# Student Satisfaction Survey

About Dataset

The dataset is regarding survey conducted at an autonomous college on college faculties performance and treatment towards students at the college. Questionnaire of 20 being rolled out to each department available at the college and randomly given to sampled list of students for the feedback.

In [1]:
```
import pandas as pd

# Load the dataset
data = pd.read_csv('student_feedback.csv')
# Display the first few rows
data.head()
```

Out[1]:

| | Unnamed: 0 | Student ID | Well versed with the subject | Explains concepts in an understandable way | Use of presentations | Degree of difficulty of assignments | Solves doubts willingly | Structuring of the course | P s st I |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | 340 | 5 | 2 | 7 | 6 | 9 | 2 | |
| **1** | 1 | 253 | 6 | 5 | 8 | 6 | 2 | 1 | |
| **2** | 2 | 680 | 7 | 7 | 6 | 5 | 4 | 2 | |
| **3** | 3 | 806 | 9 | 6 | 7 | 1 | 5 | 9 | |
| **4** | 4 | 632 | 8 | 10 | 8 | 4 | 6 | 6 | |

# Data Pre-Processing

In [2]:
```
# Check for missing values and data types
# Remove unnamed columns and student ID
df = data.drop(columns=["Unnamed: 0", "Student ID"])
#df.isnull().any()
#df.isnull().sum()
df.isnull().sum().sum()
```

Out[2]:
```
0
```

In [3]:
```
df.dtypes
```

Out[3]:
```
Well versed with the subject                                  int64
Explains concepts in an understandable way                    int64
Use of presentations                                          int64
Degree of difficulty of assignments                           int64
Solves doubts willingly                                       int64
Structuring of the course                                     int64
Provides support for students going above and beyond          int64
Course recommendation based on relevance                      int64
dtype: object
```

In [4]:
```
# Summary statistics
df.describe()
```

Out[4]:

| | Well versed with the subject | Explains concepts in an understandable way | Use of presentations | Degree of difficulty of assignments | Solves doubts willingly | Structuring of the course | Provide support fo student goin above an beyon |
|---|---|---|---|---|---|---|---|
| **count** | 1001.000000 | 1001.000000 | 1001.000000 | 1001.000000 | 1001.000000 | 1001.000000 | 1001.00000 |
| **mean** | 7.497502 | 6.081918 | 5.942058 | 5.430569 | 5.474525 | 5.636364 | 5.66233 |
| **std** | 1.692998 | 2.597168 | 1.415853 | 2.869046 | 2.874648 | 2.920212 | 2.89169 |
| **min** | 5.000000 | 2.000000 | 4.000000 | 1.000000 | 1.000000 | 1.000000 | 1.00000 |
| **25%** | 6.000000 | 4.000000 | 5.000000 | 3.000000 | 3.000000 | 3.000000 | 3.00000 |
| **50%** | 8.000000 | 6.000000 | 6.000000 | 5.000000 | 6.000000 | 6.000000 | 6.00000 |
| **75%** | 9.000000 | 8.000000 | 7.000000 | 8.000000 | 8.000000 | 8.000000 | 8.00000 |
| **max** | 10.000000 | 10.000000 | 8.000000 | 10.000000 | 10.000000 | 10.000000 | 10.00000 |

In [5]:
```
import numpy as np
# Display dataset shape and first few rows
print(f"Dataset Shape: {df.shape}")
display(df.head())

rating_cols = df.select_dtypes(include=[np.number]).columns.tolist()
print("Numeric Columns for Ratings:", rating_cols)

# Fill missing values with column mean
df[rating_cols] = df[rating_cols].fillna(df[rating_cols].mean())

# Clip ratings to valid range (1-10)
df[rating_cols] = df[rating_cols].clip(lower=1, upper=10)

# Show summary statistics
display(df[rating_cols].describe())
```

