In [62]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import datetime as dt
```

In [63]:

```
df = pd.read_csv(r"C:\Users\ak973\Downloads\Algerian_forest_fires_dataset_UPDATE.csv",skiprows=1)
```

In [64]:

```
df
```

Out[64]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes
0	01	06	2012	29	57	18	0	65.7	3.4	7.6	1.3	3.4	0.5	not fire
1	02	06	2012	29	61	13	1.3	64.4	4.1	7.6	1	3.9	0.4	not fire
2	03	06	2012	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	not fire
3	04	06	2012	25	89	13	2.5	28.6	1.3	6.9	0	1.7	0	not fire
4	05	06	2012	27	77	16	0	64.8	3	14.2	1.2	3.9	0.5	not fire
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
241	26	09	2012	30	65	14	0	85.4	16	44.5	4.5	16.9	6.5	fire
242	27	09	2012	28	87	15	4.4	41.1	6.5	8	0.1	6.2	0	not fire
243	28	09	2012	27	87	29	0.5	45.9	3.5	7.9	0.4	3.4	0.2	not fire
244	29	09	2012	24	54	18	0.1	79.7	4.3	15.2	1.7	5.1	0.7	not fire
245	30	09	2012	24	64	15	0.2	67.3	3.8	16.5	1.2	4.8	0.5	not fire

246 rows × 14 columns

In [65]:

```
df.drop([122,123], axis=0, inplace=True )
```

In [66]:

```
df.sort_index(axis=0,  ascending=True, inplace=True)
```

In [67]:

```
df.head()
```

Out[67]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes
0	01	06	2012	29	57	18	0	65.7	3.4	7.6	1.3	3.4	0.5	not fire
1	02	06	2012	29	61	13	1.3	64.4	4.1	7.6	1	3.9	0.4	not fire
2	03	06	2012	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	not fire
3	04	06	2012	25	89	13	2.5	28.6	1.3	6.9	0	1.7	0	not fire
4	05	06	2012	27	77	16	0	64.8	3	14.2	1.2	3.9	0.5	not fire

In [68]:

```
df['DateTime'] = df['day'] +"/" + df['month']+ "/"  + df['year']
```

In [69]:

```
df['DateTime']
```

Out[69]:

```
0      01/06/2012
1      02/06/2012
2      03/06/2012
3      04/06/2012
4      05/06/2012
          ...
241    26/09/2012
242    27/09/2012
243    28/09/2012
244    29/09/2012
245    30/09/2012
Name: DateTime, Length: 244, dtype: object
```

In [70]:

```
df.drop(['day'],axis=1, inplace = True)
df.drop(['month'],axis=1, inplace = True)
df.drop(['year'],axis=1, inplace = True)
#df.drop(df['day'], df['month']) #can be done in this way too
```

In [71]:

```
df
```

Out[71]:

	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	DateTime
0	29	57	18	0	65.7	3.4	7.6	1.3	3.4	0.5	not fire	01/06/2012
1	29	61	13	1.3	64.4	4.1	7.6	1	3.9	0.4	not fire	02/06/2012
2	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	not fire	03/06/2012
3	25	89	13	2.5	28.6	1.3	6.9	0	1.7	0	not fire	04/06/2012
4	27	77	16	0	64.8	3	14.2	1.2	3.9	0.5	not fire	05/06/2012
...	...	...	...	...	...	...	...	...	...	...	...	...
241	30	65	14	0	85.4	16	44.5	4.5	16.9	6.5	fire	26/09/2012
242	28	87	15	4.4	41.1	6.5	8	0.1	6.2	0	not fire	27/09/2012
243	27	87	29	0.5	45.9	3.5	7.9	0.4	3.4	0.2	not fire	28/09/2012
244	24	54	18	0.1	79.7	4.3	15.2	1.7	5.1	0.7	not fire	29/09/2012
245	24	64	15	0.2	67.3	3.8	16.5	1.2	4.8	0.5	not fire	30/09/2012

244 rows × 12 columns

checking for null values

In [72]:

```
df.columns
```

Out[72]:

```
Index(['Temperature', ' RH', ' Ws', 'Rain ', 'FFMC', 'DMC', 'DC', 'ISI', 'BUI',
       'FWI', 'Classes  ', 'DateTime'],
      dtype='object')
```

Check null values

In [73]:

```
df.isnull().sum()
```

Out[73]:

```
Temperature    0
 RH            0
 Ws            0
Rain           0
FFMC           0
DMC            0
DC             0
ISI            0
BUI            0
FWI            0
Classes        1
DateTime       0
dtype: int64
```

In [74]:

```
#we dropped the nan value since the va;ue was only one missing, if there were multiple we could've used mode or median to fill in
df.dropna()
```

Out[74]:

	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	DateTime
0	29	57	18	0	65.7	3.4	7.6	1.3	3.4	0.5	not fire	01/06/2012
1	29	61	13	1.3	64.4	4.1	7.6	1	3.9	0.4	not fire	02/06/2012
2	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	not fire	03/06/2012
3	25	89	13	2.5	28.6	1.3	6.9	0	1.7	0	not fire	04/06/2012
4	27	77	16	0	64.8	3	14.2	1.2	3.9	0.5	not fire	05/06/2012
...	...	...	...	...	...	...	...	...	...	...	...	...
241	30	65	14	0	85.4	16	44.5	4.5	16.9	6.5	fire	26/09/2012
242	28	87	15	4.4	41.1	6.5	8	0.1	6.2	0	not fire	27/09/2012
243	27	87	29	0.5	45.9	3.5	7.9	0.4	3.4	0.2	not fire	28/09/2012
244	24	54	18	0.1	79.7	4.3	15.2	1.7	5.1	0.7	not fire	29/09/2012
245	24	64	15	0.2	67.3	3.8	16.5	1.2	4.8	0.5	not fire	30/09/2012

243 rows × 12 columns

We are renaming the df column or basically encoding it

In [ ]:

```
df.info()
```

In [ ]:

```
df.loc[0:122, 'region'] = 0 #0 is bejaia
df.loc[122: , 'region'] = 1 #1 is sidi Bel Abbes
```

In [76]:

```
df
```

Out[76]:

	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	DateTime	region
0	29	57	18	0	65.7	3.4	7.6	1.3	3.4	0.5	not fire	01/06/2012	0.0
1	29	61	13	1.3	64.4	4.1	7.6	1	3.9	0.4	not fire	02/06/2012	0.0

	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	DateTime	region
2	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	not fire	03/06/2012	0.0
3	25	89	13	2.5	28.6	1.3	6.9	0	1.7	0	not fire	04/06/2012	0.0
4	27	77	16	0	64.8	3	14.2	1.2	3.9	0.5	not fire	05/06/2012	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
241	30	65	14	0	85.4	16	44.5	4.5	16.9	6.5	fire	26/09/2012	1.0
242	28	87	15	4.4	41.1	6.5	8	0.1	6.2	0	not fire	27/09/2012	1.0
243	27	87	29	0.5	45.9	3.5	7.9	0.4	3.4	0.2	not fire	28/09/2012	1.0
244	24	54	18	0.1	79.7	4.3	15.2	1.7	5.1	0.7	not fire	29/09/2012	1.0
245	24	64	15	0.2	67.3	3.8	16.5	1.2	4.8	0.5	not fire	30/09/2012	1.0

244 rows × 13 columns

In [77]:

```
df.columns
```

Out[77]:

```
Index(['Temperature', ' RH', ' Ws', 'Rain ', 'FFMC', 'DMC', 'DC', 'ISI', 'BUI',
       'FWI', 'Classes  ', 'DateTime', 'region'],
      dtype='object')
```

In [78]:

```
df['DMC'] = df.DMC.astype(float)
df['FFMC'] = df.FFMC.astype(float)
df['ISI'] = df.ISI.astype(float)
df['BUI'] = df.BUI.astype(float)
df['Temperature'] = df.Temperature.astype(int)
```

In [79]:

```
df.dtypes
```

Out[79]:

```
Temperature      int32
 RH             object
 Ws             object
Rain            object
FFMC           float64
DMC            float64
DC              object
ISI            float64
BUI            float64
FWI             object
Classes         object
DateTime        object
region         float64
dtype: object
```

In [80]:

```
df['DC'].str.strip('')
```

Out[80]:

