در ابتدا 5 درس مورد نظر با ویژگی های مورد نظر را می سازیم:

```python
# Create a dictionary to hold the data
data = {
    'Lesson': ['English Language 2', 'Physics 2', 'Discrete Mathematics', 'Electric and Electronic Circuits', 'Database Design'],
    'Semester_Passed': ['Even', 'Odd', 'Even', 'Even', 'Odd'],
    'ECTS': [2, 3, 3, 3, 3],
    'Specialized_or_General_or_Basic': ['General', 'General', 'Basic', 'Specialized', 'Specialized'],
    'Instructor': ['Hajari', 'Salimi', 'Ahmadi', 'Mohammadi', 'Bidki'],
    'Total_ECTS_in_Semester': [14, 17, 20, 14, 20],
    'Days_for_Final_Test': [2, 1, 1, 2, 0],
    'Nth_Test_of_Semester': [3, 2, 3, 3, 1],
    'Homework': ['Yes', 'Yes', 'Yes', 'Yes', 'Yes'],
    'Quiz': ['No', 'Yes', 'Yes', 'Yes', 'Yes'],
    'Passed': ['Yes', 'Yes', 'Yes', 'Yes', 'Yes'],
    'Conditionally_Passed': ['No', 'No', 'No', 'No', 'No'],
    'Grade': ['A', 'A', 'A', 'A', 'B']
}
```

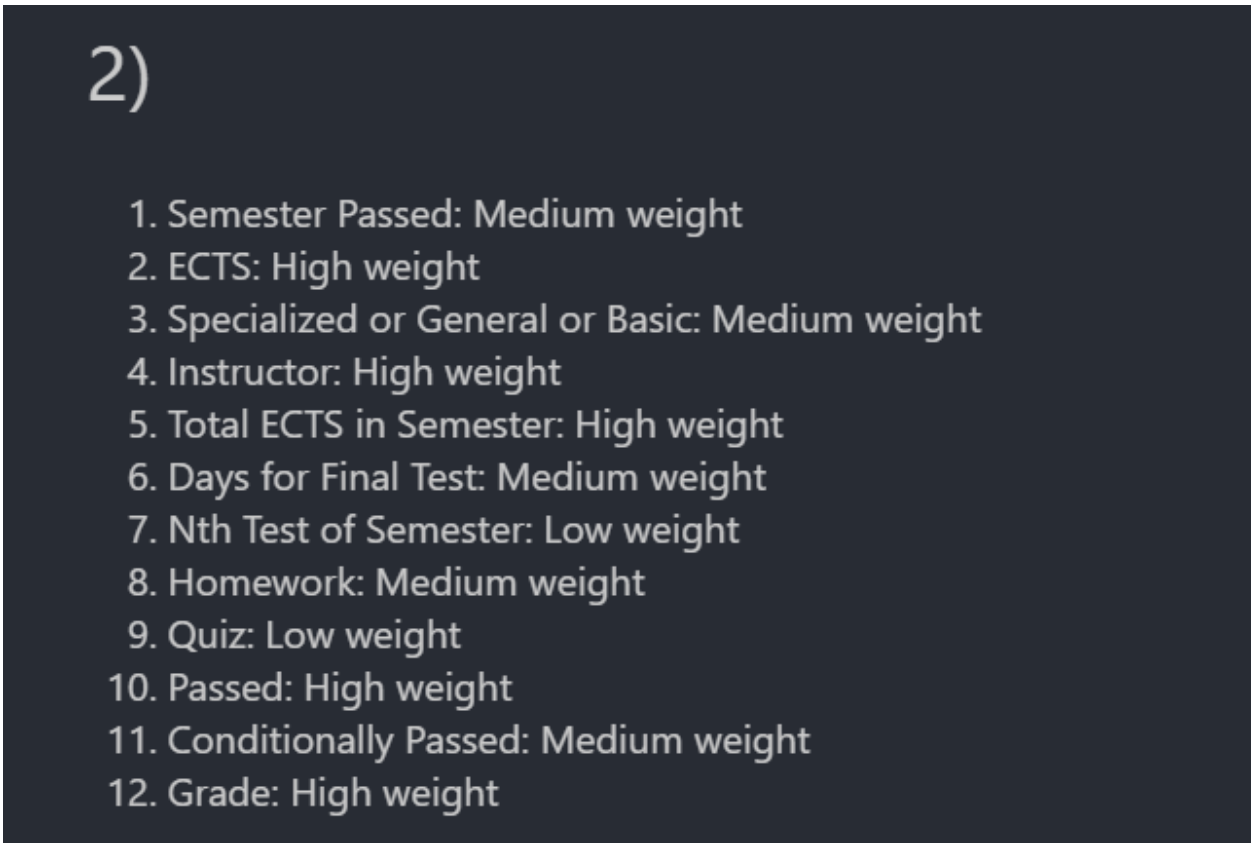
[22] ✓ 0.0s

سپس نوع هر ویژگی را مشخص می کنیم:

1)

1. Semester_Passed: Binary
2. ECTS: Numeric
3. Specialized_or_General_or_Basic: Ordianal
4. Instructor: Nominal
5. Total_ECTS_in_Semester: Numeric
6. Days_for_Final_Test: Numeric
7. Nth_Test_of_Semester: Numeric
8. Homework: Binary
9. Quiz: Binary
10. Passed: Binary
11. Conditionally_Passed: Binary
12. Grade: Ordinal

در مرحله بعد وزن هر ویژگی را مشخص می کنیم:



1. ترم در پاس کردن درس اهمیت کمی دارد
2. واحد اهمیت دوره را نشان می دهد
3. درس های اختصاصی مهم تر هستند ولی به طور کلی همه درس ها دارای اهمیت هستند
4. استاد درس می تواند به طور قابل توجهی تجربه یادگیری و نتایج یک درس را تحت تاثیر قرار دهد
5. تعداد کل واحد ها دید کلی از بار کاری و پیچیدگی ترم به عنوان یک کل ارائه می‌دهد
6. درس ها باید به طور کلی در طول ترم خوانده شود
7. امتحان چندم بودن اهمیتی ندارد
8. تکالیف می‌توانند به درک و تمرین کمک کنند
9. آزمون‌ها ممکن است توانایی درک مباحث خاص را ارزیابی کنند و به عملکرد کلی کمک کنند
10. پاس شدن درس مهم است
11. مشروطی فرق زیادی با پاس نشدن ندارد بنابراین اهمیتی ندارد
12. نمره نشان دهنده عملکرد در طول دوره است

در مرحله داده های غیر باینری و غیر عددی را مپ می کنیم:

```python
# Convert data to numpy array
num_data = len(data['Lesson'])
numeric_data = np.zeros((num_data, 5))  # 5 numeric attributes
numeric_data[:, 0] = data['ECTS']
numeric_data[:, 1] = data['Total_ECTS_in_Semester']
numeric_data[:, 2] = data['Days_for_Final_Test']
numeric_data[:, 3] = data['Nth_Test_of_Semester']

# Convert categorical data to numeric
instructor_mapping = {instructor: idx for idx, instructor in enumerate(set(data['Instructor']))}
semester_mapping = {'Odd': 0, 'Even': 1}
specialized_mapping = {'General': 0, 'Specialized': 2, 'Basic' : 1}
for i, lesson in enumerate(data['Lesson']):
    numeric_data[i, 4] = instructor_mapping[data['Instructor'][i]]
```

نمای دیتافریم بعد از تغییرات:

```
...                              Lesson Semester_Passed  ECTS  \
0                 English Language 2            Even     2
1                          Physics 2             Odd     3
2               Discrete Mathematics            Even     3
3  Electric and Electronic Circuits             Even     3
4                    Database Design             Odd     3

  Specialized_or_General_or_Basic Instructor  Total_ECTS_in_Semester  \
0                         General     Hajari                      14
1                         General     Salimi                      17
2                           Basic     Ahmadi                      20
3                     Specialized  Mohammadi                      14
4                     Specialized      Bidki                      20

   Days_for_Final_Test  Nth_Test_of_Semester Homework Quiz Passed  \
0                    2                     3      Yes   No    Yes
1                    1                     2      Yes  Yes    Yes
2                    1                     3      Yes  Yes    Yes
3                    2                     3      Yes  Yes    Yes
4                    0                     1      Yes  Yes    Yes

  Conditionally_Passed Grade
0                   No     A
1                   No     A
2                   No     A
3                   No     A
4                   No     B
```

سپس با توجه به وزن و ماتریس فاصله، ماتریس فاصله وزن دار را ترسیم می کنیم:

```python
weights = np.array([0.1, 0.5, 0.3, 0.5, 0.5, 0.3, 0.1, 0.3, 0.1, 0.5, 0.3, 0.5])

# Calculate weighted distance matrix for each attribute
weighted_dist_matrices = {}
for key, matrix in dist_matrices.items():
    weighted_dist_matrices[key] = matrix * weights[key]

# Print total weighted distance matrix for each attribute
for key, matrix in weighted_dist_matrices.items():
    print(f"Total Weighted Distance Matrix for Attribute {key+1}:")
    print(matrix)
    print()
```

نرمال سازی و محاسبه فاصله ها:

نرمال سازی : Min-Max Normalization

محاسبه فاصله بین داده ها : Euclidean Distance

در بخش دوم ابتدا 20 درس مورد نظر با ویژگی های مورد نظر را اضافه می کنیم و سپس آن را به دیتافریم تبدیل می کنیم:

```python
# Create a dictionary to hold the data
data = {
    'Lesson': ['English Language 2', 'Physics 2', 'Discrete Mathematics', 'Electric and Electronic Circuits', 'Database Design', 'andishe 1', 'zaban omomi',
    'tarbiat badani 2', 'zaban farsi', 'riazi omomi 2', 'mabani computer', 'moadelat difransiel', 'zaban takhasosi', 'kargah omomi', 'madar haye elektriki',
    'moadelat', 'az fizik 2', 'amar o ehtemal', 'jabr khati', 'madar manteghi', 'memari computer', 'signals and systems', 'mohandesi narmafzar', 'system
    haye amel', 'ravesh haye pazhohesh'],
    'Semester_Passed': ['Even', 'Odd', 'Even', 'Even', 'Odd', 'Odd','Odd','Even','Even','Odd','Odd','Odd','Even','Even','Even','Even','Odd','Odd','Odd',
    'Odd','Even','Even','Odd','Odd','Even'],
    'ECTS': [2, 3, 3, 3, 3, 2, 3, 1, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 3, 3, 3, 3, 3],
    'Specialized_or_General_or_Basic': ['General', 'General', 'Basic', 'Specialized', 'Specialized', 'General', 'General', 'General', 'General', 'General',
    'Specialized', 'Specialized', 'Specialized', 'Specialized', 'Specialized', 'Specialized', 'Specialized', 'Specialized', 'Specialized', 'Specialized',
    'Specialized', 'Specialized', 'Specialized', 'Specialized', 'Specialized' ],
    'Instructor': ['Hajari', 'Salimi', 'Ahmadi', 'Mohammadi', 'Bidki', 'sajadi', 'Hajari', 'fotohi', 'Salimi', 'motlagh', 'ranjbar', 'motlagh', 'mahmoodi',
    'ranjbar', 'mahmoodi', 'sotoodeh', 'safi nia', 'motlagh', 'joozi', 'bidki', 'ahmadi', 'mohammadi', 'ahmadi', 'ahmadi', 'ranjbar'],
    'Total_ECTS_in_Semester': [14, 17, 20, 14, 20, 5, 5, 14, 14, 17, 17, 17, 20, 20, 20, 20, 16, 16, 16, 16, 14, 14, 20, 20, 20],
    'Days_for_Final_Test': [2, 1, 1, 2, 0, 1, 0, 0, 2, 2, 2, 1, 4, 1, 0, 2, 0, 2, 2, 1, 0, 1, 2, 2, 2],
    'Nth_Test_of_Semester': [3, 2, 3, 3, 1, 2, 1, 0, 3, 2, 3, 4, 2, 3, 1, 4, 1, 2, 3, 4, 1, 2, 2, 3, 3],
    'Homework': ['Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'No', 'Yes', 'No', 'No', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes',
    'Yes', 'Yes', 'Yes', 'Yes', 'Yes'],
    'Quiz': ['No', 'Yes', 'Yes', 'Yes', 'Yes', 'No', 'Yes', 'No', 'No', 'Yes', 'Yes', 'Yes', 'No', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes',
    'Yes', 'Yes', 'Yes', 'Yes'],
    'Passed': ['Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'No', 'Yes', 'Yes', 'No', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes',
    'Yes', 'Yes', 'Yes', 'Yes', 'Yes'],
    'Conditionally_Passed': ['No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'Yes', 'Yes',
    'No', 'No', 'No', 'No', 'No'],
    'Grade': ['A', 'A', 'A', 'A', 'B', 'A', 'B', 'A', 'A', 'C', 'B', 'C', 'A', 'B', 'C', 'B', 'B', 'C', 'B', 'C', 'A', 'B', 'B', 'B', 'A']
}
```

[9] ✓ 0.0s Python

```python
df = pd.DataFrame(data)
print(df)
```

[10] ✓ 0.0s Python

باکس پلات (Box plot) یک نمودار آماری است که برای نمایش توزیع متغیرهای عددی استفاده می‌شود. باکس پلات برای مقایسه توزیع متغیرها و تشخیص داده‌های پرت (outlier) مفید است. همچنین، این نمودار به سادگی میانه را نشان می‌دهد که از معنی‌دارترین ویژگی‌های توزیع داده‌ها هستند. از باکس پلات معمولاً برای مشاهده انحرافات و شکل توزیع داده‌ها استفاده می شود.

```
# Extract numeric attributes
numeric_attributes = ['ECTS', 'Total_ECTS_in_Semester', 'Days_for_Final_Test', 'Nth_Test_of_Semester']

# Create box plots for each numeric attribute
for attribute in numeric_attributes:
    plt.figure(figsize=(8, 6))
    plt.boxplot(data[attribute])
    plt.title(f'Box Plot of {attribute}')
    plt.ylabel('Value')
    plt.show()
```

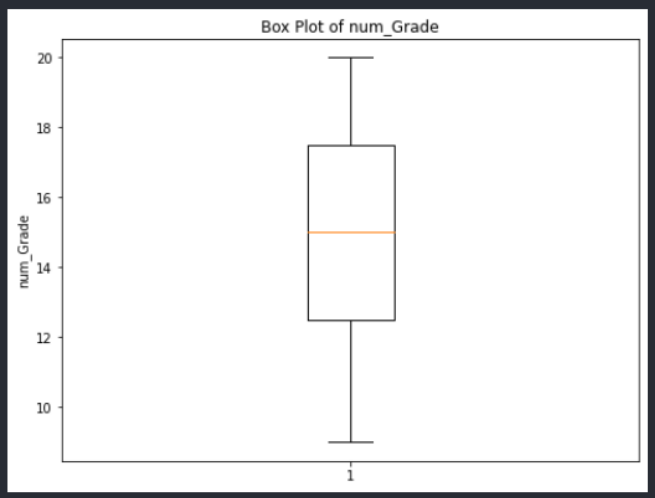
سپس ستون نمره های عددی را به دیتا فریم اضافه می کنیم:

```
# Given num_Grade values
num_Grade_values = ['19.83', '17.50', '17.10', '16', '12.50', '17.50', '15', '20', '19', '10', '12.94', '9', '20', '13', '9.9', '13.60', '15', '10', '15',
'10', '16.20', '13.50', '15', '11.65', '19.33']

# Add num_Grade to the data dictionary
data['num_Grade'] = num_Grade_values
```

سپس دوباره باکس پلات را برای این ستون رسم می کنیم:

```
# Convert num_Grade values to float
num_Grade_values = [float(value) for value in data['num_Grade']]

# Create a box plot for num_Grade
plt.figure(figsize=(8, 6))
plt.boxplot(num_Grade_values)
plt.title('Box Plot of num_Grade')
plt.ylabel('num_Grade')
plt.show()
```

[13] ✓ 0.1s



اسکاتر پلات (Scatter plot) یک نمودار است که برای نمایش ارتباط بین دو متغیر عددی استفاده می‌شود. در این نمودار، هر نقطه نمایانگر یک نمونه داده است و موقعیت آن نقطه بر اساس مقادیر دو متغیر (معمولاً یک متغیر برای هر محور) تعیین می‌شود.

از اسکاتر پلات برای تشخیص الگوها، روابط و وجود پرت‌ها (outliers) در داده‌ها استفاده می‌شود.

```python
# Extract numeric attributes
numeric_attributes = ['ECTS', 'Total_ECTS_in_Semester', 'Days_for_Final_Test', 'Nth_Test_of_Semester', 'num_Grade']

# Create scatter plots for pairs of numeric attributes
for i in range(len(numeric_attributes)):
    for j in range(i+1, len(numeric_attributes)):
        plt.figure(figsize=(8, 6))
        plt.scatter(data[numeric_attributes[i]], data[numeric_attributes[j]])
        plt.title(f'Scatter Plot of {numeric_attributes[i]} vs {numeric_attributes[j]}')
        plt.xlabel(numeric_attributes[i])
        plt.ylabel(numeric_attributes[j])
        plt.grid(True)
        plt.show()
```

✓ 1.2s