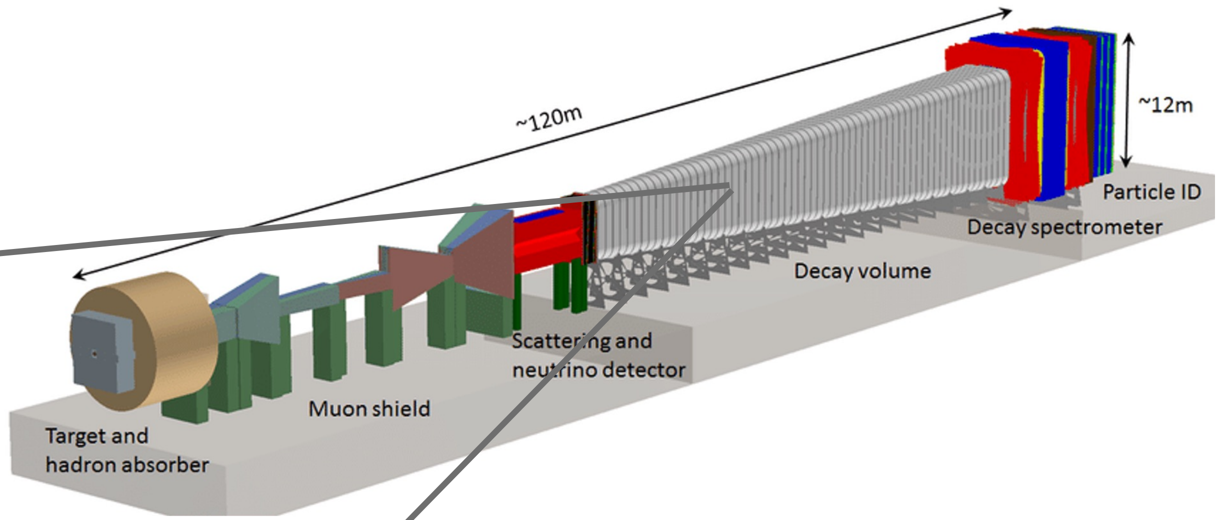
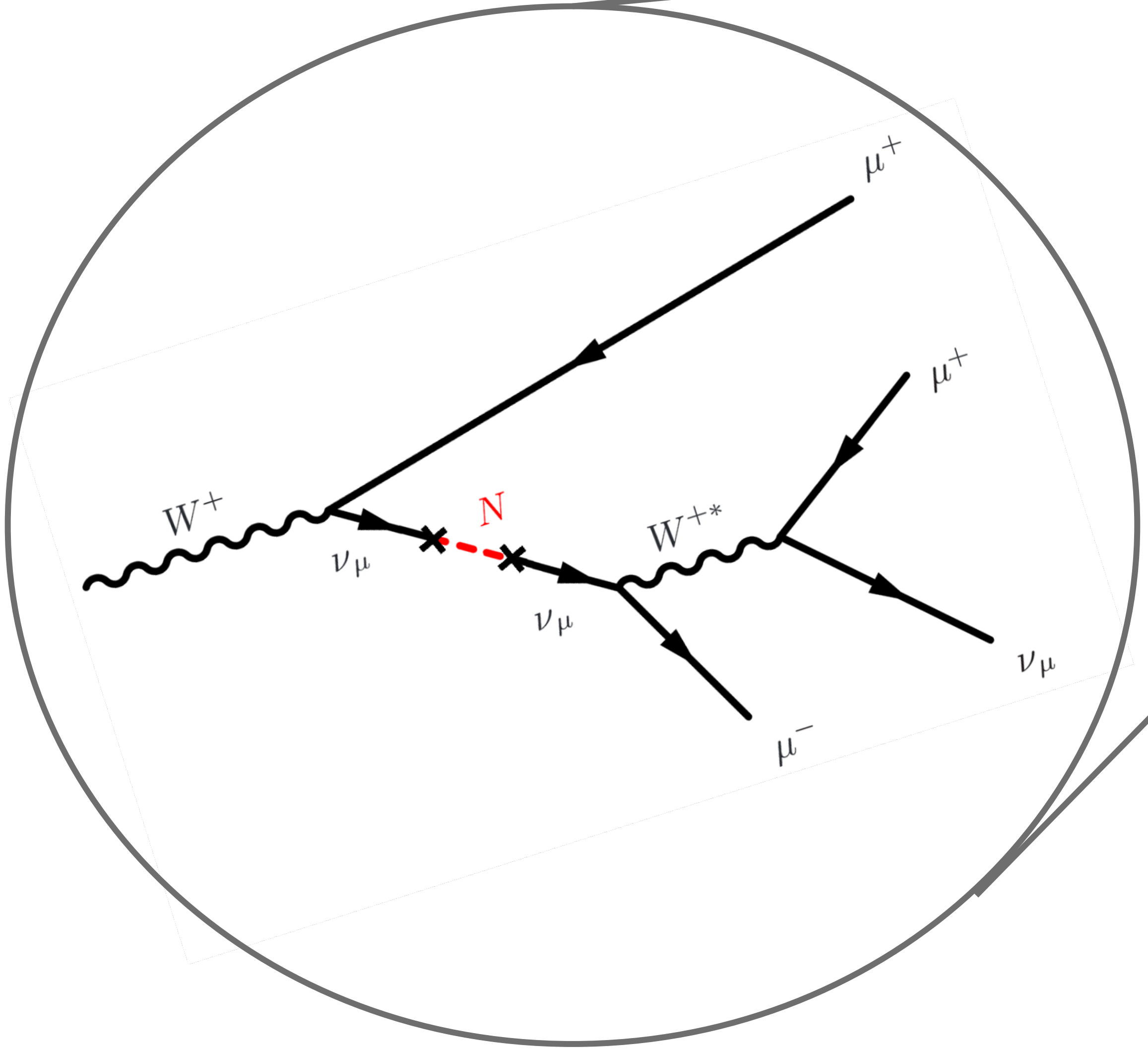




