Introduction to Machine Learning: Part III

Prof. Sean Dobbs[1] & Daniel Lersch[2]

April 13, 2021

[1] (sdobbs@fsu.edu)
[2] (dlersch@jlab.org)

About this Lecture

- **Part I:** (Covered by Prof. Dobbs)
  - Basic concepts of machine learning
    (with focus on feedforward neural networks)
  - Data manipulation and visualization with pandas dataframes
  - Training a neural network with scikit
- **Part II:**
  - Overfitting and validation data
  - Gaussian processes
- **Part III:** (Today)
  - Particle Identification
  - Classification Metrics

The individual contents might be subject to change

This Lecture...

... focusses on the basic concepts and ideas behind machine learning

This Lecture...

... focusses on the basic concepts and ideas behind machine learning

... introduces a few machine learning algorithms

This Lecture...

... focusses on the basic concepts and ideas behind machine learning

... introduces a few machine learning algorithms

... aims to familiarize with machine learning jargon / vocabulary

This Lecture...

... focusses on the basic concepts and ideas behind machine learning

... introduces a few machine learning algorithms

... aims to familiarize with machine learning jargon / vocabulary

... does NOT cover all aspects of machine learning (further reading required)

This Lecture...

... focusses on the basic concepts and ideas behind machine learning

... introduces a few machine learning algorithms

... aims to familiarize with machine learning jargon / vocabulary

... does NOT cover all aspects of machine learning (further reading required)

... will NOT turn you into a machine learning specialist

This Lecture...

- ... focusses on the basic concepts and ideas behind machine learning
- ... introduces a few machine learning algorithms
- ... aims to familiarize with machine learning jargon / vocabulary
- ... does NOT cover all aspects of machine learning (further reading required)
- ... will NOT turn you into a machine learning specialist
- ... was held last year in a different format $\rightarrow$ revised material for this edition

This Lecture...

... focusses on the basic concepts and ideas behind machine learning

... introduces a few machine learning algorithms

... aims to familiarize with machine learning jargon / vocabulary

... does NOT cover all aspects of machine learning (further reading required)

... will NOT turn you into a machine learning specialist

... was held last year in a different format $\rightarrow$ revised material for this edition

... mainly utilizes the **scikit-learn** library

This Lecture...

... focusses on the basic concepts and ideas behind machine learning

... introduces a few machine learning algorithms

... aims to familiarize with machine learning jargon / vocabulary

... does NOT cover all aspects of machine learning (further reading required)

... will NOT turn you into a machine learning specialist

... was held last year in a different format $\rightarrow$ revised material for this edition

... mainly utilizes the **scikit-learn** library

... uses **repl.it** for the hands-on sessions

This Lecture...

- ... focusses on the basic concepts and ideas behind machine learning
- ... introduces a few machine learning algorithms
- ... aims to familiarize with machine learning jargon / vocabulary
- ... does NOT cover all aspects of machine learning (further reading required)
- ... will NOT turn you into a machine learning specialist
- ... was held last year in a different format → revised material for this edition
- ... mainly utilizes the **scikit-learn** library
- ... uses **repl.it** for the hands-on sessions
- ... most likely contain several errors (→ Please send a mail to `dlersch@jlab.org`)

Homework and Literature

- Machine learning can be learned best by simply doing it!

Homework and Literature

- Machine learning can be learned best by simply doing it!
- Homework aims to perform a simple analysis and getting familiar with machine learning

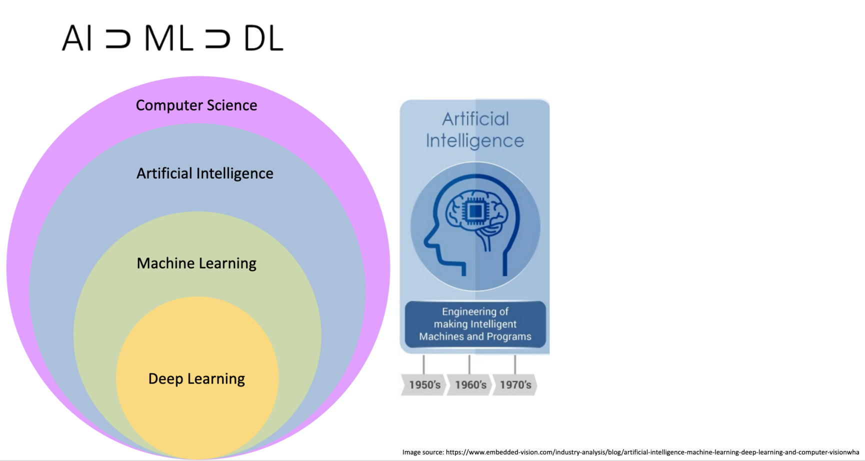
Homework and Literature

- Machine learning can be learned best by simply doing it!
- Homework aims to perform a simple analysis and getting familiar with machine learning
- Helpful literature:
  - The **scikit-learn** documentation
  - Talks from
    - The **deep learning for science school 2020**
    - The **deep learning for science school 2019**[3]
  - **Distill.pub** (many articles about state-of-the-art machine / deep learning)
  - "Hands-On Machine Learning with Scikit-Learn, Keras & Tensorflow", by Aurélien Géron
  - The internet is full of good (but also very bad!) literature[4] → browse with caution
  - Slides and scripts available at: `http://hadron.physics.fsu.edu/~dlersch/Intro_To_ML_2021/`

[3] Very good and detailed explanation of (deep) neural networks

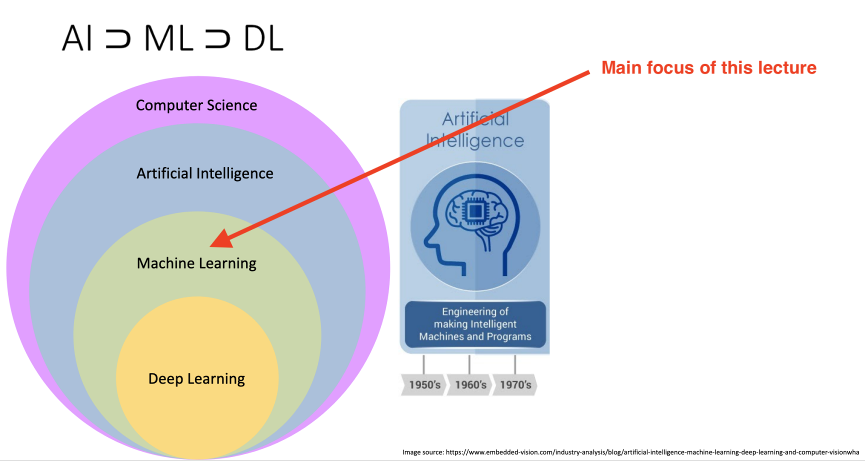
[4] Any document claiming that there is a quick way to understand machine learning without any theory / math is considered as bad

AI, ML and DL



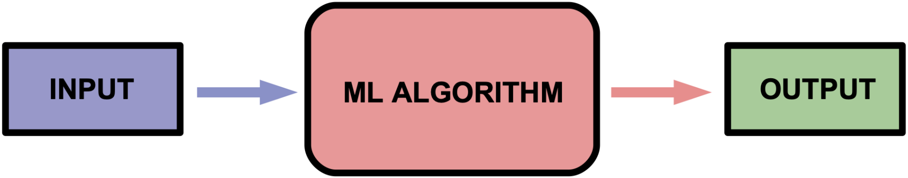
Slide taken from Brenda Ngs introductory talk at the: **deep learning for science school 2019**

AI, ML and DL

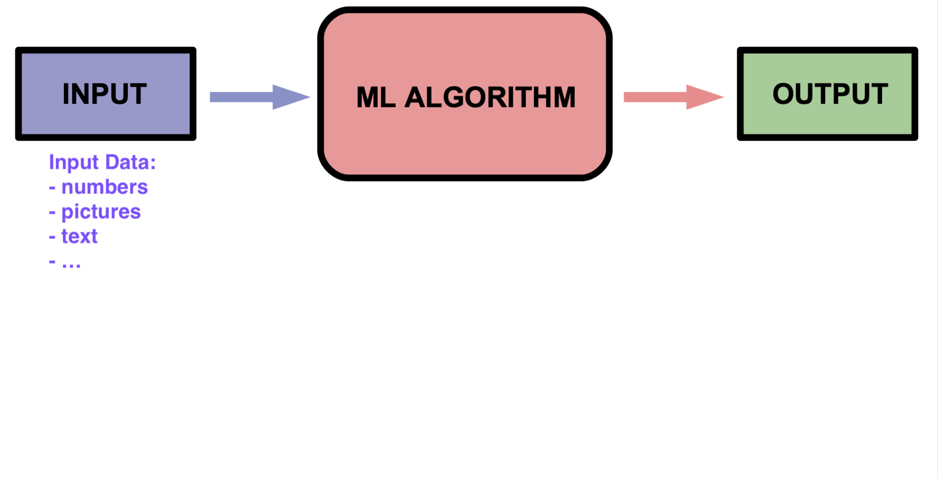


Slide taken from Brenda Ngs introductory talk at the: **deep learning for science school 2019**

