

Teaching Machine Learning with ATLAS Open Data

Kate Shaw, Tom Stevenson, Caley Yardley

University of Sussex

IOP Open Data Workshop for Higher Education

14 May 2026

Acknowledgements

Andrey Kukhmay (Sussex MSc Data Science)

Iago Rossetto (Sussex Intern, Physics)

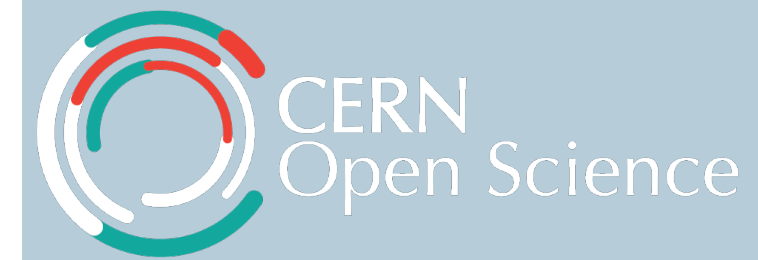
The ATLAS Collaboration, and CERN Open Data Portal

Overview

Open Data Goals



Open science principles underpin all Research activity to ensure it upholds the **UN Sustainable Development Goals** with respect to equality of education and growth.



Experiments at CERN have an obligation to make data **public** and **accessible**.

“ Open research data refers to the publishing of the data underpinning scientific research results so that they have no restrictions on their access and usage. Openly sharing data opens it up to inspection and re-use, forms the basis for research verification and reproducibility, and opens up a path to broader collaboration. ”

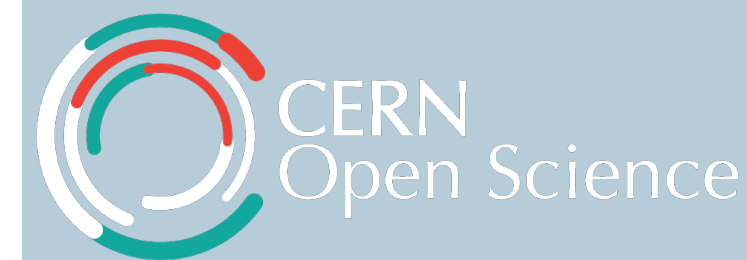
UNESCO Open research data definition

Overview

Open Data Goals



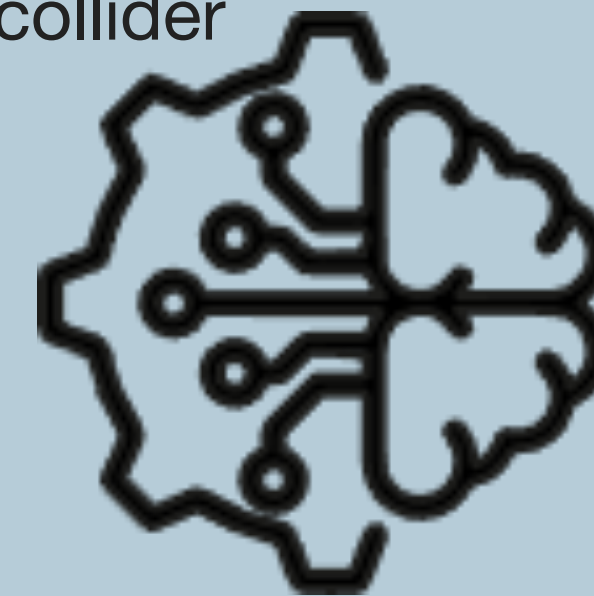
Open science principles underpin all Research activity to ensure it upholds the **UN Sustainable Development Goals** with respect to equality of education and growth.



Experiments at CERN have an obligation to make data **public and accessible**.

Machine Learning

Machine learning in ATLAS is used to search for rare particle events amid large backgrounds, leveraging many variables to detect subtle signals in high-dimensional collider datasets, enabling precise analyses and searches for new physics.



Education

Accessible, multi-level training resources developed to guide students from no prior experience to advanced machine learning, enabling **hands-on exploration** of ATLAS Open Data, skill development, and preparation for physics and data science careers.

“ Open research data refers to the publishing of the data underpinning scientific research results so that they have no restrictions on their access and usage. Openly sharing data opens it up to inspection and re-use, forms the basis for research verification and reproducibility, and opens up a path to broader collaboration. ”

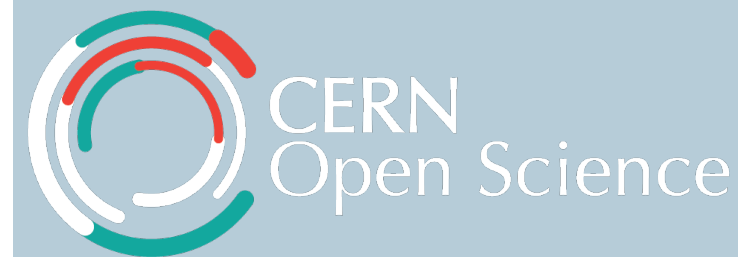
UNESCO Open research data definition

Overview

Open Data Goals



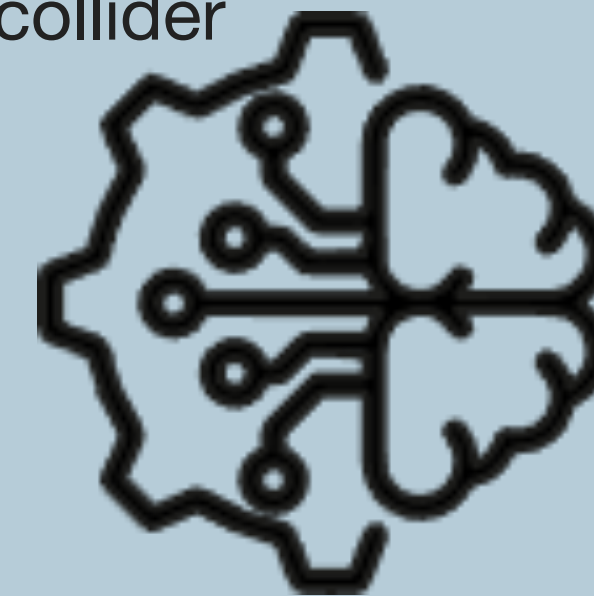
Open science principles underpin all Research activity to ensure it upholds the **UN Sustainable Development Goals** with respect to equality of education and growth.



Experiments at CERN have an obligation to make data **public and accessible**.

Machine Learning

Machine learning in ATLAS is used to search for rare particle events amid large backgrounds, leveraging many variables to detect subtle signals in high-dimensional collider datasets, enabling precise analyses and searches for new physics.



