MACHINE LEARNING FOR CLAS12 DATA ANALYSIS

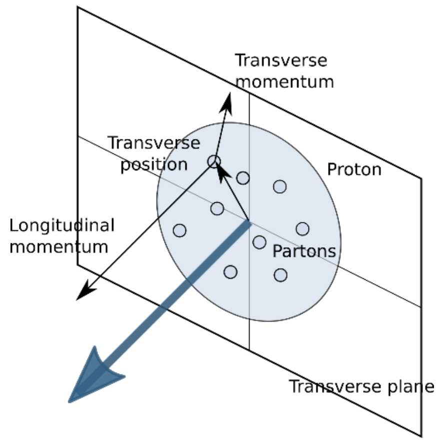
AI for nuclear physics workshop | Noëlie Cherrier

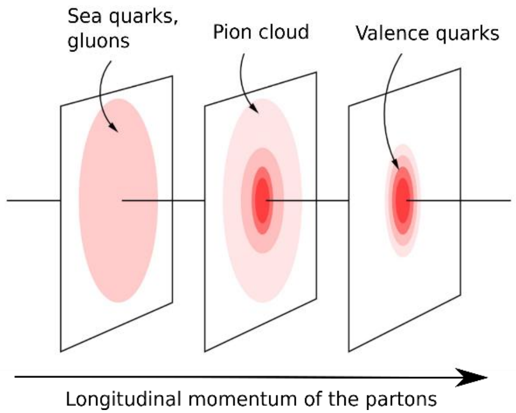
OUTLINE

- Introduction
- Feature construction: principle
- Generalized Additive Models
- CLAS12 data analysis

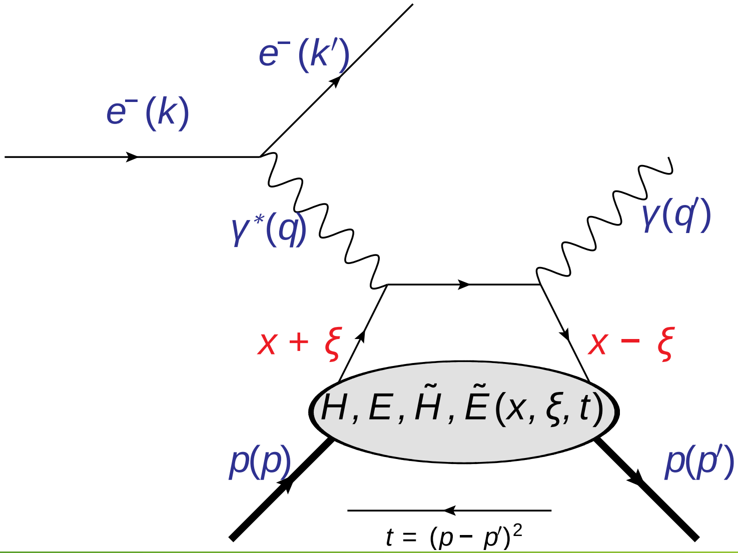
INTRODUCTION

- Physics objective: tomography of the nucleon through **Generalized Parton Distributions** (GPDs)
  - ⟶ Correlation between longitudinal momentum and transverse position of the partons in the nucleon





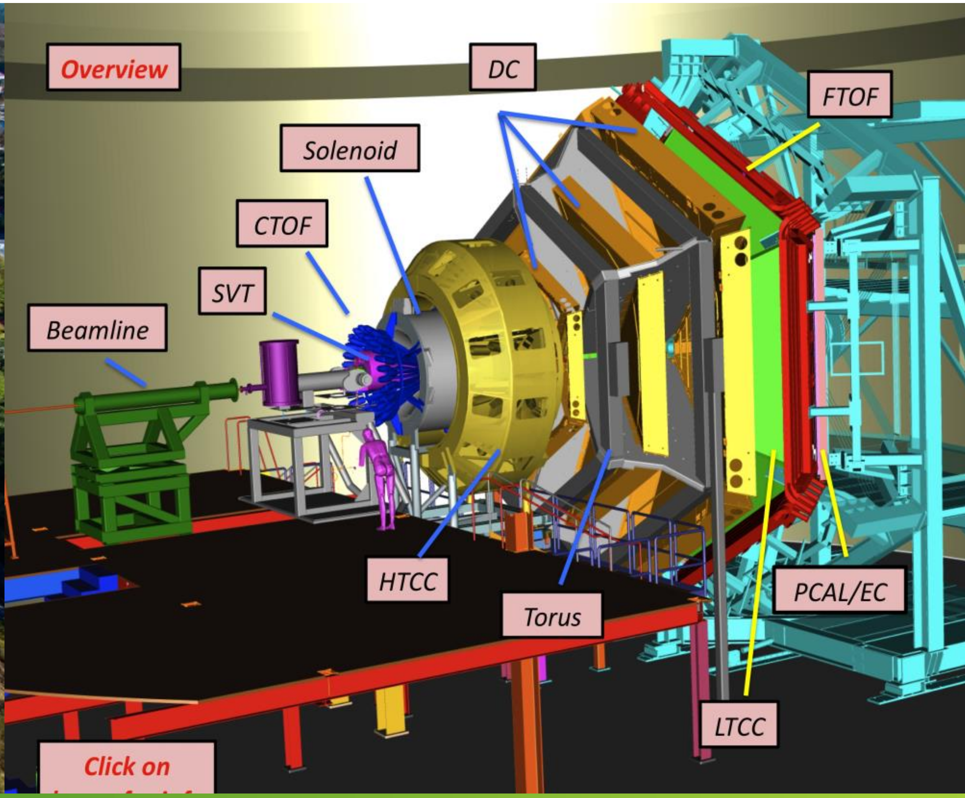
- Accessed through exclusive inelastic processes including **Deeply Virtual Compton Scattering** (DVCS)



INTRODUCTION

- Jefferson Lab: 10.6 GeV electron beam
- CLAS12 data taking since 2018: hydrogen target

Event classification task: isolate DVCS events ($ep \rightarrow ep\gamma$)

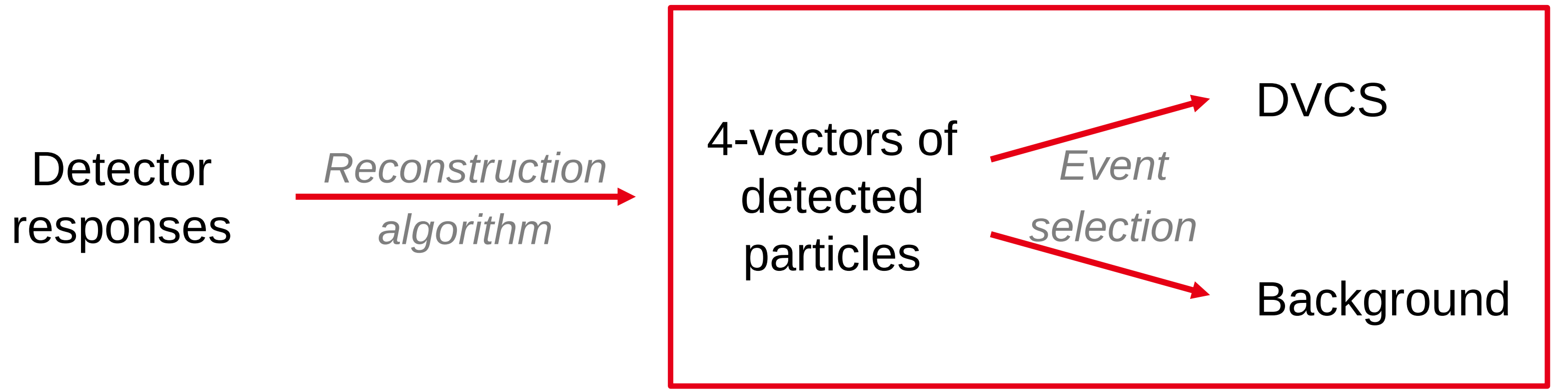


INTRODUCTION

- Jefferson Lab: 10.6 GeV electron beam
- CLAS12 data taking since 2018: hydrogen target

