# CLASSIFYING BRAIN TUMORS FROM MRI IMAGES

Authors: Mary Chau, Aditya Dave, Michelle Shen

## Introduction

Glioblastoma is an aggressive type of brain cancer with the lowest survival rate out of all brain and spinal cord tumor types [1]. Glioblastoma can be treated in several ways, one of which is using chemotherapy. One way to predict whether a patient will have a favorable prognostic when undergoing chemotherapy for glioblastoma is to identify whether the tumor has MGMT promoter region methylation.

The promoter region of a gene contains sequences that tell RNA polymerase, the main transcription enzyme for transcribing DNA to RNA, as well as other transcription factors where to bind on DNA to initiate transcription. Methylating the promoter region of a gene prevents RNA polymerase from binding, limiting transcription and thus inhibiting expression of the MGMT gene. The MGMT gene codes for $O^6$-Methylguanine-DNA Methyltransferase (MGMT) enzyme, also known as the DNA repair enzyme. Left alone, MGMT enzyme repairs tumor cells destroyed by the toxic alkylating agents used in chemotherapy, rendering chemotherapy less effective in destroying tumor cells [2]. However, when the tumor's MGMT gene is methylated, production of MGMT enzyme is suppressed, allowing chemotherapy to destroy glioblastoma cells without interference from MGMT enzyme. Currently, detection of glioblastoma to identify promoter region methylation requires biopsy and PCR or genomic sequencing of sample tissue. Biopsy is an invasive procedure and results from PCR and sequencing are may not be immediately available. Recent literature suggests machine learning applications may be promising in allowing for MGMT promoter methylation detection from MRI imaging alone [3].

The objective of this study was to find the best binary classifier out of Bernoulli Naive Bayes, logistic regression, support vector machine (SVM), neural networks to train and evaluate models to predict for the presence of MGMT gene promoter methylation in glioblastoma tumors.

## Data

### Raw Data

This study was inspired by the RSNA-MICCAI Brain Tumor Radiogenomic Classification Kaggle competition [4], which provided 582 observations as MRI scans in Digital Imaging and Communications in Medicine (DICOM) format. Scans were taken from glioblastoma patients and labeled in accordance with whether the MGMT promoter region of the tumor was methylated. A typical observation contained several hundred images from the MRI scan types (FLAIR, T1w, T1wCE, T2w). Open-source code converting MRI scans from DICOM format to TF(TensorFlow)Record format and standardizing each type

of MRI scan to contain 32 images per observation were publicly available [5]. This study leveraged a simplified dataset generated from these open-source materials on Kaggle.

The simplified dataset retained 582 observations (465 training data points and 117 validation data points) from the original dataset. Each observation contained MRI images from 4 channels, each channel representing a different MRI scan type (FLAIR, T1w, T1wCE, T2w). Each channel had its own folder containing 32 images of the respective MRI scan, as shown in Figure 1. Each image was represented as a 128 x 128 matrix of floating-point values between 0 and 1. The target variables of the dataset were binary variables of 0 and 1, denoting whether MGMT promoter methylation was detected in the subject. The simplified dataset had a total of 306 positive cases, or 52% of the total number of observations.

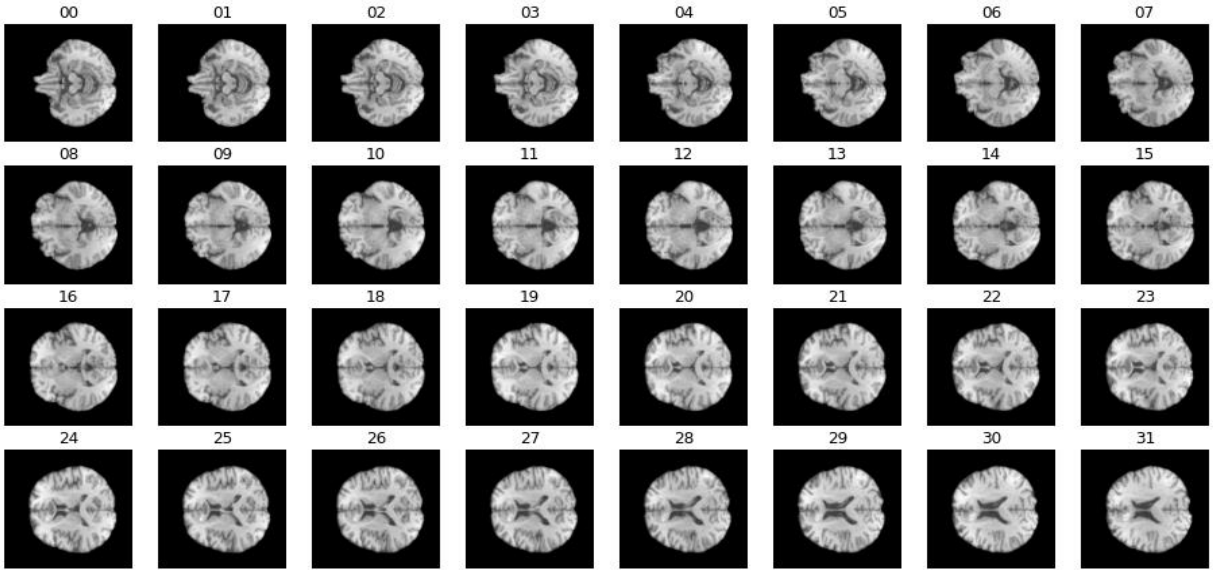


Figure 1: Sample Type T1w MRI Scan Images from One Obervation

## Image Augmentation

A sample size of 582 is small for machine learning algorithms like logistic regression, support vector machines and neural networks, which would benefit from more data points. A workaround found in existing literature working with brain tumor MRI images regarding the data scarcity problem suggested generating new images based on the existing images by rotating the images clockwise by 90° and flipping them vertically [6]. This study followed the same method to generate two new images for each observation, thus increasing the number of observations from 582 to 1746 observations.

Generated data were shuffled and split into training/development/test datasets using the 8:1:1 split, implying that 80% of the records were used for training and the remaining 20% was evenly split between both the development and test data. Applying image generation techniques resulted in a total of 1396 observations for training and 175 observations for the development and test.

## Data Dimension Reduction

Each observation has 4 MRI channels, each with 32 128x128 images equivalent to a total of 2,097,152 features per observation. Constrained by memory and compute capability available for this study,

dimensionality reduction was necessary to be able to apply any valuable machine learning algorithm to the dataset. The data was standardized with min-max scaler before applying Principal Component Analysis (PCA). Min-max scaler was selected to retain the value range of the original data, which were floating points between 0 and 1. The PCA, which was set to retain 100% of total variance and fitted on the training data, resulted in a dimension reduction from over 2 million features to just 1396 features. The same data transformation was also applied to the development and test data. The final datasets were saved as 32-bit floating point type to conserve runtime memory space.

## Approach/Methodology

Datasets were transformed from TFRecord files to NumPy array binary files that could be easily loaded into Python environment. Bernoulli Naive Bayes algorithm was used as a preliminary analysis to understand the data and to provide a baseline on accuracy, F1-score, and the area under the receiver operating characteristic curve (ROC AUC) measures achievable using a basic binary classifier. Three additional classifiers with increasing model complexity, logistic regression, support vector machine (SVM), and neural networks, were also selected. Each classifier was trained and refined with the goal of improving upon the performance of the initial Bayesian model.

All algorithms used the same training, development, and test datasets that had been preprocessed before injecting into the model training process. Figure 2 below illustrates the model development pipeline. Models were trained and fitted on training data and evaluated on development data using the following evaluation metrics: accuracy, F1-score, and ROC AUC. Development data was also used to tune each model's hyperparameters to optimize the accuracy score performance using 10-fold cross-validated grid-search. Top performing models from each algorithm were then evaluated on the three evaluation metrics using test data. The best-performing model for each algorithm was selected. Finally, algorithms were evaluated against each other based on their evaluation metrics scores.

