IDAO 2021: Baobab solution

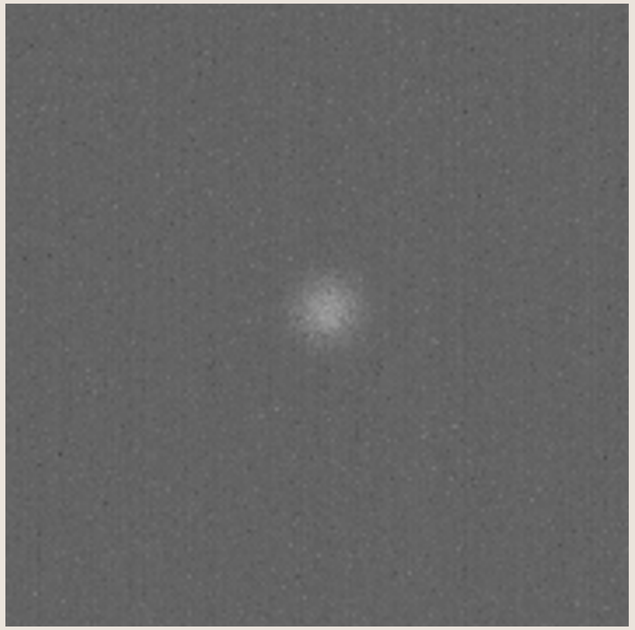
Oleg Filatov[1], Andrei Filatov[2,3], Andrey Znobishchev[4]

[1] Deutsches Elektronen-Synchrotron
[2] Moscow Institute of Physics and Technology
[3] École polytechnique fédérale de Lausanne
[4] Skolkovo Institute of Science and Technology

Our solution

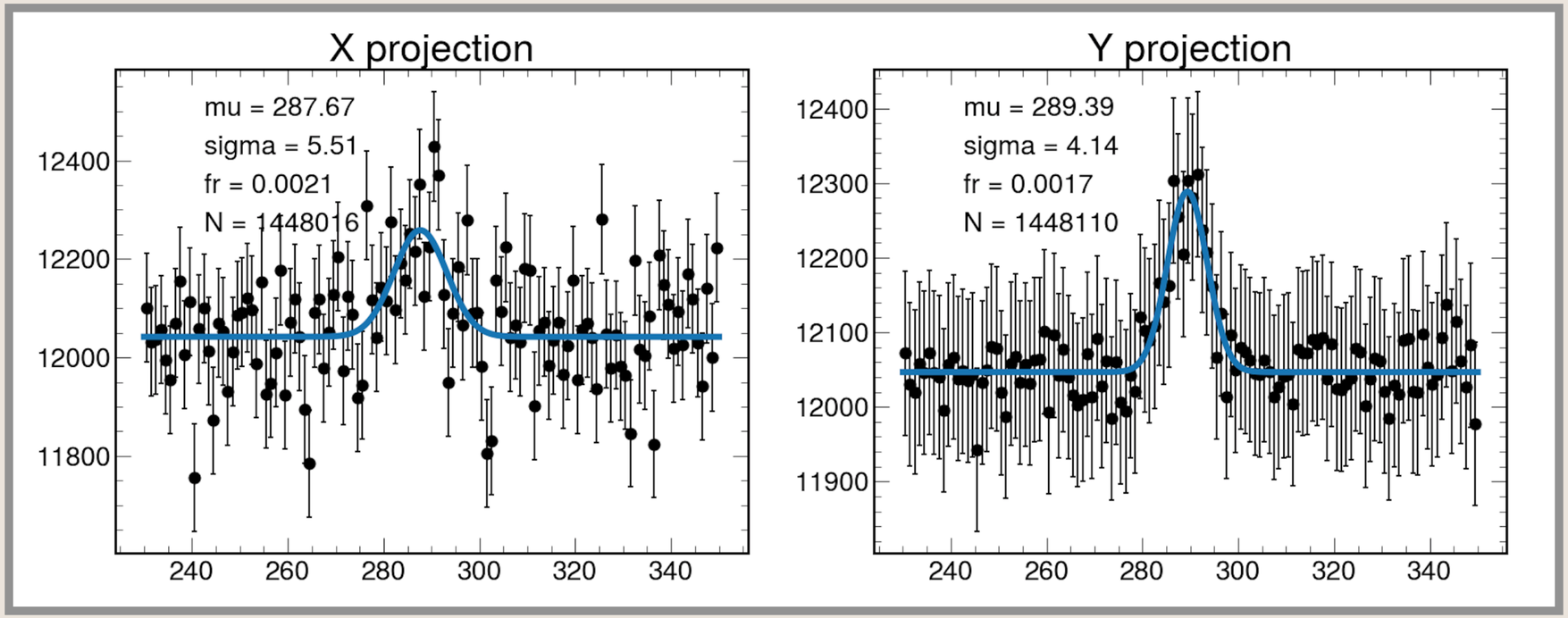
Feature engineering

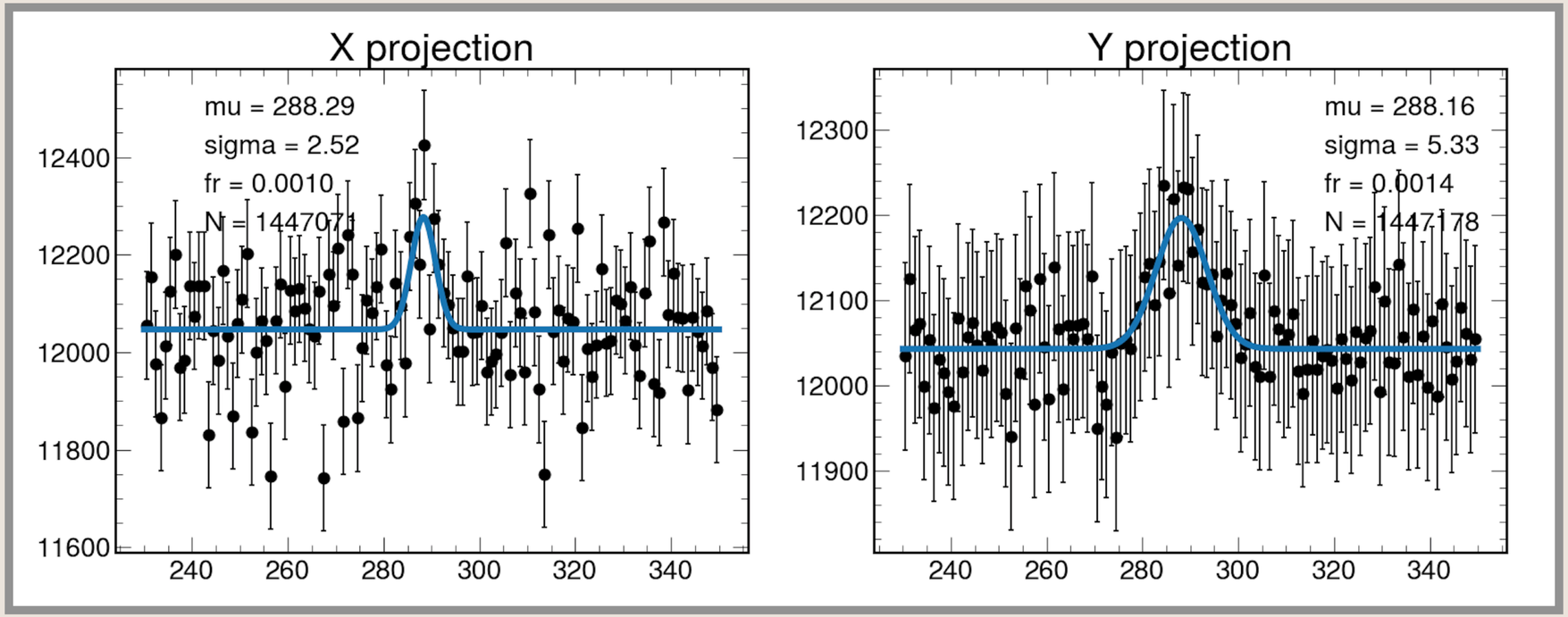
- **Idea:** firstly, extract knowledge from images by constructing informative features
- **Observation:** images look like 2D Gaussian function sitting on top of flat background
- **Approach:** fit separately X/Y projections with a mixture of flat and normal distributions *
  - Yes, we could've done 2D fit, but didn't have time to implement it
  - Took ~5 hours to process the data
- **Final features:** parameters of the fit
  - μ, σ, fr, sig_count, bkgr_count, sig_density + their errors (X,Y projections)
  - dsigma, dfr, dmu: parameter difference between X and Y fits
  - abs_dmu = abs(mu - 288) (X,Y projections)
  - χ^2, p-value under bkgr & sig+bkgr hypotheses (X,Y projections)
  - fit convergence info (e.g. HESSE_valid) (X,Y projections)
- **Note:** in the end we didn't use all of them