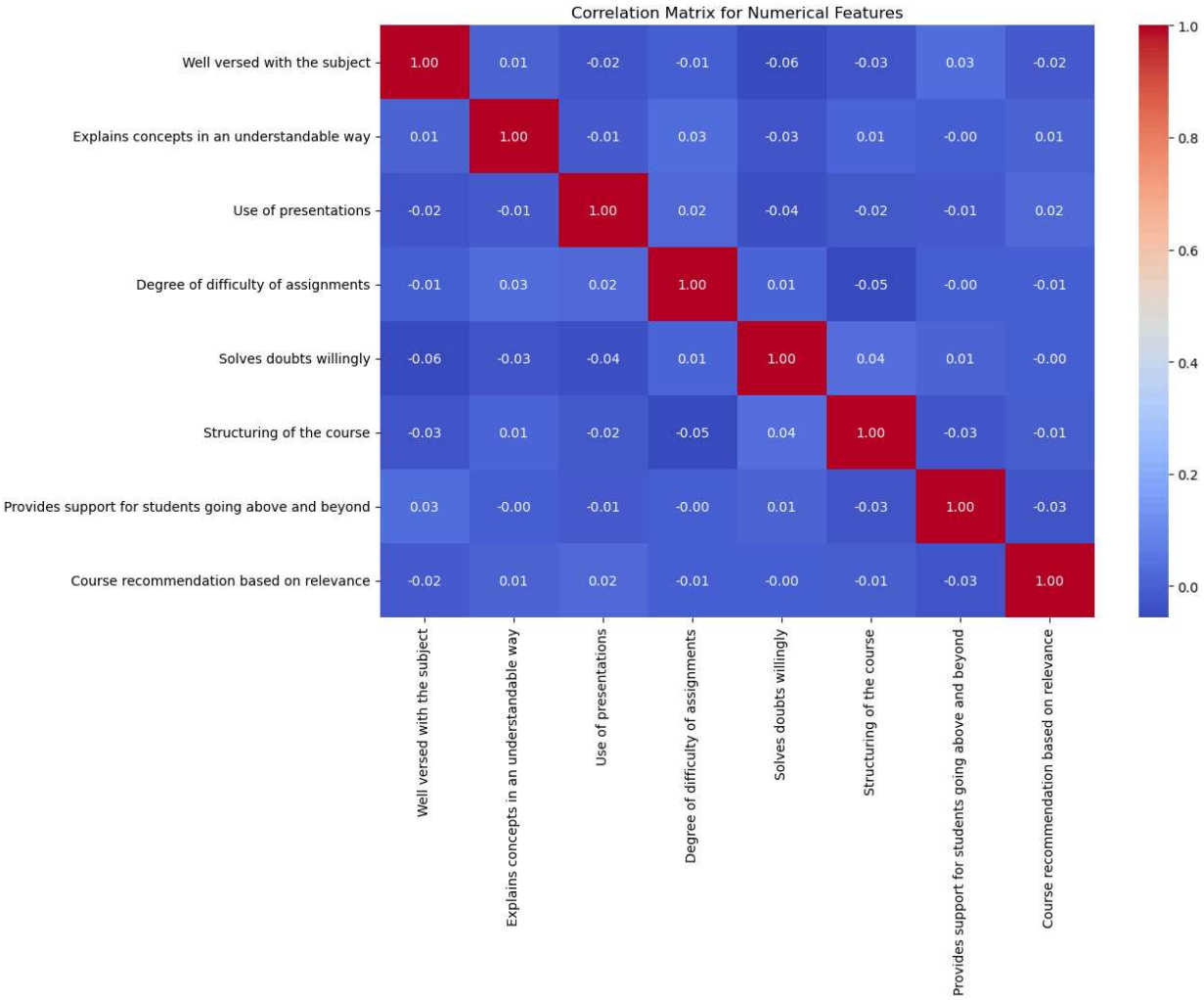
```
Dataset Shape: (1001, 8)
```

| | Well versed with the subject | Explains concepts in an understandable way | Use of presentations | Degree of difficulty of assignments | Solves doubts willingly | Structuring of the course | Provides support for students going above and beyond | Co recommenda base releva |
|---|---|---|---|---|---|---|---|---|
| **0** | 5 | 2 | 7 | 6 | 9 | 2 | 1 | |
| **1** | 6 | 5 | 8 | 6 | 2 | 1 | 2 | |
| **2** | 7 | 7 | 6 | 5 | 4 | 2 | 3 | |
| **3** | 9 | 6 | 7 | 1 | 5 | 9 | 4 | |
| **4** | 8 | 10 | 8 | 4 | 6 | 6 | 9 | |

```
Numeric Columns for Ratings: ['Well versed with the subject', 'Explains concepts in a
n understandable way', 'Use of presentations', 'Degree of difficulty of assignments',
'Solves doubts willingly', 'Structuring of the course', 'Provides support for student
s going above and beyond', 'Course recommendation based on relevance']
```

| | Well versed with the subject | Explains concepts in an understandable way | Use of presentations | Degree of difficulty of assignments | Solves doubts willingly | Structuring of the course | Provide support fo student goin above an beyon |
|---|---|---|---|---|---|---|---|
| **count** | 1001.000000 | 1001.000000 | 1001.000000 | 1001.000000 | 1001.000000 | 1001.000000 | 1001.00000 |
| **mean** | 7.497502 | 6.081918 | 5.942058 | 5.430569 | 5.474525 | 5.636364 | 5.66233 |
| **std** | 1.692998 | 2.597168 | 1.415853 | 2.869046 | 2.874648 | 2.920212 | 2.89169 |
| **min** | 5.000000 | 2.000000 | 4.000000 | 1.000000 | 1.000000 | 1.000000 | 1.00000 |
| **25%** | 6.000000 | 4.000000 | 5.000000 | 3.000000 | 3.000000 | 3.000000 | 3.00000 |
| **50%** | 8.000000 | 6.000000 | 6.000000 | 5.000000 | 6.000000 | 6.000000 | 6.00000 |
| **75%** | 9.000000 | 8.000000 | 7.000000 | 8.000000 | 8.000000 | 8.000000 | 8.00000 |
| **max** | 10.000000 | 10.000000 | 8.000000 | 10.000000 | 10.000000 | 10.000000 | 10.00000 |

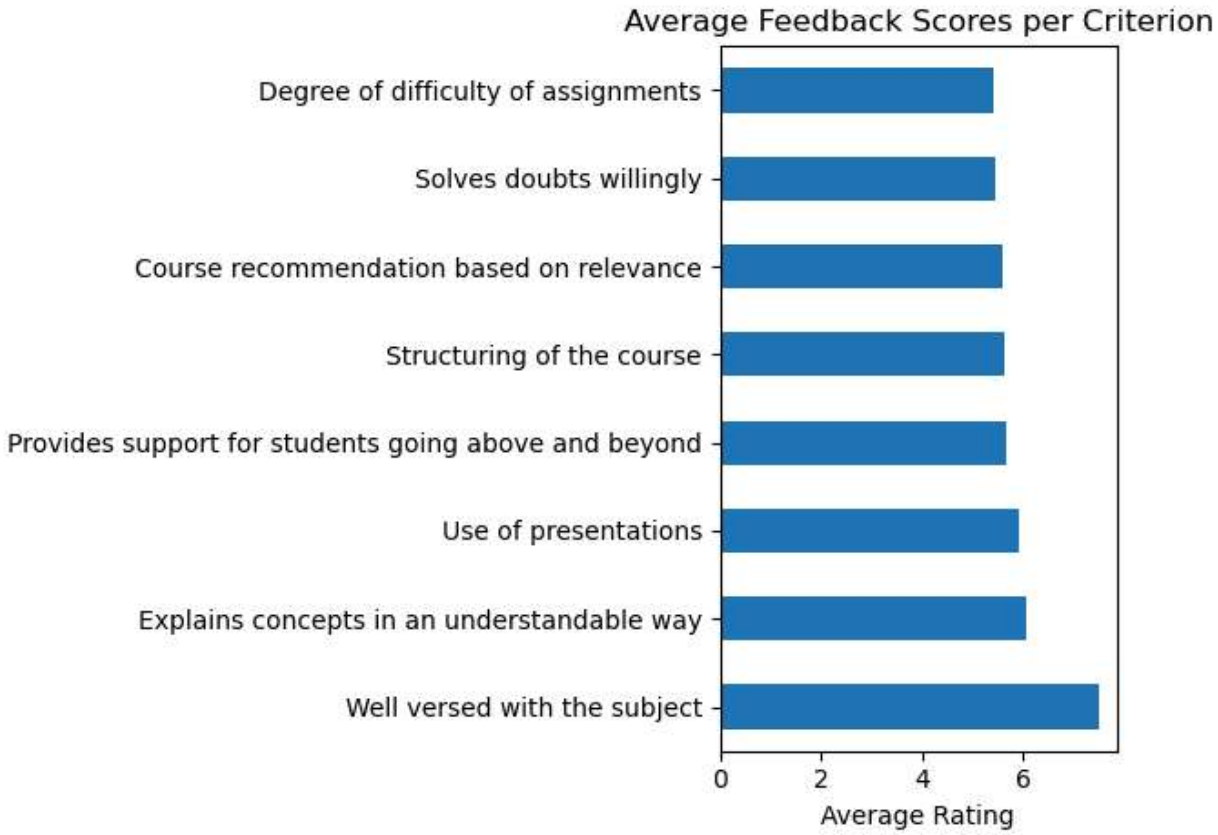
In [6]:
```python
# Correlation matrix for numerical features
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
numeric_df = df.select_dtypes(include=['int64', 'float64'])
corr_matrix = numeric_df.corr()
plt.figure(figsize=(12, 8))
sns.heatmap(corr_matrix, annot=True, fmt=".2f", cmap="coolwarm")
plt.title("Correlation Matrix for Numerical Features")
plt.show()
```




In [7]:
```
# Calculate average score per feedback criterion
avg_scores = df.mean().sort_values(ascending=False)

avg_scores
```

Out[7]:
```
Well versed with the subject                                7.497502
Explains concepts in an understandable way                  6.081918
Use of presentations                                        5.942058
Provides support for students going above and beyond        5.662338
Structuring of the course                                   5.636364
Course recommendation based on relevance                    5.598402
Solves doubts willingly                                     5.474525
Degree of difficulty of assignments                         5.430569
dtype: float64
```

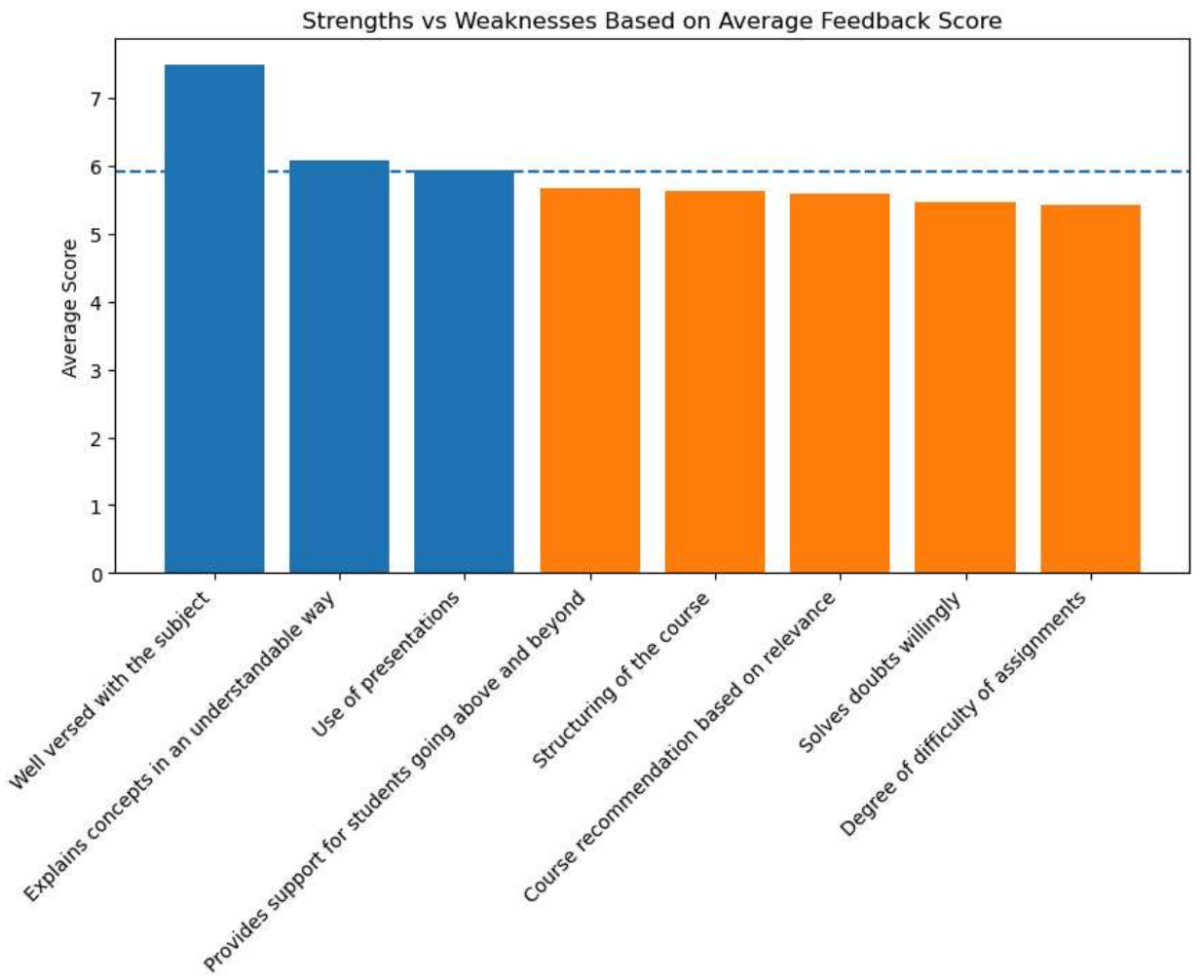
In [8]:
```
plt.figure()
avg_scores.plot(kind="barh")
plt.title("Average Feedback Scores per Criterion")
plt.xlabel("Average Rating")
plt.tight_layout()
plt.show()
```



