Data Analysis

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df1 = pd.read_csv("C:\\Users\\visha\\Desktop\\Final Project\\guvi-courses.csv")
df1.head()
```

Out[2]:

	course_id	course_title	url	price	num_subscribers	num_reviews	num
0	41295.0	Learn HTML5 Programming From Scratch	https://www.udemy.com/learn-html5-programming-...	0.0	268923.0	8629.0	
1	59014.0	Coding for Entrepreneurs Basic	https://www.udemy.com/coding-for-entrepreneurs...	0.0	161029.0	279.0	
2	625204.0	The Web Developer Bootcamp	https://www.udemy.com/the-web-developer-bootcamp/	200.0	121584.0	27445.0	
3	173548.0	Build Your First Website in 1 Week with HTML5 ...	https://www.udemy.com/build-your-first-website...	0.0	120291.0	5924.0	
4	764164.0	The Complete Web Developer Course 2.0	https://www.udemy.com/the-complete-web-develop...	200.0	114512.0	22412.0	

In [3]:
```
df1.shape
```

Out[3]:
```
(3680, 12)
```

In [4]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3680 entries, 0 to 3679
Data columns (total 12 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   course_id            3676 non-null   float64
 1   course_title         3676 non-null   object 
 2   url                  3676 non-null   object 
 3   price                3676 non-null   float64
 4   num_subscribers      3676 non-null   float64
 5   num_reviews          3676 non-null   float64
 6   num_lectures         3676 non-null   float64
 7   level                3676 non-null   object 
 8   Rating               3676 non-null   float64
 9   content_duration     3676 non-null   float64
 10  published_timestamp  3676 non-null   object 
 11  subject              3677 non-null   object 
dtypes: float64(7), object(5)
memory usage: 345.1+ KB
```

In [5]:
```
df1.isnull().sum()
```

Out[5]:
```
course_id              4
course_title           4
url                    4
price                  4
num_subscribers        4
num_reviews            4
num_lectures           4
level                  4
Rating                 4
content_duration       4
published_timestamp    4
subject                3
dtype: int64
```

Handling Missing Data for required columns

In [6]:
```
x = df1['course_title'].unique()
x
```

Out[6]:
```
array(['Learn HTML5 Programming From Scratch',
       'Coding for Entrepreneurs Basic', 'The Web Developer Bootcamp',
       ..., 'Learn Pirates of the Caribbean by Ear on the Piano',
       'Learn to Play Piano Like a Pro - Easy Piano Course 1',
       '4 Week Rhythm Mastery'], dtype=object)
```

In [7]:
```
print(len(x))
```

```
3664
```

In [8]:
```
df = df1.dropna()
```

In [9]:
```
df.isnull().sum()
```

Out[9]:
```
course_id              0
course_title           0
url                    0
price                  0
num_subscribers        0
num_reviews            0
num_lectures           0
level                  0
Rating                 0
content_duration       0
published_timestamp    0
subject                0
dtype: int64
```

In [10]:
```
df.columns
```

Out[10]:
```
Index(['course_id', 'course_title', 'url', 'price', 'num_subscribers',
       'num_reviews', 'num_lectures', 'level', 'Rating', 'content_duration',
       'published_timestamp', 'subject'],
      dtype='object')
```

Retained required Fields

In [11]:
```
df = df[['course_title', 'price', 'num_subscribers', 'num_reviews', 'num_lectures', 'l
```

In [12]:
```
df.head()
```

Out[12]:

	course_title	price	num_subscribers	num_reviews	num_lectures	level	Rating	subject
0	Learn HTML5 Programming From Scratch	0.0	268923.0	8629.0	45.0	Beginner Level	0.82	Subject: Web Development
1	Coding for Entrepreneurs Basic	0.0	161029.0	279.0	27.0	Expert Level	0.69	Subject: Web Development
2	The Web Developer Bootcamp	200.0	121584.0	27445.0	342.0	Beginner Level	0.89	Subject: Web Development
3	Build Your First Website in 1 Week with HTML5 ...	0.0	120291.0	5924.0	30.0	All Levels	0.78	Subject: Web Development
4	The Complete Web Developer Course 2.0	200.0	114512.0	22412.0	304.0	Beginner Level	0.55	Subject: Web Development

In [13]:
```
df.isna().sum()
```

Out[13]:
```
course_title       0
price              0
num_subscribers    0
num_reviews        0
num_lectures       0
level              0
Rating             0
subject            0
dtype: int64
```

In [14]:
```
df.shape
```

Out[14]:
```
(3676, 8)
```

In [15]:
```
df_x = df['course_title'].unique()
print(len(df_x))
```

```
3663
```

In [16]:
```
df.head()
```

Out[16]:

	course_title	price	num_subscribers	num_reviews	num_lectures	level	Rating	subject
0	Learn HTML5 Programming From Scratch	0.0	268923.0	8629.0	45.0	Beginner Level	0.82	Subject: Web Development
1	Coding for Entrepreneurs Basic	0.0	161029.0	279.0	27.0	Expert Level	0.69	Subject: Web Development
2	The Web Developer Bootcamp	200.0	121584.0	27445.0	342.0	Beginner Level	0.89	Subject: Web Development
3	Build Your First Website in 1 Week with HTML5 ...	0.0	120291.0	5924.0	30.0	All Levels	0.78	Subject: Web Development
4	The Complete Web Developer Course 2.0	200.0	114512.0	22412.0	304.0	Beginner Level	0.55	Subject: Web Development

In [17]:
```
df.columns
```

Out[17]:
```
Index(['course_title', 'price', 'num_subscribers', 'num_reviews',
       'num_lectures', 'level', 'Rating', 'subject'],
      dtype='object')
```

In [18]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3676 entries, 0 to 3679
Data columns (total 8 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   course_title     3676 non-null   object 
 1   price            3676 non-null   float64
 2   num_subscribers  3676 non-null   float64
 3   num_reviews      3676 non-null   float64
 4   num_lectures     3676 non-null   float64
 5   level            3676 non-null   object 
 6   Rating           3676 non-null   float64
 7   subject          3676 non-null   object 
dtypes: float64(5), object(3)
memory usage: 258.5+ KB
```

In [19]:
```
df['subject'].unique()
```

Out[19]:
```
array(['Subject: Web Development', 'Business Finance', 'Graphic Design',
       'Musical Instruments'], dtype=object)
```

In [20]:
```
df['subject'].value_counts()
```

Out[20]:
```
Subject: Web Development    1203
Business Finance            1191
Musical Instruments          680
Graphic Design               602
Name: subject, dtype: int64
```

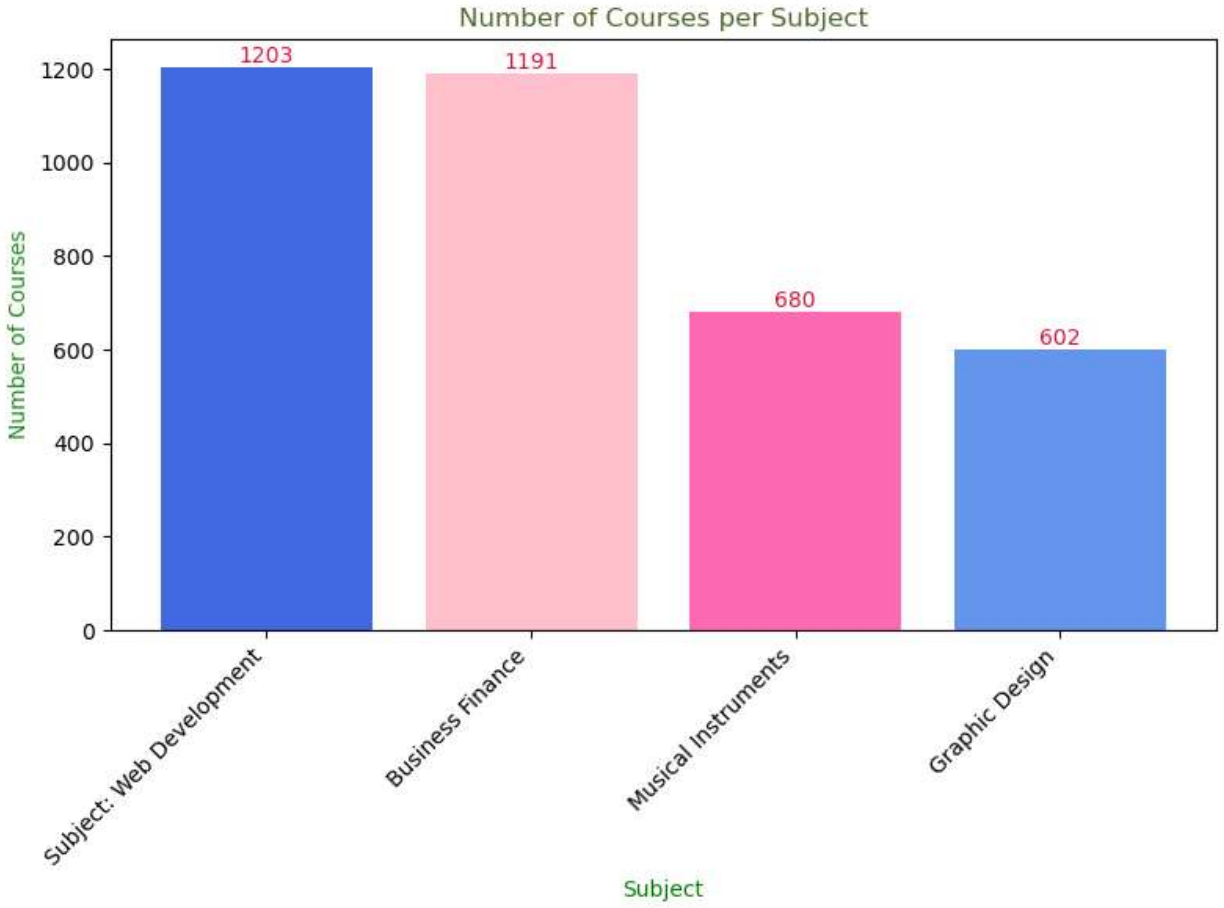
In [21]:
```
df.isna().sum()
```

Out[21]:
```
course_title       0
price              0
num_subscribers    0
num_reviews        0
num_lectures       0
level              0
Rating             0
subject            0
dtype: int64
```

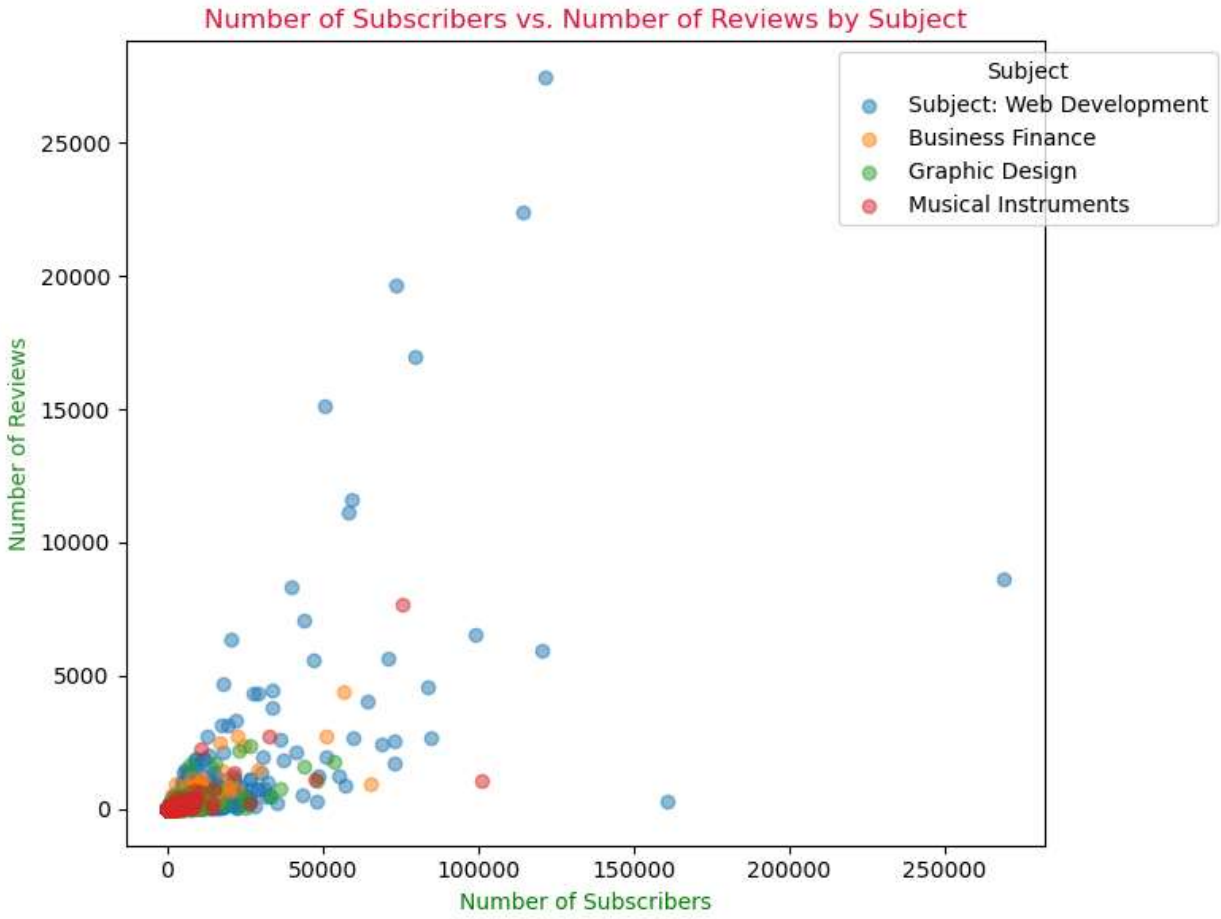
Data Visualization

In [22]:
```
subject_counts = df['subject'].value_counts().reset_index()
subject_counts.columns = ['Subject', 'Number of Courses']


plt.figure(figsize=(8, 6))
bars = plt.bar(subject_counts['Subject'], subject_counts['Number of Courses'], color=[
plt.title('Number of Courses per Subject', color='darkolivegreen')
plt.xlabel('Subject', color='green')
plt.ylabel('Number of Courses', color='forestgreen')
plt.xticks(rotation=45, ha='right')


for bar, count in zip(bars, subject_counts['Number of Courses']):
    plt.text(bar.get_x() + bar.get_width() / 2, bar.get_height(), str(count), ha='cent
```