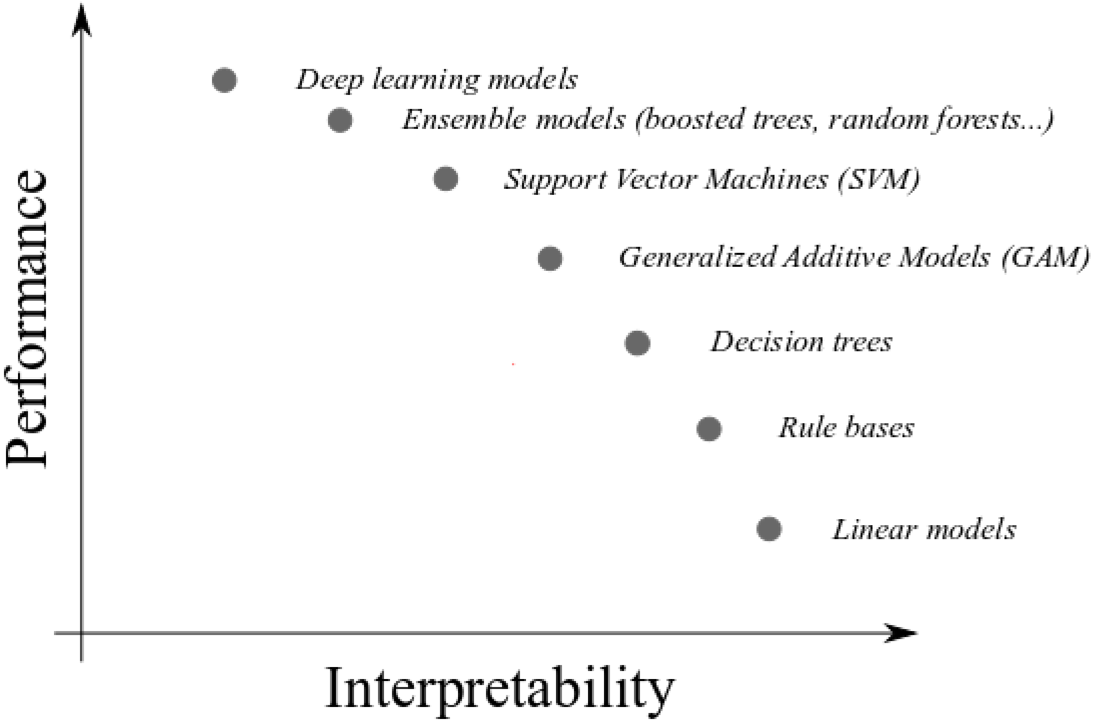
Event classification task: isolate DVCS events ($ep \rightarrow ep\gamma$)

Machine learning approach to be compared to classical approach



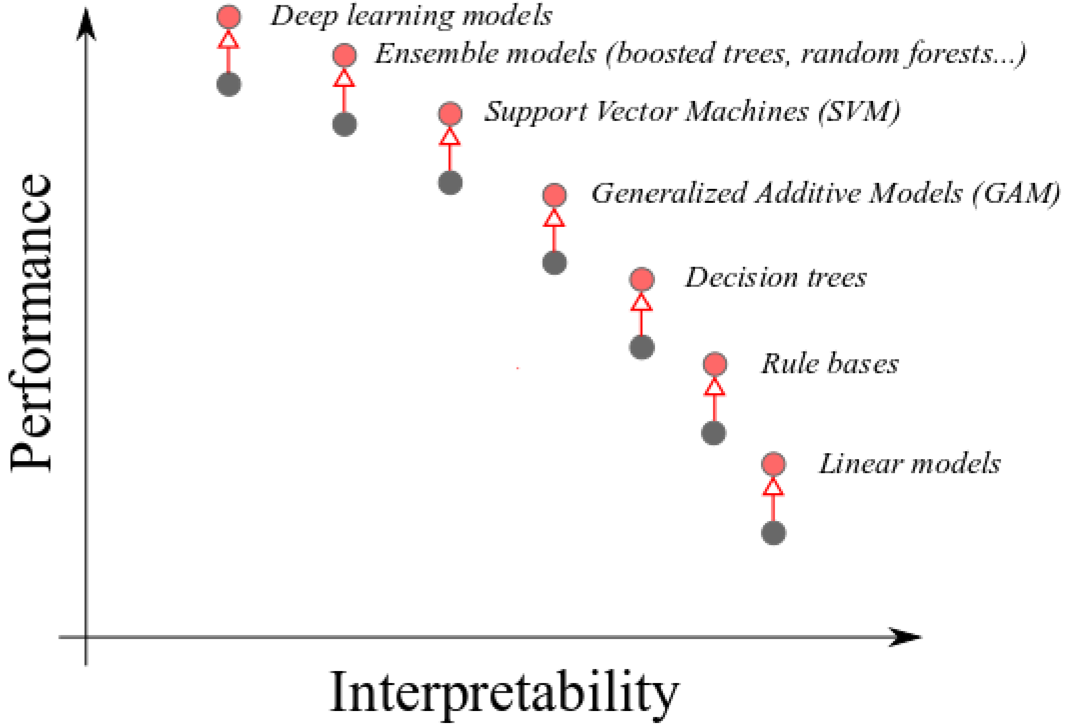
Main background: π^0-production events $ep \rightarrow ep\pi^0 \rightarrow ep\gamma\gamma$

INTERPRETABLE / TRANSPARENT / INTELLIGIBLE MACHINE LEARNING



INTERPRETABLE / TRANSPARENT / INTELLIGIBLE MACHINE LEARNING

Make up for the model drawbacks (notably internal representation)



FEATURE CONSTRUCTION: PRINCIPLE

FEATURE CONSTRUCTION

Motivation: these models do not build a sufficiently complex internal representation of the data

p_x^e φ^τ

θ^γ

p_T^{lep}

$\longrightarrow$

$p_z^e + p_z^p + p_z^{\gamma_1}$

$\cos(\theta^{lep} - \theta^\tau)$

$angle(p^{\gamma_1}, p^{\gamma_1} + p^{\gamma_2})$

$\cos(\varphi^{lep} - \varphi^\tau)$

Base variables

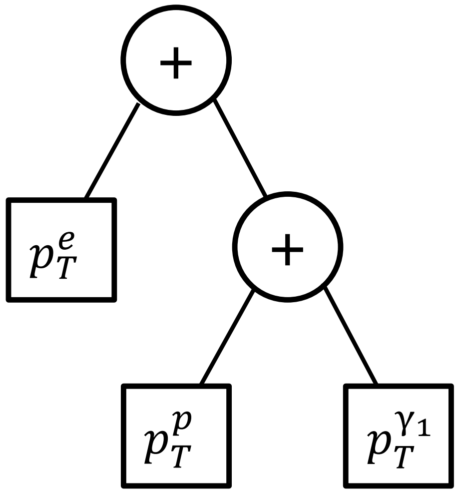
Discriminative and intelligible features

In machine learning: feature engineering, feature construction

FEATURE CONSTRUCTION

Motivation: these models do not build a sufficiently complex internal representation of the data

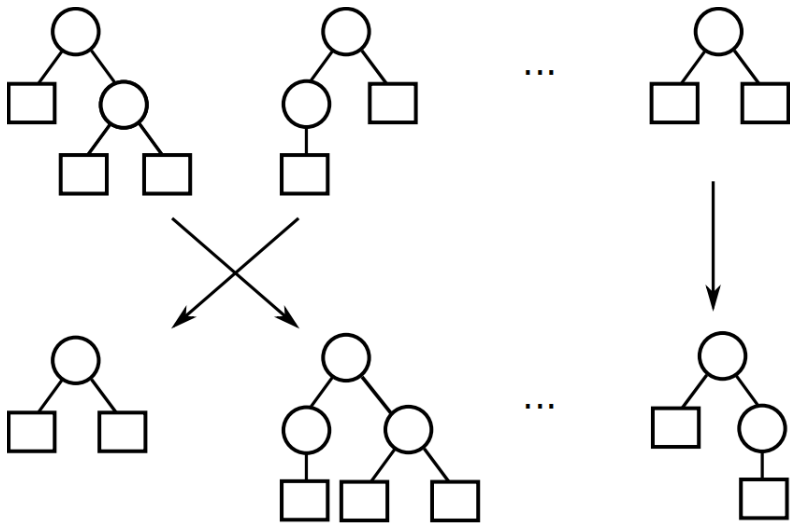
Constrained Genetic Programming: evolve a population of high-level feature candidates



Feature candidate example
→ Nodes are mathematical operators
→ Leaves are base variables

Cherrier, N., Poli, J. P., Defurne, M., & Sabatié, F. (2019, June). Consistent Feature Construction with Constrained Genetic Programming for Experimental Physics. In *2019 IEEE Congress on Evolutionary Computation (CEC)* (pp. 1650-1658). IEEE.

