In [2]:
```
import pandas as pd
```

In [3]:
```
df = pd.read_csv(r"C:\Users\asmit\Downloads\student_feedback.csv")
```

In [4]:
```
# top 5 rows
df.head()
```

Out[4]:

	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	Stu
0	0	340	5	2	7	6	9	
1	1	253	6	5	8	6	2	
2	2	680	7	7	6	5	4	
3	3	806	9	6	7	1	5	
4	4	632	8	10	8	4	6	

In [5]:
```
#bottom 5 rows
df.tail()
```

Out[5]:

	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
996	996	55	8	7	6	2	5
997	997	913	5	5	6	5	6
998	998	199	9	5	8	3	8
999	999	539	10	2	7	4	3
1000	1000	759	7	2	4	2	1

In [6]:
```
#total no of rows and columns
df.shape
```

Out[6]: (1001, 10)

In [7]:
```
#info about the Data types and the non null Count
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1001 entries, 0 to 1000
Data columns (total 10 columns):
 #   Column                                                Non-Null Count  Dtype
---  ------                                                --------------  -----
 0   Unnamed: 0                                            1001 non-null   int64
 1   Student ID                                            1001 non-null   int64
 2   Well versed with the subject                          1001 non-null   int64
 3   Explains concepts in an understandable way            1001 non-null   int64
 4   Use of presentations                                  1001 non-null   int64
 5   Degree of difficulty of assignments                   1001 non-null   int64
 6   Solves doubts willingly                               1001 non-null   int64
 7   Structuring of the course                             1001 non-null   int64
 8   Provides support for students going above and beyond  1001 non-null   int64
 9   Course recommendation based on relevance              1001 non-null   int64
dtypes: int64(10)
memory usage: 78.3 KB
```

In [8]:
```
#checking nulls
df.isnull().sum()
```

Out[8]:
```
Unnamed: 0                                              0
Student ID                                              0
Well versed with the subject                            0
Explains concepts in an understandable way              0
Use of presentations                                    0
Degree of difficulty of assignments                     0
Solves doubts willingly                                 0
Structuring of the course                               0
Provides support for students going above and beyond    0
Course recommendation based on relevance                0
dtype: int64
```

In [9]:
```
# showing total count of rows, mean values, STD, MIN, MAX
df.describe()
```

Out[9]:

	Unnamed: 0	Student ID	Well versed with the subject	Explains concepts in an understandable way	Use of presentations	Degree of difficulty of assignments
count	1001.000000	1001.000000	1001.000000	1001.000000	1001.000000	1001.000000
mean	500.000000	500.000000	7.497502	6.081918	5.942058	5.430569
std	289.108111	289.108111	1.692998	2.597168	1.415853	2.869046
min	0.000000	0.000000	5.000000	2.000000	4.000000	1.000000
25%	250.000000	250.000000	6.000000	4.000000	5.000000	3.000000
50%	500.000000	500.000000	8.000000	6.000000	6.000000	5.000000
75%	750.000000	750.000000	9.000000	8.000000	7.000000	8.000000
max	1000.000000	1000.000000	10.000000	10.000000	8.000000	10.000000

In [10]:
```
# checking duplicates
df.duplicated().sum()
```

Out[10]:
```
0
```

In [11]:
```
#unique values in different columns
df.nunique()
```

Out[11]:
```
Unnamed: 0                                            1001
Student ID                                            1001
Well versed with the subject                             6
Explains concepts in an understandable way               9
Use of presentations                                     5
Degree of difficulty of assignments                     10
Solves doubts willingly                                 10
Structuring of the course                               10
Provides support for students going above and beyond    10
Course recommendation based on relevance                10
dtype: int64
```

In [12]:
```
df.drop(columns = {"Unnamed: 0"}, inplace = True)
```

In [13]:
```
df.rename(columns={"Well versed with the subject": "Satisfaction Score"}, inplace =
```

In [14]:
```
df.rename(columns={"Explains concepts in an understandable way": "Concepts Explaina
df.rename(columns={"Use of presentations": "Presentation Use"}, inplace = True)
df.rename(columns={"Degree of difficulty of assignments": "Assignment Difficulty"},
df.rename(columns={"Solves doubts willingly": "Doubts Solving"}, inplace = True)
df.rename(columns={"Structuring of the course": "Course Structuring"}, inplace = Tr
df.rename(columns={"Provides support for students going above and beyond": "Student
df.rename(columns={"Course recommendation based on relevance": "Course Recommendati
```

