knn-classification-star-type-classification

February 5, 2022

[1]:
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
```

1 Importing Libraries

[2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from sklearn import preprocessing
from category_encoders import *
from sklearn.preprocessing import LabelEncoder
%matplotlib inline
from sklearn.neighbors import KNeighborsClassifier
from sklearn import datasets, linear_model, metrics
from sklearn.metrics import  confusion_matrix
from sklearn.metrics import accuracy_score, confusion_matrix,␣
 ↪classification_report
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report,confusion_matrix
```

2 Loading DataSet

[3]:
```
df = pd.read_csv('Stars.csv')
df
```

[3]:

	Temperature	L	R	A_M	Color	Spectral_Class	Type
0	3068	0.002400	0.1700	16.12	Red	M	0
1	3042	0.000500	0.1542	16.60	Red	M	0
2	2600	0.000300	0.1020	18.70	Red	M	0
3	2800	0.000200	0.1600	16.65	Red	M	0
4	1939	0.000138	0.1030	20.06	Red	M	0
..	…	…	…	…	…	…	…

```
235        38940  374830.000000  1356.0000  -9.93   Blue              O     5
236        30839  834042.000000  1194.0000 -10.63   Blue              O     5
237         8829  537493.000000  1423.0000 -10.73  White              A     5
238         9235  404940.000000  1112.0000 -11.23  White              A     5
239        37882  294903.000000  1783.0000  -7.80   Blue              O     5

[240 rows x 7 columns]
```

3 Exploratory Data Analysis

[4]:
```
df.head()
```

[4]:
```
   Temperature         L       R    A_M Color Spectral_Class  Type
0         3068  0.002400  0.1700  16.12   Red               M     0
1         3042  0.000500  0.1542  16.60   Red               M     0
2         2600  0.000300  0.1020  18.70   Red               M     0
3         2800  0.000200  0.1600  16.65   Red               M     0
4         1939  0.000138  0.1030  20.06   Red               M     0
```

[5]:
```
df.tail()
```

[5]:
```
     Temperature         L       R    A_M  Color Spectral_Class  Type
235        38940  374830.0  1356.0  -9.93   Blue              O     5
236        30839  834042.0  1194.0 -10.63   Blue              O     5
237         8829  537493.0  1423.0 -10.73  White              A     5
238         9235  404940.0  1112.0 -11.23  White              A     5
239        37882  294903.0  1783.0  -7.80   Blue              O     5
```

[6]:
```
df.dtypes
```

[6]:
```
Temperature         int64
L                 float64
R                 float64
A_M               float64
Color              object
Spectral_Class     object
Type                int64
dtype: object
```

[7]:
```
df.columns
```

[7]:
```
Index(['Temperature', 'L', 'R', 'A_M', 'Color', 'Spectral_Class', 'Type'],
dtype='object')
```

[8]:
```
df.shape
```

[8]:
```
(240, 7)
```

[9]:
```
df.size
```

[9]:
```
1680
```

[10]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 240 entries, 0 to 239
Data columns (total 7 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Temperature     240 non-null    int64
 1   L               240 non-null    float64
 2   R               240 non-null    float64
 3   A_M             240 non-null    float64
 4   Color           240 non-null    object
 5   Spectral_Class  240 non-null    object
 6   Type            240 non-null    int64
dtypes: float64(3), int64(2), object(2)
memory usage: 13.2+ KB
```

[11]:
```
df.describe()
```

[11]:

	Temperature	L	R	A_M	Type
count	240.000000	240.000000	240.000000	240.000000	240.000000
mean	10497.462500	107188.361635	237.157781	4.382396	2.500000
std	9552.425037	179432.244940	517.155763	10.532512	1.711394
min	1939.000000	0.000080	0.008400	-11.920000	0.000000
25%	3344.250000	0.000865	0.102750	-6.232500	1.000000
50%	5776.000000	0.070500	0.762500	8.313000	2.500000
75%	15055.500000	198050.000000	42.750000	13.697500	4.000000
max	40000.000000	849420.000000	1948.500000	20.060000	5.000000

[12]:
```
df.isnull().sum()
```

[12]:
```
Temperature       0
L                 0
R                 0
A_M               0
Color             0
Spectral_Class    0
Type              0
dtype: int64
```

[13]:
```
df.duplicated().sum()
```

[13]:
```
0
```

[14]:
```
df.skew()
```

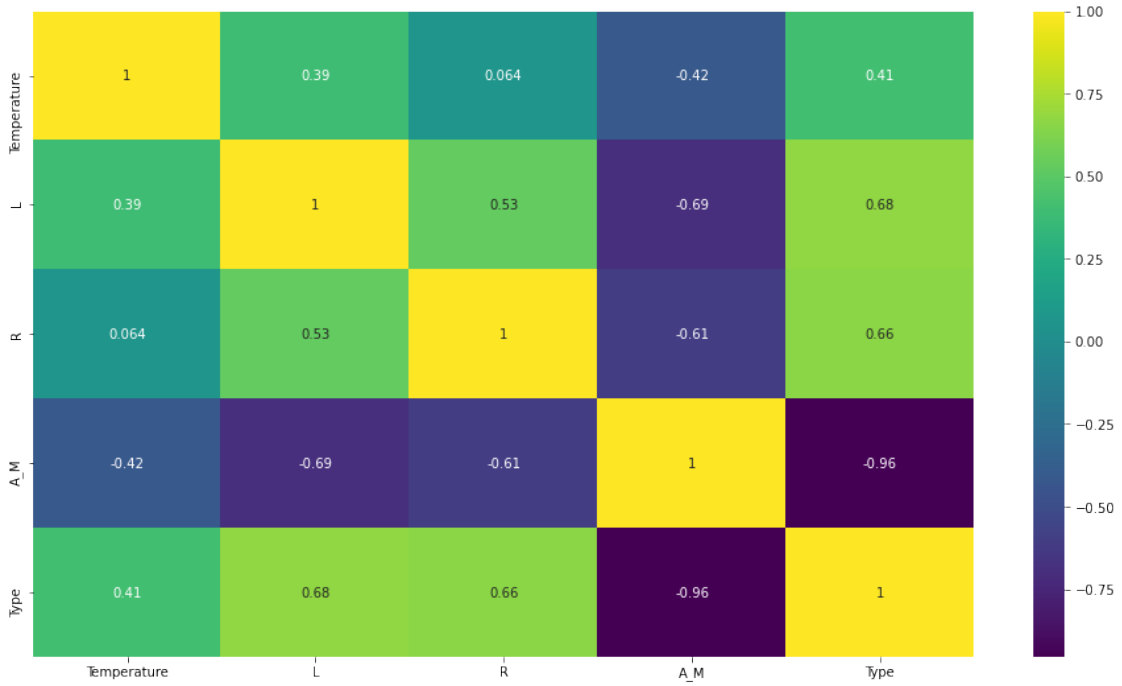
[14]:
```
Temperature    1.321568
L              2.068069
R              1.946800
A_M           -0.121540
Type           0.000000
dtype: float64
```

[15]:
```
df.corr()
```

[15]:

	Temperature	L	R	A_M	Type
Temperature	1.000000	0.393404	0.064216	-0.420261	0.411129
L	0.393404	1.000000	0.526516	-0.692619	0.676845
R	0.064216	0.526516	1.000000	-0.608728	0.660975
A_M	-0.420261	-0.692619	-0.608728	1.000000	-0.955276
Type	0.411129	0.676845	0.660975	-0.955276	1.000000

[16]:
```
plt.figure(figsize=(16,9))
x = df.drop(['Color','Spectral_Class'],axis = 1)
ax = sns.heatmap(df.corr(),annot = True,cmap = 'viridis')
plt.show()
```



4 Data Visualisation Using Autoviz

[17]:
```
! pip install Autoviz
```

```
Requirement already satisfied: Autoviz in c:\users\richard\anaconda3\lib\site-
packages (0.1.35)
Requirement already satisfied: bokeh==2.4.2 in
c:\users\richard\anaconda3\lib\site-packages (from Autoviz) (2.4.2)
Requirement already satisfied: ipython in c:\users\richard\anaconda3\lib\site-
packages (from Autoviz) (7.22.0)
Requirement already satisfied: emoji in c:\users\richard\anaconda3\lib\site-
packages (from Autoviz) (1.6.3)
Requirement already satisfied: scikit-learn in
c:\users\richard\anaconda3\lib\site-packages (from Autoviz) (0.24.1)
Requirement already satisfied: seaborn>=0.11.1 in
c:\users\richard\anaconda3\lib\site-packages (from Autoviz) (0.11.1)
Requirement already satisfied: pandas in c:\users\richard\anaconda3\lib\site-
packages (from Autoviz) (1.2.4)
Requirement already satisfied: textblob in c:\users\richard\anaconda3\lib\site-
packages (from Autoviz) (0.17.1)
Requirement already satisfied: xlrd in c:\users\richard\anaconda3\lib\site-
packages (from Autoviz) (2.0.1)
Requirement already satisfied: statsmodels in
c:\users\richard\anaconda3\lib\site-packages (from Autoviz) (0.12.2)
Requirement already satisfied: jupyter in c:\users\richard\anaconda3\lib\site-
packages (from Autoviz) (1.0.0)
Requirement already satisfied: holoviews==1.14.6 in
c:\users\richard\anaconda3\lib\site-packages (from Autoviz) (1.14.6)
Requirement already satisfied: wordcloud in c:\users\richard\anaconda3\lib\site-
packages (from Autoviz) (1.8.1)
Requirement already satisfied: xgboost in c:\users\richard\anaconda3\lib\site-
packages (from Autoviz) (1.5.2)
Requirement already satisfied: nltk in c:\users\richard\anaconda3\lib\site-
packages (from Autoviz) (3.6.1)
Requirement already satisfied: hvplot==0.7.3 in
c:\users\richard\anaconda3\lib\site-packages (from Autoviz) (0.7.3)
Requirement already satisfied: matplotlib>=3.3.3 in
c:\users\richard\anaconda3\lib\site-packages (from Autoviz) (3.3.4)
Requirement already satisfied: panel==0.12.6 in
c:\users\richard\anaconda3\lib\site-packages (from Autoviz) (0.12.6)
Requirement already satisfied: fsspec==0.8.3 in
c:\users\richard\anaconda3\lib\site-packages (from Autoviz) (0.8.3)
Requirement already satisfied: packaging>=16.8 in
c:\users\richard\anaconda3\lib\site-packages (from bokeh==2.4.2->Autoviz) (20.9)
Requirement already satisfied: pillow>=7.1.0 in
c:\users\richard\anaconda3\lib\site-packages (from bokeh==2.4.2->Autoviz)
(8.2.0)
```

```
Requirement already satisfied: typing-extensions>=3.10.0 in
c:\users\richard\anaconda3\lib\site-packages (from bokeh==2.4.2->Autoviz)
(4.0.1)
Requirement already satisfied: PyYAML>=3.10 in
c:\users\richard\anaconda3\lib\site-packages (from bokeh==2.4.2->Autoviz)
(5.4.1)
Requirement already satisfied: Jinja2>=2.9 in
c:\users\richard\anaconda3\lib\site-packages (from bokeh==2.4.2->Autoviz)
(2.11.3)
Requirement already satisfied: tornado>=5.1 in
c:\users\richard\anaconda3\lib\site-packages (from bokeh==2.4.2->Autoviz) (6.1)
Requirement already satisfied: numpy>=1.11.3 in
c:\users\richard\anaconda3\lib\site-packages (from bokeh==2.4.2->Autoviz)
(1.20.1)
Requirement already satisfied: colorcet in c:\users\richard\anaconda3\lib\site-
packages (from holoviews==1.14.6->Autoviz) (3.0.0)
Requirement already satisfied: param<2.0,>=1.9.3 in
c:\users\richard\anaconda3\lib\site-packages (from holoviews==1.14.6->Autoviz)
(1.12.0)
Requirement already satisfied: pyviz-comms>=0.7.4 in
c:\users\richard\anaconda3\lib\site-packages (from holoviews==1.14.6->Autoviz)
(2.1.0)
Requirement already satisfied: markdown in c:\users\richard\anaconda3\lib\site-
packages (from panel==0.12.6->Autoviz) (3.3.6)
Requirement already satisfied: requests in c:\users\richard\anaconda3\lib\site-
packages (from panel==0.12.6->Autoviz) (2.25.1)
Requirement already satisfied: tqdm>=4.48.0 in
c:\users\richard\anaconda3\lib\site-packages (from panel==0.12.6->Autoviz)
(4.59.0)
Requirement already satisfied: bleach in c:\users\richard\anaconda3\lib\site-
packages (from panel==0.12.6->Autoviz) (3.3.0)
Requirement already satisfied: pyct>=0.4.4 in
c:\users\richard\anaconda3\lib\site-packages (from panel==0.12.6->Autoviz)
(0.4.8)
Requirement already satisfied: MarkupSafe>=0.23 in
c:\users\richard\anaconda3\lib\site-packages (from
Jinja2>=2.9->bokeh==2.4.2->Autoviz) (1.1.1)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in
c:\users\richard\anaconda3\lib\site-packages (from matplotlib>=3.3.3->Autoviz)
(2.4.7)
Requirement already satisfied: cycler>=0.10 in
c:\users\richard\anaconda3\lib\site-packages (from matplotlib>=3.3.3->Autoviz)
(0.10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in
c:\users\richard\anaconda3\lib\site-packages (from matplotlib>=3.3.3->Autoviz)
(1.3.1)
Requirement already satisfied: python-dateutil>=2.1 in
c:\users\richard\anaconda3\lib\site-packages (from matplotlib>=3.3.3->Autoviz)
```

```
(2.8.1)
Requirement already satisfied: six in c:\users\richard\anaconda3\lib\site-
packages (from cycler>=0.10->matplotlib>=3.3.3->Autoviz) (1.15.0)
Requirement already satisfied: pytz>=2017.3 in
c:\users\richard\anaconda3\lib\site-packages (from pandas->Autoviz) (2021.1)
Requirement already satisfied: scipy>=1.0 in
c:\users\richard\anaconda3\lib\site-packages (from seaborn>=0.11.1->Autoviz)
(1.6.2)
Requirement already satisfied: webencodings in
c:\users\richard\anaconda3\lib\site-packages (from
bleach->panel==0.12.6->Autoviz) (0.5.1)
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in
c:\users\richard\anaconda3\lib\site-packages (from ipython->Autoviz) (3.0.17)
Requirement already satisfied: decorator in c:\users\richard\anaconda3\lib\site-
packages (from ipython->Autoviz) (5.0.6)
Requirement already satisfied: pygments in c:\users\richard\anaconda3\lib\site-
packages (from ipython->Autoviz) (2.8.1)
Requirement already satisfied: jedi>=0.16 in
c:\users\richard\anaconda3\lib\site-packages (from ipython->Autoviz) (0.17.2)
Requirement already satisfied: pickleshare in
c:\users\richard\anaconda3\lib\site-packages (from ipython->Autoviz) (0.7.5)
Requirement already satisfied: setuptools>=18.5 in
c:\users\richard\anaconda3\lib\site-packages (from ipython->Autoviz)
(52.0.0.post20210125)
Requirement already satisfied: backcall in c:\users\richard\anaconda3\lib\site-
packages (from ipython->Autoviz) (0.2.0)
Requirement already satisfied: colorama in c:\users\richard\anaconda3\lib\site-
packages (from ipython->Autoviz) (0.4.4)
Requirement already satisfied: traitlets>=4.2 in
c:\users\richard\anaconda3\lib\site-packages (from ipython->Autoviz) (5.0.5)
Requirement already satisfied: parso<0.8.0,>=0.7.0 in
c:\users\richard\anaconda3\lib\site-packages (from jedi>=0.16->ipython->Autoviz)
(0.7.0)
Requirement already satisfied: wcwidth in c:\users\richard\anaconda3\lib\site-
packages (from prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0->ipython->Autoviz)
(0.2.5)
Requirement already satisfied: ipython-genutils in
c:\users\richard\anaconda3\lib\site-packages (from
traitlets>=4.2->ipython->Autoviz) (0.2.0)
Requirement already satisfied: ipywidgets in
c:\users\richard\anaconda3\lib\site-packages (from jupyter->Autoviz) (7.6.3)
Requirement already satisfied: notebook in c:\users\richard\anaconda3\lib\site-
packages (from jupyter->Autoviz) (6.3.0)
Requirement already satisfied: jupyter-console in
c:\users\richard\anaconda3\lib\site-packages (from jupyter->Autoviz) (6.4.0)
Requirement already satisfied: nbconvert in c:\users\richard\anaconda3\lib\site-
packages (from jupyter->Autoviz) (6.0.7)
Requirement already satisfied: qtconsole in c:\users\richard\anaconda3\lib\site-
```

```
packages (from jupyter->Autoviz) (5.0.3)
Requirement already satisfied: ipykernel in c:\users\richard\anaconda3\lib\site-
packages (from jupyter->Autoviz) (5.3.4)
Requirement already satisfied: jupyter-client in
c:\users\richard\anaconda3\lib\site-packages (from ipykernel->jupyter->Autoviz)
(6.1.12)
Requirement already satisfied: widgetsnbextension~=3.5.0 in
c:\users\richard\anaconda3\lib\site-packages (from ipywidgets->jupyter->Autoviz)
(3.5.1)
Requirement already satisfied: nbformat>=4.2.0 in
c:\users\richard\anaconda3\lib\site-packages (from ipywidgets->jupyter->Autoviz)
(5.1.3)
Requirement already satisfied: jupyterlab-widgets>=1.0.0 in
c:\users\richard\anaconda3\lib\site-packages (from ipywidgets->jupyter->Autoviz)
(1.0.0)
Requirement already satisfied: jsonschema!=2.5.0,>=2.4 in
c:\users\richard\anaconda3\lib\site-packages (from
nbformat>=4.2.0->ipywidgets->jupyter->Autoviz) (3.2.0)
Requirement already satisfied: jupyter-core in
c:\users\richard\anaconda3\lib\site-packages (from
nbformat>=4.2.0->ipywidgets->jupyter->Autoviz) (4.7.1)
Requirement already satisfied: pyrsistent>=0.14.0 in
c:\users\richard\anaconda3\lib\site-packages (from
jsonschema!=2.5.0,>=2.4->nbformat>=4.2.0->ipywidgets->jupyter->Autoviz) (0.17.3)
Requirement already satisfied: attrs>=17.4.0 in
c:\users\richard\anaconda3\lib\site-packages (from
jsonschema!=2.5.0,>=2.4->nbformat>=4.2.0->ipywidgets->jupyter->Autoviz) (20.3.0)
Requirement already satisfied: Send2Trash>=1.5.0 in
c:\users\richard\anaconda3\lib\site-packages (from notebook->jupyter->Autoviz)
(1.5.0)
Requirement already satisfied: argon2-cffi in
c:\users\richard\anaconda3\lib\site-packages (from notebook->jupyter->Autoviz)
(20.1.0)
Requirement already satisfied: terminado>=0.8.3 in
c:\users\richard\anaconda3\lib\site-packages (from notebook->jupyter->Autoviz)
(0.9.4)
Requirement already satisfied: pyzmq>=17 in c:\users\richard\anaconda3\lib\site-
packages (from notebook->jupyter->Autoviz) (20.0.0)
Requirement already satisfied: prometheus-client in
c:\users\richard\anaconda3\lib\site-packages (from notebook->jupyter->Autoviz)
(0.10.1)
Requirement already satisfied: pywin32>=1.0 in
c:\users\richard\anaconda3\lib\site-packages (from jupyter-
core->nbformat>=4.2.0->ipywidgets->jupyter->Autoviz) (227)
Requirement already satisfied: pywinpty>=0.5 in
c:\users\richard\anaconda3\lib\site-packages (from
terminado>=0.8.3->notebook->jupyter->Autoviz) (0.5.7)
Requirement already satisfied: cffi>=1.0.0 in
```

```
c:\users\richard\anaconda3\lib\site-packages (from
argon2-cffi->notebook->jupyter->Autoviz) (1.14.5)
Requirement already satisfied: pycparser in c:\users\richard\anaconda3\lib\site-
packages (from cffi>=1.0.0->argon2-cffi->notebook->jupyter->Autoviz) (2.20)
Requirement already satisfied: importlib-metadata>=4.4 in
c:\users\richard\anaconda3\lib\site-packages (from
markdown->panel==0.12.6->Autoviz) (4.10.1)
Requirement already satisfied: zipp>=0.5 in c:\users\richard\anaconda3\lib\site-
packages (from importlib-metadata>=4.4->markdown->panel==0.12.6->Autoviz)
(3.4.1)
Requirement already satisfied: pandocfilters>=1.4.1 in
c:\users\richard\anaconda3\lib\site-packages (from nbconvert->jupyter->Autoviz)
(1.4.3)
Requirement already satisfied: mistune<2,>=0.8.1 in
c:\users\richard\anaconda3\lib\site-packages (from nbconvert->jupyter->Autoviz)
(0.8.4)
Requirement already satisfied: testpath in c:\users\richard\anaconda3\lib\site-
packages (from nbconvert->jupyter->Autoviz) (0.4.4)
Requirement already satisfied: nbclient<0.6.0,>=0.5.0 in
c:\users\richard\anaconda3\lib\site-packages (from nbconvert->jupyter->Autoviz)
(0.5.3)
Requirement already satisfied: defusedxml in
c:\users\richard\anaconda3\lib\site-packages (from nbconvert->jupyter->Autoviz)
(0.7.1)
Requirement already satisfied: entrypoints>=0.2.2 in
c:\users\richard\anaconda3\lib\site-packages (from nbconvert->jupyter->Autoviz)
(0.3)
Requirement already satisfied: jupyterlab-pygments in
c:\users\richard\anaconda3\lib\site-packages (from nbconvert->jupyter->Autoviz)
(0.1.2)
Requirement already satisfied: async-generator in
c:\users\richard\anaconda3\lib\site-packages (from
nbclient<0.6.0,>=0.5.0->nbconvert->jupyter->Autoviz) (1.10)
Requirement already satisfied: nest-asyncio in
c:\users\richard\anaconda3\lib\site-packages (from
nbclient<0.6.0,>=0.5.0->nbconvert->jupyter->Autoviz) (1.5.1)
Requirement already satisfied: regex in c:\users\richard\anaconda3\lib\site-
packages (from nltk->Autoviz) (2021.4.4)
Requirement already satisfied: click in c:\users\richard\anaconda3\lib\site-
packages (from nltk->Autoviz) (7.1.2)
Requirement already satisfied: joblib in c:\users\richard\anaconda3\lib\site-
packages (from nltk->Autoviz) (1.0.1)
Requirement already satisfied: qtpy in c:\users\richard\anaconda3\lib\site-
packages (from qtconsole->jupyter->Autoviz) (1.9.0)
Requirement already satisfied: chardet<5,>=3.0.2 in
c:\users\richard\anaconda3\lib\site-packages (from
requests->panel==0.12.6->Autoviz) (4.0.0)
Requirement already satisfied: idna<3,>=2.5 in
```

```
c:\users\richard\anaconda3\lib\site-packages (from
requests->panel==0.12.6->Autoviz) (2.10)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in
c:\users\richard\anaconda3\lib\site-packages (from
requests->panel==0.12.6->Autoviz) (1.26.4)
Requirement already satisfied: certifi>=2017.4.17 in
c:\users\richard\anaconda3\lib\site-packages (from
requests->panel==0.12.6->Autoviz) (2020.12.5)
Requirement already satisfied: threadpoolctl>=2.0.0 in
c:\users\richard\anaconda3\lib\site-packages (from scikit-learn->Autoviz)
(2.1.0)
Requirement already satisfied: patsy>=0.5 in
c:\users\richard\anaconda3\lib\site-packages (from statsmodels->Autoviz) (0.5.1)
```

[18]:
```
! pip install xlrd
```

```
Requirement already satisfied: xlrd in c:\users\richard\anaconda3\lib\site-
packages (2.0.1)
```

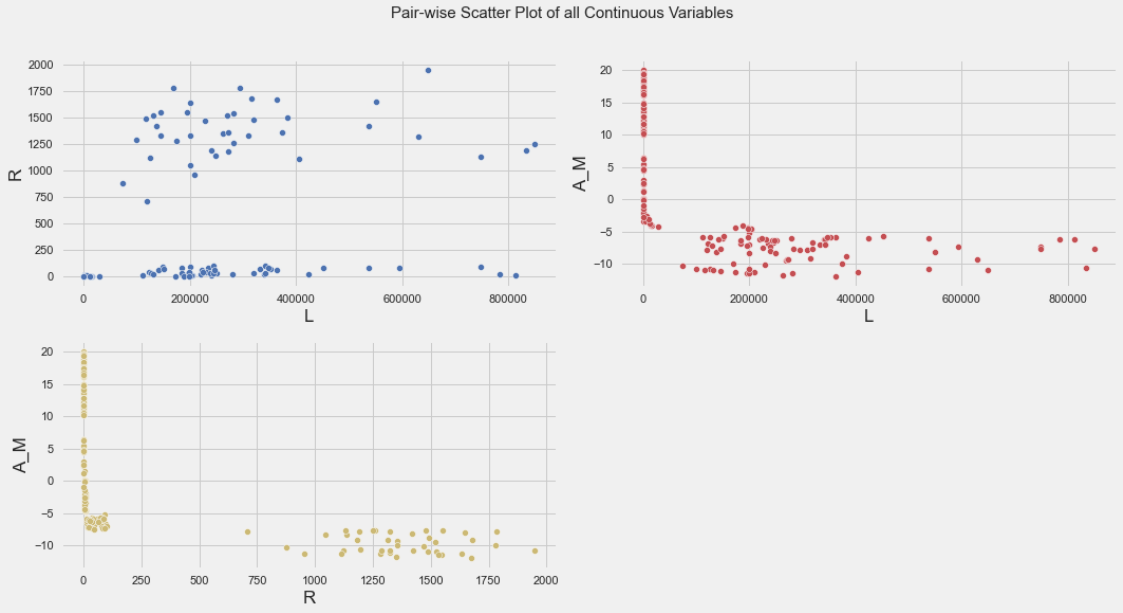
[19]:
```
pip install typing_extensions
```

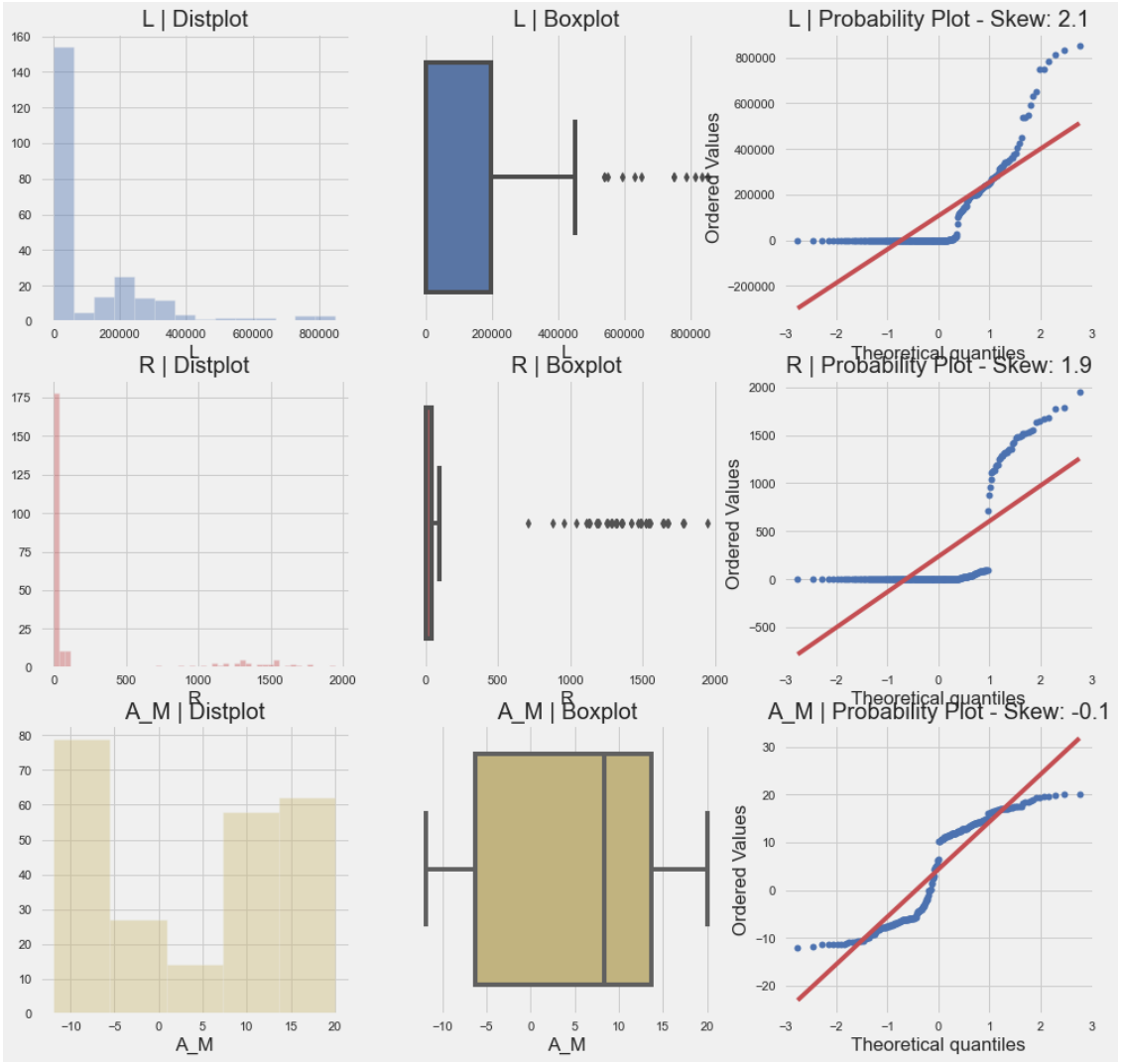
```
Requirement already satisfied: typing_extensions in
c:\users\richard\anaconda3\lib\site-packages (4.0.1)
Note: you may need to restart the kernel to use updated packages.
```

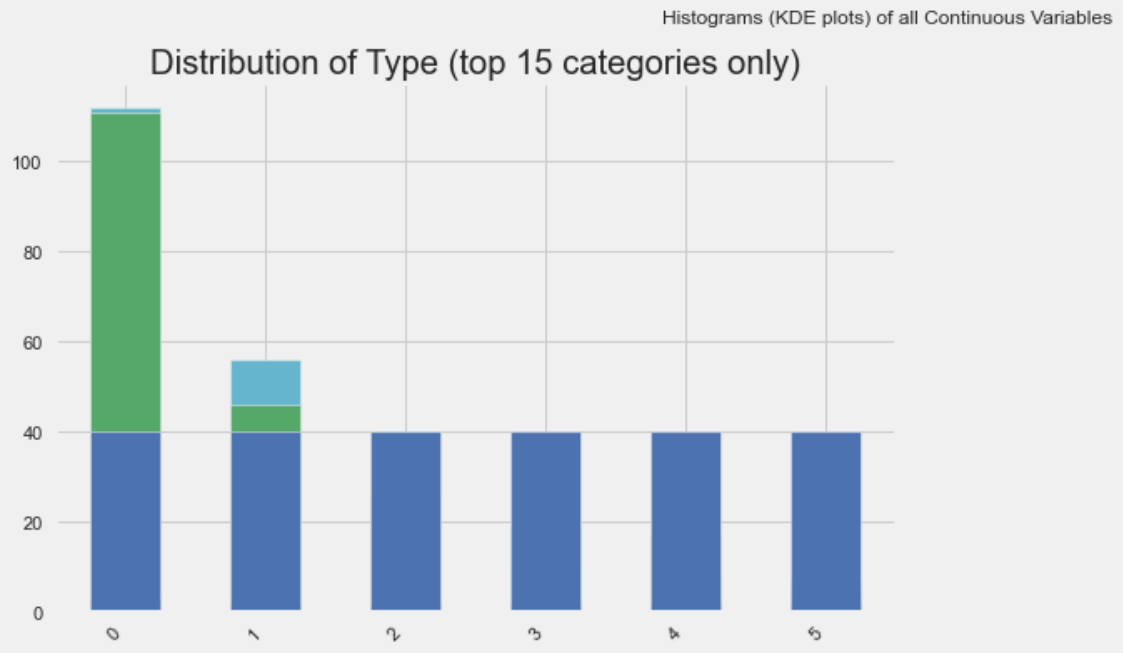
[20]:
```
from autoviz.AutoViz_Class import AutoViz_Class
AV = AutoViz_Class()
df_av = AV.AutoViz('Stars.csv')
```

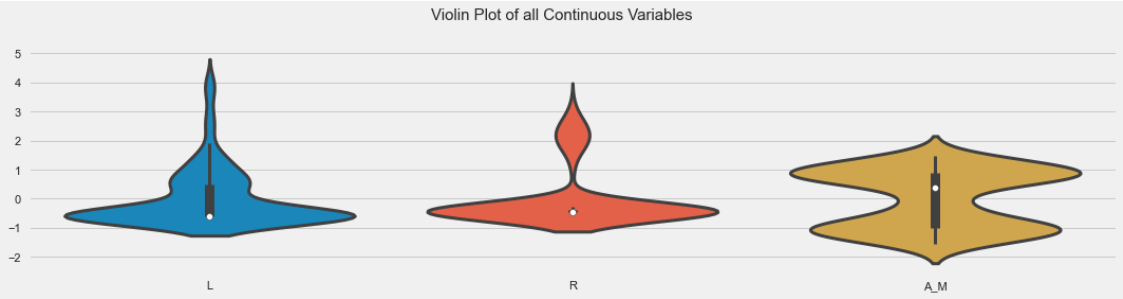
```
Alert! from autoviz version 0.1.35, after importing, you must %matplotlib inline
to display charts in Jupyter Notebooks.
    AV = AutoViz_Class()
    AV.AutoViz(filename, sep=',', depVar='', dfte=None, header=0, verbose=0,
lowess=False,
               chart_format='svg',max_rows_analyzed=150000,max_cols_analyzed=30,
save_plot_dir=None)
Note: verbose=0 or 1 generates charts and displays them in your local Jupyter
notebook.
      verbose=2 does not display plots but saves them in AutoViz_Plots folder in
local machine.
Updated: chart_format='bokeh' generates and displays charts in your local
Jupyter notebook.
      chart_format='server' generates and displays charts in the browser - one
tab for each chart.
      chart_format='html' silently saves charts HTML format - they are also
interactive!

Shape of your Data Set loaded: (240, 7)
############## C L A S S I F Y I N G  V A R I A B L E S  ####################
```

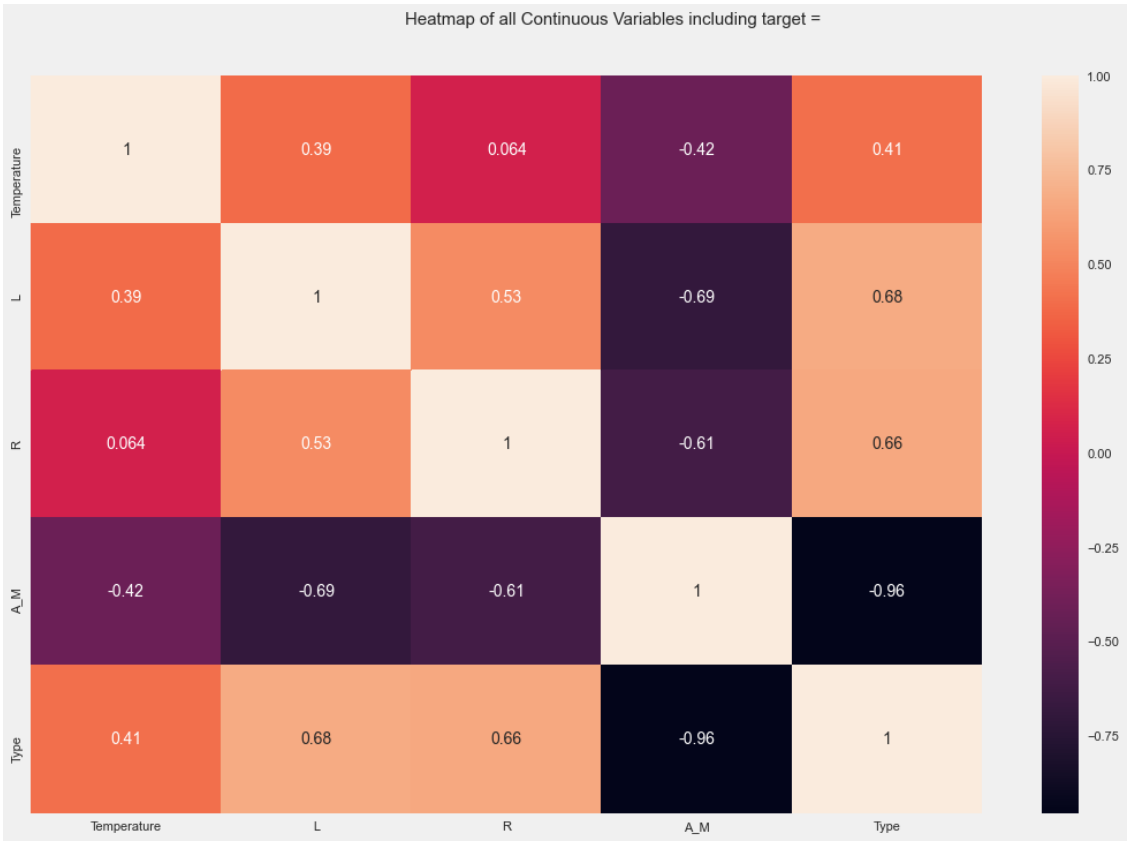
```
Classifying variables in data set…
    7 Predictors classified…
        No variables removed since no ID or low-information variables found in
data set
Number of All Scatter Plots = 6
Time to run AutoViz = 1 seconds

 ######################## AUTO VISUALIZATION Completed ########################
```

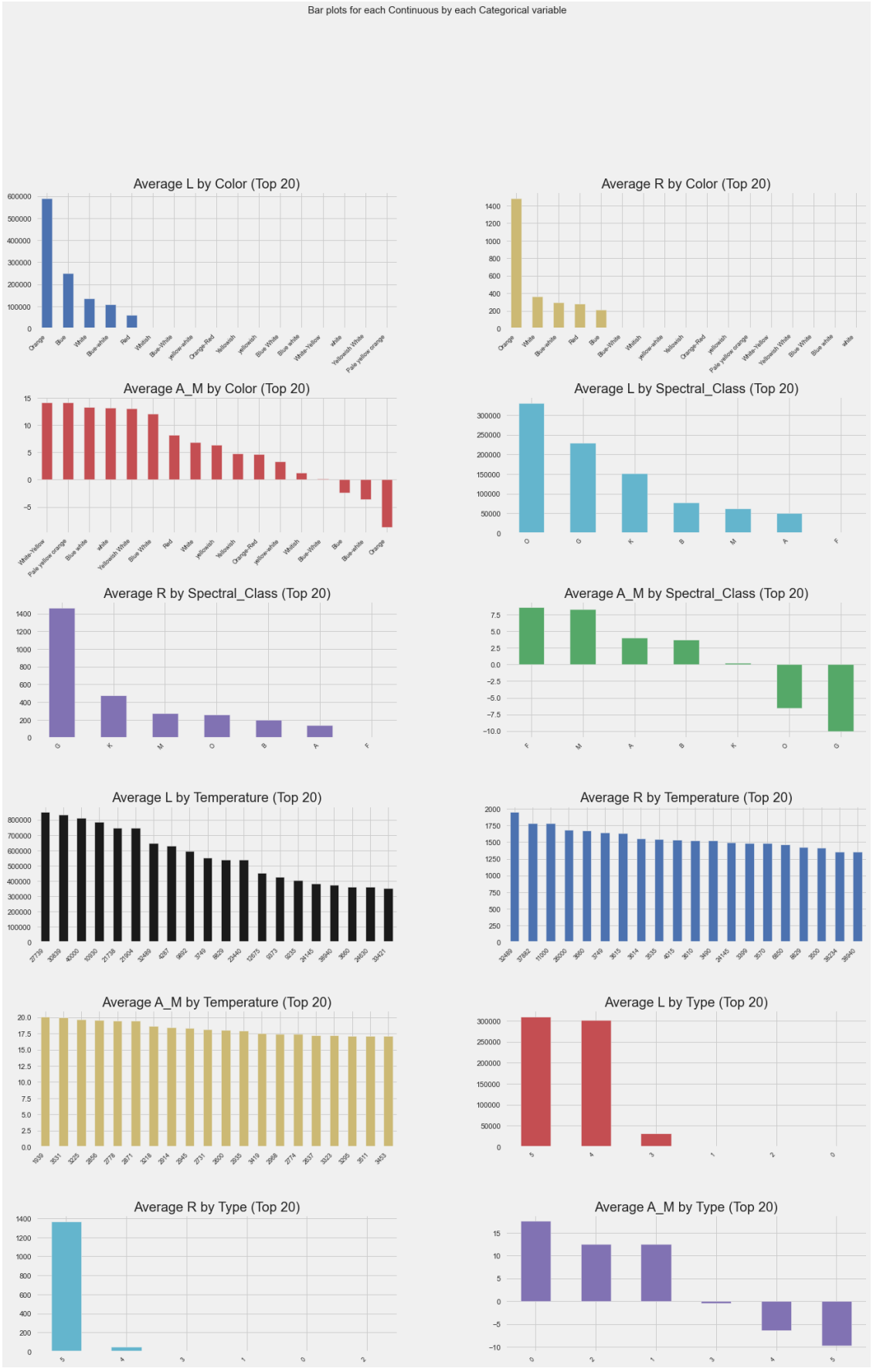



L | Distplot
L | Boxplot
L | Probability Plot - Skew: 2.1
Ordered Values
Theoretical quantiles
R | Distplot
R | Boxplot
R | Probability Plot - Skew: 1.9
Ordered Values
Theoretical quantiles
A_M | Distplot
A_M | Boxplot
A_M | Probability Plot - Skew: -0.1
Ordered Values
Theoretical quantiles


Histograms (KDE plots) of all Continuous Variables
Distribution of Type (top 15 categories only)
100
80
60
40
20
0
0
1
2
3
4
5


Violin Plot of all Continuous Variables
5
4
3
2
1
0
−1
−2
L
R
A_M


Heatmap of all Continuous Variables including target =
Temperature
L
R
A_M
Type
1
0.39
0.064
-0.42
0.41
0.39
1
0.53
-0.69
0.68
0.064
0.53
1
-0.61
0.66
-0.42
-0.69
-0.61
1
-0.96
0.41
0.68
0.66
-0.96
1
Temperature
L
R
A_M
Type
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75


Bar plots for each Continuous by each Categorical variable
Average L by Color (Top 20)
Average R by Color (Top 20)
Average A_M by Color (Top 20)
Average L by Spectral_Class (Top 20)
Average R by Spectral_Class (Top 20)
Average A_M by Spectral_Class (Top 20)
Average L by Temperature (Top 20)
Average R by Temperature (Top 20)
Average A_M by Temperature (Top 20)
Average L by Type (Top 20)
Average R by Type (Top 20)
Average A_M by Type (Top 20)

5 Basic Data Cleaning

[21]:
```
df['Color'].value_counts()
```

[21]:
```
Red                   112
Blue                   56
Blue-white             26
Blue White             10
yellow-white            8
White                   7
Blue white              4
Yellowish White         3
white                   3
yellowish               2
Whitish                 2
Orange                  2
White-Yellow            1
Orange-Red              1
Pale yellow orange      1
Blue-White              1
Yellowish               1
Name: Color, dtype: int64
```

[22]:
```
a = ['Blue White','Blue white','Blue-White','white','Whitish','Yellowish␣
 ↪White','yellowish','Yellowish','White-Yellow','Pale yellow␣
 ↪orange','Orange-Red']
for i in range(len(df['Color'])):
    if df['Color'][i] in a[:3]:
        df['Color'][i] = 'Blue-white'
    elif df['Color'][i] in a[3:5]:
        df['Color'][i] = 'White'
    elif df['Color'][i] in a[5:9]:
        df['Color'][i] = 'yellow-white'
    elif df['Color'][i] in a[9:]:
        df['Color'][i] = 'Orange'
```

6 Data Visualisation

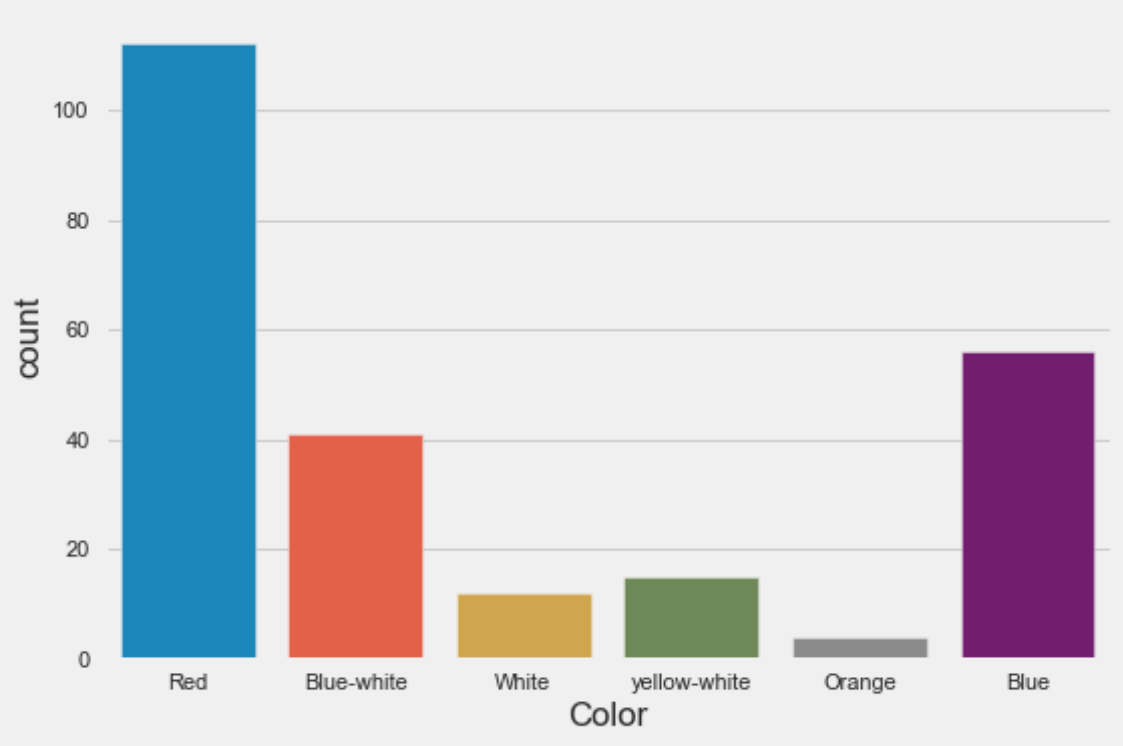
6.1 To show more realations in data I did around 200 visualisations

6.1.1 Main idea behid this is first did visualisation for every color of stars.Then, seperated each color into one dataframe. for example,I created a dataframe which has only blue colored stars like wise did to every color. Next, In blue color classified according to spectral classes and did visualistaion on them using bar and histplot and also general visualisations using heat map,pair plot,scatter plot,box plot,vilolin plot etc..

[23]:
```
df['Color'].value_counts()
```

[23]:
```
Red             112
Blue             56
Blue-white       41
yellow-white     15
White            12
Orange            4
Name: Color, dtype: int64
```

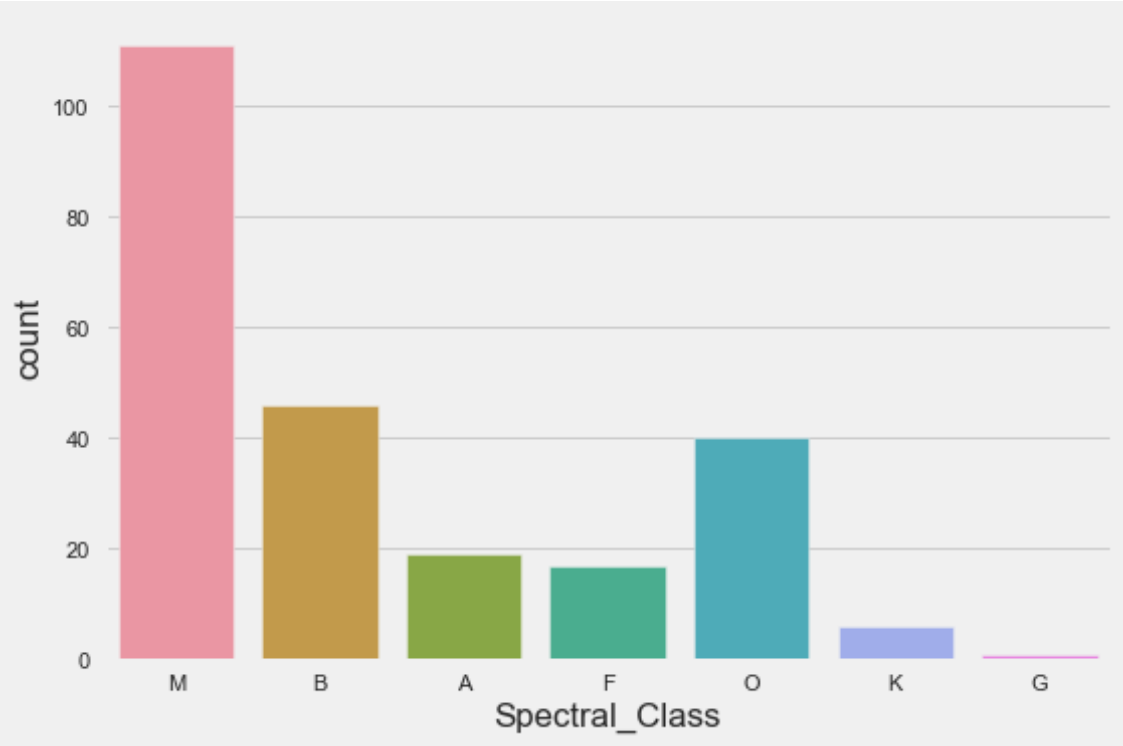
[24]:
```
sns.countplot(x = 'Color',data = df)
plt.show()
```



[25]:
```
df['Spectral_Class'].value_counts()
```

[25]:
```
M    111
B     46
O     40
A     19
F     17
K      6
G      1
Name: Spectral_Class, dtype: int64
```

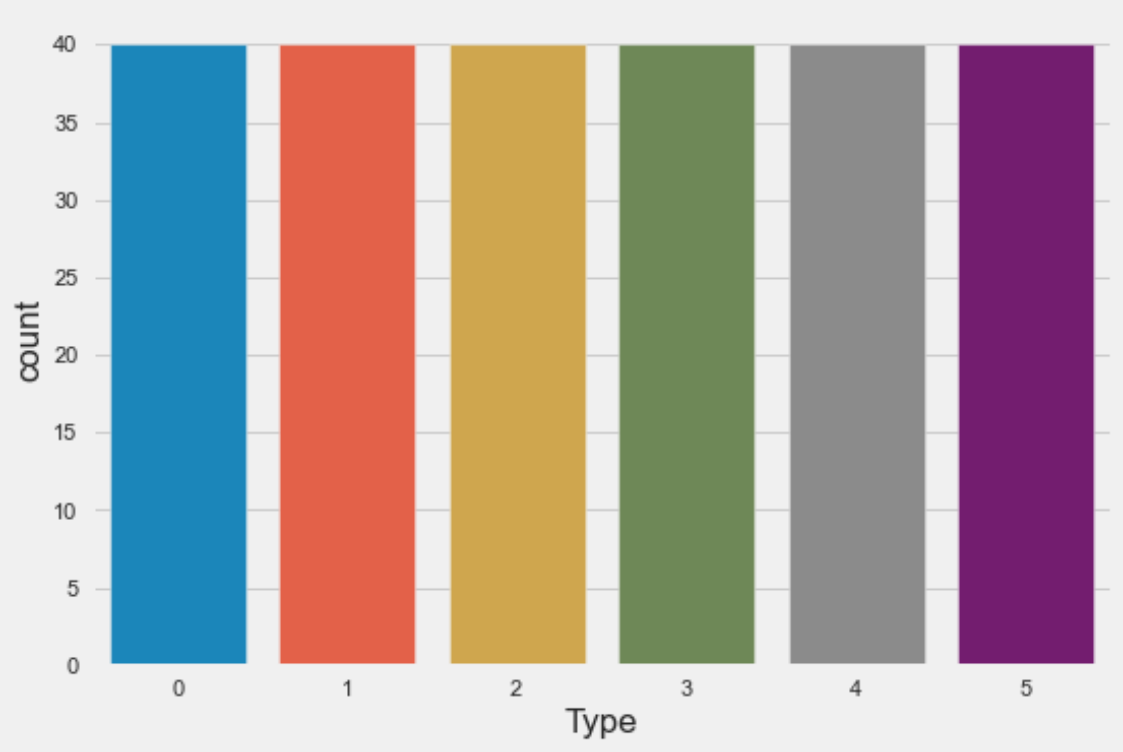
[26]:
```
sns.countplot(x='Spectral_Class', data = df)
plt.show()
```



[27]:
```
df['Type'].value_counts()
```

[27]:
```
0    40
1    40
2    40
3    40
4    40
5    40
Name: Type, dtype: int64
```

[28]:
```
sns.countplot(x='Type', data = df)
plt.show()
```



[29]:
```
fig = px.histogram(df, 'Temperature',
                   color='Color',
                   title="<b>Average Temparature by color</b>")

fig.add_vline(x=df['Temperature'].mean(), line_width=2, line_dash="dash",␣
↪line_color="black")

fig.show()