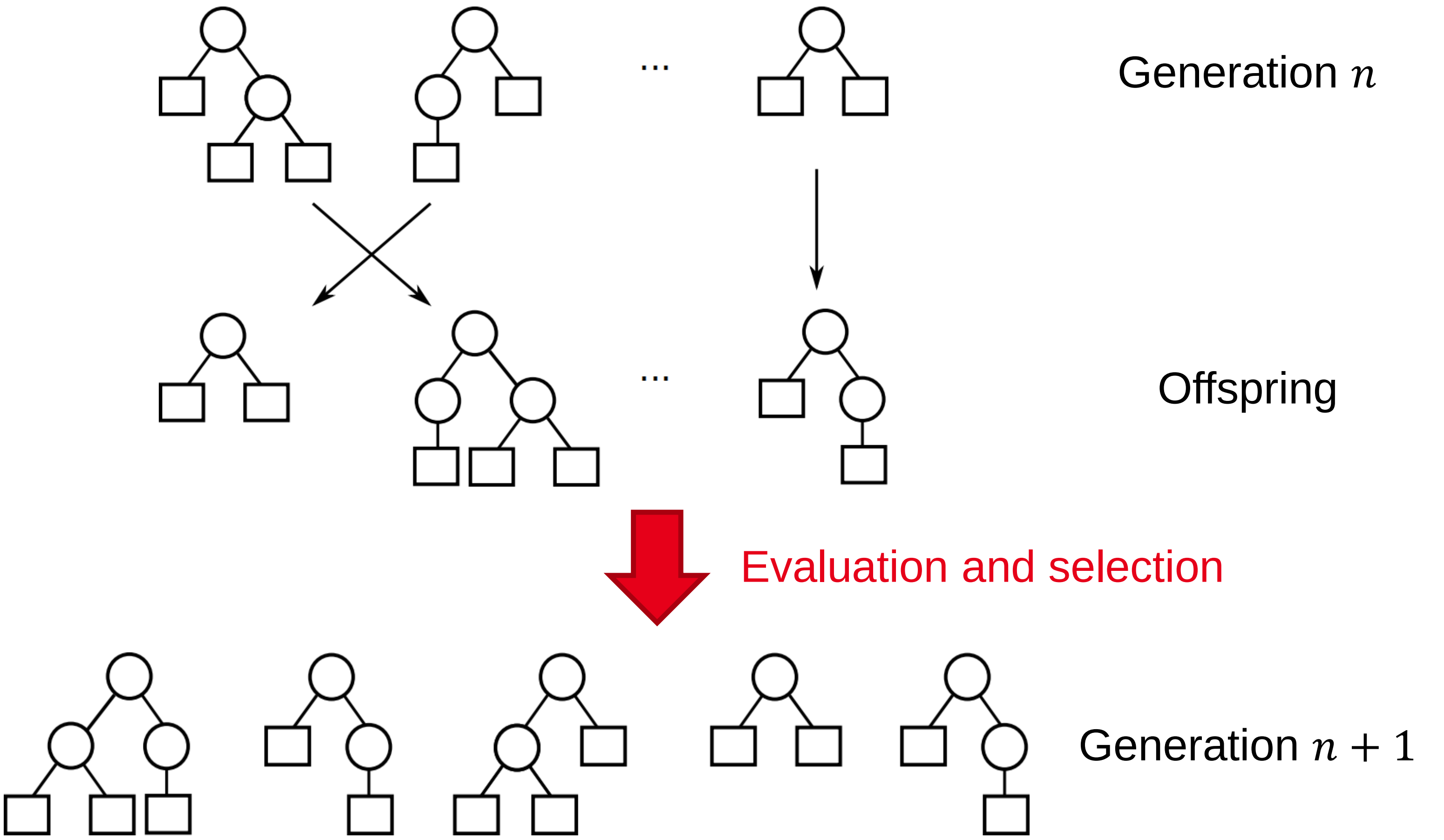
FEATURE CONSTRUCTION



Generation n

Offspring

FEATURE CONSTRUCTION



GENERALIZED ADDITIVE MODELS

GENERALIZED ADDITIVE MODELS (GAM)

$\hat{y}$ predicted output
y true output
x_1, ..., x_d input variables

Generalized Linear Models (GLM) :

$$g(\hat{y}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_d x_d$$

$g(\hat{y}) = \hat{y}$ for regression, $g(\hat{y}) = \ln(\frac{\hat{y}}{1-\hat{y}})$ for classification

GENERALIZED ADDITIVE MODELS (GAM)

$\hat{y}$ predicted output
y true output
$x_1, \ldots, x_d$ input variables

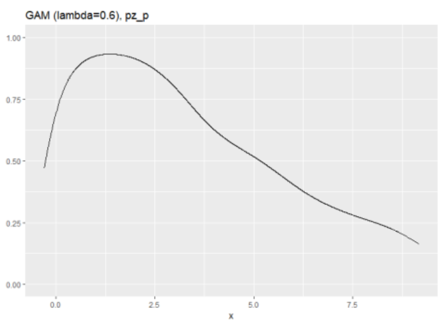
Generalized Linear Models (GLM) :

$$g(\hat{y}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_d x_d$$

$g(\hat{y}) = \hat{y}$ for regression, $g(\hat{y}) = \ln(\frac{\hat{y}}{1-\hat{y}})$ for classification

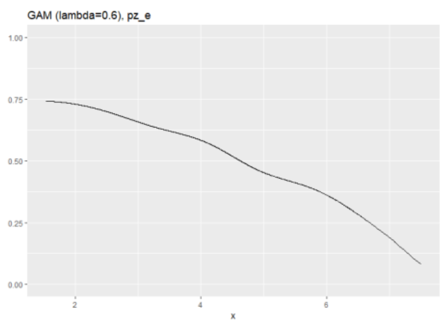
Generalized Additive Models (GAM) :

$$g(\hat{y}) = \beta_0 + f_1(x_1) + \ldots + f_d(x_d)$$



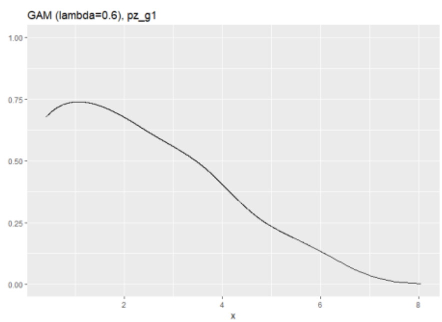
$f_1(x_1)$

$+$



$f_2(x_2)$

$+$



$f_3(x_3)$

$+ \ldots$

Hastie, T. J. (1986). Generalized additive models. In *Statistical models in S* (pp. 249-307). Routledge.
Lou, Y., Caruana, R., Gehrke, J., & Hooker, G. (2013, August). Accurate intelligible models with pairwise interactions. *ACM SIGKDD 2013.*

EMBEDDED FEATURE CONSTRUCTION: IN GAM

Idea: build one feature at a time, associated with one term of the GAM

$\longrightarrow$ **gradient boosting**

Objective function: minimize the cross entropy $-y\ln(\hat{y}) - (1-y)\ln(1-\hat{y})$

EMBEDDED FEATURE CONSTRUCTION: IN GAM

Idea: build one feature at a time, associated with one term of the GAM

$\longrightarrow$ **gradient boosting**

Objective function: minimize the cross entropy $-y\ln(\hat{y}) - (1-y)\ln(1-\hat{y})$

1) Initialize the model by predicting the proportion of the majority class p_0.
   Compute the first model: $g(\hat{y}) = \ln\left(\frac{p_0}{1-p_0}\right)$.
   Compute the residual: $r = y - \hat{y} = y - p_0$ (p_0 proportion of the majority class)

EMBEDDED FEATURE CONSTRUCTION: IN GAM

Idea: build one feature at a time, associated with one term of the GAM

$\longrightarrow$ **gradient boosting**

Objective function: minimize the cross entropy $-y\ln(\hat{y}) - (1-y)\ln(1-\hat{y})$

1) Initialize the model by predicting the proportion of the majority class p_0.
   Compute the first model: $g(\hat{y}) = \ln\left(\frac{p_0}{1-p_0}\right)$.
   Compute the residual: $r = y - \hat{y} = y - p_0$ (p_0 proportion of the majority class)

2) Build one feature x_1 discriminative wrt the residual

EMBEDDED FEATURE CONSTRUCTION: IN GAM

Idea: build one feature at a time, associated with one term of the GAM

$\longrightarrow$ **gradient boosting**

Objective function: minimize the cross entropy $-y\ln(\hat{y}) - (1-y)\ln(1-\hat{y})$

1) Initialize the model by predicting the proportion of the majority class p_0.
Compute the first model: $g(\hat{y}) = \ln\left(\frac{p_0}{1-p_0}\right)$.
Compute the residual: $r = y - \hat{y} = y - p_0$ (p_0 proportion of the majority class)

2) Build one feature x_1 discriminative wrt the residual

3) Fit a shape function $f_1(x_1)$ to the residual

EMBEDDED FEATURE CONSTRUCTION: IN GAM

Idea: build one feature at a time, associated with one term of the GAM
$\longrightarrow$ **gradient boosting**

Objective function: minimize the cross entropy $-y\ln(\hat{y}) - (1-y)\ln(1-\hat{y})$

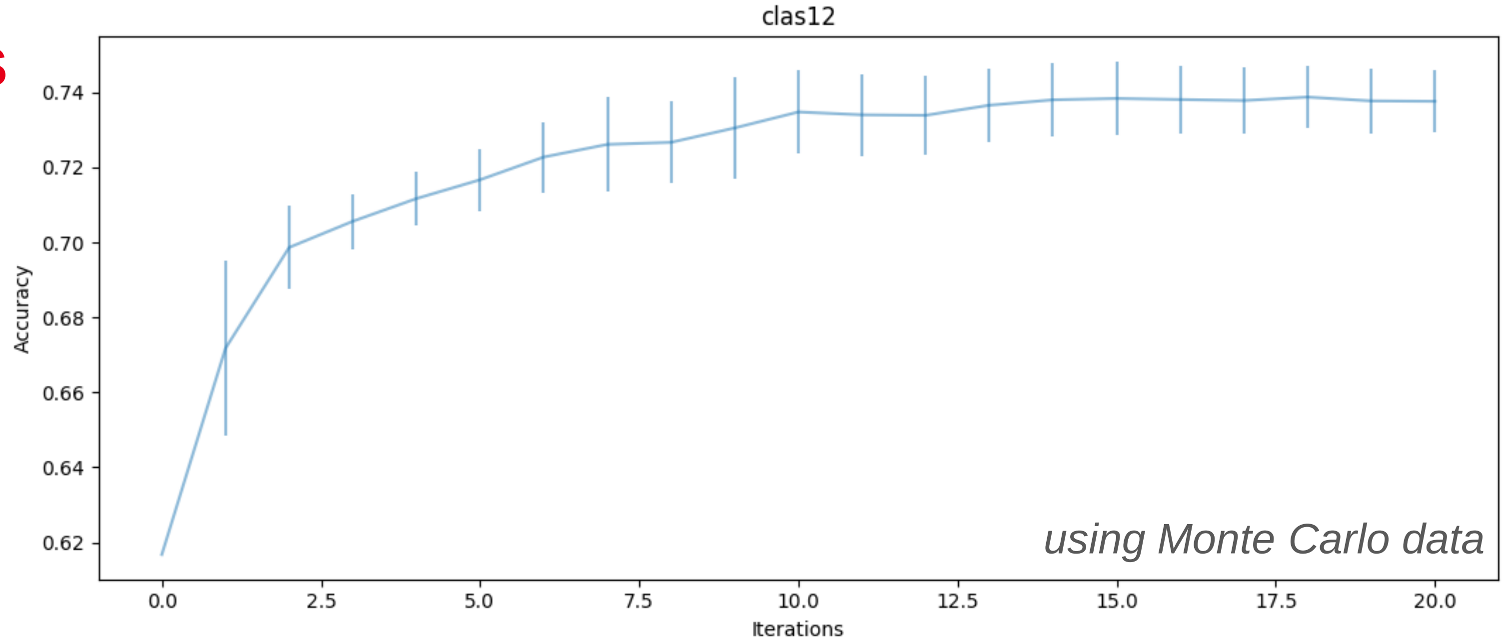
1) Initialize the model by predicting the proportion of the majority class p_0.
Compute the first model: $g(\hat{y}) = \ln\left(\frac{p_0}{1-p_0}\right)$.
Compute the residual: $r = y - \hat{y} = y - p_0$ (p_0 proportion of the majority class)

2) Build one feature x_1 discriminative wrt the residual

3) Fit a shape function $f_1(x_1)$ to the residual

4) Compute the new model: $g(\hat{y}) = g(\hat{y}) + f_1(x_1)$ and the new residual $r = y - \hat{y}$, and go back to step 2

RESULTS



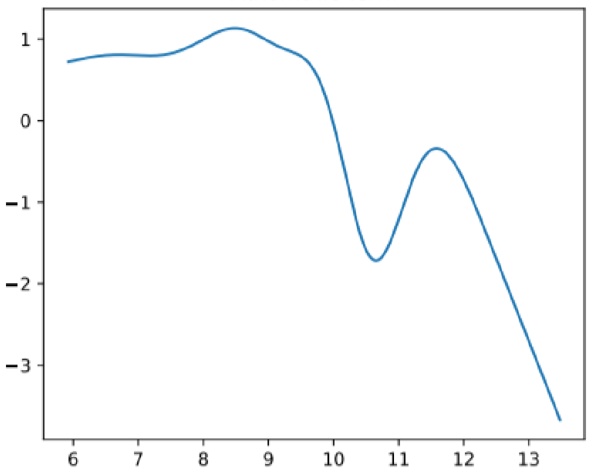
Neural network (2 hidden layers of size 100)	0.7012 ± 0,0062
Linear SVM	0.6911
C4.5 with feature construction	0.718 ± 0,020 (15 nodes using feature construction)
AdaBoost with feature construction	0.7280 ± 0.0063 (50 trees of 1 node each with feature construction)
Gradient Boosting with feature construction	0.7446 ± 0.0071 (100 trees of 7 nodes each with feature construction)

Baselines:

RESULTS

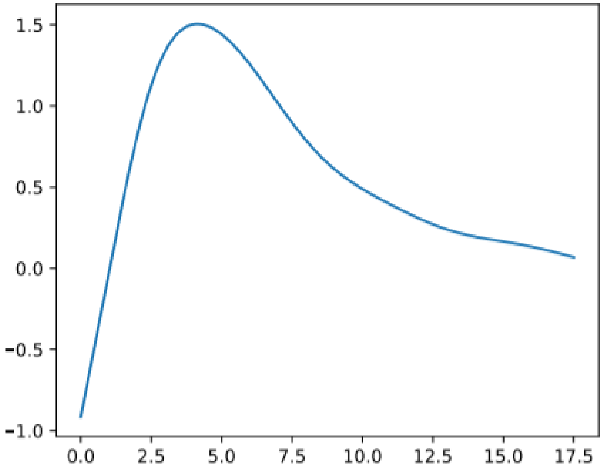
Example of a model (the lower the y value, the higher the probability to have a DVCS event):

$$p_z^e + p_z^p + p_z^{\gamma_1}$$



$+$

$$angle(p^{\gamma_2}, p^{\gamma_1} + p^{\gamma_2})$$

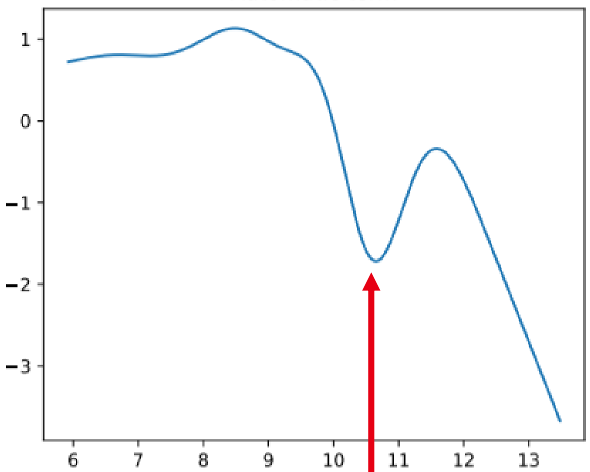


$+ \quad \ldots$

RESULTS

Example of a model (the lower the y value, the higher the probability to have a DVCS event):