In [32]:
```
df.head()
```

Out[32]:

	Student ID	Satisfaction Score	Concepts Explaination	Presentation Use	Assignment Difficulty	Doubts Solving	Course Structuring	St Sup
0	340	5	2	7	6	9	2	
1	253	6	5	8	6	2	1	
2	680	7	7	6	5	4	2	
3	806	9	6	7	1	5	9	
4	632	8	10	8	4	6	6	

In [16]:
```
def rate_to_sentiment(score):
    if score <= 3:
        return "Negative"
    elif score <= 6:
        return "Neutral"
    else:
        return "Positive"

rating_columns = ["Satisfaction Score", "Concepts Explaination", "Doubts Solving",
for col in rating_columns:
    df[f"{col}_Sentiment"] = df[col].apply(rate_to_sentiment)
```

In [17]:
```
df.head()
```

Out[17]:

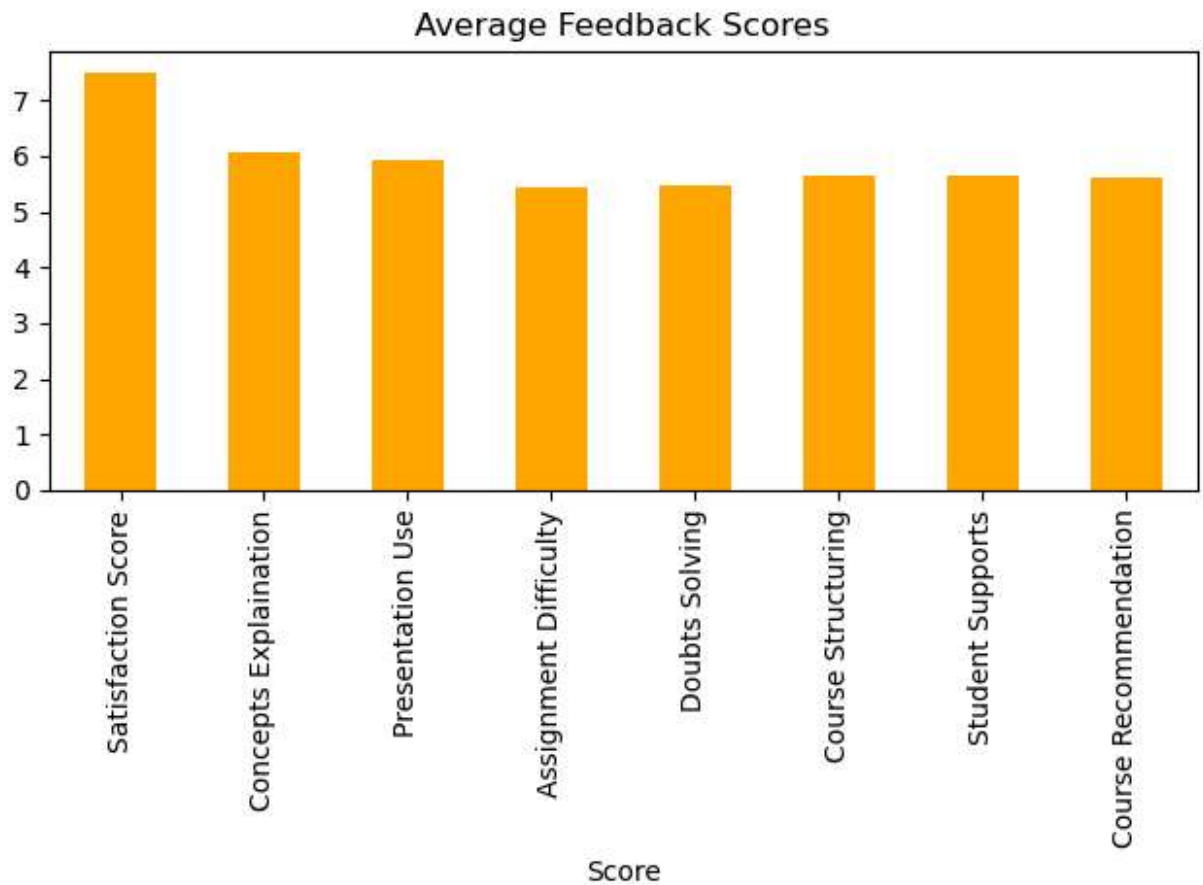
	Student ID	Satisfaction Score	Concepts Explaination	Presentation Use	Assignment Difficulty	Doubts Solving	Course Structuring	St Sup
0	340	5	2	7	6	9	2	
1	253	6	5	8	6	2	1	
2	680	7	7	6	5	4	2	
3	806	9	6	7	1	5	9	
4	632	8	10	8	4	6	6	

