EDA - Algerian Forest Fire Dataset

In [1]:

```
## importing required packages
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

%matplotlib inline
```

In [22]:

```
## load the dataset
data = pd.read_csv('Algerian_forest_fires_dataset_UPDATE.csv',skiprows=1)
```

In [24]:

```
data.head()
```

Out[24]:

	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 [25]:

```
data.iloc[120:130]
```

Out[25]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes
120	29	09	2012	26	80	16	1.8	47.4	2.9	7.7	0.3	3	0.1	not fire
121	30	09	2012	25	78	14	1.4	45	1.9	7.5	0.2	2.4	0.1	not fire
122	Sidi-Bel Abbes Region Dataset	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
123	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes
124	01	06	2012	32	71	12	0.7	57.1	2.5	8.2	0.6	2.8	0.2	not fire
125	02	06	2012	30	73	13	4	55.7	2.7	7.8	0.6	2.9	0.2	not fire
126	03	06	2012	29	80	14	2	48.7	2.2	7.6	0.3	2.6	0.1	not fire
127	04	06	2012	30	64	14	0	79.4	5.2	15.4	2.2	5.6	1	not fire
128	05	06	2012	32	60	14	0.2	77.1	6	17.6	1.8	6.5	0.9	not fire
129	06	06	2012	35	54	11	0.1	83.7	8.4	26.3	3.1	9.3	3.1	fire

In [26]:

```
data['Region'] = 'Bejaia'
```

In [27]:

```
data.head(5)
```

Out[27]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region
0	01	06	2012	29	57	18	0	65.7	3.4	7.6	1.3	3.4	0.5	not fire	Bejaia
1	02	06	2012	29	61	13	1.3	64.4	4.1	7.6	1	3.9	0.4	not fire	Bejaia

2	03	06	2012	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	not fire	Bejaia
3	04	06	2012	25	89	13	2.5	28.6	1.3	6.9	0	1.7	0	not fire	Bejaia
4	05	06	2012	27	77	16	0	64.8	3	14.2	1.2	3.9	0.5	not fire	Bejaia

In [31]:

```
data.loc[124:,'Region'] = 'Sidi-Bel Addes'
```

In [37]:

```
data.loc[:10]
```

Out[37]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region
0	01	06	2012	29	57	18	0	65.7	3.4	7.6	1.3	3.4	0.5	not fire	Bejaia
1	02	06	2012	29	61	13	1.3	64.4	4.1	7.6	1	3.9	0.4	not fire	Bejaia
2	03	06	2012	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	not fire	Bejaia
3	04	06	2012	25	89	13	2.5	28.6	1.3	6.9	0	1.7	0	not fire	Bejaia
4	05	06	2012	27	77	16	0	64.8	3	14.2	1.2	3.9	0.5	not fire	Bejaia
5	06	06	2012	31	67	14	0	82.6	5.8	22.2	3.1	7	2.5	fire	Bejaia
6	07	06	2012	33	54	13	0	88.2	9.9	30.5	6.4	10.9	7.2	fire	Bejaia
7	08	06	2012	30	73	15	0	86.6	12.1	38.3	5.6	13.5	7.1	fire	Bejaia
8	09	06	2012	25	88	13	0.2	52.9	7.9	38.8	0.4	10.5	0.3	not fire	Bejaia
9	10	06	2012	28	79	12	0	73.2	9.5	46.3	1.3	12.6	0.9	not fire	Bejaia
10	11	06	2012	31	65	14	0	84.5	12.5	54.3	4	15.8	5.6	fire	Bejaia

In [35]:

```
data.drop([122,123],inplace=True)
```

In [41]:

```
## saving the final dataframe
data.to_csv('Final-Algerian-ForestFire-Dataset.csv',index=False)
```

In [44]:

```
data.shape
```

Out[44]:

```
(244, 15)
```

In [45]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 244 entries, 0 to 245
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   day          244 non-null    object
 1   month        244 non-null    object
 2   year         244 non-null    object
 3   Temperature  244 non-null    object
 4    RH          244 non-null    object
 5    Ws          244 non-null    object
 6   Rain         244 non-null    object
 7   FFMC         244 non-null    object
 8   DMC          244 non-null    object
 9   DC           244 non-null    object
 10  ISI          244 non-null    object
 11  BUI          244 non-null    object
 12  FWI          244 non-null    object
 13  Classes      243 non-null    object
 14  Region       244 non-null    object
```

```
dtypes: object(15)
memory usage: 38.6+ KB
```

Attribute Information:

1. Date : (DD/MM/YYYY) Day, month ('june' to 'september'), year (2012) Weather data observations
2. Temp : temperature noon (temperature max) in Celsius degrees: 22 to 42
3. RH : Relative Humidity in %: 21 to 90
4. Ws :Wind speed in km/h: 6 to 29
5. Rain: total day in mm: 0 to 16.8 FWI Components
6. Fine Fuel Moisture Code (FFMC) index from the FWI system: 28.6 to 92.5
7. Duff Moisture Code (DMC) index from the FWI system: 1.1 to 65.9
8. Drought Code (DC) index from the FWI system: 7 to 220.4
9. Initial Spread Index (ISI) index from the FWI system: 0 to 18.5
10. Buildup Index (BUI) index from the FWI system: 1.1 to 68
11. Fire Weather Index (FWI) Index: 0 to 31.1
12. Classes: two classes, namely 'Fire' and 'not Fire'

From the above information we can conclude that there is olny two categorical variables, ie. Fire and Region

In [64]:
```
data[data['Classes  '].isnull()]
```

Out[64]:

day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region

In [63]:
```
data.dropna(inplace=True)
```

In [66]:
```
data.isnull().sum()
```

Out[66]:
```
day            0
month          0
year           0
Temperature    0
 RH            0
 Ws            0
Rain           0
FFMC           0
DMC            0
DC             0
ISI            0
BUI            0
FWI            0
Classes        0
Region         0
dtype: int64
```

In [105…
```
columns = ['day','month','year','Temperature','RH','Ws','Rain','FFMC','DMC','DC','ISI','
columns
```

Out[105]:
```
['day',
 'month',
 'year',
 'Temperature',
 'RH',
 'Ws',
 'Rain',
 'FFMC',
 'DMC',
 'DC',
 'ISI',
 'BUI',
 'FWI',
 'Classes',
 'Region']
```

In [98]:

```
data = pd.read_csv('Final-Algerian-ForestFire-Dataset.csv')
```

In [104…

```
data.rename(columns={'Classes  ':'Classes'},inplace=True)
```

In [106…

```
data.head()
```

Out[106]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region
0	1	6	2012	29	57	18	0.0	65.7	3.4	7.6	1.3	3.4	0.5	not fire	Bejaia
1	2	6	2012	29	61	13	1.3	64.4	4.1	7.6	1.0	3.9	0.4	not fire	Bejaia
2	3	6	2012	26	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	not fire	Bejaia
3	4	6	2012	25	89	13	2.5	28.6	1.3	6.9	0.0	1.7	0	not fire	Bejaia
4	5	6	2012	27	77	16	0.0	64.8	3.0	14.2	1.2	3.9	0.5	not fire	Bejaia

In [109…

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 244 entries, 0 to 243
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   day          244 non-null    int64  
 1   month        244 non-null    int64  
 2   year         244 non-null    int64  
 3   Temperature  244 non-null    int64  
 4    RH          244 non-null    int64  
 5    Ws          244 non-null    int64  
 6   Rain         244 non-null    float64
 7   FFMC         244 non-null    float64
 8   DMC          244 non-null    float64
 9   DC           244 non-null    object 
 10  ISI          244 non-null    float64
 11  BUI          244 non-null    float64
 12  FWI          244 non-null    object 
 13  Classes      243 non-null    object 
 14  Region       244 non-null    object 
dtypes: float64(5), int64(6), object(4)
memory usage: 28.7+ KB
```

In [112…

```
data['DC'].isnull().sum()
```

Out[112]:

```
0
```

In [117…

```
data[data['DC']=='14.6 9']
```

Out[117]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region

In [116…

```
data['DC'].replace(['14.6 9'],'14.69',inplace=True)
```

In [119…

```
data['DC']=data['DC'].astype(float)
```

In [120…

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 244 entries, 0 to 243
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   day          244 non-null    int64  
 1   month        244 non-null    int64  
 2   year         244 non-null    int64  
 3   Temperature  244 non-null    int64  
 4    RH          244 non-null    int64  
 5    Ws          244 non-null    int64  
 6   Rain         244 non-null    float64
 7   FFMC         244 non-null    float64
 8   DMC          244 non-null    float64
 9   DC           244 non-null    float64
 10  ISI          244 non-null    float64
 11  BUI          244 non-null    float64
 12  FWI          244 non-null    object 
 13  Classes      243 non-null    object 
 14  Region       244 non-null    object 
dtypes: float64(6), int64(6), object(3)
memory usage: 28.7+ KB
```

In [126…

```
data[data['FWI']=='fire   ']
```

Out[126]:

day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region

In [125…

```
data.drop(165,inplace=True)
```

In [128…

```
data['FWI']=data['FWI'].astype(float)
```

In [129…

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 243 entries, 0 to 243
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   day          243 non-null    int64  
 1   month        243 non-null    int64  
 2   year         243 non-null    int64  
 3   Temperature  243 non-null    int64  
 4    RH          243 non-null    int64  
 5    Ws          243 non-null    int64  
 6   Rain         243 non-null    float64
 7   FFMC         243 non-null    float64
 8   DMC          243 non-null    float64
 9   DC           243 non-null    float64
 10  ISI          243 non-null    float64
 11  BUI          243 non-null    float64
 12  FWI          243 non-null    float64
 13  Classes      243 non-null    object 
 14  Region       243 non-null    object 
dtypes: float64(7), int64(6), object(2)
memory usage: 30.4+ KB
```

In [132…

```
data['Classes'].replace(['fire   ','not fire   '],[1,0],inplace=True)
```

In [134…

```
data.tail()
```

Out[134]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region
239	26	9	2012	30	65	14	0.0	85.4	16.0	44.5	4.5	16.9	6.5	1	Sidi-Bel

															Addes
240	27	9	2012	28	87	15	4.4	41.1	6.5	8.0	0.1	6.2	0.0	0	Sidi-Bel Addes
241	28	9	2012	27	87	29	0.5	45.9	3.5	7.9	0.4	3.4	0.2	0	Sidi-Bel Addes
242	29	9	2012	24	54	18	0.1	79.7	4.3	15.2	1.7	5.1	0.7	0	Sidi-Bel Addes
243	30	9	2012	24	64	15	0.2	67.3	3.8	16.5	1.2	4.8	0.5	not fire	Sidi-Bel Addes

In [135…
```
data['Region'].replace(['Sidi-Bel Addes','Bejaia'],[1,0],inplace=True)
```

In [137…
```
data.head(1)
```

Out[137]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region
0	1	6	2012	29	57	18	0.0	65.7	3.4	7.6	1.3	3.4	0.5	0	0

In [138…
```
data.tail(1)
```

Out[138]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region
243	30	9	2012	24	64	15	0.2	67.3	3.8	16.5	1.2	4.8	0.5	not fire	1

In [141…
```
data['Classes'].unique()
```

Out[141]:
```
array([0, 1, 'fire', 'fire ', 'not fire', 'not fire ', 'not fire     ',
       'not fire    '], dtype=object)
```

In [142…
```
data['Classes'].replace(['fire','fire ','not fire','not fire ','not fire     ','not fire
```

In [143…
```
data['Classes'].unique()
```

Out[143]:
```
array([0, 1], dtype=int64)
```

In [144…
```
data['Classes']=data['Classes'].astype(int)
```

In [145…
```
data['Region']=data['Region'].astype(int)
```

In [146…
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 243 entries, 0 to 243
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   day          243 non-null    int64  
 1   month        243 non-null    int64  
 2   year         243 non-null    int64  
 3   Temperature  243 non-null    int64  
 4    RH          243 non-null    int64  
 5    Ws          243 non-null    int64  
 6   Rain         243 non-null    float64
 7   FFMC         243 non-null    float64
 8   DMC          243 non-null    float64
 9   DC           243 non-null    float64
 10  ISI          243 non-null    float64
 11  BUI          243 non-null    float64
 12  FWI          243 non-null    float64
 13  Classes      243 non-null    int32
 14  Region       243 non-null    int32
dtypes: float64(7), int32(2), int64(6)
memory usage: 28.5 KB
```

In [147…

```
data.isnull().sum()
```

Out[147]:

```
day            0
month          0
year           0
Temperature    0
 RH            0
 Ws            0
Rain           0
FFMC           0
DMC            0
DC             0
ISI            0
BUI            0
FWI            0
Classes        0
Region         0
dtype: int64
```

In [148…