```
plt.tight_layout()
plt.show()
```



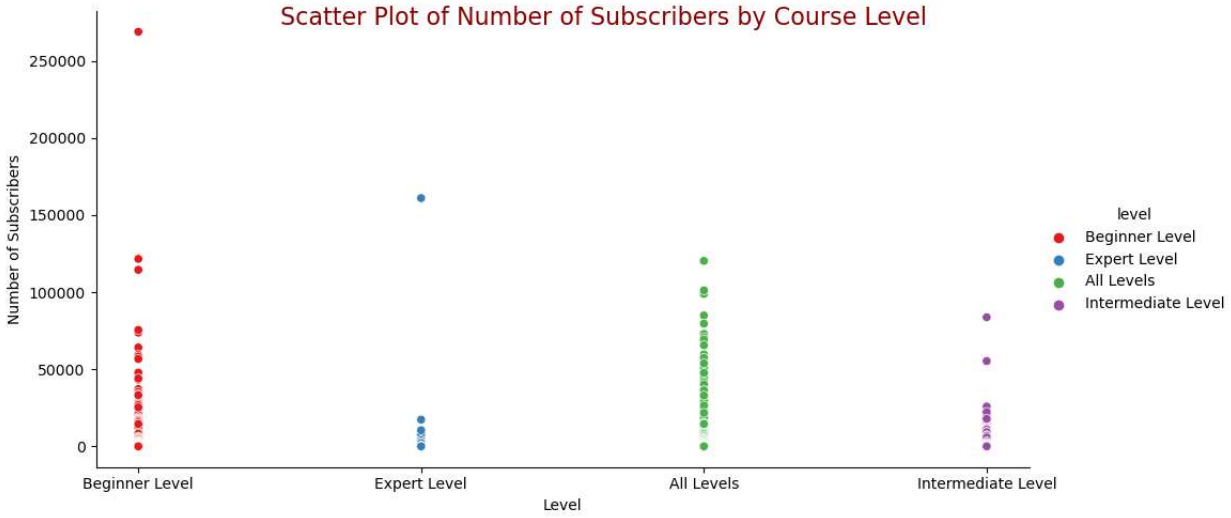
In [23]:
```
plt.figure(figsize=(8, 6))

subjects = df['subject'].unique()

for subject in subjects:
    subset = df[df['subject'] == subject]
    plt.scatter(subset['num_subscribers'], subset['num_reviews'], label=subject, alpha

plt.title('Number of Subscribers vs. Number of Reviews by Subject', color='crimson')
plt.xlabel('Number of Subscribers', color='green')
plt.ylabel('Number of Reviews', color='forestgreen')
plt.legend(title='Subject', loc='upper right', bbox_to_anchor=(1.2, 1))


plt.tight_layout()
plt.show()
```

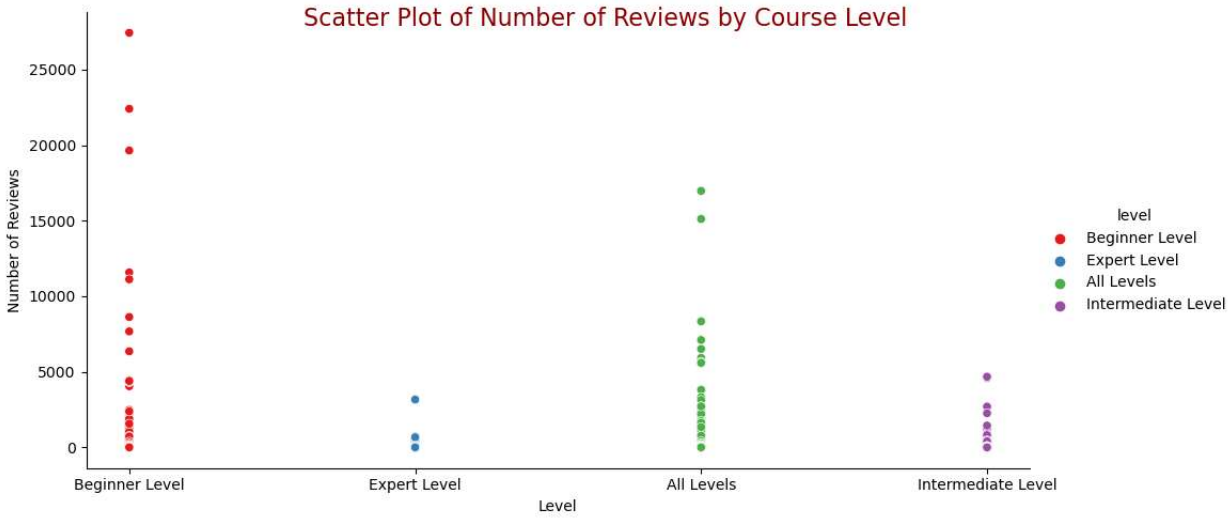


In [24]:
```
plt.figure(figsize=(10, 6))
sns.lineplot(data=df, x='level', y='price', sort=False, color='royalblue')


plt.title('Price by Course Level', fontsize=16, color='darkred')
plt.xlabel('Level', fontsize=12, color='blue')
plt.ylabel('Price', fontsize=12, color='green')


plt.xticks(rotation=45, ha='right')
plt.grid(axis='y', alpha=0.75)
plt.tight_layout()
plt.show()
```

In [25]:
```
g = sns.relplot(data=df, x='level', y='price', hue='level', palette='Set1', aspect=2)
g.set_axis_labels('Level', 'Price')
g.fig.suptitle('Scatter Plot of Price by Course Level', fontsize=16, color='darkred')

g = sns.relplot(data=df, x='level', y='num_subscribers', hue='level', palette='Set1',
g.set_axis_labels('Level', 'Number of Subscribers')
g.fig.suptitle('Scatter Plot of Number of Subscribers by Course Level', fontsize=16, c

g = sns.relplot(data=df, x='level', y='num_reviews', hue='level', palette='Set1', aspe
g.set_axis_labels('Level', 'Number of Reviews')
g.fig.suptitle('Scatter Plot of Number of Reviews by Course Level', fontsize=16, color

g = sns.relplot(data=df, x='level', y='Rating', hue='level', palette='Set1', aspect=2)
g.set_axis_labels('Level', 'Rating')
g.fig.suptitle('Scatter Plot of Rating by Course Level', fontsize=16, color='darkred')

# Show the plot with legends
plt.subplots_adjust(top=0.9)
plt.xticks(rotation=45, ha='right')
plt.grid(axis='both', alpha=0.75)
plt.tight_layout()
plt.show()
```

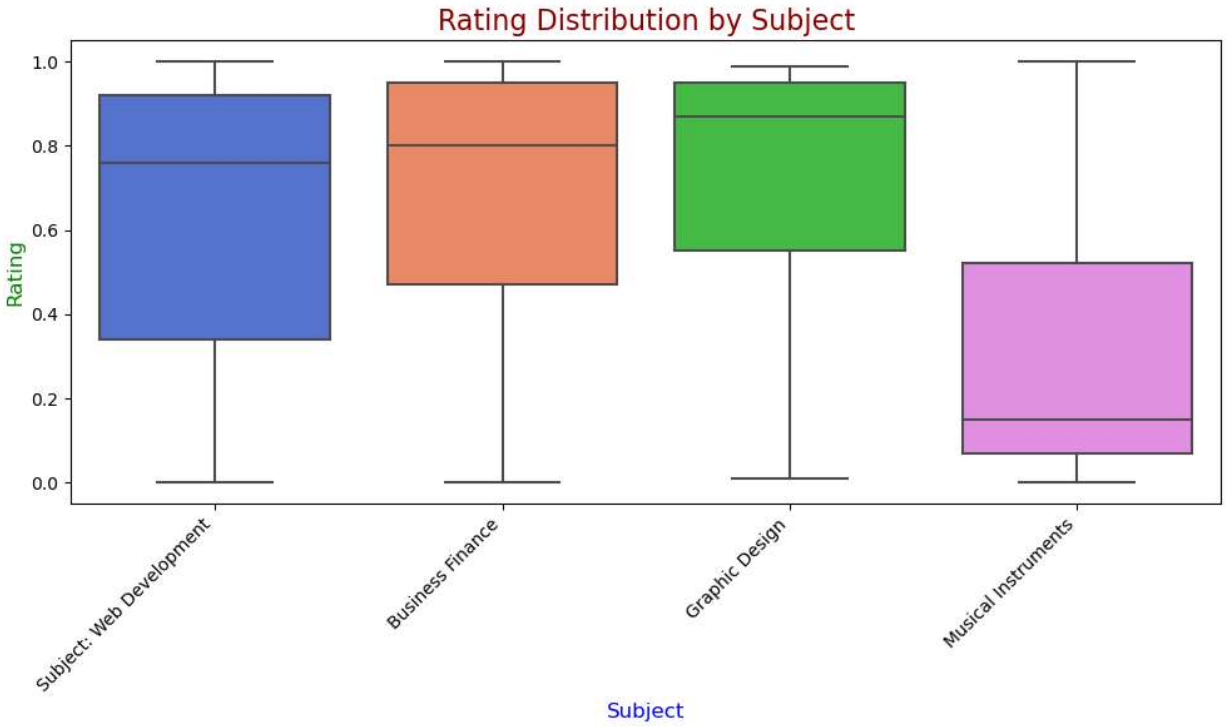

Scatter Plot of Price by Course Level
Price
Level
Beginner Level
Expert Level
All Levels
Intermediate Level
level


Scatter Plot of Number of Subscribers by Course Level
Number of Subscribers
Level
Beginner Level
Expert Level
All Levels
Intermediate Level
level


Scatter Plot of Number of Reviews by Course Level
Number of Reviews
Level
Beginner Level
Expert Level
All Levels
Intermediate Level
level

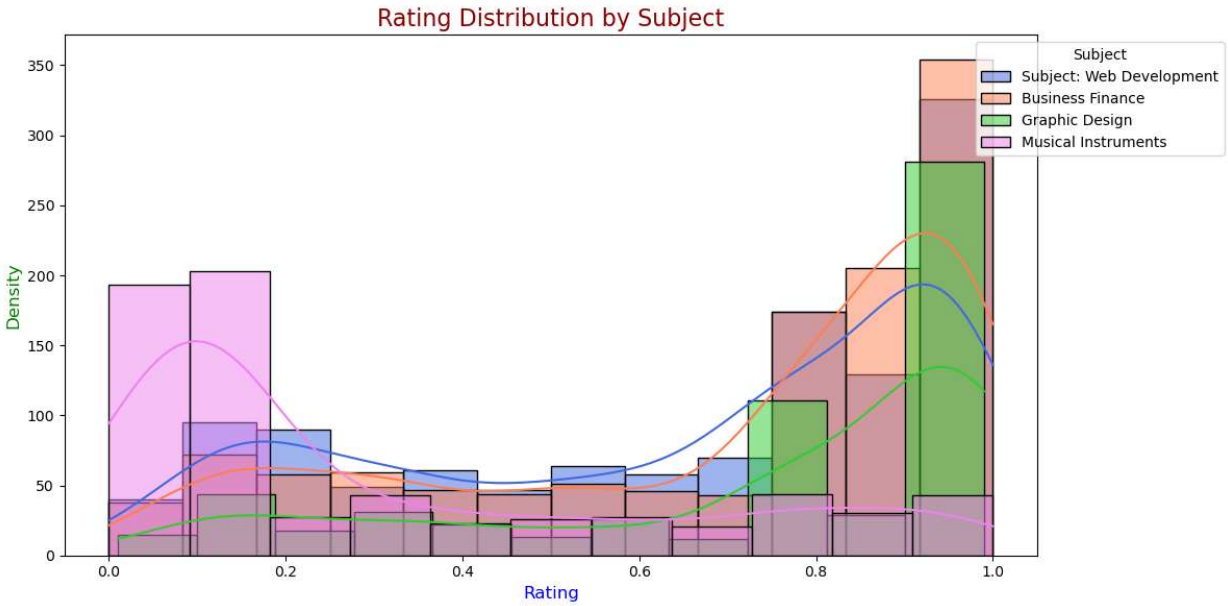
Distribution

In [26]:
```
box_colors = ['royalblue', 'coral', 'limegreen', 'violet']



plt.figure(figsize=(10, 6))
sns.boxplot(x='subject', y='Rating', data=df, palette=box_colors)

plt.title('Rating Distribution by Subject', fontsize=16, color='darkred')
plt.xlabel('Subject', fontsize=12, color='blue')
plt.ylabel('Rating', fontsize=12, color='green')



plt.xticks(rotation=45, ha='right')
# plt.grid(axis='y', alpha=0.75)

plt.tight_layout()
plt.show()
```



In [27]:

```
hist_colors = ['royalblue', 'coral', 'limegreen', 'violet']


plt.figure(figsize=(12, 6))

for i, subject in enumerate(df['subject'].unique()):
    subset = df[df['subject'] == subject]
    sns.histplot(data=subset, x='Rating', color=hist_colors[i], label=subject, kde=Tru


plt.title('Rating Distribution by Subject', fontsize=16, color='darkred')
plt.xlabel('Rating', fontsize=12, color='blue')
plt.ylabel('Density', fontsize=12, color='green')


plt.legend(title='Subject', loc='upper right', bbox_to_anchor=(1.2, 1))


plt.tight_layout()
plt.show()
```



Data Spliting

In [28]:
```
X = df[['course_title', 'price', 'num_subscribers', 'num_reviews',
       'num_lectures', 'level', 'Rating', 'subject']]
y = df['Rating']
```

Label Encode - Course Title

In [29]:
```
from sklearn.preprocessing import LabelEncoder

label_encoder = LabelEncoder()

X['course_title_Label'] = label_encoder.fit_transform(X['course_title'])

X['level_Label'] = label_encoder.fit_transform(X['level'])

X['subject_Label'] = label_encoder.fit_transform(X['subject'])
```

In [30]:
```
X.columns
```

Out[30]:
```
Index(['course_title', 'price', 'num_subscribers', 'num_reviews',
       'num_lectures', 'level', 'Rating', 'subject', 'course_title_Label',
       'level_Label', 'subject_Label'],
      dtype='object')
```

In [31]:
```
X = X[['price', 'num_subscribers', 'num_reviews', 'num_lectures', 'course_title_Label'
```

In [32]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=
```

Regression Model

In [33]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score


model = LinearRegression()

# X_train = X_train.values.reshape(-1, 1)
# X_test = X_test.values.reshape(-1, 1)


model.fit(X_train, y_train)

y_predict = model.predict(X_test)

mse = mean_squared_error(y_test, y_predict)
r2 = r2_score(y_test, y_predict)

print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

```
Mean Squared Error: 0.10690407605031713
R-squared: 0.02900287500489962
```

Random Forest

In [34]:
```
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score
```

In [35]:
```
random_forest = RandomForestRegressor(n_estimators=100, random_state=42)

# Train the model on the training data

random_forest.fit(X_train, y_train)

# Make predictions on the testing data

y_predic = random_forest.predict(X_test)

# Evaluate the model

mse = mean_squared_error(y_test, y_predic)
r2 = r2_score(y_test, y_predic)

print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

```
Mean Squared Error: 0.07191285720543479
R-squared: 0.34682399234436645
```

Gradient Boosting Regressor

In [36]:
```
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score
```

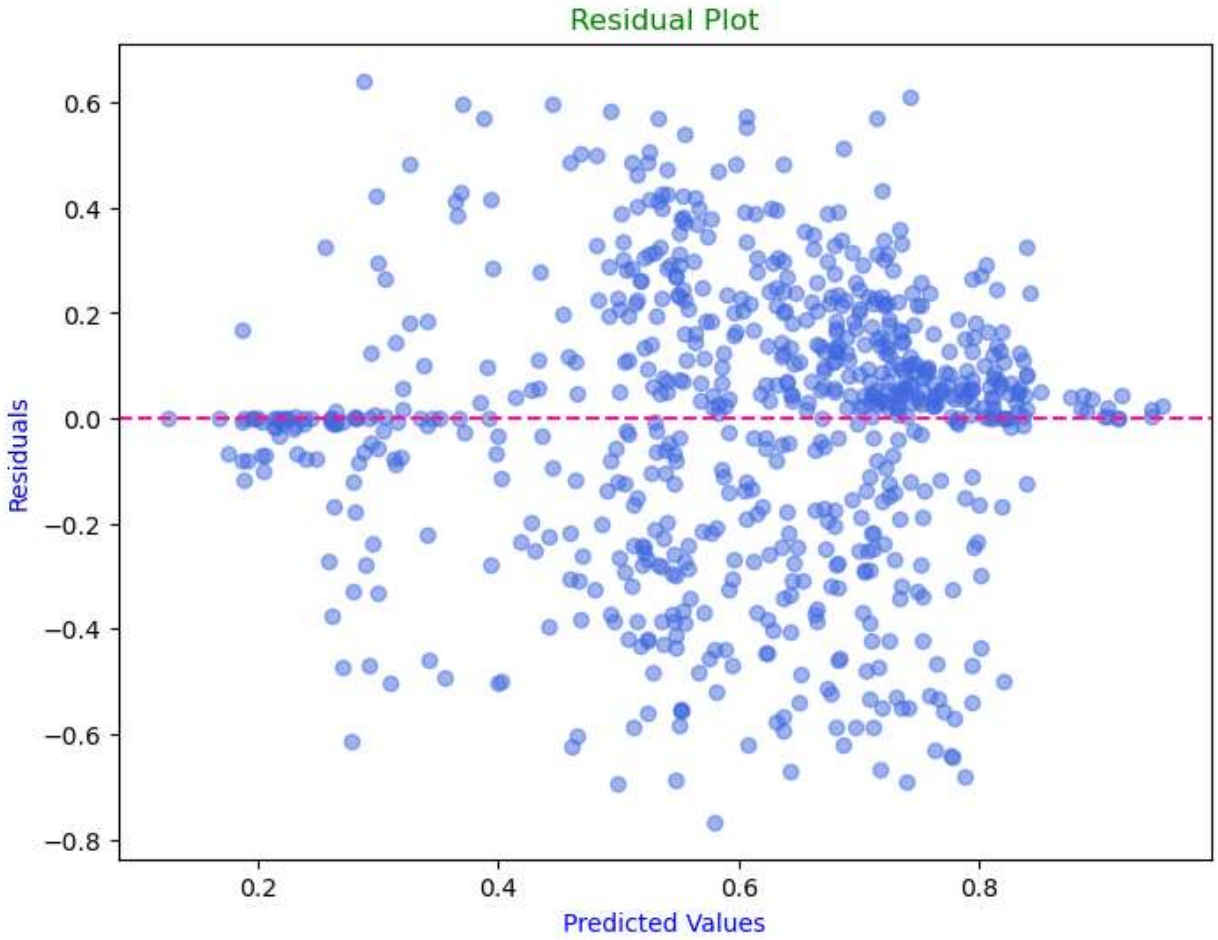
In [37]:
```
gradient_boosting = GradientBoostingRegressor(n_estimators=100, learning_rate=0.1, max

# Train the model on the training data
gradient_boosting.fit(X_train, y_train)

# Make predictions on the testing data
y_pred = gradient_boosting.predict(X_test)

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

print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

```
Mean Squared Error: 0.07587483541036925
R-squared: 0.3108378111956337
```

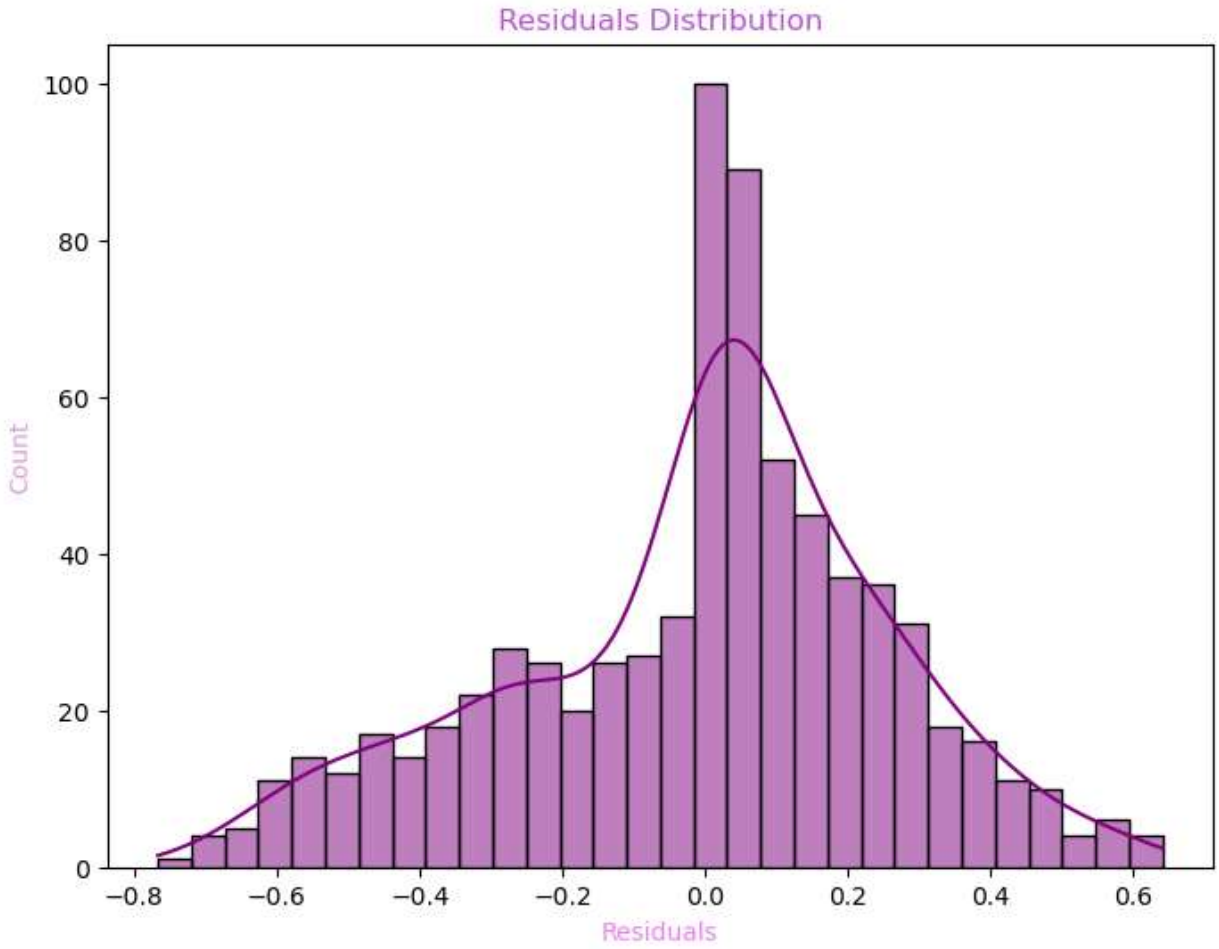
Prediction Visualization

Random Forest

In [38]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Calculate residuals
residuals = y_test - y_predic  # actual - predict


plt.figure(figsize=(8, 6))
plt.scatter(y_pred, residuals, alpha=0.5, c='royalblue')
plt.title('Residual Plot', color='green')
plt.xlabel('Predicted Values', color='blue')
plt.ylabel('Residuals', color='mediumblue')
plt.axhline(0, color='deeppink', linestyle='--')
plt.show()
```

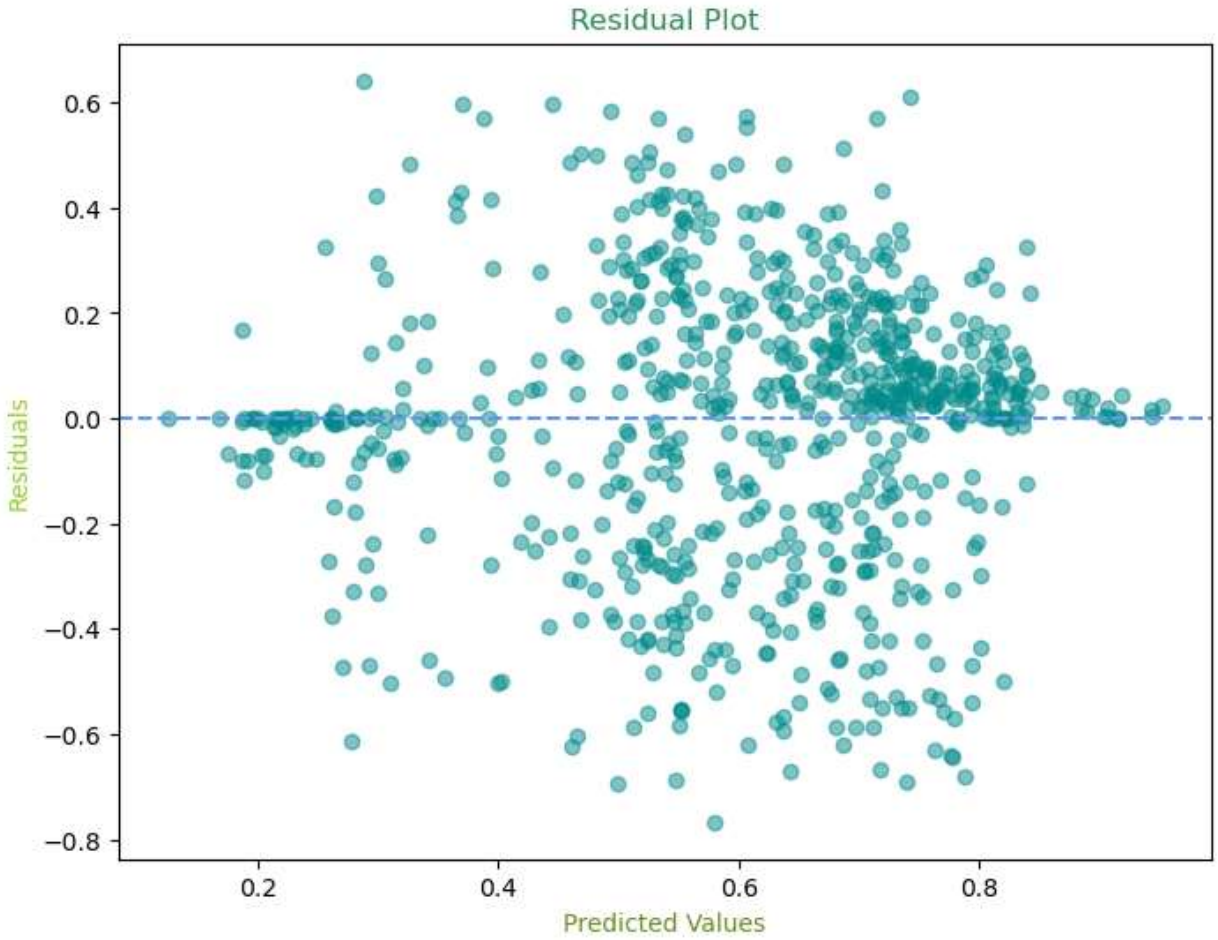


In [39]:
```
# Create a histogram of residuals
plt.figure(figsize=(8, 6))
sns.histplot(residuals, kde=True, bins=30, color='purple')
plt.title('Residuals Distribution', color='mediumorchid')
plt.xlabel('Residuals', color='violet')
plt.ylabel('Count',color='plum')
plt.show()
```

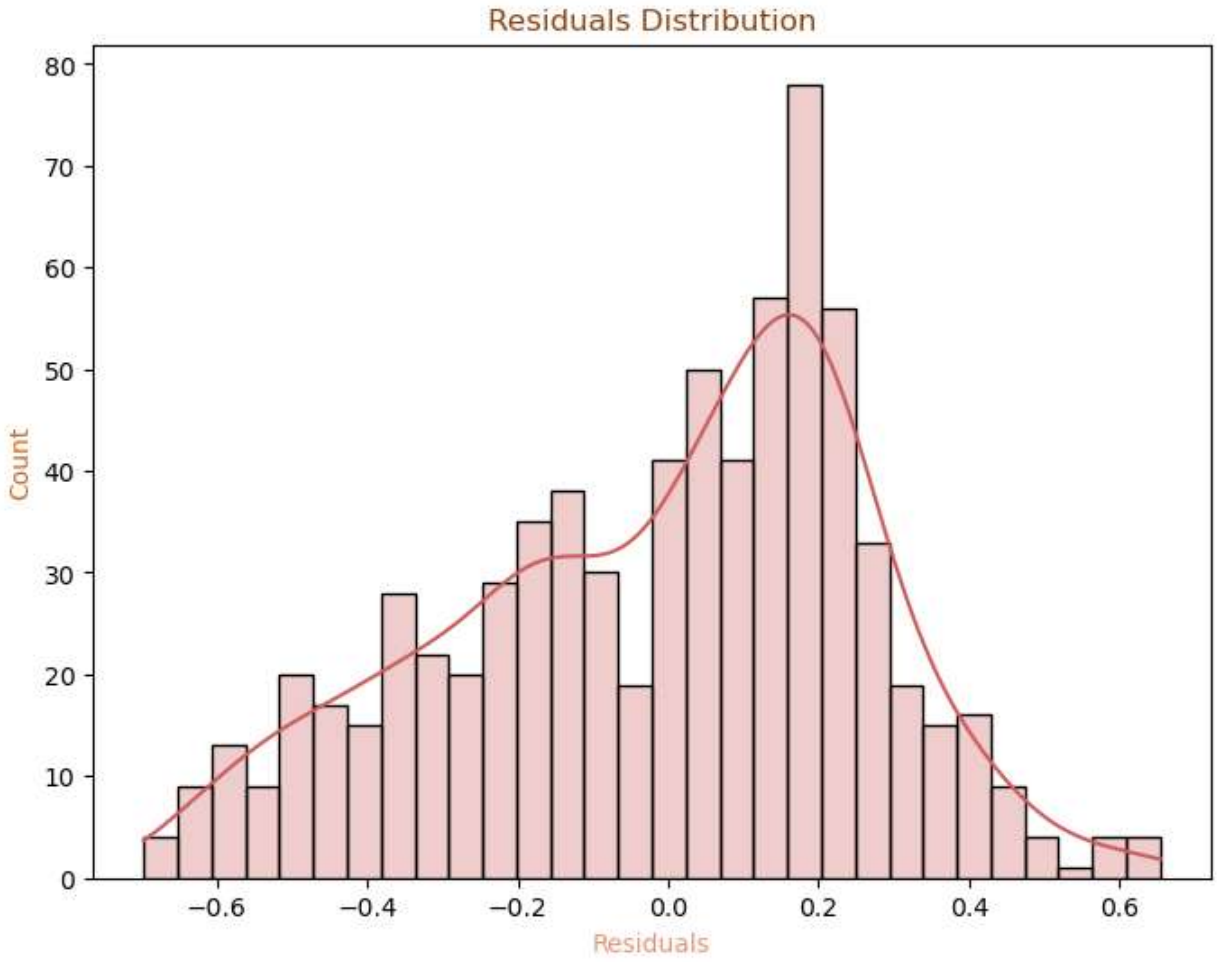


Gradient Boosting Regressor

In [40]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Calculate residuals
residual = y_test - y_pred  # actual - predict


plt.figure(figsize=(8, 6))
plt.scatter(y_pred, residuals, alpha=0.5, c='darkcyan')
plt.title('Residual Plot', color='seagreen')
plt.xlabel('Predicted Values', color='olivedrab')
plt.ylabel('Residuals', color='yellowgreen')
plt.axhline(0, color='cornflowerblue', linestyle='--')
plt.show()
```



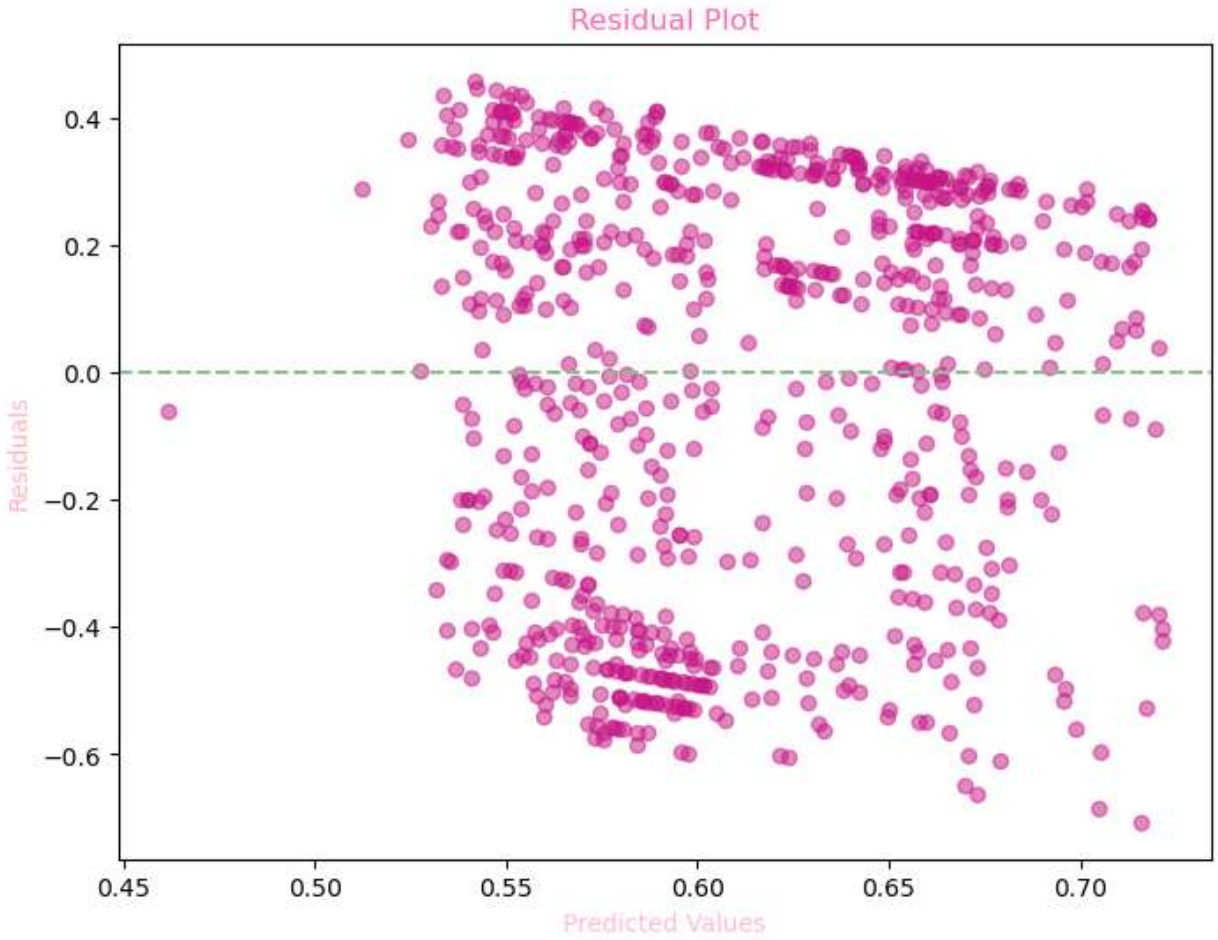
In [41]:
```
# Create a histogram of residuals
plt.figure(figsize=(8, 6))
sns.histplot(residual, kde=True, bins=30, color='indianred', alpha=0.3)
plt.title('Residuals Distribution', color='saddlebrown')
plt.xlabel('Residuals', color='darksalmon')
plt.ylabel('Count',color='chocolate')
plt.show()
```



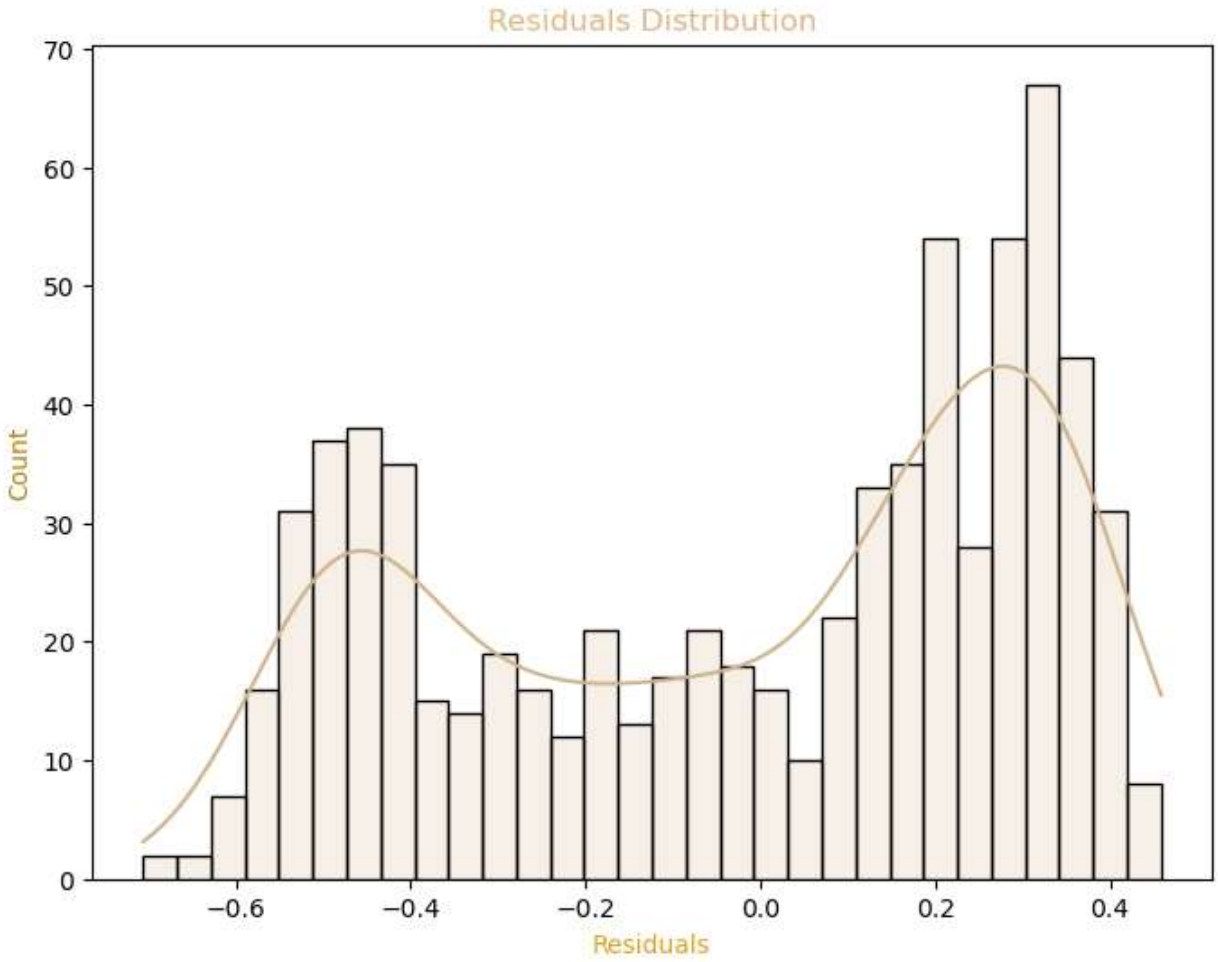
Regression Model

In [42]:

```
import matplotlib.pyplot as plt
import seaborn as sns

# Calculate residuals
resi2 = y_test - y_predict  # actual - predict


plt.figure(figsize=(8, 6))
plt.scatter(y_predict, resi2, alpha=0.5, c='mediumvioletred')
plt.title('Residual Plot', color='hotpink')
plt.xlabel('Predicted Values', color='pink')
plt.ylabel('Residuals', color='lightpink')
plt.axhline(0, color='darkseagreen', linestyle='--')
plt.show()
```



In [43]:
```
# Create a histogram of residuals
plt.figure(figsize=(8, 6))
sns.histplot(resi2, kde=True, bins=30, color='tan', alpha=0.2)
plt.title('Residuals Distribution', color='burlywood')
plt.xlabel('Residuals', color='goldenrod')
plt.ylabel('Count',color='darkgoldenrod')
plt.show()
```



Data Analysis and Modeling Summary

Data Analysis and Preprocessing:

- Imported and inspected the dataset.
- Handled missing data for required columns.
- Explored the dataset and retained relevant fields.

Visualized data with informative plots:

- Number of courses per subject.
- Comparisons of subscribers and reviews by subject.
- Price distribution by course level.
- Scatter plots of price, subscribers, reviews, and rating by level.
- Box plots and histograms of rating by subject.

Data Splitting and Regression Modeling:

- Split the data into features (X) and the target variable (y).
- Applied label encoding to course title, level, and subject.
- Performed data splitting into training and testing sets.
- Implemented three regression models:

  > - Linear Regression
  > - Random Forest Regression
  > - Gradient Boosting Regression.

- Evaluated model performance using Mean Squared Error (MSE) and R-squared (R^2).

Visualization of Model Predictions:

- Visualized model predictions using residual plots and histograms.
- Calculated residuals (actual - predicted) and created scatter plots.
- Constructed histograms to visualize the distribution of residuals for each model.

This comprehensive analysis and modeling process aimed to predict course ratings and gain insights from the data.