03 – Healthy vs Aged Classification from EIS

Chouaib (github.com/ChouaibB)

December 19, 2025

Contents

1. Overview & data choice for healthy vs aged classification

For this notebook I originally planned to use the **NASA PCoE Li-ion Battery Aging Datasets** (https://data.nasa.gov/dataset/li-ion-battery-aging-datasets), which contain full aging trajectories down to clear **end-of-life (EoL)** and are widely used for **SoH / RUL** studies. At the time of this work I could not access that dataset, so I instead reuse the SoH-from-EIS dataset from **02 – SoH from EIS (Rashid et al.)** as a **stand-in** to demonstrate a simple healthy vs aged **classification workflow**.

In real applications, a cell is typically considered to have reached EoL when its capacity (or SoH) drops to around **80%** of nominal. The Rashid SoH-from-EIS dataset I use here only covers SoH levels from **100% down to 80%** in 5% steps, and does **not** include cells degraded below 80% like the NASA PCoE data would. To still illustrate a binary screening task with the data at hand, I define:

- `healthy` = SoH 90%
- `aged` = SoH < 90%

This threshold is therefore **chosen for demonstration**, not as a universal definition of EoL, and should be interpreted as "early screening for cells that have started to age" rather than a hard safety or warranty limit. In practice, I would expect cells aged **below 80% SoH to develop a noticeably different impedance fingerprint/signature** from the SoH ranges considered here (SoH 90% and SoH < 90%), which could **substantially change** both the models and the conclusions. The current notebook should therefore be read as a **workflow demonstration** under limited data, not as a definitive analysis of fully aged cells.

---

2. Dataset and feature pipeline (from 02 – SoH from EIS)

As in
02 – SoH from EIS (Rashid et al.), I use the public dataset:

> Rashid, Muhammad; Faraji-Niri, Mona; Sansom, Jonathan; Sheikh, Muhammad; Widanage, Dhammika; Marco, James (2023),
> **"Dataset for rapid state of health estimation of lithium batteries using EIS and machine learning: Training and validation"**,
> *Data in Brief*, 48, 109157, doi: 10.1016/j.dib.2023.109157.
> Available at: https://data.mendeley.com/datasets/mn9fb7xdx6/3
> Original data: **"DIB_Data"**, Mendeley Data, V3, doi: 10.17632/mn9fb7xdx6.3 (CC0 1.0).

Dataset highlights (same as in 02 – SoH from EIS):

- **25 cylindrical Li-ion cells**, aged from SoH 100% down to 80% in 5% steps (100, 95, 90, 85, 80%).
- At each SoH stage, reference performance tests (capacity / SoH) plus **electrochemical impedance spectroscopy (EIS)** at multiple **SOC** and **temperature** conditions.
- Designed specifically to study **fast SoH estimation from EIS with machine learning**.

In this notebook I **reuse the same data loading, cleaning and feature engineering pipeline**

as in 02 – SoH from EIS:

- I load the raw EIS spectra and associated metadata,
- perform the same basic QC and filtering,
- and construct the same engineered impedance feature table (ohmic / low-frequency resistances, summary |Z| and phase statistics, and sampled spectral points).

For details of the data preparation steps, please refer to
02 – SoH from EIS (Rashid et al.). Here I start directly from the prepared feature table and focus on the **binary classification** of `healthy` vs `aged` cells based on those impedance features.

2.1 Load EIS data

	n_cells	n_eis_tests	soh_levels	soc_levels	temp_levels	\
0	24	360	80, 85, 90, 95, 100	5, 20, 50, 70, 95	15, 25, 35	

	freq_points_per_spectrum
0	61

2.2 Build the impedance feature table for ML

```
Feature table shape: (360, 27)
```

	cell_id	soh_pct	temp_c	soc_pct	R_hf_ohm	R_lf_ohm	delta_R_ohm	\
0	2	95	15	5	0.02995	0.07366	0.04371	
1	2	95	15	20	0.03001	0.07916	0.04915	
2	2	95	15	50	0.02965	0.03573	0.00608	
3	2	95	15	70	0.02960	0.03803	0.00843	
4	2	95	15	95	0.02958	0.04171	0.01213	

	Zmag_mean	Zmag_std	Zmag_min	…	Zmag_f1p0	phase_f1p0	Zmag_f10p0	\
0	0.040826	0.016416	0.024201	…	0.042847	-0.282403	0.031430	
1	0.041919	0.018060	0.024285	…	0.042731	-0.290515	0.031688	
2	0.029907	0.003850	0.023819	…	0.030956	-0.031686	0.029673	
3	0.030187	0.004274	0.023747	…	0.030942	-0.037886	0.029639	
4	0.031920	0.005565	0.023771	…	0.034760	-0.067860	0.030048	

	phase_f10p0	Zmag_f100p0	phase_f100p0	Zmag_f1000p0	phase_f1000p0	\
0	-0.123542	0.027523	-0.077613	0.024205	0.097449	
1	-0.122719	0.027668	-0.079707	0.024291	0.095818	
2	-0.047501	0.026807	-0.059273	0.023819	0.103972	
3	-0.051610	0.026685	-0.058380	0.023747	0.103440	
4	-0.092483	0.026772	-0.062605	0.023772	0.101259	

	Zmag_f10000p0	phase_f10000p0
0	0.043538	0.812199
1	0.043579	0.811200
2	0.043398	0.818643
3	0.043320	0.818540

```
4       0.043262        0.817930

[5 rows x 27 columns]
```

3. SoH-based labels for healthy vs aged classification

In this section I convert the SoH information into a **binary health label** that will be used as the target for the classifier:

- `healthy` = SoH 90%
- `aged` = SoH < 90%

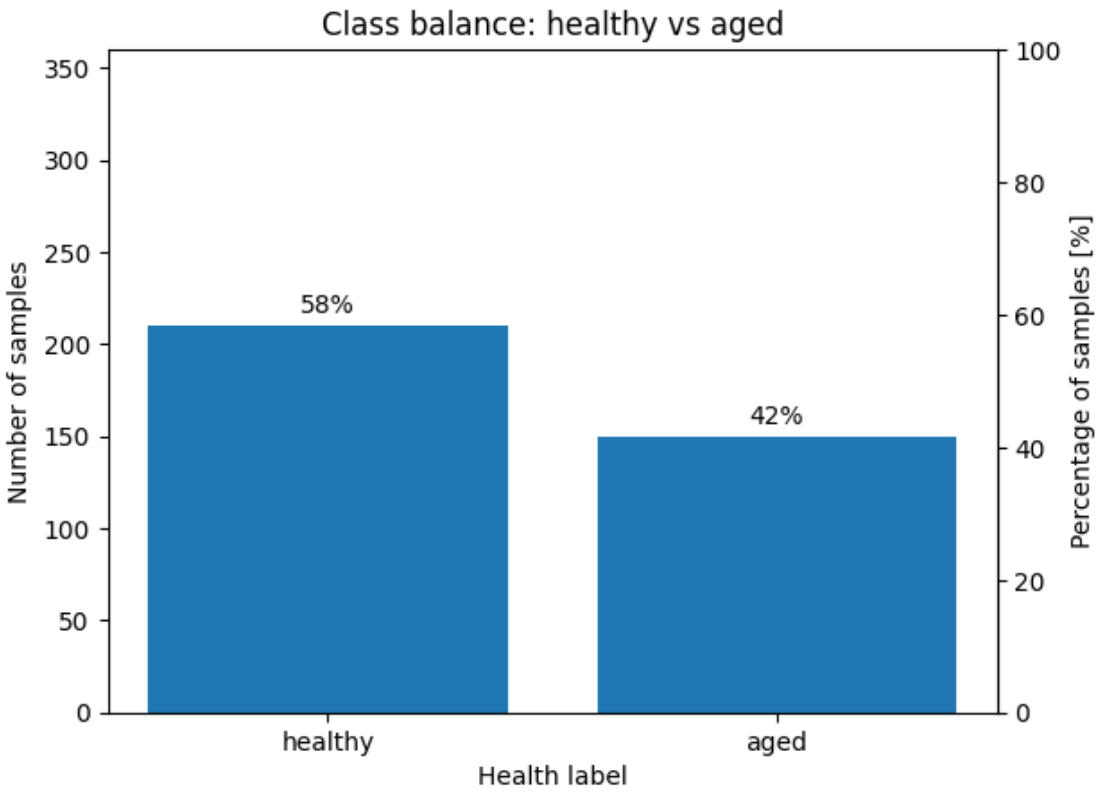
I first add a continuous SoH fraction (`soh_frac`) and then derive the binary `health_label` (and a numeric `target_aged` = 1 for aged, 0 for healthy). I also inspect the resulting **class balance** (counts and percentages) to see how many samples fall into each group before training any models.

3.1 Define SoH-based binary labels

	cell_id	soh_pct	temp_c	soc_pct	health_label
0	2	95	15	5	healthy
1	2	95	15	20	healthy
2	2	95	15	50	healthy
3	2	95	15	70	healthy
4	2	95	15	95	healthy

3.2 Inspect healthy vs aged class balance

	health_label	n_samples	pct
0	healthy	210	58.3
1	aged	150	41.7



4. Modelling: nested CV with logistic regression, SVM and GP

In this section I train three **probabilistic classifiers** on the original engineered impedance features:

1. Regularised logistic regression (`LogisticRegression`)
2. RBF-kernel SVM (`SVC(kernel="rbf", probability=True)`)
3. Gaussian Process Classifier (`GaussianProcessClassifier`)

I use a **nested cross-validation** scheme:

- **Outer loop:** 5-fold grouped splits by `cell_id` to estimate performance variability across different groups of cells.
- **Inner loop:** grouped K-fold CV (here 3 folds) on the outer-training data for hyperparameter tuning via `GridSearchCV`.

All models are wrapped in `Pipeline`s (with `StandardScaler` where needed) and evaluated using **ROC AUC** and **PR AUC** on the outer test folds. The results are collected in a single table for later summary and visualisation.

4.1 Features, target and groups

```
((360, 25), (360,))
```

4.2 Define models and simple hyperparameter grids

4.3 Main models training loop

4.3a Define CV objects and inspect outer-fold class balance

	outer_fold	train_n	train_neg	train_pos	test_n	test_neg	test_pos
0	1	285	180	105	75	30	45
1	2	285	165	120	75	45	30
2	3	285	165	120	75	45	30
3	4	285	150	135	75	60	15
4	5	300	180	120	60	30	30

4.3b Model training loop (using the same CV objects)

```
=== Model: log_reg ===

log_reg | outer CV:   0%|          | 0/5 [00:00<?, ?it/s]

  Outer fold 1/5
    best inner ROC AUC = 0.963 | outer ROC AUC = 0.834, PR AUC = 0.905
  Outer fold 2/5
    best inner ROC AUC = 0.952 | outer ROC AUC = 1.000, PR AUC = 1.000
  Outer fold 3/5
    best inner ROC AUC = 0.962 | outer ROC AUC = 0.996, PR AUC = 0.994
  Outer fold 4/5
    best inner ROC AUC = 0.978 | outer ROC AUC = 0.903, PR AUC = 0.687
  Outer fold 5/5
    best inner ROC AUC = 0.935 | outer ROC AUC = 1.000, PR AUC = 1.000

=== Model: svm_rbf ===

svm_rbf | outer CV:   0%|          | 0/5 [00:00<?, ?it/s]

  Outer fold 1/5
    best inner ROC AUC = 0.964 | outer ROC AUC = 0.689, PR AUC = 0.751
  Outer fold 2/5
    best inner ROC AUC = 0.909 | outer ROC AUC = 1.000, PR AUC = 1.000
  Outer fold 3/5
    best inner ROC AUC = 0.977 | outer ROC AUC = 0.604, PR AUC = 0.695
  Outer fold 4/5
    best inner ROC AUC = 0.996 | outer ROC AUC = 0.899, PR AUC = 0.638
  Outer fold 5/5
    best inner ROC AUC = 0.938 | outer ROC AUC = 1.000, PR AUC = 1.000

=== Model: gp_classifier ===

gp_classifier | outer CV:   0%|          | 0/5 [00:00<?, ?it/s]

  Outer fold 1/5
    best inner ROC AUC = 0.962 | outer ROC AUC = 0.861, PR AUC = 0.908
  Outer fold 2/5
```

```
    best inner ROC AUC = 0.941 | outer ROC AUC = 0.997, PR AUC = 0.995
  Outer fold 3/5
    best inner ROC AUC = 0.963 | outer ROC AUC = 0.649, PR AUC = 0.712
  Outer fold 4/5
    best inner ROC AUC = 0.961 | outer ROC AUC = 0.927, PR AUC = 0.730
  Outer fold 5/5
    best inner ROC AUC = 0.933 | outer ROC AUC = 0.946, PR AUC = 0.940
```

	outer_fold	model	roc_auc	pr_auc	best_inner_roc_auc	best_params	n_test_samples
0	2	log_reg	1.000000	1.000000	0.952222	{'clf__C': 0.1}	75
1	5	log_reg	1.000000	1.000000	0.935370	{'clf__C': 0.1}	60
2	2	svm_rbf	1.000000	1.000000	0.908593	{'clf__C': 1.0, 'clf__gamma': 0.01}	75
3	5	svm_rbf	1.000000	1.000000	0.938210	{'clf__C': 1.0, 'clf__gamma': 0.01}	60
4	2	gp_classifier	0.997037	0.995474	0.940704	{'clf__kernel': RBF(length_scale=10)}	75
5	3	log_reg	0.995556	0.993845	0.962296	{'clf__C': 0.1}	75
6	5	gp_classifier	0.945556	0.939726	0.933457	{'clf__kernel': RBF(length_scale=1)}	60
7	4	gp_classifier	0.926667	0.729529	0.961420	{'clf__kernel': RBF(length_scale=10)}	75
8	4	log_reg	0.903333	0.687388	0.978333	{'clf__C': 1.0}	75
9	4	svm_rbf	0.899444	0.638067	0.995556	{'clf__C': 10.0, 'clf__gamma': 0.01}	75
10	1	gp_classifier	0.860741	0.908004	0.961811	{'clf__kernel': RBF(length_scale=10)}	75
11	1	log_reg	0.834074	0.905359	0.963457	{'clf__C': 100.0}	75
12	1	svm_rbf	0.688889	0.750990	0.964115	{'clf__C': 1.0, 'clf__gamma': 'scale'}	75
13	3	gp_classifier	0.648889	0.712016	0.962741	{'clf__kernel': RBF(length_scale=1)}	75
14	3	svm_rbf	0.604444	0.694608	0.977481	{'clf__C': 0.1, 'clf__gamma': 1.0}	75

5. Results: outer-fold performance

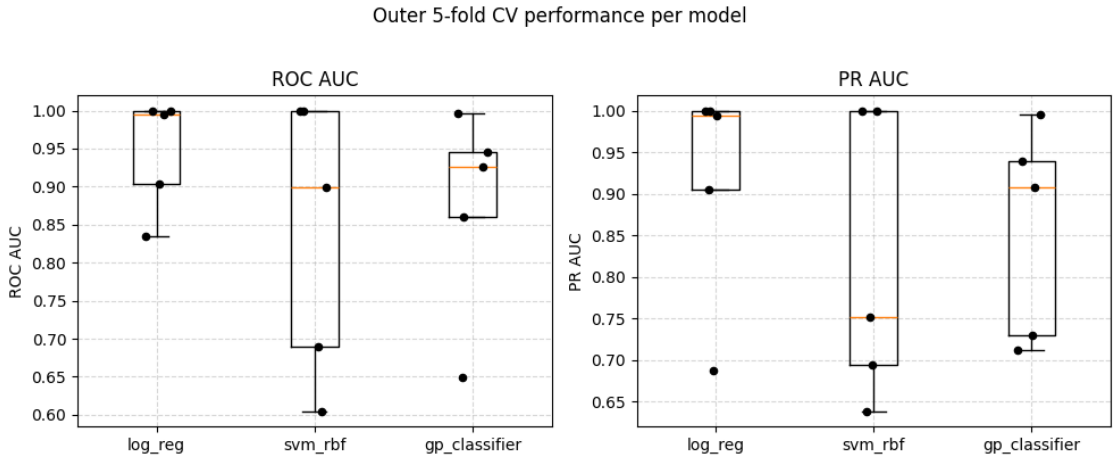
First I summarise the nested CV performance across the 5 outer folds for each model using boxplots of:

- ROC AUC (aged vs healthy)

- PR AUC (precision–recall for the aged class)

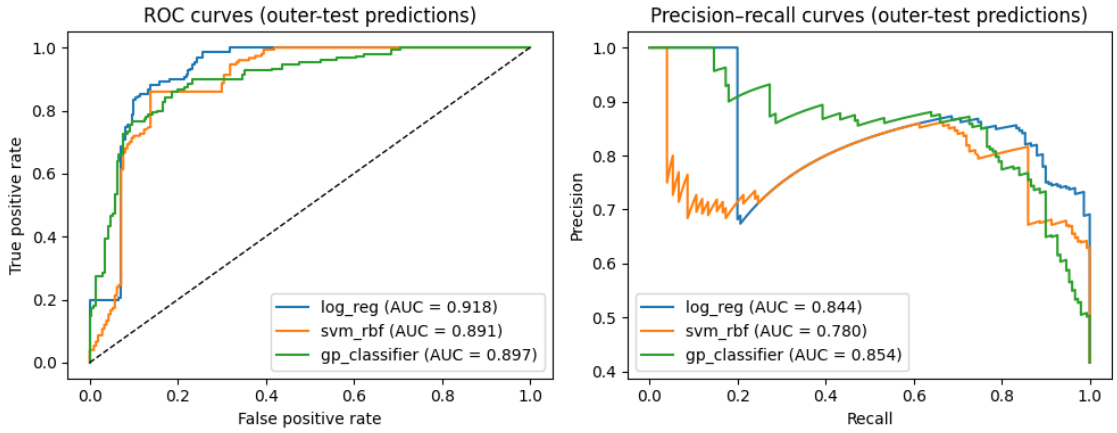
Each point comes from one outer test fold; the boxes show the spread across folds for each model.

5.1 Boxplots of ROC AUC and PR AUC



5.2 ROC and precision–recall curves

To visualise the trade-offs, I build ROC and PR curves for each model by concatenating the predictions from all outer test folds and computing the curves on these cross-validated predictions.



5.3 Interpretation of healthy vs aged classification results

Class balance & splits

- Each outer test fold has **60–75 samples** with both classes present (e.g. test sets range from 30/45 to 60/15 healthy/aged), so ROC and PR metrics are well-defined and there is no obvious class-collapse in any fold.

Overall separability

- All three models achieve **high ROC AUC (~0.84–0.95 on average)** and **PR AUC (~0.82–0.92)** across the 5 outer folds, which means the impedance features carry a strong signal for the `healthy` vs `aged` label.
- The pooled ROC/PR curves show that, for a wide range of thresholds, we can keep the **false positive rate reasonably low** while maintaining good recall for aged cells.

Model comparison

- **Logistic regression**
  - Highest and most stable performance: mean ROC AUC **0.95**, PR AUC **0.92** with relatively small fold-to-fold spread.
  - Several outer folds reach ROC/PR **1.0**, which suggests that in those particular cell splits the classes are almost perfectly linearly separable in the chosen feature space.
  - Given its simplicity, this is a very strong baseline and indicates that a mostly linear decision boundary in impedance features is already very effective for this healthy/aged split.
- **Gaussian Process Classifier**
  - Close runner-up: mean ROC AUC **0.88**, PR AUC **0.86**.
  - Shows more variability across folds (one fold drops to ROC AUC 0.65).
- **RBF SVM**
  - Slightly weaker and more variable: mean ROC AUC **0.84**, PR AUC **0.82**, with one difficult fold around ROC AUC 0.60 and others close to 1.0.
  - The higher variance across folds likely reflects sensitivity to the hyperparameters and to which cells land in train vs test.

Caveats

- The near-perfect scores on some folds are plausible given:
  - relatively small outer test sets (60–75 samples), and
  - a **constructed label** (`SoH 90%` vs `< 90%`) that may align quite cleanly with certain impedance patterns for some cells.
- However, the fact that:
  - the evaluation is **nested**,
  - splits are grouped by `cell_id`, and
  - not all folds achieve 1.0, suggests we are not seeing a trivial leak, but rather a genuinely strong separability in this dataset and threshold choice.

Overall, the results show that impedance-based features in this Rashid et al. dataset are highly informative for the early **healthy vs aged** screening task, with **regularised logistic regression** already providing a very competitive and stable classifier, and SVM / GP offering alternative non-linear and probabilistic views with slightly lower but comparable performance.

6. Feature importance and 2D decision regions

Here I look at **which impedance features matter most** for the healthy vs aged classification, and how the three classifiers behave in a **2D feature plane**.

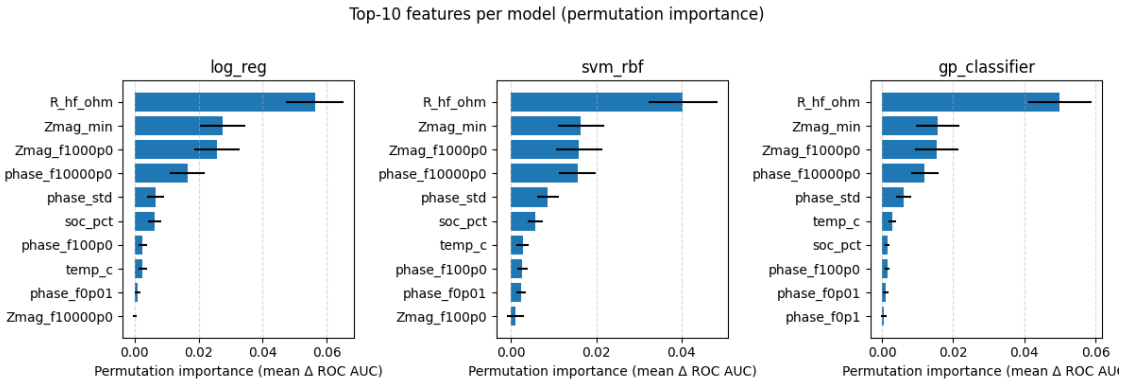
- First, I refit each model once on the **full dataset** (using grouped CV on `cell_id` for hyperparameter tuning) and compute **permutation importance**, using ROC AUC as the scoring metric.
- Then, for each model, I take its **two most important features**, train a small 2D version of the classifier on those features, and visualise the **decision regions / probability maps** together with the data points.

6.1 Final models and permutation-based feature importance

```
=== Final fit on full data: log_reg ===
  Best params: {'clf__C': 0.1}
  Mean inner ROC AUC: 0.960

=== Final fit on full data: svm_rbf ===
  Best params: {'clf__C': 1.0, 'clf__gamma': 0.01}
  Mean inner ROC AUC: 0.946

=== Final fit on full data: gp_classifier ===
  Best params: {'clf__kernel': RBF(length_scale=10)}
  Mean inner ROC AUC: 0.949
```



6.2 Decision regions in a shared 2D feature plane

From the permutation importance plots, all three models consistently rank

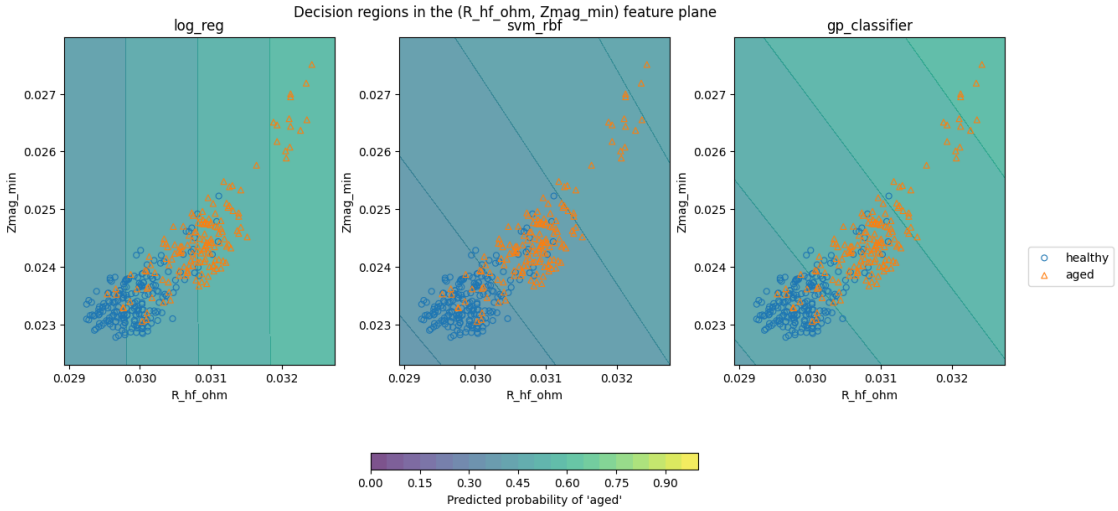
- `R_hf_ohm` (high-frequency / ohmic resistance), and
- `Zmag_min` (minimum impedance magnitude across the spectrum)

as the two most informative features.

To make the decision regions directly comparable, I now:

- fix a **common 2D feature plane** (`R_hf_ohm` vs `Zmag_min`),

- train a 2D version of each final model on these two features only, and
- plot the resulting **probability maps** and data points in that plane.



In this shared (`R_hf_ohm`, `Zmag_min`) plane, all three models agree that cells with **higher ohmic resistance and higher minimum impedance** sit in a region of high probability of being *aged* (yellow, top-right), while low values on both axes are confidently *healthy* (purple, bottom-left). The main difference is how sharply each model transitions through the **overlap band** where healthy and aged points mix: logistic regression uses an almost straight boundary, whereas the SVM and GP give slightly smoother transitions, but they all recover essentially the same separation structure.