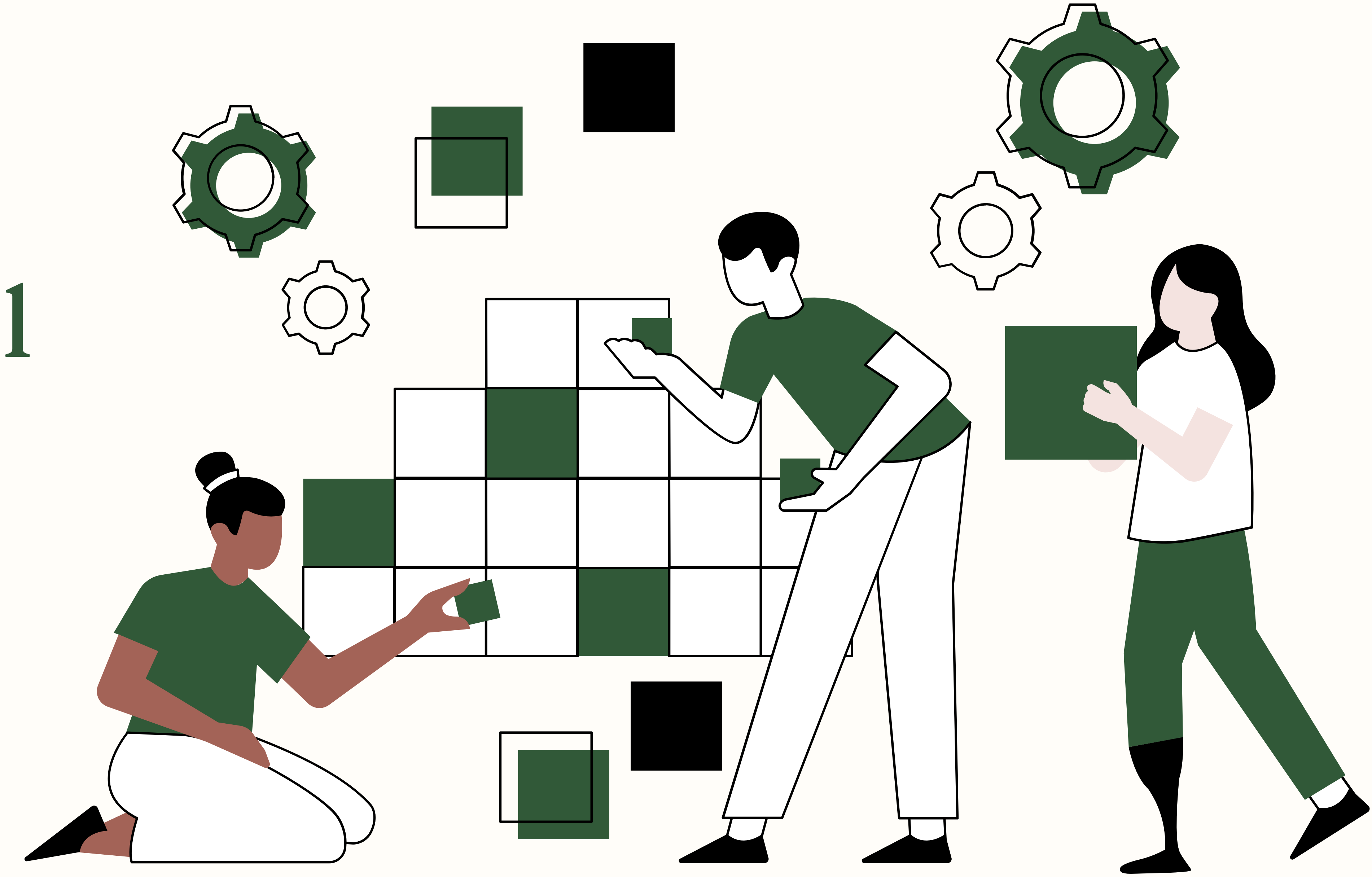