Education

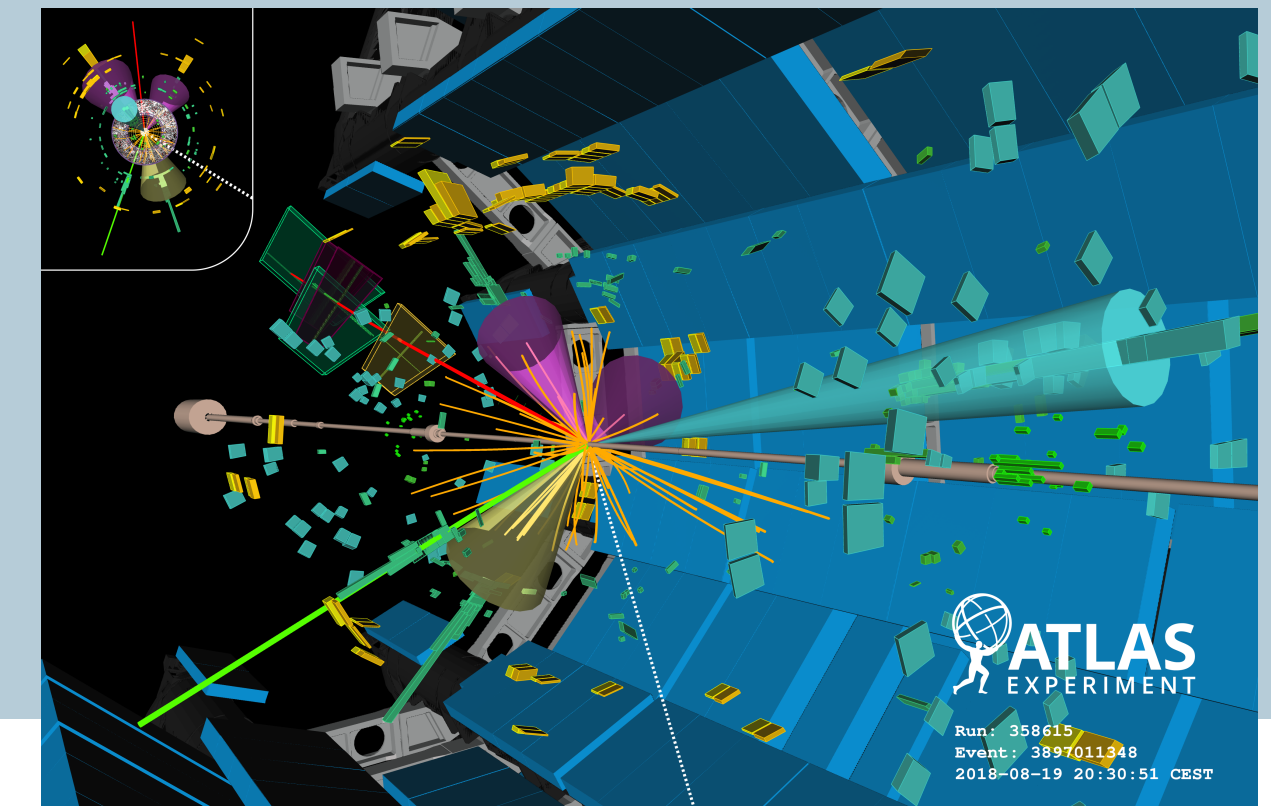
Accessible, multi-level training resources developed to guide students from no prior experience to advanced machine learning, enabling **hands-on exploration** of ATLAS Open Data, skill development, and preparation for physics and data science careers.

ATLAS @ CERN



The ATLAS experiment at CERN's Large Hadron Collider is a particle detector exploring the building blocks of matter & the forces shaping our universe.

By studying proton-proton collisions at unprecedented energies, it probes the Higgs boson, rare processes, and potential new physics, aiming to unlock the Universe's deepest mysteries.



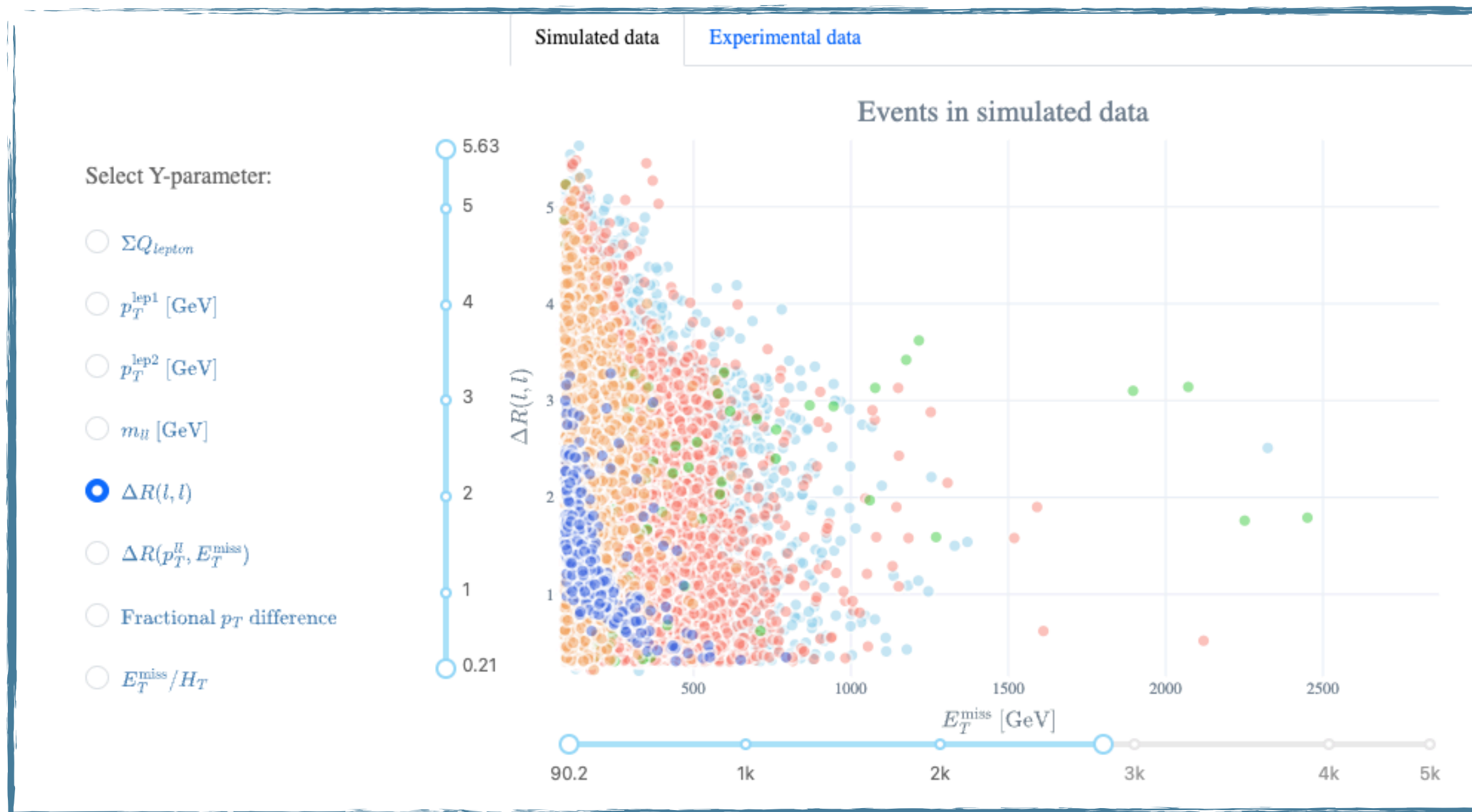
“ Open research data refers to the publishing of the data underpinning scientific research results so that they have no restrictions on their access and usage. Openly sharing data opens it up to inspection and re-use, forms the basis for research verification and reproducibility, and opens up a path to broader collaboration. ”

