Introduction to Machine Learning: Part IV

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

April 15, 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:**
  - Particle Identification
  - Classification Metrics
- **Part IV:** (Today)
  - Hyper Parameter Optimization (HPO)
  - Physics Data Analysis

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 $\rightarrow$ revised material for this edition

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

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

... most likely contain several errors ($\rightarrow$ 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

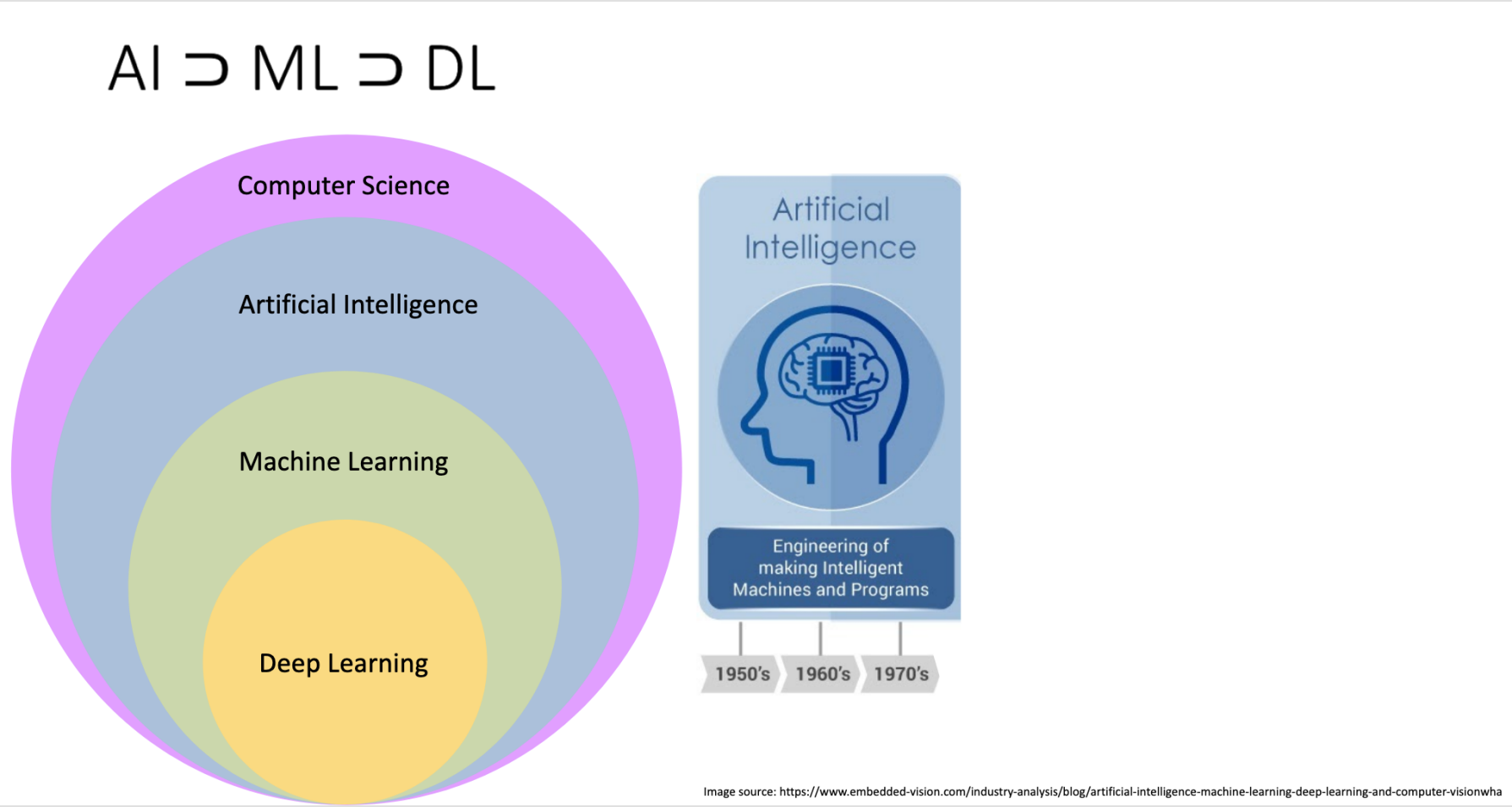
Homework and Literature

- 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

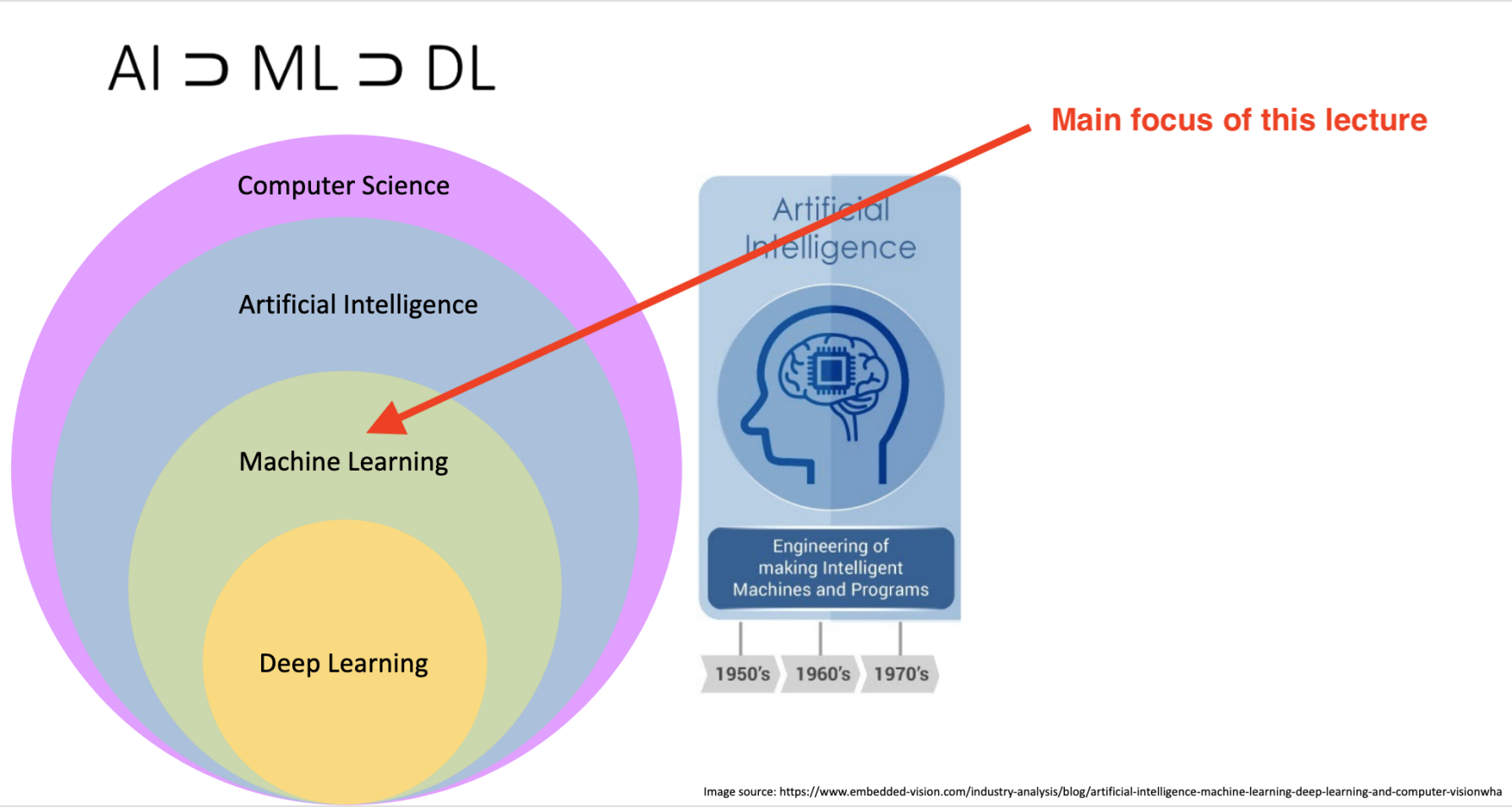
[4] Any document claiming that there is a quick way to understand machine learning without any theory / math is considered as bad

AI, ML and DL



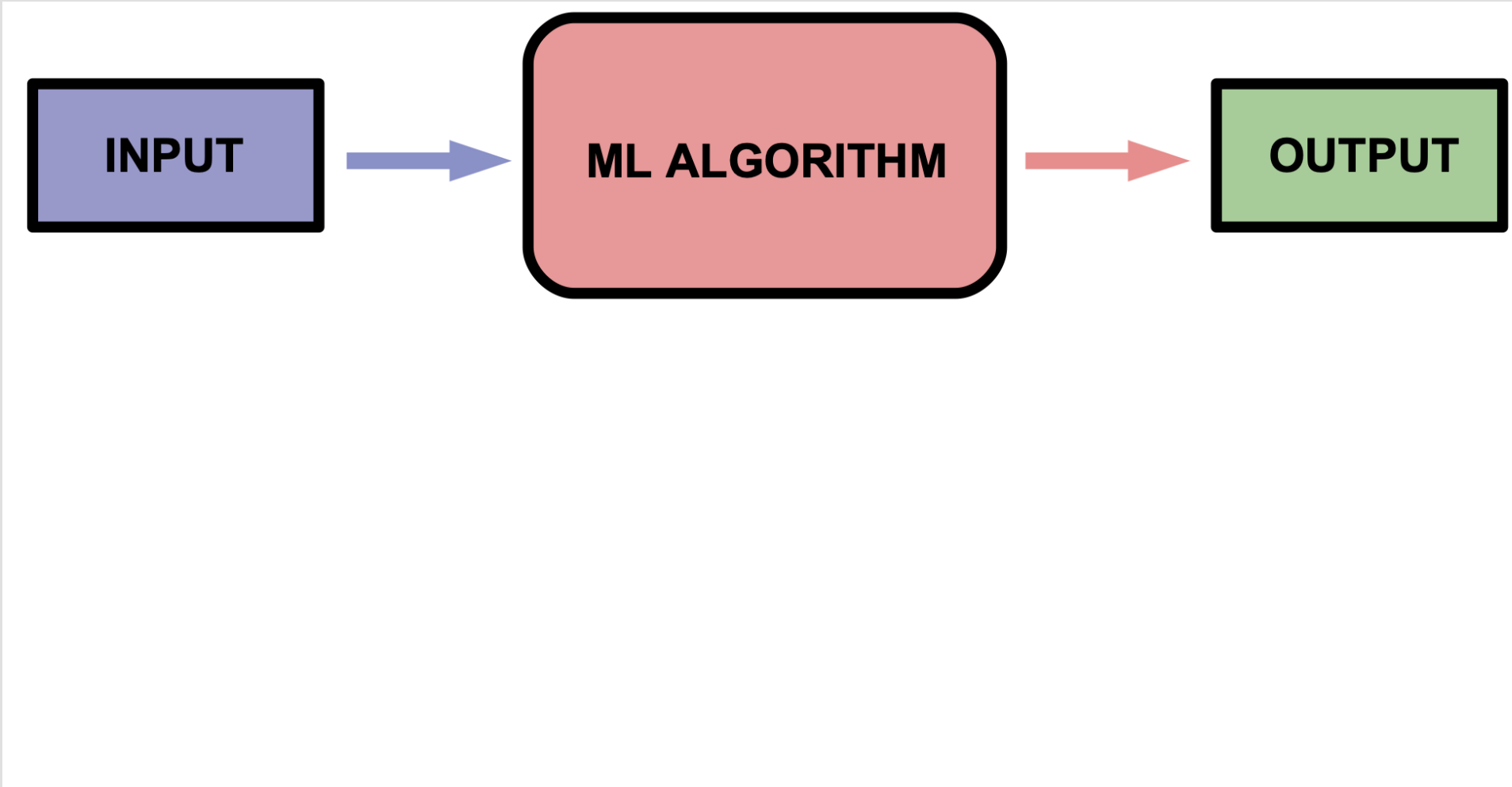
Slide taken from Brenda Ngs introductory talk at the: **deep learning for science school 2019**

AI, ML and DL

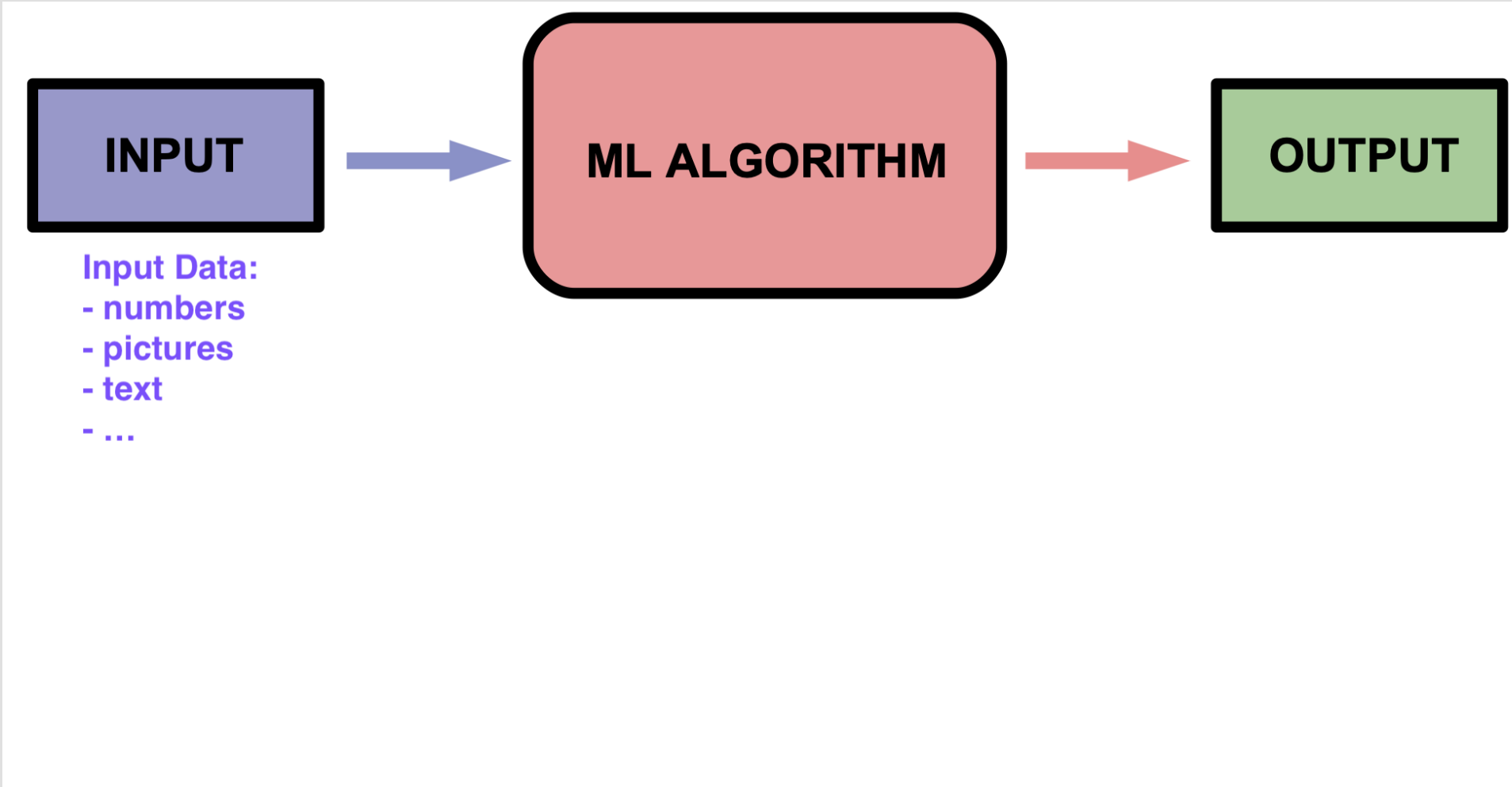


Slide taken from Brenda Ngs introductory talk at the: **deep learning for science school 2019**