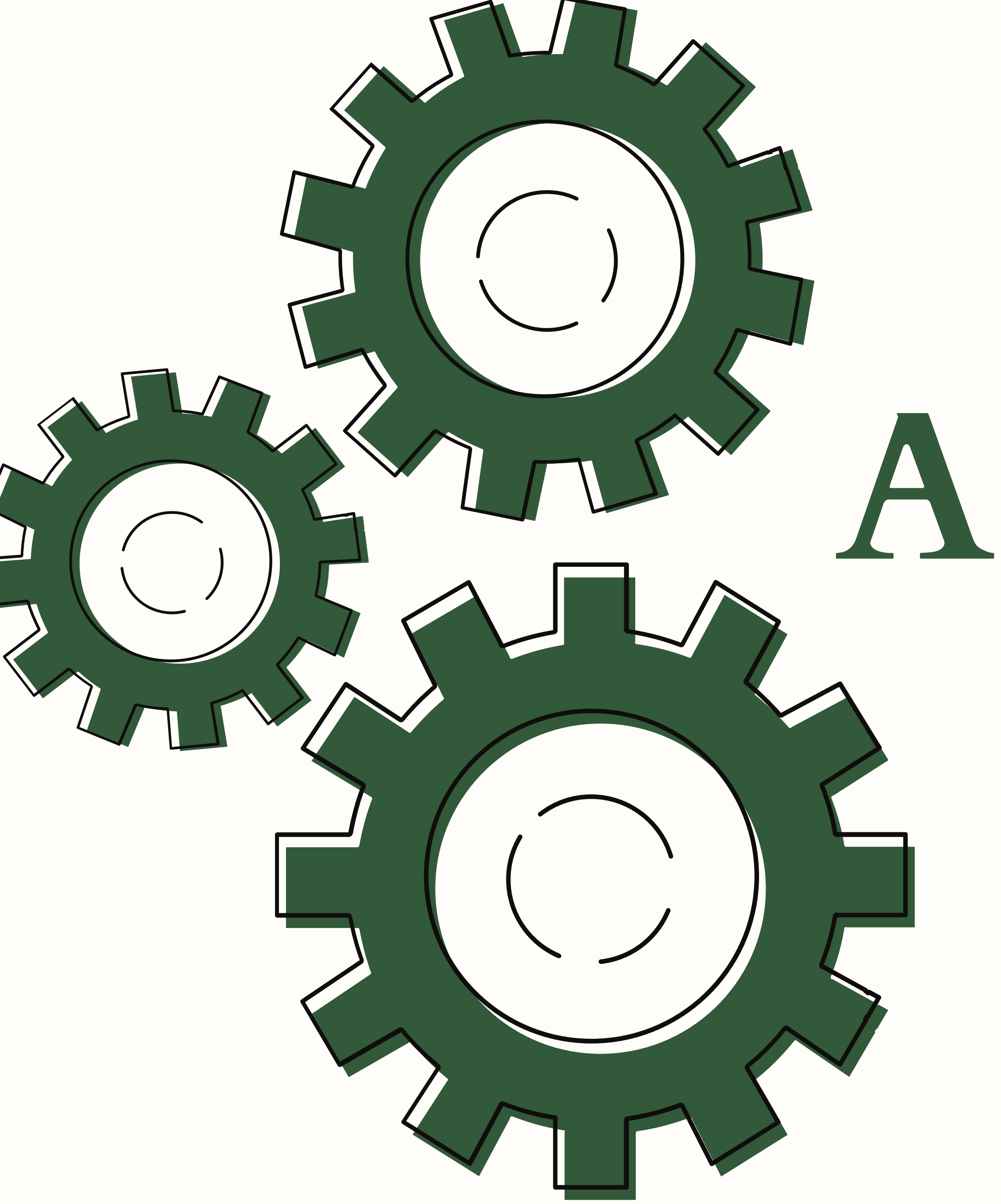
CSE 445

