Impact of Virtual Reality on Education This dataset delves into the impact of virtual reality (VR) on education, highlighting its potential to enhance learning experiences across various subjects. It encompasses data on student engagement, retention rates, and overall academic performance when VR is integrated into the curriculum. Discover how immersive technologies can revolutionize traditional teaching methods and engage learners in unprecedented ways!

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
import pandas as pd
```

```
df=pd.read_csv('//content/drive/MyDrive/AVDS_project/avds_project.csv')
```

```
df
```

	Student_ID	Age	Gender	Grade_Level	Field_of_Study	Usage_of_VR_in_Education	Hours_of_VR_Usage_Per_Week	Engagement_Level
0	STUD0001	13.0	Non-binary	Postgraduate	Science	No	6.0	1.0
1	STUD0002	16.0	Non-binary	Undergraduate	Medicine	No	6.0	1.0
2	STUD0003	15.0	Prefer not to say	High School	Science	No	4.0	5.0
3	STUD0004	24.0	Female	Postgraduate	Engineering	Yes	2.0	4.0
4	STUD0005	22.0	Non-binary	Undergraduate	Arts	Yes	10.0	3.0
...	...	...	...	...	...	...	...	...
4995	STUD4996	18.0	Male	Undergraduate	Engineering	No	10.0	4.0
4996	STUD4997	25.0	Male	High School	Medicine	No	5.0	1.0
4997	STUD4998	25.0	Female	High School	Engineering	No	1.0	4.0
4998	STUD4999	22.0	Female	Undergraduate	Science	No	4.0	3.0
4999	STUD5000	21.0	Male	Undergraduate	Medicine	Yes	1.0	1.0

5000 rows × 20 columns

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 20 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
 0   Student_ID                              5000 non-null   object
 1   Age                                     4999 non-null   float64
 2   Gender                                  4999 non-null   object
 3   Grade_Level                             4997 non-null   object
 4   Field_of_Study                          4998 non-null   object
 5   Usage_of_VR_in_Education                4999 non-null   object
 6   Hours_of_VR_Usage_Per_Week              4998 non-null   float64
 7   Engagement_Level                        4998 non-null   float64
 8   Improvement_in_Learning_Outcomes        4998 non-null   object
 9   Subject                                 4997 non-null   object
 10  Instructor_VR_Proficiency               4995 non-null   object
 11  Perceived_Effectiveness_of_VR           4999 non-null   float64
 12  Access_to_VR_Equipment                  4994 non-null   object
 13  Impact_on_Creativity                    5000 non-null   int64
 14  Stress_Level_with_VR_Usage              4998 non-null   object
 15  Collaboration_with_Peers_via_VR         4999 non-null   object
 16  Feedback_from_Educators_on_VR           4994 non-null   object
 17  Interest_in_Continuing_VR_Based_Learning  5000 non-null   object
 18  Region                                  4998 non-null   object
 19  School_Support_for_VR_in_Curriculum     5000 non-null   object
dtypes: float64(4), int64(1), object(15)
memory usage: 781.4+ KB
```

```
print(df.columns)
```

```
Index(['Student_ID', 'Age', 'Gender', 'Grade_Level', 'Field_of_Study',
       'Usage_of_VR_in_Education', 'Hours_of_VR_Usage_Per_Week',
       'Engagement_Level', 'Improvement_in_Learning_Outcomes', 'Subject',
       'Instructor_VR_Proficiency', 'Perceived_Effectiveness_of_VR',
       'Access_to_VR_Equipment', 'Impact_on_Creativity',
       'Stress_Level_with_VR_Usage', 'Collaboration_with_Peers_via_VR',
       'Feedback_from_Educators_on_VR',
       'Interest_in_Continuing_VR_Based_Learning', 'Region',
       'School_Support_for_VR_in_Curriculum'],
      dtype='object')
```

```
df.isna()
```

	Student_ID	Age	Gender	Grade_Level	Field_of_Study	Usage_of_VR_in_Education	Hours_of_VR_Usage_Per_Week	Engagement_Level
0	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...
4995	False	False	False	False	False	False	False	False
4996	False	False	False	False	False	False	False	False
4997	False	False	False	False	False	False	False	False
4998	False	False	False	False	False	False	False	False
4999	False	False	False	False	False	False	False	False

5000 rows × 20 columns

```
df.isna().sum()
```

	0
Student_ID	0
Age	1
Gender	1
Grade_Level	3
Field_of_Study	2
Usage_of_VR_in_Education	1
Hours_of_VR_Usage_Per_Week	2
Engagement_Level	2
Improvement_in_Learning_Outcomes	2
Subject	3
Instructor_VR_Proficiency	5
Perceived_Effectiveness_of_VR	1
Access_to_VR_Equipment	6
Impact_on_Creativity	0
Stress_Level_with_VR_Usage	2
Collaboration_with_Peers_via_VR	1
Feedback_from_Educators_on_VR	6
Interest_in_Continuing_VR_Based_Learning	0
Region	2
School_Support_for_VR_in_Curriculum	0

dtype: int64

```
df.isna().sum().sum()
```

```
40
```

```
df1= df.fillna(method='ffill')
```

```
<ipython-input-125-7a26809f3b85>:1: FutureWarning: DataFrame.fillna with 'method' is deprecated and will raise in a future version.
  df1= df.fillna(method='ffill')
```

```
df1
```