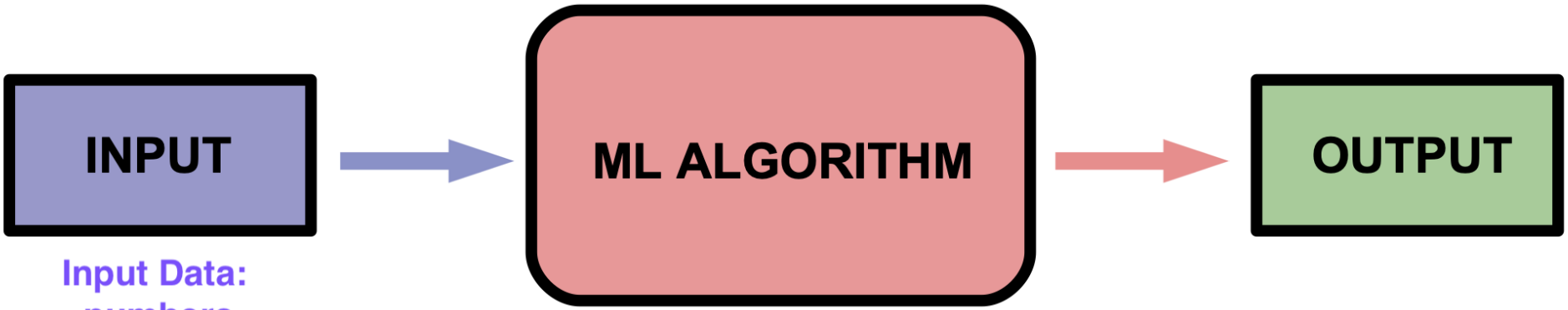
Basic Components of Machine Learning



Basic Components of Machine Learning



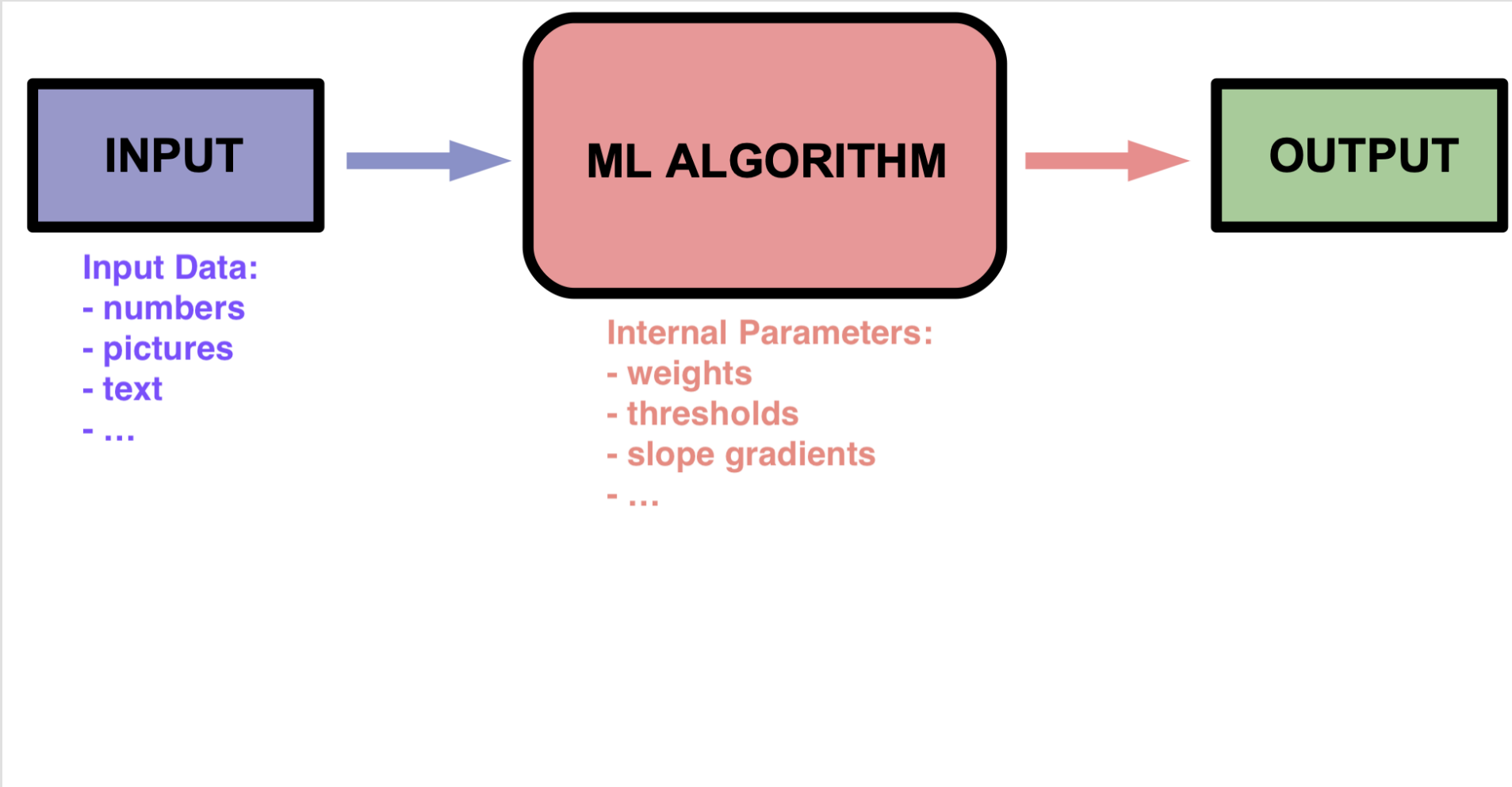
Basic Components of Machine Learning



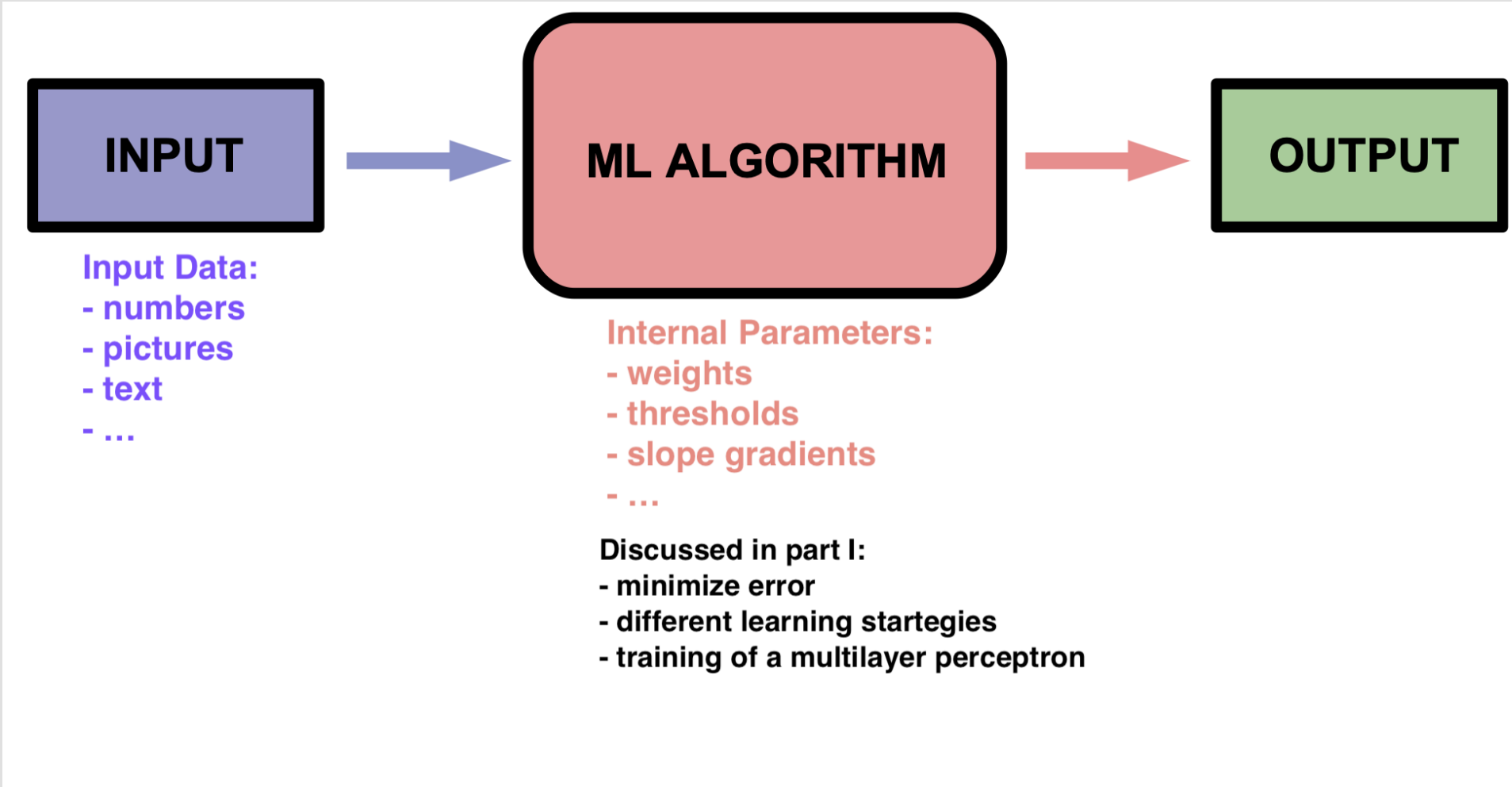
Input Data:
- numbers
- pictures
- text
- ...