# all stars with red color are below 5000 temparture
# only 6 blue stars outof 56 are lessthan 10k temparature
# only 6 blue white stars ouof 41 are lessthan 10k temparature
# A Blue Star has highest temparature among all
```

[30]:
```
fig = px.histogram(df, 'L',
                   color='Color',
                   title="<b>Average Relative Luminocity by color</b>")

fig.add_vline(x=df['L'].mean(), line_width=2, line_dash="dash",␣
↪line_color="black")
```

```
fig.show()

# around 160 stars outof 240 have less than 100k  Relative Luminocity
# A blue and a red star has greather Relative Luminocity among all
```

[31]:
```
fig = px.histogram(df, 'R',
                   color='Color',
                   title="<b>Average Relative Radius by color</b>")

fig.add_vline(x=df['R'].mean(), line_width=2, line_dash="dash",␣
 ↪line_color="black")

fig.show()

# almost 200 outof 240 stars have relative radius between 0-1
# A blue star is bigger among all
```

[32]:
```
fig = px.histogram(df, 'A_M',
                   color='Color',
                   title="<b>Average Absolute Magnitude by color</b>")

fig.add_vline(x=df['A_M'].mean(), line_width=2, line_dash="dash",␣
 ↪line_color="black")

fig.show()
# Couple of red stars has highest Absolute Magnitude am
```

[33]:
```
fig = px.histogram(df, 'Temperature',
                   color='Spectral_Class',
                   title="<b>Average Temparature by Spectral_Class</b>")

fig.add_vline(x=df['Temperature'].mean(), line_width=2, line_dash="dash",␣
 ↪line_color="black")

fig.show()

# all stars with red color are below 5000 temparture
# only 6 blue stars outof 56 are lessthan 10k temparature
# only 6 blue white stars ouof 41 are lessthan 10k temparature
# A Blue Star has highest temparature among all
```

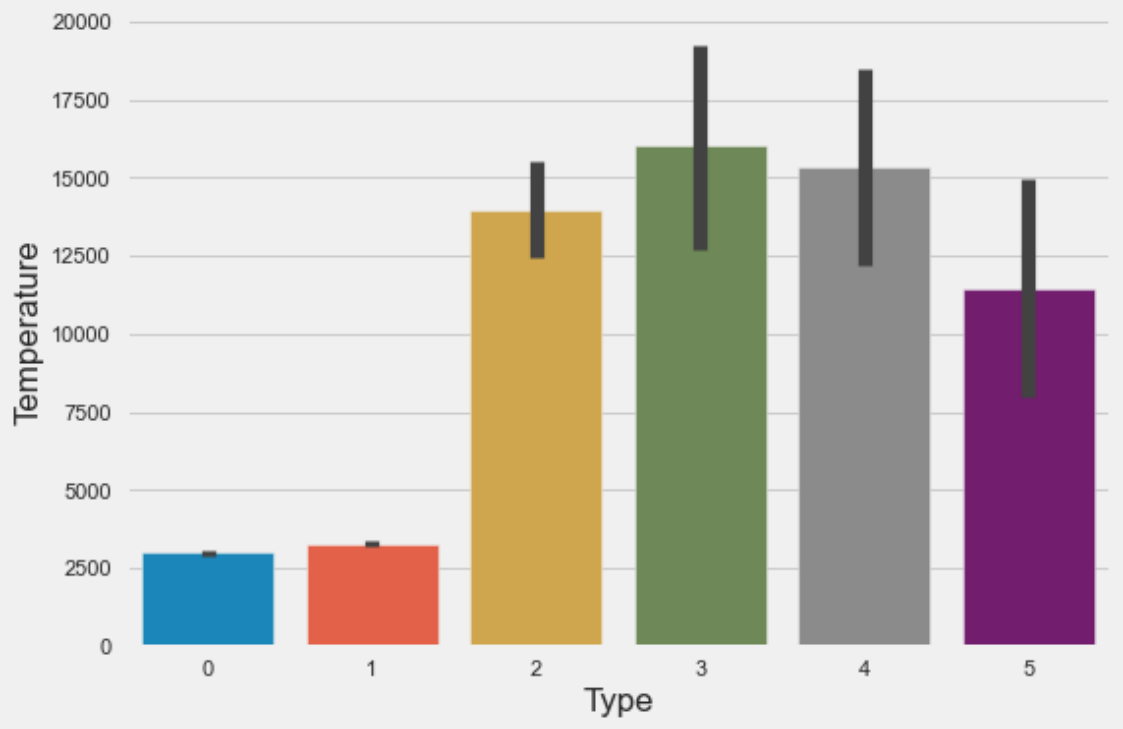
[34]:
```
fig = px.histogram(df, 'L',
                   color='Spectral_Class',
                   title="<b>Average Spectral_Class by color</b>")
```

```
fig.add_vline(x=df['L'].mean(), line_width=2, line_dash="dash",␣
 ↪line_color="black")

fig.show()
```

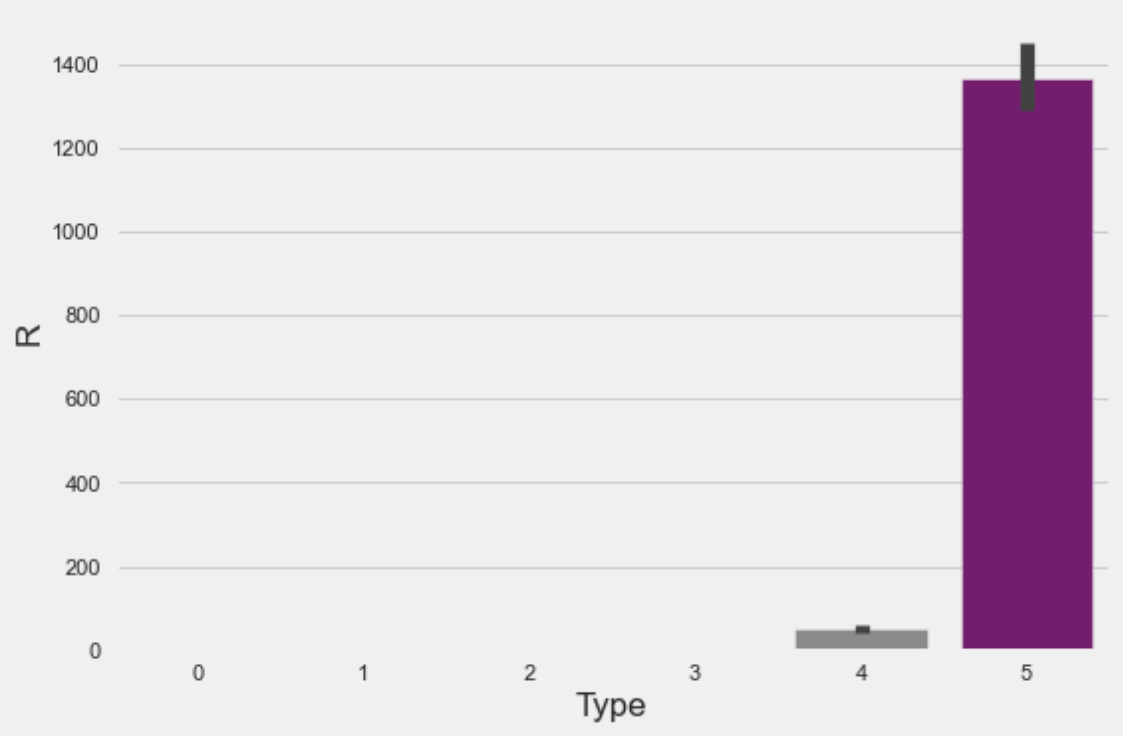
[35]:
```
fig = px.histogram(df, 'R',
                   color='Spectral_Class',
                   title="<b>Average Spectral_Class by color</b>")

fig.add_vline(x=df['R'].mean(), line_width=2, line_dash="dash",␣
 ↪line_color="black")

fig.show()
```

[36]:
```
fig = px.histogram(df, 'A_M',
                   color='Spectral_Class',
                   title="<b>Average Spectral_Class by color</b>")

fig.add_vline(x=df['A_M'].mean(), line_width=2, line_dash="dash",␣
 ↪line_color="black")

fig.show()
```

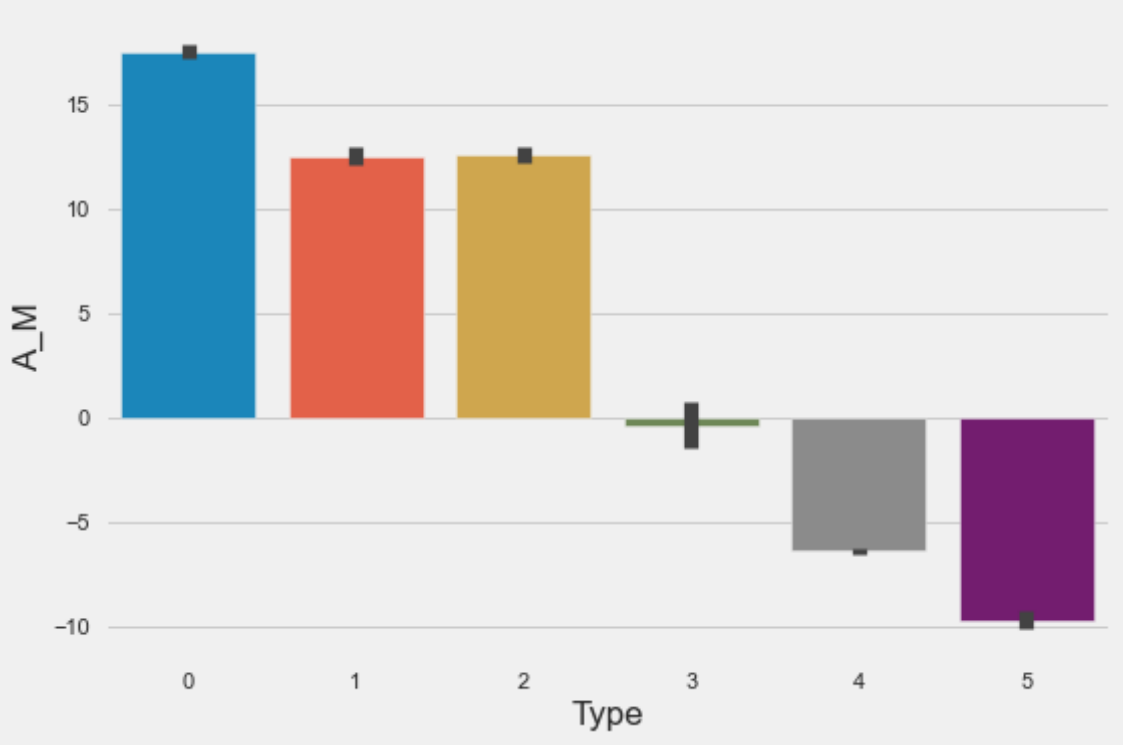
[37]:
```
sns.barplot(data = df,x = 'Type',y ='Temperature')
plt.show()
```



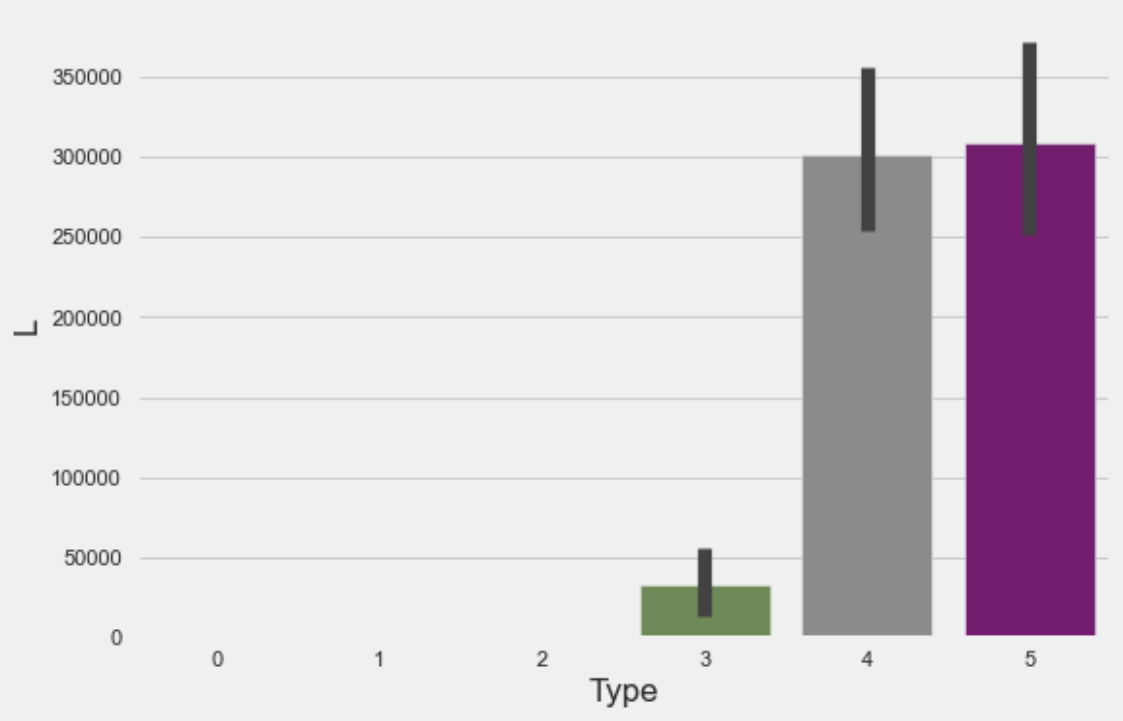
[38]:
```
sns.barplot(data = df,x = 'Type',y ='R')
plt.show()
```



[39]:
```
sns.barplot(data = df,x = 'Type',y ='A_M')
plt.show()
```



[40]:
```
sns.barplot(data = df,x = 'Type',y ='L')
plt.show()
```



[41]:
```
blue_df = df[df['Color'] == 'Blue']
blue_df# only blue color stars
```

[41]:

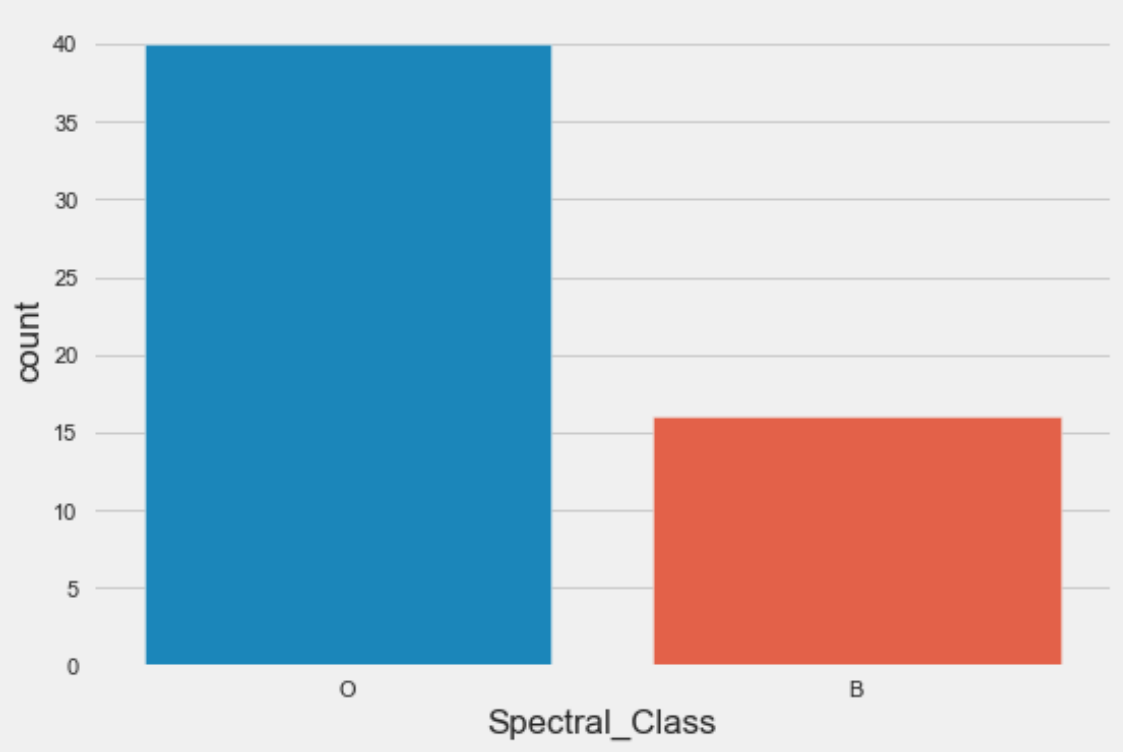
	Temperature	L	R	A_M	Color	Spectral_Class	Type
30	39000	204000.00000	10.60000	-4.700	Blue	O	3
49	33750	220000.00000	26.00000	-6.100	Blue	B	4
89	19860	0.00110	0.01310	11.340	Blue	B	2
99	36108	198000.00000	10.20000	-4.400	Blue	O	3
100	33300	240000.00000	12.00000	-6.500	Blue	B	4
101	40000	813000.00000	14.00000	-6.230	Blue	O	4
102	23000	127000.00000	36.00000	-5.760	Blue	O	4
103	17120	235000.00000	83.00000	-6.890	Blue	O	4
104	11096	112000.00000	12.00000	-5.910	Blue	O	4
105	14245	231000.00000	42.00000	-6.120	Blue	O	4
106	24630	363000.00000	63.00000	-5.830	Blue	O	4
107	12893	184000.00000	36.00000	-6.340	Blue	O	4
108	24345	142000.00000	57.00000	-6.240	Blue	O	4
109	33421	352000.00000	67.00000	-5.790	Blue	O	4
141	21020	0.00150	0.01120	11.520	Blue	B	2
142	18290	0.00130	0.00934	12.780	Blue	B	2
159	37800	202900.00000	6.86000	-4.560	Blue	O	3
160	25390	223000.00000	57.00000	-5.920	Blue	O	4
161	11567	251000.00000	36.00000	-6.245	Blue	O	4
162	12675	452000.00000	83.00000	-5.620	Blue	O	4
163	5752	245000.00000	97.00000	-6.630	Blue	O	4
164	8927	239000.00000	35.00000	-7.340	Blue	O	4
165	7282	131000.00000	24.00000	-7.220	Blue	O	4
166	19923	152000.00000	73.00000	-5.690	Blue	O	4
167	26373	198000.00000	39.00000	-5.830	Blue	O	4
168	17383	342900.00000	30.00000	-6.090	Blue	O	4
169	9373	424520.00000	24.00000	-5.990	Blue	O	4
173	26000	316000.00000	1679.00000	-9.100	Blue	B	5
176	18000	200000.00000	1045.00000	-8.300	Blue	O	5
200	16790	0.00140	0.01210	12.870	Blue	B	2
201	15680	0.00122	0.01140	11.920	Blue	B	2
202	14982	0.00118	0.01130	12.230	Blue	B	2
203	13340	0.00109	0.01160	12.900	Blue	B	2
204	18340	0.00134	0.01240	11.220	Blue	B	2
205	19920	0.00156	0.01420	11.340	Blue	B	2
206	24020	0.00159	0.01270	10.550	Blue	B	2
207	23092	0.00132	0.01040	10.180	Blue	B	2
208	17920	0.00111	0.01060	11.660	Blue	B	2
209	19360	0.00125	0.00998	11.620	Blue	B	2
214	34190	198200.00000	6.39000	-4.570	Blue	O	3
215	32460	173800.00000	6.23700	-4.360	Blue	O	3

220	23678	244290.00000	35.00000	-6.270	Blue	0	4
221	12749	332520.00000	76.00000	-7.020	Blue	0	4
222	9383	342940.00000	98.00000	-6.980	Blue	0	4
223	23440	537430.00000	81.00000	-5.975	Blue	0	4
224	16787	246730.00000	62.00000	-6.350	Blue	0	4
225	18734	224780.00000	46.00000	-7.450	Blue	0	4
226	9892	593900.00000	80.00000	-7.262	Blue	0	4
227	10930	783930.00000	25.00000	-6.224	Blue	0	4
228	23095	347820.00000	86.00000	-5.905	Blue	0	4
229	21738	748890.00000	92.00000	-7.346	Blue	0	4
231	38234	272830.00000	1356.00000	-9.290	Blue	0	5
232	32489	648430.00000	1948.50000	-10.840	Blue	0	5
235	38940	374830.00000	1356.00000	-9.930	Blue	0	5
236	30839	834042.00000	1194.00000	-10.630	Blue	0	5
239	37882	294903.00000	1783.00000	-7.800	Blue	0	5

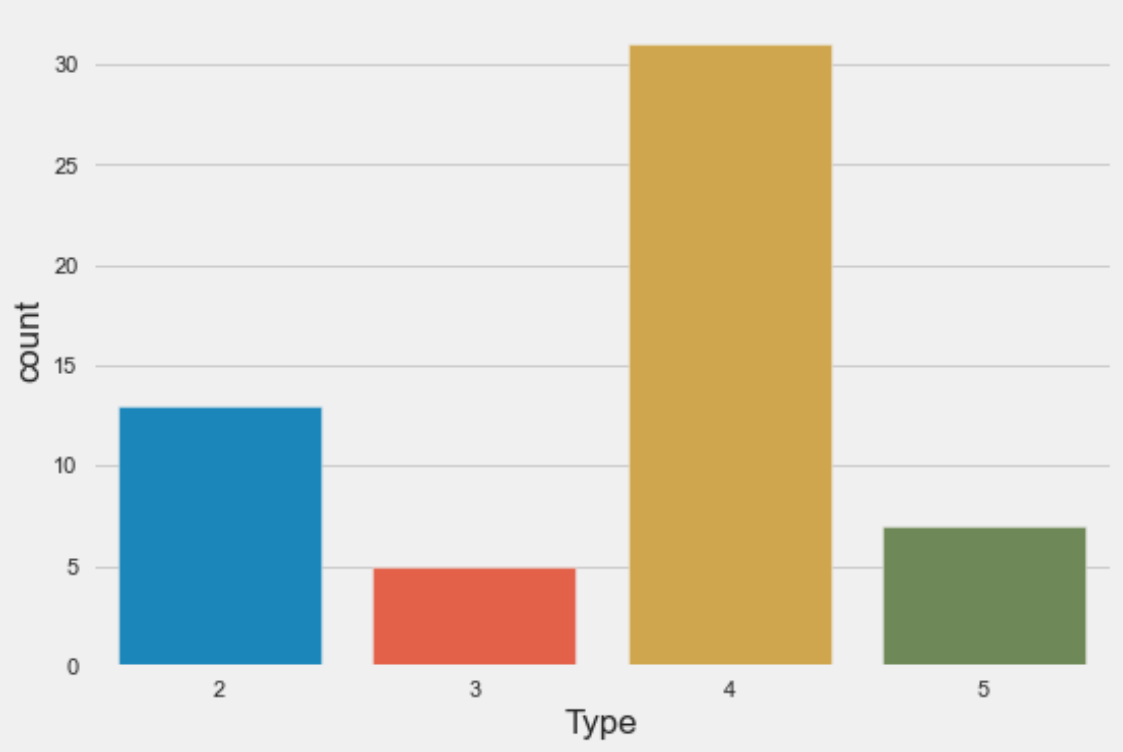
[42]:
```
blue_df['Spectral_Class'].value_counts()
```

[42]:
```
O    40
B    16
Name: Spectral_Class, dtype: int64
```

[43]:
```
sns.countplot(x = 'Spectral_Class',data = blue_df)
plt.show()

# stars with color blue have spectral_class O
```



[44]:
```
sns.countplot(x = 'Type',data = blue_df)
plt.show()

# In blue color stars most of them were type 4
```



[45]:
```
blue_df.sort_values('Temperature').tail(10)

# In blue color stars the star with highest temperature is from Spectral_class␣
↪O and of type 4
```

[45]:

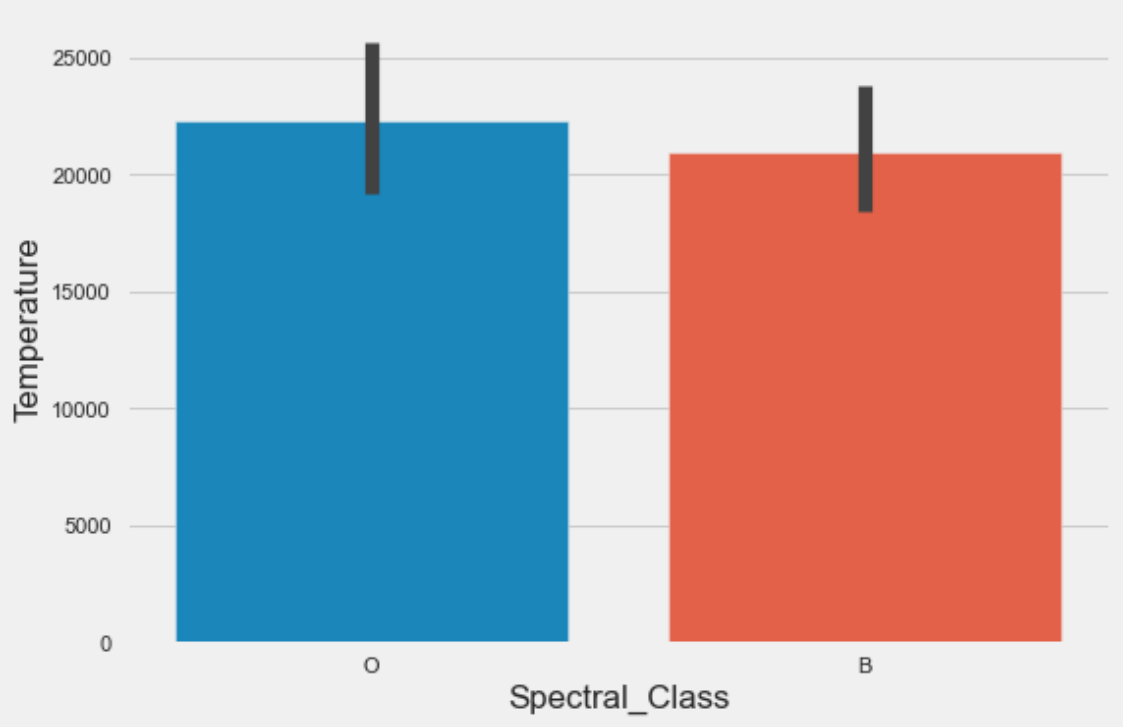
	Temperature	L	R	A_M	Color	Spectral_Class	Type
109	33421	352000.0	67.00	-5.79	Blue	O	4
49	33750	220000.0	26.00	-6.10	Blue	B	4
214	34190	198200.0	6.39	-4.57	Blue	O	3
99	36108	198000.0	10.20	-4.40	Blue	O	3
159	37800	202900.0	6.86	-4.56	Blue	O	3
239	37882	294903.0	1783.00	-7.80	Blue	O	5
231	38234	272830.0	1356.00	-9.29	Blue	O	5
235	38940	374830.0	1356.00	-9.93	Blue	O	5
30	39000	204000.0	10.60	-4.70	Blue	O	3
101	40000	813000.0	14.00	-6.23	Blue	O	4

[46]:
```
blue_df.sort_values('Temperature').head()

# In blue color stars the star with lowest temperature is from Spectral_class O␣
↪and of type 4
```

[46]:

	Temperature	L	R	A_M	Color	Spectral_Class	Type
163	5752	245000.0	97.0	-6.63	Blue	O	4
165	7282	131000.0	24.0	-7.22	Blue	O	4
164	8927	239000.0	35.0	-7.34	Blue	O	4
169	9373	424520.0	24.0	-5.99	Blue	O	4
222	9383	342940.0	98.0	-6.98	Blue	O	4

[47]:
```
sns.barplot(x = 'Spectral_Class',y = 'Temperature',data = blue_df)
plt.show()
```



[48]:
```
fig = px.histogram(blue_df, 'Temperature',
                   color = 'Type')
fig.show()

# In blue color stars type 4 is in almost every temperature(except 35k-40k)
# In blue color stars type 2 varies between 10k - 25k temperature(most of them␣
 ↪between 15k-20k)
# In blue color stars type 5 having high temparature from 25k - 40k
# There is no type 0 and type 1 in blue coloured stars
```

[49]:
```
fig = px.histogram(blue_df, 'L',
                   color = 'Type')
fig.show()
```

[50]:
```
fig = px.histogram(blue_df, 'L',
                   color = 'Spectral_Class')
fig.show()
```

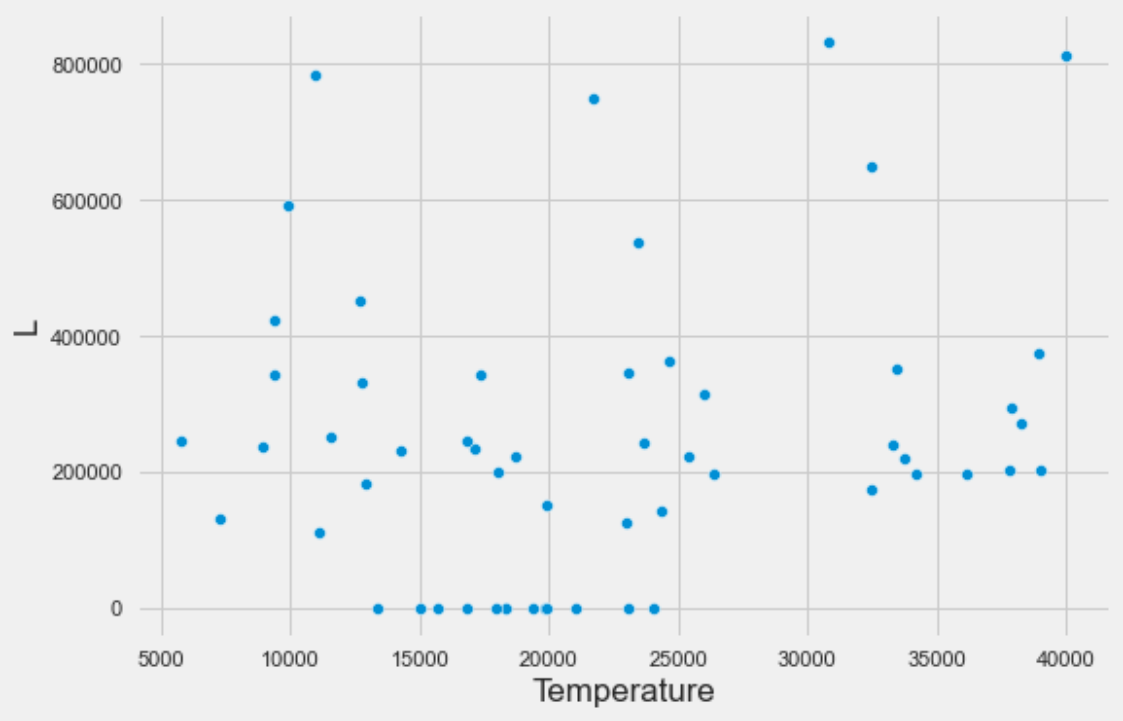
[51]:
```
fig = px.histogram(blue_df, 'R',
                   color = 'Type')
fig.show()
```

[52]:
```
fig = px.histogram(blue_df, 'R',
                   color = 'Spectral_Class')
fig.show()
```

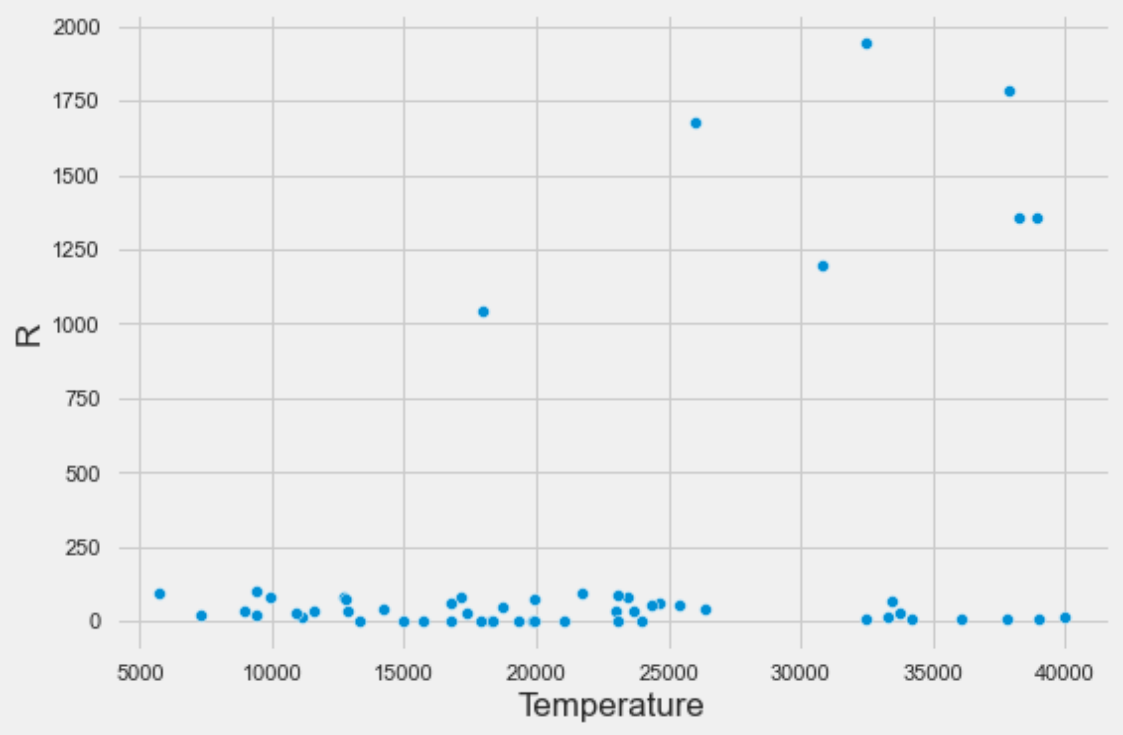
[53]:
```
fig = px.histogram(blue_df, 'A_M',
                   color = 'Type')
fig.show()
```

[54]:
```
fig = px.histogram(blue_df, 'A_M',
                   color = 'Spectral_Class')
fig.show()
```

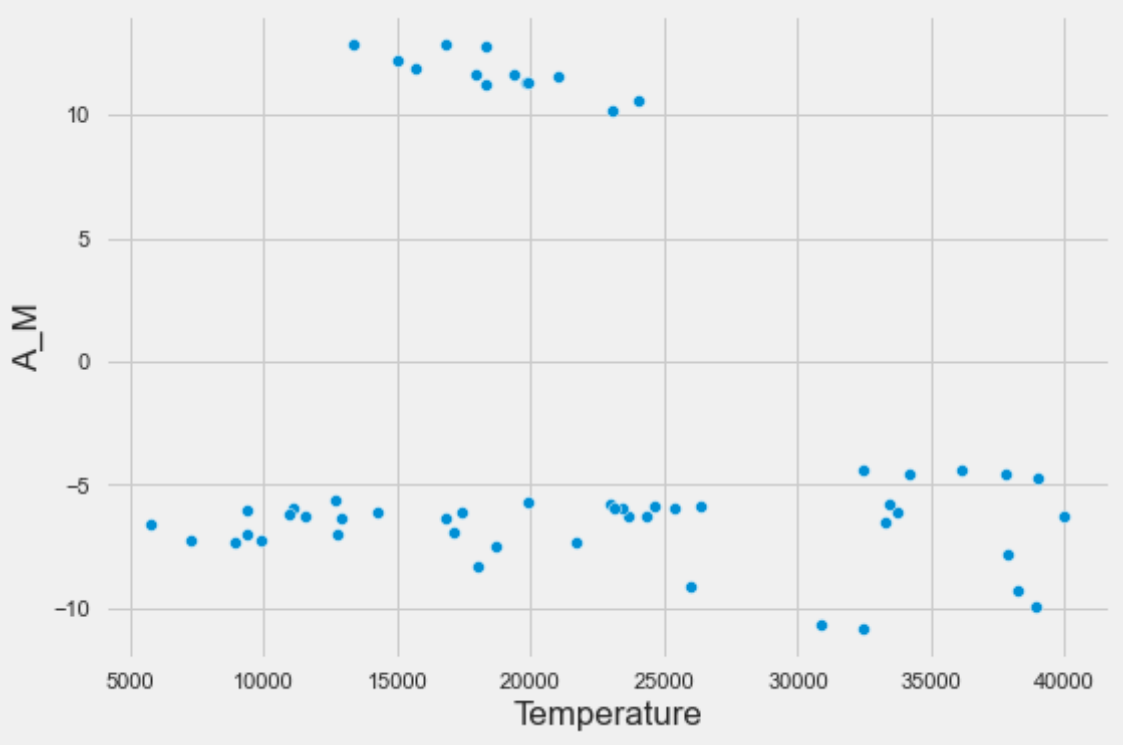
[55]:
```
sns.scatterplot(x = 'Temperature',y = 'L',data = blue_df)
plt.show()
```



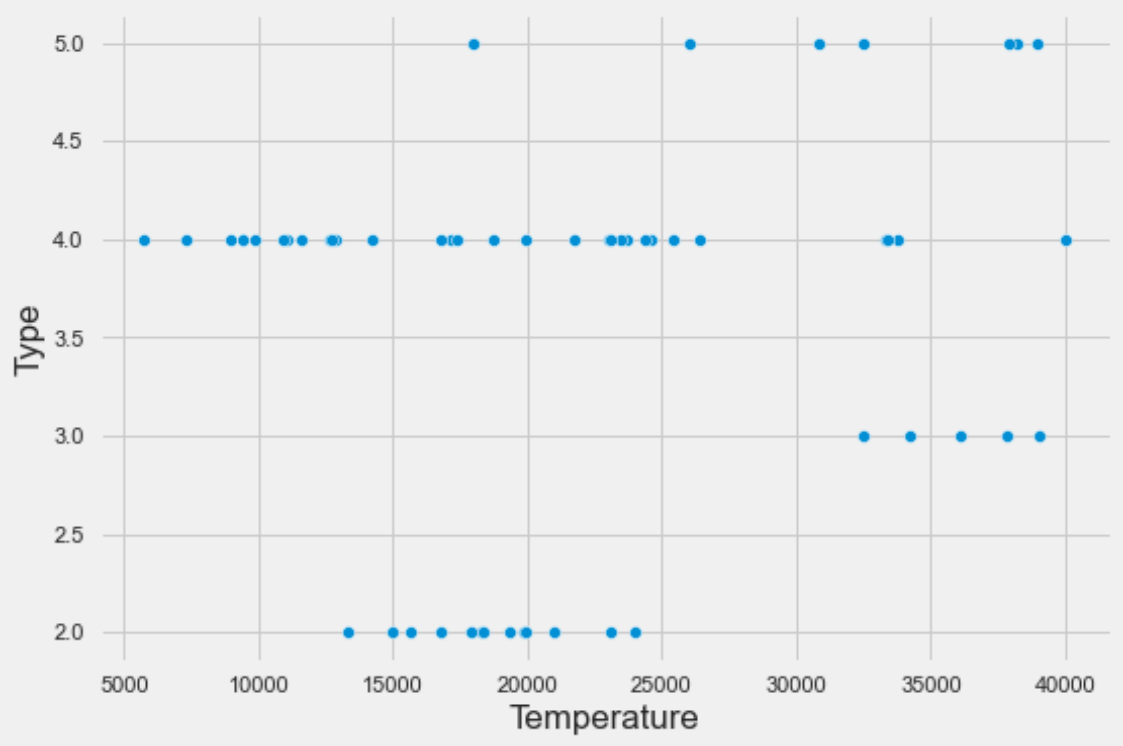
[56]:
```
sns.scatterplot(x = 'Temperature',y = 'R',data = blue_df)
plt.show()
```



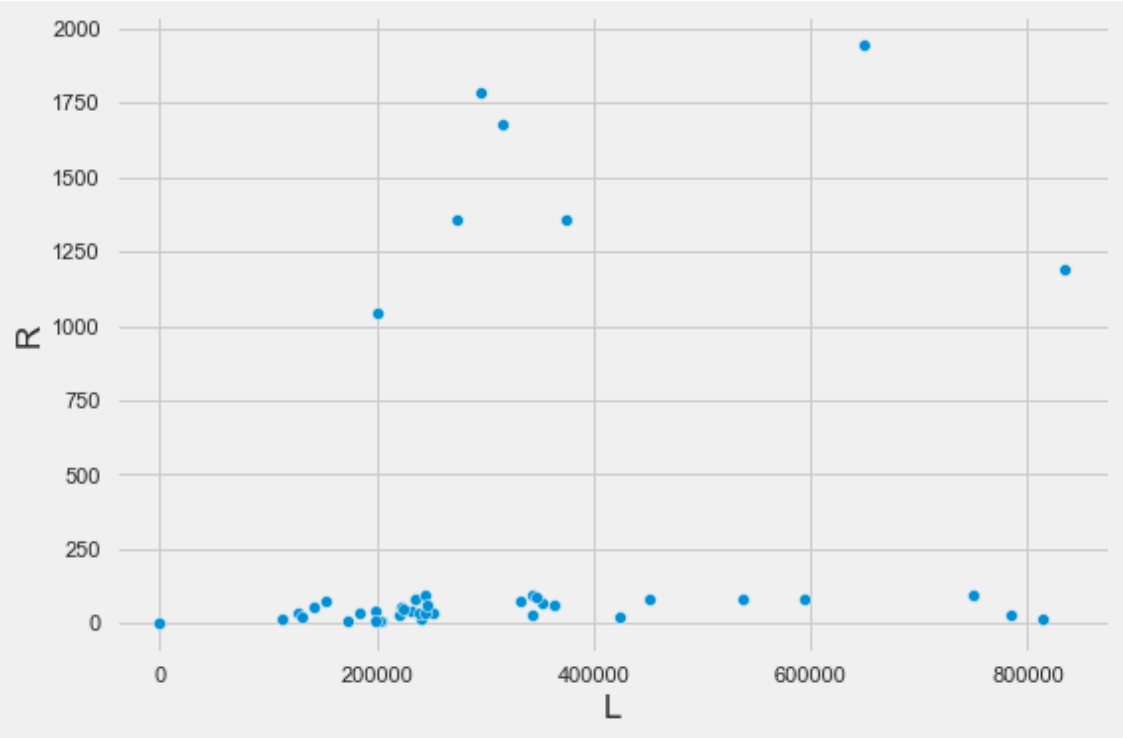
[57]:
```
sns.scatterplot(x = 'Temperature',y = 'A_M',data = blue_df)
plt.show()
```



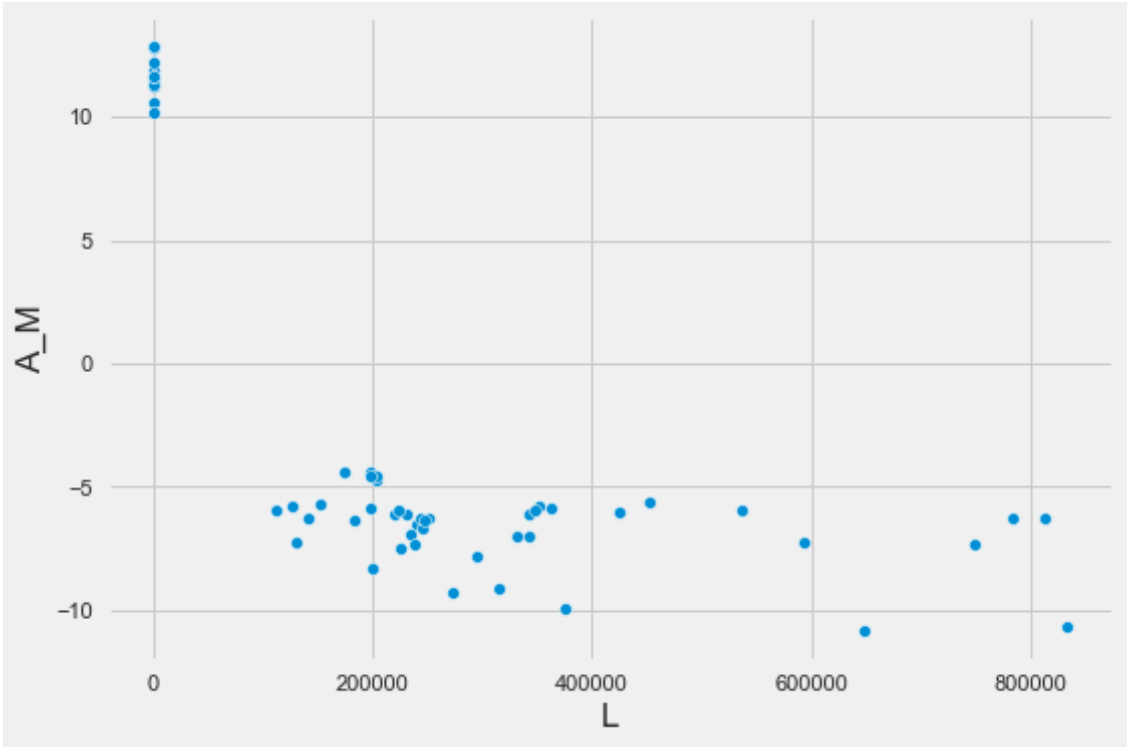
[58]:
```
sns.scatterplot(x = 'Temperature',y = 'Type',data = blue_df)
plt.show()
```



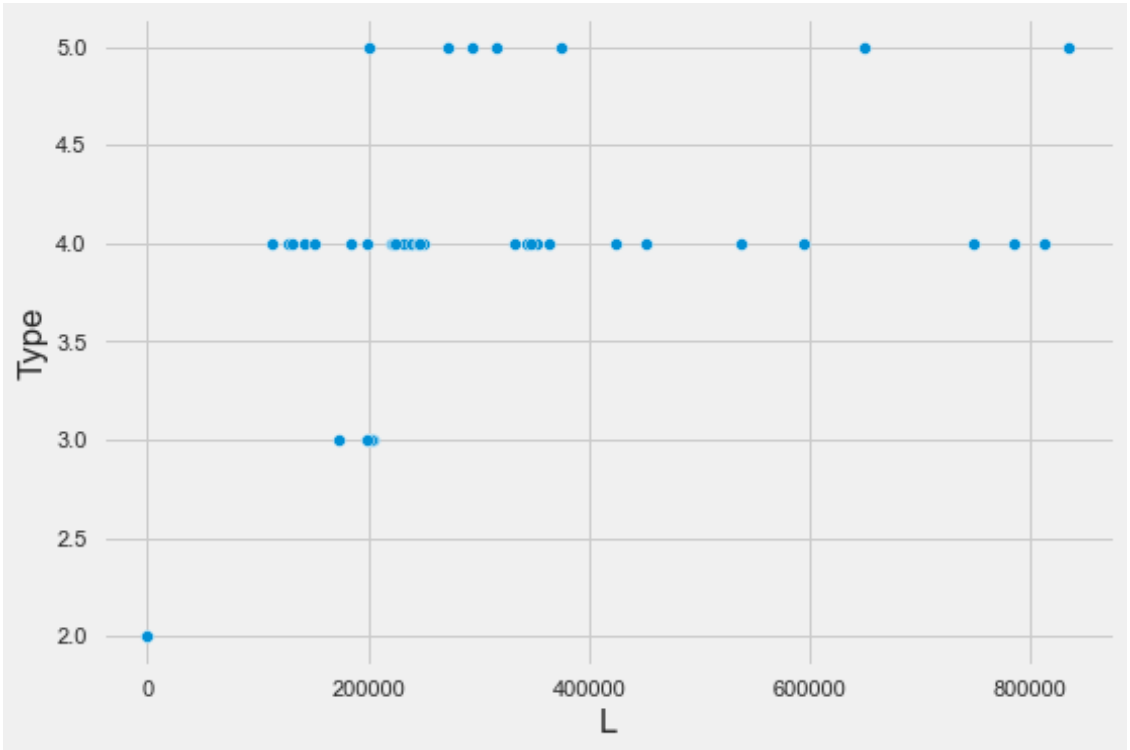
[59]:
```
sns.scatterplot(x = 'L',y = 'R',data = blue_df)
plt.show()
```



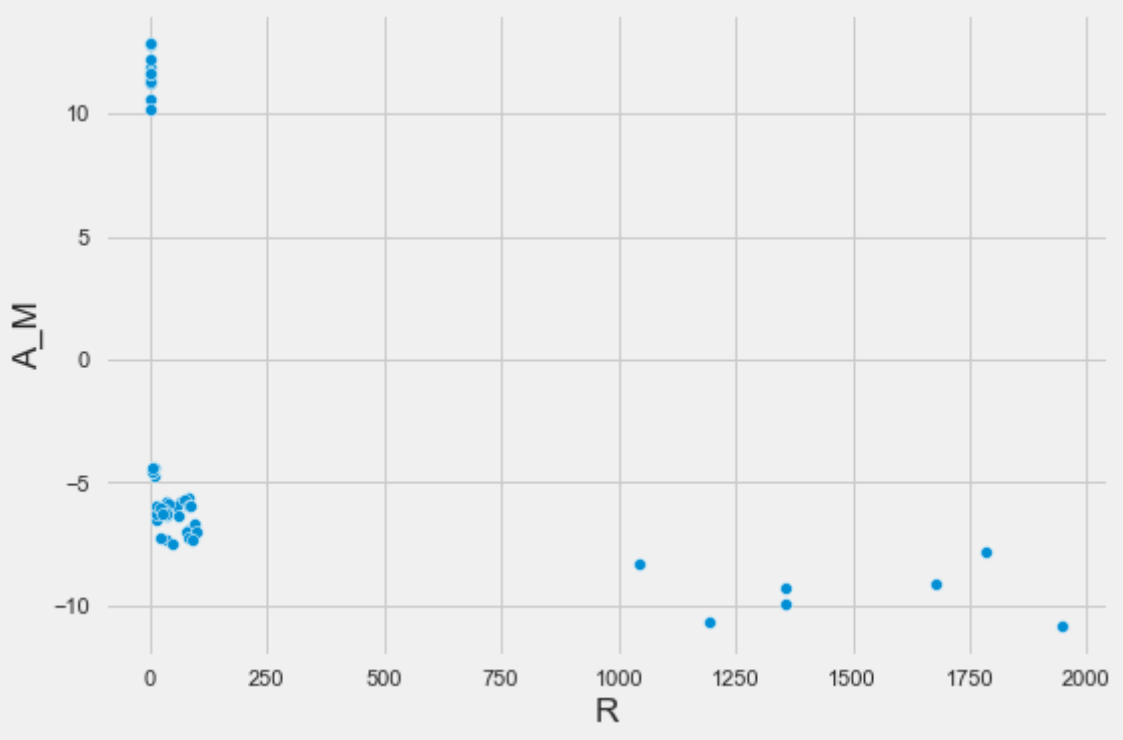
[60]:
```
sns.scatterplot(x = 'L',y = 'A_M',data = blue_df)
plt.show()
```



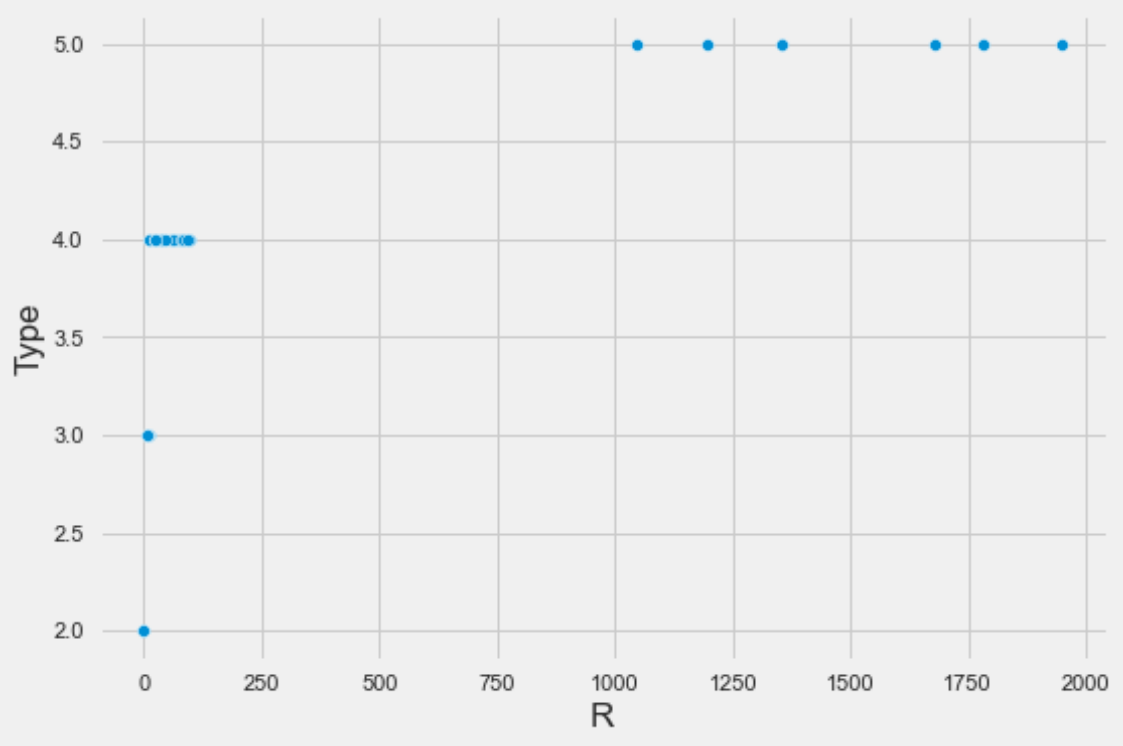
[61]:
```
sns.scatterplot(x = 'L',y = 'Type',data = blue_df)
plt.show()
```



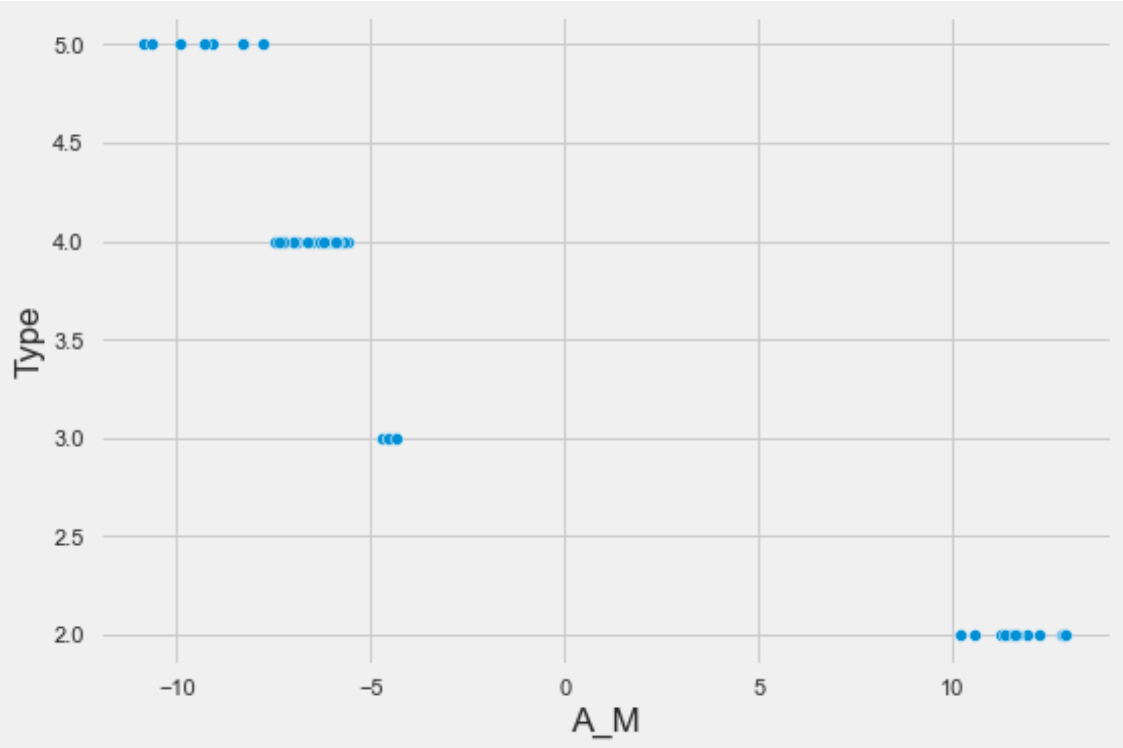
[62]:
```
sns.scatterplot(x = 'R',y = 'A_M',data = blue_df)
plt.show()
```



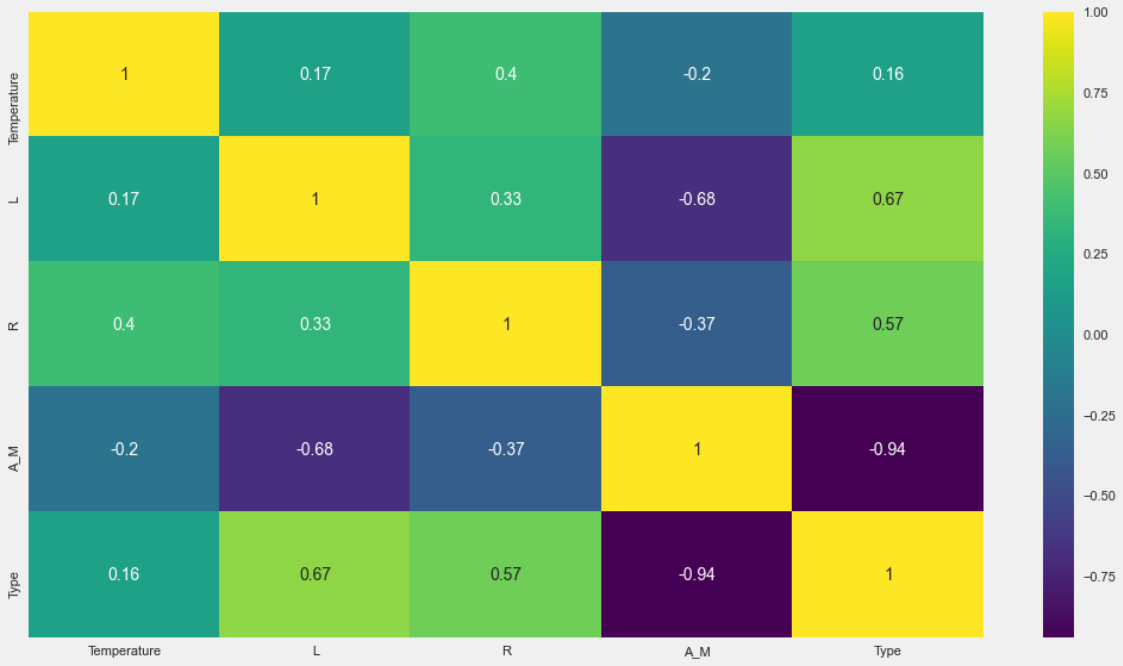
[63]:
```
sns.scatterplot(x = 'R',y = 'Type',data =blue_df)
plt.show()
```



[64]:
```
sns.scatterplot(x = 'A_M',y = 'Type',data = blue_df)
plt.show()
```

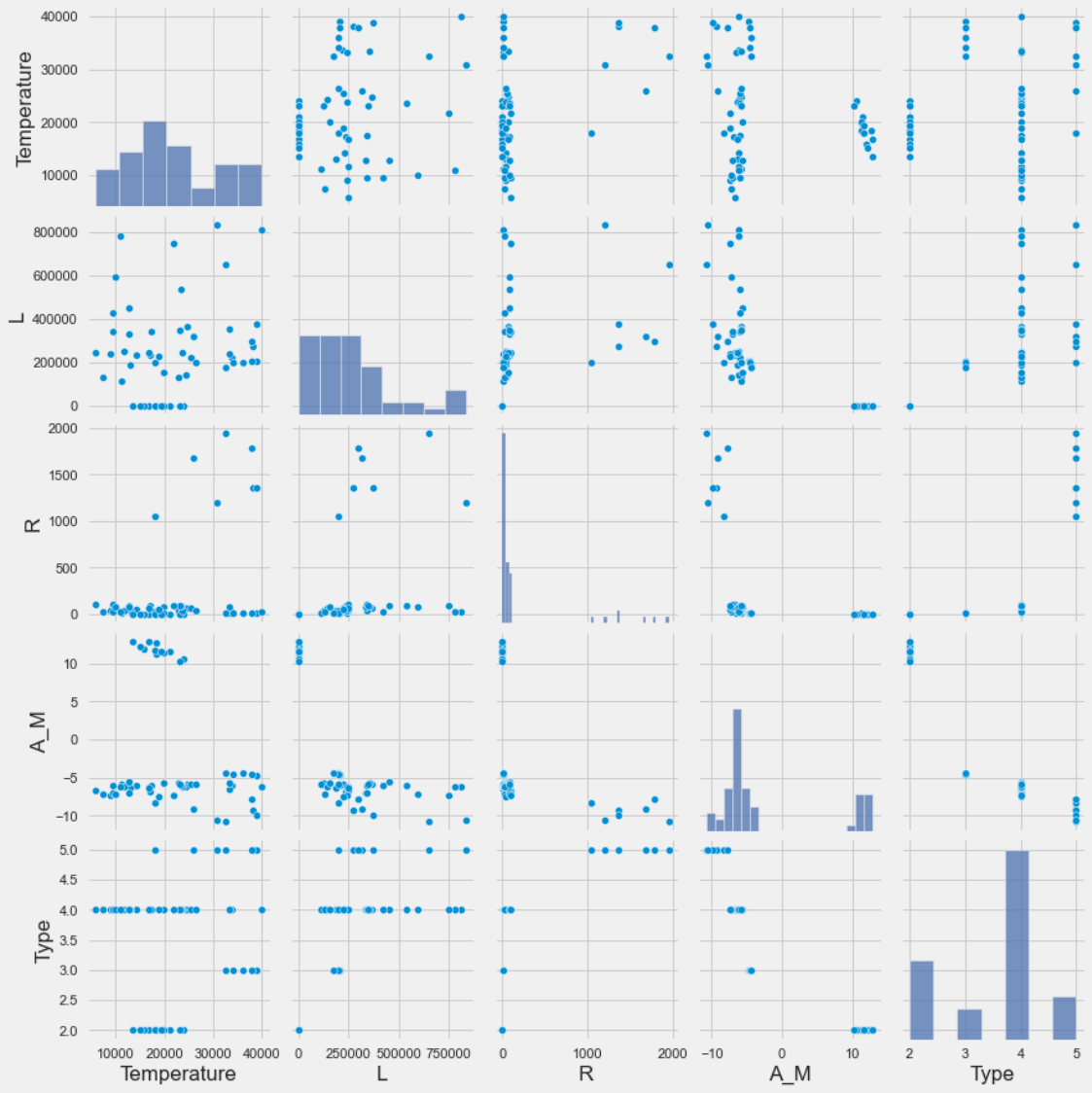


[65]:
```
plt.figure(figsize=(16,9))
x = blue_df.drop(['Color','Spectral_Class'],axis = 1)
ax = sns.heatmap(blue_df.corr(),annot = True,cmap = 'viridis')
plt.show()
```



[66]:
```
sns.pairplot(blue_df.drop(['Color','Spectral_Class'],axis = 1))
```

[66]:
```
<seaborn.axisgrid.PairGrid at 0x1f678a30130>
```



[67]:
```
f = blue_df[blue_df['Spectral_Class'] == 'O']
f
# blue color stars with specral class O
```

[67]:

	Temperature	L	R	A_M	Color	Spectral_Class	Type
30	39000	204000.0	10.600	-4.700	Blue	O	3
99	36108	198000.0	10.200	-4.400	Blue	O	3
101	40000	813000.0	14.000	-6.230	Blue	O	4
102	23000	127000.0	36.000	-5.760	Blue	O	4
103	17120	235000.0	83.000	-6.890	Blue	O	4
104	11096	112000.0	12.000	-5.910	Blue	O	4
105	14245	231000.0	42.000	-6.120	Blue	O	4
106	24630	363000.0	63.000	-5.830	Blue	O	4

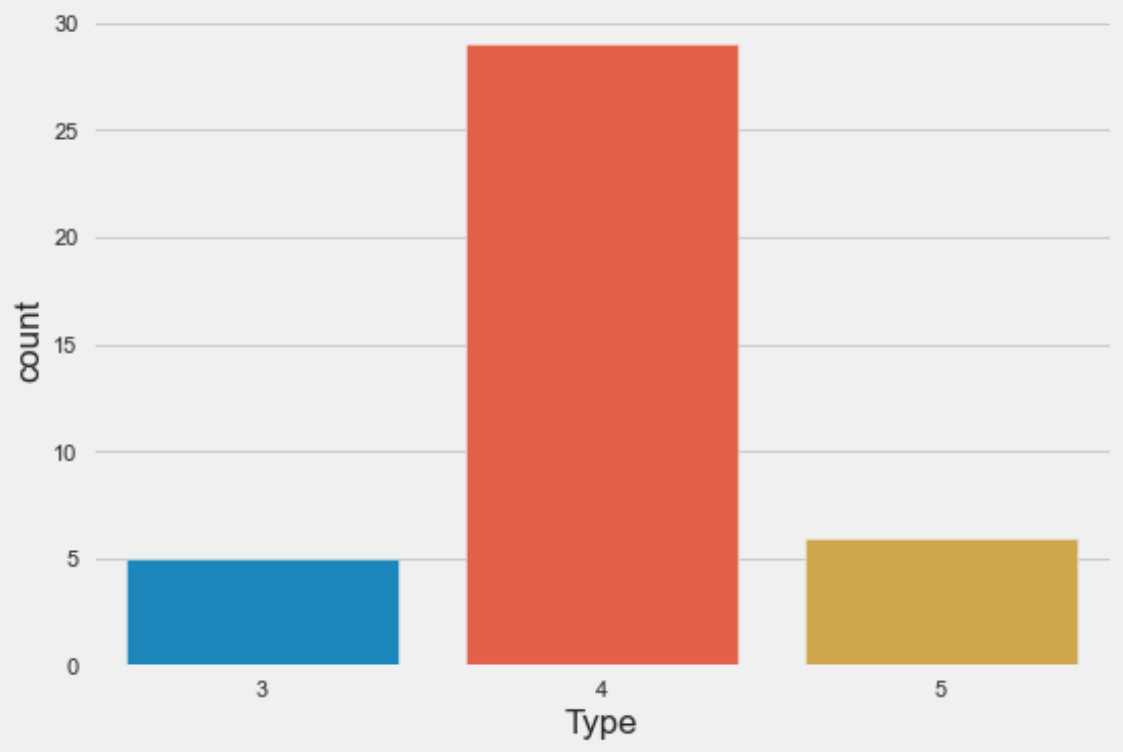
107	12893	184000.0	36.000	-6.340	Blue	0	4
108	24345	142000.0	57.000	-6.240	Blue	0	4
109	33421	352000.0	67.000	-5.790	Blue	0	4
159	37800	202900.0	6.860	-4.560	Blue	0	3
160	25390	223000.0	57.000	-5.920	Blue	0	4
161	11567	251000.0	36.000	-6.245	Blue	0	4
162	12675	452000.0	83.000	-5.620	Blue	0	4
163	5752	245000.0	97.000	-6.630	Blue	0	4
164	8927	239000.0	35.000	-7.340	Blue	0	4
165	7282	131000.0	24.000	-7.220	Blue	0	4
166	19923	152000.0	73.000	-5.690	Blue	0	4
167	26373	198000.0	39.000	-5.830	Blue	0	4
168	17383	342900.0	30.000	-6.090	Blue	0	4
169	9373	424520.0	24.000	-5.990	Blue	0	4
176	18000	200000.0	1045.000	-8.300	Blue	0	5
214	34190	198200.0	6.390	-4.570	Blue	0	3
215	32460	173800.0	6.237	-4.360	Blue	0	3
220	23678	244290.0	35.000	-6.270	Blue	0	4
221	12749	332520.0	76.000	-7.020	Blue	0	4
222	9383	342940.0	98.000	-6.980	Blue	0	4
223	23440	537430.0	81.000	-5.975	Blue	0	4
224	16787	246730.0	62.000	-6.350	Blue	0	4
225	18734	224780.0	46.000	-7.450	Blue	0	4
226	9892	593900.0	80.000	-7.262	Blue	0	4
227	10930	783930.0	25.000	-6.224	Blue	0	4
228	23095	347820.0	86.000	-5.905	Blue	0	4
229	21738	748890.0	92.000	-7.346	Blue	0	4
231	38234	272830.0	1356.000	-9.290	Blue	0	5
232	32489	648430.0	1948.500	-10.840	Blue	0	5
235	38940	374830.0	1356.000	-9.930	Blue	0	5
236	30839	834042.0	1194.000	-10.630	Blue	0	5
239	37882	294903.0	1783.000	-7.800	Blue	0	5

[68]:
```
blue_df[blue_df['Spectral_Class'] == 'O']['Type'].value_counts()
# blue color stars with specral class O and count of all its types
```

[68]:
```
4    29
5     6
3     5
Name: Type, dtype: int64
```

[69]:
```
sns.countplot(x = 'Type',data = f)