Detection of Heavy Neutral Leptons in SHiP

Anna Bjerregaard & Unik Gyanu

UNIVERSITY OF COPENHAGEN

Overview

- The physics
- Data preparation
- ML algorithms used
  - XGBoost
  - LightGBM
  - MLPClassifier
  - PyTorch
  - Tensorflow
  - Projection and clustering
- Conclusion and improvements

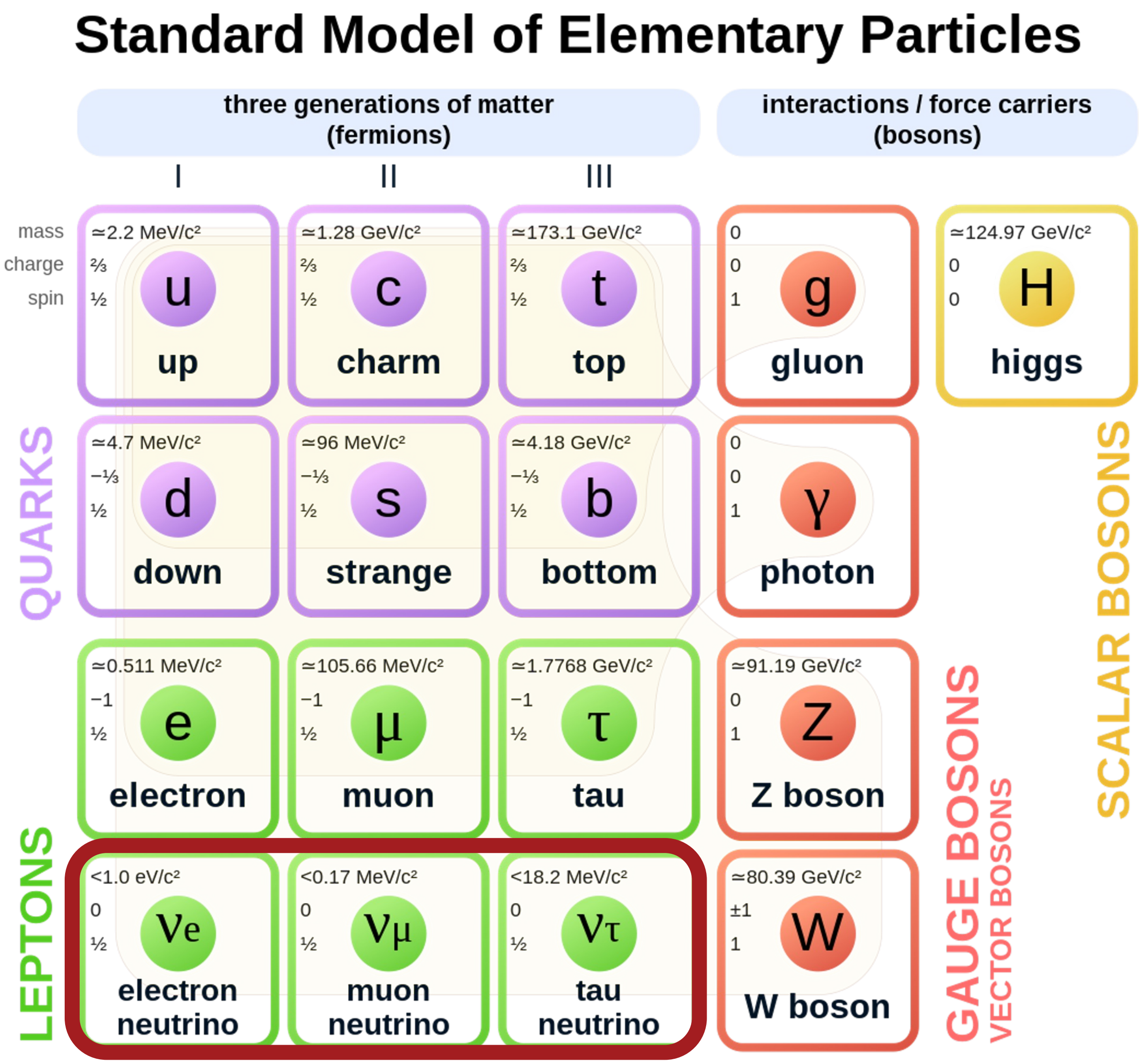
Heavy Neutral Leptons

- Explanation of Neutrino mass
  - “Seesaw mechanism”