```
data.corr()
```

Out[148]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	
day	1.000000	-0.000369	NaN	0.097227	-0.076034	0.047812	-0.112523	0.224956	0.491514	0.52
month	-0.000369	1.000000	NaN	-0.056781	-0.041252	-0.039880	0.034822	0.017030	0.067943	0.12
year	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
Temperature	0.097227	-0.056781	NaN	1.000000	-0.651400	-0.284510	-0.326492	0.676568	0.485687	0.37
RH	-0.076034	-0.041252	NaN	-0.651400	1.000000	0.244048	0.222356	-0.644873	-0.408519	-0.22
Ws	0.047812	-0.039880	NaN	-0.284510	0.244048	1.000000	0.171506	-0.166548	-0.000721	0.07
Rain	-0.112523	0.034822	NaN	-0.326492	0.222356	0.171506	1.000000	-0.543906	-0.288773	-0.29
FFMC	0.224956	0.017030	NaN	0.676568	-0.644873	-0.166548	-0.543906	1.000000	0.603608	0.50
DMC	0.491514	0.067943	NaN	0.485687	-0.408519	-0.000721	-0.288773	0.603608	1.000000	0.87
DC	0.527952	0.126511	NaN	0.376284	-0.226941	0.079135	-0.298023	0.507397	0.875925	1.00
ISI	0.180543	0.065608	NaN	0.603871	-0.686667	0.008532	-0.347484	0.740007	0.680454	0.50
BUI	0.517117	0.085073	NaN	0.459789	-0.353841	0.031438	-0.299852	0.592011	0.982248	0.94
FWI	0.350781	0.082639	NaN	0.566670	-0.580957	0.032368	-0.324422	0.691132	0.875864	0.73
Classes	0.202840	0.024004	NaN	0.516015	-0.432161	-0.069964	-0.379097	0.769492	0.585658	0.51
Region	0.000821	0.001857	NaN	0.269555	-0.402682	-0.181160	-0.040013	0.222241	0.192089	-0.07

In [150…

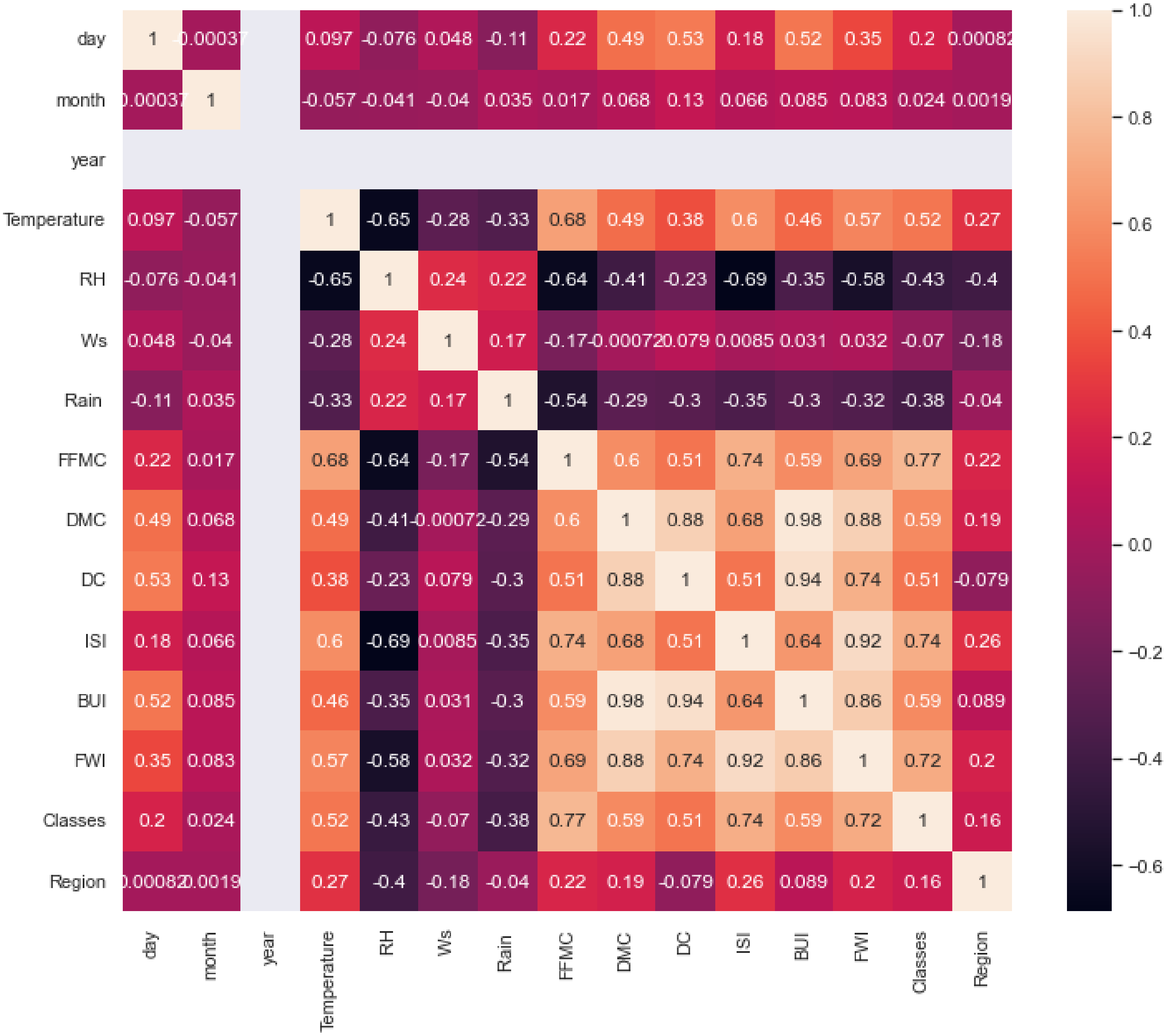
```
import seaborn as sns
```

In [154…

```
sns.set(rc={'figure.figsize':(12,10)})
sns.heatmap(data.corr(),annot=True)
```

Out[154]:

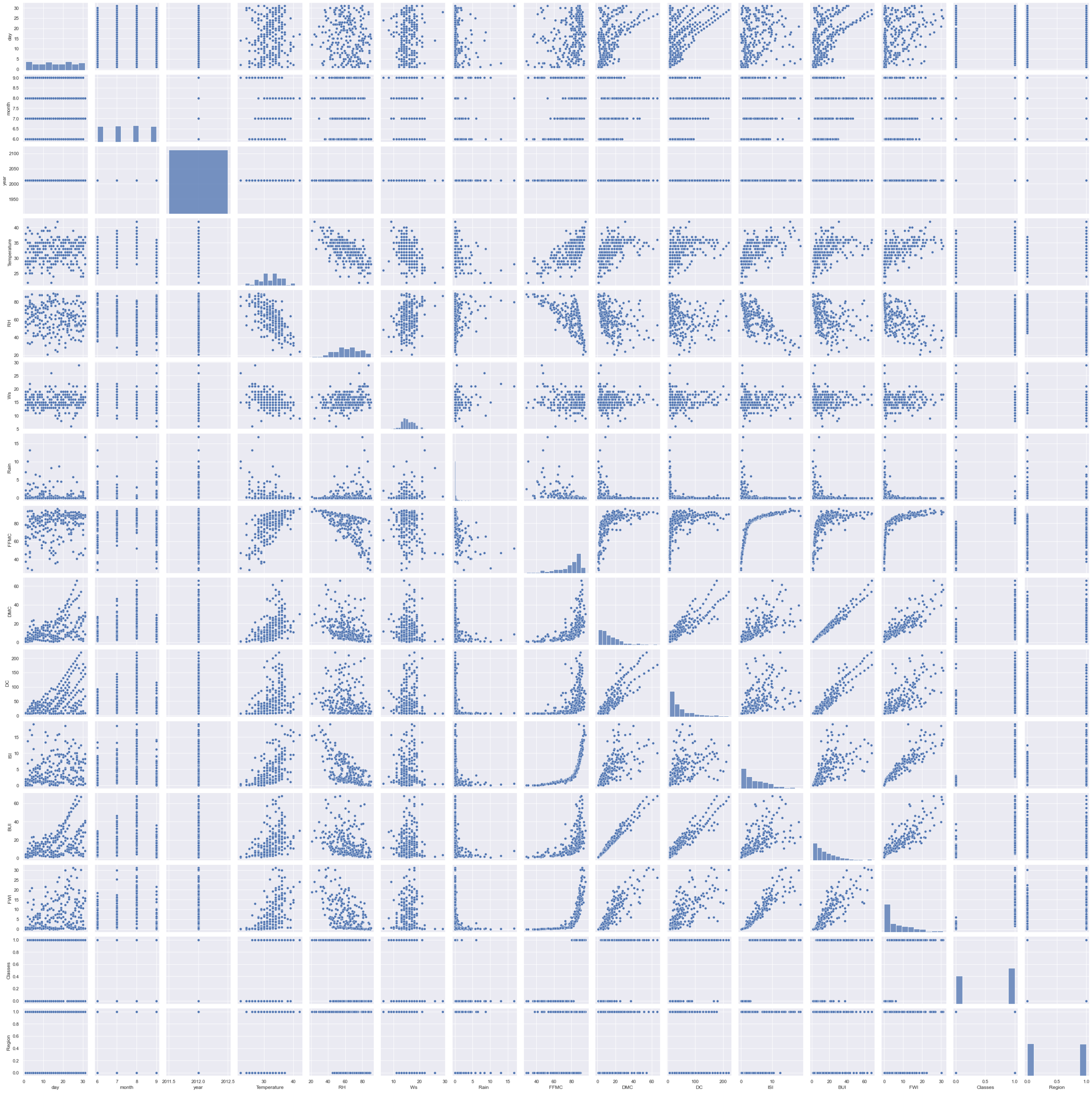
```
<AxesSubplot:>
```



	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region
day	1	-0.00037		0.097	-0.076	0.048	-0.11	0.22	0.49	0.53	0.18	0.52	0.35	0.2	0.00082
month	0.00037	1		-0.057	-0.041	-0.04	0.035	0.017	0.068	0.13	0.066	0.085	0.083	0.024	0.0019
year															
Temperature	0.097	-0.057		1	-0.65	-0.28	-0.33	0.68	0.49	0.38	0.6	0.46	0.57	0.52	0.27
RH	-0.076	-0.041		-0.65	1	0.24	0.22	-0.64	-0.41	-0.23	-0.69	-0.35	-0.58	-0.43	-0.4
Ws	0.048	-0.04		-0.28	0.24	1	0.17	-0.17	-0.00072	0.079	0.0085	0.031	0.032	-0.07	-0.18
Rain	-0.11	0.035		-0.33	0.22	0.17	1	-0.54	-0.29	-0.3	-0.35	-0.3	-0.32	-0.38	-0.04
FFMC	0.22	0.017		0.68	-0.64	-0.17	-0.54	1	0.6	0.51	0.74	0.59	0.69	0.77	0.22
DMC	0.49	0.068		0.49	-0.41	-0.00072	-0.29	0.6	1	0.88	0.68	0.98	0.88	0.59	0.19
DC	0.53	0.13		0.38	-0.23	0.079	-0.3	0.51	0.88	1	0.51	0.94	0.74	0.51	-0.079
ISI	0.18	0.066		0.6	-0.69	0.0085	-0.35	0.74	0.68	0.51	1	0.64	0.92	0.74	0.26
BUI	0.52	0.085		0.46	-0.35	0.031	-0.3	0.59	0.98	0.94	0.64	1	0.86	0.59	0.089
FWI	0.35	0.083		0.57	-0.58	0.032	-0.32	0.69	0.88	0.74	0.92	0.86	1	0.72	0.2
Classes	0.2	0.024		0.52	-0.43	-0.07	-0.38	0.77	0.59	0.51	0.74	0.59	0.72	1	0.16
Region	0.00082	0.0019		0.27	-0.4	-0.18	-0.04	0.22	0.19	-0.079	0.26	0.089	0.2	0.16	1

In [155…
```
data['year'].unique()
```
Out[155]:
```
array([2012], dtype=int64)
```

In [156…
```
sns.pairplot(data)
```
Out[156]:
```
<seaborn.axisgrid.PairGrid at 0x147ae1c8fd0>
```

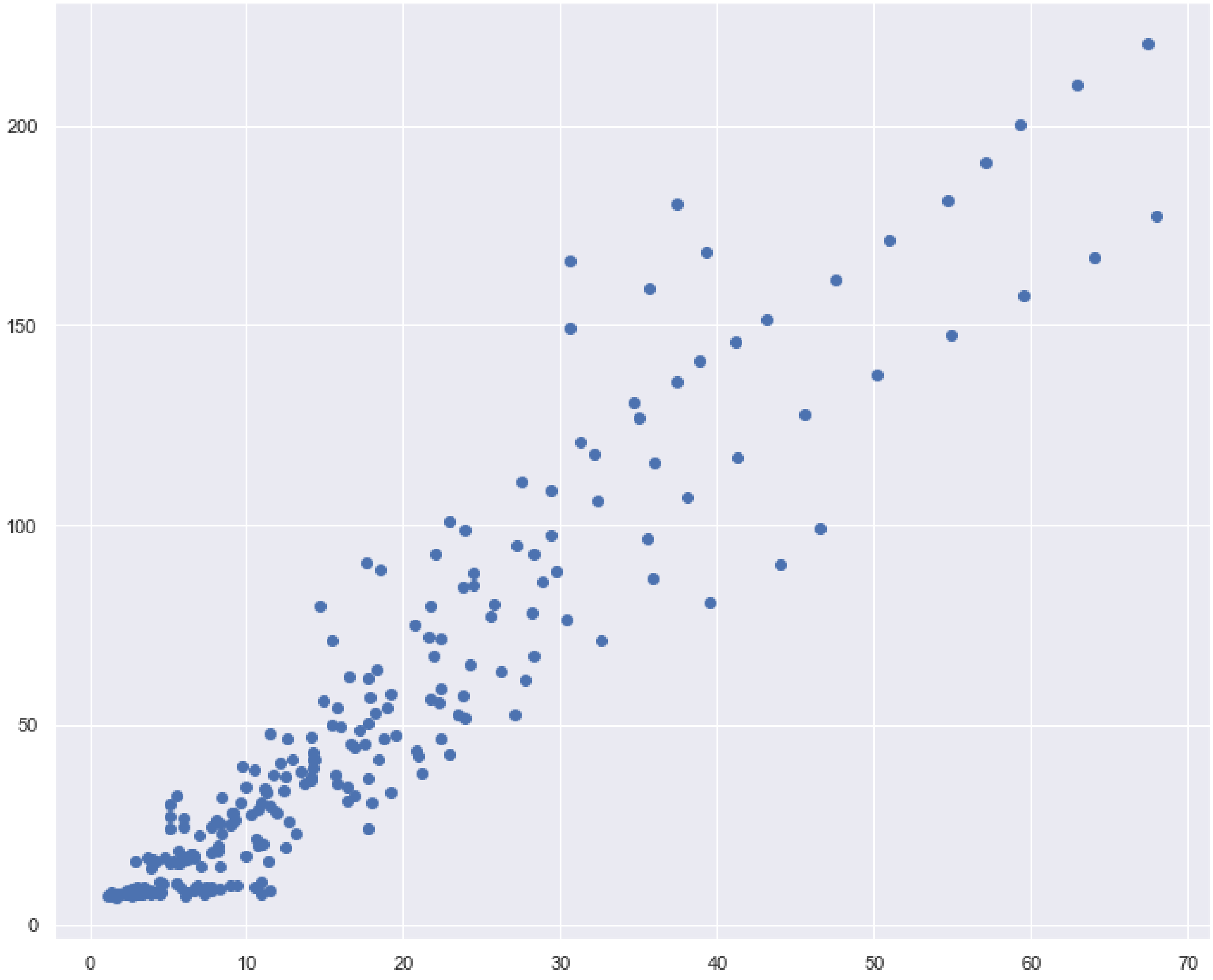


In [167…

```
plt.scatter(data['BUI'],data['DC'])
```

Out[167]:

```
<matplotlib.collections.PathCollection at 0x147ba256cd0>
```

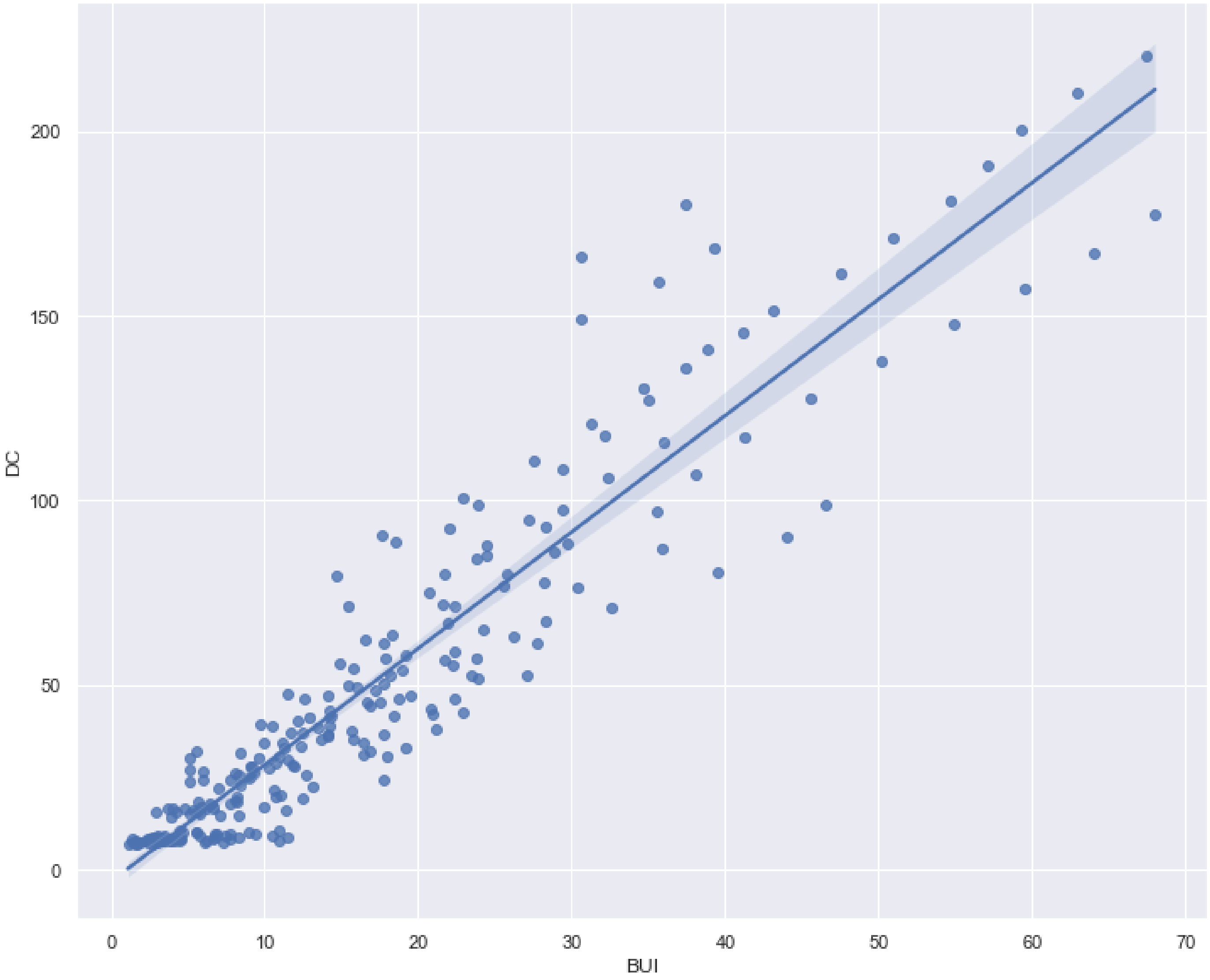


In [169…

```
sns.regplot(x=data['BUI'],y=data['DC'])
```

Out[169]:

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



In [171…

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 243 entries, 0 to 243
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   day          243 non-null    int64  
 1   month        243 non-null    int64  
 2   year         243 non-null    int64  
 3   Temperature  243 non-null    int64  
 4    RH          243 non-null    int64  
 5    Ws          243 non-null    int64  
 6   Rain         243 non-null    float64
 7   FFMC         243 non-null    float64
 8   DMC          243 non-null    float64
 9   DC           243 non-null    float64
 10  ISI          243 non-null    float64
 11  BUI          243 non-null    float64
 12  FWI          243 non-null    float64
 13  Classes      243 non-null    int32  
 14  Region       243 non-null    int32  
dtypes: float64(7), int32(2), int64(6)
memory usage: 36.6 KB
```

In [174…

```
data.columns
```

Out[174]:

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

In [177…

```
data.rename(columns={'Rain ':'Rain'},inplace=True)
```

In [178…

```
data.rename(columns={' RH':'RH'},inplace=True)
```

In [179…

```
data.columns
```

Out[179]:

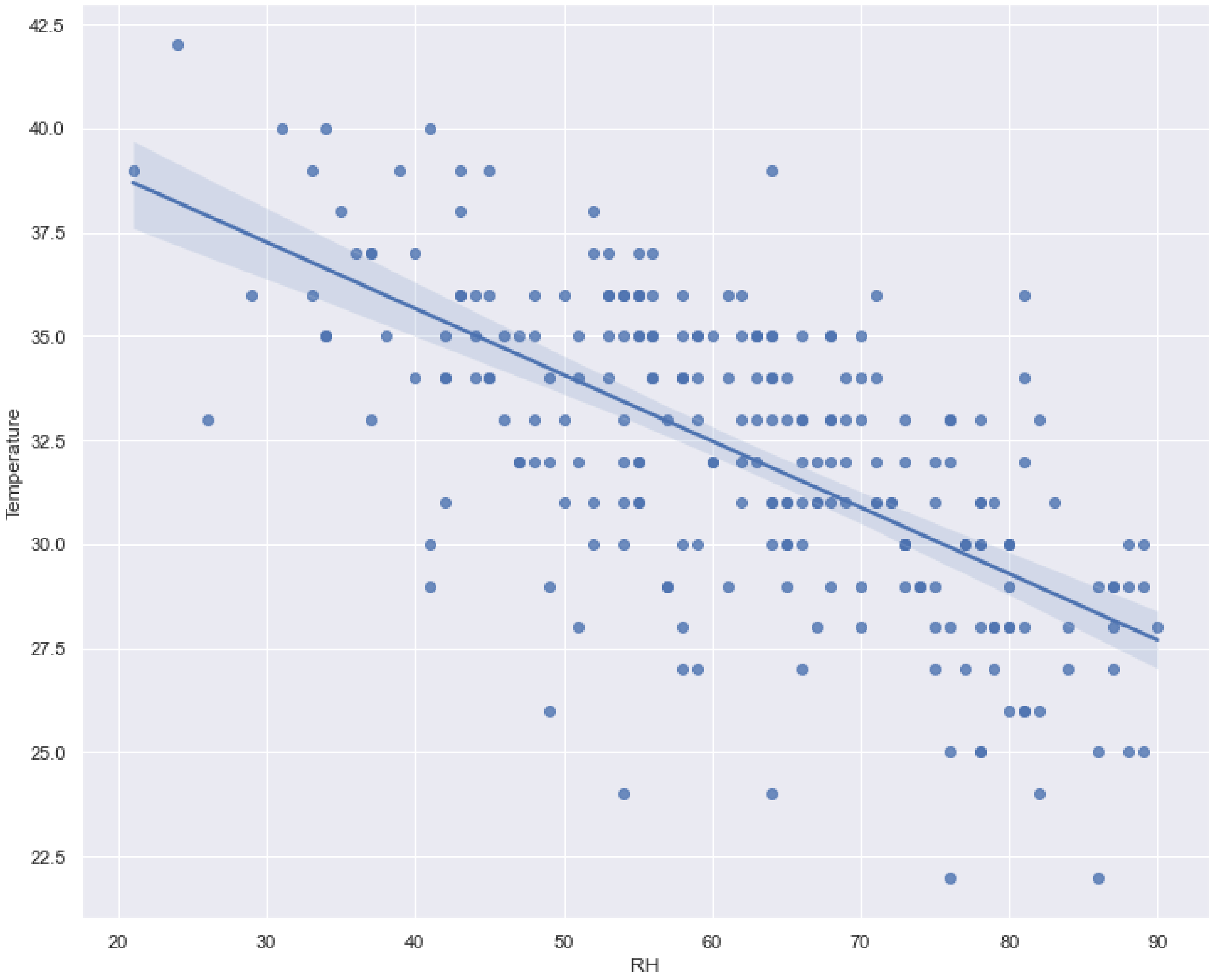
```
Index(['day', 'month', 'year', 'Temperature', 'RH', ' Ws', 'Rain', 'FFMC',
       'DMC', 'DC', 'ISI', 'BUI', 'FWI', 'Classes', 'Region'],
      dtype='object')
```

In [180…

```
sns.regplot(x=data['RH'],y=data['Temperature'])
```

Out[180]:

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



Prepared final dataset for model creation

In [181…

```
data.shape
```

Out[181]:

```
(243, 15)
```

In [182…

```
data.head(2)
```

Out[182]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region
0	1	6	2012	29	57	18	0.0	65.7	3.4	7.6	1.3	3.4	0.5	0	0

1	2	6	2012	29	61	13	1.3	64.4	4.1	7.6	1.0	3.9	0.4	0	0

In [183…

```
data.tail(2)
```

Out[183]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region
242	29	9	2012	24	54	18	0.1	79.7	4.3	15.2	1.7	5.1	0.7	0	1
243	30	9	2012	24	64	15	0.2	67.3	3.8	16.5	1.2	4.8	0.5	0	1

In [185…

```
data.describe()
```

Out[185]:

	day	month	year	Temperature	RH	Ws	Rain	FFMC	DMC
count	243.000000	243.000000	243.0	243.000000	243.000000	243.000000	243.000000	243.000000	243.000000
mean	15.761317	7.502058	2012.0	32.152263	62.041152	15.493827	0.762963	77.842387	14.680658
std	8.842552	1.114793	0.0	3.628039	14.828160	2.811385	2.003207	14.349641	12.393040
min	1.000000	6.000000	2012.0	22.000000	21.000000	6.000000	0.000000	28.600000	0.700000
25%	8.000000	7.000000	2012.0	30.000000	52.500000	14.000000	0.000000	71.850000	5.800000
50%	16.000000	8.000000	2012.0	32.000000	63.000000	15.000000	0.000000	83.300000	11.300000
75%	23.000000	8.000000	2012.0	35.000000	73.500000	17.000000	0.500000	88.300000	20.800000
max	31.000000	9.000000	2012.0	42.000000	90.000000	29.000000	16.800000	96.000000	65.900000

