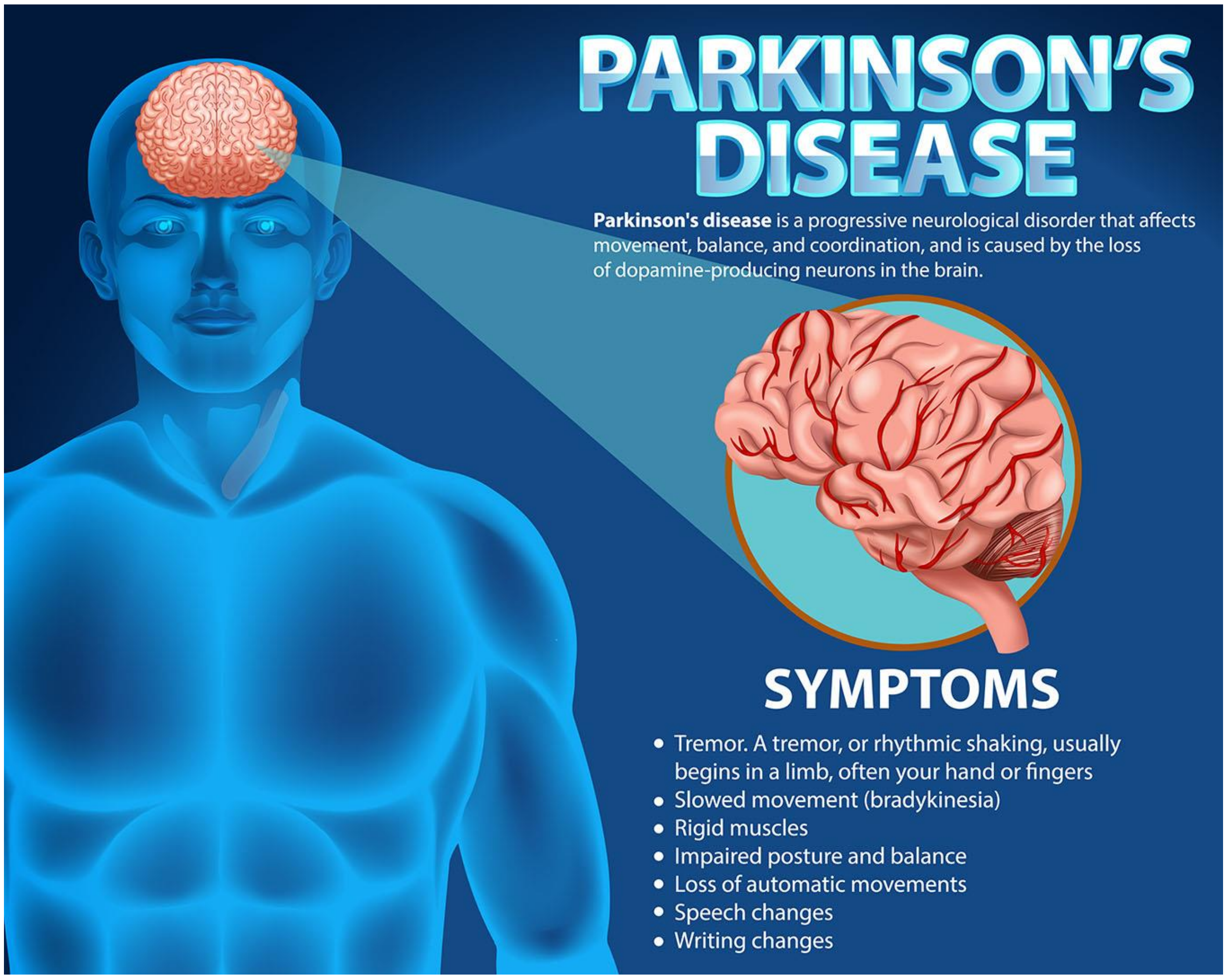
Parkinson's Disease

MS2

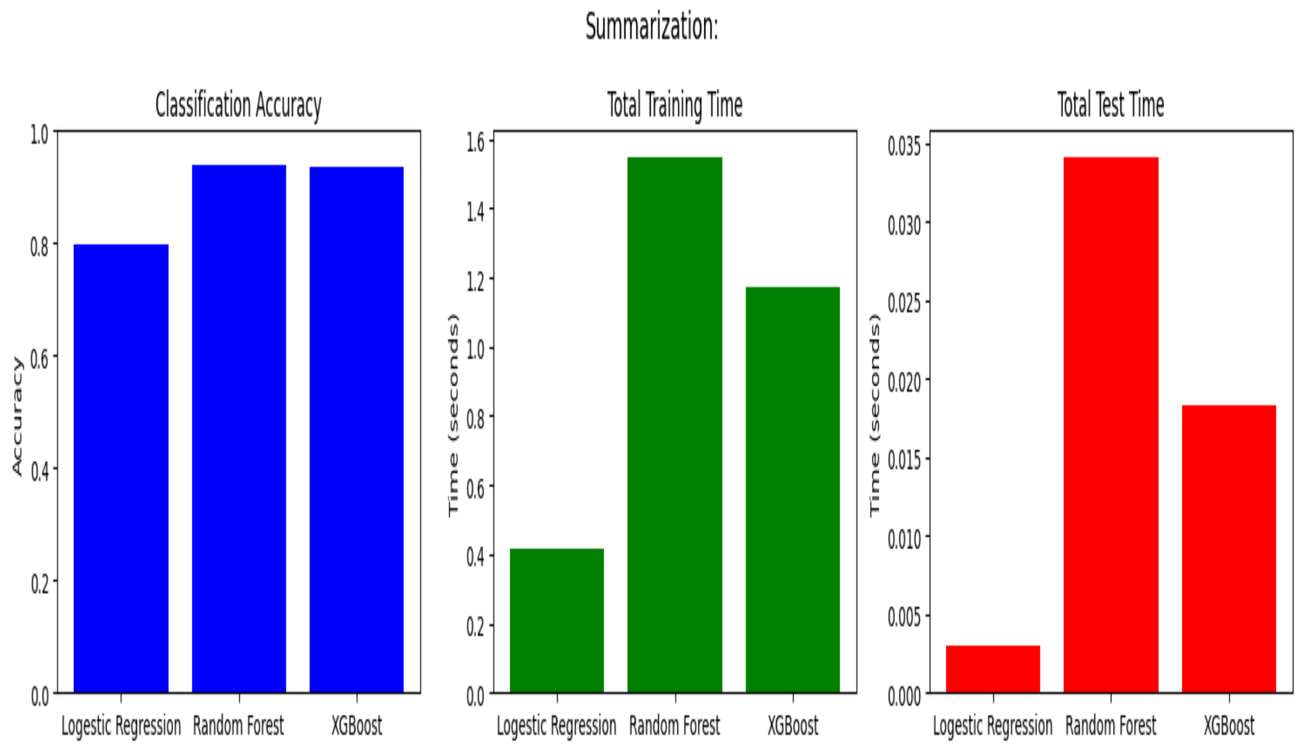
CLASSIFICATION


PARKINSON'S DISEASE
Parkinson's disease is a progressive neurological disorder that affects movement, balance, and coordination, and is caused by the loss of dopamine-producing neurons in the brain.
SYMPTOMS
• Tremor. A tremor, or rhythmic shaking, usually begins in a limb, often your hand or fingers
• Slowed movement (bradykinesia)
• Rigid muscles
• Impaired posture and balance
• Loss of automatic movements
• Speech changes
• Writing changes

CS_10

NAME	ID
عمرمحمد سعيد محمد عبده	2022170282
عبدالله ابراهيم احمد ابراهيم	2022170220
على بسام المصرى	2022170259
عبدالرحمن حمدى عبدالحليم عمران	2022170217
مصطفى محمد مجدى مصطفى	2022170425
اسلام عمرو عبدالعزيز السيد	2022170060

Summarization:



```
Logestic accuracy= 0.797  test time:= 0.0029  training time:= 0.416
Random Forest accuracy= 0.94  test time:= 0.034  training time:= 1.54
XGBoost accuracy= 0.935  test time:= 0.018  training time:= 1.169
```

Feature Selection in Classification

In the classification phase, the feature selection process. Was almost the same as regression phase, but we used RandomForestClassifier instead of Regressor **Key Adjustments:**

- We reviewed feature importances using `RandomForestClassifier`'s `.feature_importances_` attribute.