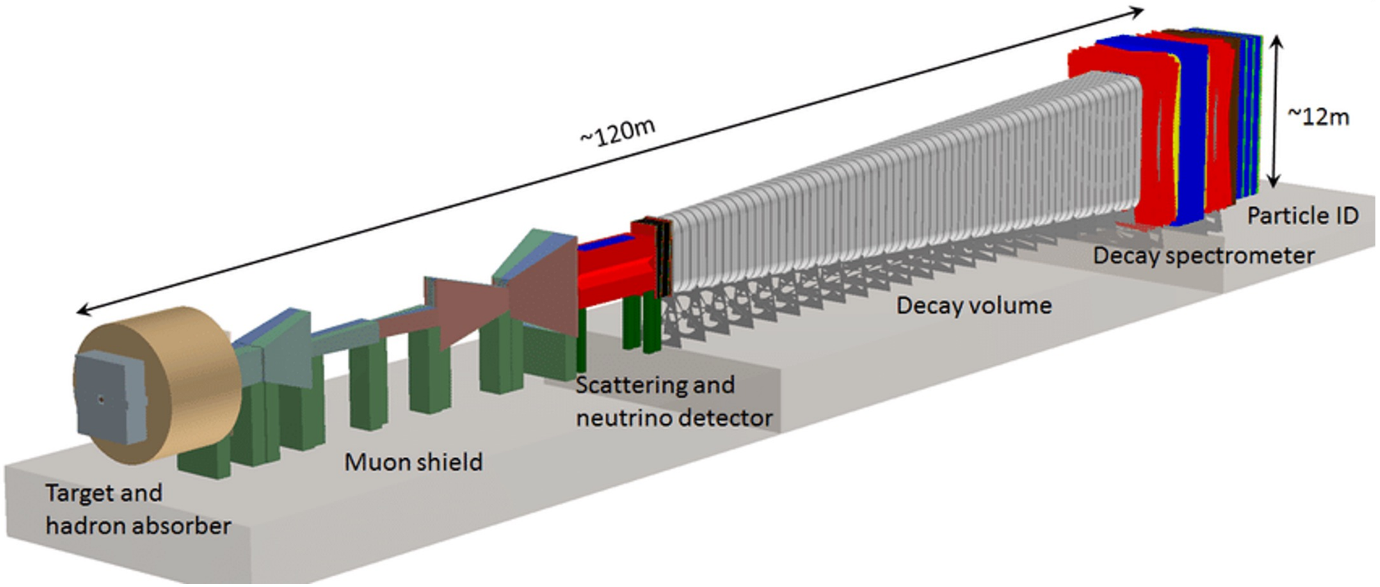
$$\mathcal{L}_{K+D+M} = \sum_{\alpha=e,\mu,\tau} \sum_{I}^{N} \Big(i\nu^\dagger_{\alpha L}\overline{\sigma}^\mu \partial_\mu \nu_{\alpha L}$$

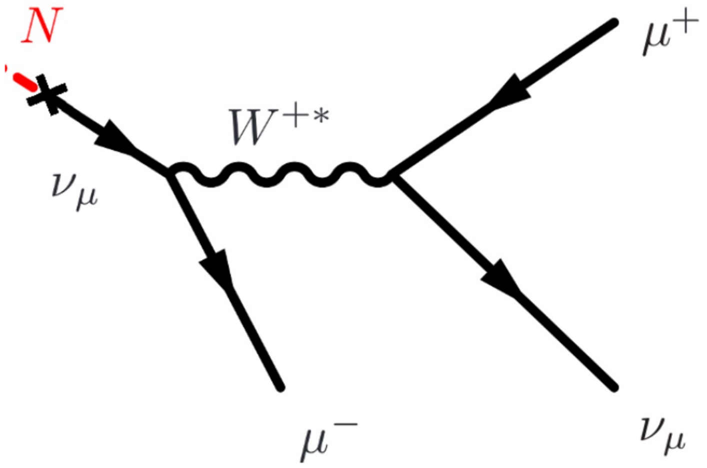
$$+ iN^\dagger_{IR}\sigma^\mu\partial_\mu N_{IR}$$

$$- \Big(m_{\alpha I}\nu^\dagger_{\alpha L} N_{IR} - \frac{iM_I}{2} N^\dagger_{IR}\sigma_2 N^*_{IR} + h.c. \Big)\Big)$$



SHiP Detector





Decay or random muon noise?

Data preparation

- HNL decay simulation provided by Mads Hyttel, Edis Tireli and Oleg Ruchayskiy (1e5 data points for 21 different HNL masses)
- Muon noise simulation provided by us :) (1e6 data points)
  - Details can be provided, but are more physics than machine learning
- Variables: four-momenta of outgoing particles

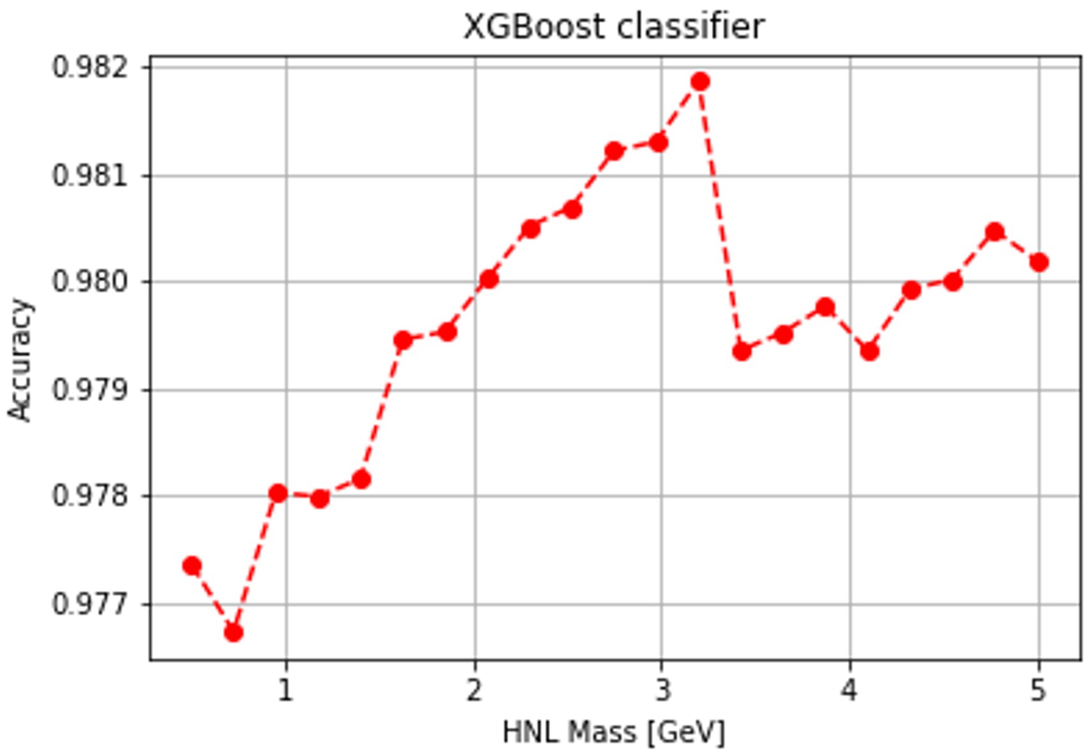
```
["E_mu_plus", "E_mu_minus", "p_mu_plus_x", "p_mu_minus_x", "p_mu_plus_y", "p_mu_minus_y", "p_mu_plus_z", "p_mu_minus_z"]
```