Project : A segmentation model that segment accurately exact the nuclei from WSIs.



November 26, 2024

CSE 445



Agenda

- Introduction
- Timeline
- Methodology
- Results
- Conclusion

Introduction

A segmentation model that segment accurately exact the nuclei from WSIs Using Machine Learning Model

About the Project

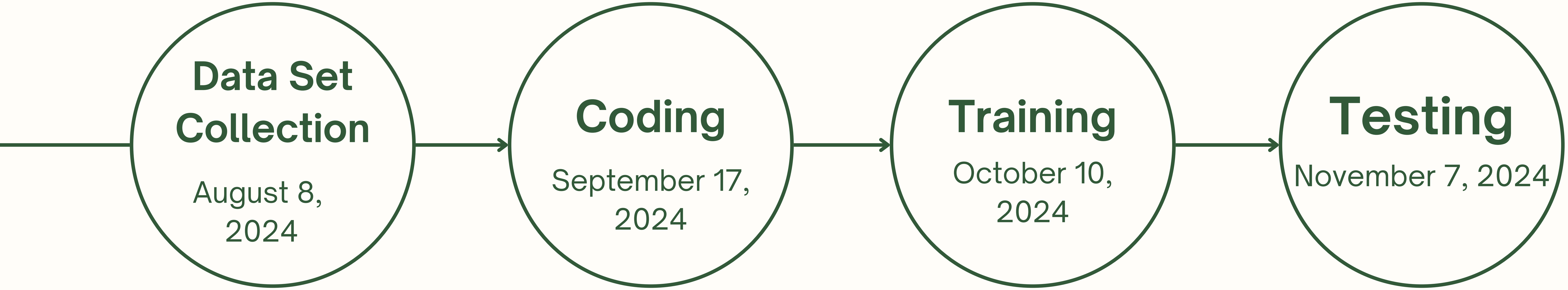
This project focuses on regression tasks using various machine learning models. Initially, the following models will be employed for training and prediction:

- Linear Regression: A simple model for predicting a continuous target variable based on linear relationships between input features.
- Decision Tree Regressor: A model that uses a decision tree to predict continuous values by learning simple decision rules inferred from the input features.
- Random Forest Regressor: An ensemble learning method that combines multiple decision trees to improve accuracy and prevent overfitting.
- K-Nearest Neighbors Regressor (KNN): A non-parametric model that predicts the target variable based on the average of the nearest K data points.

Future plans involve exploring more advanced models, including UNET for segmentation tasks and its variants, such as RESUNET and DEEPLABV3+, for more complex tasks related to image segmentation.

Timeline

Project Timeline



Data Set Collection

August 8, 2024

Coding

September 17, 2024

Training

October 10, 2024

Testing

November 7, 2024

RIMBERIO

Methodology

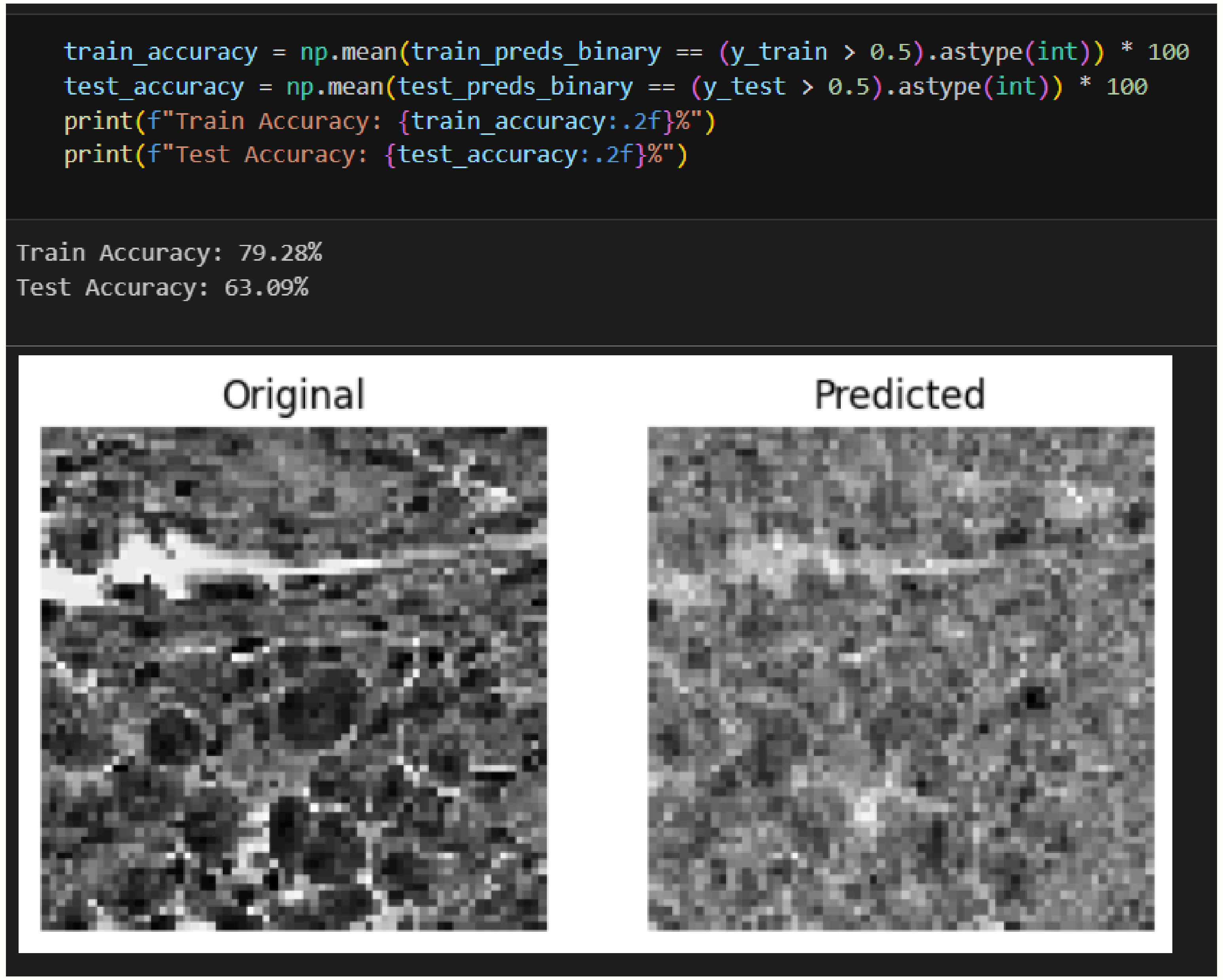
Methodology

CSE445

Model Selection

To explore the performance of different machine learning models for segmentation, the following models will be implemented:

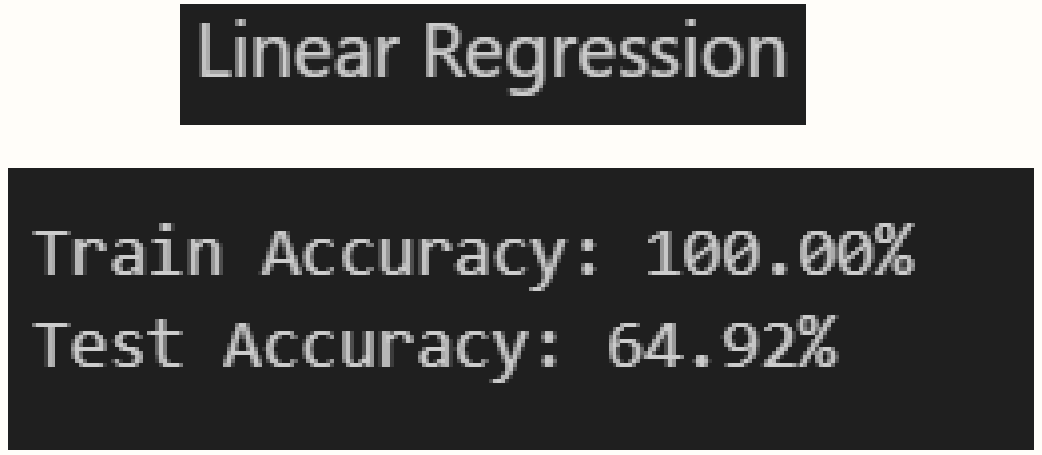
1. Linear Regression:
   - Baseline model to predict pixel intensities based on linear relationships.
   - Converts the output into binary masks using a threshold.
2. Decision Tree:
   - Build a tree structure to classify pixels based on feature splits.
   - Pruning techniques will be applied to prevent overfitting.
3. Random Forest:
   - Use an ensemble of decision trees to improve segmentation performance.
   - Perform pixel-wise classification by aggregating predictions from multiple trees.
4. k-Nearest Neighbors (kNN):
   - Classify pixels by considering the majority label of their nearest neighbors in the feature space.
   - Tune the number of neighbors (k) for optimal performance.



```
train_accuracy = np.mean(train_preds_binary == (y_train > 0.5).astype(int)) * 100
test_accuracy = np.mean(test_preds_binary == (y_test > 0.5).astype(int)) * 100
print(f"Train Accuracy: {train_accuracy:.2f}%")
print(f"Test Accuracy: {test_accuracy:.2f}%")
```

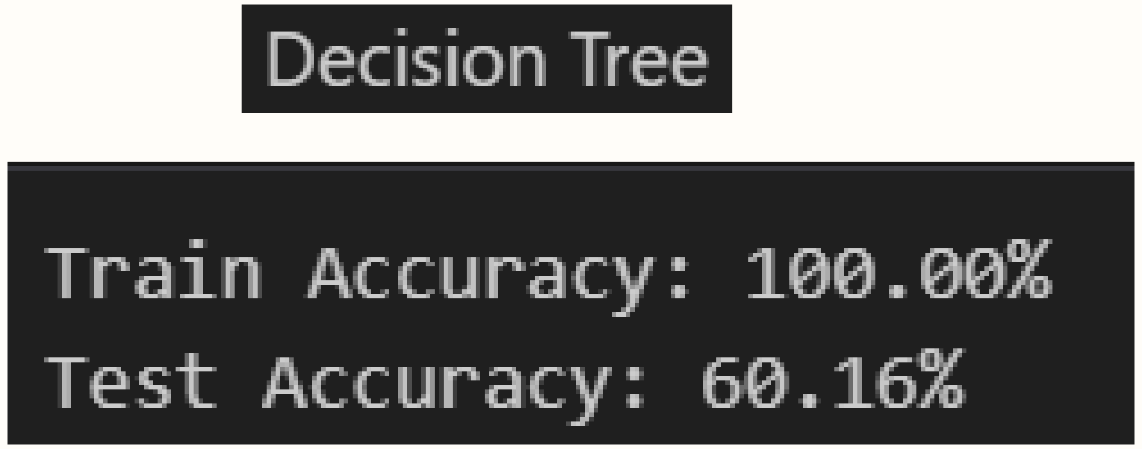
```
Train Accuracy: 79.28%
Test Accuracy: 63.09%
```

Results



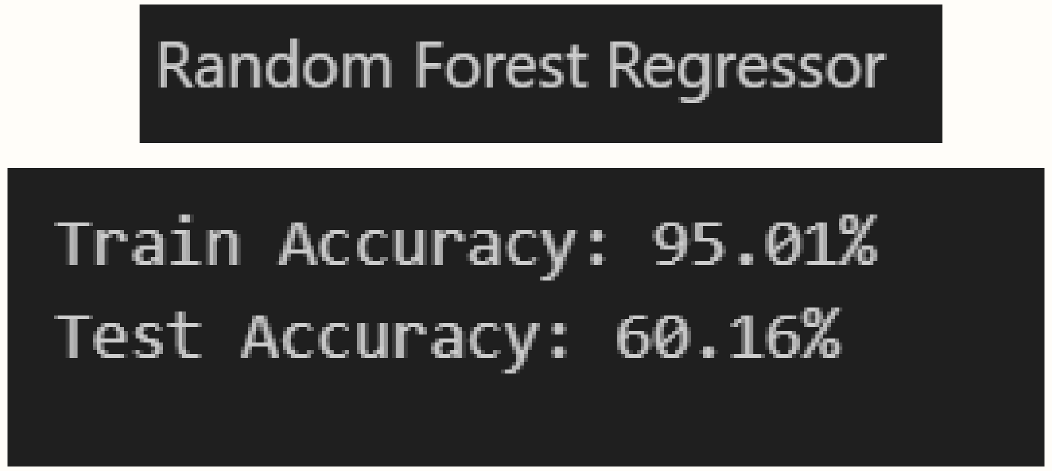
Linear Regression

```
Train Accuracy: 100.00%
Test Accuracy: 64.92%
```



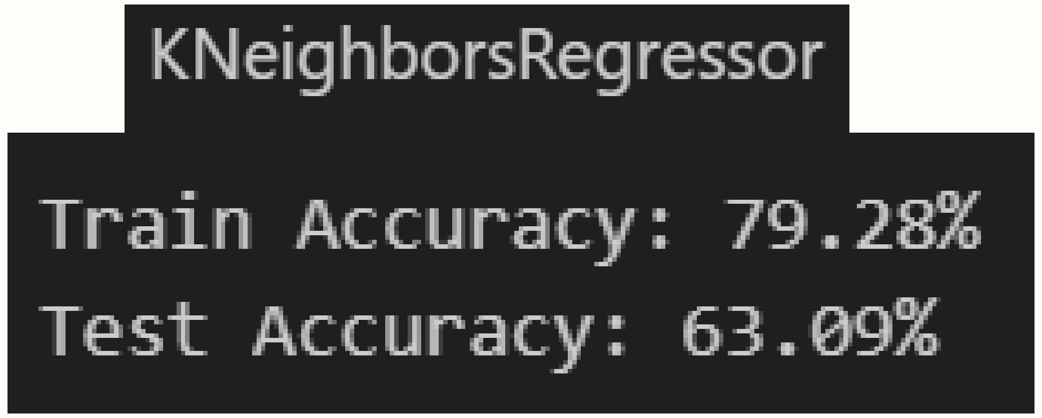
Decision Tree

```
Train Accuracy: 100.00%
Test Accuracy: 60.16%
```



Random Forest Regressor

```
Train Accuracy: 95.01%
Test Accuracy: 60.16%
```

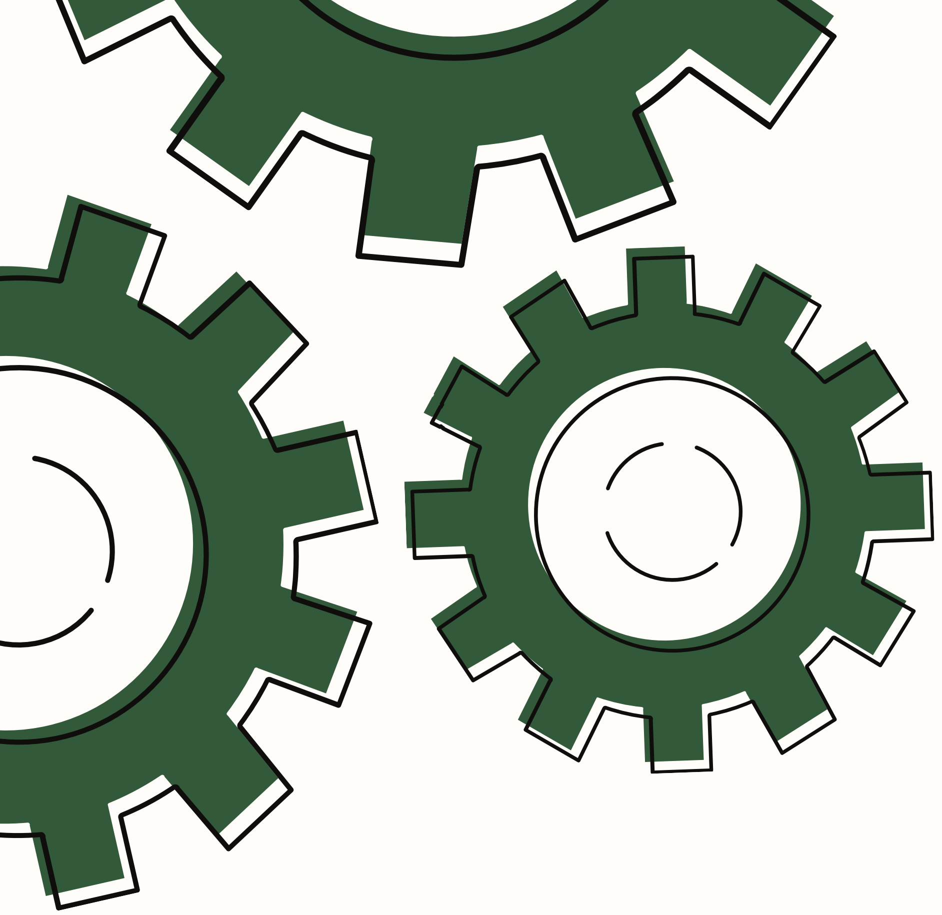


KNeighborsRegressor

```
Train Accuracy: 79.28%
Test Accuracy: 63.09%
```

Conclusion

- We find KNN regressor with the highest testing accuracy.
- We have future plans involve exploring more advanced models, including UNET for segmentation tasks and its variants, such as RESUNET and DEEPLABV3+, for more complex tasks related to image segmentation.



Thank you!