In [186…

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 243 entries, 0 to 243
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   day          243 non-null    int64  
 1   month        243 non-null    int64  
 2   year         243 non-null    int64  
 3   Temperature  243 non-null    int64  
 4   RH           243 non-null    int64  
 5    Ws          243 non-null    int64  
 6   Rain         243 non-null    float64
 7   FFMC         243 non-null    float64
 8   DMC          243 non-null    float64
 9   DC           243 non-null    float64
 10  ISI          243 non-null    float64
 11  BUI          243 non-null    float64
 12  FWI          243 non-null    float64
 13  Classes      243 non-null    int32  
 14  Region       243 non-null    int32  
dtypes: float64(7), int32(2), int64(6)
memory usage: 36.6 KB
```

Problem statement: We have to predict temperature based on other features

So, here our temperature feature is the dependent feature and rest of all are independent features

In [203…

```
# Dependent feature
y = data['Temperature']
```

In [196…

```
# Independent features
X = data.drop('Temperature',axis=1)
```

In [197…

```
X
```

Out[197]:

	day	month	year	RH	Ws	Rain	FFMC	DMC	DC	ISI	BUI	FWI	Classes	Region
0	1	6	2012	57	18	0.0	65.7	3.4	7.6	1.3	3.4	0.5	0	0
1	2	6	2012	61	13	1.3	64.4	4.1	7.6	1.0	3.9	0.4	0	0
2	3	6	2012	82	22	13.1	47.1	2.5	7.1	0.3	2.7	0.1	0	0
3	4	6	2012	89	13	2.5	28.6	1.3	6.9	0.0	1.7	0.0	0	0
4	5	6	2012	77	16	0.0	64.8	3.0	14.2	1.2	3.9	0.5	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
239	26	9	2012	65	14	0.0	85.4	16.0	44.5	4.5	16.9	6.5	1	1
240	27	9	2012	87	15	4.4	41.1	6.5	8.0	0.1	6.2	0.0	0	1
241	28	9	2012	87	29	0.5	45.9	3.5	7.9	0.4	3.4	0.2	0	1
242	29	9	2012	54	18	0.1	79.7	4.3	15.2	1.7	5.1	0.7	0	1
243	30	9	2012	64	15	0.2	67.3	3.8	16.5	1.2	4.8	0.5	0	1

243 rows × 14 columns

In [204…

```
y
```

Out[204]:

```
0      29
1      29
2      26
3      25
4      27
       ..
239    30
240    28
241    27
242    24
243    24
Name: Temperature, Length: 243, dtype: int64
```

In [200…

```
X.shape
```

Out[200]:

```
(243, 14)
```

In [205…

```
y.shape
```

Out[205]:

```
(243,)
```

In [207…

```
from sklearn.model_selection import train_test_split
```

In [208…

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=26)
```

In [209…

```
X_train.shape
```

Out[209]:

```
(170, 14)
```

In [211…

```
X_test.shape
```

Out[211]:
```
(73, 14)
```

In [212…
```
y_train.shape
```

Out[212]:
```
(170,)
```

In [213…
```
y_test.shape
```

Out[213]:
```
(73,)
```

In [214…
```
## Standardize or feature scaling the datasets
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
```

In [215…
```
scaler
```

Out[215]:
```
StandardScaler()
```

In [216…
```
X_train = scaler.fit_transform(X_train)
```

In [217…
```
X_test = scaler.transform(X_test)
```

In [218…
```
X_train
```

Out[218]:
```
array([[-1.59963855, -1.37600597,  0.        , ..., -0.89724684,
        -1.13898959, -0.976741  ],
       [-0.683635  ,  0.43033391,  0.        , ...,  0.49296572,
         0.87797115,  1.02381286],
       [-1.14163677, -0.47283603,  0.        , ..., -0.03342544,
         0.87797115, -0.976741  ],
       ...,
       [-1.02713633, -1.37600597,  0.        , ...,  0.0205634 ,
         0.87797115, -0.976741  ],
       [-1.59963855,  0.43033391,  0.        , ..., -0.843258  ,
        -1.13898959, -0.976741  ],
       [ 0.91937121, -0.47283603,  0.        , ..., -0.77577195,
        -1.13898959, -0.976741  ]])
```

In [219…
```
X_test
```

Out[219]:
```
array([[-0.22563323, -1.37600597,  0.        , ..., -0.91074405,
        -1.13898959,  1.02381286],
       [ 0.46136943,  1.33350385,  0.        , ..., -0.10091149,
         0.87797115, -0.976741  ],
       [-1.4851381 , -1.37600597,  0.        , ..., -0.93773846,
        -1.13898959,  1.02381286],
       ...,
       [-0.45463411,  1.33350385,  0.        , ..., -0.88374963,
        -1.13898959, -0.976741  ],
       [ 0.34686899,  1.33350385,  0.        , ..., -0.19539195,
         0.87797115,  1.02381286],
       [-1.25613722,  0.43033391,  0.        , ..., -0.12790591,
         0.87797115, -0.976741  ]])
```

Model Training : Linear Regression

In [220…
```
from sklearn.linear_model import LinearRegression
```

In [221…
```
regression = LinearRegression()
```

In [222…

```
regression
```

Out[222]:

```
LinearRegression()
```

In [223…

```
regression.fit(X_train,y_train)
```

Out[223]:

```
LinearRegression()
```

In [224…

```
## Coefficients
regression.coef_
```

Out[224]:

```
array([-5.61529842e-01, -3.61751839e-01,  2.22044605e-16, -1.31764101e+00,
       -4.60234910e-01,  1.84639299e-01,  1.04485441e+00,  7.46555404e-01,
        1.14651430e+00, -4.38758105e-01, -1.54170511e+00,  8.71088011e-01,
        9.55726641e-02,  2.17433101e-01])
```

In [225…

```
## Intercept
regression.intercept_
```

Out[225]:

```
32.311764705882354
```

In [226…

```
y_pred = regression.predict(X_test)
```

In [227…

```
y_pred
```

Out[227]:

```
array([28.8157676 , 29.299373  , 29.6764605 , 30.83193012, 33.08225409,
       29.68778145, 32.69761834, 37.28664419, 34.3226546 , 28.74770707,
       33.01273727, 26.44514572, 33.74632555, 25.94880372, 32.39401691,
       32.93549637, 31.43204681, 31.87347861, 36.72701154, 32.50104684,
       26.14360757, 35.0822357 , 33.69825968, 32.83611651, 33.2710649 ,
       37.14937922, 29.93664287, 31.80639816, 29.98367129, 33.31476127,
       29.01784349, 30.64701681, 35.22697307, 33.79926417, 34.60979178,
       33.71553812, 31.17467031, 32.40380095, 33.85297654, 34.10677695,
       30.83759873, 33.63423066, 30.99135726, 32.85563015, 32.67361638,
       31.10867653, 30.58178255, 31.76036358, 39.05004122, 32.75527886,
       32.66660564, 29.24933943, 34.19667261, 31.5165678 , 35.34506598,
       36.52482734, 28.41439365, 36.27316289, 28.58906536, 28.03992483,
       28.94508158, 28.67306446, 28.1185017 , 28.96345325, 35.77676935,
       36.63694396, 31.81701949, 34.33936227, 34.0118185 , 33.09414797,
       28.67513728, 35.63055223, 33.10178721])
```

In [228…

```
y_test
```

Out[228]:

```
135    27
111    28
124    29
227    28
125    30
       ..
155    34
60     35
103    29
232    29
65     34
Name: Temperature, Length: 73, dtype: int64
```