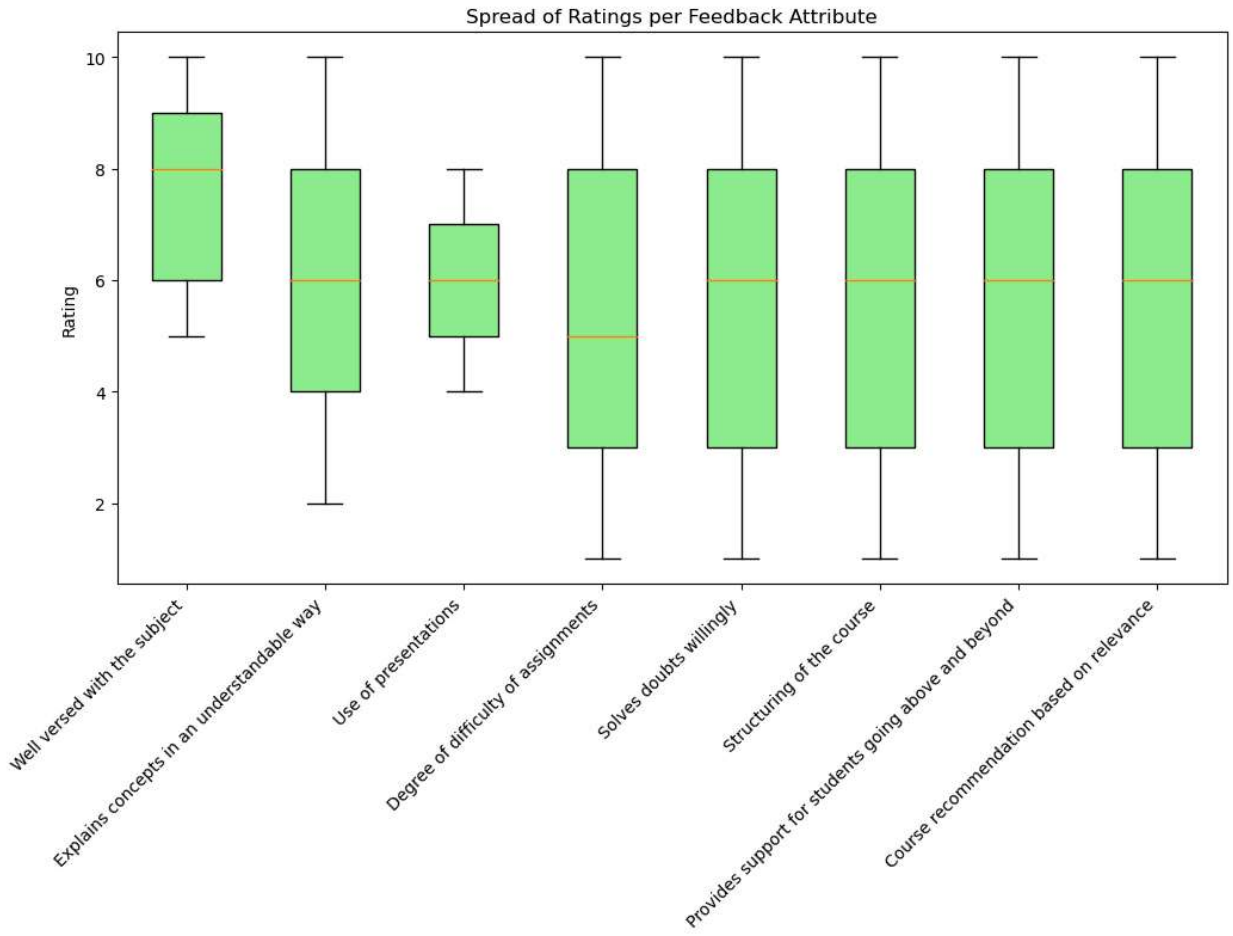

In [9]:
```
mean_score = avg_scores.mean()

strengths = avg_scores[avg_scores >= mean_score]
weaknesses = avg_scores[avg_scores < mean_score]

print("Strengths:\n", strengths)
print("\nNeeds Improvement:\n", weaknesses)
```

```
Strengths:
 Well versed with the subject                  7.497502
Explains concepts in an understandable way    6.081918
Use of presentations                          5.942058
dtype: float64

Needs Improvement:
 Provides support for students going above and beyond    5.662338
Structuring of the course                               5.636364
Course recommendation based on relevance                5.598402
Solves doubts willingly                                 5.474525
Degree of difficulty of assignments                     5.430569
dtype: float64
```

In [10]:
```
plt.figure(figsize=(10,5))

plt.bar(strengths.index, strengths.values)
plt.bar(weaknesses.index, weaknesses.values)

plt.axhline(mean_score, linestyle="--")
plt.ylabel("Average Score")
plt.title("Strengths vs Weaknesses Based on Average Feedback Score")
plt.xticks(rotation=45, ha="right")

plt.show()
```