In [18]:
```
import seaborn as sns
import matplotlib.pyplot as plt
```

In [19]:
```
cols = [
    "Satisfaction Score",
    "Concepts Explaination",
    "Presentation Use",
    "Assignment Difficulty",
    "Doubts Solving",
    "Course Structuring",
    "Student Supports",
```

```
    "Course Recommendation"
]

avg_scores = df[cols].mean()
```

In [20]:
```
avg_scores.plot(kind='bar', color='orange')
plt.title("Average Feedback Scores")
plt.xlabel("Score")
plt.tight_layout()
plt.show()
```

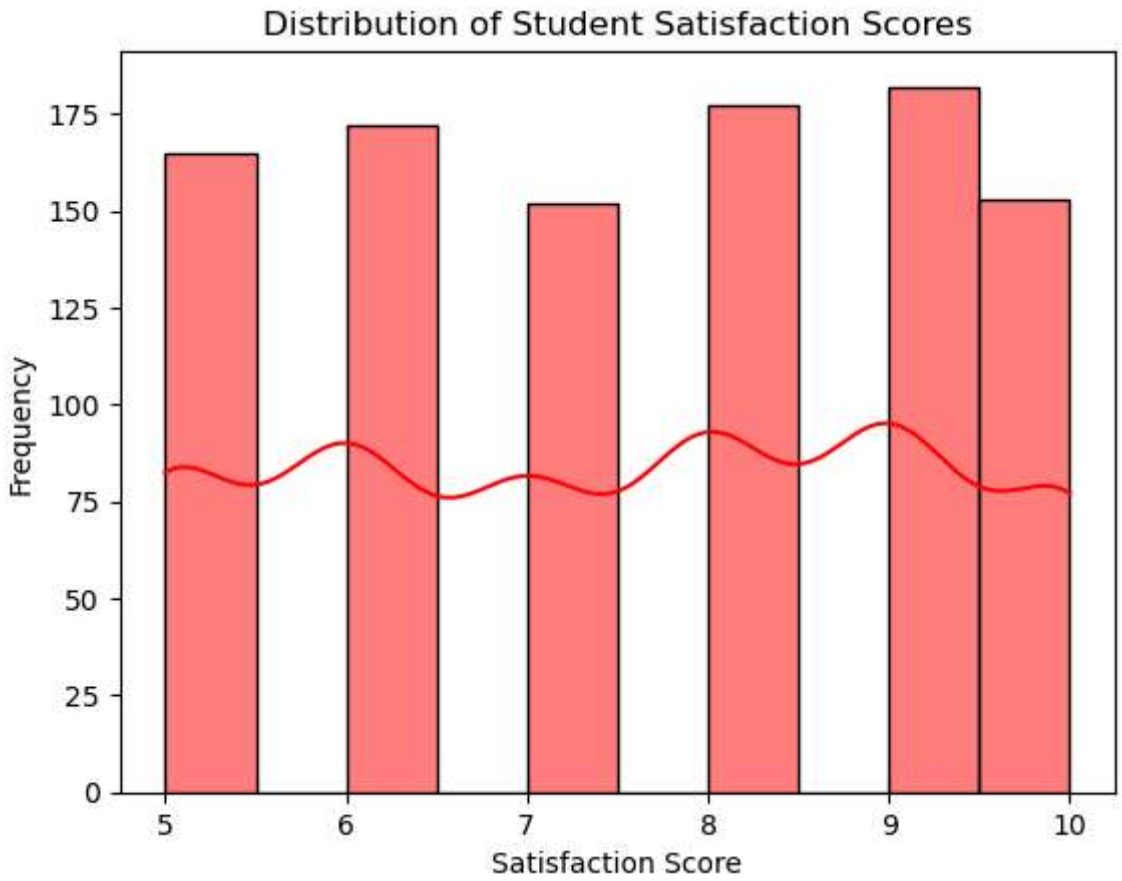


In [26]:
```
