Progress Report

1. Introduction

Recap of Project Overview:
In this project, I am working to predict patient recovery time using MRI scans combined with demographic data and medical history. The idea is to build a model that can forecast how long patients might take to recover from their injuries, which could be very useful in a clinical setting. By integrating high-dimensional imaging data with information like age, sex, and concussion history, I hope to uncover patterns that relate to recovery outcomes.

Motivation:
The primary aim of this project is to develop a predictive model that estimates the recovery time (in days) for patients based on their blood flow MRI (data and demographic/medical features). By achieving this, we aim to provide valuable insights into patient recovery and improve medical decision-making processes.

Research Questions:
The key question driving this project is: Can we accurately predict recovery time based on blood flow MRI data, demographic features (e.g., age, gender), and medical history (e.g., history of concussion)? This project involves leveraging a deep learning approach to analyze blood flow MRI images and other structured data, enabling the development of a robust predictive tool.

Some of the other questions driving me to the key questions include:

- Can MRI scans, along with demographic and medical data, accurately predict patient recovery time?
- Does combining imaging features with demographic data enhance the prediction performance?
- Which specific features are most closely associated with prolonged recovery times?

2. Data Summary

Datasets:
For this project, I am using two main datasets. The first dataset, **clean_output.csv**, contains patient metadata such as age, sex, history of concussion (HOC), days since injury (dInj), and days of recovery (dRecov), along with summary statistics for MRI scans and corresponding file names. The second dataset, **cleaned_output.h5**, consists of preprocessed MRI data stored as NumPy arrays, each indexed by a unique subject_id.

Key EDA Findings:
Through exploratory data analysis (EDA), I have uncovered several important insights. I looked at the variation in recovery time across different categories, including age, HOC, and sex. The recovery time distribution is highly skewed, which may impact model performance. These findings have helped guide my approach to feature engineering and model development, and they underscore the need for further analysis to capture the complex relationships between these variables and recovery time.

It also revealed that there are no major issues with missing values or data alignment, providing a solid and consistent foundation for further investigation. The demographic data shows that the age of subjects is concentrated in the late teens to mid-twenties, indicating a relatively homogeneous cohort. Moreover, both sex and history of concussion (HOC) are roughly balanced between their respective categories (male/female, yes/no for concussion), which minimizes concerns related to class imbalance.

In terms of recovery time (dRecov), the distribution is notably right-skewed, with a few outliers extending to high values, such as over 300 days. This skewness suggests that while most patients recover within a relatively short period, there are exceptional cases that may need special attention. For the MRI data, applying a brain mask proved to be beneficial as it removed non-brain regions, thereby mitigating skew and reducing the impact of outliers that arise from large areas of zero or near-zero intensities.

Dimensionality reduction using PCA further highlighted the benefits of masking. Before masking, the PCA results were dominated by outliers, resulting in a tight cluster of points with a few extreme cases. After masking, however, the points were more evenly distributed in the principal component space, suggesting that this approach captures more meaningful anatomical variation. Additionally, the pairplot analysis confirmed that demographic features such as sex, HOC, and days since injury do not exhibit strong linear correlations with age or each other, which reduces concerns about multicollinearity in subsequent modeling efforts.

3 Modeling Approach

In this project, I have implemented multiple models to predict recovery time based on fMRI data and clinical features. My initial models served as baselines, while later models incorporated more advanced techniques, such as K-Fold cross-validation and hyperparameter tuning, to improve performance.

Model 1: Neural Network with PyTorch

The first model I implemented was a simple neural network using PyTorch. This model consists of multiple fully connected layers with dropout regularization and ReLU activation. It was trained using the Adam optimizer and Mean Squared Error (MSE) loss function. The training process involved splitting the dataset into training and testing sets, followed by iteratively optimizing the network over 50 epochs.

Despite these efforts, Model 1 produced a high RMSE, indicating poor predictive performance. This suggested that the model either lacked the necessary complexity to capture patterns in the data or required more extensive hyperparameter tuning.

Model 2: Random Forest

To compare deep learning with traditional machine learning, I implemented a Random Forest model. This ensemble learning method leverages multiple decision trees to improve predictive accuracy. However, like Model 1, this approach resulted in a relatively high RMSE, suggesting that it struggled to fully utilize the complex MRI features.

Given the poor performance of these initial models, I shifted my focus toward more advanced deep learning techniques with systematic hyperparameter tuning.

Model 3: K-Fold Cross-Validation with GridSearchCV

To improve model reliability and avoid overfitting, I implemented **Model 3**, which uses K-Fold cross-validation. This approach trains the neural network on different subsets of the data and evaluates performance across multiple validation sets. Additionally, I introduced **GridSearchCV**, which allows me to systematically explore different combinations of hyperparameters, including:

- **Neurons per layer**: 64 or 128
- **Dropout rate**: 0.2 or 0.5
- **Learning rate**: 0.001 or 0.01

After testing multiple configurations, Model 3 achieved an **RMSE of approximately 58.09 days** with an optimal set of hyperparameters (**64 neurons, 0.5 dropout rate, and a learning rate of 0.001**). While this was an improvement over the earlier models, there was still room for further optimization.

Model 4: Hyperparameter Tuning with Additional Regularization

Building on the findings from Model 3, **Model 4** introduced additional hyperparameter tuning, including:

- **More fine-tuned neuron values** (64, 80, 96, 128)
- **More granular dropout rates** (0.2, 0.3, 0.5)
- **Learning rates ranging from 0.0005 to 0.005**
- **Weight decay (L2 regularization)**
- **Number of layers** (2 or 3)
- **Extended training epochs** (100 or 150)

Through this refined tuning process, Model 4 achieved a **best RMSE of 55.14 days**, further improving upon the previous models. The optimal configuration included **96 neurons, a dropout rate of 0.3, a learning rate of 0.005, weight decay of 1e-05, and training for 150 epochs**.

These results show that our hyperparameter tuning process is working well to reduce the prediction error. In our initial experiment (Model 3), using K-Fold cross-validation, we found that the best configuration with 64 neurons, a dropout rate of 0.5, and a learning rate of 0.001 produced an average RMSE of about 58.09 days. This means that, on average, our model's prediction of recovery time was off by roughly 58 days.

After further tuning with additional parameters such as weight decay, different numbers of neurons, layers, and epochs, we achieved even better performance. The best result came from a configuration with 96 neurons, a dropout rate of 0.3, a learning rate of 0.005, a weight decay of 1e-05, using 2 layers and training for 150 epochs, which yielded an RMSE of approximately 55.14 days.

In summary, by systematically tuning the hyperparameters, we managed to lower our error margin, suggesting that the refined model can predict recovery time more accurately. This improvement demonstrates the value of hyperparameter tuning in capturing the complex relationships in high-dimensional fMRI data combined with clinical features.

Key Takeaways from Model Comparisons

1. **Baseline Models Performed Poorly**
   - Both the initial neural network (Model 1) and the Random Forest model (Model 2) produced high RMSE values, indicating that they were not effectively capturing the complexity of the data.

2. **Cross-Validation and Hyperparameter Tuning Improved Performance**
    - Model 3 (K-Fold with GridSearchCV) significantly reduced error by systematically exploring different parameter configurations.
    - Model 4 (Further Tuning with Regularization) yielded the best results, demonstrating the importance of weight decay and more refined hyperparameter selection.

4. Challenges and Next Steps

Although Model 4 demonstrated significant improvements, achieving an RMSE of 55.14 days, this value remains high for clinical applicability. To enhance predictive accuracy and make the model more reliable, I will focus on several key areas, including better handling of outliers, refining feature representations, improving model architecture, and further tuning hyperparameters.

One of the biggest challenges in modeling recovery time is the highly skewed distribution of the target variable (dRecov). Extreme outliers, ie patients with recovery times far beyond the majority—can distort model predictions and increase RMSE. To address this, I will experiment with log transformation or other normalization techniques to reduce the impact of extreme values. Additionally, I plan to test alternative loss functions such as Huber Loss, which is less sensitive to outliers than the commonly used Mean Squared Error (MSE). These approaches should help the model generalize better and make more stable predictions. Currently, the MRI features in my dataset are extracted from whole-brain intensity distributions. However, certain brain regions may be more predictive of recovery time than others. To refine feature extraction, I will apply Region-of-Interest (ROI) segmentation, focusing on specific brain areas, such as the frontal and temporal lobes, that may have stronger clinical relevance. By computing summary statistics (e.g., mean, variance) within each ROI, I aim to enhance the model's ability to detect meaningful patterns that contribute to recovery time.

