x8hsusyf4

January 27, 2025

[ ]:
```
#FINAL PROJECT REPORT:

#Project Title: Decision Tree Regression Model for Student Performance Dataset

#Dataset: Student Performance Dataset
#Objective: Predicting Student Performance based on some feauters
#Methodology:
#1. Data Importing and Preprocessing
#2.  Encoding
#3. Decision Tree Regressor Modeling
#4. Model Evaluation
#Results:
#Mean Squared Error (MSE): 2.5822784810126582
#Mean Absolute Error (MAE): 1.0379746835443038
#Root Mean Squared Error (RMSE): 1.6069469440565418
#Coefficient of Determination (R-squared): 0.8476345346594562
#Mean Absolute Percentage Error (MAPE): inf
#Median Absolute Error (MdAE): 1.0

# ACCURACY : 84.7

#Conclusion:
#The Decision Tree Regressor model achieved high accuracy in predicting student␣
 ↪Performance.

#Summary of Graphs:
#  Actual vs. Predicted (Scatter Plot): Compares the predicted grades to actual␣
 ↪grades.
I#deal model predictions should lie on a 45-degree line.
# Residual Plot: Checks for randomness in the residuals to verify that the␣
 ↪model is not biased.

# Distribution of Residuals (Histogram/KDE): Analyzes the normality of␣
 ↪residuals to ensure that the errors are evenly distributed.
# Feature Importance: Shows which features most strongly influence the model's␣
 ↪predictions,
#  helping to understand the key factors impacting students' grades.
```

[19]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import math
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import mean_squared_error,mean_absolute_error, r2_score
```

[2]:
```
#loading the dataset
data = pd.read_csv(r"C:\Users\91703\Downloads\student_data.csv")
```

[3]:
```
data.head()
```

[3]:
```
  school sex  age address famsize Pstatus  Medu  Fedu     Mjob      Fjob  … \
0     GP   F   18       U     GT3       A     4     4  at_home   teacher  …
1     GP   F   17       U     GT3       T     1     1  at_home     other  …
2     GP   F   15       U     LE3       T     1     1  at_home     other  …
3     GP   F   15       U     GT3       T     4     2   health  services  …
4     GP   F   16       U     GT3       T     3     3    other     other  …

  famrel freetime  goout  Dalc  Walc health absences  G1  G2  G3
0      4        3      4     1     1      3        6   5   6   6
1      5        3      3     1     1      3        4   5   5   6
2      4        3      2     2     3      3       10   7   8  10
3      3        2      2     1     1      5        2  15  14  15
4      4        3      2     1     2      5        4   6  10  10

[5 rows x 33 columns]
```

[8]:
```
data.columns
```

[8]:
```
Index(['school', 'sex', 'age', 'address', 'famsize', 'Pstatus', 'Medu', 'Fedu',
       'Mjob', 'Fjob', 'reason', 'guardian', 'traveltime', 'studytime',
       'failures', 'schoolsup', 'famsup', 'paid', 'activities', 'nursery',
       'higher', 'internet', 'romantic', 'famrel', 'freetime', 'goout', 'Dalc',
       'Walc', 'health', 'absences', 'G1', 'G2', 'G3'],
      dtype='object')
```

[4]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 395 entries, 0 to 394
Data columns (total 33 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
```

```
 0   school      395 non-null    object
 1   sex         395 non-null    object
 2   age         395 non-null    int64
 3   address     395 non-null    object
 4   famsize     395 non-null    object
 5   Pstatus     395 non-null    object
 6   Medu        395 non-null    int64
 7   Fedu        395 non-null    int64
 8   Mjob        395 non-null    object
 9   Fjob        395 non-null    object
 10  reason      395 non-null    object
 11  guardian    395 non-null    object
 12  traveltime  395 non-null    int64
 13  studytime   395 non-null    int64
 14  failures    395 non-null    int64
 15  schoolsup   395 non-null    object
 16  famsup      395 non-null    object
 17  paid        395 non-null    object
 18  activities  395 non-null    object
 19  nursery     395 non-null    object
 20  higher      395 non-null    object
 21  internet    395 non-null    object
 22  romantic    395 non-null    object
 23  famrel      395 non-null    int64
 24  freetime    395 non-null    int64
 25  goout       395 non-null    int64
 26  Dalc        395 non-null    int64
 27  Walc        395 non-null    int64
 28  health      395 non-null    int64
 29  absences    395 non-null    int64
 30  G1          395 non-null    int64
 31  G2          395 non-null    int64
 32  G3          395 non-null    int64
dtypes: int64(16), object(17)
memory usage: 102.0+ KB
```

[5]:
```
data.describe()
```

[5]:

	age	Medu	Fedu	traveltime	studytime	failures	\
count	395.000000	395.000000	395.000000	395.000000	395.000000	395.000000	
mean	16.696203	2.749367	2.521519	1.448101	2.035443	0.334177	
std	1.276043	1.094735	1.088201	0.697505	0.839240	0.743651	
min	15.000000	0.000000	0.000000	1.000000	1.000000	0.000000	
25%	16.000000	2.000000	2.000000	1.000000	1.000000	0.000000	
50%	17.000000	3.000000	2.000000	1.000000	2.000000	0.000000	
75%	18.000000	4.000000	3.000000	2.000000	2.000000	0.000000	
max	22.000000	4.000000	4.000000	4.000000	4.000000	3.000000	

```
            famrel    freetime       goout        Dalc        Walc      health  \
count  395.000000  395.000000  395.000000  395.000000  395.000000  395.000000
mean     3.944304    3.235443    3.108861    1.481013    2.291139    3.554430
std      0.896659    0.998862    1.113278    0.890741    1.287897    1.390303
min      1.000000    1.000000    1.000000    1.000000    1.000000    1.000000
25%      4.000000    3.000000    2.000000    1.000000    1.000000    3.000000
50%      4.000000    3.000000    3.000000    1.000000    2.000000    4.000000
75%      5.000000    4.000000    4.000000    2.000000    3.000000    5.000000
max      5.000000    5.000000    5.000000    5.000000    5.000000    5.000000

          absences          G1          G2          G3
count  395.000000  395.000000  395.000000  395.000000
mean     5.708861   10.908861   10.713924   10.415190
std      8.003096    3.319195    3.761505    4.581443
min      0.000000    3.000000    0.000000    0.000000
25%      0.000000    8.000000    9.000000    8.000000
50%      4.000000   11.000000   11.000000   11.000000
75%      8.000000   13.000000   13.000000   14.000000
max     75.000000   19.000000   19.000000   20.000000
```

[6]:
```
data.isnull().sum()
```

[6]:
```
school        0
sex           0
age           0
address       0
famsize       0
Pstatus       0
Medu          0
Fedu          0
Mjob          0
Fjob          0
reason        0
guardian      0
traveltime    0
studytime     0
failures      0
schoolsup     0
famsup        0
paid          0
activities    0
nursery       0
higher        0
internet      0
romantic      0
famrel        0
```

```
freetime      0
goout         0
Dalc          0
Walc          0
health        0
absences      0
G1            0
G2            0
G3            0
dtype: int64
```

[7]:
```
numerical_cols = data.select_dtypes(include=[np.number]).columns
from scipy.stats import zscore
z_scores = data[numerical_cols].apply(zscore)
threshold = 3
outliers = (z_scores.abs() > threshold)
outlier_rows = data[outliers.any(axis=1)]

# Display rows with outliers
print("Outlier Rows:")
print(outlier_rows)
```

```
Outlier Rows:
    school sex  age address famsize Pstatus  Medu  Fedu      Mjob      Fjob  \
2       GP   F   15       U     LE3       T     1     1   at_home     other
18      GP   M   17       U     GT3       T     3     2  services  services
25      GP   F   16       U     GT3       T     2     2  services  services
29      GP   M   16       U     GT3       T     4     4   teacher   teacher
61      GP   F   16       U     GT3       T     1     1  services  services
66      GP   M   15       U     GT3       A     4     4     other  services
74      GP   F   16       U     GT3       T     3     3     other  services
78      GP   M   17       U     GT3       T     2     1     other     other
100     GP   M   16       U     GT3       T     4     4  services  services
108     GP   M   15       R     GT3       T     4     4     other     other
127     GP   F   19       U     GT3       T     0     1   at_home     other
134     GP   M   15       R     GT3       T     3     4   at_home   teacher
144     GP   M   17       U     GT3       T     2     1     other     other
146     GP   F   15       U     GT3       T     3     2    health  services
149     GP   M   15       U     LE3       A     2     1  services     other
150     GP   M   18       U     LE3       T     1     1     other     other
153     GP   M   19       U     GT3       T     3     2  services   at_home
157     GP   F   18       R     GT3       T     1     1   at_home     other
164     GP   M   17       R     LE3       T     1     1     other  services
173     GP   F   16       U     GT3       T     1     3   at_home  services
183     GP   F   17       U     LE3       T     3     3     other     other
184     GP   F   16       U     GT3       T     3     2     other     other
206     GP   F   16       U     GT3       A     3     1  services     other
```