- We dropped some weak or noisy features that contributed little to classification accuracy.

- We also compared model performance with and without those features to confirm impact.

Proved: Removing low-importance features led to slight improvements in accuracy and reduced overfitting.
Disproved: Some features that were strong predictors in regression did not help classification and were dropped.

Effect of Hyperparameter Tuning

We manually tuned hyperparameters for both **Logistic Regression** and **Random Forest**, focusing on key parameters:

Logistic Regression:

- Tuned `C` (regularization strength) and `max_iter`.

- **Accuracy test**

```
C=0.01 | max_iter=10000 | Accuracy=0.7506

C=1 | max_iter=10000 | Accuracy=0.7580

C=100 | max_iter=10000 | Accuracy=0.7580

C=1 | max_iter=500 | Accuracy=0.7481

C=1 | max_iter=1000 | Accuracy=0.7457

C=1 | max_iter=10000 | Accuracy=0.7580
```

- **Observation:**
  - Best performance was achieved at `C=1` with `max_iter = 10000`.
  - Low max_iter values worsened accuracy due to underfitting.
  - Low C values worsened accuracy.

Random Forest:

- Tuned `n_estimators` and `max_depth`.
- **Accuracy test**

```
max_depth=10 | n_estimators=50 | Accuracy=0.9160

max_depth=10 | n_estimators=100 | Accuracy=0.9333

max_depth=10 | n_estimators=200 | Accuracy=0.9432

n_estimators=100 | max_depth=5 | Accuracy=0.9086

n_estimators=100 | max_depth=10 | Accuracy=0.9432

n_estimators=100 | max_depth=30 | Accuracy=0.9407
```

```
n_estimators=100 | max_depth=None | Accuracy=0.9407
```

- **Observation:**
  - Increasing `n_estimators` improved performance up to a point, after which it seems to start overfitting.
  - `max_depth=10` gave the best balance between performance and generalization.

- **Overall Effect:** Hyperparameter tuning led to measurable improvements in accuracy. It helped prevent overfitting in both models by controlling model complexity.

XGBoost:

- Tuned `n_estimators` and `max_depth`.
- **Accuracy test**

```
max_depth=10 | n_estimators=50  | Accuracy=0.9259
max_depth=10 | n_estimators=100 | Accuracy=0.9210
max_depth=10 | n_estimators=200 | Accuracy=0.9185
n_estimators=100 | max_depth=5   | Accuracy=0.9407
n_estimators=100 | max_depth=10  | Accuracy=0.9210
n_estimators=100 | max_depth=15  | Accuracy=0.9185
```

- **Observation:**
  - Increasing `n_estimators` beyond 50 **did not improve accuracy** when `max_depth=10`. In fact, it slightly **reduced performance**, likely due to **overfitting**.

    - The best performance came from **max_depth=5 with n_estimators=100**, suggesting that **shallow trees** with more boosting rounds generalize better for this dataset.
    - `max_depth=10` gave decent results but started to degrade as the model complexity increased.

- **Overall Effect:** Hyperparameter tuning improved the model It helped strike a balance between underfitting and overfitting by adjusting tree depth and the number of boosting rounds.

Conclusion

Our intuition was that certain features would be equally helpful in both regression and classification — this was only partially correct. The classification models responded better to features that helped **discriminate between classes**, rather than those that had strong numeric relationships with a continuous target.

- Random Forest was more robust and required tuning of tree depth and ensemble size to avoid overfitting.
- Feature importance analysis was vital for understanding which features actually contributed to classification accuracy.

In summary, careful feature analysis and systematic hyperparameter tuning significantly improved classification performance and helped validate our understanding of the problem.