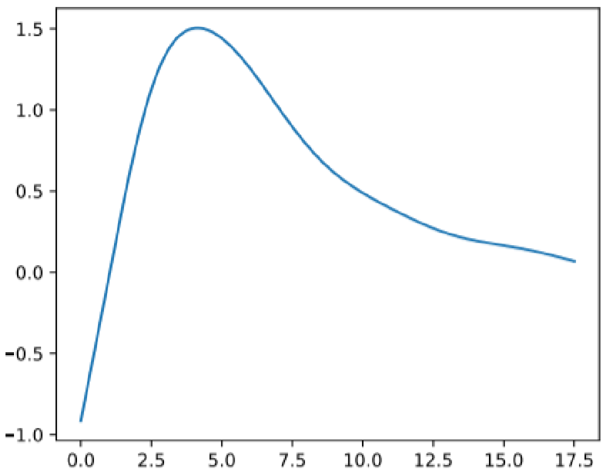
$$p_z^e + p_z^p + p_z^{\gamma_1}$$



Beam energy

$+$

$$angle(p^{\gamma_2}, p^{\gamma_1} + p^{\gamma_2})$$



$+ \quad \ldots$

CLAS12 DATA ANALYSIS

COMPARISON WITH OTHER ANALYSIS APPROACHES

Classical DVCS event selection

$$-0{,}05\ GeV^2 \leq {MM^2}_{ep\rightarrow ep\gamma X} \leq 0{,}05\ GeV^2$$

$$0{,}1\ GeV \leq MM_{ep\rightarrow e\gamma X} \leq 1{,}7\ GeV$$

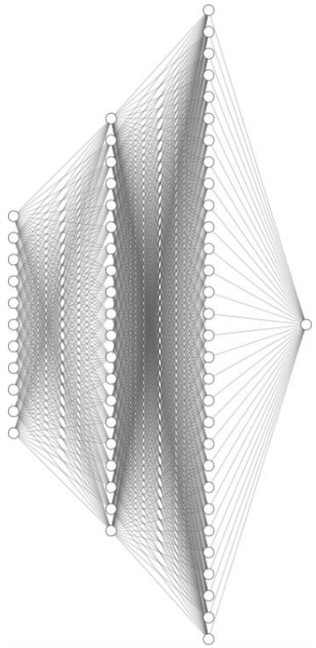
$$-1\ GeV \leq missing\ energy \leq 2\ GeV$$