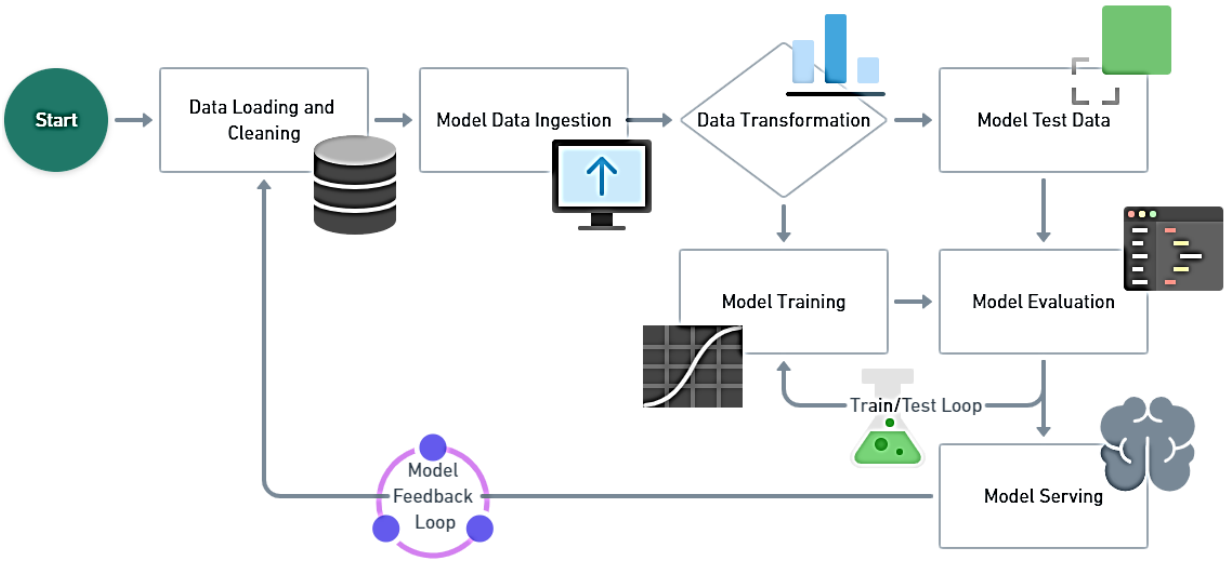


Figure 2: The general model-building and refining pipeline

## Evaluation Parameters

Three classification metrics were chosen to evaluate the models: Accuracy, F1 Score, and ROC AUC score. Accuracy was used as primary metric for tuning the model, while F1 Score and ROC AUC scores would provide substantiation to the core accuracy metric for each model.

Accuracy was the most intuitive classification metric. It is a ratio of the number of correct predictions divided by the total number of observations. This ratio may give a false sense of high accuracy if the target data is skewed or imbalanced (e.g. if a model simply predicts that it will always rain in Seattle, it will easily achieve a high accuracy). However, since our target result were well balanced (52% positive, 48% negative), accuracy was a valid choice of evaluation metric.

F1 Score is an essential metric for the real-life application of this study because classifying the potential for tumor patients to benefit from chemotherapy can mean the difference between life and death. F1 Score, the harmonic mean of precision and recall, provides a balanced evaluation between precision and recall, and allows the model to place equal importance on false positive (FP) and false negative (FN) cases. Both FP and FN cases may impact a patient's treatment outcomes and potential survival rate. FP cases would result in patients classified as having MGMT promoter methylation receiving chemotherapy despite not having favorable reactions to the chemotherapy. Vice versa, the FN cases would result in patients not receiving appropriate chemotherapy when it would have been effective for them. Hence, it was necessary to have a metric with equal weight on precision and recall.

Lastly, the ROC AUC score measures how well a classifier can differentiate classes and provides a useful metric to better understand the model. The metric was selected not only because it is the evaluation metric of the Kaggle competition that this study was motivated from, but also because it is invariant to class distribution and classification-threshold. The classification-threshold-invariance provides fair evaluation of model's prediction quality regardless of the classification threshold value chosen, allowing proper evaluation of models from different machine learning algorithms. Though class imbalance is not a big issue for the dataset used for this study, such characteristics would prevent the evaluation metric being affected by class imbalance if such a scenario did exist.

## Success & Failure Criteria

Success and failure were determined based on model accuracies. The initial expectation was to consider the study successful if any model reached at least 80% accuracy and to consider the study a failure if models were unable to reach 80% accuracy. These criteria were based on the idea that an accuracy lower than 80% would not be reliable if applied to a clinical setting where treatment decisions may be the difference between life and death. The model also needed to be easily scalable and yield a more accurate model if additional observations were added. However, the initial expectation was altered upon the completion of the Kaggle competition from which this study was inspired. The best performing submission to the competition had an ROC AUC score of 62.17%. The reason for low performance was due to several factors, such as the nature of the problem size of the dataset, number of features, and maturity of the data [7]. Therefore, success criteria were modified to redefine study success as the best model achieving at least 62% accuracy, F1, and ROC AUC scores, comparable to the performance level of the top-performing Kaggle submission.

# Experiments

The experiments in this study explored and tuned four types of machine learning algorithms: Bernoulli Naive Bayes, logistic regression, support vector machine (SVM), and neural networks. All experiments were conducted through Jupyter Notebook running on Google Colab Pro with Python 3.6.9 and used standard libraries such as NumPy (version1.19.5), Pandas (version 1.1.5), and scikit-learn (version 1.0.1) available in the Google Colab environment. The only exception was Neural Network, which used additional libraries: TensorFlow (version 2.7.0) and SciKeras (versions: 0.4.1).

## Bernoulli Naive Bayes

Bernoulli Naive Bayes was selected as the preliminary model because it was the fastest and most straightforward binary classifier out of the four target algorithms. A 10-fold cross-validated grid-search was performed using both training and development data to find the optimal parameters, the binarize threshold and alpha, with the highest accuracy score. The parameter grid included 11 alpha values spanning from 1.0e-10 to 25 and five binarize thresholds spanning equally from 0.1 to 0.9. The optimal parameter resulting from the grid-search had an accuracy of 0.56 with alpha set to 1.5 and binarize threshold set to 0.25.

## Logistic Regression

The logistic regression classifier was selected as a fast, scalable binary classifier. The algorithm defaults to minimizing the multinomial loss for all solvers except when data is binary or when the liblinear solver is selected, in which case the fit is binary. Training data was normalized before model fitting due to the need for scaled data using certain solvers. Consequently, all development and test data used were scaled to training data before applying the model.

Ten-fold cross-validated grid-search was performed on training and development data to tune the logistic regression model hyperparameters of C-value (inverse of regularization), solver (minimization of cost function), and penalty terms compatible with each solver. The solver was modeled using both L1, L2, and L1 ratio (Elastic-Net mixing) hyperparameters; however, it ultimately scored lower in accuracy than when using the liblinear solver so L1 ratio was disregarded for tuning. Because the liblinear solver was used, the intercept scaling parameter was briefly explored with no improvement to accuracy scoring. Final hyperparameters used were the liblinear solver with L2 penalty term and C-value of 1291.5 which had an uncontested accuracy score of 63%, F1 score of 66%, and ROC AUC score of 63% on development data.

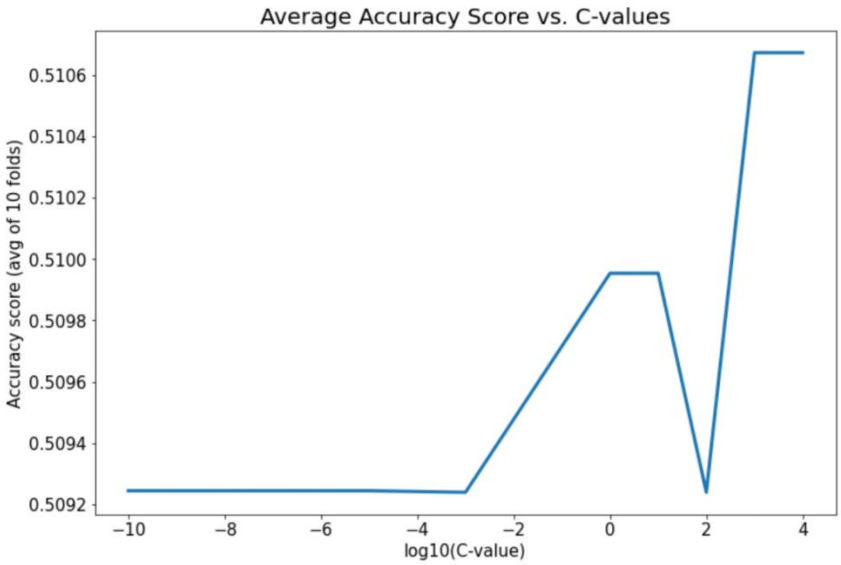


Figure 3: Average Accuracy Score vs Log of C-values

Because C-value was large, seven values, all multiples of 10 ranging from 1e-10 to 1e4, were selected for 10-fold cross-validated grid-search on C-value with fixed liblinear solver and L2 penalty term in order to quantify the differences in scoring accuracies using very small and very large C-values. Grid-search returned a 10x7 score matrix with accuracy scores per value per fold, with each row representing a fold and each column corresponding to one C-value. Then, the score matrix was averaged across columns and resulted in a 1x7 matrix representing the average accuracy score across all ten folds for each C-value tested. In Figure 3, C-values were plotted against averaged accuracy scores to visualize the change in average accuracy when increasing or decreasing C tenfold. It is evident that the average accuracy score has a difference of less than 0.15% when C-value changes from 1e-10 to 1e4. Final C-value was manually set to 1.0 to maximize accuracy while minimizing C-value. Scoring using these parameters on development data remained the same. Applied to test data, the tuned model returned an accuracy score of 63%, F1 score of 64%, and ROC AUC score of 64%. Logistic regression performed similarly on both development and test data after hyperparameter tuning on development data. Logistic Regression performed the same or better on development data for accuracy and F1-score; however, it performed better on test data for ROC AUC score.

## Support Vector Machine (SVM)

The support vector machine model was selected mainly due to its effectiveness in classifying data in high dimensions. Additionally, the SVM model is effective specifically when the number of dimensions for an observation exceeds the number of observations themselves in the dataset. When creating a baseline model, a regularization of 1 and a gamma value of 3 were used since they are the default hyperparameters for a support vector classifier. Fitting on the training data and evaluating on the development dataset, an accuracy of 64.0% was achieved and when evaluating on the test dataset, an accuracy of 62.0% was achieved. To begin experimenting with the SVM model, 3 metrics had to be optimized: C (regularization), gamma (influence) and the kernel (Linear, Radial Basis, etc.). To find the best parameters given the training data, a 10-fold cross-validated grid-search was performed using a

range of values for C and gamma spanning from 0.01 to 1000 by multiples of 10 for C and from 0.001 to 10 by multiples of 10 for gamma. A linear kernel was used due to the data linearity brought out by PCA.

Other kernels such as RBF (Radial Basis Kernel) were used to attempt classifying overlapping points. However, augmented data with dimensionality reduction deceased overlapping points, diminishing the performance of this kernel trick. The results of simply using the grid-search to find the optimal C and gamma did not improve the model's accuracy at all, although the grid-search resulted in a C of 0.01 and gamma of 0.001 as being the most performant hyperparameters.