```
0       7.6
1       7.6
2       7.1
3       6.9
4      14.2
       ...
241    44.5
242       8
243     7.9
244    15.2
```

```
245    16.5
Name: DC, Length: 244, dtype: object
```

In [81]:

```
num_col = [feature for feature in df.columns if df[feature].dtypes!='object']
num_col
```

Out[81]:

```
['Temperature', 'FFMC', 'DMC', 'ISI', 'BUI', 'region']
```

In [82]:

```
#could not convert some of the columns, hence resulting empty list
cat_col = [feature for feature in df.columns if df[feature].dtypes==['float','int']]
cat_col
```

Out[82]:

```
[]
```

In [83]:

```
df.shape
```

Out[83]:

```
(244, 13)
```

Univariate analysis

In [85]:

```
df.info()
```

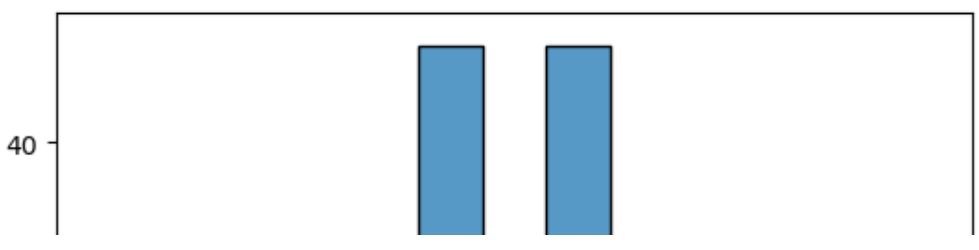
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 244 entries, 0 to 245
Data columns (total 13 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Temperature  244 non-null    int32  
 1    RH          244 non-null    object 
 2    Ws          244 non-null    object 
 3   Rain         244 non-null    object 
 4   FFMC         244 non-null    float64
 5   DMC          244 non-null    float64
 6   DC           244 non-null    object 
 7   ISI          244 non-null    float64
 8   BUI          244 non-null    float64
 9   FWI          244 non-null    object 
 10  Classes      243 non-null    object 
 11  DateTime     244 non-null    object 
 12  region       244 non-null    float64
dtypes: float64(5), int32(1), object(7)
memory usage: 33.8+ KB
```

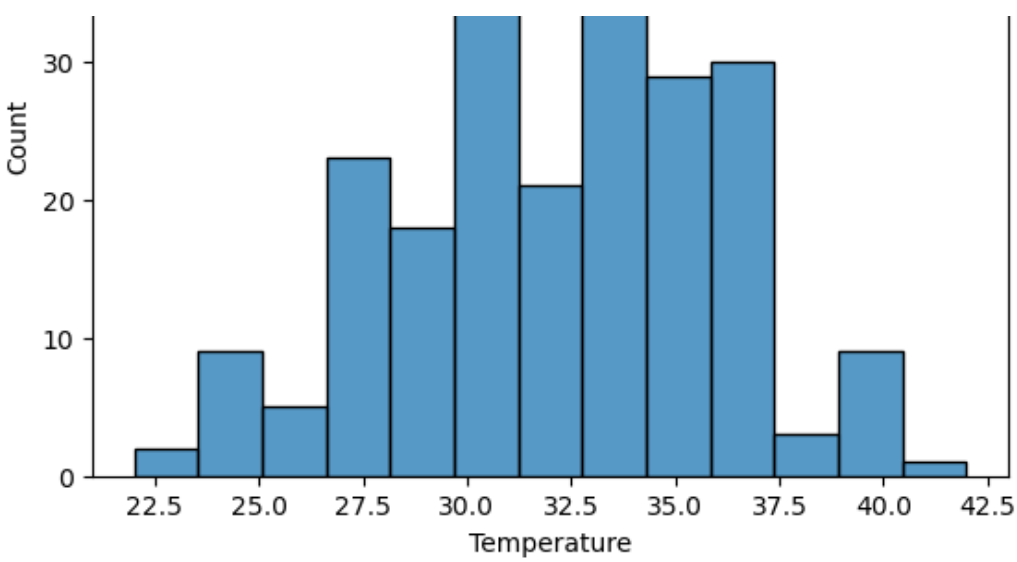
In [86]:

```
sns.histplot(df['Temperature'])
```

