Boosted decision trees (BDTs) is a trusted and widely used tool in HEP.
- *dmlc* **XGBoost** is a popular BDT library successfully used by many Kagglers. It has recently won the "HEP meets ML" award from CERN.
- The model is based on two BDT classifiers using different sets of features and trained independently. The first classifier is geometric and the second one is kinematic. Final prediction is

$$\text{prediction} = \frac{1}{2}\Big[\text{w1} \cdot \text{prediction1} + (1 - \text{w1}) \cdot \text{prediction2}\Big], \quad \text{w1} = 0.78.$$

- The geometric classifier is weaker, but helps to pass the correlation test. To pass the agreement test, it was enough to not use the SPDhits feature.
- Source code in Python: https://github.com/gramolin/flavours-of-physics.

  It contains the following files:
  - features.py (implementation of the additional features)
  - parameters.py (tunable parameters of the model)
  - train.py (training the classifiers)
  - predict.py (making a prediction)

parameters.py

```
# Random seed:
random_state = 1

# Weight for the first classifier:
w1 = 0.78

# Numbers of trees:
num_trees1 = 200 # Classifier 1
num_trees2 = 100 # Classifier 2

# Parameters of the classifiers:
params = {'objective': 'binary:logistic',
          'eta': 0.05,
          'max_depth': 4,
          'scale_pos_weight': 5.,
          'silent': 1,
          'seed': random_state}
```

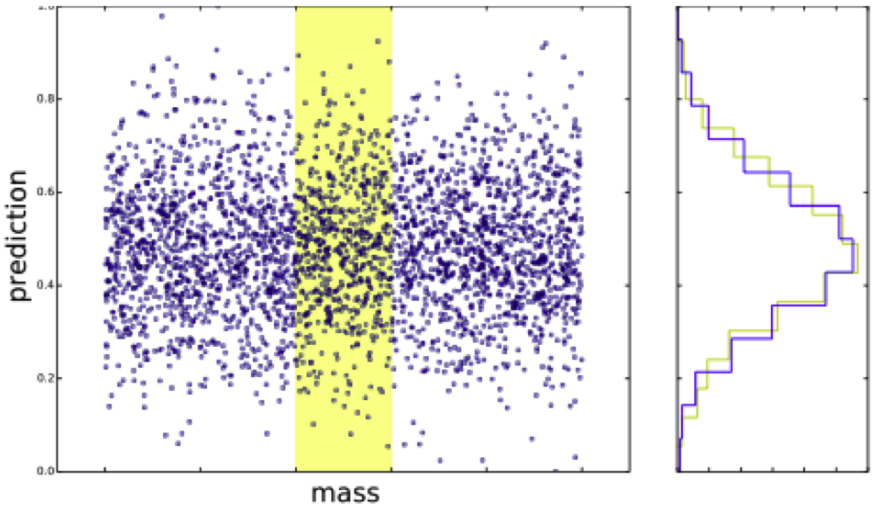
train.py

```
import pandas, xgboost, features, parameters

# Read the training dataset:
train = pandas.read_csv('data/training.csv', index_col='id')
train = train[train['min_ANNmuon'] > 0.4]

# Add extra features:
train = features.add_features(train)

# Train the first (geometric) XGBoost classifier:
bst1 = xgboost.train(parameters.params,
                     xgboost.DMatrix(train[features.list1],
                     train['signal']), parameters.num_trees1)
bst1.save_model('bst1.model')

# Train the second (kinematic) XGBoost classifier:
bst2 = xgboost.train(parameters.params,
                     xgboost.DMatrix(train[features.list2],
                     train['signal']), parameters.num_trees2)
bst2.save_model('bst2.model')
```

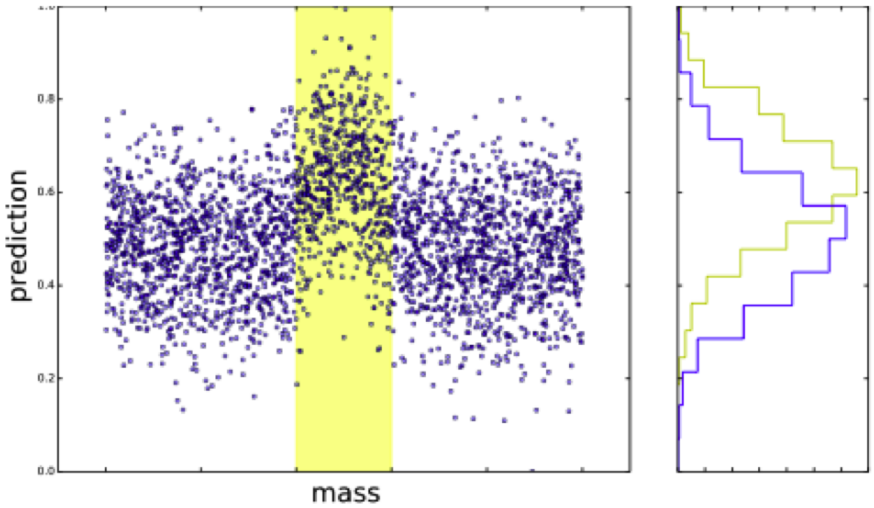
Correlation test

A Cramer–von Mises test is performed to compare the predicted values for the entire dataset with the predicted values within a rolling window.

No correlation with mass:

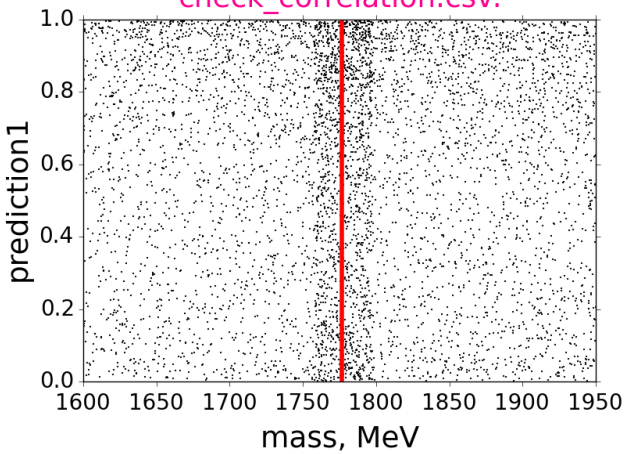


Strong correlation:

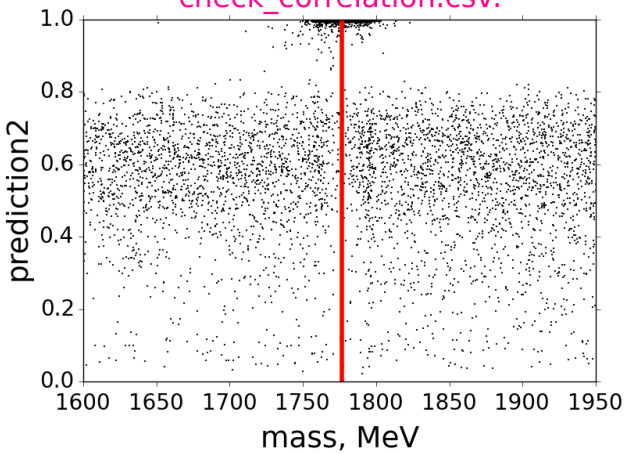


Predictions for the correlation dataset:

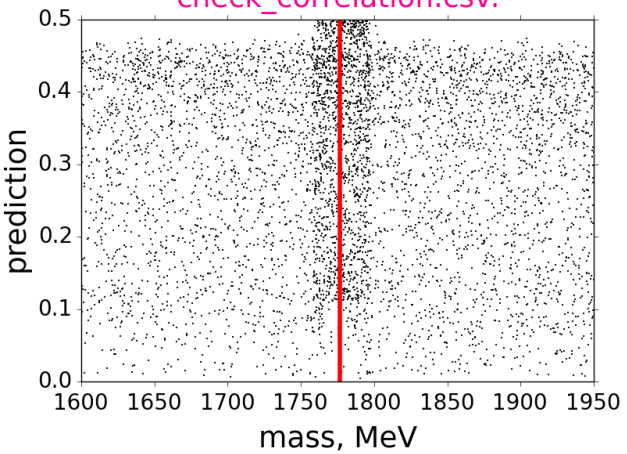
check_correlation.csv:



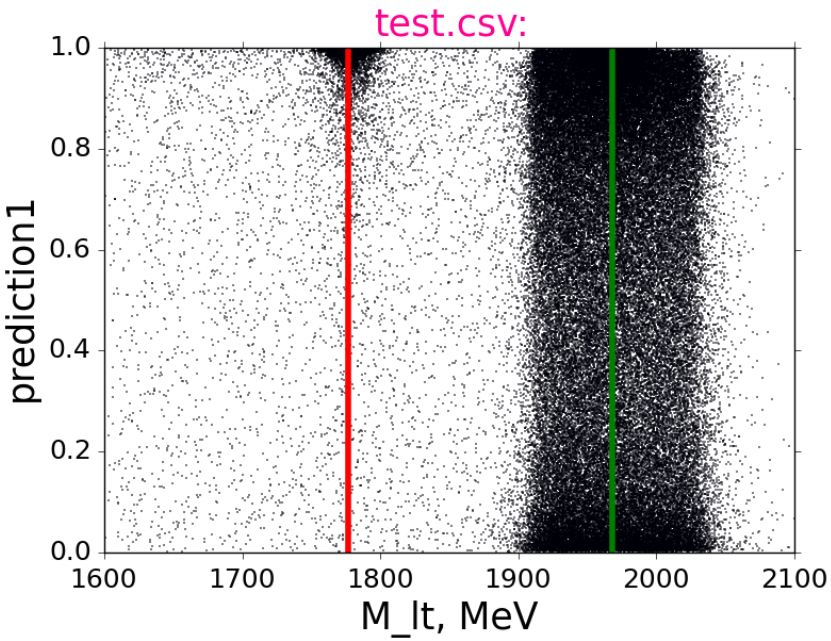
check_correlation.csv:

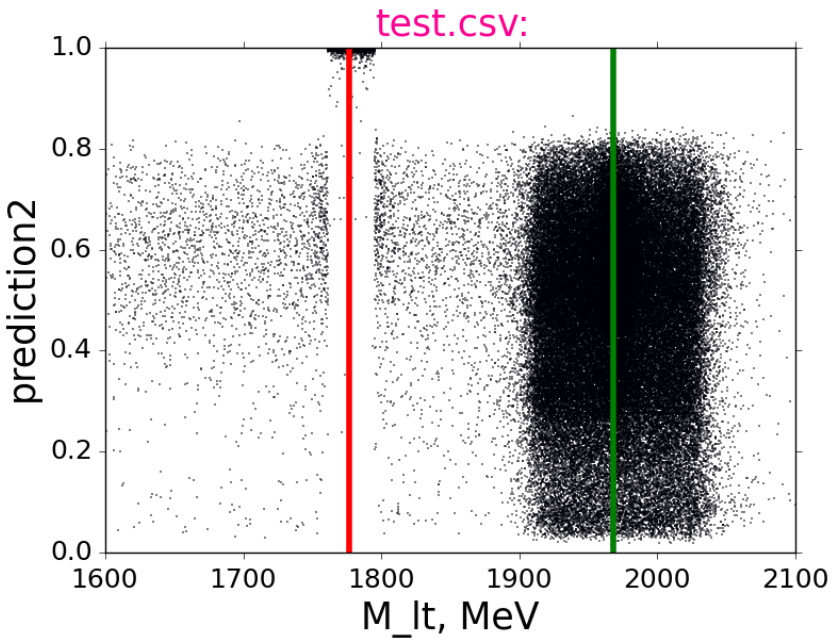


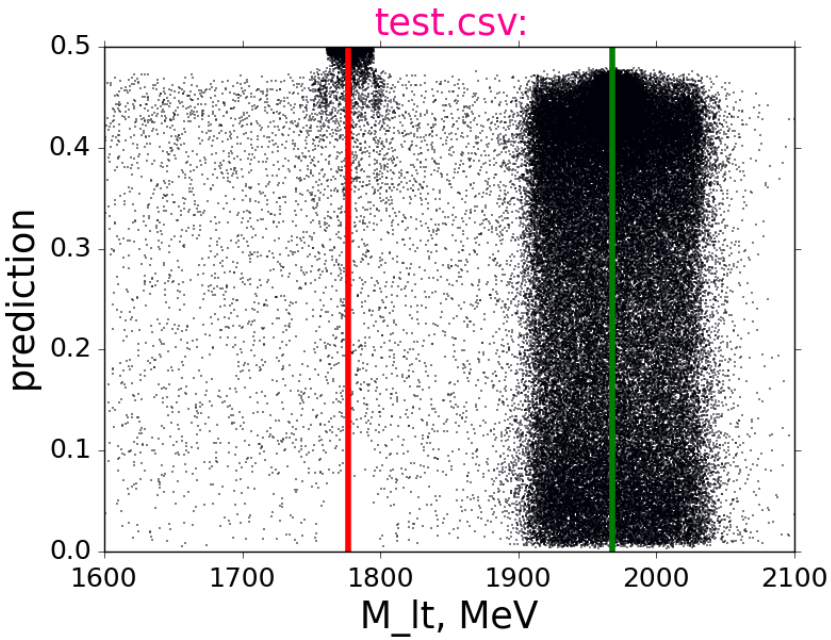
check_correlation.csv:

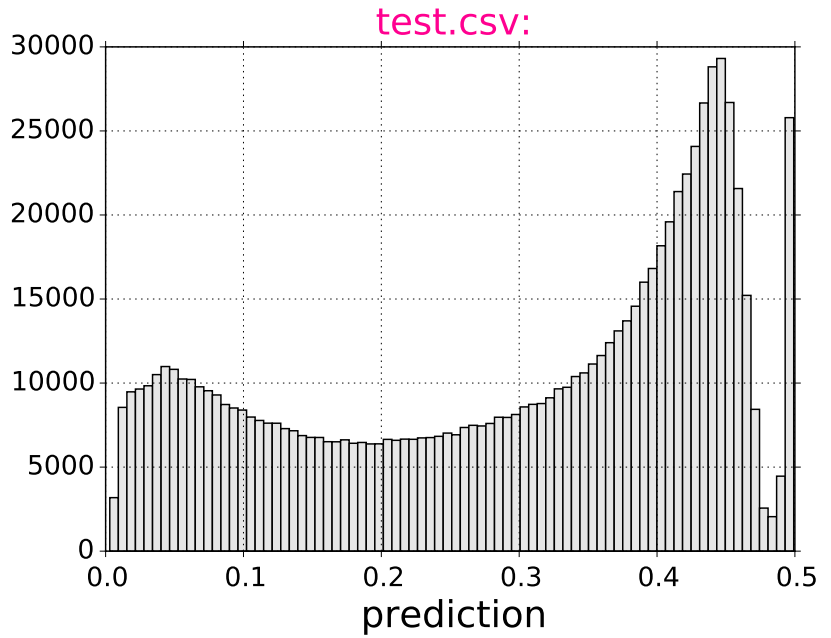


Predictions for the test dataset









Conclusion

- The model is very simple, but scored 0.999998 on the Private Leaderboard.
- Such a high score is possible because the mass can easily be reconstructed and allows us to almost perfectly distinguish between signal and background. The evaluation metric used (weighted area under the ROC curve) helps too.
- At the same time, the correlation test can be passed by combining a strong (kinematic) prediction with a weaker (geometric) one.
- XGBoost is a powerful BDT library that can certainly be useful for HEP.
- This was the first time when real (background) data from the LHC were provided in a public ML challenge. Combining real and simulated data is really difficult.
- I encourage the Organizers to make the datasets publicly available and to provide the rest of the kinematic features (azimuthal angles of p0, p1, p2).
- We all look forward to new Kaggle challenges from CERN and Yandex!

Thank you for your attention!