ID: 16

Predicting Feed Rate of CNC Machine Using Random Forest Regression

Objective

The objective of this experiment is to predict the feed rate (`M1_CURRENT_FEEDRATE`) using a Random Forest Regressor and evaluate the model's performance. Additionally, the experiment includes visualization of the relationship between actual and predicted feed rates with a fitted linear regression line.

Methodology

Dataset

The dataset used in this experiment is stored in a file named `experiment_01.csv`. It contains several features, including:
Target Variable`M1_CURRENT_FEEDRATE`
Exclusion:The categorical feature `Machining_Process` was excluded from the analysis.

Steps

Data Loading: The dataset was loaded using the `pandas` library.

Data Preprocessing: The target variable, `M1_CURRENT_FEEDRATE`, was separated from the features. The `Machining_Process` column, a categorical variable, was excluded from the features. The features were split into training and testing sets in an 80:20 ratio.

Feature Scaling: The features were standardized using `StandardScaler` to improve the model's performance.

Model Training: A Random Forest Regressor was trained on the scaled training data using default hyperparameters and a random seed of 42.

Prediction and Evaluation: The trained model was used to predict feed rates for the testing data. Model performance was evaluated using Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared (R^2) Score.

Feature Importance: The relative importance of each feature was extracted and ranked.

Visualization: A scatter plot of actual vs predicted feed rates was created. A red linear regression line was overlaid to show the trend.

Results

Model Performance

Mean Absolute Error (MAE): 0.32530805687

Mean Squared Error (MSE): 3.65546161

R² Score: 0.974511097

Feature Importance

The following table shows the ranked importance of features:

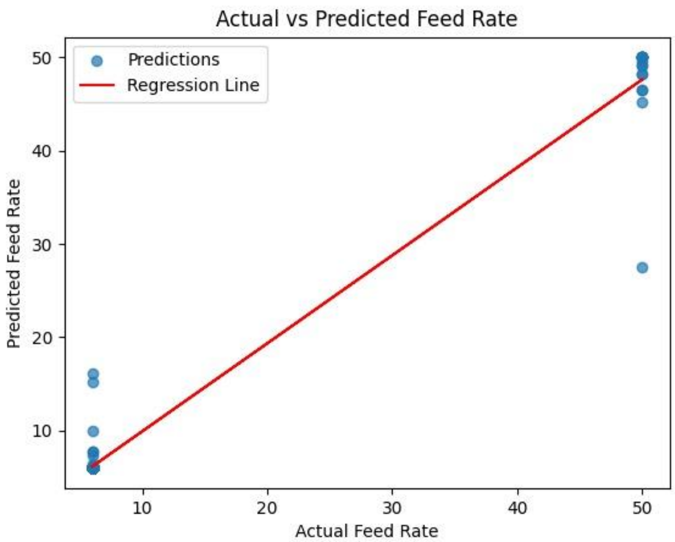
```
C:\Users\HP\PycharmProjects\pythonProject\venv\Scripts\python.exe "C:\Users\
Mean Absolute Error (MAE): 0.325308056872038
Mean Squared Error (MSE): 3.6554616113744087
R^2 Score: 0.9745110976349303

Feature Importance:

                     Feature  Importance
25       Z1_CommandPosition    0.472072
22        Z1_ActualPosition    0.329151
23        Z1_ActualVelocity    0.056336
26       Z1_CommandVelocity    0.038534
38       S1_CurrentFeedback    0.011524
42           S1_OutputPower    0.011118
24    Z1_ActualAcceleration    0.009244
33        S1_ActualVelocity    0.009087
8          X1_OutputCurrent    0.008532
39          S1_DCBusVoltage    0.008213
40         S1_OutputCurrent    0.007737
36       S1_CommandVelocity    0.006369
41         S1_OutputVoltage    0.006233
19         Y1_OutputCurrent    0.005695
18          Y1_DCBusVoltage    0.005174
35       S1_CommandPosition    0.004180
32        S1_ActualPosition    0.003490
27   Z1_CommandAcceleration    0.002519
```

Visualization

A scatter plot of actual vs predicted feed rates was generated. A linear regression line was fitted to the scatter data and plotted in red to highlight the trend. The visualization reveals a positive correlation between the actual and predicted values, suggesting good model performance.



Conclusion

The Random Forest Regressor effectively predicted the feed rate, achieving an R^2 score of 0.974511097,indicating the model's ability to explain the variability in the data. The visualization demonstrated that the predictions closely align with the actual values, supported by the linear regression line.

Future Work

- Experiment with hyperparameter tuning for the Random Forest Regressor.

- Incorporate feature engineering to extract additional relevant features.

- Explore alternative machine learning models, such as Gradient Boosting or Neural Networks.

Code

Below is the Python code used in the experiment:

```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
import numpy as np
import matplotlib.pyplot as plt

# Load dataset
file_path = 'experiment_01.csv'
data = pd.read_csv(file_path)

# Define target variable and features
```

```
target = "M1_CURRENT_FEEDRATE"
X = data.drop(columns=[target, "Machining_Process"])
y = data[target]

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Scale features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Train a Random Forest Regressor
model = RandomForestRegressor(random_state=42, n_estimators=100)
model.fit(X_train_scaled, y_train)

# Make predictions
y_pred = model.predict(X_test_scaled)

# Evaluate the model
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Absolute Error (MAE): {mae}")
print(f"Mean Squared Error (MSE): {mse}")
print(f"R^2 Score: {r2}")

# Feature Importance
feature_importance = pd.DataFrame({
    'Feature': X.columns,
    'Importance': model.feature_importances_
}).sort_values(by='Importance', ascending=False)

print("\nFeature Importance:\n")
print(feature_importance)

# Visualization with regression line
plt.scatter(y_test, y_pred, alpha=0.7, label='Predictions')
plt.xlabel("Actual Feed Rate")
plt.ylabel("Predicted Feed Rate")
plt.title("Actual vs Predicted Feed Rate")

# Add linear regression line
```

```
slope, intercept = np.polyfit(y_test, y_pred, 1)
line = slope * y_test + intercept
plt.plot(y_test, line, color='red', label='Regression Line')

plt.legend()
plt.show()
```