UNESCO Open research data definition

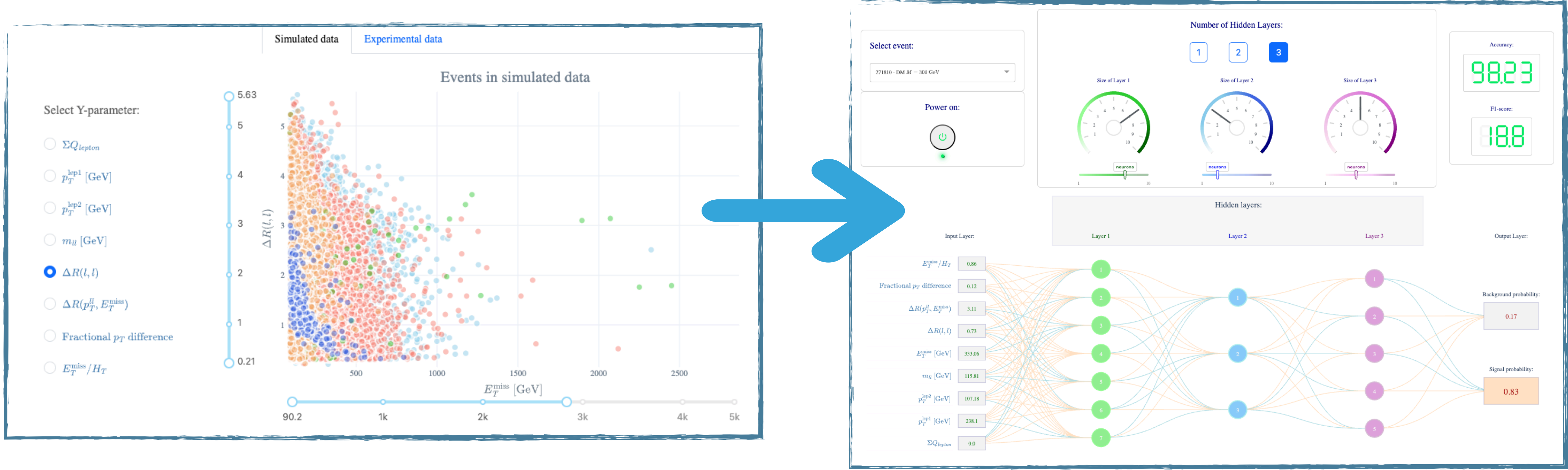
Training Resources and Learning Pathways



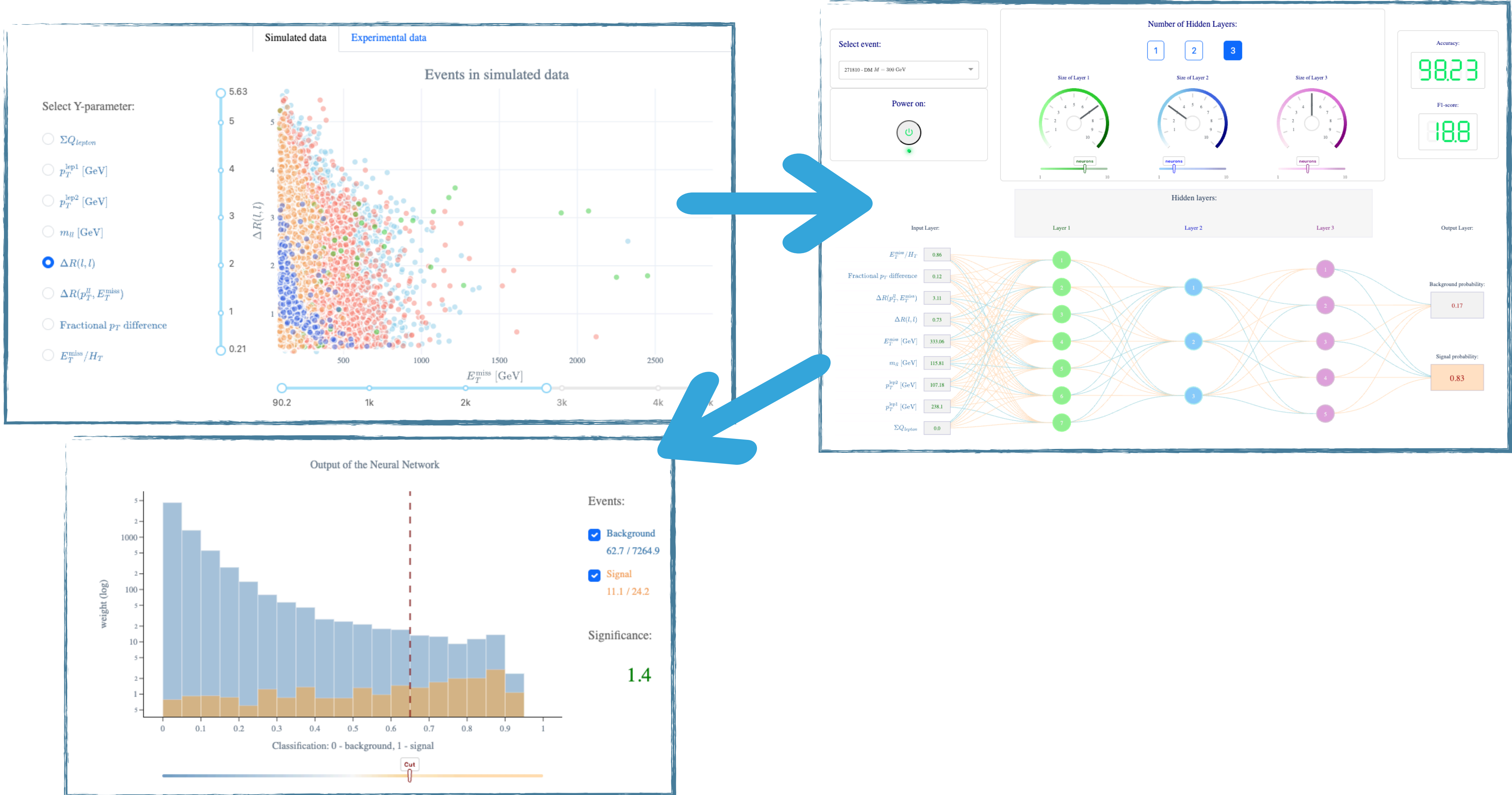
Web-Based Interactive Application for ML