- Target: “Truth” (1 or 0), thus accuracy is easy to measure

XGBoost

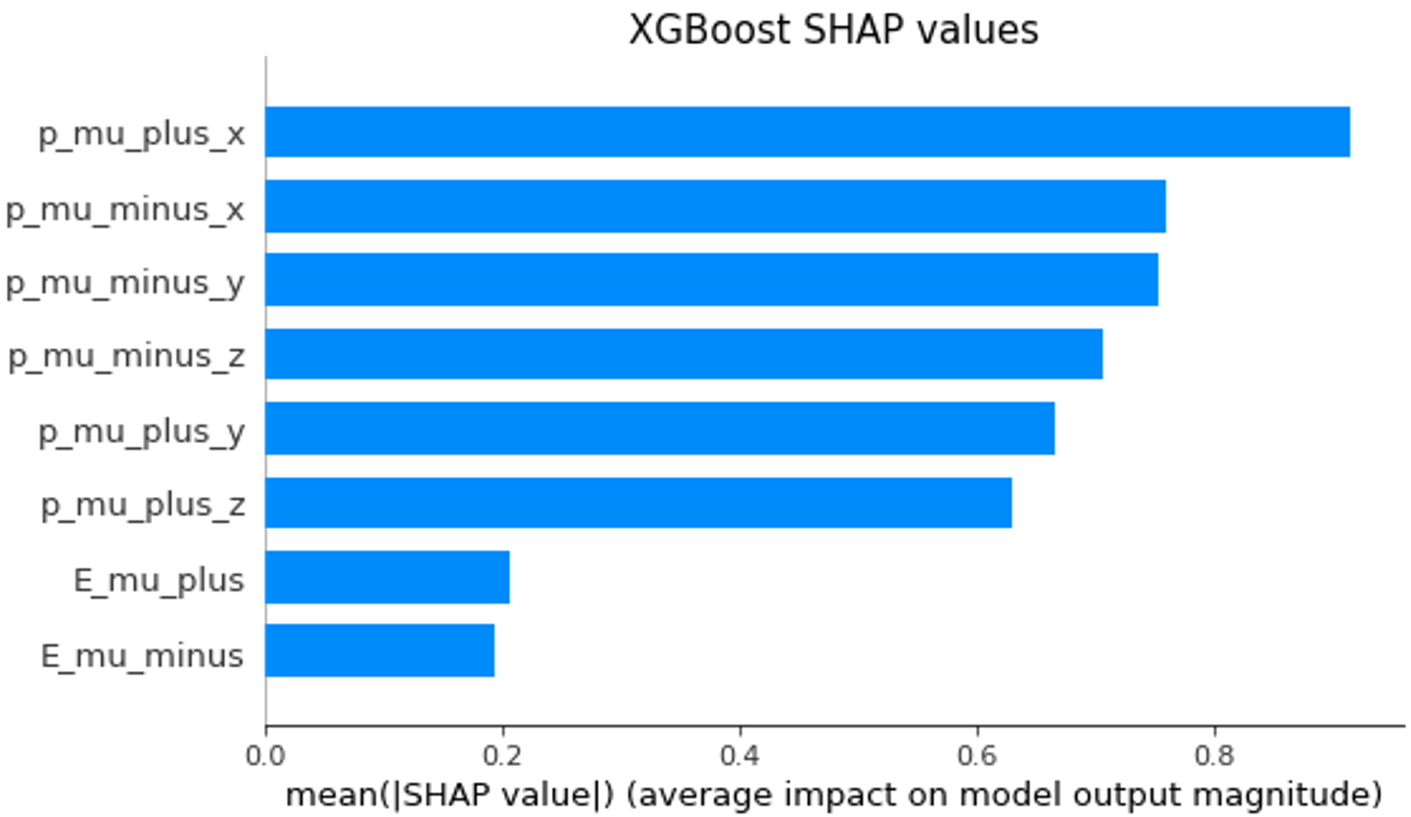
- Bayesian Optimization on N = 1e5 sample

```
The best hyperparameters are :

{'colsample_bytree': 0.7560154270067307, 'gamma': 8.81647216347955, 'max_depth': 9.0, 'min_child_weight': 9.0, 'reg_alpha': 101.0, 'reg_lambda': 0.6158033470421029}
```



XGBoost

- SHAP Values



LightGBM

- RandomSearch with 10 iterations
- No correlation between the HNL mass and accuracy