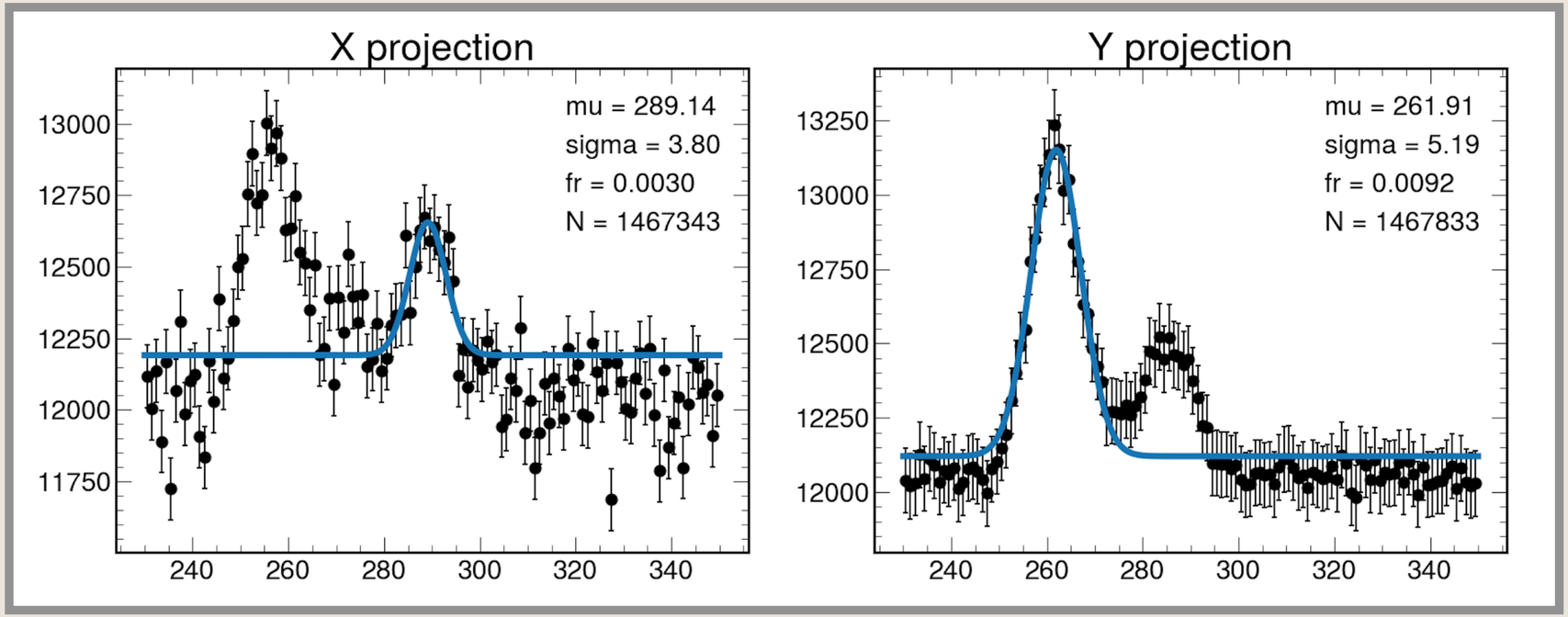
*if you're curious, we used custom implementation of χ^2 binned fit in iMinuit speeded up by numba

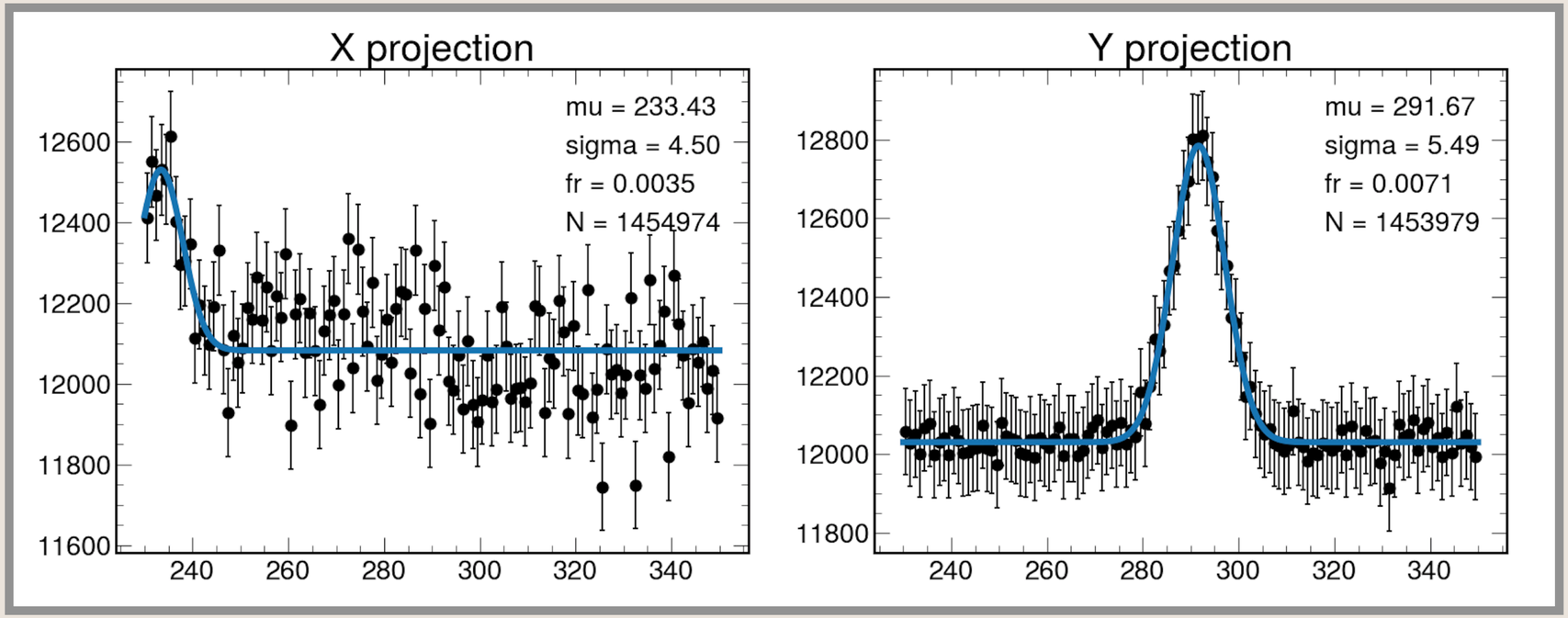
ER, 3 keV


X projection
mu = 287.67
sigma = 5.51
fr = 0.0021
N = 1448016
Y projection
mu = 289.39
sigma = 4.14
fr = 0.0017
N = 1448110

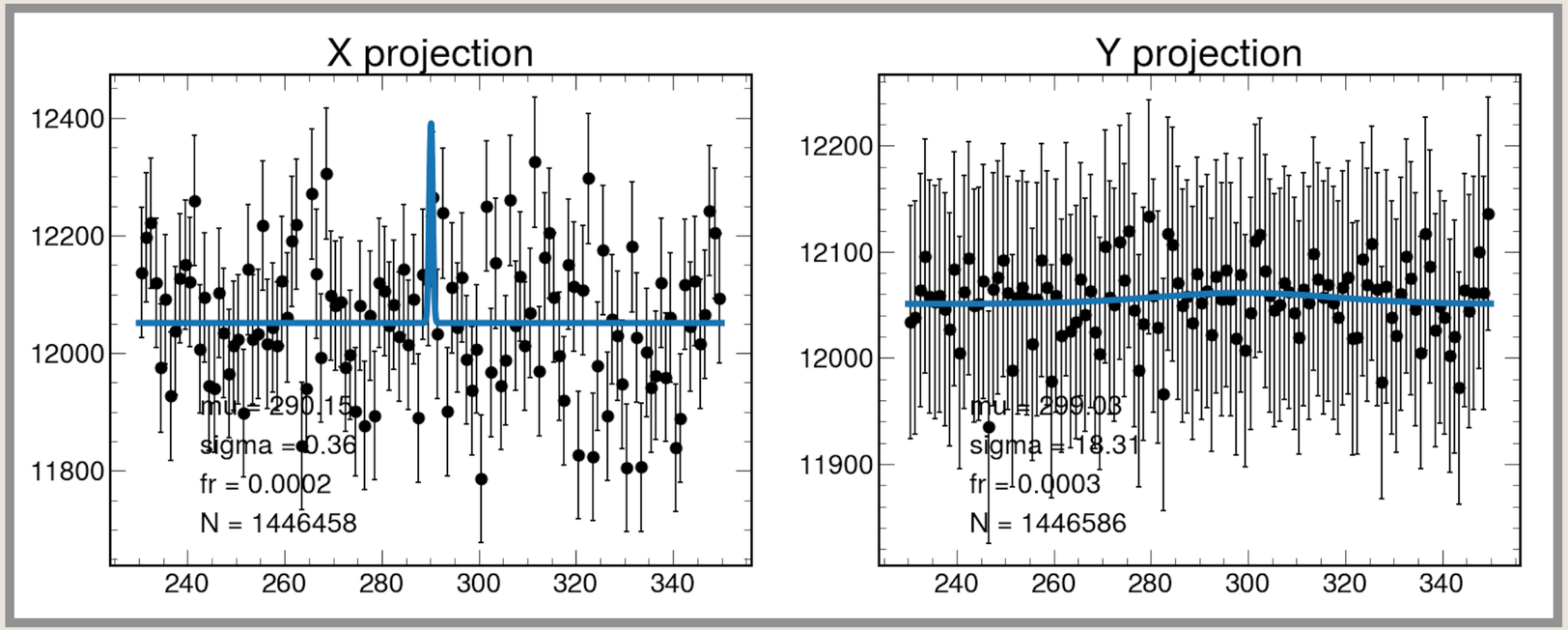

X projection
mu = 288.29
sigma = 2.52
fr = 0.0010
N = 1447071
Y projection
mu = 288.16
sigma = 5.33
fr = 0.0014
N = 1447178

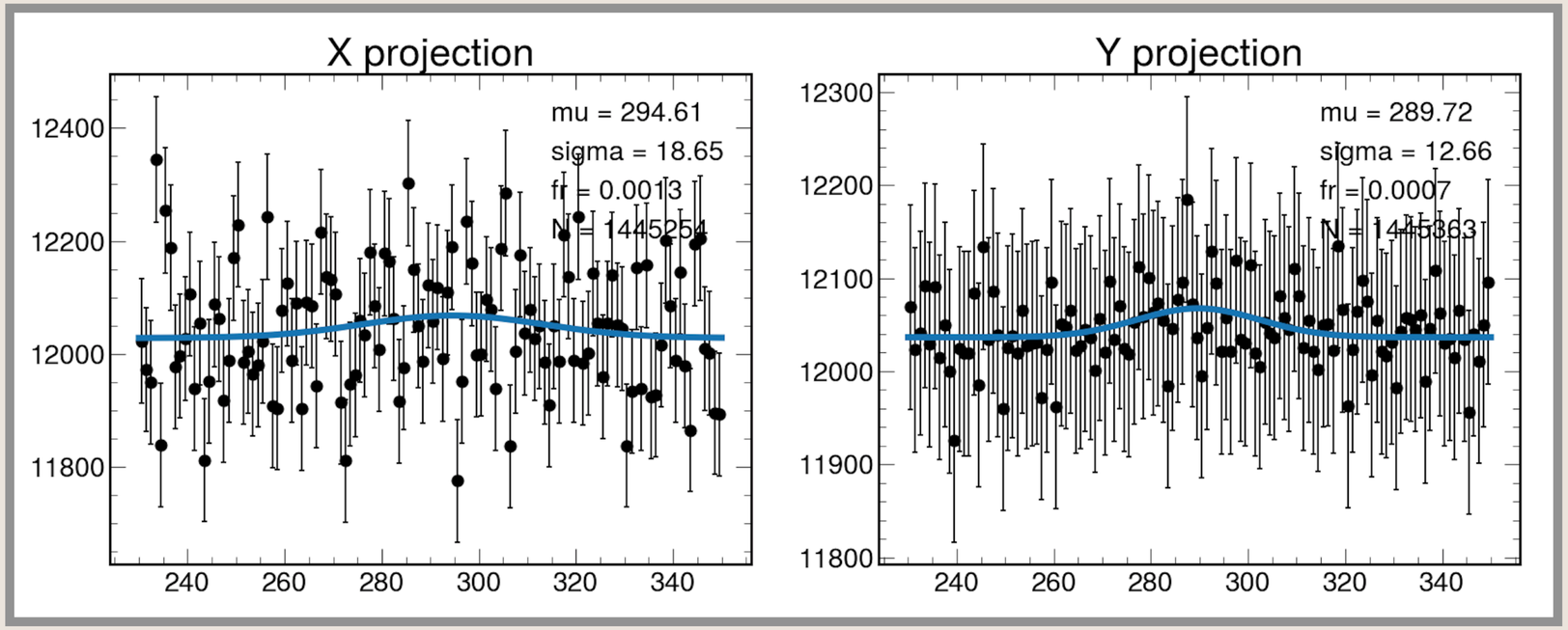
ER, 30 keV


X projection
mu = 289.14
sigma = 3.80
fr = 0.0030
N = 1467343
Y projection
mu = 261.91
sigma = 5.19
fr = 0.0092
N = 1467833

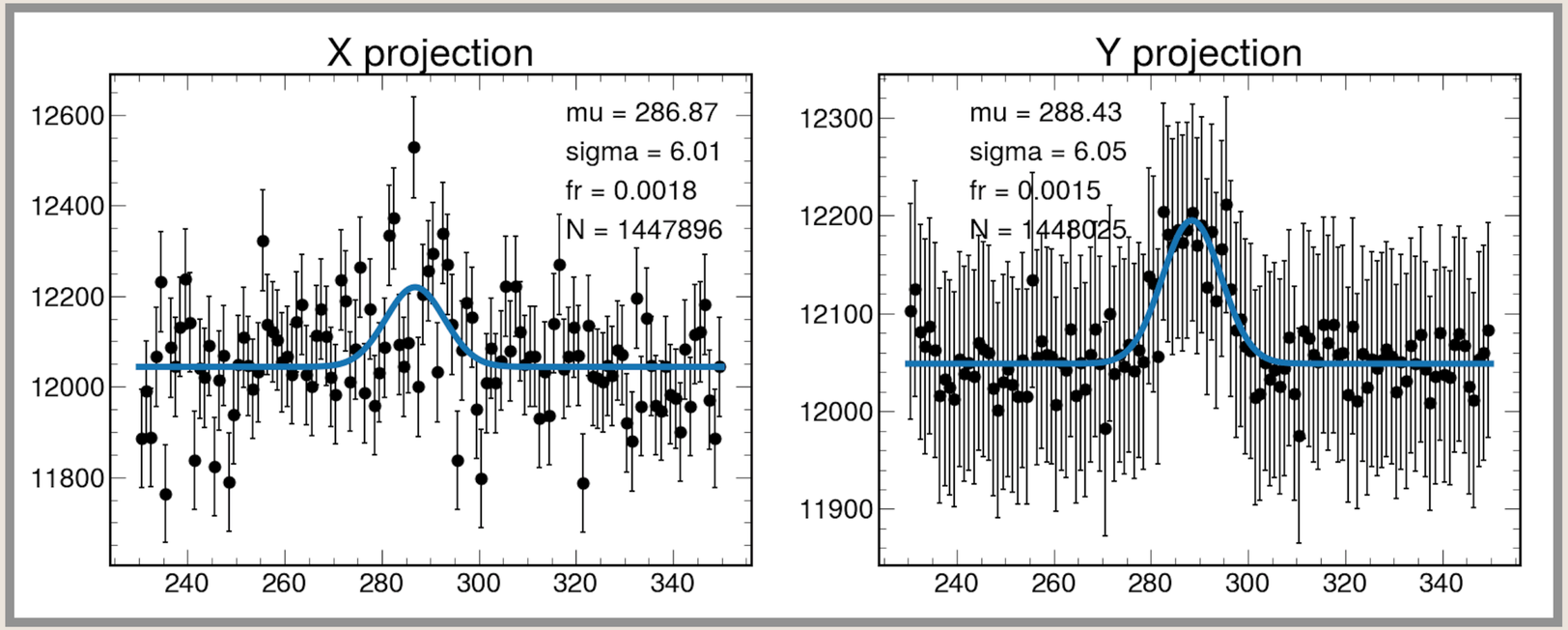

X projection
mu = 233.43
sigma = 4.50
fr = 0.0035
N = 1454974
Y projection
mu = 291.67
sigma = 5.49
fr = 0.0071
N = 1453979

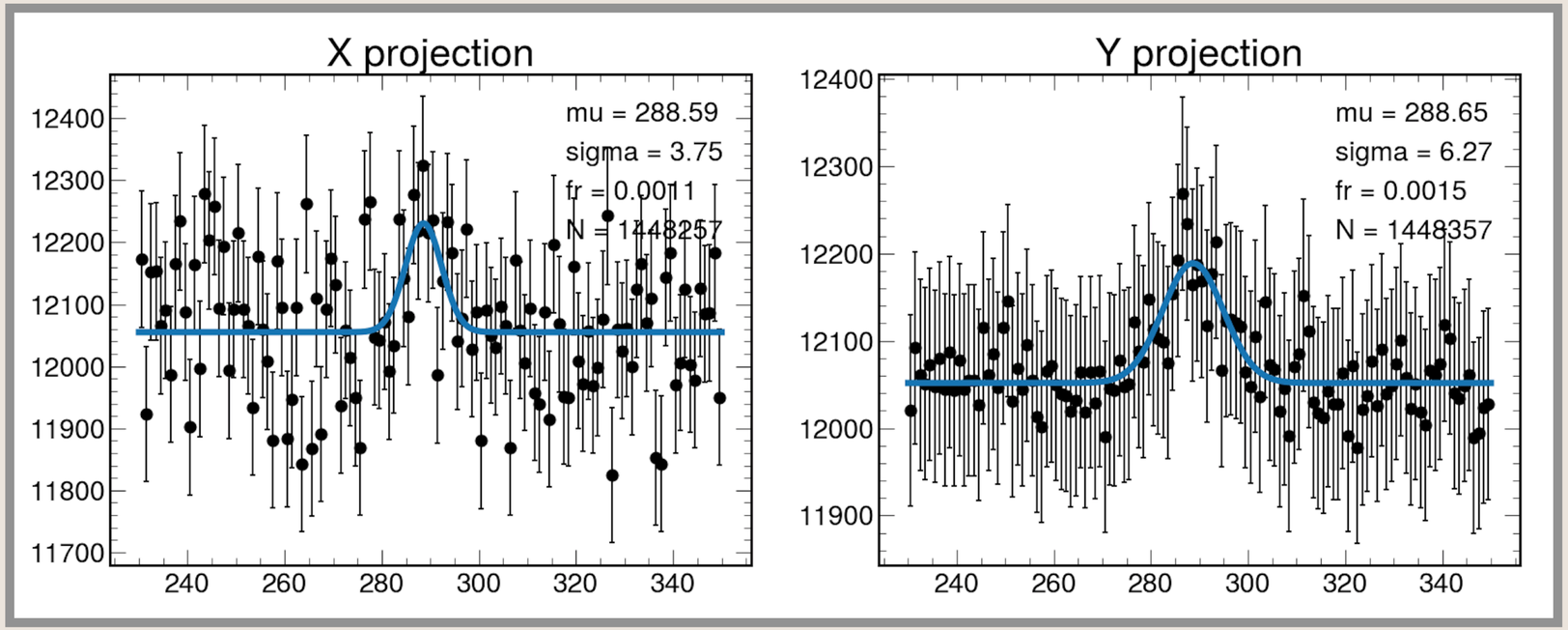
NR, 1 keV


X projection
mu = 290.15
sigma = 0.36
fr = 0.0002
N = 1446458
Y projection
mu = 299.03
sigma = 18.31
fr = 0.0003
N = 1446586


X projection
mu = 294.61
sigma = 18.65
fr = 0.0013
N = 1445254
Y projection
mu = 289.72
sigma = 12.66
fr = 0.0007
N = 1445363

NR, 6 keV


X projection
mu = 286.87
sigma = 6.01
fr = 0.0018
N = 1447896
Y projection
mu = 288.43
sigma = 6.05
fr = 0.0015
N = 1448025


X projection
mu = 288.59
sigma = 3.75
fr = 0.0011
N = 1448257
Y projection
mu = 288.65
sigma = 6.27
fr = 0.0015
N = 1448357

Our solution

First steps

#	Participant	A 1092/2430	B 138/1039	Score
126	Baobab	-1537.93 30d. 10h.	-2295.33 30d. 10h.	-3833.26
1	random team	998.00 30d. 9h.	1000.00 30d. 10h.	1998.00
2	White Material	997.34 26d. 3h.	1000.00 26d. 6h.	1997.34
3	fit_predict	996.00 30d. 10h.	1000.00 30d. 10h.	1996.00
4	DataCrackers	996.67 30d. 5h.	996.67 27d. 7h.	1993.34
5	CooperFactory	994.67 26d. 11h.	998.00 28d. 12h.	1992.67

- **Observation: easy to get high score on public, but that's pointless**
  - e.g. by building good model on training sample - they have similar domains
  - Teams are evaluated on private →no sense to fight for leaderboard positions
- Instead, **crucial to do good job on private sample**
  - However, sizeable differences in two domains expected
  - Meaning model trained on training sample expected to perform worse on private

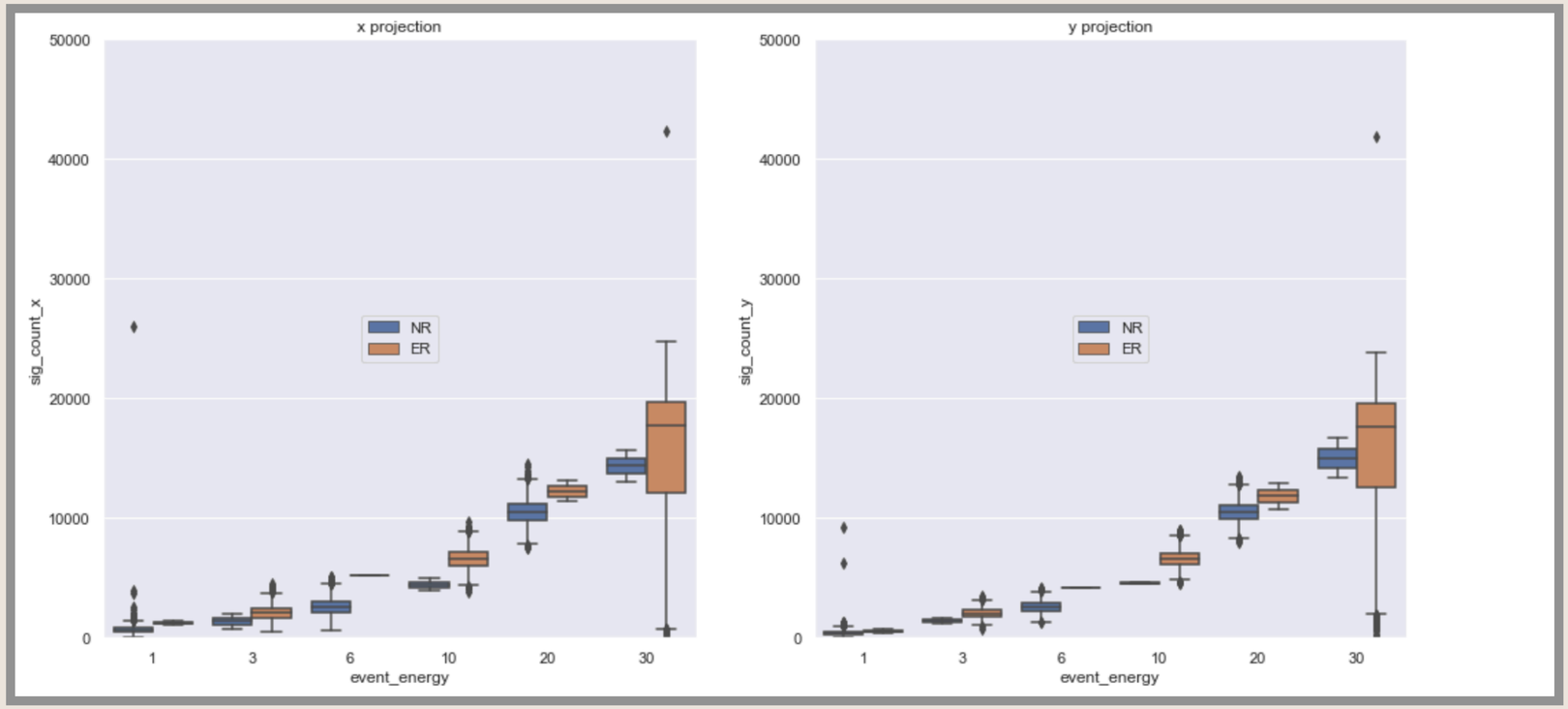
Energy, keV	He	e
1	*	-
3	-	*
6	*	-
10	-	*
20	*	-
30	-	*

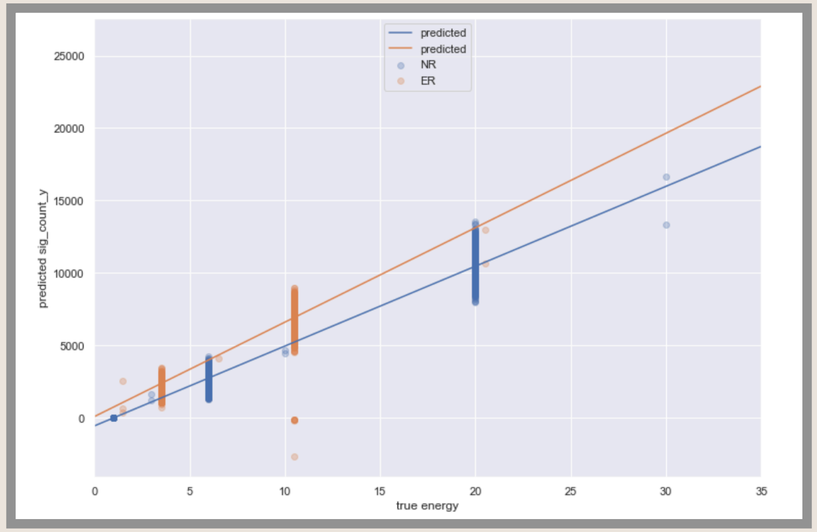
* is training; - is testing

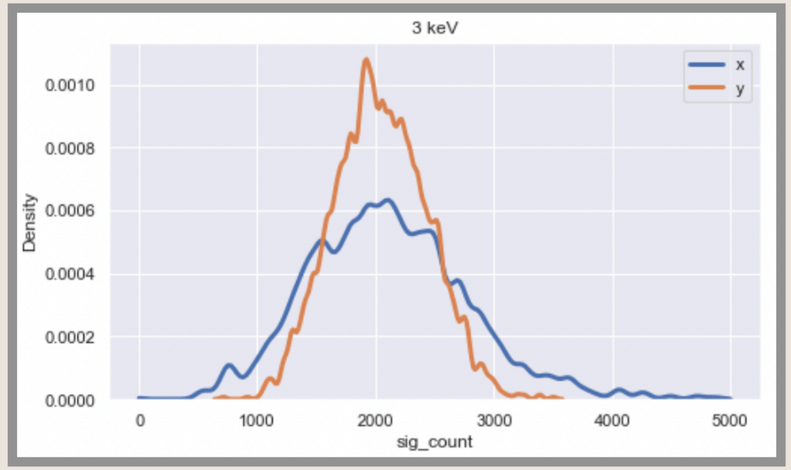
Our solution

Calibration curve

- Initially, we **thought that domain difference isn't that big**
- So **focused firstly on the following idea:**
  - It is clear that `sig_count` should be highly correlated with `true_energy`
  - Can try to build a simple regression on that → calibration curve
  - Using this curve we can label the data
    - Class prediction can be inferred from energy one (recall the table)
- **NB:** we decided to use Y projection for that, since it turned out to provide better resolution at low energies
- **NB:** hereafter use NR curve coefficients for calibration



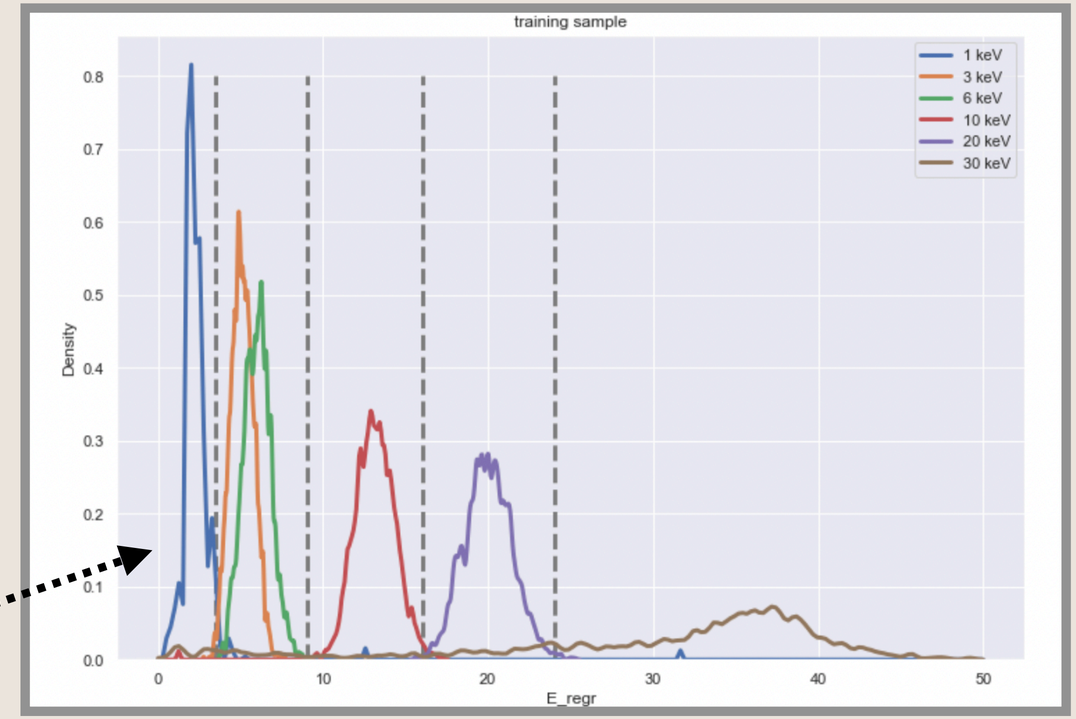


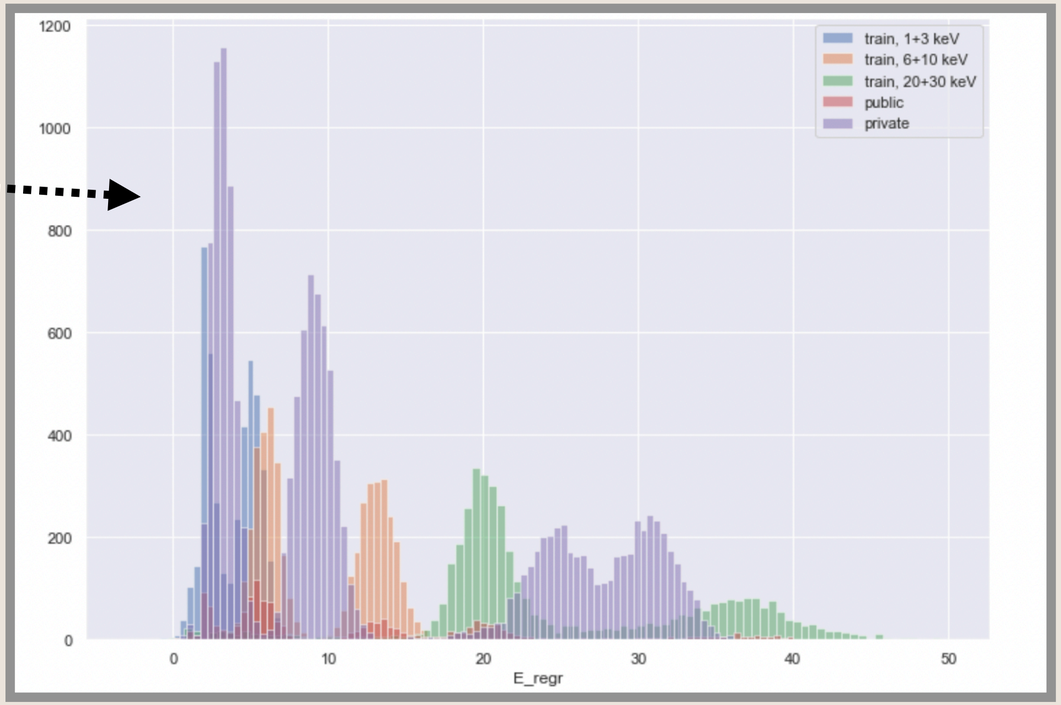


Our solution

From `sig_count` to `E_regr`

- **(Implicitly) we didn't want to predict "soft" energies**
  - But rather round them to those which appear in data sample
  - Makes sense if we're confident in prediction (we were, classes easily separated in `E_regr`)
- **This would require applying a cut on `E_regr`**
  - Kinda easily separated on training sample
  - Basically, no need for ML (unless wanna be super-accurate)
- **Decided to plot `E_regr` distribution for private sample**
- ***horrow***
  - Not only the peaks are shifted (expected)
  - They also heavily overlap: e.g. ER (3) vs NR (6), ER (10) vs NR (20)
  - Any classifier trained on training sample would be literally useless on private
- **This made us rethink the whole strategy**

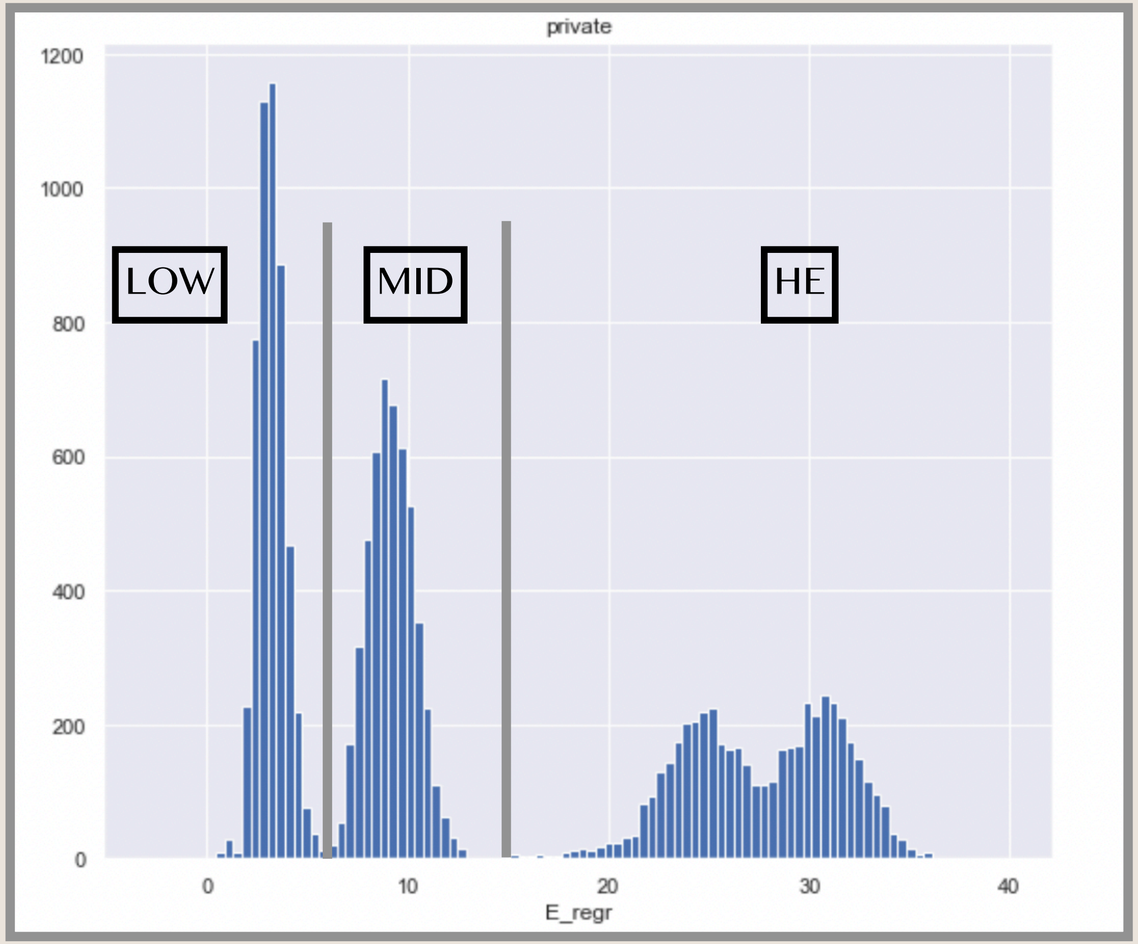




Our solution

Final strategy

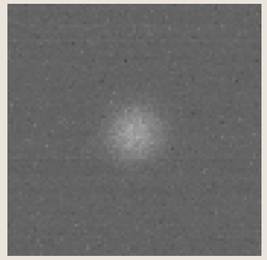
- **Refrain from nicely predicting public and focus entirely on labelling private**
  - Public labelled with the same (not optimal) approach as private
- **Based on `E_regr` define 3 regions of interest:**
  - HE (high energy): `E_regr>=14`
  - MID (middle): `6<=E_regr<14`
  - LOW: `E_regr<6`
- **In each of these regions we have almost pure selection of classes:**
  - HE: ER (20) & NR (30)
  - MID: ER (6) & NR (10)
  - LOW: ER (1) & NR (3)
- **Now, problem is split into 3 mutually exclusive binary classification problems**



Our solution

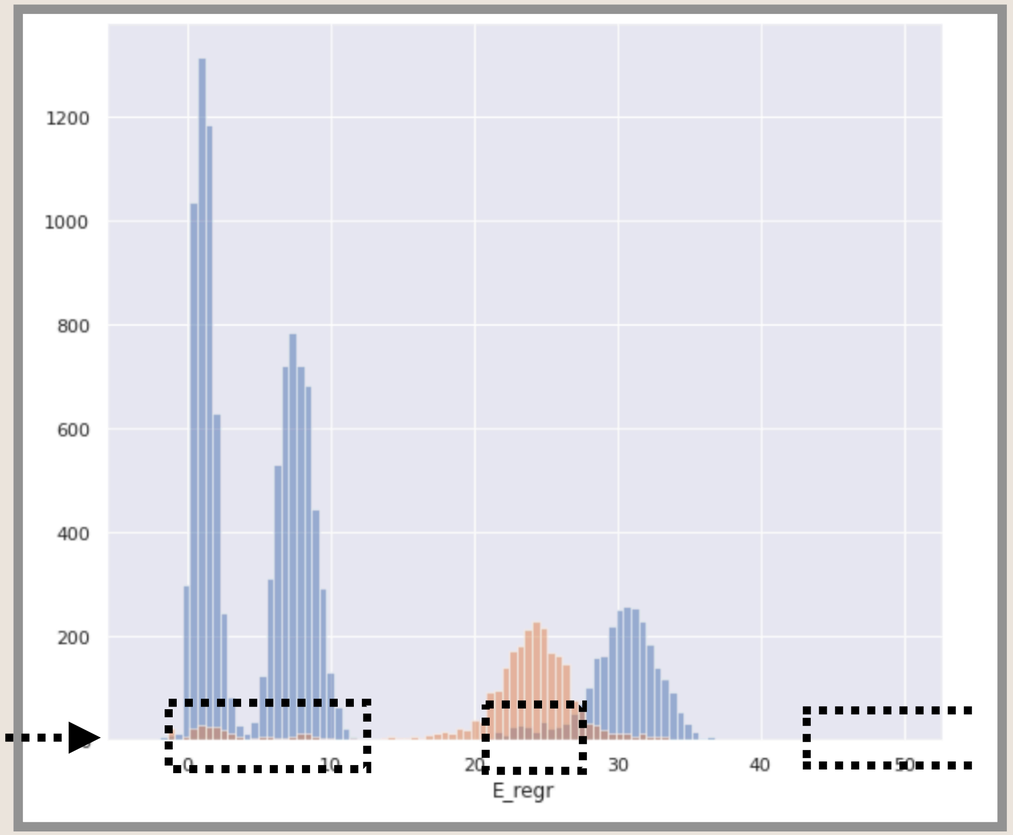
HE region

ER(20)



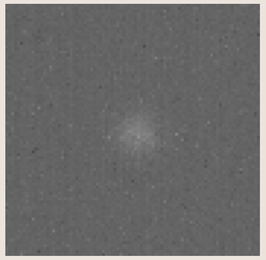
NR(30)

- **Gradient boosting on training dataset**
  - "wiggly classifier" ER(30) vs all
  - Because effectively in this regions we need to separate "wiggly" electrons from "round" nuclea
  - Features: chi2_pvalue_x, chi2_pvalue_y, abs_dmu_x, abs_dmu_y
  - ROC AUC ≈1 on training sample
- **Can afford using training dataset here**
  - "wiggliness" is ~invariant under ER(30) vs NR(20) <—> ER(20) vs NR(30)
- **Validate on 4 test domain samples from training set**
  - Got very confident and correct predictions → looks promising
- **Apply cut > 0.5 to get deterministic predictions**
- **Remaining misID observed → check & force these events into ER(20) category**
- **Overall approach results in expected ~2500 vs ~2500 split**

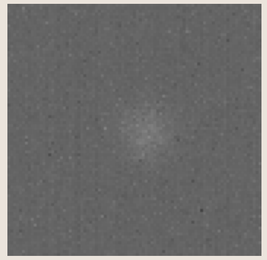


Our solution

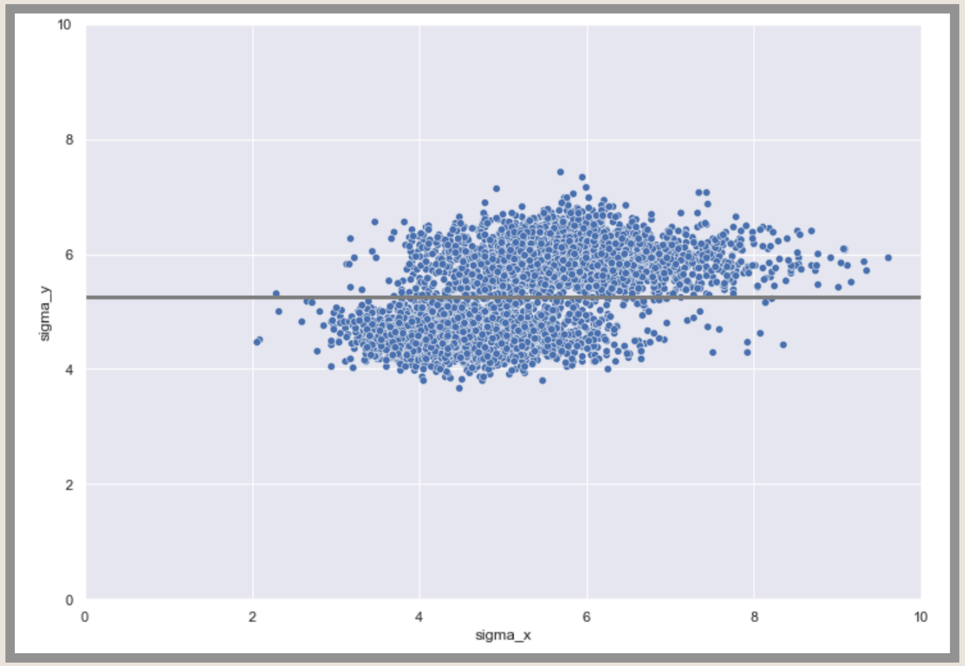
MID region



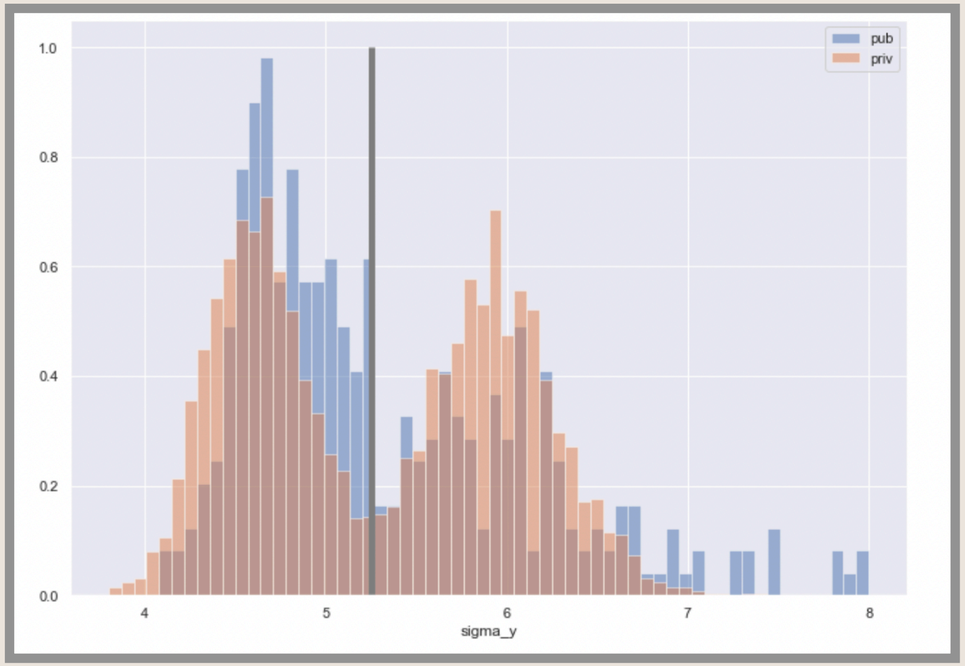
ER(6)



NR(10)

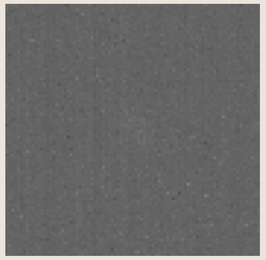


- **Can't use training data as in HE region**
  - Domain shift is significant, ER <—> NR changes distributions dramatically
- **However, turned out that there's a feature where two clearly separated clusters seen**
  - It is σ_y
  - In fact, it makes sense because expect electrons to produce more compact energy deposits
- **Simply go ahead with applying cut $\sigma_y > 5.25$**
- **Validate that it predicts correct labels on 4 private domain samples**
- **Approach results in expected ~2500 vs ~2500 split**

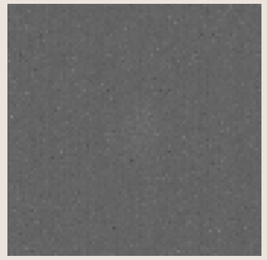


Our solution

LOW region

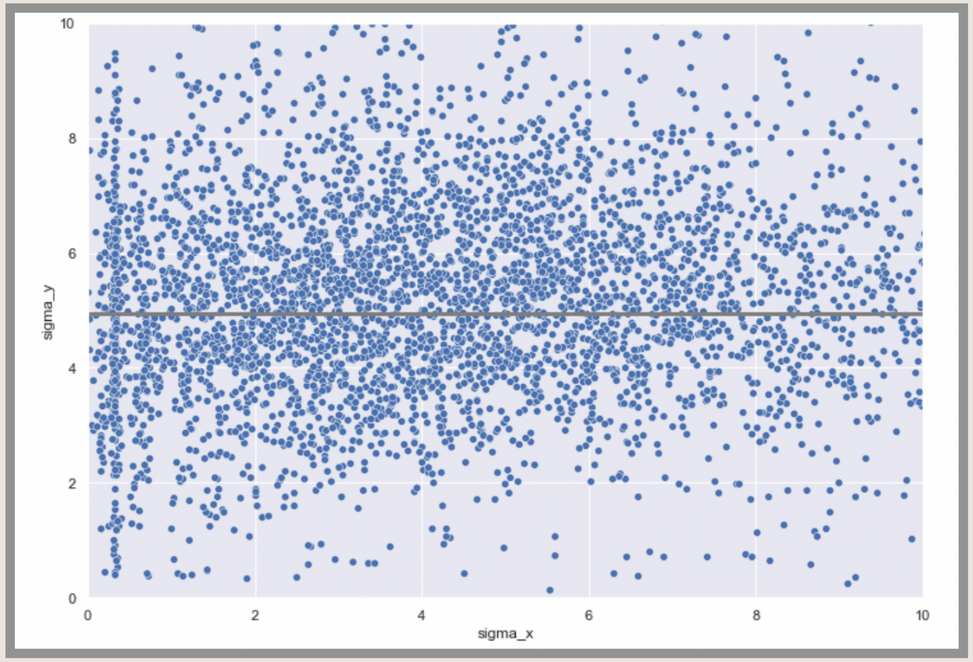


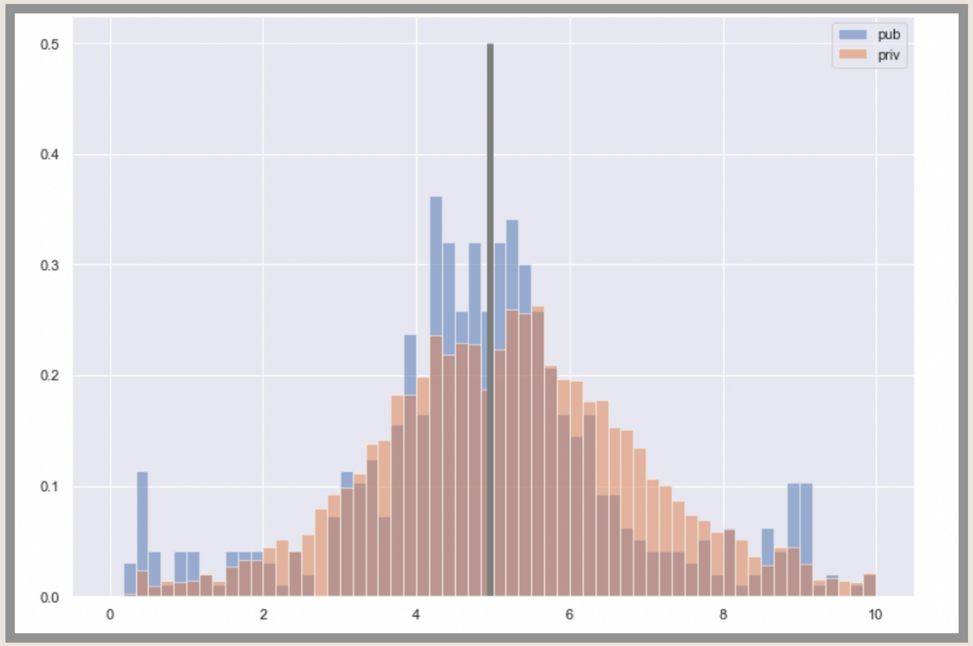
ER(1)



NR(3)

- **The most tough one**
  - Significant overlap of ER(1) and NR(3)
  - Can’t use supervised ML because of domain shift
  - Failed to find "killer feature" as in MID region
- **Decided to pick the most discriminative (by eye) feature and applied cut there**
  - σ_y once again with $\sigma_y > 4.95$
  - 1/5 mispredicted on private domain samples from train
- **Resulted in asymmetric split ~2800 vs ~2200 but that’s OK**
  - Expected since σ_y (NR) > σ_y(ER)
  - However, should be more optimal wrt. forcing balanced option





Our solution

Track 2

- **Approach described above submitted to track 1**
- **But can't use it for track 2**
  - fitting of images takes O(1) hours → can't pass 15 min time limit
- **Well, actually we can if we use surrogate features**
  - `sig_count:` subtract average background count from signal region (per projection)
  - `abs_dmu:` finding bin position with max count (per projection)
  - `sigma:` count number of bins with excess over average bkgr. (per projection)
- **Apply cut on these feature in each region (borders on `sig_count_y` were recalculated) to discriminate btw. corresponding classes**
  - HE: `abs_dmu_y > 2`
  - MID: `sigma_y > 13.5`
  - LOW: `sigma_y > 1.99`
- **Were quite in a rush but managed to make last minute submit before deadline**

Results & summary

- **Track 1** $(-\infty < \text{score} <= 1000)$
  - Public: $\text{AUC} = 0.830,\ \text{MAE} = 2.368,\ \text{total} = -1537.9$
  - Private: $\text{AUC} = 0.878,\ \text{MAE} = 0.283,\ \mathbf{total = 595.49}$
- **Track 2** $(-\infty < \text{score} <= 1000)$
  - Public: $\text{AUC} = 0.687,\ \text{MAE} = 2.982,\ \text{total} = -2295.3$
  - Private: $\text{AUC} = 0.596,\ \text{MAE} = 3.055,\ \text{total} = -2459.04$
- **The code:** https://github.com/depot-hep/idao-2021

	Track 1	Country
1	It doesn't matter	France
2	Veni Vidi Vici	Belarus, Russia
3	SquattingSlavs	Germany, United States
4	DS19	Russia
5	Baobab	Germany, Russia
6	tsubasa	Azerbaijan
7	Shizika	Russia
8	Super Mario Bros	Bangladesh
9	BegInnors	Russia
10	Data Pro	Australia, Hong Kong
11	Sabhya log	India
12	oski	Russia
13	ai.max.roar	Russia
14	Анти материалисты	Russia
15	data o plomo	France

	Track 2	Country
1	Optimization 4 KO	Russia
2	random team	Switzerland
3	!	Russia
4	mn team 2	Korea, Mongolia
5	Alpha Analysts	Malaysia, Russia, Egypt
6	Made As Described Earlier	Russia
7	data siens	Russia
8	Magic City	Russia
9	CHAD DATA SCIENTISTS	Russia
10	QuantumCurious	India
11	Tonatiuh	Russia
12	misclass_classifier	India
13	NotExperts	Russia
14	Getsuga Tenshou	Russia
15	White Material	Russia