## 1. Introduction

Brain tumors are among the most life-threatening forms of cancer. Accurate early diagnosis using MRI imaging is crucial for effective treatment. In this project, we developed a machine learning model using traditional techniques such as Random Forest, PCA (Principal Component Analysis), and feature selection to classify brain MRI images into four categories: Glioma, Meningioma, Pituitary, and No Tumo**r.**

## 2. Dataset Description

The dataset used in this study is a publicly available MRI brain tumor dataset that includes Training and Testing folders, each containing images from four classes:

1) Glioma Tumor
2) Meningioma Tumor
3) Pituitary Tumor
4) No Tumor

Each class consists of grayscale MRI images with varying orientations (axial, sagittal, coronal) and resolutions. The dataset is moderately balanced across classes.

- **Feature Extraction**

Each image was converted to grayscale and resized to 128x128. From each image, we extracted:

- Mean pixel intensity
- Standard deviation
- Variance
- Histogram with 32 bins

- **Descriptive Statistics (from extracted features)**

Using the pandas library, statistical features such as mean pixel intensity, standard deviation, variance, and histogram bins were extracted from each image. A summary of these features is shown below:

Image Count Per Class (Training Set):
- Glioma: 826
- Meningioma: 822
- Pituitary: 827
- No Tumor: 395

Feature Summary (Mean Intensity, Std Dev, etc.):
- Mean Intensity: Mean ≈ 119.2, Std ≈ 35.6
- Std Dev: Mean ≈ 29.8, Std ≈ 12.4
- Histogram Bins (32): varied values showing brightness distribution

## 3. Source of the Dataset

The dataset was sourced from Kaggle:

Dataset Link: https://www.kaggle.com/datasets/masoudnickparvar/brain-tumor-mri-dataset

## 4. Objective

The main objective is to classify brain MRI images into one of the four categories: Glioma, Meningioma, Pituitary, or No Tumor.

This is a multi-class classification problem using traditional machine learning algorithms. The goal is to:

- Achieve high classification accuracy
- Explore the impact of feature extraction, feature selection, and dimensionality reduction on model performance

## 5. Suitable Machine Learning Algorithm

We explored the following algorithms:

1. Random Forest

An ensemble learning method that builds multiple decision trees and outputs the mode of their predictions. It's robust to noise and works well with tabular feature data.

2. SVM (Support Vector Machine)

SVM tries to find the optimal hyperplane to separate data into classes. It performed suboptimally in our case due to overlapping feature distributions.

3. KNN and KMeans

Not used in final evaluation due to the overlapping nature of MRI features, which makes distance-based methods less reliable.

## 6. Why Random Forest Was Suitable

1) Performs well on non-linearly separable and overlapping data
2) Can handle high-dimensional feature space without scaling
3) Provides feature importance metrics for selection
4) Works well with small to medium-sized datasets

## 7. Feature Engineering and PCA

We applied the following steps:

1) Extracted features: mean, std, variance, histogram (32 bins)
2) Selected top 20 features using Random Forest importance (feature selection)
3) Applied PCA (Principal Component Analysis) to reduce to 10 components

This reduced the model's complexity while retaining over 90% of the original variance.

## 8. Model Performance

| Model Version | Accuracy (%) | Notes |
|---|---|---|
| Random Forest (Raw Features) | ~91–93% | Strong baseline |
| Random Forest + PCA (10 comp.) | 90.85% | Good dimensionality reduction |
| Feature Selection + PCA + RF | 90.01% | Leaner model, slightly lower accuracy |
| 5-Fold Cross-Validation Avg | 86.53% | Realistic estimate of generalization |

All models were evaluated using accuracy, classification reports, and confusion matrices. PCA scatter plots also helped visualize feature overlap.

## 9. Research Papers Using This Dataset

1. Chakrabarty et al. (2022)

Conducted a comparative study of traditional ML models (SVM, Random Forest) and deep learning approaches (CNNs). Their findings confirmed that while CNNs achieved the highest accuracy (~94–96%), Random Forests provided strong performance (~88–90%) and remained a viable solution for resource-constrained applications.

- Title: Comparative Study of ML and DL Models for Brain Tumor Detection
- Approach: Compared SVM, KNN, Random Forest, and CNNs on the same dataset
- Outcome: Random Forest achieved ~88%, CNNs performed best overall
- Link: https://www.sciencedirect.com/science/article/abs/pii/S1878875024005382

2. Masoud Nickparvar et al. (2021)

Applied CNN and transfer learning to brain tumor classification and achieved around 97% accuracy, showing the potential of deep learning models when trained end-to-end with sufficient data.

- Title: Brain tumor detection using CNN and transfer learning
- Approach: Used Convolutional Neural Networks (CNNs) for end-to-end learning

- Outcome: Achieved ~97% accuracy using deep features
- Link: https://arxiv.org/abs/1802.10200

3. Afshar et al. (2020)

Proposed a Capsule Network (CapsNet)-based framework for classifying brain tumor types. The study highlighted that CapsNet could outperform conventional CNNs, especially in terms of preserving spatial hierarchies in image features. The model achieved accuracy in the 90–93% range.

- Title: Brain Tumor Type Classification via Capsule Networks
- Approach: Used advanced deep learning (CapsNet) for improved spatial reasoning
- Outcome: Achieved ~90–93% accuracy
- Link: https://www.researchgate.net/publication/347496862_BayesCap_A_Bayesian_Approach_to_Brain_Tumor_Classification_Using_Capsule_Networks

These papers highlight the potential of deep learning, but also validate Random Forest as a strong traditional ML baseline, especially when computational resources are limited.

## Conclusion

This project successfully demonstrates the use of traditional machine learning techniques for brain tumor classification. Despite the availability of deep learning methods, a well-engineered Random Forest model with feature selection and PCA achieved high performance while remaining computationally efficient and interpretable.

**Tools Used:** Python, Google Colab, Matplotlib, Pandas

**Google Colab Notebook:**
https://colab.research.google.com/drive/1c2Nvyc1Eo_MpjBFT0sTeHYbzOhr3c6Kd?usp=sharing

**Future Work:** Integrating CNN for improved accuracy.