Technische Universität Dortmund

Physics Department

International Master Advanced Methods in Particle Physics

Detecting Brain Tumours using a Convolutional Neural Network

Machine Learning Report

Date: July 30, 2024

Bastian Schuchardt - bastian.schuchardt@tu-dortmund.de

Dortmund, Germany JULY, 2024

Contents

Detecting Brain Tumours using a Convolutional Neural Network

Machine Learning Seminar

Bastian Schuchardt

July 30, 2024

Abstract

Brain tumours are growths of cells in the brain that multiply abnormally. They can be classified as benign (non-cancerous) or malignant (cancerous). The detection of tumors especially malignant ones is crucial for doctors. Machine learning can be used as a supplementary tool for medical diagnosis. Four machine learning models will be trained on a labelled dataset of 3762 magnetic resonance imaging (MRI) scans with 13 tumour features derived from the images. The performances for training on the MRI scan images and training on the tumour features will be compared to evaluate how much performance is lost by skipping the feature derivation and directly taking the MRI scans as input. The highest accuracy of 98.01 % for training on the features was achieved by a Support Vector Machine (SVM), and the highest accuracy of 96.94 % for training directly on the images was achieved by a Convolutional Neural Network (CNN). The accuracy difference is notable, which is a sign of future areas of improvement that will be discussed.

1 Introduction

Malignant brain tumours are cancerous cell growths in the brain that are life-threatening if undetected and untreated. One non-invasive method to detect brain tumours is to use MRI scans to analyse and classify them as malignant or benign. Doctors usually carry out this task which is time-consuming and can be subjective. A machine learning model can assist doctors to have a higher combined prediction accuracy. Cases in which the doctor classifies the tumour as benign and the machine learning model predicts it to be malignant are especially important because the doctor can reevaluate their decision and potentially save more lives.

The goal of this project is to compare the performance between a Convolutional Neural Network trained directly on the MRI scans and a k-Nearest Neighbours (k-NN) model, a Logistic Regression, and a Support Vector Machine trained on the features of the tumours derived from the MRI scans. The CNN should have an accuracy close to the other three models trained on the features to justify skipping the tedious feature derivation.

2 Dataset

The dataset is labelled and consists of 3762 MRI scans of benign and malignant tumours and features of the tumours derived from the scans. It was published on Kaggle [1] under the CC BY-NC-SA 4.0 license [2]. Nine MRI scans are shown in Figure 1. The five first-order features are mean, variance, standard deviation, skewness, and kurtosis. Additionally, there are eight second-order features: contrast, energy, angular second moment (ASM), entropy, homogeneity, dissimilarity, correlation, and coarseness. The dataset has a class distribution of 1,683 malignant and 2,079 benign cases. The grey-scale images are stored in the jpg format and have a size 240 × 240 pixels.

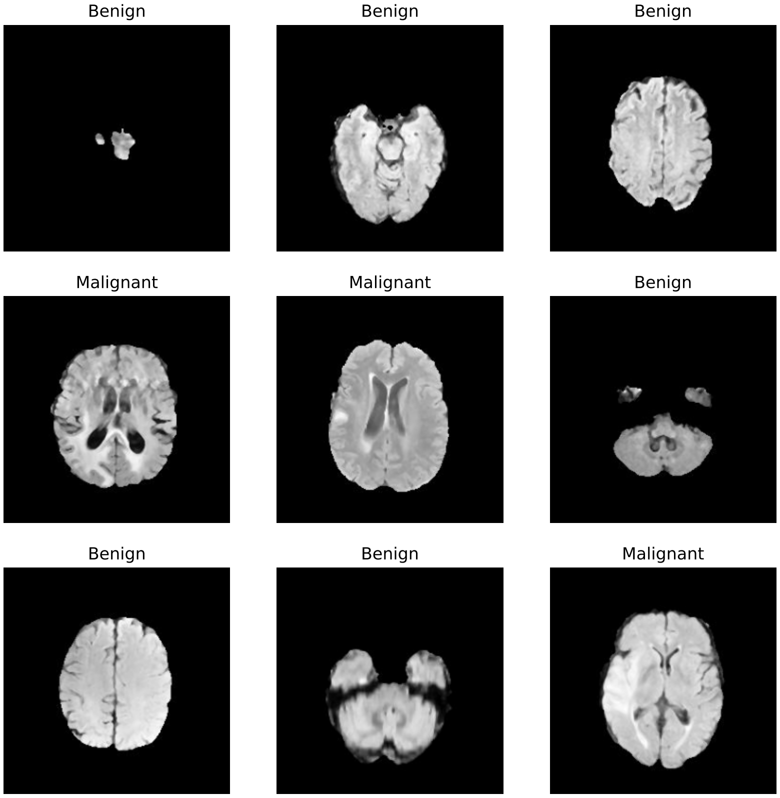

Figure 1: MRI scans of malignant and benign brain tumours.

The feature distribution for benign and malignant tumours can be seen in Figure 2. The malignant and benign distributions have a comparable shape for the mean, variance, standard deviation, skewness, kurtosis, contrast, dissimilarity, correlation, and contrast. Only entropy, energy, ASM, and homogeneity have a clear separation between the malignant and benign distributions. This trend can be noticed in the correlation between the features and the diagnosis, where the four features are strongly negatively correlated with a malignant diagnosis. The correlation coefficients of the tumour features are given in Table 1. In addition, the scatter matrix of the features and a heatmap of the correlation matrix are given in Appendix A.