Introduced in part I: DataFrames —> handle and manipulate data

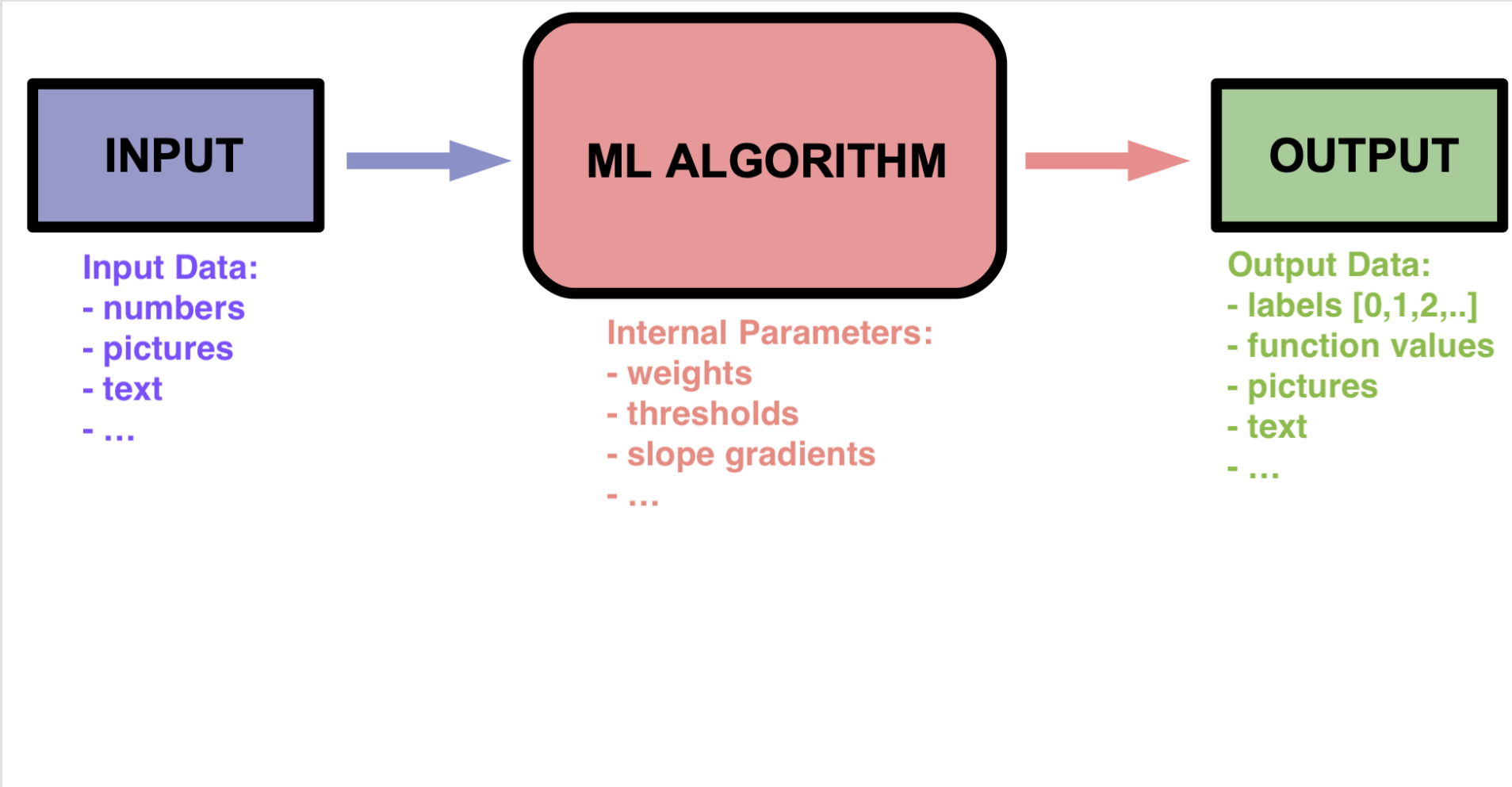
Basic Components of Machine Learning



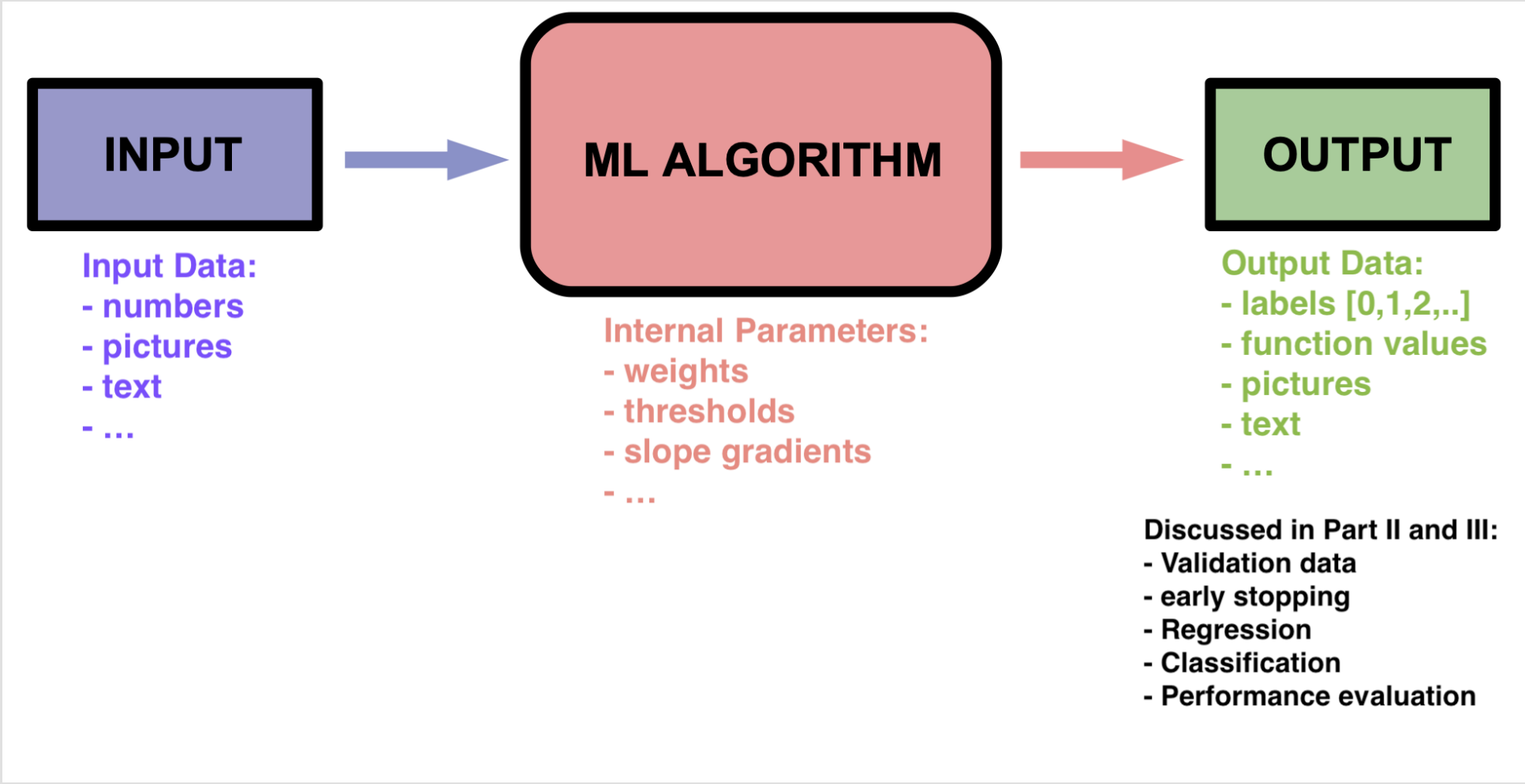
Basic Components of Machine Learning



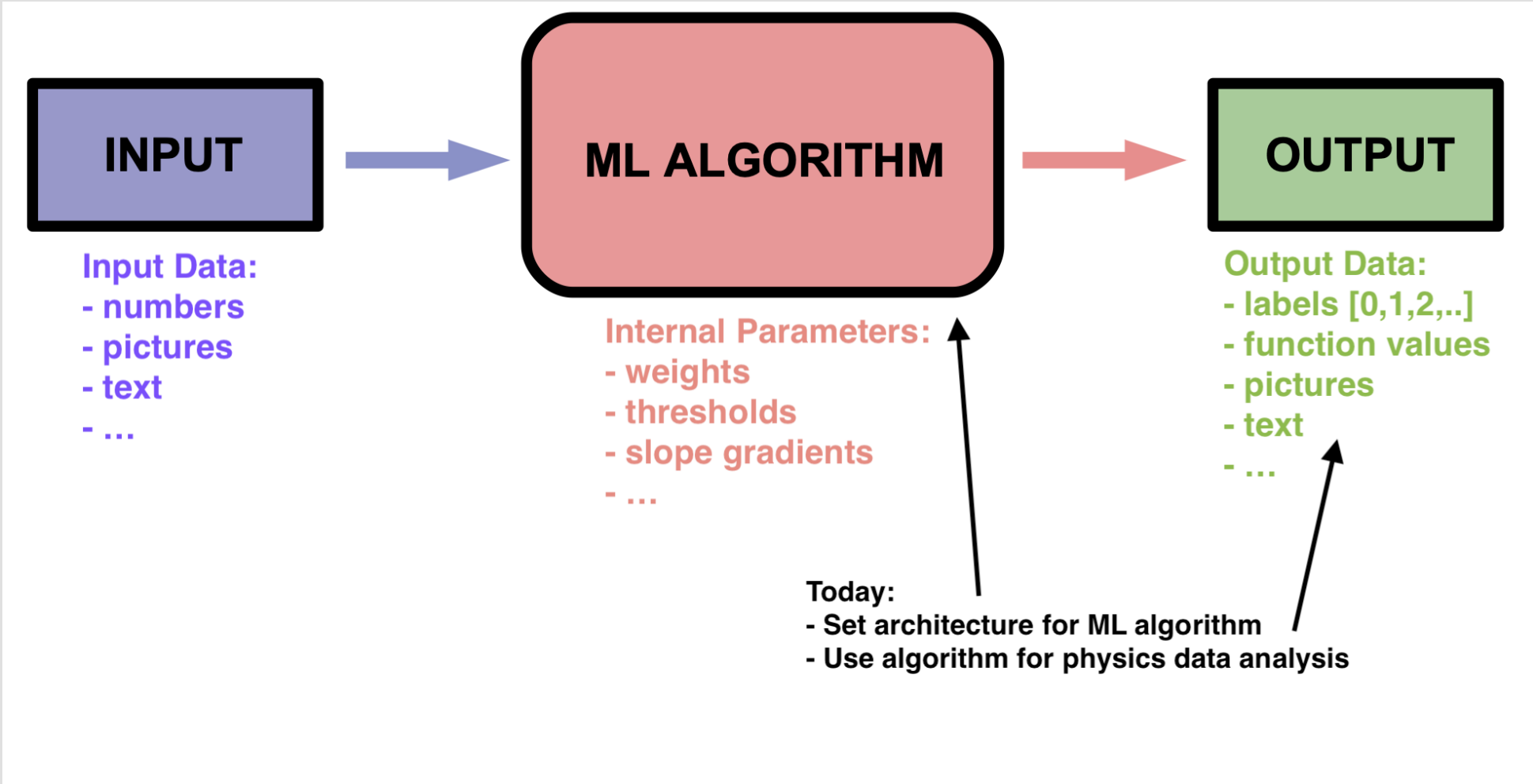
Basic Components of Machine Learning



Basic Components of Machine Learning



Basic Components of Machine Learning



Hyper Parameters

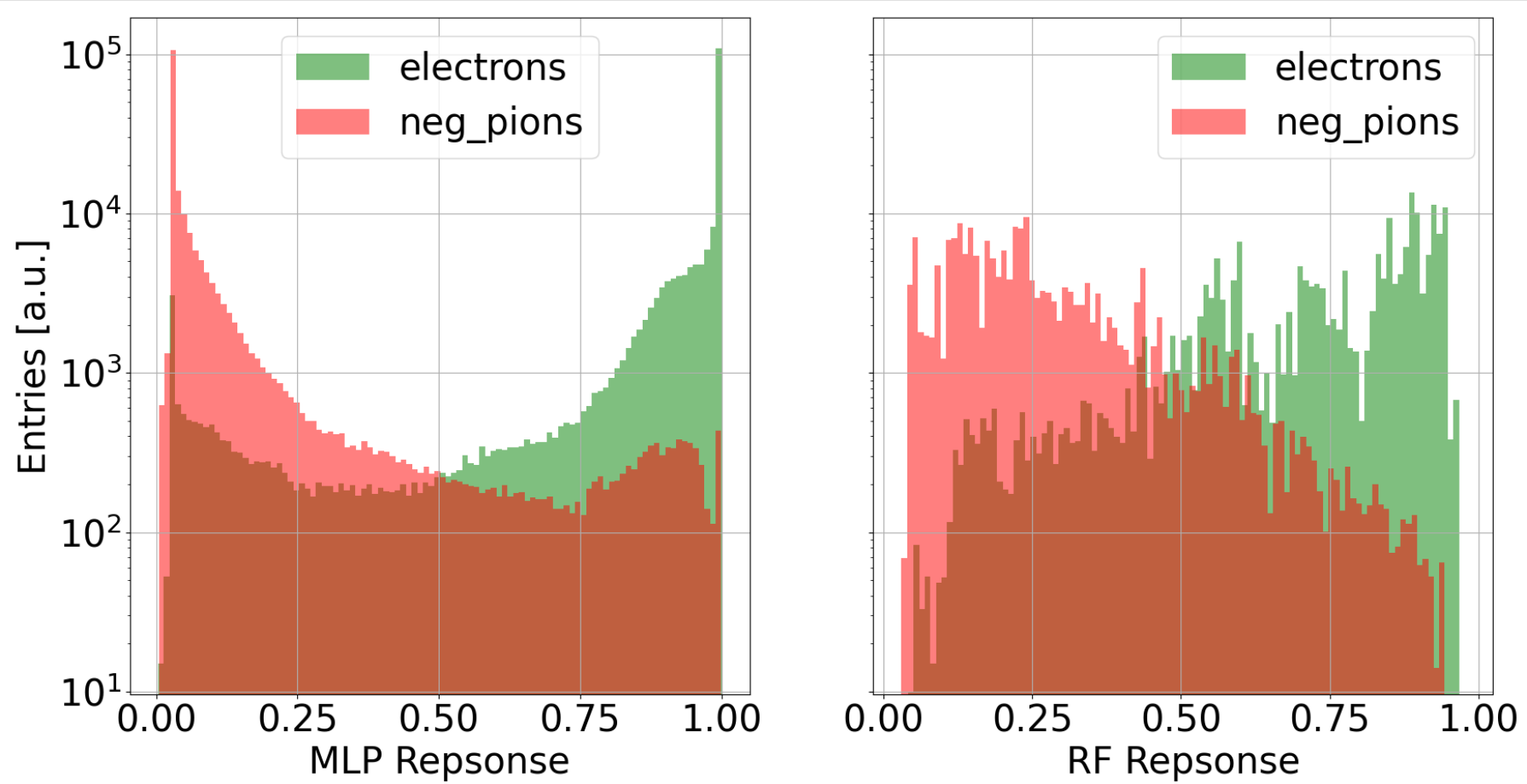
- **Fit Parameters:** Model internal parameters → Set by optimization procedure → driven by data
- **Hyper Parameters:** Determine model architecture / performance → Set by user

Model	Fit Parameters	Hyper Parameters
$pol(N) = p_N x^N + p_{N-1} x^{N-1} + ... + p_0$	$p_N, .., p_0$	N, minimizer,...
Multilayer Perceptron	weights, biases	#Hidden Layers, #Neurons, #Epochs,...
Random Forest	Thresholds, splitting level	#Trees, max. depth of trees,...

Hyper Parameter Optimization (HPO)

- **Goal:** Find set of hyper parameters which maximizes the prediction performance $\leftrightarrow$ Minimize prediction error
- **To consider:**
  - Use computational time efficiently
  - Generalizability $\leftrightarrow$ avoid overfitting $\rightarrow$ Use validation data!
  - Actually find optimum
- **How to: HPO?**
  - Test different settings manually $\rightarrow$ can be painful
  - Use algorithm
    - ★ Grid search: simple, but ineffective if parameter ranges are unknown
    - ★ Random parameter search
    - ★ Bayesian optimization
    - ★ and many more...

Parameter Grid Search for the random Forest Classifier



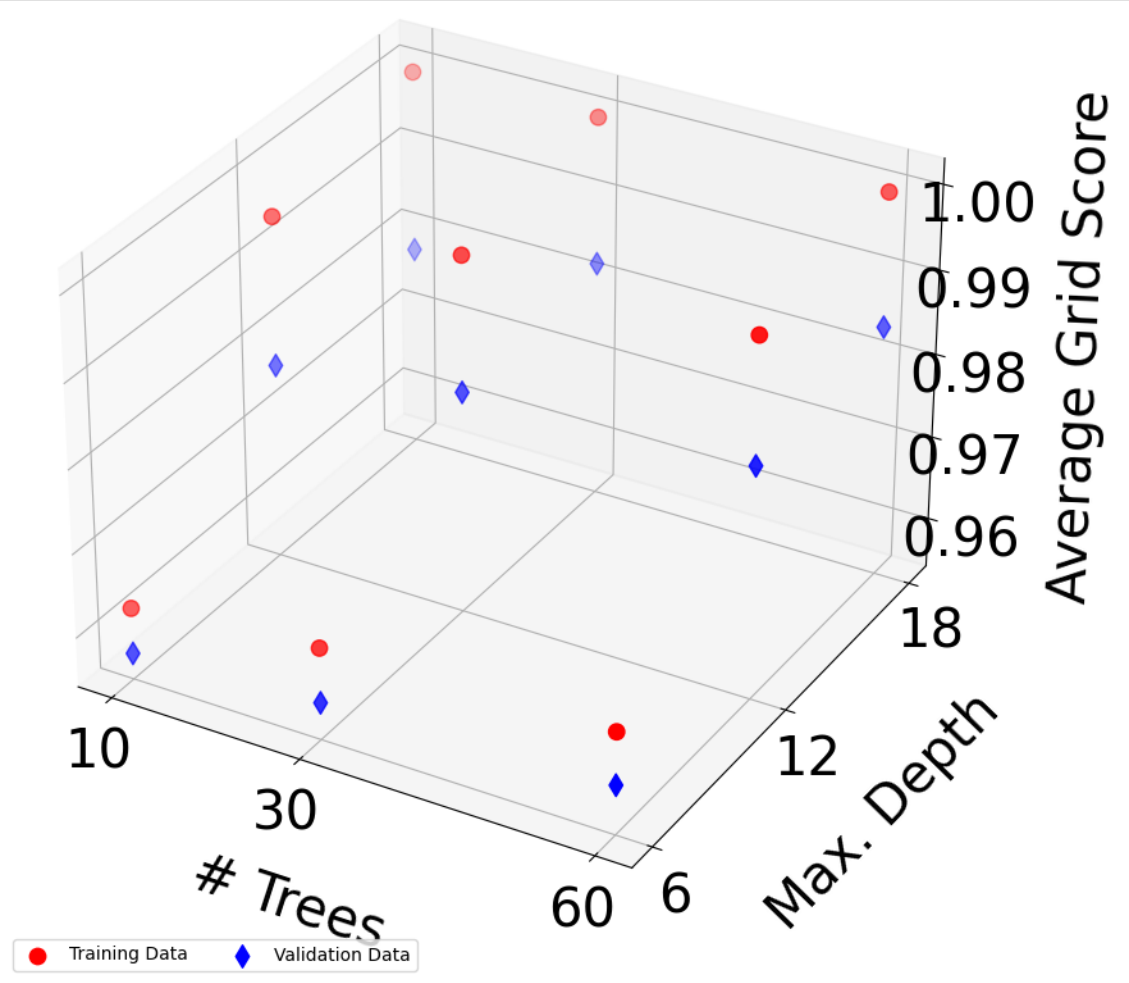
- Random forest classifier trained last lecture shows insufficient performance
- Use parameter grid search to optimize this classifier

Parameter Grid Search in scikit

```
#1.) Define the parameters you want to tune:
#n_estimator = number or trees within the ensemble
#max_depth: Depth you allow each tree to grow during training
parameters_to_tune = [{'n_estimators': [10,30,60],'max_depth': [6,12,18]}]
#2.) Define a scorer function
tuning_score = 'roc_auc' #---> Area under the ROC-curve
#<=> Should be 1 for the ideal ROC-curve
#3.) #Set up the grid search function:
rf_scan = GridSearchCV(
      RandomForestClassifier(), #--> The algorithm you want tune
      parameters_to_tune, #---> Specify the hyper parameters
      scoring=tuning_score, #---> Which metric to judge the performance
      return_train_score=True #---> Calculate score for the training data
)
#4.) Run the grid search:
rf_scan.fit(X, Y)
```

- Checks EVERY parameter combination (here: 9)
- Internally splits data into training / test samples
  (e.g. via k-fold cross-validation)
- Uses highest score on the test set to determine the best parameter configuration
- Details can be found under this link

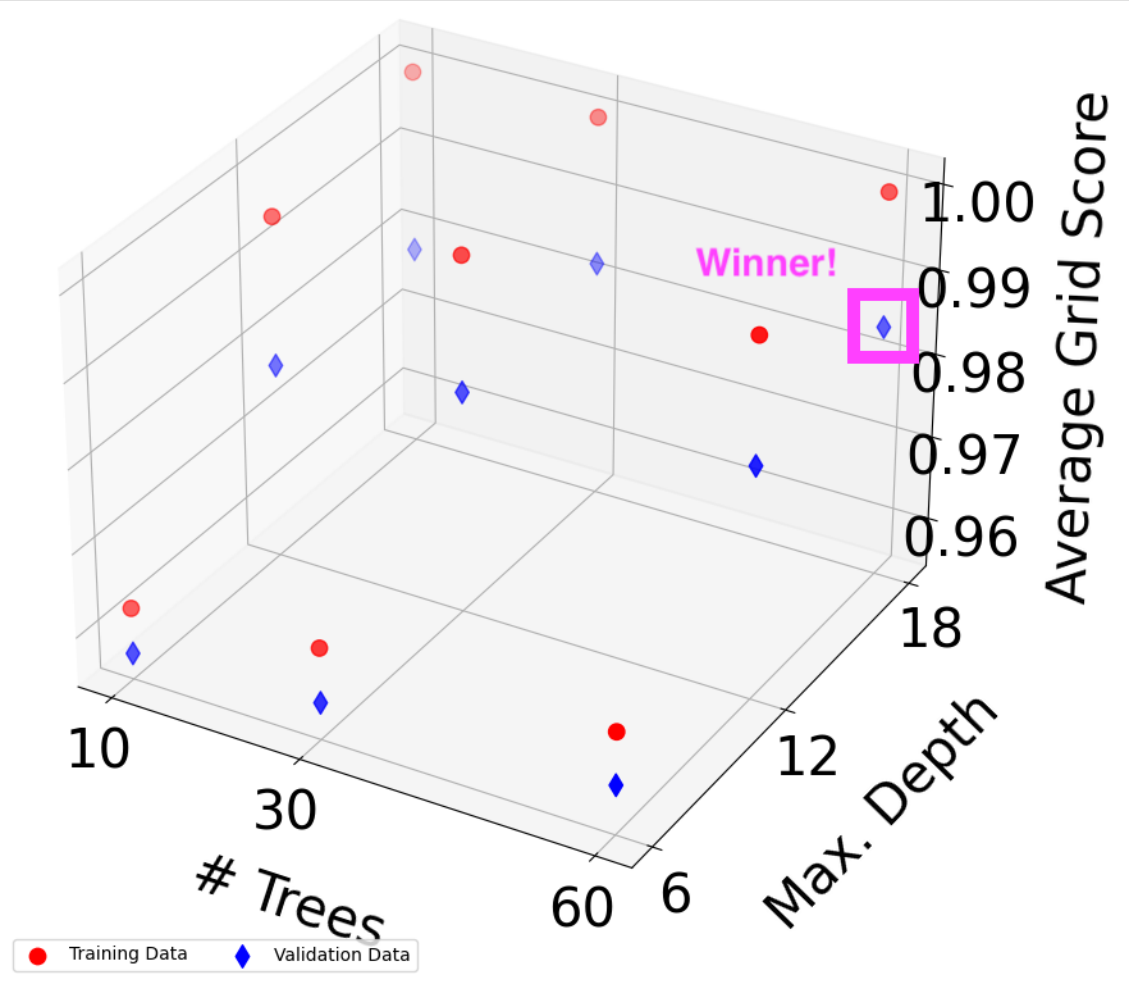
Results from the Grid Search



- Performed grid search on small sub-sample[5] of e^-/π^- data
- Best model: 60 Trees with depth 18 $\leftrightarrow$ edge of parameter values that we specified

[5] Due to time constraints

Results from the Grid Search

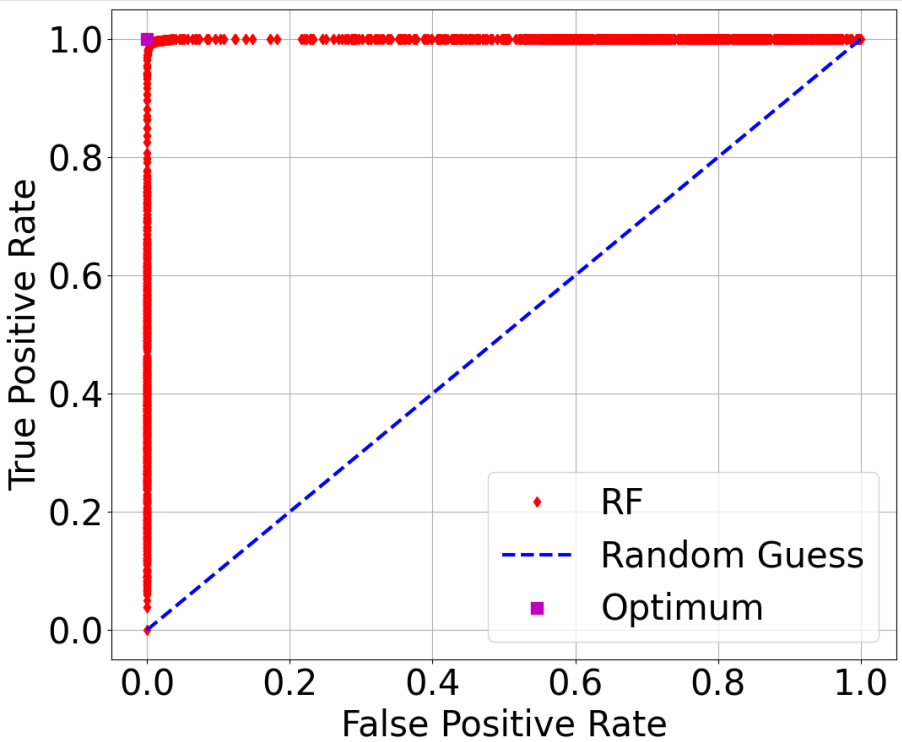


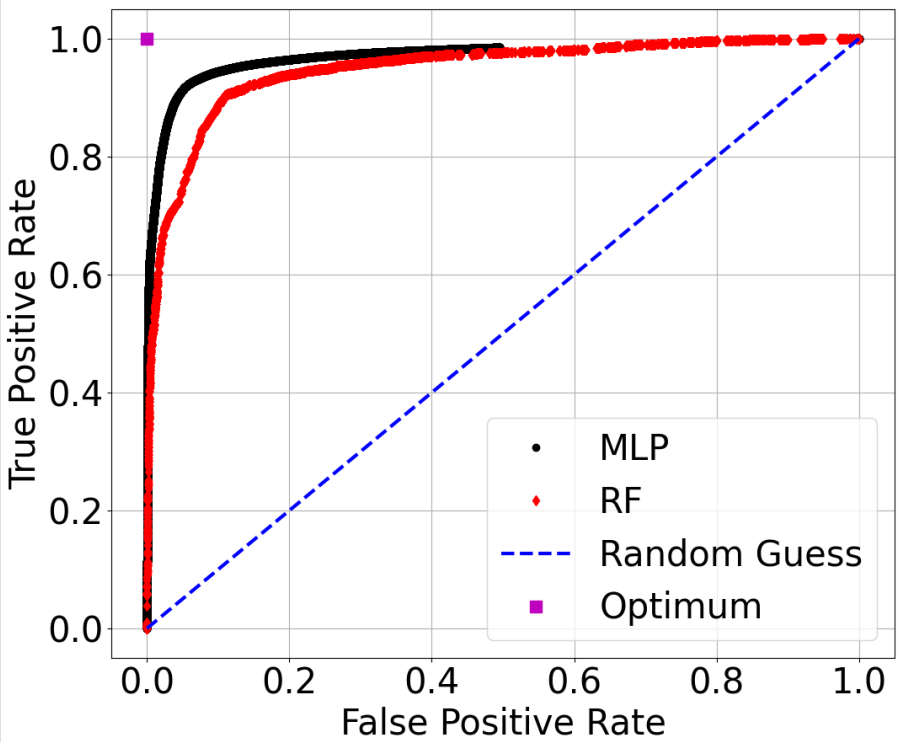
- Performed grid search on small sub-sample[5] of e^-/π^- data
- Best model: 60 Trees with depth 18 $\leftrightarrow$ edge of parameter values that we specified

[5] Due to time constraints

Optimized Random Forest: ROC-Curve

- **Bottom Left:** Random Forest after grid search
- **Bottom Right:** Random Forest trained with best guess parameters[6]

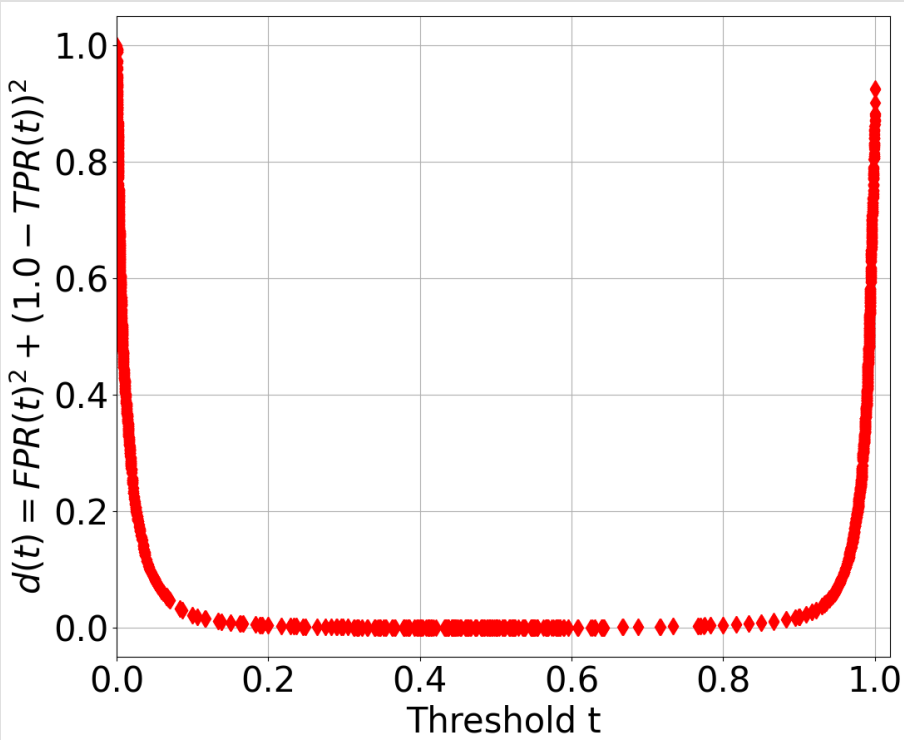


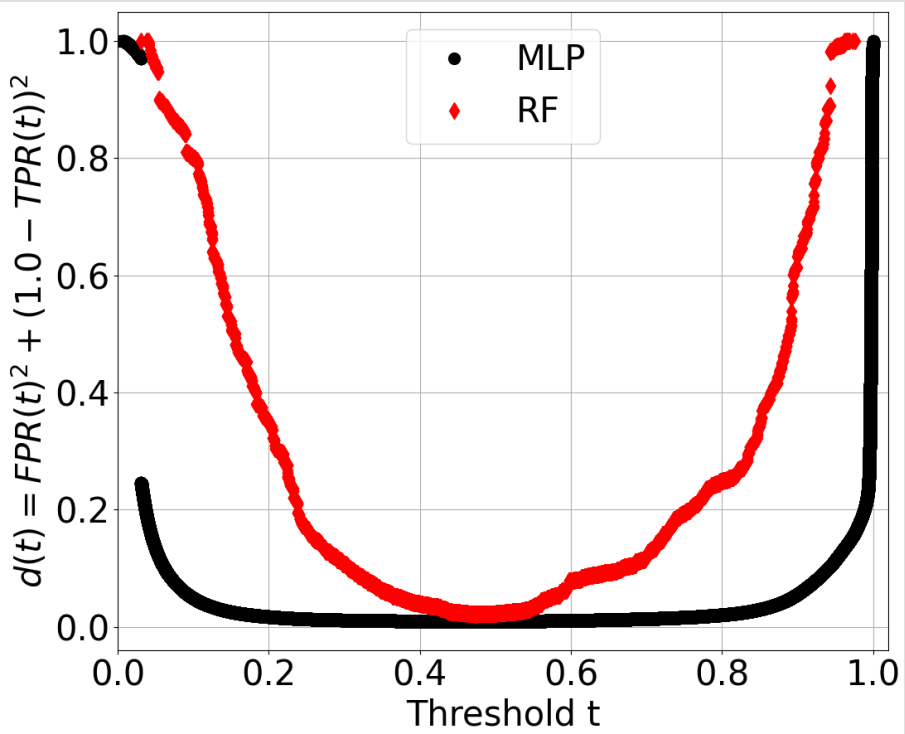


[6] **NOTE:** The statistics used for training here were larger than those used for the grid search

Optimized Random Forest: ROC-Curve

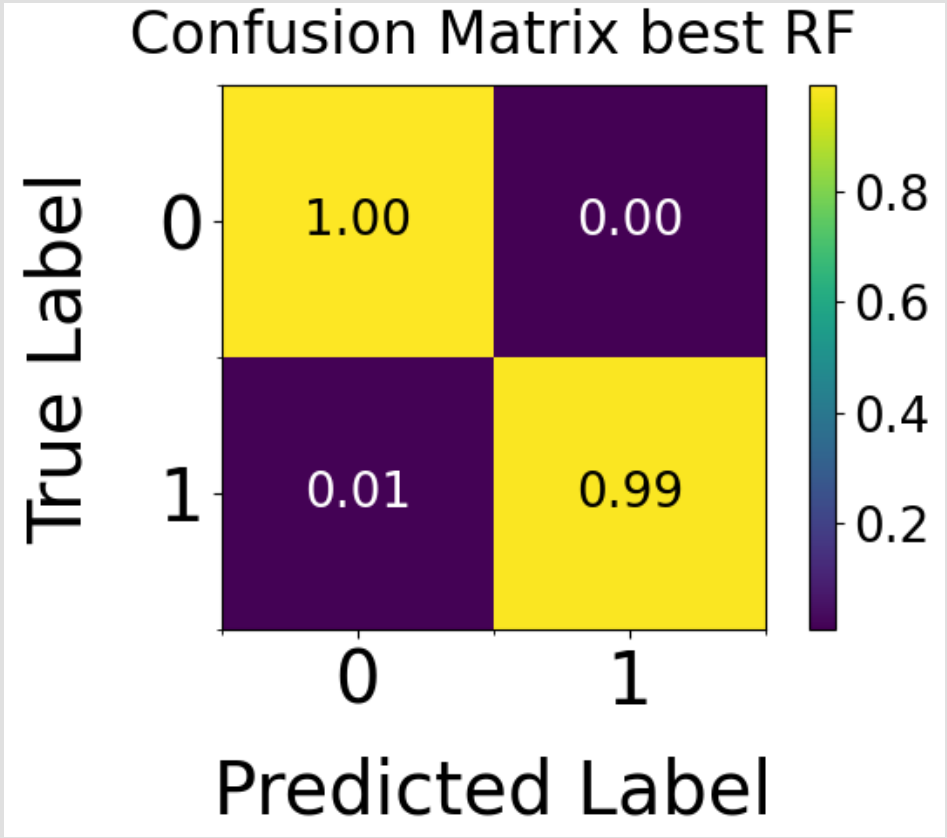
- **Bottom Left:** Random Forest after grid search
- **Bottom Right:** Random Forest trained with best guess parameters[6]





[6] **NOTE:** The statistics used for training here were larger than those used for the grid search

Optimized Random Forest: Classification Performance



Algorithm	Min. $d(t)$	TPR	FPR	threshold t	Accuracy	MCC[7]
RF (last lecture)	0.021	0.9	0.11	0.48	0.89	0.79
Best RF	$8 \cdot 10^{-5}$	0.9935	0.0065	0.48	0.99	0.99

[7] Matthews Correlation Coefficient, not discussed today

Intermediate Summary: Grid Search

- Used grid search to tune a random forest classifier

Intermediate Summary: Grid Search

- Used grid search to tune a random forest classifier
- Performance improved drastically
  $\rightarrow$ Might change if algorithm is applied on full statistics data set!

Intermediate Summary: Grid Search

- Used grid search to tune a random forest classifier
- Performance improved drastically
  $\rightarrow$ Might change if algorithm is applied on full statistics data set!
- Grid search is simple, but can be computationally expensive:

  $$\#\text{Searches} = \prod_i \#\text{Settings[Parameter(i)]}$$

Intermediate Summary: Grid Search

- Used grid search to tune a random forest classifier
- Performance improved drastically
  $\rightarrow$ Might change if algorithm is applied on full statistics data set!
- Grid search is simple, but can be computationally expensive:

  $$\#\text{Searches} = \prod_i \#\text{Settings[Parameter(i)]}$$

- No guarantee, that best parameter is found after an extensive grid search
  $\rightarrow$ Might simply "miss" best performing set $\rightarrow$ parameter grid too coarse / fine

Intermediate Summary: Grid Search

- Used grid search to tune a random forest classifier
- Performance improved drastically
  $\rightarrow$ Might change if algorithm is applied on full statistics data set!
- Grid search is simple, but can be computationally expensive:

  $$\#\text{Searches} = \prod_i \#\text{Settings[Parameter(i)]}$$

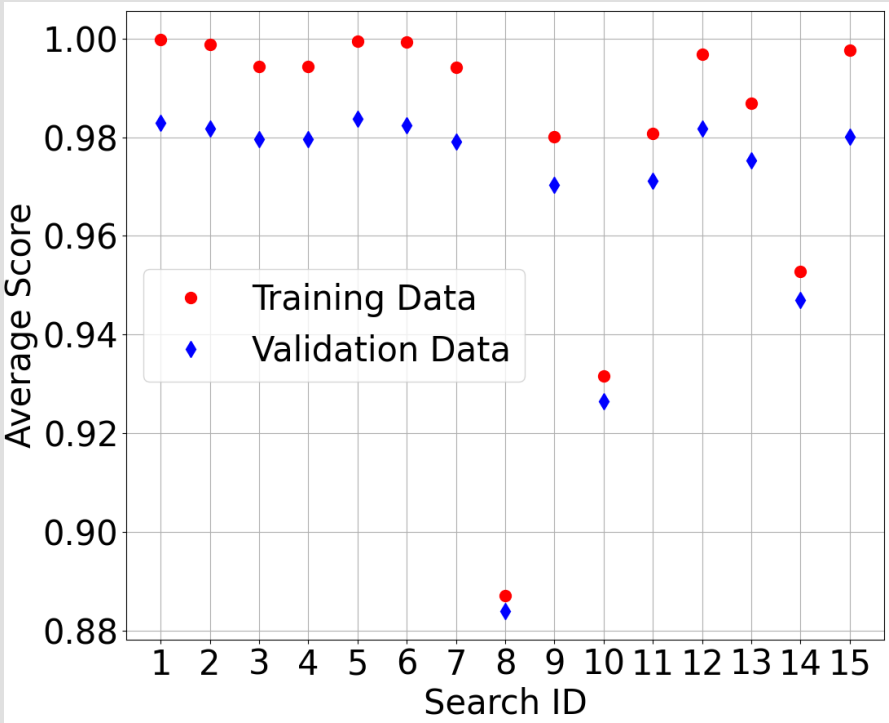
- No guarantee, that best parameter is found after an extensive grid search
  $\rightarrow$ Might simply "miss" best performing set $\rightarrow$ parameter grid too coarse / fine
- (more effective) Alternatives
  - Random parameter search (discussed here)
  - Successive halving
  - Bayesian optimization

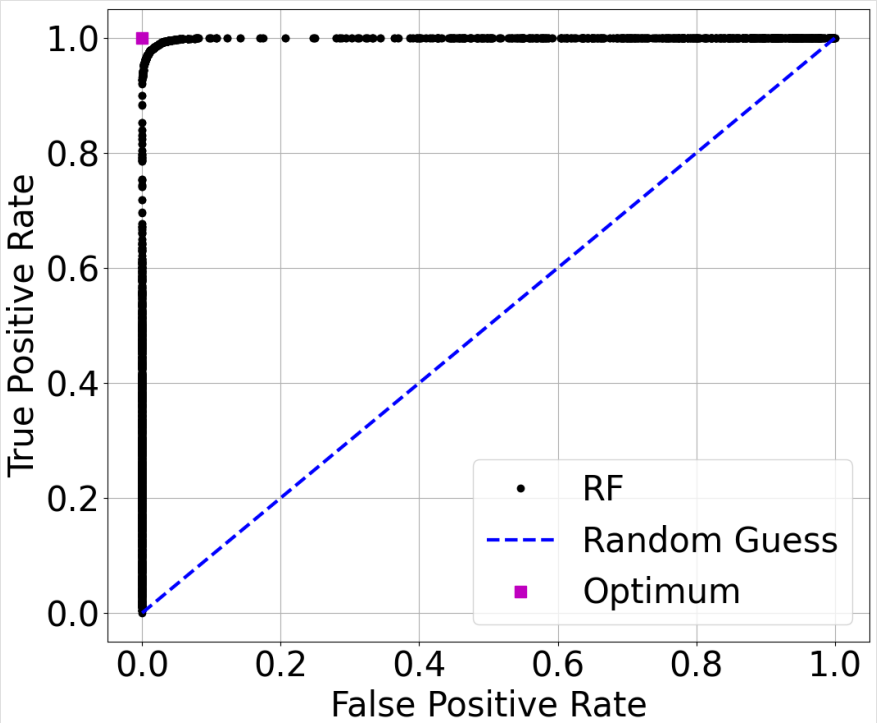
Random Parameter Search in scikit

```
#1.) Define the parameters you want to tune:
parameters_to_tune_rf = {
   'n_estimators': stats.randint(5,60),#---> Define parameter ranges instead
   'max_depth': stats.randint(3,18)
}
#2.) Define the number of searches
n_searches = 15#--> Control over computational time
tuning_score = 'roc_auc'#---> Area under the ROC-curve
#3.) Set up search function:
random_search = RandomizedSearchCV(
   RandomForestClassifier(), #--> The algorithm you want tune
   param_distributions=parameters_to_tune_rf, #---> Specify hyper parameters
   n_iter=n_searches,
   scoring=tuning_score, #---> Which metric to judge the performance
 )
 #4.) Run the search:
 random_search.fit(X,Y)
```

- Draws random parameter samples
- Internally splits data into training / test samples
  (e.g. via k-fold cross-validation)
- Uses highest score on the test set to determine the best parameter configuration
- Details can be found under this link

Results from random Search

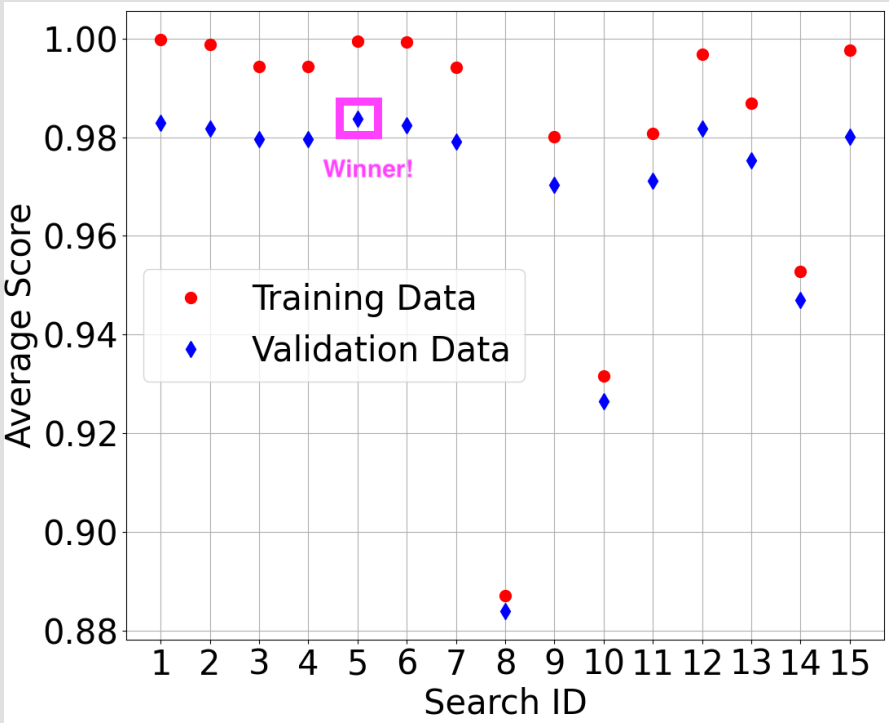


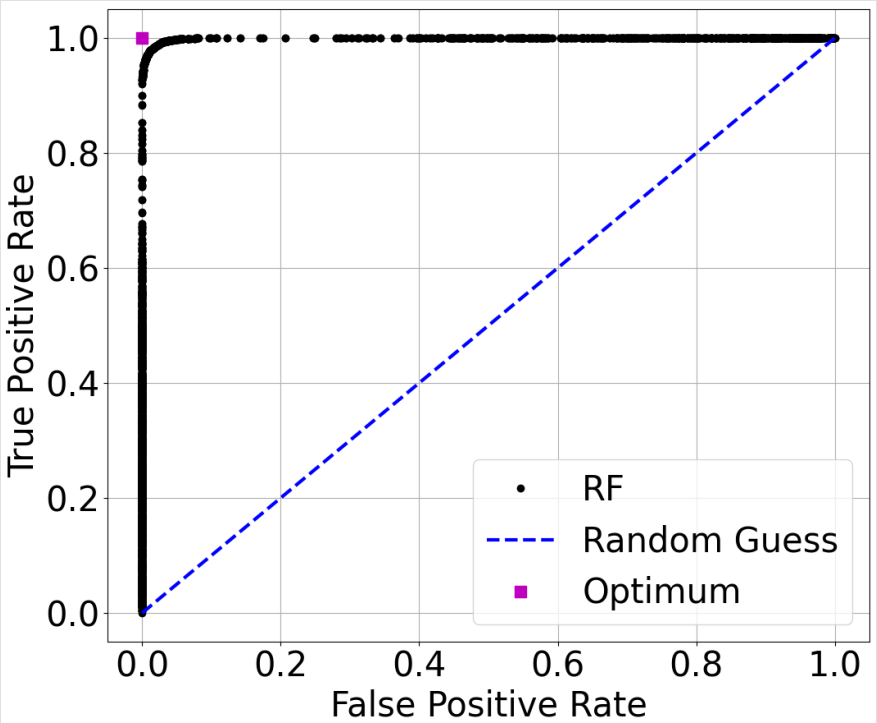


Algorithm	Min. $d(t)$	TPR	FPR	threshold t	Accuracy	MCC[8]
RF (last lecture)	0.021	0.9	0.11	0.48	0.89	0.79
RF (grid search)	$8 \cdot 10^{-5}$	0.9935	0.0061	0.48	0.99	0.99
RF (rnd search)	$6 \cdot 10^{-4}$	0.98	0.02	0.47	0.98	0.97

[8] Matthews Correlation Coefficient, not discussed today

Results from random Search





Algorithm	Min. $d(t)$	TPR	FPR	threshold t	Accuracy	MCC[8]
RF (last lecture)	0.021	0.9	0.11	0.48	0.89	0.79
RF (grid search)	$8 \cdot 10^{-5}$	0.9935	0.0061	0.48	0.99	0.99
RF (rnd search)	$6 \cdot 10^{-4}$	0.98	0.02	0.47	0.98	0.97

[8] Matthews Correlation Coefficient, not discussed today

Intermediate Summary: Random Search

- Performance competitive to grid search
- Define parameter ranges (and not specific values)
- Control over computational time $\leftrightarrow$ Define number of searches
- Still might miss optimum $\rightarrow$ Number of searches set too low

DIY: HPO

1.) Go to: https://replit.com/@daniel49/FSUMLLecture4

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

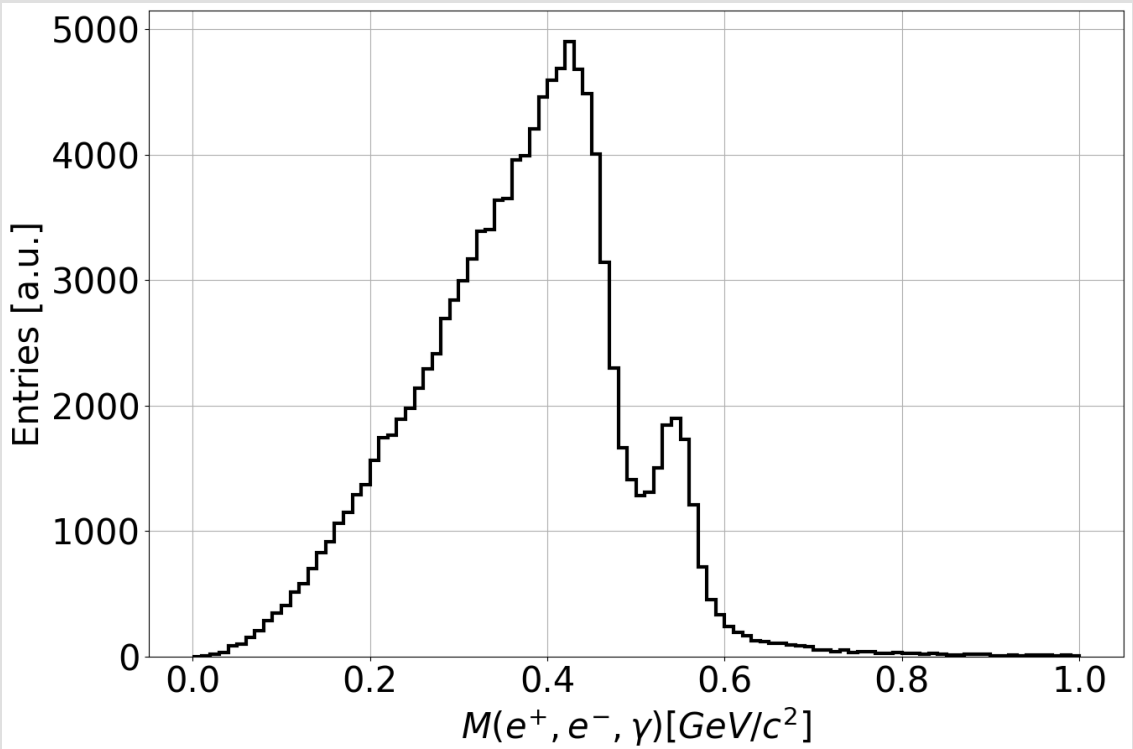
3.) Sign in or log in with your credentials (repl is free)

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/

Application in Physics Data Analysis

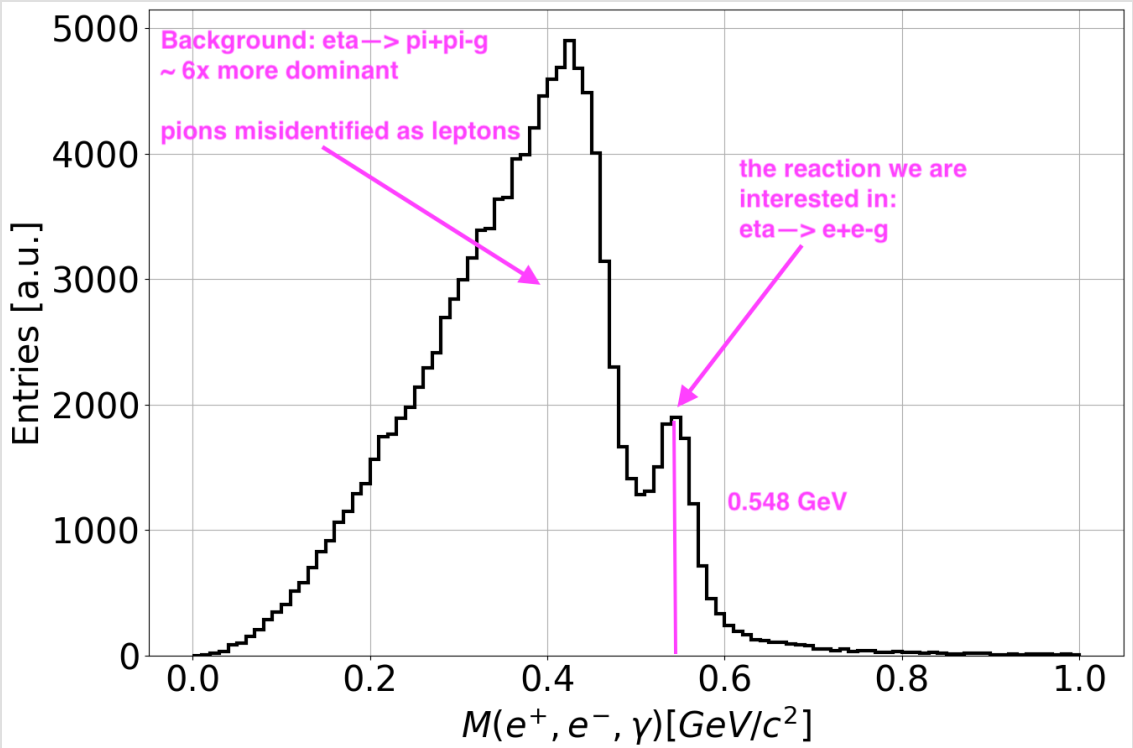
- Look at toy data: $\gamma p \to p\eta$
  - i) $\eta \to e^+e^-\gamma$
  - ii) $\eta \to \pi^+\pi^-\gamma$ (6× more likely than i))
- **Goal:** Want to reconstruct the reaction: $\eta \to e^+e^-\gamma$
  i.e. $M(e^+, e^-, \gamma) = m_\eta = 0.548\,\mathrm{GeV/c^2}$



- **Problem:** Pions are misidentified as electrons: $\eta \to \pi^+\pi^-\gamma$ treated as $\eta \to e^+e^-\gamma$
- **Use classifier trained on $e^\pm/\pi^\pm$ tracks:** suppress $\eta \to \pi^+\pi^-\gamma$ background

Application in Physics Data Analysis

- Look at toy data: $\gamma p \to p\eta$
  - i) $\eta \to e^+e^-\gamma$
  - ii) $\eta \to \pi^+\pi^-\gamma$ (6× more likely than i))
- **Goal:** Want to reconstruct the reaction: $\eta \to e^+e^-\gamma$
  i.e. $M(e^+, e^-, \gamma) = m_\eta = 0.548\,\mathrm{GeV/c^2}$



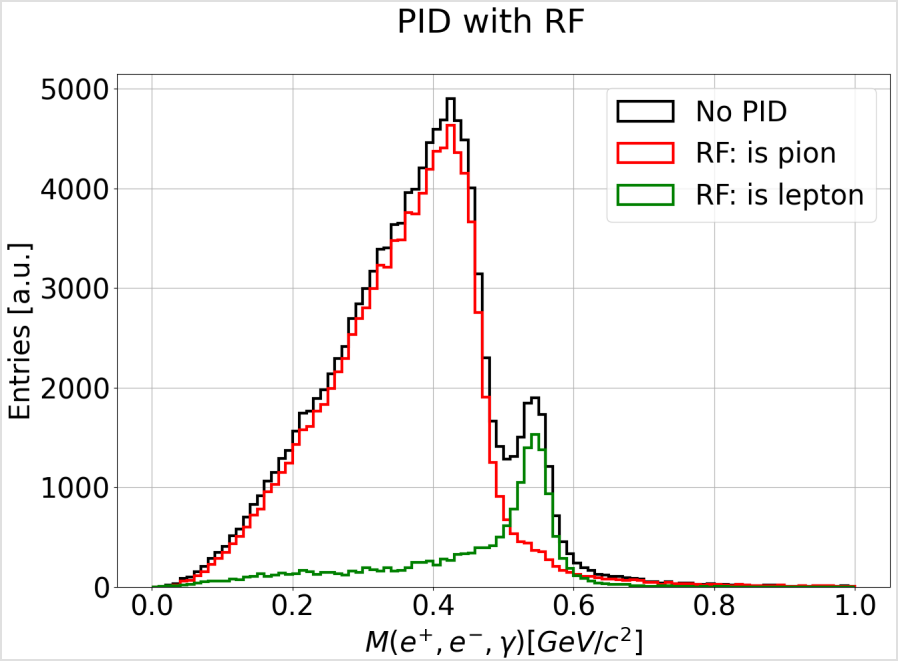
- **Problem:** Pions are misidentified as electrons: $\eta \to \pi^+\pi^-\gamma$ treated as $\eta \to e^+e^-\gamma$
- **Use classifier trained on $e^\pm/\pi^\pm$ tracks:** suppress $\eta \to \pi^+\pi^-\gamma$ background

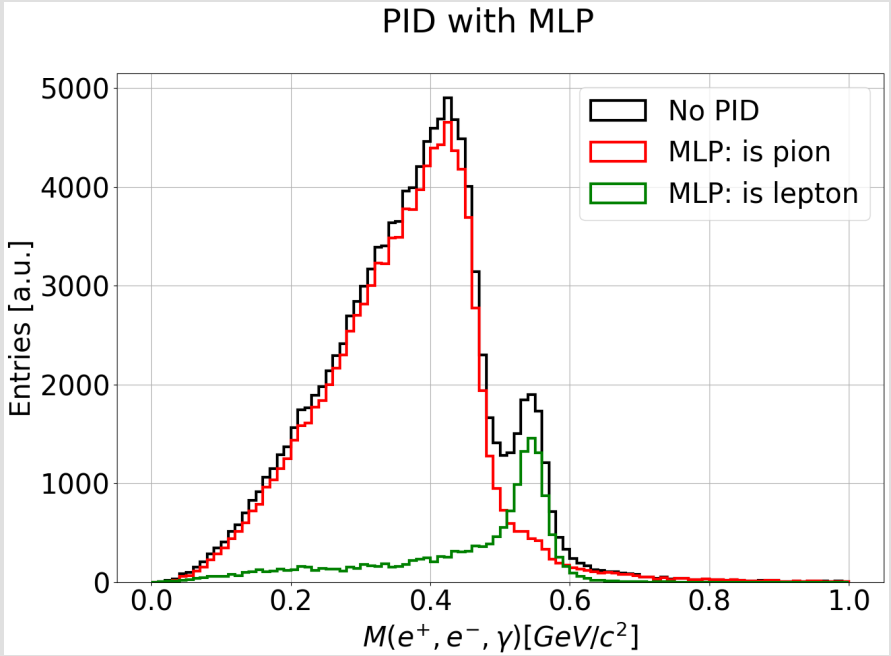
Random Forest and Neural Network Classifier

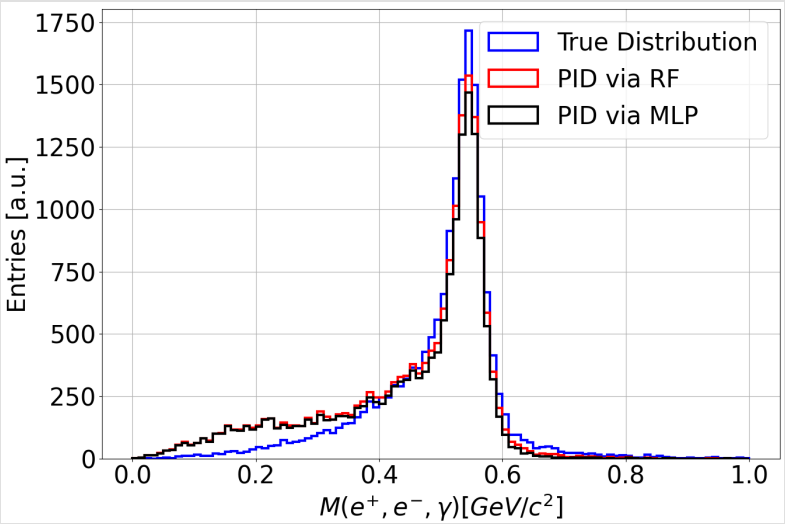
- Trained on simulated $e^{\pm}/\pi^{\pm}$ single tracks
- Hyper parameters set via random search
- One classifier type per charge and particle $\rightarrow$ 4 classifiers in total

Algorithm	Min. $d(t)$	TPR	FPR	threshold t	Accuracy	MCC
MLP (e^+/π^+)	0.009	0.92	0.05	0.45	0.93	0.87
MLP (e^-/π^-)	0.011	0.91	0.06	0.51	0.93	0.85
RF (e^+/π^+)	0.002	0.97	0.027	0.5	0.97	0.95
RF (e^-/π^-)	$6 \cdot 10^{-4}$	0.94	0.02	0.47	0.98	0.97

Reconstructing $\eta \to e^+e^-\gamma$ events



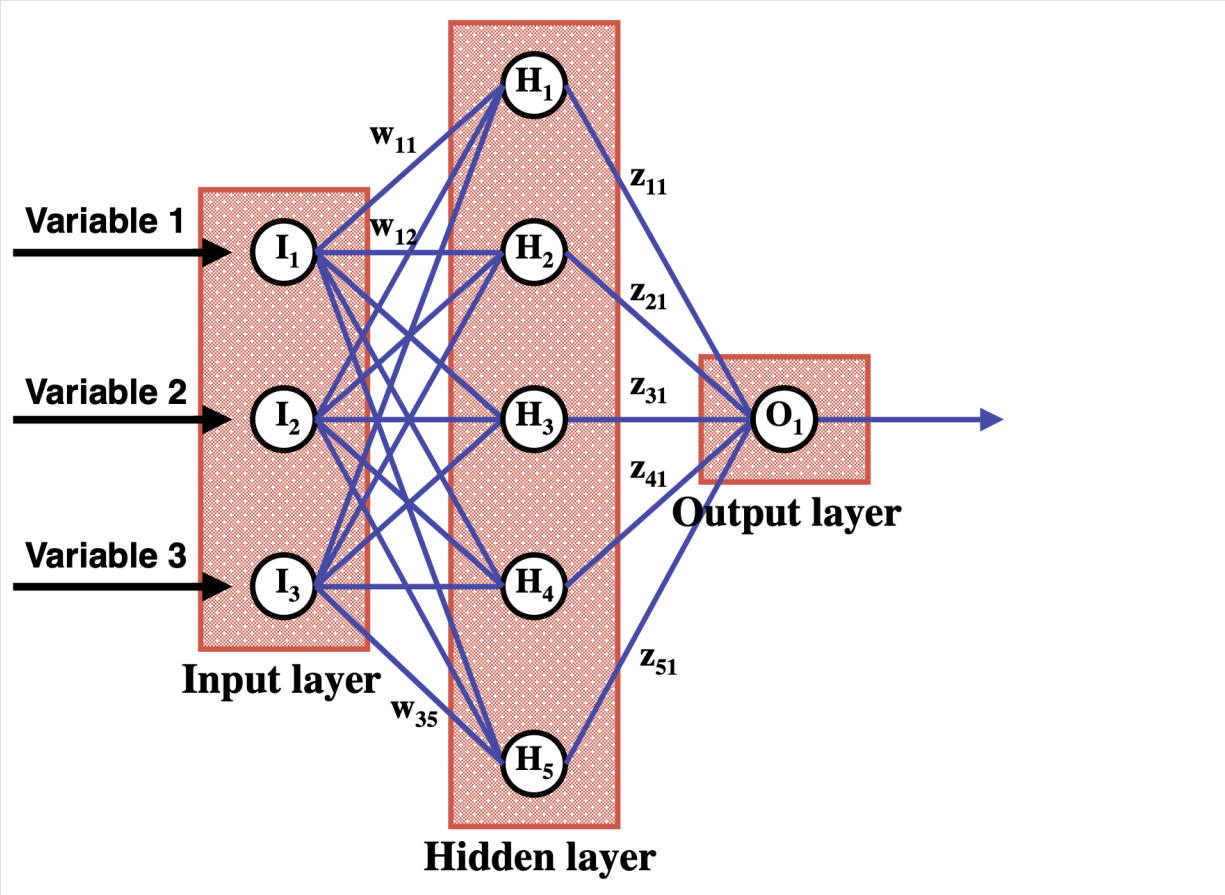




- Both classifiers are capable to suppress $\eta \to \pi^+\pi^-\gamma$ events
- Random forest outperforms neural network (see table on previous slide)
- $\eta \to e^+e^-\gamma$ signal not significantly altered by classifiers

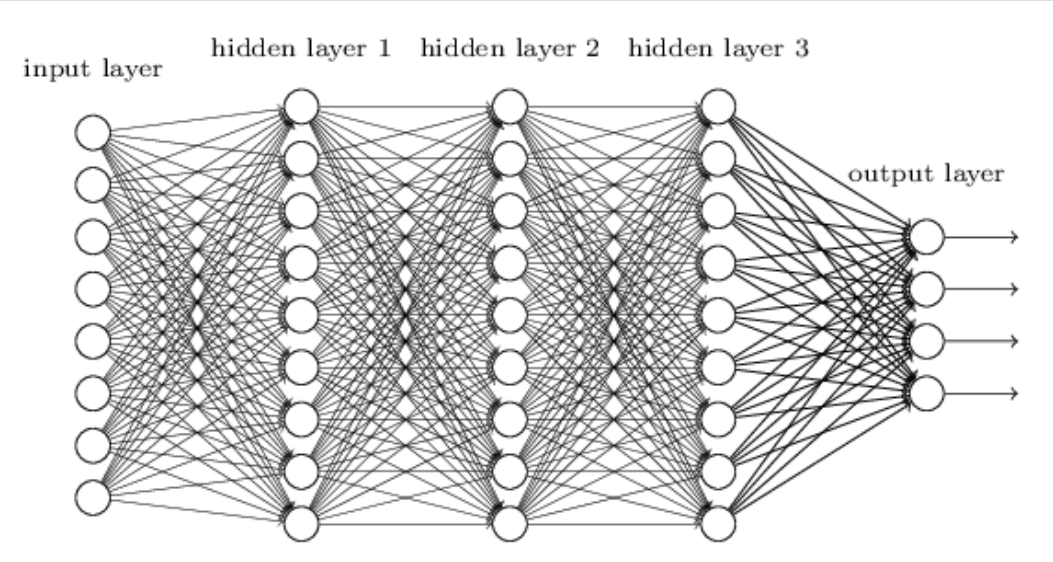
From Machine to Deep Learning (in a very naive picture)

- "Classical" Machine learning
  - Features obtained after pre-processing (calibration, analysis cuts, variable selection,...) $\rightarrow$ **feature-engineering**
  - Pre-processing encodes information into data
  - **Moderate model size** (e.g 1-2 hidden layers in a neural network)



From Machine to Deep Learning (in a very naive picture)

- "Classical" Machine learning
  - Features obtained after pre-processing (calibration, analysis cuts, variable selection,...) $\rightarrow$ **feature-engineering**
  - Pre-processing encodes information into data
  - **Moderate model size** (e.g 1-2 hidden layers in a neural network)
- Deep learning
  - Still machine learning, but uses neural networks only
  - **Leave out (certain) pre-processing steps** $\rightarrow$ Let the model do the work for you
  - The neural **network becomes deep** $\rightarrow$ Pre-processing is basically done in extra hidden layers



Picture taken from here

From Machine to Deep Learning (in a very naive picture)

- "Classical" Machine learning
  - Features obtained after pre-processing (calibration, analysis cuts, variable selection,...) $\rightarrow$ **feature-engineering**
  - Pre-processing encodes information into data
  - **Moderate model size** (e.g 1-2 hidden layers in a neural network)
- Deep learning
  - Still machine learning, but uses neural networks only
  - **Leave out (certain) pre-processing steps** $\rightarrow$ Let the model do the work for you
  - The neural **network becomes deep** $\rightarrow$ Pre-processing is basically done in extra hidden layers
- Deep learning is NOT trivial, but fortunately there are many frameworks
  - Pytorch
  - **Keras**
  - **Tensorflow**
  - ROOT
  - ...

DIY: Physics Data Analysis

1.) Go to: https://replit.com/@daniel49/FSUMLLecture4

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/

Machine / Deep Learning vs. Fitting a Function

	ML / DL	Fit Function (e.g Gauss, pol(N),...)
Parameters	weights, nodes,...	mean, width, $p_i \cdot x^i$, ...
Hyper Parameters	learning rate, architecture, learning epochs, tolerance,...	order of pol., include tails, fit iterations, tolerance,...
Optimizer	Adam, SGD, L-BFGS,...	χ^2, Log-Likelihood
Data Sets	training, validation,.. bootstrapping	same here
Performance Evaluation	Mean squared error,...	χ^2/ndf,...
Basic question	Which model?	Which fit function?
...	...	...

Machine / Deep Learning vs. Fitting a Function

- At its heart, using machine (deep) learning is not much different to fitting a function

Machine / Deep Learning vs. Fitting a Function

- At its heart, using machine (deep) learning is not much different to fitting a function
- Machine / Deep learning algorithms are successful, because they consist of many (~ 100 - $\sim 10^9$) parameters $\Rightarrow$ (Many) hyper parameters to tune

Machine / Deep Learning vs. Fitting a Function

- At its heart, using machine (deep) learning is not much different to fitting a function
- Machine / Deep learning algorithms are successful, because they consist of many (~ 100 - $\sim 10^9$) parameters $\Rightarrow$ (Many) hyper parameters to tune
- Provocative statement: Machine / Deep learning is:
  Reduction of N adjustable parameters to M $<<$ N adjustable hyper parameters

Machine / Deep Learning vs. Fitting a Function

- At its heart, using machine (deep) learning is not much different to fitting a function
- Machine / Deep learning algorithms are successful, because they consist of many (~ 100 - $\sim 10^9$) parameters $\Rightarrow$ (Many) hyper parameters to tune
- Provocative statement: Machine / Deep learning is:
  Reduction of N adjustable parameters to M $<<$ N adjustable hyper parameters
- Depending on the data set (complexity / size) $\Rightarrow$ Training of ML / DL is far more challenging than a pol. fit

Machine / Deep Learning vs. Fitting a Function

- At its heart, using machine (deep) learning is not much different to fitting a function
- Machine / Deep learning algorithms are successful, because they consist of many (~ 100 - $\sim 10^9$) parameters $\Rightarrow$ (Many) hyper parameters to tune
- Provocative statement: Machine / Deep learning is:
  Reduction of N adjustable parameters to M $<<$ N adjustable hyper parameters
- Depending on the data set (complexity / size) $\Rightarrow$ Training of ML / DL is far more challenging than a pol. fit
- Do not be afraid to use ML / DL

Open Questions

- I want to use machine learning in my analysis. Do I have to switch to DataFrames?

Open Questions

- I want to use machine learning in my analysis. Do I have to switch to DataFrames?
  - No you do not have to, but you can.

Open Questions

- I want to use machine learning in my analysis. Do I have to switch to DataFrames?
  - No you do not have to, but you can.
  - Use DataFrames to prepare the training data and train/evaluate your algorithm(s)

Open Questions

- I want to use machine learning in my analysis. Do I have to switch to DataFrames?
  - No you do not have to, but you can.
  - Use DataFrames to prepare the training data and train/evaluate your algorithm(s)
  - Once the algorithm(s) are trained you can deploy them to your (ROOT / Java / python / ...) analysis

Open Questions

- I want to use machine learning in my analysis. Do I have to switch to DataFrames?
  - No you do not have to, but you can.
  - Use DataFrames to prepare the training data and train/evaluate your algorithm(s)
  - Once the algorithm(s) are trained you can deploy them to your (ROOT / Java / python / ...) analysis
- Which framework is the best for me? Or which is the best to start with?

Open Questions

- I want to use machine learning in my analysis. Do I have to switch to DataFrames?
  - No you do not have to, but you can.
  - Use DataFrames to prepare the training data and train/evaluate your algorithm(s)
  - Once the algorithm(s) are trained you can deploy them to your (ROOT / Java / python / ...) analysis
- Which framework is the best for me? Or which is the best to start with?
  - It depends on what you want to do

Open Questions

- I want to use machine learning in my analysis. Do I have to switch to DataFrames?
  - No you do not have to, but you can.
  - Use DataFrames to prepare the training data and train/evaluate your algorithm(s)
  - Once the algorithm(s) are trained you can deploy them to your (ROOT / Java / python / ...) analysis
- Which framework is the best for me? Or which is the best to start with?
  - It depends on what you want to do
  - Efficient and quick filtering of large data sets without tuning too much $\rightarrow$ Apache Spark (not discussed in this lecture)

Open Questions

- I want to use machine learning in my analysis. Do I have to switch to DataFrames?
  - No you do not have to, but you can.
  - Use DataFrames to prepare the training data and train/evaluate your algorithm(s)
  - Once the algorithm(s) are trained you can deploy them to your (ROOT / Java / python / ...) analysis
- Which framework is the best for me? Or which is the best to start with?
  - It depends on what you want to do
  - Efficient and quick filtering of large data sets without tuning too much $\rightarrow$ Apache Spark (not discussed in this lecture)
  - R & D on many different algorithms with flexibility on parameter tuning $\rightarrow$ Scikit

Open Questions

- I want to use machine learning in my analysis. Do I have to switch to DataFrames?
  - No you do not have to, but you can.
  - Use DataFrames to prepare the training data and train/evaluate your algorithm(s)
  - Once the algorithm(s) are trained you can deploy them to your (ROOT / Java / python / ...) analysis
- Which framework is the best for me? Or which is the best to start with?
  - It depends on what you want to do
  - Efficient and quick filtering of large data sets without tuning too much $\rightarrow$ Apache Spark (not discussed in this lecture)
  - R & D on many different algorithms with flexibility on parameter tuning $\rightarrow$ Scikit
  - R & D on analyzing large and complex data sets, various options to customize own model $\rightarrow$ Tensorflow (not discussed in this lecture)

Open Questions

- I want to use machine learning in my analysis. Do I have to switch to DataFrames?
  - No you do not have to, but you can.
  - Use DataFrames to prepare the training data and train/evaluate your algorithm(s)
  - Once the algorithm(s) are trained you can deploy them to your (ROOT / Java / python / ...) analysis
- Which framework is the best for me? Or which is the best to start with?
  - It depends on what you want to do
  - Efficient and quick filtering of large data sets without tuning too much $\rightarrow$ Apache Spark (not discussed in this lecture)
  - R & D on many different algorithms with flexibility on parameter tuning $\rightarrow$ Scikit
  - R & D on analyzing large and complex data sets, various options to customize own model $\rightarrow$ Tensorflow (not discussed in this lecture)
  - Of course, there is some overlap between the different frameworks

Open Questions

- I want to use machine learning in my analysis. Do I have to switch to DataFrames?
  - No you do not have to, but you can.
  - Use DataFrames to prepare the training data and train/evaluate your algorithm(s)
  - Once the algorithm(s) are trained you can deploy them to your (ROOT / Java / python / ...) analysis
- Which framework is the best for me? Or which is the best to start with?
  - It depends on what you want to do
  - Efficient and quick filtering of large data sets without tuning too much $\rightarrow$ Apache Spark (not discussed in this lecture)
  - R & D on many different algorithms with flexibility on parameter tuning $\rightarrow$ Scikit
  - R & D on analyzing large and complex data sets, various options to customize own model $\rightarrow$ Tensorflow (not discussed in this lecture)
  - Of course, there is some overlap between the different frameworks
  - Good documentation for each of them

Open Questions

- I want to use machine learning in my analysis. Do I have to switch to DataFrames?
  - No you do not have to, but you can.
  - Use DataFrames to prepare the training data and train/evaluate your algorithm(s)
  - Once the algorithm(s) are trained you can deploy them to your (ROOT / Java / python / ...) analysis
- Which framework is the best for me? Or which is the best to start with?
  - It depends on what you want to do
  - Efficient and quick filtering of large data sets without tuning too much $\rightarrow$ Apache Spark (not discussed in this lecture)
  - R & D on many different algorithms with flexibility on parameter tuning $\rightarrow$ Scikit
  - R & D on analyzing large and complex data sets, various options to customize own model $\rightarrow$ Tensorflow (not discussed in this lecture)
  - Of course, there is some overlap between the different frameworks
  - Good documentation for each of them
- Which model shall I use for my analysis?

Open Questions

- I want to use machine learning in my analysis. Do I have to switch to DataFrames?
  - No you do not have to, but you can.
  - Use DataFrames to prepare the training data and train/evaluate your algorithm(s)
  - Once the algorithm(s) are trained you can deploy them to your (ROOT / Java / python / ...) analysis
- Which framework is the best for me? Or which is the best to start with?
  - It depends on what you want to do
  - Efficient and quick filtering of large data sets without tuning too much $\rightarrow$ Apache Spark (not discussed in this lecture)
  - R & D on many different algorithms with flexibility on parameter tuning $\rightarrow$ Scikit
  - R & D on analyzing large and complex data sets, various options to customize own model $\rightarrow$ Tensorflow (not discussed in this lecture)
  - Of course, there is some overlap between the different frameworks
  - Good documentation for each of them
- Which model shall I use for my analysis?
  - Again, it depends on what you want to do (classify small data set with few features vs. customized model on individual problem)

Open Questions

- I want to use machine learning in my analysis. Do I have to switch to DataFrames?
  - No you do not have to, but you can.
  - Use DataFrames to prepare the training data and train/evaluate your algorithm(s)
  - Once the algorithm(s) are trained you can deploy them to your (ROOT / Java / python / ...) analysis
- Which framework is the best for me? Or which is the best to start with?
  - It depends on what you want to do
  - Efficient and quick filtering of large data sets without tuning too much $\rightarrow$ Apache Spark (not discussed in this lecture)
  - R & D on many different algorithms with flexibility on parameter tuning $\rightarrow$ Scikit
  - R & D on analyzing large and complex data sets, various options to customize own model $\rightarrow$ Tensorflow (not discussed in this lecture)
  - Of course, there is some overlap between the different frameworks
  - Good documentation for each of them
- Which model shall I use for my analysis?
  - Again, it depends on what you want to do (classify small data set with few features vs. customized model on individual problem)
  - There is no need to use a deep model with 10^9 parameters, if a random forest classifier shows (after a careful analysis) a close to perfect performance

Open Questions

- I want to use machine learning in my analysis. Do I have to switch to DataFrames?
  - No you do not have to, but you can.
  - Use DataFrames to prepare the training data and train/evaluate your algorithm(s)
  - Once the algorithm(s) are trained you can deploy them to your (ROOT / Java / python / ...) analysis
- Which framework is the best for me? Or which is the best to start with?
  - It depends on what you want to do
  - Efficient and quick filtering of large data sets without tuning too much $\rightarrow$ Apache Spark (not discussed in this lecture)
  - R & D on many different algorithms with flexibility on parameter tuning $\rightarrow$ Scikit
  - R & D on analyzing large and complex data sets, various options to customize own model $\rightarrow$ Tensorflow (not discussed in this lecture)
  - Of course, there is some overlap between the different frameworks
  - Good documentation for each of them
- Which model shall I use for my analysis?
  - Again, it depends on what you want to do (classify small data set with few features vs. customized model on individual problem)
  - There is no need to use a deep model with 10^9 parameters, if a random forest classifier shows (after a careful analysis) a close to perfect performance
  - Do not try to fine tune a random forest classifier on a complex data set (e.g. ~ 100 different correlated variables), if you can not achieve a reasonable performance

References and Further Reading

- Most of the information / code pieces have been taken from / inspired by the following web-sites: (the blue items are clickable links)
  - **Apache Spark**
  - **Apache Spark ML**
  - **Pyspark documentation**
  - **Python scikit-learn**
  - **Tensorflow**
  - **Tensorflow Keras Models**
  - **Keras**
  - **stackoverflow**
  - **distill.pub: Current problems and issues in machine/deep learning**
  - **A Recipe for Training Neural Networks (Andrej Karpathy)**
  - "Hands-On Machine Learning with Scikit-Learn, Keras & Tensorflow", by Aurélien Géron $\rightarrow$ Really good book!
  - Talks from the **deep learning for science school 2019**
  - Talks from the **deep learning for science school 2020**

References and Further Reading

- Most of the information / code pieces have been taken from / inspired by the following web-sites: (the blue items are clickable links)
  - **Apache Spark**
  - **Apache Spark ML**
  - **Pyspark documentation**
  - **Python scikit-learn**
  - **Tensorflow**
  - **Tensorflow Keras Models**
  - **Keras**
  - **stackoverflow**
  - **distill.pub: Current problems and issues in machine/deep learning**
  - **A Recipe for Training Neural Networks (Andrej Karpathy)**
  - "Hands-On Machine Learning with Scikit-Learn, Keras & Tensorflow", by Aurélien Géron $\rightarrow$ Really good book!
  - Talks from the **deep learning for science school 2019**
  - Talks from the **deep learning for science school 2020**
- If you are stuck with a problem / framework $\Rightarrow$ Do not spend weeks to solve it $\Rightarrow$ Look it up (stackoverflow, google, yahoo,...), most likely someone has a similar problem

References and Further Reading

- Most of the information / code pieces have been taken from / inspired by the following web-sites: (the blue items are clickable links)
  - **Apache Spark**
  - **Apache Spark ML**
  - **Pyspark documentation**
  - **Python scikit-learn**
  - **Tensorflow**
  - **Tensorflow Keras Models**
  - **Keras**
  - **stackoverflow**
  - **distill.pub: Current problems and issues in machine/deep learning**
  - **A Recipe for Training Neural Networks (Andrej Karpathy)**
  - "Hands-On Machine Learning with Scikit-Learn, Keras & Tensorflow", by Aurélien Géron $\rightarrow$ Really good book!
  - Talks from the **deep learning for science school 2019**
  - Talks from the **deep learning for science school 2020**
- If you are stuck with a problem / framework $\Rightarrow$ Do not spend weeks to solve it $\Rightarrow$ Look it up (stackoverflow, google, yahoo,...), most likely someone has a similar problem
- Try not to start from scratch (sometimes not avoidable)

References and Further Reading

- Most of the information / code pieces have been taken from / inspired by the following web-sites: (the blue items are clickable links)
  - **Apache Spark**
  - **Apache Spark ML**
  - **Pyspark documentation**
  - **Python scikit-learn**
  - **Tensorflow**
  - **Tensorflow Keras Models**
  - **Keras**
  - **stackoverflow**
  - **distill.pub: Current problems and issues in machine/deep learning**
  - **A Recipe for Training Neural Networks (Andrej Karpathy)**
  - "Hands-On Machine Learning with Scikit-Learn, Keras & Tensorflow", by Aurélien Géron $\rightarrow$ Really good book!
  - Talks from the **deep learning for science school 2019**
  - Talks from the **deep learning for science school 2020**
- If you are stuck with a problem / framework $\Rightarrow$ Do not spend weeks to solve it $\Rightarrow$ Look it up (stackoverflow, google, yahoo,...), most likely someone has a similar problem
- Try not to start from scratch (sometimes not avoidable)
- Again, there is no easy / quick way to learn all the aspects of machine learning

References and Further Reading

- Most of the information / code pieces have been taken from / inspired by the following web-sites: (the blue items are clickable links)
  - **Apache Spark**
  - **Apache Spark ML**
  - **Pyspark documentation**
  - **Python scikit-learn**
  - **Tensorflow**
  - **Tensorflow Keras Models**
  - **Keras**
  - **stackoverflow**
  - **distill.pub: Current problems and issues in machine/deep learning**
  - **A Recipe for Training Neural Networks (Andrej Karpathy)**
  - "Hands-On Machine Learning with Scikit-Learn, Keras & Tensorflow", by Aurélien Géron $\rightarrow$ Really good book!
  - Talks from the **deep learning for science school 2019**
  - Talks from the **deep learning for science school 2020**
- If you are stuck with a problem / framework $\Rightarrow$ Do not spend weeks to solve it $\Rightarrow$ Look it up (stackoverflow, google, yahoo,...), most likely someone has a similar problem
- Try not to start from scratch (sometimes not avoidable)
- Again, there is no easy / quick way to learn all the aspects of machine learning
- My personal recommendation: Try it out yourself! (i.e. pick an example data set and start playing around)

Final Remarks

- Tried to give a rough impression on machine learning in physics data analysis

Final Remarks

- Tried to give a rough impression on machine learning in physics data analysis
- Did not cover all aspects in machine learning

Final Remarks

- Tried to give a rough impression on machine learning in physics data analysis
- Did not cover all aspects in machine learning
- Not all shown code snippets are best programming practice $\rightarrow$ They shall simply give you an idea on how to implement various functions

Final Remarks

- Tried to give a rough impression on machine learning in physics data analysis
- Did not cover all aspects in machine learning
- Not all shown code snippets are best programming practice $\rightarrow$ They shall simply give you an idea on how to implement various functions
- scikit provides a much larger functionality than shown in this lecture

Final Remarks

- Tried to give a rough impression on machine learning in physics data analysis
- Did not cover all aspects in machine learning
- Not all shown code snippets are best programming practice $\rightarrow$ They shall simply give you an idea on how to implement various functions
- scikit provides a much larger functionality than shown in this lecture
- Many approaches shown here are based on my own experience, i.e. NOT the ultimate truth $\rightarrow$ Someone else might have tackled the problems differently

Final Remarks

- Tried to give a rough impression on machine learning in physics data analysis
- Did not cover all aspects in machine learning
- Not all shown code snippets are best programming practice $\rightarrow$ They shall simply give you an idea on how to implement various functions
- scikit provides a much larger functionality than shown in this lecture
- Many approaches shown here are based on my own experience, i.e. NOT the ultimate truth $\rightarrow$ Someone else might have tackled the problems differently
- I hope you had fun!