CS135: Project B Report

Will Cusato and Constantine Moseley

August 13, 2026

Problem 1

1A: Preprocessing Description

Preprocessing of the data was instilled to make all entries binary. This is primarily seen in the 'background_father,' 'background_mother,' and 'region' features in the raw data, as these are not binary features. Every entry for these fields is represented by a boolean feature of whether that entry was present in the data. Because the training data is larger than test data, there are some entries in the aforementioned fields that are present in the training data but not the test data. Because of this, there were only 53 features in the test set, opposed to 57 in the training set. These missing features were added to the test set and filled with false values for every patient. This was done to ensure that the feature representation for each patient was identical. Additionally, some of the training set pictures were augmented to increase the number of photos for the MEL, SEK and SCC classifications as they were disproportionately outweighed by the BCC classification in particular. Each photo belonging to these classes was rotated 90, 180 and 270 degrees to produce three more inputs for each original entry. Finally, no normalization was applied to the features, as random forests are invariant to feature scaling.

Additionally, three unique fields, leading to nine features, were added to both the training and test sets. The first of these fields was the color variation feature introduced in the project specification. This feature takes the standard deviation of the color in the entire image to provide data on whether a lesion is malignant. This is based on the observation that malignant lesions exhibit more color variations. Three features, one for each color in the RGB sequence, were added to the tabular test and training data. In a similar vein, a color intensity metric was added to the data. This color intensity metric was instilled to define differences in color between the base skin complexion and the lesion present. The color intensity of the middle 400 pixels, representing a 20×20 grid, was averaged and divided by the average color intensity of the pixels on the perimeter of the image. Three features, one representing the relative color intensity of each color in the RGB sequence, were added to the training and test features. This feature relies on the image being centered in the frame, which was found to be true for most images observed. While observing the image data, it was noticed that certain non-malignant lesions, particularly NEV and SEK, had less intense colors compared to the base complexion, which is predominately white. Therefore, this feature was added to provide additional information in differentiating non-malignant lesions. Finally, a field representing lesion symmetry was added to the training and test sets. Each photo was split into quadrants and rotated such that the lesion occupied the same area of the quadrant. The quadrants were superimposed, and the gradient of the color intensity was then computed for each color in the RGB sequence. The 100 largest gradients were averaged for each color and added as the three final numeric features.

1B: Cross Validation Design Description

All cross-validation was performed using `scikit-learn`'s `KFold` with 10 folds, a shuffled split, and a random seed of 7 to ensure reproducibility. Stratified Group KFold was used to produce the training and validation splits for the data. This was chosen because some patients have multiple entries within the training set. If these patients were split across the training and validation sets, performance would be artificially bolstered; therefore, they were grouped. Stratification was also necessary due to the immense class imbalance. Despite augmenting less represented classes, BCC still disproportionately outweighed other classes (BCC: 676, ACK: 218, NEV: 228, SEK: 128, MEL: 164, SCC: 154), making stratification necessary. With data augmentation, there were 1568 photos of lesions, making each fold 156 to 157 photos in size. The performance metric used throughout is AUCROC, evaluated on both training and validation folds; validation AUCROC guides hyperparameter selection for generalization, while comparison to training AUCROC helps diagnose over and underfitting.

One cross-validation grid search was run on the hyperparameters `max_depth` and `max_features`. `n_estimators` was considered to be included in the hyperparameter grid search but was excluded to reduce computation time, knowing the broad property of `n_estimators` to not guide the over- and underfitting regimes. Both `max_depth` and `max_features` were run over linear space vectors with five total values for each. The values of these hyperparameters were intentionally left low so as to not overfit to the training data and to create a greater difference between trees in the ensemble. In addition, hyperparameter `min_samples_leaf` was held at one to simplify the grid search. `Sklearn`'s `GridSearchCV` was used to compute the grid search.

After the cross-validation grid search was completed, a single ensemble using the optimal `max_depth` and `max_features` as well as `n_estimators` equal to 800 and `min_samples_leaf` equal to one was created to compute the test scores.

1C: Hyperparameter Selection

Random forests are well suited to classify lesions based on tabular datasets, as they are often cited as some of the better-performing models on tabular data. Because of this, creating tabular features based on the images aids in the performance of the model as a whole. Additionally, adding features increases the number of possible trees that can be created, further improving the random forest, as each tree will be closer to independent.`max_depth` and `max_features` were used in a grid search to determine the best parameters for the random forest classifier. Both hyperparameters were run on the same vector of values ([2, 7, 12, 17, 22]). These hyperparameter values were intentionally left small due to the relatively low number of features (66) in the dataset. If they were increased, a tree could be created that fits the noise in the validation set. Small values for `max_depth` and `num_features` are likely to create highly variable trees that often miss important parameters, which causes underfitting. Alternatively, trees with a `max_depth` and `max_features` at the top end of the ranges lead to fitting to dataset noise, a marker of overfitting. It should be noted that `n_estimators` was set at 800 due to the parameters' relative independence to overfitting, and `max_samples_leaf` was set to one. Additionally, `class_weight` was left at its default, as data augmentation was employed to help rebalance the class distribution.

It can be seen that training AUCROC readily increases until hitting the maximum of one at a `max_depth` of 12 and stays there for the remainder of the hyperparameter grid search. Meanwhile,

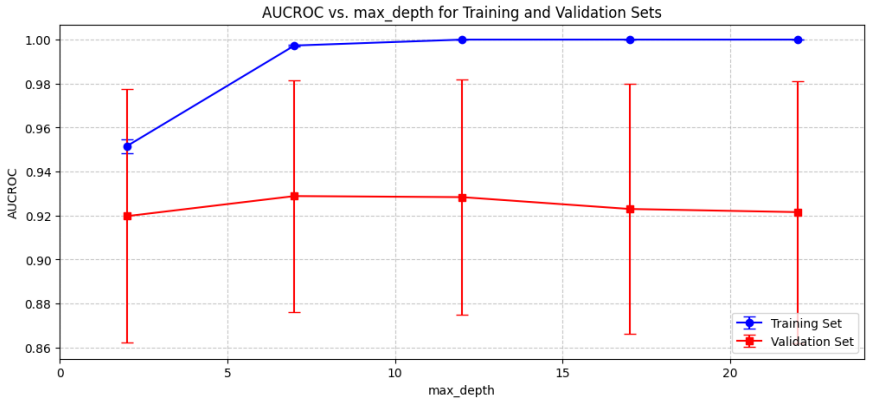

Figure 1: AUCROC vs. `max_depth` Hyperparameter with Error Bars Representing the Standard Deviation Across the 10 Folds with `max_features` Equal to 2.

validation AUCROC increases steadily from a `max_depth` of 2 to 7, stays steady from 7 to 12, and decreases steadily after 12. This is while training AUCROC increases. This shows a typical underfitting-to-overfitting regime.

It can be seen that the preferred parameter for controlling the complexity of the model is `max_depth`, as `max_features` had little correlation with model performance. To confirm this judgment, a separate grid search fixing `max_depth` equal to seven was run, with a vector of `max_features` of ([1, 20, 40, 66]). This grid search confirmed that `max_features` has little effect on model performance, as the reported AUCROCs of [0.92657878 0.9256088 0.9267907 0.92179645] only showed overfitting once every feature was included. It was found that the hyperparameters that perform best are a `max_depth` of 7 and a `max_features` of 2 with a validation AUCROC over ten folds of 0.9286.

1D: Reflection on Test Set Performance

The final hyperparameters used were `max_depth` equal to seven and `max_features` equal to two. The AUCROC achieved by the random forest was 0.90635, which falls slightly below the validation AUCROC of 0.9288 achieved by our classifier on the validation set with the best hyperparameters. one striking result of the classifier is the TNR of 0.21739. This TNR implies that we are misclassifying a large percentage of cancer-free lesions as positive. This is also seen in a balanced accuracy of 0.6087 and TPR of 1.0.

Our initial efforts to increase the TNR of our classifier involved augmenting the disproportionately small number of non-cancerous lesions to better represent a balanced dataset. These efforts, however, led to a decrease in TNR, despite a more balanced dataset. Our first submission, run without augmentation, had a TNR of 0.43478 and a TPR of 1.0. Additionally, we implemented features that we believed would increase negative class prediction fidelity. Particularly, it was noticed that non-cancerous lesions were more symmetric and had defined edges, which guided the use of color gradient in the 'quadrant' features. Our results show that these methods are insufficient. This is possibly due to a 'blurring' of the images used in the 'quadrant' features. The four rotated quadrants were superimposed onto a single image, which blurs the true edges. Future efforts of defining symmetry should be isolated from efforts of defining edges to reduce this confoundment.

Despite this fault, it should be noted that the cost of a false negative classification is much more costly than a false positive classification. A false negative classification leads to a patient disregarding a potentially life-ending ailment, a severely high cost. This is opposed to a false-positive classification, which likely leads to a patient receiving a biopsy for the lesion, which will correctly classify it. While this has an associated cost, it is not on the same magnitude of a false negative. This potentially explains our TNR and TPR, as the threshold for a positive result was likely placed low for the test set to avoid false negative results.

Problem 2

2A: Feature Representation Description

Images were augmented more heavily in problem 2. All images in the training set were rotated as done in problem 1. Additionally, all non-BCC images were reflected across the vertical axes and rotated three more times. Finally, to improve performance on the older ages and non-BCC classes, BCCs under the age of 55 were removed for the final image set. For problem 2, six features were added to the problem 1 feature representation. Using the pretrained model, `EfficientNet_b0` scores were created for each lesion class, representing the model's confidence in labeling an image with each class. The pre-trained weights were used, along with `PyTorch's Adam` optimizer. A grid search covering learning rate ([1e-5, 1e-4, 1e-3, 1e-2, 1e-1]) and batch size ([32, 64]) was implemented to find the best-performing combination of hyperparameters for the neural network. It was found that a learning rate of 0.001 and batch size of 32 performed best over seven epochs. The computed scores for each image were added to feature representation, leading to 72 final features. `EfficientNet_b0` has 5.3 million parameters, fitting within the acceptable range of 10 million. The initial layer is a 3×3 convolution with 32 filters and a stride of 2. The body of the NN has seven stages, which feature a 1×1 convolution that increases channels, a 3×3 convolution that processes channels independently and a 1×1 convolution that shrinks the channels back down. The head contains global average pooling to flatten the NN and a softmax calculation for predictions.

2B: Cross Validation Design Description

We trained two models independently: a Random Forest, which was used to classify older people, and a XGBoost model, which was used to classify adults. The two models were trained on the same features, but with independent hyperparameter searches. For the elder-classifying Random Forest model, we fixed `min_samples_leaf`, `n_estimators`, and `min_samples_split`, and searched over `max_depth` and `max_features`. We used entropy as our split criterion as it was what we used in Homework 7 when training decision trees, and was introduced in lecture as one of the two best split criteria, along with Gini impurity. Similar to Problem 1A, we used `StratifiedGroupKFold`, grouping by patient, to avoid data leakage between the training and validation sets and to maintain class balance within each fold. Our scoring metric was AUCROC for patients over the age of 60, which was averaged over $K = 8$ validation folds to obtain a score for each hyperparameter pairing. We chose $K = 8$ to balance computational cost with reliable estimates of variance. There were 5868 inputs after augmentation, meaning each fold was roughly 730 entries large. Since this model is ultimately being evaluated on AUCROC on people over 60 years old in the test set, we thought making this our validation evaluation metric would yield the best test performance. We defined this scoring metric using `sklearn.metrics.make_scorer()`.

Since we used a XGBoost model to classify middle-aged adults, the hyperparameter search dif-

fered than that of the Random Forest. We used a binary classification objective and searched over 5 hyperparameters, which are specified in 2C, in Table 1. Again, we used `StratifiedGroupKFold`, grouping by patient, for the reasons noted in the previous paragraph. Our scoring metric was AUCROC for *all* patients, because we found that using general AUCROC still yielded good performance on the target age group. Again, we averaged over $K = 8$ validation folds. Both models were trained on `X_dev` and `y_dev`, which contain only the provided training data. To predict on the test set, we wrote a function that checks the age of each patient, passes each patient's data to the corresponding model, and compiles the predictions into a 1D array.

2C: Classifier Description and Hyperparameter Search

XGBoost models are considered state of the art on tabular data, making them well-suited to classify lesions based on our dataset. As stated in 2B, we trained a Random Forest to classify people over the age of 60, and a XGBoost model to classify people between the ages of 30 and 60. For the elder-classifying Random Forest model, we fixed `min_samples_leaf` to be 1, `n_estimators` to be 500, `min_samples_split` to be 2, and used entropy as our split criterion. We searched over `max_depth` values of [1, 5, 10, 15, 20] and `max_features` values of [1, 5, 10, 15, 20]. `max_depth` controls model complexity by limiting the size of each tree, preventing each data point from having its own leaf node, and `max_features` limits the number of features that can be considered at each split, resulting in diverse trees. The best-performing model had a `max_depth` of 15 and a `max_features` value of 10. All of the values searched for both hyperparameters are rather small, which we chose to reflect the size of the dataset. We did not encounter issues with step size or convergence.

Since we used a XGBoost model for 30 - 60 year-olds used a log-loss objective function, specified with `objective='binary:logistic'`. This is a standard objective function for binary classifiers. We searched over 5 hyperparameters – `max_depth`, `n_estimators`, `learning_rate`, `subsample`, and `colsample_bytree`. The values searched for each hyperparameter and the results are reported in Table 1. All of the values of `max_depth` over which we searched are very small, again, which reflects the size of our dataset. While they don't directly control model complexity, we searched over `n_estimators`, `subsample`, and `colsample_bytree` to find values that resulted in the best model flexibility. Finally, we searched over `learning_rate` to find a model that trains as quickly as possible without diverging. Since we have so many parameters, we trained over smaller ranges.

Hyperparameter	Values Searched	Best Value
`max_depth`	[1, 3, 5, 7, 9]	5
`n_estimators`	[200, 500]	500
`learning_rate`	[0.01, 0.1, 0.3]	0.1
`subsample`	[0.8, 1.0]	1.0
`colsample_bytree`	[0.8, 1.0]	1.0

Table 1: Summary of hyperparameter search for XGBoost model used to classify middle-aged adults

Figure 2 clearly shows evidence of underfitting at `max_depth` = 1, with sub-optimal performance on both training and validation sets, as well as evidence of overfitting at `max_depth` = 7 and 9, with validation performance decreasing while training AUCROC remains near 1. The value resulting in the best performance, is in the middle at 5. This evidence, however, is fairly uncertain: the error bars, representing standard deviation between folds, are much taller than the range of mean AUCROCs across the 5 values. While we choose `max_depth` = 5, we bear in mind that there may be another value that generalizes better.

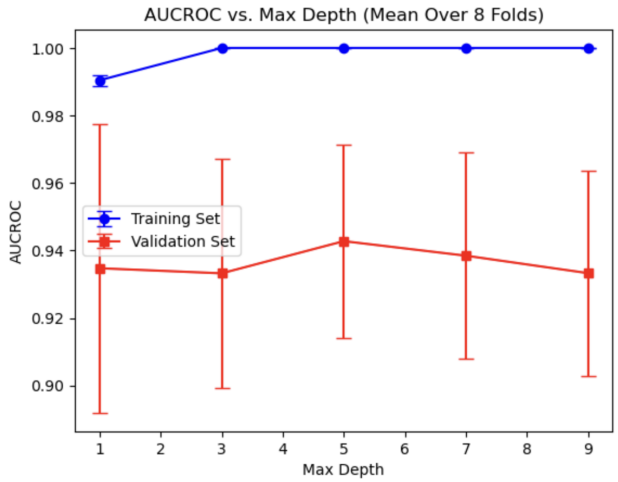

Figure 2: Training and validation performance over values of `max_depth` at best values of other parameters searched for the XGBoost model.

2D: Reflection on Model Performance

Our problem 1 AUCROC is **0.90635**, and our worst performing AUCROC for problem 2 is **0.88125** for adults. For Problem 1, our most confidently correct and most confidently incorrect predictions are reported in Tables 2 and 3 with images in Figures 3 and 4, and for Problem 2, our most confidently correct and most confidently incorrect predictions are reported in Tables 4 and 5 with images in Figures 5 and 6.

Patient ID	Image ID	Probability	True Label	Age
PAT_117	PAT_117_1145	0.9725	1	64
PAT_158	PAT_158_1106	0.9684	1	64
PAT_158	PAT_158_017	0.9670	1	64

Table 2: Most confidently correct predictions on held-out data for problem 1

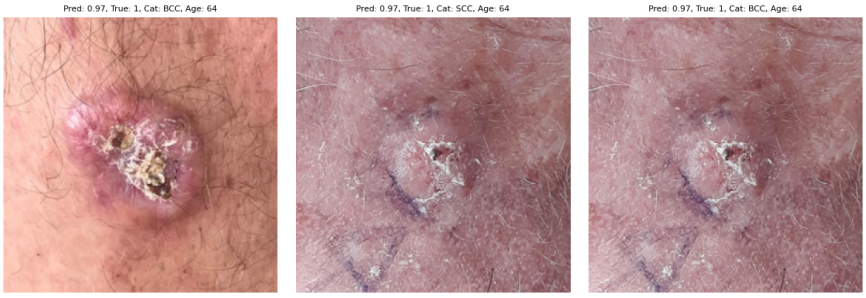

Figure 3: Most confidently correct predictions on held-out data for problem 1

For problem 1, our model classifies cancerous lesions most confidently on older patients. This is evident from our model's largest feature importance of 9.00e-02 being attributed to the 'age' feature. Additionally, all three confident predictions also shared true classifications for the important features of 'hurt' and 'elevation.' This shows that the random forest classifier gives very confident

Patient ID	Image ID	Probability	True Label	Age
PAT_174	PAT_174_1301	0.7613	0	77
PAT_093	PAT_093_1251	0.7367	0	54
PAT_114	PAT_114_568	0.7030	0	74

Table 3: Most confidently incorrect images on held-out data for problem 1

Figure 4: Most confidently incorrect images on held-out data for problem 1

Patient ID	Image ID	Probability	True Label	Age
PAT_333	PAT_333_121	1.0	1	71
PAT_143	PAT_143_677	1.0	1	64
PAT_267	PAT_267_1085	1.0	1	81

Table 4: Most confidently correct predictions on held-out data for problem 2

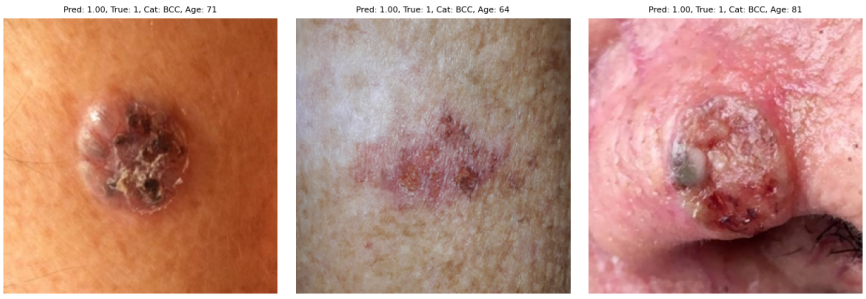

Figure 5: Most confidently correct images on held-out data for problem 2

Patient ID	Image ID	Probability	True Label	Age
PAT_251	PAT_251_312	0.0308	1	35
PAT_243	PAT_243_1207	0.1698	1	33
PAT_961	PAT_961_006	0.2570	1	34

Table 5: Most confidently incorrect predictions on held-out data for problem 2

predictions to older individuals who display these common cancer signs. Additionally, our created features had large feature importances. The assymetry in PAT_117 represents a common skin cancer sign and leads to a stronger positive prediction. We believe the pen markings on PAT_158 yield a stronger prediction as well. It should be noted that both PAT_158 entries have different classifications. This is a product of the dataset. Our problem 1 classifier, however, tends to give

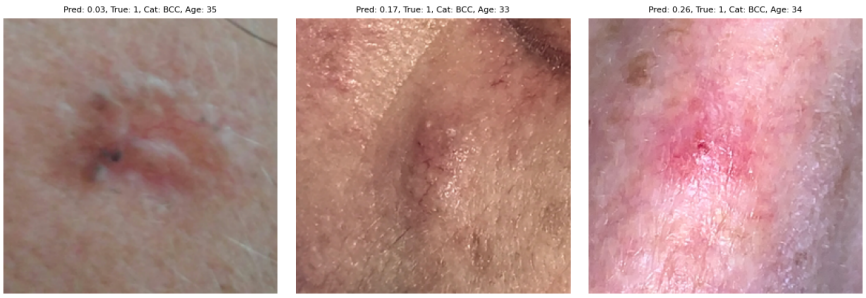

Figure 6: Most confidently incorrect images on held-out data for problem 2

more positive predictions than negative predictions, which is shown in our most confident incorrect predictions, which give positive predictions to noncancerous lesions. These predictions come from patients older than 50, showing again that older patients receive more positive predictions. The two strongest incorrect predictions share a history of cancer, another feature with high importance, which is believed to yield strong predictions despite other high-importance features, such as the 'hurt' being negative. The third most incorrect prediction, belonging to PAT_114, has both a larger diameter_1 and diameter_2 and is elevated, a combination that lends itself to cancer. With this being said, our problem 1 classifier weights age very importantly when determining a coarse label. This comes to its detriment when classifying older patients who do not have cancer. With these weaknesses highlighted, if given more time, we would create more augmented images of noncancerous lesions for those over 50 years old, and implement a custom scorer in the hyperparameter grid search that maximizes AUROC on those over 50.

As described in 2B and 2C, our problem 2 model is composed of a Random Forest and an XGBoost model, which are deployed for data points in different age groups. These classifiers exhibit different behavior. Our strongest, most positive predictions come from the elder classifiers, which give predictions of 1.0 for many predictions. It can be seen in Figure 5 that the three most confident and correct predictions have very noticeable colorations when compared to the base skin. This feeds into the color intensity feature that was created for boths problems 1 and 2, which has a strong feature importance in all classifiers used in this project. Additionally, the symmetry classifier lends itself well to classifying PAT_143. The asymmetry in PAT_267 is largely due to the location of the lesion on the nose. It was found that the location features, such as 'location_back,' have high importance and can be used within each tree to distinguish true asymmetry. Our elder classifier shows strong positive predictions but also fewer confidently incorrect predictions. This is evidenced by the fact that almost all confident incorrect predictions come from the adult classifier. In this case, all incorrect classifications are cancerous lesions that are predicted as non-cancerous. This in large part has to do with the age of the patients being in their low thirties but also has to do with the nature of the lesion. The three lesions both reflect the base skin tone. It is believed that this tricks the fine label score features created by the pretrained neural network. It was found that the NN correctly predicts very pronounced lesions as cancerous, particularly BCC, but incorrectly creates scores on subtle lesions. This also lends itself to the strong correct predictions explained in Figure 5. XGBoost was used to correct for these subtleties in cancerous lesion appearance but did not do so in all cases. If given more time, we would work on further refining the features given by the pre-trained neural network: since these features had large importance, we believe this may have been a point of confusion. We could have fine-tuned additional layers of `EfficientNet-b0` on our cancerous lesion dataset, rather than keeping all weights frozen at their pretrained values.