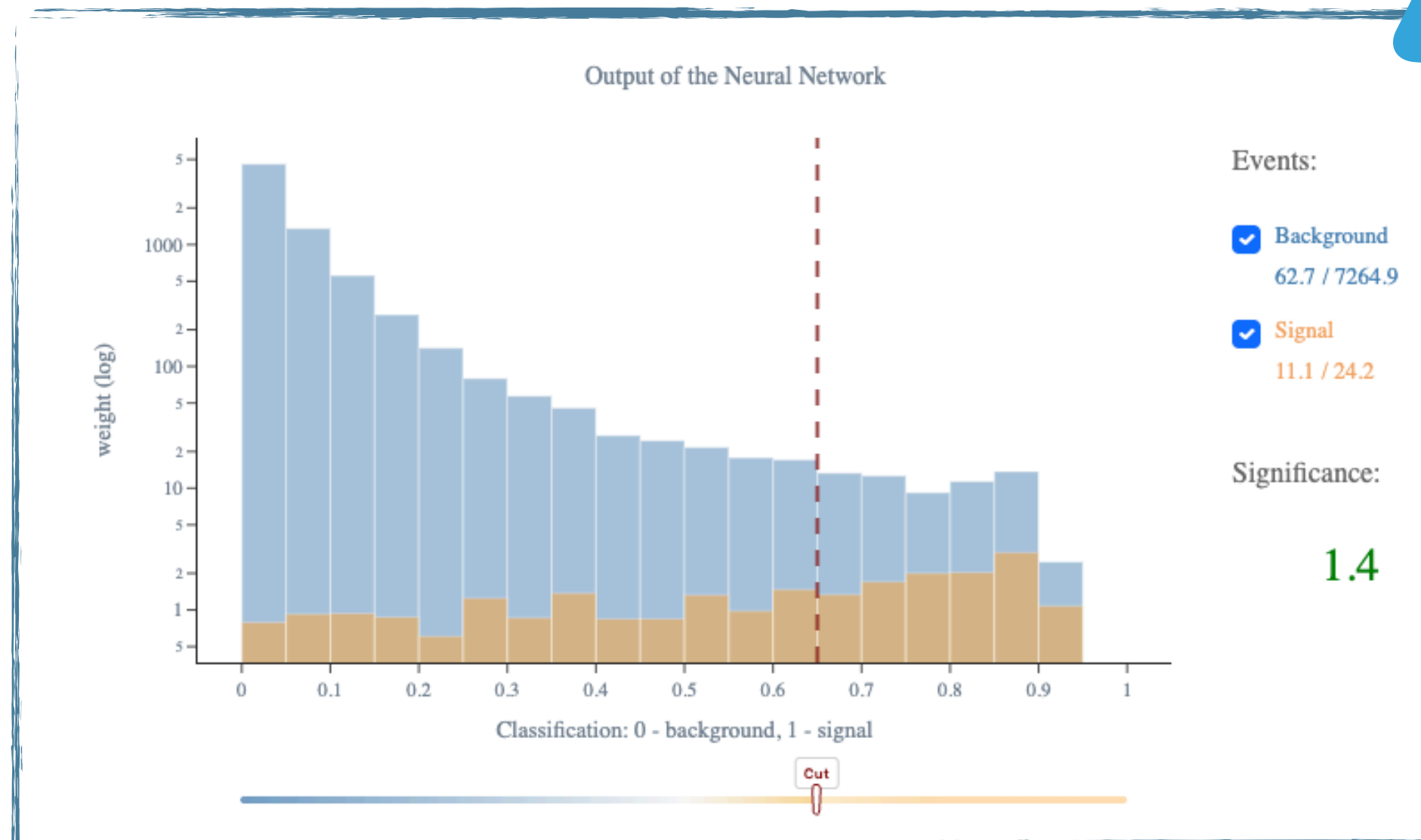
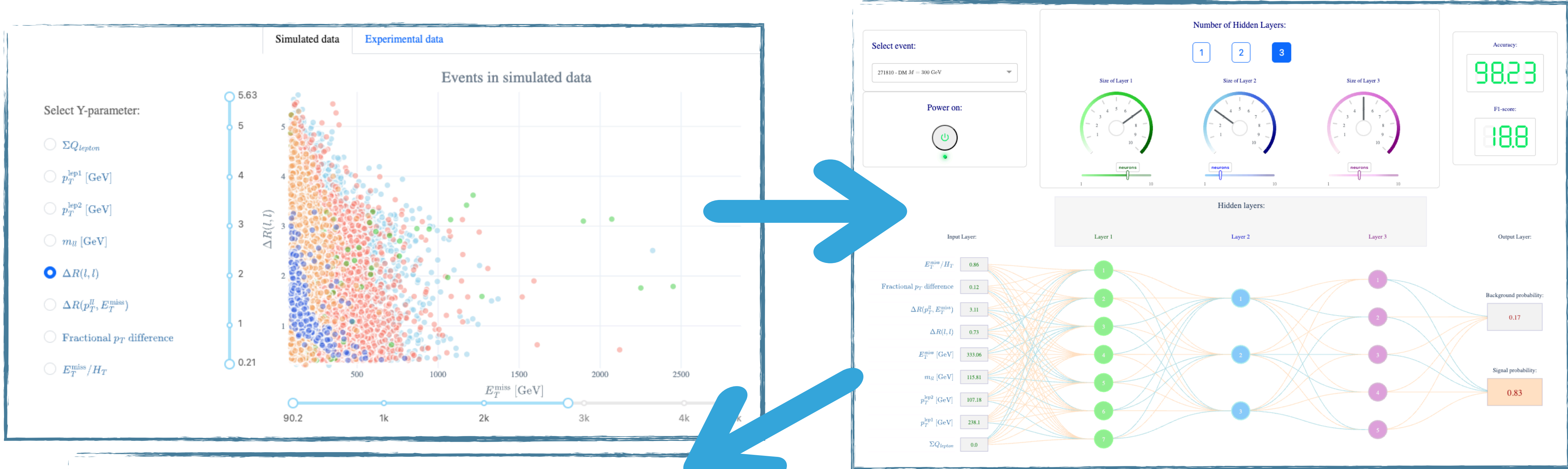
Web-Based Interactive Application for ML



Web-Based Interactive Application for ML



Web-Based Interactive Application for ML



Web-based interactive app provides guided introduction to ML and data analysis, accessible to students with no prior experience.

Demonstrates how ML separates rare signal events from large backgrounds in ATLAS. Users **explore variables** and **adjust neural network hyper-parameters** via point-and-click controls and evaluate performance using metrics and output distributions.

