The BSM-AI project

Eur. Phys. J. C (2017) 77: 257
DOI: 10.1140/epjc/s10052-017-4814-9

SUSY-AI: Reinterpreting SUSY LHC Limits with Machine Learning

Sascha Caron, Jong Soo Kim, Krzysztof Rolbiecki, Roberto Ruiz de Austri, Bob Stienen
b.stienen@science.ru.nl

netherlands eScience center

Radboud University

Supersymmetry (SUSY)

- Theoretical model of new physics, introducing a symmetry between fermions and bosons

- Predicts > 2 times the amount of particles we know from experiment:
  SM particles and SUSY partners of these particles

- In perfect SUSY: SM particles and their partners only differ in spin
  In broken SUSY: e.g. masses may differ, but coupling types are identical

Supersymmetry (SUSY)

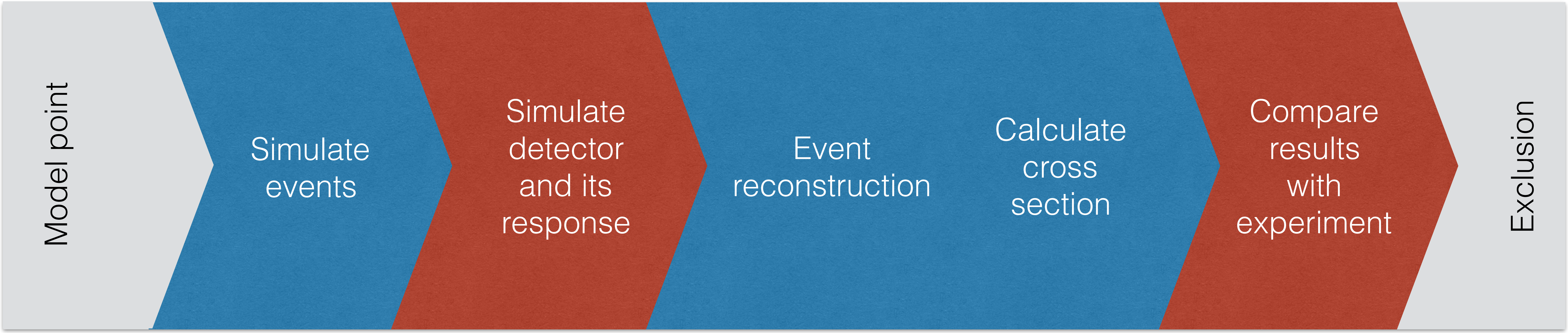
- Minimal version (MSSM) adds ~O(100) free parameters

- ~19 parameters if only looking at the phenomenologically relevant ones (pMSSM)

Regardless: SUSY has not been discovered (yet), so…

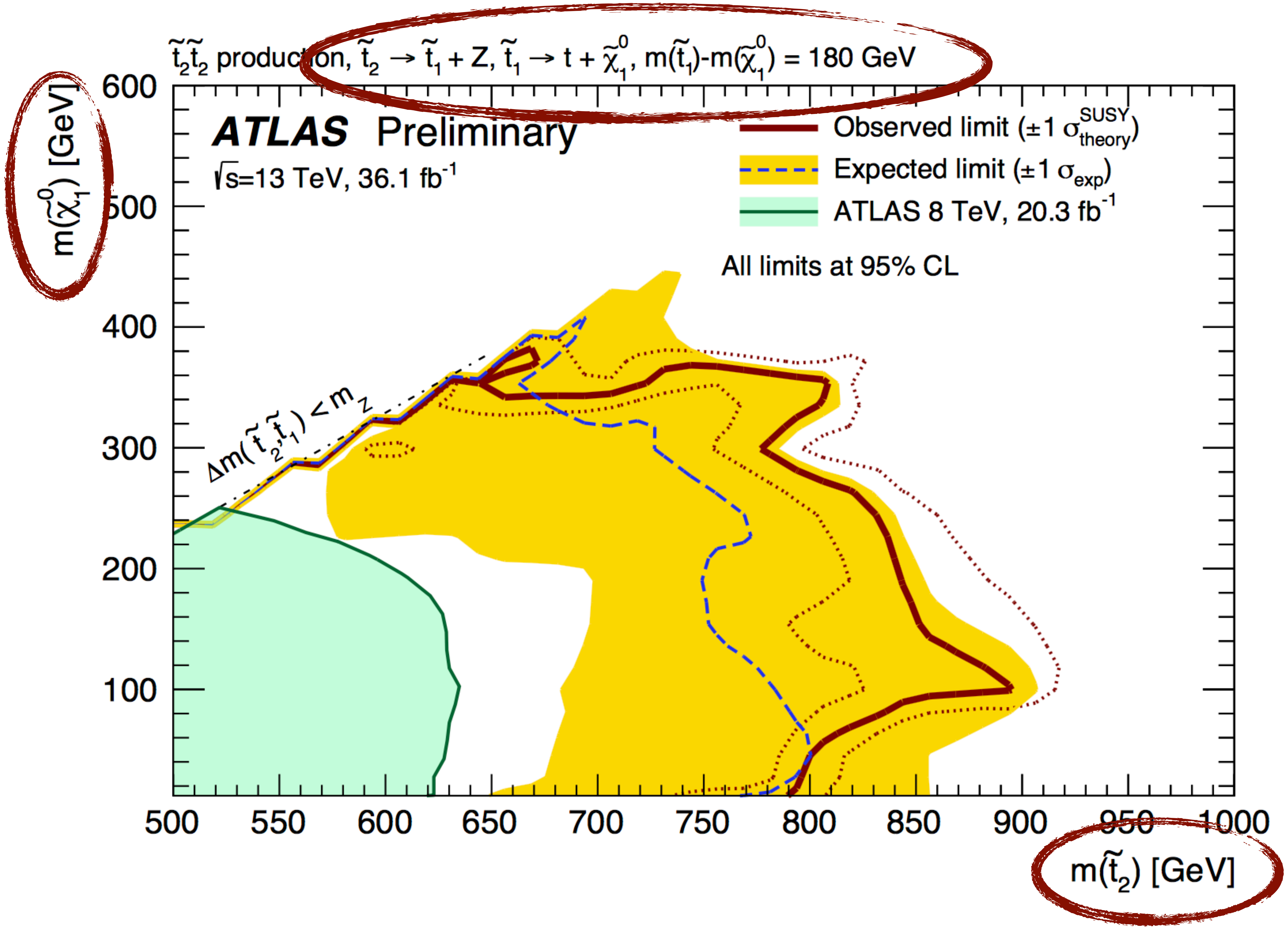
The Analysis Problem

Time = O(hours)



The Plot Problem

atlas_conf_2017_019



Contents

- Machine Learning
- Data and approach
- Results
- Confidence
- Applicability
- Conclusions

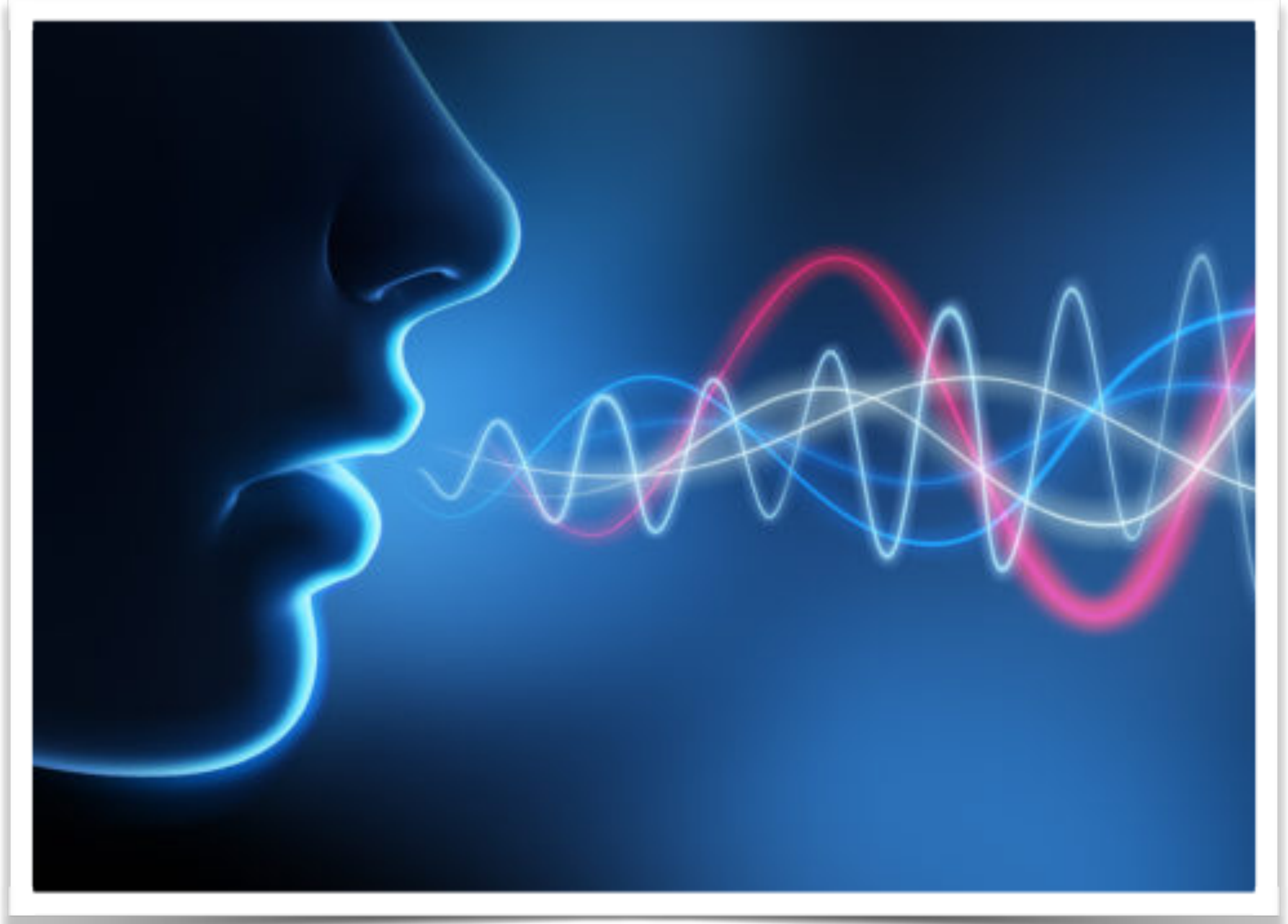
Machine Learning

Getting to know our machinery

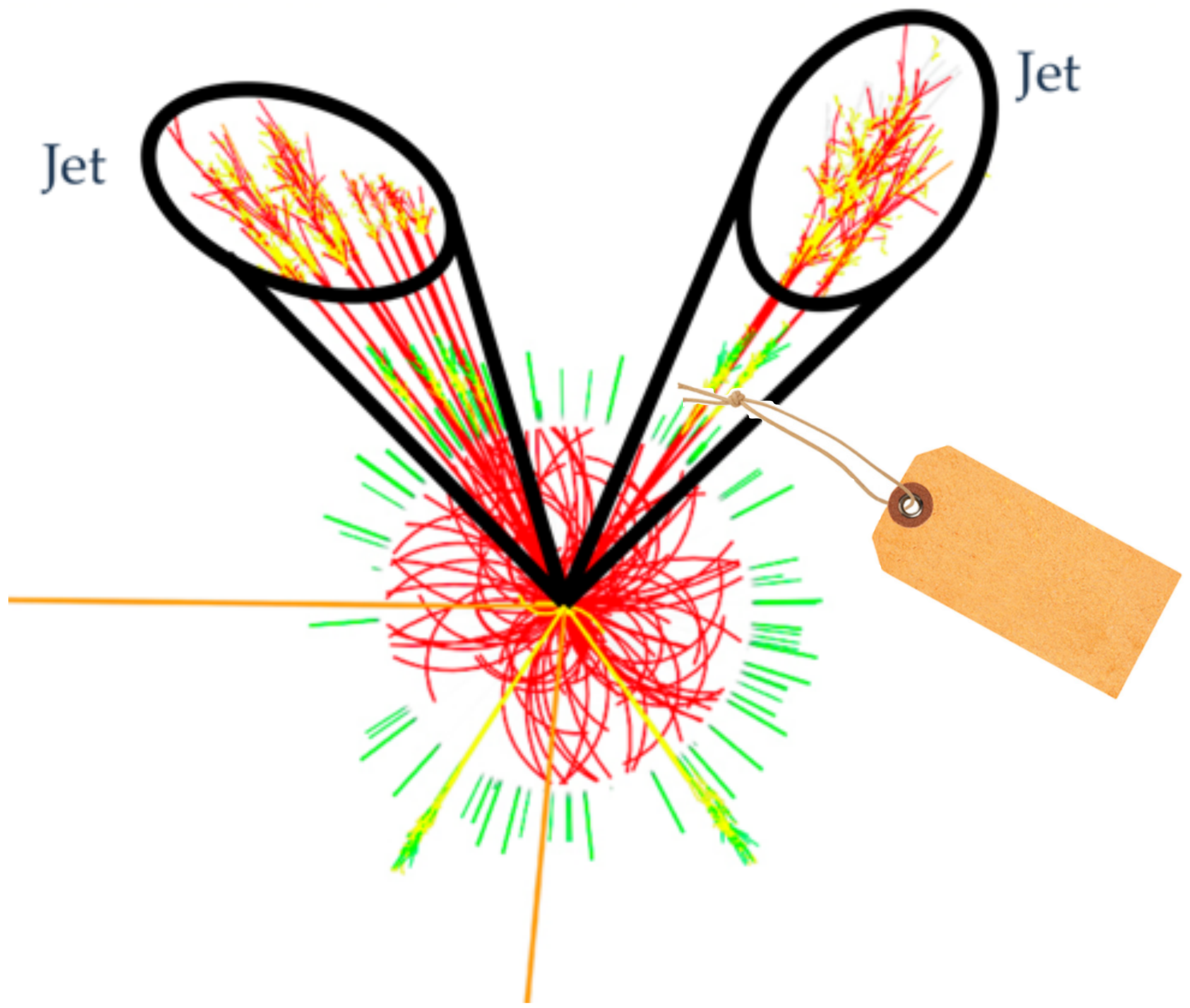
Machine Learning

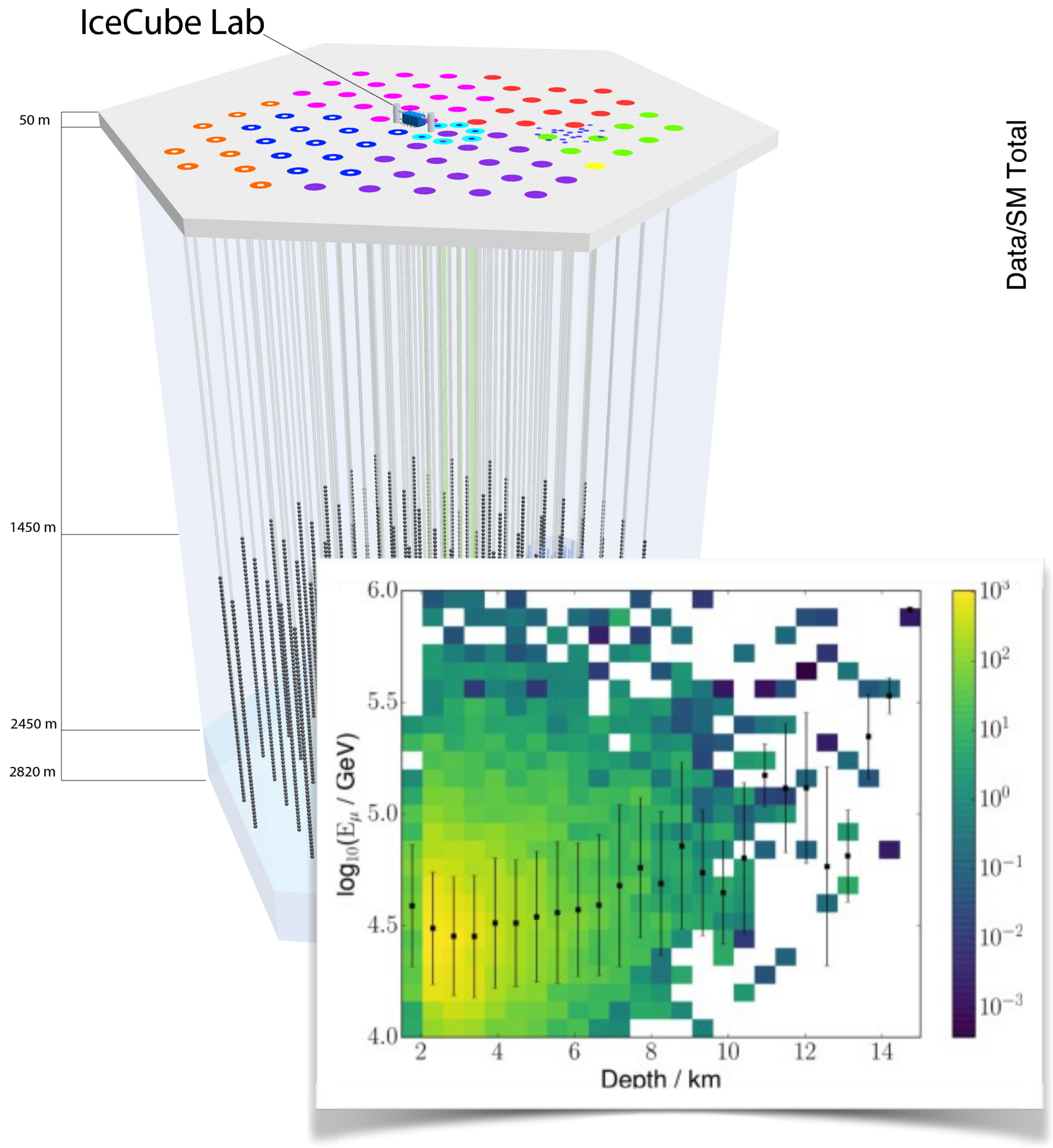
- Statistics of big data
  - Prediction of data properties based on example (training) data via smart interpolation
- Wide range of algorithms…
  (e.g. boosted decision trees, k-nearest neighbours, neural networks)
- … and applications

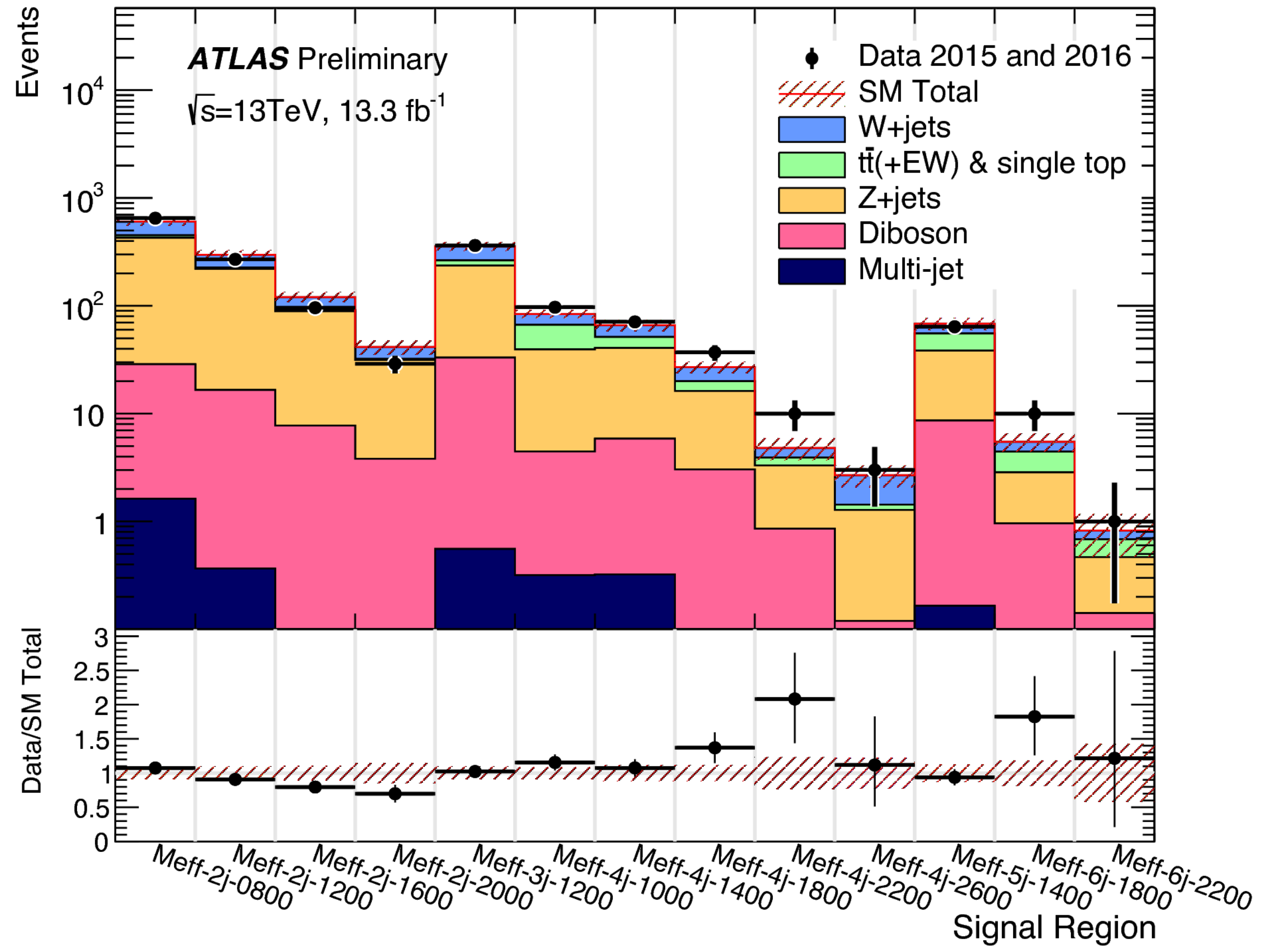
Examples of Machine Learning



Examples of ML in HEP

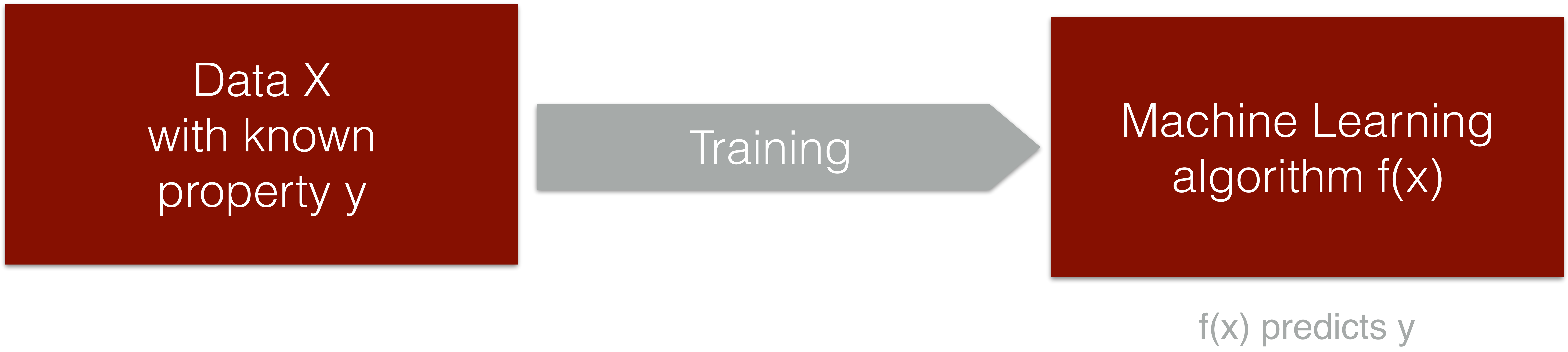




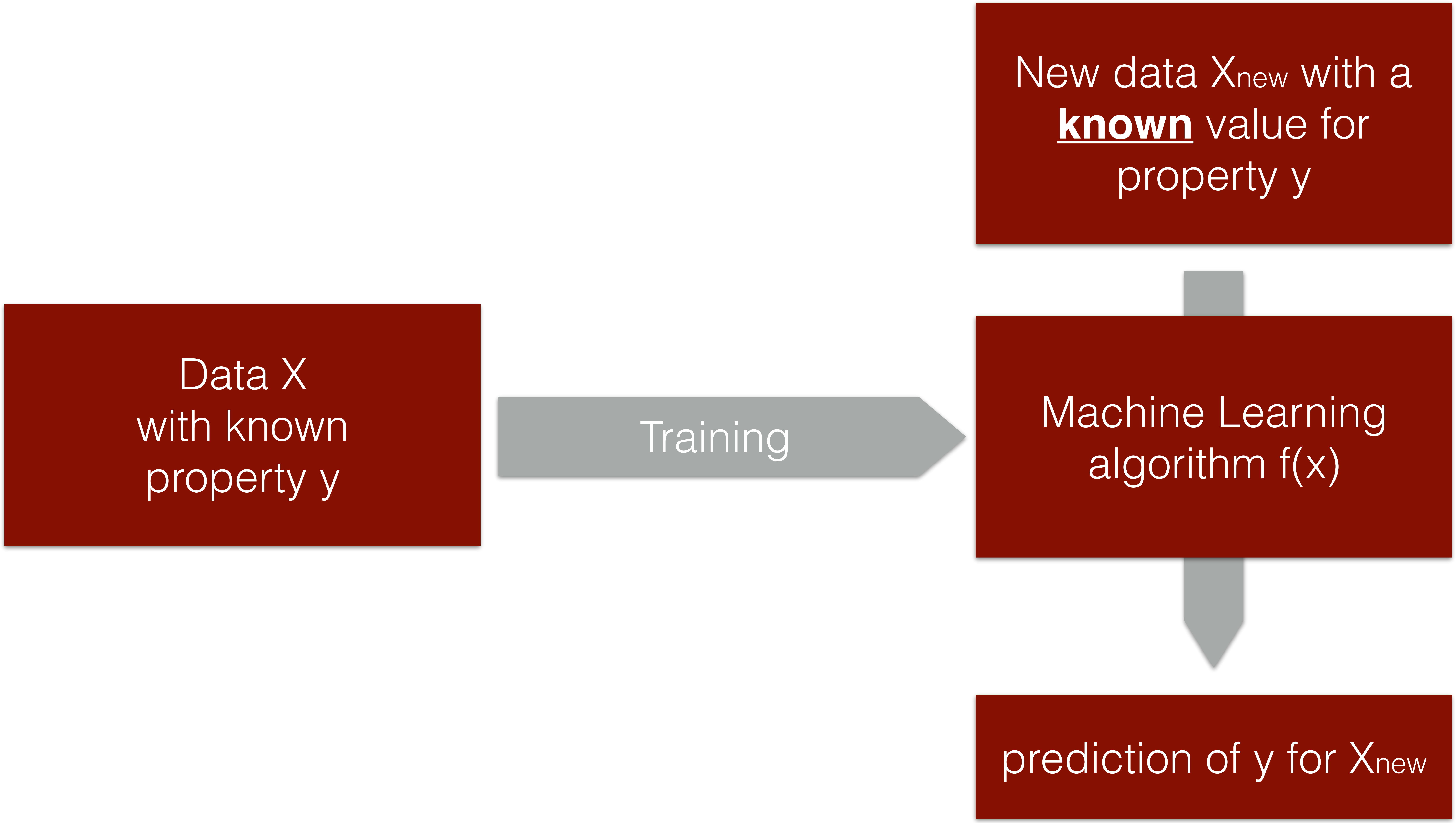


All these are examples of property prediction

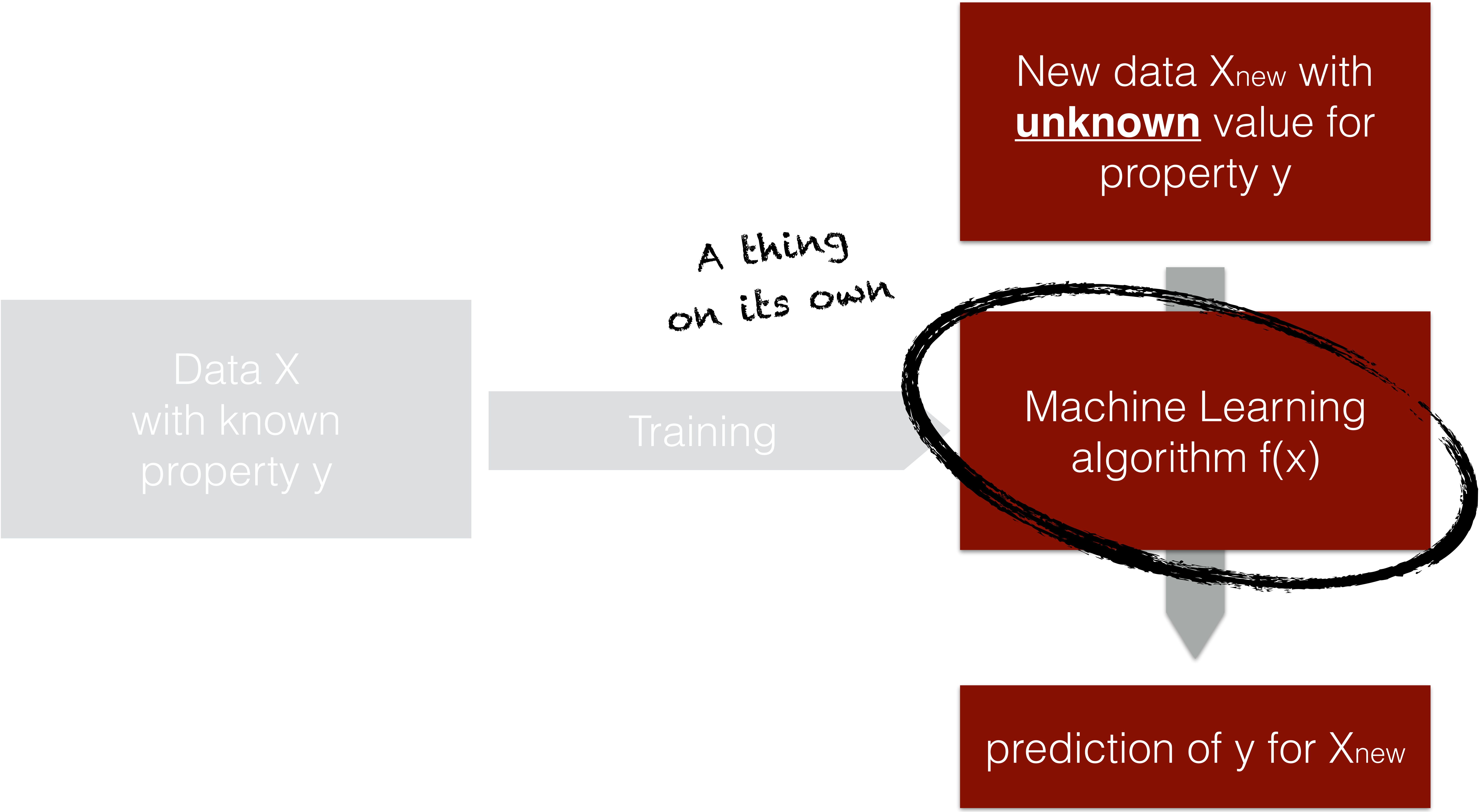
Property prediction



Property prediction



Property prediction



The idea

Machine Learning as a tool to reinterpret experimental results and to determine the exclusion of model points

Training data

>300,000 model points in pMSSM with exclusion as determined by:

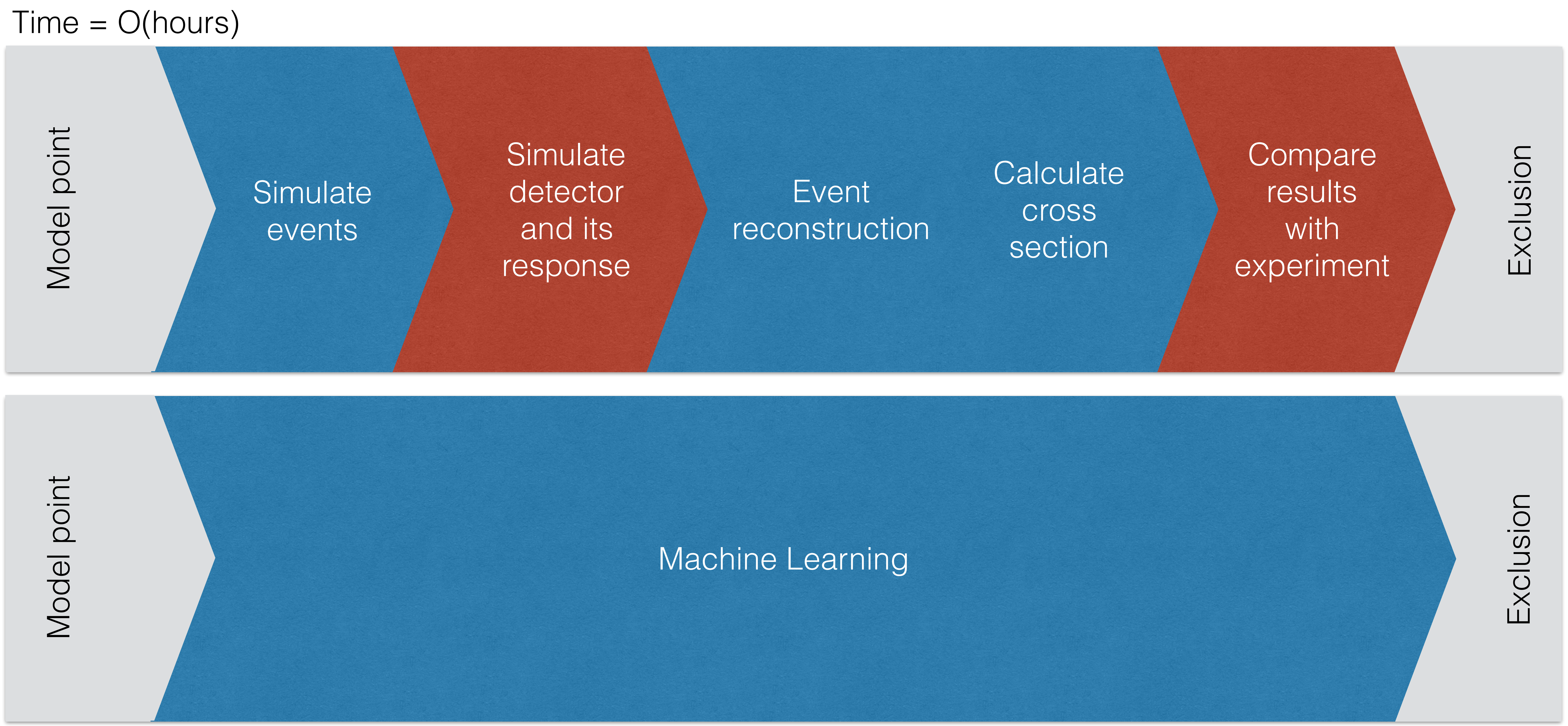
- ATLAS at 8TeV [arXiv: 1508.06608]
- Barr & Liu at 13TeV [arXiv: 1605.09502]

All data has correct Higgs mass and relic density (upper limit), and is not excluded by precision experiments (LHCb, e.g. B_s decay) or by LUX or Xenon100

Algorithm

Random Forest (a smartly constructed set of decision trees) in scikit-learn Python package

The idea


Time = O(hours)
Model point
Simulate events
Simulate detector and its response
Event reconstruction
Calculate cross section
Compare results with experiment
Exclusion
Model point
Machine Learning
Exclusion

Data and Approach

What is the problem exactly?

Dataset: pMSSM

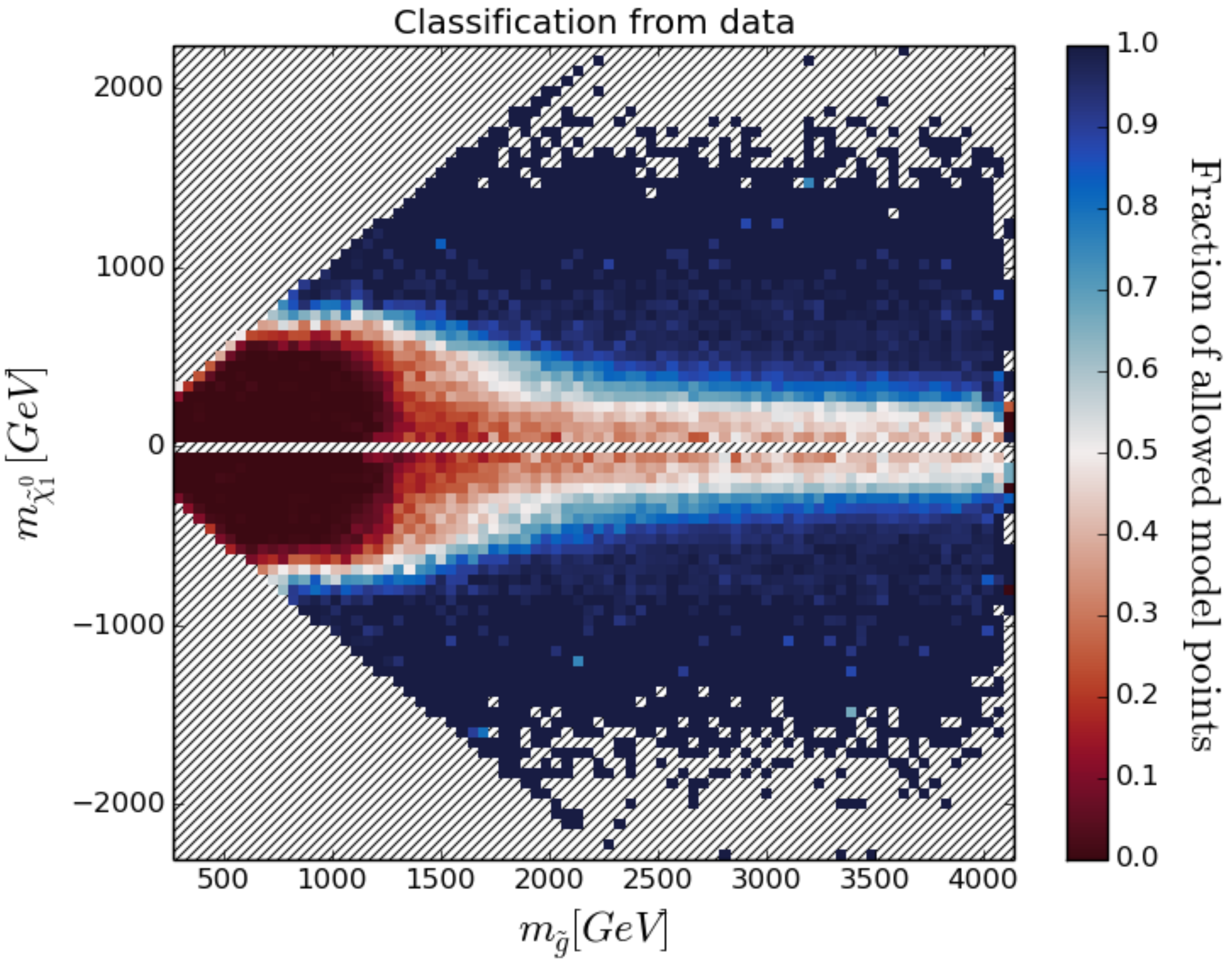
1. **R-parity is conserved**
2. **No symmetry breaking mechanism is assumed**
3. Minimal flavour violation
4. **Lightest neutralino is the lightest SUSY particle**
5. First two sfermion generations are mass degenerate
6. First two generations have negligible Yukawa couplings

Parameter	Description	Scanned range
$m_{\tilde{L}_1}$	$1^{\text{st}}/2^{\text{nd}}$ gen. $SU(2)$ doublet soft breaking slepton mass	[90 GeV, 4 TeV]
$m_{\tilde{E}_1}$	$1^{\text{st}}/2^{\text{nd}}$ gen. $SU(2)$ singlet soft breaking slepton mass	[90 GeV, 4 TeV]
$m_{\tilde{L}_3}$	3^{rd} gen. $SU(2)$ doublet soft breaking slepton mass	[90 GeV, 4 TeV]
$m_{\tilde{E}_3}$	3^{rd} gen. $SU(2)$ singlet soft breaking slepton mass	[90 GeV, 4 TeV]
$m_{\tilde{Q}_1}$	$1^{\text{st}}/2^{\text{nd}}$ gen. $SU(2)$ doublet soft breaking squark mass	[200 GeV, 4 TeV]
$m_{\tilde{U}_1}$	$1^{\text{st}}/2^{\text{nd}}$ gen. $SU(2)$ singlet soft breaking squark mass	[200 GeV, 4 TeV]
$m_{\tilde{D}_1}$	$1^{\text{st}}/2^{\text{nd}}$ gen. $SU(2)$ singlet soft breaking squark mass	[200 GeV, 4 TeV]
$m_{\tilde{Q}_3}$	3^{rd} gen. $SU(2)$ doublet soft breaking squark mass	[100 GeV, 4 TeV]
$m_{\tilde{U}_3}$	3^{rd} gen. $SU(2)$ singlet soft breaking squark mass	[100 GeV, 4 TeV]
$m_{\tilde{D}_3}$	3^{rd} gen. $SU(2)$ singlet soft breaking squark mass	[100 GeV, 4 TeV]
A_t	Stop trilinear coupling	[−8 TeV, 8 TeV]
A_b	Sbottom trilinear coupling	[−4 TeV, 4 TeV]
A_τ	Stau trilinear coupling	[−4 TeV, 4 TeV]
$\|\mu\|$	Higgsino mass parameter	[80 GeV, 4 TeV]
$\|M_1\|$	Bino mass parameter	[0 TeV, 4 TeV]
$\|M_2\|$	Wino mass parameter	[70 GeV, 4 TeV]
M_3	Gluino mass parameter	[200 GeV, 4 TeV]
M_A	Pseudoscalar Higgs mass	[100 GeV, 4 TeV]
$\tan\beta$	Ratio of vacuum expectation values	[1, 60]

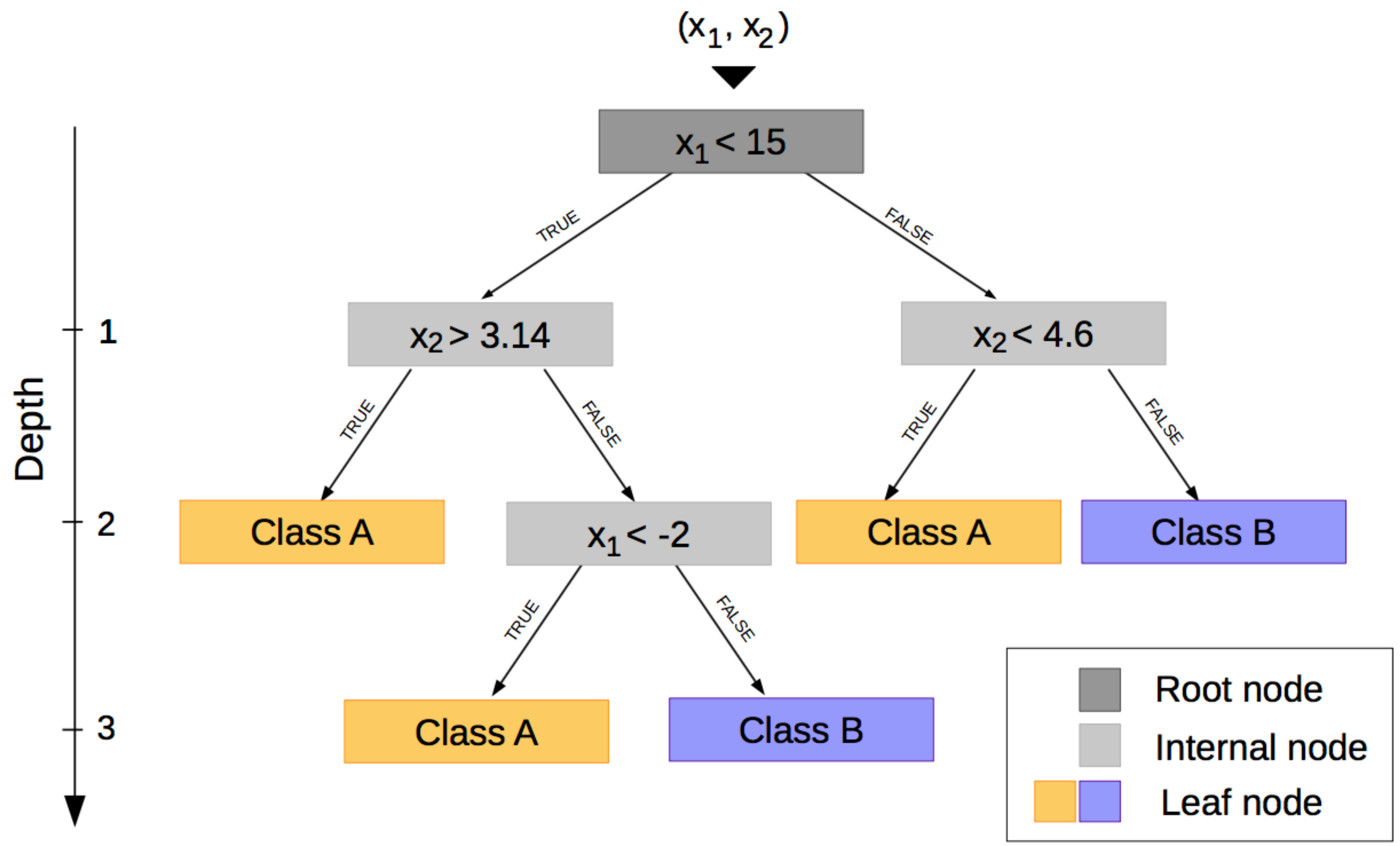
Analyses

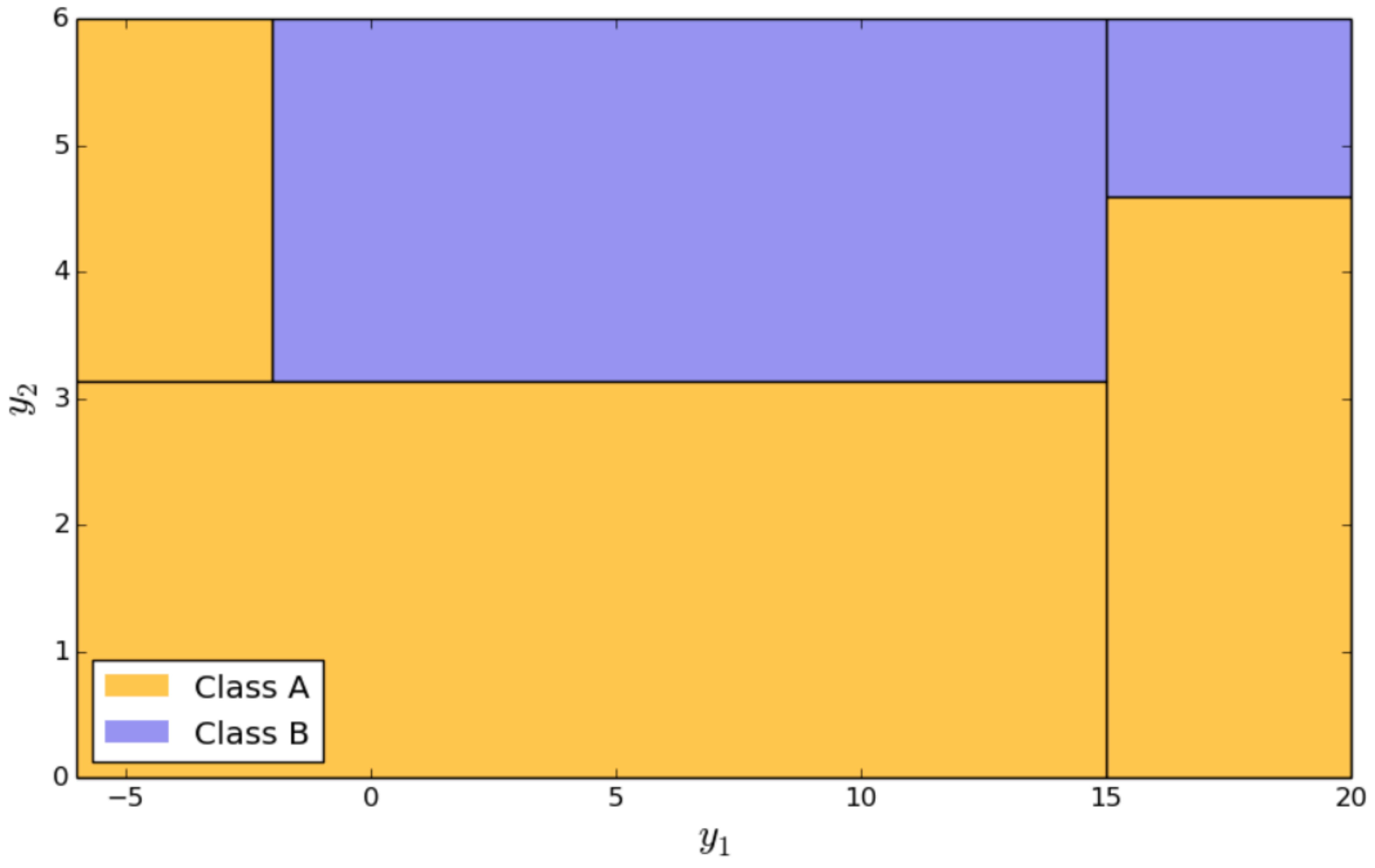
Final State	Category
0 lepton + 2–6 jets + $\not{E}_T$	Inclusive
0 lepton + 7–10 jets + $\not{E}_T$	
1 lepton + jets + $\not{E}_T$	
$\tau(\tau/\ell)$ + jets + $\not{E}_T$	
SS/3 lepton + jets + $\not{E}_T$	
b-jets + 0/1 lepton + $\not{E}_T$	
monojet	
0 lepton stop search	Third generation squarks
1 lepton stop search	
2 lepton stop search	
monojet search	
stop search with Z in final state	
2b-jets sbottom search	
asymmetric stop search	
1 lepton plus Higgs final state	Electroweak
dilepton final state	
2τ final state	
trilepton final state	
four-lepton final state	
disappearing track	
Long-lived particle search	Other
$H/A \to \tau\tau$ search	

Dataset: pMSSM



Decision trees

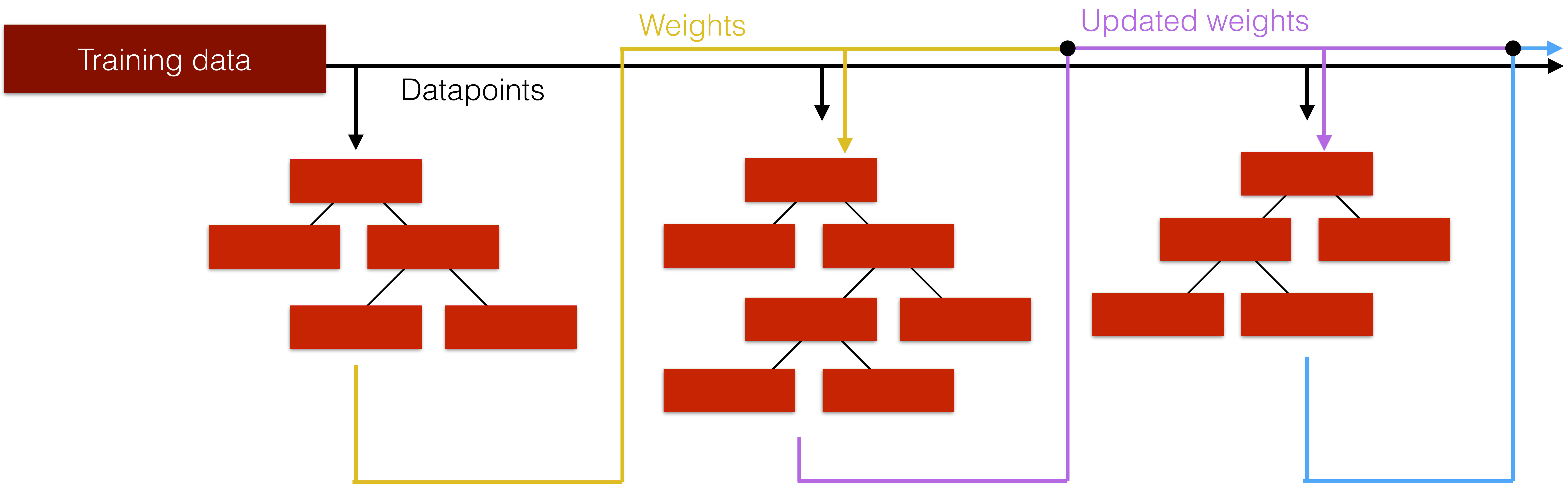




Boosted decision trees

- Trees are combined (ensemble) into single classifier

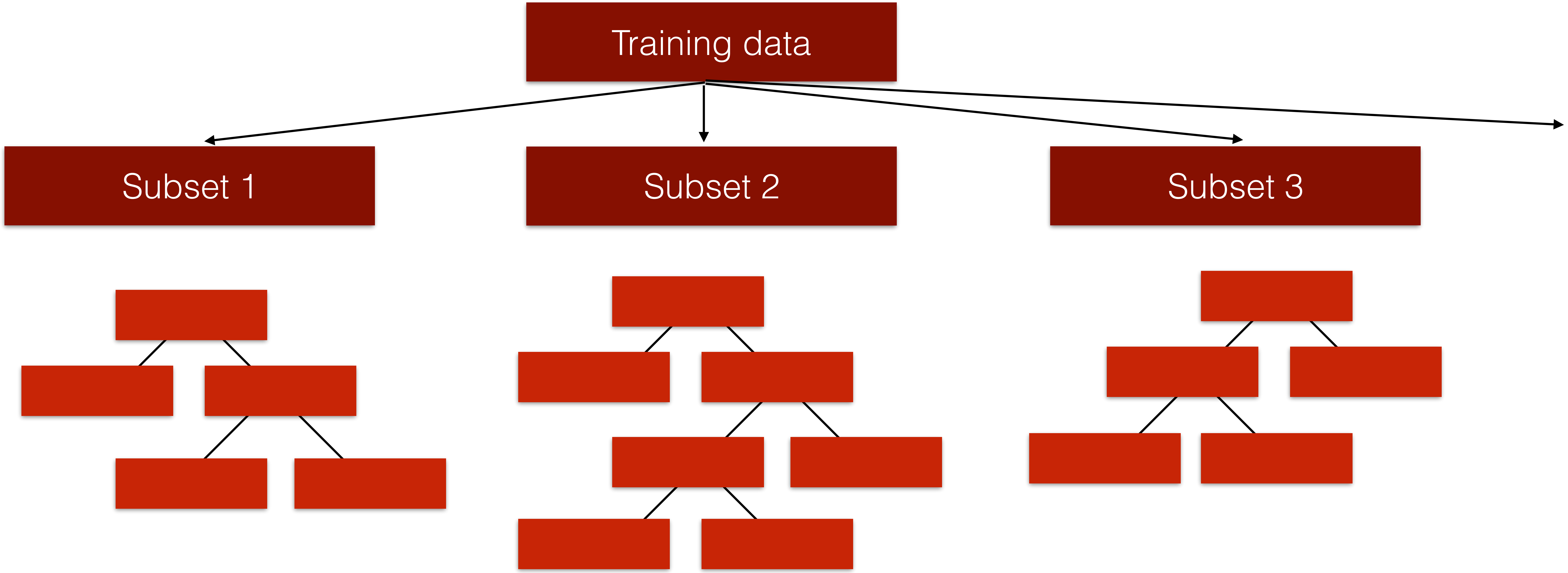
- Each next tree is trained on same data set with updated weights, so misclassifications of previous tree(s) are predicted better



Random Forest (1/2)

- Combination of multiple decision trees (ensemble), prediction by majority vote

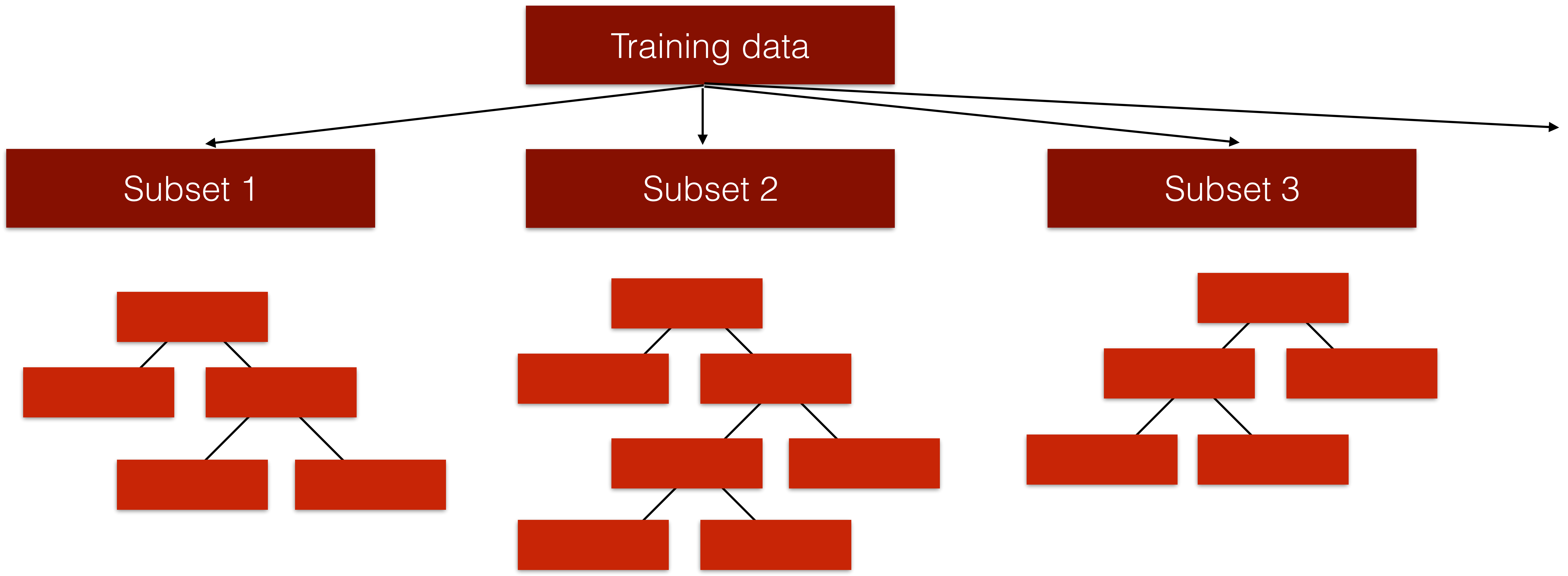
- Introducing the randomness in the forest: trees are constructed with *bagging* (each tree trained on unique subset of training data)



Random Forest (2/2)

- Subsets are of the same size as training data set and data points are selected with replacement —> same datapoint can be selected multiple times
  ~63.2% of model points in subset are unique

- Moreover, only subset of parameters is considered at each node to split on



Random Forest vs Boosted Decision Trees (1/2)

- Both are sets of decisions trees, but constructed in different ways: bagging vs boosting respectively
  - Boosting: train each tree iteratively to do better on the mistakes of the previous trees (increase weight of misclassified points by previous tree)
  - Bagging: introduce randomness in training of the trees and average over them.
- Both bagging and boosting are well understood methods to reduce overtraining.

Random Forest vs Boosted Decision Trees (2/2)

- Boosting reduces in theory both bias and variance, but does tend to overfit sometimes. It uses shorter trees and is faster in training and use.

- Bagging is less sensitive to outliers and its output is more closely linked to prediction confidence.
  Also: out-of-bag estimation

Out-of-bag estimation

- Only ~63.2% of training data is used in training of a single tree

- Use remaining 37.8% for independent testing

- This can be done for every single tree in the forest

- Lots of trees —> independent test on *all* training data

- Combined output is independent prediction by forest on its training data —> useful for testing purposes
  No train:test split needed!

Random Forest configuration

Optimal configuration was found via a grid search

- Number of trees
  900

- Maximum features considered each split
  12 (out of a total of 19)

- Maximum depth of each individual tree
  30

Results

Applying Machine Learning

Out-of-bag vs train:test split

Accuracy:
(TP+TN) / all

Precision:
TP / (TP+FP)

Sensitivity
TP / (TP+FN)

Negative prediction value
TN / (TN+FN)

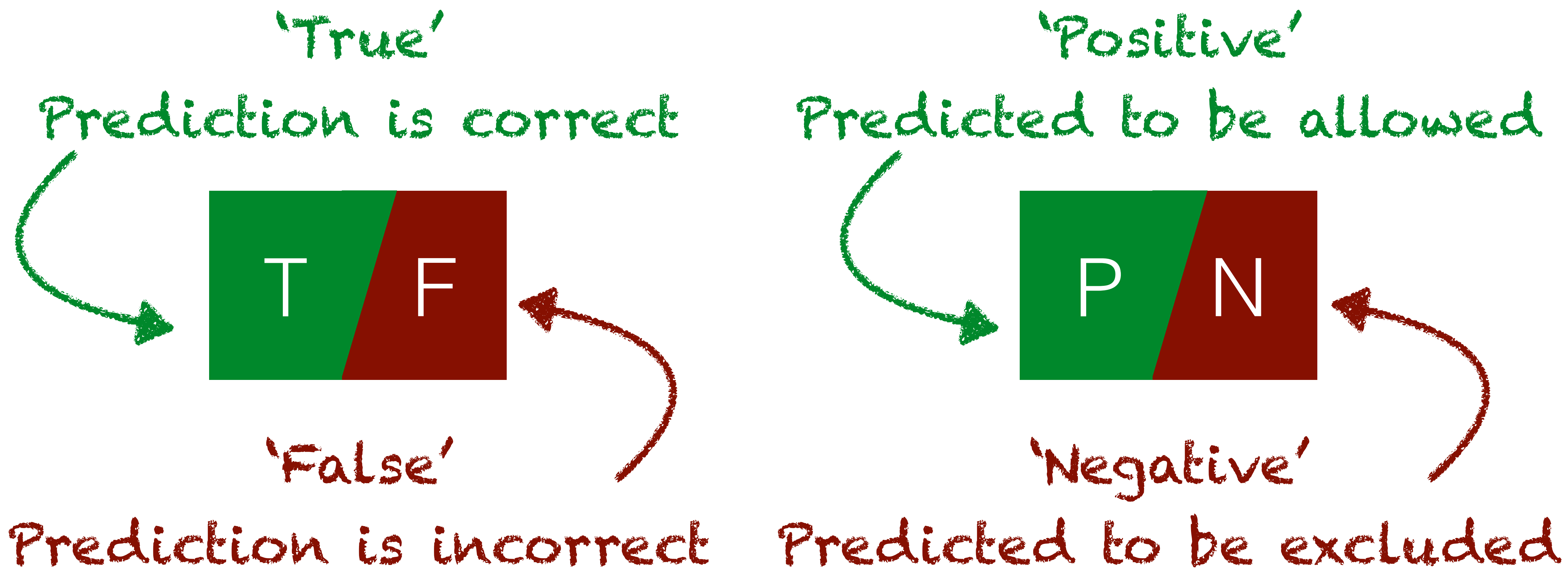
Specificity
TN / (TN+FP)

Out-of-bag

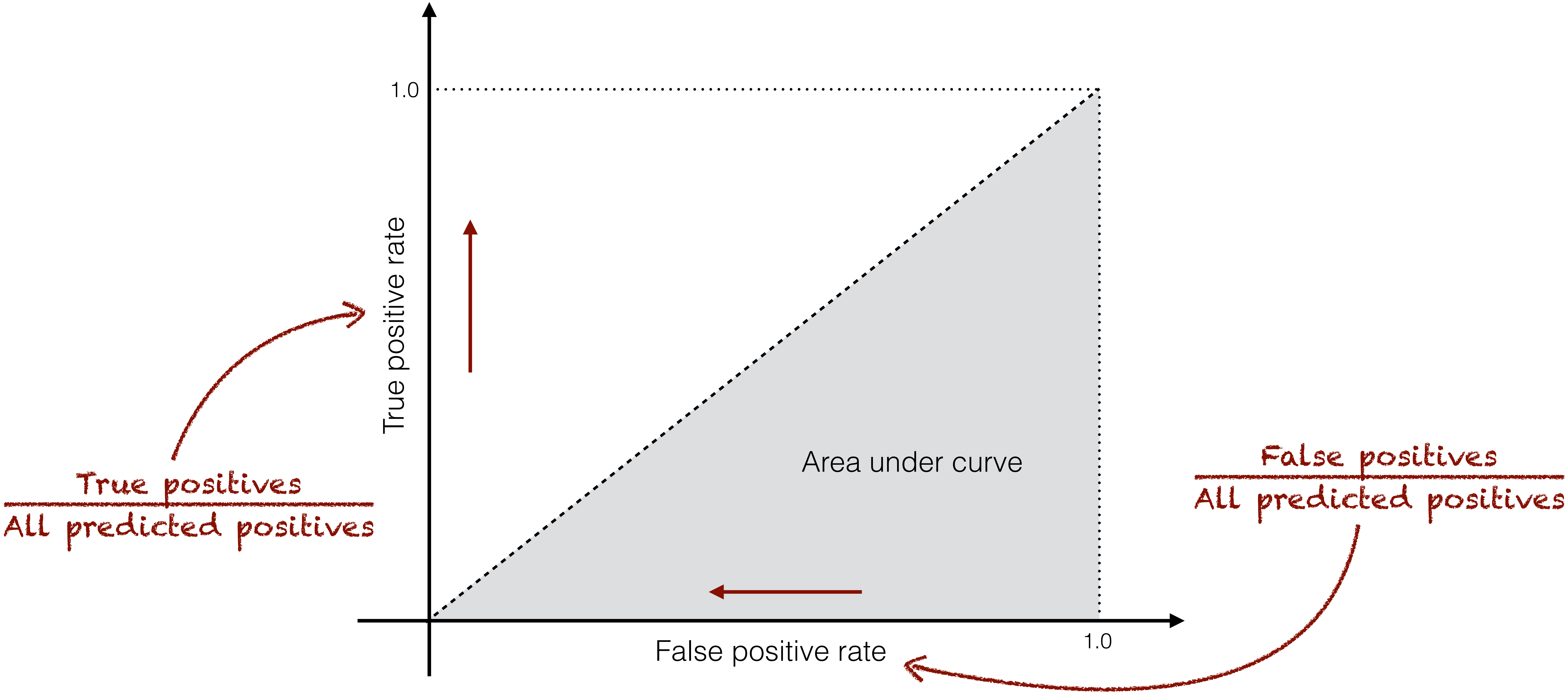
#	# / total	Accuracy	Precision	Sensitivity	NPV	Specificity
310 324	1.0000	0.93226	0.93951	0.94665	0.92152	0.91133

Dataset splitting train:test = 75:25

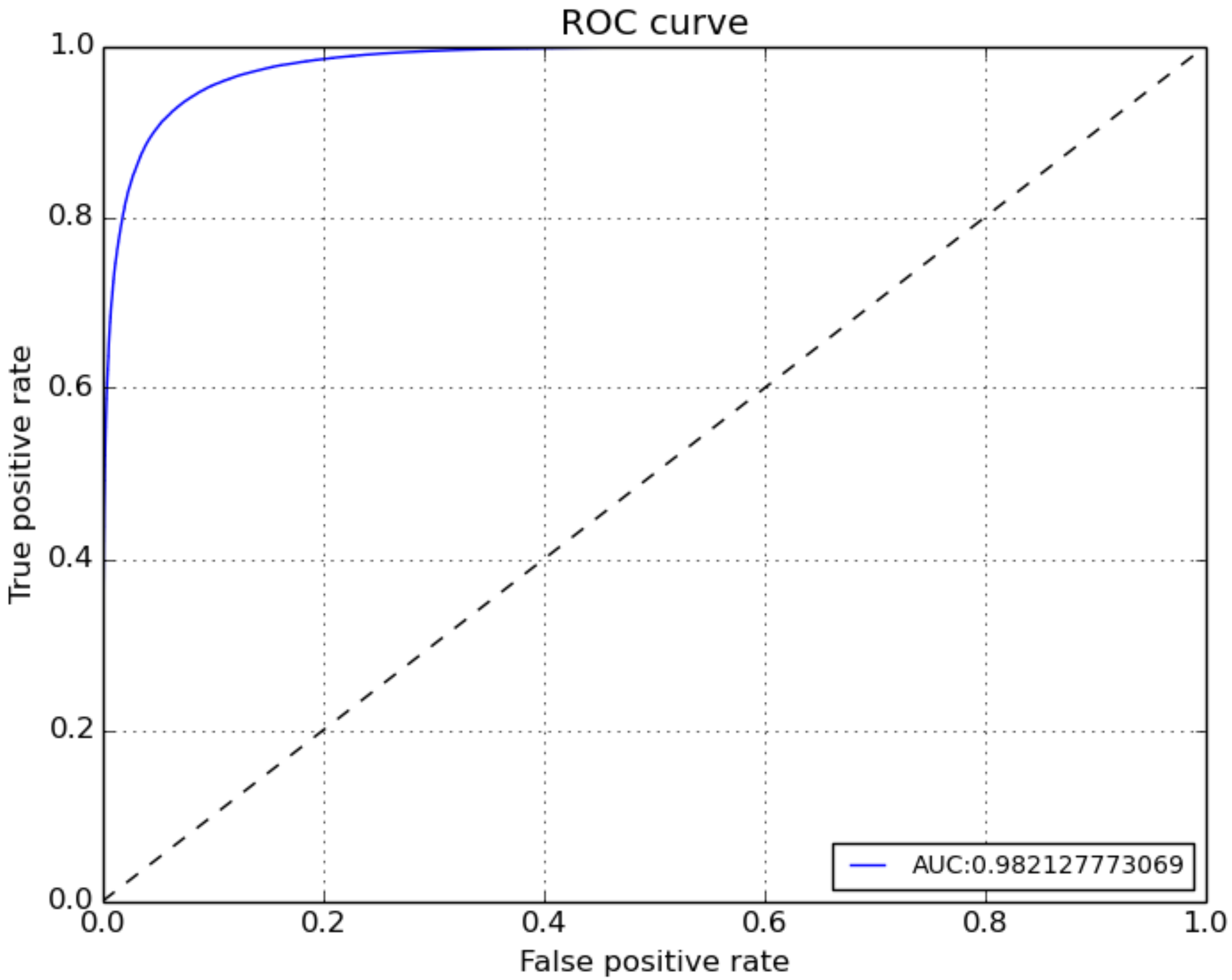
#	# / total	Accuracy	Precision	Sensitivity	NPV	Specificity
77 581	1.0000	0.92271	0.91653	0.93049	0.92912	0.91491



Introduction to ROC curves



ROC curve

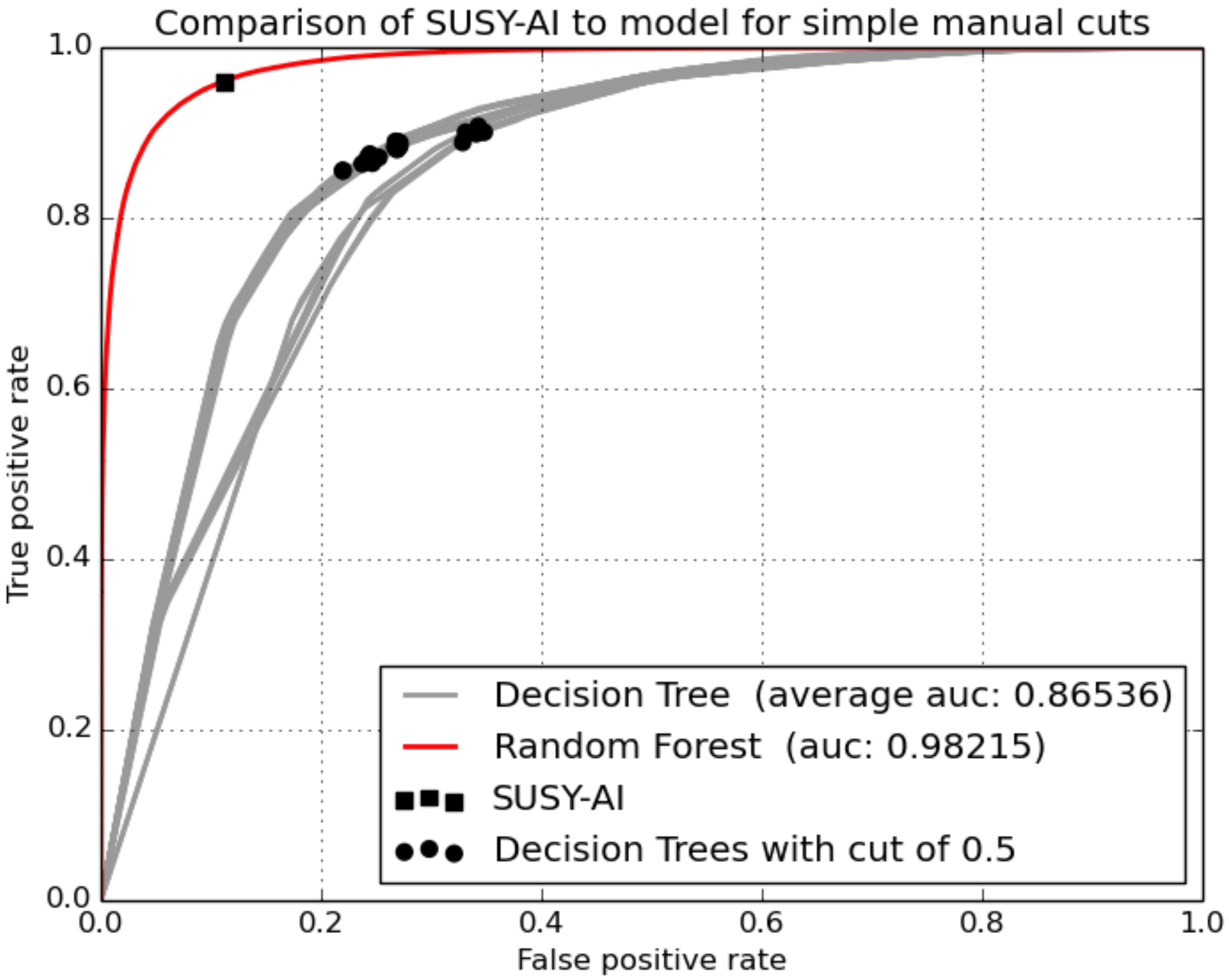


ROC curve
1.0
0.8
0.6
0.4
0.2
0.0
True positive rate
0.0
0.2
0.4
0.6
0.8
1.0
False positive rate
AUC:0.982127773069

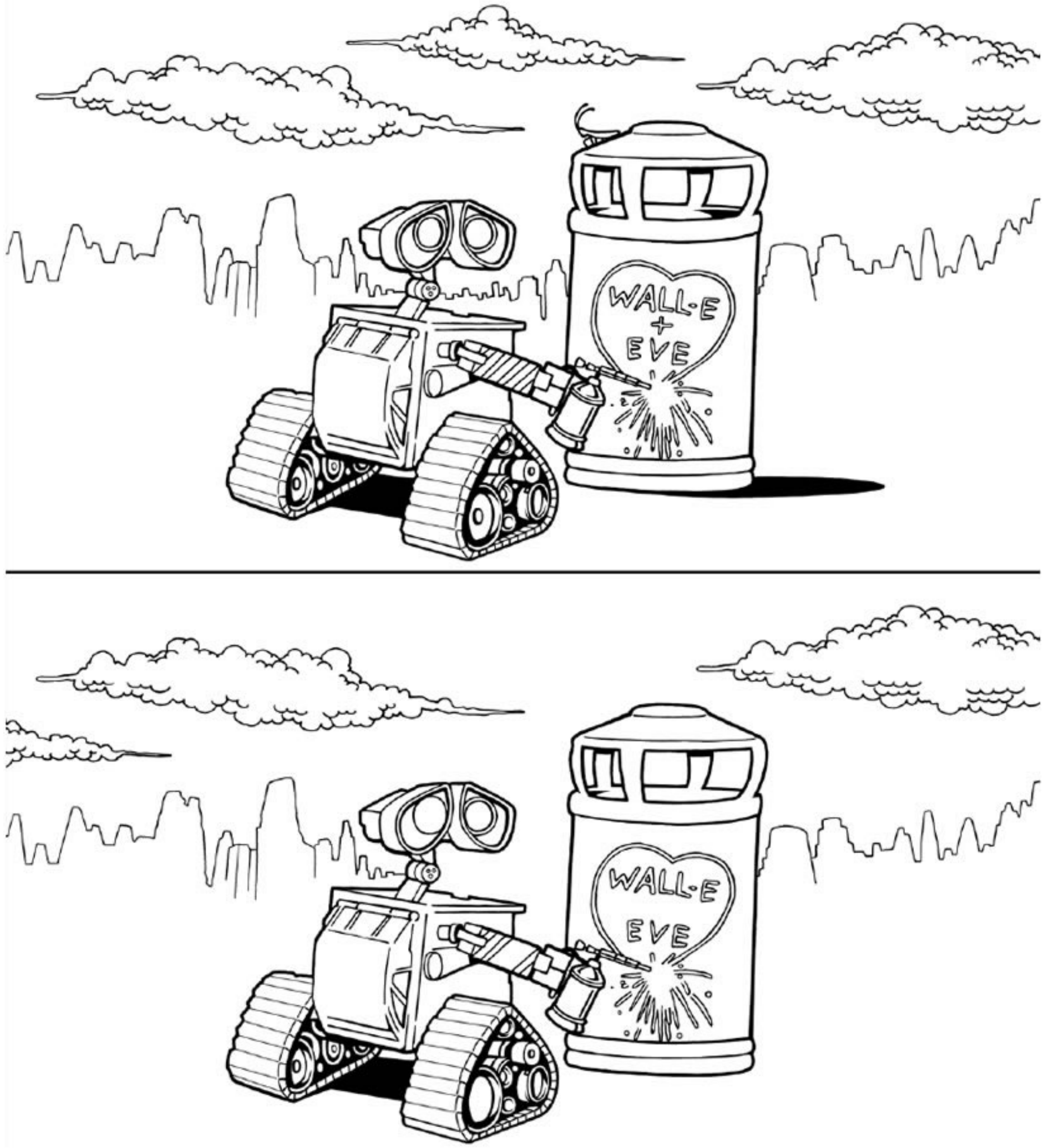
Comparison to model for human

- 20 individual decision trees with maximum depth of 5 (=21 cuts in parameter space)

- Markers are placed at value for cut with the highest accuracy

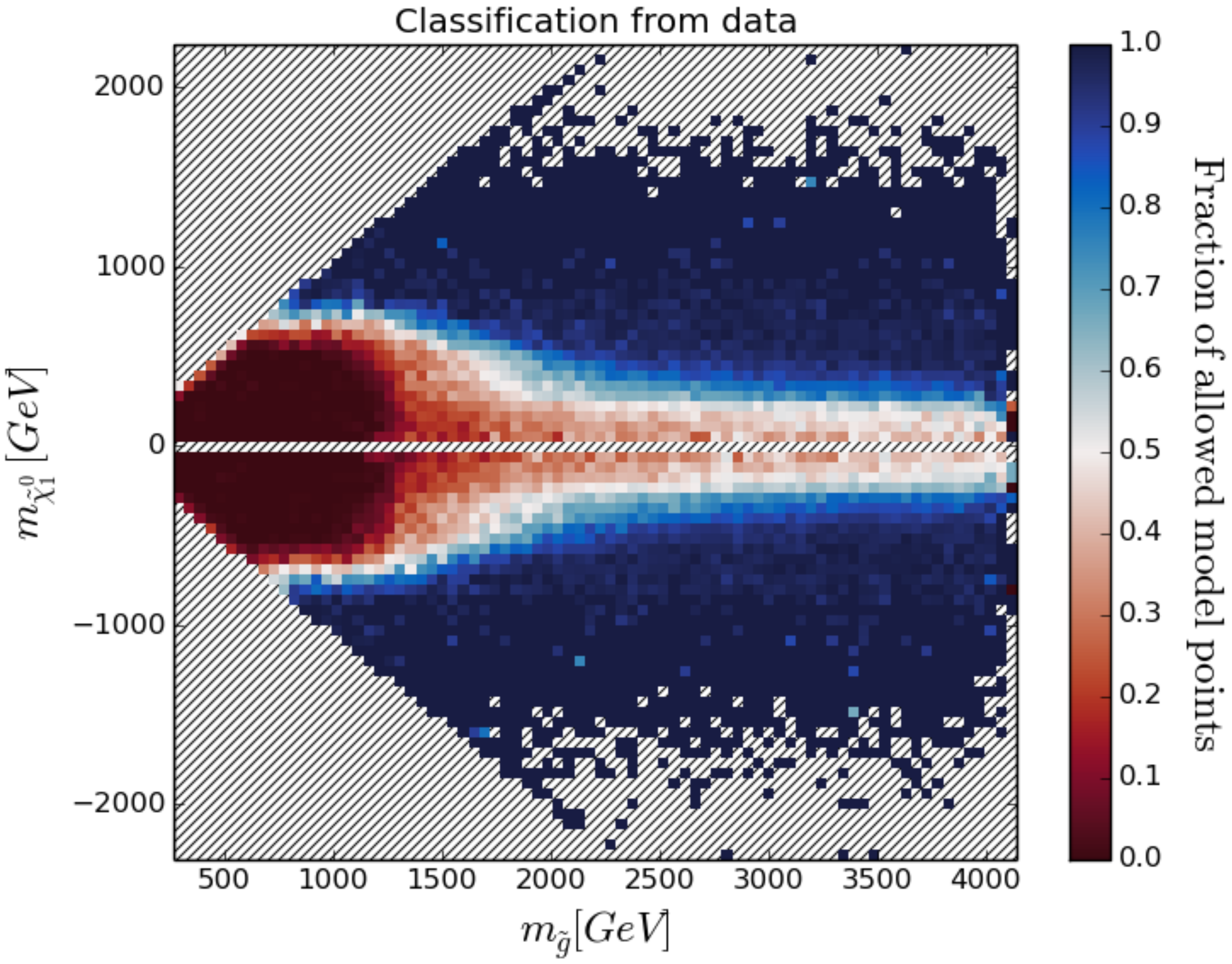


Spot the differences



Spot the differences

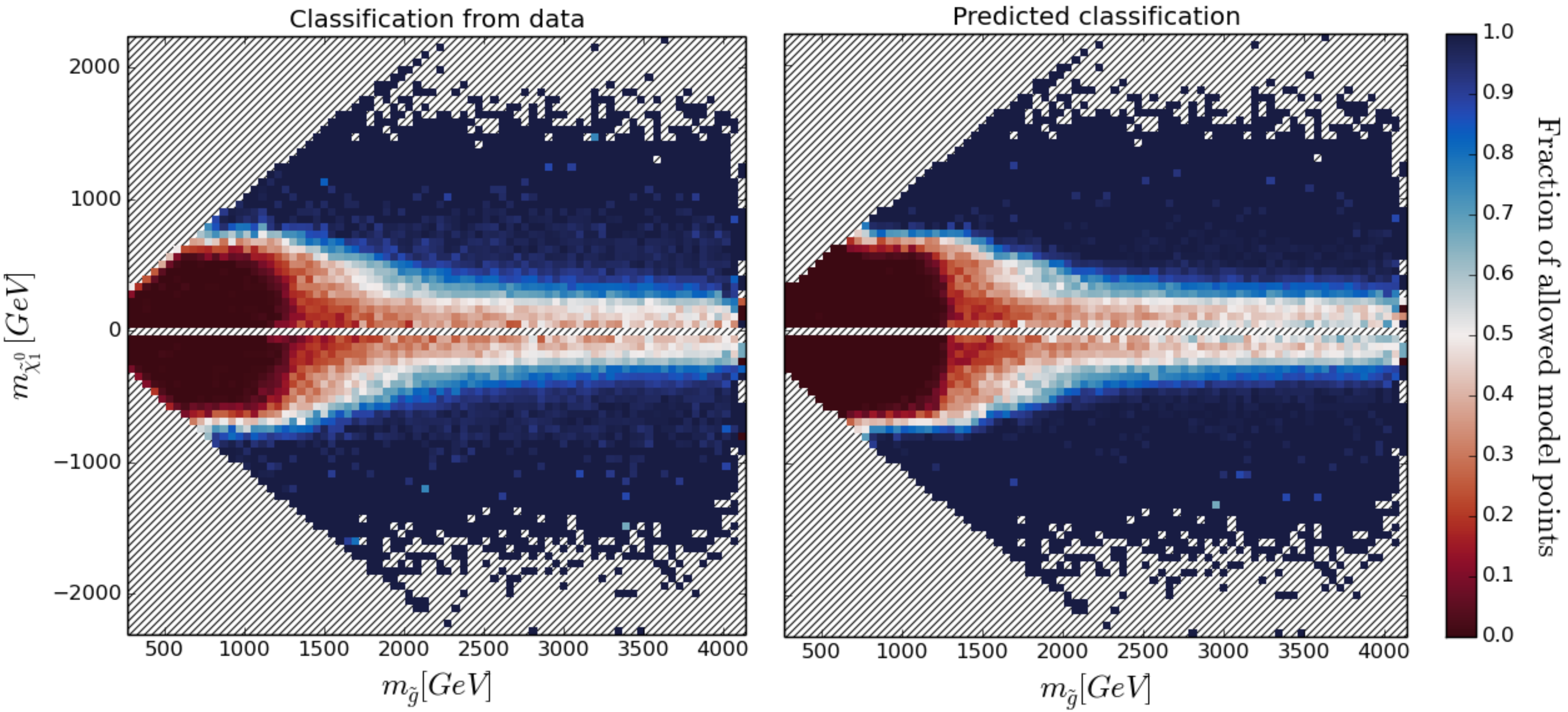
Performance gluino vs neutralino1



Performance gluino vs neutralino1

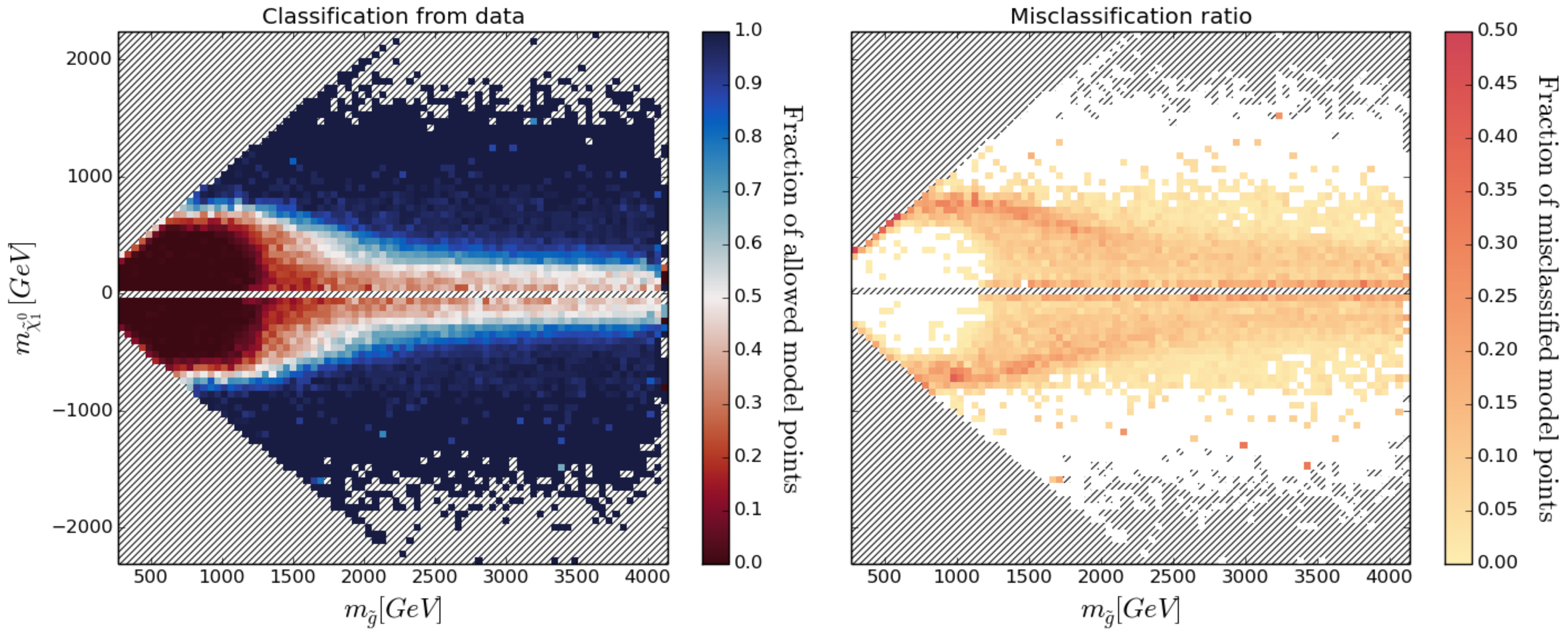
93.2% accuracy @ 8TeV

92.7% accuracy @ 13 TeV



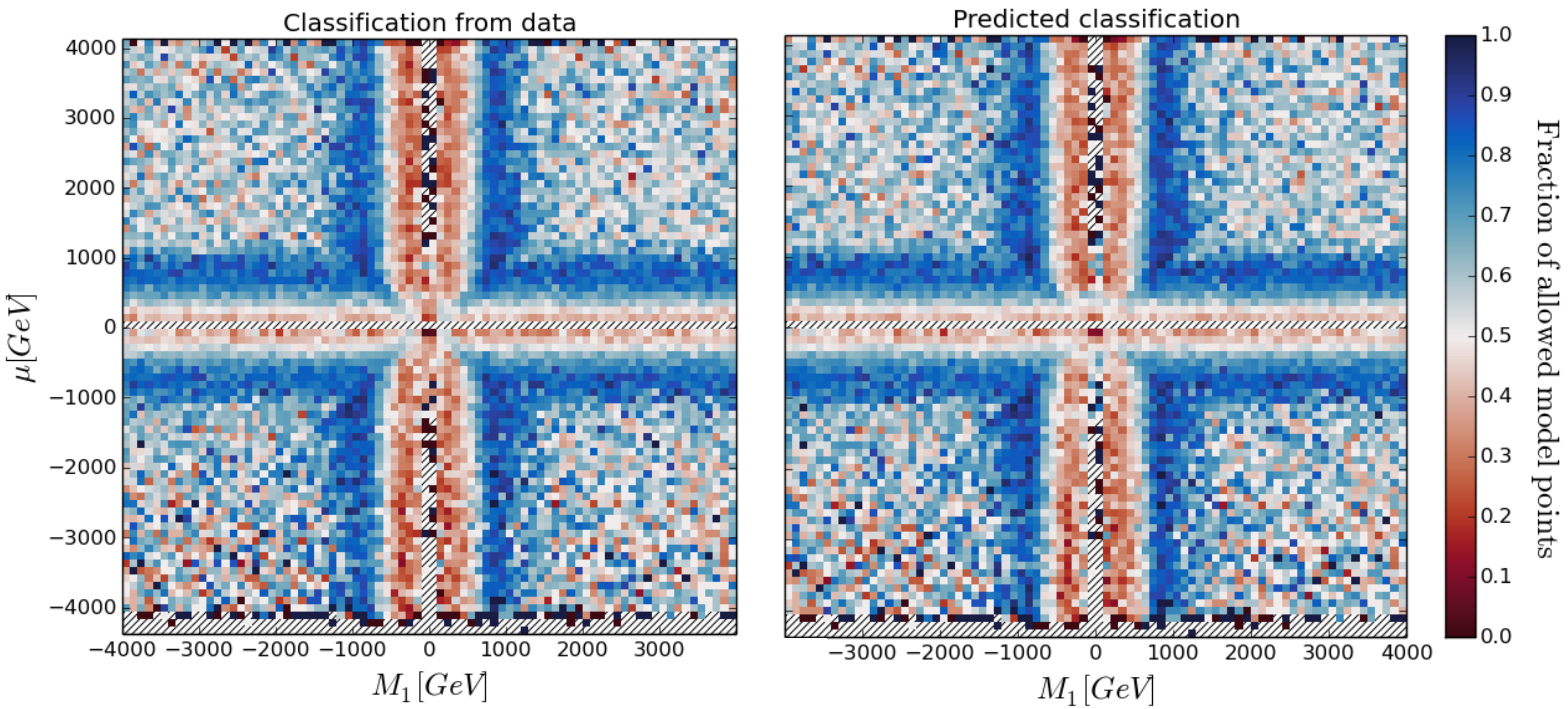
Performance gluino vs neutralino1

93.2% accuracy @ 8TeV 92.7% accuracy @ 13 TeV



Performance M1 vs mu

93.2% accuracy @ 8TeV   92.7% accuracy @ 13 TeV

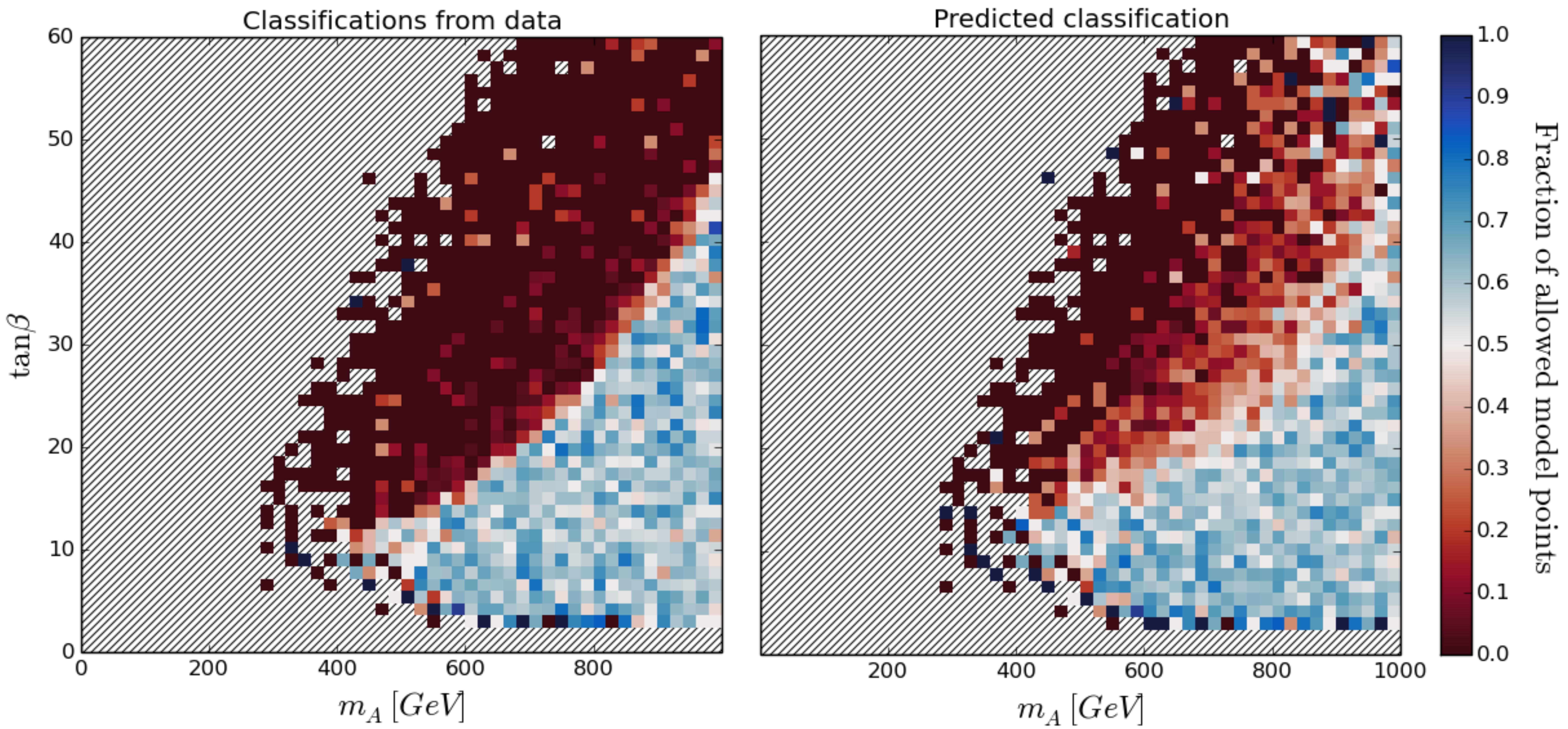


Performance mA vs tan(beta)

Zoomed in

93.2% accuracy @ 8TeV

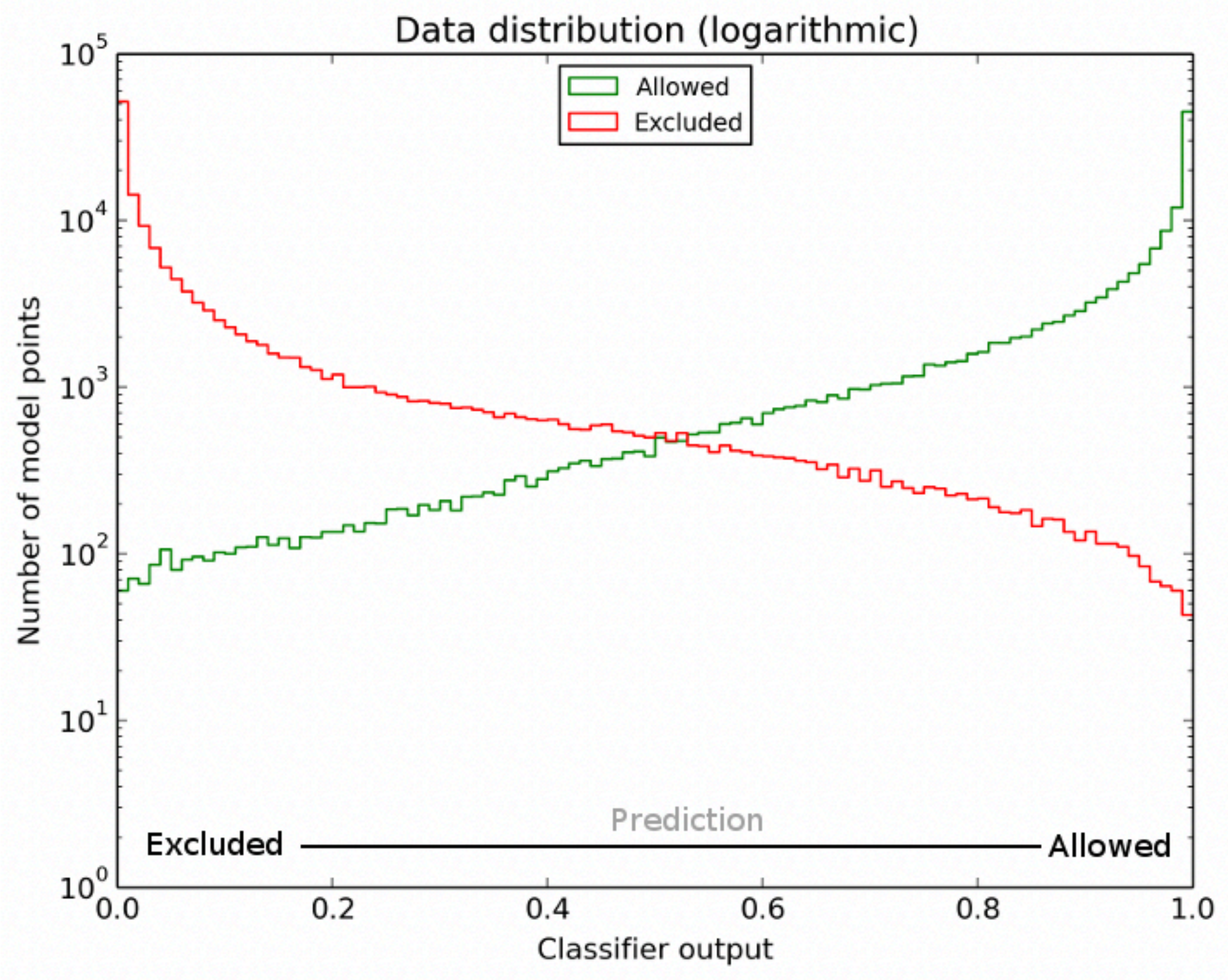
92.7% accuracy @ 13 TeV

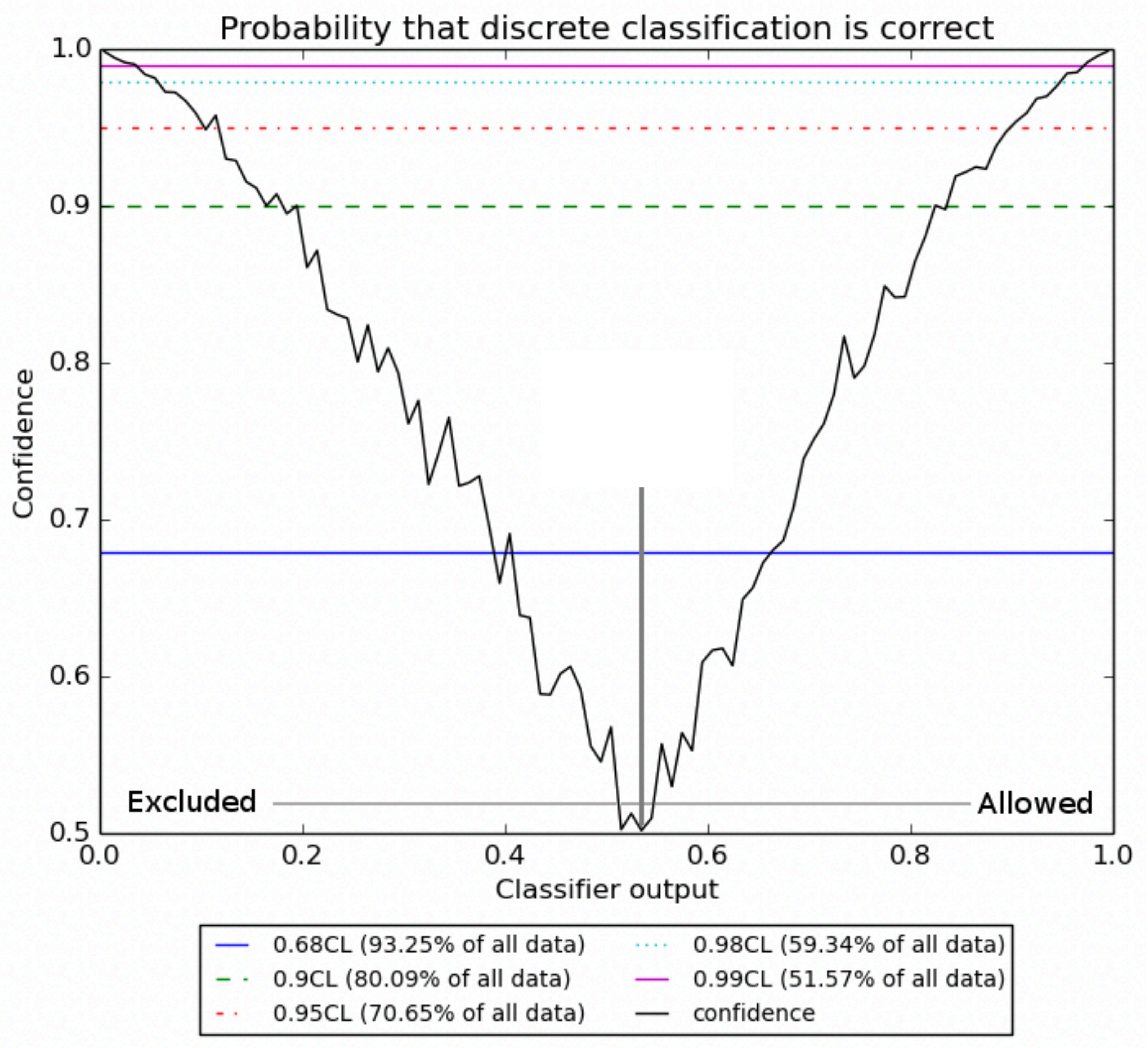


Confidence

Improving predictions

Confidence



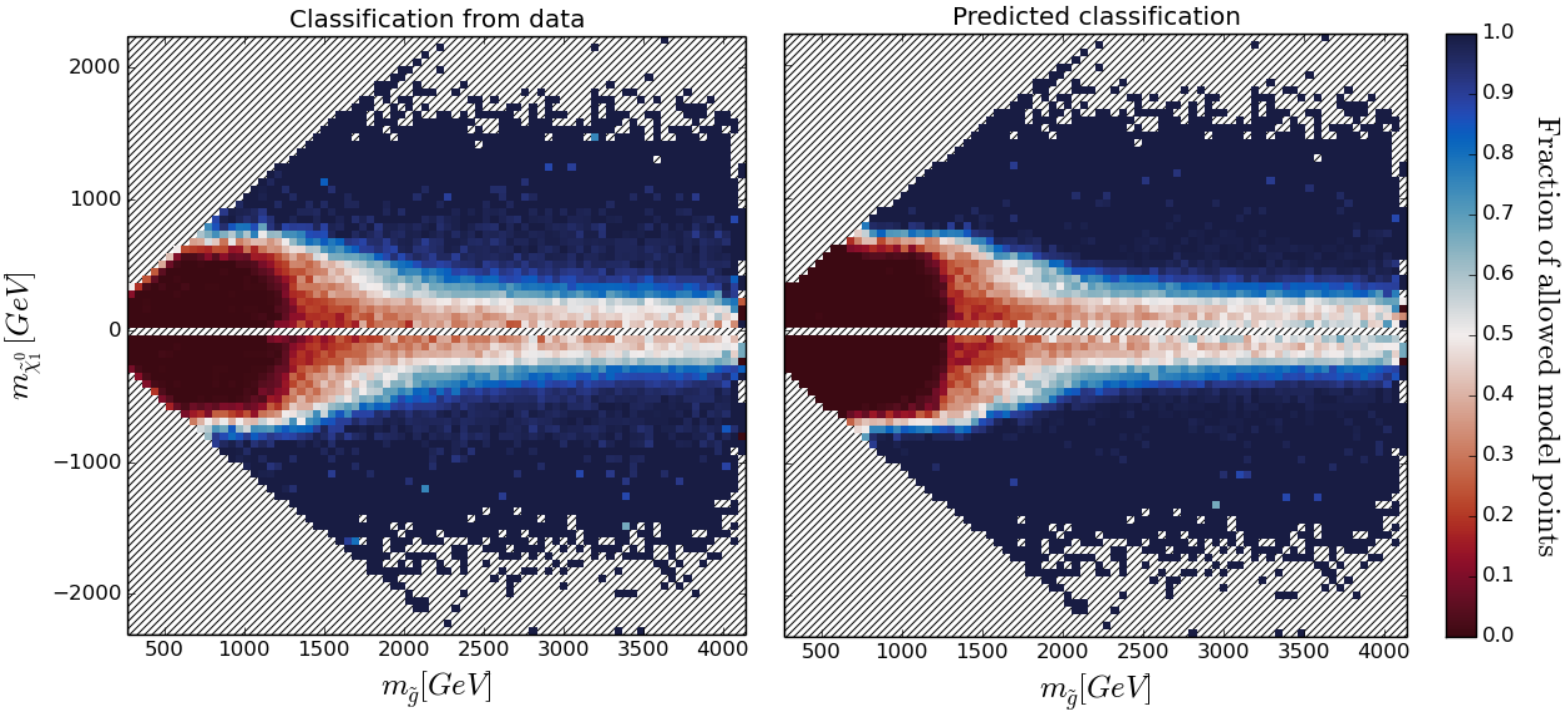


- Allows for requiring minimum degree of confidence

Performance gluino vs neutralino1

93.2% accuracy @ 8TeV

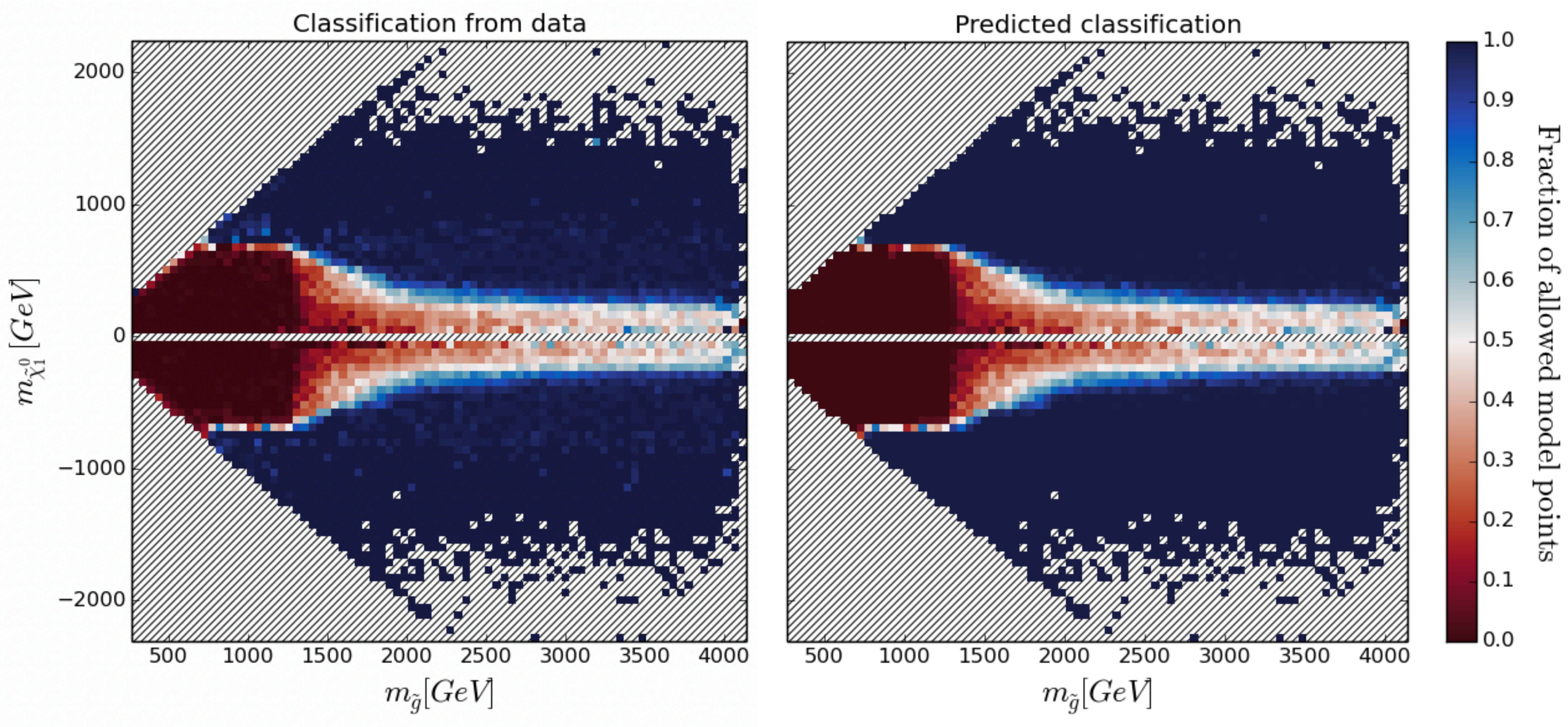
92.7% accuracy @ 13 TeV



Confidence (>95%) gluino vs neutralino1

99.1% accuracy on 70.6% of total data @ 8TeV

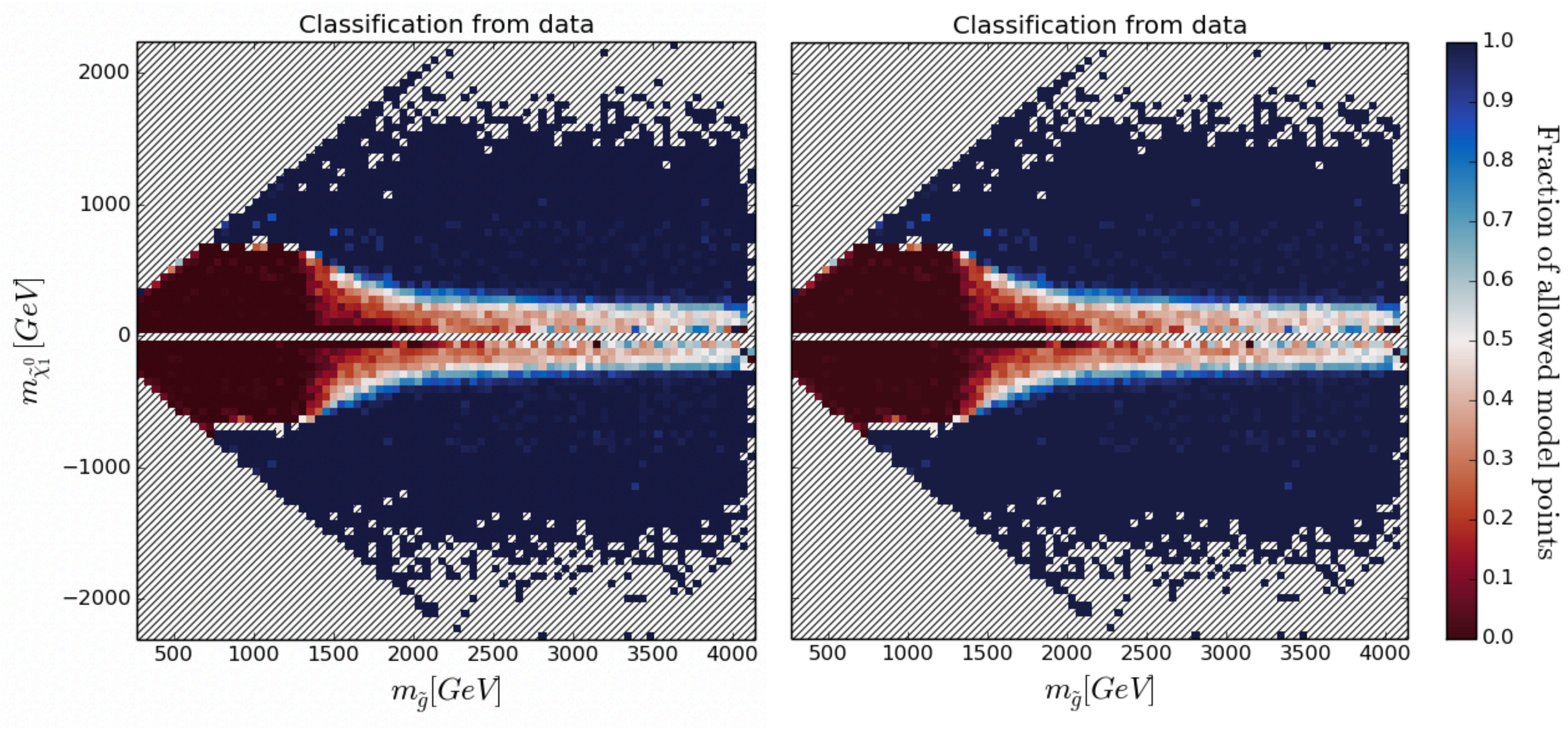
99.0% accuracy on 68.0% of total data @ 13 TeV



Confidence (>99%) gluino vs neutralino1

99.7% accuracy on 51.6% of total data @ 8TeV

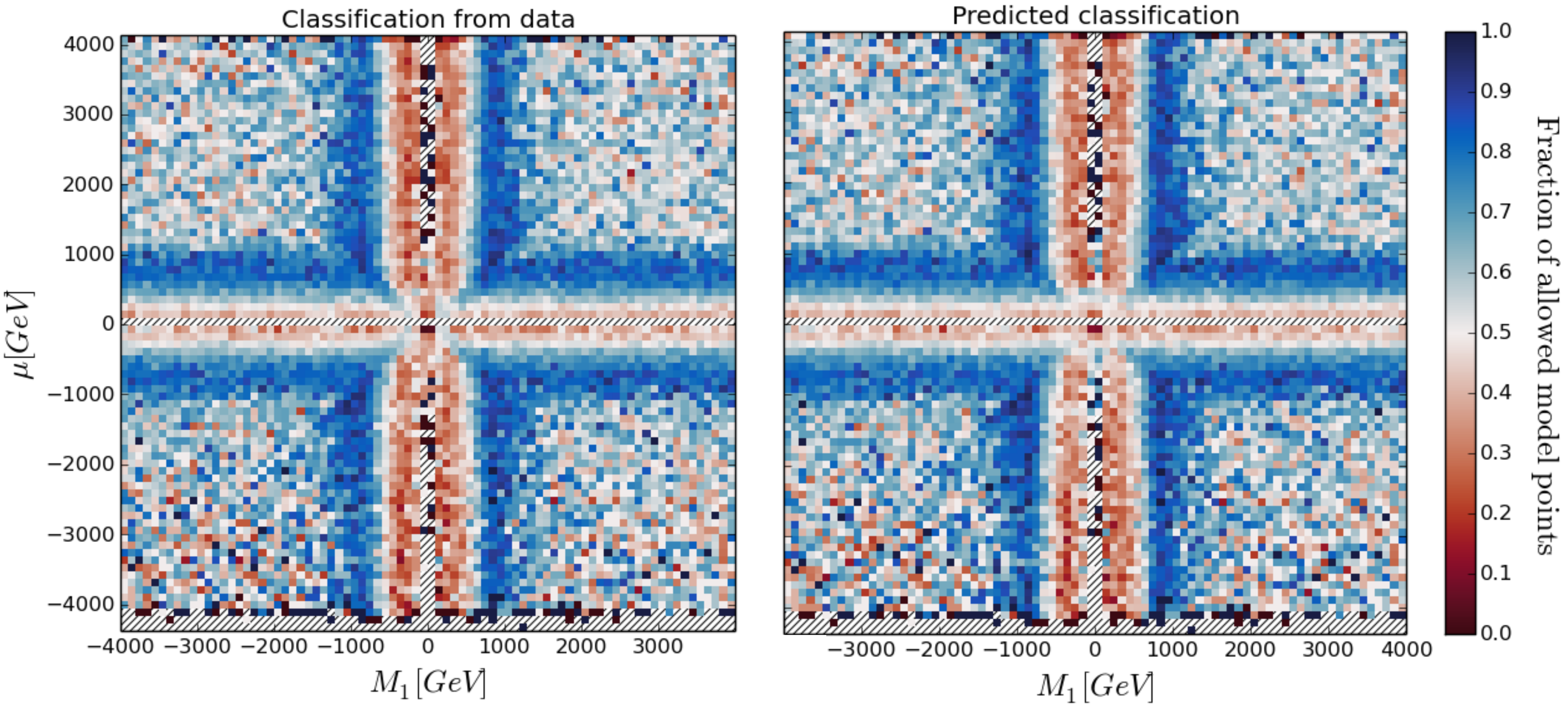
99.7% accuracy on 47.6% of total data @ 13 TeV



Performance M1 vs mu

93.2% accuracy @ 8TeV

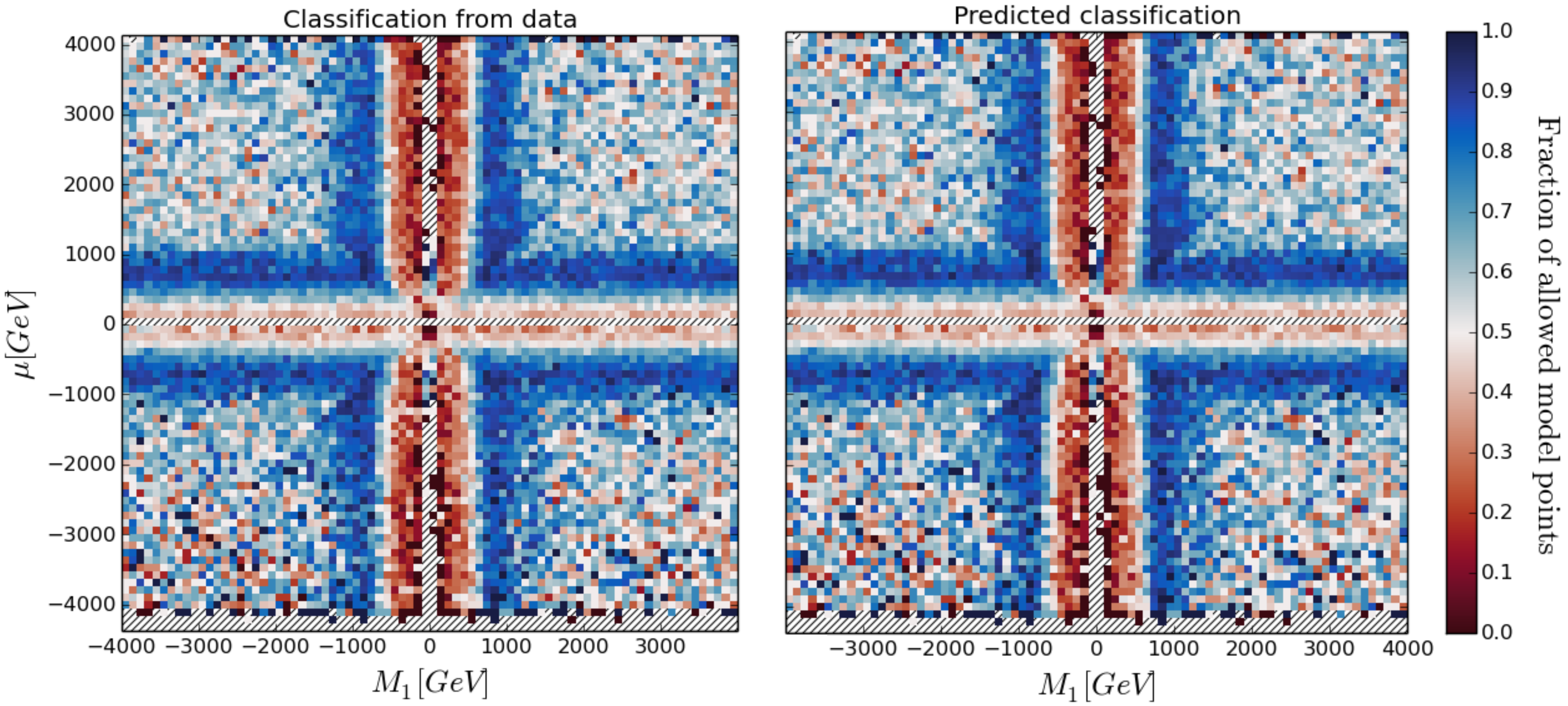
92.7% accuracy @ 13 TeV



Confidence (>95%) M1 vs mu

99.1% accuracy on 70.6% of total data @ 8TeV

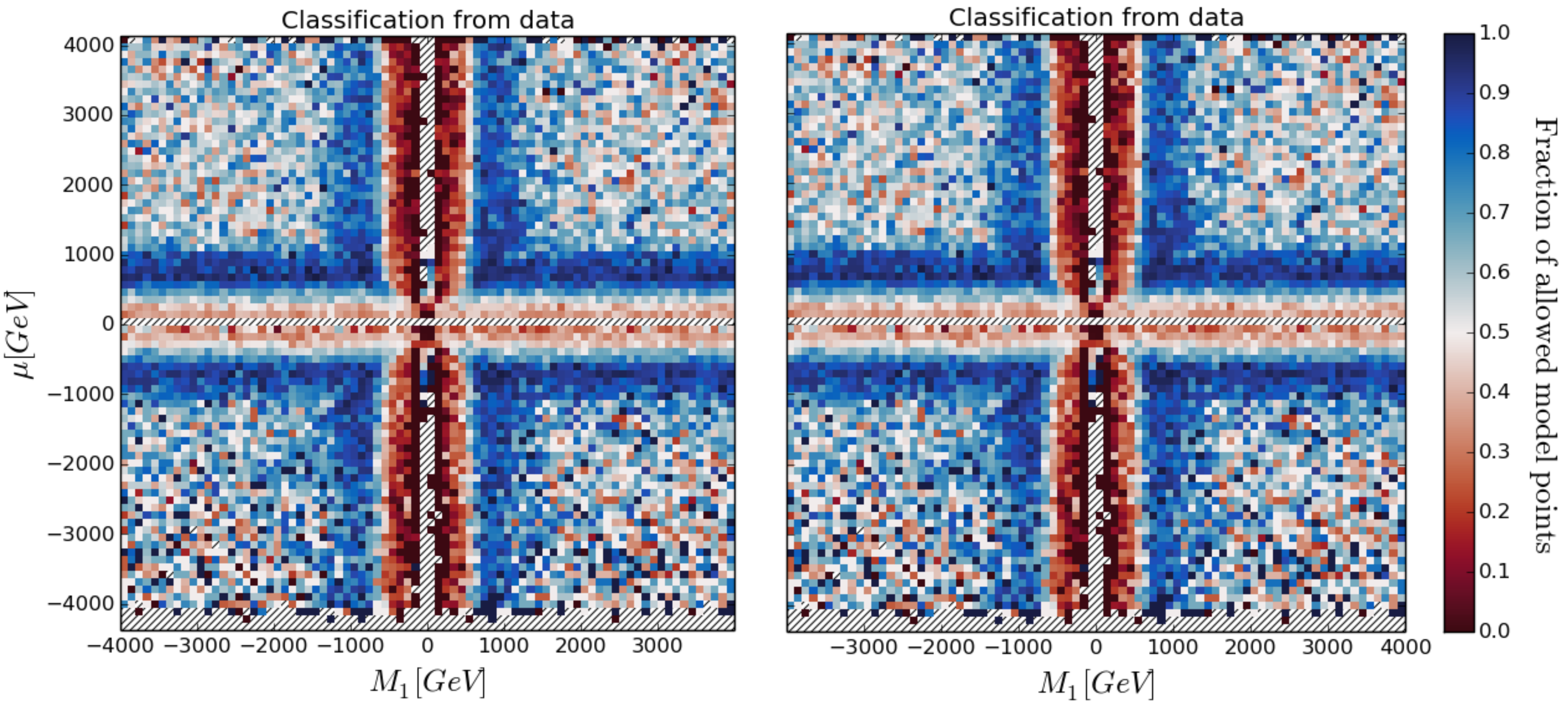
99.0% accuracy on 68.0% of total data @ 13 TeV



Confidence (>99%) M1 vs mu

99.7% accuracy on 51.6% of total data @ 8TeV

99.7% accuracy on 47.6% of total data @ 13 TeV

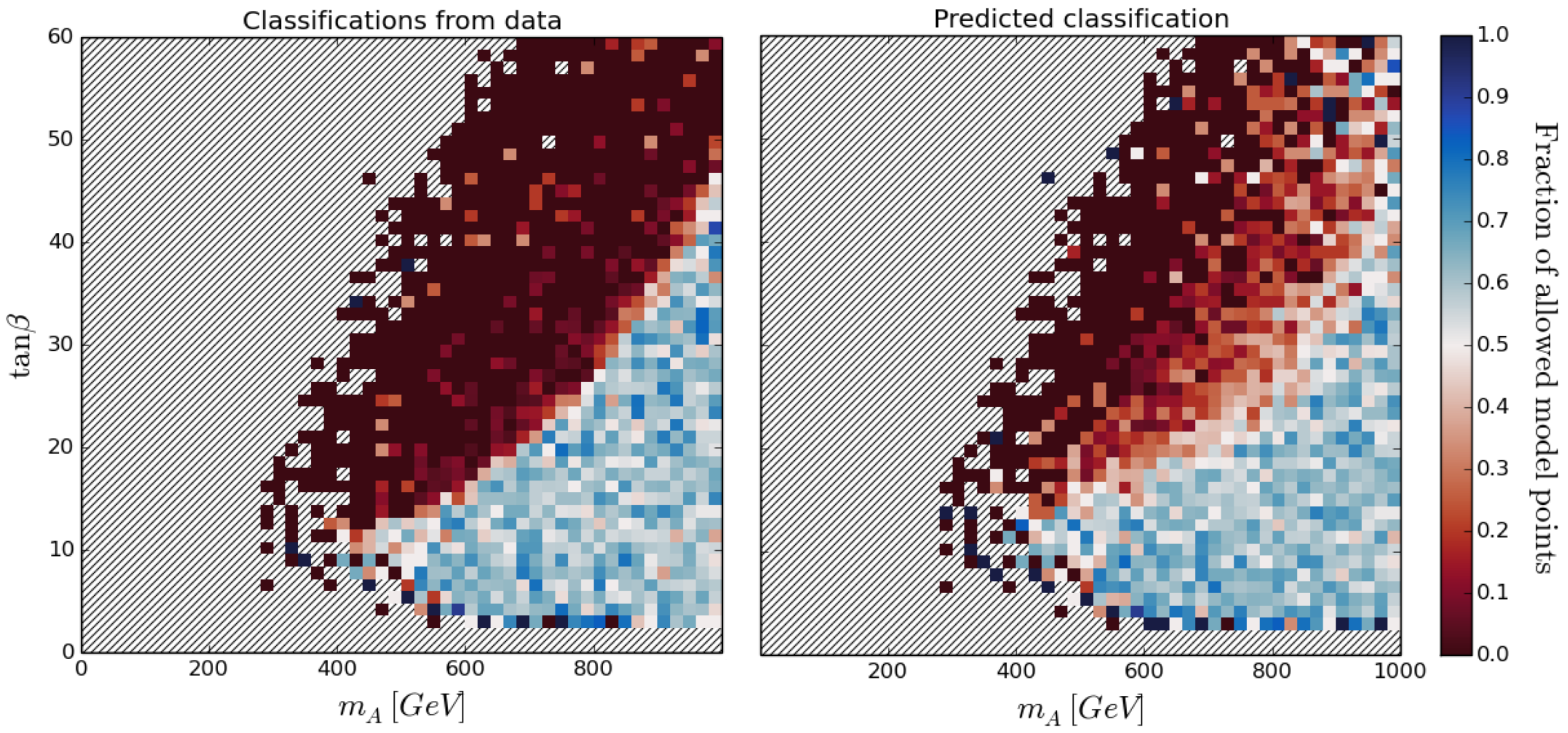


Performance mA vs tan(beta)

Zoomed in

93.2% accuracy @ 8TeV

92.7% accuracy @ 13 TeV

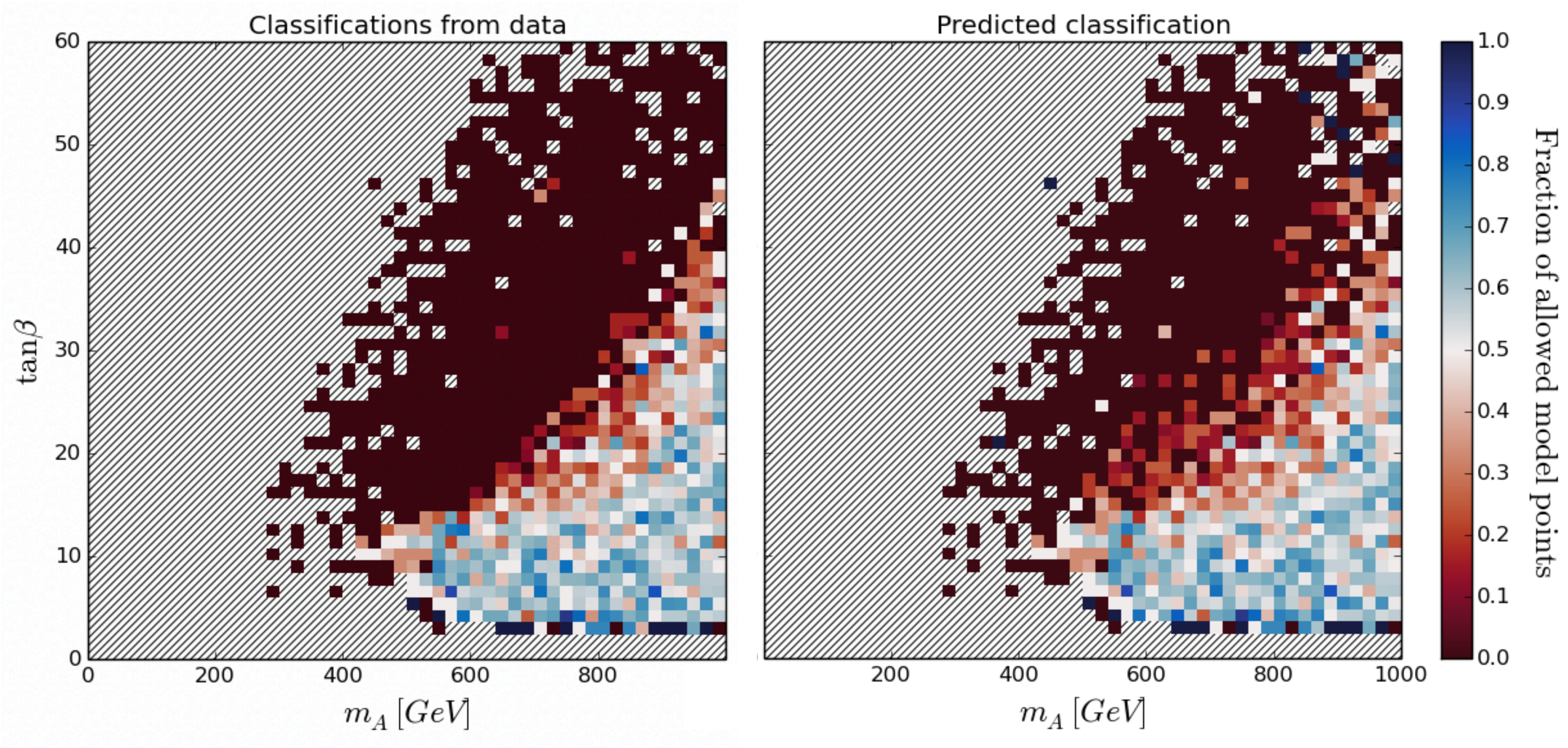


Confidence (>95%) mA vs tan(beta)

Zoomed in

99.1% accuracy on 70.6% of total data @ 8TeV

99.0% accuracy on 68.0% of total data @ 13 TeV

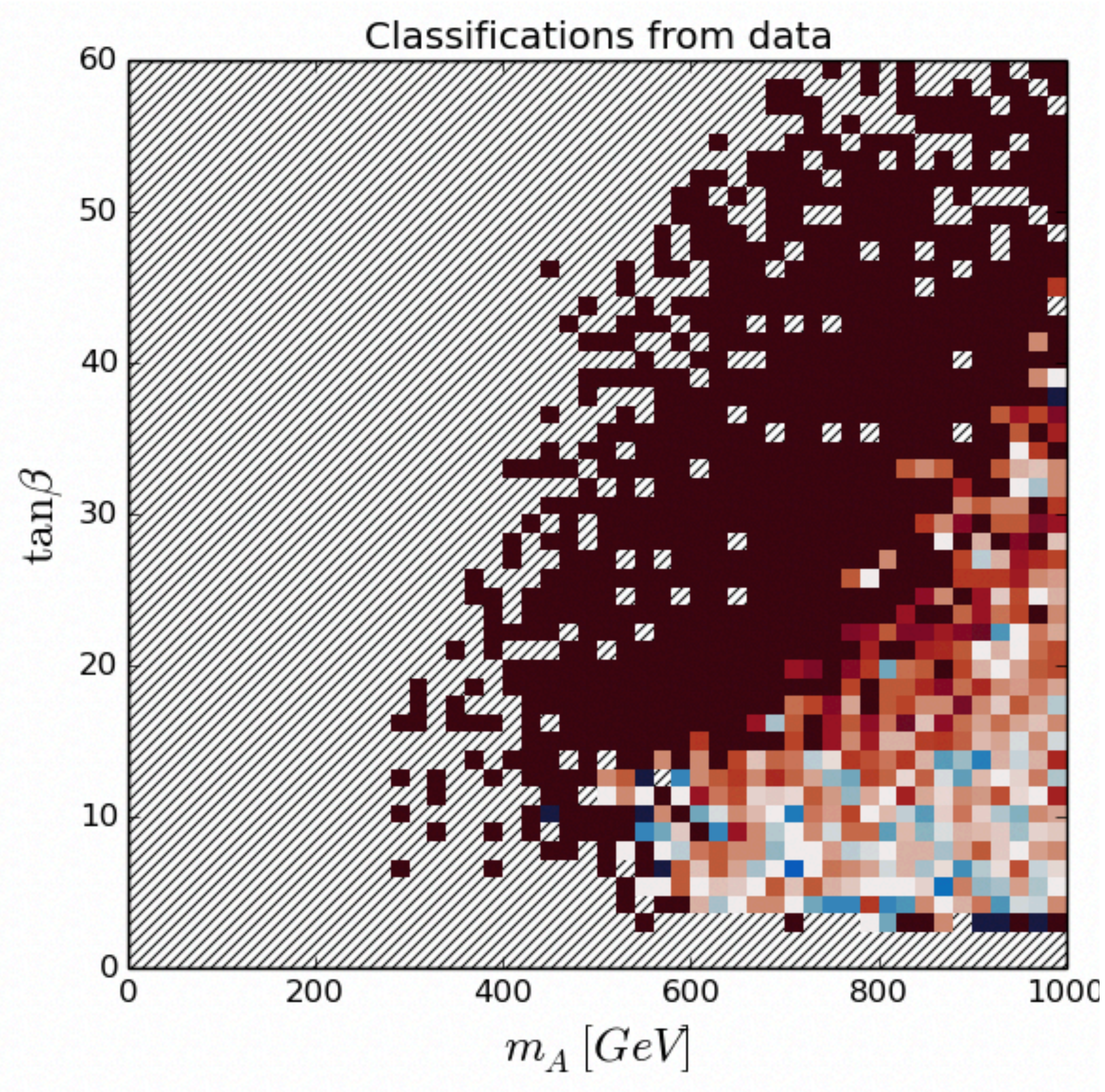


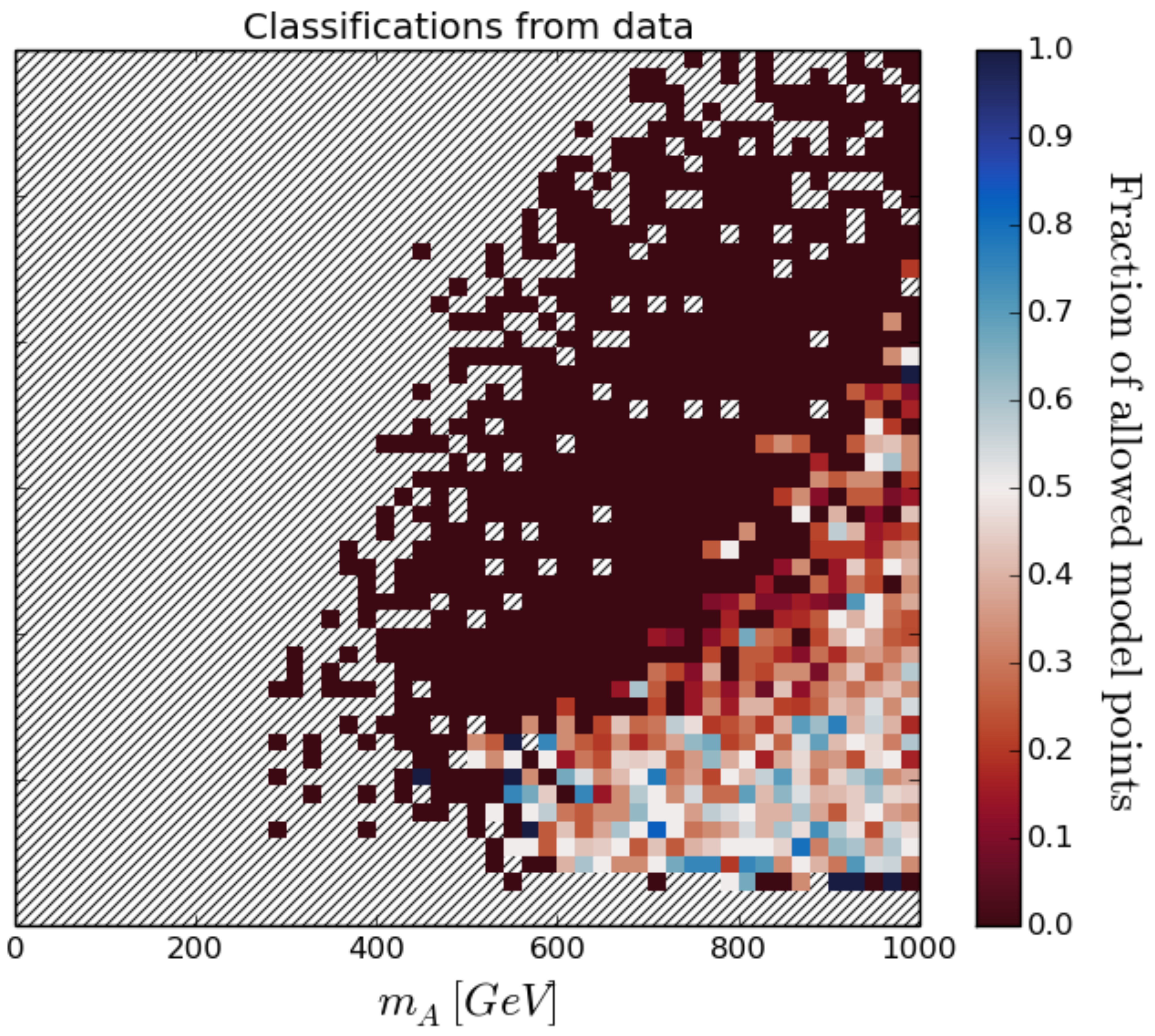
Confidence (>99%) mA vs tan(beta)

Zoomed in

99.7% accuracy on 51.6% of total data @ 8TeV

99.7% accuracy on 47.6% of total data @ 13 TeV





Out-of-bag vs train:test split

Accuracy:
(TP+TN) / all

Precision:
TP / (TP+FP)

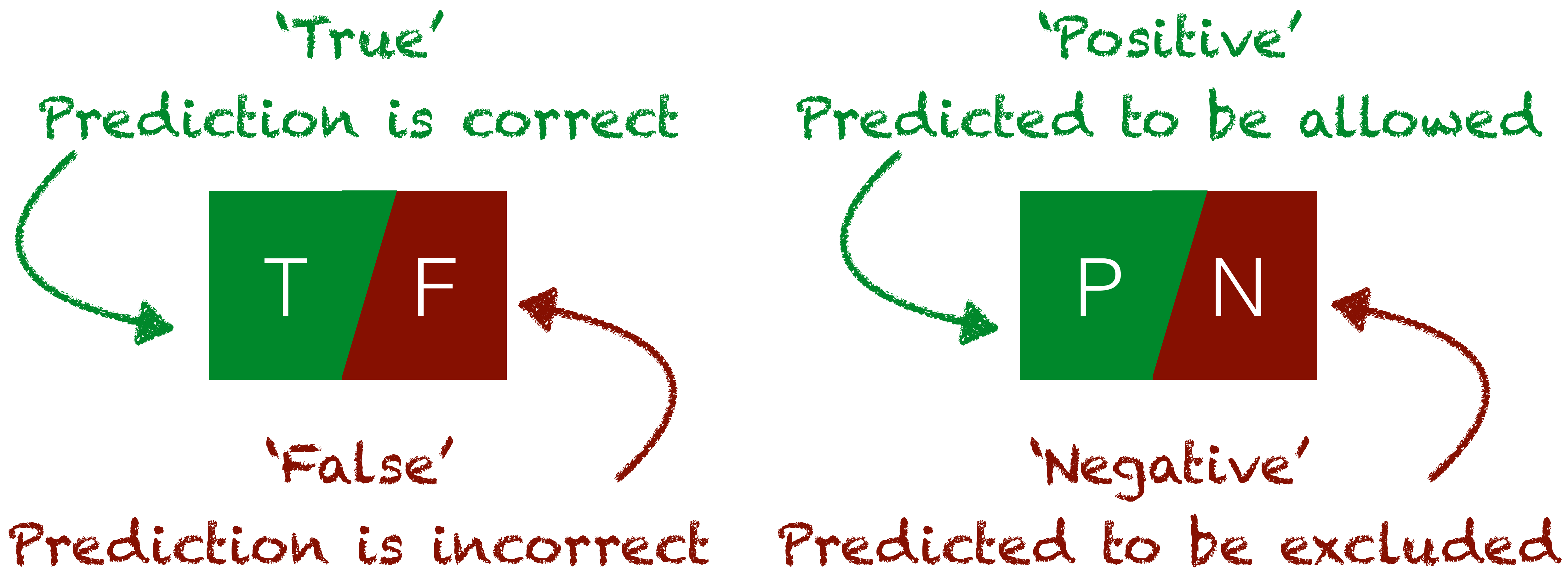
Sensitivity
TP / (TP+FN)

Negative prediction value
TN / (TN+FN)

Specificity
TN / (TN+FP)

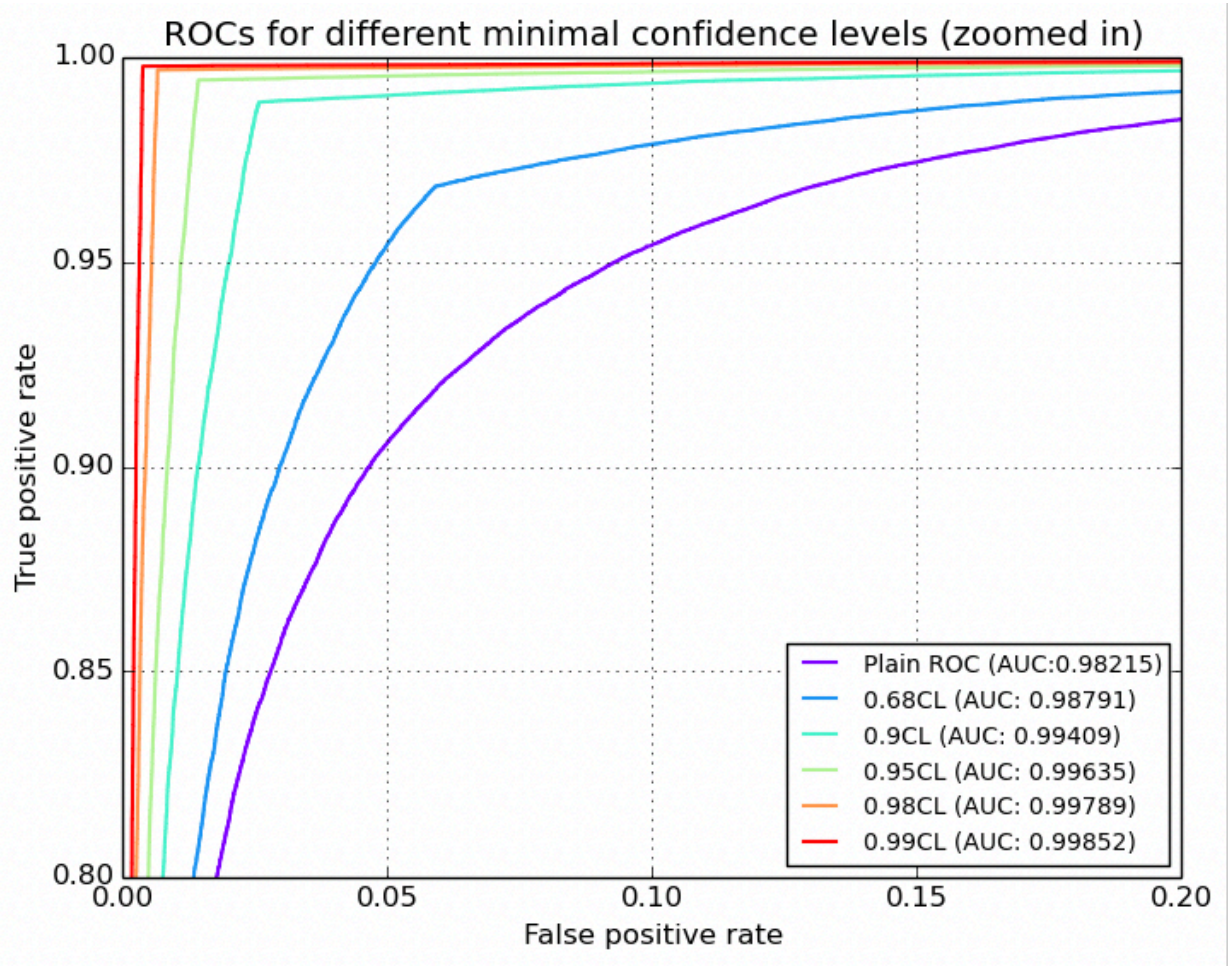
Out-of-bag

CL	#	# / total	Accuracy	Precision	Sensitivity	NPV	Specificity
0.0	310 324	1.0000	0.93226	0.93951	0.94665	0.92152	0.91133
0.68	289 371	0.93248	0.95735	0.96072	0.96835	0.95222	0.94094
0.95	219 233	0.70646	0.99094	0.99092	0.99426	0.99096	0.98573
0.98	184 230	0.59367	0.99543	0.99573	0.99672	0.99496	0.99346
0.99	160 034	0.51570	0.99708	0.99747	0.99764	0.99649	0.99624



ROC curve

Zoomed in

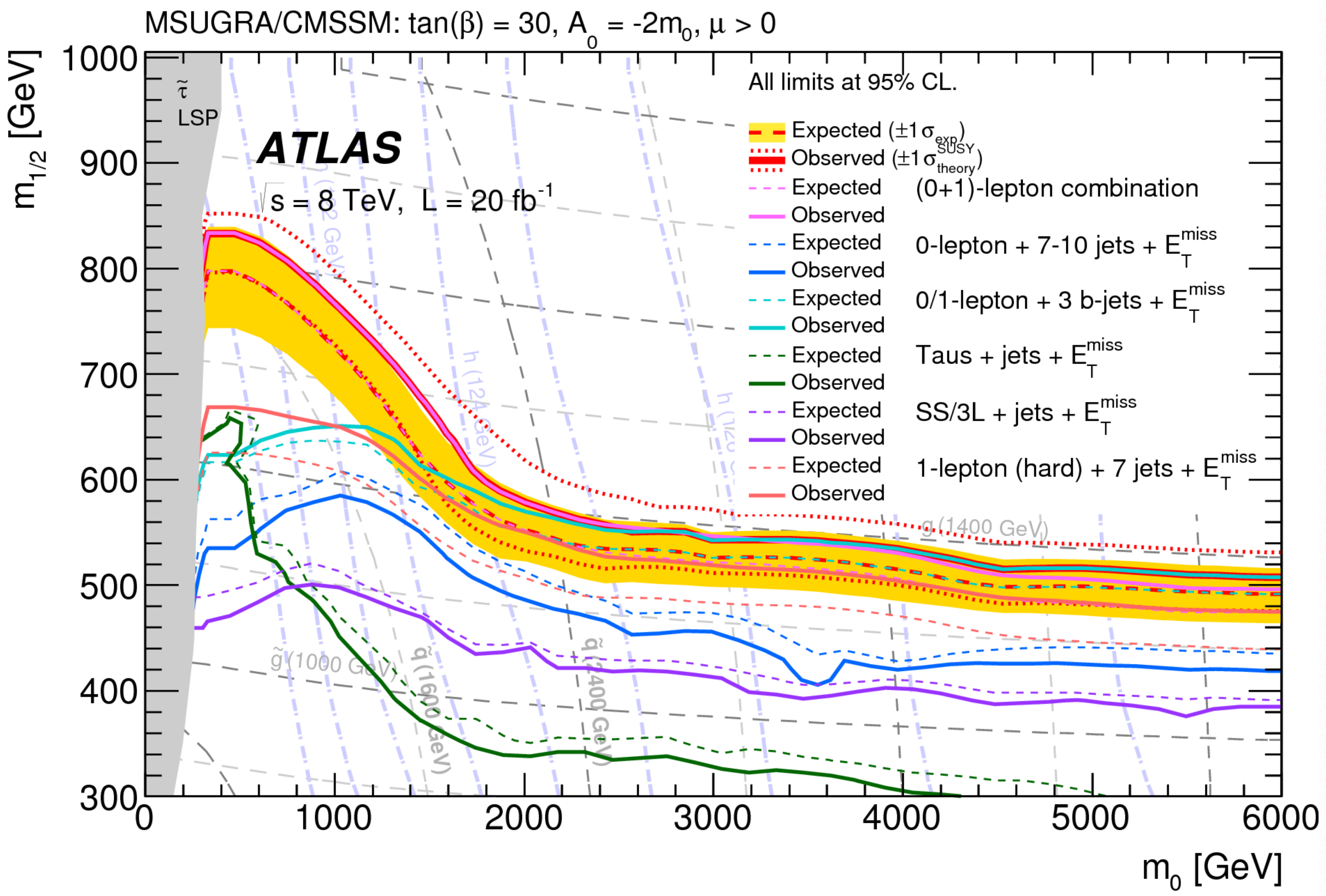


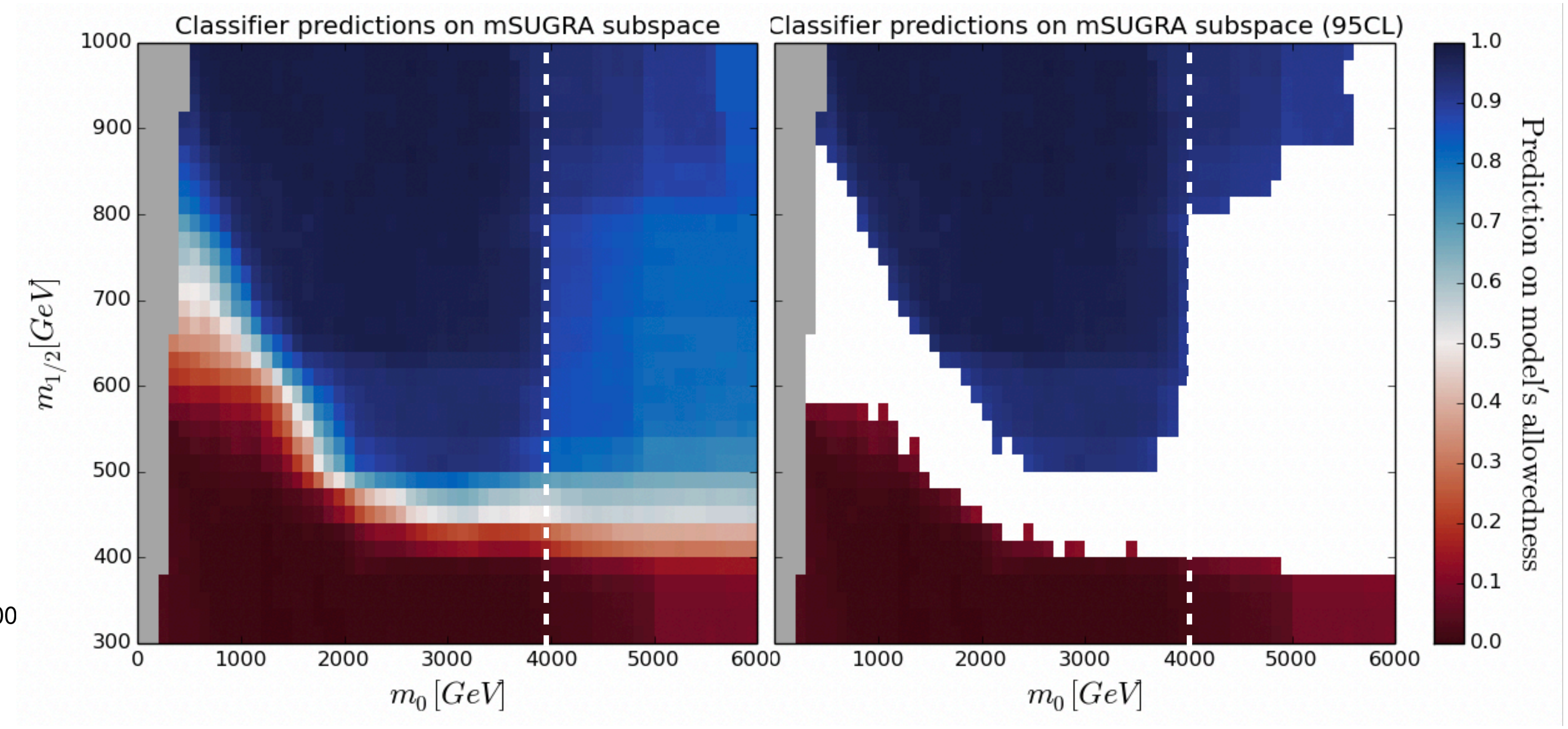
Applicability

Can it always be used?

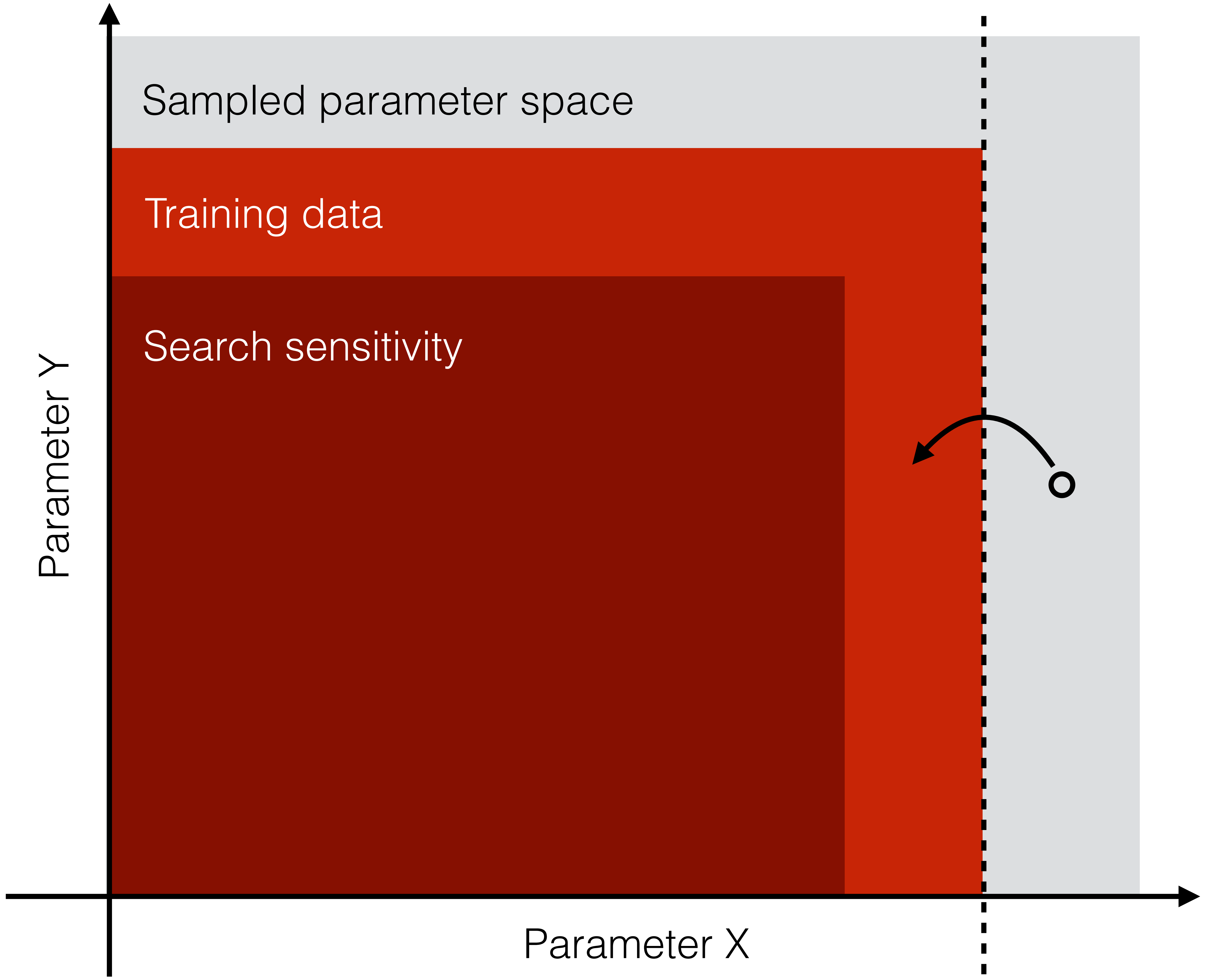
mSUGRA

Parameter	Description	Scanned range [GeV]
m_0	Sbosonic particle masses	[0, 6000]
$m_{1/2}$	Sfermionic particle masses	[300, 1000]
A_0	Coupling proportionality constant	$2 \cdot m_0$
$\tan\beta$	Ratio of vacuum expectation values of H_u^0 and H_d^0	30
$\mathrm{sign}(\mu)$	Sign of the higgsino mass parameter	+1

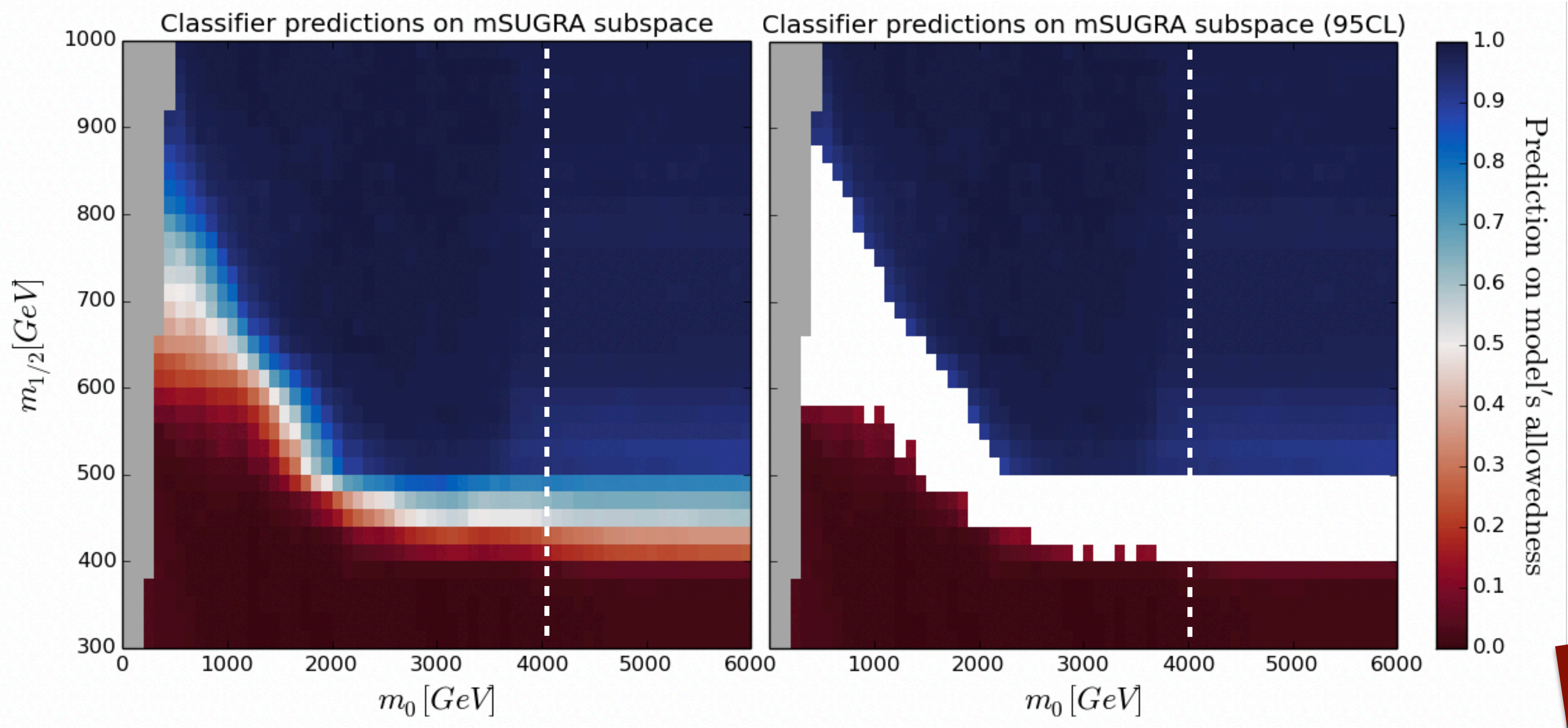


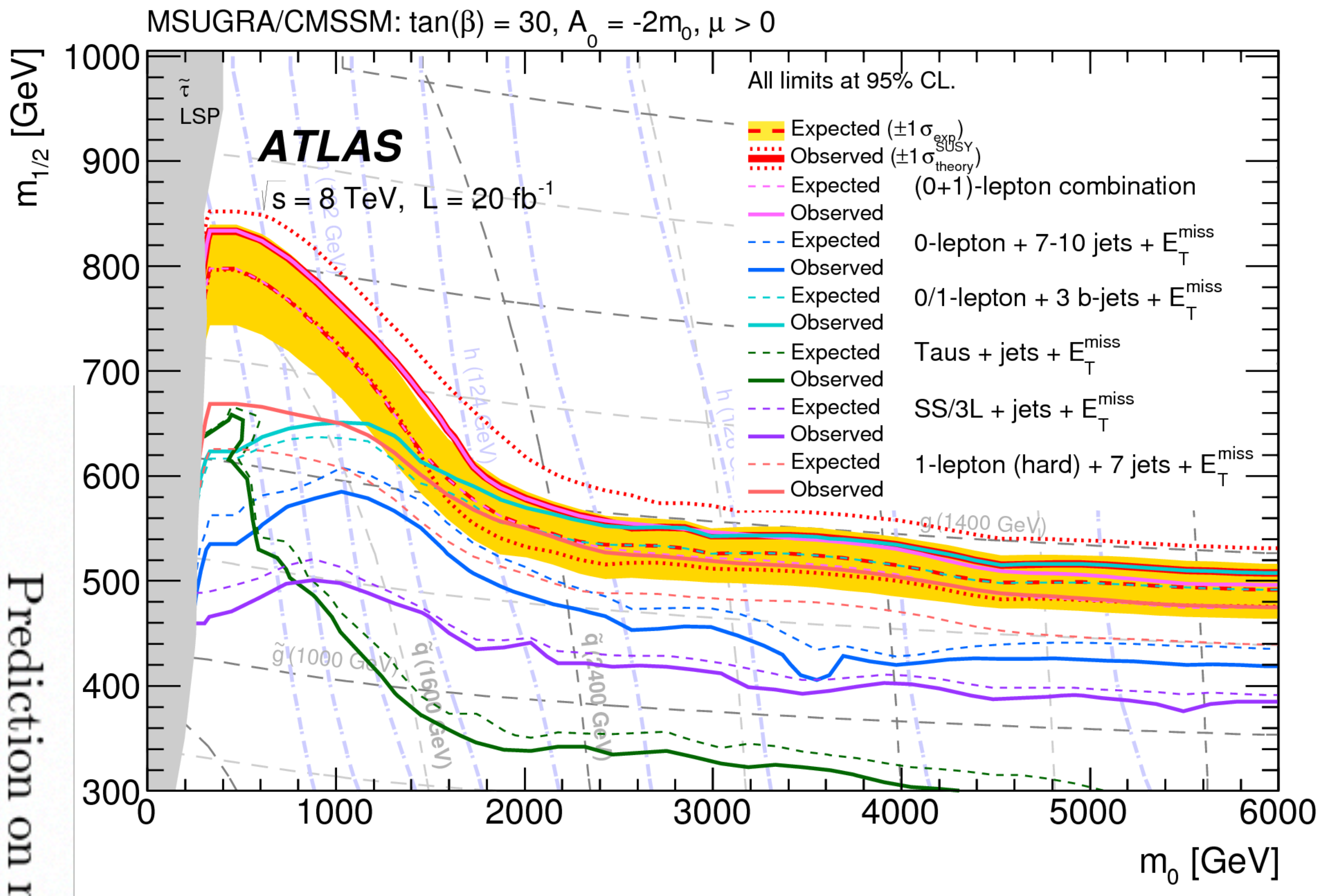


Outlier mapping



mSUGRA with mapping





So SUSY-AI is also applicable outside the range of the training data!

Other contexts

- Zoomed in parts of pMSSM
- CMS Analyses
- Exclusion based on other experiments (Xenon100, IceCube etc.)
- Higgs likelihood based on kappa values
- Dark Matter models
- …

This ML application can be applied to any model space!

Also yours!

Conclusions

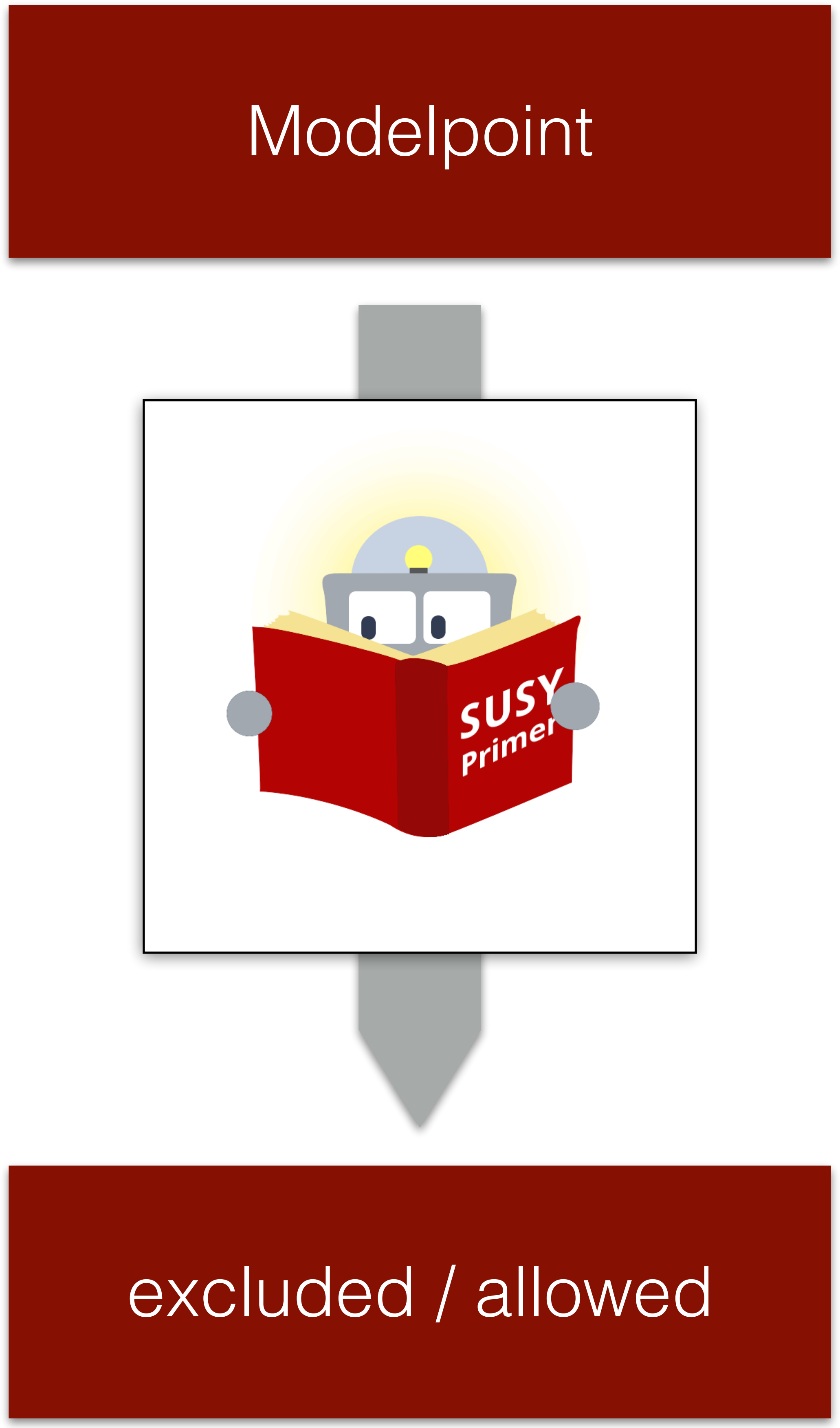
and about the software

SUSY-AI

- Algorithms (both 8TeV and 13TeV) are publicly available at http://susyai.hepforge.org

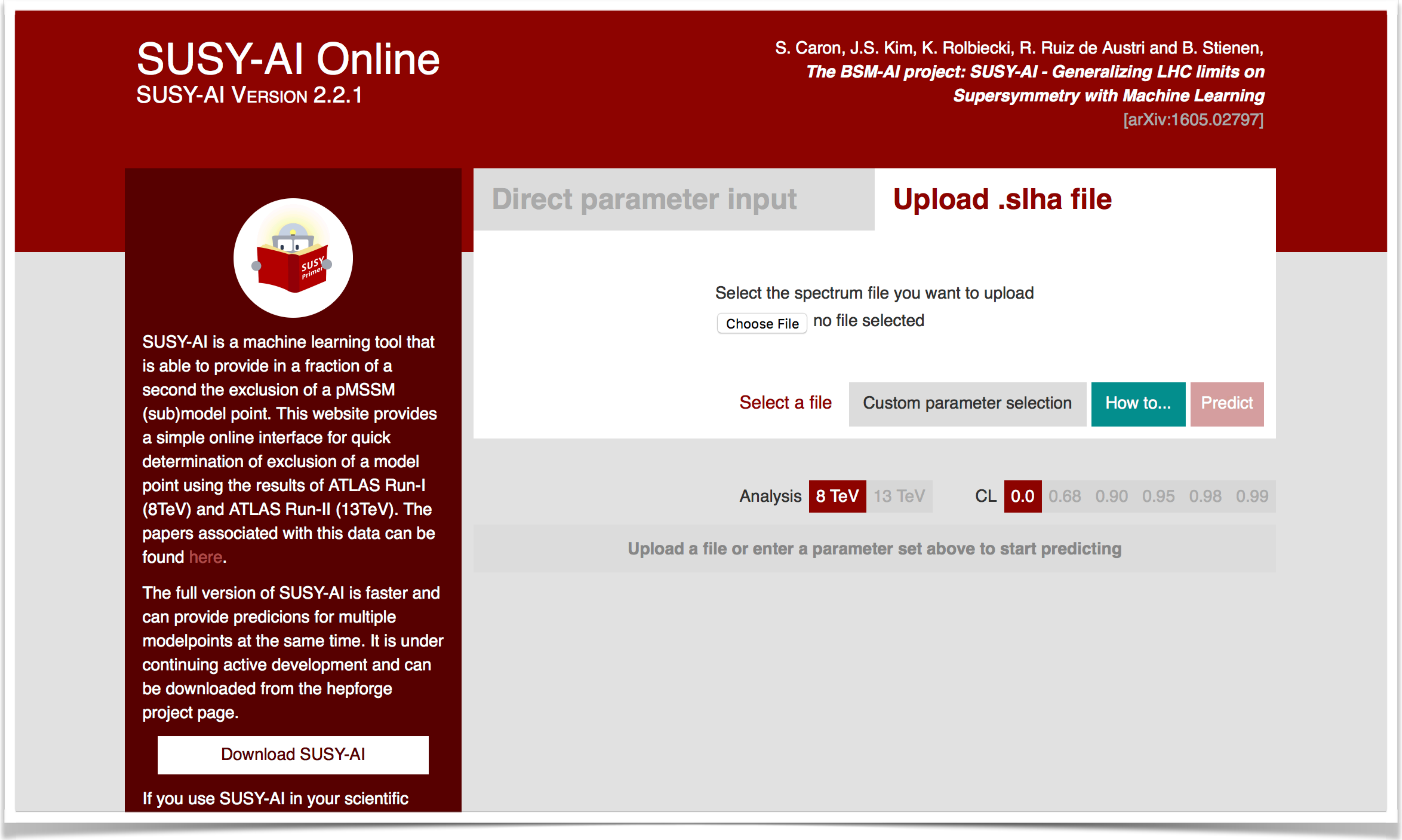
```python
from susyai import susyai
import numpy as np

sa = susyai("susyai_classifier_python_v3.pkl")
data = np.array([[30, 4.0276e2, 7.3196e2, 2.1862e3, 1.0,
          4.0713e3, 4.4890e3, 4.4752e3, 4.4743e3, 2.8806e3,
          3.7855e3, 1.3240e3, 2.9076e3, 4.2226e3, 4.2056e3,
          3.4290e3, 3.8608e3, -4.3154e3, -8.1538e3, -7.3680e3]])
clas, pred, cert = sa.predict(data)
```

- Up to 5,000 model point predictions per second / CPU



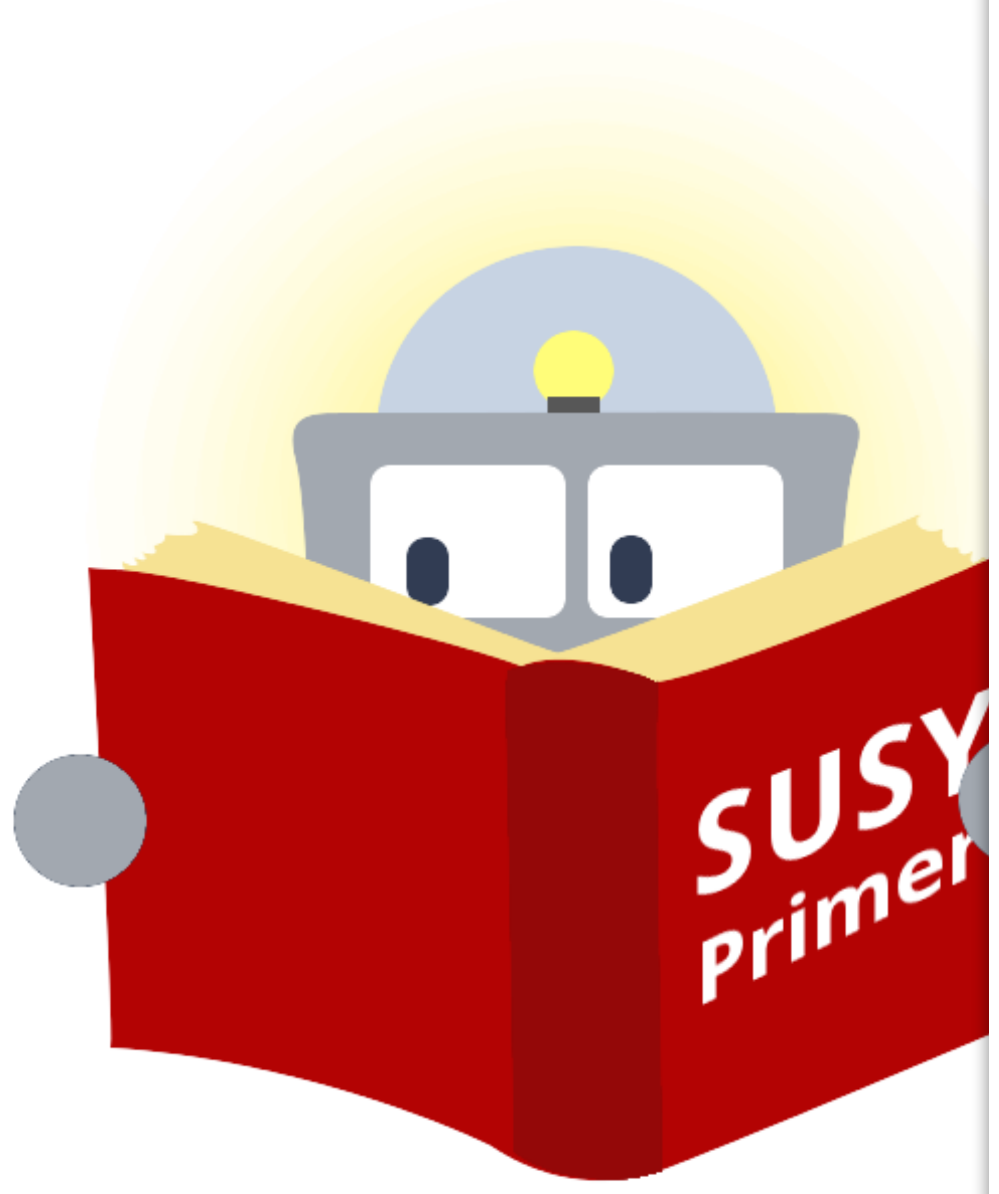
SUSY-AI online

http://www.susy-ai.org/



Eur. Phys. J. C (2017) 77: 257
DOI: 10.1140/epjc/s10052-017-4814-9

Conclusion



- We created a Machine Learning algorithm that can predict model point exclusion in a fraction of a second

- Website is online and algorithm is publicly available (you can start applying LHC limits to your data right away!)

- It works within the general pMSSM, but method is not limited to this parameter space (let me know if you have data!)

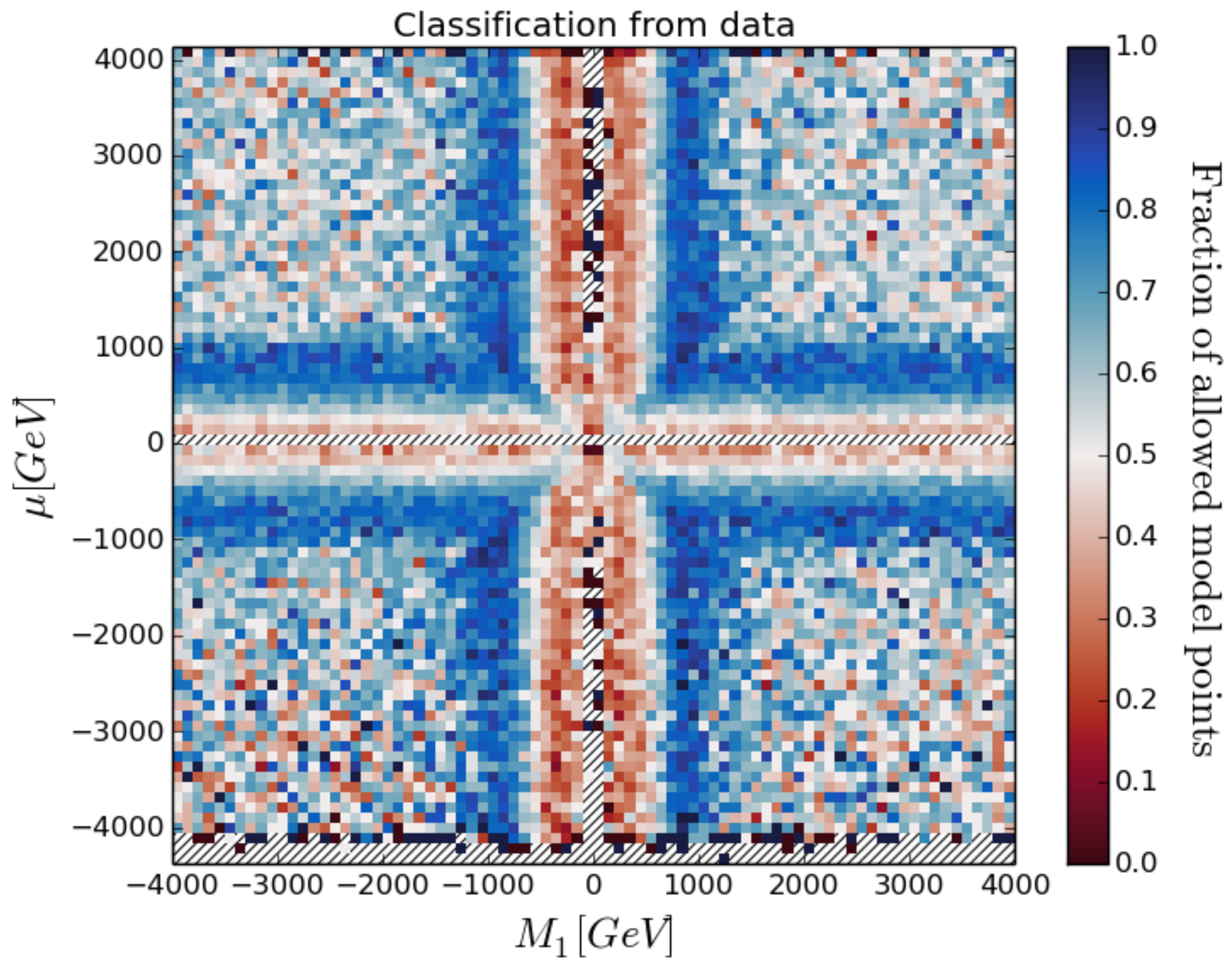
- Algorithm can be stored: method can be used to communicate multivariate results and analyses

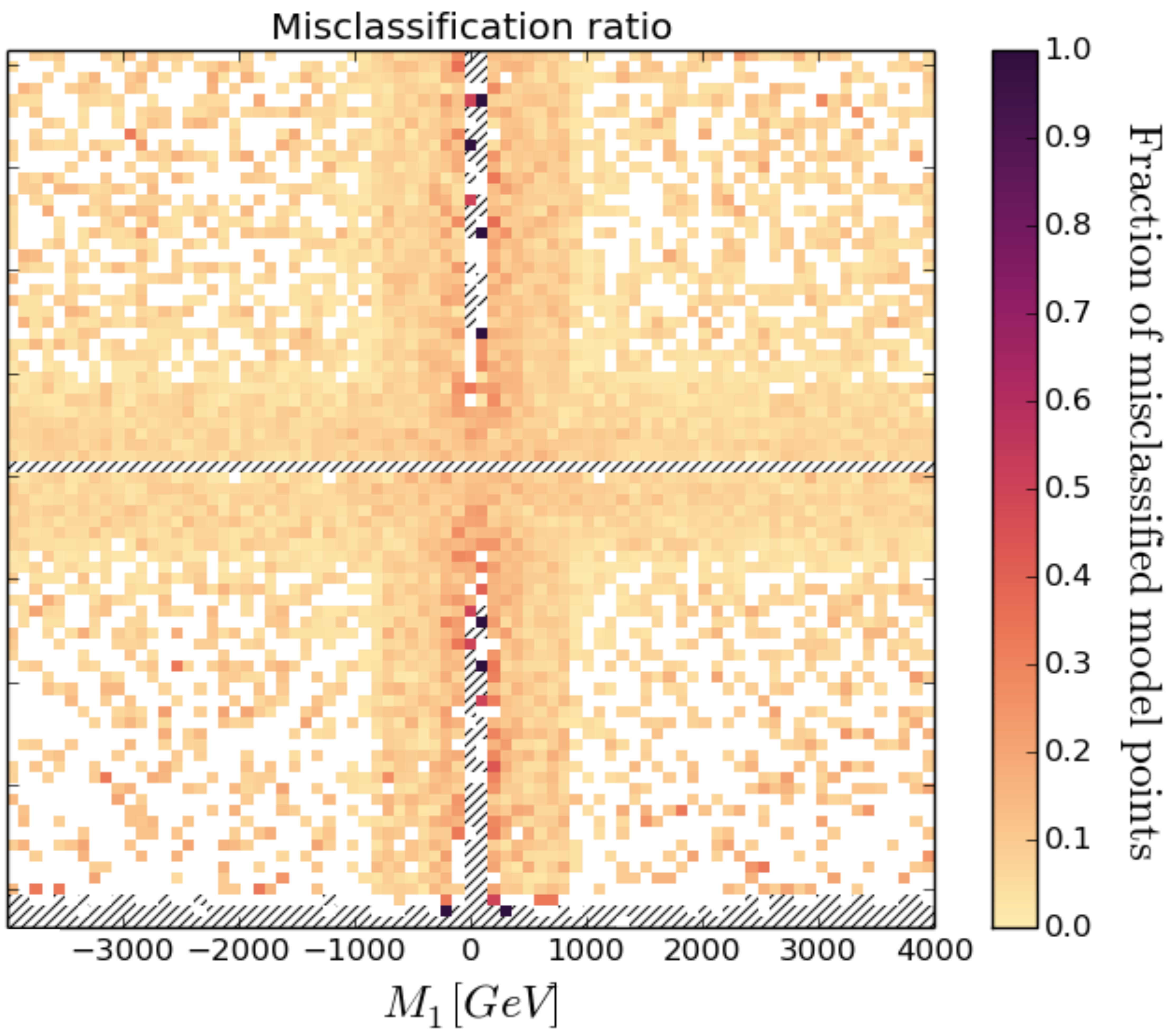

Back-up

Performance M1 vs mu

93.2% accuracy @ 8TeV

92.7% accuracy @ 13 TeV

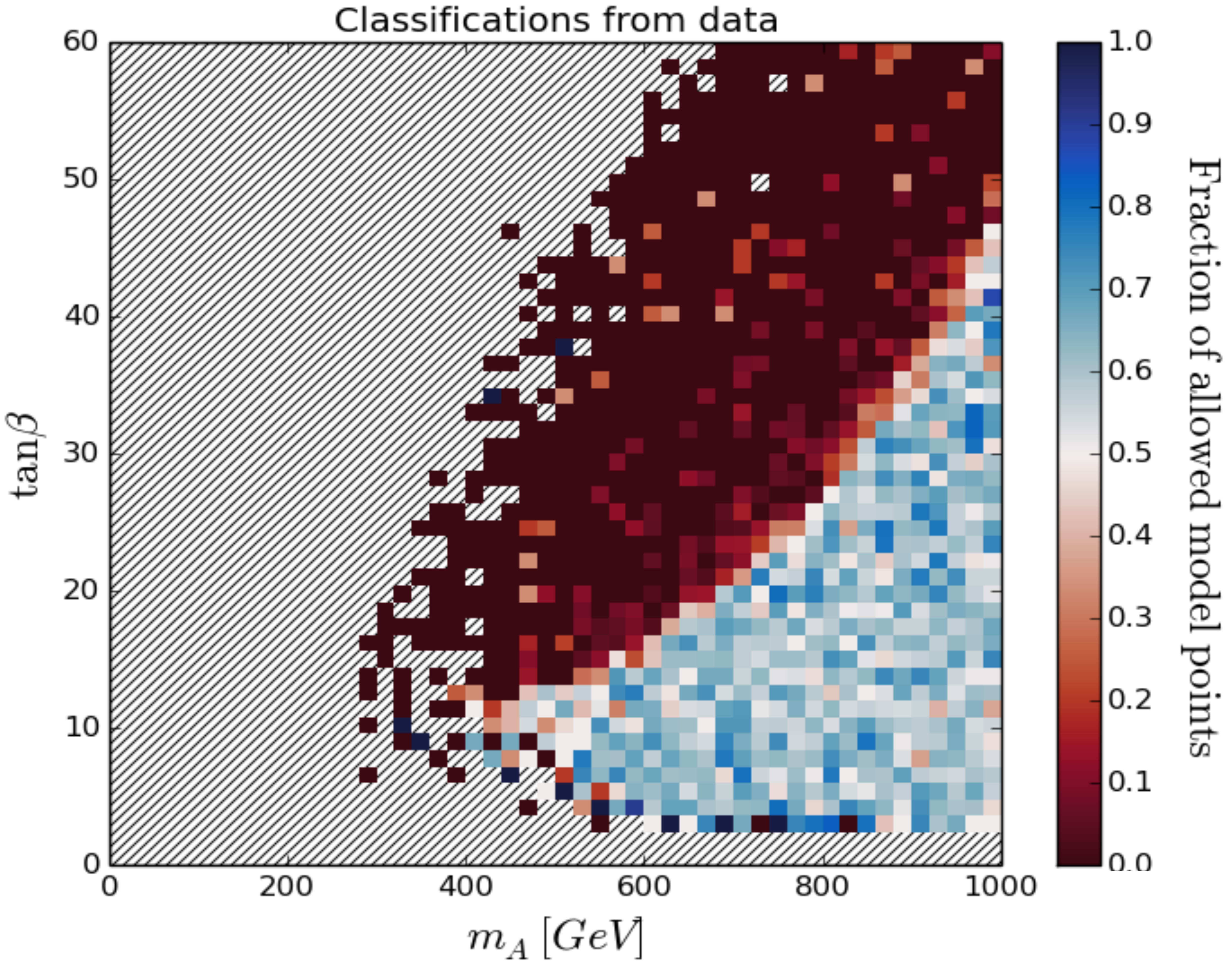


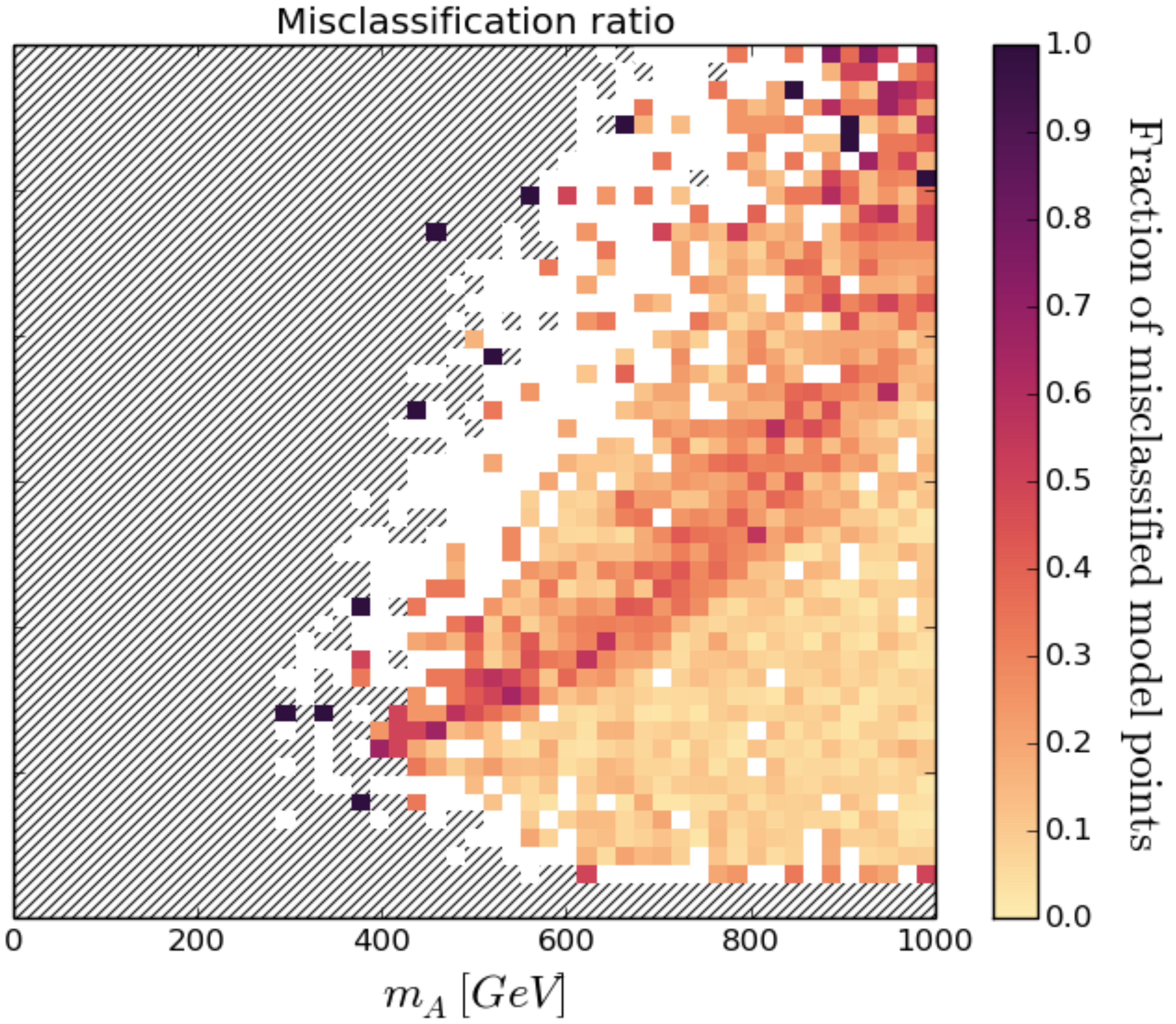


Performance mA vs tan(beta)

93.2% accuracy @ 8TeV

92.7% accuracy @ 13 TeV

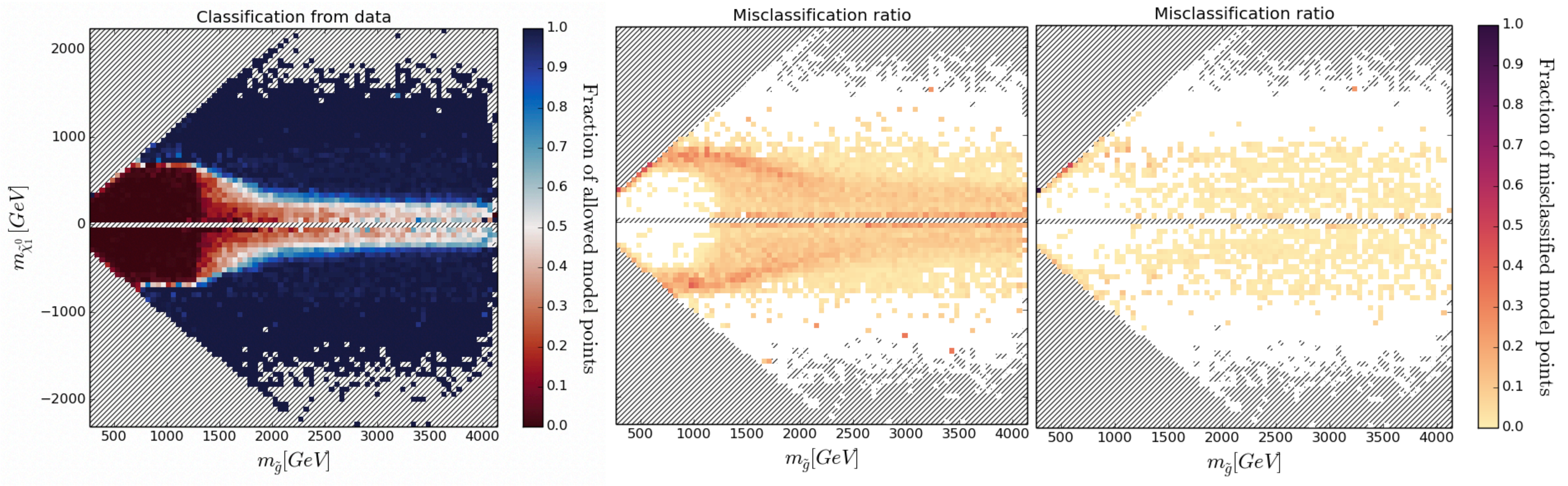




Confidence (>95%) gluino vs neutralino1

99.1% accuracy on 70.6% of total data @ 8TeV

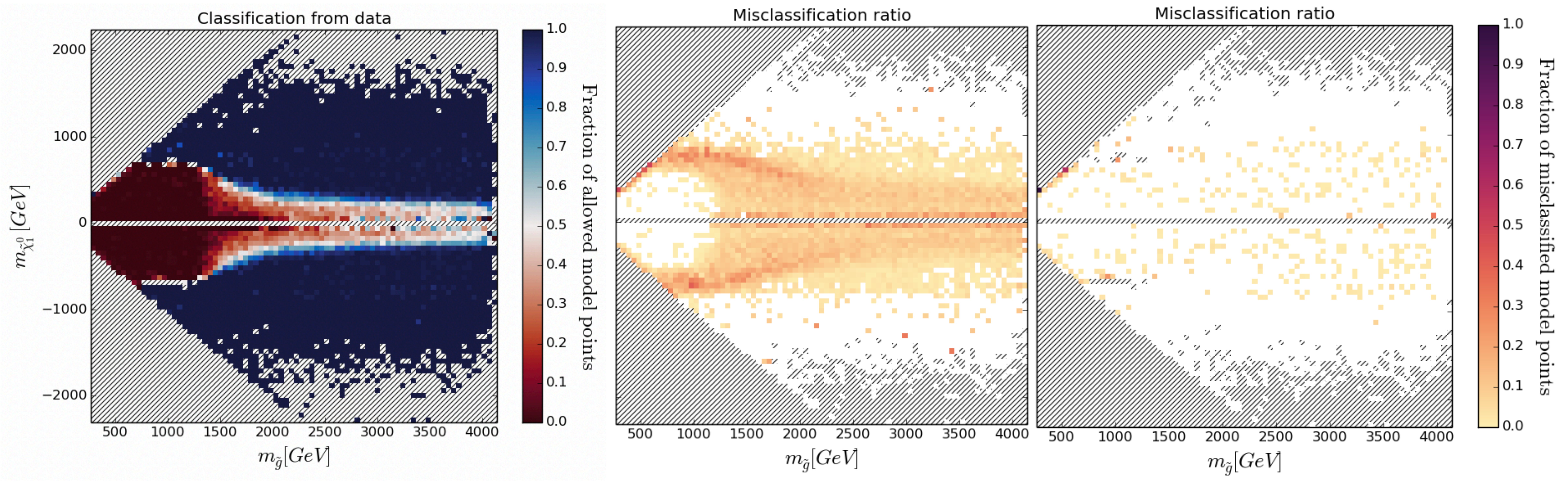
99.0% accuracy on 68.0% of total data @ 13 TeV



Confidence (>99%) gluino vs neutralino1

99.7% accuracy on 51.6% of total data @ 8TeV

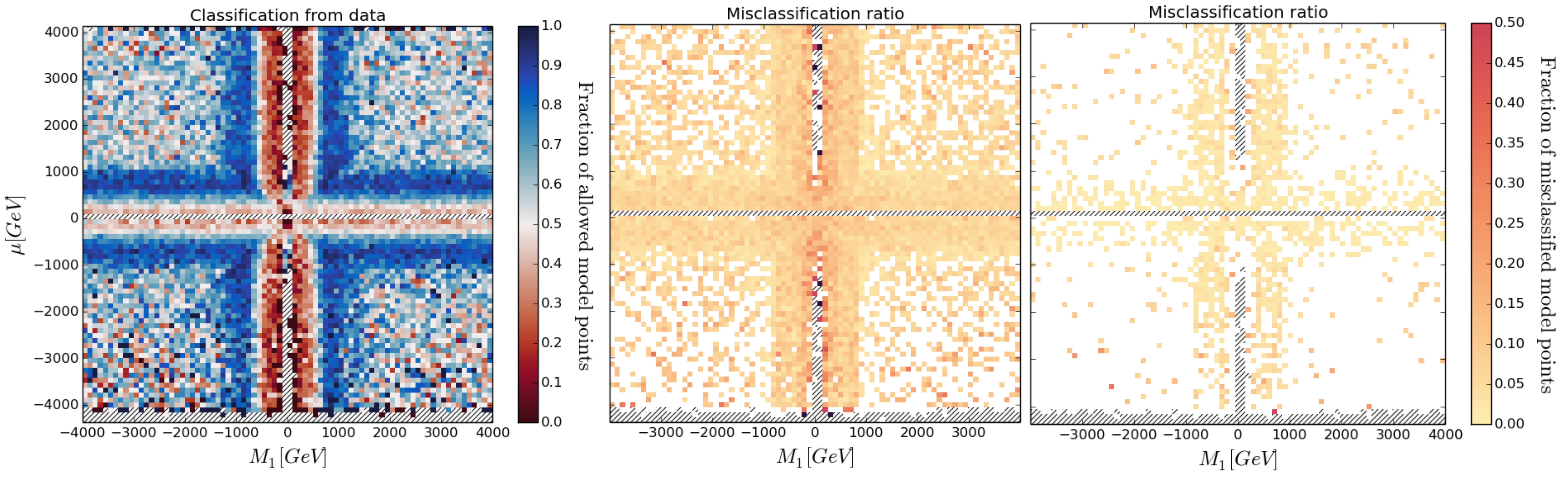
99.7% accuracy on 47.6% of total data @ 13 TeV



Confidence (>95%) M1 vs mu

99.1% accuracy on 70.6% of total data @ 8TeV

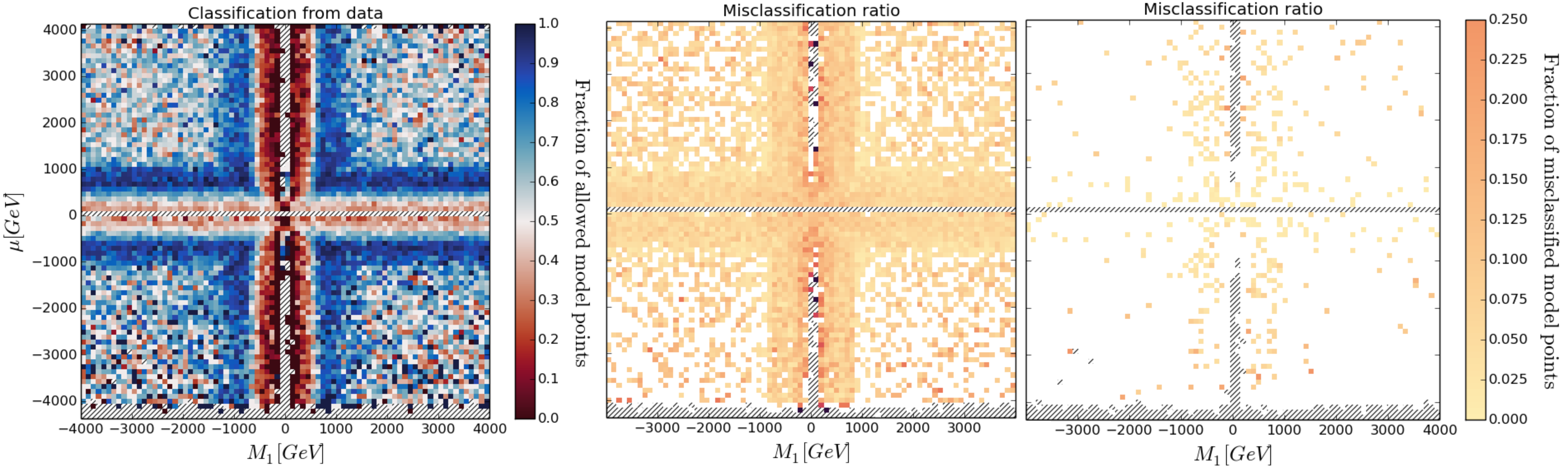
99.0% accuracy on 68.0% of total data @ 13 TeV



Confidence (>99%) M1 vs mu

99.7% accuracy on 51.6% of total data @ 8TeV

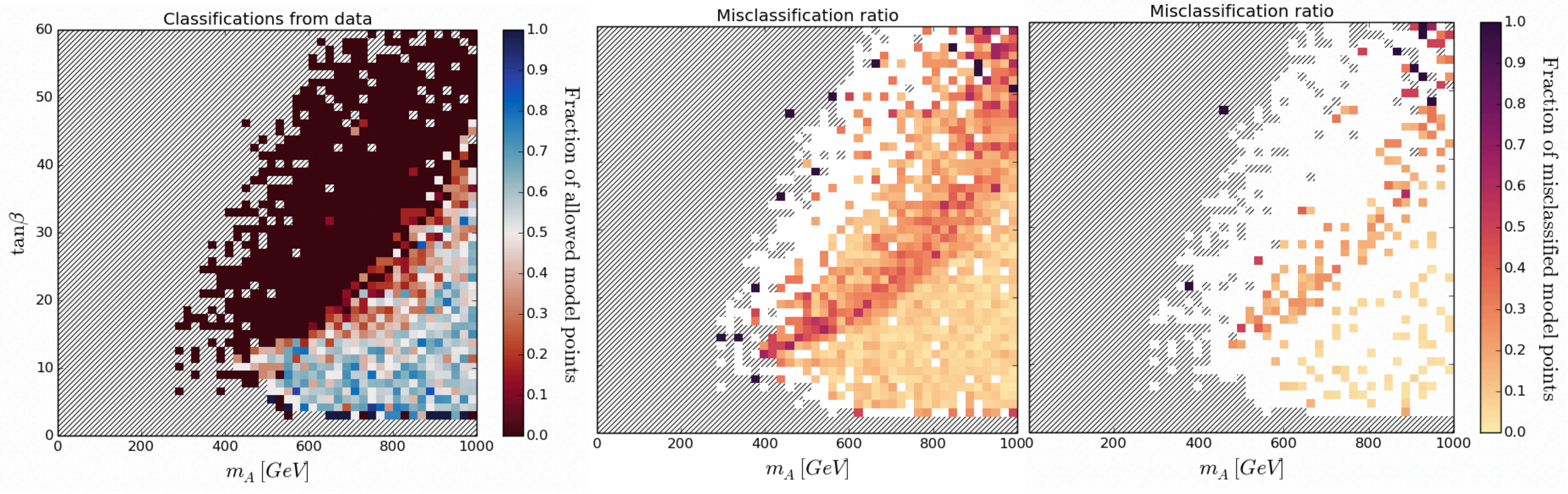
99.7% accuracy on 47.6% of total data @ 13 TeV



Confidence (>95%) mA vs tan(beta)

99.1% accuracy on 70.6% of total data @ 8TeV

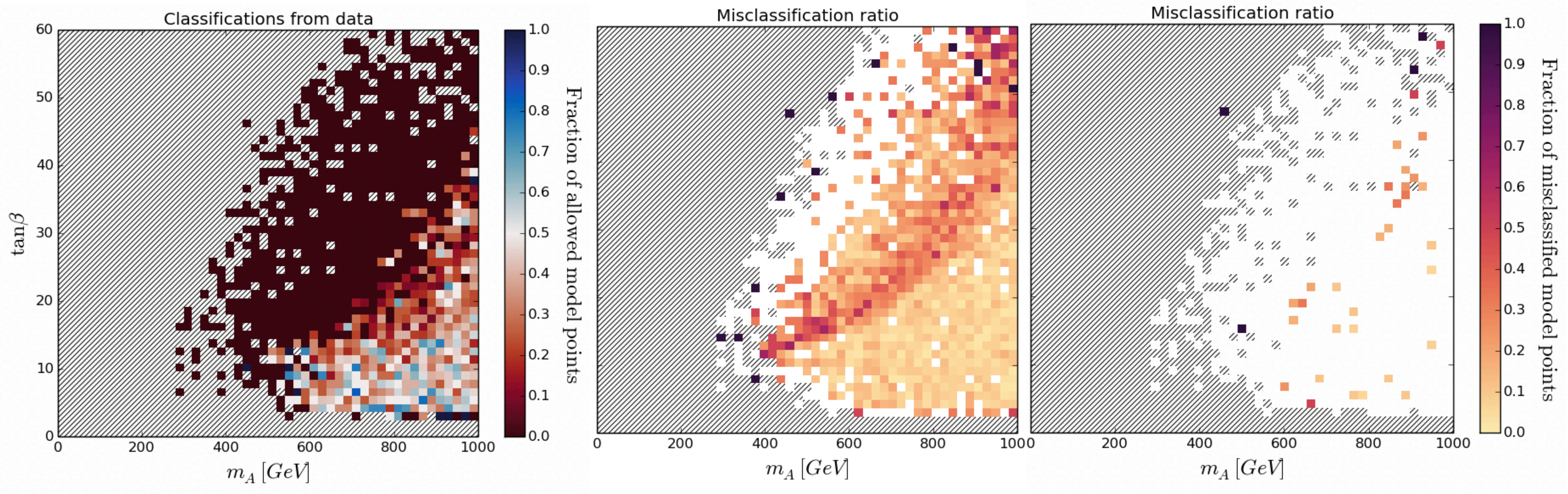
99.0% accuracy on 68.0% of total data @ 13 TeV



Confidence (>99%) mA vs tan(beta)

99.7% accuracy on 51.6% of total data @ 8TeV

99.7% accuracy on 47.6% of total data @ 13 TeV



Feature importances

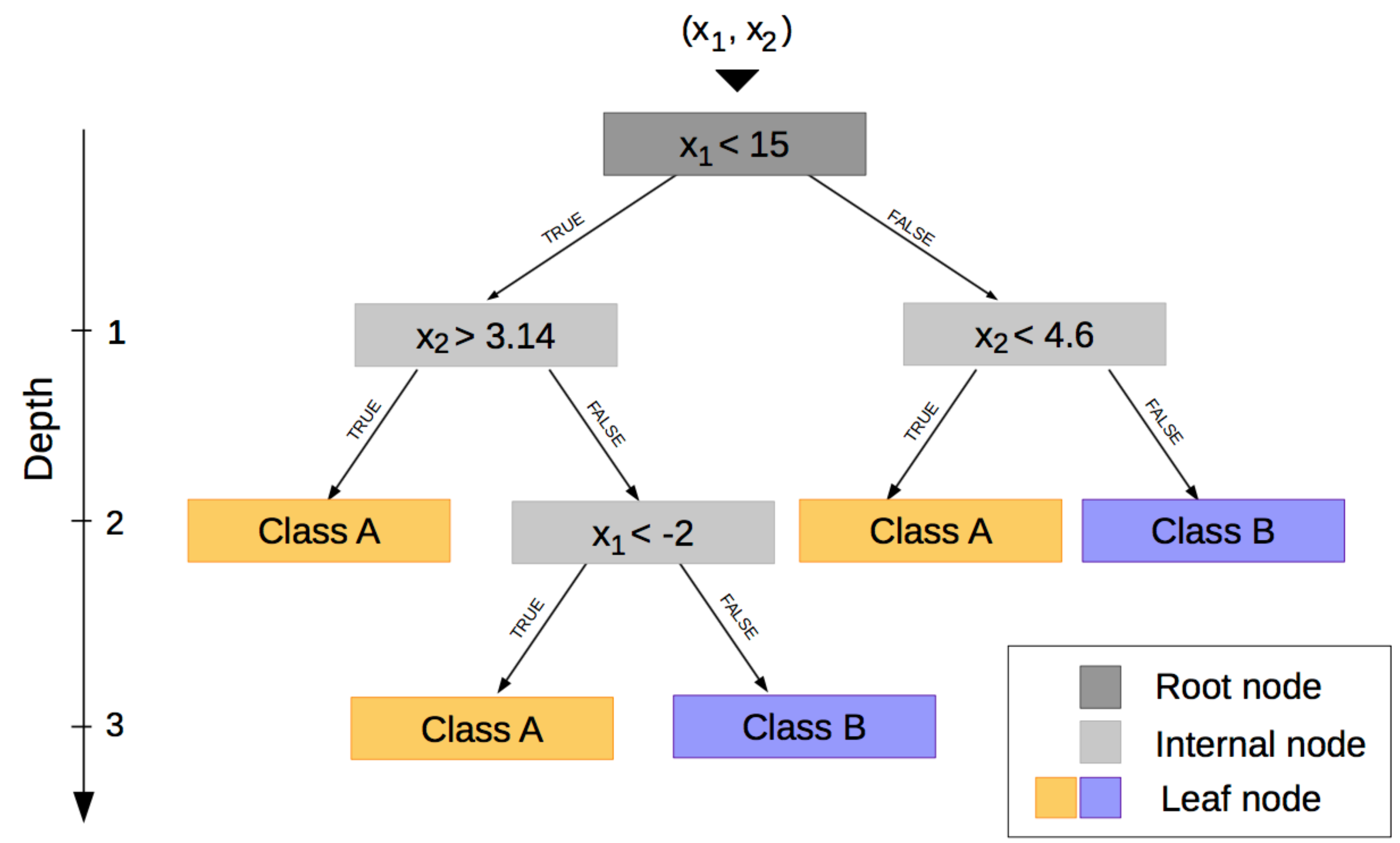
- Splits in Decision Trees are made based on Gini impurity

$$I = \sum_{i=1}^{C} f_i \cdot (1 - f_i) = 1 - \sum_{i=1}^{C} f_i^2$$

- Weighted impurity (variable importance) per feature can be calculated via:

$$\sum_{k \in \text{nodes splitting } j} \frac{\text{model points at node } k}{\text{total number of model points}} \cdot \text{impurity change}$$

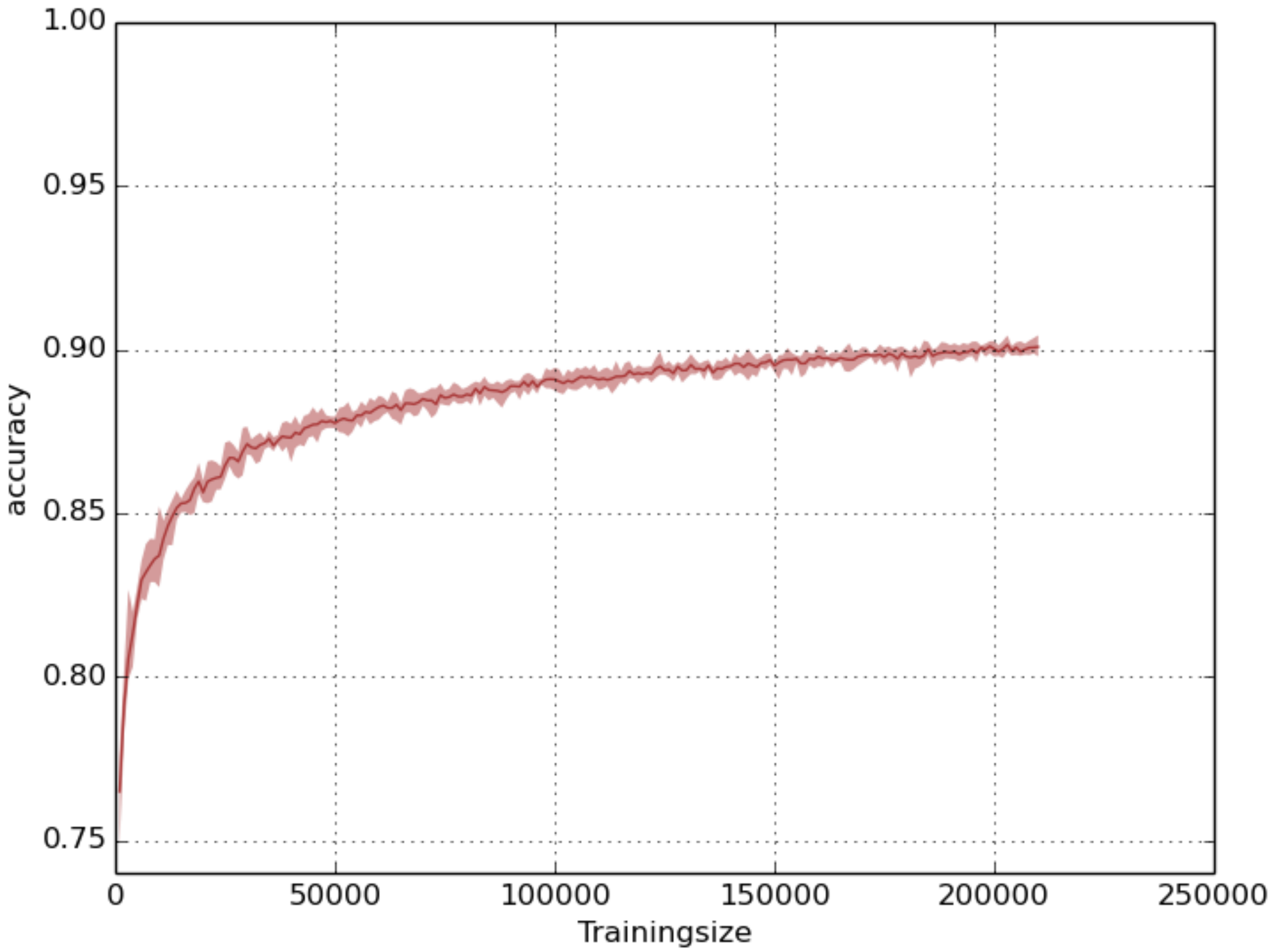
- Significant differences in variable importance between features!



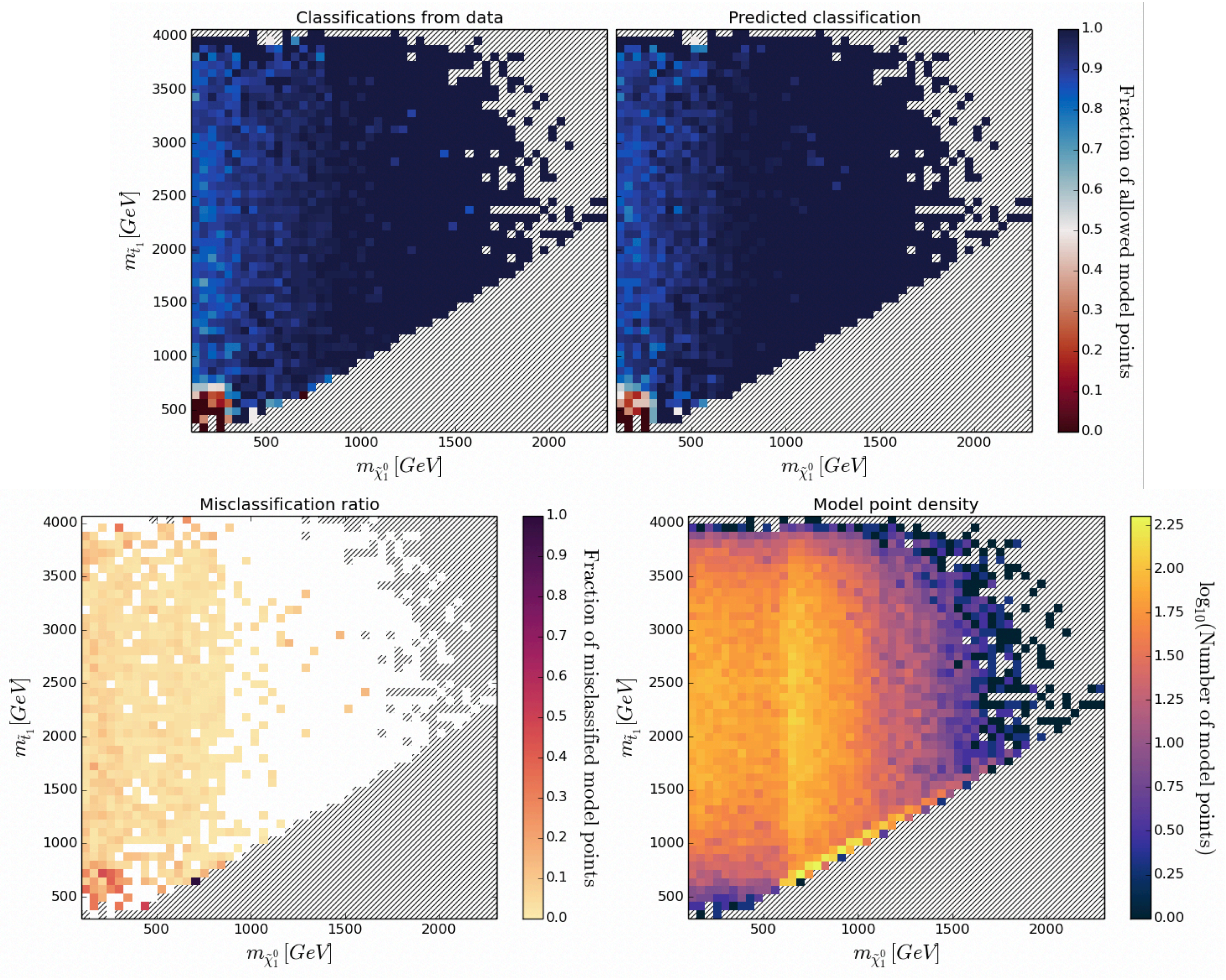
Parameter	Importance
mL1	0.021
me1	0.019
mL3	0.014
me3	0.014
mQ1	0.079
mu1	0.066
md1	0.037
mQ3	0.026
mu3	0.018
md3	0.026

Parameter	Importance
M1	0.058
M2	0.164
mu	0.130
M3	0.242
At	0.013
Ab	0.012
Atau	0.012
mA2	0.031
tanbeta	0.019

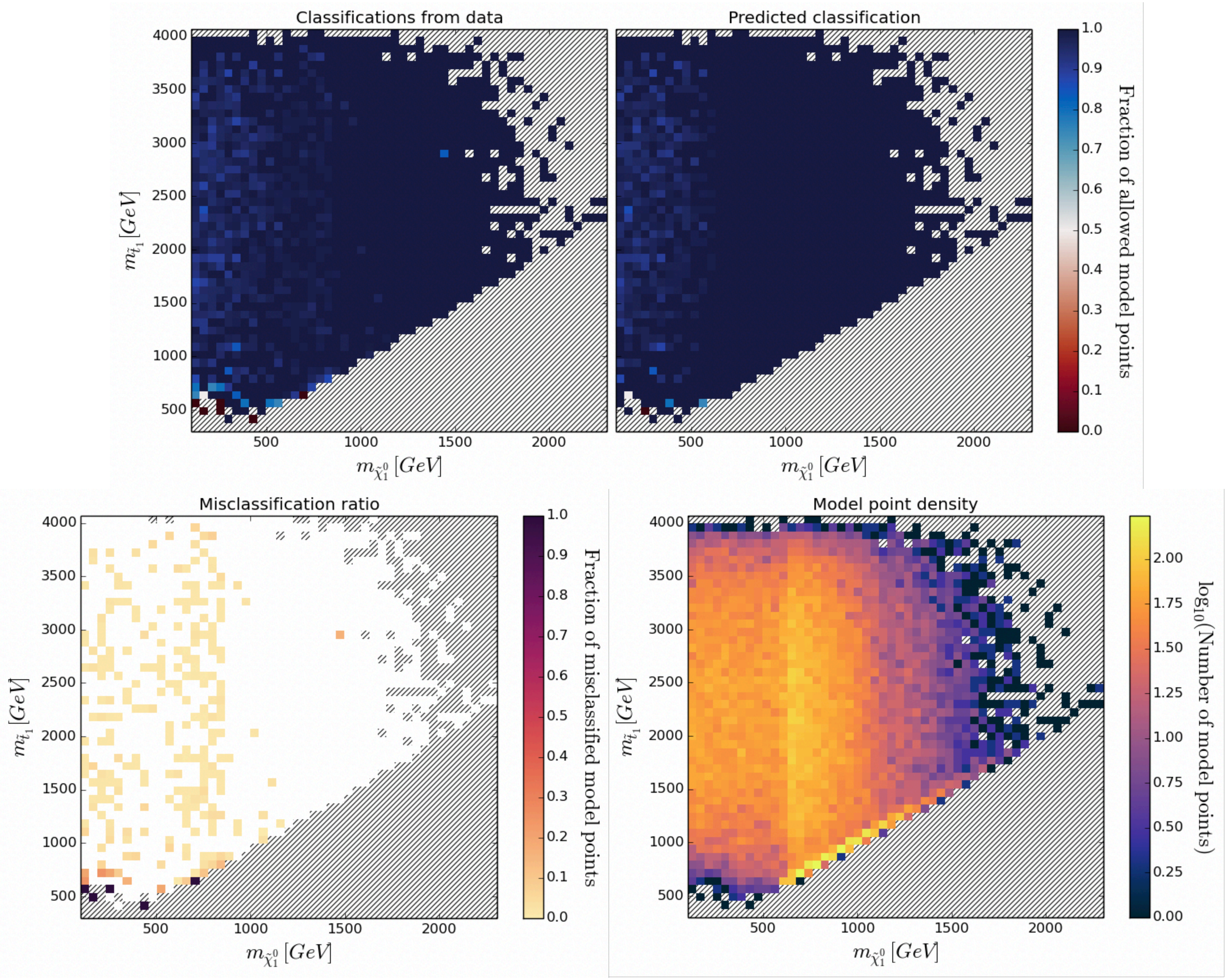
Learning curve



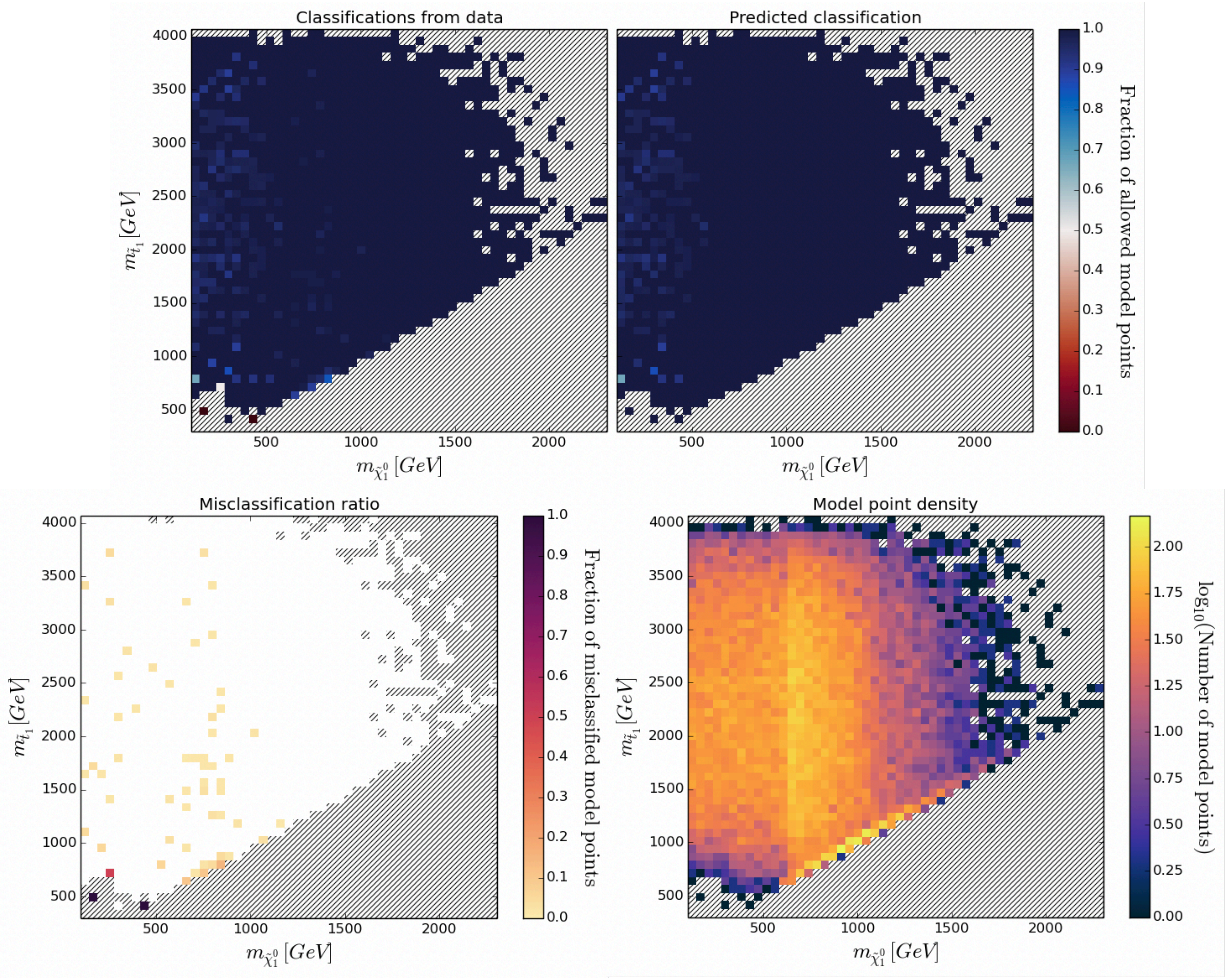
Sparsity


Classifications from data
Predicted classification
Fraction of allowed model points
Misclassification ratio
Fraction of misclassified model points
Model point density
$\log_{10}$(Number of model points)
$m_{\tilde{t}_1}$ [GeV]
$m_{\tilde{\chi}_1^0}$ [GeV]

Sparsity
95% CL



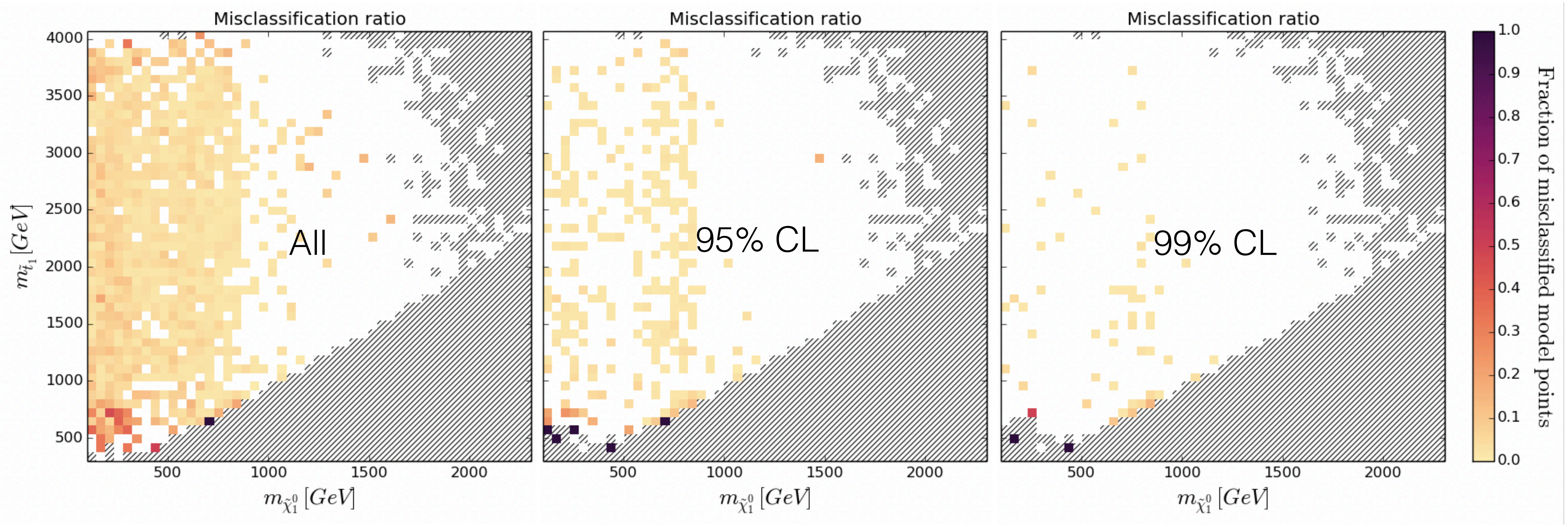
Sparsity
99% CL



Sparsity

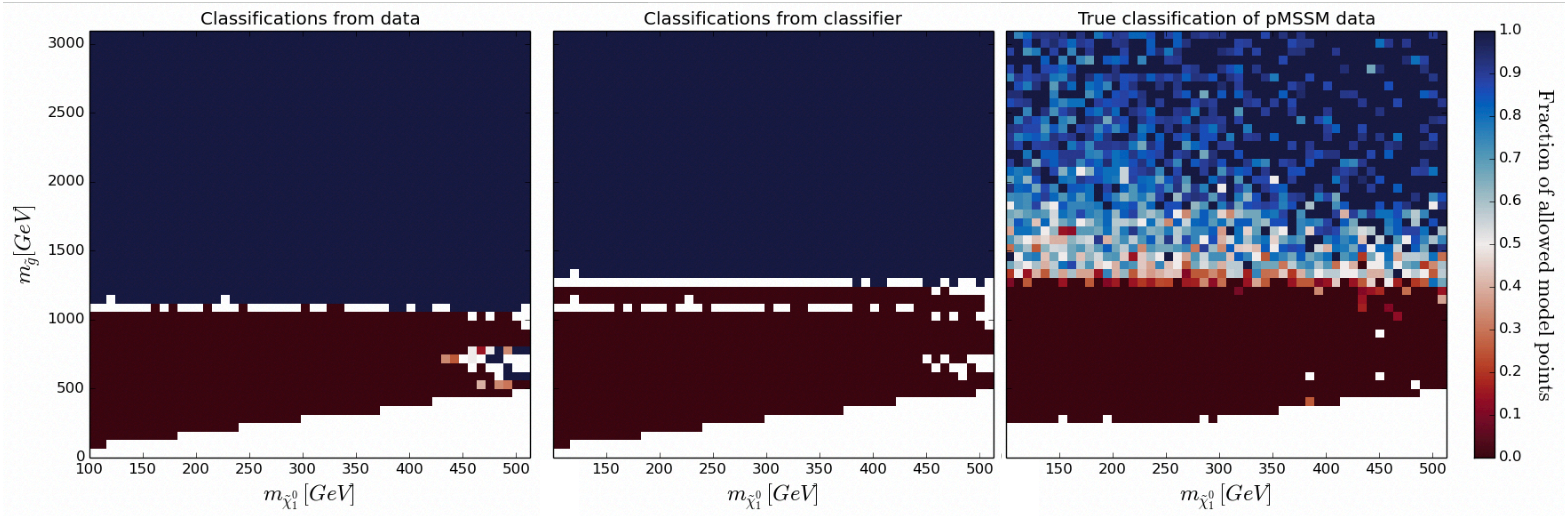
- Errors in low energy region can be taken care of by applying confidence limits

- However: comes at the cost of sensitivity due to data sparsity
  —> more data is needed



Natural SUSY

	Definition	Scanned range [GeV]
$m_{\tilde{Q}_3}$	3rd generation left-handed squark breaking mass	[100, 1500]
$m_{\tilde{U}_3}$	3rd generation up-type right-handed squark breaking mass	[100, 1500]
M_3	Gluino mass parameter	[100, 3000]
A_t	Stop trilinear coupling	[-3000, 3000]
μ	Higgsino mass parameter	[100, 500]
$\tan\beta$	Ratio of vacuum expectation values of H_u^0 and H_d^0	[1, 20]



Out-of-bag vs train:test split

Accuracy:
(TP+TN) / all

Precision:
TP / (TP+FP)

Sensitivity
TP / (TP+FN)

Negative prediction value
TN / (TN+FN)

Specificity
TN / (TN+FP)

Out-of-bag

CL	#	# / total	Accuracy	Precision	Sensitivity	NPV	Specificity
0.0	310 324	1.0000	0.93226	0.93951	0.94665	0.92152	0.91133
0.68	289 371	0.93248	0.95735	0.96072	0.96835	0.95222	0.94094
0.95	219 233	0.70646	0.99094	0.99092	0.99426	0.99096	0.98573
0.98	184 230	0.59367	0.99543	0.99573	0.99672	0.99496	0.99346
0.99	160 034	0.51570	0.99708	0.99747	0.99764	0.99649	0.99624

Dataset splitting train:test = 75:25

CL	#	# / total	Accuracy	Precision	Sensitivity	NPV	Specificity
0.0	77 581	1.0000	0.92271	0.91653	0.93049	0.92912	0.91491
0.68	70 375	0.90712	0.9545	0.95516	0.95302	0.95386	0.95595
0.95	48 900	0.63031	0.99022	0.99047	0.9893	0.99	0.99109
0.98	39 815	0.51321	0.99485	0.99559	0.99353	0.99419	0.99604
0.99	34 004	0.43830	0.99644	0.99685	0.99554	0.99608	0.99724

SUSY-AI Online