$$missing\ p_T\ (ep \rightarrow epX) \leq 0{,}4\ GeV$$

$$cone\ angle \leq 4°$$

PhD student at Saclay+Glasgow

Neural network approach

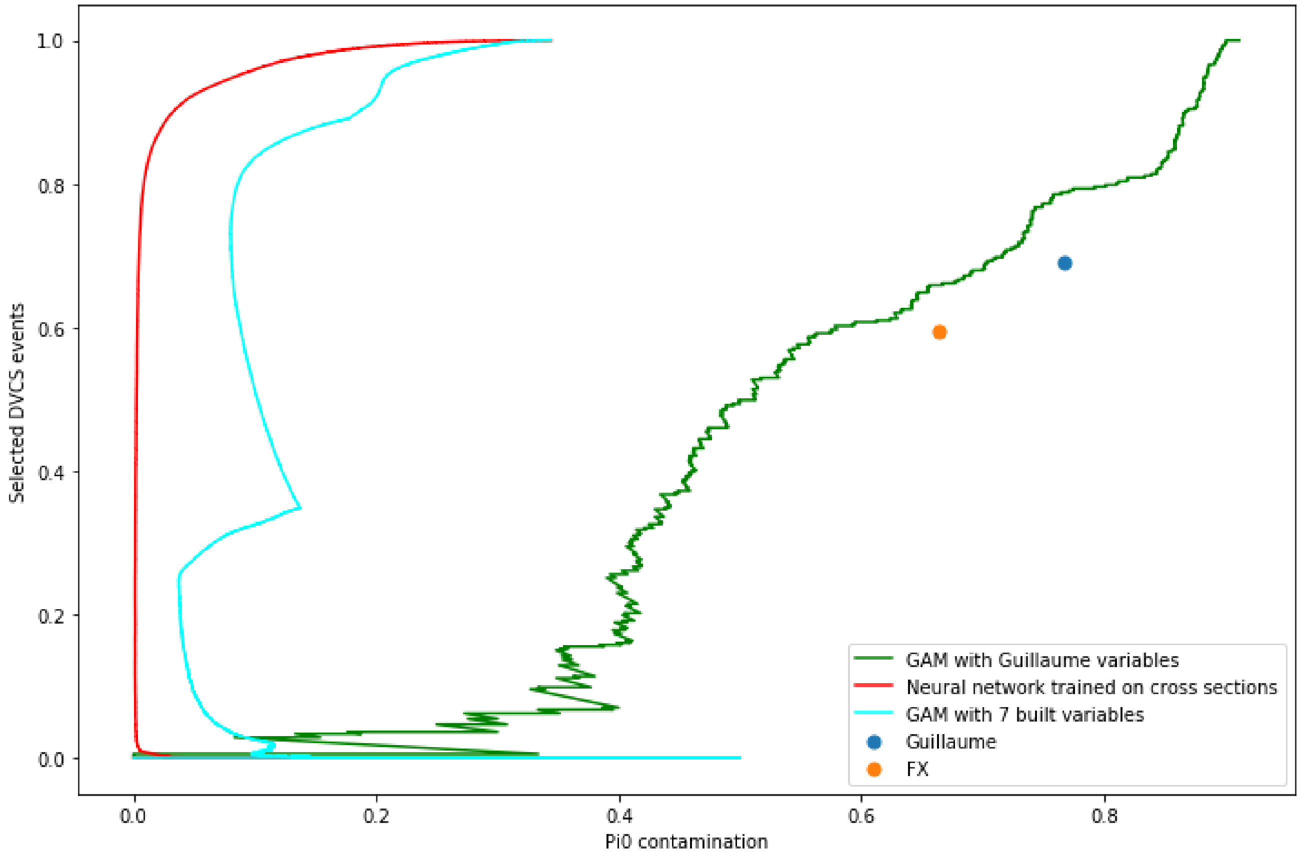


2 hidden layers of size (20, 30)
11 high-level input features

Post-doc at Saclay

COMPARISON WITH OTHER ANALYSIS APPROACHES

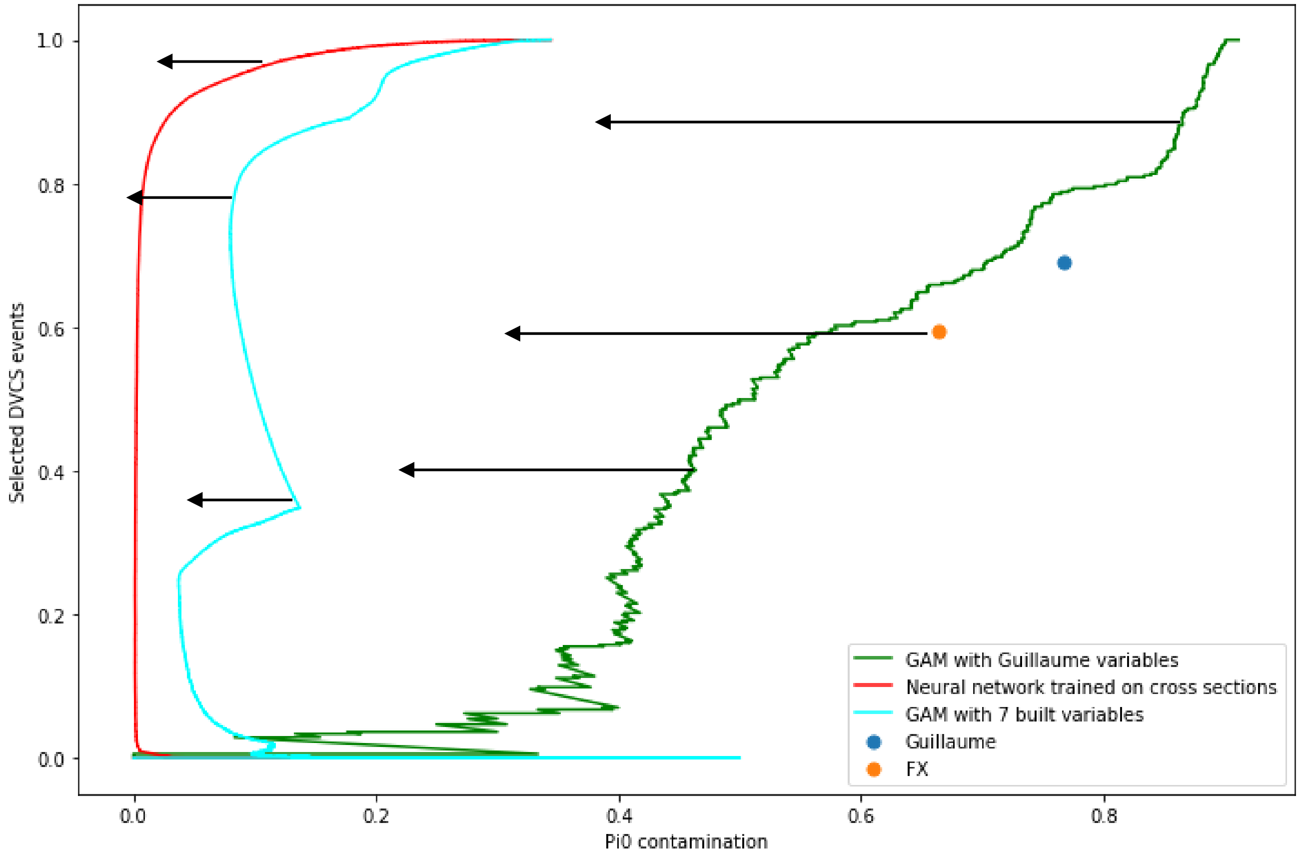
Y axis: percentage of selected DVCS events among all existing DVCS in simulated data
X axis: percentage of Pi0 events still present in the selected subset



COMPARISON WITH OTHER ANALYSIS APPROACHES

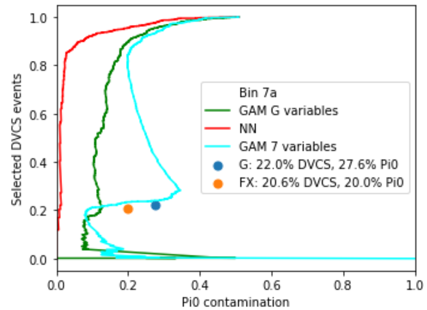
Y axis: percentage of selected DVCS events among all existing DVCS in simulated data
X axis: percentage of Pi0 events still present in the selected subset

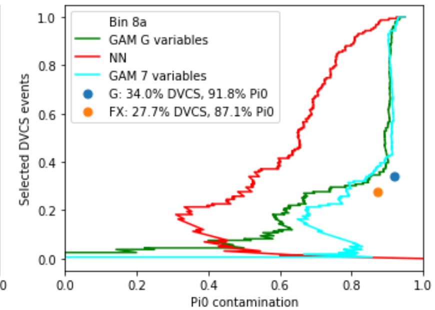
! Pi0 subtraction method

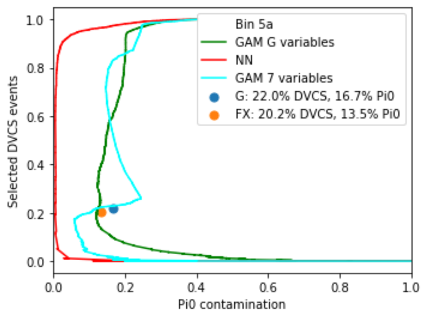


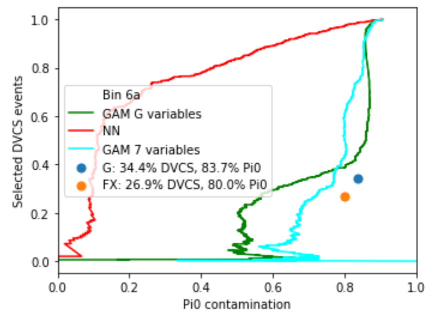
COMPARISON WITH OTHER ANALYSIS APPROACHES

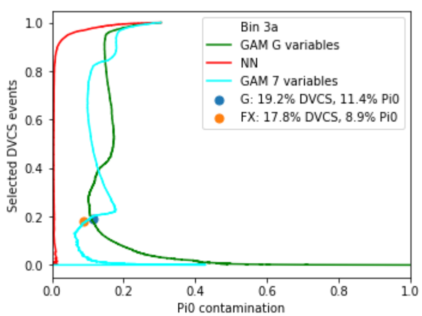
$$-\frac{t}{Q^2} \geq 0{,}25$$

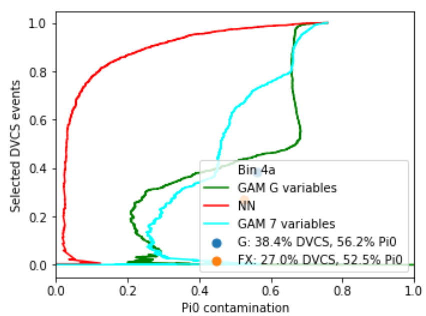


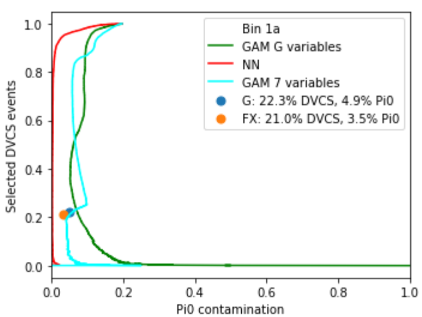


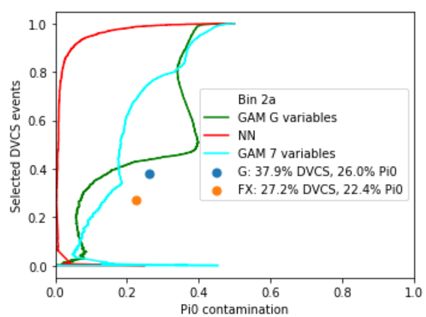








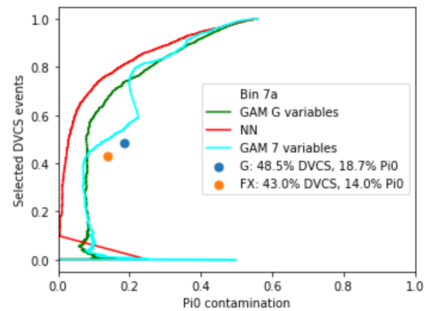


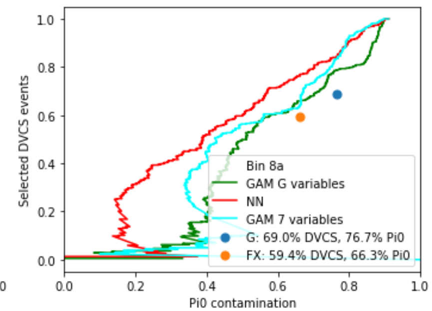


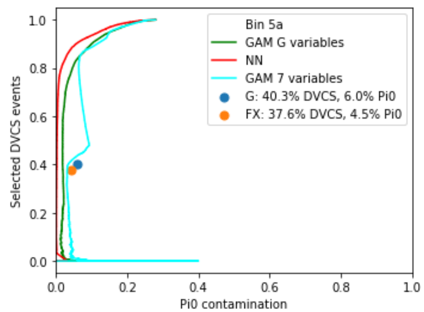
- Neural network always perform better
  Better use of correlations?

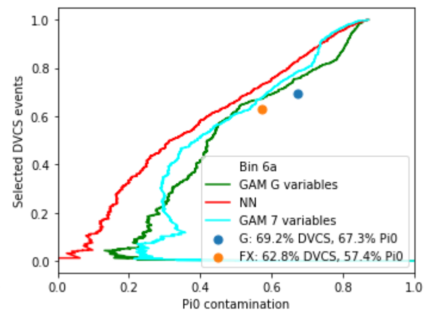
COMPARISON WITH OTHER ANALYSIS APPROACHES

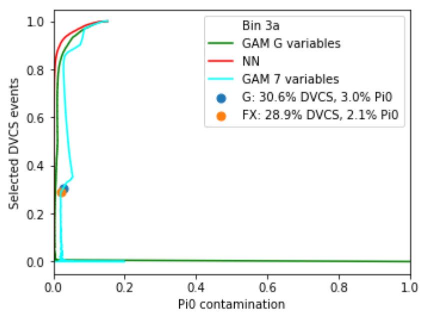
$$-\frac{t}{Q^2} < 0{,}25$$

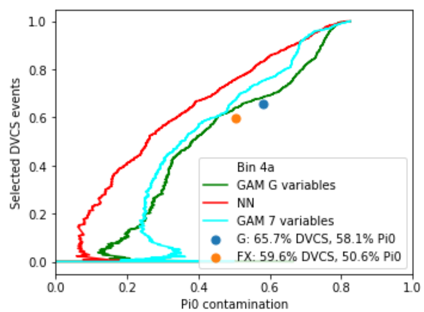


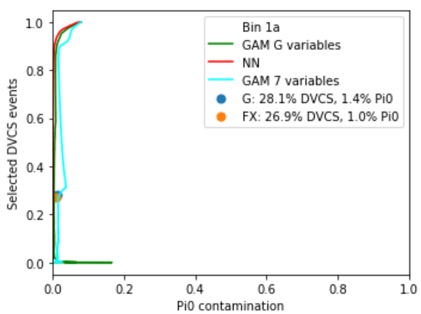


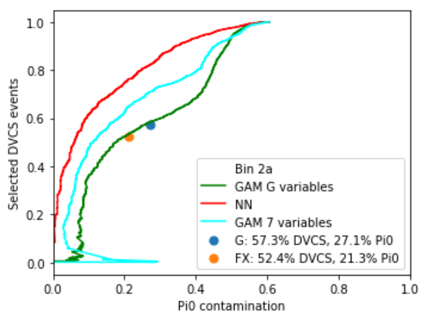












- Neural network always perform better
  Better use of correlations?

- Less significant differences here
  These are the most interesting bins for physics

CONCLUSION

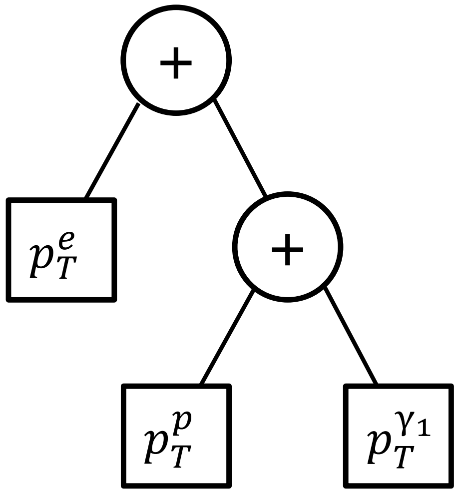
- Analysis of CLAS12 data to select DVCS events
- Feature construction principle: get new discriminative high-level variables
- Implementation in Generalized Additive Models
- Comparison with other analysis methods
- Open questions:
  - ⟶ How to do a fair comparison of the different methods (test data, phase space bins, systematic errors...)?
  - ⟶ How to assess interpretability in an objective manner?
  - ⟶ How to transfer these models on real data?

BACKUP

FEATURE CONSTRUCTION

Motivation: these models do not build a sufficiently complex internal representation of the data

Constrained Genetic Programming: evolve a population of high-level feature candidates



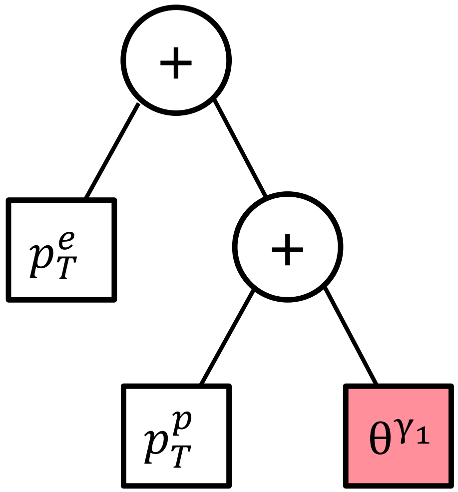
Feature candidate example
→ Nodes are mathematical operators
→ Leaves are base variables

Cherrier, N., Poli, J. P., Defurne, M., & Sabatié, F. (2019, June). Consistent Feature Construction with Constrained Genetic Programming for Experimental Physics. In *2019 IEEE Congress on Evolutionary Computation (CEC)* (pp. 1650-1658). IEEE.

FEATURE CONSTRUCTION

Motivation: these models do not build a sufficiently complex internal representation of the data

Constrained Genetic Programming: evolve a population of high-level feature candidates



Feature candidate example
→ Nodes are mathematical operators
→ Leaves are base variables

Cherrier, N., Poli, J. P., Defurne, M., & Sabatié, F. (2019, June). Consistent Feature Construction with Constrained Genetic Programming for Experimental Physics. In *2019 IEEE Congress on Evolutionary Computation (CEC)* (pp. 1650-1658). IEEE.

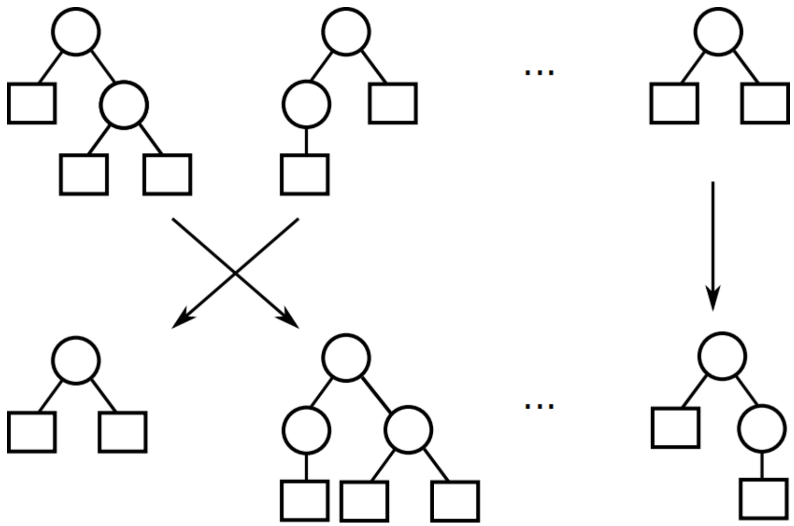
FEATURE CONSTRUCTION

Grammar-guided Genetic Programming

```
<start> ::= <E> | <A> | <F>
<E> ::= <E> + <E> | <E> - <E> | <E> * <F>
       | <E> / <F> | sqrt(<E2>) | <termE>
<A> ::= <A> + <A> | <A> - <A> | acos(<F>)
       | asin(<F>) | atan(<F>) | <termA>
<F> ::= <F> + <F> | <F> - <F> | <F> * <F>
       | <F> / <F> | <E> / <E> | <A> / <A>
       | cos(<A>) | sin(<A>) | tan(<A>)
       | <termF>
<E2> ::= <E2> + <E2> | <E2> - <E2>
       | <E> * <E> | <E2> * <F> | <E2> / <F>
       | square(<E>) | <termE2>
```

Ratle, A., & Sebag, M. (2001). Grammar-guided genetic programming and dimensional consistency: application to non-parametric identification in mechanics. *Applied Soft Computing*, 1(1), 105-118.