#Most of the blue color stars with spectral class O are from type 4
```

[69]:
```
<AxesSubplot:xlabel='Type', ylabel='count'>
```



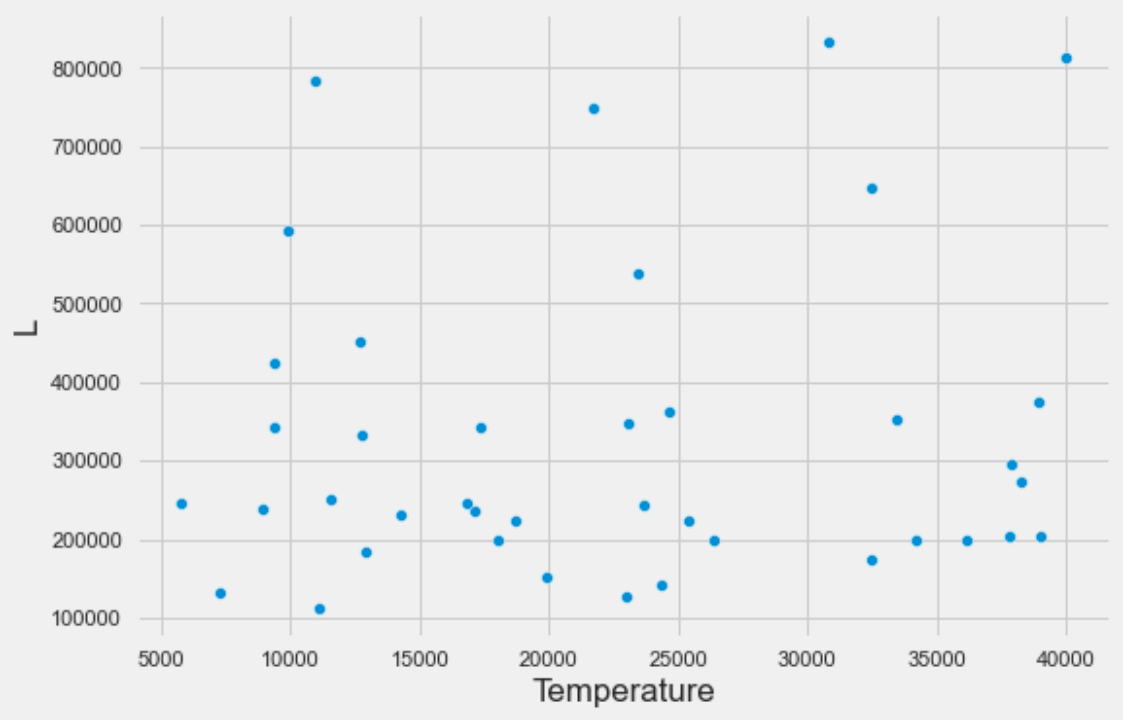
[70]:
```
fig = px.histogram(f, 'Temperature',
                   color = 'Type')
fig.show()
```

[71]:
```
fig = px.histogram(f, 'L',
                   color = 'Type')
fig.show()
```

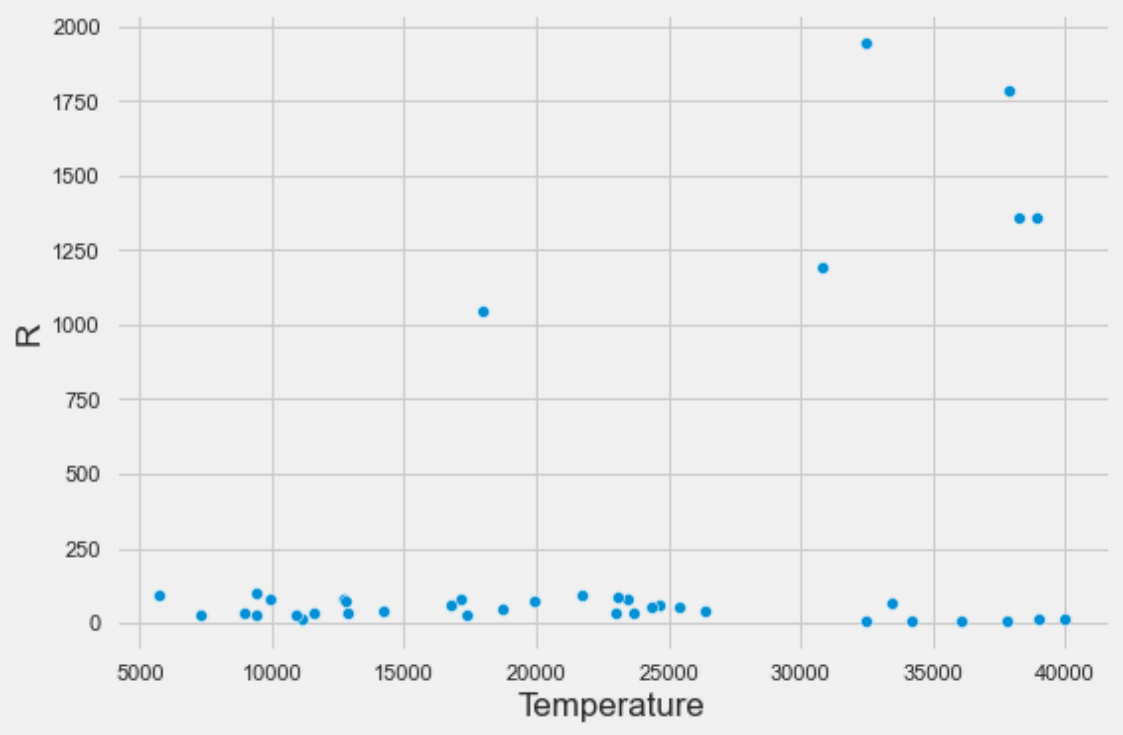
[72]:
```
fig = px.histogram(f, 'R',
                   color = 'Type')
fig.show()
```

[73]:
```
fig = px.histogram(f, 'A_M',
                   color = 'Type')
fig.show()
```

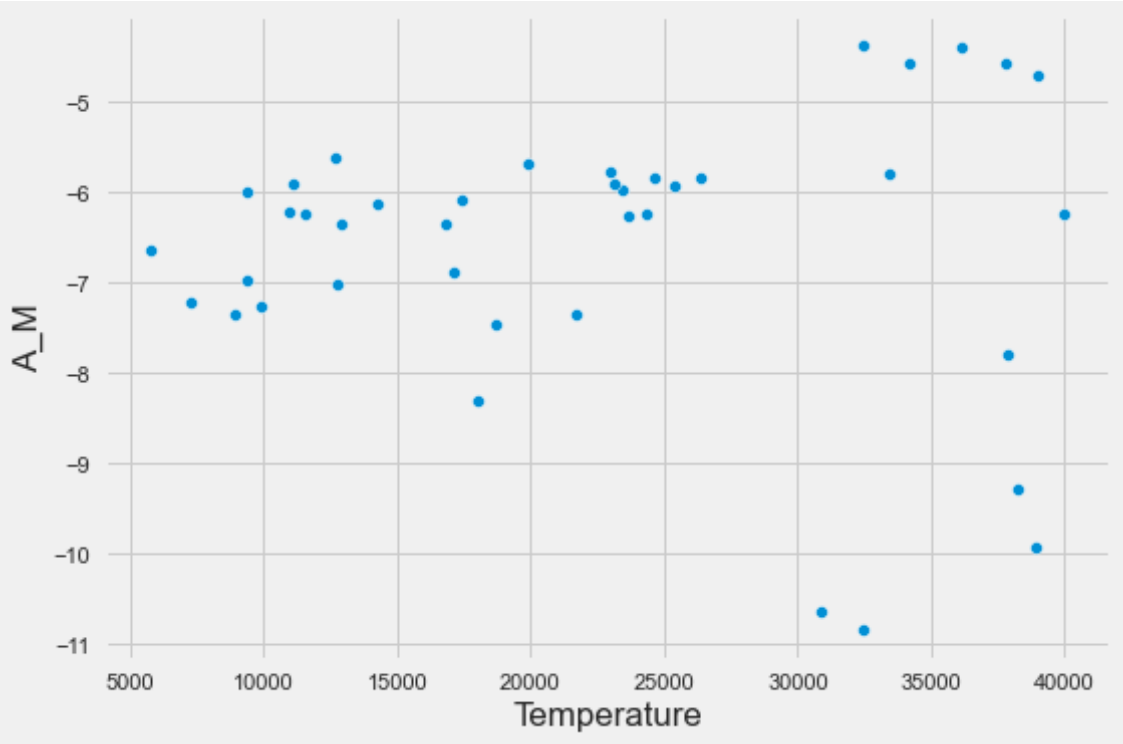
[74]:
```
sns.scatterplot(x = 'Temperature',y = 'L',data = f)
plt.show()
```



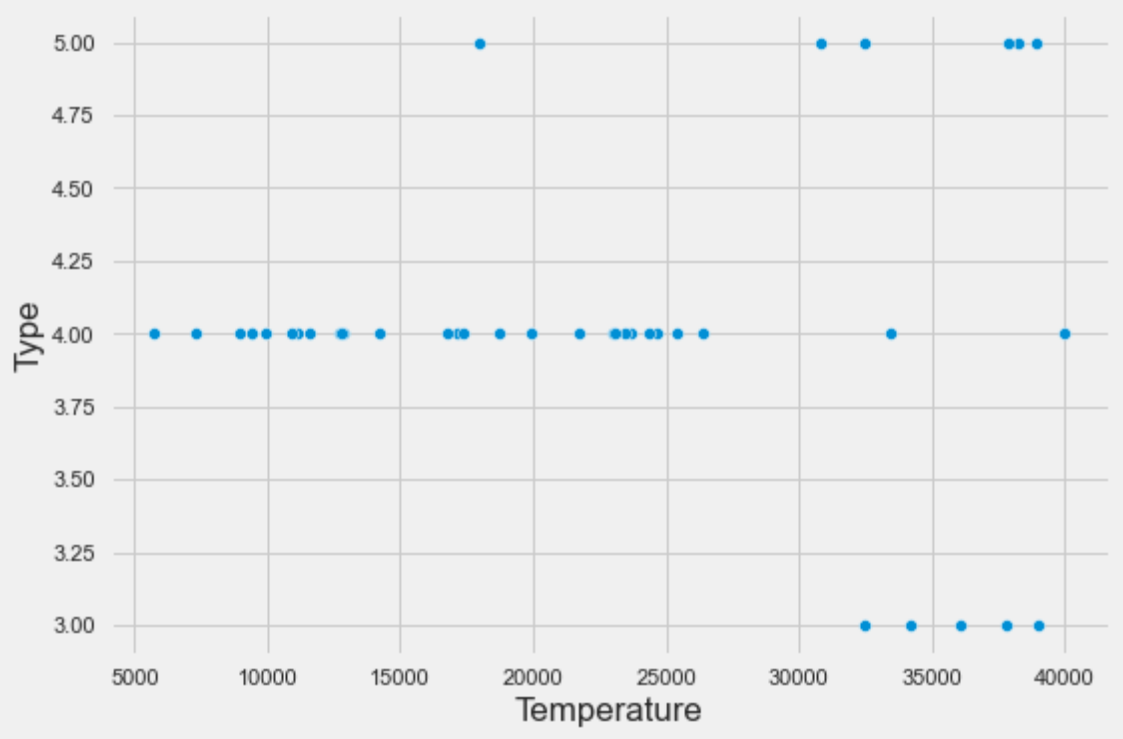
[75]:
```
sns.scatterplot(x = 'Temperature',y = 'R',data = f)
plt.show()
```



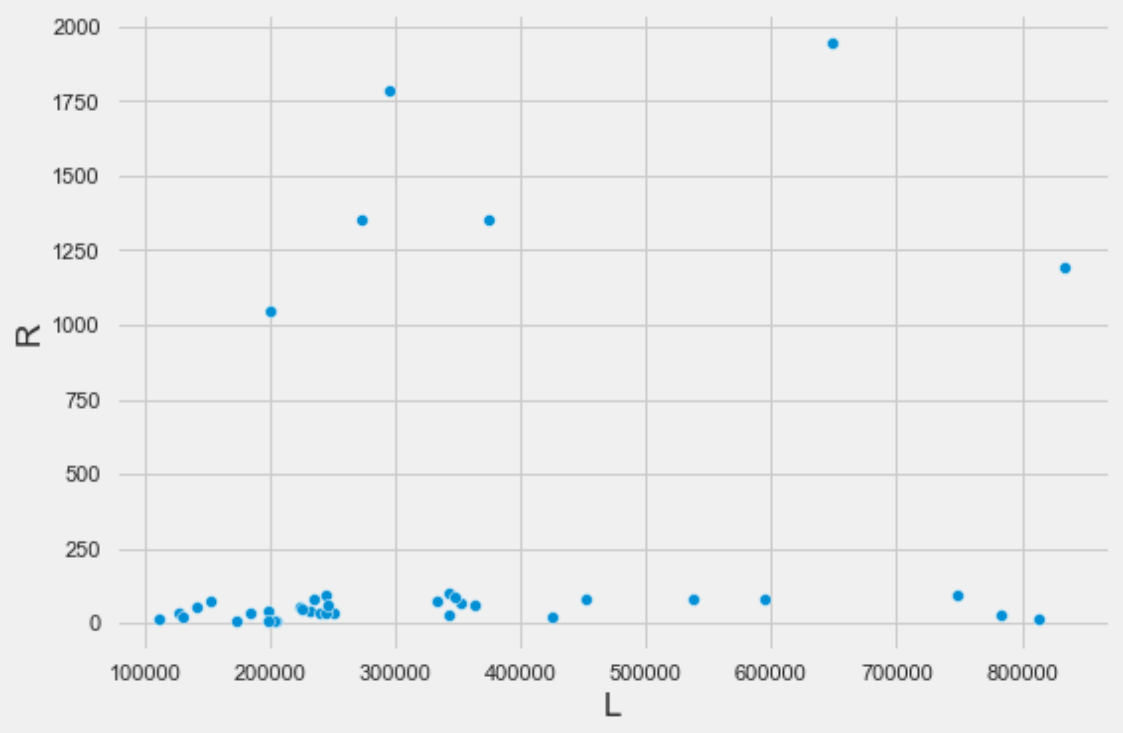
[76]:
```
sns.scatterplot(x = 'Temperature',y = 'A_M',data = f)
plt.show()
```



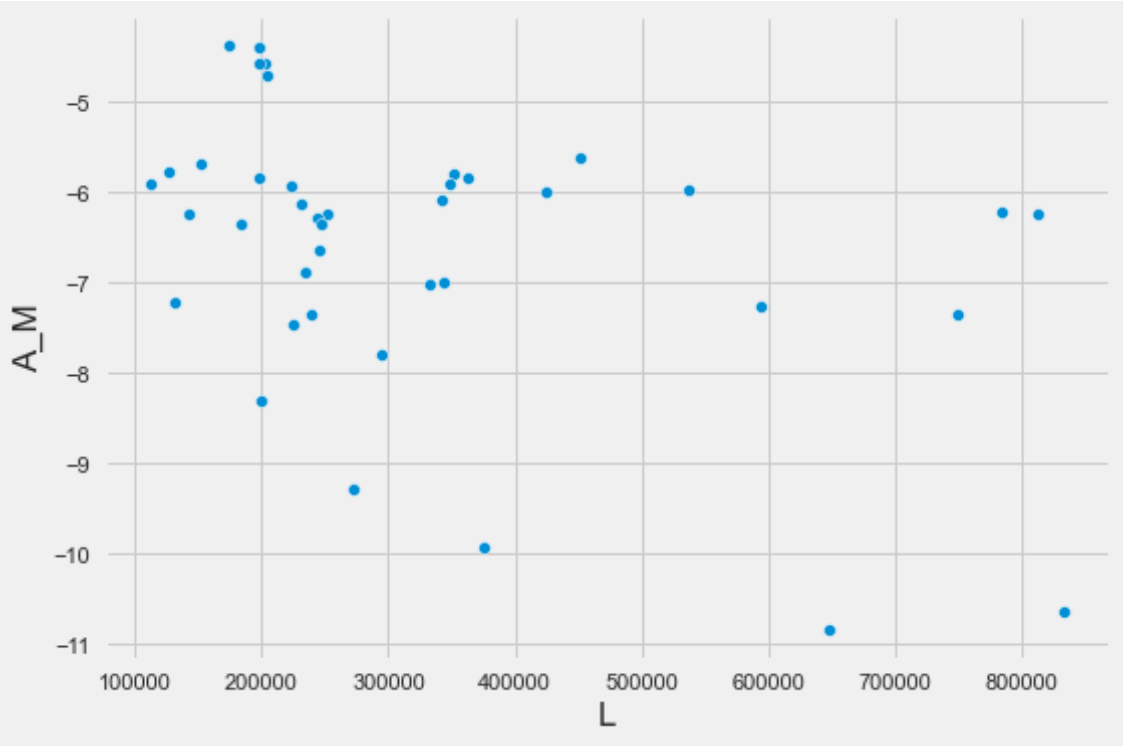
[77]:
```
sns.scatterplot(x = 'Temperature',y = 'Type',data = f)
plt.show()
```



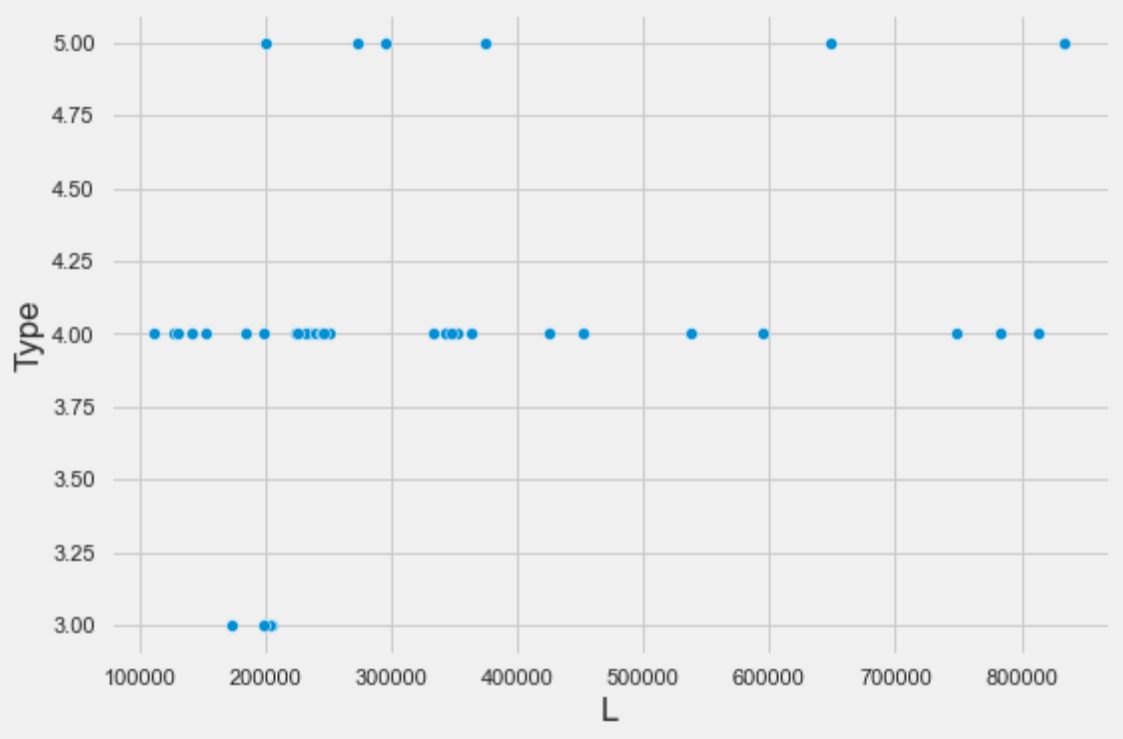
[78]:
```
sns.scatterplot(x = 'L',y = 'R',data = f)
plt.show()
```



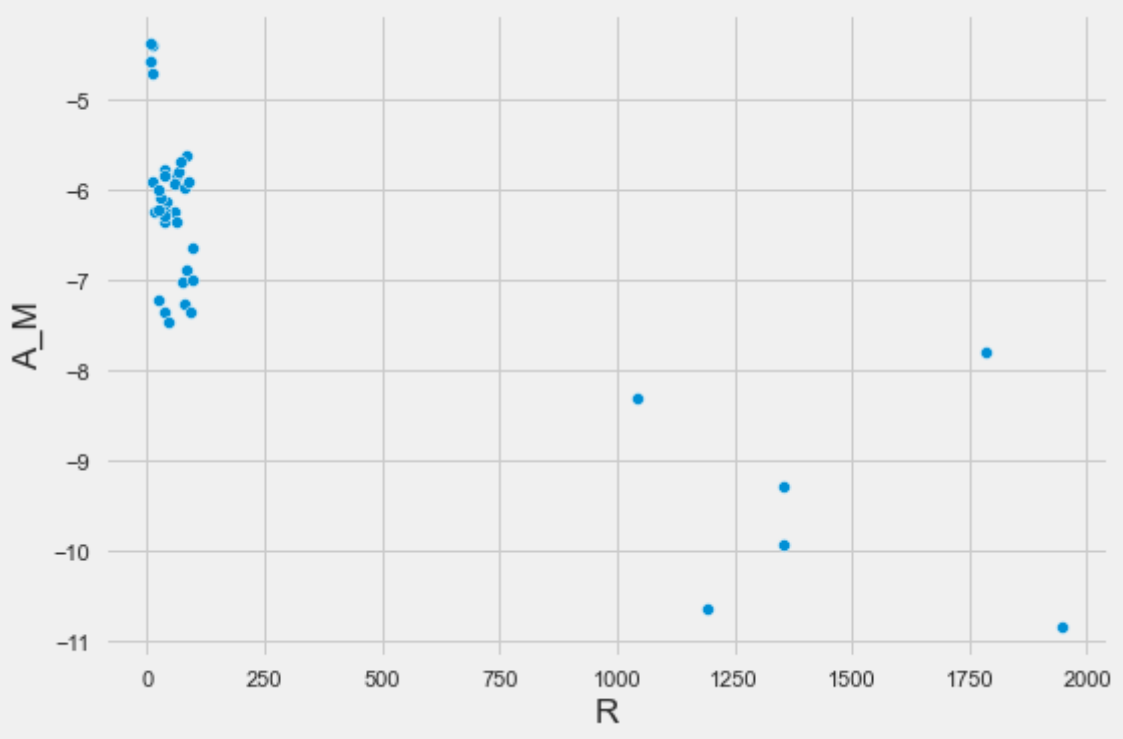
[79]:
```
sns.scatterplot(x = 'L',y = 'A_M',data = f)
plt.show()
```



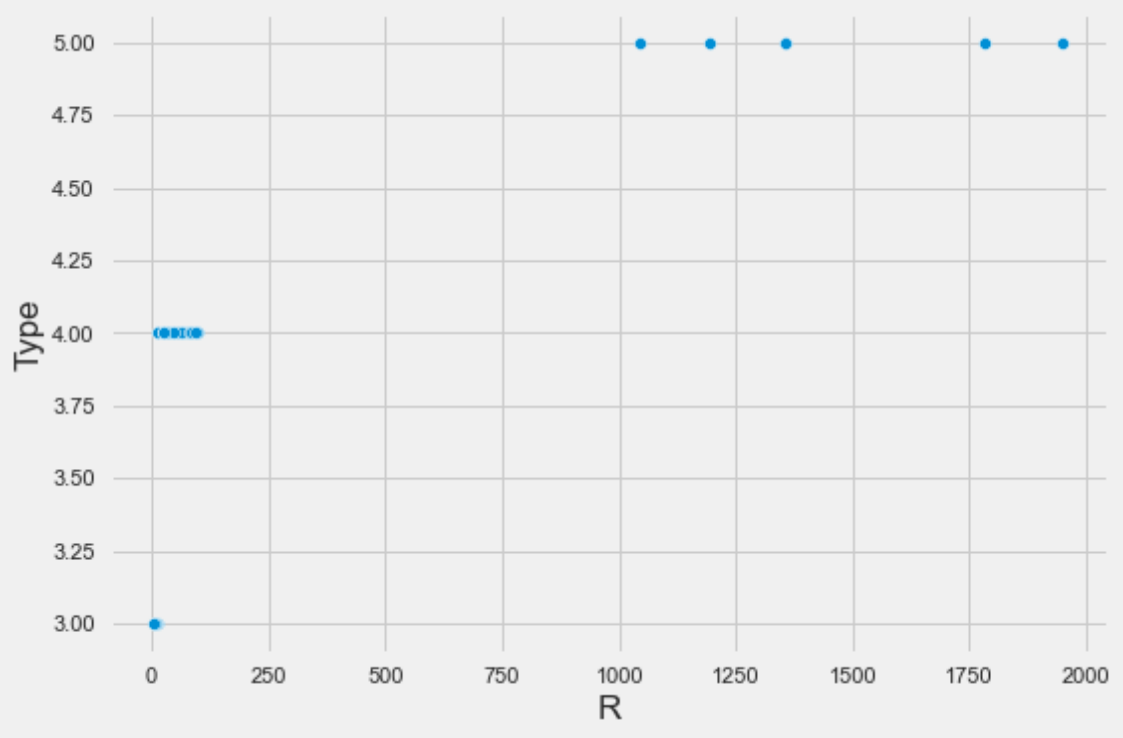
[80]:
```
sns.scatterplot(x = 'L',y = 'Type',data = f)
plt.show()
```



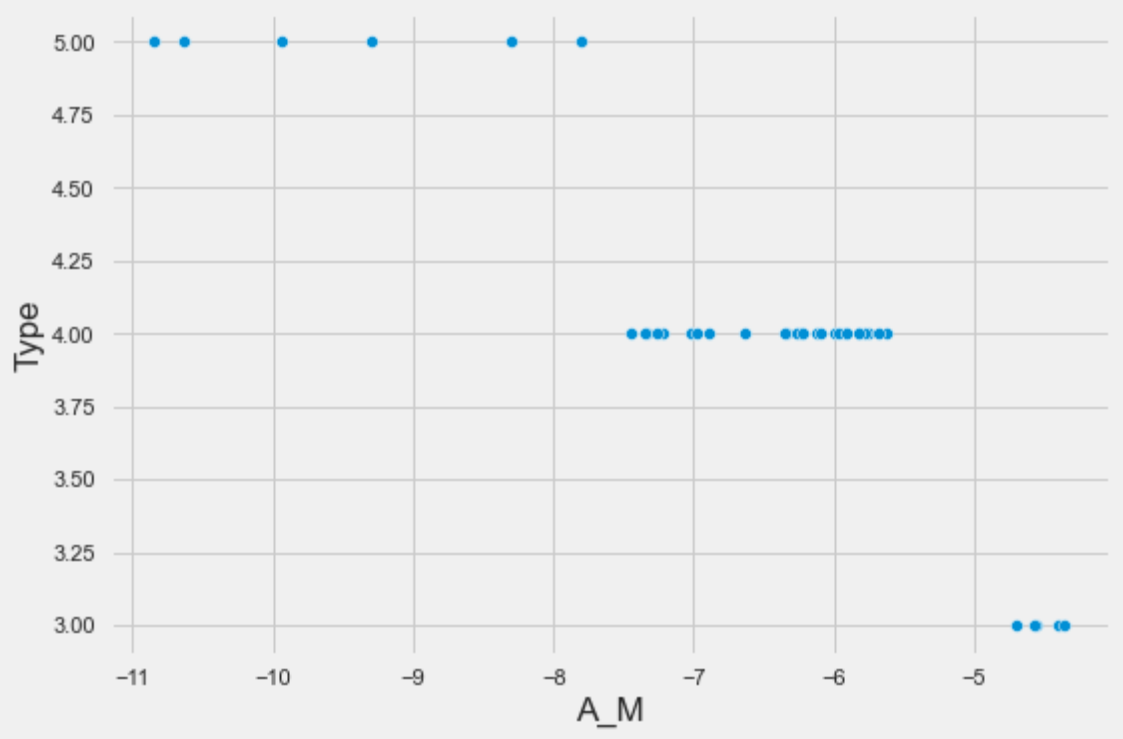
[81]:
```
sns.scatterplot(x = 'R',y = 'A_M',data = f)
plt.show()
```



[82]:
```
sns.scatterplot(x = 'R',y = 'Type',data = f)
plt.show()
```



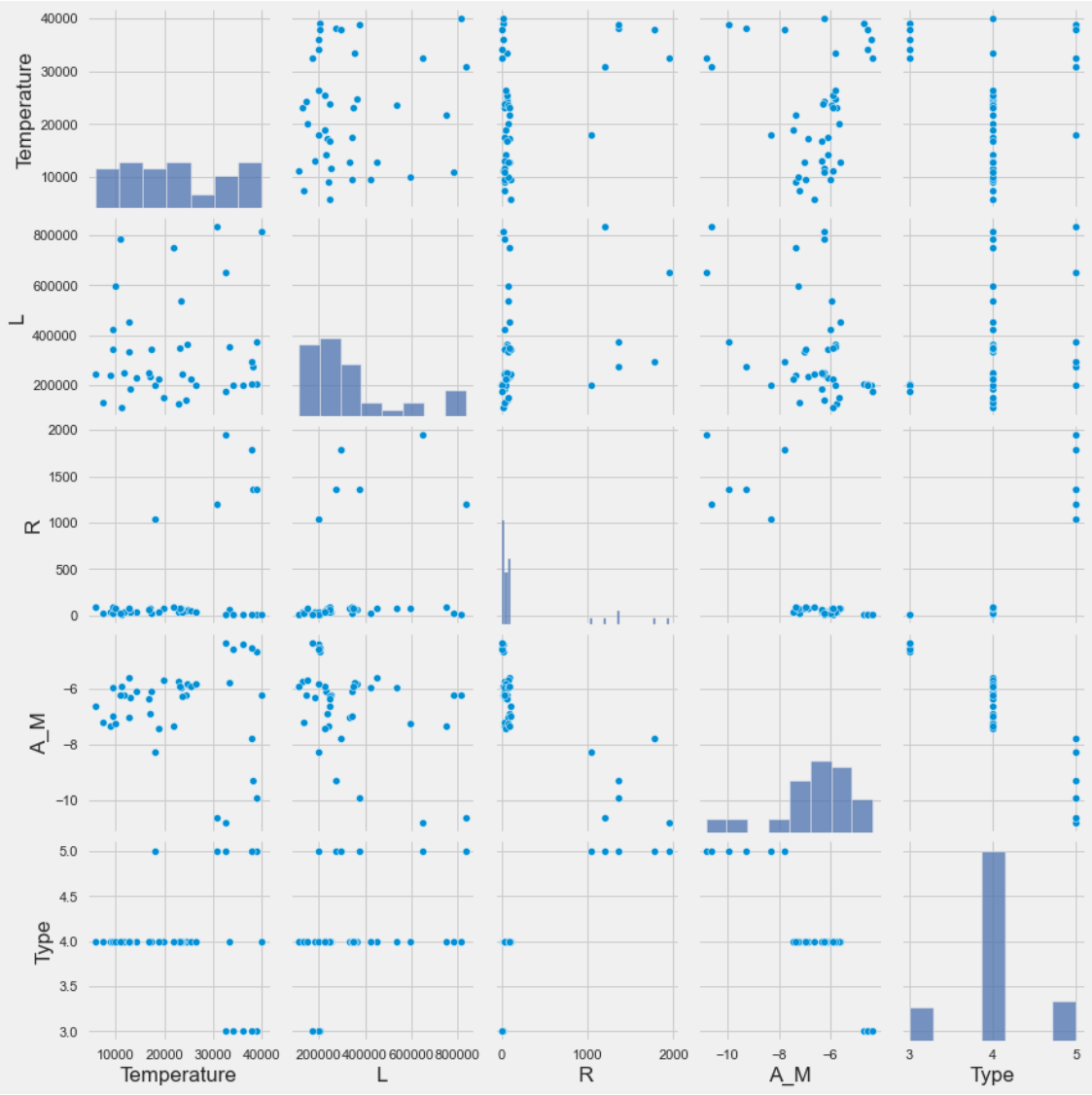
[83]:
```
sns.scatterplot(x = 'A_M',y = 'Type',data = f)
plt.show()
```



[84]:
```
plt.figure(figsize=(16,9))
x = f.drop(['Color','Spectral_Class'],axis = 1)
ax = sns.heatmap(f.corr(),annot = True,cmap = 'viridis')
plt.show()
```

[85]:
```
sns.pairplot(f.drop(['Color','Spectral_Class'],axis = 1))
```

[85]: <seaborn.axisgrid.PairGrid at 0x1f67952f760>



[86]:
```
g = blue_df[blue_df['Spectral_Class'] == 'B']
g
# blue color stars with specral class B
```

[86]:

	Temperature	L	R	A_M	Color	Spectral_Class	Type
49	33750	220000.00000	26.00000	-6.10	Blue	B	4

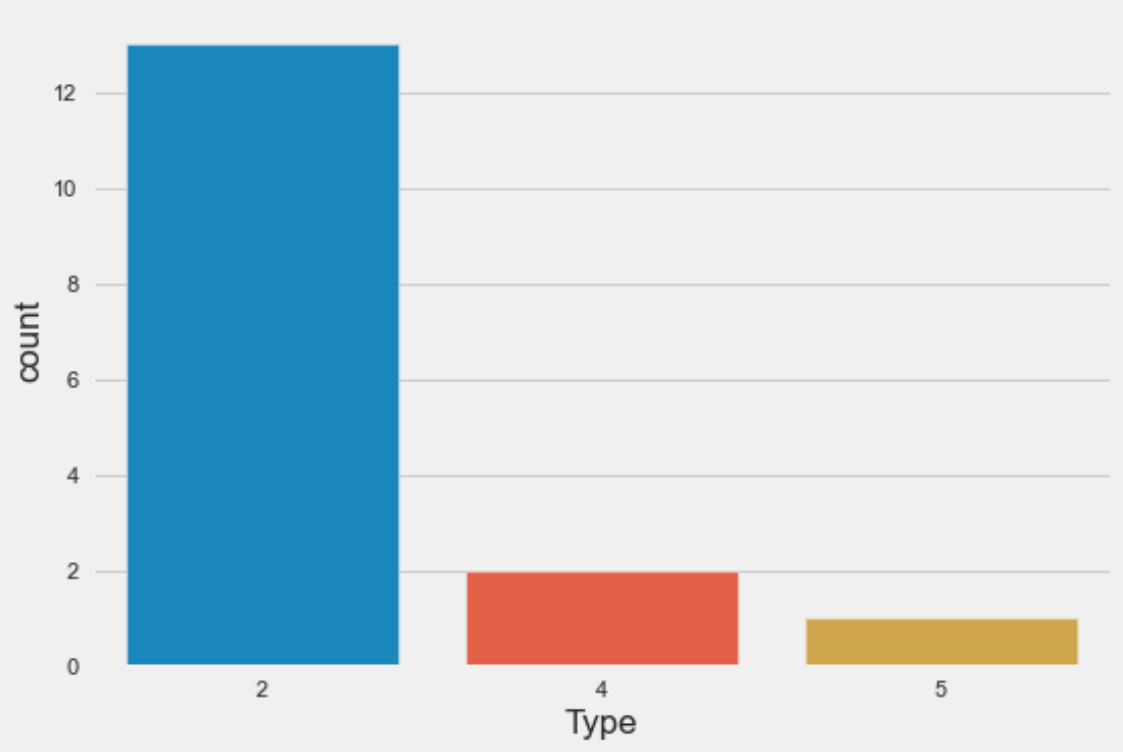
89	19860	0.00110	0.01310	11.34	Blue	B	2
100	33300	240000.00000	12.00000	-6.50	Blue	B	4
141	21020	0.00150	0.01120	11.52	Blue	B	2
142	18290	0.00130	0.00934	12.78	Blue	B	2
173	26000	316000.00000	1679.00000	-9.10	Blue	B	5
200	16790	0.00140	0.01210	12.87	Blue	B	2
201	15680	0.00122	0.01140	11.92	Blue	B	2
202	14982	0.00118	0.01130	12.23	Blue	B	2
203	13340	0.00109	0.01160	12.90	Blue	B	2
204	18340	0.00134	0.01240	11.22	Blue	B	2
205	19920	0.00156	0.01420	11.34	Blue	B	2
206	24020	0.00159	0.01270	10.55	Blue	B	2
207	23092	0.00132	0.01040	10.18	Blue	B	2
208	17920	0.00111	0.01060	11.66	Blue	B	2
209	19360	0.00125	0.00998	11.62	Blue	B	2

[87]:
```
blue_df[blue_df['Spectral_Class'] == 'B']['Type'].value_counts()
# blue color stars with specral class B and count of all its types
```

[87]:
```
2    13
4     2
5     1
Name: Type, dtype: int64
```

[88]:
```
sns.countplot(x = 'Type',data = g)