Another consideration was to use a Bagging ensemble learning strategy, in which n SVM's (estimators) are trained independently of each other using randomly chosen training observations through bootstrapping. Bagging is a method to decrease the variance in the predictions by creating more training data from combinations and repetitions of the original data points. The number of estimators ranged from 5 to 100 by 5 inclusive for a total of 20 different Bagging executions. The base SVM model was used as the Bagging classifier's base estimator. The Bagging ensemble method yielded slightly higher accuracies for several of the estimator counts when the model was evaluated on the development dataset. For 5 estimators, an accuracy of 65.7% was achieved, which is roughly a 2.0% increase from the base model. When evaluating the model on the test dataset, the accuracy diminished by a roughly 2.0% from the base model evaluated on the test dataset from 62% to 60% when 5 estimators were used. The top performing support vector machines were for 5 and 25 estimators as shown in Table 1**.** The model was evaluated again on the test data to determine the more performant parameters. The model with the best results used 10 estimators and had an accuracy score of 62.86%. Since 10 estimators for the Bagging ensemble yielded the more performant model, it was chosen as the final SVM model.

| Number of Estimators | On Dev Data | | | On Test Data | | |
|---|---|---|---|---|---|---|
| | **Accuracy** | **F1 Score** | **ROC AUC** | **Accuracy** | **F1 Score** | **ROC AUC** |
| **5** | 0.6571 | 0.6537 | 0.6555 | 0.6057 | 0.6043 | 0.6075 |
| **10** | 0.6286 | 0.6277 | 0.6278 | 0.6286 | 0.6369 | 0.6297 |
| **25** | 0.6514 | 0.6497 | 0.6502 | 0.5829 | 0.5814 | 0.5846 |

Table 1: Evaluation Metrics of Top 3 Support Vector Machine Model Candidates

## Neural Networks

Due to limited memory and compute resources, a simple neural network with minimal number of layers was chosen over more sophisticated deep convolutional neural networks. The network architecture consisted of an input layer, a hidden layer, and an output layer, shown in Table 2. Several 10-fold cross-validated grid-searches were performed over parameters like number of neurons of the hidden layer, dropout rate, learning rate, batch size, optimizer, activation function, and weight initialization type to find the combination that yielded the highest accuracy.

Model: "Simple NN"

| Layer (type) | Output Shape | Param # |
|---|---|---|
| input_3 (InputLayer) | [(None, 1396)] | 0 |
| dense_4 (Dense) | (None, 64) | 89408 |
| dropout_2 (Dropout) | (None, 64) | 0 |
| dense_5 (Dense) | (None, 1) | 65 |

Total params: 89,473

Trainable params: 89,473

Non-trainable params: 0

Table 2: Summary of the Neural Network

The resulting neural network architecture had an input layer with 1369 features, a hidden layer of 64 neurons with SoftPlus activation function[1], uniform weight initialization, a dropout of 0.5 to filter, and an output layer to produce a binary result with a "Sigmoid" activation function. The model also used Adam optimizer with an exponential decay learning function using a learning rate of 0.01.

The final model was chosen from the 10 models produced by a 10-fold cross-validation using combined training and development datasets. The training used a batch size of 16 and had early stopping enabled, which would stop the training when accuracy stopped improving after 15 epochs.

The resulting top performing models were from fold #4 and fold #6. As shown in Table 3, both models tied in accuracy and ROC AUC scores when evaluated on development data. The F1 score of the fold #4 model was 4% higher than that of the fold #6 model. However, the fold #6 model had better loss. Hence, both models were evaluated again on the test data to determine the better one. The fold #6 model scored higher across all evaluation metrics when evaluated on test data. Therefore, the fold #6 model was selected as the optimal model for the neural network algorithm.

| | On Dev Data | | | | On Test Data | | | |
|---|---|---|---|---|---|---|---|---|
| | **Accuracy** | **F1 Score** | **ROC AUC** | **Loss** | **Accuracy** | **F1 Score** | **ROC AUC** | **Loss** |
| **Fold #4** | 1.0000 | 0.7476* | 1.0000 | 0.0006 | 0.6571 | 0.6471 | 0. 6569 | 15.04* |
| **Fold #6** | 1.0000 | 0.7064 | 1.0000 | 0.0005* | 0.6857* | 0.6821* | 0.6859* | 15.34 |

** indicates the better metric out of the two candidates.*

Table 3: Evaluation Metrics of Top 2 Neural Network Model Candidates

[1] softplus is very similar to the popular activation function, ReLu, as both produce output in scale of (0, $+\infty$). The main difference between the two is near x=0: the output of softplus has a smooth and gradual transition as x moves from negative to positive values, while the output of ReLu stays strictly as 0 when x is less than 0 and take sharp increase when x becomes greater than 0. Softplus preforms slightly better in the cross-validated grid search result, hence softplus activation function selected as the activation function of the hidden layer.