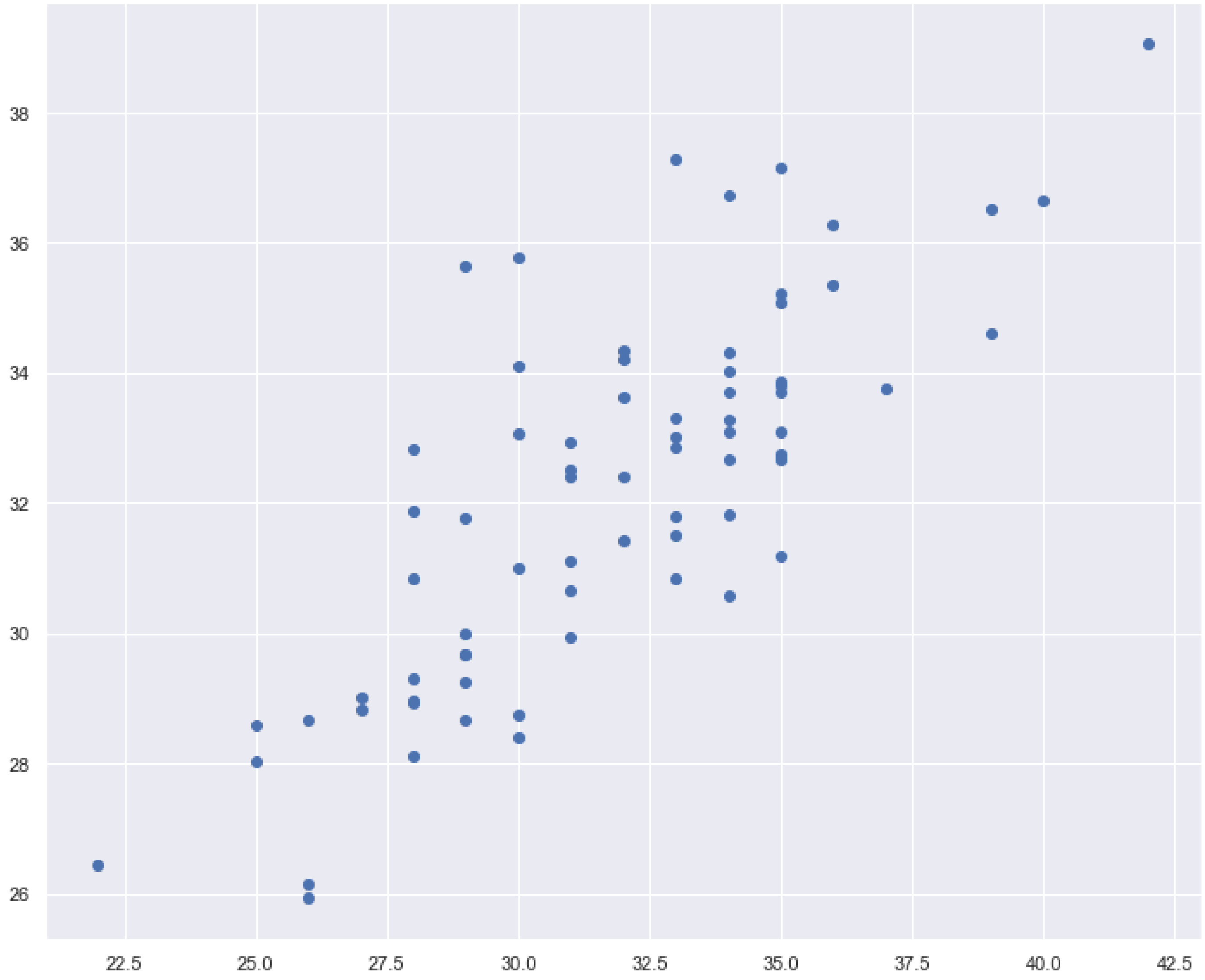
Assumptions

In [232…

```
plt.scatter(y_test,y_pred)
```

Out[232]:

```
<matplotlib.collections.PathCollection at 0x147bdf3f6d0>
```



Original data and Predicted data should have some linear relationship

In [233…
```
residuals = y_test - y_pred
```

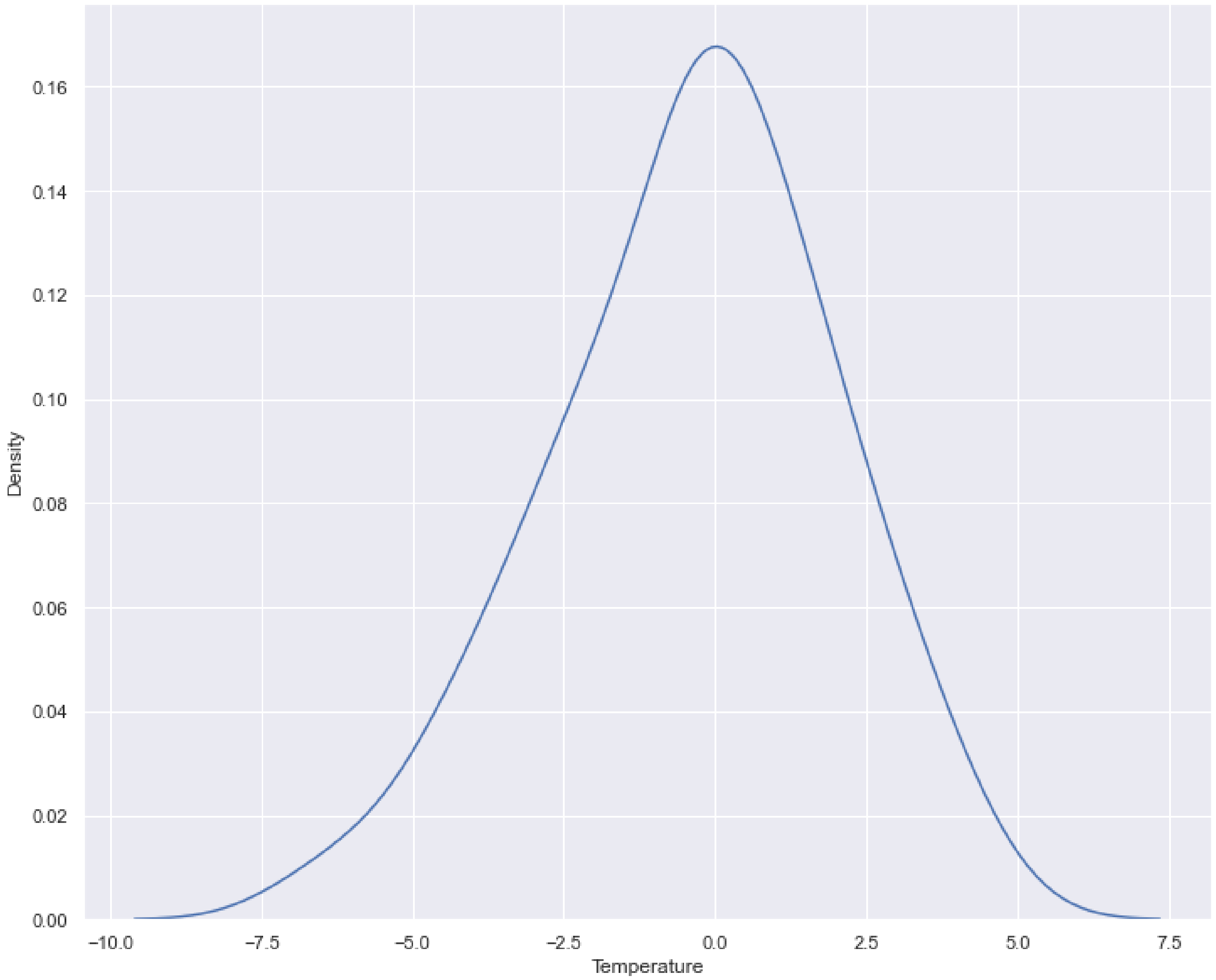
In [234…
```
residuals
```

Out[234]:
```
135   -1.815768
111   -1.299373
124   -0.676461
227   -2.831930
125   -3.082254
         ...
155   -0.011818
60     1.905852
103    0.324863
232   -6.630552
65     0.898213
Name: Temperature, Length: 73, dtype: float64
```

In [237…
```
sns.distplot(residuals,hist=False)
```

```
C:\Users\chatt\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning:
`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `kdeplot` (an axes-level function for kernel density plots).
  warnings.warn(msg, FutureWarning)
```

Out[237]:
```
<AxesSubplot:xlabel='Temperature', ylabel='Density'>
```



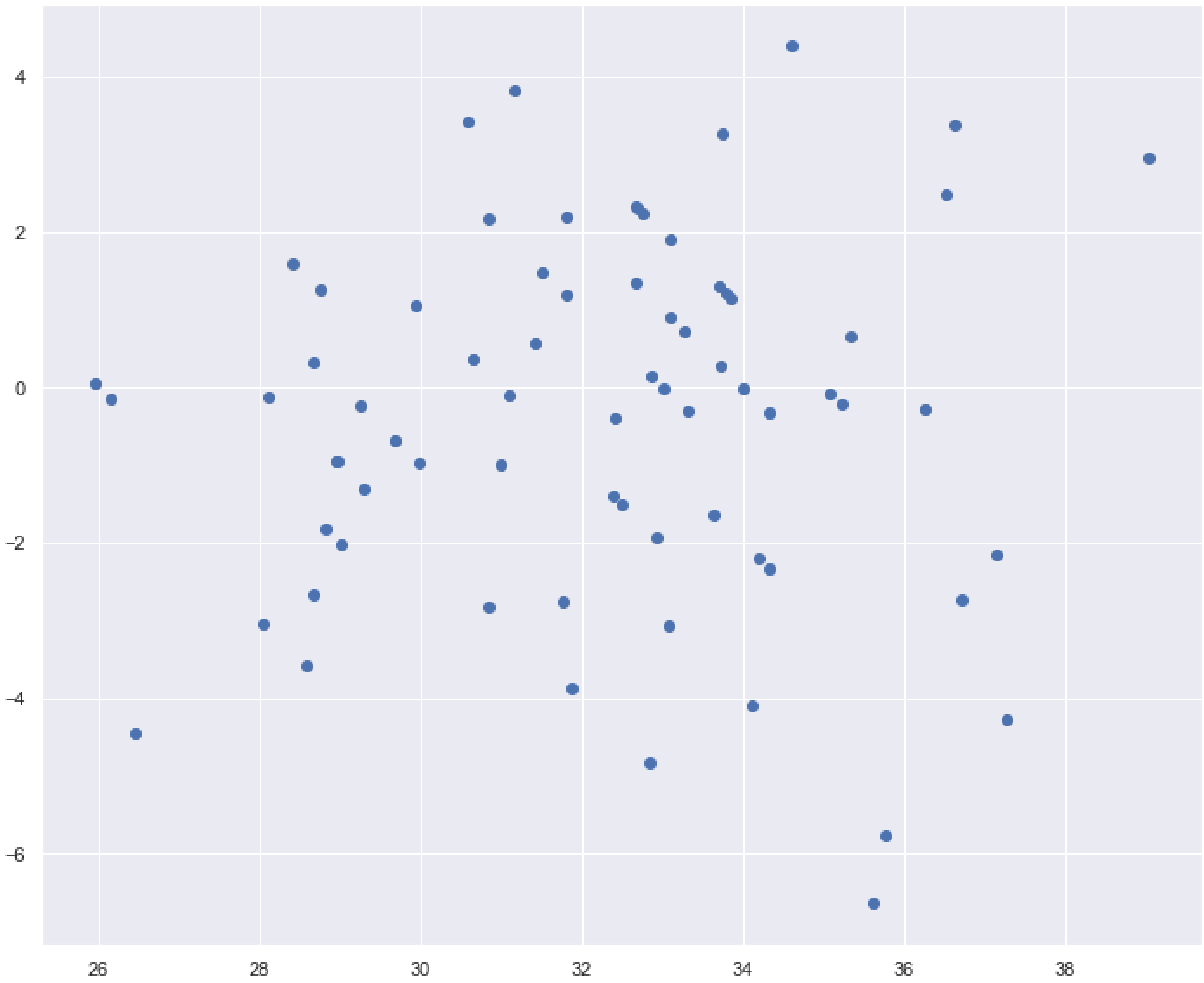
Residuals should follow a Normal Gaussian Distribution

In [238…

```
## Residuals and predicted values should be uniformly distributed
plt.scatter(y_pred,residuals)
```

Out[238]:

```
<matplotlib.collections.PathCollection at 0x147bef410a0>
```



In [239…

```
## Check the perform metrics
from sklearn.metrics import mean_squared_error
from sklearn.metrics import median_absolute_error
```

In [242…

```
print("MSE:  ",mean_squared_error(y_test,y_pred))
```

```
MSE:   5.4945847118297175
```

In [243…

```
print("MAE:  ",median_absolute_error(y_test,y_pred))
```

```
MAE:   1.4834321979678933
```

In [244…

```
print("RMSE:  ",np.sqrt(mean_squared_error(y_test,y_pred)))
```

```
RMSE:   2.344053052264329
```

R-Squared and Adjusted R-Squared for model performance

In [245…

```
from sklearn.metrics import r2_score
```

In [247…

```
score = r2_score(y_test,y_pred)
```

In [248…

```
score
```

Out[248]:

```
0.6061783197129715
```

In [249…
```
## Adjusted R square
#display adjusted R-squared
1 - (1-score)*(len(y_test)-1)/(len(y_test)-X_test.shape[1]-1)
```

Out[249]:
```
0.5111179141264475
```

Ridge Regression

In [250…
```
from sklearn.linear_model import Ridge
```

In [251…
```
ridge  = Ridge()
```

In [252…
```
ridge.fit(X_train,y_train)
```

Out[252]:
```
Ridge()
```

In [253…
```
ridge.coef_
```

Out[253]:
```
array([-0.55348071, -0.34865427,  0.        , -1.30689518, -0.45921557,
        0.16830843,  0.99993902, -0.10994351,  0.62383067, -0.28294097,
       -0.11162647,  0.65796468,  0.11191881,  0.2126251 ])
```

In [254…
```
ridge.intercept_
```

Out[254]:
```
32.311764705882354
```

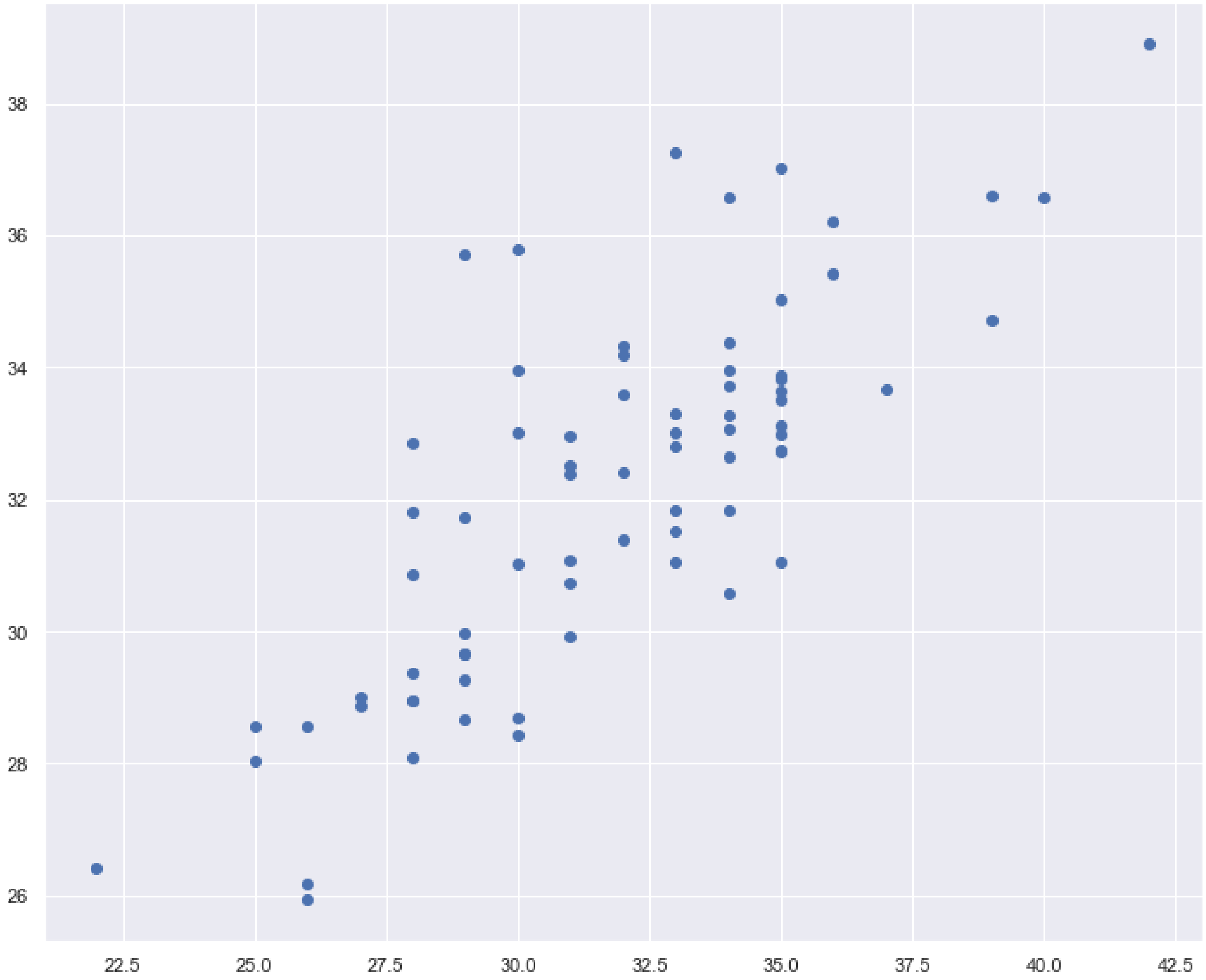
In [256…
```
ridge_y_pred = ridge.predict(X_test)
```

In [257…
```
ridge_y_pred
```

Out[257]:
```
array([28.87187682, 29.36187881, 29.65747886, 30.86192924, 33.02361569,
       29.65224501, 32.98141415, 37.24812834, 34.36957818, 28.69418439,
       33.02133163, 26.41912425, 33.66654756, 25.9540052 , 32.3737563 ,
       32.96846908, 31.38898194, 31.81359756, 36.57298278, 32.50988057,
       26.19090235, 35.03766908, 33.51729393, 32.85769689, 33.26538901,
       37.02697638, 29.91470198, 31.83403789, 29.96952777, 33.30577846,
       29.01333366, 30.74789748, 33.64773959, 33.81242166, 34.70705709,
       33.71360545, 31.03857969, 32.40107895, 33.87107627, 33.95958811,
       31.0377332 , 33.60237006, 31.01848392, 32.81347914, 32.72054887,
       31.06627153, 30.57718966, 31.7200776 , 38.89436872, 32.74894602,
       32.65175034, 29.27687424, 34.19877987, 31.53377958, 35.41107236,
       36.61194963, 28.43898174, 36.19834973, 28.55475975, 28.0436946 ,
       28.95158455, 28.57419822, 28.10055277, 28.96580545, 35.79869966,
       36.57421405, 31.82664234, 34.3333526 , 33.96273724, 33.11414946,
       28.66407717, 35.69853868, 33.07794882])
```

In [258…
```
plt.scatter(y_test,ridge_y_pred)
```

Out[258]:
```
<matplotlib.collections.PathCollection at 0x147ba27b880>
```



In [259…

```
ridge_residuals = y_test - ridge_y_pred
```

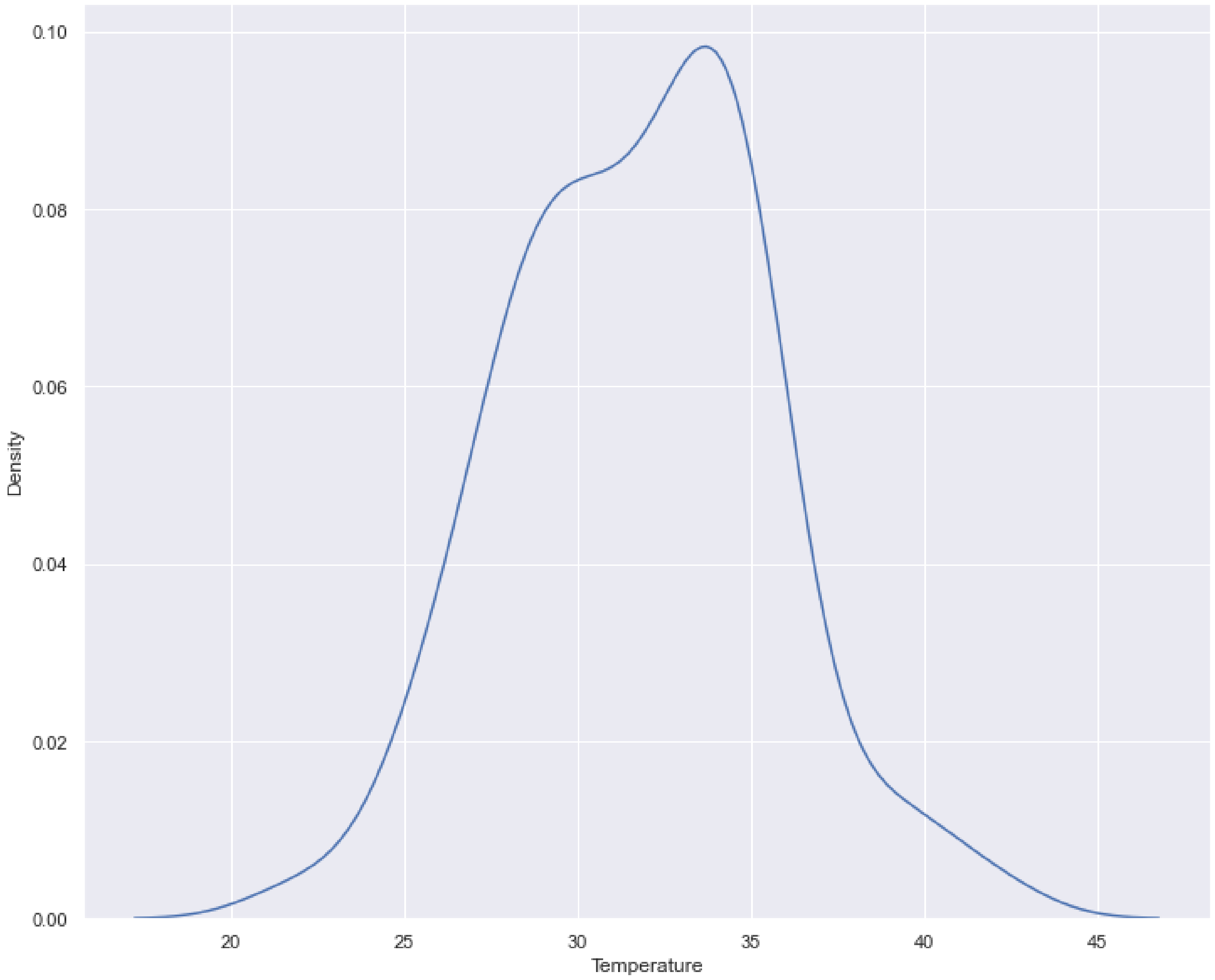
In [260…

```
sns.distplot(y_test,ridge_residuals,hist=False)
```

```
C:\Users\chatt\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning:
`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `kdeplot` (an axes-level function for kernel density plots).
  warnings.warn(msg, FutureWarning)
```

Out[260]:

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

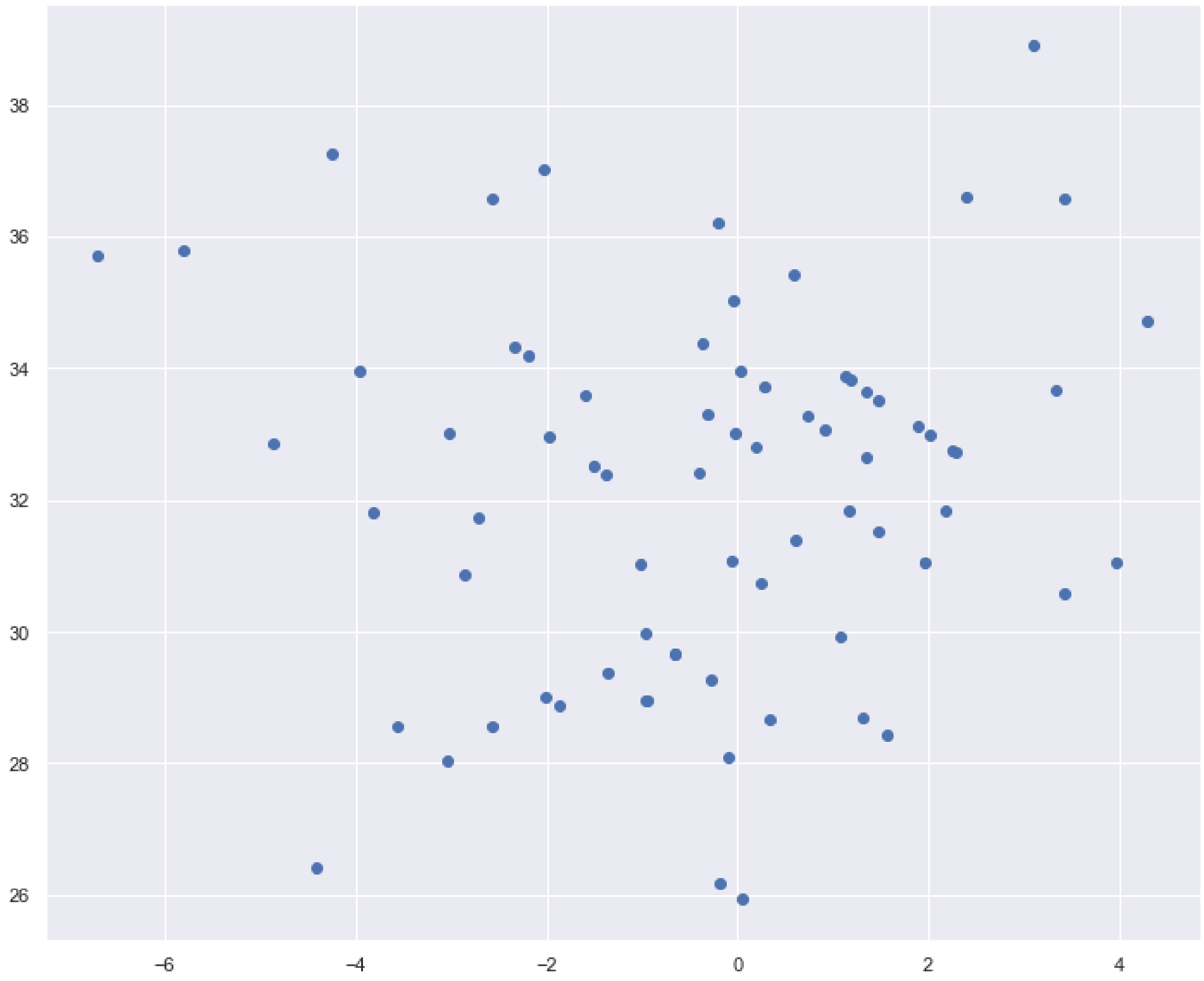


In [282…

```
plt.scatter(ridge_residuals,ridge_y_pred)
```

Out[282]:

```
<matplotlib.collections.PathCollection at 0x147c3ba0fd0>
```



```
In [262…  ridge_r2_score = r2_score(y_test,ridge_y_pred)
```

```
In [263…  ridge_r2_score
```

```
Out[263]: 0.6075790252064115
```

```
In [264…  ridge_adj_re_score = 1 - (1-ridge_r2_score)*(len(y_test)-1)/(len(y_test)-X_test.shape[1]
```

```
In [265…  ridge_adj_re_score
```

```
Out[265]: 0.5128567209458901
```

Lasso Regression

```
In [267…  from sklearn.linear_model import Lasso
```

```
In [268…  lasso = Lasso()
```

```
In [269…  lasso.fit(X_train,y_train)
```

```
Out[269]: Lasso()
```

```
In [270…  lasso.coef_
```

Out[270]:

```
array([-0.        , -0.        ,  0.        , -0.85386496, -0.        ,
       -0.        ,  0.71959202,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ])
```

In [271…

```
lasso.intercept_
```

Out[271]:

```
32.311764705882354
```

In [273…

```
lasso_y_pred = lasso.predict(X_test)
```

In [274…

```
lasso_y_pred
```

Out[274]:

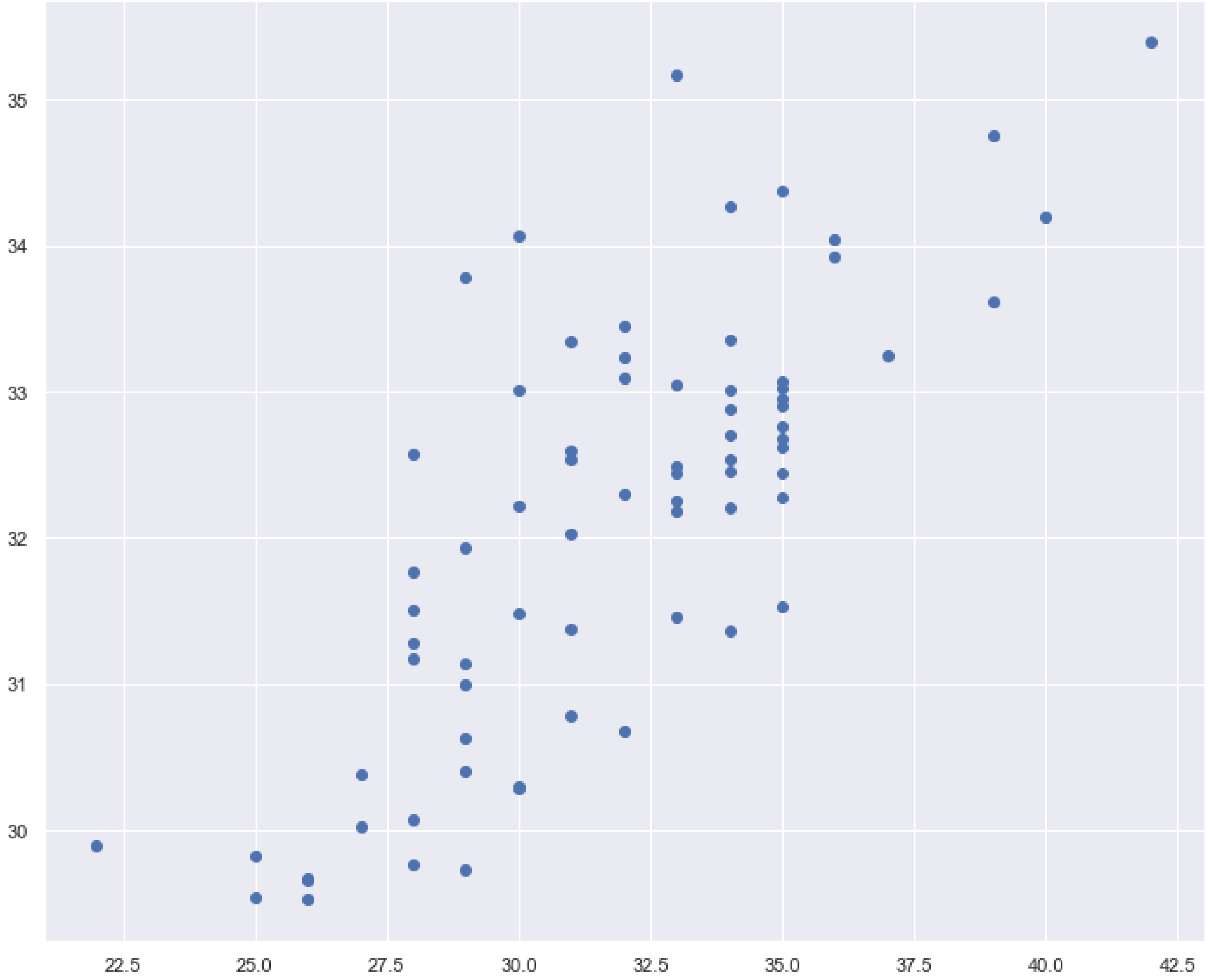
```
array([30.02796648, 31.2881686 , 29.73026093, 31.50302723, 32.22419019,
       31.14104144, 32.95796465, 35.16841905, 32.7086602 , 30.29632589,
       32.4396289 , 29.89604657, 33.2512424 , 29.66714086, 32.60156683,
       33.34370861, 30.68081788, 31.77166205, 34.26433934, 32.53911865,
       29.66396503, 33.02368046, 32.9032119 , 32.57048044, 32.88156119,
       34.38038011, 30.78088771, 32.18204894, 30.99767019, 32.49572543,
       30.37734076, 31.38188678, 32.6781539 , 33.07534919, 33.62235919,
       33.35198412, 31.53438902, 32.29616585, 32.76668059, 33.00896149,
       31.4603059 , 33.09315219, 31.48455235, 33.04340731, 32.276439  ,
       32.03482998, 31.36023607, 31.93167612, 35.39155319, 32.44598056,
       32.45376778, 30.62914915, 33.24104303, 32.24910852, 33.93026411,
       34.75207698, 30.28169872, 34.04438103, 29.82465099, 29.53522095,
       30.07309995, 29.53262521, 29.76595872, 31.18116703, 34.06862748,
       34.20121927, 32.21264705, 33.45272584, 33.01088535, 32.62196557,
       30.40332746, 33.78814484, 32.54354644])
```

In [276…

```
plt.scatter(y_test,lasso_y_pred)
```

Out[276]:

```
<matplotlib.collections.PathCollection at 0x147bfa9c4f0>
```



In [277…

```
lasso_residuals = y_test-lasso_y_pred
```

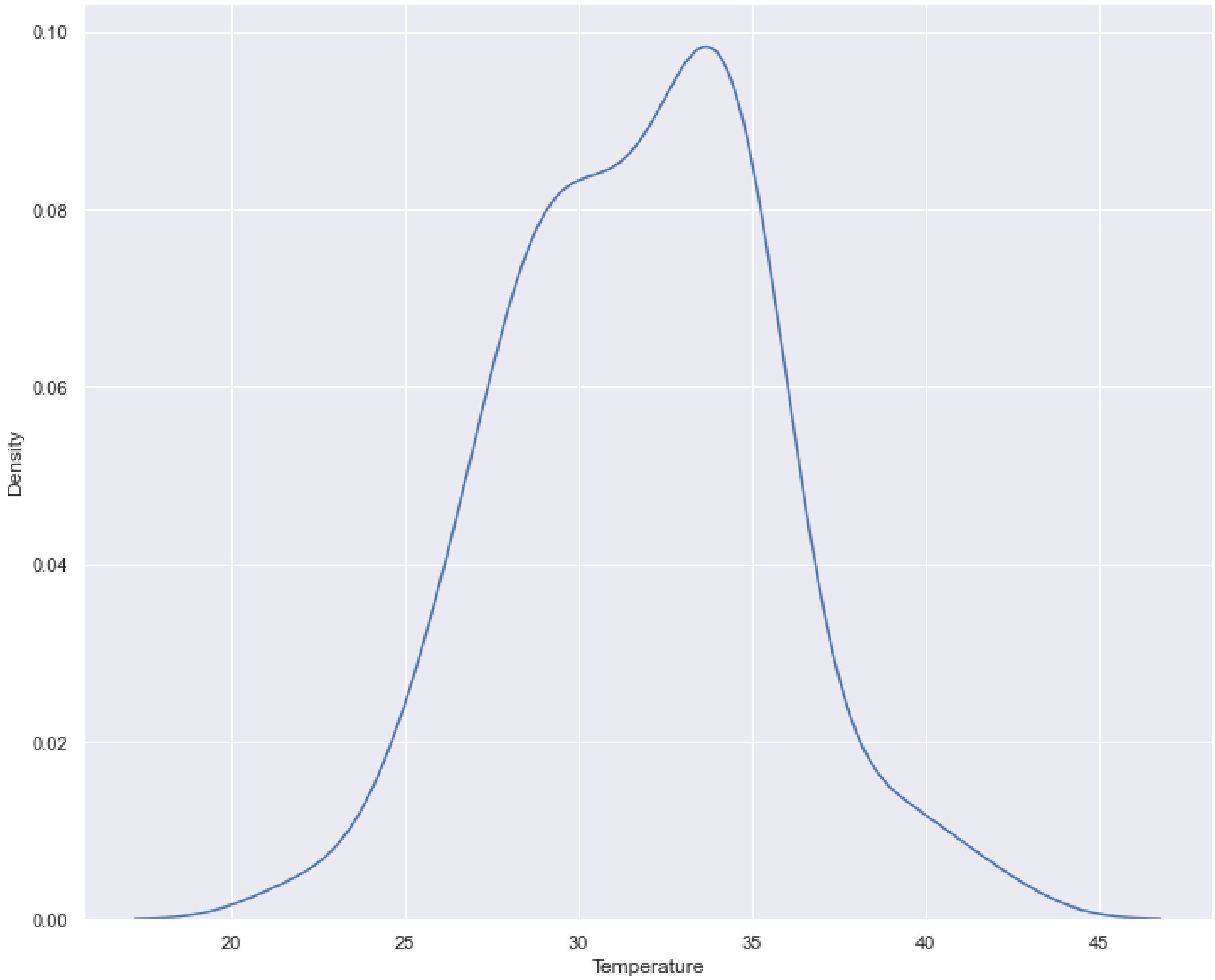
In [279…

```
sns.distplot(y_test,lasso_residuals,hist=False)
```

```
C:\Users\chatt\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning:
`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `kdeplot` (an axes-level function for kernel density plots).
  warnings.warn(msg, FutureWarning)
```

Out[279]:

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

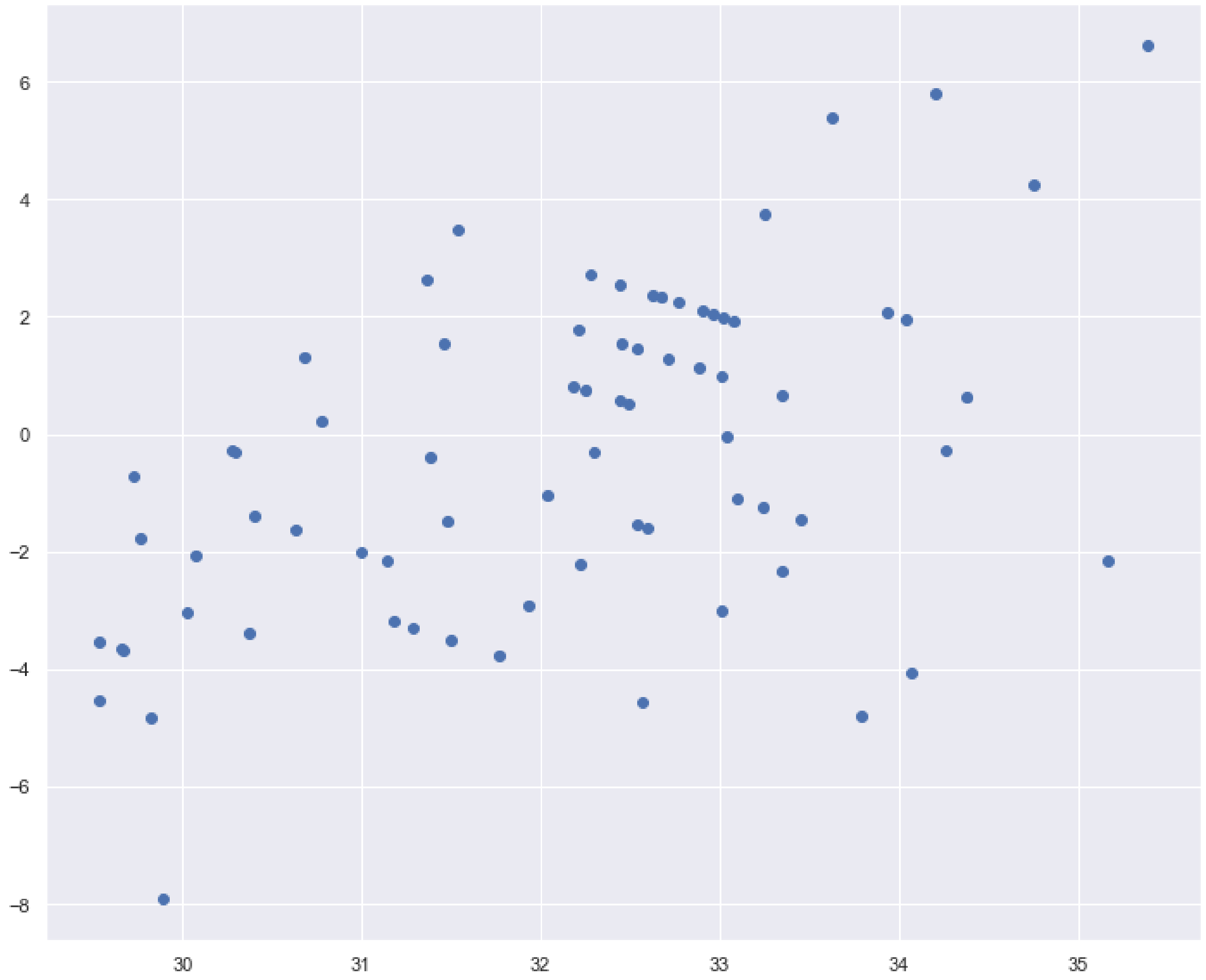


In [281…

```
plt.scatter(lasso_y_pred,lasso_residuals)
```

Out[281]:

```
<matplotlib.collections.PathCollection at 0x147c399ffa0>
```



In [283…
```
lasso_r2_score = r2_score(y_test,lasso_y_pred)
```

In [284…
```
lasso_r2_score
```

Out[284]:
```
0.4390701620494102
```

In [285…
```
lasso_adj_r2_score = lasso_r2_score = 1 - (1-lasso_r2_score)*(len(y_test)-1)/(len(y_test
```

In [286…
```
lasso_adj_r2_score
```

Out[286]:
```
0.30367330461306097
```

ElasticNet Regression

In [287…
```
from sklearn.linear_model import ElasticNet
```

In [288…
```
elasticnet = ElasticNet()
```

In [289…
```
elasticnet.fit(X_train,y_train)
```

Out[289]:
```
ElasticNet()
```

In [290…
```
elasticnet.coef_
```

```
Out[290]: array([-0.        , -0.        ,  0.        , -0.73305525, -0.04412262,
                 -0.        ,  0.58163161,  0.04026034,  0.        ,  0.2269735 ,
                  0.        ,  0.19209178,  0.14177145,  0.        ])
```

```
In [291…  elasticnet.intercept_
```

```
Out[291]: 32.311764705882354
```

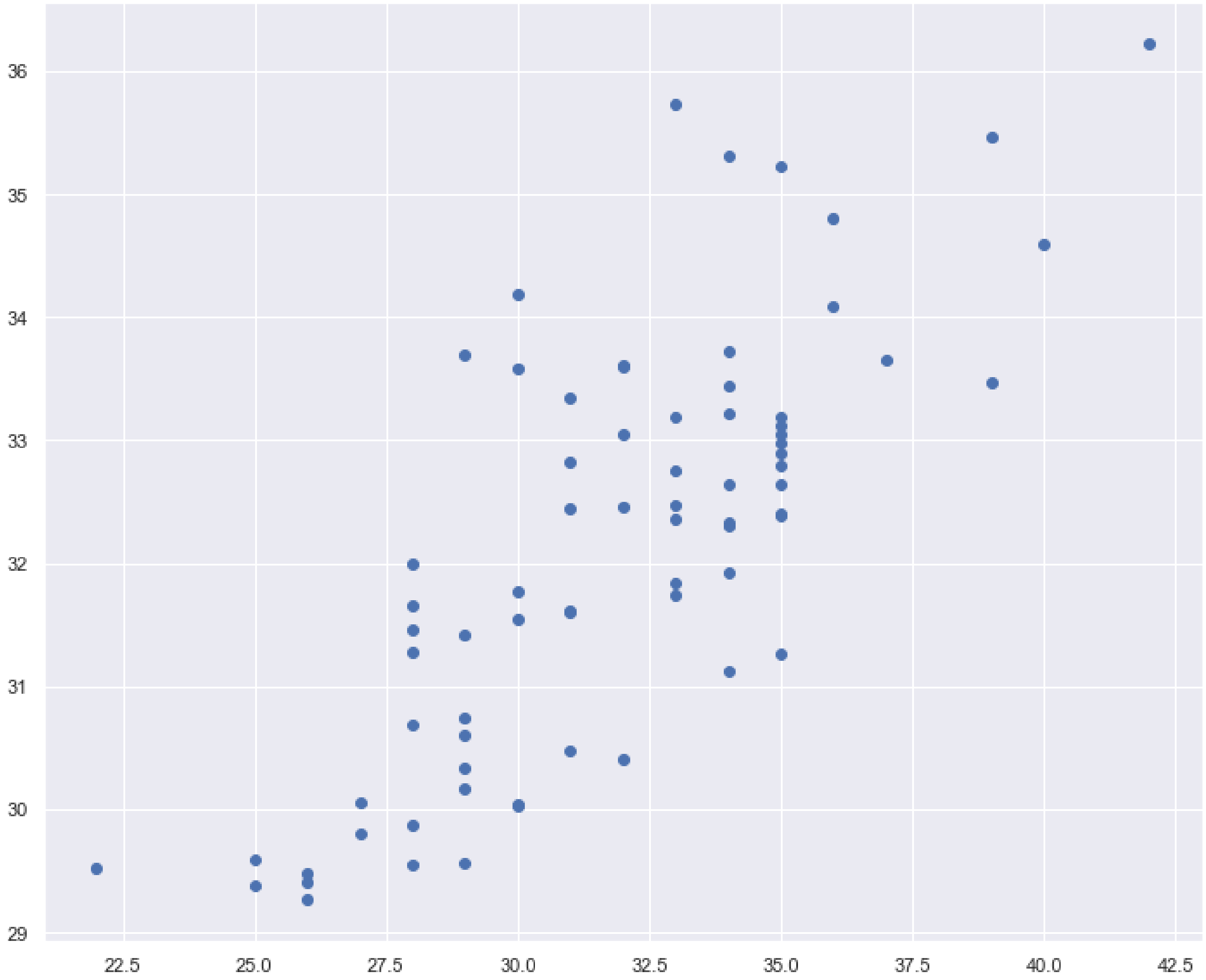
```
In [292…  elas_y_pred = elasticnet.predict(X_test)
```

```
In [293…  elas_y_pred
```

```
Out[293]: array([29.80709878, 31.46148336, 29.5654218 , 31.66221748, 31.76426509,
                 30.74750995, 33.18794852, 35.73554973, 33.44376303, 30.0354734 ,
                 32.46873391, 29.52199938, 33.64833079, 29.40822434, 32.82652791,
                 33.33825087, 30.41165406, 31.27848586, 35.31200794, 32.44057738,
                 29.48234128, 32.88718833, 32.78888506, 31.99942232, 32.30787046,
                 35.21989434, 30.47556576, 32.36006969, 30.59749312, 32.75738971,
                 30.05648001, 31.61236737, 32.3865618 , 32.63971133, 33.47631   ,
                 33.72751988, 31.26352413, 32.45674493, 33.11594517, 33.58404231,
                 31.8430754 , 33.04219951, 31.53935625, 33.18743894, 32.39515962,
                 31.60526857, 31.12104578, 31.41682768, 36.2187701 , 32.97477457,
                 31.92576768, 30.33416224, 33.59316529, 31.73536502, 34.08203678,
                 35.4694917 , 30.02795397, 34.80039451, 29.59724835, 29.38093762,
                 29.87585392, 29.27359357, 29.55463493, 30.68385869, 34.18892201,
                 34.59057365, 32.33825306, 33.61608097, 33.21338799, 33.04324665,
                 30.16259338, 33.68840979, 32.635962  ])
```

```
In [294…  plt.scatter(y_test,elas_y_pred)
```

```
Out[294]: <matplotlib.collections.PathCollection at 0x147c03ac910>
```



In [295…

```
elas_residuals = y_test - elas_y_pred
```

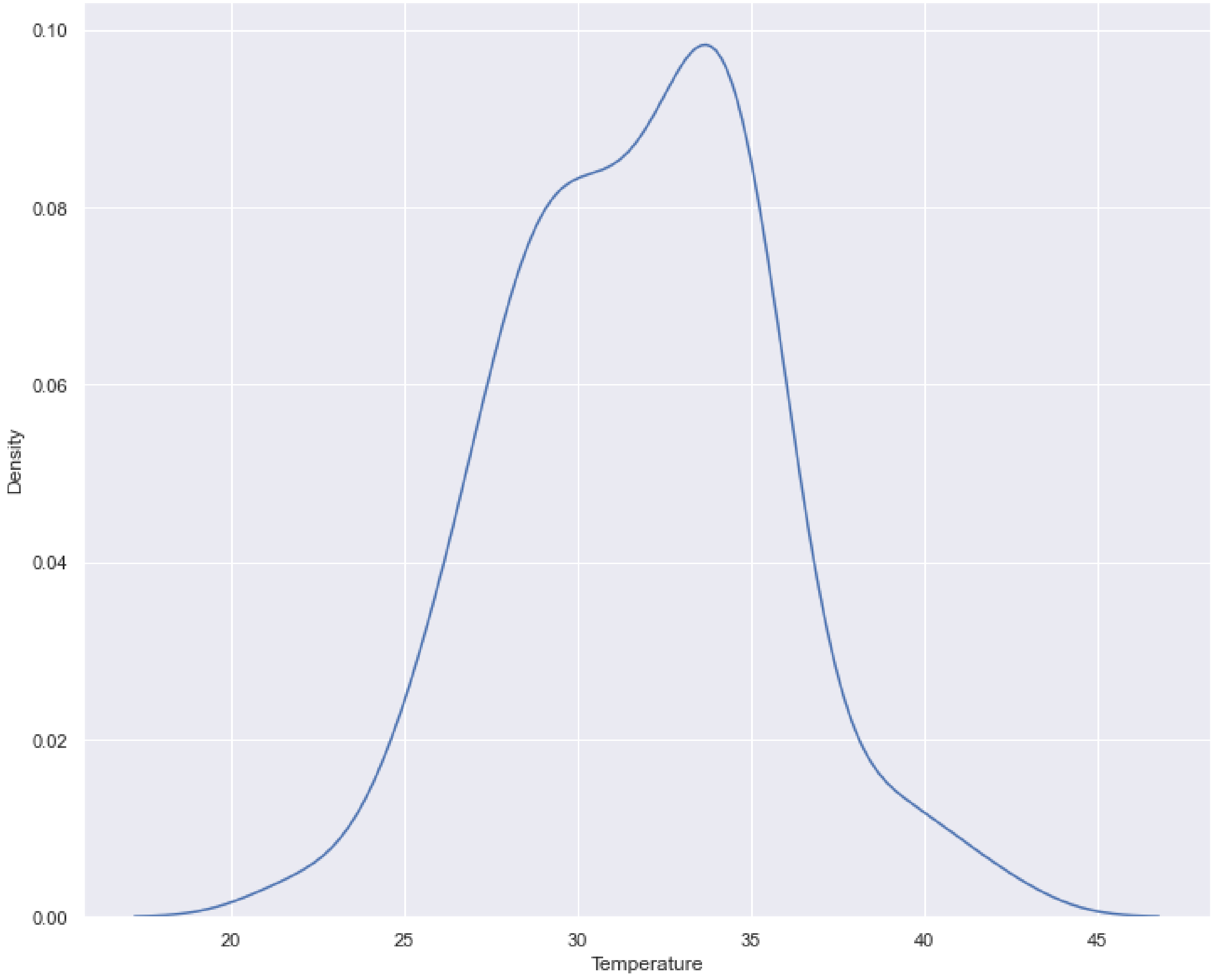
In [296…

```
sns.distplot(y_test,elas_residuals,hist=False)
```

```
C:\Users\chatt\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning:
`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `kdeplot` (an axes-level function for kernel density plots).
  warnings.warn(msg, FutureWarning)
```

Out[296]:

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

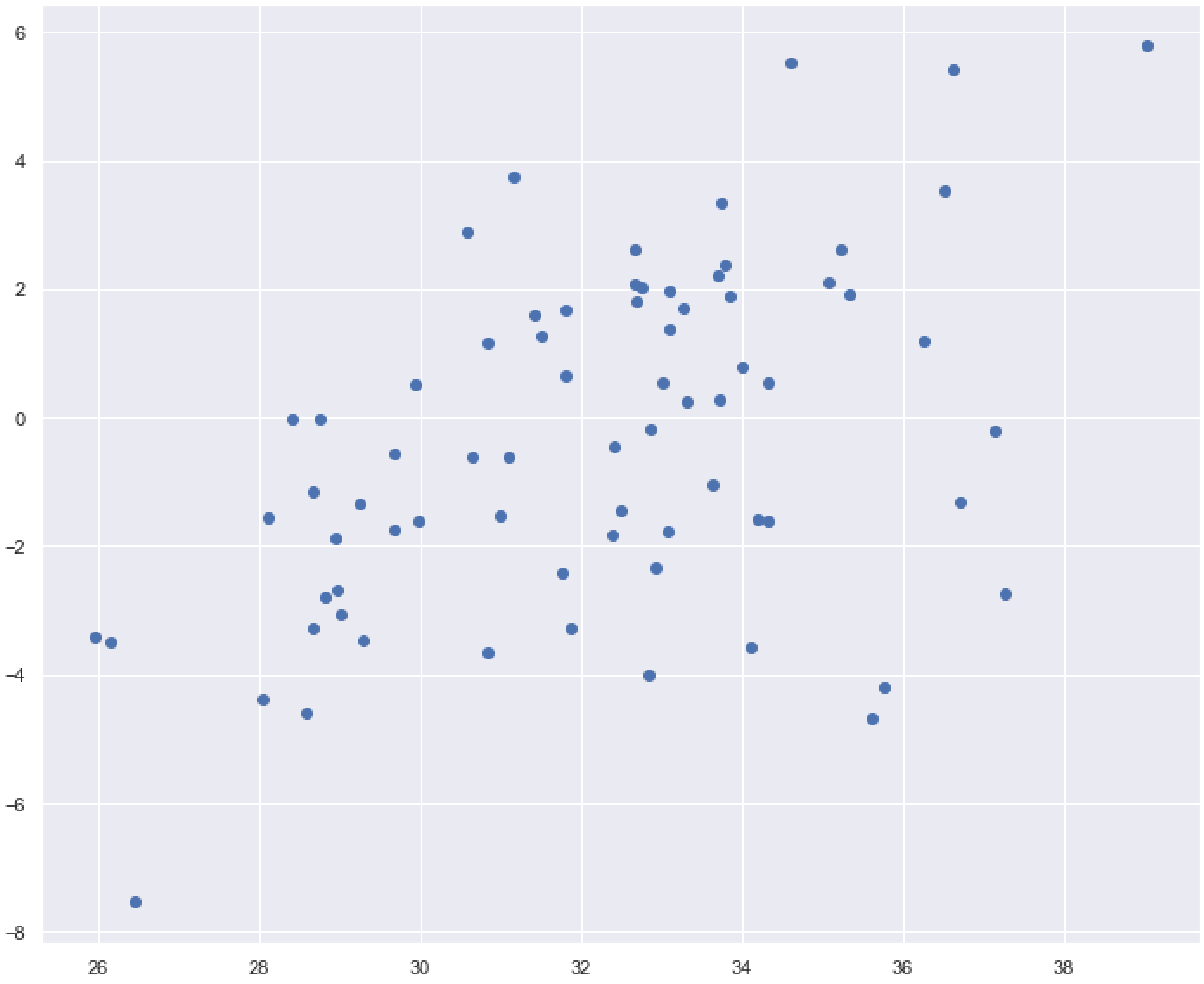


In [297…

```
plt.scatter(y_pred,elas_residuals)
```

Out[297]:

```
<matplotlib.collections.PathCollection at 0x147bf9b6bb0>
```



```
In [298…  elas_r2_score = r2_score(y_test,elas_y_pred)
```

```
In [299…  elas_r2_score
```

```
Out[299]: 0.4886420890206643
```

```
In [300…  elas_adj_r2_score = elas_r2_score = 1 - (1-elas_r2_score)*(len(y_test)-1)/(len(y_test)-X
```

```
In [301…  elas_adj_r2_score
```

```
Out[301]: 0.3652108691291005
```

Final Conclusion

1. Linear Regression

- r2_score = 60
- adj_r2_score = 51

1. Ridge Regression

- r2_score = 60
- adj_r2_score = 51

- Predicted and Residuals not fully normally distributed

1. Lasso Regression

- r2_score = 43
- adj_r2_score = 30

1. ElasticNet Regression

- r2_score = 48
- adj_r2_score = 36

Linear Regression provides us more accurate predictions

In [ ]:

```

```