GERMAN INTERNATIONAL UNIVERSITY
الجامعة الألمانية الدولية

Advanced Machine Learning

(Random Forest, AdaBoost, and Gradient Boost)

Done by:

Mohamed Bassem 2003731
Mina Ehab 2005830

[13/3/2024]

Data processing

Dataset 1 (“Banknote Authentication”)

```
# Check duplicates,missing values,types
data_banknote_authentication.duplicated().sum()
# Remove duplicates
data_banknote_authentication.drop_duplicates()
# Check types
data_banknote_authentication.dtypes
# Check missing values
data_banknote_authentication.isnull().sum()
# Apply scaling to features
scaler = StandardScaler()
target = data_banknote_authentication['Class']
features = data_banknote_authentication.drop('Class',axis = 1)
scaled_features = scaler.fit_transform(features)
scaled_df = pd.DataFrame(scaled_features, columns=features.columns)
scaled_data_banknote_authentication = pd.concat([scaled_df, target], axis=1)
scaled_data_banknote_authentication
```

What have we Done ?

1-Check for duplicates: 0 duplicates found.
2-Check types of features: All features are numerical.
3-Check for missing values: No missing values found.
4-Apply feature scaling using StandardScaler.
5-Split data into features and target.
6-Split data into train and test sets.

Dataset 2 (“Glass Type prediction”)

```
# Check duplicates,missing values,types
glasstypePrediction.duplicated().sum()
# Remove duplicates
glasstypePrediction.drop_duplicates()
# Check types
glasstypePrediction.dtypes
# Check missing values
glasstypePrediction.isnull().sum()
# Apply scaling to features
scaler = StandardScaler()
target = glasstypePrediction['Type']
features = glasstypePrediction.drop('Type',axis = 1)
scaled_features = scaler.fit_transform(features)
scaled_df = pd.DataFrame(scaled_features, columns=features.columns)
scaled_glasstypePrediction = pd.concat([scaled_df, target], axis=1)
scaled_glasstypePrediction
```

What have we Done ?

1-Check for duplicates: 0 duplicates found.
2-Check types of features: All features are numerical.
3-Check for missing values: No missing values found.
4-Apply feature scaling using StandardScaler.
5-Split data into features and target.
6-Split data into train and test sets.

Dataset 3 (“House Price Prediction”)

```
# Apply label encoding
label_encoder = LabelEncoder()

housePricePrediction['Encoded_POSTED_BY'] = label_encoder.fit_transform(housePricePrediction['POSTED_BY'])
housePricePrediction['Encoded_BHK_OR_RK'] = label_encoder.fit_transform(housePricePrediction['BHK_OR_RK'])
housePricePrediction = housePricePrediction.drop(['POSTED_BY','BHK_OR_RK','ADDRESS'],axis =1)

# Check duplicates,missing values,types
housePricePrediction.duplicated().sum()
# Remove duplicates
housePricePrediction.drop_duplicates()
# Check missing values
housePricePrediction.isnull().sum()
# Check types
housePricePrediction.dtypes
```

What have we Done ?

1-Check for duplicates: 0 duplicates found.
2-Check types of features: Features include numerical and categorical data.
3-Check for missing values: No missing values found.
4-Apply label encoding to categorical features.
5-Split data into features and target.
6-Split data into train and test sets.

Hyperparameter Tuning Process (Random Forest)

What have we Done In the 3 Dataset1 ?

1-Define the parameter grid for Random Forest.
2-Perform hyperparameter tuning using GridSearchCV.
3-Evaluate the model using accuracy, precision, recall, and F1-score.

Dataset 1 ("Banknote Authentication")

Accuracy	Precision	Recall	F1-score
99.27%	1.0	0.98	0.99

Dataset 2 ("Glass Type prediction")

Accuracy	Precision	Recall	F1-score
83.72%	0.87	0.84	0.83

Dataset 3 ("House Price Prediction")

Mean Squared Error
175161.02

Hyperparameter Tuning Process (AdaBoost)

What have we Done In the 3 Dataset1 ?

1-Define the parameter grid for Random Forest.
2-Perform hyperparameter tuning using GridSearchCV.
3-Evaluate the model using accuracy, precision, recall, and F1-score.

Dataset 1 ("Banknote Authentication")

Accuracy	Precision	Recall	F1-score
97.09%	0.97	0.97	0.97

Dataset 2 ("Glass Type prediction")

Accuracy	Precision	Recall	F1-score
69.77%	0.66	0.70	0.67

Dataset 3 ("House Price Prediction")

Mean Squared Error
123601.31

Hyperparameter Tuning Process (Gradient Boosting)

What have we Done In the 3 Dataset1 ?

1-Define the parameter grid for Random Forest.
2-Perform hyperparameter tuning using GridSearchCV.
3-Evaluate the model using accuracy, precision, recall, and F1-score.

Dataset 1 ("Banknote Authentication")

Accuracy	Precision	Recall	F1-score
100.0%	1.0	1.0	1.0

Dataset 2 ("Glass Type prediction")

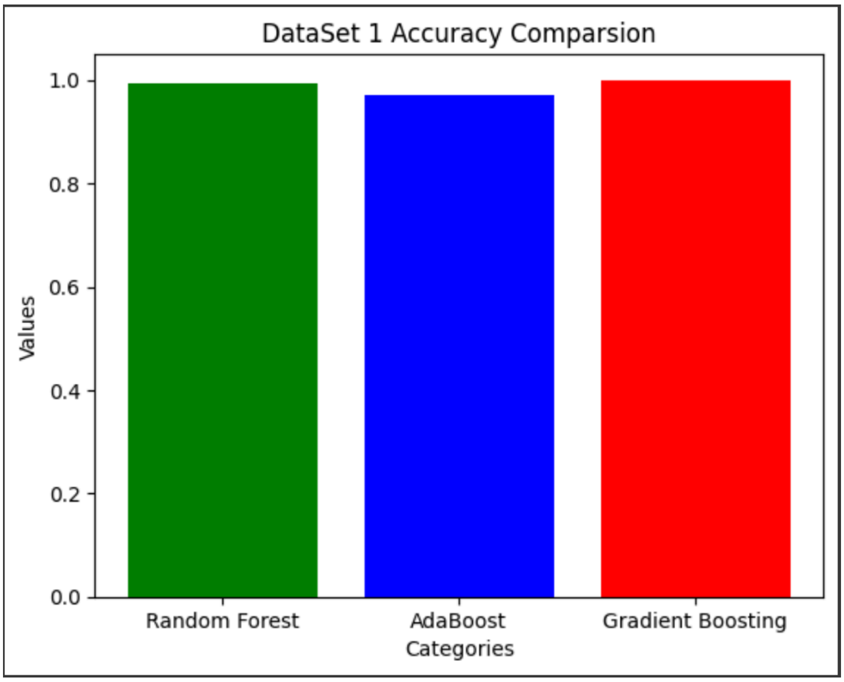
Accuracy	Precision	Recall	F1-score
86.05%	0.88	0.86	0.85

Dataset 3 ("House Price Prediction")

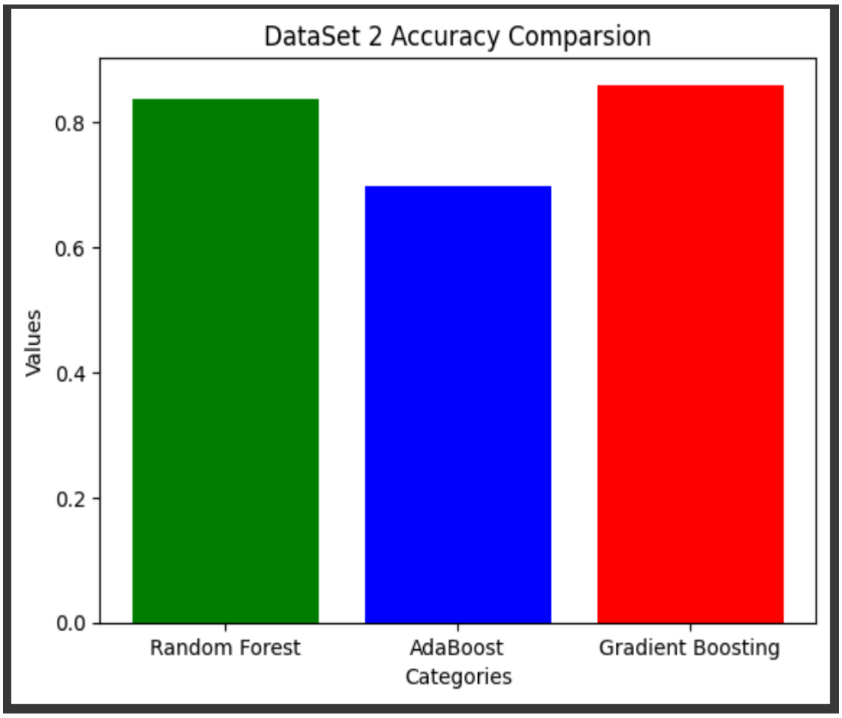
Mean Squared Error
120975.64

Comparisons in accuracy between models performance on each Dataset Using Barcharts

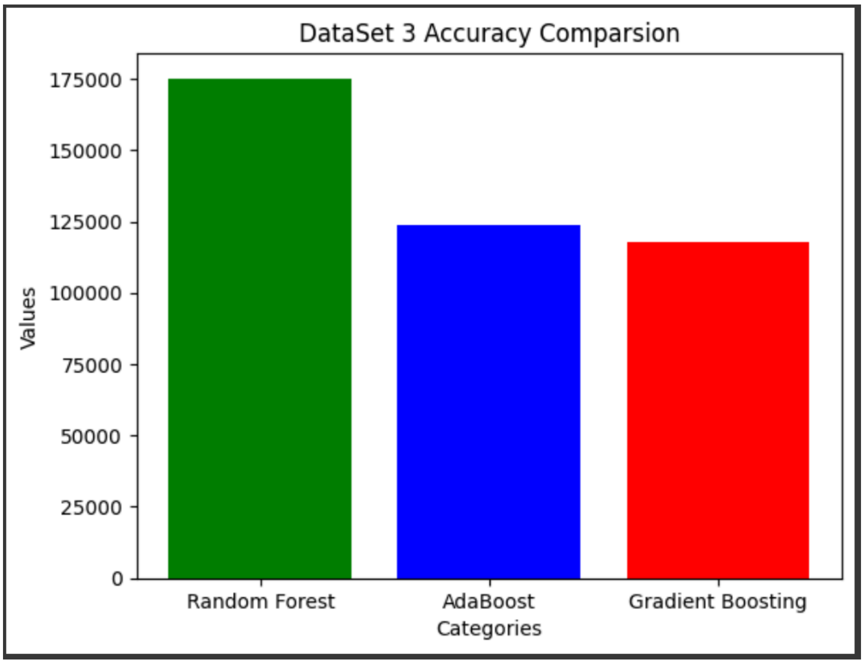
Dataset 1 ("Banknote Authentication")



Dataset 2 ("Glass Type prediction")



Dataset 3 ("House Price Prediction")

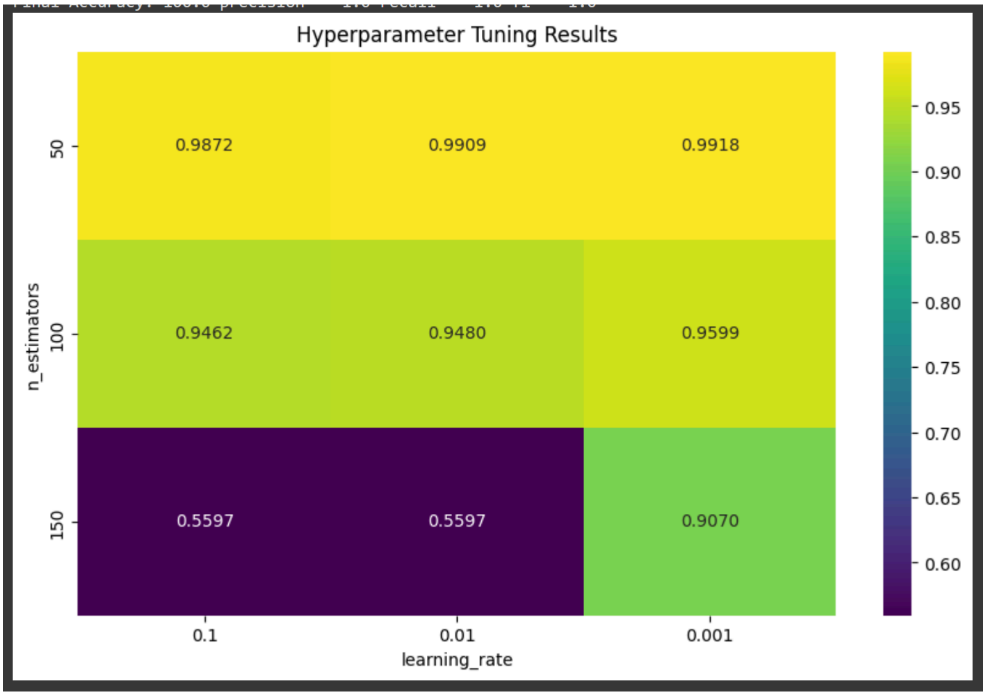


Conclusion Table("Accuracy")

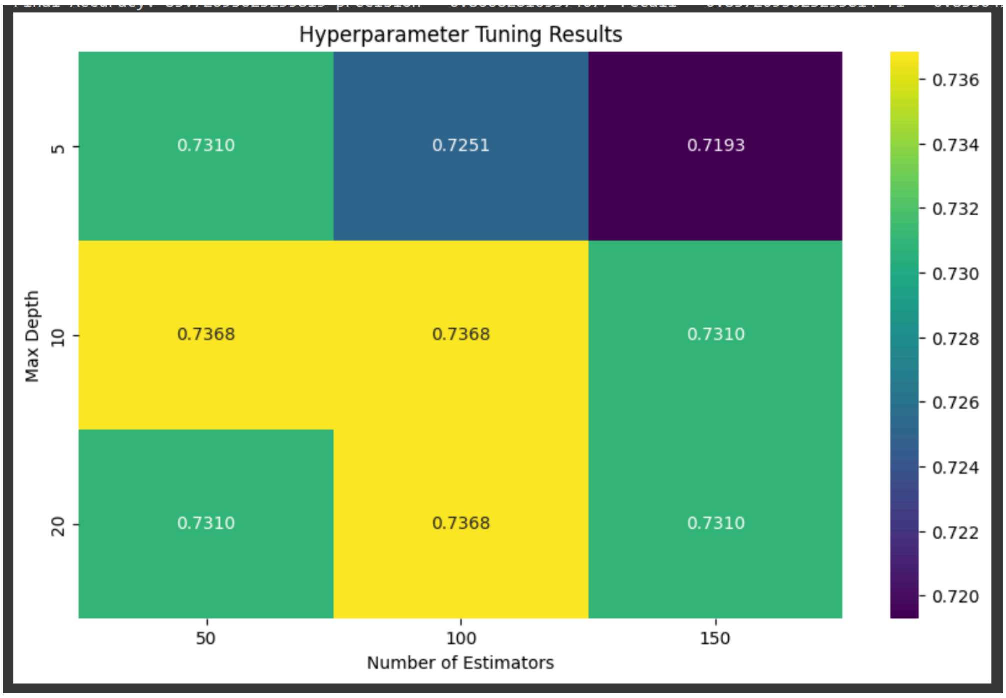
Datasets	Random Forest (%)	AdaBoost (%)	Gradient Boosting (%)
Banknote Authentication	99.27	97.09	100.00
Glass Type prediction	83.72	69.77	86.05
House Price Prediction	175161.02	123601.31	120975.64

HeatMaps for the Models worked Best with each Dataset

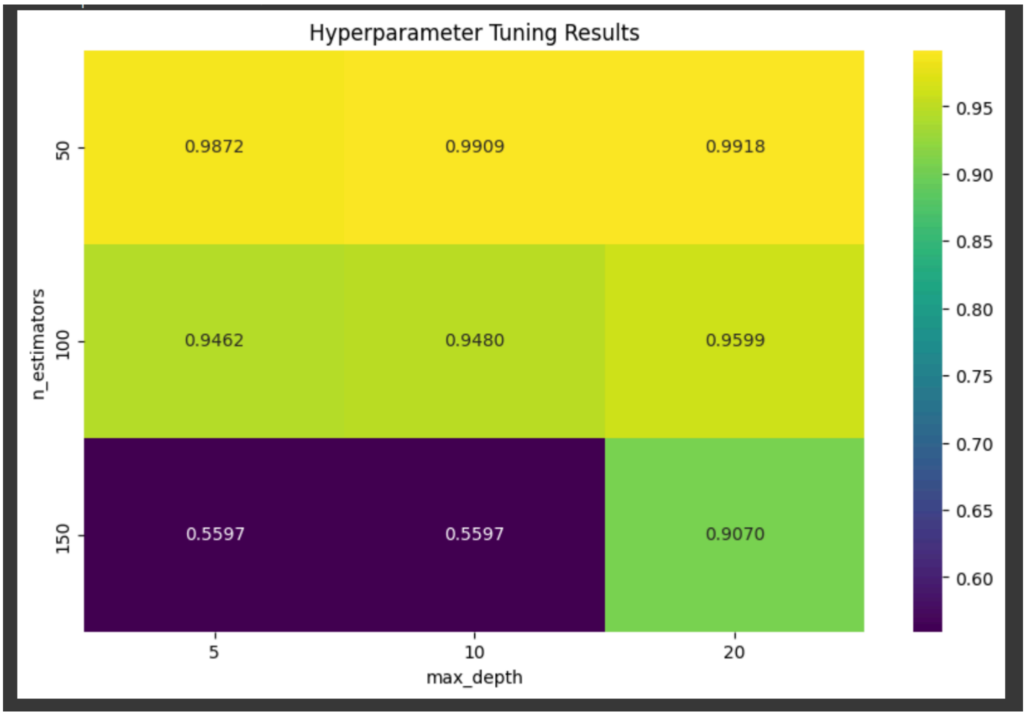
Dataset 1 (“Banknote Authentication”)(“Gradient Boosting”)



Dataset 2 (“Glass Type prediction”)(“Random Forest”)



Dataset 3 ("House Price Prediction")("Gradient Boosting")



⭐This heatmap visualizes the mean test scores for different combinations of hyperparameters. Each cell in the heatmap represents the mean test score for a particular combination of hyperparameters. The x-axis and y-axis of the heatmap represent the values of the hyperparameters specified in param_grid

Our Insights;-

1-Model Performance:-
Gradient Boosting consistently performs best, achieving the highest accuracy in most cases. Random Forest and AdaBoost follow, with Random Forest slightly outperforming AdaBoost on some datasets.
2-Regression Task Performance;-
Gradient Boosting demonstrates superior performance in regression tasks, indicated by the lowest Mean Squared Error (MSE) on housePricePrediction dataset.
3-Model Suitability:-
Random Forest and AdaBoost are suitable for classification tasks, providing a good balance between performance and computational efficiency. Gradient Boosting is more computationally intensive but offers higher accuracy and is suitable for both classification and regression tasks.
4-Dataset Characteristics:-
Model performance varies based on dataset characteristics, such as feature complexity and data distribution. It's essential to choose the model that fits the dataset's characteristics and task requirements.
5-Hyperparameter Tuning:-
Fine-tuning hyperparameters, especially for Gradient Boosting, significantly improves model performance.
6-Generalization:-
While Gradient Boosting consistently performs well, model generalization should be evaluated using techniques like cross-validation for real-world applications.