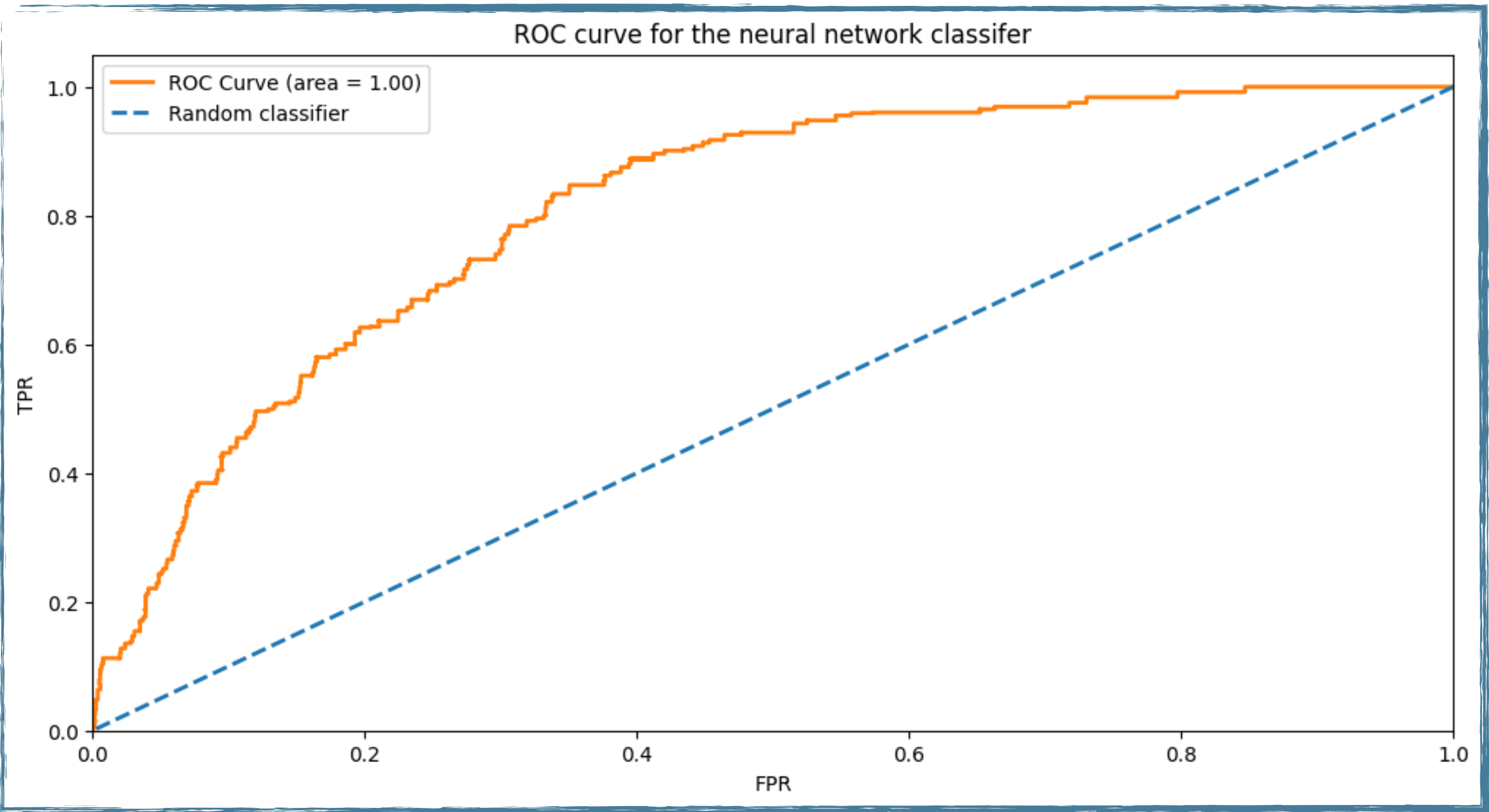
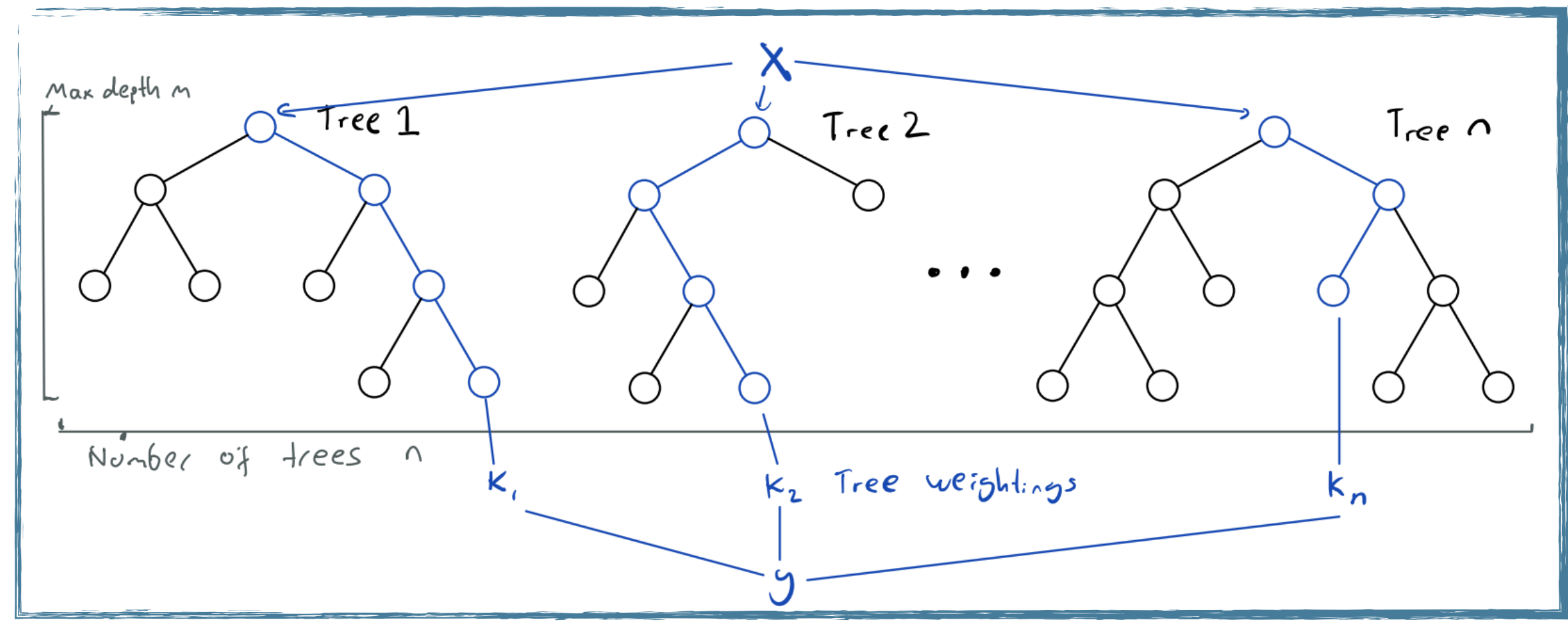
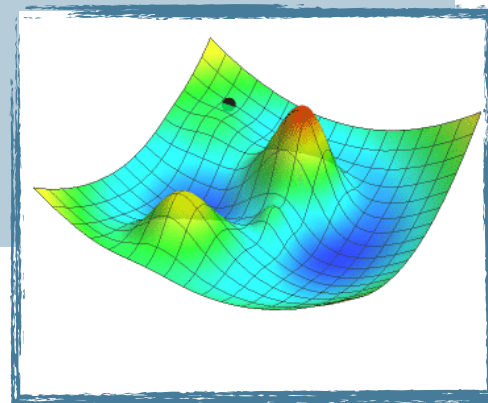
All activities usable **without coding**, offering a practical, engaging, and **research-focused** experience for students and all publics that introduces students to core **ML concepts** and their applications in experimental physics.

Undergraduate Jupyter Notebooks for ML

Jupyter Notebook introduces machine learning fundamentals using Python's **scikit-learn** to create classifiers.

Students actively explore **ROC curves**, **gradient descent**, **random forests**, and **neural networks**, with real ATLAS physics applications.

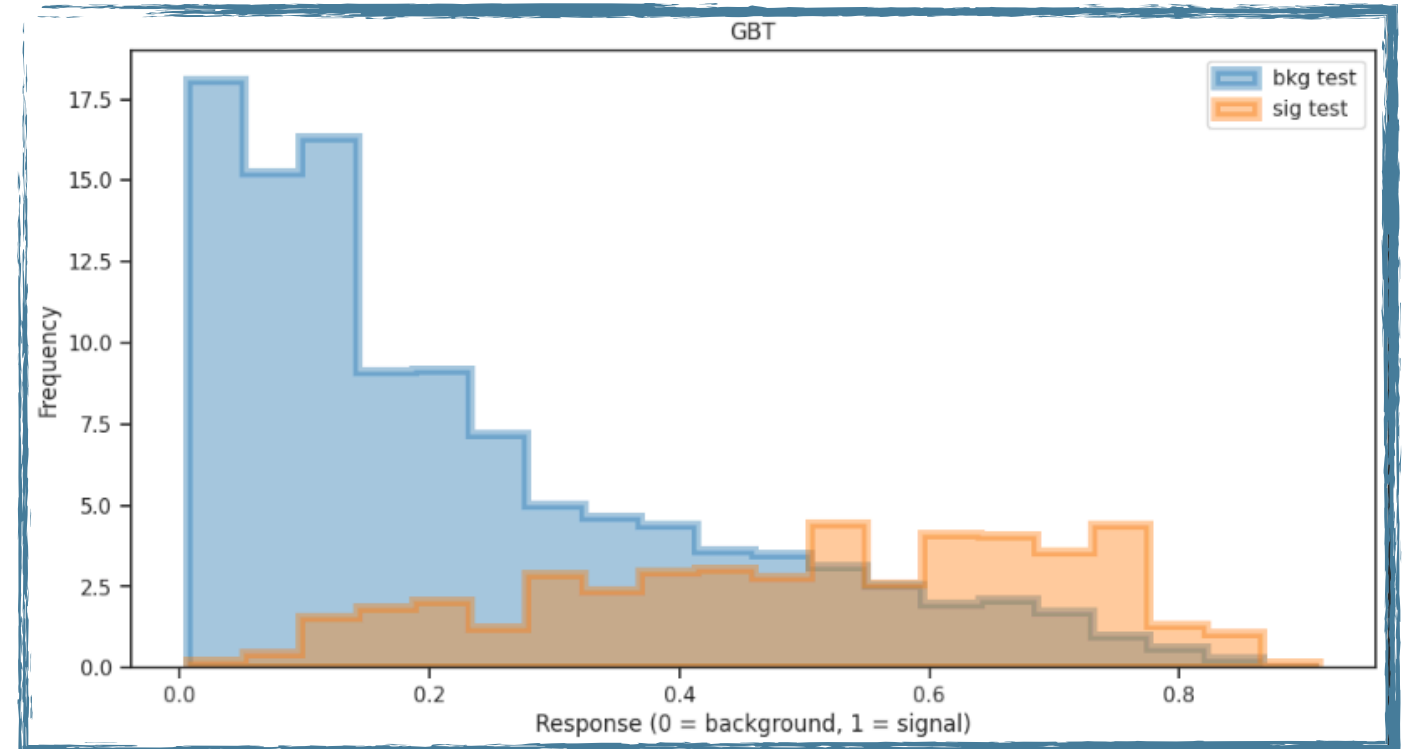
Also learning material about **more advanced libraries** such as PyTorch, TensorFlow, and Keras.



```
In [ ]: from keras import Sequential
        from keras.layers import Input,Dense
        from keras.utils import plot_model
        NN_model = Sequential()
        NN_model.add(Input(shape=(len(ML_inputs),)))
        NN_model.add(Dense(20, activation='relu'))
        NN_model.add(Dense(20, activation='relu'))
        NN_model.add(Dense(1, activation='sigmoid'))

        opt = keras.optimizers.Adam(learning_rate=0.005)

        NN_model.compile(loss='binary_crossentropy', optimizer=opt, metrics=['accuracy'])
```



		Predicted	
		S	B
Actual	S	TP	FN
	B	FP	TN

Advanced / Early Career Jupyter Notebooks

Machine Learning Workbook

```

In [ ]: data_for_ml = {} # define empty dictionary to hold dataframes that will be used to train the NN
# inputs = ['lead_pt', 'Emiss'] # list of features for neural network
# loop over the different keys in the dictionary of dataframes
for key in data: # loop over the different keys in the dictionary of dataframes
    data_for_ml[key] = data[key].copy()
data_for_ml

Organise data ready for the ML

In [ ]: # for sklearn data to usually organized
# into one 2D array of shape (n_samples x n_features)
# containing all the data and one array of categories
# of length n_samples

all_X = [] # define empty list that will contain all features for the ML
for key in data: # loop over the different keys in the dictionary of dataframes
    if key != 'data': # only MC should pass this
        all_X.append(data_for_ml[key]) # append the MC dataframe to the list containing all MC features
# concatenate all MC dataframes into a single 2D array of features, called X
X = pd.concat(all_X)

all_y = [] # define empty list that will contain labels whether an event is signal or background
for key in data: # loop over the different keys in the dictionary of dataframes
    if key == 'signal': # only background MC should pass this
        all_y.append(np.zeros(data_for_ml[key].shape[0])) # background events are labeled with 0
    else: # signal events are labeled with 1
        all_y.append(np.ones(data_for_ml[key].shape[0])) # signal events are labeled with 1
# concatenate the list of labels into a single 1D array of labels, called y
y = np.concatenate(all_y)

The Training and Testing split

One of the first things to do is split your data into a training and testing set. This will split your data into train-test sets: 90%-10%. It will also shuffle entries so you will not get the first 90% of data for training and the last 10% for testing. This is particularly important in cases where you load all signal events first and then the background events.

Here we split our data into two independent samples. The split is to create a training and testing set. The first will be used for training the classifier and the second to evaluate its performance.

We don't want to test on events that we used to train on, this prevents overfitting to some subset of data so the network would be good for the test data but much worse
    
```

[ACCESS WORKBOOK HERE](#)

This online workbook, created by Thomas Stevenson, will introduce the basics of Machine Learning and how it can be exploited in an experimental physics context. Experience with python is required, however no knowledge of ML is necessary. Warning: it may take a long time to load the workbook and associated repositories! A preview is available by clicking on the image.