While fully connected neural networks (FCNs) can capture complex relationships in the data, they may not fully utilize the spatial information inherent in MRI scans. Convolutional Neural Networks (CNNs), on the other hand, are well-suited for analyzing spatial structures in imaging data. Therefore, I will implement CNN-based models that process 3D fMRI scans directly. By comparing CNN performance with my existing FCN models, I hope to determine whether spatial patterns in MRI scans can provide additional predictive power.

Although Principal Component Analysis (PCA) has been useful for dimensionality reduction, it may not be the best method for capturing non-linear relationships in MRI data. As an alternative, I plan to experiment with t-SNE (t-distributed Stochastic Neighbor Embedding) and UMAP (Uniform Manifold Approximation and Projection) to explore whether different recovery profiles emerge when visualizing MRI features in a lower-dimensional space. If clusters corresponding to distinct clinical patterns are found, I will use these features to enhance my predictive models.

Hyperparameter tuning played a crucial role in improving Model 4's performance, demonstrating the importance of optimizing parameters like dropout rate, learning rate, and weight decay. While I previously used GridSearchCV, I will now explore Bayesian Optimization, a more efficient technique for navigating high-dimensional parameter spaces. Additionally, I will test adaptive learning rates, which allow the model to adjust its training speed dynamically based on convergence patterns. These optimizations should help the model learn more efficiently while avoiding overfitting.

Currently, my models process MRI and demographic features together, but a more structured approach to integrating these data sources may improve performance. One potential solution is to build a multi-input model, where CNNs extract imaging features while a separate neural network processes clinical and demographic data. These two branches would then be merged before the final prediction layer, allowing the model to make better use of both types of information. This fusion approach could improve generalization and predictive accuracy.

5. Appendix

Snip of the result

Here's a snip of the result, full code is avaliable on github:

```
if __name__ == "__main__":

    X_combined, y = load_and_prepare_data('cleaned_output.csv', 'brain_overlayed_new.h5')

    # Run Model 3 (K-Fold CV with GridSearchCV)
    model3_rmse = model3(X_combined, y)
    print(f"Model 3 RMSE: {model3_rmse}")
```

```
Running Model 3: K-Fold Cross-Validation with PyTorch...
Training with neurons=64, dropout_rate=0.2, learning_rate=0.001
Epoch [10/50], Loss: 20276.6414
Epoch [20/50], Loss: 16265.1587
Epoch [30/50], Loss: 6490.2791
Epoch [40/50], Loss: 5100.2111
Epoch [50/50], Loss: 3664.1124
```

```
Fold 1, RMSE: 58.99284744262695
Epoch [10/50], Loss: 16862.3271
Epoch [20/50], Loss: 19716.2688
Epoch [30/50], Loss: 9087.2522
Epoch [40/50], Loss: 6594.4599
Epoch [50/50], Loss: 6652.3967
Fold 2, RMSE: 63.448673248291016
Epoch [10/50], Loss: 12629.6711
Epoch [20/50], Loss: 6264.5955
Epoch [30/50], Loss: 5806.6519
Epoch [40/50], Loss: 3285.1958
Epoch [50/50], Loss: 2643.7082
Fold 3, RMSE: 93.34349060058594
Epoch [10/50], Loss: 27332.9478
Epoch [20/50], Loss: 17970.0107
Epoch [30/50], Loss: 17378.7285
Epoch [40/50], Loss: 8668.6591
Epoch [50/50], Loss: 10027.6521
Fold 4, RMSE: 36.986846923828125
Epoch [10/50], Loss: 68122.9775
Epoch [20/50], Loss: 12435.8416
Epoch [30/50], Loss: 8093.8364
Epoch [40/50], Loss: 6136.7468
Epoch [50/50], Loss: 7385.2642
Fold 5, RMSE: 48.890933990478516
Avg RMSE for neurons=64, dropout_rate=0.2, learning_rate=0.001: 60.33255844116211
Training with neurons=64, dropout_rate=0.2, learning_rate=0.01
Epoch [10/50], Loss: 42513.7905
Epoch [20/50], Loss: 22795.3855
Epoch [30/50], Loss: 13326.1963
Epoch [40/50], Loss: 16072.4006
Epoch [50/50], Loss: 22045.8499
Fold 1, RMSE: 48.913021087646484
Epoch [10/50], Loss: 95480.2451
Epoch [20/50], Loss: 11673.5508
Epoch [30/50], Loss: 9050.8478
Epoch [40/50], Loss: 12631.6567
Epoch [50/50], Loss: 11503.8414
Fold 2, RMSE: 73.2386474609375
Epoch [10/50], Loss: 258736.9336
Epoch [20/50], Loss: 15485.4111
Epoch [30/50], Loss: 7152.9576
Epoch [40/50], Loss: 6540.6458
Epoch [50/50], Loss: 8095.0743
Fold 3, RMSE: 86.30178833007812
Epoch [10/50], Loss: 390350.6172
Epoch [20/50], Loss: 33568.3809
Epoch [30/50], Loss: 24154.3047
Epoch [40/50], Loss: 16760.2864
Epoch [50/50], Loss: 28553.6162
Fold 4, RMSE: 46.999755859375
```

```
model4_rmse = model_with_tuning(X_combined, y)
print(f"Model 4 RMSE: {model4_rmse}")
```

```
Running Model with Further Hyperparameter Tuning...
Training with neurons=64, dropout_rate=0.2, learning_rate=0.0005, weight_decay=0.0, num_layers=2, epochs=100
Epoch [10/100], Loss: 9166.6407
Epoch [20/100], Loss: 5711.6373
Epoch [30/100], Loss: 6046.6249
Epoch [40/100], Loss: 6292.8608
Epoch [50/100], Loss: 3266.7817
Epoch [60/100], Loss: 3131.8219
Epoch [70/100], Loss: 3309.0095
Epoch [80/100], Loss: 3796.9489
Epoch [90/100], Loss: 4521.7365
Epoch [100/100], Loss: 3735.5137
Fold 1, RMSE: 63.81948471069336
Epoch [10/100], Loss: 12176.0835
Epoch [20/100], Loss: 8450.9704
Epoch [30/100], Loss: 5848.2982
Epoch [40/100], Loss: 4235.7450
Epoch [50/100], Loss: 1817.0803
Epoch [60/100], Loss: 6538.1494
Epoch [70/100], Loss: 1706.3610
Epoch [80/100], Loss: 2191.0188
Epoch [90/100], Loss: 3514.7665
Epoch [100/100], Loss: 3509.8644
Fold 2, RMSE: 77.2696762084961
Epoch [10/100], Loss: 9744.3691
Epoch [20/100], Loss: 5762.5269
Epoch [30/100], Loss: 5863.4464
Epoch [40/100], Loss: 5965.6083
Epoch [50/100], Loss: 3368.4945
Epoch [60/100], Loss: 3036.0735
Epoch [70/100], Loss: 2172.2342
Epoch [80/100], Loss: 1243.2361
Epoch [90/100], Loss: 2919.9233
Epoch [100/100], Loss: 1941.6379
Fold 3, RMSE: 92.50481414794922
Epoch [10/100], Loss: 18527.5024
Epoch [20/100], Loss: 12260.4685
Epoch [30/100], Loss: 7923.8274
Epoch [40/100], Loss: 16006.7214
Epoch [50/100], Loss: 3268.8077
Epoch [60/100], Loss: 2273.7982
Epoch [70/100], Loss: 2895.2681
Epoch [80/100], Loss: 1069.1440
Epoch [90/100], Loss: 4552.7239
Epoch [100/100], Loss: 5738.2180
Fold 4, RMSE: 41.42780303955078
Epoch [10/100], Loss: 12745.8208
Epoch [20/100], Loss: 10761.9856
Epoch [30/100], Loss: 6322.2347
Epoch [40/100], Loss: 4491.1542
Epoch [50/100], Loss: 4806.7697
Epoch [60/100], Loss: 3351.1547
Epoch [70/100], Loss: 4922.5372
Epoch [80/100], Loss: 2607.3788
Epoch [90/100], Loss: 2965.4514
Epoch [100/100], Loss: 2422.7382
```