	Student_ID	Age	Gender	Grade_Level	Field_of_Study	Usage_of_VR_in_Education	Hours_of_VR_Usage_Per_Week	Engagement_Level
0	STUD0001	13.0	Non-binary	Postgraduate	Science	No	6.0	1.0
1	STUD0002	16.0	Non-binary	Undergraduate	Medicine	No	6.0	1.0
2	STUD0003	15.0	Prefer not to say	High School	Science	No	4.0	5.0
3	STUD0004	24.0	Female	Postgraduate	Engineering	Yes	2.0	4.0
4	STUD0005	22.0	Non-binary	Undergraduate	Arts	Yes	10.0	3.0
...	...	...	...	...	...	...	...	...
4995	STUD4996	18.0	Male	Undergraduate	Engineering	No	10.0	4.0
4996	STUD4997	25.0	Male	High School	Medicine	No	5.0	1.0
4997	STUD4998	25.0	Female	High School	Engineering	No	1.0	4.0
4998	STUD4999	22.0	Female	Undergraduate	Science	No	4.0	3.0
4999	STUD5000	21.0	Male	Undergraduate	Medicine	Yes	1.0	1.0

5000 rows × 20 columns

```
df1.isna().sum()
```

	0
Student_ID	0
Age	0
Gender	0
Grade_Level	0
Field_of_Study	0
Usage_of_VR_in_Education	0
Hours_of_VR_Usage_Per_Week	0
Engagement_Level	0
Improvement_in_Learning_Outcomes	0
Subject	0
Instructor_VR_Proficiency	0
Perceived_Effectiveness_of_VR	0
Access_to_VR_Equipment	0
Impact_on_Creativity	0
Stress_Level_with_VR_Usage	0
Collaboration_with_Peers_via_VR	0
Feedback_from_Educators_on_VR	0
Interest_in_Continuing_VR_Based_Learning	0
Region	0
School_Support_for_VR_in_Curriculum	0

dtype: int64

```
df1.isna().sum().sum()
```

```
0
```

```
df1.head()
```

	Student_ID	Age	Gender	Grade_Level	Field_of_Study	Usage_of_VR_in_Education	Hours_of_VR_Usage_Per_Week	Engagement_Level	In
0	STUD0001	13.0	Non-binary	Postgraduate	Science	No	6.0	1.0	
1	STUD0002	16.0	Non-binary	Undergraduate	Medicine	No	6.0	1.0	
2	STUD0003	15.0	Prefer not to say	High School	Science	No	4.0	5.0	
3	STUD0004	24.0	Female	Postgraduate	Engineering	Yes	2.0	4.0	
4	STUD0005	22.0	Non-binary	Undergraduate	Arts	Yes	10.0	3.0	

```
df1.tail()
```

	Student_ID	Age	Gender	Grade_Level	Field_of_Study	Usage_of_VR_in_Education	Hours_of_VR_Usage_Per_Week	Engagement_Level
4995	STUD4996	18.0	Male	Undergraduate	Engineering	No	10.0	4.0
4996	STUD4997	25.0	Male	High School	Medicine	No	5.0	1.0
4997	STUD4998	25.0	Female	High School	Engineering	No	1.0	4.0
4998	STUD4999	22.0	Female	Undergraduate	Science	No	4.0	3.0
4999	STUD5000	21.0	Male	Undergraduate	Medicine	Yes	1.0	1.0

SHOW STUDENTS THAT USE VR IN EDUCATION AND ARE NOT INTERESTED IN CONTINUING VR BASED LEARNING

```
df2=df1[(df1['Usage_of_VR_in_Education']=='Yes')&(df1['Interest_in_Continuing_VR_Based_Learning']=='No')]
df2
```

	Student_ID	Age	Gender	Grade_Level	Field_of_Study	Usage_of_VR_in_Education	Hours_of_VR_Usage_Per_Week	Engagement_Level
3	STUD0004	24.0	Female	Postgraduate	Engineering	Yes	2.0	4.0
5	STUD0006	28.0	Male	High School	Science	Yes	10.0	5.0
6	STUD0007	19.0	Male	Undergraduate	Business	Yes	9.0	4.0
11	STUD0012	15.0	Non-binary	Undergraduate	Arts	Yes	5.0	1.0
15	STUD0016	18.0	Female	Postgraduate	Law	Yes	3.0	1.0
...	...	...	...	...	...	...	...	...
4983	STUD4984	19.0	Female	High School	Arts	Yes	8.0	1.0
4987	STUD4988	24.0	Prefer not to say	High School	Science	Yes	10.0	2.0
4992	STUD4993	28.0	Prefer not to say	Undergraduate	Education	Yes	8.0	1.0
4993	STUD4994	17.0	Non-binary	Undergraduate	Law	Yes	7.0	4.0
4994	STUD4995	19.0	Male	Postgraduate	Education	Yes	9.0	5.0

1243 rows × 20 columns

```
len(df2)
```

```
1243
```

SHOW DATA OF STUDENTS WITH NEUTRAL FEEDBACK AND ENGAGEMENT LEVEL GREATER THAN 3

```
df3=df1[(df1['Feedback_from_Educators_on_VR']=='Neutral')&(df1['Engagement_Level']>3)]
df3
```

	Student_ID	Age	Gender	Grade_Level	Field_of_Study	Usage_of_VR_in_Education	Hours_of_VR_Usage_Per_Week	Engagement_Level
2	STUD0003	15.0	Prefer not to say	High School	Science	No	4.0	5.0
3	STUD0004	24.0	Female	Postgraduate	Engineering	Yes	2.0	4.0
5	STUD0006	28.0	Male	High School	Science	Yes	10.0	5.0
6	STUD0007	19.0	Male	Undergraduate	Business	Yes	9.0	4.0
7	STUD0008	19.0	Male	High School	Education	No	1.0	5.0
...	...	...	...	...	...	...	...	...
4975	STUD4976	20.0	Male	Undergraduate	Science	Yes	7.0	5.0
4980	STUD4981	16.0	Non-binary	Postgraduate	Arts	No	7.0	5.0
4993	STUD4994	17.0	Non-binary	Undergraduate	Law	Yes	7.0	4.0
4994	STUD4995	19.0	Male	Postgraduate	Education	Yes	9.0	5.0
4995	STUD4996	18.0	Male	Undergraduate	Engineering	No	10.0	4.0

655 rows × 20 columns

```
len(df3)
```

```
655
```

```
df1['Instructor_VR_Proficiency'].unique()
```

```
array(['Intermediate', 'Beginner', 'Advanced'], dtype=object)
```

```
def VR_Proficiency_grade(Instructor_VR_Proficiency):
  if Instructor_VR_Proficiency=='Beginner':
    return 'c+'
  elif Instructor_VR_Proficiency=='Intermediate':
    return 'B+'
  elif Instructor_VR_Proficiency=='Advanced':
    return 'A+'
```

```
df4=df1['Instructor_VR_Proficiency'].apply(VR_Proficiency_grade)
```

```
df4
```

	Instructor_VR_Proficiency
0	B+
1	c+
2	A+
3	c+
4	c+
...	...
4995	A+
4996	A+
4997	c+
4998	c+
4999	B+

5000 rows × 1 columns

dtype: object

Statistical analysis of data using pandas

```
df1['Engagement_Level'].mean()
```

```
3.0212
```

```
df1['Age'].median()
```

```
21.0
```

```
mean_overall=df1[['Age','Hours_of_VR_Usage_Per_Week','Engagement_Level','Perceived_Effectiveness_of_VR','Impact_on_Creativity']].mean()
print(mean_overall)
```

```
Age                             21.1820
Hours_of_VR_Usage_Per_Week       5.0282
Engagement_Level                 3.0212
Perceived_Effectiveness_of_VR    2.9518
Impact_on_Creativity             3.0196
dtype: float64
```

```
mode_overall=df1[['Age','Hours_of_VR_Usage_Per_Week','Engagement_Level','Perceived_Effectiveness_of_VR','Impact_on_Creativity']].mode()
print(mode_overall)
```

```
    Age  Hours_of_VR_Usage_Per_Week  Engagement_Level  \
0  21.0                         6.0               5.0

   Perceived_Effectiveness_of_VR  Impact_on_Creativity
0                            1.0                     5
```

```
df1['Hours_of_VR_Usage_Per_Week'].mode()
```

	Hours_of_VR_Usage_Per_Week
0	6.0

dtype: float64

```
from sklearn.preprocessing import LabelEncoder
```

```
df1=df1.drop(['Gender'],axis=1)
```

```
data=df1.drop(['Grade_Level'],axis=1)
```

```
le = LabelEncoder()
data['Instructor_VR_Proficiency_encoded'] = le.fit_transform(data['Instructor_VR_Proficiency'])
data['Field_of_Study_encoded'] = le.fit_transform(data['Field_of_Study'])
data['Improvement_in_Learning_Outcomes_encoded'] = le.fit_transform(data['Improvement_in_Learning_Outcomes'])
data['Usage_of_VR_in_Education  _encoded'] = le.fit_transform(data['Usage_of_VR_in_Education'])
data['Access_to_VR_Equipment_encoded'] = le.fit_transform(data['Access_to_VR_Equipment'])
data['Subject_encoded'] = le.fit_transform(data['Subject'])
data['Stress_Level_with_VR_Usage_encoded'] = le.fit_transform(data['Stress_Level_with_VR_Usage'])
data['Collaboration_with_Peers_via_VR_encoded'] = le.fit_transform(data['Collaboration_with_Peers_via_VR'])
data['Feedback_from_Educators_on_VR_encoded'] = le.fit_transform(data['Feedback_from_Educators_on_VR'])
data['Region_encoded'] = le.fit_transform(data['Region'])
data['School_Support_for_VR_in_Curriculum_encoded'] = le.fit_transform(data['School_Support_for_VR_in_Curriculum'])
```

```
data
```

	Student_ID	Age	Field_of_Study	Usage_of_VR_in_Education	Hours_of_VR_Usage_Per_Week	Engagement_Level	Improvement_in_Learn
0	STUD0001	13.0	Science	No	6.0	1.0	
1	STUD0002	16.0	Medicine	No	6.0	1.0	
2	STUD0003	15.0	Science	No	4.0	5.0	
3	STUD0004	24.0	Engineering	Yes	2.0	4.0	
4	STUD0005	22.0	Arts	Yes	10.0	3.0	
...	...	...	...	...	...	...	
4995	STUD4996	18.0	Engineering	No	10.0	4.0	
4996	STUD4997	25.0	Medicine	No	5.0	1.0	
4997	STUD4998	25.0	Engineering	No	1.0	4.0	
4998	STUD4999	22.0	Science	No	4.0	3.0	
4999	STUD5000	21.0	Medicine	Yes	1.0	1.0	

5000 rows × 29 columns

```
data=data.drop(['Instructor_VR_Proficiency','Field_of_Study','Improvement_in_Learning_Outcomes','Usage_of_VR_in_Education','Interest_in_
```

```
data
```

	Student_ID	Age	Hours_of_VR_Usage_Per_Week	Engagement_Level	Perceived_Effectiveness_of_VR	Impact_on_Creativity	Instructo
0	STUD0001	13.0	6.0	1.0	3.0	5	
1	STUD0002	16.0	6.0	1.0	2.0	3	
2	STUD0003	15.0	4.0	5.0	5.0	2	
3	STUD0004	24.0	2.0	4.0	5.0	3	
4	STUD0005	22.0	10.0	3.0	4.0	1	
...	...	...	...	...	...	...	
4995	STUD4996	18.0	10.0	4.0	1.0	1	
4996	STUD4997	25.0	5.0	1.0	4.0	5	
4997	STUD4998	25.0	1.0	4.0	2.0	3	
4998	STUD4999	22.0	4.0	3.0	3.0	3	
4999	STUD5000	21.0	1.0	1.0	2.0	4	

5000 rows × 17 columns

```
data=data.drop(['Interest_in_Continuing_VR_Based_Learning'],axis=1)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-151-36b585896afb> in <cell line: 1>()
----> 1 data=data.drop(['Interest_in_Continuing_VR_Based_Learning'],axis=1)

                                 3 frames
/usr/local/lib/python3.10/dist-packages/pandas/core/indexes/base.py in drop(self, labels, errors)
   7068         if mask.any():
   7069             if errors != "ignore":
-> 7070                 raise KeyError(f"{labels[mask].tolist()} not found in axis")
   7071             indexer = indexer[~mask]
   7072         return self.delete(indexer)

KeyError: "['Interest_in_Continuing_VR_Based_Learning'] not found in axis"
```

```
data=data.drop(['Student_ID'],axis=1)
```

```
correlation_matrix = data.corr()
```

```
print(correlation_matrix)
```

```
                                                 Age  \
Age                                         1.000000
Hours_of_VR_Usage_Per_Week                  0.012832
Engagement_Level                            0.007589
Perceived_Effectiveness_of_VR               0.012834
Impact_on_Creativity                       -0.008505
Instructor_VR_Proficiency_encoded          -0.017905
Field_of_Study_encoded                     -0.006167
Improvement_in_Learning_Outcomes_encoded    0.008336
Usage_of_VR_in_Education\t_encoded          0.004987
Access_to_VR_Equipment_encoded             -0.017800
Subject_encoded                            -0.001632
Stress_Level_with_VR_Usage_encoded         -0.008761
Collaboration_with_Peers_via_VR_encoded     0.000293
Feedback_from_Educators_on_VR_encoded      -0.009463
Region_encoded                             -0.021031
School_Support_for_VR_in_Curriculum_encoded -0.007539

                                            Hours_of_VR_Usage_Per_Week  \
Age                                                           0.012832
Hours_of_VR_Usage_Per_Week                                    1.000000
Engagement_Level                                              0.006205
Perceived_Effectiveness_of_VR                                 0.004169
Impact_on_Creativity                                          0.010690
Instructor_VR_Proficiency_encoded                            -0.006268
Field_of_Study_encoded                                        0.004176
Improvement_in_Learning_Outcomes_encoded                      0.001931
Usage_of_VR_in_Education\t_encoded                           -0.006403
Access_to_VR_Equipment_encoded                                0.019309
Subject_encoded                                              -0.002538
Stress_Level_with_VR_Usage_encoded                            0.012053
Collaboration_with_Peers_via_VR_encoded                      -0.003964
Feedback_from_Educators_on_VR_encoded                        -0.010199
Region_encoded                                               -0.023179
School_Support_for_VR_in_Curriculum_encoded                  -0.004049

                                            Engagement_Level  \
Age                                                 0.007589
Hours_of_VR_Usage_Per_Week                          0.006205
Engagement_Level                                    1.000000
Perceived_Effectiveness_of_VR                      -0.009974
Impact_on_Creativity                               -0.004200
Instructor_VR_Proficiency_encoded                  -0.005129
Field_of_Study_encoded                             -0.017001
Improvement_in_Learning_Outcomes_encoded           -0.009602
Usage_of_VR_in_Education\t_encoded                 -0.012214
Access_to_VR_Equipment_encoded                     -0.022623
Subject_encoded                                    -0.036225
Stress_Level_with_VR_Usage_encoded                 -0.014311
Collaboration_with_Peers_via_VR_encoded             0.006036
Feedback_from_Educators_on_VR_encoded              -0.007255
Region_encoded                                      0.012006
School_Support_for_VR_in_Curriculum_encoded        -0.011694

                                            Perceived_Effectiveness_of_VR  \
Age                                                              0.012834
Hours_of_VR_Usage_Per_Week                                       0.004169
Engagement_Level                                                -0.009974
```

```
df1.describe()
#generates descriptive statistics
```

	Age	Hours_of_VR_Usage_Per_Week	Engagement_Level	Perceived_Effectiveness_of_VR	Impact_on_Creativity
count	5000.000000	5000.00000	5000.00000	5000.000000	5000.000000
mean	21.182000	5.02820	3.02120	2.951800	3.019600
std	5.461945	3.14041	1.42715	1.417843	1.437508
min	12.000000	0.00000	1.00000	1.000000	1.000000
25%	16.000000	2.00000	2.00000	2.000000	2.000000
50%	21.000000	5.00000	3.00000	3.000000	3.000000
75%	26.000000	8.00000	4.00000	4.000000	4.000000
max	30.000000	10.00000	5.00000	5.000000	5.000000

```
df1.corr(numeric_only=True)
```