In [11]:
```
# Create box plot
plt.figure(figsize=(12,6))
plt.boxplot(df.values, labels=df.columns, vert=True, patch_artist=True,
            boxprops=dict(facecolor='lightgreen'))
plt.xticks(rotation=45, ha="right")
plt.ylabel("Rating")
plt.title("Spread of Ratings per Feedback Attribute")
plt.show()
```




# Sentiment Analysis

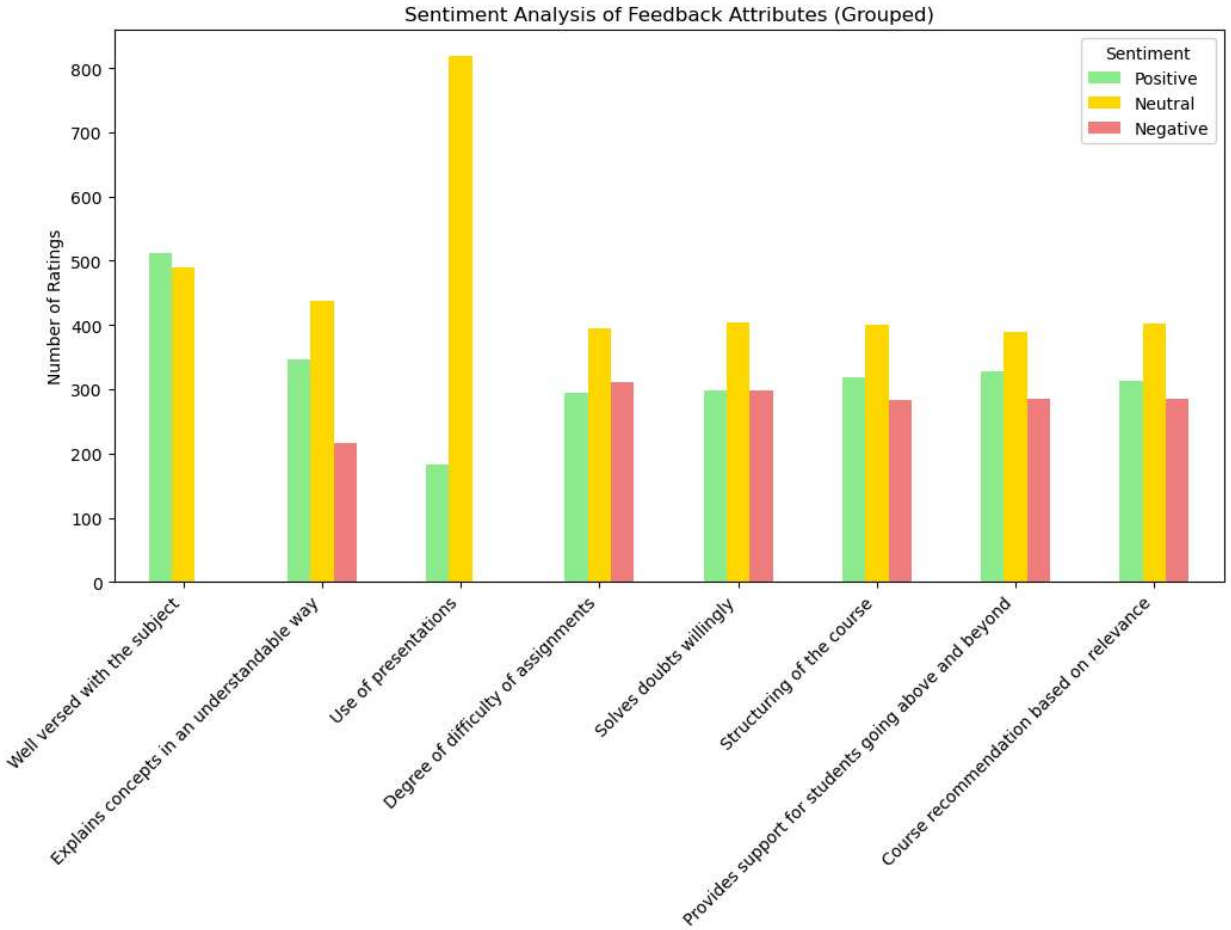
In [12]:
```
def rating_to_sentiment(x):
    if x <= 3:
        return 'Negative'
    elif x <= 7:
        return 'Neutral'
    else:
        return 'Positive'

sentiment_df = df.applymap(rating_to_sentiment)
print(sentiment_df)
```

```
     Well versed with the subject Explains concepts in an understandable way  \
0                        Neutral                                    Negative
1                        Neutral                                     Neutral
2                        Neutral                                     Neutral
3                       Positive                                     Neutral
4                       Positive                                    Positive
...                          ...                                         ...
996                     Positive                                     Neutral
997                      Neutral                                     Neutral
998                     Positive                                     Neutral
999                     Positive                                    Negative
1000                     Neutral                                    Negative

     Use of presentations Degree of difficulty of assignments  \
0                 Neutral                             Neutral
1                Positive                             Neutral
2                 Neutral                             Neutral
3                 Neutral                            Negative
4                Positive                             Neutral
...                   ...                                 ...
996               Neutral                            Negative
997               Neutral                             Neutral
998              Positive                            Negative
999               Neutral                             Neutral
1000              Neutral                            Negative

     Solves doubts willingly Structuring of the course  \
0                   Positive                   Negative
1                   Negative                   Negative
2                    Neutral                   Negative
3                    Neutral                   Positive
4                    Neutral                    Neutral
...                      ...                        ...
996                  Neutral                    Neutral
997                  Neutral                    Neutral
998                 Positive                   Negative
999                 Negative                    Neutral
1000                Negative                    Neutral

     Provides support for students going above and beyond  \
0                                           Negative
1                                           Negative
2                                           Negative
3                                            Neutral
4                                           Positive
...                                              ...
996                                          Neutral
997                                          Neutral
998                                         Negative
999                                         Positive
1000                                        Positive

     Course recommendation based on relevance
0                                   Positive
1                                   Positive
2                                   Negative
3                                    Neutral
4                                   Positive
...                                      ...
996                                 Positive
```

```
997                                   Negative
998                                   Negative
999                                   Negative
1000                                  Positive

[1001 rows x 8 columns]
```