FEATURE CONSTRUCTION

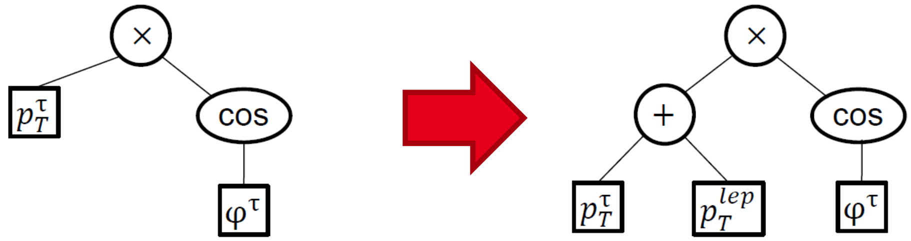


Generation n

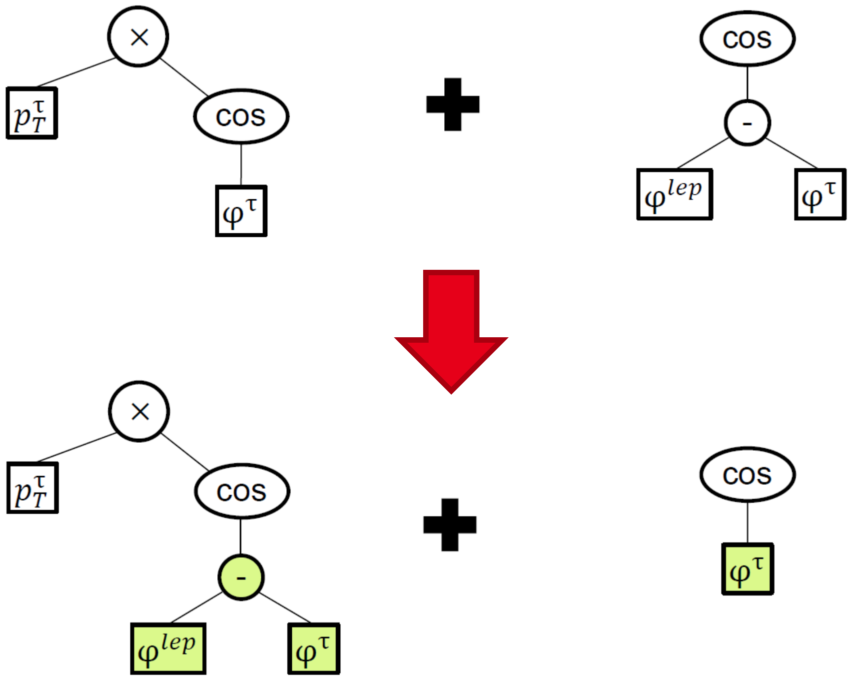
Offspring

FEATURE CONSTRUCTION

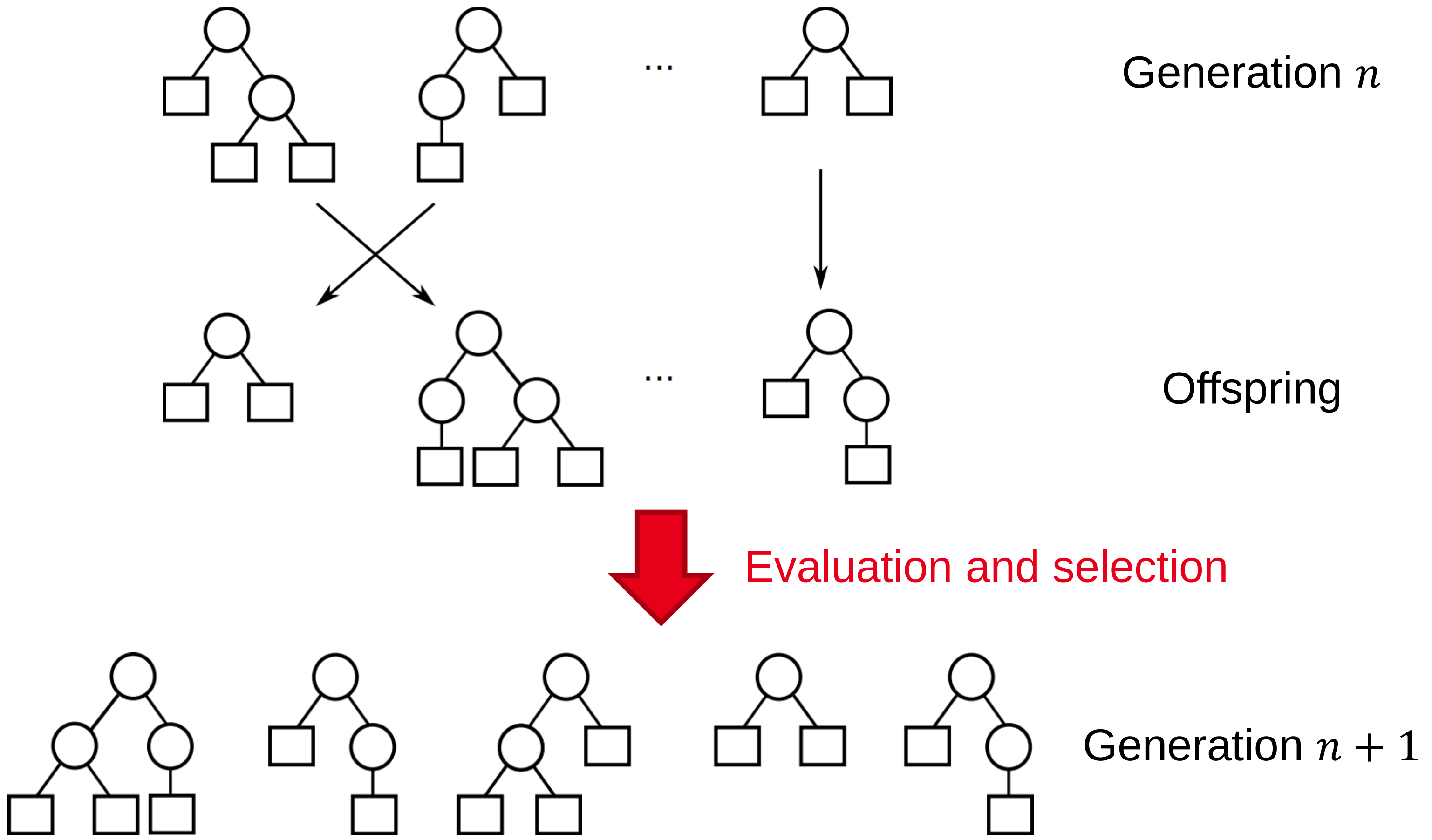
Mutation



Crossover



FEATURE CONSTRUCTION

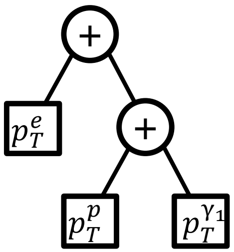


FEATURE CONSTRUCTION

Different FC methods, main difference = how to evaluate the feature candidates

$\underbrace{\text{Filter, wrapper,}}_{}$ or embedded methods

prior FC
(before learning the ML model)



$\longrightarrow \quad p_T^e + p_T^p + p_T^{\gamma_1}$

Filter

Information gain, Gini index, ... of the candidate feature

Wrapper

Inclusion into the initial list:

$p_T^e, \Theta^e, \varphi^e, p_T^p, \Theta^p, \varphi^p$, etc., $p_T^e + p_T^p + p_T^{\gamma_1}$

and training of a ML algorithm
(the fitness of the candidate is the test score of the ML algorithm)

Embedded

Build features during the induction process, usually with filter fitness functions

- *Decision trees and ensemble methods*
- *Generalized Additive Models*

EMBEDDED FEATURE CONSTRUCTION: IN GAM

Idea: build one feature at a time, associated with one term of the GAM

$\longrightarrow$ **gradient boosting**

Objective function: minimize the cross entropy $-y\ln(\hat{y}) - (1-y)\ln(1-\hat{y})$

1) Compute $\beta_0 = \ln\left(\frac{p_0}{1-p_0}\right)$ to form the 1st model $g(\hat{y}) = \beta_0$.
The residual is $r = y - \hat{y} = y - p_0$ (p_0 proportion of the majority class)

2) Build one feature x_1 discriminative wrt the residual

Fitness function for the Genetic Programming algorithm:

- Shallow tree (maximum 4 leaves)
- Feature fitness: RMS error of the inducted tree with the residual

3) Fit a shape function $f_1(x_1)$ to the residual

4) Compute the new model: $g(\hat{y}) = g(\hat{y}) + f_1(x_1)$ and the new residual $r = y - \hat{y}$, and go back to step 2