#Most of the blue color stars with spectral class B are from type 2
```

[88]:
```
<AxesSubplot:xlabel='Type', ylabel='count'>
```



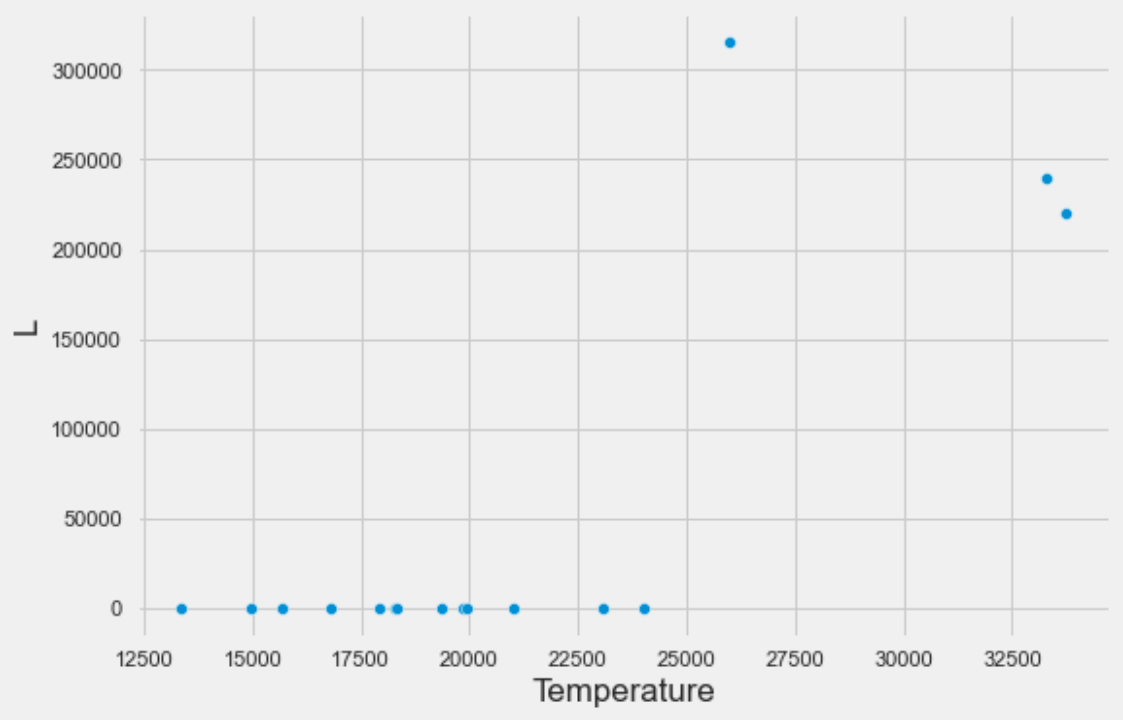
[89]:
```
fig = px.histogram(g, 'Temperature',
                   color = 'Type')
fig.show()
```

[90]:
```
fig = px.histogram(g, 'L',
                   color = 'Type')
fig.show()
```

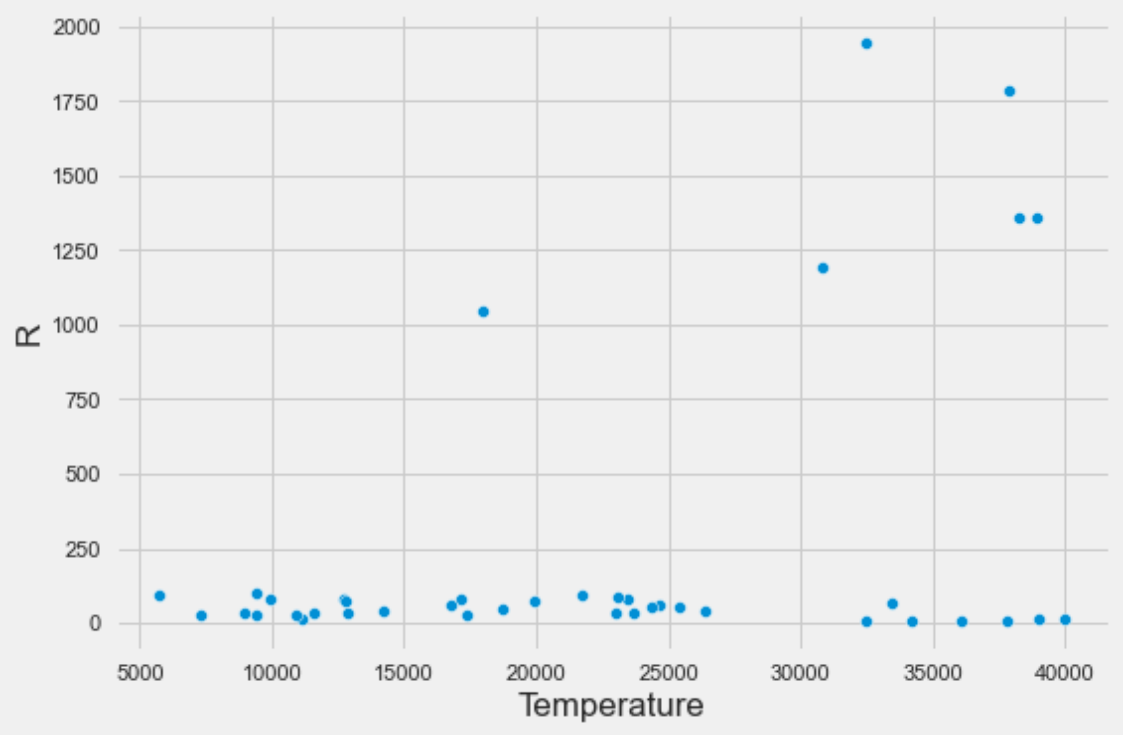
[91]:
```
fig = px.histogram(g, 'R',
                   color = 'Type')
fig.show()
```

[92]:
```
fig = px.histogram(g, 'A_M',
                   color = 'Type')
fig.show()
```

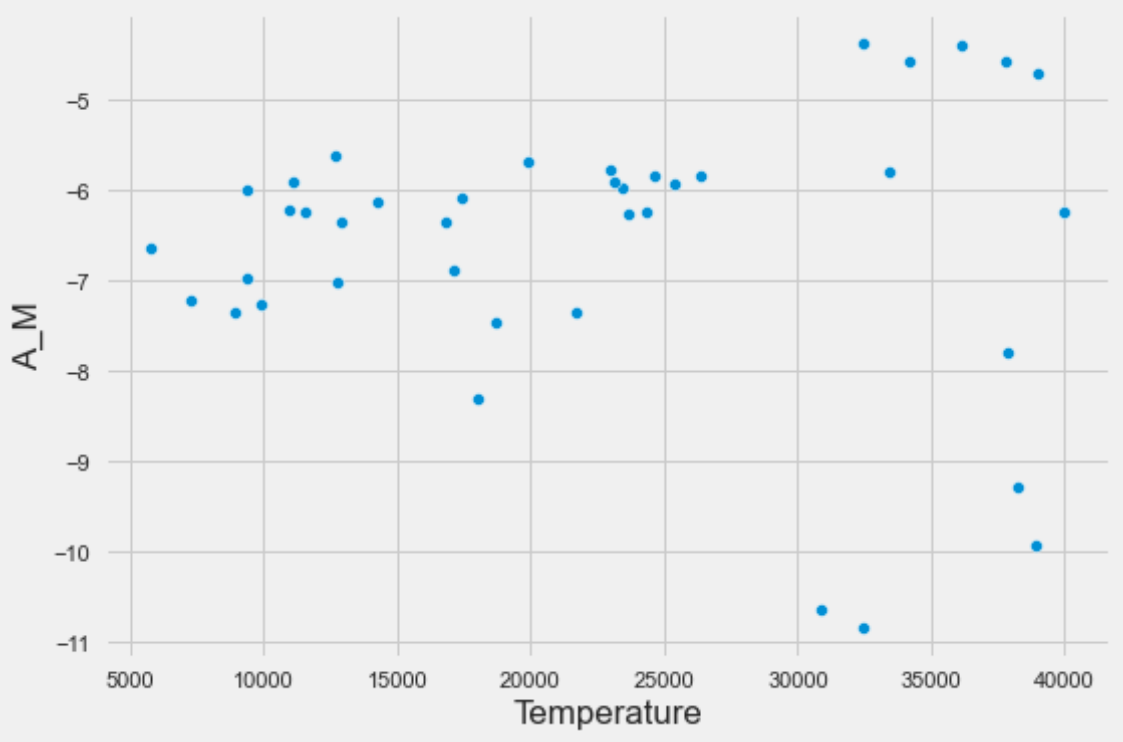
[93]:
```
sns.scatterplot(x = 'Temperature',y = 'L',data = g)
plt.show()
```



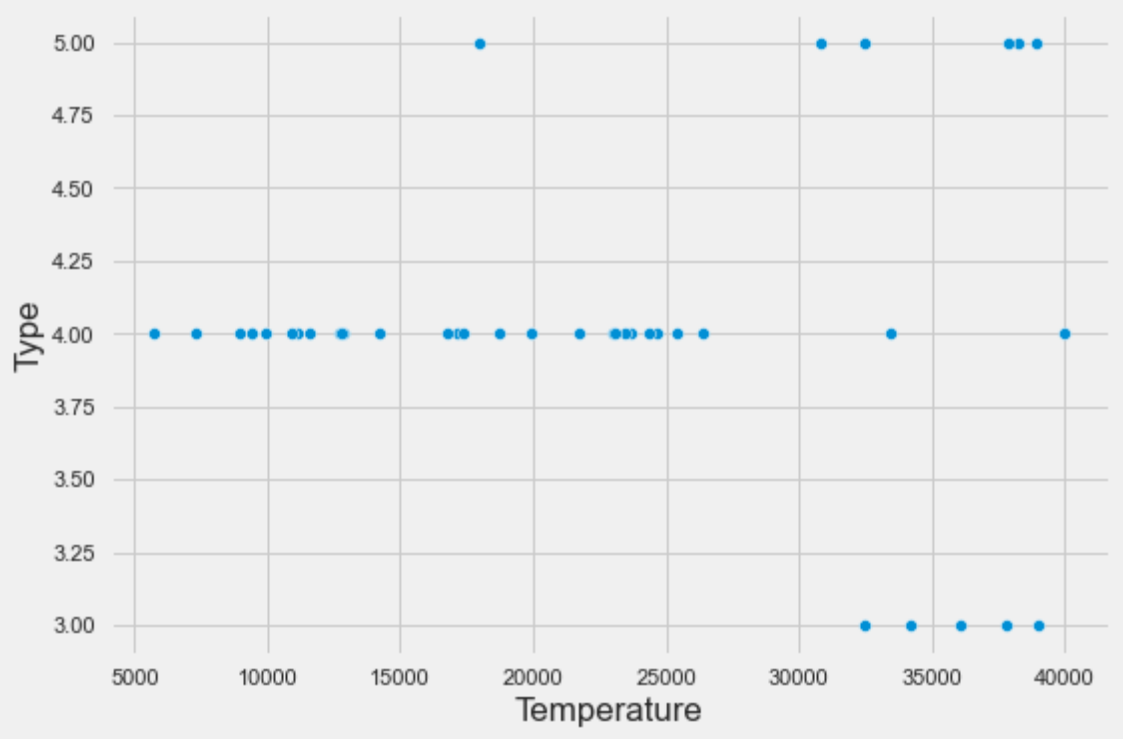
[94]:
```
sns.scatterplot(x = 'Temperature',y = 'R',data = f)
plt.show()
```



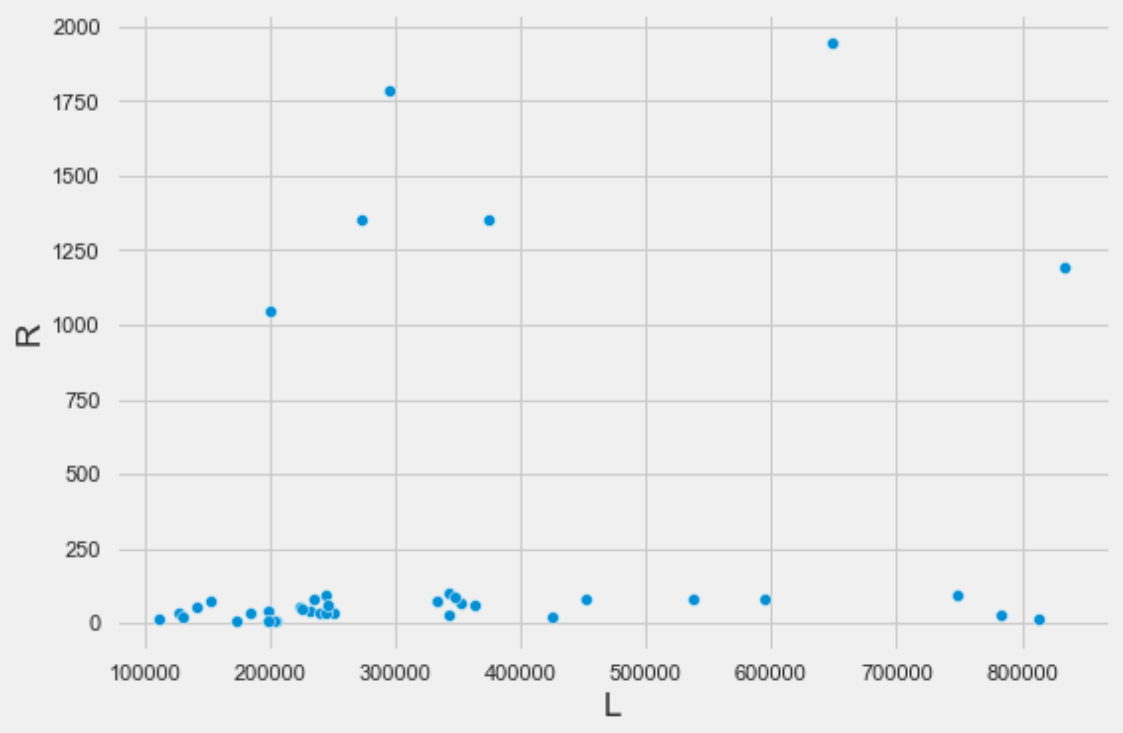
[95]:
```
sns.scatterplot(x = 'Temperature',y = 'A_M',data = f)
plt.show()
```



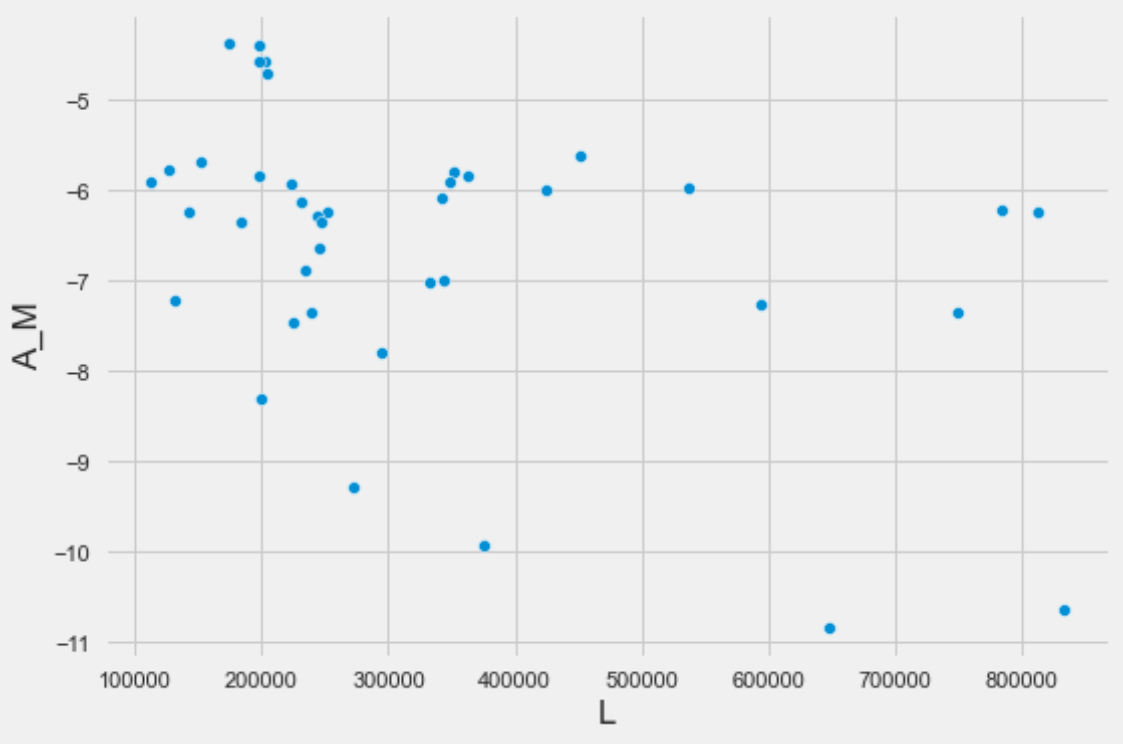
[96]:
```
sns.scatterplot(x = 'Temperature',y = 'Type',data = f)
plt.show()
```



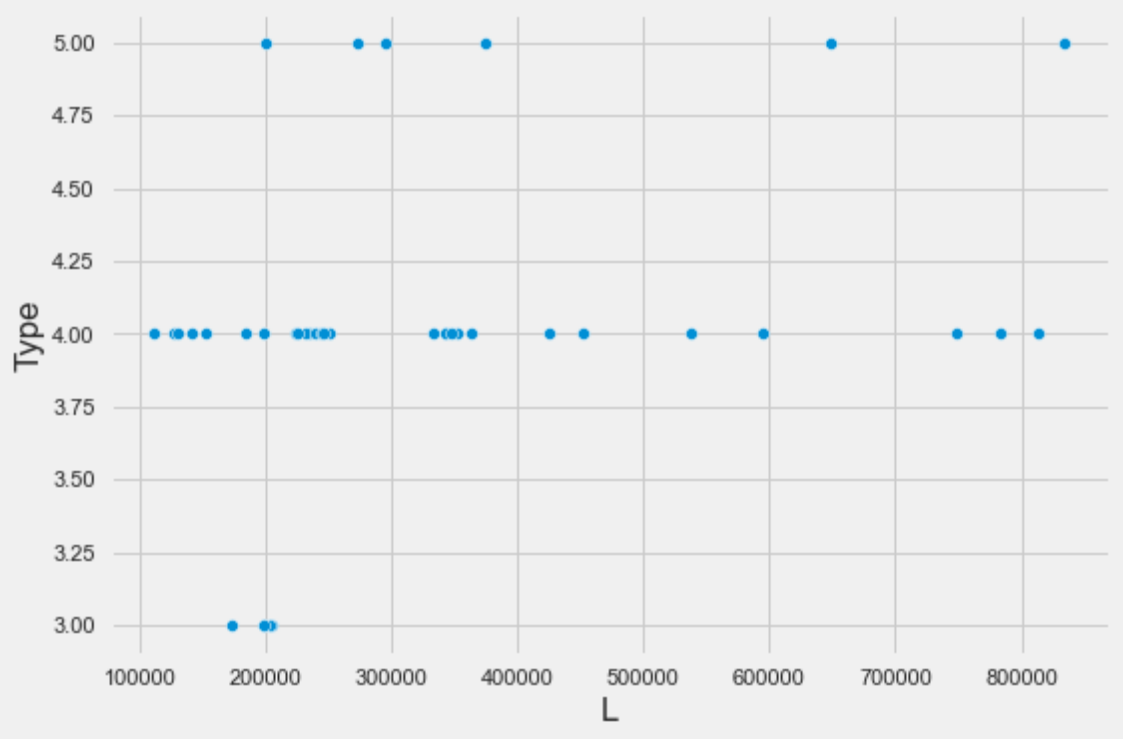
[97]:
```
sns.scatterplot(x = 'L',y = 'R',data = f)
plt.show()
```



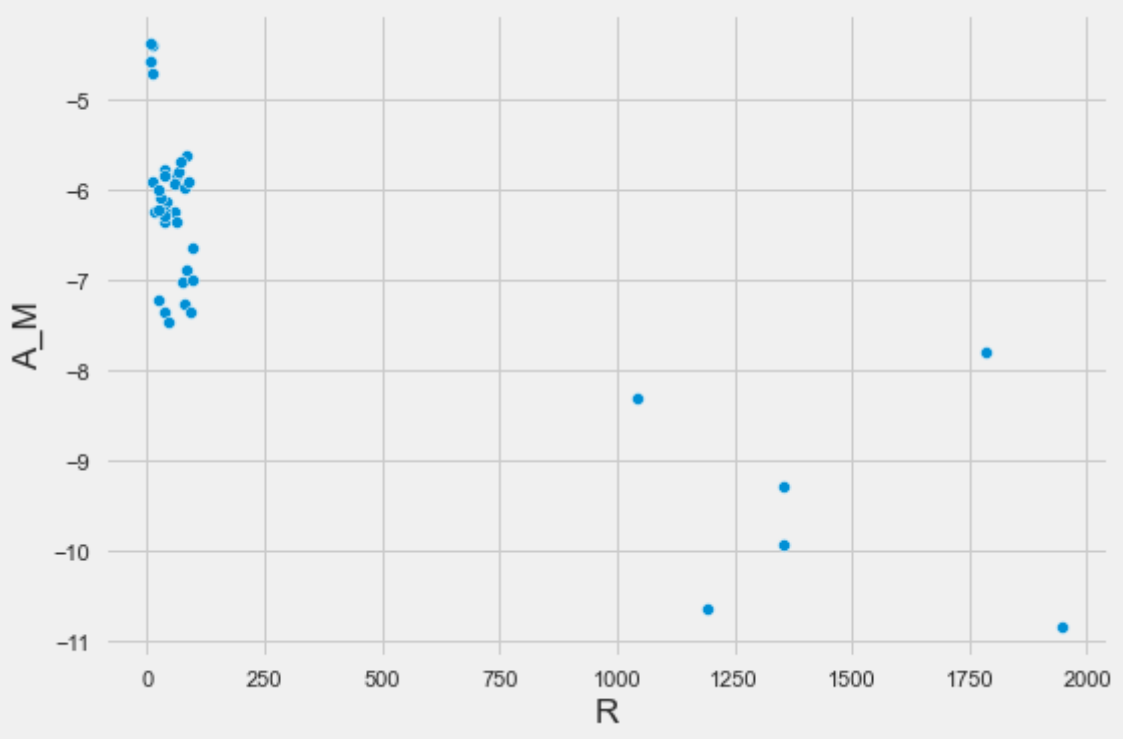
[98]:
```
sns.scatterplot(x = 'L',y = 'A_M',data = f)
plt.show()
```



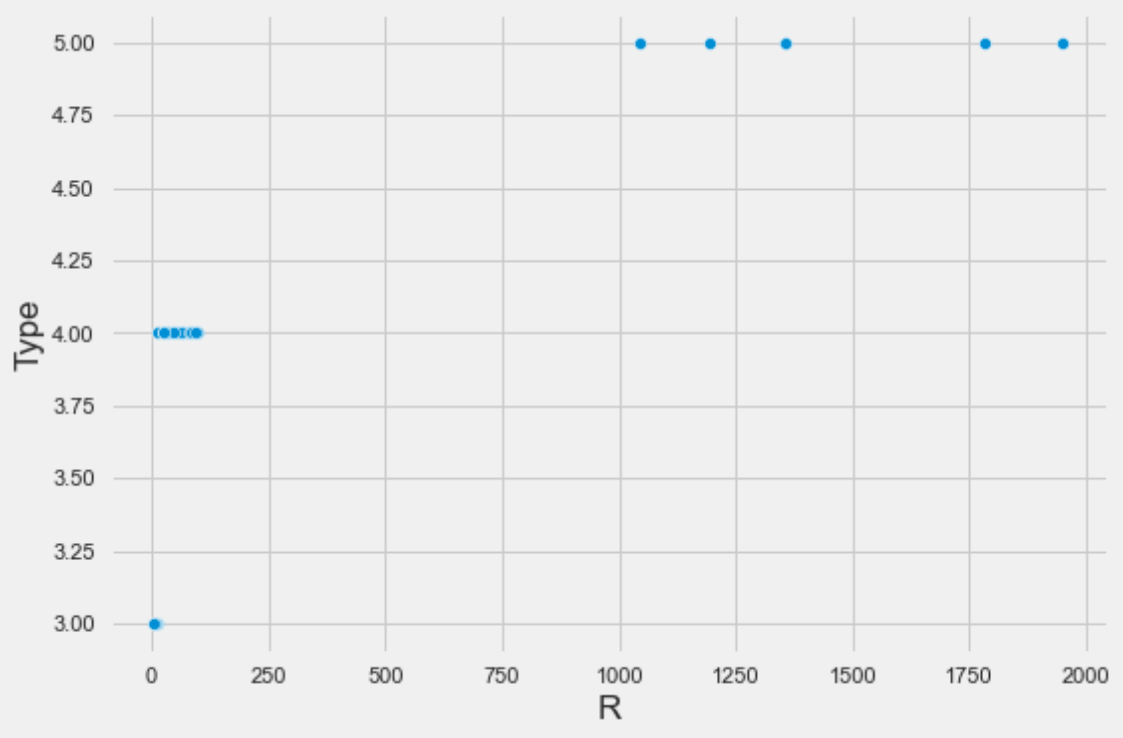
[99]:
```
sns.scatterplot(x = 'L',y = 'Type',data = f)
plt.show()
```



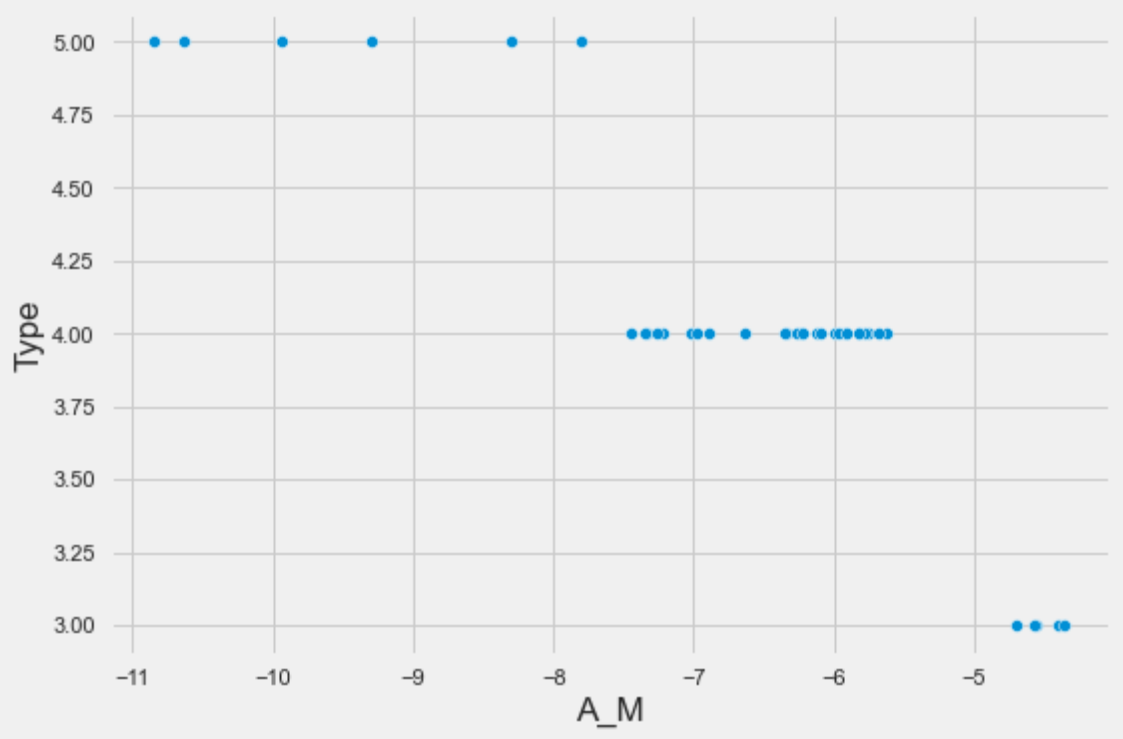
[100]:
```
sns.scatterplot(x = 'R',y = 'A_M',data = f)
plt.show()
```



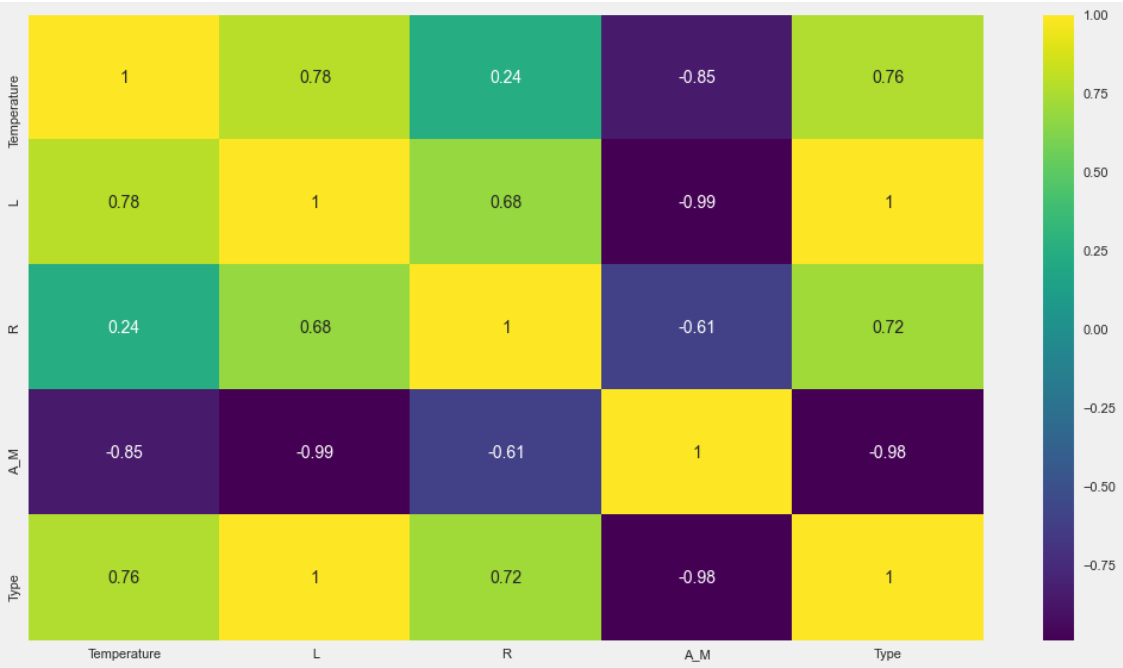
[101]:
```
sns.scatterplot(x = 'R',y = 'Type',data = f)
plt.show()
```



[102]:
```
sns.scatterplot(x = 'A_M',y = 'Type',data = f)
plt.show()
```



[103]:
```
plt.figure(figsize=(16,9))
x = g.drop(['Color','Spectral_Class'],axis = 1)
ax = sns.heatmap(g.corr(),annot = True,cmap = 'viridis')
plt.show()
```



[104]:
```
red_df = df[df['Color'] == 'Red']
red_df
```

[104]:

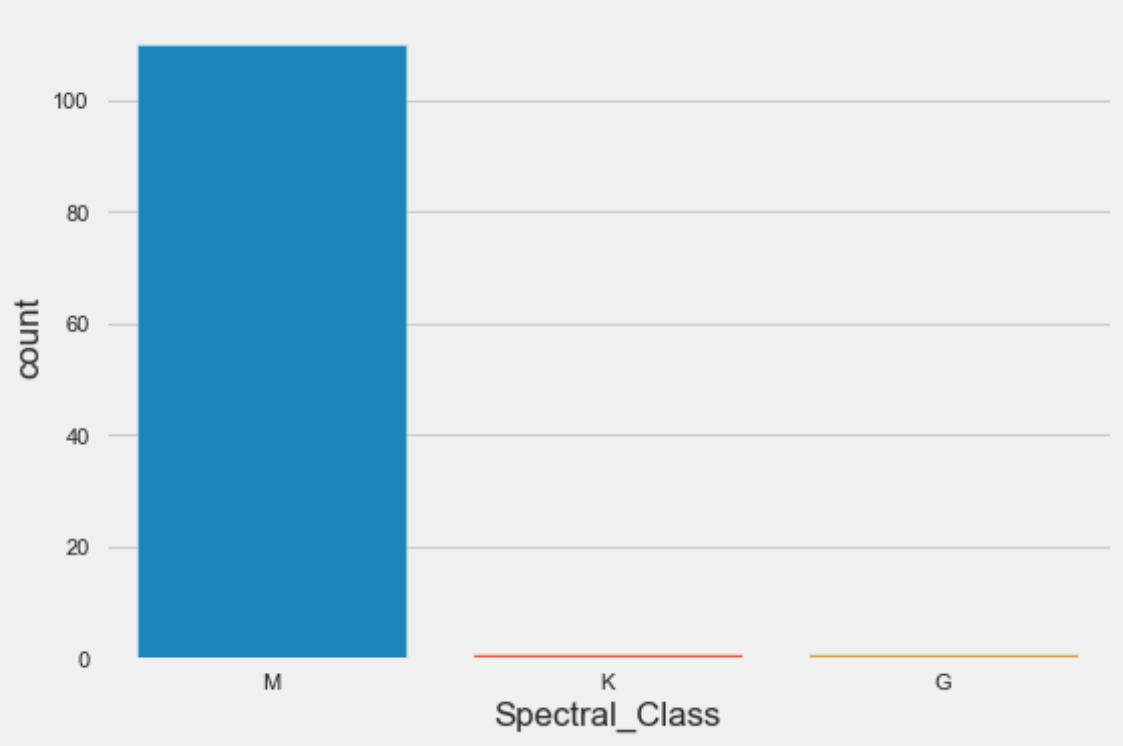
	Temperature	L	R	A_M	Color	Spectral_Class	Type
0	3068	0.002400	0.1700	16.120	Red	M	0
1	3042	0.000500	0.1542	16.600	Red	M	0
2	2600	0.000300	0.1020	18.700	Red	M	0
3	2800	0.000200	0.1600	16.650	Red	M	0
4	1939	0.000138	0.1030	20.060	Red	M	0
..	…	…	…	…	…	…	…
195	3598	0.002700	0.6700	13.667	Red	M	1
196	3142	0.001320	0.2580	14.120	Red	M	1
197	3496	0.001250	0.3360	14.940	Red	M	1
198	3324	0.006500	0.4710	12.780	Red	M	1
199	3463	0.002700	0.6750	14.776	Red	M	1

[112 rows x 7 columns]

[105]:
```
red_df['Spectral_Class'].value_counts()
```

[105]:
```
M    110
G      1
K      1
Name: Spectral_Class, dtype: int64
```

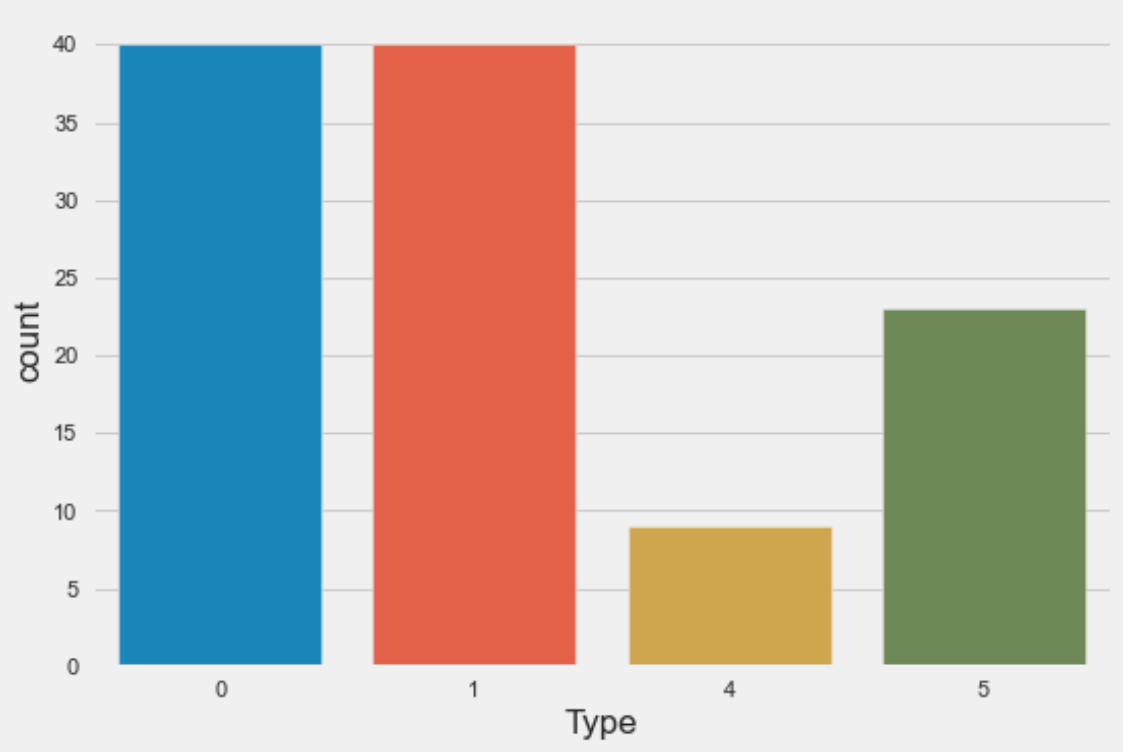
[106]:
```
sns.countplot(x = 'Spectral_Class',data = red_df)
plt.show()
```



[107]:
```
red_df['Type'].value_counts()
```

[107]:
```
0    40
1    40
5    23
4     9
Name: Type, dtype: int64
```

[108]:
```
sns.countplot(x = 'Type',data = red_df)
plt.show()
```



[109]:
```
red_df.sort_values('Temperature').tail(10)
```

[109]:

	Temperature	L	R	A_M	Color	Spectral_Class	Type
54	3650	310000.00000	1324.00	-7.79	Red	M	5
56	3660	363000.00000	1673.00	-11.92	Red	M	5
17	3692	0.00367	0.47	10.80	Red	M	1
51	3750	283000.00000	1260.00	-7.63	Red	M	5
58	3752	209000.00000	955.00	-11.24	Red	M	5
119	3780	200000.00000	1324.00	-10.70	Red	M	5
40	3826	200000.00000	19.00	-6.93	Red	M	4
52	3834	272000.00000	1183.00	-9.20	Red	M	5
116	4015	282000.00000	1534.00	-11.39	Red	K	5
118	6850	229000.00000	1467.00	-10.07	Red	G	5

[110]:
```
red_df.sort_values('Temperature').head(10)
```

[110]:

	Temperature	L	R	A_M	Color	Spectral_Class	Type
4	1939	0.000138	0.1030	20.060	Red	M	0
2	2600	0.000300	0.1020	18.700	Red	M	0
7	2600	0.000400	0.0960	17.400	Red	M	0
78	2621	0.000600	0.0980	12.810	Red	M	1
6	2637	0.000730	0.1270	17.220	Red	M	0
8	2650	0.000690	0.1100	17.450	Red	M	0

```
14          2650  0.000600  0.1400  11.782   Red              M     1
9           2700  0.000180  0.1300  16.050   Red              M     0
129         2731  0.000437  0.0856  18.090   Red              M     0
68          2774  0.000360  0.1180  17.390   Red              M     0
```

[111]:
```
fig = px.histogram(red_df, 'Temperature',
                   color = 'Spectral_Class')
fig.show()
```

[112]:
```
fig = px.histogram(red_df, 'Temperature',
                   color = 'Type')
fig.show()
```

[113]:
```
fig = px.histogram(red_df, 'L',
                   color = 'Type')
fig.show()
```

[114]:
```
fig = px.histogram(red_df, 'L',
                   color = 'Spectral_Class')
fig.show()
```

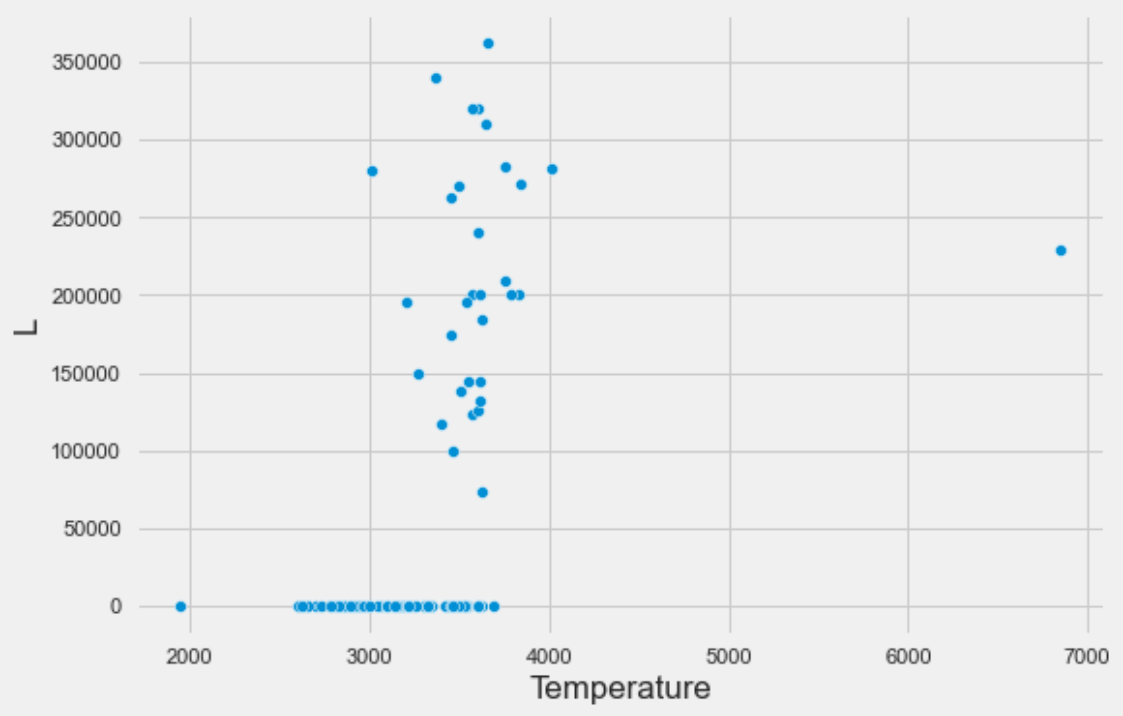
[115]:
```
fig = px.histogram(red_df, 'R',
                   color = 'Type')
fig.show()
```

[116]:
```
fig = px.histogram(red_df, 'R',
                   color = 'Spectral_Class')
fig.show()
```

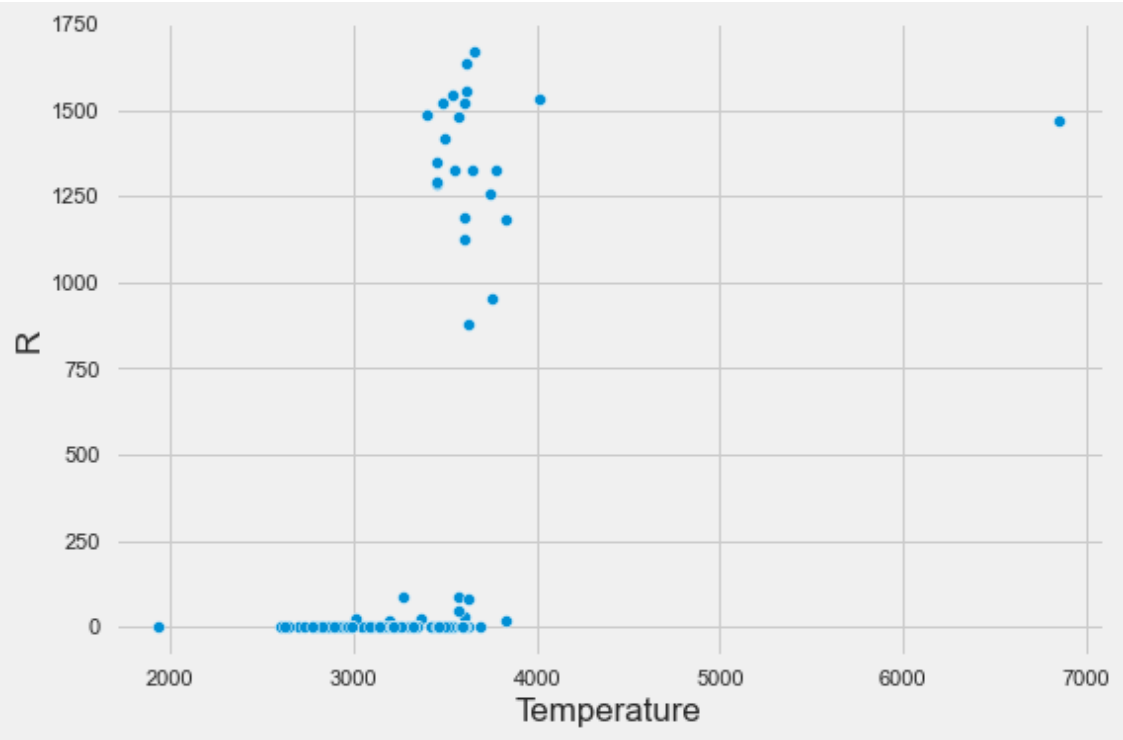
[117]:
```
fig = px.histogram(red_df, 'A_M',
                   color = 'Type')
fig.show()
```

[118]:
```
fig = px.histogram(red_df, 'A_M',
                   color = 'Spectral_Class')
fig.show()
```

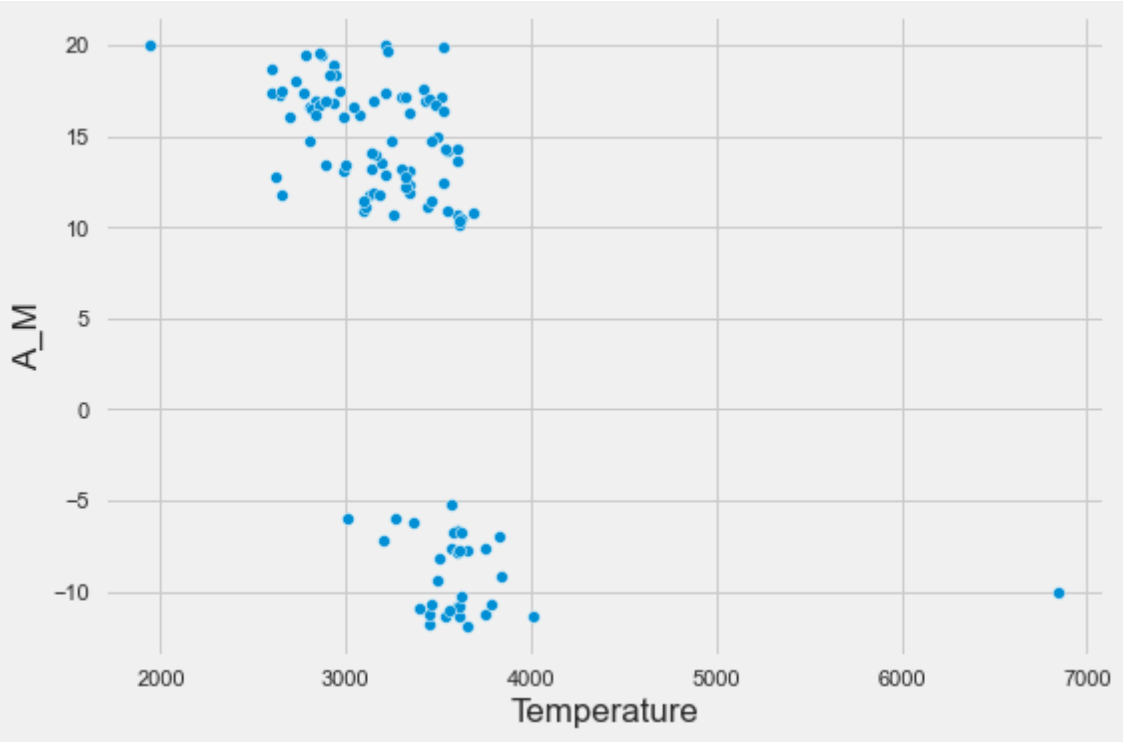
[119]:
```
sns.scatterplot(x = 'Temperature',y = 'L',data = red_df)
plt.show()
```



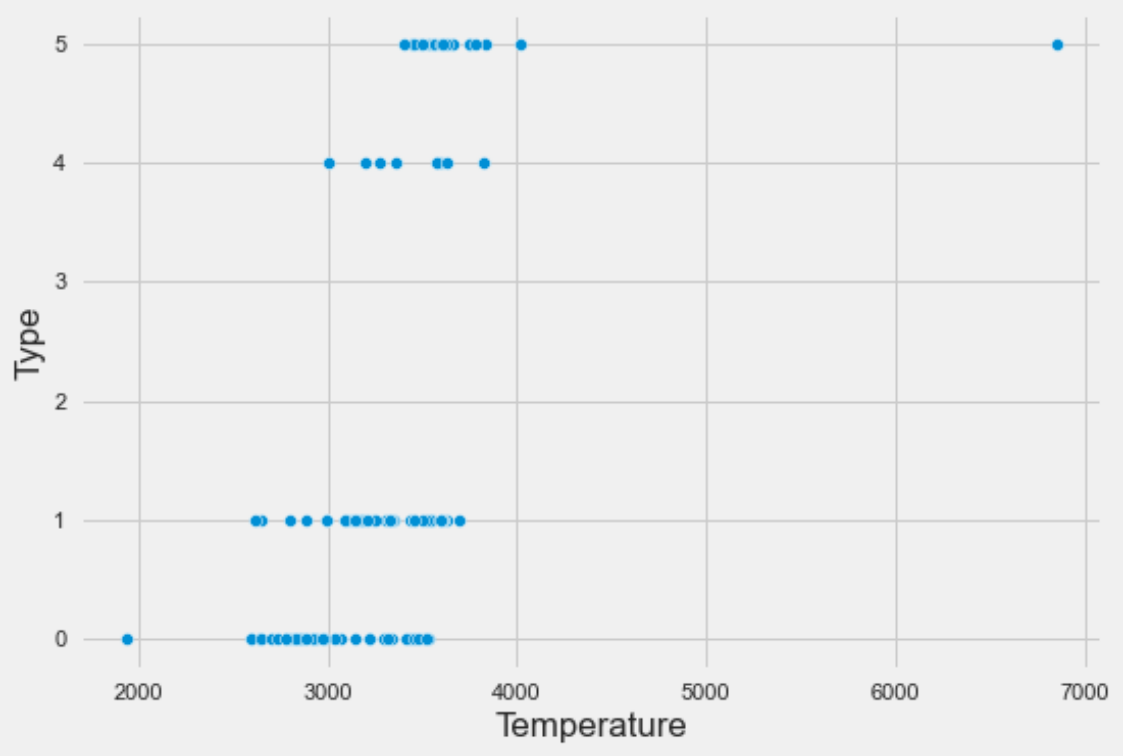
[120]:
```
sns.scatterplot(x = 'Temperature',y = 'R',data = red_df)
plt.show()
```



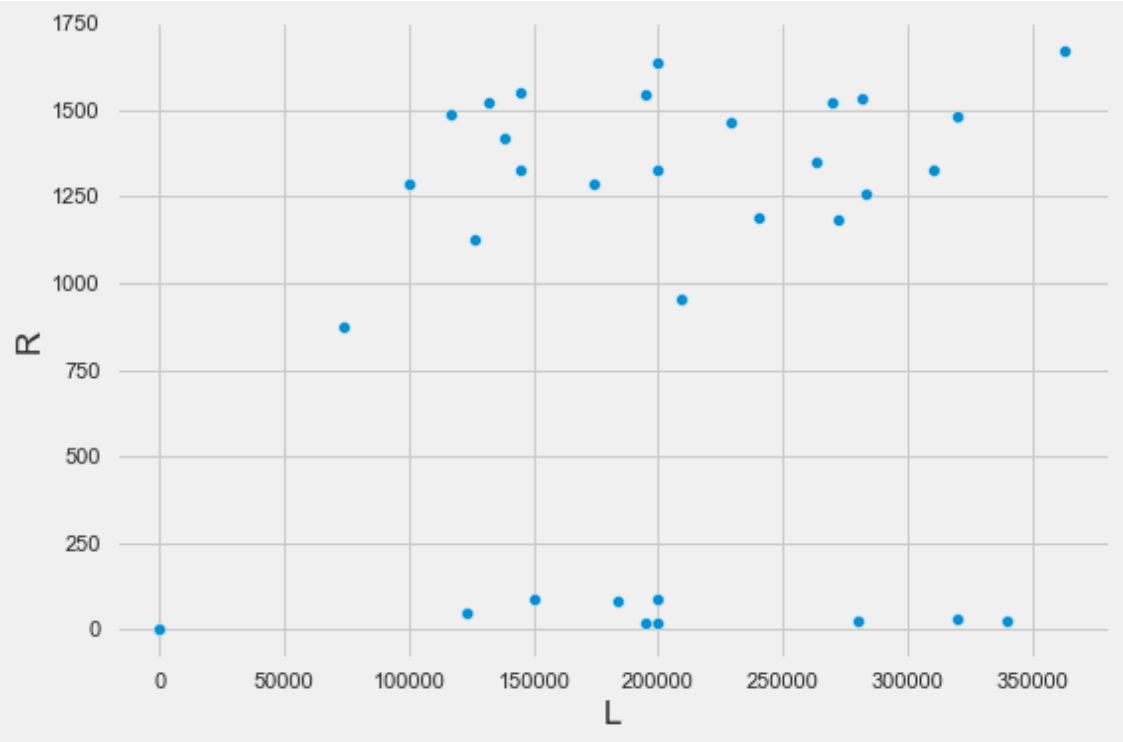
[121]:
```
sns.scatterplot(x = 'Temperature',y = 'A_M',data = red_df)
plt.show()
```



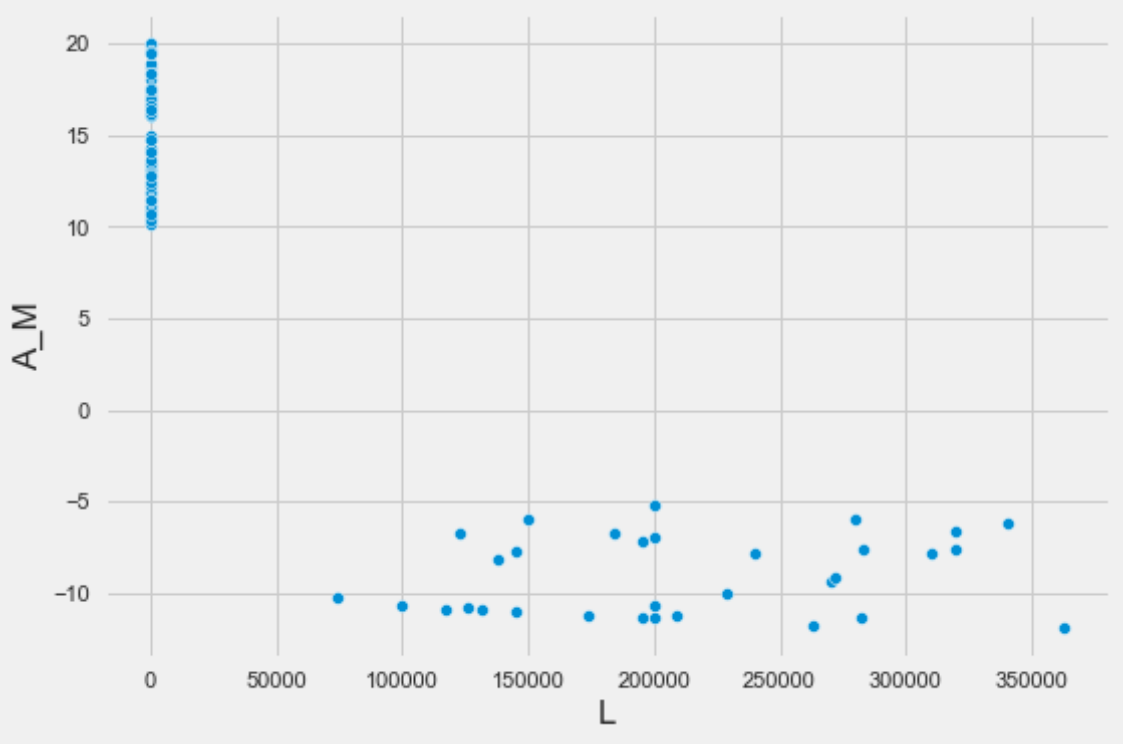
[122]:
```
sns.scatterplot(x = 'Temperature',y = 'Type',data = red_df)
plt.show()
```



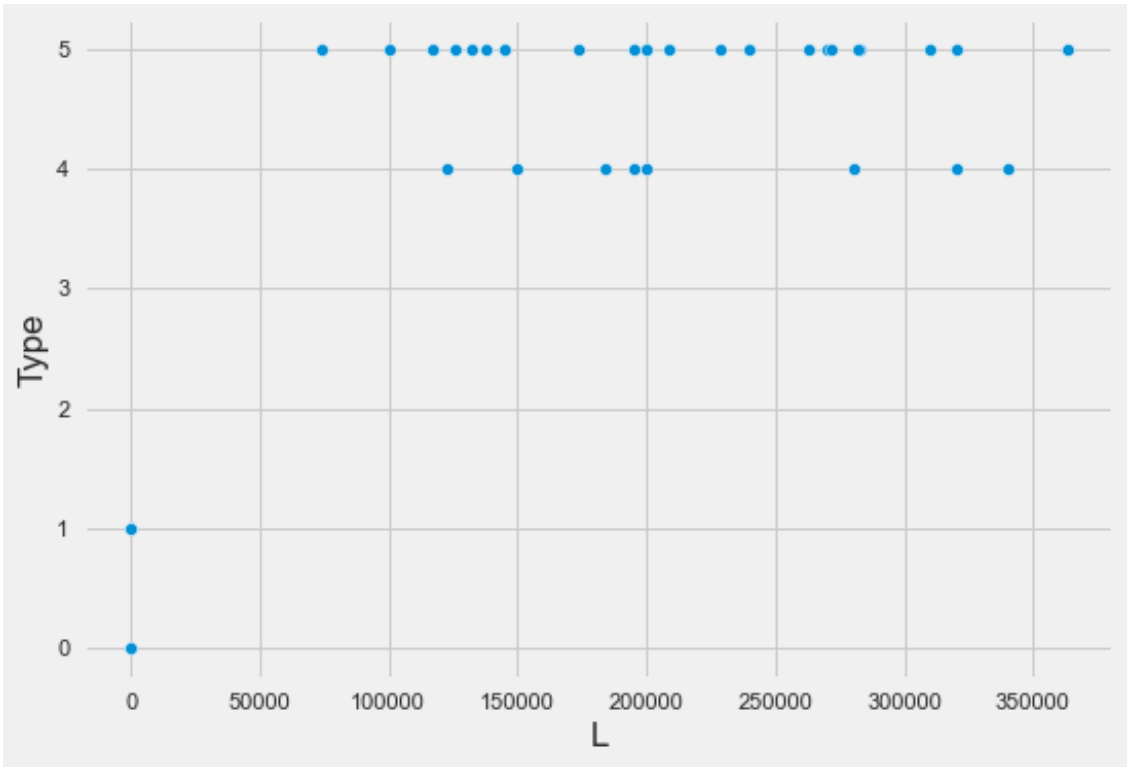
[123]:
```
sns.scatterplot(x = 'L',y = 'R',data = red_df)
plt.show()
```



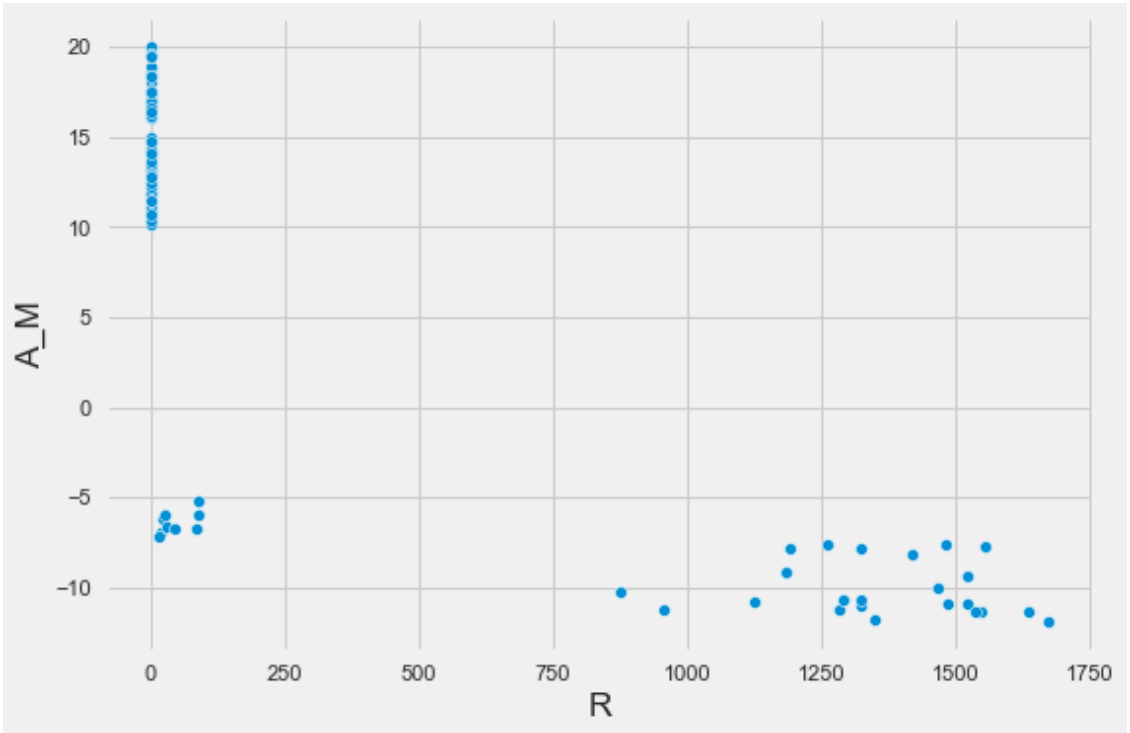
[124]:
```
sns.scatterplot(x = 'L',y = 'A_M',data = red_df)
plt.show()
```



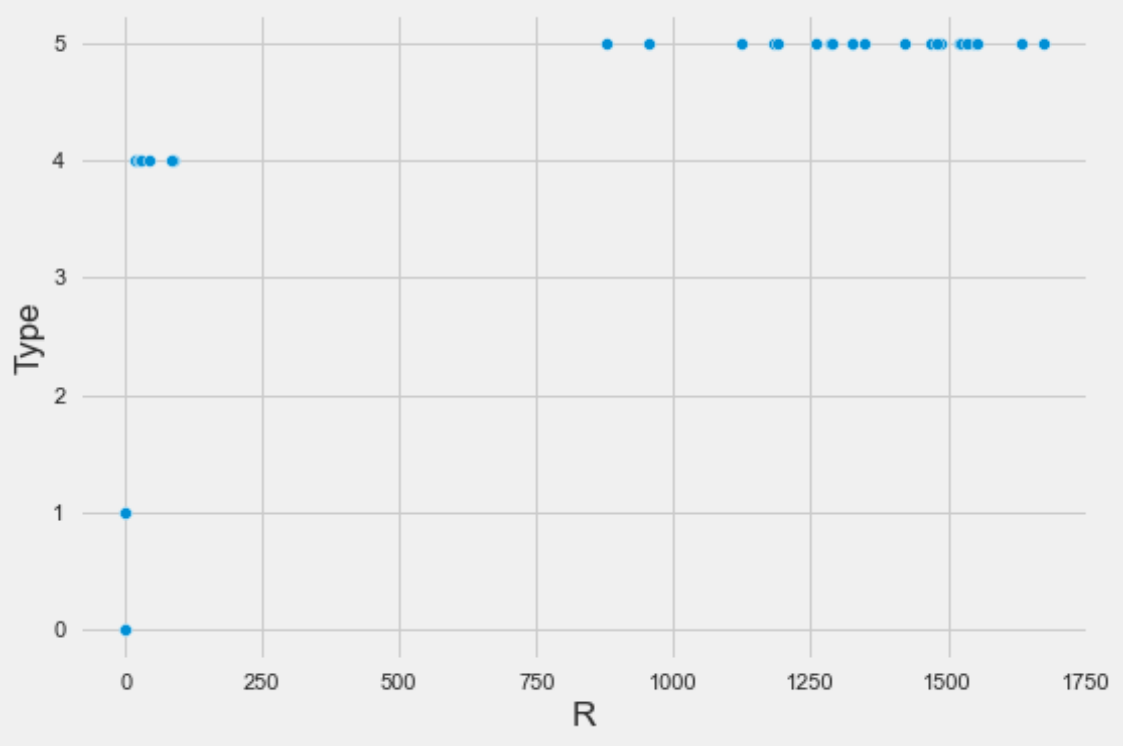
[125]:
```
sns.scatterplot(x = 'L',y = 'Type',data = red_df)
plt.show()
```



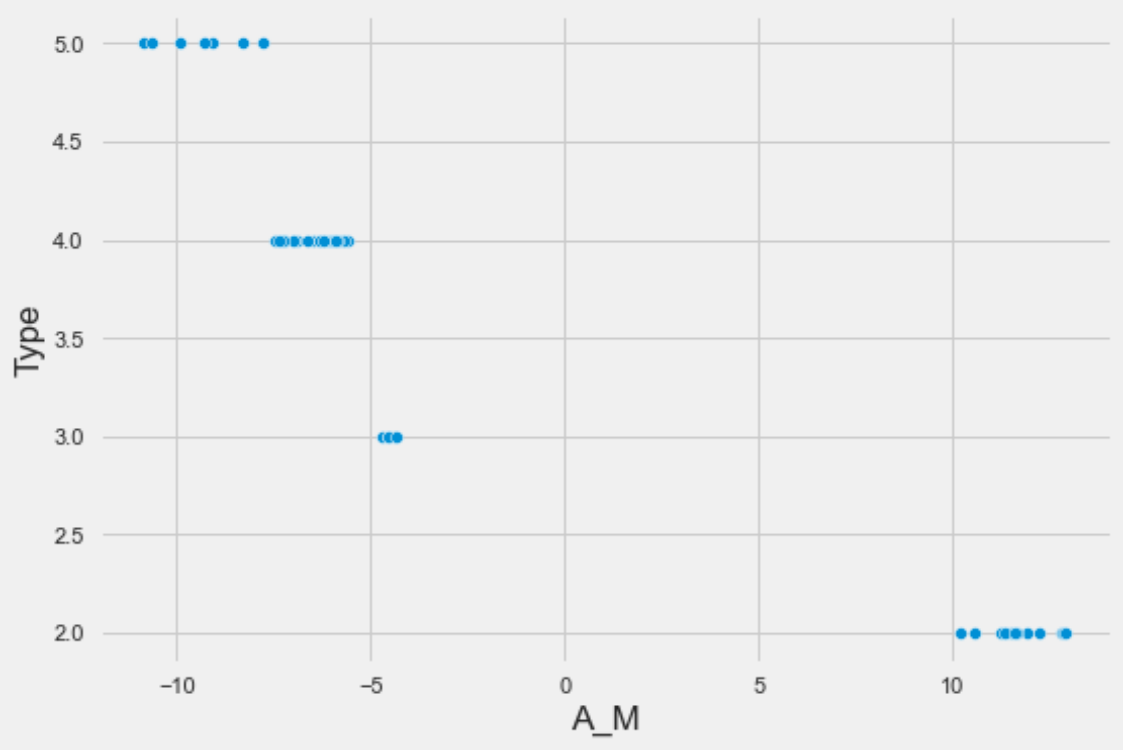
[126]:
```
sns.scatterplot(x = 'R',y = 'A_M',data = red_df)
plt.show()
```



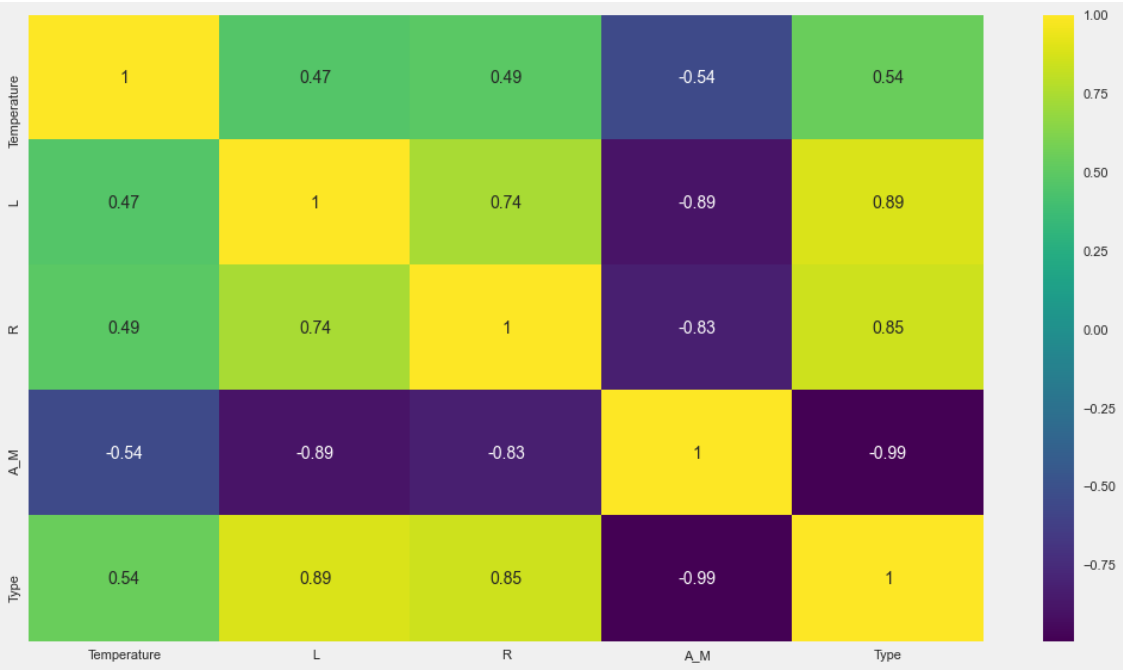
[127]:
```
sns.scatterplot(x = 'R',y = 'Type',data =red_df)
plt.show()
```



[128]:
```
sns.scatterplot(x = 'A_M',y = 'Type',data = blue_df)
plt.show()
```



[129]:
```
plt.figure(figsize=(16,9))
x = red_df.drop(['Color','Spectral_Class'],axis = 1)
ax = sns.heatmap(red_df.corr(),annot = True,cmap = 'viridis')
plt.show()
```



[130]:
```
p = red_df[red_df['Spectral_Class'] == 'M']
p
```

[130]:

	Temperature	L	R	A_M	Color	Spectral_Class	Type
0	3068	0.002400	0.1700	16.120	Red	M	0
1	3042	0.000500	0.1542	16.600	Red	M	0
2	2600	0.000300	0.1020	18.700	Red	M	0
3	2800	0.000200	0.1600	16.650	Red	M	0
4	1939	0.000138	0.1030	20.060	Red	M	0
..	…	…	…	…	…	…	…
195	3598	0.002700	0.6700	13.667	Red	M	1
196	3142	0.001320	0.2580	14.120	Red	M	1
197	3496	0.001250	0.3360	14.940	Red	M	1
198	3324	0.006500	0.4710	12.780	Red	M	1
199	3463	0.002700	0.6750	14.776	Red	M	1

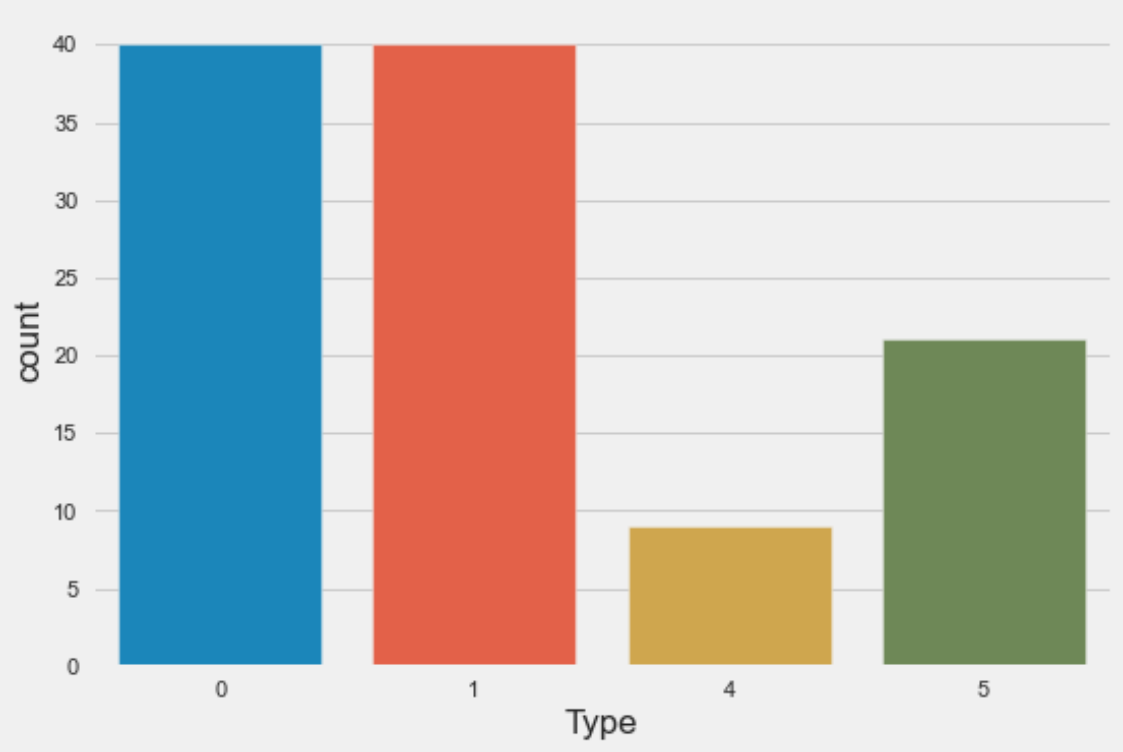
[110 rows x 7 columns]

[131]:
```
red_df[red_df['Spectral_Class'] == 'M']['Type'].value_counts()
```

[131]:
```
0    40
1    40
5    21
4     9
Name: Type, dtype: int64
```

[132]:
```
sns.countplot(x = 'Type',data = p)
```

[132]:
```
<AxesSubplot:xlabel='Type', ylabel='count'>
```



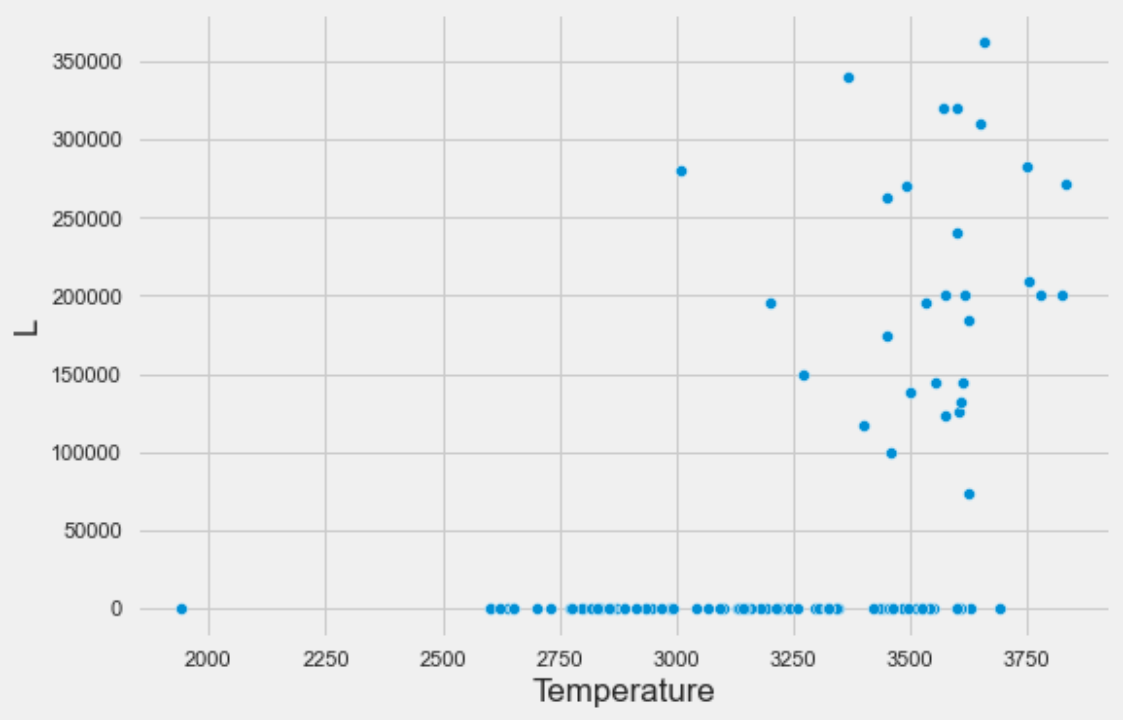
[133]:
```
fig = px.histogram(p, 'Temperature',
                   color = 'Type')
fig.show()
```

[134]:
```
fig = px.histogram(p, 'L',
                   color = 'Type')
fig.show()
```

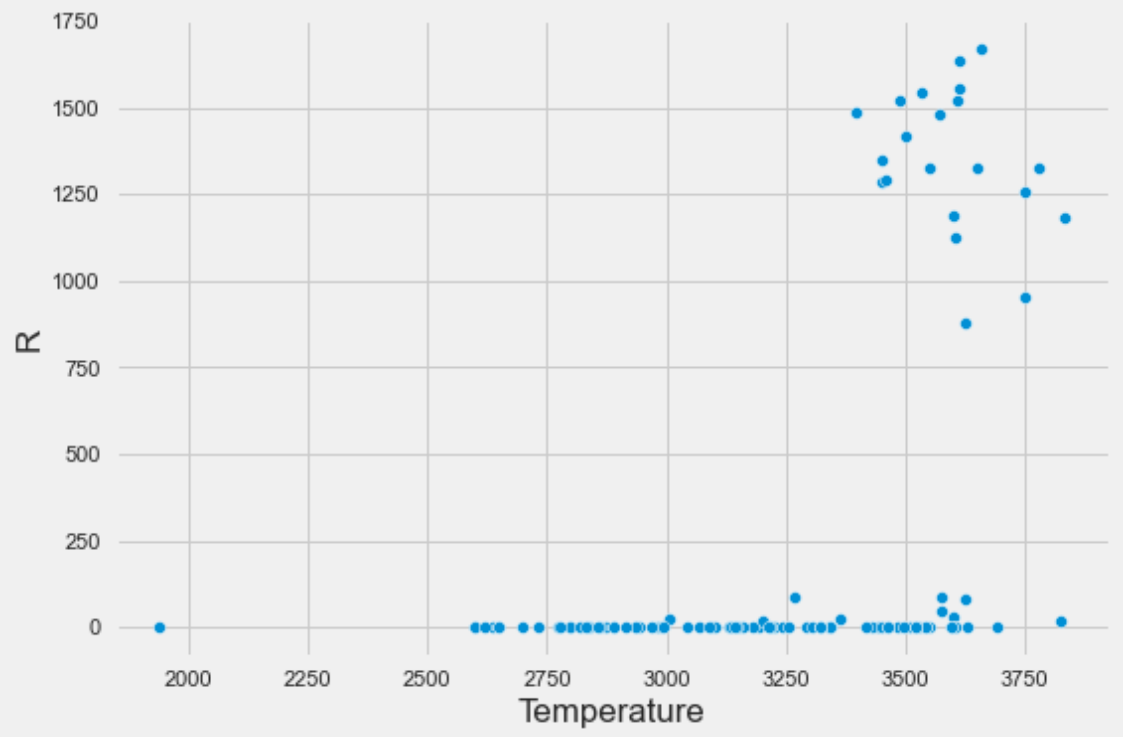
[135]:
```
fig = px.histogram(p, 'R',
                   color = 'Type')
fig.show()
```

[136]:
```
fig = px.histogram(p, 'A_M',
                   color = 'Type')
fig.show()
```

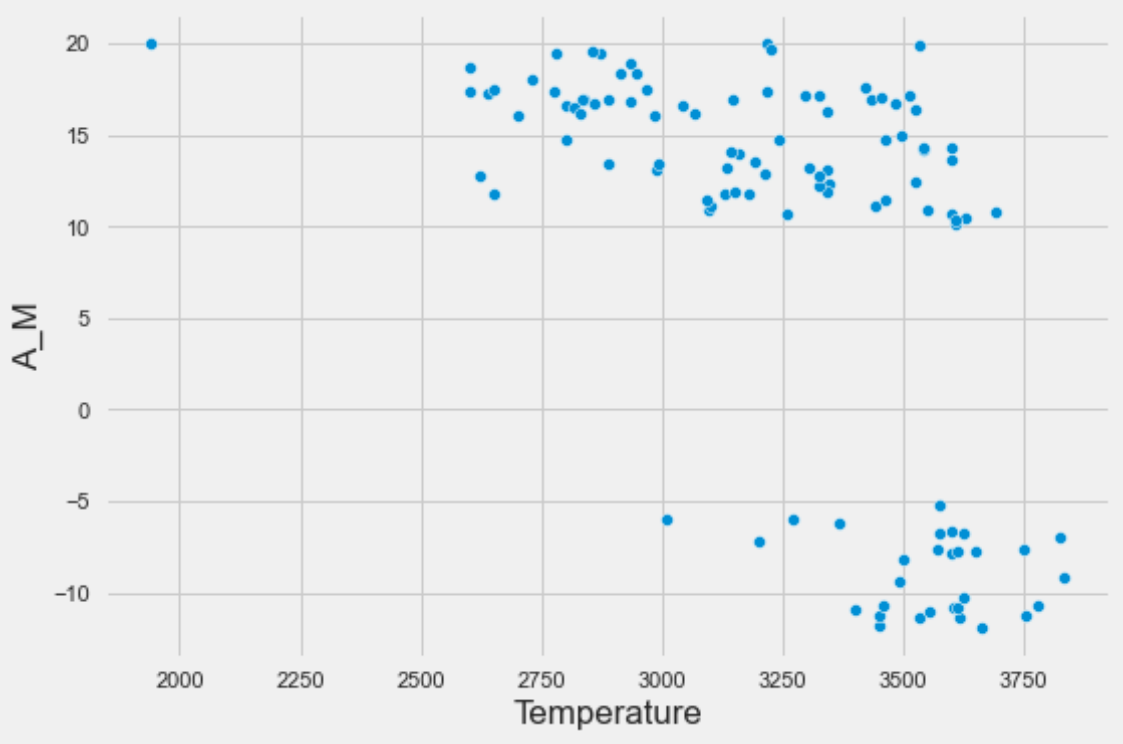
[137]:
```
sns.scatterplot(x = 'Temperature',y = 'L',data = p)
plt.show()
```



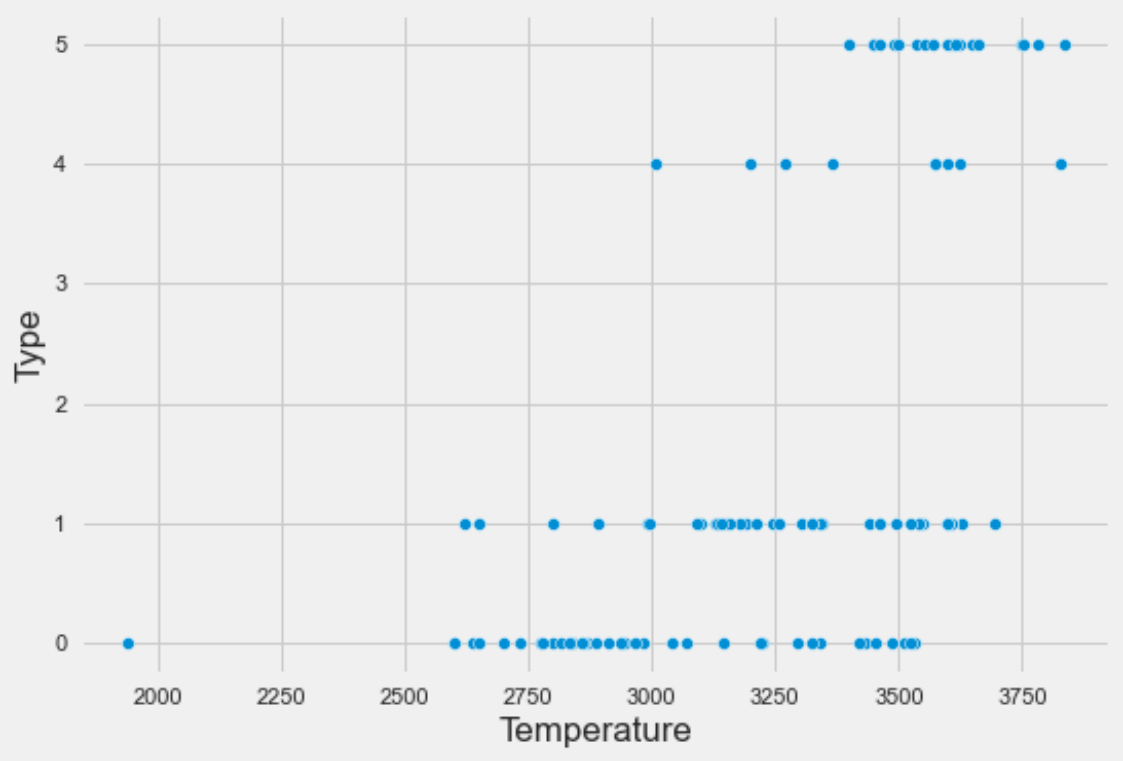
[138]:
```
sns.scatterplot(x = 'Temperature',y = 'R',data = p)
plt.show()
```



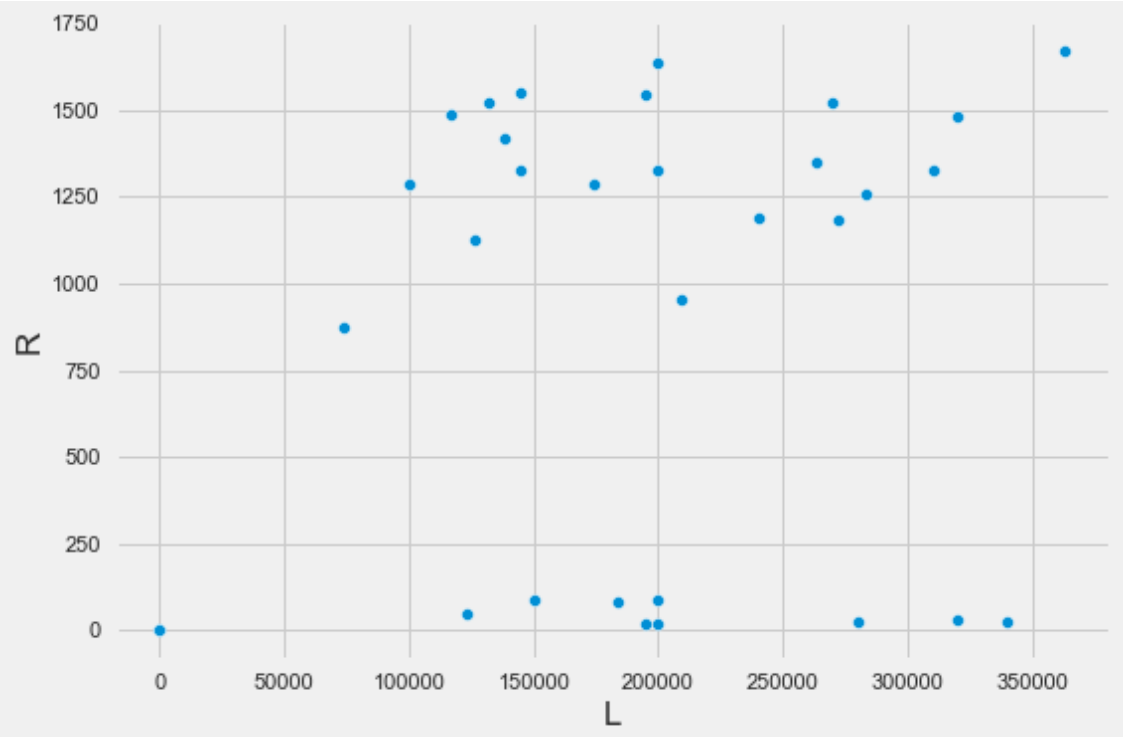
[139]:
```
sns.scatterplot(x = 'Temperature',y = 'A_M',data = p)
plt.show()
```



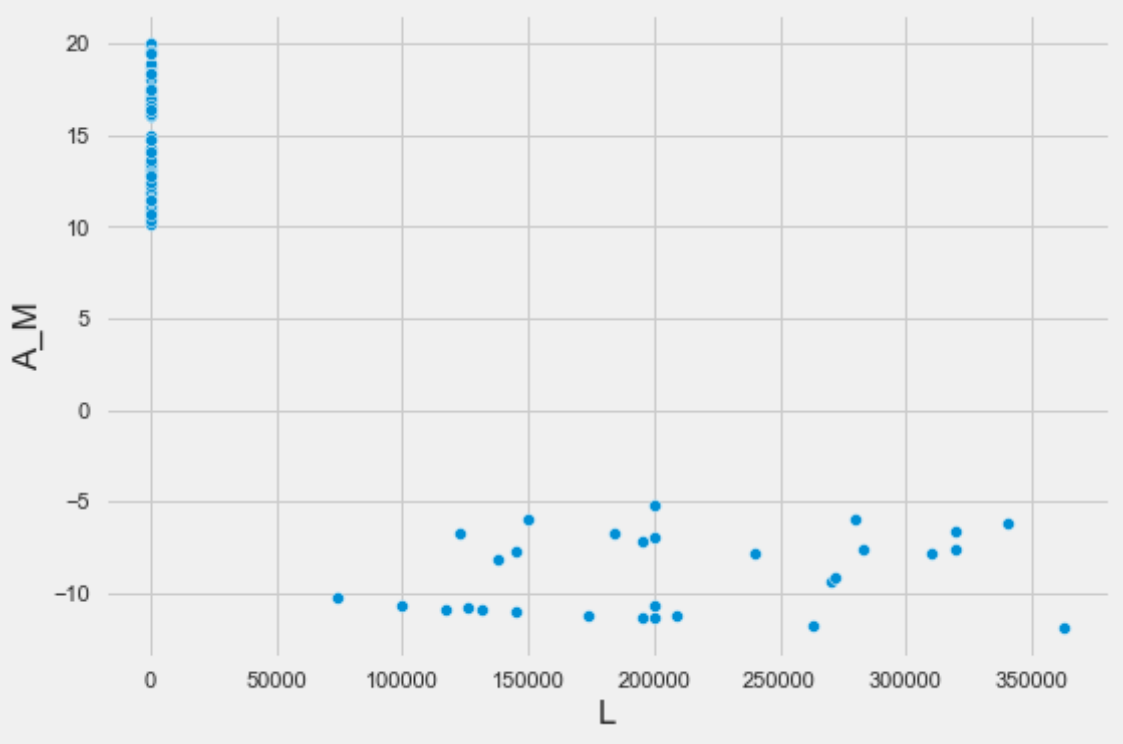
[140]:
```
sns.scatterplot(x = 'Temperature',y = 'Type',data = p)
plt.show()
```



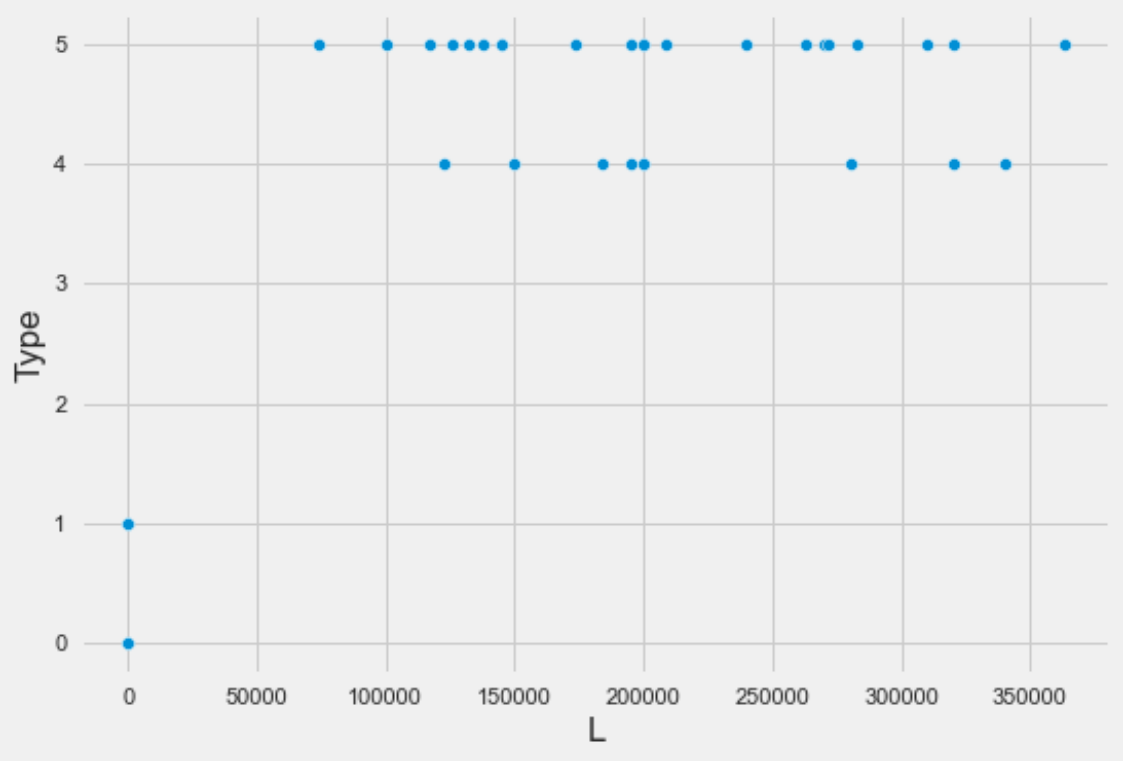
[141]:
```
sns.scatterplot(x = 'L',y = 'R',data = p)
plt.show()
```



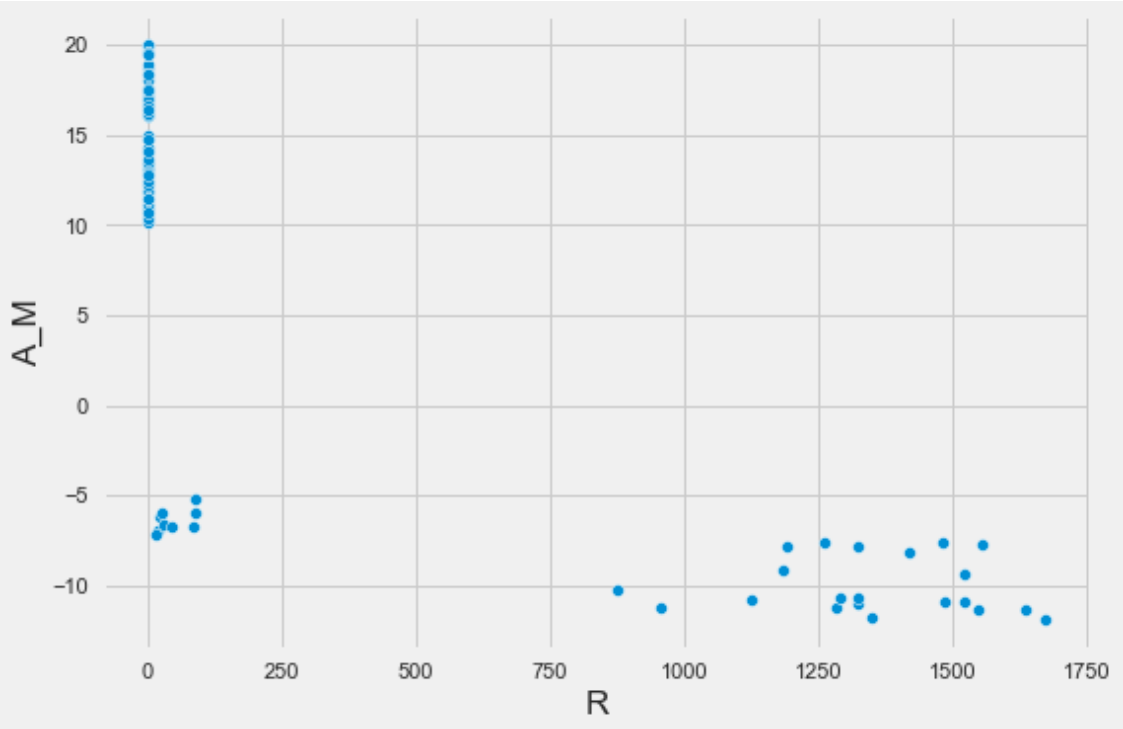
[142]:
```
sns.scatterplot(x = 'L',y = 'A_M',data = p)
plt.show()
```



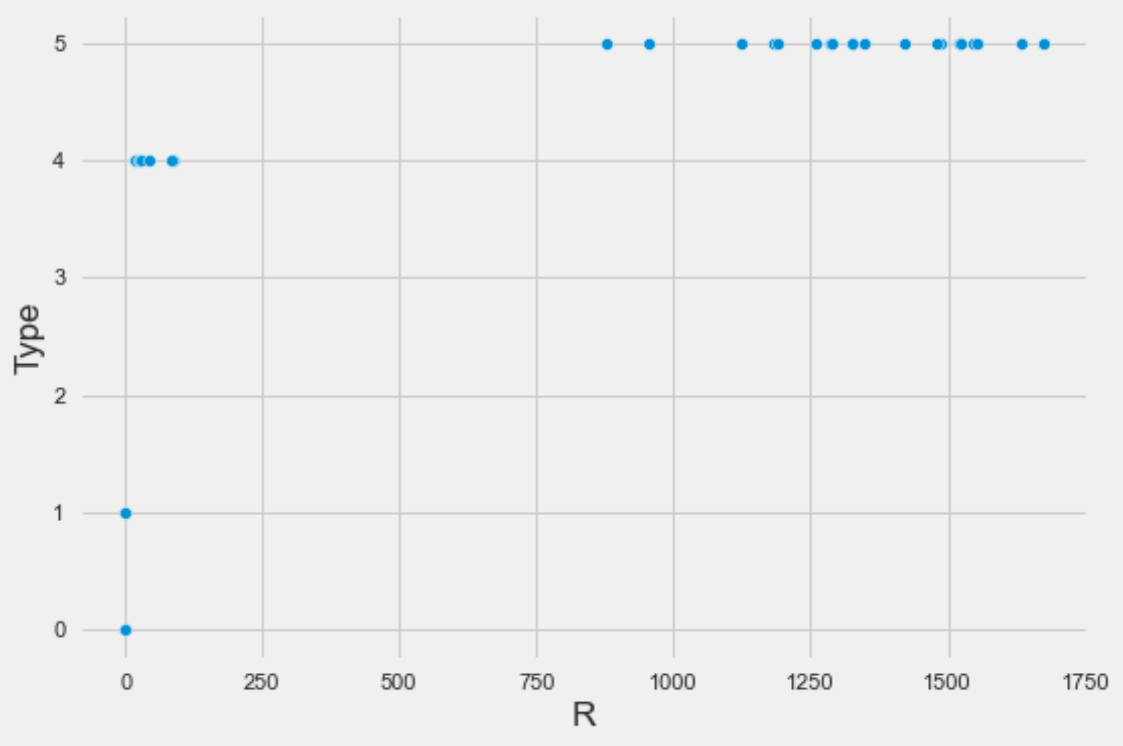
[143]:
```
sns.scatterplot(x = 'L',y = 'Type',data = p)
plt.show()
```



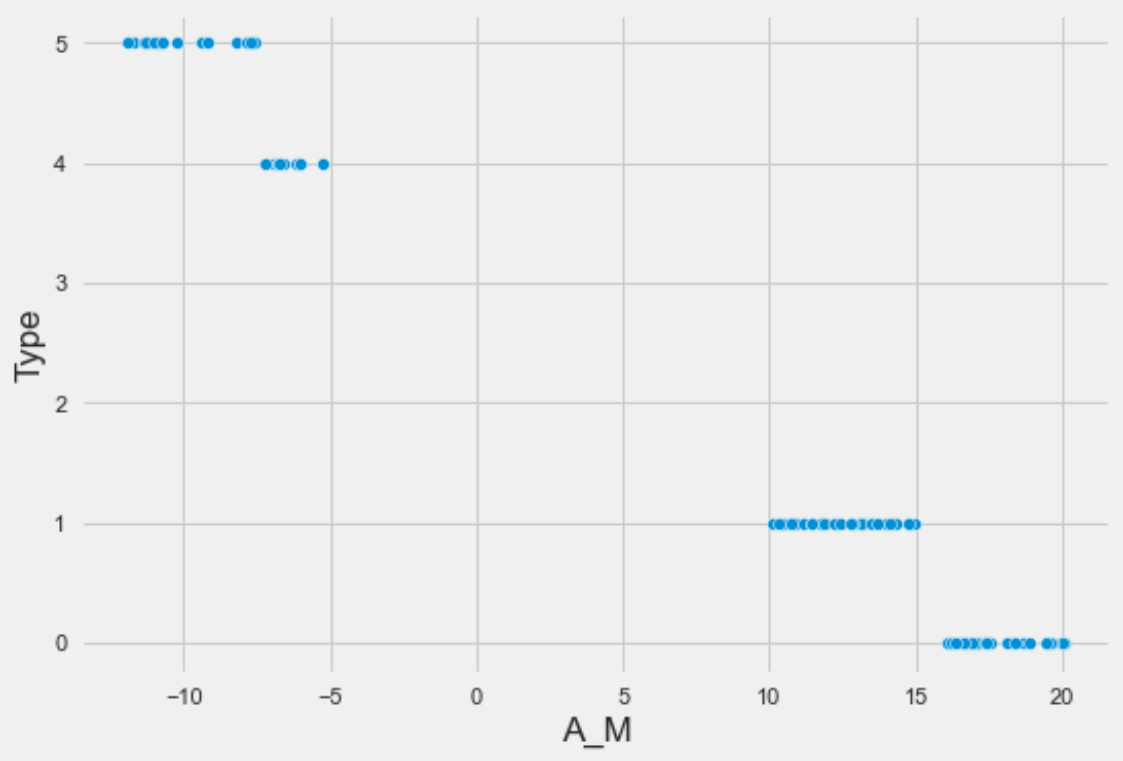
[144]:
```
sns.scatterplot(x = 'R',y = 'A_M',data = p)
plt.show()
```



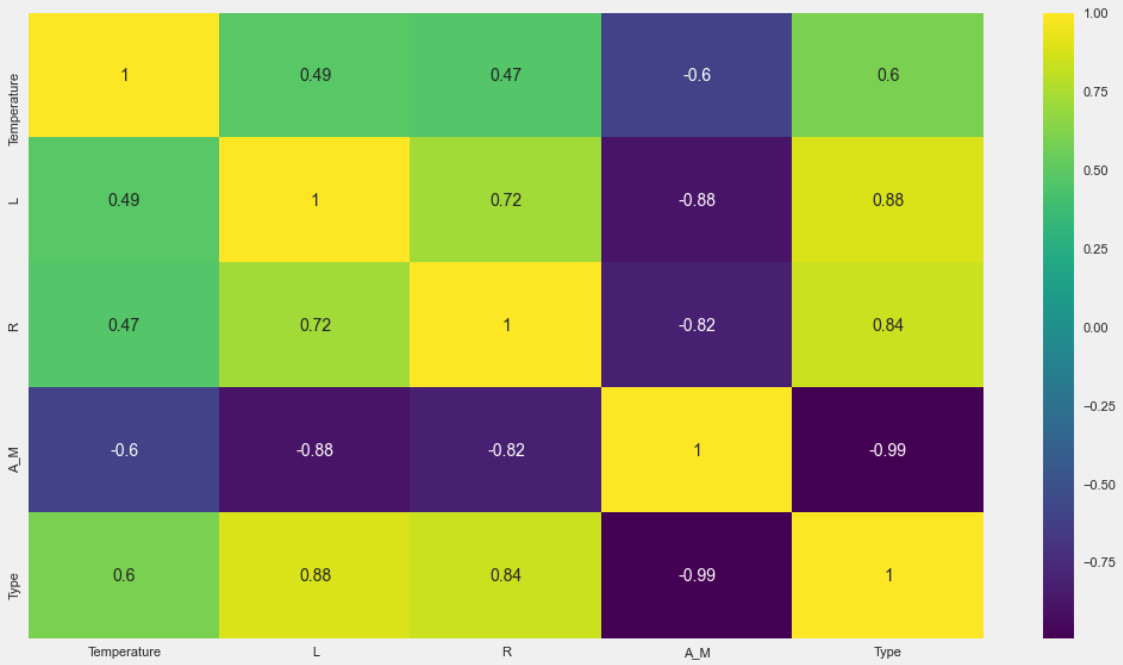
[145]:
```
sns.scatterplot(x = 'R',y = 'Type',data = p)
plt.show()
```



[146]:
```
sns.scatterplot(x = 'A_M',y = 'Type',data = p)
plt.show()
```



[147]:
```
plt.figure(figsize=(16,9))
x = p.drop(['Color','Spectral_Class'],axis = 1)
ax = sns.heatmap(x.corr(),annot = True,cmap = 'viridis')
plt.show()
```



[148]:
```
s = red_df[red_df['Spectral_Class'] == 'K']
s
```

[148]:

	Temperature	L	R	A_M	Color	Spectral_Class	Type
116	4015	282000.0	1534.0	-11.39	Red	K	5

[149]:
```
n = red_df[red_df['Spectral_Class'] == 'G']
n
```

[149]:

	Temperature	L	R	A_M	Color	Spectral_Class	Type
118	6850	229000.0	1467.0	-10.07	Red	G	5

[150]:
```
bluew_df = df[df['Color'] == 'Blue-white']
bluew_df
```

[150]:

	Temperature	L	R	A_M	Color	Spectral_Class	\
20	25000	0.05600	0.00840	10.580	Blue-white	B	
24	16500	0.01300	0.01400	11.890	Blue-white	B	
26	8570	0.00081	0.00970	14.200	Blue-white	A	
31	30000	28840.00000	6.30000	-4.200	Blue-white	B	
32	15276	1136.00000	7.20000	-1.970	Blue-white	B	
83	17200	0.00098	0.01500	12.450	Blue-white	B	
84	14100	0.00067	0.00890	12.170	Blue-white	B	
85	9675	0.00045	0.01090	13.980	Blue-white	A	
86	12010	0.00078	0.00920	12.130	Blue-white	B	
87	10980	0.00074	0.00870	11.190	Blue-white	B	
94	9030	45.00000	2.63000	1.450	Blue-white	A	
95	11250	672.00000	6.98000	-2.300	Blue-white	A	
98	12098	689.00000	7.01000	0.020	Blue-white	A	
140	13420	0.00059	0.00981	13.670	Blue-white	B	
143	14520	0.00082	0.00972	11.920	Blue-white	B	
144	11900	0.00067	0.00898	11.380	Blue-white	B	
145	8924	0.00028	0.00879	14.870	Blue-white	A	
146	12912	0.00071	0.00945	12.830	Blue-white	B	
149	12984	0.00088	0.00996	11.230	Blue-white	B	
150	29560	188000.00000	6.02000	-4.010	Blue-white	B	
151	8945	38.00000	2.48700	0.120	Blue-white	A	
152	14060	1092.00000	5.74500	-2.040	Blue-white	A	
153	16390	1278.00000	5.68000	-3.320	Blue-white	B	
154	25070	14500.00000	5.92000	-3.980	Blue-white	B	
155	28700	16790.00000	6.40000	-4.090	Blue-white	B	
156	26140	14520.00000	5.49000	-3.800	Blue-white	B	
157	20120	4720.00000	6.78000	-3.400	Blue-white	B	
158	13023	998.00000	6.21000	-1.380	Blue-white	A	
177	11000	170000.00000	1779.00000	-9.900	Blue-white	B	

178	12100	120000.00000	708.90000	-7.840	Blue-white	B
179	24490	248490.00000	1134.50000	-8.240	Blue-white	B
210	22350	12450.00000	6.36000	-3.670	Blue-white	B
211	10012	552.00000	5.85600	0.013	Blue-white	A
212	13089	788.00000	5.99200	-0.120	Blue-white	A
213	22012	6748.00000	6.64000	-2.550	Blue-white	B
216	9320	29.00000	1.91000	1.236	Blue-white	A
217	19400	10920.00000	6.03000	-3.080	Blue-white	B
218	17140	883.00000	5.65300	-2.640	Blue-white	B
230	24145	382993.00000	1494.00000	-8.840	Blue-white	B
233	27739	849420.00000	1252.00000	-7.590	Blue-white	B
234	21904	748490.00000	1130.00000	-7.670	Blue-white	B

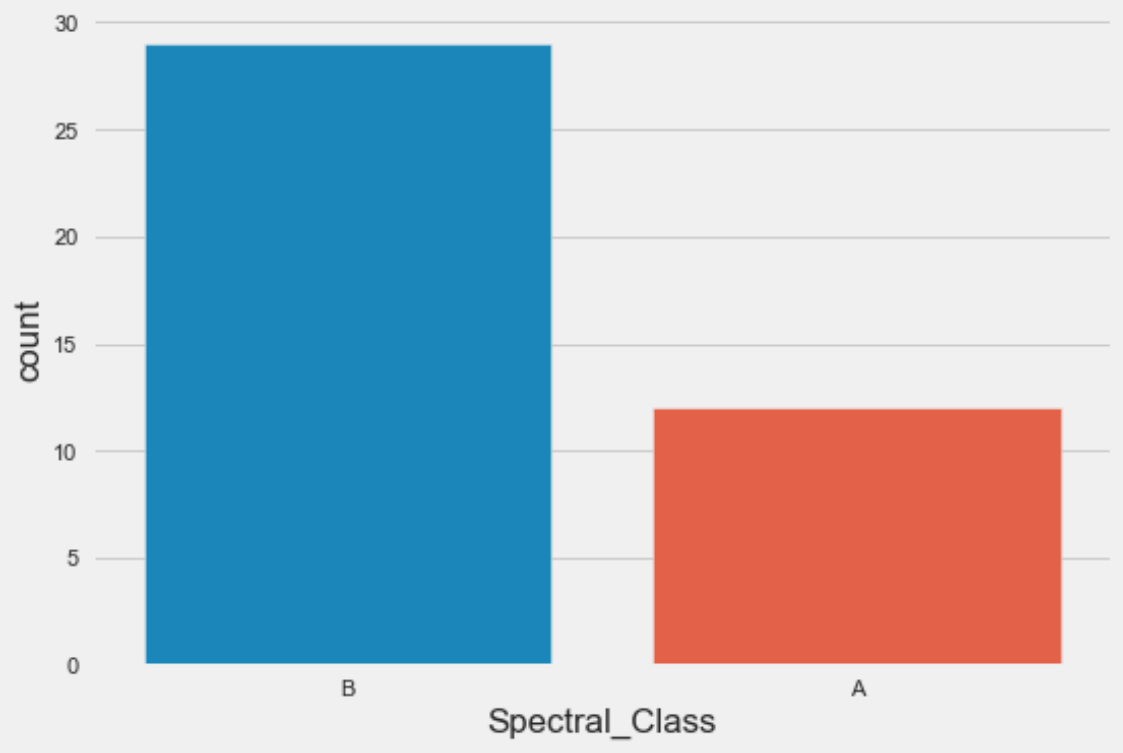
	Type
20	2
24	2
26	2
31	3
32	3
83	2
84	2
85	2
86	2
87	2
94	3
95	3
98	3
140	2
143	2
144	2
145	2
146	2
149	2
150	3
151	3
152	3
153	3
154	3
155	3
156	3
157	3
158	3
177	5
178	5
179	5
210	3
211	3

```
212    3
213    3
216    3
217    3
218    3
230    5
233    5
234    5
```

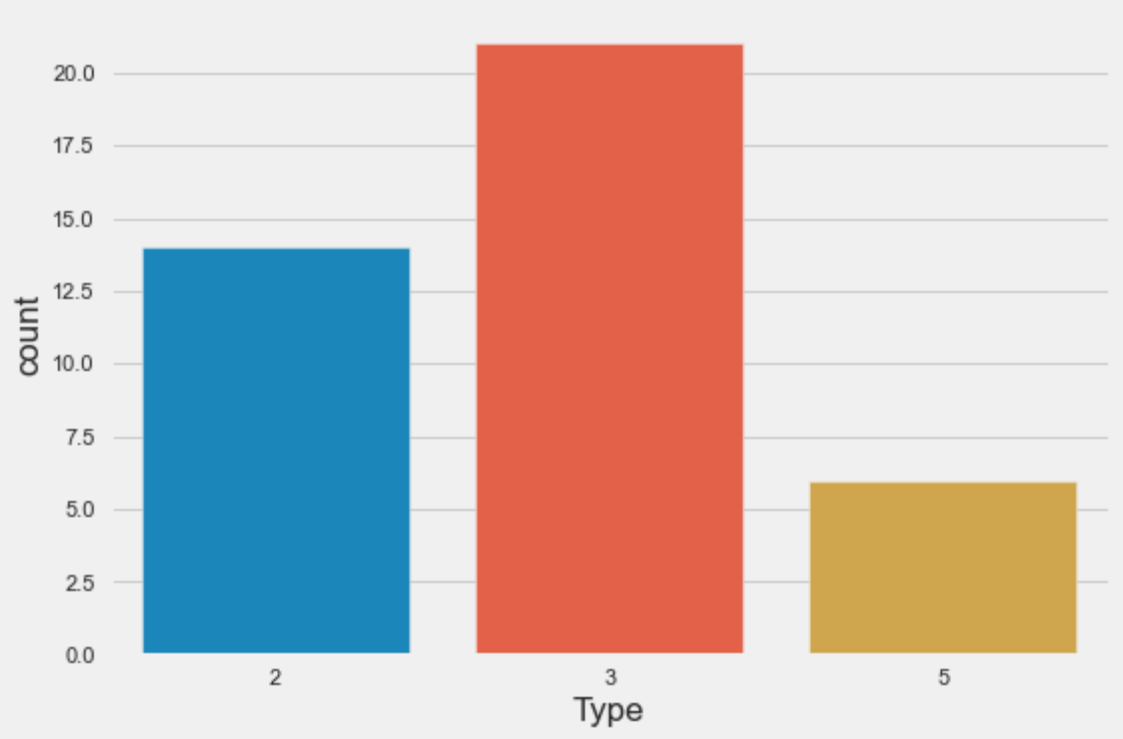
[151]:
```
bluew_df['Spectral_Class'].value_counts()
```

[151]:
```
B    29
A    12
Name: Spectral_Class, dtype: int64
```

[152]:
```
sns.countplot(x = 'Spectral_Class',data = bluew_df)
plt.show()
```



[153]:
```
sns.countplot(x = 'Type',data = bluew_df)
plt.show()
```



[154]:
```
bluew_df.sort_values('Temperature').tail(10)
```

[154]:

	Temperature	L	R	A_M	Color	Spectral_Class	Type
210	22350	12450.000	6.3600	-3.67	Blue-white	B	3
230	24145	382993.000	1494.0000	-8.84	Blue-white	B	5
179	24490	248490.000	1134.5000	-8.24	Blue-white	B	5
20	25000	0.056	0.0084	10.58	Blue-white	B	2
154	25070	14500.000	5.9200	-3.98	Blue-white	B	3
156	26140	14520.000	5.4900	-3.80	Blue-white	B	3
233	27739	849420.000	1252.0000	-7.59	Blue-white	B	5
155	28700	16790.000	6.4000	-4.09	Blue-white	B	3
150	29560	188000.000	6.0200	-4.01	Blue-white	B	3
31	30000	28840.000	6.3000	-4.20	Blue-white	B	3

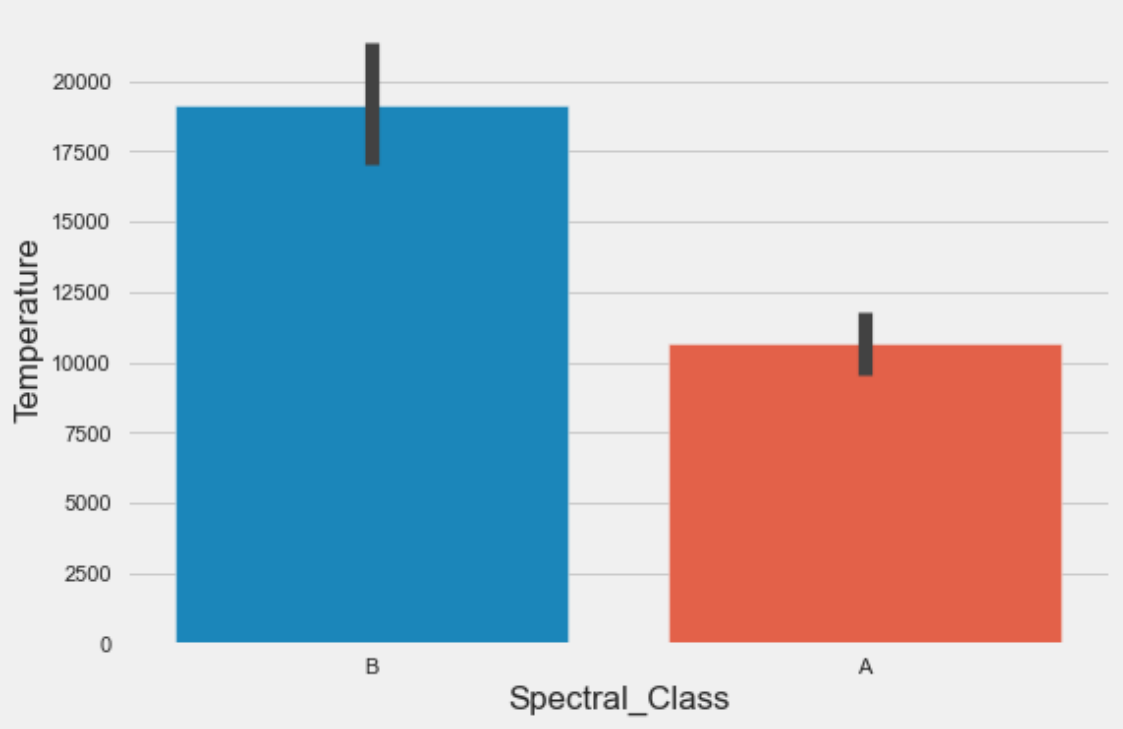
```
150     3
31      3
```

[155]:
```
blue_df.sort_values('Temperature').head(10)
```

[155]:

	Temperature	L	R	A_M	Color	Spectral_Class	Type
163	5752	245000.0	97.0	-6.630	Blue	0	4
165	7282	131000.0	24.0	-7.220	Blue	0	4
164	8927	239000.0	35.0	-7.340	Blue	0	4
169	9373	424520.0	24.0	-5.990	Blue	0	4
222	9383	342940.0	98.0	-6.980	Blue	0	4
226	9892	593900.0	80.0	-7.262	Blue	0	4
227	10930	783930.0	25.0	-6.224	Blue	0	4
104	11096	112000.0	12.0	-5.910	Blue	0	4
161	11567	251000.0	36.0	-6.245	Blue	0	4
162	12675	452000.0	83.0	-5.620	Blue	0	4

[156]:
```
sns.barplot(x = 'Spectral_Class',y = 'Temperature',data = bluew_df)
plt.show()
```



[157]:
```
fig = px.histogram(bluew_df, 'Temperature',
                   color = 'Type')
fig.show()
```

[158]:
```
fig = px.histogram(bluew_df, 'L',
                   color = 'Type')
fig.show()
```

[159]:
```
fig = px.histogram(bluew_df, 'L',
                   color = 'Spectral_Class')
fig.show()
```

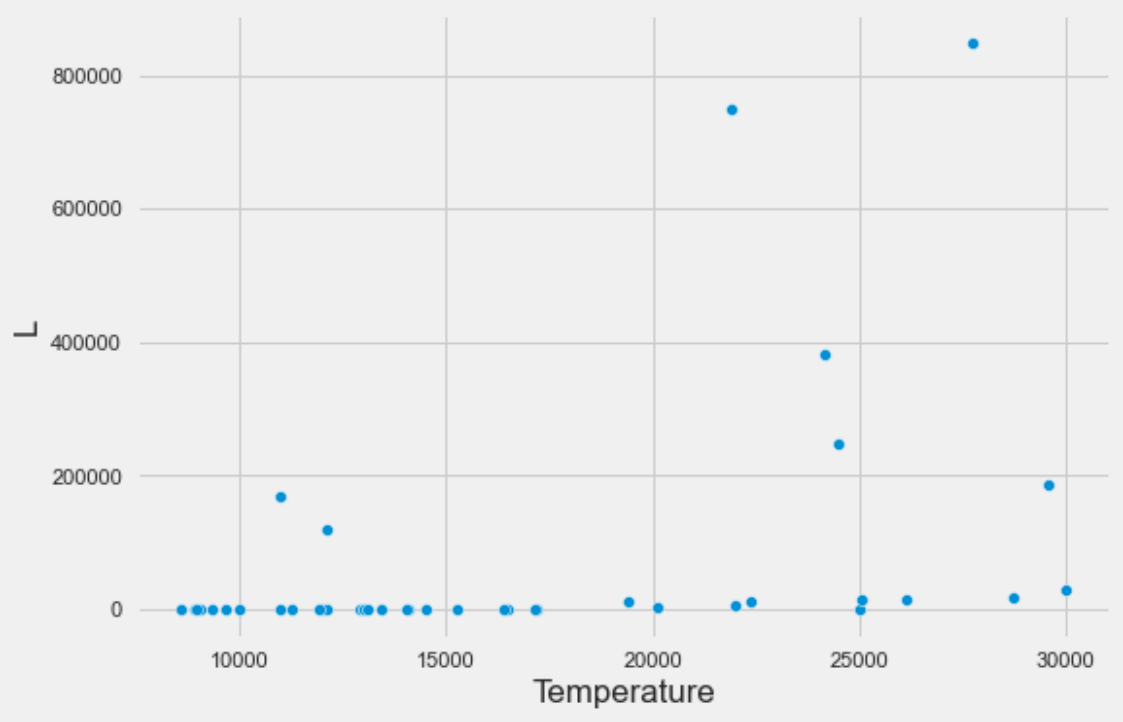
[160]:
```
fig = px.histogram(bluew_df, 'R',
                   color = 'Type')
fig.show()
```

[161]:
```
fig = px.histogram(bluew_df, 'R',
                   color = 'Spectral_Class')
fig.show()
```

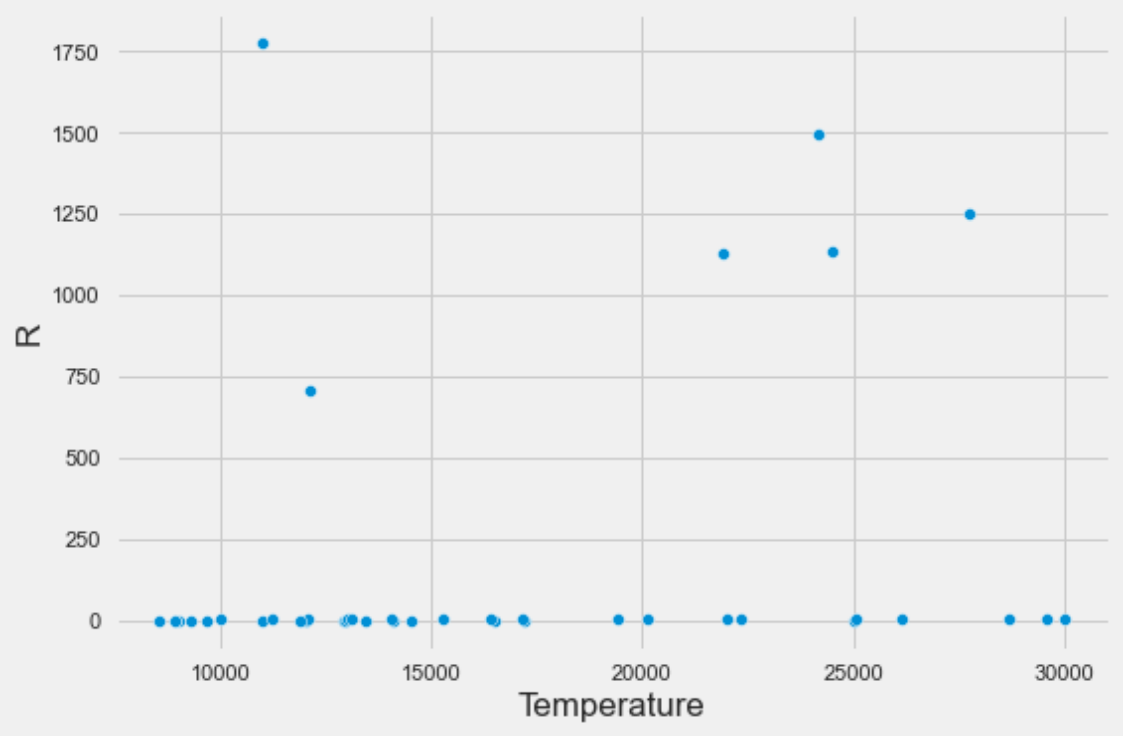
[162]:
```
fig = px.histogram(bluew_df, 'A_M',
                   color = 'Type')
fig.show()
```

[163]:
```
fig = px.histogram(bluew_df, 'A_M',
                   color = 'Spectral_Class')
fig.show()
```

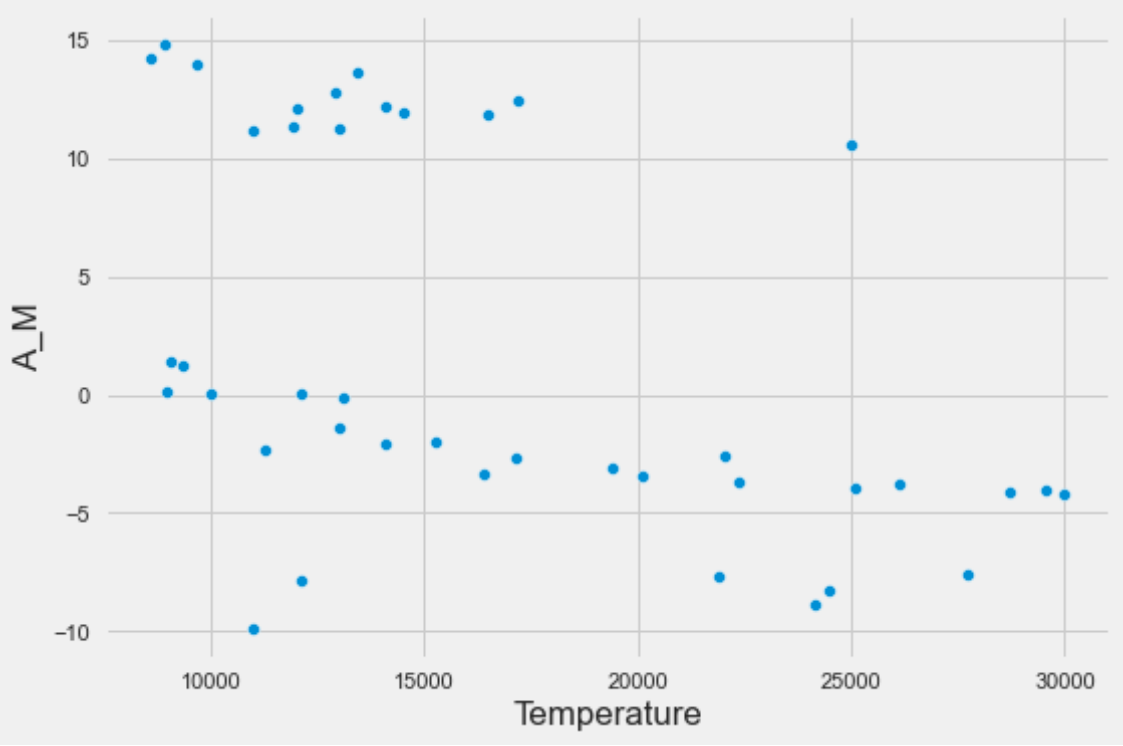
[164]:
```
sns.scatterplot(x = 'Temperature',y = 'L',data = bluew_df)
plt.show()
```



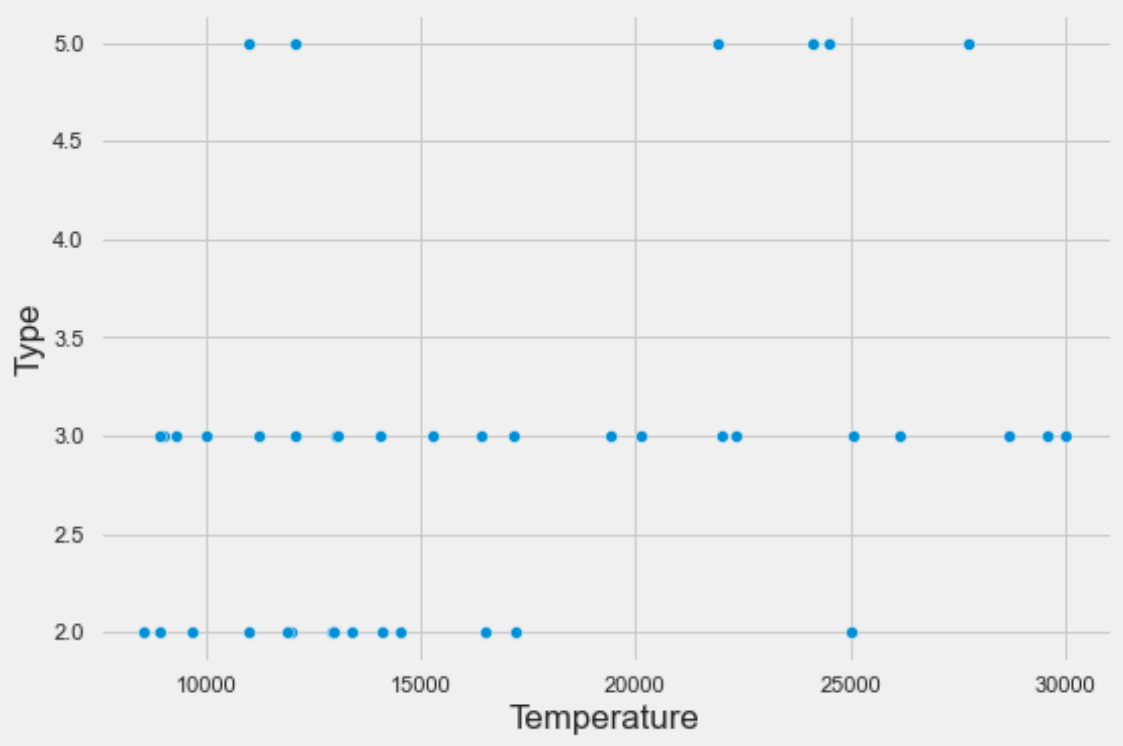
[165]:
```
sns.scatterplot(x = 'Temperature',y = 'R',data = bluew_df)
plt.show()
```



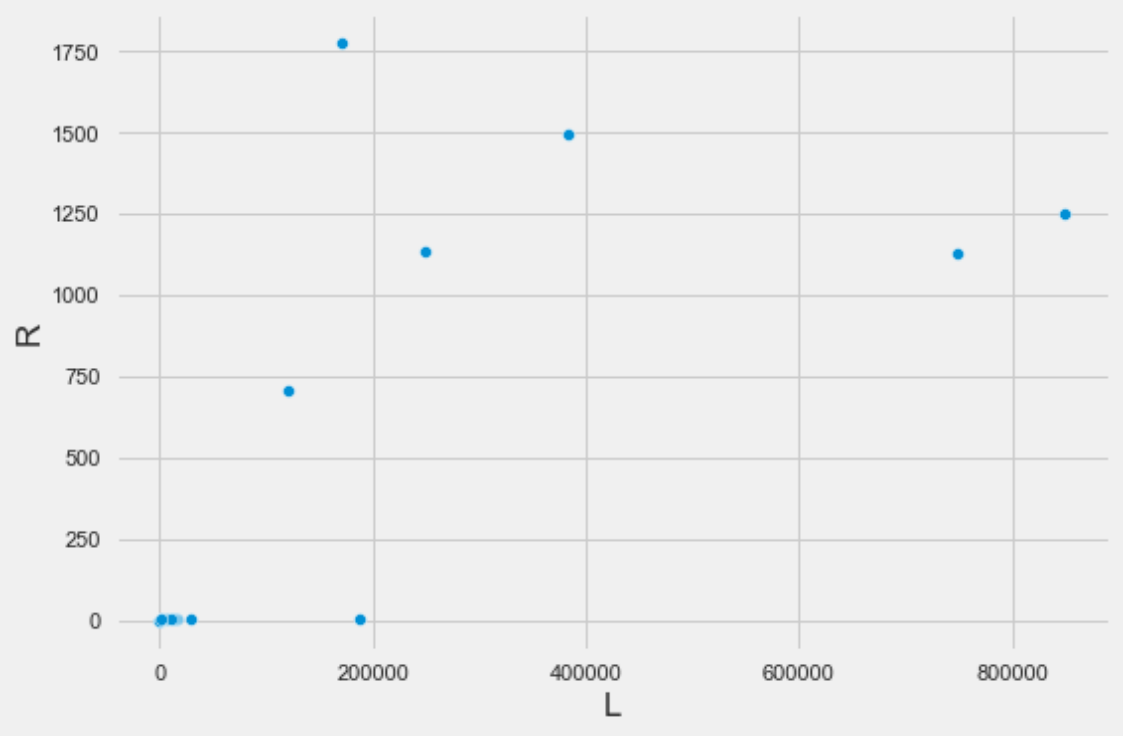
[166]:
```
sns.scatterplot(x = 'Temperature',y = 'A_M',data = bluew_df)
plt.show()
```



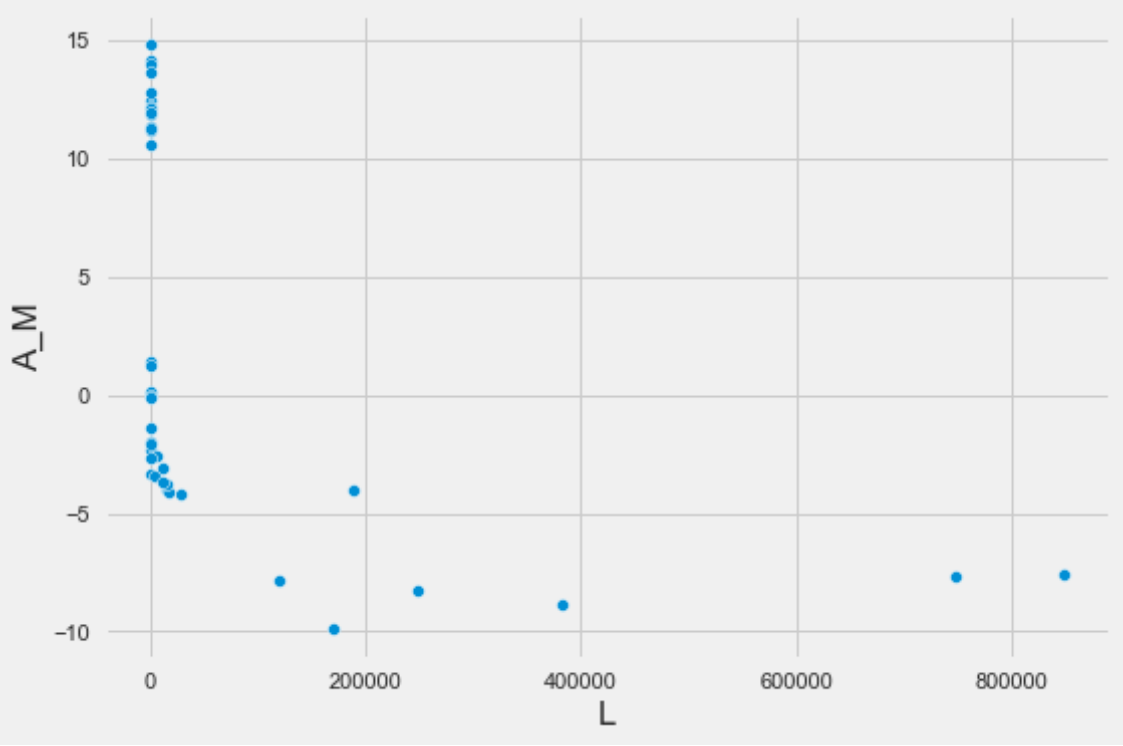
[167]:
```
sns.scatterplot(x = 'Temperature',y = 'Type',data = bluew_df)
plt.show()
```



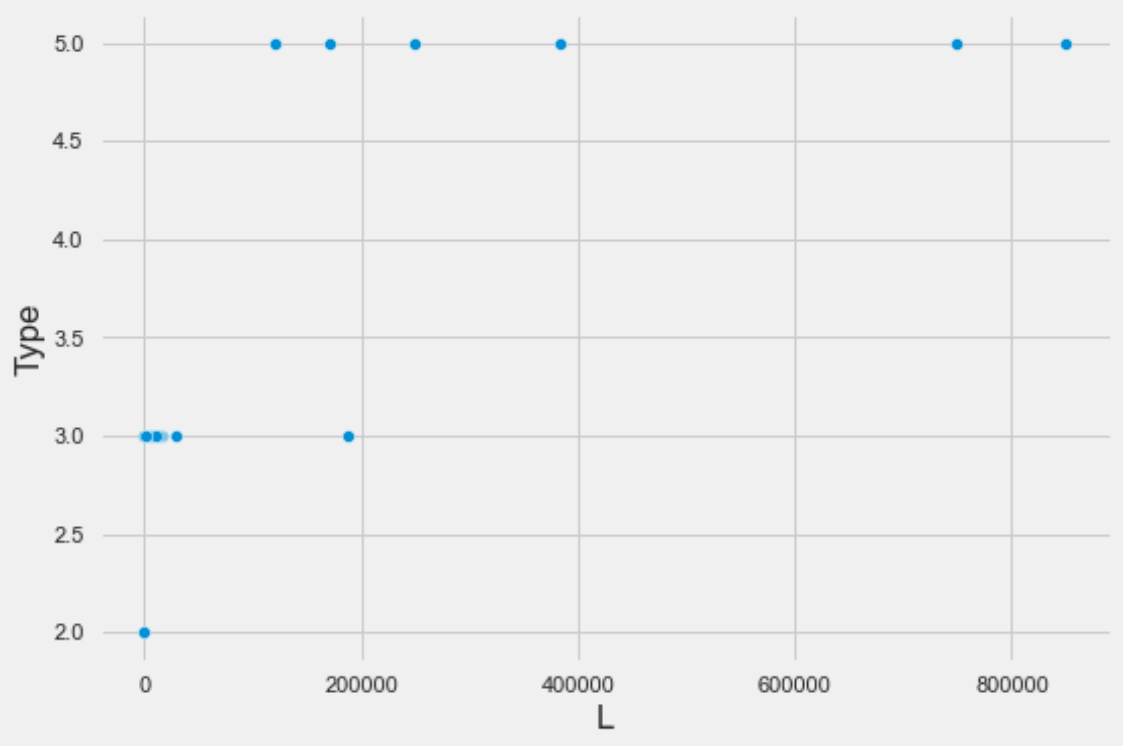
[168]:
```
sns.scatterplot(x = 'L',y = 'R',data = bluew_df)
plt.show()
```



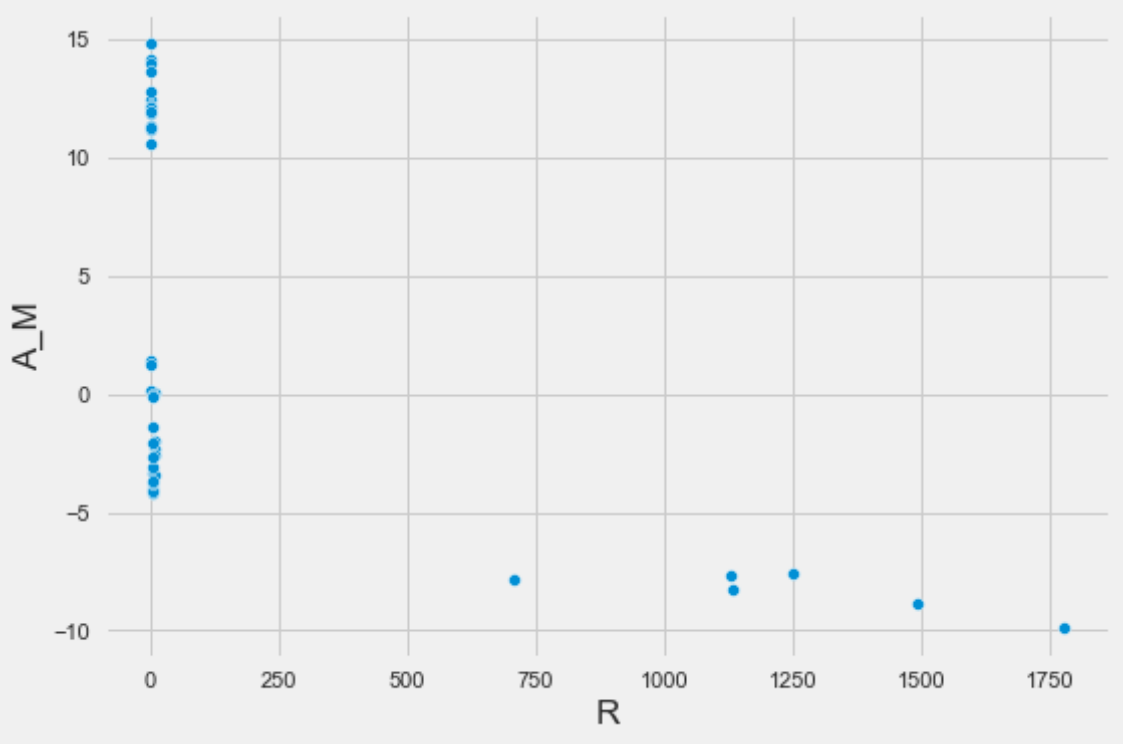
[169]:
```
sns.scatterplot(x = 'L',y = 'A_M',data = bluew_df)
plt.show()
```



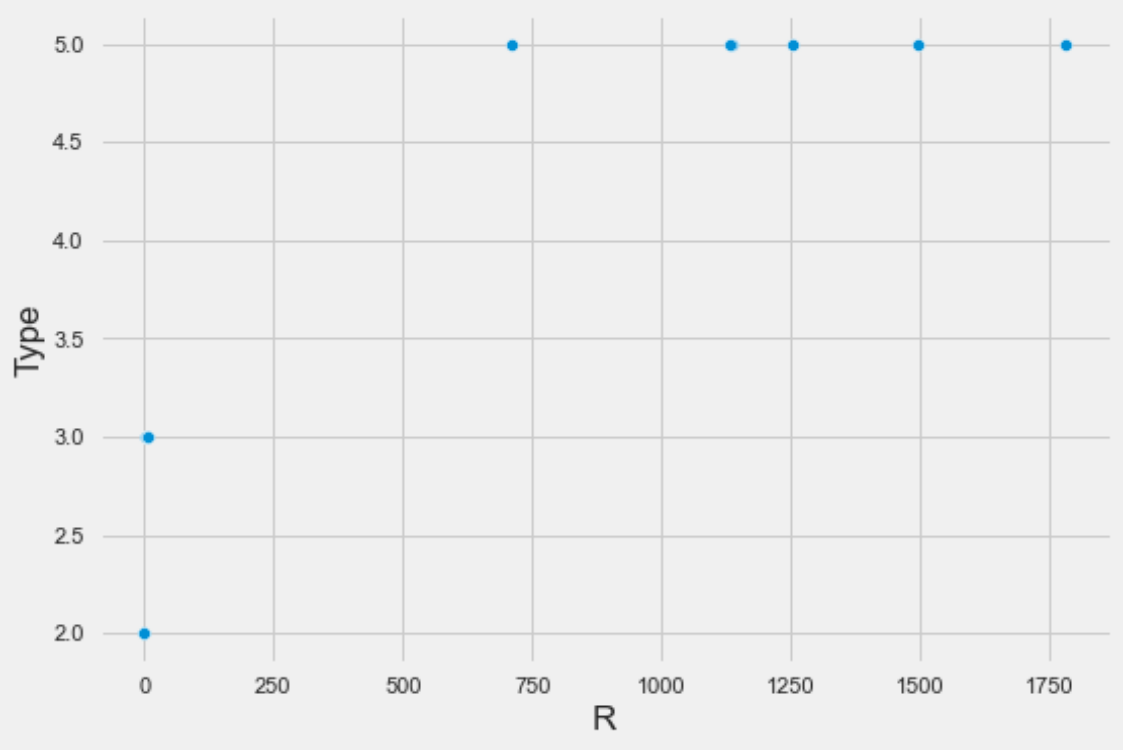
[170]:
```
sns.scatterplot(x = 'L',y = 'Type',data = bluew_df)
plt.show()
```



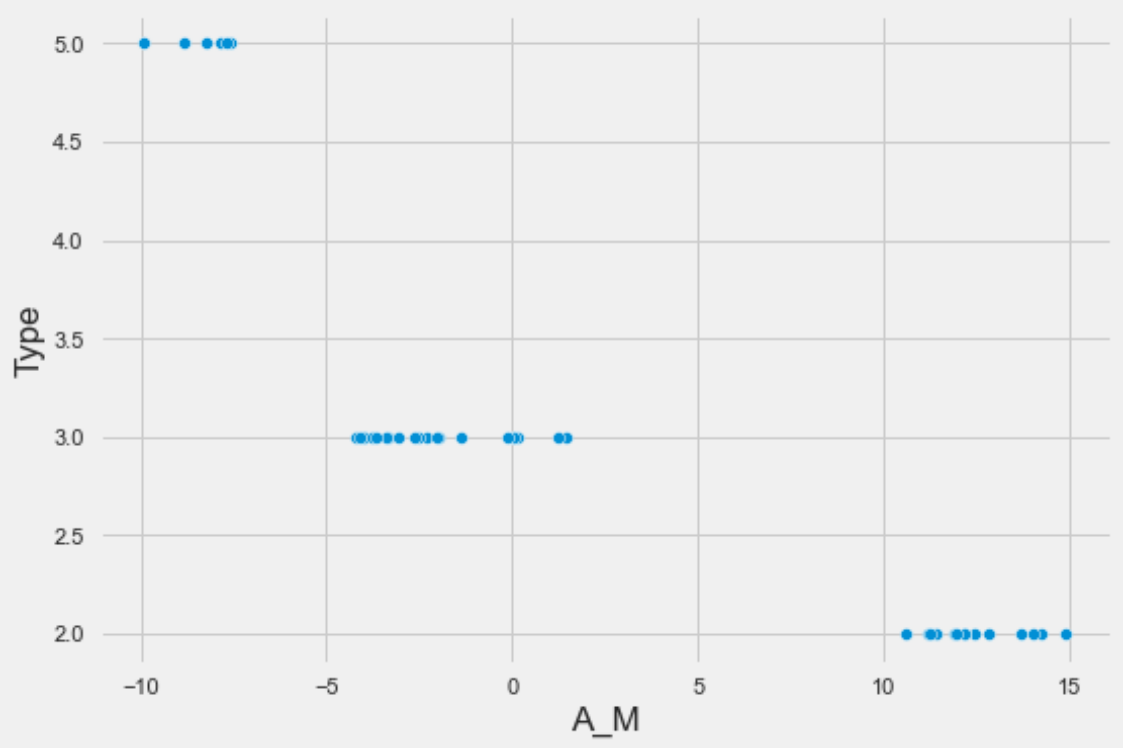
[171]:
```
sns.scatterplot(x = 'R',y = 'A_M',data = bluew_df)
plt.show()
```



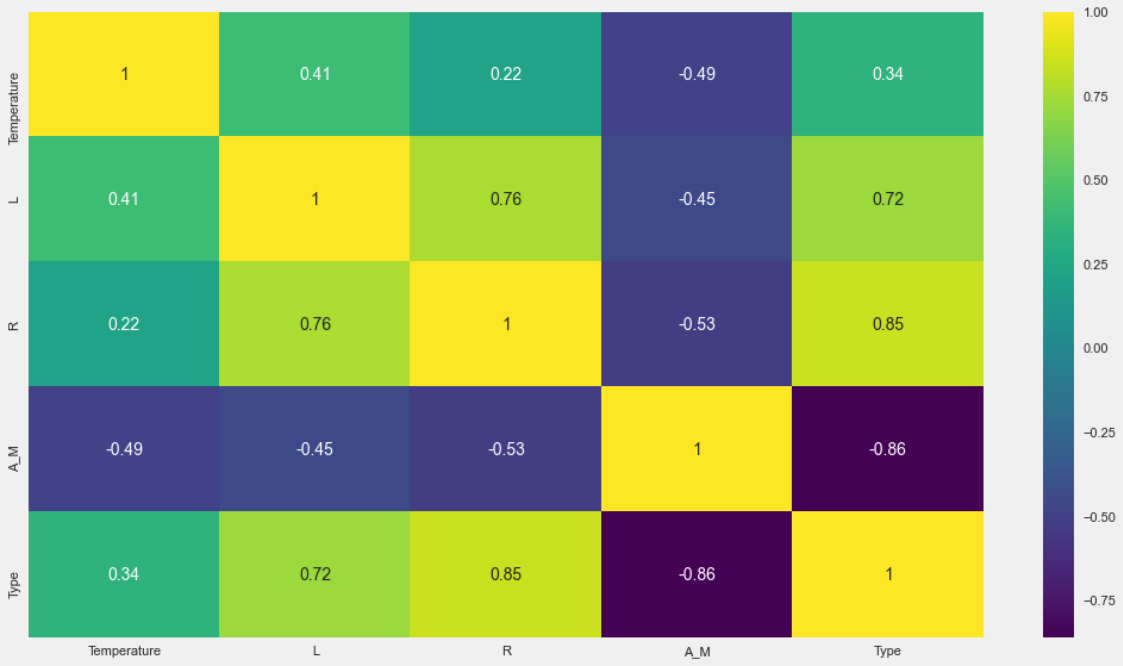
[172]:
```
sns.scatterplot(x = 'R',y = 'Type',data =bluew_df)
plt.show()
```



[173]:
```
sns.scatterplot(x = 'A_M',y = 'Type',data = bluew_df)
plt.show()
```



[174]:
```
plt.figure(figsize=(16,9))
x = bluew_df.drop(['Color','Spectral_Class'],axis = 1)
ax = sns.heatmap(x.corr(),annot = True,cmap = 'viridis')
plt.show()
```



[175]:
```
t = bluew_df[bluew_df['Spectral_Class'] == 'A']
t
```

[175]:

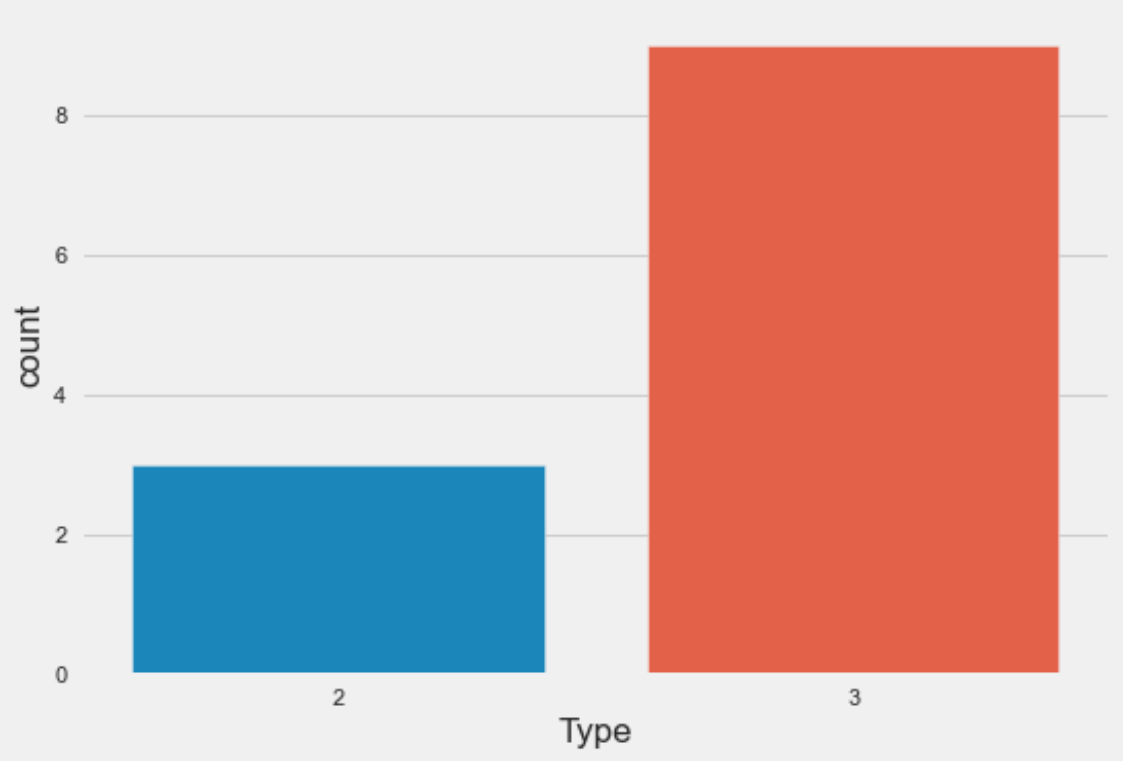
	Temperature	L	R	A_M	Color	Spectral_Class	Type
26	8570	0.00081	0.00970	14.200	Blue-white	A	2
85	9675	0.00045	0.01090	13.980	Blue-white	A	2
94	9030	45.00000	2.63000	1.450	Blue-white	A	3
95	11250	672.00000	6.98000	-2.300	Blue-white	A	3
98	12098	689.00000	7.01000	0.020	Blue-white	A	3
145	8924	0.00028	0.00879	14.870	Blue-white	A	2
151	8945	38.00000	2.48700	0.120	Blue-white	A	3
152	14060	1092.00000	5.74500	-2.040	Blue-white	A	3
158	13023	998.00000	6.21000	-1.380	Blue-white	A	3
211	10012	552.00000	5.85600	0.013	Blue-white	A	3
212	13089	788.00000	5.99200	-0.120	Blue-white	A	3
216	9320	29.00000	1.91000	1.236	Blue-white	A	3

[176]:
```
bluew_df[bluew_df['Spectral_Class'] == 'A']['Type'].value_counts()
```

[176]:
```
3    9
2    3
Name: Type, dtype: int64
```

[177]:
```
sns.countplot(x = 'Type',data = t)
```

[177]:
```
<AxesSubplot:xlabel='Type', ylabel='count'>
```



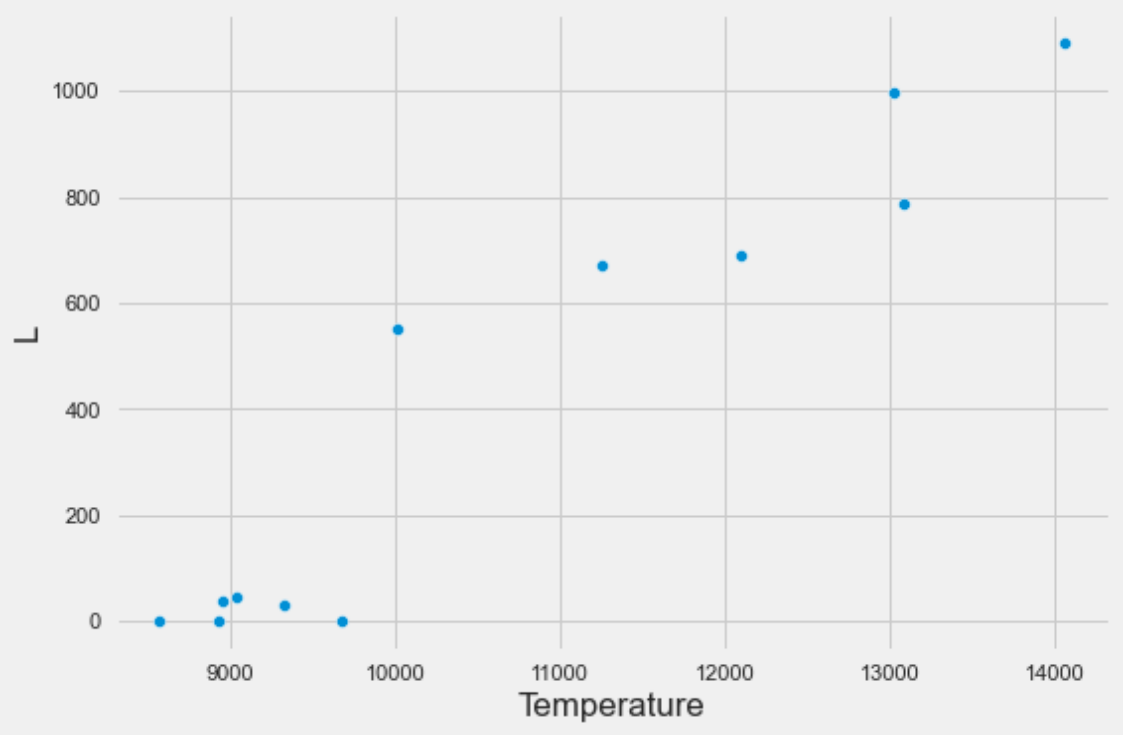
[178]:
```
fig = px.histogram(t, 'Temperature',
                   color = 'Type')
fig.show()
```

[179]:
```
fig = px.histogram(t, 'L',
                   color = 'Type')
fig.show()
```

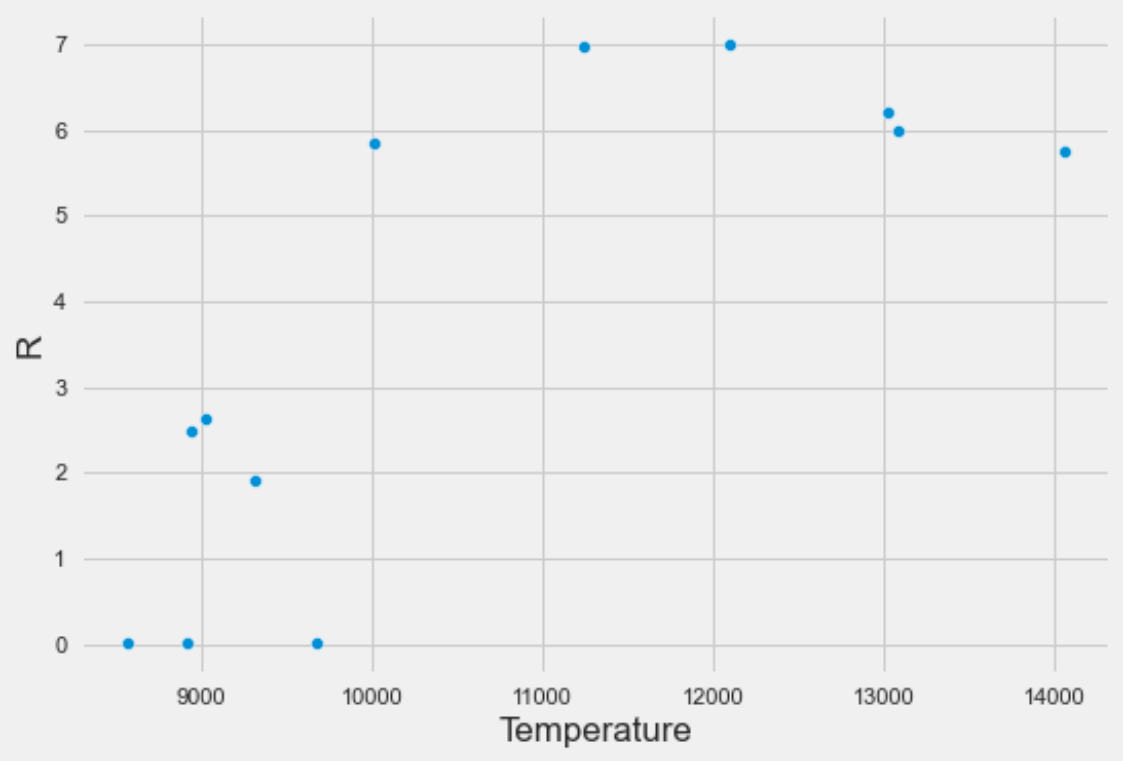
[180]:
```
fig = px.histogram(t, 'R',
                   color = 'Type')
fig.show()
```

[181]:
```
fig = px.histogram(t, 'A_M',
                   color = 'Type')
fig.show()
```

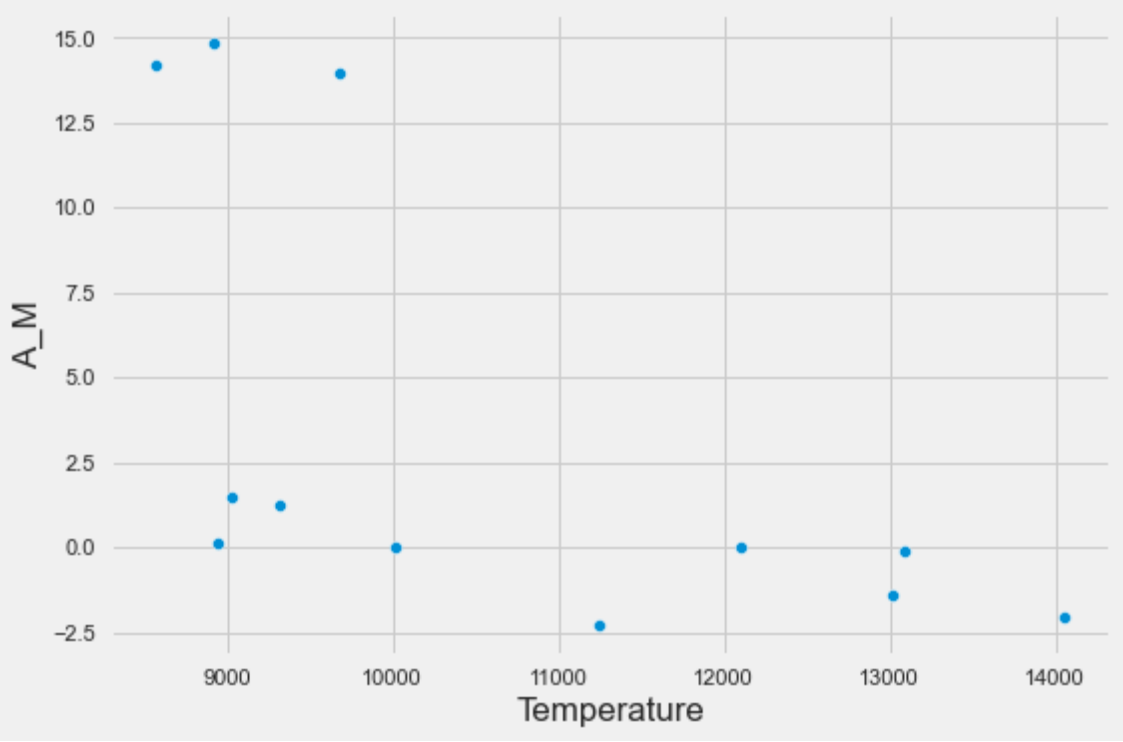
[182]:
```
sns.scatterplot(x = 'Temperature',y = 'L',data = t)
plt.show()
```



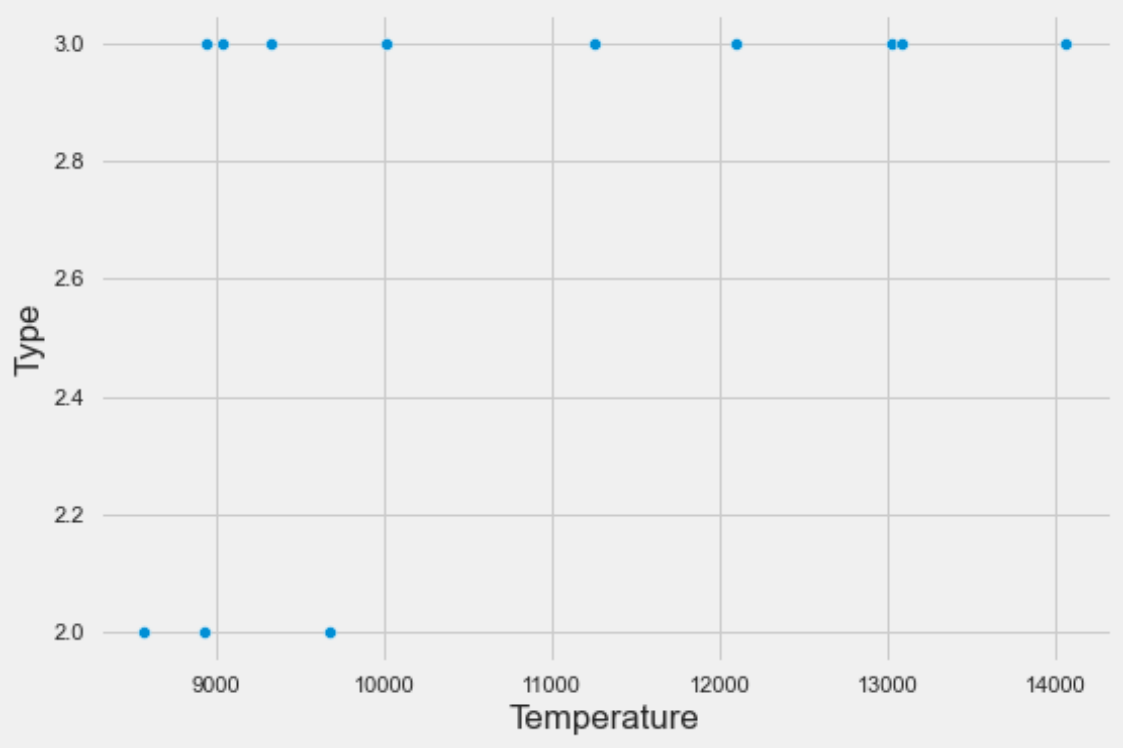
[183]:
```
sns.scatterplot(x = 'Temperature',y = 'R',data = t)
plt.show()
```



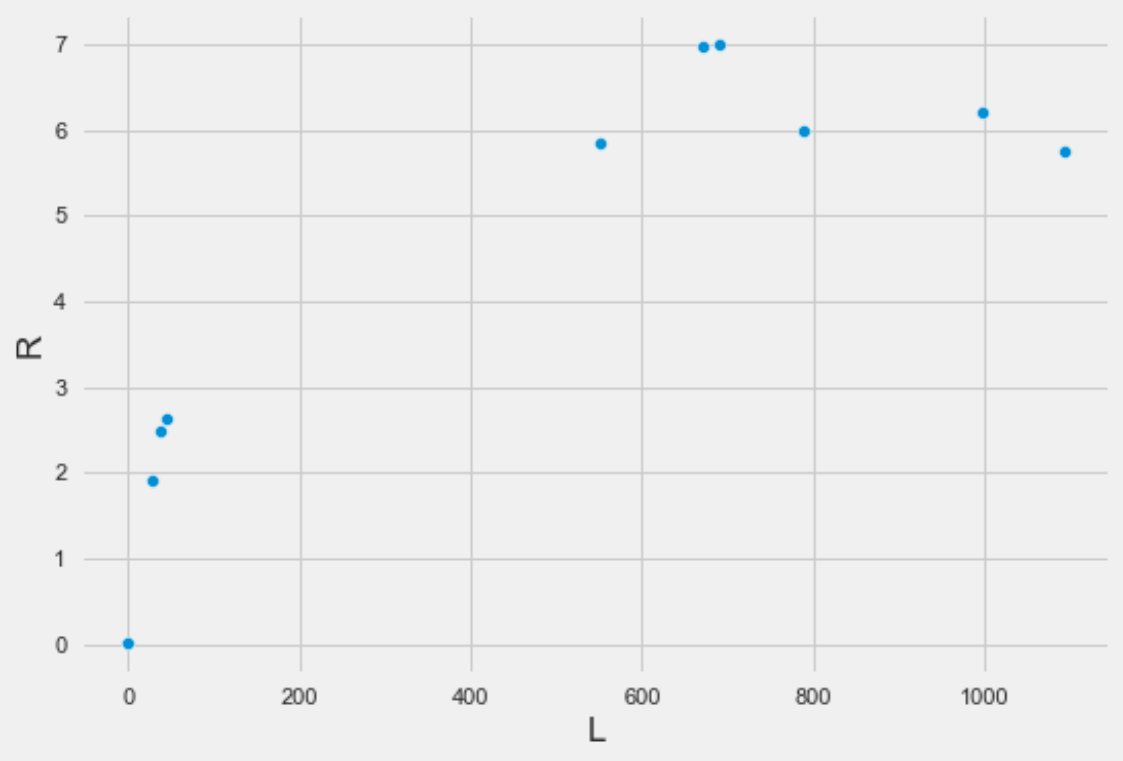
[184]:
```
sns.scatterplot(x = 'Temperature',y = 'A_M',data = t)
plt.show()
```



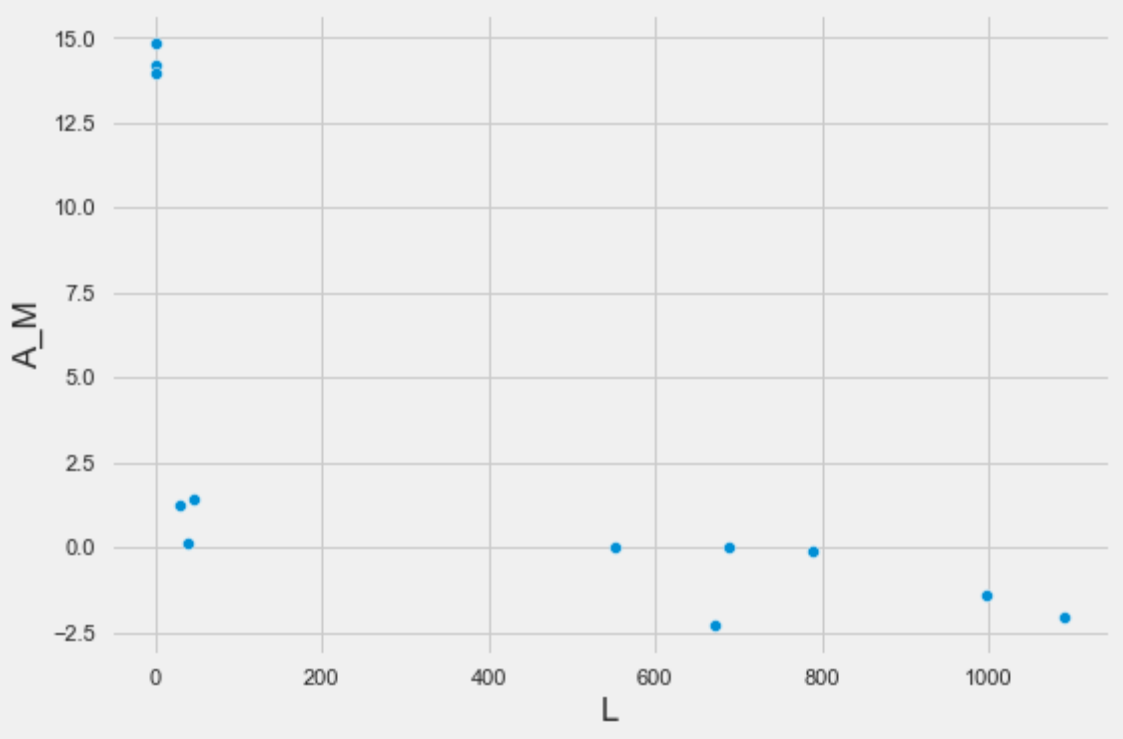
[185]:
```
sns.scatterplot(x = 'Temperature',y = 'Type',data = t)
plt.show()
```



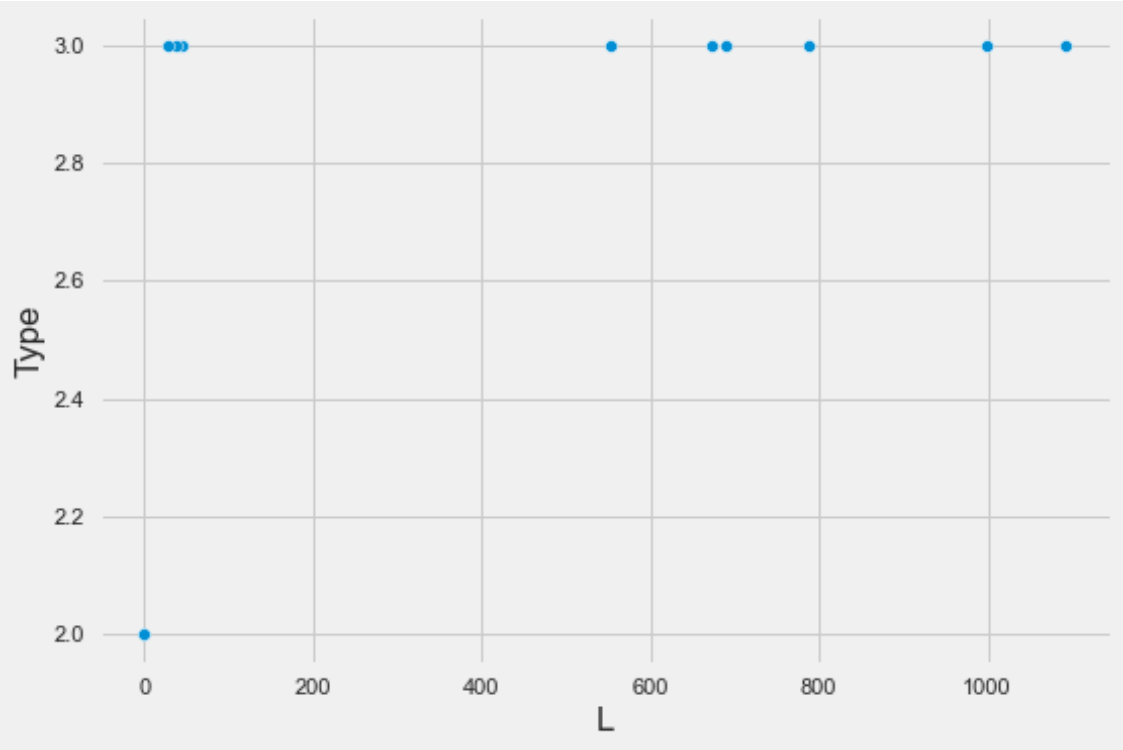
[186]:
```
sns.scatterplot(x = 'L',y = 'R',data = t)
plt.show()
```



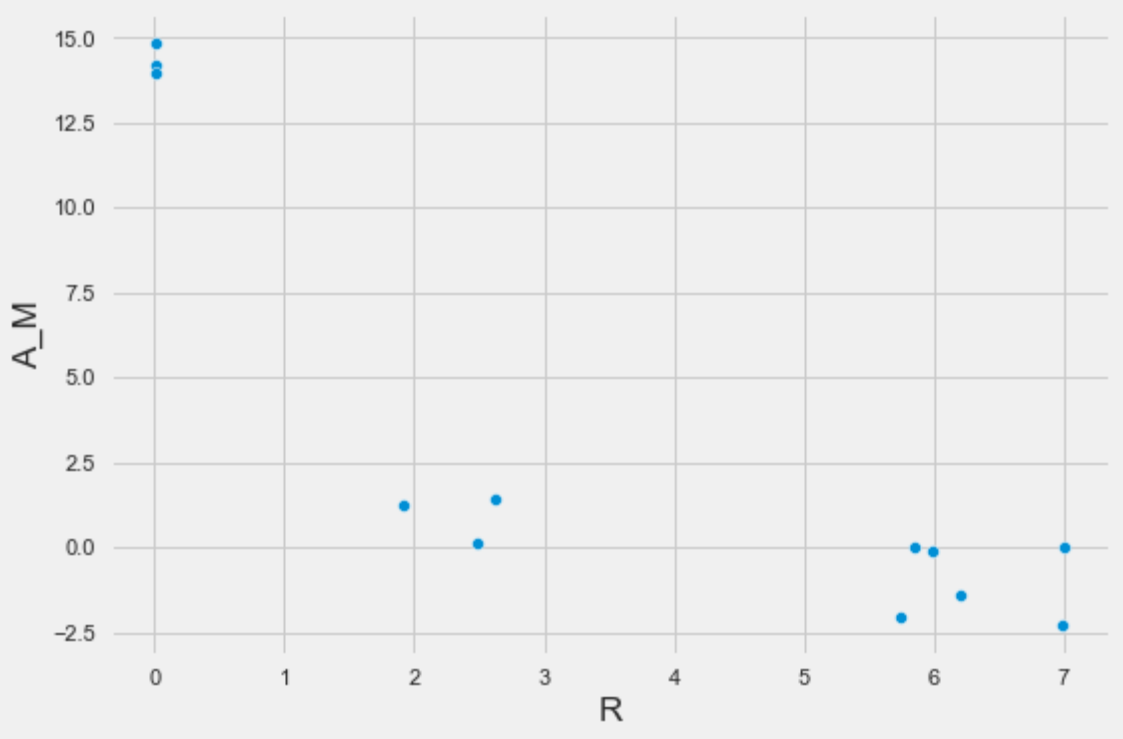
[187]:
```
sns.scatterplot(x = 'L',y = 'A_M',data = t)
plt.show()
```



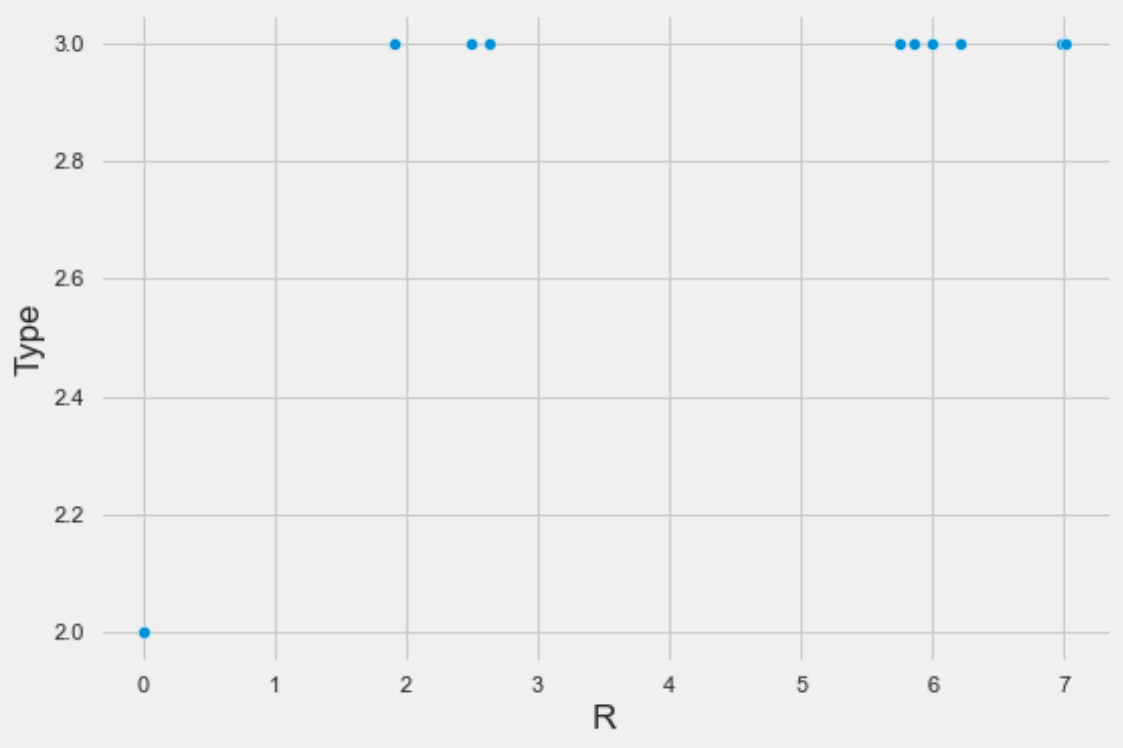
[188]:
```
sns.scatterplot(x = 'L',y = 'Type',data = t)
plt.show()
```



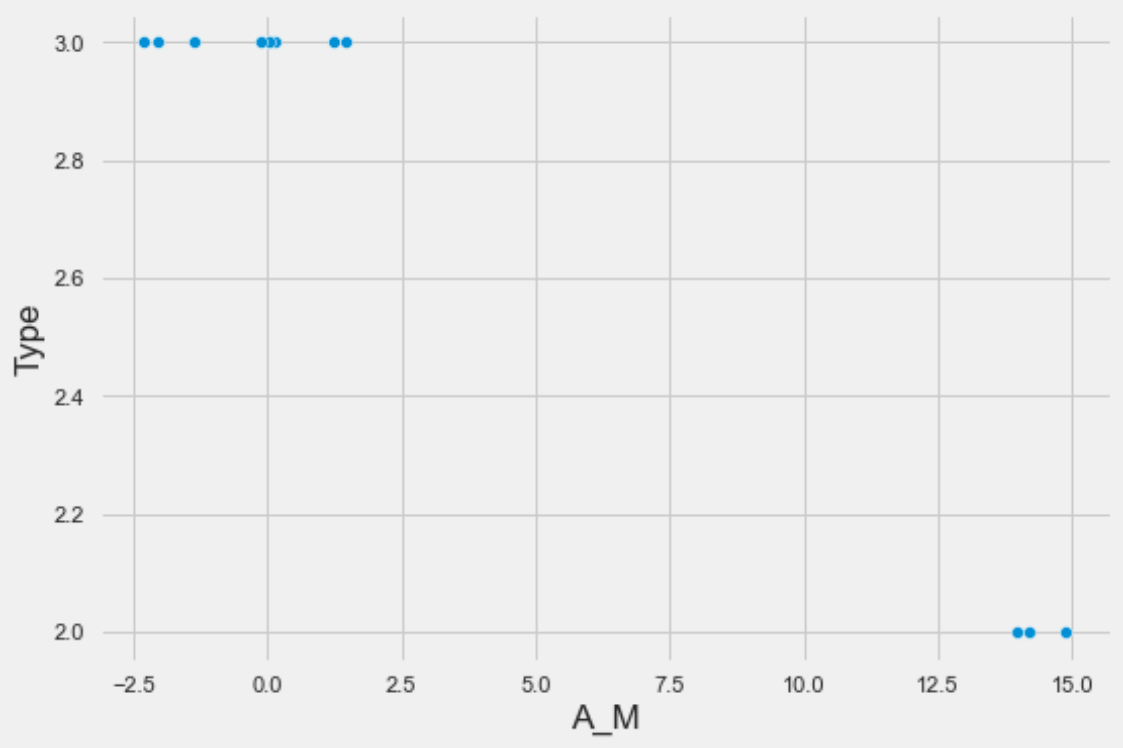
[189]:
```
sns.scatterplot(x = 'R',y = 'A_M',data = t)
plt.show()
```



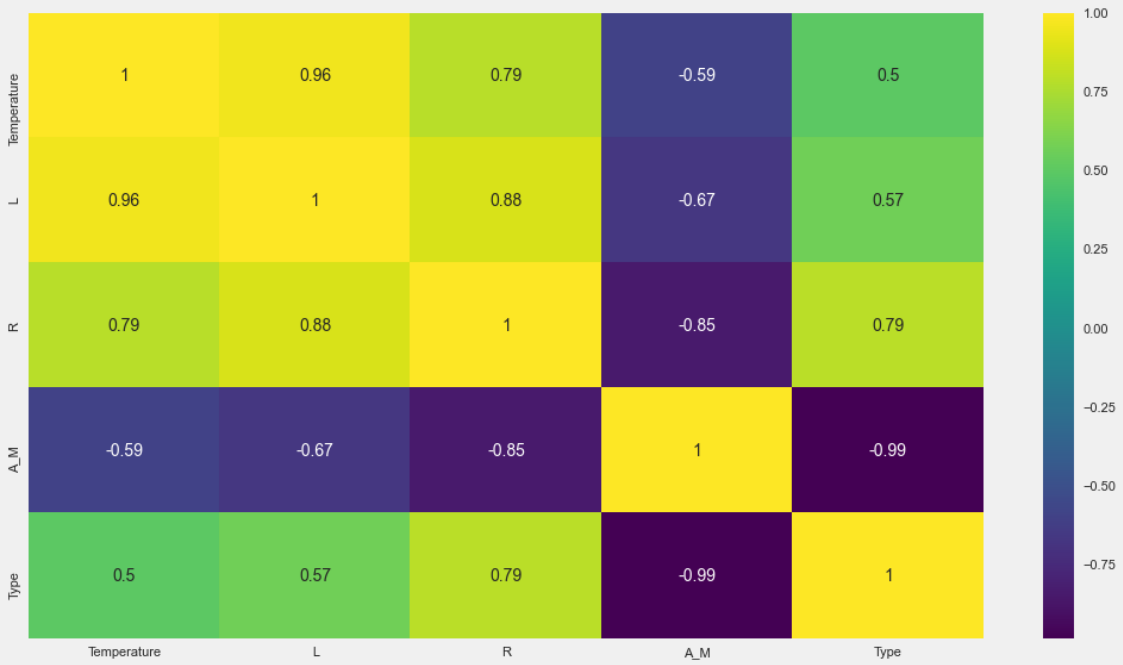
[190]:
```
sns.scatterplot(x = 'R',y = 'Type',data = t)
plt.show()
```



[191]:
```
sns.scatterplot(x = 'A_M',y = 'Type',data = t)
plt.show()
```



[192]:
```
plt.figure(figsize=(16,9))
x = t.drop(['Color','Spectral_Class'],axis = 1)
ax = sns.heatmap(x.corr(),annot = True,cmap = 'viridis')
plt.show()
```



[193]:
```
q = bluew_df[bluew_df['Spectral_Class'] == 'B']
q
```

[193]:

	Temperature	L	R	A_M	Color	Spectral_Class	\
20	25000	0.05600	0.00840	10.58	Blue-white	B	
24	16500	0.01300	0.01400	11.89	Blue-white	B	
31	30000	28840.00000	6.30000	-4.20	Blue-white	B	
32	15276	1136.00000	7.20000	-1.97	Blue-white	B	
83	17200	0.00098	0.01500	12.45	Blue-white	B	
84	14100	0.00067	0.00890	12.17	Blue-white	B	
86	12010	0.00078	0.00920	12.13	Blue-white	B	
87	10980	0.00074	0.00870	11.19	Blue-white	B	
140	13420	0.00059	0.00981	13.67	Blue-white	B	
143	14520	0.00082	0.00972	11.92	Blue-white	B	
144	11900	0.00067	0.00898	11.38	Blue-white	B	
146	12912	0.00071	0.00945	12.83	Blue-white	B	
149	12984	0.00088	0.00996	11.23	Blue-white	B	
150	29560	188000.00000	6.02000	-4.01	Blue-white	B	
153	16390	1278.00000	5.68000	-3.32	Blue-white	B	
154	25070	14500.00000	5.92000	-3.98	Blue-white	B	
155	28700	16790.00000	6.40000	-4.09	Blue-white	B	
156	26140	14520.00000	5.49000	-3.80	Blue-white	B	
157	20120	4720.00000	6.78000	-3.40	Blue-white	B	
177	11000	170000.00000	1779.00000	-9.90	Blue-white	B	
178	12100	120000.00000	708.90000	-7.84	Blue-white	B	

```
179        24490  248490.00000  1134.50000  -8.24  Blue-white              B
210        22350   12450.00000     6.36000  -3.67  Blue-white              B
213        22012    6748.00000     6.64000  -2.55  Blue-white              B
217        19400   10920.00000     6.03000  -3.08  Blue-white              B
218        17140     883.00000     5.65300  -2.64  Blue-white              B
230        24145  382993.00000  1494.00000  -8.84  Blue-white              B
233        27739  849420.00000  1252.00000  -7.59  Blue-white              B
234        21904  748490.00000  1130.00000  -7.67  Blue-white              B
```

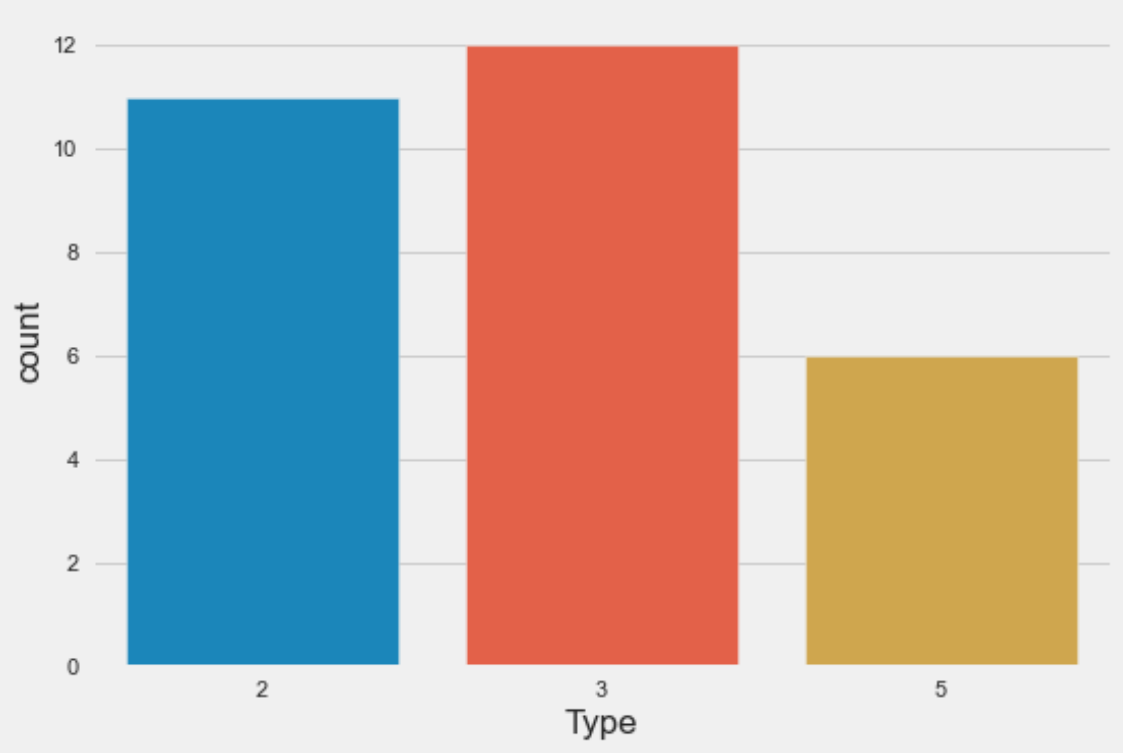
```
     Type
20      2
24      2
31      3
32      3
83      2
84      2
86      2
87      2
140     2
143     2
144     2
146     2
149     2
150     3
153     3
154     3
155     3
156     3
157     3
177     5
178     5
179     5
210     3
213     3
217     3
218     3
230     5
233     5
234     5
```

[194]:
```
bluew_df[bluew_df['Spectral_Class'] == 'B']['Type'].value_counts()
```

[194]:
```
3    12
2    11
5     6
Name: Type, dtype: int64
```

[195]:
```
sns.countplot(x = 'Type',data = q)
```

[195]:
```
<AxesSubplot:xlabel='Type', ylabel='count'>
```



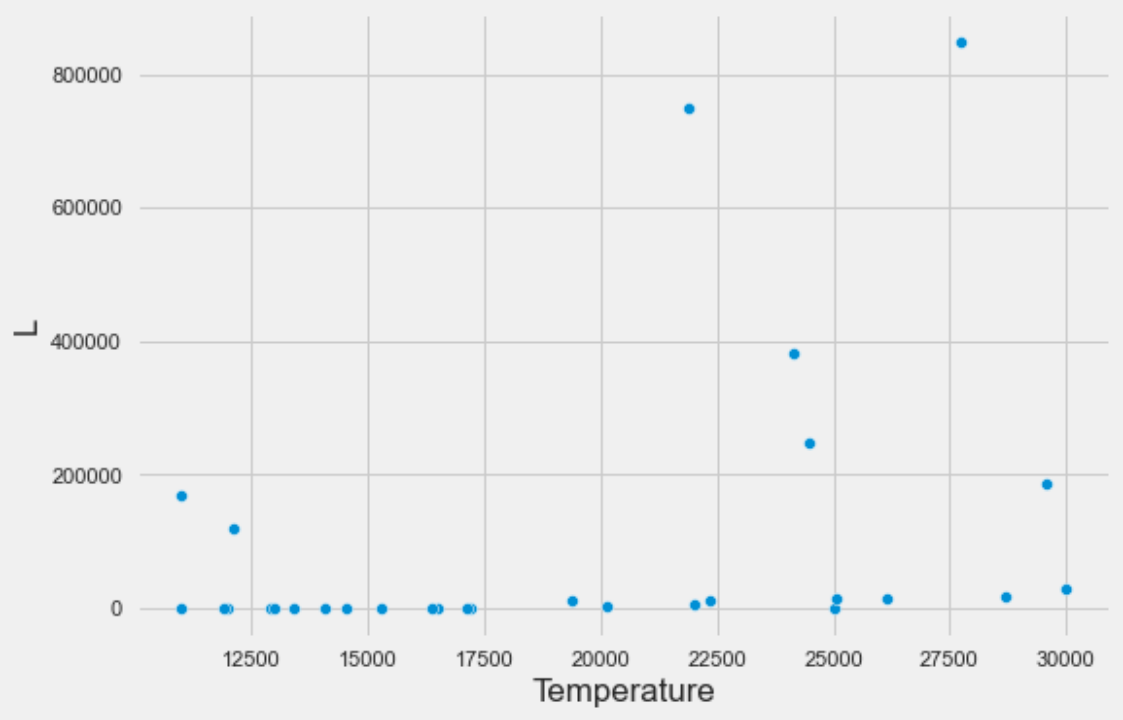
[196]:
```
fig = px.histogram(q, 'Temperature',
                   color = 'Type')
fig.show()
```

[197]:
```
fig = px.histogram(q, 'L',
                   color = 'Type')
fig.show()
```

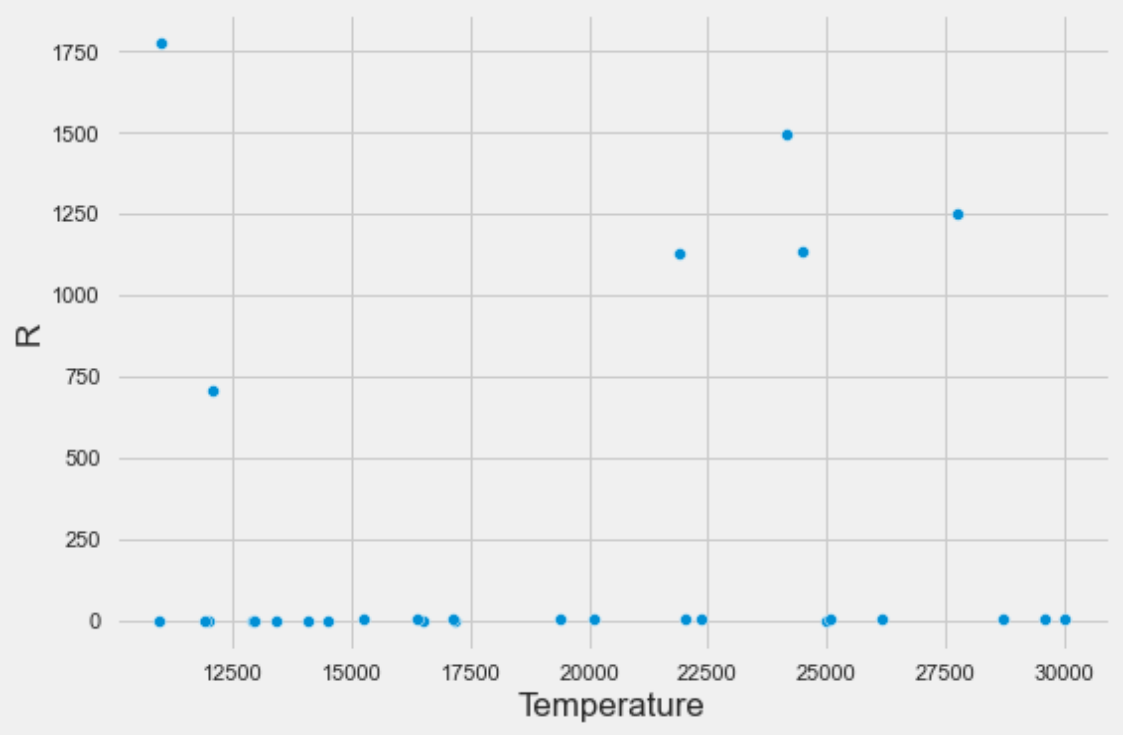
[198]:
```
fig = px.histogram(q, 'R',
                   color = 'Type')
fig.show()
```

[199]:
```
fig = px.histogram(q, 'A_M',
                   color = 'Type')
fig.show()
```

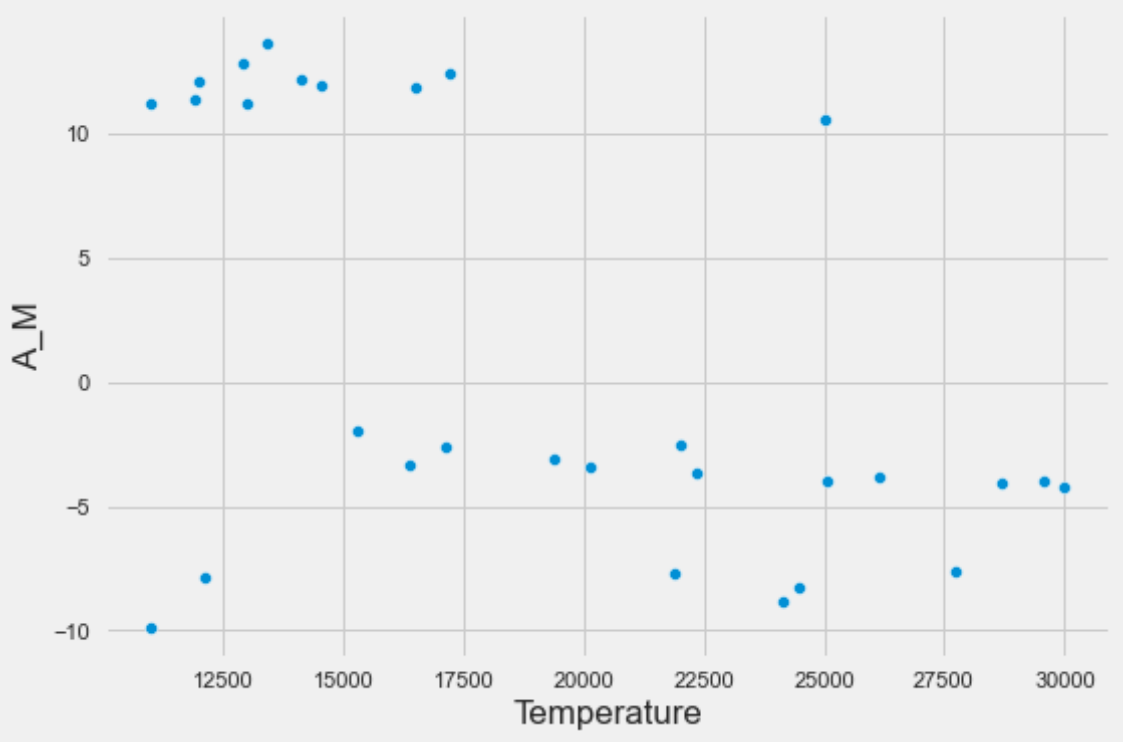
[200]:
```
sns.scatterplot(x = 'Temperature',y = 'L',data = q)
plt.show()
```



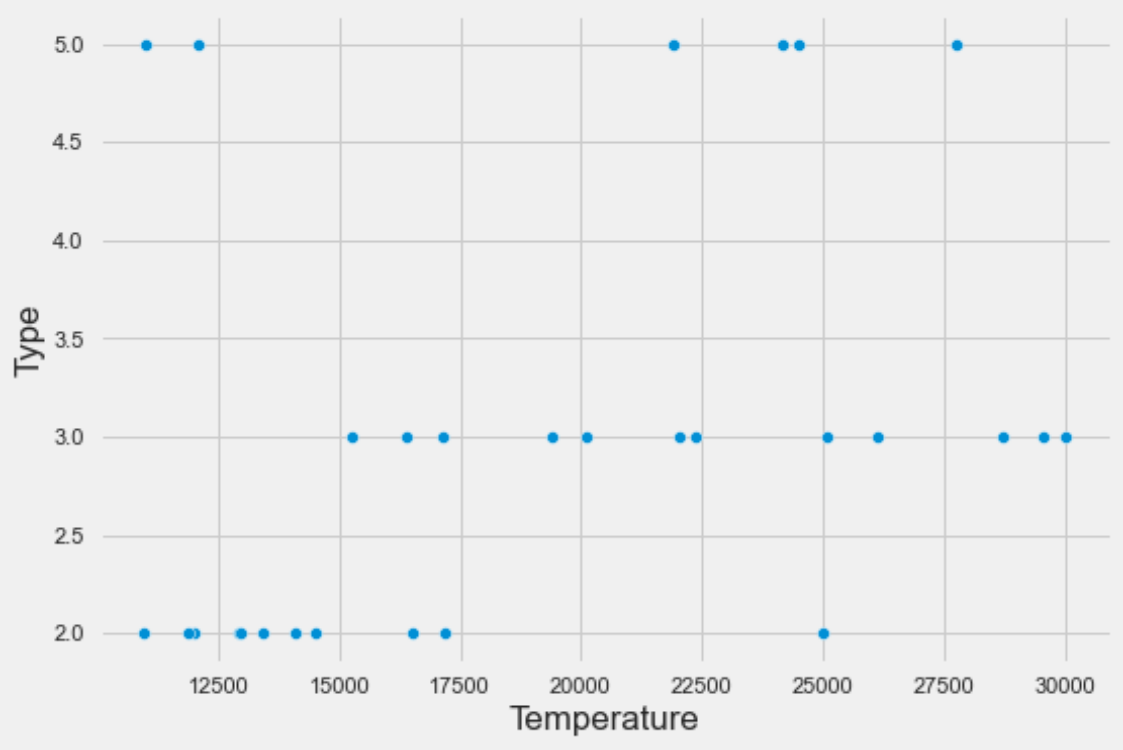
[201]:
```
sns.scatterplot(x = 'Temperature',y = 'R',data = q)
plt.show()
```



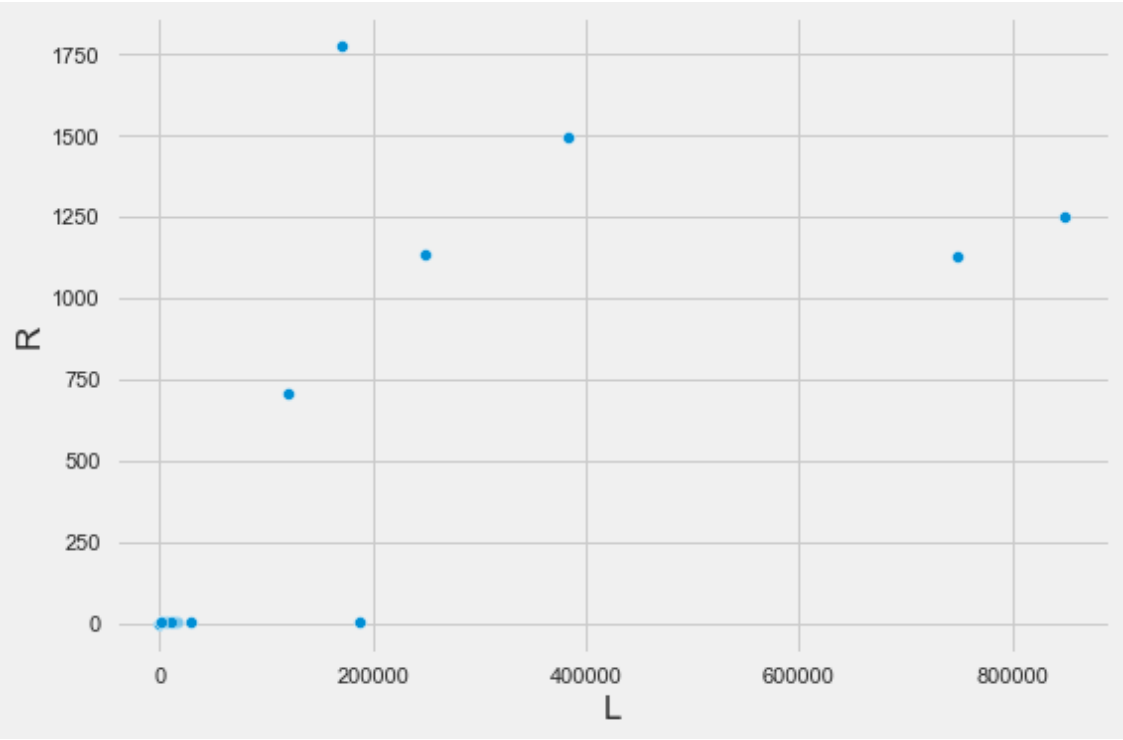
[202]:
```
sns.scatterplot(x = 'Temperature',y = 'A_M',data = q)
plt.show()
```



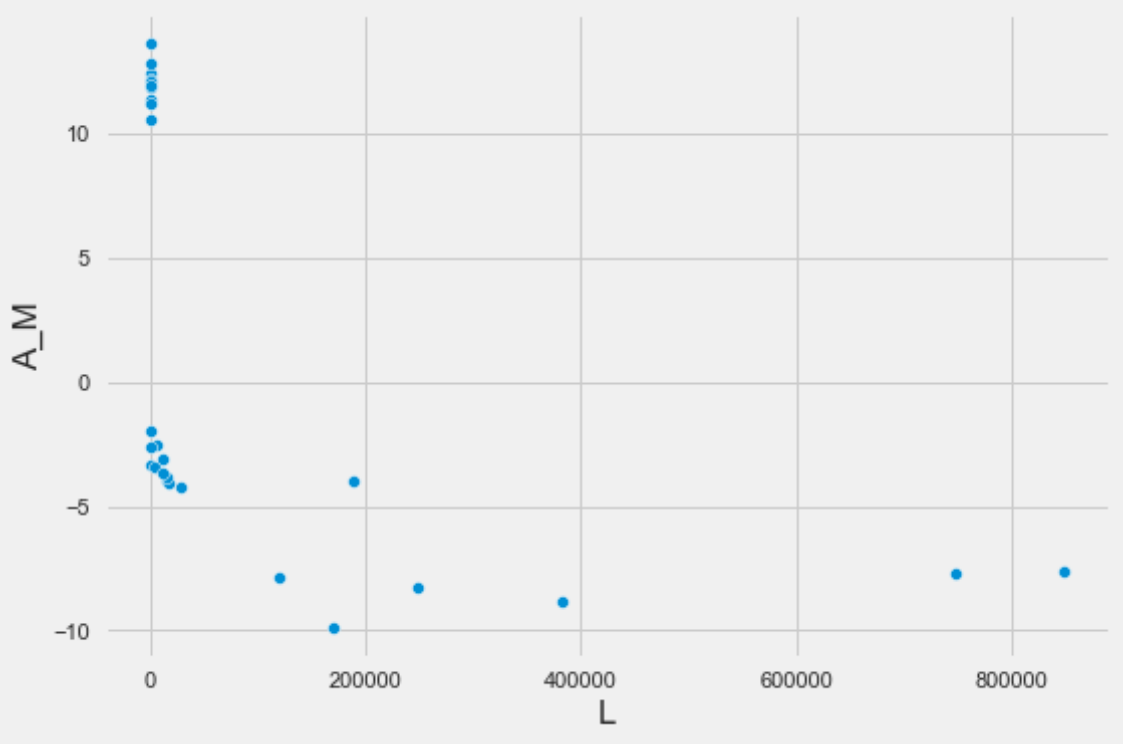
[203]:
```
sns.scatterplot(x = 'Temperature',y = 'Type',data = q)
plt.show()
```



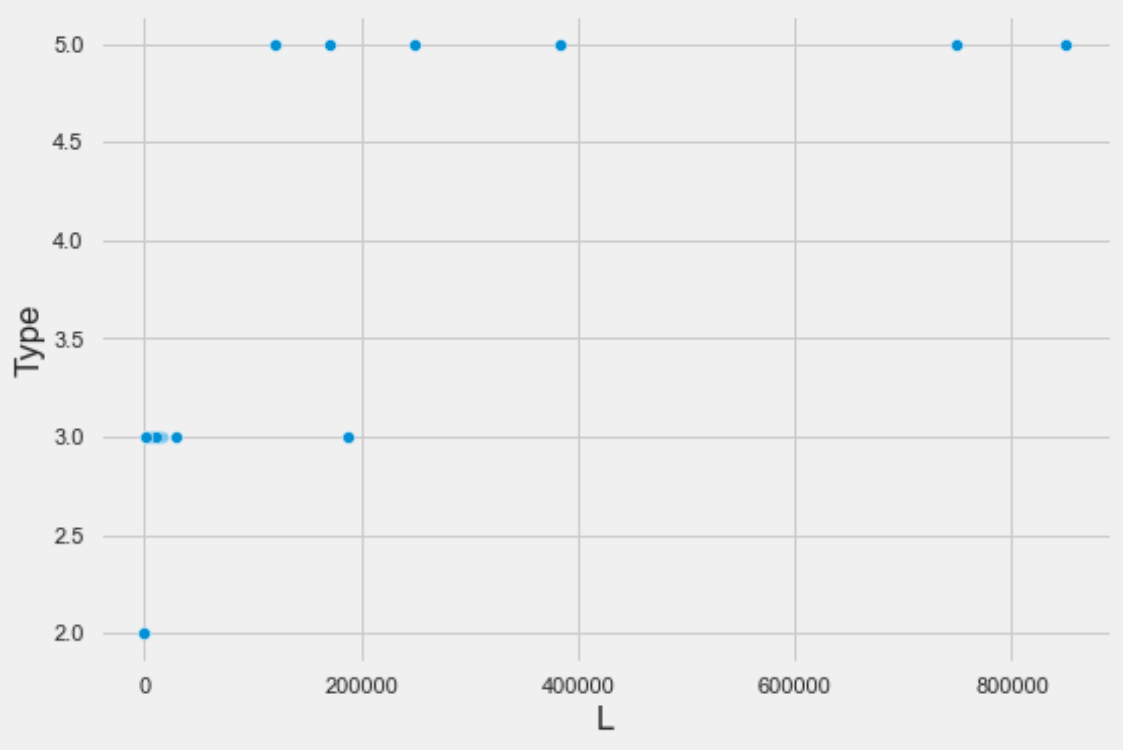
[204]:
```
sns.scatterplot(x = 'L',y = 'R',data = q)
plt.show()
```



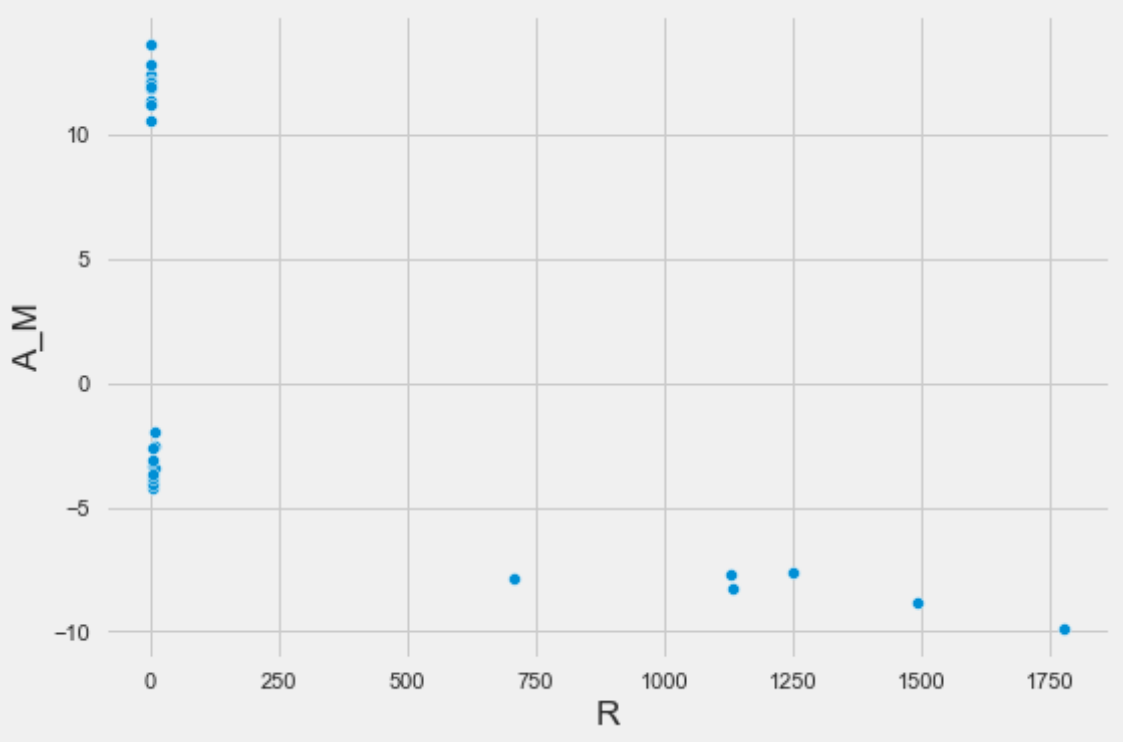
[205]:
```
sns.scatterplot(x = 'L',y = 'A_M',data = q)
plt.show()
```



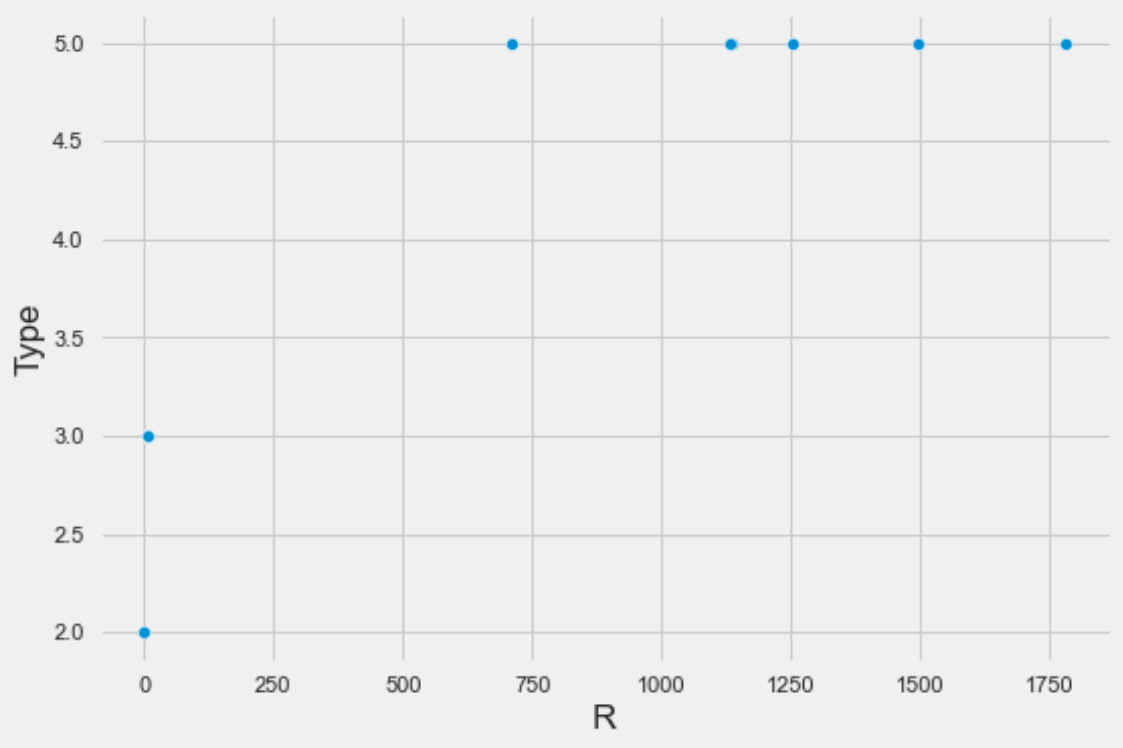
[206]:
```
sns.scatterplot(x = 'L',y = 'Type',data = q)
plt.show()
```



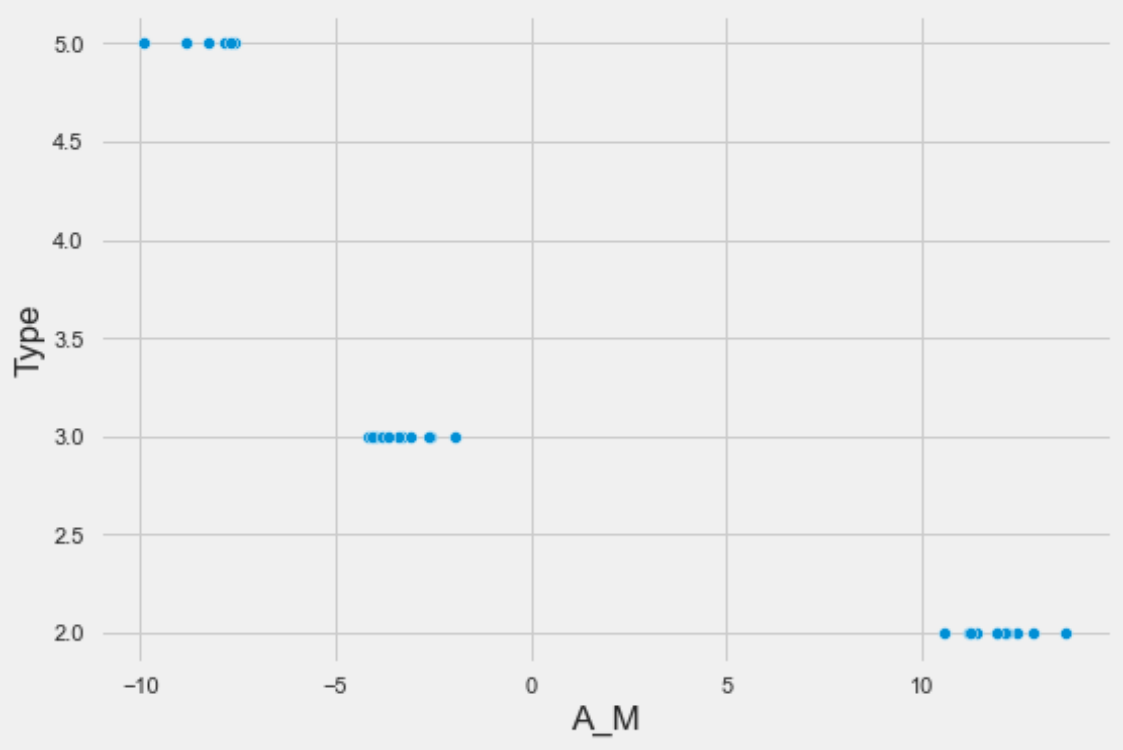
[207]:
```
sns.scatterplot(x = 'R',y = 'A_M',data = q)
plt.show()
```



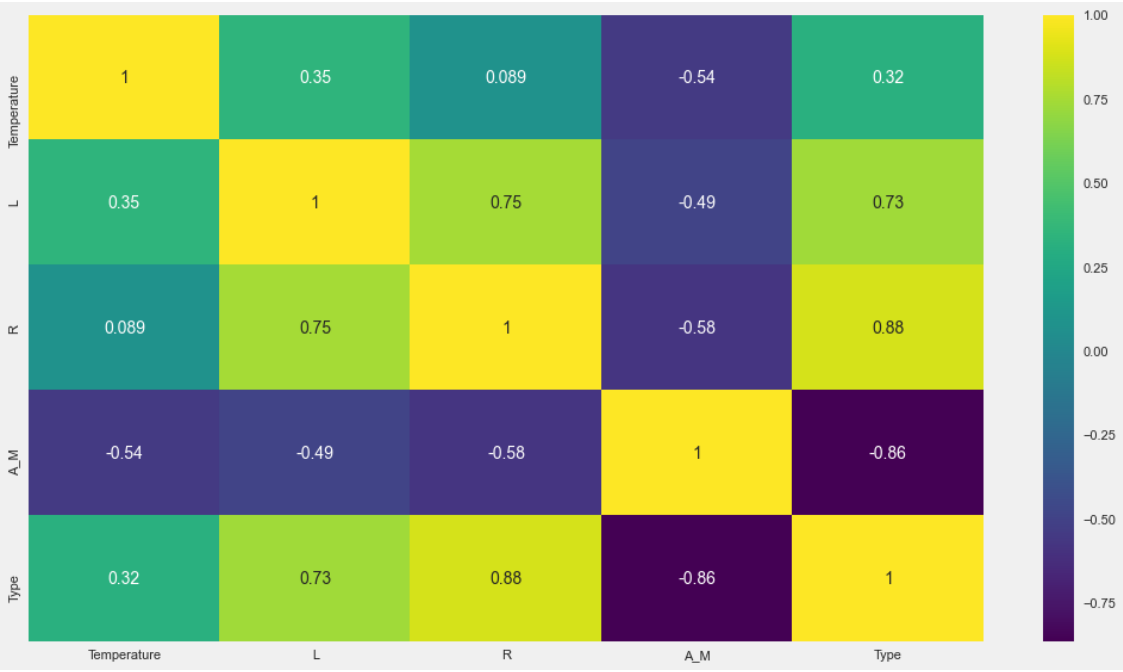
[208]:
```
sns.scatterplot(x = 'R',y = 'Type',data = q)
plt.show()
```



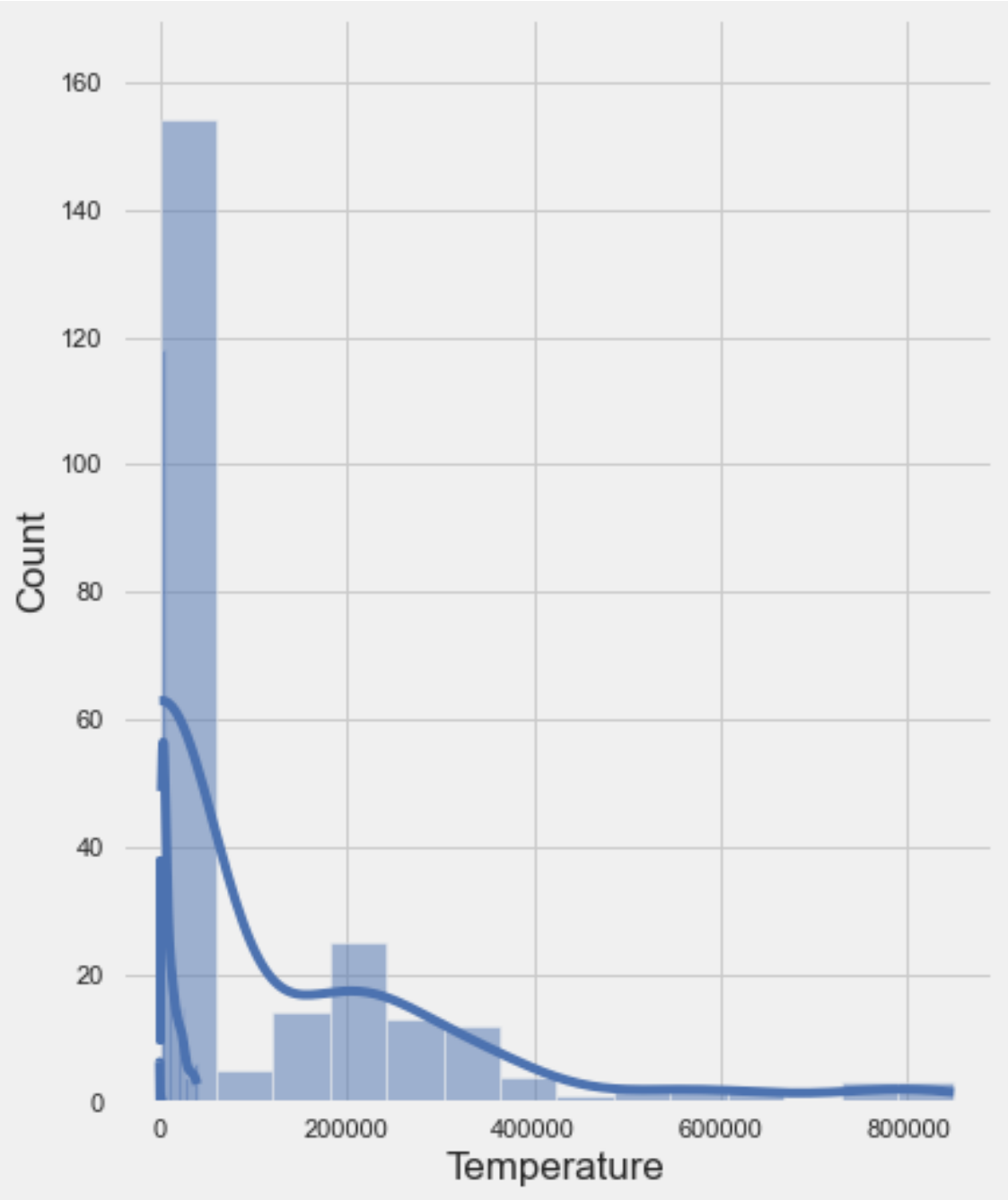
[209]:
```
sns.scatterplot(x = 'A_M',y = 'Type',data = q)
plt.show()
```



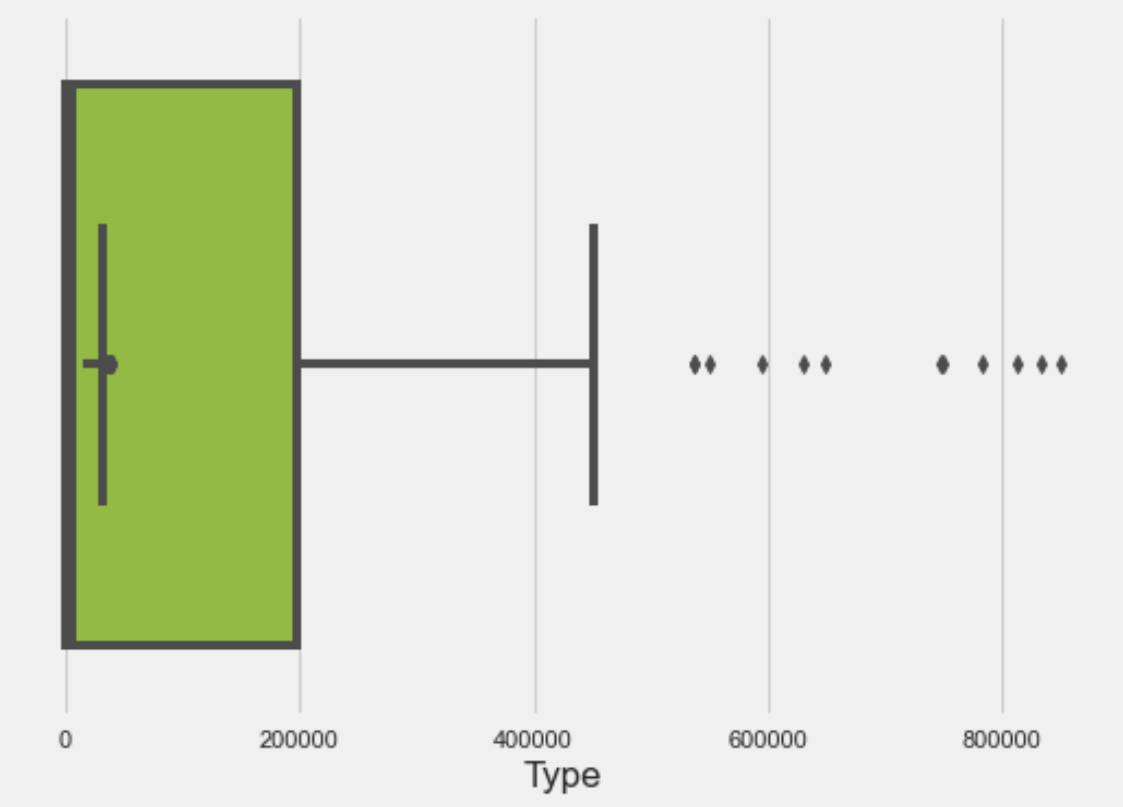
[210]:
```
plt.figure(figsize=(16,9))
x = q.drop(['Color','Spectral_Class'],axis = 1)
ax = sns.heatmap(q.corr(),annot = True,cmap = 'viridis')
plt.show()
```



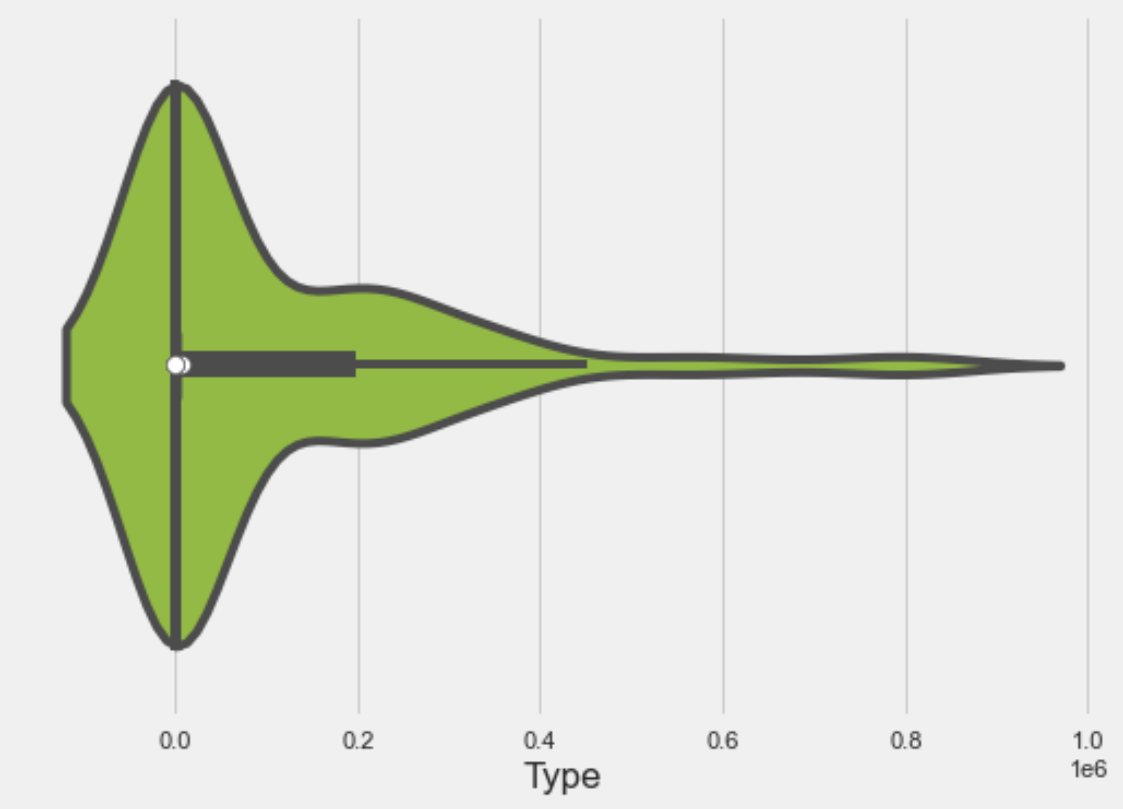
[211]:
```
plt.figure(figsize=(6,8))
x = df.drop(['Color','Spectral_Class'],axis = 1)
for i in x.columns:
    sns.histplot(x[i],kde = True)
    plt.show()
```



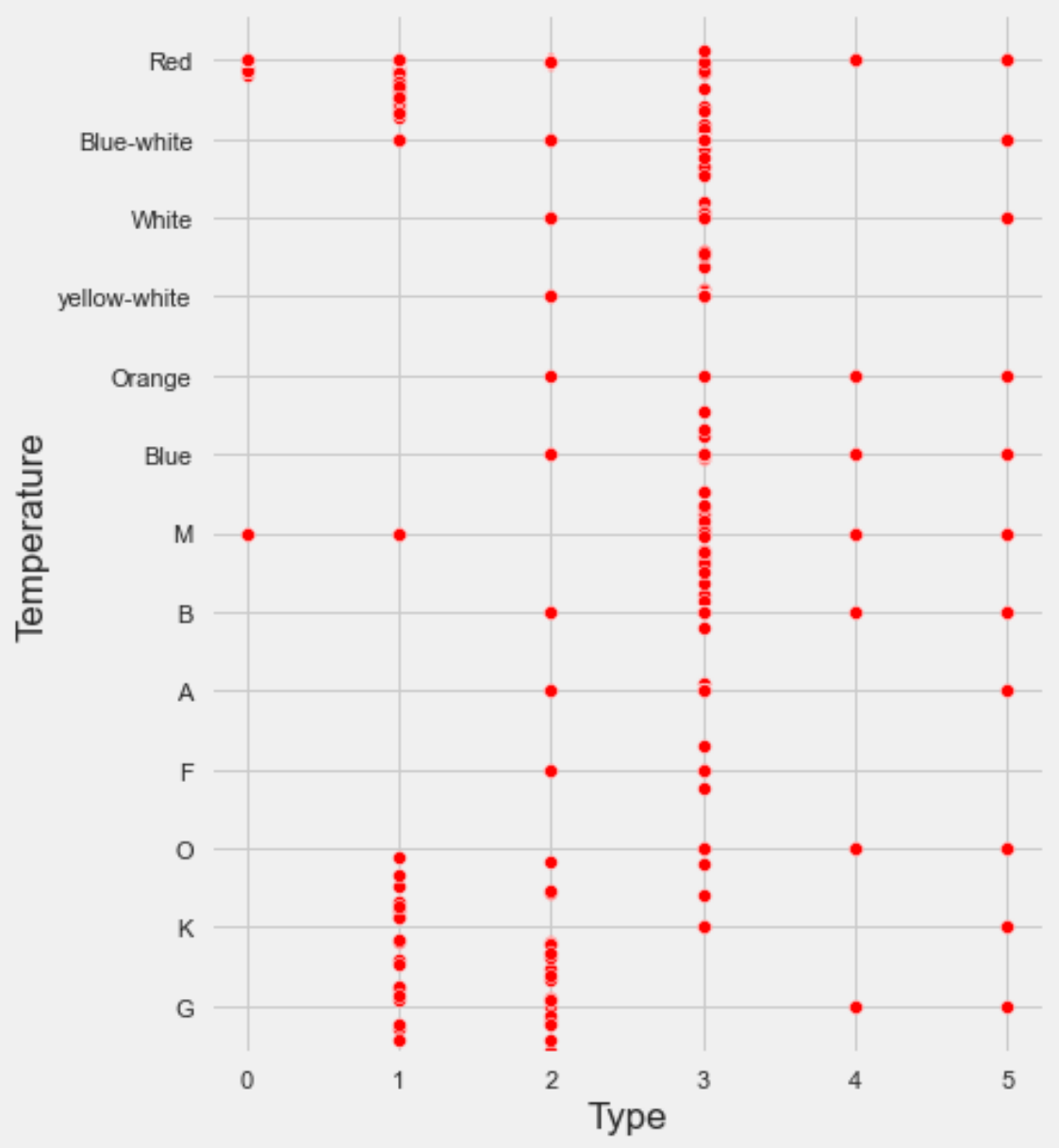
[212]:
```
x = df.drop(['Color','Spectral_Class'],axis = 1)
for i in x.columns:
    sns.boxplot(x = i, data = x,color = 'yellowgreen')
    plt.xlabel(i)
    plt.show()
```



[213]:
```
x = df.drop(['Color','Spectral_Class'],axis = 1)
for i in x.columns:
    sns.violinplot(x = i, data = x,color = 'yellowgreen')
    plt.xlabel(i)
    plt.show()
```

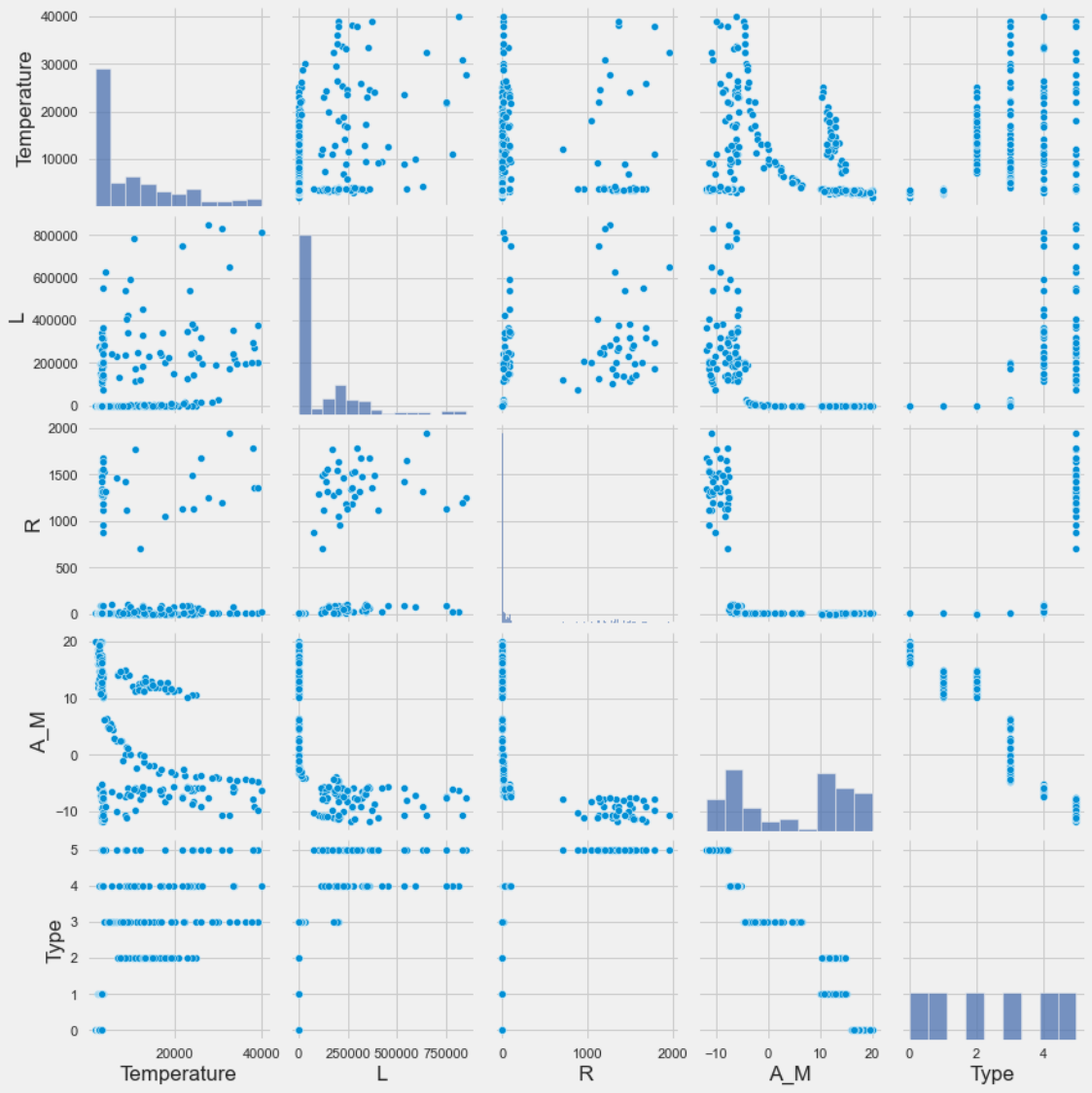


[214]:
```
plt.figure(figsize=(6,8))
for i in df.columns:
    sns.scatterplot(x = 'Type',y = i,data = df,color = 'Red')
    plt.show()
```



[215]:
```
sns.pairplot(df)
```

[215]: <seaborn.axisgrid.PairGrid at 0x1f67a44f970>



7 Count of outliers

[216]:
```
def count_outliers(data,col):
        q1 = data[col].quantile(0.25,interpolation='nearest')
        q2 = data[col].quantile(0.5,interpolation='nearest')
        q3 = data[col].quantile(0.75,interpolation='nearest')
        q4 = data[col].quantile(1,interpolation='nearest')
        IQR = q3 -q1
        global LLP
        global ULP
        LLP = q1 - 1.5*IQR
        ULP = q3 + 1.5*IQR
```

```
        if data[col].min() > LLP and data[col].max() < ULP:
            print("No outliers in",i)
        else:
            print("There are outliers in",i)
            x = data[data[col]<LLP][col].size
            y = data[data[col]>ULP][col].size
            a.append(i)
            print('Count of outliers are:',x+y)
global a
a = []
for i in x.columns:
    count_outliers(df,i)
```

```
There are outliers in Temperature
Count of outliers are: 13
There are outliers in L
Count of outliers are: 12
There are outliers in R
Count of outliers are: 40
No outliers in A_M
No outliers in Type
```

8 Encoding

[217]:
```
df1=pd.get_dummies(data=df,columns=["Color","Spectral_Class"],drop_first=True)
```

[218]:
```
df1
```

[218]:

	Temperature	L	R	A_M	Type	Color_Blue-white	\
0	3068	0.002400	0.1700	16.12	0	0	
1	3042	0.000500	0.1542	16.60	0	0	
2	2600	0.000300	0.1020	18.70	0	0	
3	2800	0.000200	0.1600	16.65	0	0	
4	1939	0.000138	0.1030	20.06	0	0	
..	…	…	…	…	…	…	
235	38940	374830.000000	1356.0000	-9.93	5	0	
236	30839	834042.000000	1194.0000	-10.63	5	0	
237	8829	537493.000000	1423.0000	-10.73	5	0	
238	9235	404940.000000	1112.0000	-11.23	5	0	
239	37882	294903.000000	1783.0000	-7.80	5	0	

	Color_Orange	Color_Red	Color_White	Color_yellow-white	\
0	0	1	0	0	
1	0	1	0	0	
2	0	1	0	0	
3	0	1	0	0	

```
4                0          1            0                    0
..             …          …            …                  …
235              0          0            0                    0
236              0          0            0                    0
237              0          0            1                    0
238              0          0            1                    0
239              0          0            0                    0

     Spectral_Class_B  Spectral_Class_F  Spectral_Class_G  Spectral_Class_K  \
0                   0                 0                 0                 0
1                   0                 0                 0                 0
2                   0                 0                 0                 0
3                   0                 0                 0                 0
4                   0                 0                 0                 0
..                …                 …               …                 …
235                 0                 0                 0                 0
236                 0                 0                 0                 0
237                 0                 0                 0                 0
238                 0                 0                 0                 0
239                 0                 0                 0                 0

     Spectral_Class_M  Spectral_Class_O
0                   1                 0
1                   1                 0
2                   1                 0
3                   1                 0
4                   1                 0
..                …                 …
235                 0                 1
236                 0                 1
237                 0                 0
238                 0                 0
239                 0                 1

[240 rows x 16 columns]
```

8.1 Reaaranging Columns

[219]:
```
df1 = df1[['Temperature', 'L', 'R', 'A_M', 'Color_Blue-white',
       'Color_Orange', 'Color_Red', 'Color_White', 'Color_yellow-white',
       'Spectral_Class_B', 'Spectral_Class_F', 'Spectral_Class_G',
       'Spectral_Class_K', 'Spectral_Class_M', 'Spectral_Class_O', 'Type']]
```

[220]:
```
df1
```

[220]:

	Temperature	L	R	A_M	Color_Blue-white	\
0	3068	0.002400	0.1700	16.12	0	
1	3042	0.000500	0.1542	16.60	0	
2	2600	0.000300	0.1020	18.70	0	
3	2800	0.000200	0.1600	16.65	0	
4	1939	0.000138	0.1030	20.06	0	
..	…	…	…	…	…	
235	38940	374830.000000	1356.0000	-9.93	0	
236	30839	834042.000000	1194.0000	-10.63	0	
237	8829	537493.000000	1423.0000	-10.73	0	
238	9235	404940.000000	1112.0000	-11.23	0	
239	37882	294903.000000	1783.0000	-7.80	0	

	Color_Orange	Color_Red	Color_White	Color_yellow-white	\
0	0	1	0	0	
1	0	1	0	0	
2	0	1	0	0	
3	0	1	0	0	
4	0	1	0	0	
..	…	…	…	…	
235	0	0	0	0	
236	0	0	0	0	
237	0	0	1	0	
238	0	0	1	0	
239	0	0	0	0	

	Spectral_Class_B	Spectral_Class_F	Spectral_Class_G	Spectral_Class_K	\
0	0	0	0	0	
1	0	0	0	0	
2	0	0	0	0	
3	0	0	0	0	
4	0	0	0	0	
..	…	…	…	…	
235	0	0	0	0	
236	0	0	0	0	
237	0	0	0	0	
238	0	0	0	0	
239	0	0	0	0	

	Spectral_Class_M	Spectral_Class_O	Type
0	1	0	0
1	1	0	0
2	1	0	0
3	1	0	0
4	1	0	0
..	…	…	…
235	0	1	5

```
236                0                1     5
237                0                0     5
238                0                0     5
239                0                1     5

[240 rows x 16 columns]
```

9 Feature Scaling

[221]:
```
scaler = StandardScaler()
```

[222]:
```
scaler.fit(df1.drop('Type',axis = 1))
```

[222]:
```
StandardScaler()
```

[223]:
```
scaled_features = scaler.transform(df1.drop('Type',axis = 1))
```

10 Feature Selection

[224]:
```
df_feat = pd.DataFrame(scaled_features,columns = df1.columns[:-1])
df_feat.head()
```

[224]:
```
   Temperature         L         R       A_M  Color_Blue-white  Color_Orange  \
0    -0.779382 -0.598624 -0.459210  1.116745         -0.453905     -0.130189
1    -0.782110 -0.598624 -0.459241  1.162414         -0.453905     -0.130189
2    -0.828477 -0.598624 -0.459342  1.362213         -0.453905     -0.130189
3    -0.807496 -0.598624 -0.459229  1.167171         -0.453905     -0.130189
4    -0.897819 -0.598624 -0.459340  1.491607         -0.453905     -0.130189

   Color_Red  Color_White  Color_yellow-white  Spectral_Class_B  \
0   1.069045    -0.229416           -0.258199         -0.486943
1   1.069045    -0.229416           -0.258199         -0.486943
2   1.069045    -0.229416           -0.258199         -0.486943
3   1.069045    -0.229416           -0.258199         -0.486943
4   1.069045    -0.229416           -0.258199         -0.486943

   Spectral_Class_F  Spectral_Class_G  Spectral_Class_K  Spectral_Class_M  \
0         -0.276104         -0.064685         -0.160128          1.078036
1         -0.276104         -0.064685         -0.160128          1.078036
2         -0.276104         -0.064685         -0.160128          1.078036
3         -0.276104         -0.064685         -0.160128          1.078036
4         -0.276104         -0.064685         -0.160128          1.078036

   Spectral_Class_O
0         -0.447214
```

```
1         -0.447214
2         -0.447214
3         -0.447214
4         -0.447214
```

[225]:
```
X = df_feat
y = df1['Type']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,␣
↪random_state=101)
knn = KNeighborsClassifier(n_neighbors = 1)
knn.fit(X_train,y_train)
```

[225]:
```
KNeighborsClassifier(n_neighbors=1)
```

[226]:
```
pred = knn.predict(X_test)
pred
```

[226]:
```
array([2, 3, 0, 3, 0, 4, 4, 3, 0, 0, 5, 3, 5, 3, 3, 1, 1, 5, 3, 2, 2, 0,
       5, 3, 0, 5, 5, 2, 3, 2, 5, 1, 3, 2, 1, 3, 3, 2, 3, 0, 3, 3, 0, 0,
       2, 4, 1, 1, 2, 0, 2, 4, 2, 4, 2, 2, 5, 1, 5, 2, 5, 5, 0, 2, 1, 0,
       5, 5, 0, 3, 4, 5], dtype=int64)
```

[227]:
```
print(confusion_matrix(y_test,pred))
```

```
[[13  0  0  0  0  0]
 [ 0  8  0  0  0  0]
 [ 0  0 12  0  0  0]
 [ 0  0  3 14  0  0]
 [ 0  0  0  1  6  0]
 [ 0  0  0  1  0 14]]
```

[228]:
```
print(classification_report(y_test,pred))
```

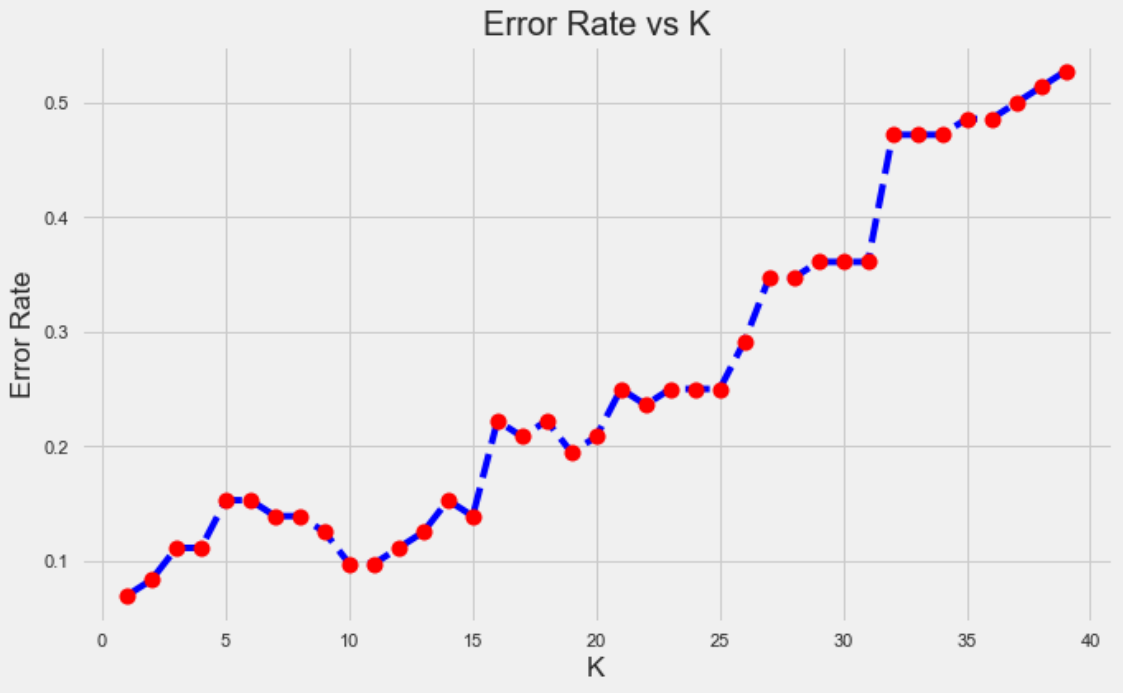
```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        13
           1       1.00      1.00      1.00         8
           2       0.80      1.00      0.89        12
           3       0.88      0.82      0.85        17
           4       1.00      0.86      0.92         7
           5       1.00      0.93      0.97        15

    accuracy                           0.93        72
   macro avg       0.95      0.94      0.94        72
weighted avg       0.94      0.93      0.93        72
```

11 finding least value for K to apply KNN

[229]:
```
error_rate= []
for i in range(1,40):
    knn = KNeighborsClassifier(n_neighbors = i)
    knn.fit(X_train,y_train)
    pred_i = knn.predict(X_test)
    error_rate.append(np.mean(pred_i != y_test))
```

[230]:
```
plt.figure(figsize = (10,6))
plt.plot(range(1,40),error_rate,color = 'blue',linestyle = '--',marker =␣
 ↪'o',markerfacecolor='red',markersize = 10)
plt.title('Error Rate vs K')
plt.xlabel('K')
plt.ylabel('Error Rate')
```

[230]:
```
Text(0, 0.5, 'Error Rate')
```



[231]:
```
print(metrics.accuracy_score(y_test, pred))
```

```
0.9305555555555556
```