In [13]:
```
# Count sentiments per attribute and ensure all sentiment categories exist
sentiment_counts = sentiment_df.apply(lambda x: x.value_counts()).fillna(0)
sentiment_counts = sentiment_counts.reindex(['Positive', 'Neutral', 'Negative']).T  #
```

In [14]:
```
# Grouped bar chart
sentiment_counts.plot(kind='bar', figsize=(12,6), color=['lightgreen','gold','lightcor
plt.xticks(rotation=45, ha='right')
plt.ylabel('Number of Ratings')
plt.title('Sentiment Analysis of Feedback Attributes (Grouped)')
plt.legend(title='Sentiment')
plt.show()
```




# Conclusion

The overall findings indicate a generally positive academic environment with clear strengths and identifiable areas for improvement. Faculty members are perceived to be highly knowledgeable in their respective subjects, as reflected by the highest average score in the survey. Additionally, the ability of instructors to explain concepts clearly and make effective use of presentations

stands out as a major strength, suggesting that teaching methodologies are largely effective and well-received by students.

However, the analysis also highlights areas that require attention. Attributes such as assignment difficulty, course structuring, doubt resolution, and extended student support received comparatively lower average scores. While these ratings are not poor, they indicate opportunities to enhance student engagement through better course planning, more balanced assignment workloads, and increased academic support mechanisms.

The sentiment analysis further reinforces these findings, showing a predominance of neutral to positive feedback across most attributes, with relatively fewer negative responses. This suggests that while students are generally satisfied, targeted improvements could significantly elevate their overall learning experience.

In [ ]: