

```
from google.colab import files
uploaded = files.upload()
```

[Choose Files](#) student_feedback.csv
student_feedback.csv(text/csv) - 24877 bytes, last modified: 12/2/2025 - 100% done
Saving student_feedback.csv to student_feedback (2).csv

```
import pandas as pd
df = pd.read_csv("student_feedback.csv")
df.head()
```

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	Provides support for students going above and beyond	Course recommendation based on relevance	
0	0	340	5	2	7	6	9	2	1	8
1	1	253	6	5	8	6	2	1	2	9

Next steps: [Generate code with df](#) [New interactive sheet](#)

```
df.columns
```

```
Index(['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',
      'Provides support for students going above and beyond',
      'Course recommendation based on relevance'],
      dtype='object')
```

```
df.info()
df.head()
```

```
<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
```

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	Provides support for students going above and beyond	Course recommendation based on relevance	
0	0	340	5	2	7	6	9	2	1	8
1	1	253	6	5	8	6	2	1	2	9

Next steps: [Generate code with df](#) [New interactive sheet](#)

```
# Remove the unwanted index column
df = df.drop(columns=['Unnamed: 0'])

# Rename columns to simpler names
df.columns = ['
```

```

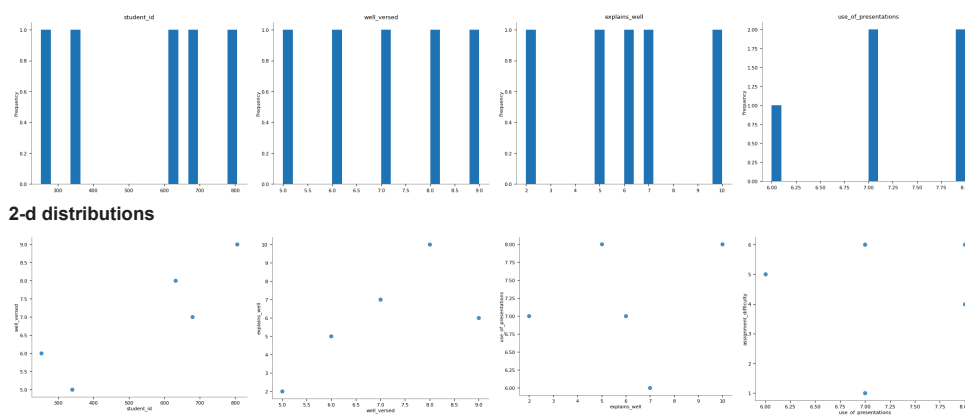
'student_id',
'well_versed',
'explains_well',
'use_of_presentations',
'assignment_difficulty',
'solves_doubts',
'course_structure',
'provides_support',
'course_recommendation'
]

```

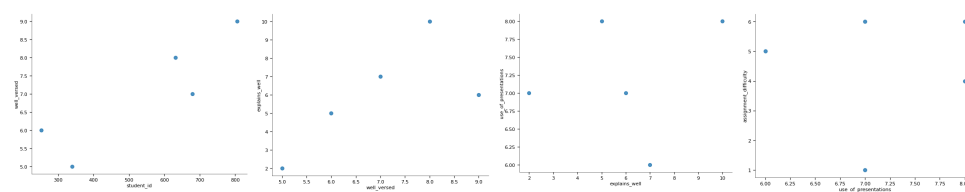
```
df.head()
```

	student_id	well_versed	explains_well	use_of_presentations	assignment_difficulty	solves_doubts	course_structure	pro
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	

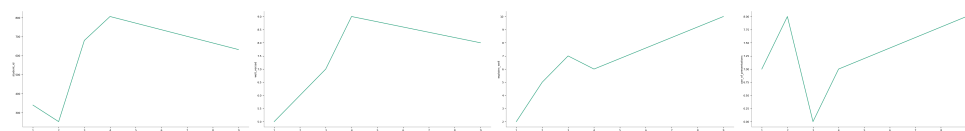
Distributions



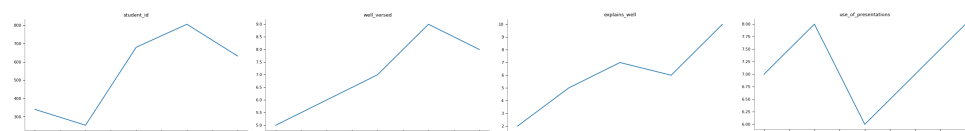
2-d distributions



Time series



Values



Next steps: [Generate code with df](#) [New interactive sheet](#)

```

import seaborn as sns
import matplotlib.pyplot as plt

print(df.describe())

avg_ratings = df.mean().drop('student_id')
avg_ratings.plot(kind='bar', color='skyblue')
plt.title("Average Ratings per Feedback Question")
plt.ylabel("Average Rating (1-10)")
plt.xticks(rotation=45, ha='right')
plt.show()

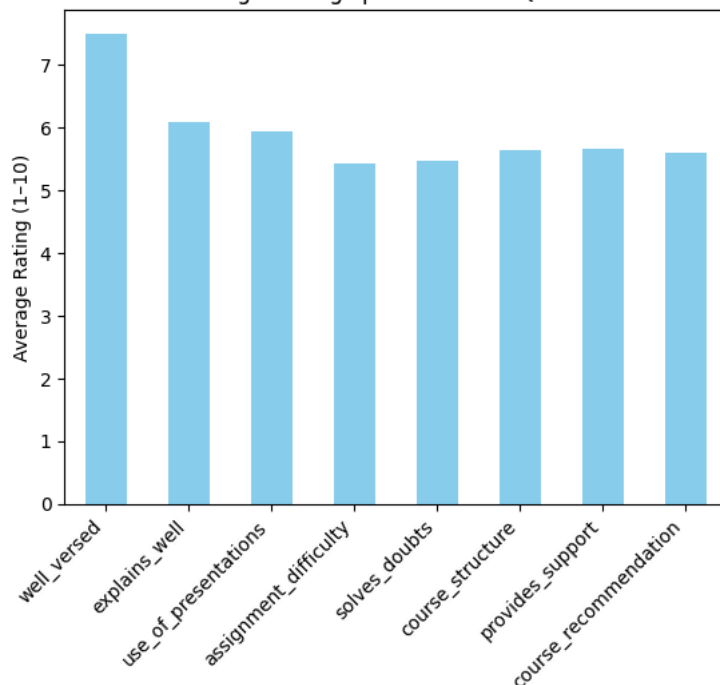
```

	student_id	well_versed	explains_well	use_of_presentations	\
count	1001.000000	1001.000000	1001.000000	1001.000000	
mean	500.000000	7.497502	6.081918	5.942058	
std	289.108111	1.692998	2.597168	1.415853	
min	0.000000	5.000000	2.000000	4.000000	
25%	250.000000	6.000000	4.000000	5.000000	
50%	500.000000	8.000000	6.000000	6.000000	
75%	750.000000	9.000000	8.000000	7.000000	
max	1000.000000	10.000000	10.000000	8.000000	

	assignment_difficulty	solves_doubts	course_structure	\
count	1001.000000	1001.000000	1001.000000	
mean	5.430569	5.474525	5.636364	
std	2.869046	2.874648	2.920212	
min	1.000000	1.000000	1.000000	
25%	3.000000	3.000000	3.000000	
50%	5.000000	6.000000	6.000000	
75%	8.000000	8.000000	8.000000	
max	10.000000	10.000000	10.000000	

	provides_support	course_recommendation
count	1001.000000	1001.000000
mean	5.662338	5.598402
std	2.891690	2.886617
min	1.000000	1.000000
25%	3.000000	3.000000
50%	6.000000	6.000000
75%	8.000000	8.000000
max	10.000000	10.000000

Average Ratings per Feedback Question



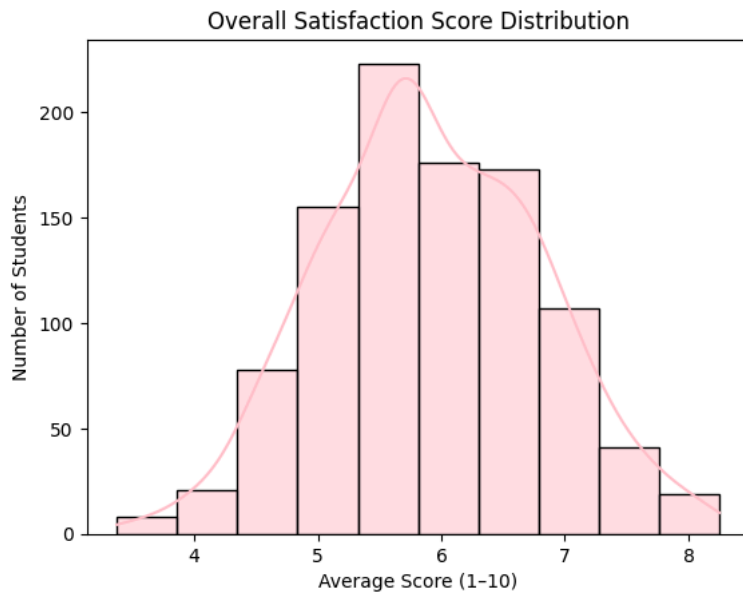
```
rating_columns = [
    'well_versed', 'explains_well', 'use_of_presentations',
    'assignment_difficulty', 'solves_doubts', 'course_structure',
    'provides_support', 'course_recommendation'
]
```

```
df['overall_score'] = df[rating_columns].mean(axis=1)
df['overall_score'].head()
```

