In [25]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier, AdaBoostClassifier, ExtraTreesClas
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression, RidgeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor, AdaBoostRegressor, ExtraTreesRegress
from sklearn.linear_model import LinearRegression, Ridge
from sklearn.metrics import accuracy_score, classification_report, mean_absolute_error, mean_squared_error, r2_scor
```

In [26]:
```
# Load dataset (Wine Quality dataset from UCI Repository)
df = pd.read_csv("https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-red.csv", sep=


```

In [27]:
```
#define features and target
X = df.drop(columns=['quality'])
y= df['quality']
```

In [28]:
```
# Convert target into binary classification (Good vs. Bad wine)
y_class = y.apply(lambda x: 1 if x >= 6 else 0)

# Split the dataset for classification
X_train_c, X_test_c, y_train_c, y_test_c = train_test_split(X, y_class, test_size=0.2, random_state=42)


```

In [29]:
```
#classification models
classification_models = {
    'Logistic Regression' : LogisticRegression(),
    'Ridge Regression' : RidgeClassifier(),
    'Random Forest' : RandomForestClassifier(n_estimators=50, random_state=41),
    'Gradient Boosting': GradientBoostingClassifier(n_estimators=100, random_state=42),
    'AdaBoost' : AdaBoostClassifier(n_estimators=50 , random_state=41),
    'Extra Trees' : ExtraTreesClassifier(n_estimators=50, random_state=41),
    'SVM' : SVC(kernel='rbf' , probability=True),
    'K-Nearest Neighbours' : KNeighborsClassifier(n_neighbors=5),
    'Decission Tree' : DecisionTreeClassifier(random_state=41)

}
```

In [30]:
```
best_classification_model = None
best_classification_accuracy = 0

# Iterate over classification models and evaluate
print("\nClassification Models Evaluation")
for name, model in classification_models.items():
    pipeline = Pipeline([
        ('scaler', StandardScaler()),
        ('classifier', model)
    ])

    # Train the model
    pipeline.fit(X_train_c, y_train_c)

    # Predict
    y_pred_c = pipeline.predict(X_test_c)

    # Evaluation metrics
    accuracy = accuracy_score(y_test_c, y_pred_c)
    report = classification_report(y_test_c, y_pred_c)

    print(f"\n{name} Classification Performance:")
    print(f"Accuracy: {accuracy:.4f}")
    print("Classification Report:")
    print(report)

    if accuracy > best_classification_accuracy:
        best_classification_accuracy = accuracy
        best_classification_model = name


```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.80      0.80      0.80       141
           1       0.84      0.84      0.84       179

    accuracy                           0.82       320
   macro avg       0.82      0.82      0.82       320
weighted avg       0.82      0.82      0.82       320


SVM Classification Performance:
Accuracy: 0.7719
Classification Report:
              precision    recall  f1-score   support

           0       0.73      0.77      0.75       141
           1       0.81      0.77      0.79       179

    accuracy                           0.77       320
```

In [31]:
```
#split for regression

X_train_r,X_test_r,y_train_r,y_test_r = train_test_split(X, y, test_size=0.25, random_state=41)

#reg models
regression_models = {
    'Linear Regression' : LinearRegression(),
    'Ridge Regression' : Ridge(),
    'Random Forest Regressor' : RandomForestRegressor(n_estimators=100, random_state=41),
    'Gradient Boosting Regressor' : GradientBoostingRegressor(n_estimators=50, random_state=41),
    'AdaBoost Regressor' : AdaBoostRegressor(n_estimators=50, random_state=41),
    'Extra Trees Regressor': ExtraTreesRegressor(n_estimators=50, random_state=41)

}
```

In [37]:
```
best_regression_model = None
best_r2_score = -float("inf")

# Iterate over regression models and evaluate
print("\nRegression Models Evaluation")
for name, model in regression_models.items():
    pipeline = Pipeline([
        ('scaler', StandardScaler()),
        ('regressor', model)
    ])

    # Train the model
    pipeline.fit(X_train_r, y_train_r)

    # Predict
    y_pred_r = pipeline.predict(X_test_r)

    # Evaluation metrics
    mae = mean_absolute_error(y_test_r, y_pred_r)
    mse = mean_squared_error(y_test_r, y_pred_r)
    r2 = r2_score(y_test_r, y_pred_r)

    print(f"\n{name} Regression Performance:")
    print(f"Mean Absolute Error: {mae:.4f}")
    print(f"Mean Squared Error: {mse:.4f}")
    print(f"R^2 Score: {r2:.4f}")

    if r2 > best_r2_score:
        best_r2_score = r2
        best_regression_model = name


```

```
Regression Models Evaluation

Linear Regression Regression Performance:
Mean Absolute Error: 0.5032
Mean Squared Error: 0.4199
R^2 Score: 0.3619

Ridge Regression Regression Performance:
Mean Absolute Error: 0.5033
Mean Squared Error: 0.4199
R^2 Score: 0.3619

Random Forest Regressor Regression Performance:
Mean Absolute Error: 0.3964
Mean Squared Error: 0.3032
R^2 Score: 0.5392

Gradient Boosting Regressor Regression Performance:
Mean Absolute Error: 0.4623
Mean Squared Error: 0.3519
R^2 Score: 0.4652

AdaBoost Regressor Regression Performance:
Mean Absolute Error: 0.4758
Mean Squared Error: 0.3838
R^2 Score: 0.4166

Extra Trees Regressor Regression Performance:
Mean Absolute Error: 0.3511
Mean Squared Error: 0.2763
R^2 Score: 0.5800
```

In [38]:
```
# Conclusion
print("\nConclusion:")
print(f"The best classification model is {best_classification_model} with an accuracy of {best_classification_accur
print(f"The best regression model is {best_regression_model} with an R^2 score of {best_r2_score:.4f}.")
```

```
Conclusion:
The best classification model is Extra Trees with an accuracy of 0.8219.
The best regression model is Extra Trees Regressor with an R^2 score of 0.5800.
```

In [ ]:
```

```