import plotly.graph_objects as go

scores = [8.5, 9.0, 7.5, 6.0, 8.0, 7.0, 8.8, 9.2]

fig = go.Figure()

fig.add_trace(go.Scatterpolar(
    r=scores,
    theta=cols,
    mode='lines+markers',
    fill='toself',
    name='Feedback Score',
    hoverinfo='text',
    text=[f"{label}: {score}" for label, score in zip(cols, scores)]
))

fig.update_layout(
```

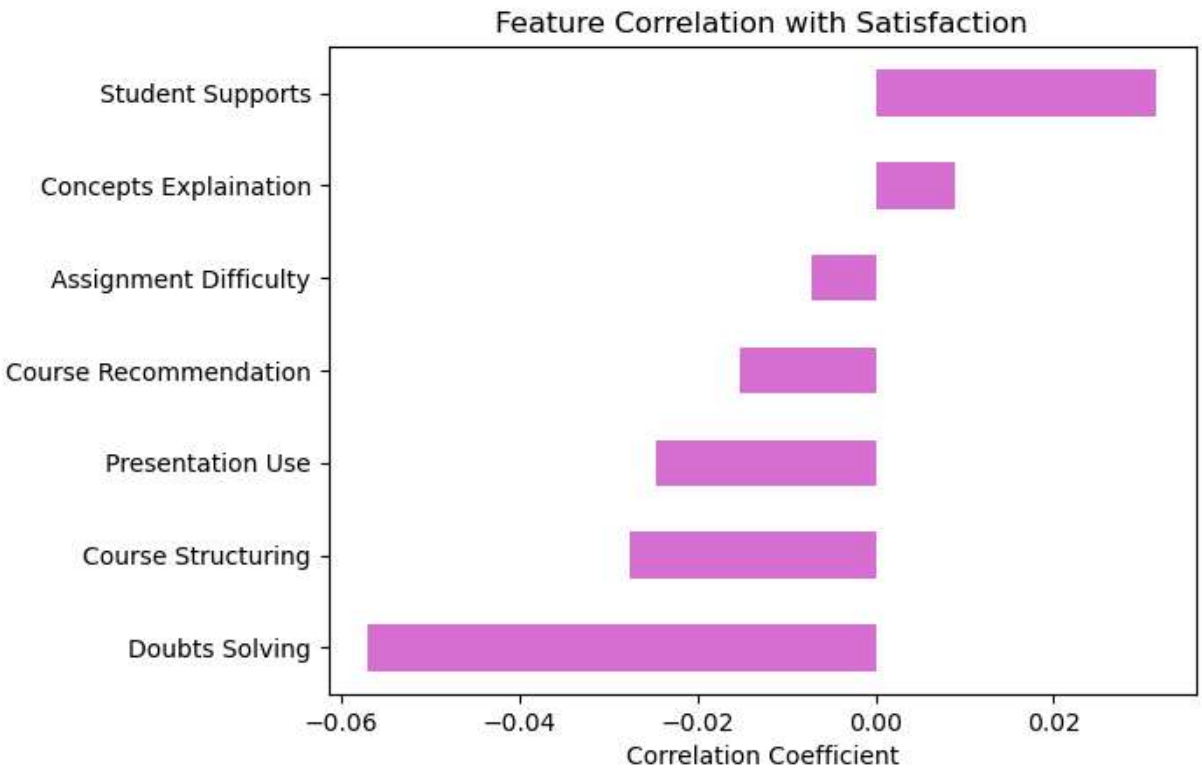
```
    polar=dict(
        radialaxis=dict(visible=True, range=[0, 10])
    ),
    showlegend=False,
    title='Course Feedback Radar Chart'
)

fig.show()
```

In [37]:
```
sns.histplot(df['Satisfaction Score'], bins=10, kde=True, color='red')
plt.title('Distribution of Student Satisfaction Scores')
plt.xlabel('Satisfaction Score')
plt.ylabel('Frequency')
plt.show()
```

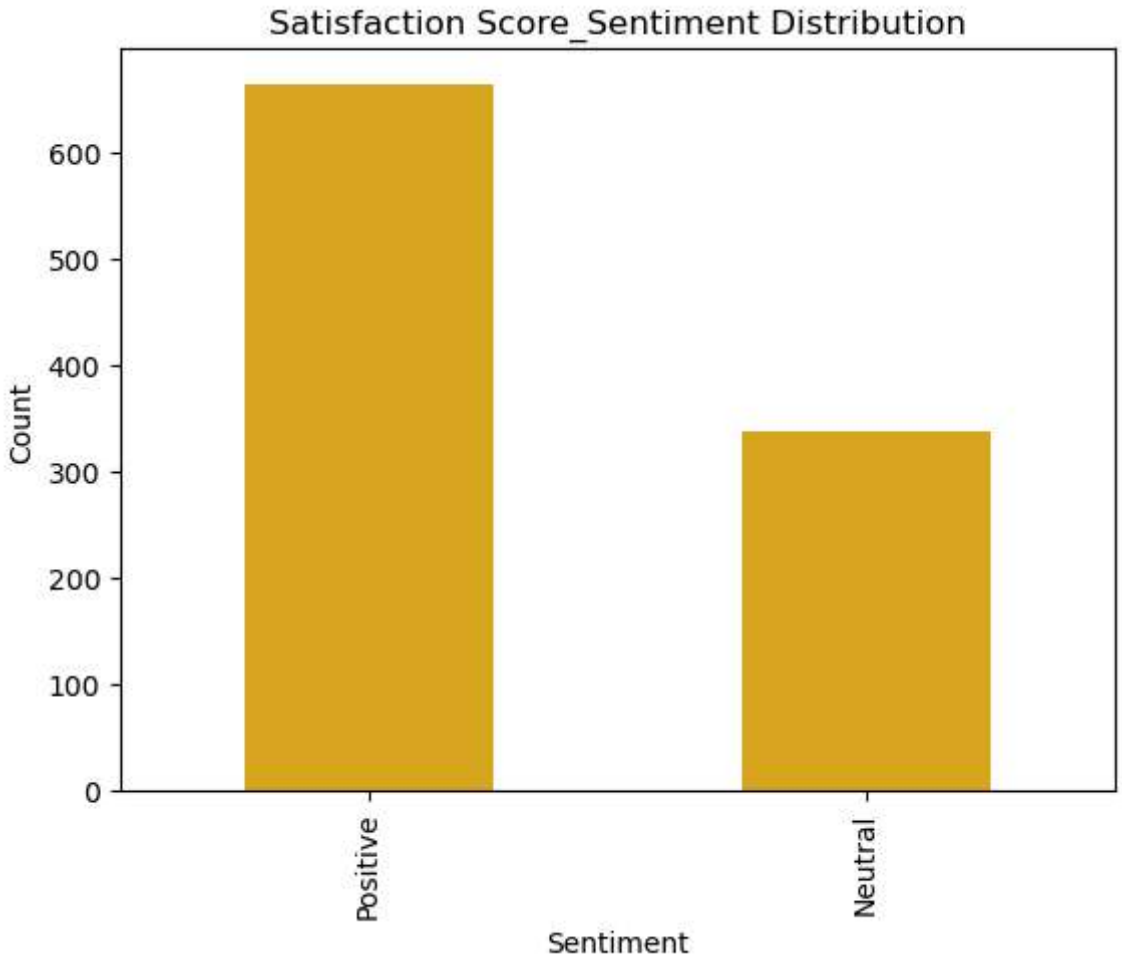
```
C:\Users\asmit\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version. Convert
inf values to NaN before operating instead.
```

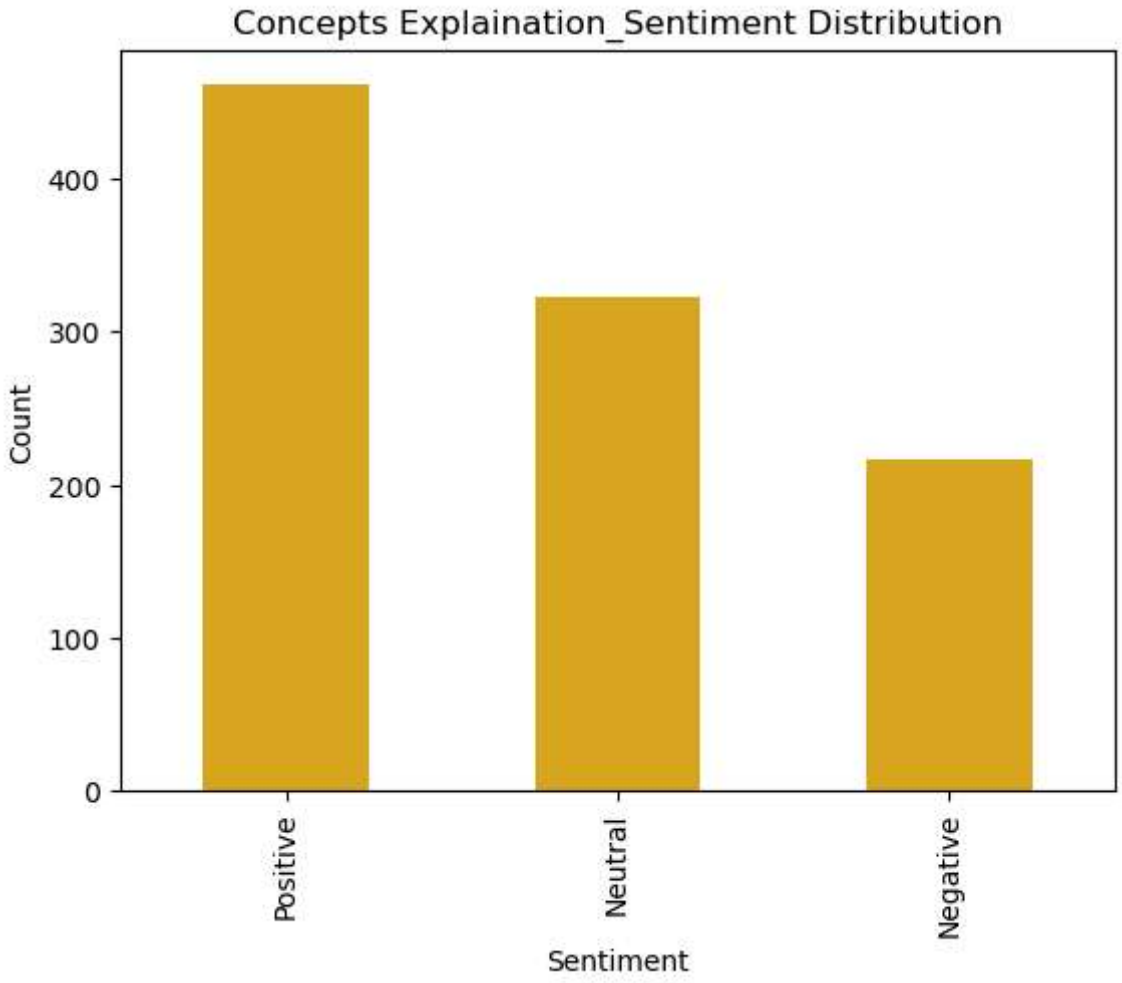


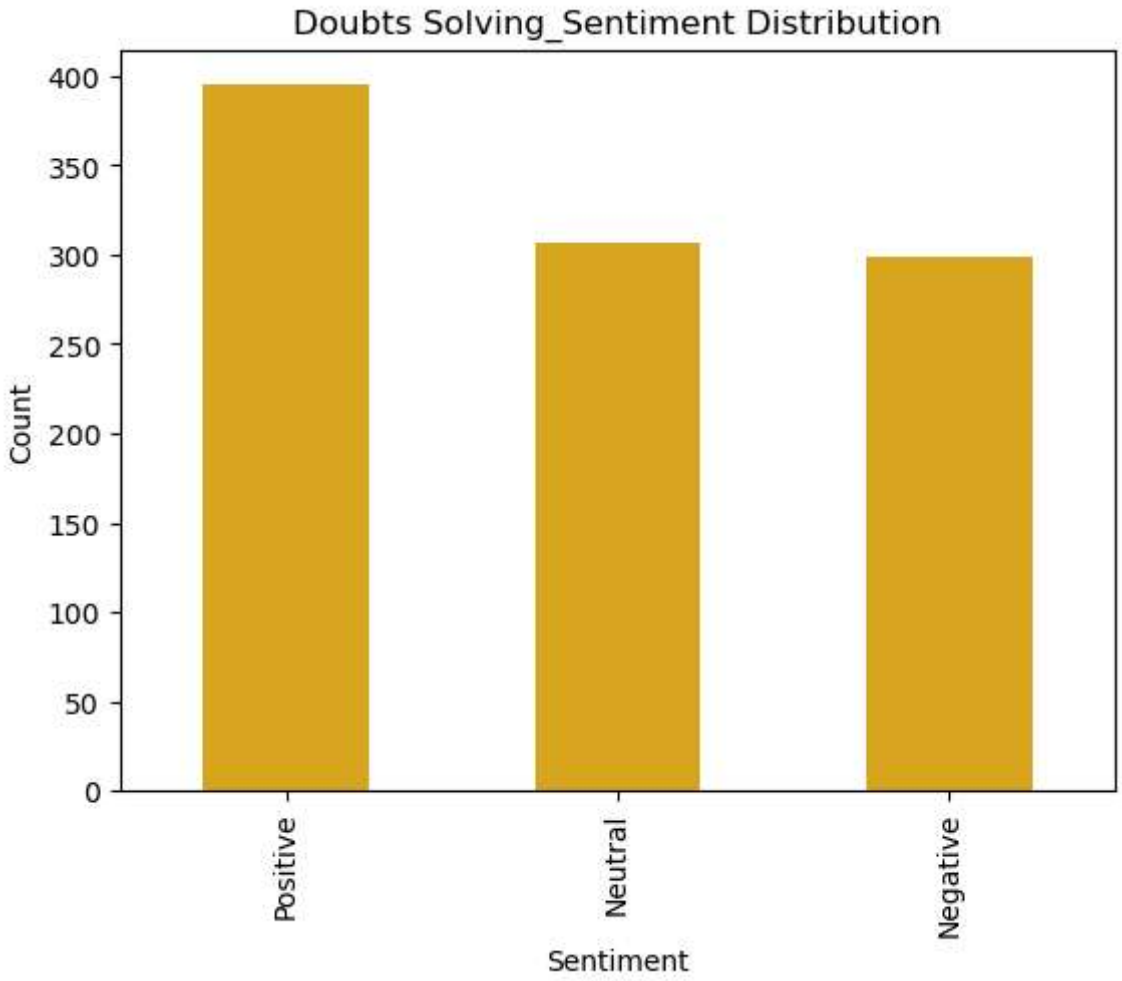
In [39]:
```
features = ['Concepts Explaination', 'Presentation Use', 'Assignment Difficulty',
            'Doubts Solving', 'Course Structuring', 'Student Supports', 'Course Rec

correlations = df[features + ['Satisfaction Score']].corr()['Satisfaction Score'][f
correlations.sort_values().plot(kind='barh', color='orchid')
plt.title('Feature Correlation with Satisfaction')
plt.xlabel('Correlation Coefficient')
plt.show()
```

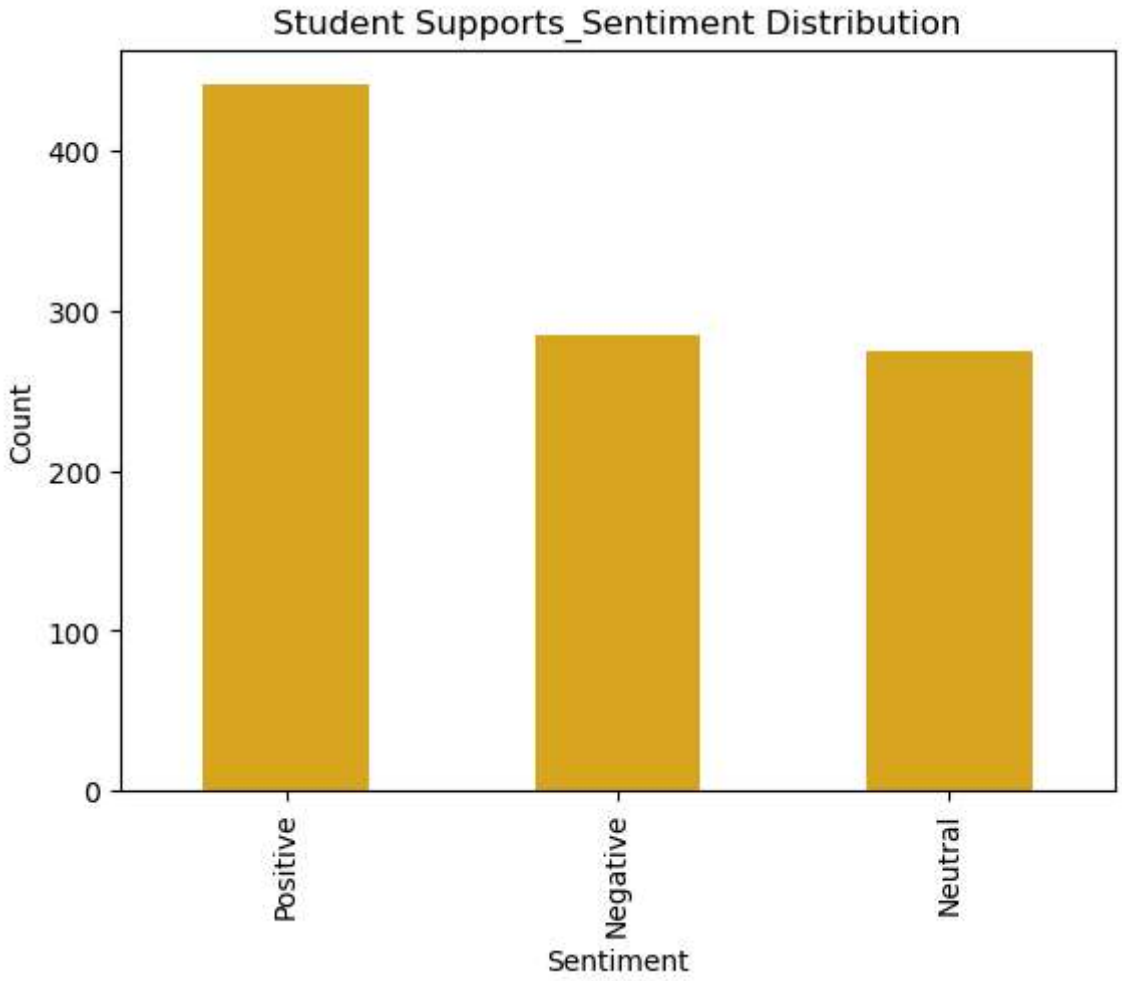


In [40]:
```
sentiment_cols = ['Satisfaction Score_Sentiment', 'Concepts Explaination_Sentiment'
                  'Doubts Solving_Sentiment', 'Student Supports_Sentiment']

for col in sentiment_cols:
    sentiment_counts = df[col].value_counts()
    sentiment_counts.plot(kind='bar', color='goldenrod')
    plt.title(f'{col} Distribution')
    plt.xlabel('Sentiment')
    plt.ylabel('Count')
    plt.show()
```


Satisfaction Score_Sentiment Distribution
Count
600
500
400
300
200
100
0
Positive
Neutral
Sentiment


Concepts Explaination_Sentiment Distribution
Count
0
100
200
300
400
Positive
Neutral
Negative
Sentiment


Doubts Solving_Sentiment Distribution
400
350
300
250
200
150
100
50
0
Count
Positive
Neutral
Negative
Sentiment


Student Supports_Sentiment Distribution
Count
400
300
200
100
0
Positive
Negative
Neutral
Sentiment

In [ ]:

In [ ]: