SKIN CANCER CLASSIFICATION

Introduction:

The project involves analysing a dermatology dataset containing images of skin lesions, represented through pixel values in a CSV format (`hmnist_28_28_RGB.csv`). Another accompanying dataset (`HAM10000_metadata.csv`) provides metadata for these images, including information about the patients' sex and the localization of the lesions. The primary goal is to develop a predictive model that can accurately classify skin lesions based on these images and metadata.

Preprocessing:

The preprocessing phase was designed to transform the raw data into a format suitable for machine learning algorithms, focusing on normalisation, dimensionality reduction, and categorical data encoding:

Normalisation and PCA: Given the high dimensionality of the image data (each image represented by 2352-pixel values), PCA was applied after standardising the pixel values. Standardisation (using StandardScaler) ensured that each feature contributed equally to the analysis, preventing features with larger scales from dominating. PCA was then used to reduce the dataset to a manageable size while retaining 95% of the variance, striking a balance between computational efficiency and information retention.

Metadata Encoding: The patient metadata underwent two significant transformations. The binary encoding of the sex column transformed textual descriptions into a binary format (0 for male, 1 for female), simplifying model interpretation. For the localization of lesions, one-hot encoding expanded this categorical variable into multiple binary columns, each representing the presence or absence of lesions in specific body locations. This method allows models to discern patterns across different lesion locations without assuming an ordinal relationship between them.

Most common category: We have very few examples of missing values that is why we thought that putting by hand or giving the mean values to the missing values will just give us noise, taking out the good quality of the data.

Imbalance Analysis and Mitigation:

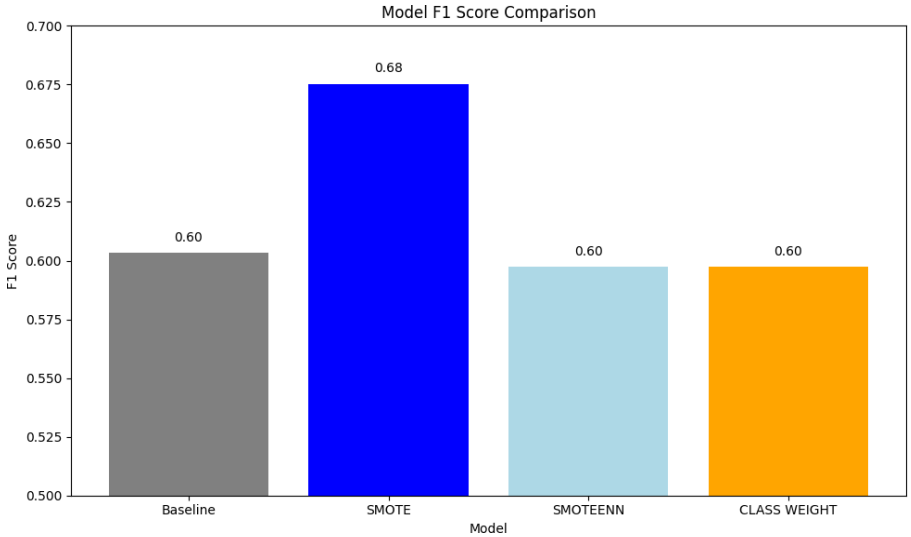
Class imbalance was identified as a potential challenge, with some classes of lesions significantly outnumbering others. This imbalance can lead to biassed models that preferentially predict the majority class, potentially overlooking critical diagnostic categories.

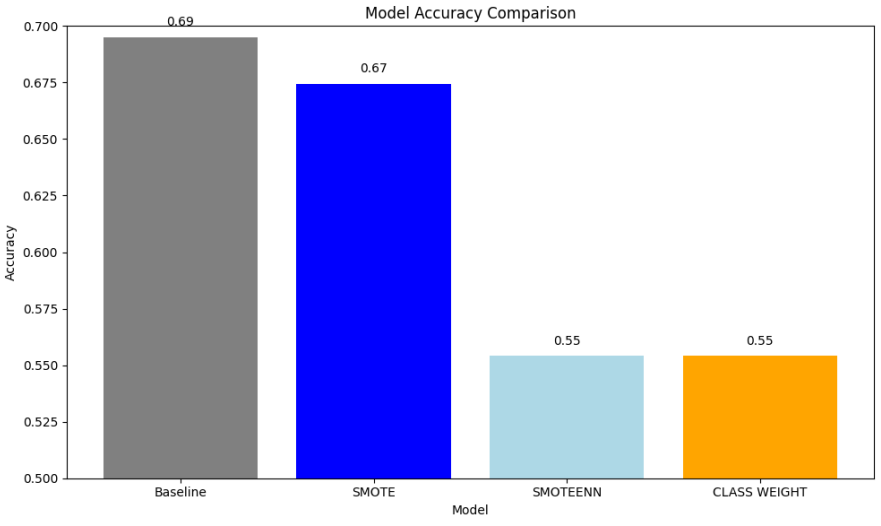
To counteract this, two strategies were used:

SMOTE *(Synthetic Minority Over-sampling Technique)*: This approach synthesises new examples from the minority classes, aiming to balance the class distribution. By interpolating between existing instances, SMOTE attempts to introduce meaningful variability into the training set, enabling models to learn more generalizable patterns across all classes.

SMOTEENN *(SMOTE + Edited Nearest Neighbours)*: Combining SMOTE's oversampling capabilities with the cleaning effect of Edited Nearest Neighbours, SMOTEENN aims to refine the oversampled dataset by removing synthetic examples that are too close to the class boundary. This method seeks to enhance class separability and reduce the noise introduced by oversampling.

We have experienced a significant improvement in the model performance over the baseline where it addressed imbalance data increasing it by 8%. This outcome suggests that SMOTE is the right model to capture the patterns of the data effectively.





Cleaning Process with ENN: ENN removes samples that do not agree well with their neighbours. If class 4 had a lot of overlap or was closely surrounded by samples from other classes, ENN might have removed a significant number of the synthesised samples for class 4, believing them to be noise or mislabelled.

```
Class Distribution after SMOTE:
 label
4    5367
2    5367
0    5367
3    5367
6    5367
1    5367
5    5367
Name: count, dtype: int64
```

```
Class Distribution after SMOTEENN:
 label
3    5367
0    5361
5    5361
1    5356
2    5268
6    5235
4    2820
Name: count, dtype: int64
```

Model Comparison and Hyperparameter Tuning:

Several models were trained and evaluated, including:

- MLPClassifier (Multi-layer Perceptron)

The MLPClassifier trained without smote demonstrates moderate, performance across the board with particularly high f1 score (0.88) in class 4. This trend will later be maintained throughout other models, suggesting that this class is intrinsically easier to classify with high confidence (as seen in the support). Upon training using SMOTE however, this model does not reflect the performance improvement of the random forest used for testing. (Images: No SMOTE, SMOTE)

Evaluation of MLPClassifier:

	precision	recall	f1-score	support
0	0.36	0.42	0.38	60
1	0.42	0.42	0.42	97
2	0.47	0.42	0.44	224
3	0.38	0.34	0.36	29
4	0.87	0.88	0.88	1336
5	0.36	0.57	0.44	21
6	0.36	0.35	0.36	223
accuracy			0.72	1990
macro avg	0.46	0.49	0.47	1990
weighted avg	0.72	0.72	0.72	1990

Evaluation of MLPClassifier:

	precision	recall	f1-score	support
0	0.27	0.20	0.23	69
1	0.40	0.52	0.45	93
2	0.39	0.34	0.37	228
3	0.25	0.21	0.23	28
4	0.84	0.82	0.83	1338
5	0.40	0.48	0.43	21
6	0.30	0.35	0.32	226
accuracy			0.67	2003
macro avg	0.41	0.42	0.41	2003
weighted avg	0.67	0.67	0.67	2003

- Logistic Regression

Logistic Regression also shows commendable performance, particularly in its accuracy (0.74) and weighted average recall (0.74), achieving a 0.72 f1-score, almost matching the leading model, Gradient Boosting. This model's strength in handling class 4, suggests, as we mentioned before, that this class seems to be specially simple. The high performance of this model suggests that there is a strong linear relationship in this classification problem. Likewise, the SMOTE version seems to perform significantly worse across the board.

Evaluation of Logistic Regression:

	precision	recall	f1-score	support
0	0.39	0.28	0.33	60
1	0.54	0.54	0.54	97
2	0.55	0.47	0.51	224
3	0.29	0.14	0.19	29
4	0.82	0.93	0.87	1336
5	0.47	0.33	0.39	21
6	0.43	0.25	0.31	223
accuracy			0.74	1990
macro avg	0.50	0.42	0.45	1990
weighted avg	0.71	0.74	0.72	1990

Evaluation of Logistic Regression:

	precision	recall	f1-score	support
0	0.15	0.33	0.21	69
1	0.30	0.48	0.37	93
2	0.35	0.45	0.39	228
3	0.05	0.25	0.09	28
4	0.92	0.55	0.68	1338
5	0.17	0.62	0.27	21
6	0.27	0.49	0.34	226
accuracy			0.51	2003
macro avg	0.32	0.45	0.34	2003
weighted avg	0.70	0.51	0.57	2003

- K-Nearest Neighbours

K-Nearest Neighbors (KNN) exhibits weaker performance overall. KNN, with low averages, shows high inconsistency across class prediction capability which might be due to its sensitivity to the local data structure and noise in the dataset.

Evaluation of K-Nearest Neighbors:

	precision	recall	f1-score	support
0	0.27	0.27	0.27	60
1	0.31	0.41	0.36	97
2	0.39	0.41	0.40	224
3	0.10	0.03	0.05	29
4	0.83	0.92	0.87	1336
5	0.50	0.10	0.16	21
6	0.44	0.16	0.23	223
accuracy			0.71	1990
macro avg	0.41	0.33	0.33	1990
weighted avg	0.68	0.71	0.69	1990

Evaluation of K-Nearest Neighbors:

	precision	recall	f1-score	support
0	0.26	0.36	0.30	69
1	0.25	0.45	0.32	93
2	0.29	0.47	0.36	228
3	0.07	0.25	0.11	28
4	0.93	0.58	0.72	1338
5	0.06	0.38	0.11	21
6	0.27	0.38	0.32	226
accuracy			0.53	2003
macro avg	0.31	0.41	0.32	2003
weighted avg	0.71	0.53	0.59	2003

- Random Forest

In comparison, random forest shows mild performance all across the board with specially high recall on class 4, which again suggests class 4 instances are hard to miss. This model doesn't excel particularly on overall results, with a weighted average of f1 of

0.68 both with and without smote, challenging our findings on the SMOTE analysis, however, the accuracy seems to go down when using SMOTE.

Evaluation of Random Forest:

	precision	recall	f1-score	support
0	0.38	0.05	0.09	60
1	0.59	0.28	0.38	97
2	0.58	0.37	0.45	224
3	0.00	0.00	0.00	29
4	0.76	0.98	0.86	1336
5	0.00	0.00	0.00	21
6	0.55	0.17	0.26	223
accuracy			0.73	1990
macro avg	0.41	0.26	0.29	1990
weighted avg	0.68	0.73	0.68	1990

Evaluation of Random Forest:

	precision	recall	f1-score	support
0	0.28	0.30	0.29	69
1	0.41	0.56	0.47	93
2	0.46	0.33	0.38	228
3	1.00	0.04	0.07	28
4	0.84	0.82	0.83	1338
5	0.46	0.29	0.35	21
6	0.32	0.45	0.37	226
accuracy			0.67	2003
macro avg	0.54	0.40	0.40	2003
weighted avg	0.70	0.67	0.68	2003

- Gradient Boosting

Gradient Boosting stands out as the top-performing model across the board, achieving the highest overall accuracy at 0.75. This model also maintains a good balance between precision (0.71) and recall (0.75), scoring a 0.72 overall f1-score in the weighted averages, indicating effective handling of both false positives and false negatives. At this point, it doesn't make much sense to keep adding the SMOTE results, but I feel bad for it so there they are, shameful, as always.

Evaluation of Gradient Boosting:

	precision	recall	f1-score	support
0	0.49	0.30	0.37	60
1	0.56	0.46	0.51	97
2	0.55	0.42	0.47	224
3	0.20	0.03	0.06	29
4	0.82	0.95	0.88	1336
5	0.25	0.14	0.18	21
6	0.47	0.29	0.36	223
accuracy			0.75	1990
macro avg	0.48	0.37	0.40	1990
weighted avg	0.71	0.75	0.72	1990

Evaluation of Gradient Boosting:

	precision	recall	f1-score	support
0	0.29	0.46	0.36	69
1	0.35	0.58	0.43	93
2	0.35	0.38	0.37	228
3	0.20	0.21	0.21	28
4	0.93	0.67	0.78	1338
5	0.23	0.43	0.30	21
6	0.29	0.59	0.39	226
accuracy			0.61	2003
macro avg	0.38	0.48	0.40	2003
weighted avg	0.73	0.61	0.65	2003

- AdaBoost

A boosting algorithm that focuses on improving the performance of weak classifiers, trains multiple weak classifiers sequentially, assigning higher weight to instances misclassified by previous iterations. Effective to improve the classification accuracy, on imbalanced data.

Evaluation of AdaBoost:

	precision	recall	f1-score	support
0	0.26	0.33	0.29	60
1	0.38	0.33	0.35	97
2	0.44	0.38	0.41	224
3	0.13	0.07	0.09	29
4	0.85	0.79	0.82	1336
5	0.10	0.24	0.14	21
6	0.28	0.43	0.34	223
accuracy			0.65	1990
macro avg	0.35	0.37	0.35	1990
weighted avg	0.68	0.65	0.66	1990

Evaluation of AdaBoost:

	precision	recall	f1-score	support
0	0.20	0.46	0.28	69
1	0.30	0.45	0.36	93
2	0.34	0.25	0.29	228
3	0.12	0.25	0.16	28
4	0.91	0.53	0.67	1338
5	0.26	0.52	0.34	21
6	0.24	0.69	0.36	226
accuracy			0.51	2003
macro avg	0.34	0.45	0.35	2003
weighted avg	0.70	0.51	0.56	2003

Bagging shows an extreme case with the highest precision (1.00) for class 0 but with very low recall (0.05), indicating that while it is highly accurate when it predicts class 0, it rarely does so, this could be due to a weird fit, but it could be particularly useful in scenarios where false positives are much more detrimental than false negatives. However, its overall balanced weighted average performance shows it has merits, particularly in environments where reliability across multiple classes is less critical than precision in specific classes. It also hardly misses any class 4 (as shown by its 0.99 recall).

Evaluation of Bagging:

	precision	recall	f1-score	support
0	1.00	0.05	0.10	60
1	0.67	0.21	0.31	97
2	0.56	0.29	0.38	224
3	0.00	0.00	0.00	29
4	0.74	0.99	0.84	1336
5	0.00	0.00	0.00	21
6	0.59	0.13	0.21	223
accuracy			0.72	1990
macro avg	0.51	0.24	0.26	1990
weighted avg	0.69	0.72	0.65	1990

Evaluation of Bagging:

	precision	recall	f1-score	support
0	0.37	0.33	0.35	69
1	0.41	0.61	0.49	93
2	0.45	0.35	0.39	228
3	0.75	0.11	0.19	28
4	0.87	0.78	0.82	1338
5	0.60	0.43	0.50	21
6	0.32	0.56	0.41	226
accuracy			0.67	2003
macro avg	0.54	0.45	0.45	2003
weighted avg	0.71	0.67	0.68	2003

Across all models, performance on class 4 is generally strong, suggesting that features related to this class might be more distinctly defined or captured in the dataset than those for other classes.

Results and Evaluation:

The performance of individual models was evaluated using classification metrics such as precision, recall, and F1-score. Despite attempts to balance the class distribution, the Random Forest model trained without explicit imbalance handling techniques yielded the best results. This outcome underscores the Random Forest's inherent robustness to class imbalance.

A notable observation was that even though SMOTE had a significant enhancement of the model performance, it was not enough to see much change in the ensemble models. So this could indicate that the dataset's specific characteristics and the models' inherent capabilities might already provide a strong baseline that is difficult to improve upon through oversampling methods.

In our case logistic regression, random forest, and gradient boosting are our best models so we use a Voting Classifier with voting='soft' and voting='hard' (majority voting instead of averaging probabilities) .

voting → soft

Evaluation of Voting Classifier with Pre-trained Models:

	precision	recall	f1-score	support
0	0.46	0.37	0.41	60
1	0.53	0.48	0.51	97
2	0.52	0.42	0.46	224
3	0.67	0.14	0.23	29
4	0.84	0.96	0.89	1336
5	0.62	0.24	0.34	21
6	0.52	0.30	0.39	223
accuracy			0.76	1990
macro avg	0.59	0.42	0.46	1990
weighted avg	0.74	0.76	0.74	1990

voting → hard

Evaluation of Voting Classifier with Pre-trained Models:

	precision	recall	f1-score	support
0	0.45	0.33	0.38	60
1	0.50	0.47	0.49	97
2	0.55	0.47	0.51	224
3	0.33	0.03	0.06	29
4	0.82	0.97	0.89	1336
5	0.50	0.05	0.09	21
6	0.63	0.20	0.30	223
accuracy			0.76	1990
macro avg	0.54	0.36	0.39	1990
weighted avg	0.73	0.76	0.72	1990

Interesting Findings and Patterns:

The ensemble method, specifically a Voting Classifier combining K-Nearest Neighbours, Random Forest, MLP Classifier, and Gradient Boosting, was used in two configurations: soft voting and hard voting. This approach aimed to leverage the strengths of individual models by aggregating their predictions. The evaluation of the Voting Classifier suggested that aggregating multiple models' predictions could improve the overall performance, showcasing the power of ensemble learning in achieving more accurate and robust predictions.

The project's findings highlight the complexity of predictive modelling in medical imagery and underscore the potential of machine learning techniques in improving diagnostic processes.

We found SMOTE to consistently damage our predictive capability in almost all models and all across the board, with little exception suggesting that even if sometimes it might seem to work, it is very likely just a fluke.