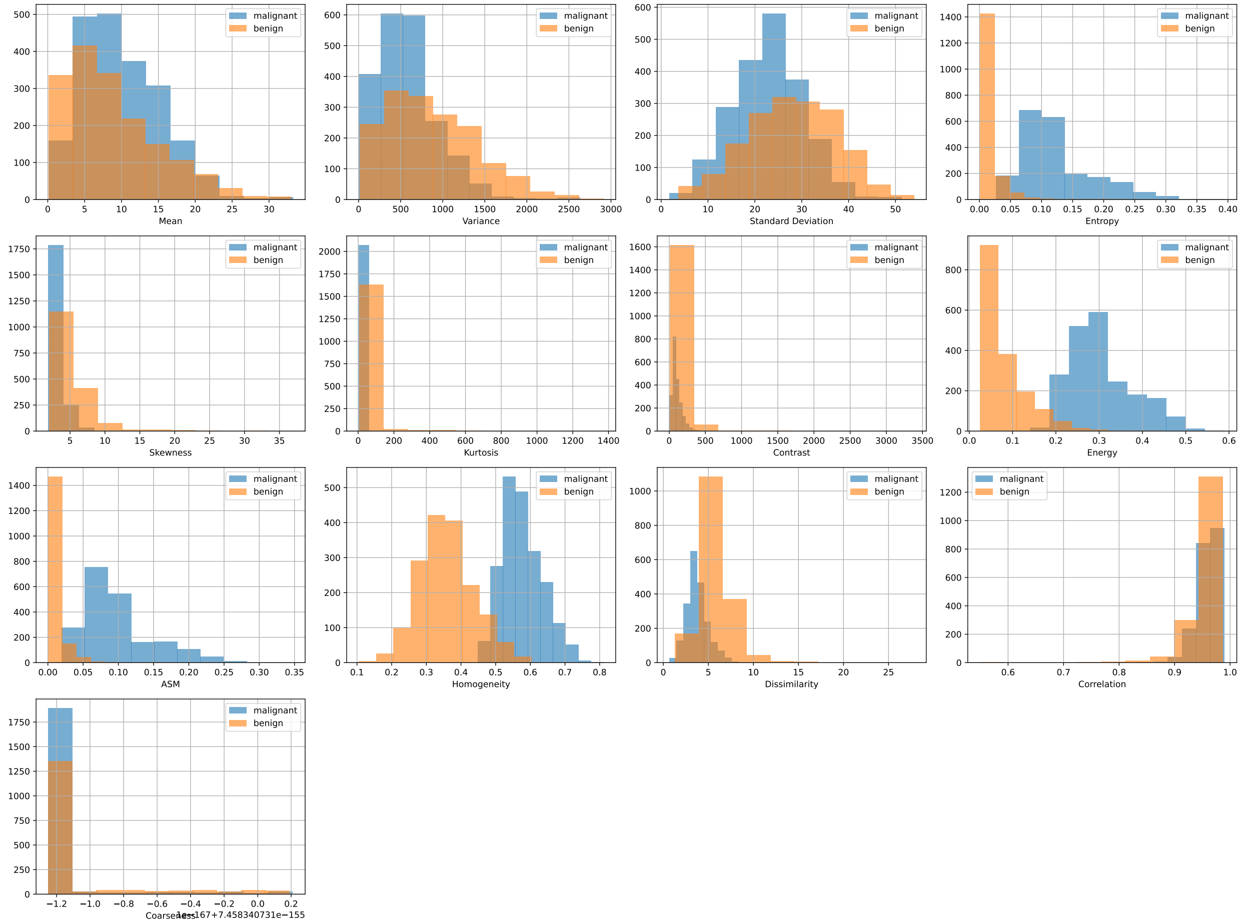

Figure 2: Histograms of the distribution of the five first-order and eight second-order features derived from the MRI scans. The malignant distributions are highlighted in an orange colour and the benign distributions in a blue colour.

Even without much knowledge of medicine and brain tumours, it is possible to make an educated guess that the variance, standard deviation, skewness, and dissimilarity should positively correlate with a malignant diagnosis.

Table 1: Correlation coefficients of the features for a malignant diagnosis.

	Mean	Variance	Standard Deviation	Entropy	Skewness	Kurtosis
Correlation	-0.096	0.31	0.29	-0.78	0.4	0.24
	Contrast	Energy	ASM	Homogeneity	Dissimilarity	Correlation
Correlation	0.21	-0.86	-0.76	-0.85	0.56	-0.11

Albeit, the correlations are only weak to moderate with a correlation coefficient of 0.56 for the dissimilarity as the strongest correlating feature. Some features are strongly negatively correlated and some are weak to moderately positively correlated with a malignant diagnosis so it can be concluded that the models trained on the tumour features will probably perform well.

2.1 Preprocessing

The dataset has no missing values so no data cleaning is needed. To boost the performance of all models a simple preprocessing was applied for the MRI scans and the tumour features. The MRI scan images have been rescaled by a factor of $\frac{1}{255}$ to be between zero and one, and the tumour features were scaled by the scikit-learn [3] standard scaler. The tumour features and MRI scan images were split into a train and a test dataset with a test size of 20 %. The split for the tumour features was carried out with the help of scikit-learn and the split for the MRI scans was achieved by moving the images to subfolders that made it possible to load them into tensorflow datasets.

3 Analysis

An SVM, logistic regression and k-NN model all implemented in scikit-learn [3] were chosen to be trained on the tumour features. Additionally, a custom CNN network implemented in TensorFlow [4] was trained on the MRI scan images. The jupyter notebook attached to this report [5] does work with the environment given for this course but TensorFlow must be updated to version 2.16.1 so the Keras [6] function image_dataset_from_directory is usable. Without the update, the CNN model cannot be trained.