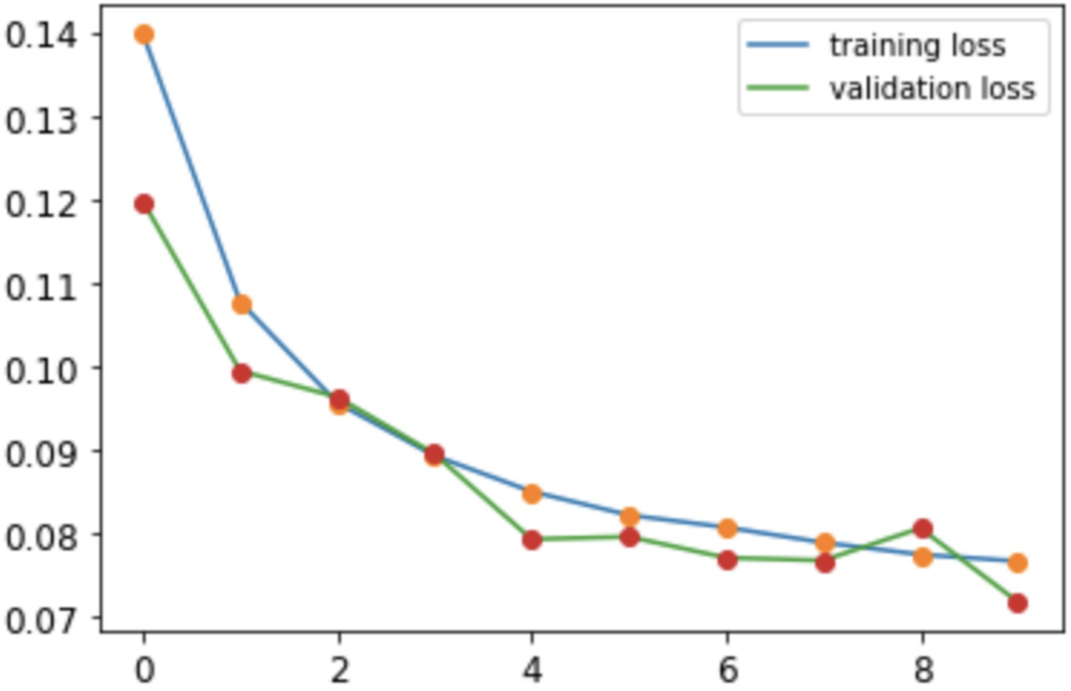
```
The best parameters are: {'max_depth': 8, 'n_estimators': 8, 'num_leaves': 67}
```

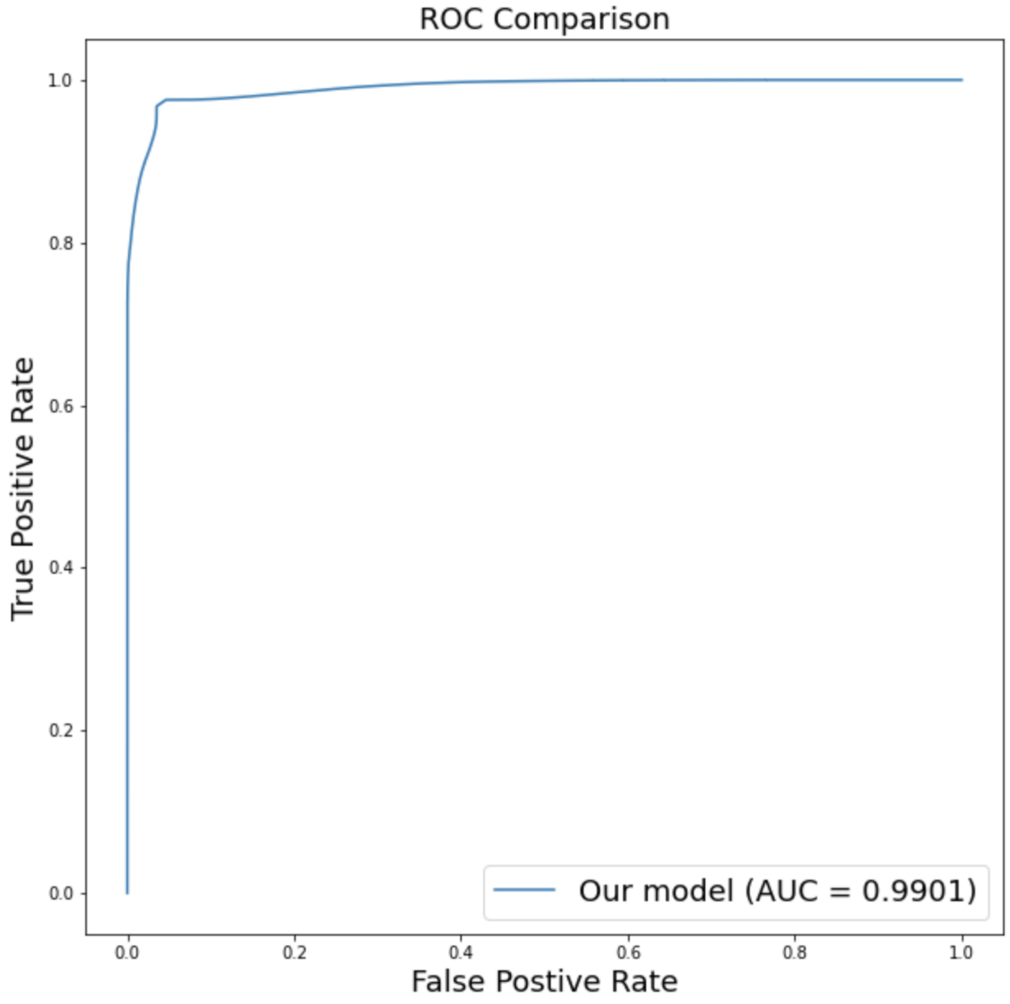
MLPClassifier

- RandomSearch with 10 iterations
- No correlation between the HNL mass and accuracy

```
The best parameters are: {'hidden_layer_sizes': 88, 'activation': 'tanh', 'solver': 'sgd', 'alpha': 0.05, 'learning
_rate': 0.001}
```

PyTorch

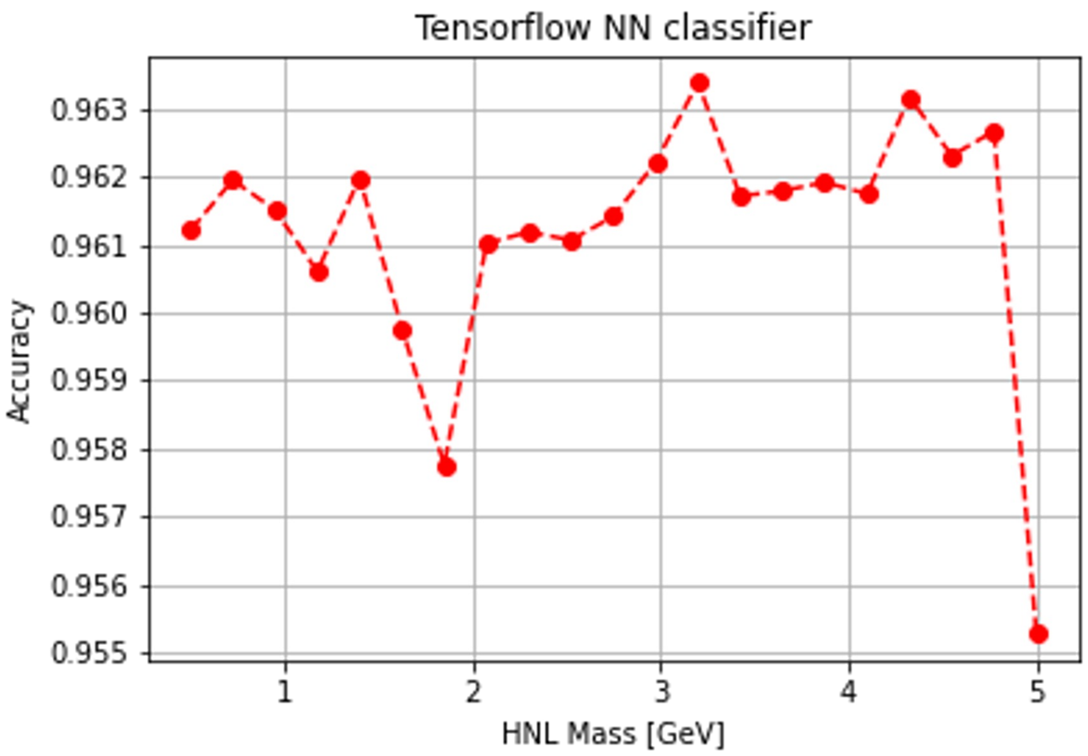

training loss
validation loss
0.14
0.13
0.12
0.11
0.10
0.09
0.08
0.07
0
2
4
6
8


ROC Comparison
True Positive Rate
False Postive Rate
Our model (AUC = 0.9901)
0.0
0.2
0.4
0.6
0.8
1.0

Tensorflow

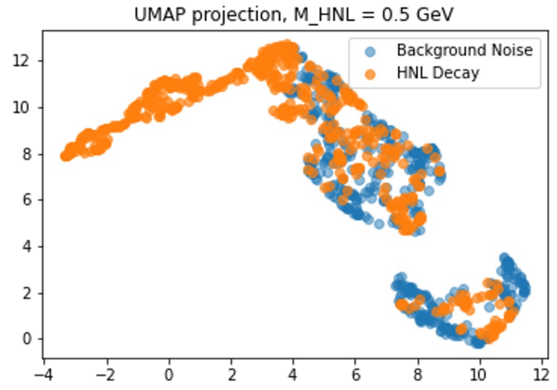
- Bayesian optimization
  - Learning rate = 0.01

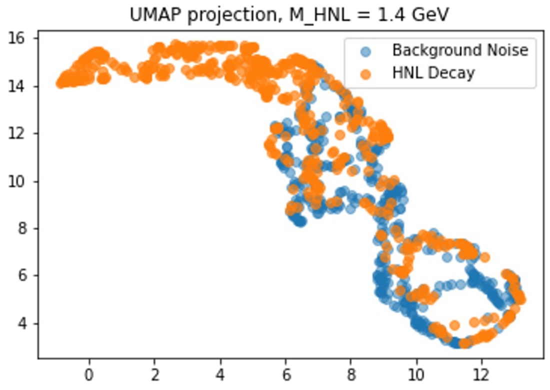
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_layer (Dense)         (32, 12)                  108

 dense (Dense)               (32, 25)                  325

 dense_1 (Dense)             (32, 10)                  260

 output (Dense)              (32, 1)                   11

=================================================================
Total params: 704
Trainable params: 704
Non-trainable params: 0
_________________________________________________________________
```

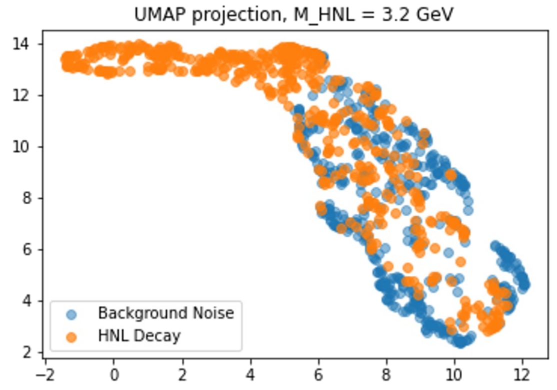


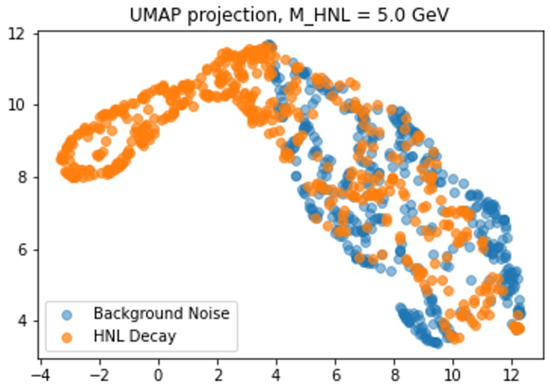
Projection

UMAP

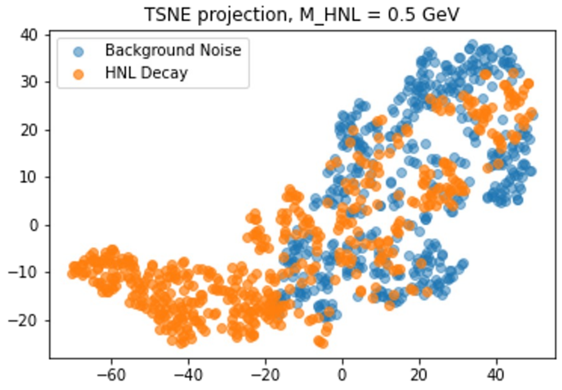


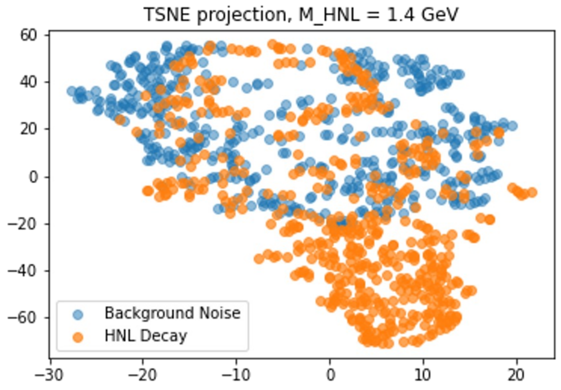


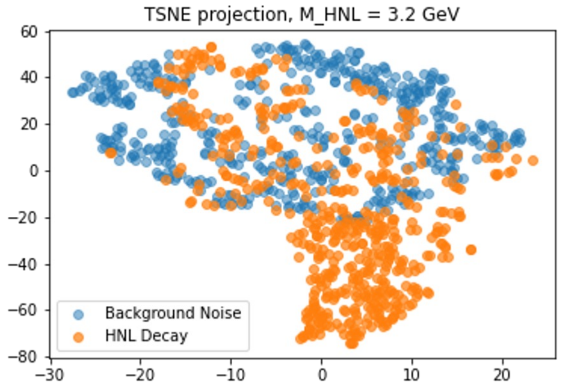