	Age	Hours_of_VR_Usage_Per_Week	Engagement_Level	Perceived_Effectiveness_of_VR	Impact_on_Creat
Age	1.000000	0.012832	0.007589	0.012834	-0.(
Hours_of_VR_Usage_Per_Week	0.012832	1.000000	0.006205	0.004169	0.(
Engagement_Level	0.007589	0.006205	1.000000	-0.009974	-0.(
Perceived_Effectiveness_of_VR	0.012834	0.004169	-0.009974	1.000000	0.(
Impact_on_Creativity	-0.008505	0.010690	-0.004200	0.007432	1.(

```
df1.cov(numeric_only=True)
```

	Age	Hours_of_VR_Usage_Per_Week	Engagement_Level	Perceived_Effectiveness_of_VR	Impact_on_Crea
Age	29.832843	0.220112	0.059153	0.099392	-0.
Hours_of_VR_Usage_Per_Week	0.220112	9.862177	0.027808	0.018563	0.
Engagement_Level	0.059153	0.027808	2.036758	-0.020182	-0.
Perceived_Effectiveness_of_VR	0.099392	0.018563	-0.020182	2.010279	0.
Impact_on_Creativity	-0.066781	0.048257	-0.008617	0.015148	2.

```
data.cov()
```

	Age	Hours_of_VR_Usage_Per_Week	Engagement_Level	Perceived_Effectiveness_of_VR
Age	29.832843	0.220112	0.059153	0.099392
Hours_of_VR_Usage_Per_Week	0.220112	9.862177	0.027808	0.018563
Engagement_Level	0.059153	0.027808	2.036758	-0.020182
Perceived_Effectiveness_of_VR	0.099392	0.018563	-0.020182	2.010279
Impact_on_Creativity	-0.066781	0.048257	-0.008617	0.015148
Instructor_VR_Proficiency_encoded	-0.079490	-0.016001	-0.005950	0.016941
Field_of_Study_encoded	-0.067982	0.026466	-0.048965	-0.012340
Improvement_in_Learning_Outcomes_encoded	0.022768	0.003033	-0.006852	0.010618
Usage_of_VR_in_Education\t_encoded	0.013620	-0.010054	-0.008716	0.000560
Access_to_VR_Equipment_encoded	-0.048611	0.030320	-0.016143	0.014621
Subject_encoded	-0.017879	-0.015987	-0.103709	0.006857
Stress_Level_with_VR_Usage_encoded	-0.039023	0.030869	-0.016656	-0.009992
Collaboration_with_Peers_via_VR_encoded	0.000799	-0.006225	0.004308	0.002113
Feedback_from_Educators_on_VR_encoded	-0.042358	-0.026248	-0.008485	-0.005102
Region_encoded	-0.197621	-0.125229	0.029478	0.097356
School_Support_for_VR_in_Curriculum_encoded	-0.020584	-0.006357	-0.008343	-0.004089

```
df2['Hours_of_VR_Usage_Per_Week_rating']=np.where(df2['Hours_of_VR_Usage_Per_Week']>2.0,'yes','no')
print(df2[['Hours_of_VR_Usage_Per_Week','Hours_of_VR_Usage_Per_Week_rating']])
```

```
      Hours_of_VR_Usage_Per_Week Hours_of_VR_Usage_Per_Week_rating
3                            2.0                                no
5                           10.0                               yes
6                            9.0                               yes
11                           5.0                               yes
15                           3.0                               yes
...                          ...                               ...
4983                         8.0                               yes
4987                        10.0                               yes
4992                         8.0                               yes
4993                         7.0                               yes
4994                         9.0                               yes

[1243 rows x 2 columns]
<ipython-input-159-f9c3a98ba809>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  df2['Hours_of_VR_Usage_Per_Week_rating']=np.where(df2['Hours_of_VR_Usage_Per_Week']>2.0,'yes','no')
```

```
interval=[1.0,2.0,3.0,4.0,5.0,6.0,7.0,8.0,9.0,10.0]
label=['1.0-2.0','2.0-3.0','3.0-4.0','4.0-5.0','5.0-6.0','6.0-7.0','7.0-8.0','8.0-9.0','9.0-10.0']
df2['Hours_of_VR_Usage_Per_Week']=pd.cut(df2['Hours_of_VR_Usage_Per_Week'],bins=interval,labels=label,include_lowest='True')
print(df2[['Hours_of_VR_Usage_Per_Week']].head())
```

```
   Hours_of_VR_Usage_Per_Week
3                     1.0-2.0
5                    9.0-10.0
6                     8.0-9.0
11                    4.0-5.0
15                    2.0-3.0
<ipython-input-160-f4fb7f311f3e>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  df2['Hours_of_VR_Usage_Per_Week']=pd.cut(df2['Hours_of_VR_Usage_Per_Week'],bins=interval,labels=label,include_lowest='True')
```

```
x=df1.groupby('Usage_of_VR_in_Education')['Instructor_VR_Proficiency']
x.count()
```

Usage_of_VR_in_Education	Instructor_VR_Proficiency
No	2473
Yes	2527

dtype: int64

```
x.value_counts()
```

Usage_of_VR_in_Education	Instructor_VR_Proficiency	count
No	Beginner	832
	Intermediate	826
	Advanced	815
Yes	Beginner	865
	Intermediate	843
	Advanced	819

dtype: int64

```
y=df1.groupby('Instructor_VR_Proficiency')['Usage_of_VR_in_Education']
y.count()
```