3.1 SVM

Support vector machines perform optimal data transformations to determine boundaries between classes. In this case, the SVM was initialised with a radial basis function (RBF) kernel that measures the similarity between two data points and computes the decision boundary by majority vote. The RBF kernel function utilises the Euclidian distance, which means that data points that are far from each other are more likely to be dissimilar [7]. SVMs perform particularly well for high-dimensional datasets, so it is expected that this model will have the best performance compared to the other models.

3.2 Logistic Regression

Logistic regression predicts if a feature is malignant or benign by utilising the logistic or sigmoid function to map the features to probabilities ranging between zero and one. The threshold for a malignant diagnosis was chosen to be 0.5. Furthermore, the L2 or Ridge norm was selected as the penalty function to avoid overfitting. Logistic regression is one of the most widely used models for binary classification and should perform well, although it will probably be outperformed from the SVM.

3.3 k-NN

A data point is classified as malignant or benign by taking a plurality vote of the class of the number of k-nearest neighbours. The number of nearest neighbours k was chosen to be five and no weighting to nearest neighbours was applied. The distance was calculated with the Euclidian metric. It should be noted that the choice of metric can vastly influence the performance of the model and different metrics were not checked in this case. Nevertheless, the model should perform comparably to the other models.

3.4 CNN

A convolutional neural network is used for machine vision and usually consists of a convolutional layer, a pooling layer, and a fully connected layer. In the convolutional layer a kernel is applied that acts as a filter for the input image. Mathematically the kernel is a matrix and it reduces the input by applying a convolution to it. This step allows it to reduce the number of weights vastly compared to fully connected neural networks. The pooling layer reduces the input further by e.g. just taking the maximum of a slice of the kernel. This process is called max pooling. After the pooling layer, the 2D data will be flattened to a 1D array and fed into a deep neural network that classifies the input images.

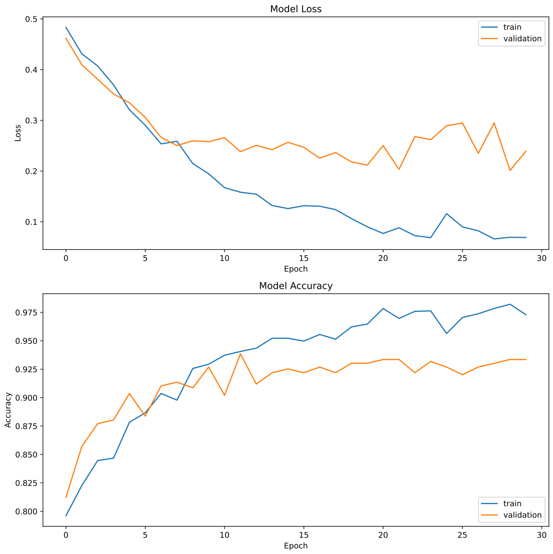

Figure 3: Loss and accuracy curve for the CNN model for the training and validation dataset.

The CNN was initialised with three convolutional layers with a kernel size of 3×3 followed by a max pooling layer with a pool size of 2×2 each. The number of filters is reduced for each layer. The first convolutional layer has 24, the second one 12 or $\frac{1}{2}$ of 24, and the third one has six filters or $\frac{1}{3}$ of 24. Afterwards, there is a flattening layer followed by a dropout layer with a dropout rate of 0.5 to reduce overfitting, a dense layer with four units and finally another dense layer acting as the output layer. The activation function for the convolutional and the dense layers is the rectifier linear unit (ReLU) function. The loss function is the sparse categorical cross entropy and the metric to be optimised is the accuracy. The optimisation was done with the Adam optimiser [8] for a learning rate of 0.005 and batch size of 32. The validation data is 20 % of the training data and the class distribution was kept the same for the training and validation data. The training history for 30 epochs can be seen in Figure 3. The validation loss curve is higher than the training loss curve which is a strong indicator for overtraining. In general, the number of filters, the dropout rate, and the number of dense units were rather arbitrarily chosen to have a starting point for further improvements.

3.5 Hyperparameter Optimisation

A hyperparameter optimisation was done to improve the accuracy of the CNN model, where the number of filters, the number of units in the dense layer, and the dropout rate were taken as hyperparameters. The optimisation was done by a grid search [9] for the possible number of filters of 24, 36, 48, 60, and 72. The possible values of dense units were chosen to be 4, 8, 12, 16, 20, and 24, and the possible dropout rates were chosen to be 0.4, 0.5, and 0.6. The pairplots for these three hyperparameters are given in Figure 4.

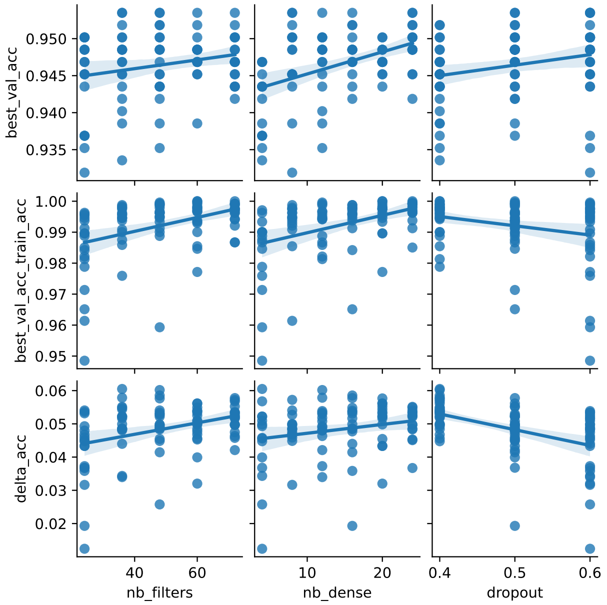

Figure 4: Pairplots of the best accuracy from top to bottom on the validation dataset, the best value on the training dataset, and the difference between both accuracies (delta accuracy) against the tuned hyperparameter number of filters, the number of dense units, and the dropout rates.