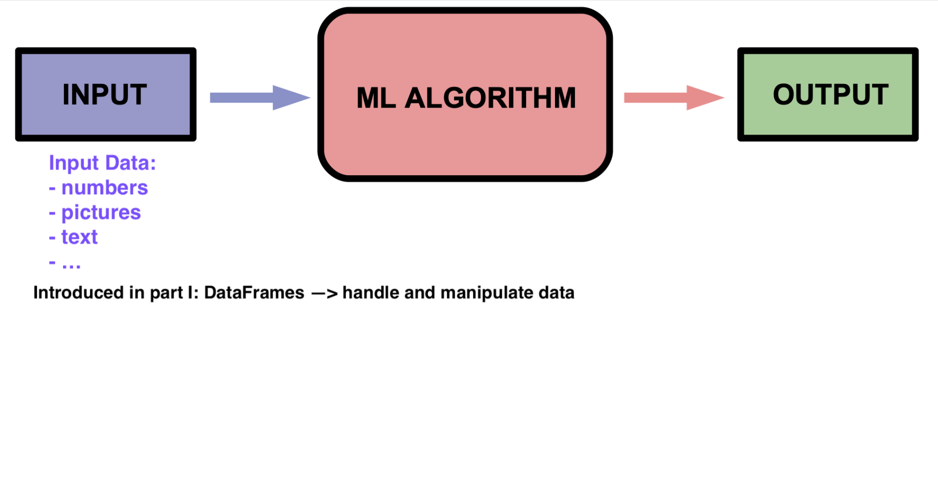
Basic Components of Machine Learning



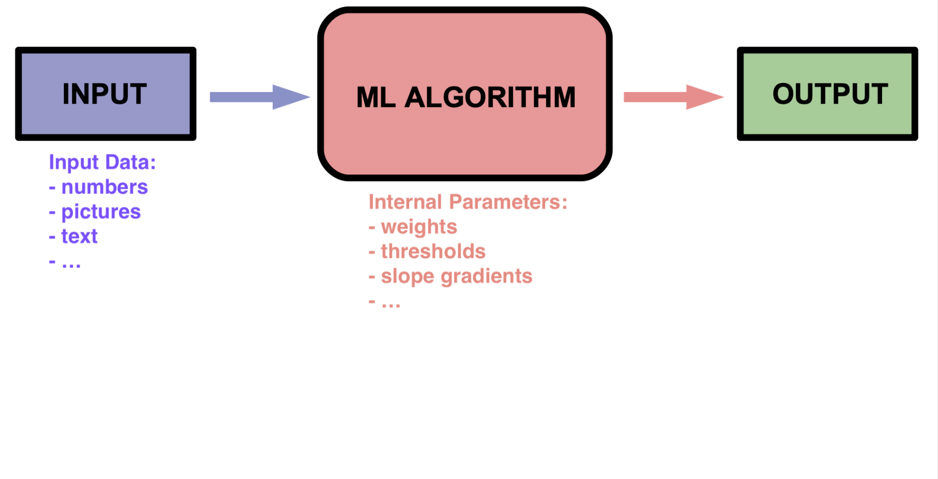
Basic Components of Machine Learning



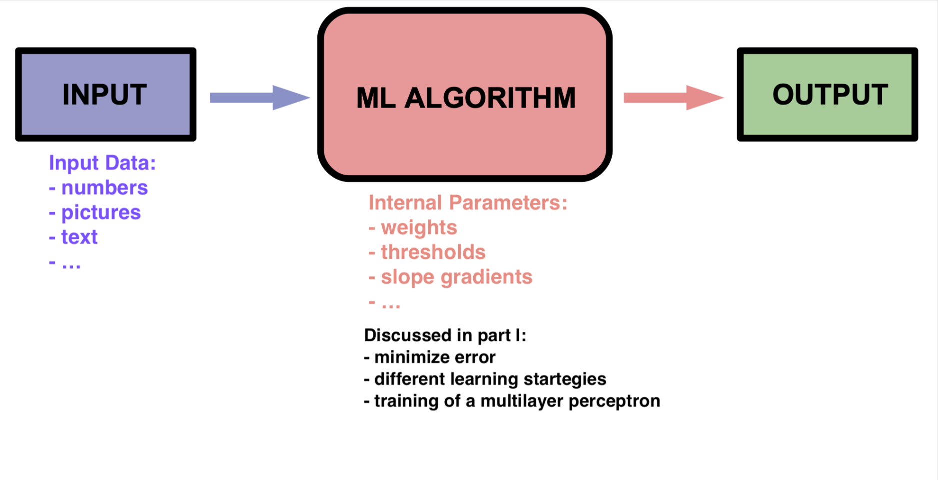
Basic Components of Machine Learning



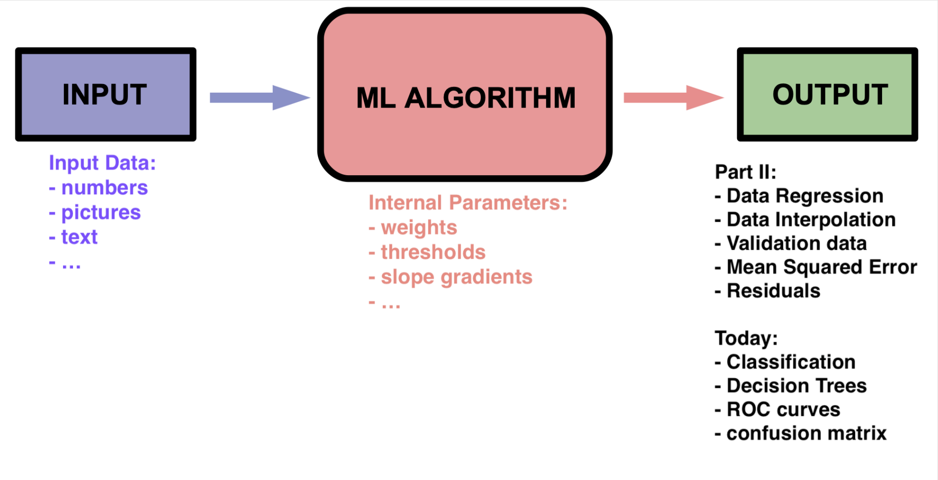
Basic Components of Machine Learning



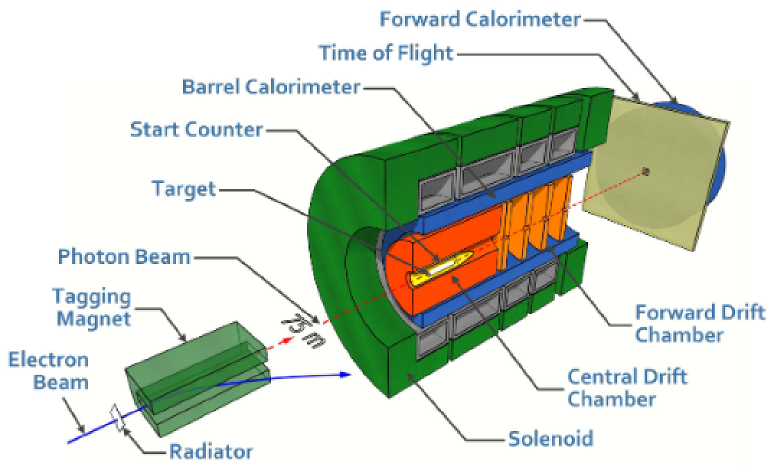
Basic Components of Machine Learning



Basic Components of Machine Learning



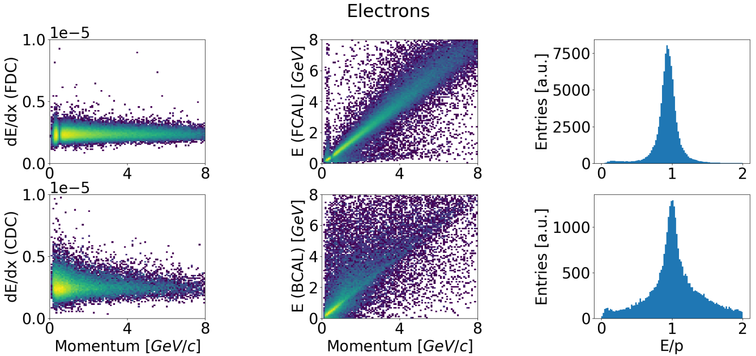
Particle Identification at GlueX



- Many particles produced during GlueX experiments
- **Goal:** Want to classify different particle types $\leftrightarrow$ Particle Identification (PID)
- **Need:** Understand correlations between different sub-detector systems