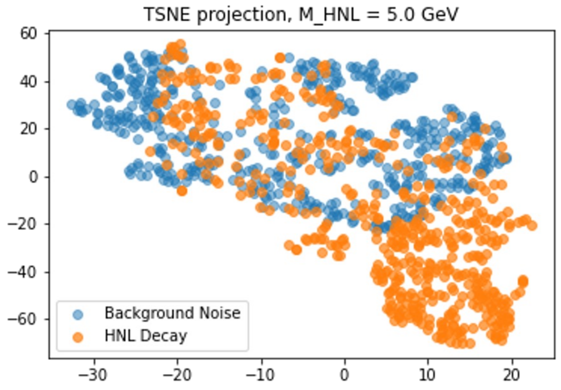


TSNE

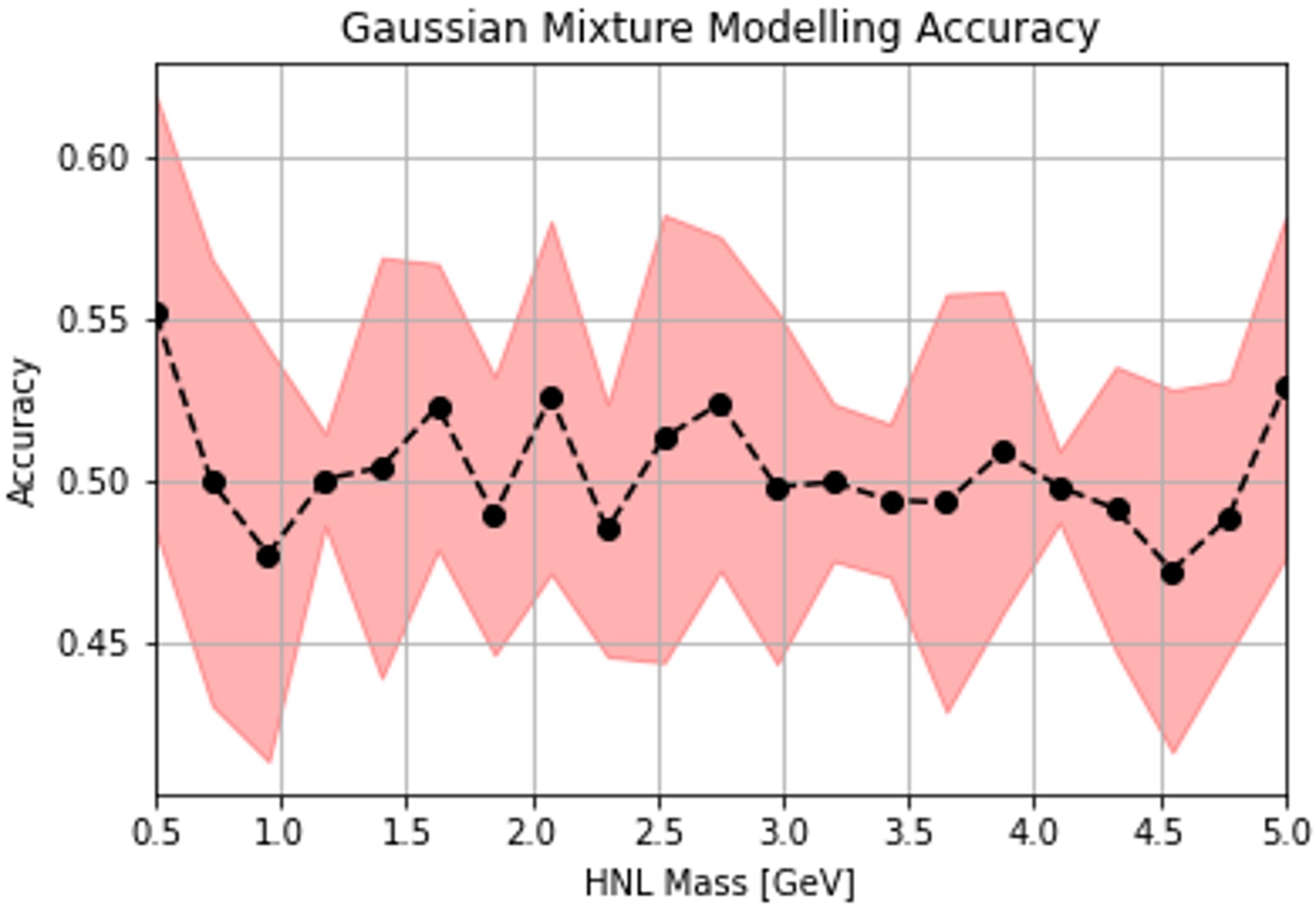








Gaussian Mixture Modelling



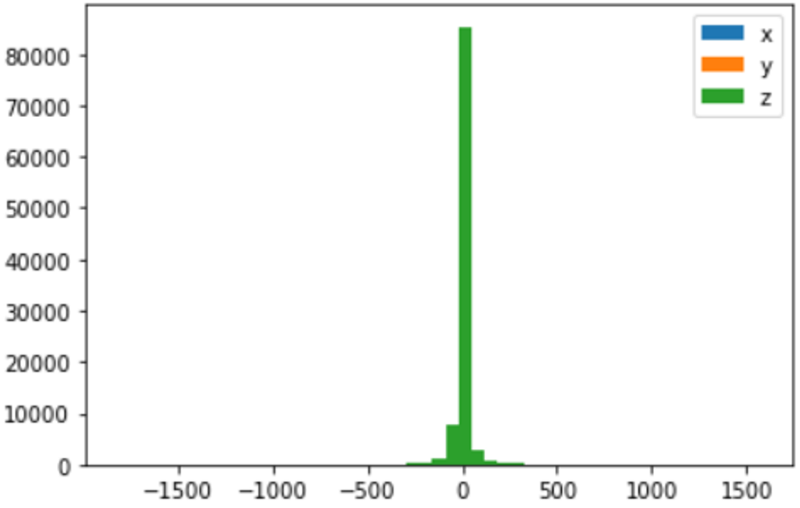
Conclusion and Improvements

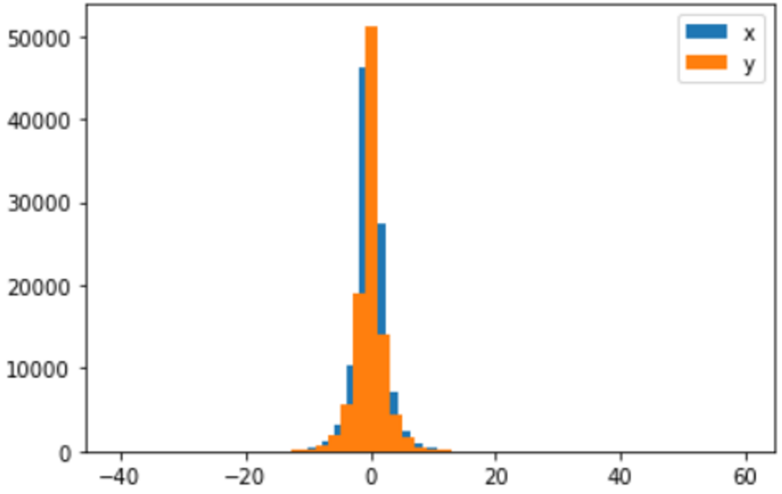
- In general very good results
  - Tree based > NN > Unsupervised
- Improvements
  - More sophisticated simulations (“It would take a small group 1 year to do” - Oleg)
  - Less shortcuts on computational power

Thank you for your time!

Back-up slide: Momentum histograms (Reminder: The noise/signal ratio is 10)

HNL Decay





Noise