Three kinds of accuracies are shown in the pairplot. First, the best accuracy achieved on the validation dataset, second the best accuracy on the training dataset, and lastly the difference between them that is called delta accuracy. It can be seen that both accuracies improve for a higher number of filters and units in the dense layer. The number of filters is only applied in the first convolutional layer. The second layer has $\frac{1}{2}$, and the third one has $\frac{1}{3}$ of the number of filters of the first layer. The dropout rate only improves the accuracy on the validation dataset and reduces it for the two other accuracies. However, the dropout rate should be large to avoid overfitting and consecutively yield a smaller delta accuracy. The loss curves for the three best-performing models are given in Figure 5.

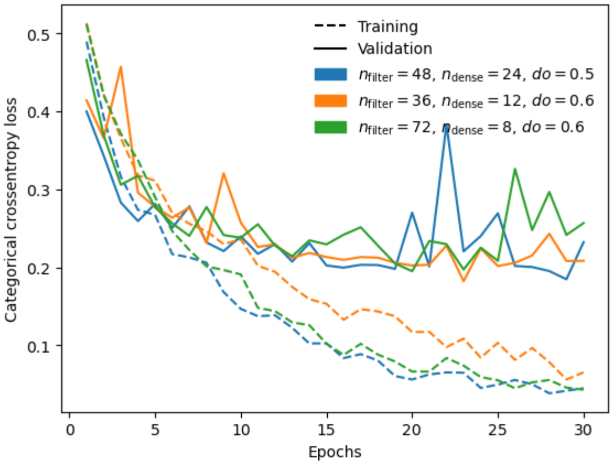

Figure 5: Training and validation loss curves for the three best-performing models with given hyperparameters for 30 epochs.

It can be seen, that the best models show signs of overfitting because the validation loss curve is much higher than the training loss curve.

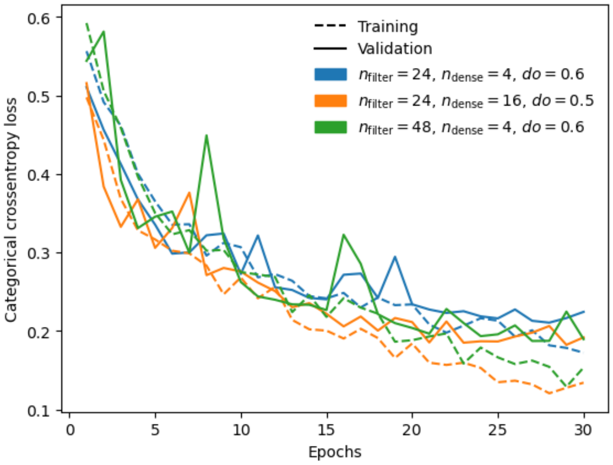

Figure 6: Training and validation loss curves for the three models with minimal delta accuracy for 30 epochs.

The three best-performing models with the least delta accuracy are given in Figure 6 and they show no signs of overfitting. All validation and training loss curves converge so that the number of filters can be chosen to be 24, the number of dense units to be four, and the dropout rate to be 0.6. The training and validation loss curves and the accuracy on the train and validation dataset for 100 epochs are shown in Figure 7. The accuracy has improved significantly compared to the CNN without tuned hyperparameters and the loss curves show no sign of overfitting.

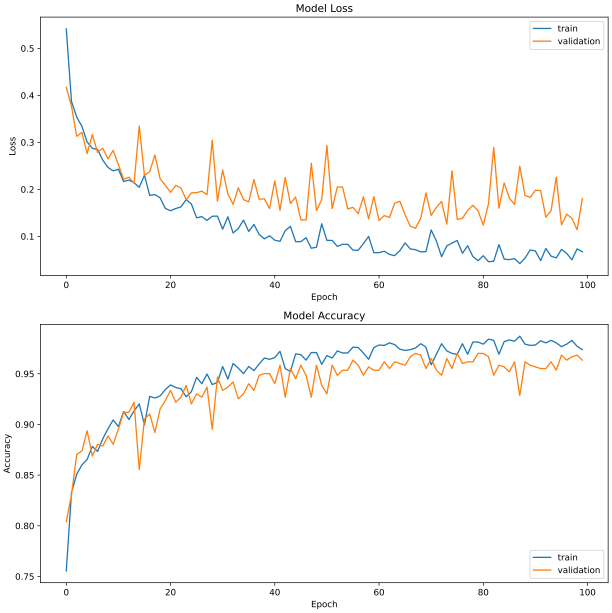

Figure 7: Loss and accuracy curve for the CNN model with tuned hyperparameters for the training and validation dataset.

3.6 Performance Evaluation

All models are evaluated with recall, accuracy, precision, F1-score, R2-score, and mean squared error (MSE) on a test dataset.

Table 2: Recall, accuracy, precision, F1-score, R2-score, and MSE for SVM, logistic regression, k-NN, CNN, and CNN with optimised hyperparameter values. The best scores achieved are written in bold.

	SVM	Logistic Regression	k-NN	CNN	CNN HP
Recall	**0.9617**	**0.9617**	0.9469	0.8726	0.9519
Accuracy	**0.9801**	0.9788	0.9761	0.9229	0.9694
Precision	0.9939	0.9909	**1.0000**	0.9864	0.9925
F1	**0.9775**	0.9760	0.9727	0.9260	0.9718
R2	**0.9195**	0.9142	0.9034	0.6880	0.8763
MSE	**0.0199**	0.0212	0.0239	0.0771	0.0306

The respective values of the performance metrics are given in Table 2. In addition, the confusion matrix and the distribution of the predicted and the true classes have been plotted. The plots for all models are given in Appendix B. It can be seen that the SVM model performed the best of all the models for the most performance metrics, which was expected. It is only trumped for the precision value, where the k-NN model had a perfect score of 1.0. Additionally, the logistic regression model achieved the same recall value as the SVM model with a value of 0.9617. It should be stressed that the recall value is of high importance because the higher the value the lower the number of false negative predictions are. False negative predictions should be avoided at all costs because they result in the worst case in no treatment for a malignant tumour and the potential death of a patient. An accuracy of 98.01 % was achieved so that the SVM model is suitable for accurately classifying brain tumours. The accuracy score could be further improved by tuning the hyperparameters of the models or choosing a better-suited model like a neural network for example.

As expected, the CNN model performs significantly worse for all performance metrics, e.g. the accuracy is 92.29 % and therefore 5.72 % lower than the accuracy of the SVM model. After tuning the hyperparameters, the CNN model performed significantly better and now has only an accuracy difference compared to the SVM model of 1.07 % with an accuracy of 96.94 %. It even has a better recall score than the k-NN model but it still falls slightly short compared to the logistic regression and SVM model.

4 Discussion

The CNN model with tuned hyperparameters performs only slightly worse than the SVM model trained on the tumour features so it can potentially be used in medical applications to support doctors. Nevertheless, it is still an open question if a CNN model trained on MRI scans will outperform models trained on tumour features. This would need to be checked by changing the CNN architecture and trying a larger scoped hyperparameter optimisation. Moreover, it would be interesting to investigate how much the accuracy of the models trained on the tumour features can be improved because better-performing models are crucial for saving lives. If it turns out that the models trained on the tumour features will always outperform a CNN trained on MRI scans then it could be proposed that the CNN shall be trained to extract the tumour features from the MRI scans and then let another model classify the tumour from the derived features. Another investigation area would be to train the models for a higher recall value instead of training them for optimal accuracy to have fewer false negative predictions.

5 Conclusion

An SVM, a logistic regression, and a k-NN model were trained on the tumour features, where the SVM model outperformed the other models with an accuracy of 98.01 %. Moreover, a CNN model with a simple architecture was trained on the MRI scan images, yielding a significantly lower accuracy of 92.29 %. After tuning the hyperparameters by a grid search, the best hyperparameters were chosen so that the CNN model needed less time to train and showed no signs of overfitting. An accuracy of 96.94 %, which is only 1.07 % lower than the accuracy of the best-performing SVM, was achieved with this model. From this, it can be concluded that CNNs trained on MRI scans could potentially be used to help doctors diagnose brain tumours. The results are satisfying for the scope of this machine learning project but further research is needed. It is still an open question how much the performance of a model trained on the tumour features and a CNN trained on the MRI scans can be improved and if the CNN will ever outperform a model trained on the tumour features. A CNN model will in principle yield faster results but the prediction of a model will influence the doctor deciding the treatment of a patient. Considering this it should be discussed if it is not better to derive the tumour features by hand, albeit slower, and train a model on them for the last bit of accuracy improvement. Another solution that can be proposed is to train a CNN to derive the tumour features from the MRI scans and then let a model classify the tumour from the features.

References

[1] Jakesh Bohaju. *Brain Tumor*. 2020. DOI: 10.34740/KAGGLE/DSV/1370629. URL: https://www.kaggle.com/dsv/1370629.

[2] *Attribution-NonCommercial-ShareAlike 4.0 International*. Version 4.0. Creative Commons Corporationn, Aug. 26, 2021. URL: https://creativecommons.org/licenses/by-nc-sa/4.0/.

[3] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[4] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[5] Bastian Schuchardt and Joel Koch. *Detecting Brain Tumours using a Convolutional Neural Network*. https://github.com/joeyko2706/MachineLearning24/tree/main/Project. Version 1.0. July 2024.

[6] François Chollet et al. *Keras*. https://keras.io. 2015.

[7] sci-kit learn developers. *Plot classification boundaries with different SVM Kernels*. https://scikit-learn.org/stable/auto_examples/svm/plot_svm_kernels.html. 2024.

[8] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG]. URL: https://arxiv.org/abs/1412.6980.

[9] Steven M LaValle, Michael S Branicky, and Stephen R Lindemann. "On the relationship between classical grid search and probabilistic roadmaps". In: *The International Journal of Robotics Research* 23.7-8 (2004), pp. 673–692.