# Model Results & Comparison

Judging on the evaluation metrics calculated on test data used the best performing model of each of the four machine learning algorithms: Naïve Bayes, logistic regression, SVM, and neural networks. Algorithm performance improved uniformly across all evaluation metrics in the order of Naïve Bayes, SVM, logistic regression, and neural networks, as shown in Table 4. The baseline model, Naïve Bayes, had metrics values around 55%, while logistic regression and SVM both had metrics around low 60%, about a 6~10% improvement from the baseline result. Neural networks had the highest performance cross all metrics of about 68%, which was about 13% improvement from baseline results.

| Model Type | Accuracy | F1 Score | ROC AUC |
|---|---|---|---|
| **Naive Bayes** | 0.56 | 0.5443 | 0.5595 |
| **Logistic Regression** | 0.6343 | 0.6444 | 0.6356 |
| **SVM** | 0.6286 | 0.6369 | 0.6297 |
| **Neural Network** | 0.6857 | 0.6821 | 0.6859 |

Table 4 Model Result Comparison Over the Evaluation Metrics - Accuracy, F2 Score, and ROC AUC

Figure 4 below shows model performance in terms of ROC curves, allowing for visualization of the diagnostic ability for these binary classifiers. Models with curves closer to the top-left corners indicate better performance, while models with flatter curves, essentially a 45-degree diagonal line, indicate less desirable performance. Applying this interpretation to Figure 4, it is observed that the ROC curve improved in order of Naïve Bayes, SVM, logistic regression, then neural networks, which had the ROC curve closest to the top-left corner. Note that the sharp elbows of the ROC curves might be due less to the scarcity of the test dataset, which had 175 samples, but due more to the fact that the models predicted results as discrete binary values (0 or 1) instead of continuous probability values between the range of 0 and 1. Such model prediction behavior resulted in their ROC curves consisting of only 3 points: one assuming a threshold of 0, and one assuming a threshold of 1, and lastly, one assuming a threshold of Inference of the model.

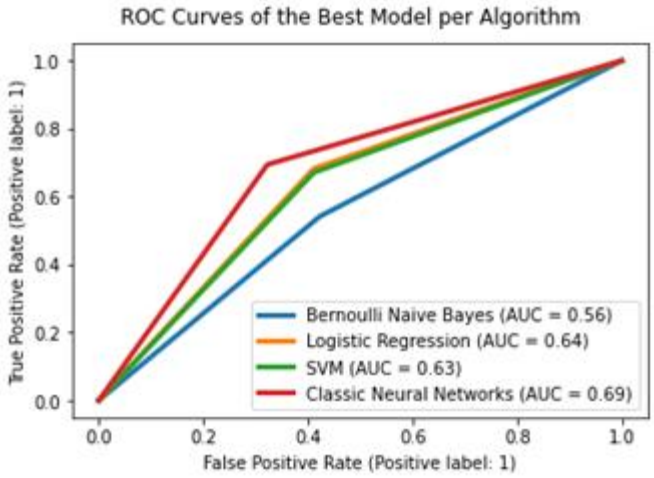


Figure 4: ROC Curves of the Best Performing Models of Each Algorithm

# Discussion

There were many tradeoffs between the four algorithms that were developed. Starting with the Naive Bayes classifier, tuning and training the preliminary model was fast and straightforward, but the accuracies returned from the evaluation were only slightly better than uniform random distribution. The model relied on a stronger independence of predictors assumption, so the interpretation should always be approached with caution, especially with the number of features in this dataset.

Logistic regression weights variables similar to a linear model. While certain logistic regression solvers can learn multinomial logistic regression models, the liblinear solver selected from grid-search uses a coordinate descent algorithm to fit a linear model for binary outcomes. The roughly 63% accuracy, F1, and ROC AUC scores were not unexpected, as the data in this study do not appear to have a strong linear relationship with the feature of interest. However, logistic regression has been successfully used for binary categorization of biomedical imaging data, so slightly better performance on determining methylation status may have been expected [8]. Also noteworthy was that the inverse of regularization strength has minimal effect in accuracy on this logistic regression model.

Next, the SVM model seemed to be a promising solution due to its ability to work well with the high dimensional data. However, the metrics from this model surpassed 60.0% and were similar to those achieved using logistic regression. One possible explanation for the less desirable performance of the SVM model was that training was performed on the PCA-reduced data. Better performance may have been observed if the SVM had been fit on the non-reduced dataset. A main issue with the SVM models was the training time, especially using ensemble techniques such as Bagging. Additionally, it was very difficult to visualize the support vectors because of the high number of predictors, making interpretation quite convoluted. With respect to increasing the accuracy of the model, combining the development and train data into a singular dataset allowed for a larger training dataset. However, using this implementation forfeits the tuning step to determine the optimal number of estimators from the Bagging ensemble.

The simple neural network was able to achieve the highest evaluation metrics at the cost of interpretability. This performance aligned with the intuition that a more complex model like Neural Networks that can capture non-linear relationships in the data may yield better results. However, this does not mean that increasing the number of neurons in the hidden layer would increase the model performance. Figure 5 illustrates that models with a larger number of neurons (500+) did not necessarily perform better than models with fewer neurons.

In addition, training neural networks is costly in time and computing resources. For the minimalistic neural network architecture and the dataset used in this study, it took many hours to tune the hyperparameters. Trade-offs between several hyperparameters made it difficult to parallelize the model tuning work. Using dropout rate and number of neurons shown in Figure 5 as an example, a higher number of neurons seemed to result in better performance with lower dropout rate, while a lower number of neurons seemed work better with higher dropout rate. Therefore, a very time-consuming exhaustive grid search through all hyperparameters may be necessary to find the absolute best combinations of all hyperparameters. Some combinations of hyperparameters like learning rate and number of neurons might result in much longer training times, but only result in a couple percentage points of improvement in evaluation metrics. Hence it was important to understand the trade-off

between model performance and computing/time constraints for model tuning and training to create an efficient model that satisfied the training constraints without sacrificing too much accuracy.

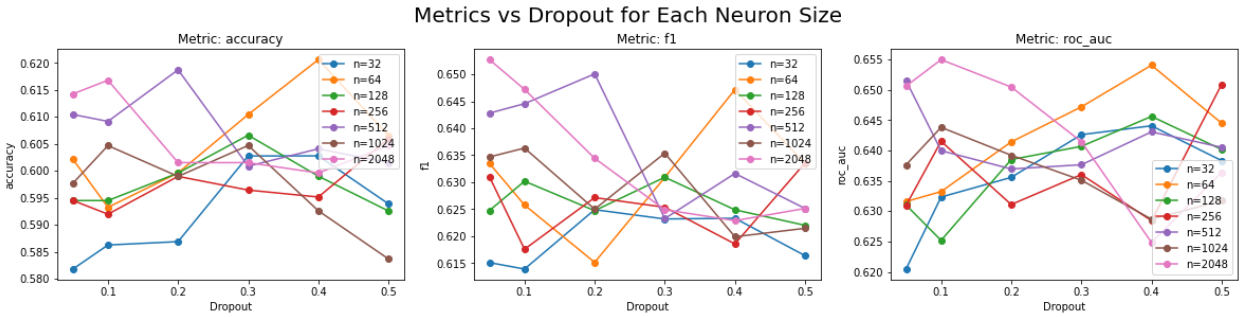


Figure 5: Evaluation Metrics vs Dropout Rate for Each Neuron Size

## Constraints

There were several constraints while working on this study in multiple domains, including the machines' raw computational power, level of machine learning application, and the lack of clinical subject matter expertise.

Since the initial dataset contained over 2 million floating points per observation, roadblocks emerged in the form of out of memory exceptions and session crashes in Google Colab and Jupyter notebook. Google Colab Pro subscriptions were purchased to increase runtime RAM from 12GB to 25GB. However, some high-memory operations, such as performing PCA on the dataset, were still crashing even with the significantly increased memory allocation. To overcome this challenge, an Azure Virtual Machine with 64GB of runtime memory was used to reduce the size of the dataset, which allowed the rest of the study to be conducted on Google Colab, which was more sustainable for prolonged use. However, the memory and computation strain also pushed other memory- and compute-intensive algorithms, like deep neural networks with 3D convolution, out of the reach of this study.

Machine learning models and techniques used for this study were limited to the coursework taught in the W207 Applied Machine Learning. Though personal research and reading of scientific studies were conducted to supplement coursework, a lack of experience in executing a full-on machine learning study posed as a constraint on the depth and breadth of the analysis covered. Due to limitations in modeling experience, only basic computer vision techniques were used; however, the overall analysis was performed using classification on floating point matrices.

Finally, another key constraint would be the lack of clinical knowledge. No information was provided on how MRI images were examined in real life to identify presence of MGMT promoter methylation. Supplemental readings and consultations with researchers' family and friends in the medical field provided little benefit, as detailed examination of this study's MRI scans typically requires experienced glioblastoma and radiology specialists. If this clinical knowledge were available, better intuition in model creation or feature extraction may have been available.

## Limitations of Study

Applying PCA to the original dataset enabled the study to be conducted with limited computing resources. However, this dimension reduction technique relied on the linear relationship in the dataset and removed nonlinear relationships that may have existed in the original dataset. The linearity imposed by PCA on the dataset benefited algorithm like logistic regression, but it might have made algorithms like SVM and neural networks that worked well in capturing non-linear decision boundaries less effective than they could have been. This study was unable to analyze the effect of applying a non-linear dimension reduction technique.

In addition, this particular research problem is an unusual application of machine learning in which the predicted feature, MGMT gene promoter methylation, is not a directly visible imaging finding, but rather is a genomic biomarker determined by molecular analysis of biopsied specimens [7]. Studies documented in prior literature explore the potential for MGMT promoter methylation detection using machine learning applications. While one study achieved an average accuracy score 87.7% on test data using an L1-regularized neural network, this study used a sample size of 59, which was ten times smaller than the size of our original dataset [9] [10]. A study with slightly greater numbers of subjects did not see the same levels of success with an accuracy score of 62% using neural networks [11], while still another study suggests that MGMT promoter methylation detection using radiomics is not highly reliable [12]. Therefore, there are no known solutions to this problem that have had success in real-world clinical applications.

# Conclusion

Based on the study, neural networks was the best binary classifier out of Bernoulli Naive Bayes, logistic regression, support vector machine, and neural networks in predicting a glioblastoma tumor's MGMT gene promoter region methylation status from MRI scans.

The metrics of the models generated from this study show that our best-performing Neural Networks model scored well below 80% in all metrics, and too low to be viable in a clinical context where high accuracy and F1 scores are extremely important to the wellbeing of the patient. We did not meet the initial objective of achieving 80% accuracy, F1, and ROC AUC scores. However, our Neural Networks model scored over 68% in all evaluation metrics and was able to match and surpass the Kaggle competition first place leaderboard ROC AUC score of 62.17%. We acknowledge that while our model may have benefitted from advantages such as not being confined to the Kaggle development environment, we consider matching the competition leaderboard score a success.

## Future Work

In the future, input from medical professionals and having a clinical expert on the study team could be highly beneficial. With respect to improved algorithms and techniques, more advanced computer vision and other dimensionality reduction techniques, such as Kernel PCA or Autoencoder, that can preserve the nonlinear relationship in the original data could be utilized for more accurate classifications. It would also be very interesting to apply an ensemble method involving multiple algorithms to this problem. Additionally, over time more data may be made available for the study, improving the baseline accuracies that the models would yield.

# References


[1] Q. Ostrom, G. Cioffi, H. Gittleman and et al, "CBTRUS statistical report: Primary brain and other central nervous system tumors diagnosed in the United States in 2012–2016," *Neuro-Oncol,* vol. 21, no. Suppl 5, p. v1–v100, 2019.

[2] "MGMT Promoter Methylation, Tumor," Mayo Foundation for Medical Education and Research, [Online]. Available: https://www.mayocliniclabs.com/test-catalog/Clinical+and+Interpretive/36733. [Accessed 5 12 2021].

[3] I. Levner, S. Drabycz, G. Roldan, P. De Robles, J. G. Cairncross and R. Mitchell, "Predicting MGMT methylation status of glioblastomas from MRI texture," *Med Image Comput Comput Assist Interv,* vol. 12, no. Pt 2, pp. 522-30, 2009.

[4] U. Baid and et al., "The RSNA-ASNR-MICCAI BraTS 2021 Benchmark on Brain Tumor Segmentation and Radiogenomic Classification," *arXiv:2107.02314,* 2021.

[5] K. Shahhosseini, "RSNA Brain Tumor Classification TFRecords," 2021. [Online]. Available: https://www.kaggle.com/kavehshahhosseini/rsna-brain-tumor-classification-tfrecords.

[6] M. M. Badža and M. Č. Barjaktarović, "Classification of Brain Tumors from MRI Images," *Applied Sciences,* vol. 10, no. 6, p. 1999, 15 March 2020.

[7] J. Mongan, "On the low performance of models in this challenge," 28 Oct 2021. [Online]. Available: https://www.kaggle.com/c/rsna-miccai-brain-tumor-radiogenomic-classification/discussion/284024. [Accessed 04 12 2021].

[8] L. Zammataro, "Logistic Regression for malignancy prediction in cancer - Applying Logistic Regression to the Wisconsin Breast Cancer (Diagnostic) Data Set," 22 12 2019. [Online]. Available: https://towardsdatascience.com/logistic-regression-for-malignancy-prediction-in-cancer-27b1a1960184. [Accessed 5 12 2021].

[9] S. Drabycz, G. Roldán, P. de Robles, D. Adler, J. B. McIntyre, A. M. Magliocco, J. G. Cairncross and J. R. Mitchell, "An analysis of image texture, tumor location, and MGMT promoter methylation in glioblastoma using magnetic resonance imaging," *NeuroImage,* vol. 49, no. 2, p. 1398–1405, 2010.

[10] P. Korfiatis, T. L. Kline, L. Coufalova, D. H. Lachance, I. F. Parney, R. E. Carter, J. C. Buckner and B. J. Erickson, "MRI texture features as biomarkers to predict MGMT methylation status in glioblastomas," *Medical physics,* vol. 43, no. 6, p. 2835–2844, 2016.

[11] L. Han and M. R. Kamdar, "MRI to MGMT: predicting methylation status in glioblastoma patients using convolutional recurrent neural networks," *Biocomputing 2018,* pp. 331-342, 2018.

[12] E. Calabrese, . J. E. Villanueva-Meyer and S. Cha , "A fully automated artificial intelligence method for non-invasive, imaging-based identification of genetic alterations in glioblastomas," *Scientific Reports,* vol. 10, no. 11852, 2020.