Instructor_VR_Proficiency	Usage_of_VR_in_Education
Advanced	1634
Beginner	1697
Intermediate	1669

dtype: int64

```
y.value_counts()
```

Instructor_VR_Proficiency	Usage_of_VR_in_Education	count
Advanced	Yes	819
	No	815
Beginner	Yes	865
	No	832
Intermediate	Yes	843
	No	826

dtype: int64

```
a=df1.groupby('Field_of_Study')['Usage_of_VR_in_Education']
a.count()
```

Field_of_Study	Usage_of_VR_in_Education
Arts	710
Business	727
Education	694
Engineering	693
Law	691
Medicine	737
Science	748

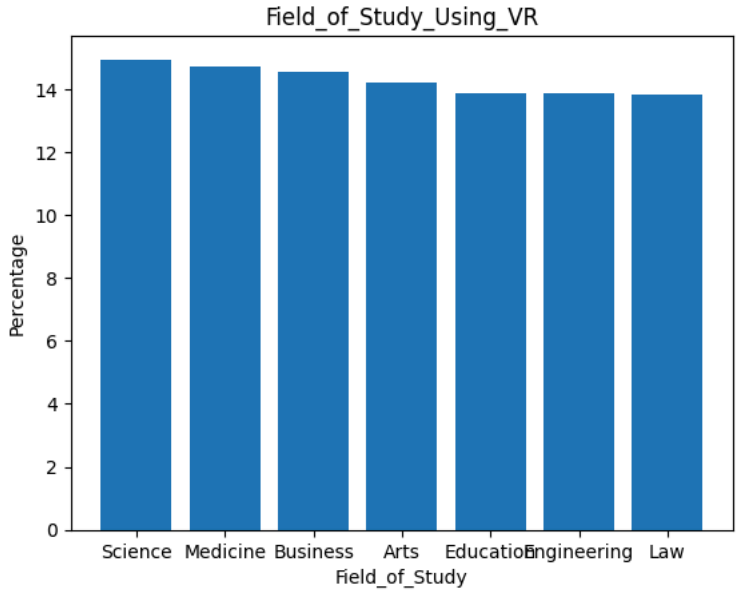
dtype: int64

```
a.value_counts()
```

Field_of_Study	Usage_of_VR_in_Education	count
Arts	Yes	364
	No	346
Business	Yes	367
	No	360
Education	Yes	357
	No	337
Engineering	Yes	349
	No	344
Law	Yes	352
	No	339
Medicine	Yes	378
	No	359
Science	No	388
	Yes	360

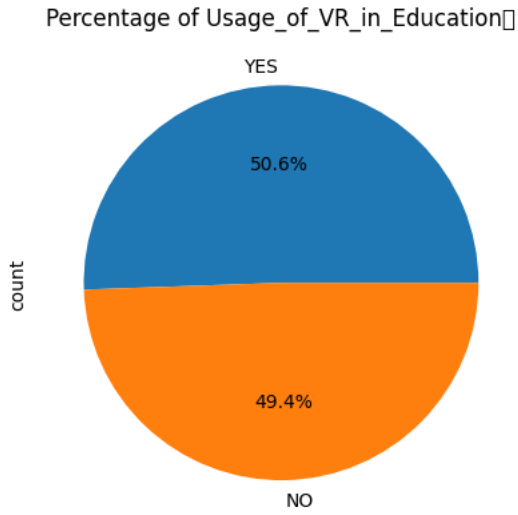
dtype: int64

```
Field_of_Study_counts = df1['Field_of_Study'].value_counts()
Field_of_Study_percentage = Field_of_Study_counts / Field_of_Study_counts.sum() * 100
import matplotlib.pyplot as plt
plt.bar(Field_of_Study_percentage.index,Field_of_Study_percentage.values)
plt.xlabel("Field_of_Study")
plt.ylabel("Percentage")
plt.title("Field_of_Study_Using_VR")
plt.show()
```

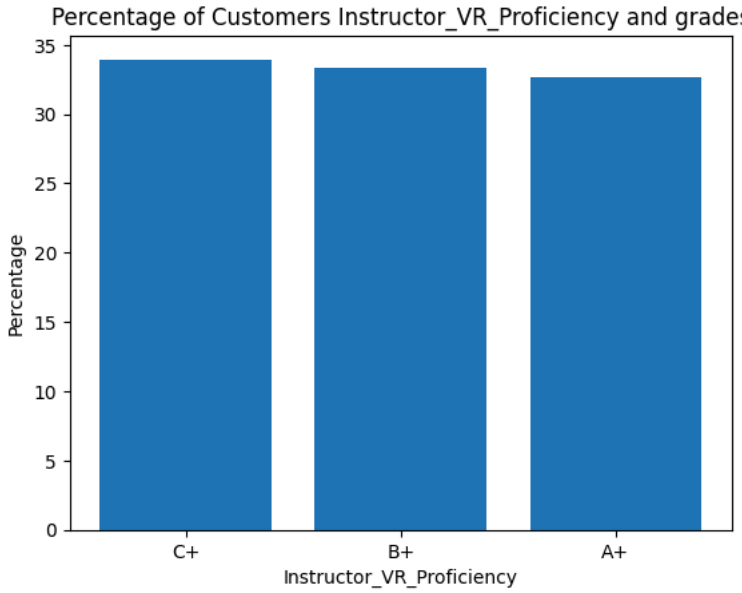



```
Usage_of_VR_in_Education_counts=df['Usage_of_VR_in_Education'].value_counts()
Usage_of_VR_in_Education_percentage=Usage_of_VR_in_Education_counts/Usage_of_VR_in_Education_counts.sum()*100
label=['YES','NO']
Usage_of_VR_in_Education_percentage.plot(kind='pie',labels=label,autopct='%1.1f%%')
plt.title('Percentage of Usage_of_VR_in_Education   ')
plt.show()
```