Out[86]:

```
<AxesSubplot:xlabel='Temperature', ylabel='Count'>
```





OBSERVATIONS

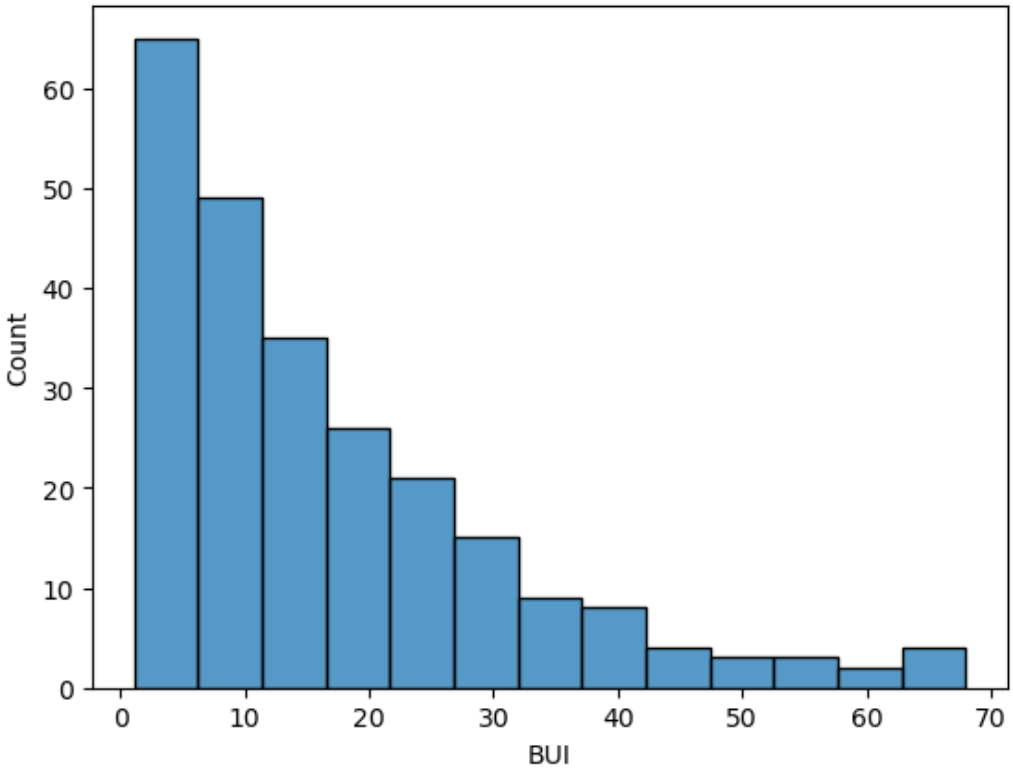
1. **we can see that the highest temperature is 42.5**
2. **maximum amount of occurences is 30 and 32.5**
3. **also observe that 42.5 has the least weightage in the plot**
4. **35 and 37.5 degree has the second highest weightage**

In [91]:

```
sns.histplot(df['BUI'])
```

Out[91]:

```
<AxesSubplot:xlabel='BUI', ylabel='Count'>
```



BUI stands for buildup index and basically indicates fire builup

1. **looking at the curve one can say its right skewed or positively skewed**
2. **Between the range of 0 to 5 approximately the build up is huge**
3. **between the range of 60 to 70 its low**

BIVARIATE ANALYSIS

but first lets check if we have any outliers

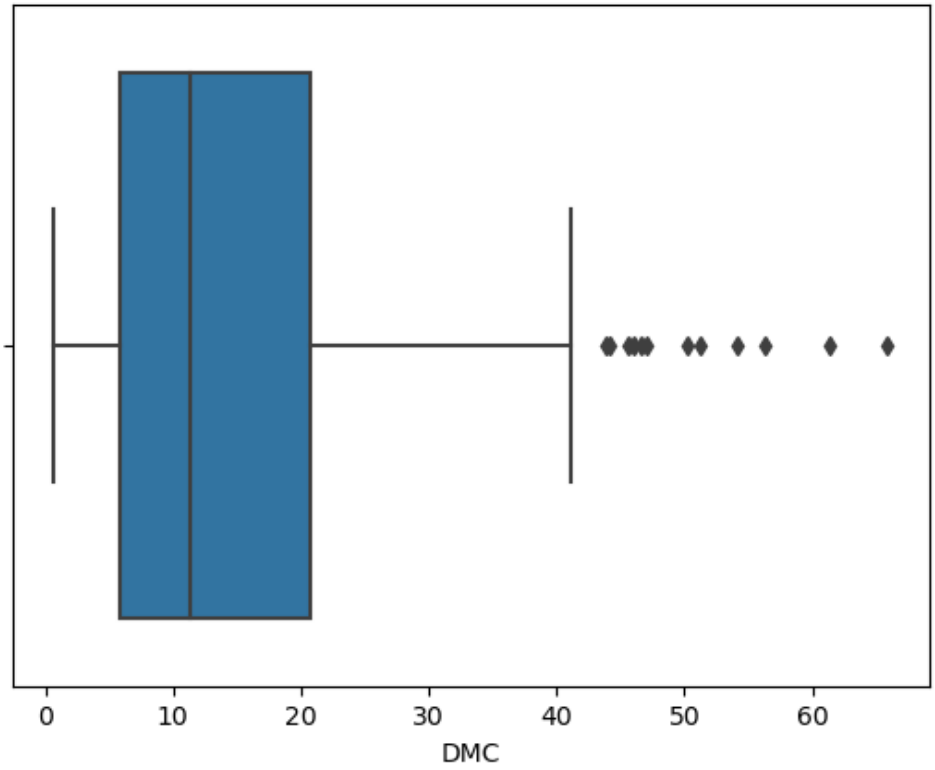
In [96]:

```
#we are checking for outliers in a particular column called DMC
sns.boxplot(df['DMC'], orient = 'v')
```

```
C:\Users\ak973\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass
the following variable as a keyword arg: x. From version 0.12, the only valid positional
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
C:\Users\ak973\anaconda3\lib\site-packages\seaborn\_core.py:1326: UserWarning: Vertical o
rientation ignored with only `x` specified.
  warnings.warn(single_var_warning.format("Vertical", "x"))
```

Out[96]:

```
<AxesSubplot:xlabel='DMC'>
```



OBSERVATION

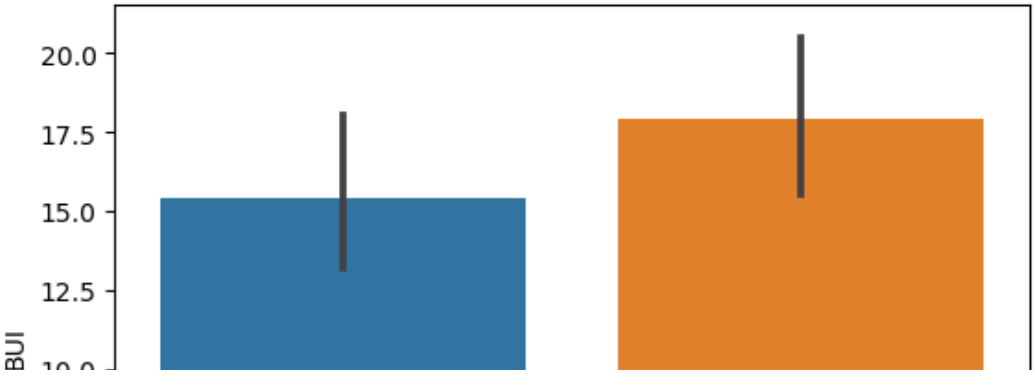
- **we can easily see the outliers on the right end of the plot**

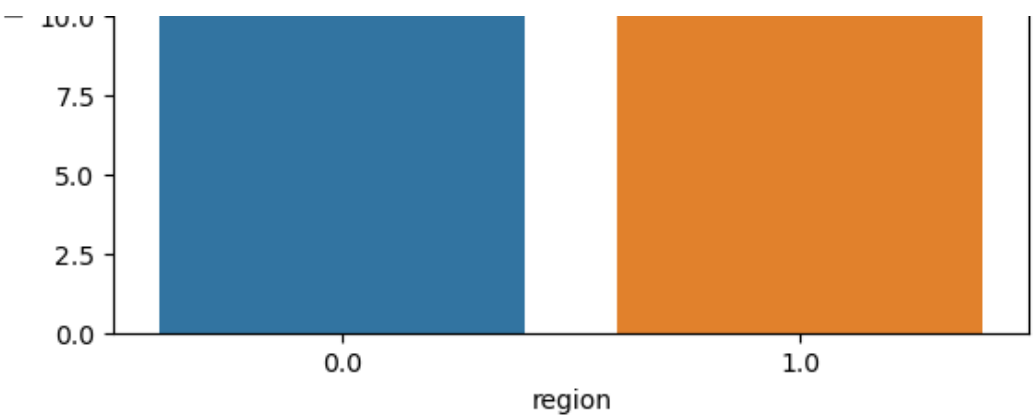
In [99]:

```
sns.barplot(x='region',y='BUI', data = df)
```

Out[99]:

```
<AxesSubplot:xlabel='region', ylabel='BUI'>
```





OBSERVATIONS

- **1.0 has much more build up index compared to 0**
- **highest BUI is around 17.5 on an average**
- **lowest BUI is somewhere around 15.2 or 15.3**

In [ ]:

```
```

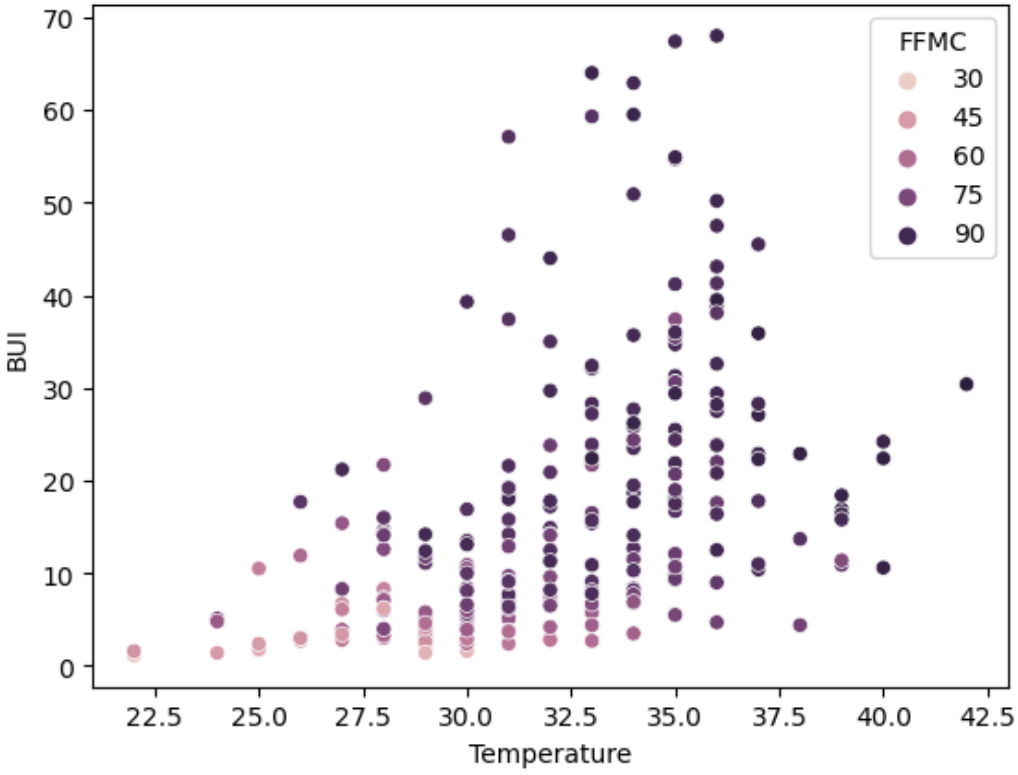
MULTIVARIATE ANALYSIS

In [120]:

```
sns.scatterplot(x='Temperature', y='BUI', hue='FFMC', data = df)
```

Out[120]:

```
<AxesSubplot:xlabel='Temperature', ylabel='BUI'>
```



OBSERVATIONS

- **as and when temperature rises BUI rises**
- **as and when temperature rises even FFMC rises(ffmc is moisture code index)**
- **20 being lowest FFMC one can see temperature is also at par low with it**

In [ ]: