In [30]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```

In [2]:
```
df = pd.read_csv('temperatures.csv')
```

In [3]:
```
df
```

Out[3]:

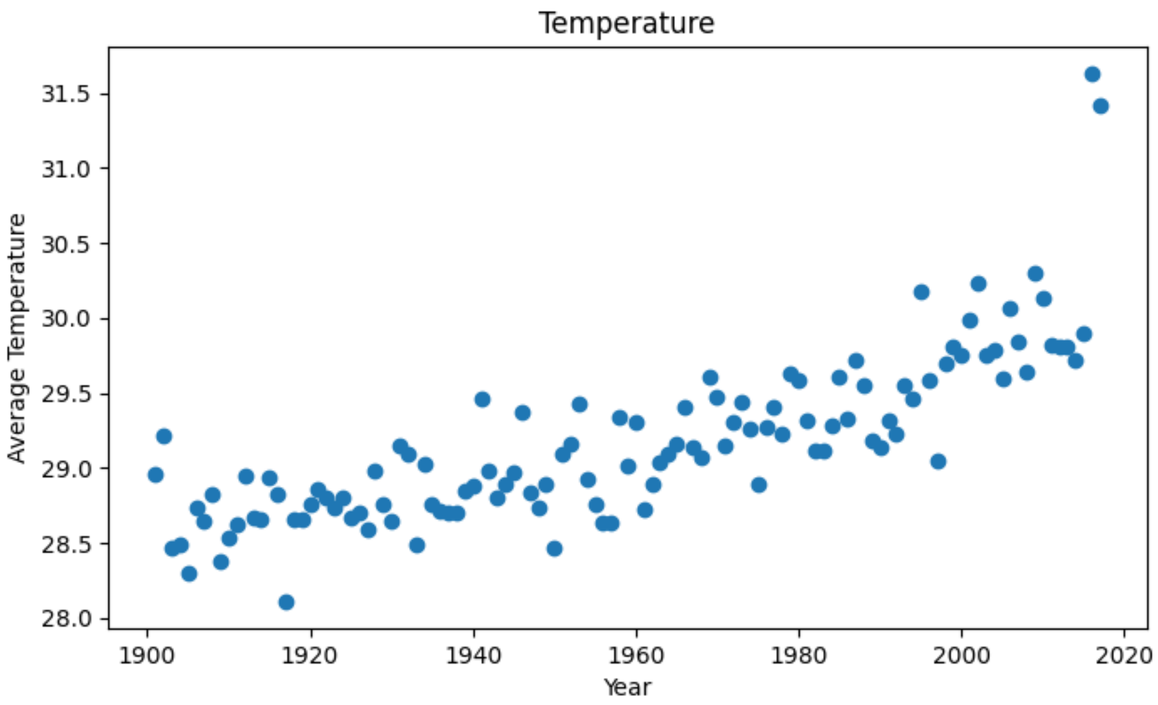
	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	N
0	1901	22.40	24.14	29.07	31.91	33.41	33.18	31.21	30.39	30.47	29.97	2
1	1902	24.93	26.58	29.77	31.78	33.73	32.91	30.92	30.73	29.80	29.12	2
2	1903	23.44	25.03	27.83	31.39	32.91	33.00	31.34	29.98	29.85	29.04	2
3	1904	22.50	24.73	28.21	32.02	32.64	32.07	30.36	30.09	30.04	29.20	2
4	1905	22.00	22.83	26.68	30.01	33.32	33.25	31.44	30.68	30.12	30.67	2
...	...	...	...	...	...	...	...	...	...	...	...	
112	2013	24.56	26.59	30.62	32.66	34.46	32.44	31.07	30.76	31.04	30.27	2
113	2014	23.83	25.97	28.95	32.74	33.77	34.15	31.85	31.32	30.68	30.29	2
114	2015	24.58	26.89	29.07	31.87	34.09	32.48	31.88	31.52	31.55	31.04	2
115	2016	26.94	29.72	32.62	35.38	35.72	34.03	31.64	31.79	31.66	31.98	
116	2017	26.45	29.46	31.60	34.95	35.84	33.82	31.88	31.72	32.22	32.29	2

117 rows × 18 columns

In [4]:
```
x = df['YEAR']
y = df['ANNUAL']
```

In [7]:
```
plt.figure(figsize=(8,4.5))
plt.title('Temperature')
plt.xlabel('Year')
plt.ylabel('Average Temperature')
plt.scatter(x,y)
```

Out[7]:
```
<matplotlib.collections.PathCollection at 0x7ffa0268dff0>
```



```
In [8]:  from sklearn.linear_model import LinearRegression

In [12]: x = x.values

In [13]: x = x.reshape(117,1)

In [10]: regressor = LinearRegression()

In [14]: regressor.fit(x,y)

Out[14]: ▾ LinearRegression
         LinearRegression()

In [15]: regressor.coef_

Out[15]: array([0.01312158])

In [16]: regressor.intercept_

Out[16]: 3.4761897126187016

In [20]: regressor.predict([[2024]])

Out[20]: array([30.03427031])

In [21]: predicted = regressor.predict([[2078]])

In [25]: predicted = regressor.predict(x)

In [27]: predicted
```

```
Out[27]:  array([28.4203158 , 28.43343739, 28.44655897, 28.45968055, 28.47280213,
                 28.48592371, 28.49904529, 28.51216687, 28.52528846, 28.53841004,
                 28.55153162, 28.5646532 , 28.57777478, 28.59089636, 28.60401794,
                 28.61713952, 28.63026111, 28.64338269, 28.65650427, 28.66962585,
                 28.68274743, 28.69586901, 28.70899059, 28.72211218, 28.73523376,
                 28.74835534, 28.76147692, 28.7745985 , 28.78772008, 28.80084166,
                 28.81396324, 28.82708483, 28.84020641, 28.85332799, 28.86644957,
                 28.87957115, 28.89269273, 28.90581431, 28.91893589, 28.93205748,
                 28.94517906, 28.95830064, 28.97142222, 28.9845438 , 28.99766538,
                 29.01078696, 29.02390855, 29.03703013, 29.05015171, 29.06327329,
                 29.07639487, 29.08951645, 29.10263803, 29.11575961, 29.1288812 ,
                 29.14200278, 29.15512436, 29.16824594, 29.18136752, 29.1944891 ,
                 29.20761068, 29.22073227, 29.23385385, 29.24697543, 29.26009701,
                 29.27321859, 29.28634017, 29.29946175, 29.31258333, 29.32570492,
                 29.3388265 , 29.35194808, 29.36506966, 29.37819124, 29.39131282,
                 29.4044344 , 29.41755599, 29.43067757, 29.44379915, 29.45692073,
                 29.47004231, 29.48316389, 29.49628547, 29.50940705, 29.52252864,
                 29.53565022, 29.5487718 , 29.56189338, 29.57501496, 29.58813654,
                 29.60125812, 29.6143797 , 29.62750129, 29.64062287, 29.65374445,
                 29.66686603, 29.67998761, 29.69310919, 29.70623077, 29.71935236,
                 29.73247394, 29.74559552, 29.7587171 , 29.77183868, 29.78496026,
                 29.79808184, 29.81120342, 29.82432501, 29.83744659, 29.85056817,
                 29.86368975, 29.87681133, 29.88993291, 29.90305449, 29.91617608,
                 29.92929766, 29.94241924])
```

```
In [28]:  y
```

```
Out[28]:  0      28.96
          1      29.22
          2      28.47
          3      28.49
          4      28.30
                 ...
          112    29.81
          113    29.72
          114    29.90
          115    31.63
          116    31.42
          Name: ANNUAL, Length: 117, dtype: float64
```

```
In [31]:  #mean absolute error
          np.mean(abs(y -  predicted))
```

```
Out[31]:  0.22535284978630413
```

```
In [32]:  from sklearn.metrics import mean_squared_error
          mean_squared_error(y,predicted)
```

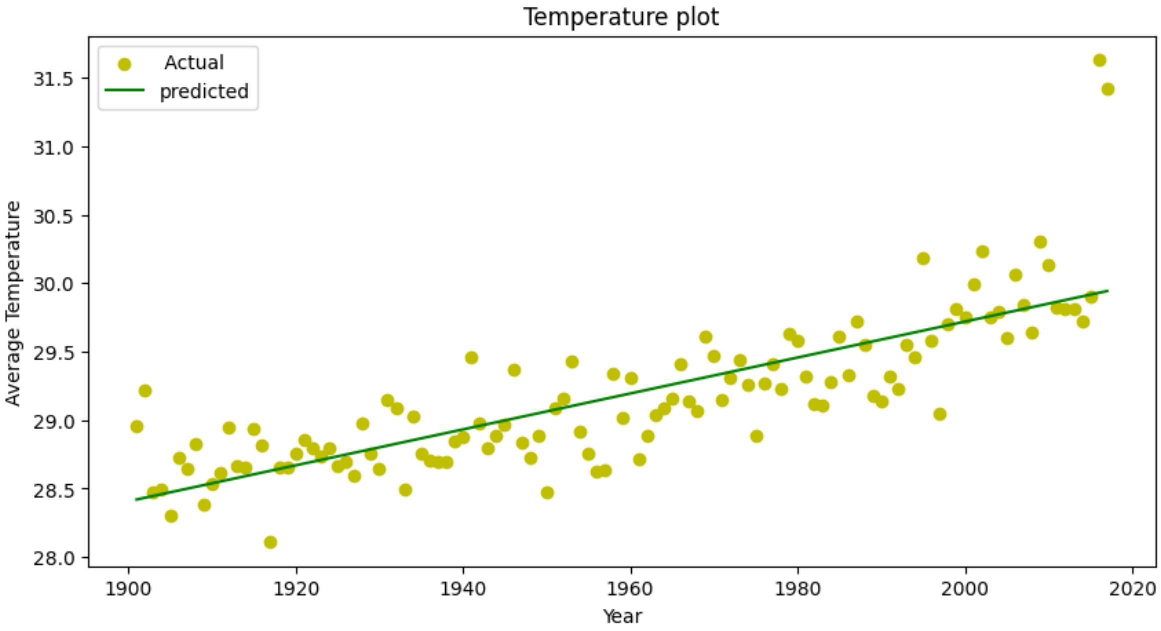
```
Out[32]:  0.10960795229110352
```

```
In [33]:  from sklearn.metrics import r2_score
          r2_score(y,predicted)
```

```
Out[33]:  0.6418078912783682
```

```
In [41]:  plt.figure(figsize=(10,5))
          plt.title('Temperature plot')
          plt.xlabel('Year')
          plt.ylabel('Average Temperature')
          plt.scatter(x,y, label= ' Actual', color = 'y')
          plt.plot(x, predicted, label = 'predicted' , color = 'g')
          plt.legend()
```

```
Out[41]: <matplotlib.legend.Legend at 0x7ff9fe073ca0>
```



```
In [ ]:
```