207	GP	F	16	U	GT3	T	4	3	teacher	other
223	GP	M	18	U	GT3	T	2	2	other	other
228	GP	M	18	U	LE3	T	2	1	at_home	other
236	GP	M	17	U	LE3	T	2	2	other	other
247	GP	M	22	U	GT3	T	3	1	services	services
276	GP	F	18	R	GT3	A	3	2	other	services
280	GP	M	17	U	LE3	A	4	1	services	other
299	GP	M	18	U	LE3	T	4	4	teacher	teacher
307	GP	M	19	U	GT3	T	4	4	teacher	services
315	GP	F	19	R	GT3	T	2	3	other	other
327	GP	M	17	R	GT3	T	2	2	services	other
349	MS	M	18	R	GT3	T	3	2	other	other
350	MS	M	19	R	GT3	T	1	1	other	services
357	MS	F	17	U	LE3	A	3	2	services	other
375	MS	F	18	R	GT3	T	1	1	other	other
389	MS	F	18	U	GT3	T	1	1	other	other
392	MS	M	21	R	GT3	T	1	1	other	other

	…	famrel	freetime	goout	Dalc	Walc	health	absences	G1	G2	G3
2	…	4	3	2	2	3	3	10	7	8	10
18	…	5	5	5	2	4	5	16	6	5	5
25	…	1	2	2	1	3	5	14	6	9	8
29	…	4	4	5	5	5	5	16	10	12	11
61	…	5	5	5	5	5	5	6	10	8	11
66	…	1	3	3	5	5	3	4	13	13	12
74	…	4	3	3	2	4	5	54	11	12	11
78	…	4	5	1	1	1	3	2	8	8	10
100	…	4	5	5	5	5	4	14	7	7	5
108	…	1	3	5	3	5	1	6	10	13	13
127	…	3	4	2	1	1	5	2	7	8	9
134	…	5	3	3	1	1	5	0	9	0	0
144	…	5	4	5	1	2	5	0	5	0	0
146	…	3	3	2	1	1	3	0	6	7	0
149	…	4	5	5	2	5	5	0	8	9	10
150	…	2	3	5	2	5	4	0	6	5	0
153	…	4	5	4	1	1	4	0	5	0	0
157	…	5	2	5	1	5	4	6	9	8	10
164	…	5	3	5	1	5	5	0	5	8	7
173	…	4	3	5	1	1	3	0	8	7	0
183	…	5	3	3	2	3	1	56	9	9	8
184	…	1	2	2	1	2	1	14	12	13	12
206	…	2	3	3	2	2	4	5	7	7	7
207	…	1	3	2	1	1	1	10	11	12	13
223	…	3	3	3	5	5	4	0	12	13	13
228	…	4	3	2	4	5	3	14	10	8	9
236	…	4	4	2	5	5	4	4	14	13	13
247	…	5	4	5	5	5	1	16	6	8	8
276	…	4	1	1	1	1	5	75	10	9	9

```
280  …       4          5       4       2       4       5         30    8    8    8
299  …       1          4       2       2       2       1          5   16   15   16
307  …       4          3       4       1       1       4         38    8    9    8
315  …       4          1       2       1       1       3         40   13   11   11
327  …       4          4       5       5       5       4          8   11   10   10
349  …       2          5       5       5       5       5         10   11   13   13
350  …       5          4       4       3       3       2          8    8    7    8
357  …       1          2       3       1       2       5          2   12   12   11
375  …       4          3       2       1       2       4          2    8    8   10
389  …       1          1       1       1       1       5          0    6    5    0
392  …       5          5       3       3       3       3          3   10    8    7

[40 rows x 33 columns]
```

[9]:
```
# Select relevant columns and convert categorical variables
data = data[['school', 'sex', 'age', 'Pstatus', 'traveltime', 'studytime',
       'failures', 'schoolsup', 'famsup', 'health', 'absences', 'G1',␣
 ↪'G2','G3']]
```

[10]:
```
data.head()
```

[10]:
```
  school sex  age Pstatus  traveltime  studytime  failures schoolsup famsup  \
0     GP   F   18       A           2          2         0       yes     no
1     GP   F   17       T           1          2         0        no    yes
2     GP   F   15       T           1          2         3       yes     no
3     GP   F   15       T           1          3         0        no    yes
4     GP   F   16       T           1          2         0        no    yes

   health  absences  G1  G2  G3
0       3         6   5   6   6
1       3         4   5   5   6
2       3        10   7   8  10
3       5         2  15  14  15
4       5         4   6  10  10
```

[13]:
```
#converting the characterstical data into numericaldata by using label encoder
categorical_columns = ['school', 'sex', 'age', 'Pstatus', 'traveltime',␣
 ↪'studytime',
       'failures', 'schoolsup', 'famsup', 'health', 'absences', 'G1', 'G2','G3']
label_encoder = LabelEncoder()
for col in categorical_columns:
    data[col] = label_encoder.fit_transform(data[col])
```

[14]:
```
# Define target variable (y) and feature variables (X)
X = data.drop('G3', axis=1)  # Independent variables (features)
y = data['G3']  # Dependent variable (target)
```

[15]:
```
# 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)
```

[16]:
```
# Create and train Decision Tree Regressor model
model = DecisionTreeRegressor()
model.fit(X_train, y_train)
```

[16]:
```
DecisionTreeRegressor()
```

[17]:
```
y_pred = model.predict(X_test)
print(y_pred)
```

```
[ 5.  8.  5.  7.  6. 11. 16.  2.  0. 10. 11.  5. 12.  8. 11.  7.  3.  7.
 12.  0. 11. 12. 12.  2.  8. 15.  9.  7. 15.  7.  6.  8. 12. 10.  0.  5.
  0. 13.  8.  5.  3.  7. 11.  7. 13.  7. 10. 11. 10. 13. 10. 11.  7.  7.
  3. 10.  9.  0. 13. 12. 11.  7.  5.  4.  5. 15.  5.  7.  7. 13.  5.  8.
 10. 14.  8.  3.  7. 11.  5.]
```

[20]:
```
# Evaluate model performance
mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
rmse = math.sqrt(mse)
r2 = r2_score(y_test, y_pred)
mape = np.mean(np.abs((y_test - y_pred) / y_test)) * 100
mdae = np.median(np.abs(y_test - y_pred))
print("Mean Squared Error (MSE):", mse)
print("Mean Absolute Error (MAE):", mae)
print("Root Mean Squared Error (RMSE):", rmse)
print("Coefficient of Determination (R-squared):", r2)
print("Mean Absolute Percentage Error (MAPE):", mape)
print("Median Absolute Error (MdAE):", mdae)
```

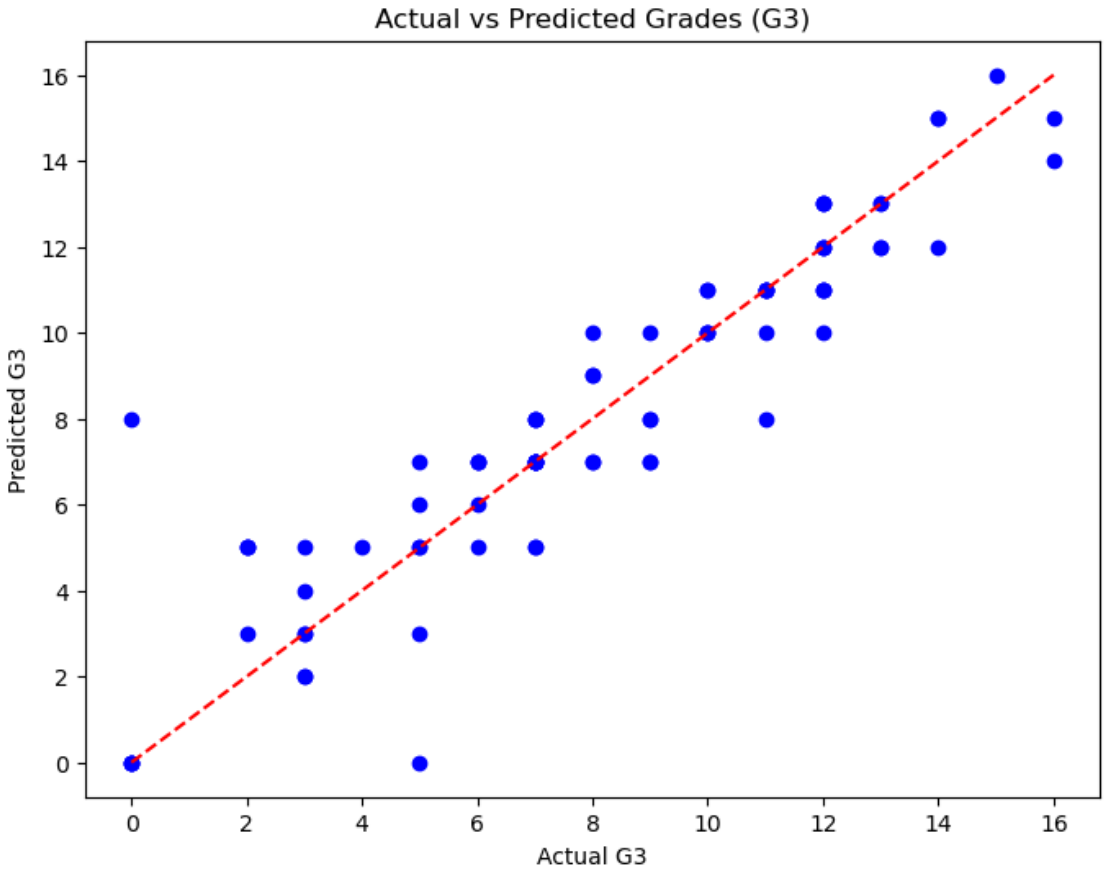
```
Mean Squared Error (MSE): 2.5822784810126582
Mean Absolute Error (MAE): 1.0379746835443038
Root Mean Squared Error (RMSE): 1.6069469440565418
Coefficient of Determination (R-squared): 0.8476345346594562
Mean Absolute Percentage Error (MAPE): inf
Median Absolute Error (MdAE): 1.0
```

[29]:
```
# to get the accuracey
print("ACCURACY :", 0.847 * 100)
```

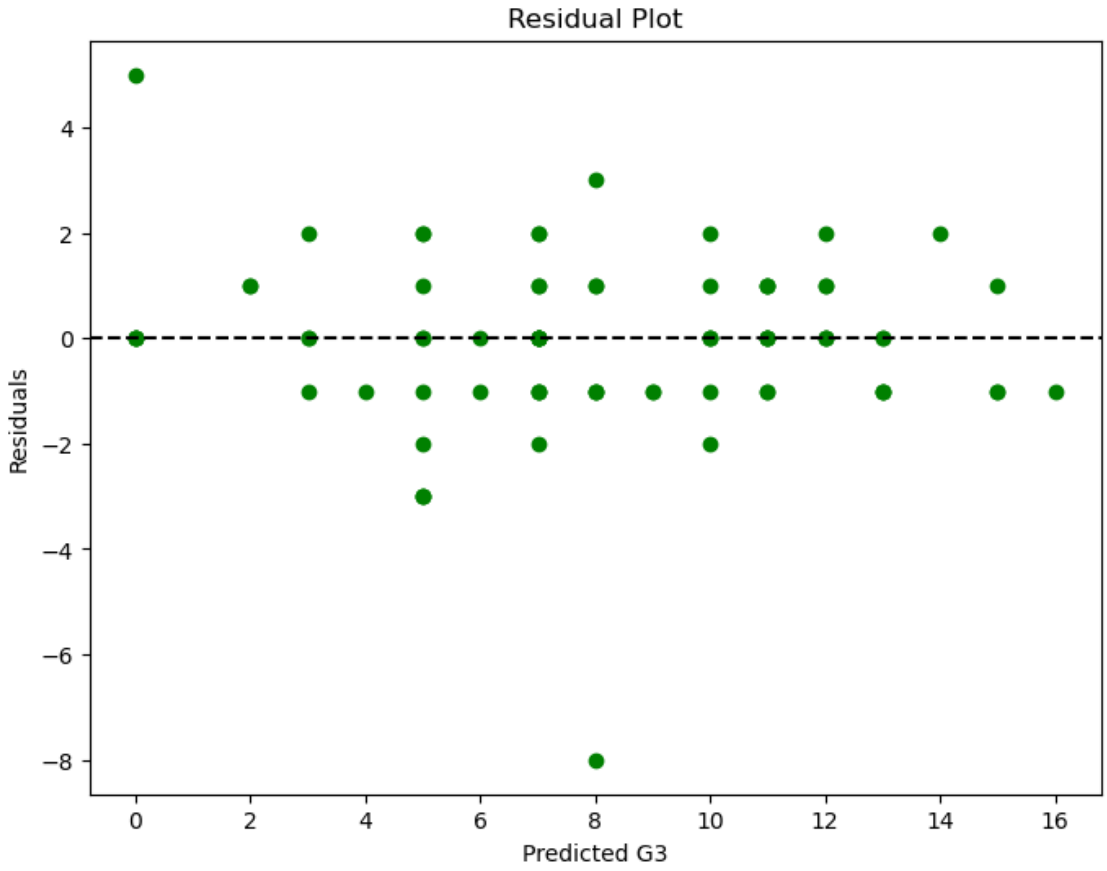
```
ACCURACY : 84.7
```

[25]:
```
# Scatter plot of actual vs. predicted values
plt.figure(figsize=(8,6))
plt.scatter(y_test, y_pred, color='blue')
```

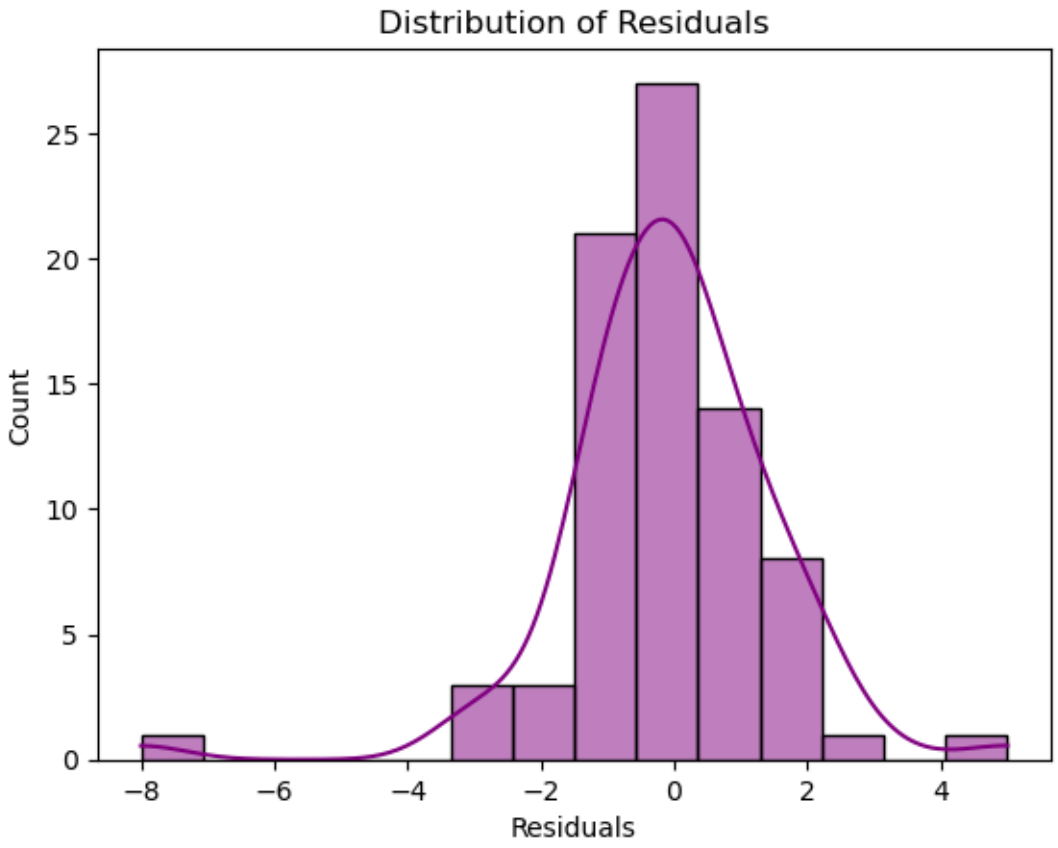
```
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='red',␣
 ↪linestyle='--')  # Ideal line (y = x)
plt.xlabel("Actual G3")
plt.ylabel("Predicted G3")
plt.title("Actual vs Predicted Grades (G3)")
plt.show()
```



[26]:
```
# Residual plot
residuals = y_test - y_pred
plt.figure(figsize=(8,6))
plt.scatter(y_pred, residuals, color='green')
plt.axhline(y=0, color='black', linestyle='--')
plt.xlabel("Predicted G3")
plt.ylabel("Residuals")
plt.title("Residual Plot")
plt.show()
```

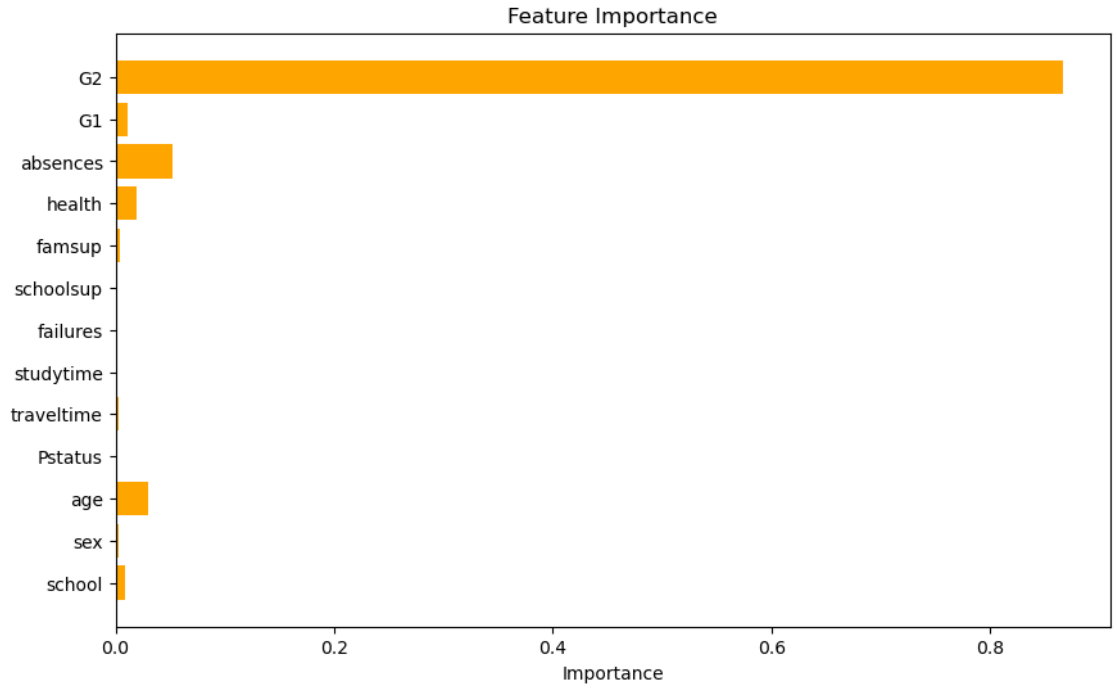


[28]:
```
#A histogram or KDE (Kernel Density Estimation) plot of residuals will show␣
 ↪whether errors are normally distributed.
#Normally distributed residuals indicate that your model is doing well.
# Plotting the distribution of residuals
sns.histplot(residuals, kde=True, color='purple')
plt.xlabel("Residuals")
plt.title("Distribution of Residuals")
plt.show()
```



[27]:
```
#Feature Importance (Optional, if you want to interpret your model)
#For decision trees or tree-based models, plotting feature importance can help␣
 ↪you understand which features
#(e.g., studytime, age, etc.) are the most important in predicting the final␣
 ↪grade.
# Feature importance plot for decision tree models
feature_importances = model.feature_importances_
features = X.columns

plt.figure(figsize=(10,6))
plt.barh(features, feature_importances, color='orange')
plt.xlabel("Importance")
plt.title("Feature Importance")
plt.show()
```


Feature Importance
G2
G1
absences
health
famsup
schoolsup
failures
studytime
traveltime
Pstatus
age
sex
school
0.0
0.2
0.4
0.6
0.8
Importance

[ ]: