Breast-Cancer Attributes Prediction

Authors: | Avi Klings | Shay Morad | Yehonatan Ezra | Lior Zats |

We chose to do the second challenge, predicting clinical attributes of breast cancer, a meaningful and complex real-world challenge. In this short report, we'd like to walk you through our process and to explain how we cleaned the data, explored hidden patterns, and gradually built up a modeling pipeline we were proud of.

Data Cleaning and Pre-Processing

Our first step was to prepare the raw data. We focused on cleaning and simplifying the features through manual checks, correlation analysis, and compact encodings.

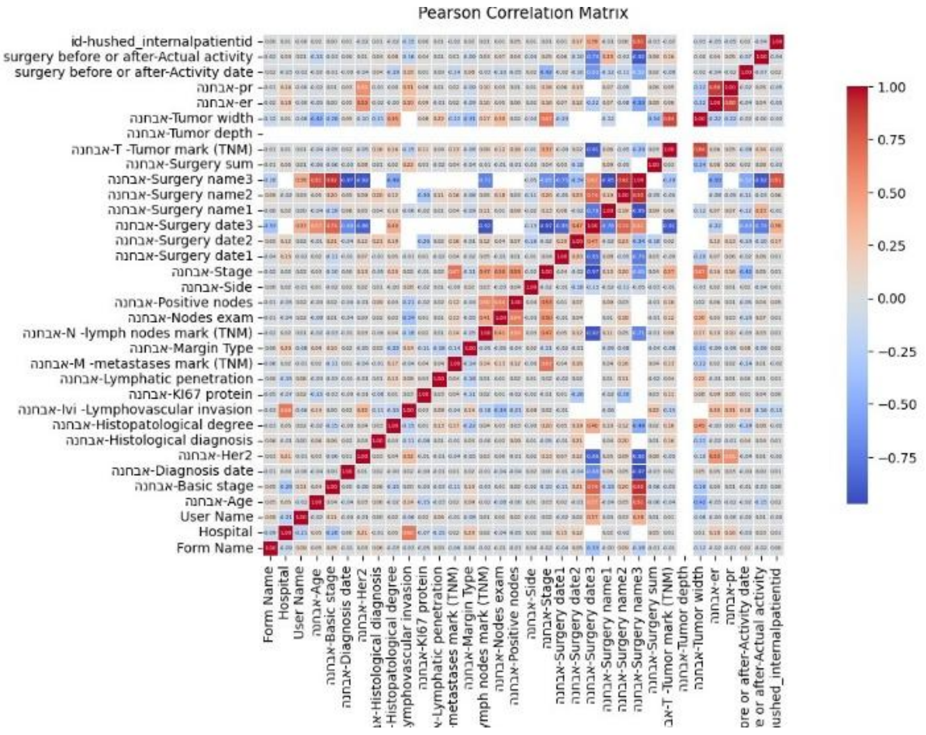
Initial Manual Screening:

At the outset we attempted a column-by-column inspection of the 34 raw features provided in train.feats.csv. The goal was to catalogue data types, admissible values, and obvious anomalies. While this manual pass helped us familiarize ourselves with the dataset, it quickly became tedious and error-prone - many variables contain dozens of heterogeneous string representations and many more rows made spreadsheet - style checks impractical.

Automated Correlation-Based Screening:

To streamline the feature set we transformed every column into a provisional numeric form timestamps for dates, ordinal codes for categorical strings, and 0/1 for Booleans - then visualized the resulting Pearson - correlation matrix. We used this correlation analysis to remove columns that were not relevant for modelling, replacing exhaustive manual comparisons with a fast, objective filter.

In the graph below you can see our Pearson correlation Matrix:



Embedding of High-Cardinality Categorical:
After finishing the correlation step, we addressed the few columns with dozens of unique values. Instead of sprawling one-hot vectors, we learned compact 8–16-dimension embeddings that summarize each category's clinical signal while keeping the feature matrix dense. This was used for fields that made sense to apply this to, such as the types of surgeries a patient went to and so on of which we do not want to have an order ratio between the values in such columns.

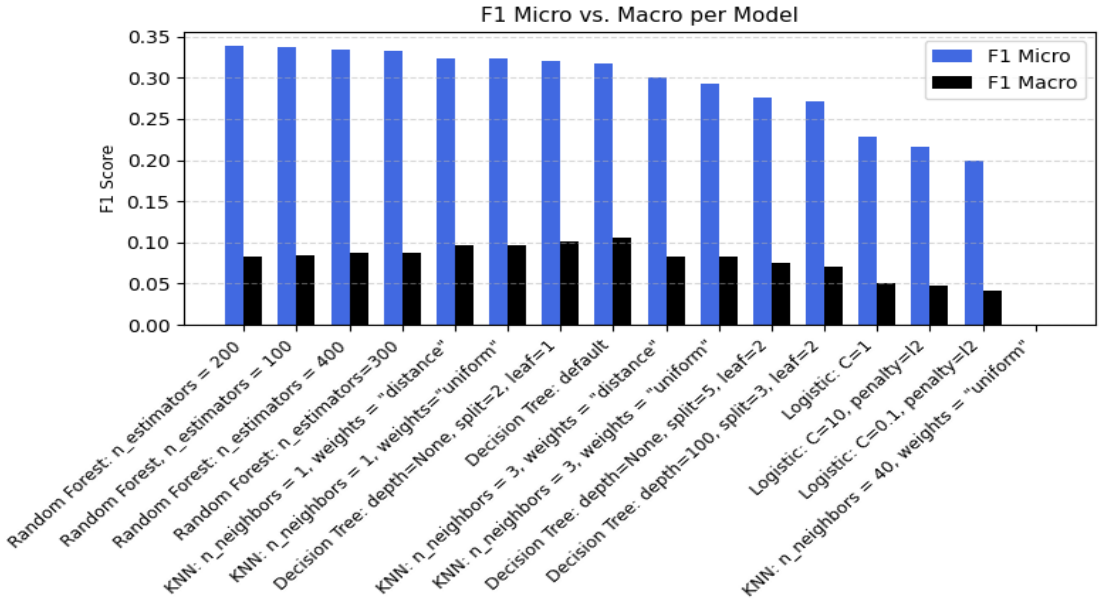
Missing - Value Imputation:
Numeric gaps were filled with each column's median, and missing categorical received an explicit default/missing token. We also quantized values based on similar values such as "pos", "+", "100%pos" and such for a common value.

Part 1 - Predicting Metastases

This task required predicting a **multi-label, multi-class** output: each patient could have zero to three metastasis sites. We framed the problem as a set of binary classification tasks (one-vs-rest), training a separate model per metastasis site.
We evaluated three main model families with various hyperparameters: Random Forest, Decision Tree, and K-Nearest Neighbors. Each model type was tested with multiple configurations to explore its sensitivity and generalization capacity.
After exhaustively grid-searching every hyper, we found that the deeper, tree-rich Random Forest consistently delivered the highest score on F1-micro (class-frequency-weighted) and almost best score on F1-macro (class-balanced) metrics. As illustrated in the accompanying bar chart, its margin over the next-best alternatives was clear, so we selected this Random Forest configuration as our final model for predicting metastasis sites.

Step	**Choice**	**Rationale**
Encoder	custom LabelEncoder → 11 binary targets	0–3 sites per visit.
Split	75 / 25 stratified split (patient-time duplicates removed)	reproducible with seed.
Model sweep	k-NN, Logistic, Ridge, Decision-Tree, Random-Forest	baseline comparison and parameters tuning
Best so far	**RandomForest (200 trees)**	robust, no per-label tuning required.

In the graph below we can see the results:



Part 2 – Tumor-size Regression

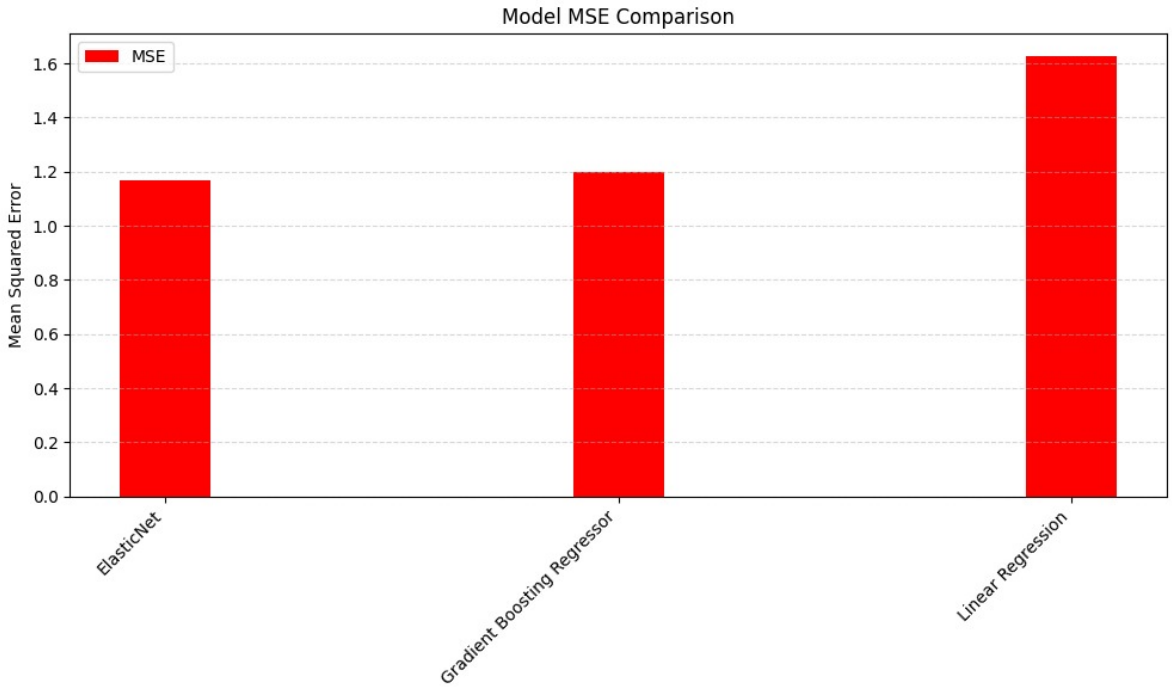
Cleaning the data (similar to part 1)

- kept every numeric variable that might hint at size (T-stage, KI-67, node counts, etc.)
- filled in missing numbers with the median of the column
- clipped any later negative predictions to 0 mm (a tumour cannot be smaller than nothing).

Tested three off-the-shelf models

Model	How it works	Dev-set error
Linear Regression	Straight line through the data	1.6277
ElasticNet	Line + L1/L2 Regularization	1.2195
Gradient Boosting Regression	Many decision trees added together by boosting	1.1958

*Mean-squared-error in mm², averaged over five random splits.



Why we kept Gradient-Boosting

- clearly the lowest error without heavy tuning
- copes well with a mix of numeric, ordinal and one-hot features
- runs fast enough for a hackathon submission.

This task required predicting a **multi-label, multi-class** output: each patient could have zero to three metastasis sites. We framed the problem as a set of binary classification tasks (one-vs-rest), training a separate model per metastasis site.
We evaluated three main model families with various hyperparameters: Random Forest, Decision Tree, and K-Nearest Neighbors. Each model type was tested with multiple configurations to explore its sensitivity and generalization capacity.

Part 3 – Unsupervised Learning

Goal: look for patterns in train.feats.csv that doctors might find interesting.

Principal-component analysis (PCA)

- first 9 components already explain ≈ 50 % of the total variance → engineered features do capture new information.
- biggest axis separates “large, surgically aggressive cases” from “small, early-stage cases”.

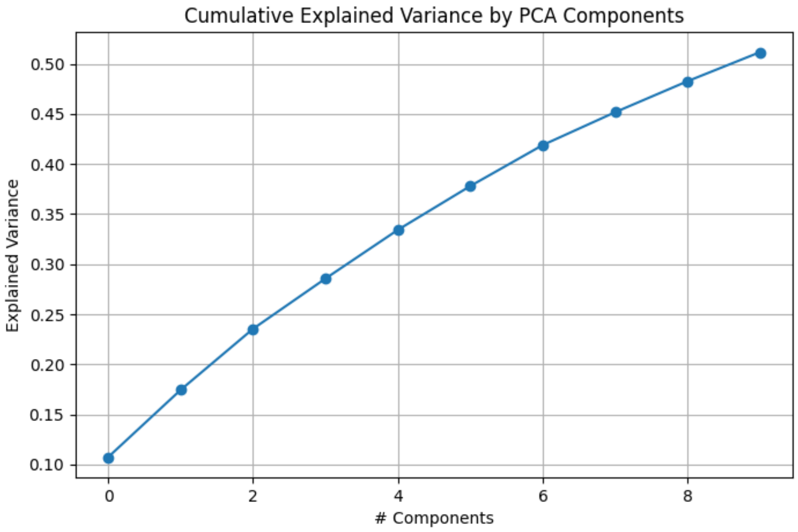
K-means clustering (k = 5) on the PCA space

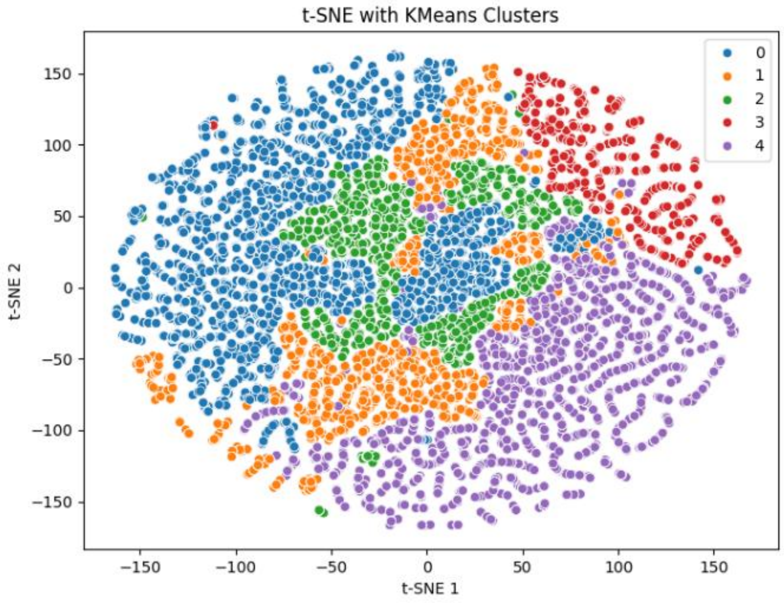
- three big, partly overlapping clusters trace a smooth severity gradient.
- one tight cluster is dominated by triple-negative tumors (ER-, PR-, HER2-).
- another cluster groups most post-mastectomy visits.

t-SNE visualization

- confirms that those five clusters are not random blobs: similar visits sit close, dissimilar ones are far.
- silhouette score ≈ 0. 1177 – modest but typical for high-dimensional clinical data.

Take-away: even with basic engineering and no labels, the data naturally organizes along biologically sensible dimensions (size, aggressiveness, subtype). These insights can guide future feature design and help clinicians spot outlier cases quickly.


Cumulative Explained Variance by PCA Components
Explained Variance
# Components


t-SNE with KMeans Clusters
● 0
● 1
● 2
● 3
● 4
t-SNE 2
t-SNE 1