	overall_score
0	5.000
1	4.875
2	4.375
3	5.875
4	7.500

dtype: float64

```
sns.histplot(df['overall_score'], bins=10, kde=True, color='pink')
plt.title("Overall Satisfaction Score Distribution")
plt.xlabel("Average Score (1-10)")
plt.ylabel("Number of Students")
plt.show()
```



```
print("Top-rated aspects:")
print(avg_ratings.sort_values(ascending=False).head(3))

print("\nAspects needing improvement:")
print(avg_ratings.sort_values().head(3))
```

```
Top-rated aspects:
well_versed          7.497502
explains_well       6.081918
use_of_presentations 5.942058
dtype: float64
```

```
Aspects needing improvement:
assignment_difficulty 5.430569
solves_doubts         5.474525
course_recommendation 5.598402
dtype: float64
```

```
feedback_list = [
    "Very good teaching style",
    "Concepts were confusing at times",
    "Loved the use of examples",
    "Assignments were too hard",
    "Excellent and interactive sessions",
    "Good, but more practical examples needed",
    "The course was engaging and informative",
    "Slides were boring and text-heavy",
    "Teacher was very supportive and helpful",
    "Overall an average experience"
] * 101 # make sure it's more than 1001

df['feedback_comment'] = feedback_list[:len(df)]
```

```
!pip install textblob
from textblob import TextBlob
```

```
Requirement already satisfied: textblob in /usr/local/lib/python3.12/dist-packages (0.19.0)
Requirement already satisfied: nltk>=3.9 in /usr/local/lib/python3.12/dist-packages (from textblob) (3.9.1)
Requirement already satisfied: click in /usr/local/lib/python3.12/dist-packages (from nltk>=3.9->textblob) (8.3.1)
Requirement already satisfied: joblib in /usr/local/lib/python3.12/dist-packages (from nltk>=3.9->textblob) (1.5.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.12/dist-packages (from nltk>=3.9->textblob) (2025.11.3)
Requirement already satisfied: tqdm in /usr/local/lib/python3.12/dist-packages (from nltk>=3.9->textblob) (4.67.1)
```

```
def get_sentiment(text):
    polarity = TextBlob(text).sentiment.polarity
    if polarity > 0:
```

```

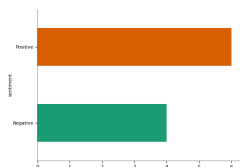
        return 'Positive'
    elif polarity == 0:
        return 'Neutral'
    else:
        return 'Negative'

df['sentiment'] = df['feedback_comment'].apply(get_sentiment)
df[['feedback_comment', 'sentiment']].head(10)

```

	feedback_comment	sentiment
0	Very good teaching style	Positive
1	Concepts were confusing at times	Negative
2	Loved the use of examples	Positive
3	Assignments were too hard	Negative
4	Excellent and interactive sessions	Positive
5	Good, but more practical examples needed	Positive
6	The course was engaging and informative	Positive
7	Slides were boring and text-heavy	Negative
8	Teacher was very supportive and helpful	Positive
9	Overall an average experience	Negative

Categorical distributions



```

import seaborn as sns
import matplotlib.pyplot as plt

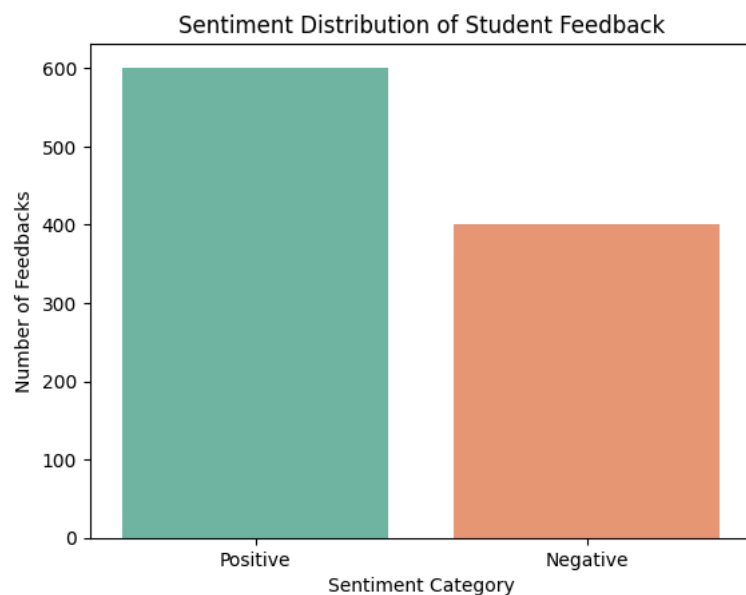
sns.countplot(x='sentiment', data=df, palette='Set2')
plt.title("Sentiment Distribution of Student Feedback")
plt.xlabel("Sentiment Category")
plt.ylabel("Number of Feedbacks")
plt.show()

```

/tmp/ipython-input-1176212608.py:4: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and

```
sns.countplot(x='sentiment', data=df, palette='Set2')
```



```

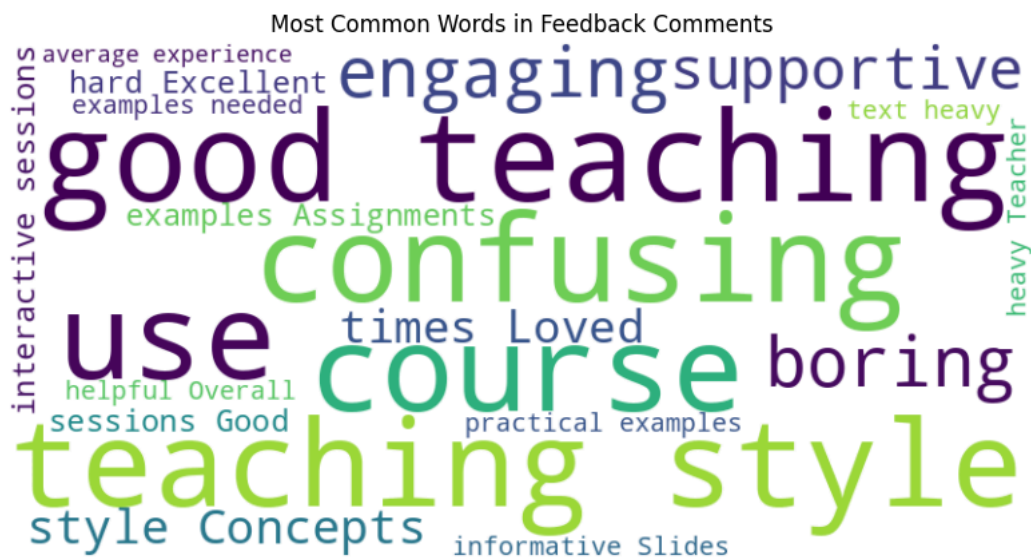
!pip install wordcloud
from wordcloud import WordCloud

```

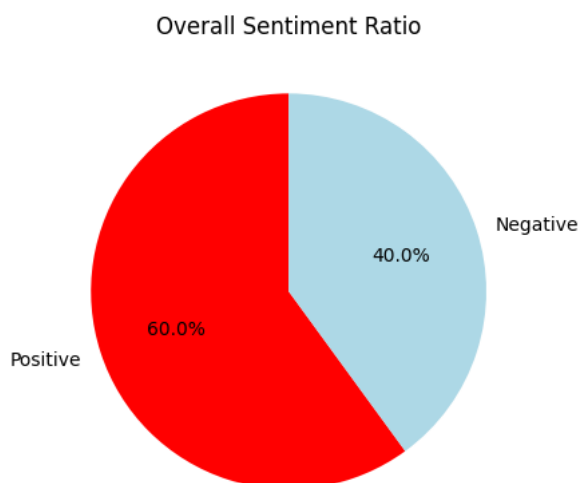
```
all_text = " ".join(df['feedback_comment'])
wordcloud = WordCloud(width=800, height=400, background_color='white').generate(all_text)

plt.figure(figsize=(10,5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title("Most Common Words in Feedback Comments")
plt.show()
```

Requirement already satisfied: wordcloud in /usr/local/lib/python3.12/dist-packages (1.9.4)
 Requirement already satisfied: numpy>=1.6.1 in /usr/local/lib/python3.12/dist-packages (from wordcloud) (2.0.2)
 Requirement already satisfied: pillow in /usr/local/lib/python3.12/dist-packages (from wordcloud) (11.3.0)
 Requirement already satisfied: matplotlib in /usr/local/lib/python3.12/dist-packages (from wordcloud) (3.10.0)
 Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib->wordcloud) (1.3.0)
 Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.12/dist-packages (from matplotlib->wordcloud) (0.12.1)
 Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.12/dist-packages (from matplotlib->wordcloud) (4.53.0)
 Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib->wordcloud) (1.4.7)
 Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packages (from matplotlib->wordcloud) (25.0)
 Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.12/dist-packages (from matplotlib->wordcloud) (3.2.0)
 Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.12/dist-packages (from matplotlib->wordcloud) (2.9.0)
 Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.12/dist-packages (from python-dateutil->matplotlib->wordcloud) (1.17.0)



```
# Pie chart for sentiment ratio
df['sentiment'].value_counts().plot(
    kind='pie', autopct='%1.1f%%', startangle=90, colors=['red', 'lightblue', 'lightcoral'])
plt.title("Overall Sentiment Ratio")
plt.ylabel("")
plt.show()
```



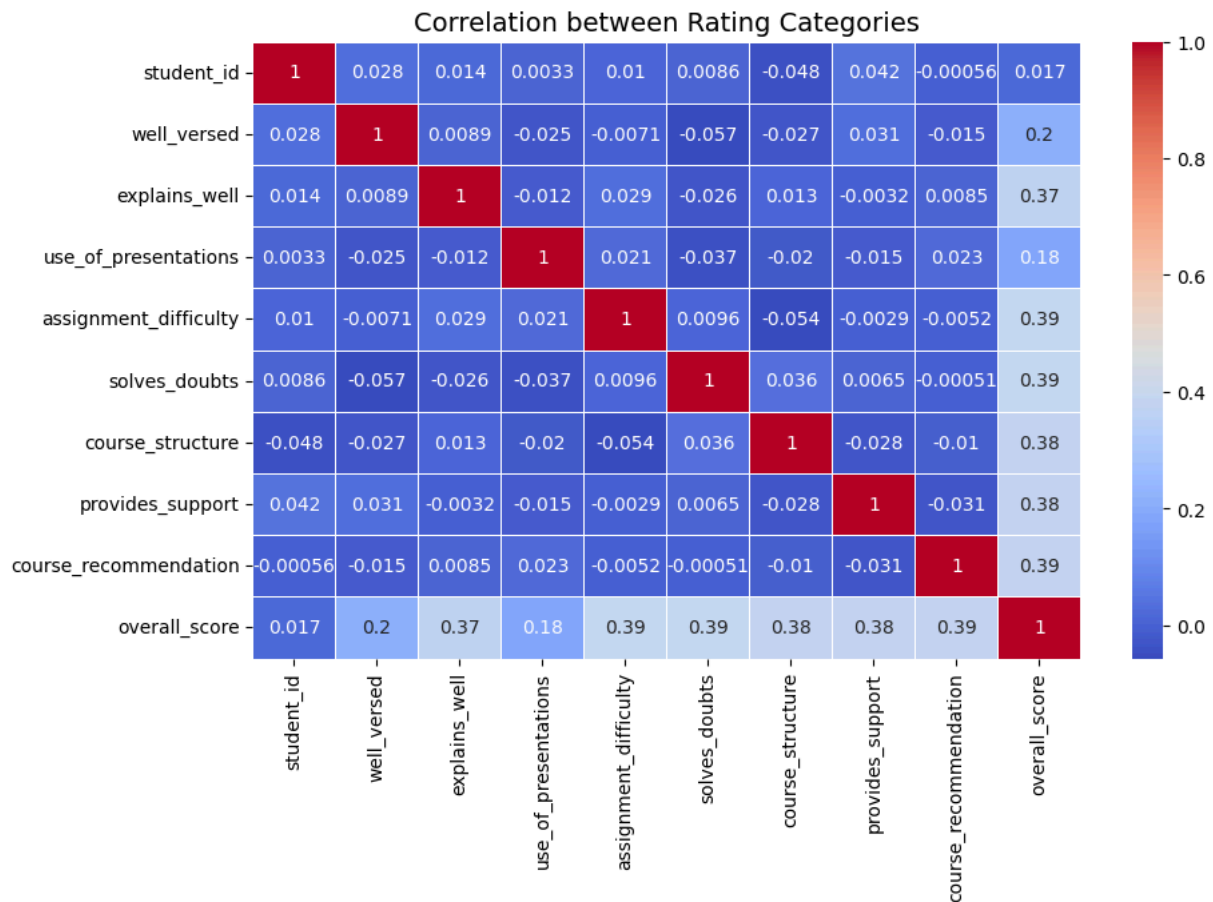
```
# Select only numeric columns for correlation
numeric_df = df.select_dtypes(include=['int64', 'float64'])

# Check which columns were selected
numeric_df.columns
```

```
Index(['student_id', 'well_versed', 'explains_well', 'use_of_presentations',
      'assignment_difficulty', 'solves_doubts', 'course_structure',
      'provides_support', 'course_recommendation', 'overall_score'],
      dtype='object')
```

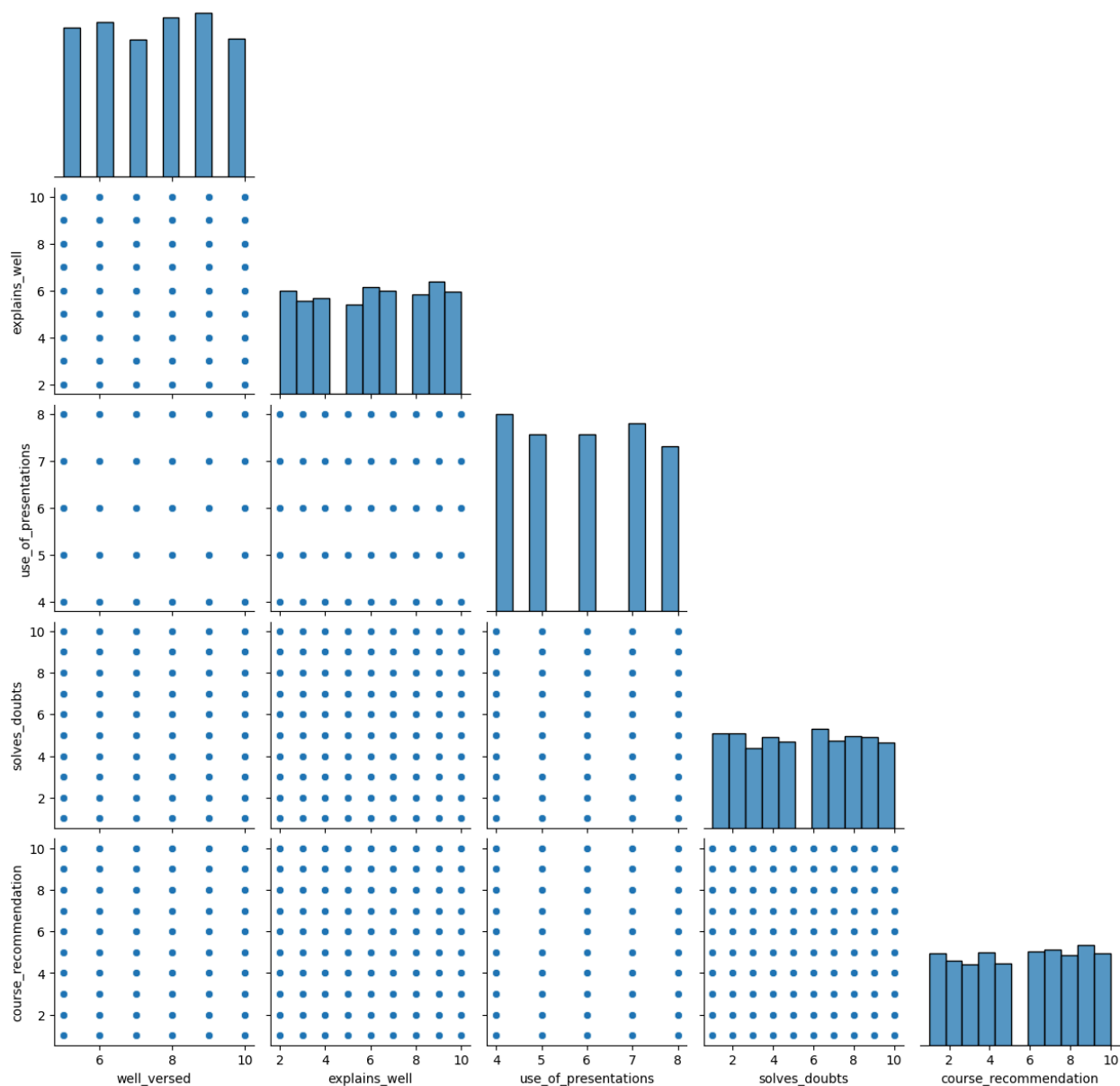
```
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(10,6))
sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm', linewidths=0.5)
plt.title("Correlation between Rating Categories", fontsize=14)
plt.show()
```



```
sns.pairplot(df[['well_versed', 'explains_well', 'use_of_presentations',
                 'solves_doubts', 'course_recommendation']], corner=True)
plt.suptitle("Pairwise Relationships Between Rating Factors", y=1.02)
plt.show()
```

Pairwise Relationships Between Rating Factors

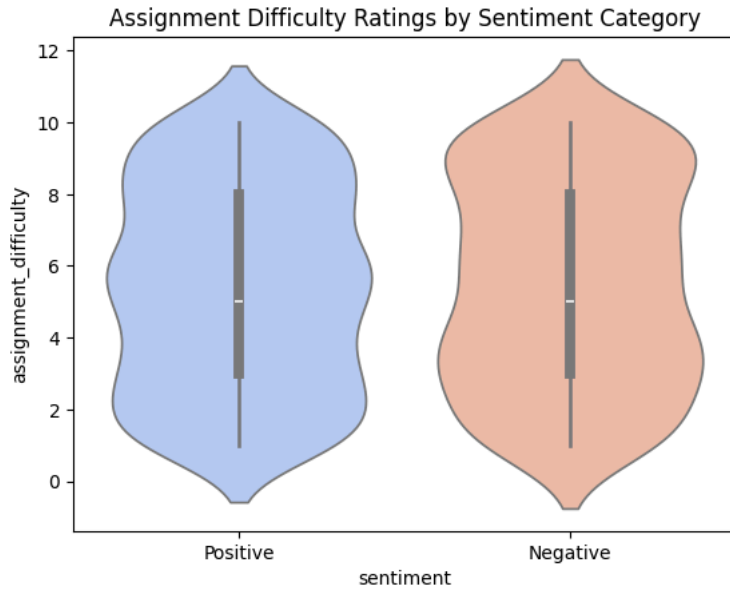


```
sns.violinplot(x='sentiment', y='assignment_difficulty', data=df, palette='coolwarm')
plt.title("Assignment Difficulty Ratings by Sentiment Category")
plt.show()
```

/tmp/ipython-input-117077460.py:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and

```
sns.violinplot(x='sentiment', y='assignment_difficulty', data=df, palette='coolwarm')
```

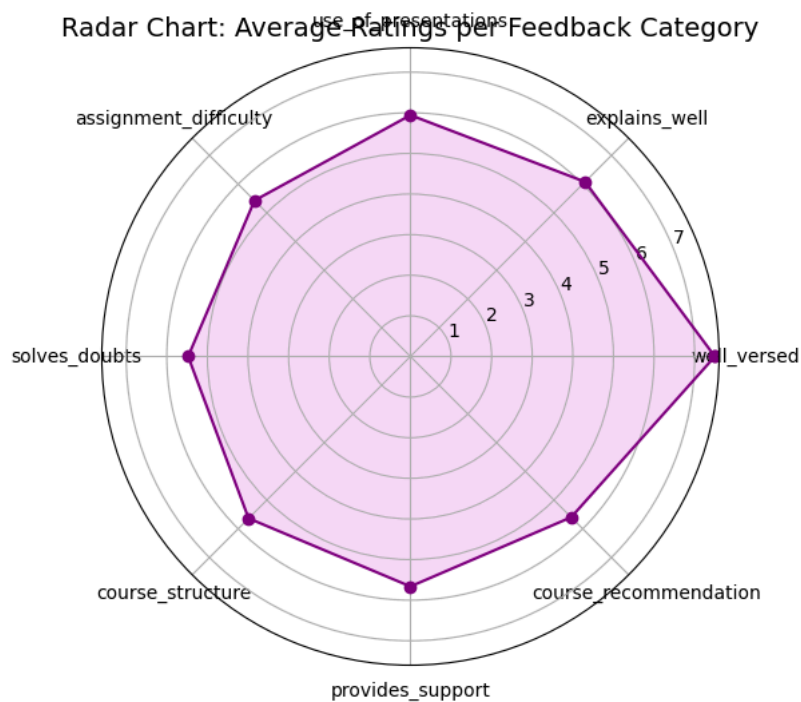


```
import matplotlib.pyplot as plt
import numpy as np

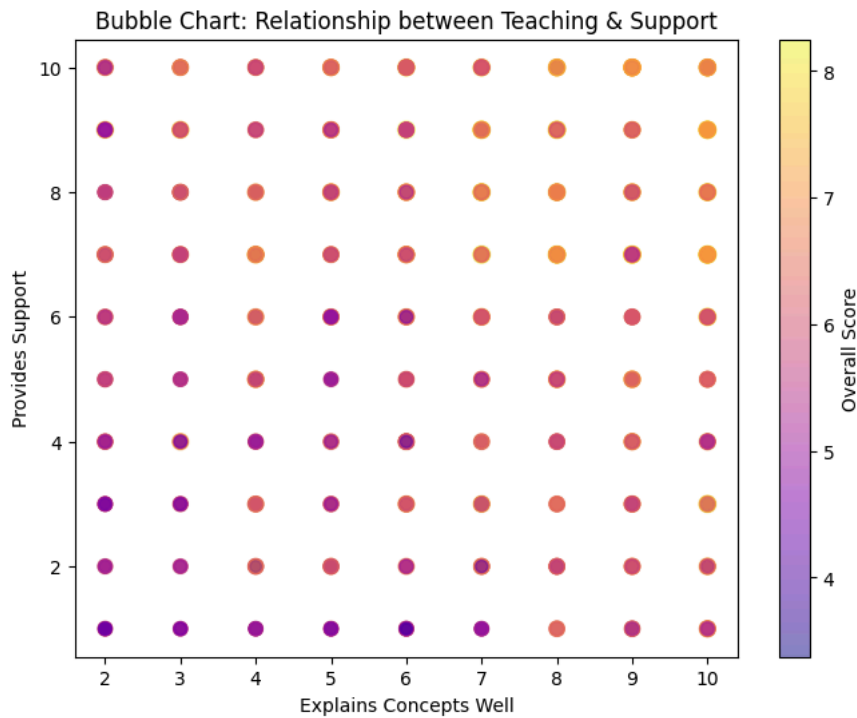
# Average ratings per question
categories = ['well_versed', 'explains_well', 'use_of_presentations',
             'assignment_difficulty', 'solves_doubts', 'course_structure',
             'provides_support', 'course_recommendation']
values = df[categories].mean().tolist()

# Create angles
angles = np.linspace(0, 2 * np.pi, len(categories), endpoint=False).tolist()
values += values[:1]
angles += angles[:1]

# Plot
plt.figure(figsize=(6,6))
plt.polar(angles, values, color='purple', marker='o')
plt.fill(angles, values, alpha=0.3, color='violet')
plt.xticks(angles[:-1], categories, color='black', size=10)
plt.title("Radar Chart: Average Ratings per Feedback Category", size=14)
plt.show()
```



```
plt.figure(figsize=(8,6))
plt.scatter(df['explains_well'], df['provides_support'],
            s=df['overall_score']*10, alpha=0.5, c=df['overall_score'], cmap='plasma')
plt.colorbar(label='Overall Score')
plt.xlabel("Explains Concepts Well")
plt.ylabel("Provides Support")
plt.title("Bubble Chart: Relationship between Teaching & Support")
plt.show()
```



```
avg_ratings = df[categories].mean().sort_values(ascending=False)
plt.figure(figsize=(10,5))
sns.barplot(x=avg_ratings.values, y=avg_ratings.index, palette='Blues_r')
plt.title("Average Ratings per Category")
for i, v in enumerate(avg_ratings.values):
    plt.text(v + 0.1, i, f'{v:.2f}', color='black', va='center')
plt.xlabel("Average Rating (1-10)")
plt.show()
```

/tmp/ipython-input-3362035581.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and

```
sns.barplot(x=avg_ratings.values, y=avg_ratings.index, palette='Blues_r')
```