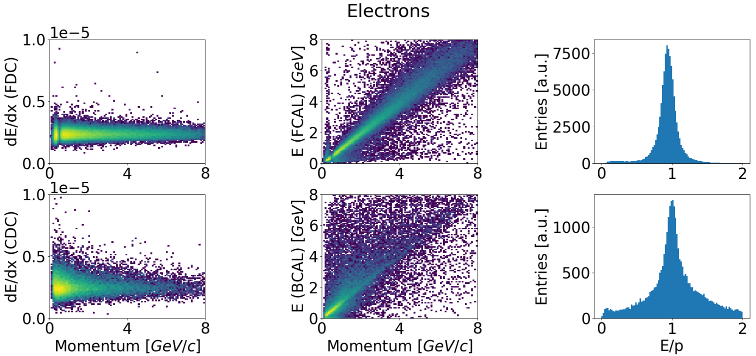
Studying simulated single Particle Tracks

- Simulated $200\,\mathrm{k}$ single $e^{\pm}$ / $\pi^{\pm}$ tracks



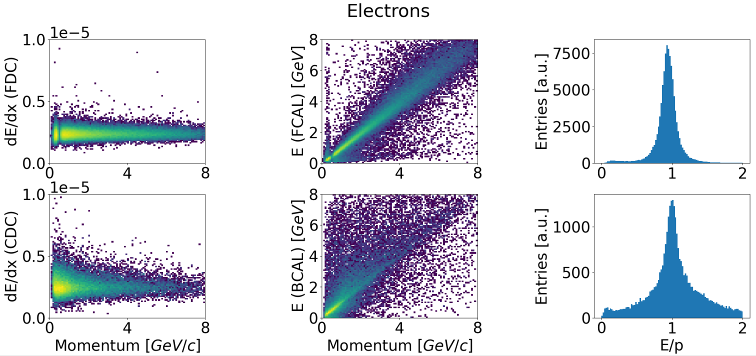
Studying simulated single Particle Tracks

- Simulated 200 k single $e^{\pm}$ / $\pi^{\pm}$ tracks
- Data is balanced: N(electrons) = N(pions)



Studying simulated single Particle Tracks

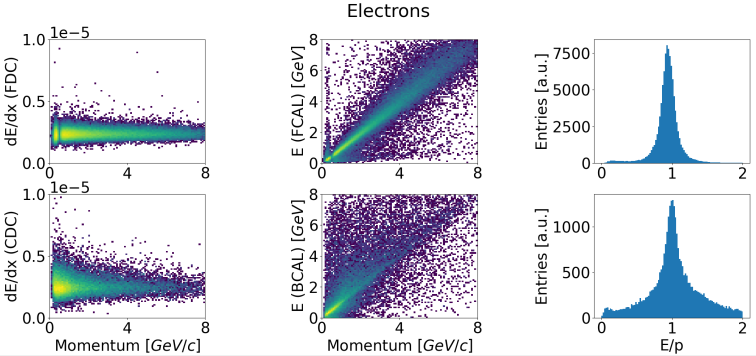
- Simulated 200 k single $e^{\pm}$ / $\pi^{\pm}$ tracks
- Data is balanced: N(electrons) = N(pions)
- Events are labeled: $e^{\pm}$: 1 and $\pi^{\pm}$: 0



Studying simulated single Particle Tracks

- Simulated 200 k single $e^{\pm}$ / $\pi^{\pm}$ tracks
- Data is balanced: N(electrons) = N(pions)
- Events are labeled: $e^{\pm}$: 1 and $\pi^{\pm}$: 0
- Azimuthal angle:

$$\theta < \begin{cases} < 10^\circ \text{ particle detected in GlueX forward part (FDC, FCAL),} \\ \geq 10^\circ \text{ particle detected in GlueX central part (CDC, BCAL)} \end{cases}$$

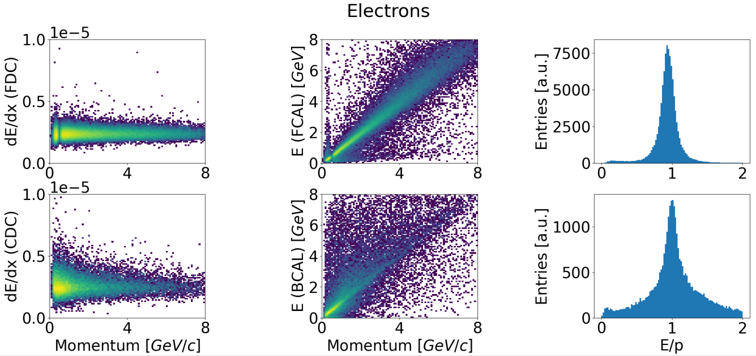


Studying simulated single Particle Tracks

- Simulated 200 k single $e^{\pm}$ / $\pi^{\pm}$ tracks
- Data is balanced: N(electrons) = N(pions)
- Events are labeled: $e^{\pm}$: 1 and $\pi^{\pm}$: 0
- Azimuthal angle:

$$\theta < \begin{cases} < 10^\circ \text{ particle detected in GlueX forward part (FDC, FCAL),} \\ \geq 10^\circ \text{ particle detected in GlueX central part (CDC, BCAL)} \end{cases}$$

- E/p : Energy deposit in calorimeter / Particle momentum

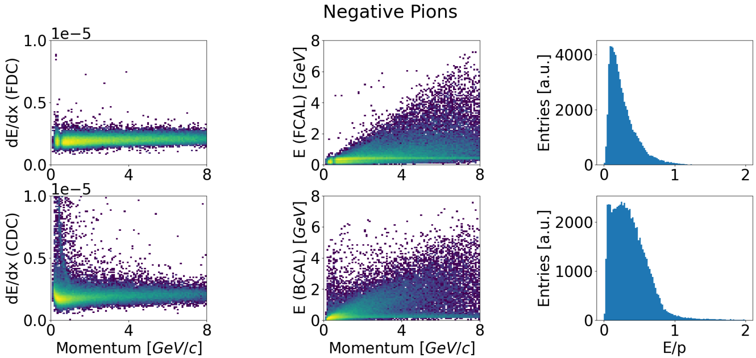


Studying simulated single Particle Tracks

- Simulated 200 k single $e^{\pm}$ / $\pi^{\pm}$ tracks
- Data is balanced: N(electrons) = N(pions)
- Events are labeled: $e^{\pm}$: 1 and $\pi^{\pm}$: 0
- Azimuthal angle:

$$\theta < \begin{cases} < 10^{\circ} \text{ particle detected in GlueX forward part (FDC, FCAL),} \\ \geq 10^{\circ} \text{ particle detected in GlueX central part (CDC, BCAL)} \end{cases}$$

- E/p : Energy deposit in calorimeter / Particle momentum

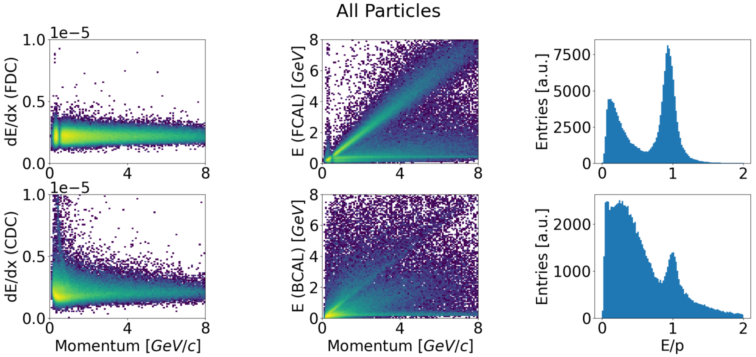


Studying simulated single Particle Tracks

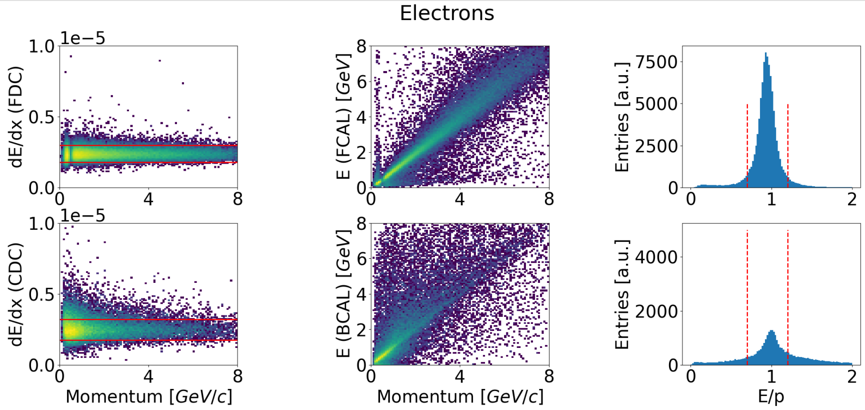
- Simulated $200\,\mathrm{k}$ single $e^{\pm}$ / $\pi^{\pm}$ tracks
- Data is balanced: N(electrons) = N(pions)
- Events are labeled: $e^{\pm}$: 1 and $\pi^{\pm}$: 0
- Azimuthal angle:

$$\theta < \begin{cases} < 10^{\circ} \text{ particle detected in GlueX forward part (FDC, FCAL),} \\ \geq 10^{\circ} \text{ particle detected in GlueX central part (CDC, BCAL)} \end{cases}$$

- E/p : Energy deposit in calorimeter / Particle momentum

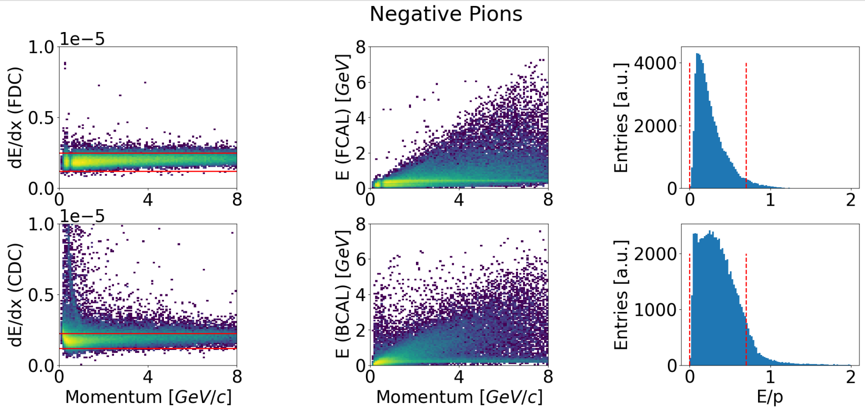


Crude Electron Selection



- Solid / dashed red lines indicate event selection
- Event is labeled $\begin{cases} 1 : \text{Event passes selection criteria,} \\ 0 : \text{Event does NOT pass selection criteria} \end{cases}$

Crude Pion Selection



- Solid / dashed red lines indicate event selection
- Event is labeled $\begin{cases} 0: \text{Event passes selection criteria,} \\ 1: \text{Event does NOT pass selection criteria} \end{cases}$

Performance Evaluation

- How do we know if our event selection was good / bad?

Performance Evaluation

- How do we know if our event selection was good / bad?
- Luckily, our data is labeled

Performance Evaluation

- How do we know if our event selection was good / bad?
- Luckily, our data is labeled
- Compare true labels with those from our event selection

Performance Evaluation

- How do we know if our event selection was good / bad?
- Luckily, our data is labeled
- Compare true labels with those from our event selection
- **Ideally:** True and predicted labels are identical

Performance Evaluation

- How do we know if our event selection was good / bad?
- Luckily, our data is labeled
- Compare true labels with those from our event selection
- **Ideally:** True and predicted labels are identical
- **Define:**

i) True Positive Rate $= \frac{\#\text{Events CORRECTLY labeled as 1 (0)}}{\#\text{Events truly labeled as 1 (0)}}$

ii) False Positive Rate $= \frac{\#\text{Events FALSELY labeled as 1 (0)}}{\#\text{Events truly labeled as 0 (1)}}$

Performance Evaluation

- How do we know if our event selection was good / bad?
- Luckily, our data is labeled
- Compare true labels with those from our event selection
- **Ideally:** True and predicted labels are identical
- **Define:**

i) True Positive Rate $= \frac{\text{\#Events CORRECTLY labeled as 1 (0)}}{\text{\#Events truly labeled as 1 (0)}}$

ii) False Positive Rate $= \frac{\text{\#Events FALSELY labeled as 1 (0)}}{\text{\#Events truly labeled as 0 (1)}}$

- These are the building blocks for nearly all classification performance metrics

Identification Rates

Particle	True Positive Rate (TPR)	False Positive Rate (FPR)
Electrons	0.85	0.11
Negative Pions	0.89	0.15

- **Ideally:** TPR $= 1.0$ and FPR $= 0.0$
- TPR(Electron / Pion) $+$ FPR(Pion / Electron) $= 1.0$
  $\Rightarrow$ Number of particles is conserved!

Identification Rates

Particle	True Positive Rate (TPR)	False Positive Rate (FPR)
Electrons	0.85	0.11
Negative Pions	0.89	0.15

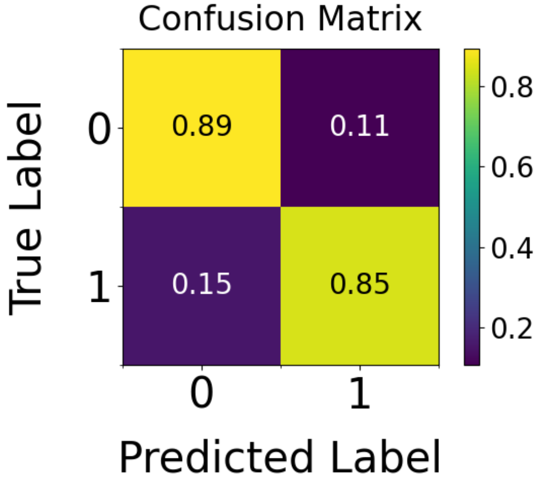
- **Ideally:** TPR = 1.0 and FPR = 0.0
- TPR(Electron / Pion) + FPR(Pion / Electron) = 1.0
  $\Rightarrow$ Number of particles is conserved!

Identification Rates

Particle	True Positive Rate (TPR)	False Positive Rate (FPR)
Electrons	0.85	0.11
Negative Pions	0.89	0.15

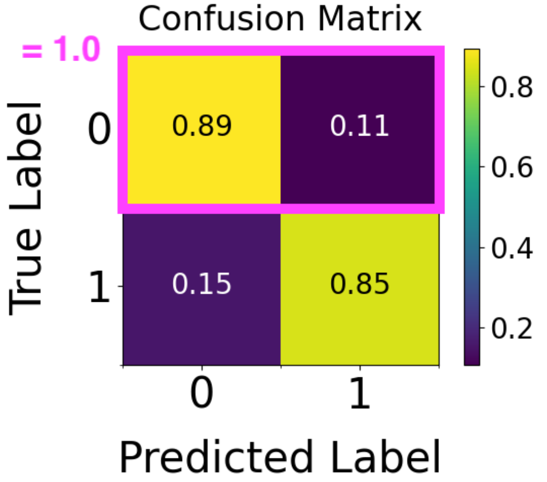
- **Ideally:** TPR = 1.0 and FPR = 0.0
- TPR(Electron / Pion) + FPR(Pion / Electron) = 1.0
  $\Rightarrow$ Number of particles is conserved!

The Confusion Matrix



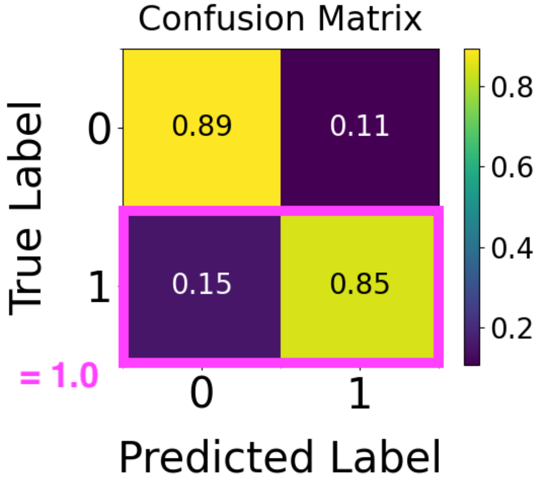
- The confusion matrix is another way to display identification rates
- True positive rates are shown along the diagonal
- **Note:** No rule for which axis holds true / predicted labels

The Confusion Matrix



- The confusion matrix is another way to display identification rates
- True positive rates are shown along the diagonal
- **Note:** No rule for which axis holds true / predicted labels

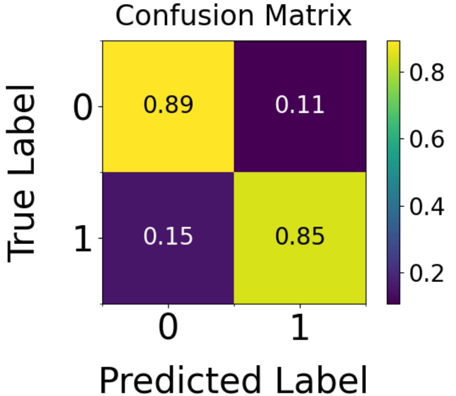
The Confusion Matrix



- The confusion matrix is another way to display identification rates
- True positive rates are shown along the diagonal
- **Note:** No rule for which axis holds true / predicted labels

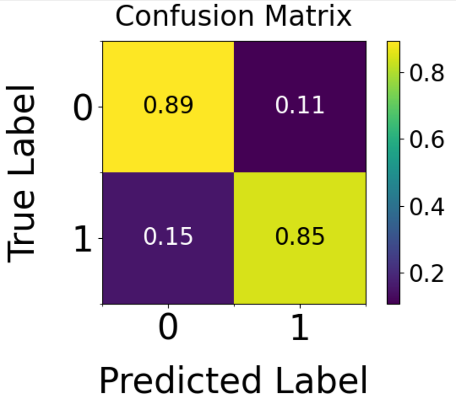
The Accuracy

- Many performance metrics can be (directly) derived from the confusion matrix



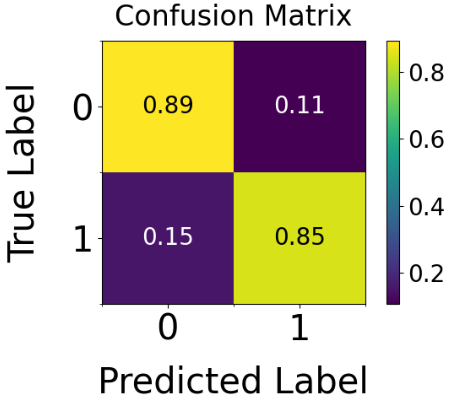
The Accuracy

- Many performance metrics can be (directly) derived from the confusion matrix
- **Example: Accuracy** $= \sum_{i}^{\#species} FPR(i) \cdot R(i)$



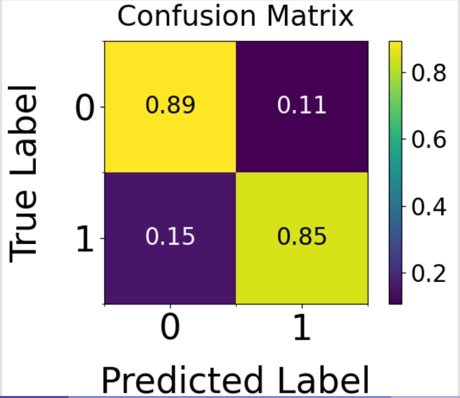
The Accuracy

- Many performance metrics can be (directly) derived from the confusion matrix
- **Example: Accuracy** $= \sum_{i}^{\#species} FPR(i) \cdot R(i)$
- R(i) denotes abundance of each particle species in the data



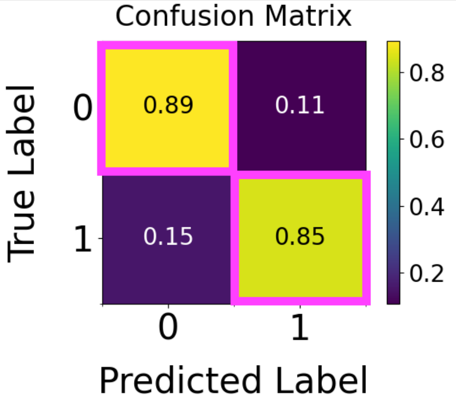
The Accuracy

- Many performance metrics can be (directly) derived from the confusion matrix
- **Example: Accuracy** $= \sum_{i}^{\#species} FPR(i) \cdot R(i)$
- R(i) denotes abundance of each particle species in the data
- Balanced data (each species equally represented): R(i) = 1 / number species



The Accuracy

- Many performance metrics can be (directly) derived from the confusion matrix
- **Example: Accuracy** $= \sum_{i}^{\#species} FPR(i) \cdot R(i)$
- R(i) denotes abundance of each particle species in the data
- Balanced data (each species equally represented): R(i) = 1 / number species
- **Here: R(i) = 0.5 and Accuracy** $= 0.89 \cdot 0.5 + 0.85 \cdot 0.5 = 0.87$



Optimizing the Identification Performance

- Obtained somewhat good PID performance $\rightarrow$ Want to do better

Optimizing the Identification Performance

- Obtained somewhat good PID performance $\rightarrow$ Want to do better
- Used six different particle features:
  1.) Is particle detected in forward / central part of GlueX $\equiv \theta$
  2.) Particle Momentum
  3.) Information from forward drift chamber (FDC)
  4.) Information from forward calorimeter (FCAL)
  5.) Information from central drift chamber (CDC)
  6.) Information from central calorimeter (BCAL)

Optimizing the Identification Performance

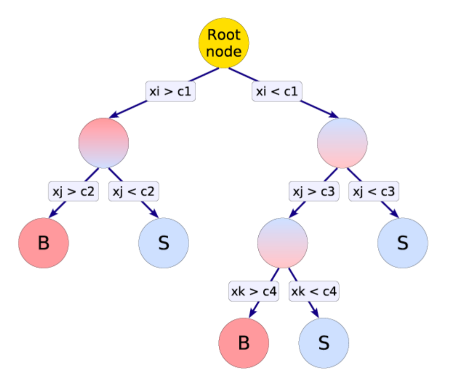
- Obtained somewhat good PID performance $\rightarrow$ Want to do better
- Used six different particle features:
  1.) Is particle detected in forward / central part of GlueX $\equiv \theta$
  2.) Particle Momentum
  3.) Information from forward drift chamber (FDC)
  4.) Information from forward calorimeter (FCAL)
  5.) Information from central drift chamber (CDC)
  6.) Information from central calorimeter (BCAL)
- Could tune each feature by hand...
  ... sometimes necessary
  ... time consuming
  ... can be too complex

Optimizing the Identification Performance

- Obtained somewhat good PID performance $\rightarrow$ Want to do better
- Used six different particle features:
  1.) Is particle detected in forward / central part of GlueX $\equiv \theta$
  2.) Particle Momentum
  3.) Information from forward drift chamber (FDC)
  4.) Information from forward calorimeter (FCAL)
  5.) Information from central drift chamber (CDC)
  6.) Information from central calorimeter (BCAL)
- Could tune each feature by hand...
  ... sometimes necessary
  ... time consuming
  ... can be too complex
- Use a machine learning algorithm
  - Designed to handle a multidimensional feature space
  - Able to "see" correlations that we might miss

A Decision Tree

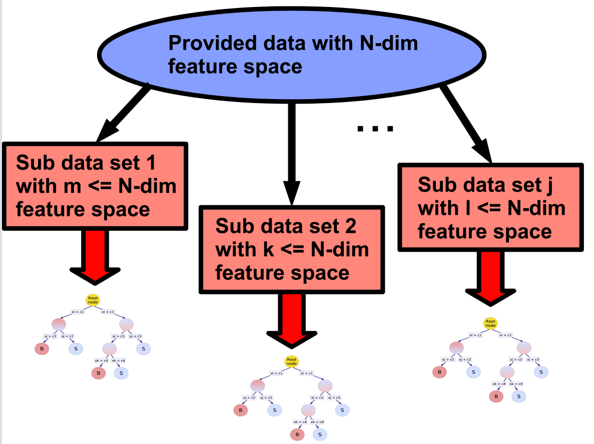
- Already know the multilayer perceptron neural network
- Decision tree is an other machine learning algorithm



- **Basic Idea:**
  i) Create sub-nodes to include different features
  ii) Tune node thresholds until maximum separation / purity is achieved

Many Trees make a Forest

- An ensemble of decision trees defines a random forest
- Each decision tree...
  - ... has its own set of features
  - ... is trained on an individual data set (e.g. bootstrapping)
- Combine predictions of all trees to one output for the random forest



DIY: Single Track Analysis and Algorithm Training

1.) Go to: https://replit.com/@daniel49/FSUMLLecture3

2.) Klick on the **Fork** button

3.) Sign in or log in with your credentials (repl is free)

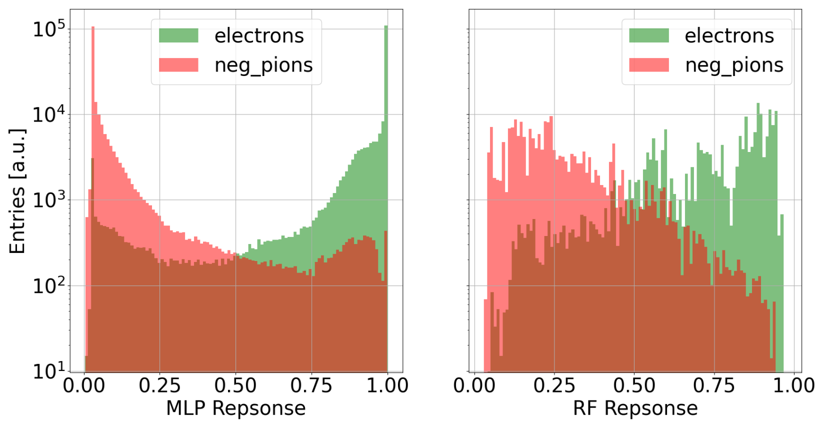
4.) Follow instructions in **main.py**

   NOTE: The data you are able to analyze on repl, is just a sub-set ($\sim 20\,\mathrm{k}$ events) of the data presented here ($\sim 400\,\mathrm{k}$ events) $\rightarrow$ However, both data sets (the one used here and the sub-sample) are available at:
   http://hadron.physics.fsu.edu/~dlersch/Intro_To_ML_2021/data/

Analyze single Track Data with a Neural Network and a random Forest Classifier

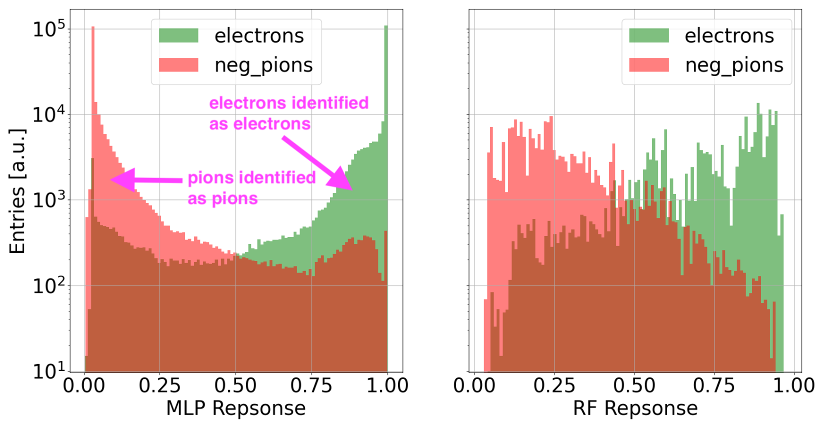
- Both algorithms available via scikit package
- Train each algorithm to separate electrons from pions in single track data
- Use 6 features for each classifier:
    1. Momentum p
    2. Azimuthal angle θ
    3. ddEdx(FDC)
    4. E(FCAL)
    5. ddEdx(CDC)
    6. E(BCAL)
- **About the neural network:**
    - Two hidden layers with 8 / 5 neurons each
    - Trained for 200 epochs
    - Validation data used ($\sim$ 50%)
- **About the random forest:**
    - 10 decision trees
    - Maximum depth of 6
- Which algorithm is better?

The Classifier Response



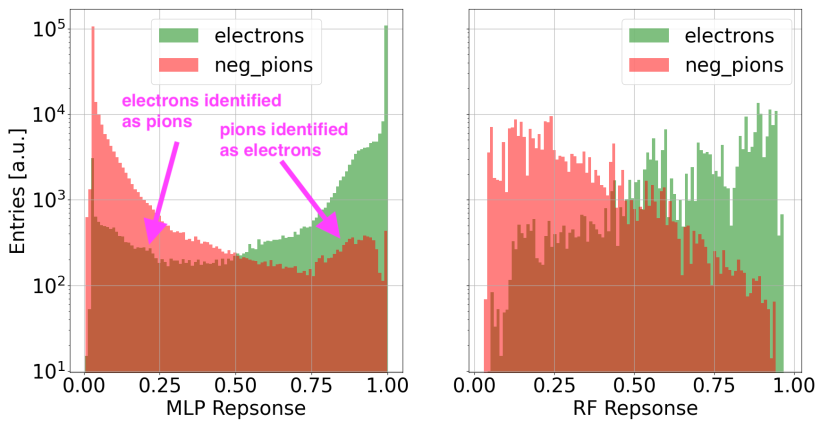
- **Very first thing to check!**
- Does the response function "make sense" ?
  - Neural network response looks "reasonable"
  - Random forest response indicates a bad performance

The Classifier Response



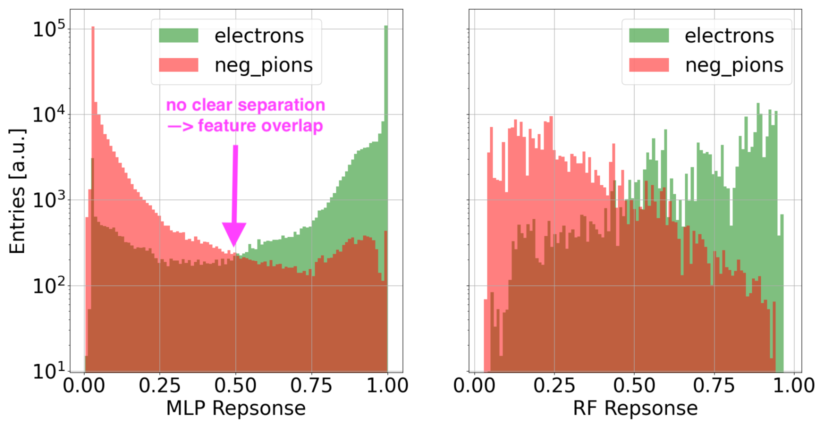
- **Very first thing to check!**
- Does the response function "make sense" ?
  - Neural network response looks "reasonable"
  - Random forest response indicates a bad performance

The Classifier Response



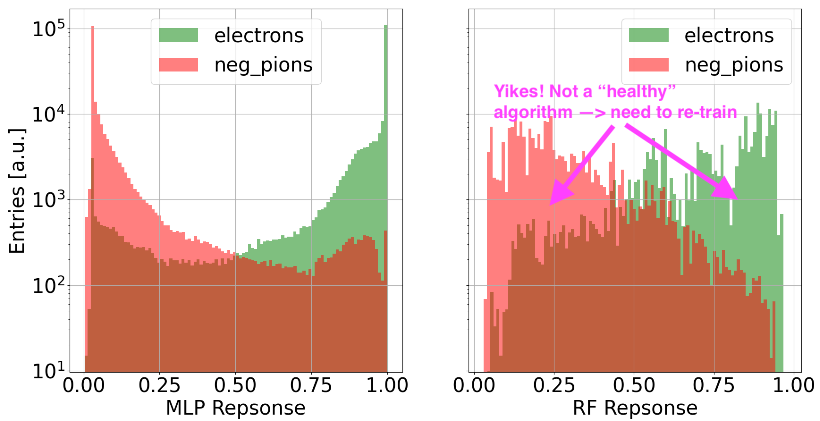
- **Very first thing to check!**
- Does the response function "make sense" ?
  - Neural network response looks "reasonable"
  - Random forest response indicates a bad performance

The Classifier Response



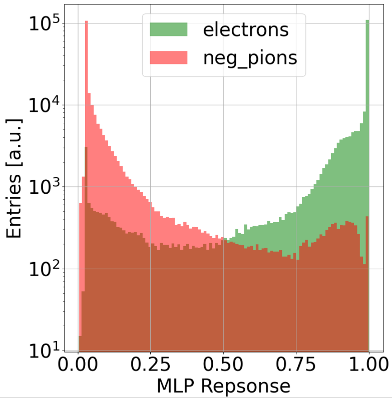
- **Very first thing to check!**
- Does the response function "make sense" ?
  - Neural network response looks "reasonable"
  - Random forest response indicates a bad performance

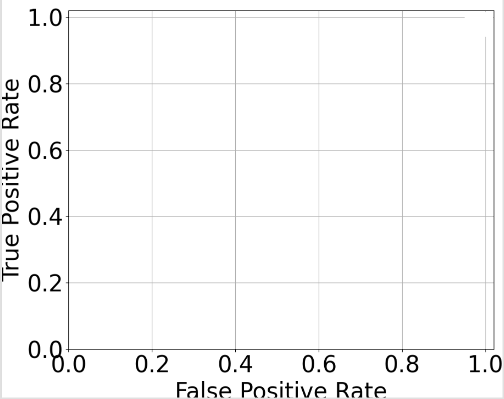
The Classifier Response



- **Very first thing to check!**
- Does the response function “make sense” ?
  - Neural network response looks “reasonable”
  - Random forest response indicates a bad performance

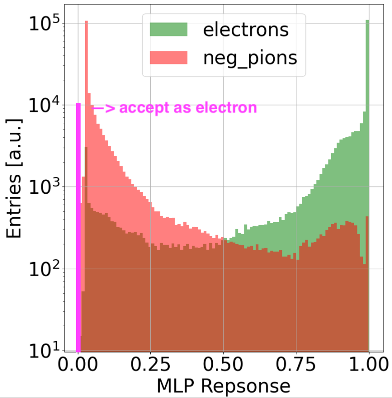
Scanning the MLP Response Distribution

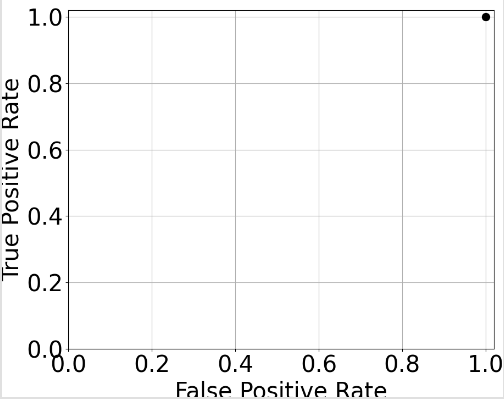




- Want to use neural network response to separate electrons from pions
- Which response value shall be used?
- Perform threshold scan on response distribution → calculate TPR / FRP and compare

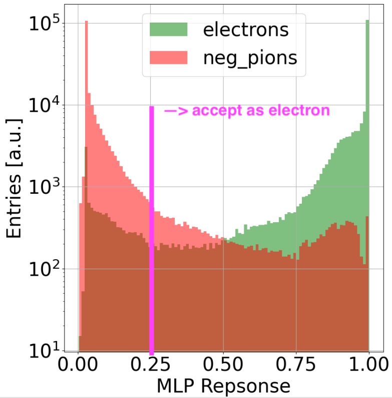
Scanning the MLP Response Distribution

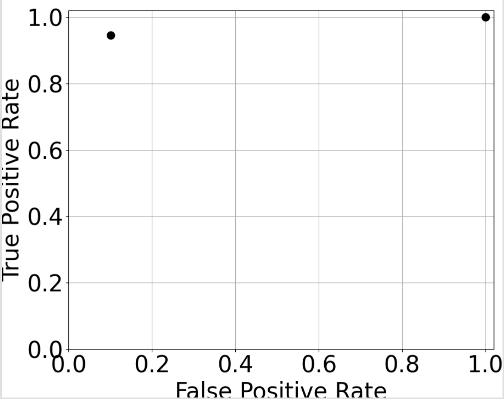




- Want to use neural network response to separate electrons from pions
- Which response value shall be used?
- Perform threshold scan on response distribution $\rightarrow$ calculate TPR / FRP and compare

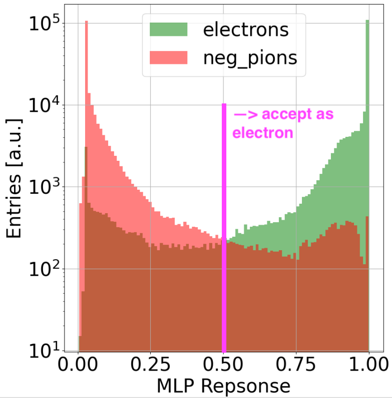
Scanning the MLP Response Distribution

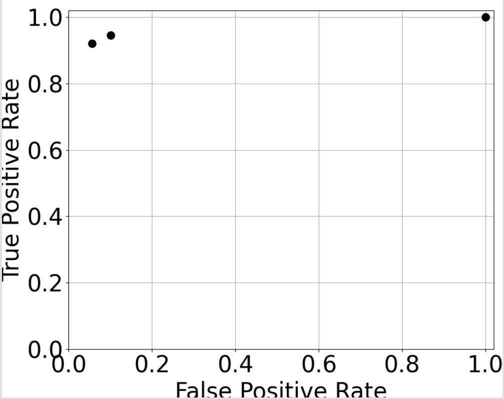




- Want to use neural network response to separate electrons from pions
- Which response value shall be used?
- Perform threshold scan on response distribution → calculate TPR / FRP and compare

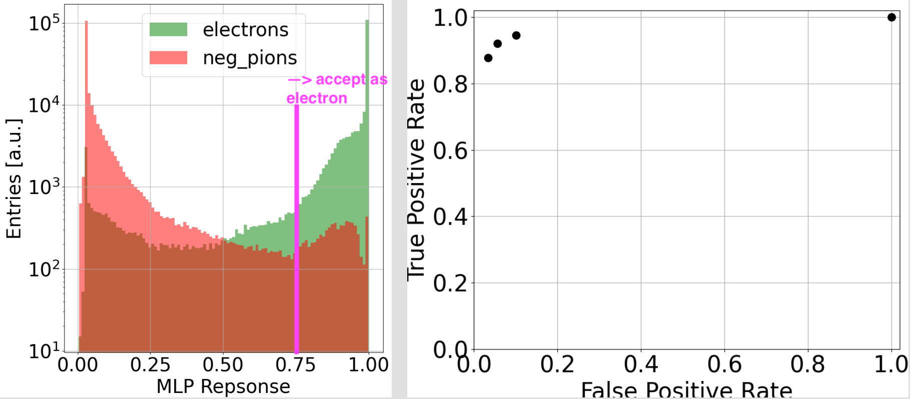
Scanning the MLP Response Distribution





- Want to use neural network response to separate electrons from pions
- Which response value shall be used?
- Perform threshold scan on response distribution → calculate TPR / FRP and compare

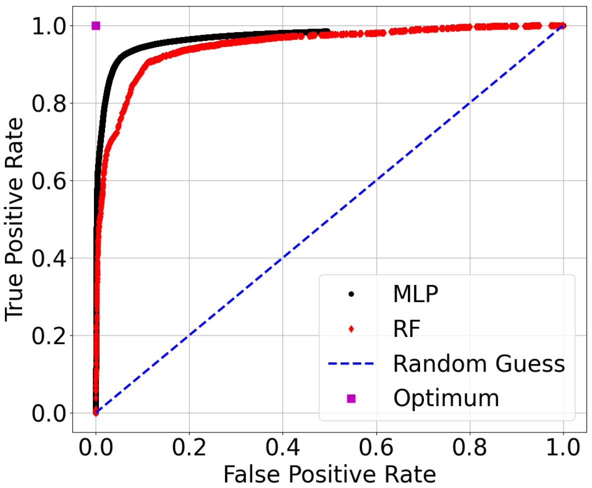
Scanning the MLP Response Distribution



- Want to use neural network response to separate electrons from pions
- Which response value shall be used?
- Perform threshold scan on response distribution $\rightarrow$ calculate TPR / FRP and compare
- **ROC- (Receiver Operating Characteristic) curve**

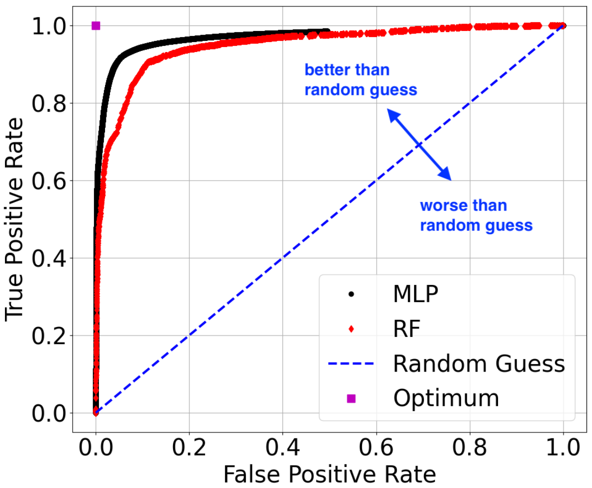
Understanding the ROC-Curve

- ROC-curve directly reflects properties of response function
- Allows to compare multiple algorithms
- Easy to interpret



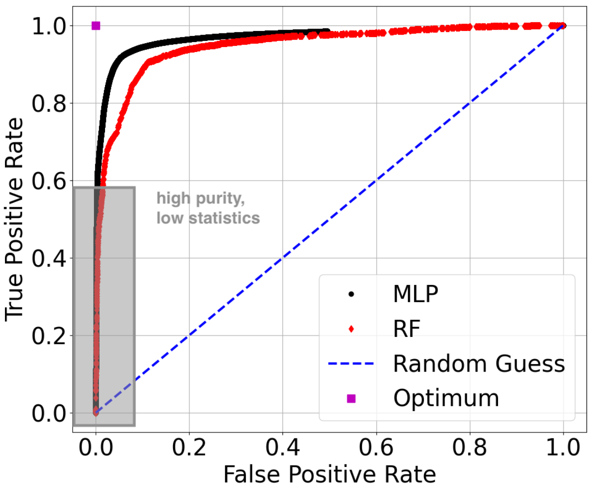
Understanding the ROC-Curve

- ROC-curve directly reflects properties of response function
- Allows to compare multiple algorithms
- Easy to interpret



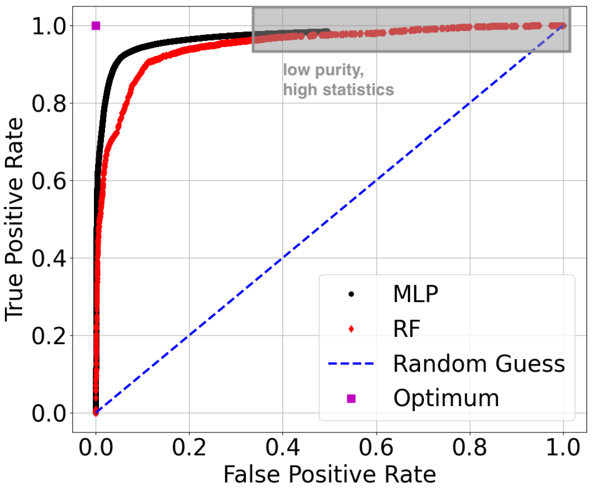
Understanding the ROC-Curve

- ROC-curve directly reflects properties of response function
- Allows to compare multiple algorithms
- Easy to interpret



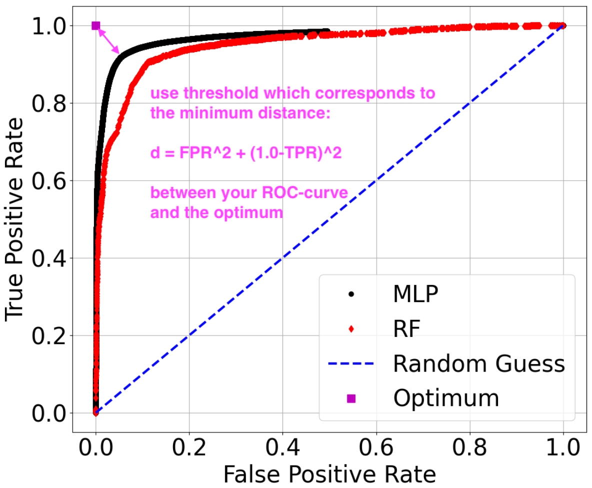
Understanding the ROC-Curve

- ROC-curve directly reflects properties of response function
- Allows to compare multiple algorithms
- Easy to interpret



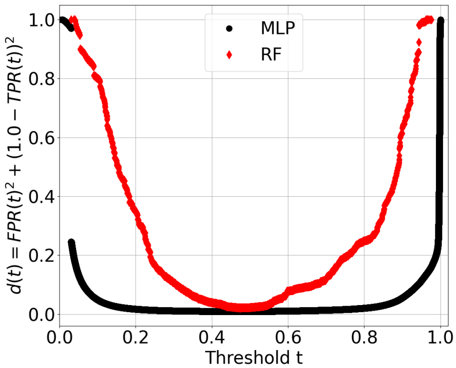
Understanding the ROC-Curve

- ROC-curve directly reflects properties of response function
- Allows to compare multiple algorithms
- Easy to interpret



Optimum Threshold

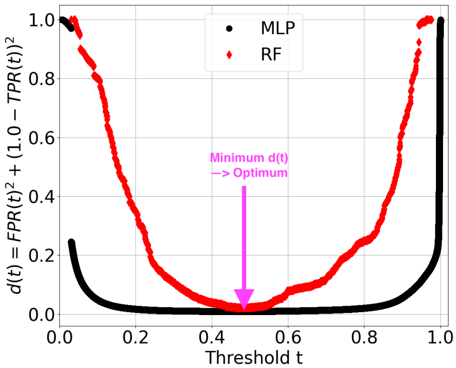
- ROC curves shown previously were produced by scikit:sklearn.metrics.roc_curve
- Threshold scan performed internally



Algorithm	Min. $d(t)$	TPR	FPR	threshold t
MLP	0.009	0.92	0.06	0.47
RF	0.021	0.9	0.11	0.48
Crude analysis	na	0.85	0.11	na

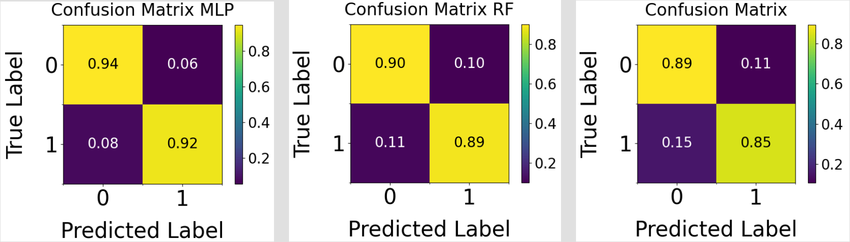
Optimum Threshold

- ROC curves shown previously were produced by scikit:sklearn.metrics.roc_curve
- Threshold scan performed internally



Algorithm	Min. $d(t)$	TPR	FPR	threshold t
MLP	0.009	0.92	0.06	0.47
RF	0.021	0.9	0.11	0.48
Crude analysis	na	0.85	0.11	na

Comparing Confusion Matrices and other Metrics



Algorithm	Min. $d(t)$	TPR	FPR	threshold t	Accuracy	MCC[5]
MLP	0.009	0.92	0.06	0.47	0.93	0.87
RF	0.021	0.9	0.11	0.48	0.89	0.79
Crude analysis	na	0.85	0.11	na	0.87	0.74

- Neural network outperforms other approaches
- Might improve (RF) performance by re-training
- **Perfect classifier:** TPR = Accuracy = MCC = 1 and FPR = 0
- **Poor classifier:** TPR = Accuracy = 0 / MCC = -1 and FPR = 1

[5] Matthews Correlation Coefficient, not discussed today

Summary

- Used different metrics to compare a neural network and a random forest classifier

Summary

- Used different metrics to compare a neural network and a random forest classifier
- Always x-check different performance metrics $\leftrightarrow$ Consistency!

Summary

- Used different metrics to compare a neural network and a random forest classifier
- Always x-check different performance metrics $\leftrightarrow$ Consistency!
- Sometimes the ROC-optimum is NOT desirable
  $\leftrightarrow$ Think about what you want to do!

Summary

- Used different metrics to compare a neural network and a random forest classifier
- Always x-check different performance metrics $\leftrightarrow$ Consistency!
- Sometimes the ROC-optimum is NOT desirable
  $\leftrightarrow$ Think about what you want to do!
- Used a crude (NOT machine learning based) analysis
  - Always have one!
  - Helps to understand / debug machine learning model
  - Helps to understand your data
  - Easier to understand (in most cases)
  - Use as reference / benchmark!

Summary

- Used different metrics to compare a neural network and a random forest classifier
- Always x-check different performance metrics $\leftrightarrow$ Consistency!
- Sometimes the ROC-optimum is NOT desirable
  $\leftrightarrow$ Think about what you want to do!
- Used a crude (NOT machine learning based) analysis
  - Always have one!
  - Helps to understand / debug machine learning model
  - Helps to understand your data
  - Easier to understand (in most cases)
  - Use as reference / benchmark!
- scikit handles threshold tuning for you
  (i.e. returned labels correspond to optimum threshold)
  Nevertheless, check algorithm response!

Summary

- Used different metrics to compare a neural network and a random forest classifier
- Always x-check different performance metrics $\leftrightarrow$ Consistency!
- Sometimes the ROC-optimum is NOT desirable
  $\leftrightarrow$ Think about what you want to do!
- Used a crude (NOT machine learning based) analysis
  - Always have one!
  - Helps to understand / debug machine learning model
  - Helps to understand your data
  - Easier to understand (in most cases)
  - Use as reference / benchmark!
- scikit handles threshold tuning for you
  (i.e. returned labels correspond to optimum threshold)
  Nevertheless, check algorithm response!
- scikit offers many more evaluation metrics see link here

Summary

- Used different metrics to compare a neural network and a random forest classifier
- Always x-check different performance metrics $\leftrightarrow$ Consistency!
- Sometimes the ROC-optimum is NOT desirable
  $\leftrightarrow$ Think about what you want to do!
- Used a crude (NOT machine learning based) analysis
  - Always have one!
  - Helps to understand / debug machine learning model
  - Helps to understand your data
  - Easier to understand (in most cases)
  - Use as reference / benchmark!
- scikit handles threshold tuning for you
  (i.e. returned labels correspond to optimum threshold)
  Nevertheless, check algorithm response!
- scikit offers many more evaluation metrics see link here
- NOT discussed today
  - Feature correlations $\leftrightarrow$ Which features do I need to include in my analysis?
  - Hyper Parameter Optimization $\leftrightarrow$ Tune the machine learning algorithm

DIY: Performance Evaluation

1.) Go to: https://replit.com/@daniel49/FSUMLLecture3

2.) Klick on the **Fork** button

3.) Sign in or log in with your credentials (repl is free)

4.) Follow instructions in **main.py**

NOTE: The data you are able to analyze on repl, is just a sub-set ($\sim 20\,\mathrm{k}$ events) of the data presented here ($\sim 400\,\mathrm{k}$ events) $\rightarrow$ However, both data sets (the one used here and the sub-sample) are available at:
http://hadron.physics.fsu.edu/~dlersch/Intro_To_ML_2021/data/