Appendix A

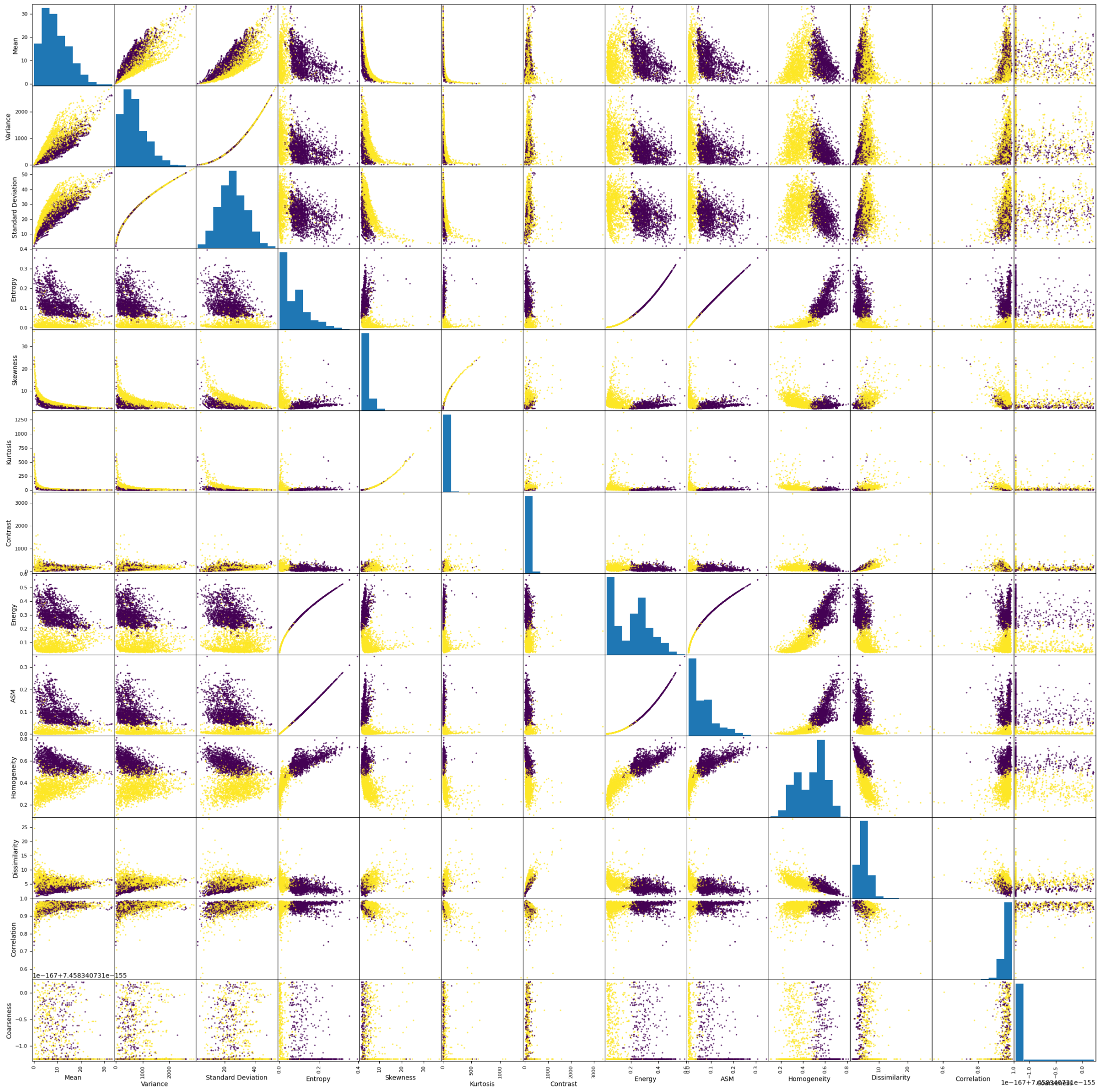

Figure 8: Scatter matrix of all the tumour features.