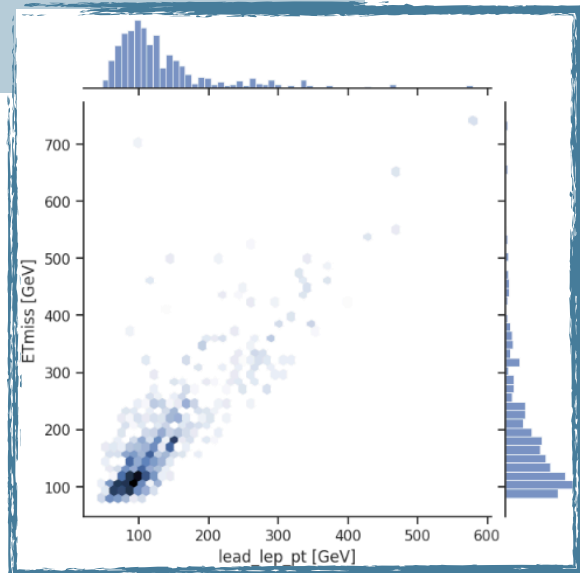
Jupyter Notebook, accessible via Binder, trains students in applied machine learning **using hands-on exercises**, including k-Fold Cross-Validation and ROC curve analysis.

Demystifies common ML terminology, making concepts approachable for Python users with no prior experience.

```

In [ ]: from sklearn.model_selection import ShuffleSplit
ABDT = AdaBoostClassifier(tree.DecisionTreeClassifier(max_depth=6),
                           algorithm="SAMME",
                           n_estimators=20)

cv = ShuffleSplit(n_splits=2, test_size=0.2, random_state=0)
errorVsSize(ABDT, cv, X_train, y_train, 4)
    
```



Advanced / Early Career Jupyter Notebooks

Machine Learning Workbook

```

# Create variables for use in our algorithms
In [ ]: data_for_ml = {} # define empty dictionary to hold dataframes that will be used to train the NN
# Inputs = [200, 100, 100] # list of features for neural network
# For key in data: # loop over the different keys in the dictionary of dataframes
# data_for_ml[key] = data[key].copy()
data_for_ml

Organise data ready for the ML

In [ ]: # for sklearn data is usually organized
# into one 2D array of shape (n_samples, n_features)
# containing all the data and one array of categories
# of length n_samples

all_X = [] # define empty list that will contain all features for the NN
# For key in data: # loop over the different keys in the dictionary of dataframes
# if key in 'data': # only NN should pass this
all_X.append(data_for_ml[key]) # append the NN dataframe to the list containing all NN features
# n = np.concatenate(all_X) # concatenate the list of NN dataframes into a single 2D array of features, called X
X = pd.concat(all_X)

all_y = [] # define empty list that will contain labels whether an event is signal or background
# For key in data: # loop over the different keys in the dictionary of dataframes
# if key in ['signal', 'background']: # only background NN should pass this
all_y.append(np.zeros(data_for_ml[key].shape[0])) # background events are labeled with 0
all_y.append(np.ones(data_for_ml[key].shape[0])) # signal events are labeled with 1
# n = np.concatenate(all_y) # concatenate the list of labels into a single 1D array of labels, called y

The Training and Testing split
One of the first things to do is split your data into a training and testing set. This will split your data into train-test sets: 90%-10%. It will also shuffle entries so you will not get the first 90% of for training and the last 10% for testing. This is particularly important in cases where you load all signal events first and then the background events.
Here we split our data into two independent samples. The split is to create a training and testing set. The first will be used for training the classifier and the second to evaluate its performance.
We don't want to test on events that we used to train on, this prevents overfitting to some subset of data so the network would be good for the test data but much worse
    
```

ACCESS WORKBOOK HERE

This online workbook, created by Thomas Stevenson, will introduce the basics of Machine Learning and how it can be exploited in an experimental physics context. Experience with python is required, however no knowledge of ML is necessary. Warning: it may take a long time to load the workbook and associated repositories! A preview is available by clicking on the image.

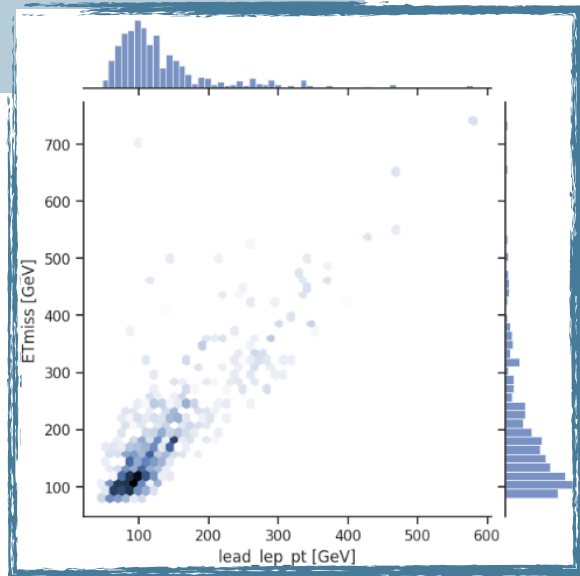
Jupyter Notebook, accessible via Binder, trains students in applied machine learning **using hands-on exercises**, including k-Fold Cross-Validation and ROC curve analysis.

Demystifies common ML terminology, making concepts approachable for Python users with no prior experience.

```

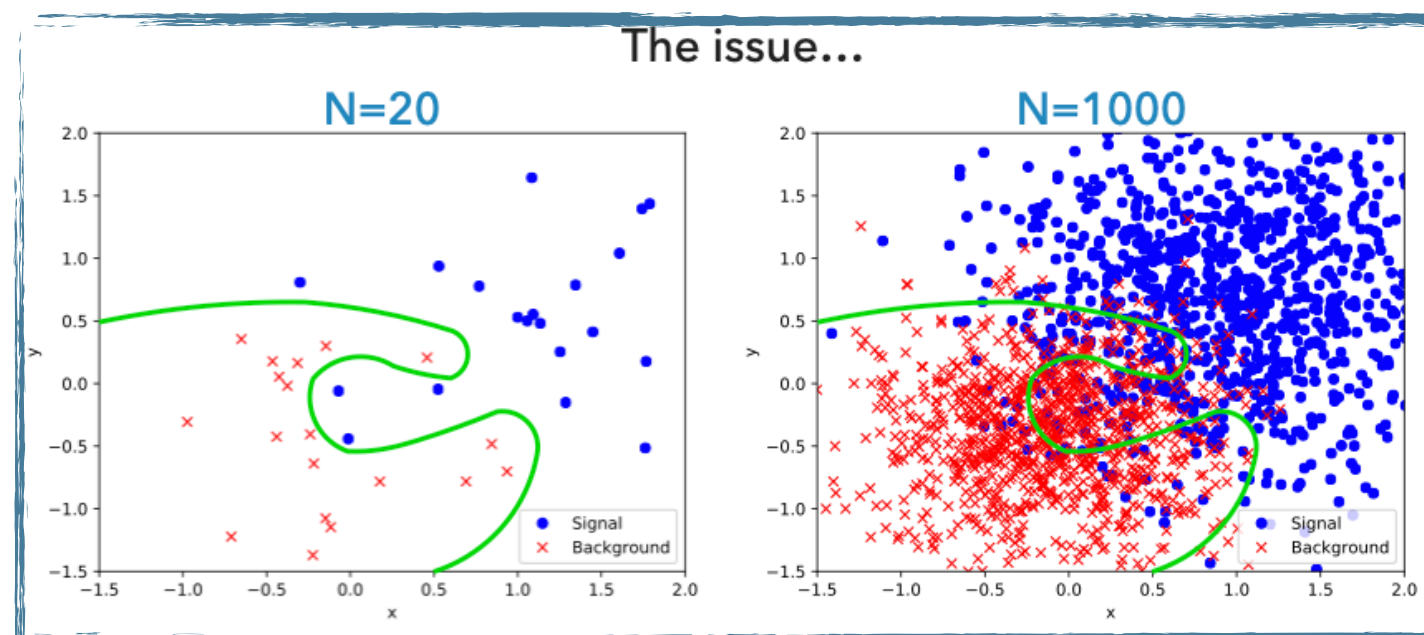
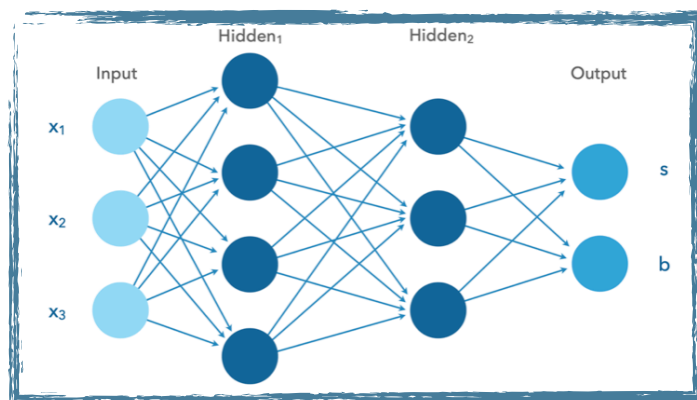
In [ ... ] from sklearn.model_selection import ShuffleSplit
ABDT = AdaBoostClassifier(tree.DecisionTreeClassifier(max_depth=6),
                          algorithm="SAMME",
                          n_estimators=20)

cv = ShuffleSplit(n_splits=2, test_size=0.2, random_state=0)
errorVsSize(ABDT, cv, X_train, y_train, 4)
    
```



An accompanying presentation, available as downloadable slides and an explanatory video, provides additional guidance.

Together, they allow students to **explore concepts** at their own pace, gain practical skills, and **build research-focused expertise** applying machine learning to high-dimensional physics datasets.

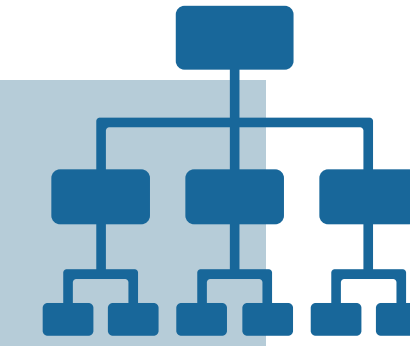


Introduction to Machine Learning using ATLAS Open Data

Dr. Tom Stevenson and Caley Yardley
University of Sussex

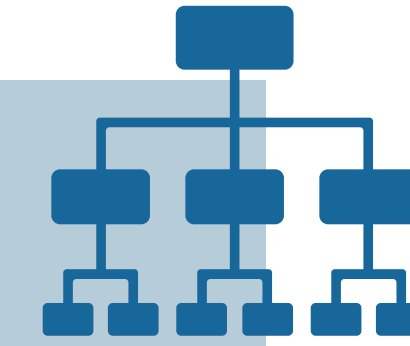
Expand

Expand resources to include advanced ML techniques and more complex ATLAS analyses



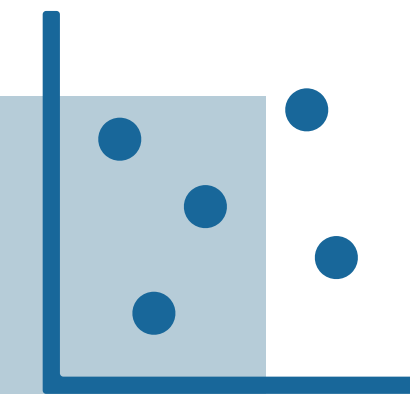
Expand

Expand resources to include advanced ML techniques and more complex ATLAS analyses



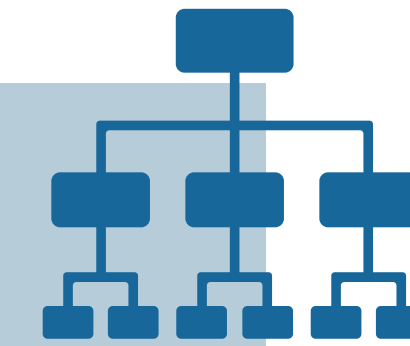
Integrate

Integrate additional LHC datasets to provide broader, real-world data experiences



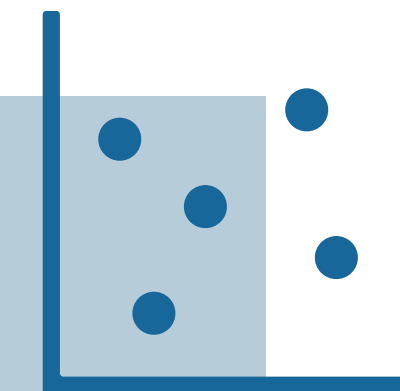
Expand

Expand resources to include advanced ML techniques and more complex ATLAS analyses



Integrate

Integrate additional LHC datasets to provide broader, real-world data experiences



Train

Train the next generation of physicists and data scientists in practical, hands-on high-dimensional data analysis skills



Thanks!

- ◎ **References and links:**

- **ATLAS Collaboration (2020). ATLAS 13 TeV samples collection exactly two leptons (electron or muon), for 2020 Open Data release. CERN Open Data Portal. DOI:[10.7483/OPENDATA.ATLAS.OHZ7.RFNH](https://doi.org/10.7483/OPENDATA.ATLAS.OHZ7.RFNH)**
- <https://opendata.atlas.cern/docs/category/web-apps-for-education>
- <https://github.com/atlas-outreach-data-tools/ml-visual-dashboard>
- https://github.com/atlas-outreach-data-tools/Histogram_Analyser_Dark_Matter