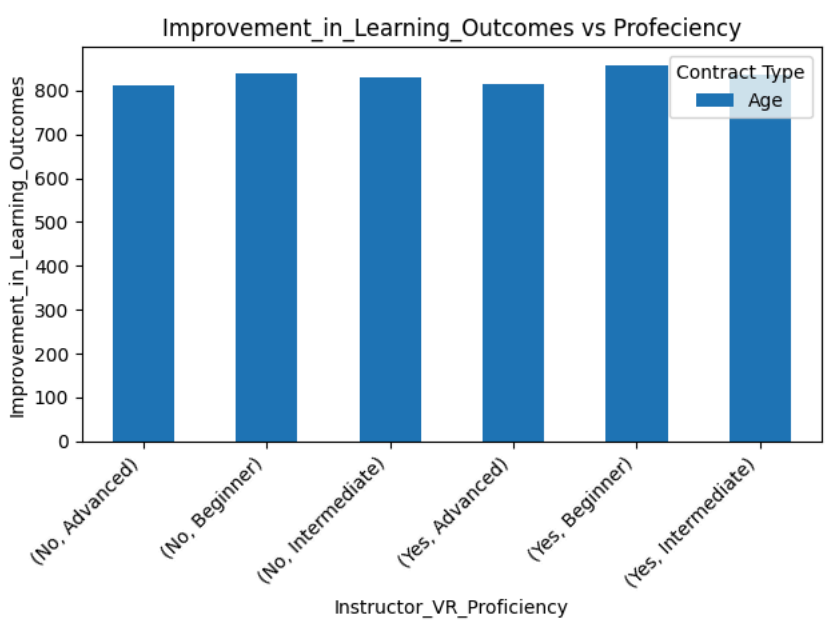
```
/usr/local/lib/python3.10/dist-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 9 (   ) missing from current font.
  fig.canvas.print_figure(bytes_io, **kw)
```



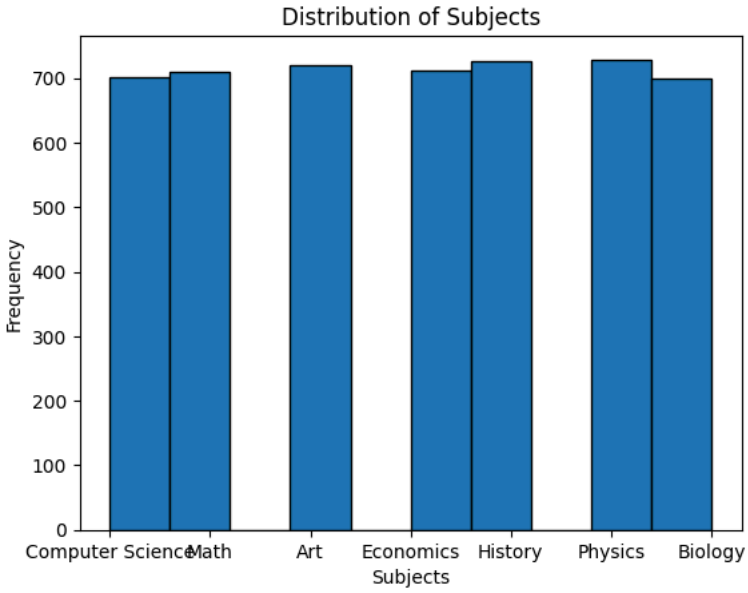

```
Instructor_VR_Proficiency_counts = df['Instructor_VR_Proficiency'].value_counts()
Instructor_VR_Proficiency_percentage =Instructor_VR_Proficiency_counts / Instructor_VR_Proficiency_counts.sum() * 100
import matplotlib.pyplot as plt
plt.bar(Instructor_VR_Proficiency_percentage.index, Instructor_VR_Proficiency_percentage.values)
plt.xlabel("Instructor_VR_Proficiency")
plt.ylabel("Percentage")
plt.title("Percentage of Customers Instructor_VR_Proficiency and grades")
plt.xticks(["Beginner","Intermediate","Advanced"], ['C+', 'B+','A+'])  # Customize x-axis labels for better readability
plt.show()
```



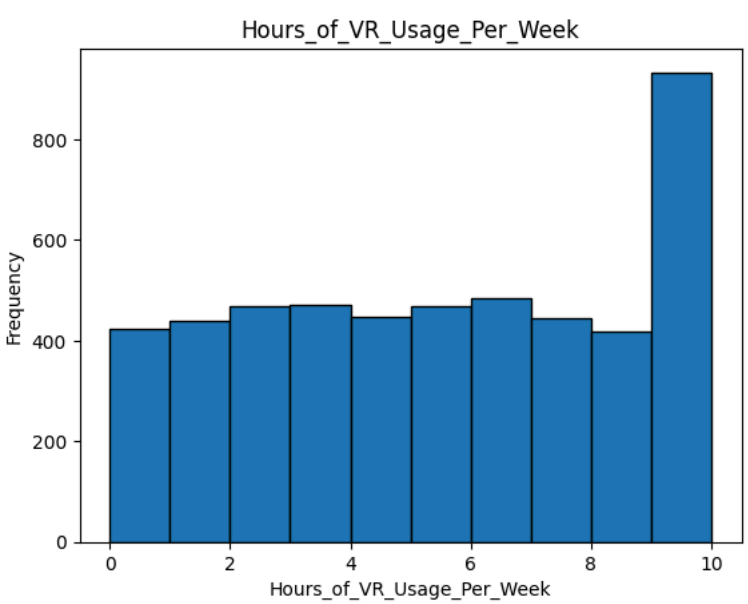

```
internet_contract_counts = df.groupby(['Improvement_in_Learning_Outcomes', 'Instructor_VR_Proficiency'])['Age'].count()
internet_contract_counts.plot(kind='bar')
plt.xlabel("Instructor_VR_Proficiency")
plt.ylabel("Improvement_in_Learning_Outcomes")
plt.title("Improvement_in_Learning_Outcomes vs Profeciency")
plt.xticks(rotation=45, ha='right')  # Rotate x-axis labels for readability
plt.legend(title="Contract Type")
plt.tight_layout()  # Adjust layout to prevent overlapping labels
plt.show()
```



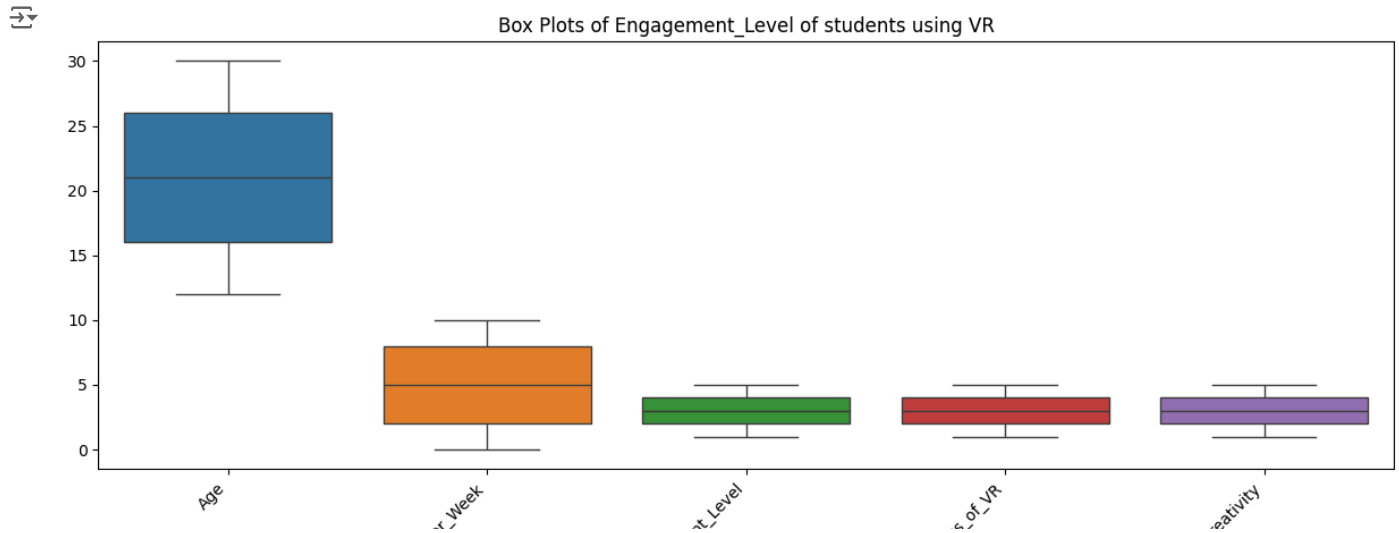

```
import matplotlib.pyplot as plt
plt.hist(df1['Subject'], bins=10, edgecolor='black')  # Adjust 'bins' for desired granularity
plt.xlabel("Subjects")
plt.ylabel("Frequency")
plt.title("Distribution of Subjects")
plt.show()
```




```
import matplotlib.pyplot as plt
plt.hist(df1['Hours_of_VR_Usage_Per_Week'], bins=10, edgecolor='black')  # Adjust 'bins' for desired granularity
plt.xlabel("Hours_of_VR_Usage_Per_Week")
plt.ylabel("Frequency")
plt.title("Hours_of_VR_Usage_Per_Week")
plt.show()
```




```
import matplotlib.pyplot as plt
import seaborn as sns
# Select numerical columns
Engagement_Level_cols = df1.select_dtypes(include=['number']).columns
# Create box plots
plt.figure(figsize=(12, 6))  # Adjust figure size as needed
sns.boxplot(data=df1[Engagement_Level_cols])
plt.xticks(rotation=45, ha='right')  # Rotate x-axis labels for readability
plt.title("Box Plots of Engagement_Level of students using VR")
plt.tight_layout()  # Adjust layout to prevent overlapping labels
plt.show()
```



Decision Tree

Double-click (or enter) to edit

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn import tree
import matplotlib.pyplot as plt
import numpy as np
# Import the necessary function
from sklearn.metrics import accuracy_score

# Drop 'customerID' column as it's not useful for prediction
X=data.drop(['Engagement_Level', 'Age'],axis=1)
y=data['Engagement_Level']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
clf = DecisionTreeClassifier(random_state=42)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
y_pred = clf.predict(X_test)
# Now you can use accuracy_score
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.2f}")
```

```
Accuracy: 0.19
```

Naive Bayes Algorithm

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score
X = data[['Age','Access_to_VR_Equipment_encoded']] # Use a list of column names within double brackets
y = data['Age']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
clf = GaussianNB()
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
```