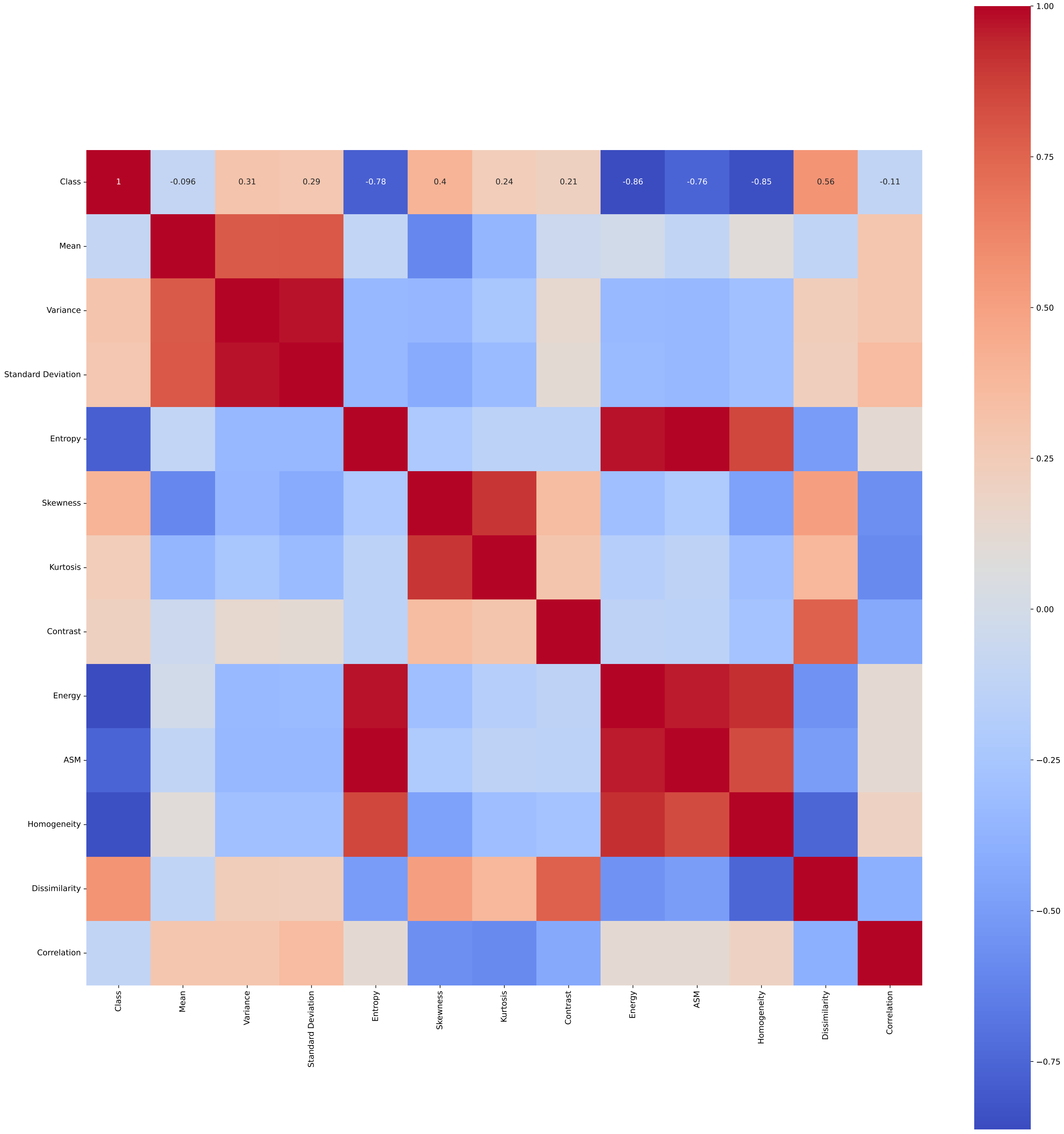

Figure 9: Correlation matrix of the class, where one is a malignant and zero is a benign diagnosis, and the other tumour features. Coarseness has been omitted because it is numerically much smaller than the other values and the computed correlations are negligible.

Appendix B

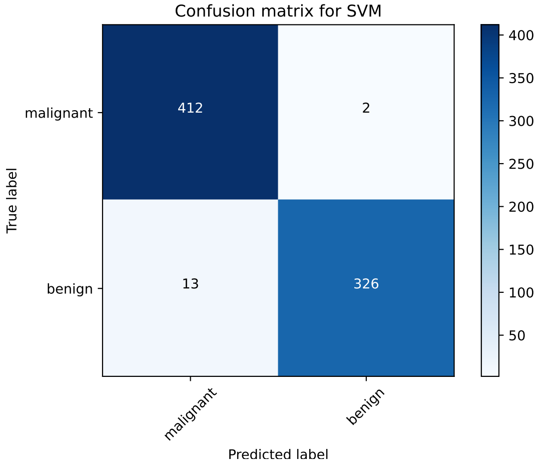

Figure 10: Confusion matrix of SVM model.

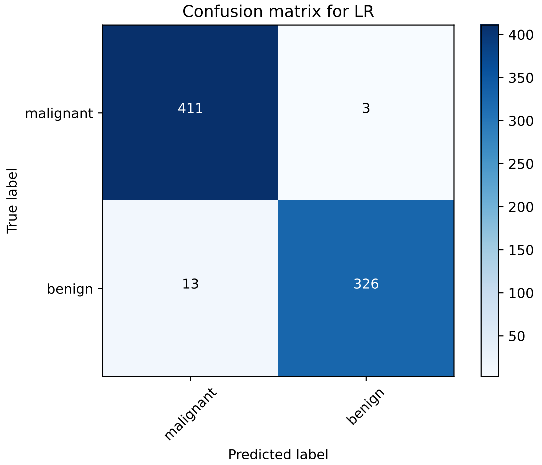

Figure 11: Confusion matrix of logistic regression model.

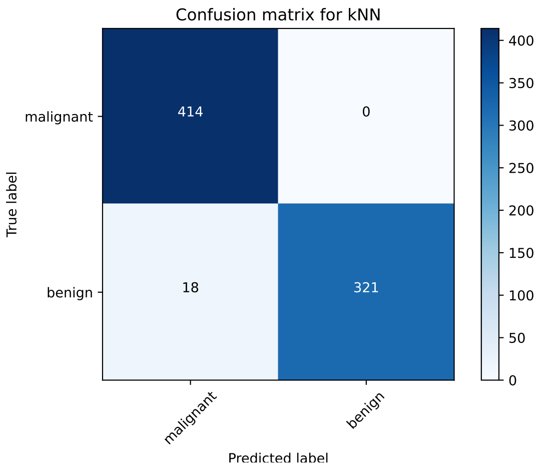

Figure 12: Confusion matrix of k-NN model.

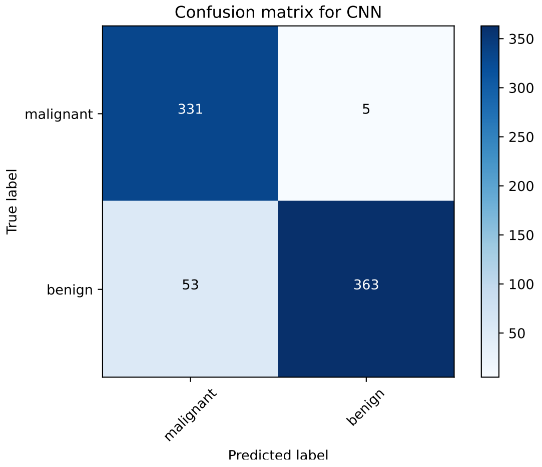

Figure 13: Confusion matrix of CNN model.

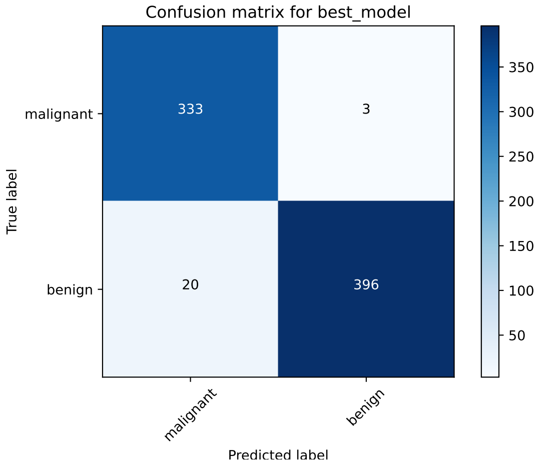

Figure 14: Confusion matrix of CNN model with tuned hyperparameters.

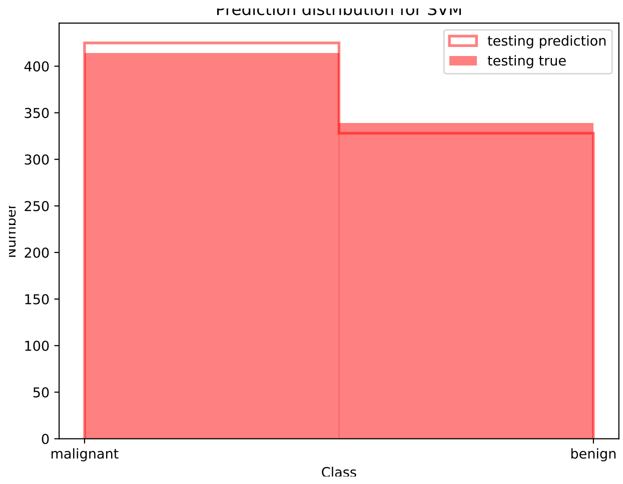

Figure 15: Distribution of predicted and true classes for the SVM model.

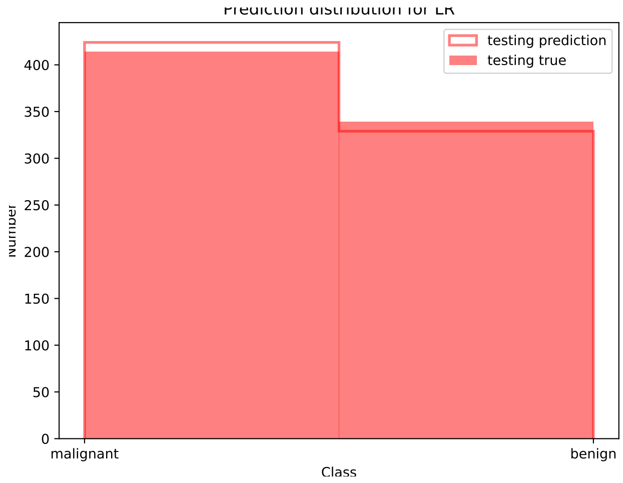

Figure 16: Distribution of predicted and true classes for the logistic regression model.

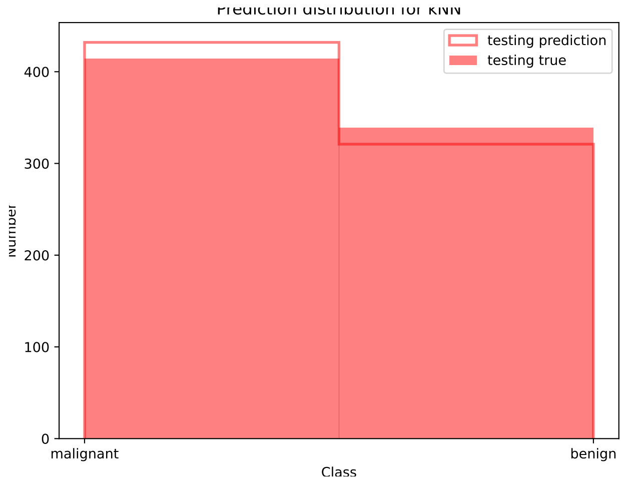

Figure 17: Distribution of predicted and true classes for the k-NN model.

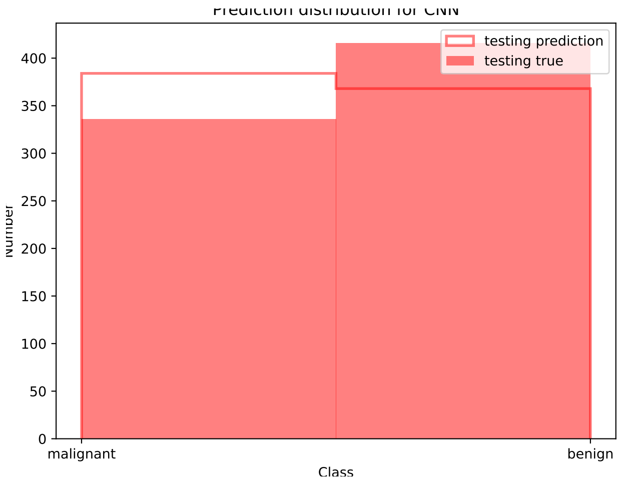

Figure 18: Distribution of predicted and true classes for the CNN model.

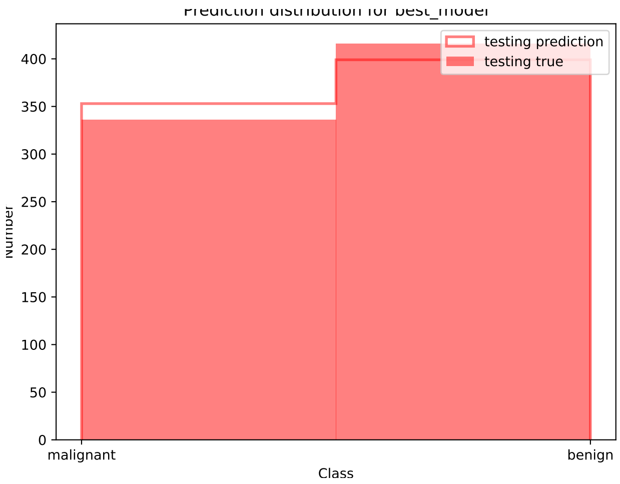

Figure 19: Distribution of predicted and true classes for the CNN model with tuned hyperparameters.