Seattle Weather

May 31, 2024

1 Has rain gotten more frequent in Seattle?

1.1 Madhavi Ghanta

https://www.kaggle.com/rtatman/did-it-rain-in-seattle-19482017

```
[1]: import pandas as pd
     import matplotlib.pyplot as plt
     import seaborn as sns
     import numpy as np
     from scipy.stats import binom
     %matplotlib inline
```

```
[3]: rain = pd.read_csv("C:/Users/mghan/Documents/MSDS/Portfolio/
       ↪seattleWeather_1948-2017.csv")
```

```
[4]: rain.describe()
     rain.head()
```

```
[4]:        DATE  PRCP  TMAX  TMIN  RAIN
     0  1/1/1948  0.47    51    42  True
     1  1/2/1948  0.59    45    36  True
     2  1/3/1948  0.42    45    35  True
     3  1/4/1948  0.31    45    34  True
     4  1/5/1948  0.17    45    32  True
```

```
[8]: #Bad Values within the data set
     rain[pd.isnull(rain).any(axis=1)]
```

```
[8]:            DATE  PRCP  TMAX  TMIN RAIN
     18415  6/2/1998   NaN    72    52  NaN
     18416  6/3/1998   NaN    66    51  NaN
     21067  9/5/2005   NaN    70    52  NaN
```

```
[9]: #Cleaning data set
     rain.dropna(axis=1, how='all', inplace=True)
```

2 Hypothesis:

Rain has been steadily increasing in Seattle. Null Hypothesis: Rain has has not increased or changed over time

2.1 Describe what the 5 variables mean in the dataset (Chapter 1).

DATE: the date of the observation

PRCP: the amount of precipitation, in inches

TMAX: the maximum temperature for that day, in degrees Fahrenheit

TMIN: the minimum temperature for that day, in degrees Fahrenheit

RAIN: TRUE if rain was observed on that day, FALSE if it was not

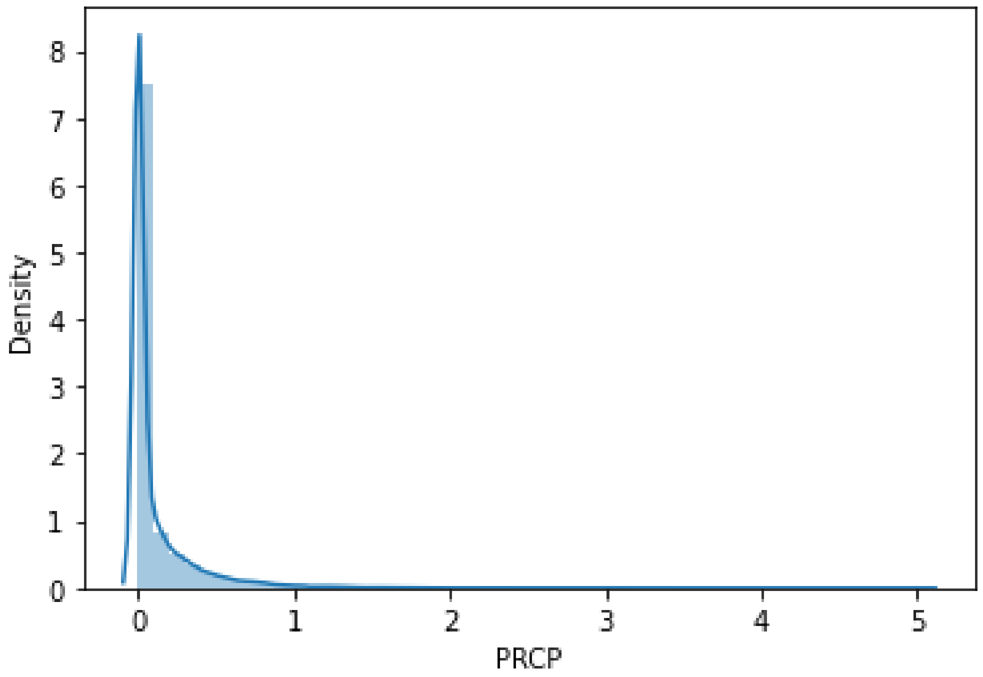
---

Include a histogram of each of the 5 variables – in your summary and analysis, identify any outliers and explain the reasoning for them being outliers and how you believe they should be handled (Chapter 2).

```
[10]: #Skipping date on histogram
      sns.distplot(rain['PRCP'].dropna())
```

```
C:\Users\mghan\anaconda3\lib\site-packages\seaborn\distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed in a
future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for
histograms).
  warnings.warn(msg, FutureWarning)
```

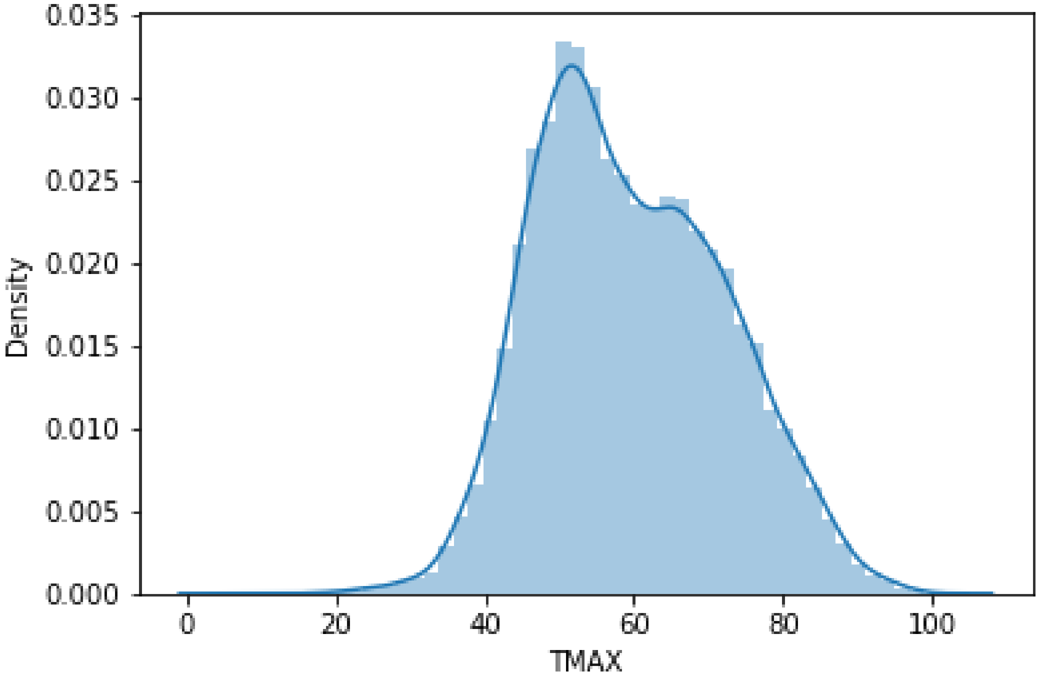
```
[10]: <AxesSubplot:xlabel='PRCP', ylabel='Density'>
```



[11]:
```
sns.distplot(rain['TMAX'].dropna())
```

```
C:\Users\mghan\anaconda3\lib\site-packages\seaborn\distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed in a
future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for
histograms).
  warnings.warn(msg, FutureWarning)
```

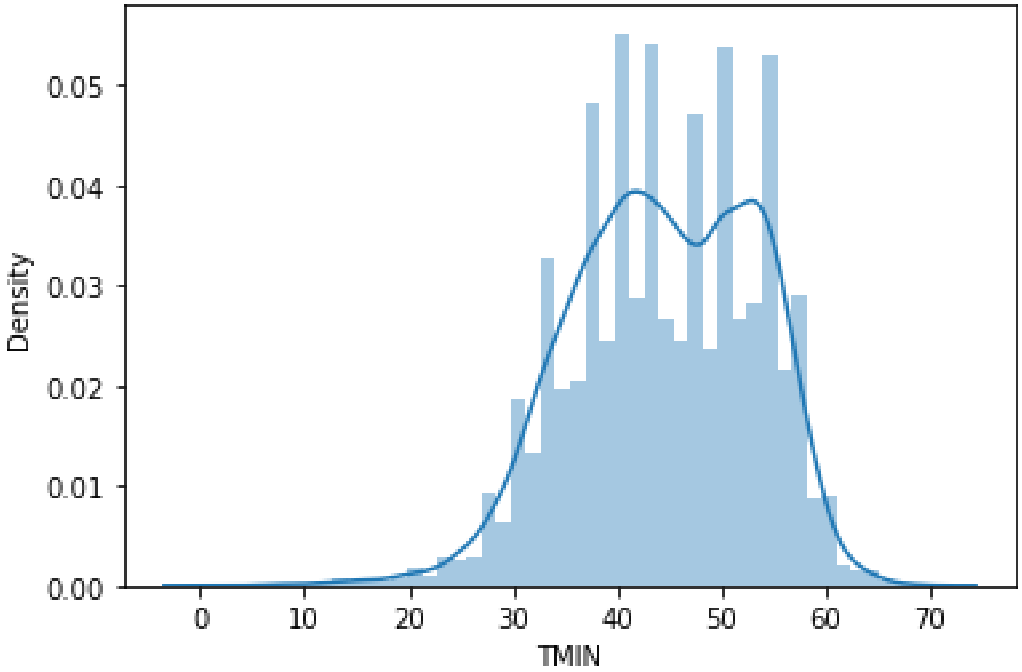
[11]:
```
<AxesSubplot:xlabel='TMAX', ylabel='Density'>
```



```
[12]: sns.distplot(rain['TMIN'].dropna())
```

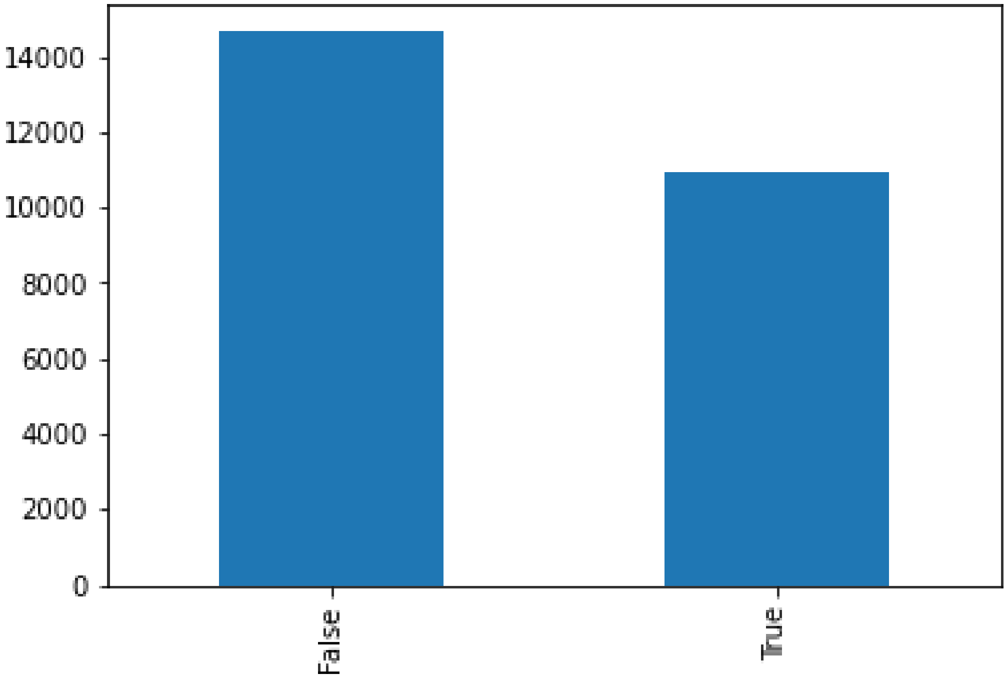
```
C:\Users\mghan\anaconda3\lib\site-packages\seaborn\distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed in a
future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for
histograms).
  warnings.warn(msg, FutureWarning)
```

```
[12]: <AxesSubplot:xlabel='TMIN', ylabel='Density'>
```



```
[13]: rain['RAIN'].value_counts().plot(kind='bar', label='Rain')
```

```
[13]: <AxesSubplot:>
```



Include the other descriptive characteristics about the variables: * Mean * Mode * Spread * Tails

```
[14]: rain['TMIN'].mean(), rain['TMIN'].median(), rain['TMIN'].mode(), rain['TMIN'].
       ↪std()
```

```
[14]: (44.51422644906266,
       45.0,
       0    42
       Name: TMIN, dtype: int64,
       8.892835742411922)
```

```
[15]: rain['DATE'].max(), rain['DATE'].min()
```

```
[15]: ('9/9/2017', '1/1/1948')
```

```
[16]: rain['DATE']= pd.to_datetime(rain['DATE'])
```

```
[17]: rain['YEAR']= rain['DATE'].dt.year
```

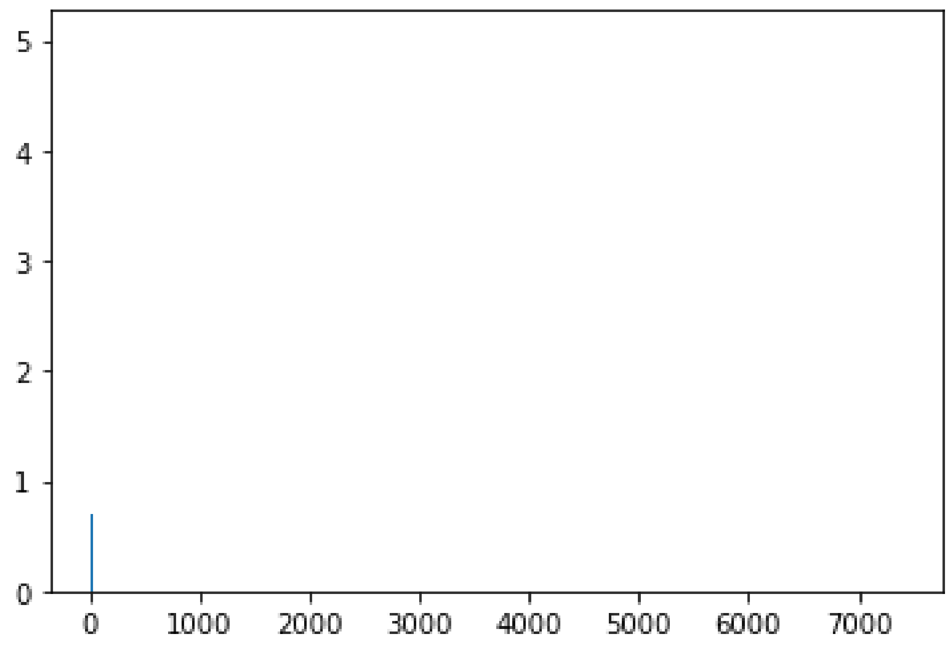
```
[18]: rain.head()
```

```
[18]:         DATE  PRCP  TMAX  TMIN  RAIN  YEAR
      0 1948-01-01  0.47    51    42  True  1948
      1 1948-01-02  0.59    45    36  True  1948
      2 1948-01-03  0.42    45    35  True  1948
      3 1948-01-04  0.31    45    34  True  1948
      4 1948-01-05  0.17    45    32  True  1948
```

```
[19]: first_half = rain[rain["DATE"]>='1982-12-22 00:00:00']
      second_half = rain[rain["DATE"]<='1982-12-23 00:00:00']
```

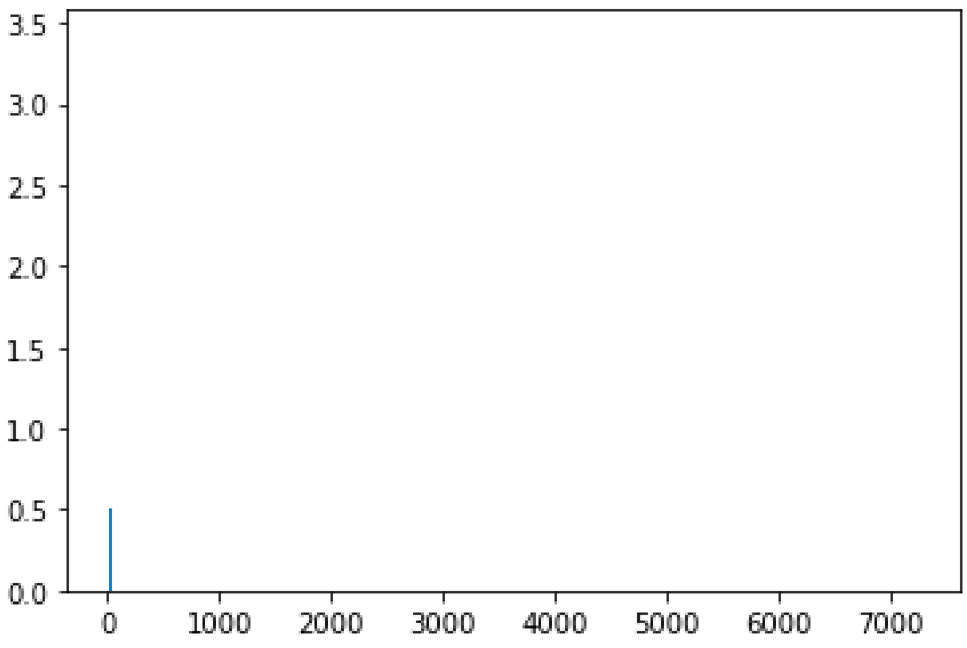
```
[22]: plt.bar(list(pmf_first.values()),list(pmf_first.keys()))
```

```
[22]: <BarContainer object of 183 artists>
```



[25]:
```
plt.bar(list(pmf_second.values()), list(pmf_second.keys()))
```

[25]:
```
<BarContainer object of 175 artists>
```



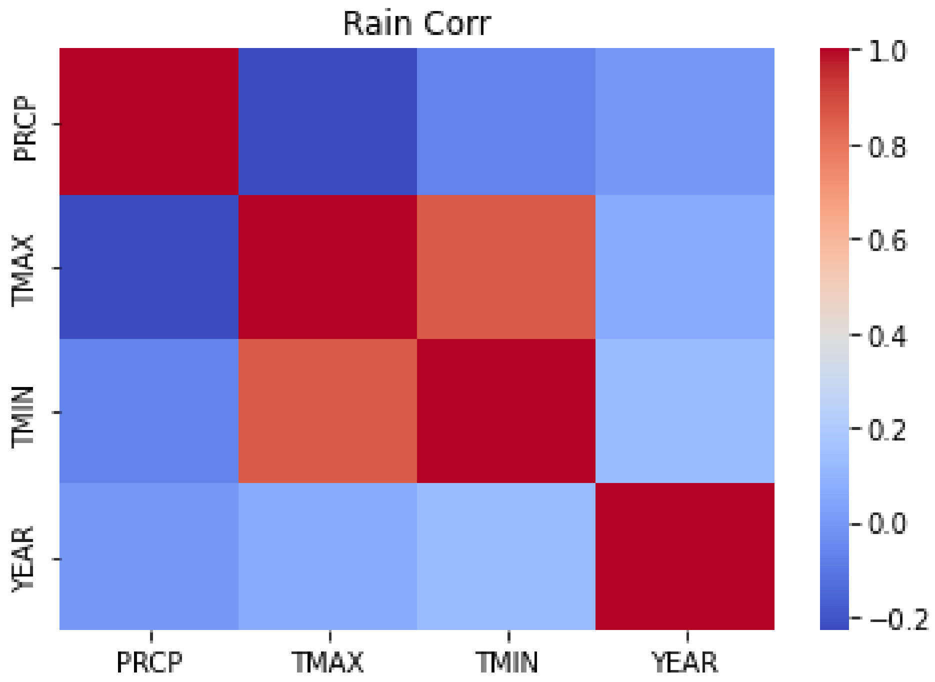
```
[113]: pmf
```

```
[113]: {0.0: 7246,
        0.01: 462,
        0.02: 353,
        0.03: 258,
        0.04: 198,
        0.05: 212,
        0.06: 209,
        0.07: 170,
        0.08: 174,
        0.09: 130,
        0.1: 129,
        0.11: 142,
        0.12: 146,
        0.13: 130,
        0.14: 112,
        0.15: 108,
        0.16: 97,
        0.17: 96,
        0.18: 70,
        0.19: 63,
        0.2: 101,
        0.21: 73,
        0.22: 62,
        0.23: 77,
        0.24: 70,
        0.25: 62,
        0.26: 59,
        0.27: 51,
        0.28: 68,
        0.29: 64,
        0.3: 67,
        0.31: 54,
        0.32: 38,
        0.33: 51,
        0.34: 47,
        0.35: 53,
        0.36: 49,
        0.37: 41,
        0.38: 44,
        0.39: 42,
        0.4: 29,
        0.41: 28,
        0.42: 32,
        0.43: 26,
        0.44: 33,
        0.45: 46,
        0.46: 27,
        0.47: 30,
        0.48: 22,
        0.49: 31,
        0.5: 26,
        0.51: 24,
        0.52: 17,
        0.53: 25,
        0.54: 24,
        0.55: 19,
        0.56: 35,
        0.57: 15,
        0.58: 21,
        0.59: 13,
        0.6: 24,
        0.61: 16,
        0.62: 16,
        0.63: 13,
        0.64: 13,
        0.65: 9,
        0.66: 17,
        0.67: 18,
        0.68: 8,
        0.69: 9,
        0.7: 17,
        0.71: 15,
        0.72: 9,
        0.73: 13,
        0.74: 20,
        0.75: 7,
        0.76: 11,
        0.77: 9,
        0.78: 9,
        0.79: 13,
        0.8: 13,
        0.81: 8,
        0.82: 4,
        0.83: 13,
        0.84: 10,
        0.85: 6,
        0.86: 5,
        0.87: 8,
        0.88: 12,
        0.89: 3,
        0.9: 2,
        0.91: 6,
        0.92: 7,
        0.93: 4,
        0.94: 5,
        0.95: 4,
        0.96: 8,
        0.97: 3,
        0.98: 6,
        0.99: 6,
        1.0: 0.00023090169067000628,
        1.01: 3,
        1.02: 2,
        1.03: 3,
        1.04: 6,
        1.05: 7,
        1.06: 6,
        1.07: 2,
        1.08: 5,
        1.09: 1,
        1.11: 1,
        1.12: 1,
        1.13: 5,
        1.14: 1,
        1.15: 1,
        1.16: 3,
        1.17: 5,
        1.18: 5,
        1.19: 4,
        1.2: 2,
        1.21: 2,
        1.22: 3,
        1.23: 6,
        1.24: 2,
        1.26: 5,
        1.27: 9,
        1.28: 4,
        1.29: 3,
        1.3: 1,
        1.31: 1,
        1.32: 4,
        1.33: 1,
        1.34: 1,
        1.36: 2,
        1.37: 2,
        1.38: 1,
        1.39: 2,
        1.4: 1,
        1.45: 3,
        1.46: 3,
        1.48: 2,
        1.49: 1,
        1.5: 1,
        1.51: 1,
        1.52: 1,
        1.53: 1,
        1.54: 2,
        1.55: 2,
        1.56: 1,
        1.6: 2,
        1.61: 3,
        1.63: 3,
        1.64: 2,
        1.65: 2,
        1.66: 2,
        1.67: 2,
        1.68: 1,
        1.7: 1,
        1.75: 2,
        1.76: 1,
        1.78: 2,
        1.83: 2,
        1.85: 2,
        1.93: 1,
        2: 0.00010331872260488416,
        2.04: 1,
        2.08: 1,
        2.14: 1,
        2.18: 1,
        2.23: 1,
        2.26: 1,
        2.58: 1,
        2.7: 1,
        2.72: 1,
        2.98: 1,
        3: 7.827175954915466e-05,
        3.41: 1,
        4: 0.00010644959298685035,
        5: 0.00010488415779586725,
        6: 8.140262993112086e-05,
        7: 6.105197244834064e-05,
        8: 6.966186599874766e-05,
        9: 6.809643080776456e-05,
        10: 5.165936130244208e-05,
        11: 5.9486537257357546e-05,
        12: 5.5572949279899807e-05,
        13: 5.8703819661866e-05,
        14: 5.3224796493425174e-05,
        15: 4.8528490920475895e-05,
        16: 3.757044458359424e-05,
        17: 4.53976205385097e-05,
        18: 5.0876643706950534e-05,
        19: 4.618033813400125e-05,
        20: 4.304946775203507e-05,
        22: 4.148403256105198e-05,
        23: 4.226675015654352e-05,
        25: 3.8353162179085784e-05,
        27: 3.678772698810269e-05,
        28: 3.209142141515341e-05,
        29: 3.913587977457733e-05,
        30: 3.6005009392611145e-05,
        31: 3.287413901064496e-05,
        32: 3.52222917971196e-05,
        33: 3.443957420162805e-05,
        35: 3.052598622417032e-05,
        38: 2.739511584220413e-05,
        40: 2.582968065122104e-05,
        43: 2.8960551033187226e-05,
        44: 2.4264245460237948e-05,
        45: 2.9743268628678773e-05,
        51: 2.3481527864746398e-05,
        52: 2.6612398246712587e-05,
        54: 2.113337507827176e-05,
        56: 2.269881026925485e-05,
        57: 1.8002504696305572e-05,
        61: 2.5046963055729494e-05,
        68: 2.0350657482780215e-05,
        69: 1.643706950532248e-05,
        71: 1.7219787100814026e-05,
        72: 1.5654351909830933e-05,
        88: 1.4871634314339386e-05,
        90: 1.408891671884784e-05,
        94: 1.3306199123356293e-05,
        111: 1.1740763932373199e-05,
        112: 1.0958046336881654e-05,
        113: 1.0175328741390108e-05,
        118: 1.2523481527864747e-05,
        124: 9.39261114589856e-06,
        132: 7.827175954915466e-06,
        136: 7.04445835942392e-06,
        157: 5.479023168440827e-06,
        161: 6.2617407639323735e-06,
        199: 4.69630557294928e-06,
        207: 3.9135879774577733e-06,
        230: 3.1308703819661867e-06,
        235: 2.348152786474464e-06,
        354: 1.5654351909830934e-06,
        471: 7.827175954915467e-07,
        7403: 0.0}
```

[27]:
```
#Create 1 CDF with one of your variables, using page 41-44 as your guide, what does this tell you about your variable and how does it address the question you are trying to answer (Chapter 4).
```

[29]:
```
sns.heatmap(rain.corr(), cmap='coolwarm')
plt.title('Rain Corr')
```

[29]:
```
Text(0.5, 1.0, 'Rain Corr')
```



[30]:
```
rain_per_year = rain[(rain['RAIN']==True)].groupby('YEAR').count()
```

[31]:
```
year_rain_count=rain[rain['RAIN']==True].groupby('YEAR').count()
```

[32]:
```
rain_per_year['RAIN']=year_rain_count['RAIN']
```

[33]:
```
rain_per_year['TMIN MEAN']=rain.groupby('YEAR')['TMIN'].mean()
rain_per_year['TMAX MEAN']=rain.groupby('YEAR')['TMAX'].mean()
rain_per_year['PRCP MEAN']=rain.groupby('YEAR')['PRCP'].mean()
```

```
[ ]:
```

```
[34]: rain_per_year.reset_index(inplace=True)
```

```
[35]: rain_per_year = rain_per_year[['YEAR', 'RAIN', 'TMIN MEAN', 'TMAX MEAN', 'PRCP␣
        ↪MEAN']]
```

```
[36]: rain_per_year.head()
```

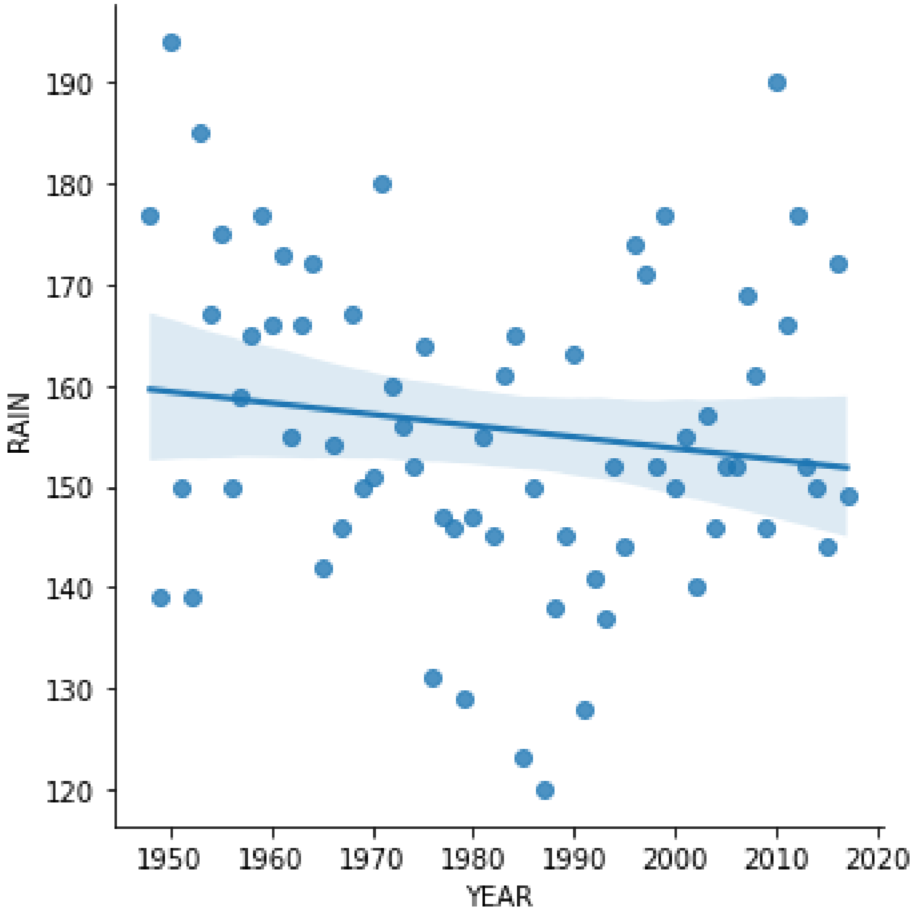
```
[36]:    YEAR  RAIN  TMIN MEAN  TMAX MEAN  PRCP MEAN
      0  1948   177  41.196721  57.013661   0.125109
      1  1949   139  41.391781  59.147945   0.088932
      2  1950   194  41.000000  57.035616   0.151068
      3  1951   150  41.052055  58.545205   0.110411
      4  1952   139  41.467213  58.743169   0.064973
```

```
[37]: rain_per_year.columns
```

```
[37]: Index(['YEAR', 'RAIN', 'TMIN MEAN', 'TMAX MEAN', 'PRCP MEAN'], dtype='object')
```

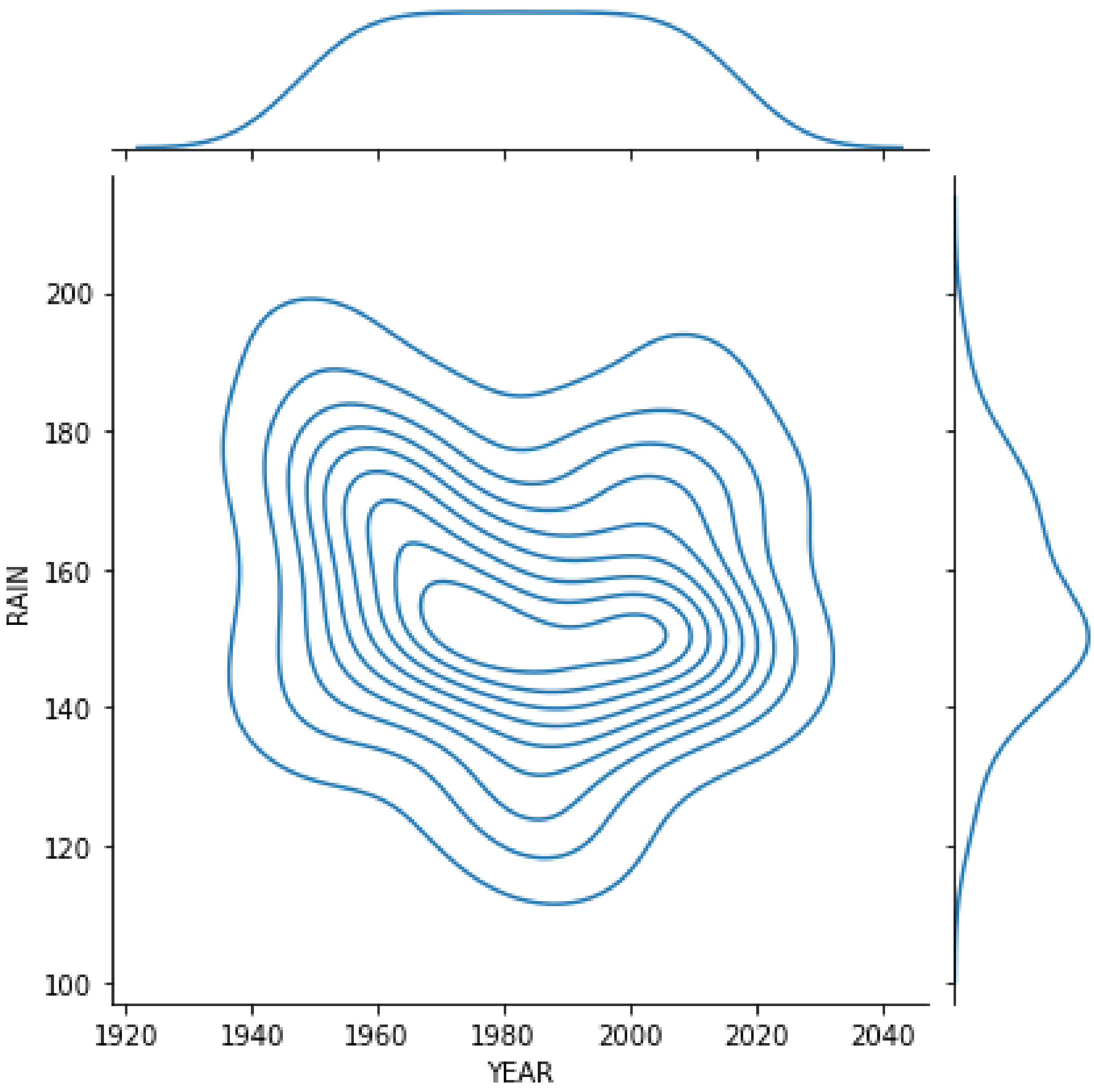
```
[38]: sns.lmplot(x='YEAR', y='RAIN', data=rain_per_year)
```

```
[38]: <seaborn.axisgrid.FacetGrid at 0x2b720ec2a90>
```



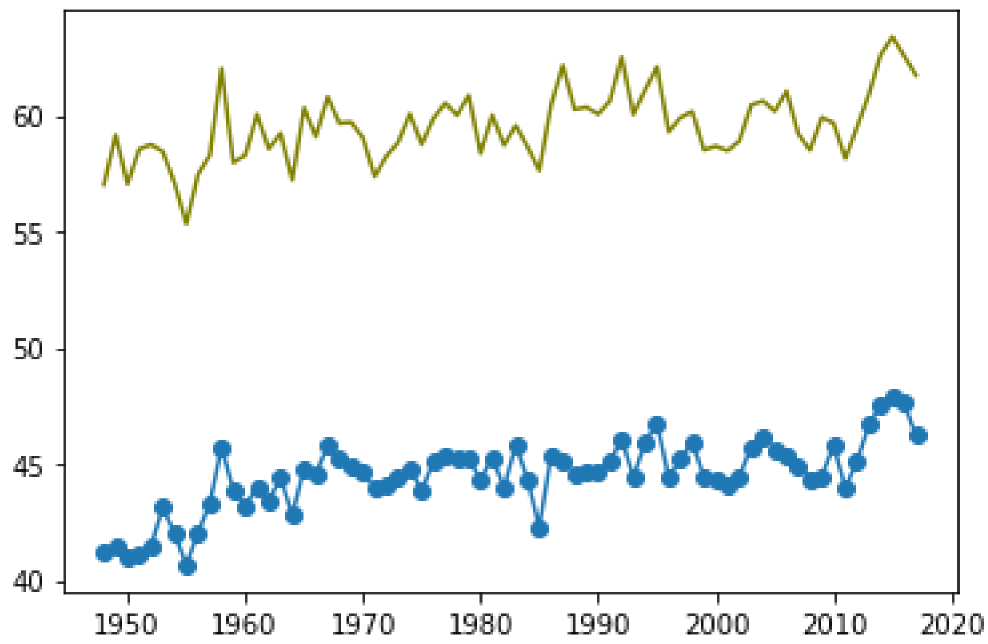
```
[39]: sns.jointplot(x='YEAR', y='RAIN', data=rain_per_year, kind='kde')
```

```
[39]: <seaborn.axisgrid.JointGrid at 0x2b720e14d00>
```



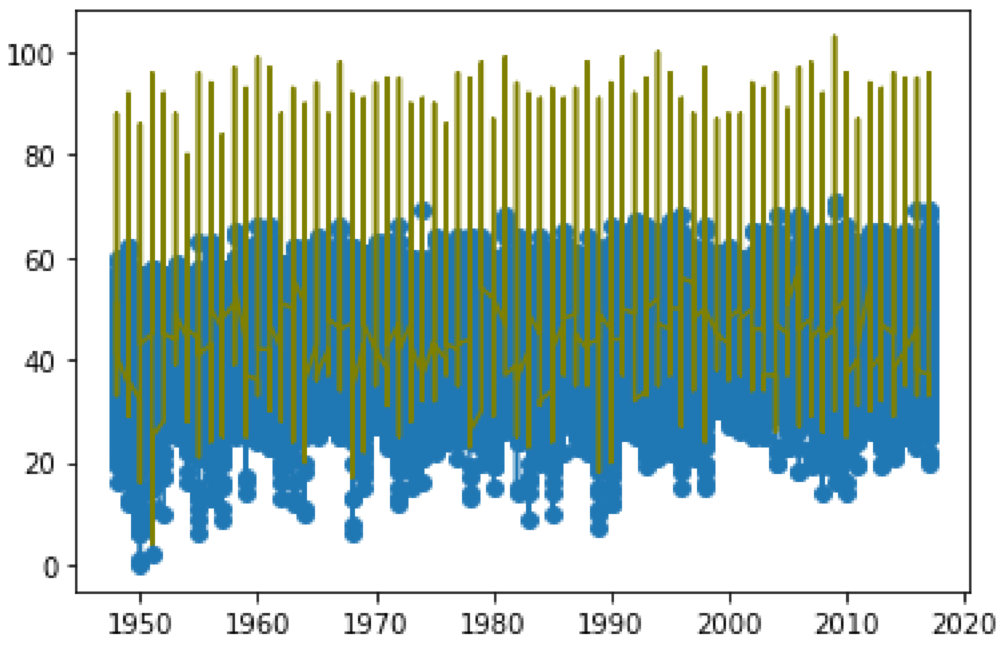
```
[40]: plt.plot('YEAR', 'TMIN MEAN', data=rain_per_year, marker='o')
      plt.plot('YEAR', 'TMAX MEAN', data=rain_per_year, marker='', color='olive')
```

```
[40]: [<matplotlib.lines.Line2D at 0x2b721153160>]
```



[41]:
```
plt.plot('YEAR', 'TMIN', data=rain, marker='o')
plt.plot('YEAR', 'TMAX', data=rain, marker='', color='olive')
```

[41]:
```
[<matplotlib.lines.Line2D at 0x2b7211afb80>]
```



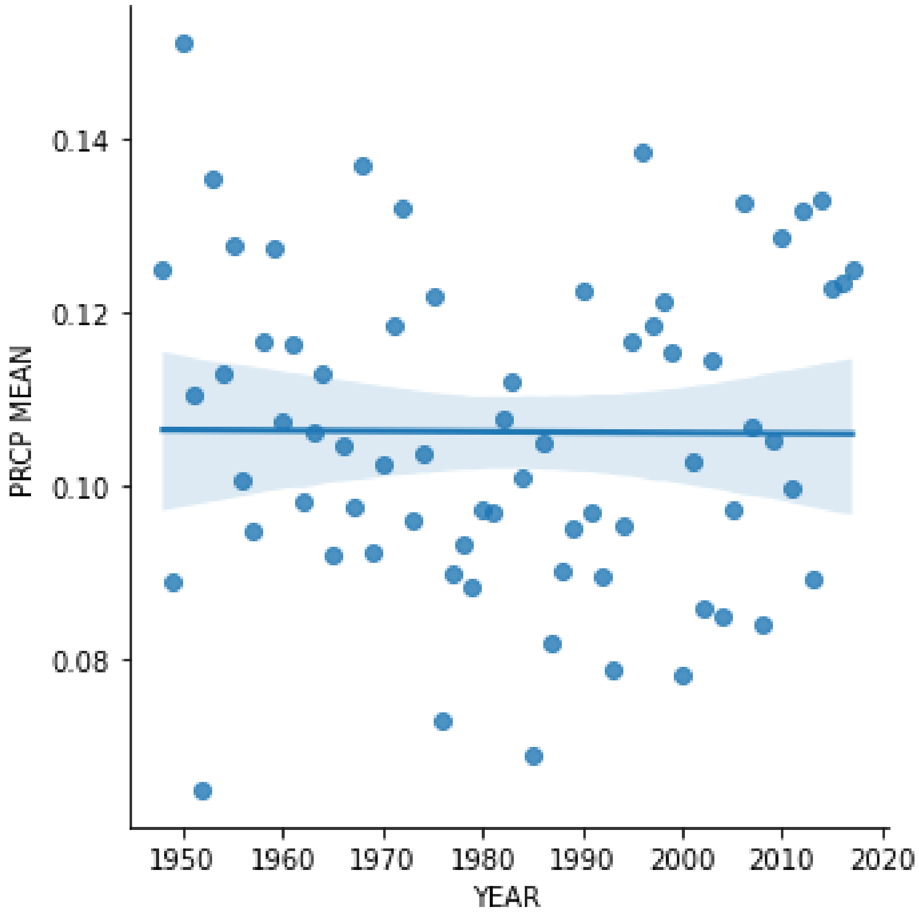
[42]:

```
sns.lmplot('YEAR', 'PRCP MEAN', data=rain_per_year)
```

```
C:\Users\mghan\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

[42]:

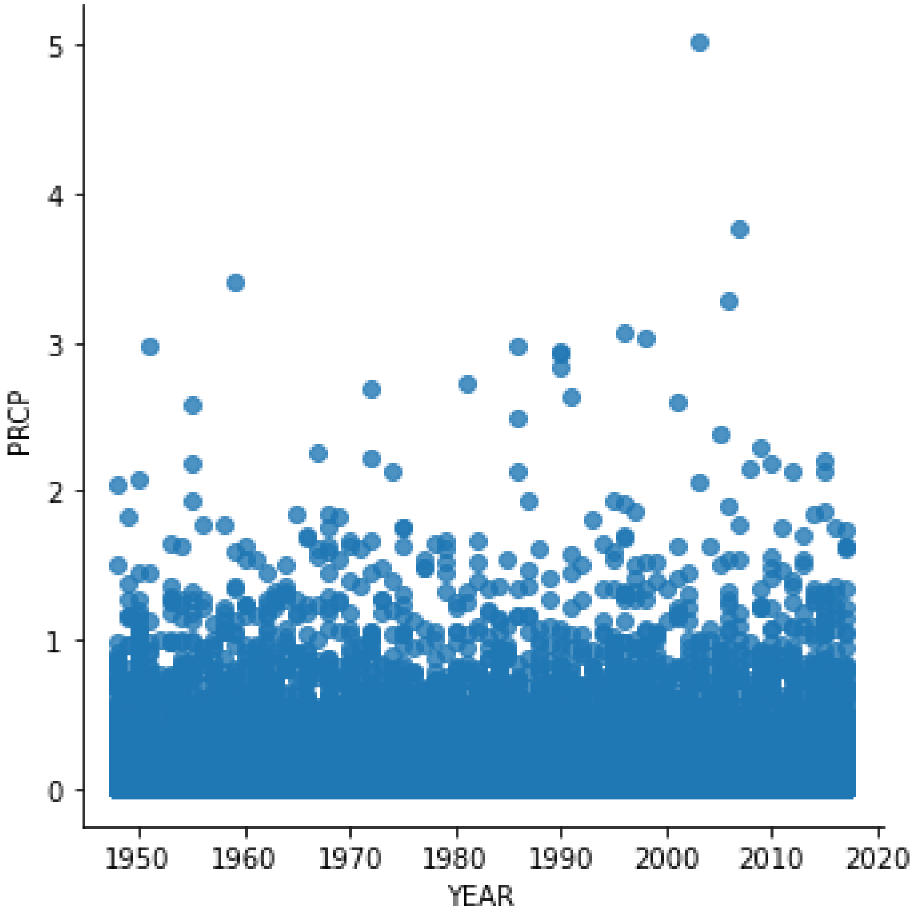
```
<seaborn.axisgrid.FacetGrid at 0x2b7211dd1f0>
```



[43]:

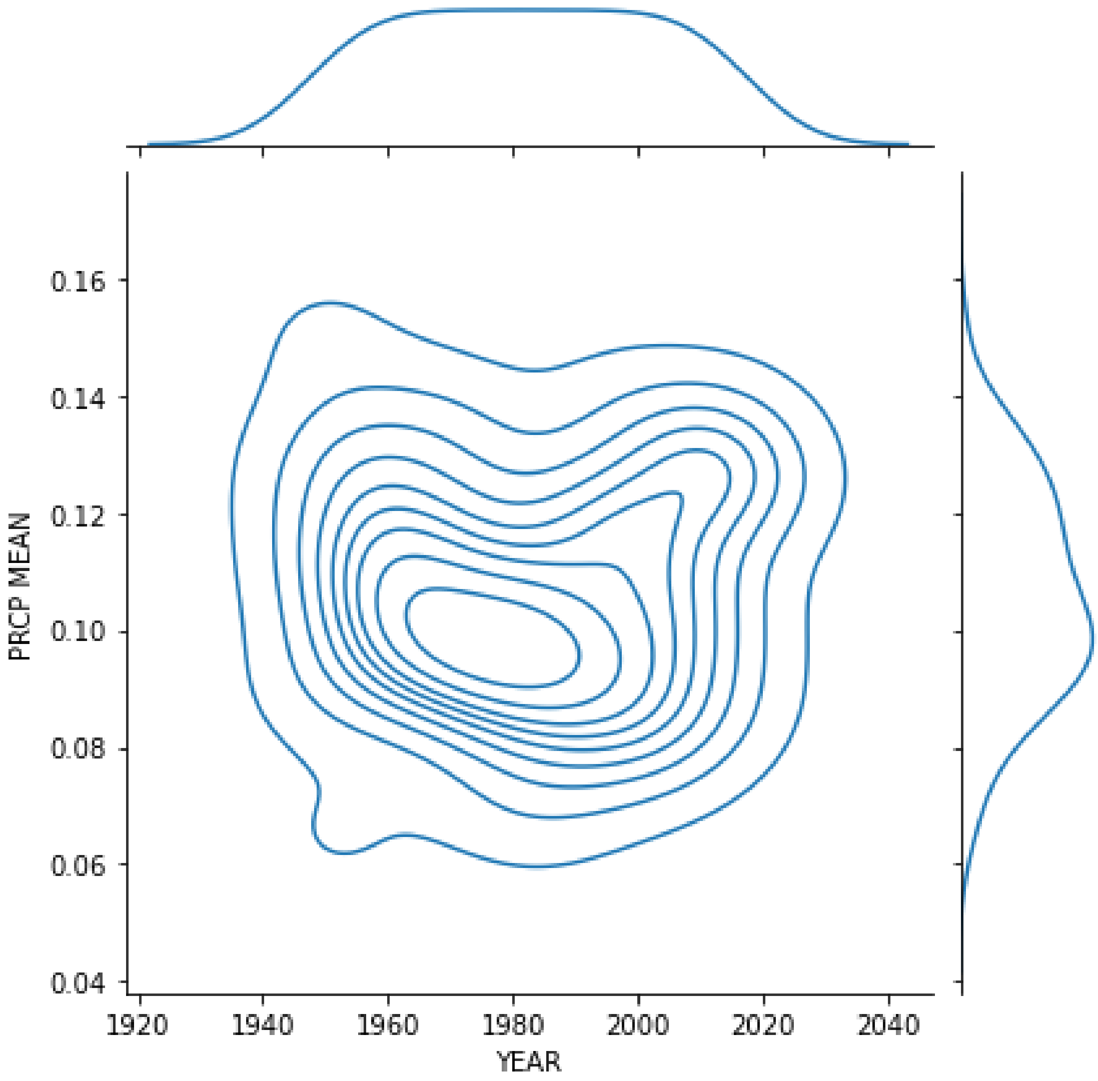
```
sns.lmplot('YEAR', 'PRCP', data=rain)
```

```
C:\Users\mghan\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
[43]: <seaborn.axisgrid.FacetGrid at 0x2b7211d6b50>
```



```
[44]: sns.jointplot(x='YEAR', y='PRCP MEAN', data=rain_per_year, kind='kde')
```

```
[44]: <seaborn.axisgrid.JointGrid at 0x2b721232520>
```



```
[50]: super_column = rain_per_year[['YEAR', 'TMIN MEAN', 'TMAX MEAN']]
```

```
[51]: super_column['Max']='Max'
```

```
C:\Users\mghan\AppData\Local\Temp\ipykernel_1592\3469924724.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  super_column['Max']='Max'
```

```
[52]: super_column['Min']='Min'
```

[53]:

```
max = super_column[['YEAR', 'TMAX MEAN', 'Max']]
min = super_column[['YEAR', 'TMIN MEAN', 'Min']]
```

[54]:

```
max['temp']=max['TMAX MEAN']
max['type']=max['Max']
min['temp']=min['TMIN MEAN']
min['type']=min['Min']
```

```
C:\Users\mghan\AppData\Local\Temp\ipykernel_1592\3706309177.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  max['temp']=max['TMAX MEAN']
C:\Users\mghan\AppData\Local\Temp\ipykernel_1592\3706309177.py:3:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  min['temp']=min['TMIN MEAN']
```

[55]:

```
max = max[['YEAR', 'temp', 'type']]
min = min[['YEAR', 'temp', 'type']]
```

[56]:

```
combined = max.append(min)
```

```
C:\Users\mghan\AppData\Local\Temp\ipykernel_1592\299298275.py:1: FutureWarning:
The frame.append method is deprecated and will be removed from pandas in a
future version. Use pandas.concat instead.
  combined = max.append(min)
```

[ ]:

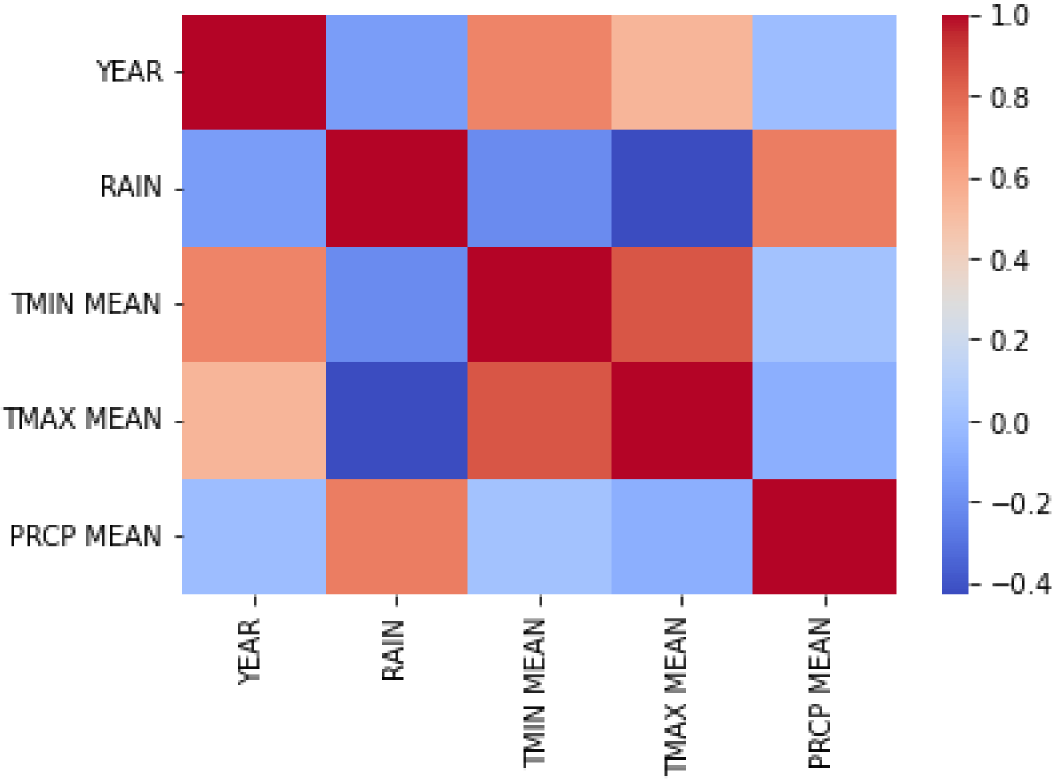
```

```

[57]:

```
sns.heatmap(rain_per_year.corr(), cmap='coolwarm')
```

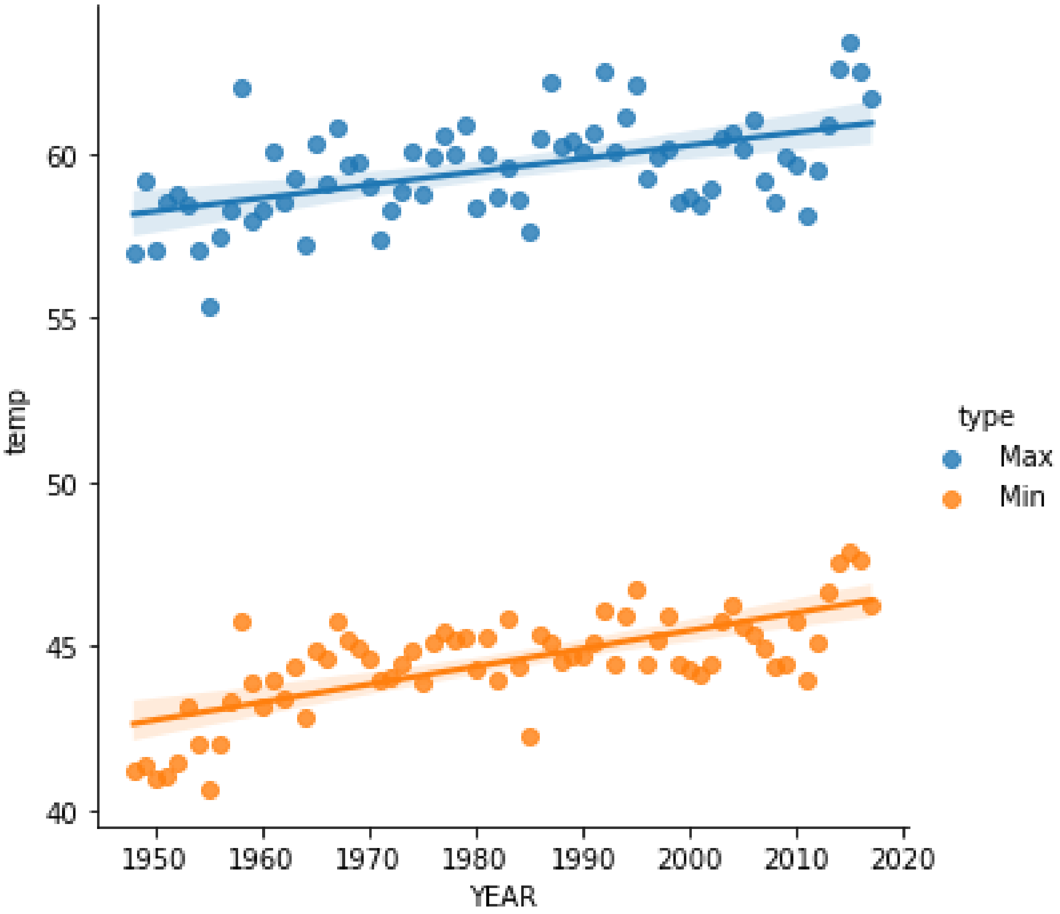
[57]:

```
<AxesSubplot:>
```



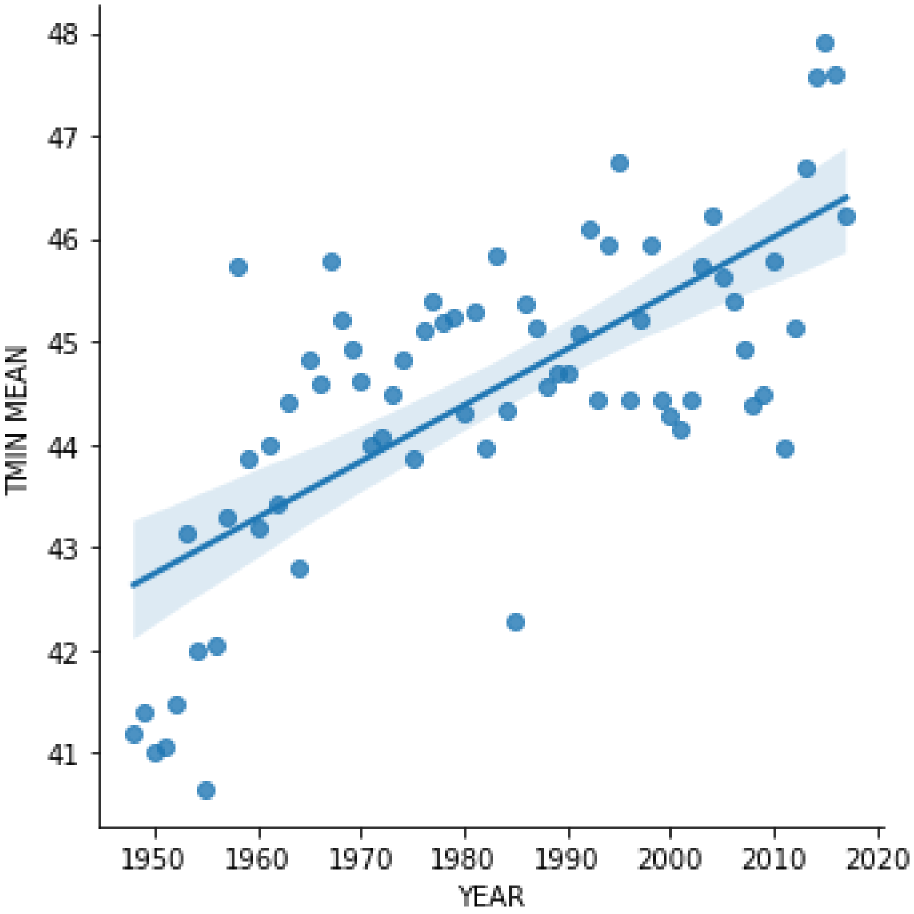
[58]:
```
sns.lmplot(x='YEAR', y='temp', hue='type', data=combined)
```

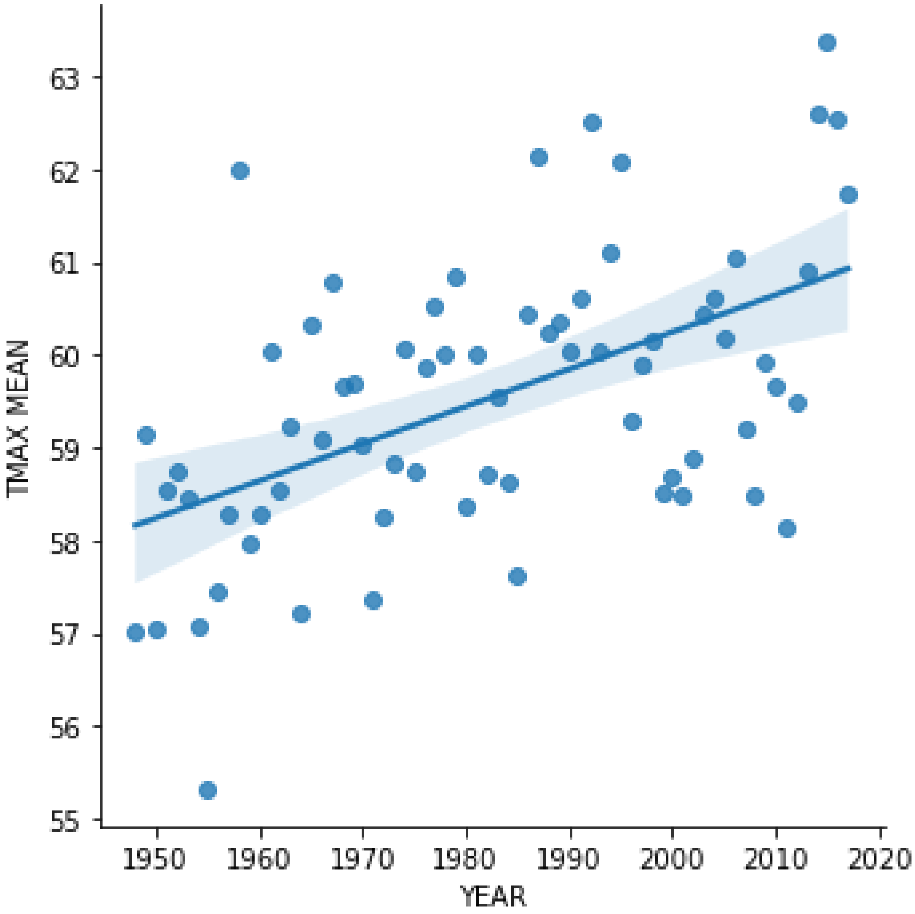
[58]:
```
<seaborn.axisgrid.FacetGrid at 0x2b7224527f0>
```



[59]:
```
sns.lmplot(x='YEAR', y='TMIN MEAN', data=rain_per_year)
sns.lmplot(x='YEAR', y='TMAX MEAN', data=rain_per_year)
```

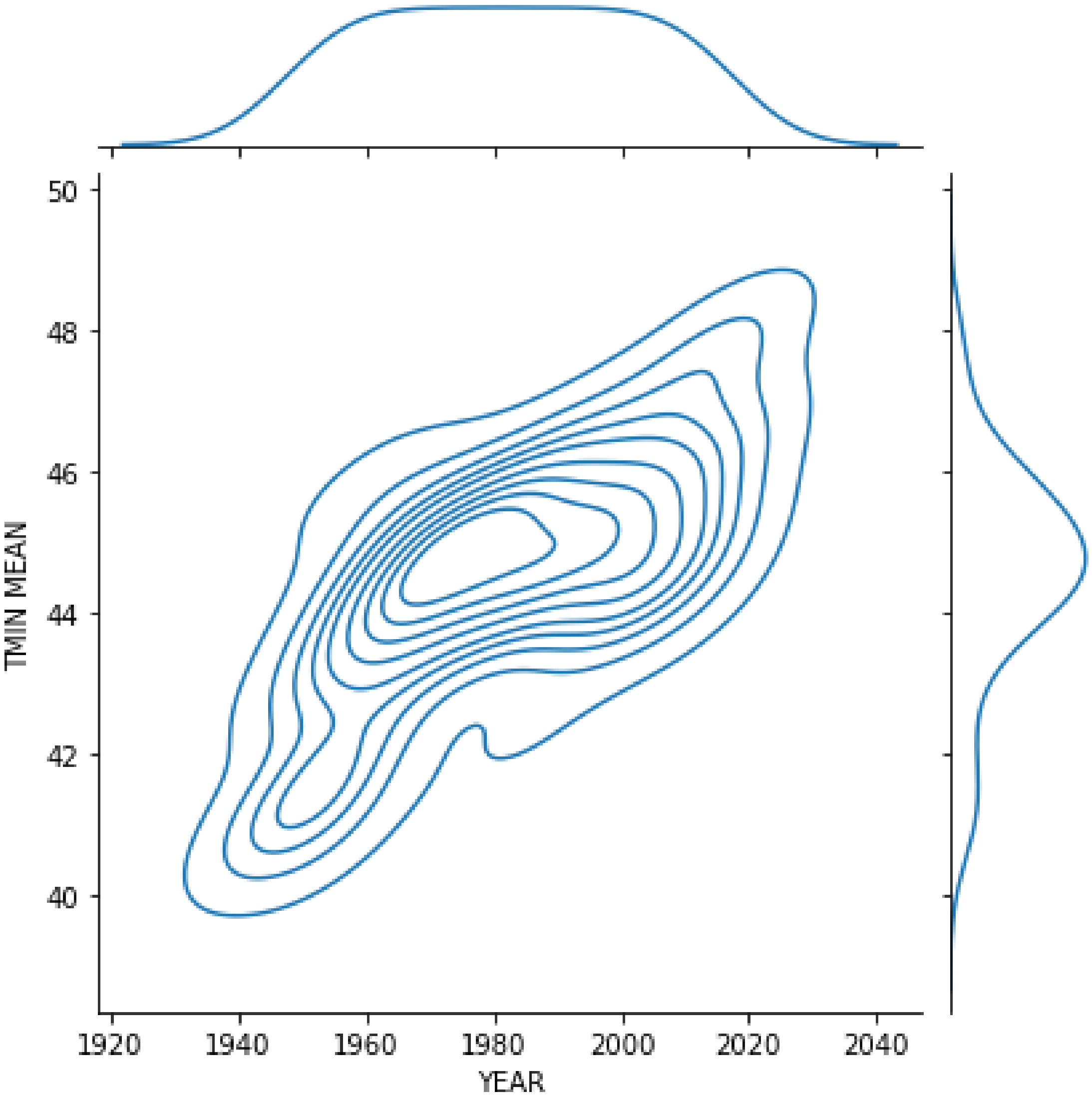
[59]:
```
<seaborn.axisgrid.FacetGrid at 0x2b7223c47c0>
```

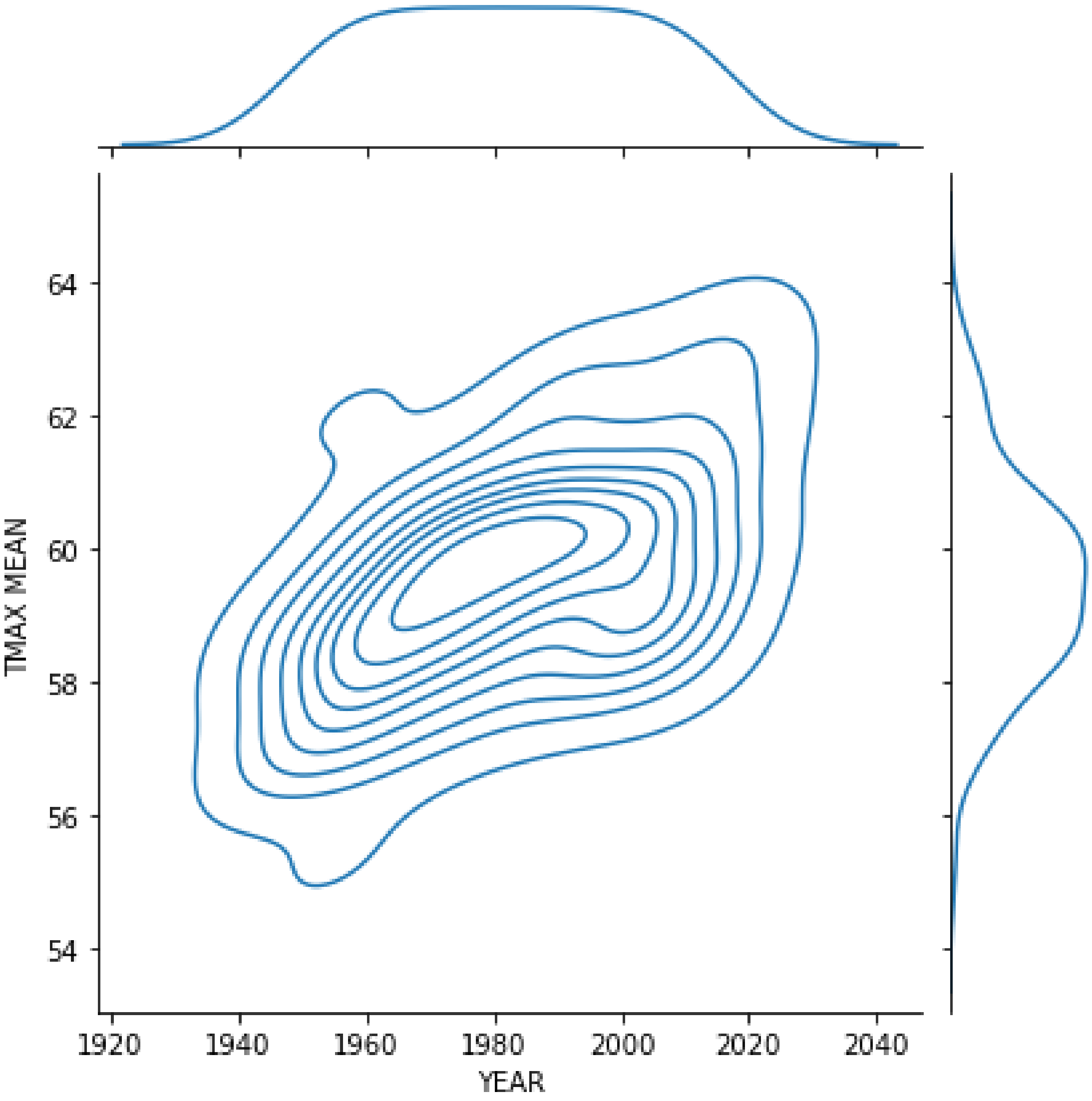

TMIN MEAN
48
47
46
45
44
43
42
41
1950
1960
1970
1980
1990
2000
2010
2020
YEAR



```
[60]: sns.jointplot(x='YEAR', y='TMIN MEAN', data=rain_per_year, kind='kde')
      sns.jointplot(x='YEAR', y='TMAX MEAN', data=rain_per_year, kind='kde')
```

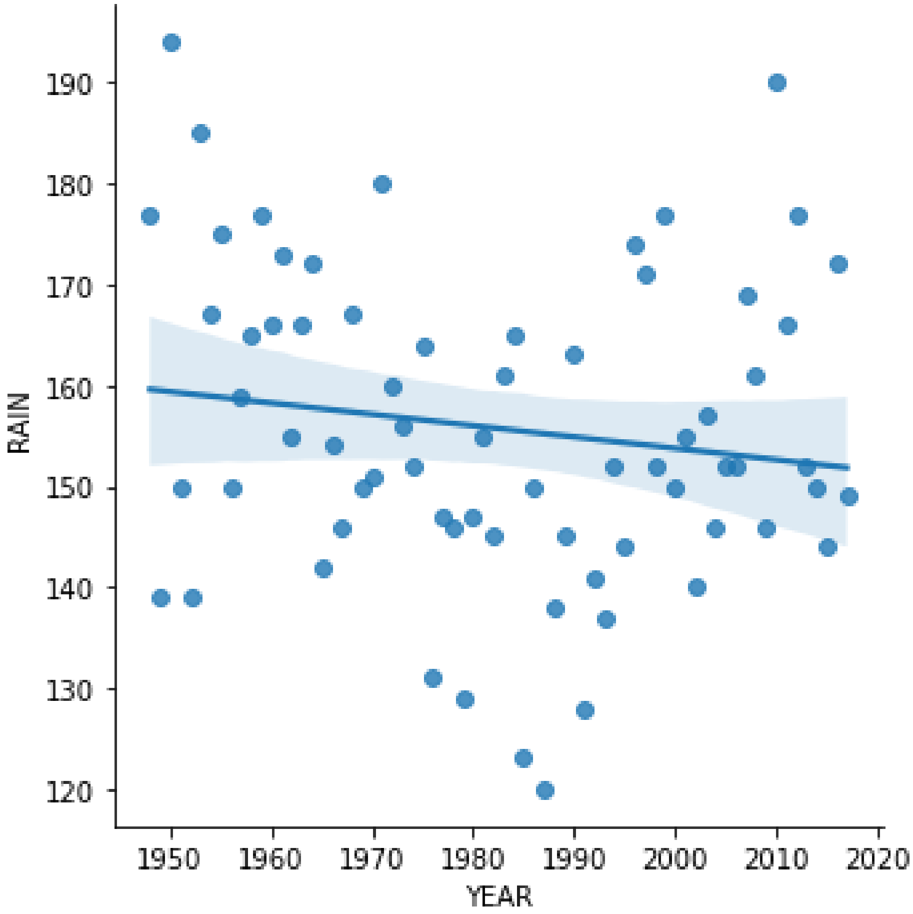
```
[60]: <seaborn.axisgrid.JointGrid at 0x2b7226bc580>
```


50
48
46
44
42
40
TMIN MEAN
1920
1940
1960
1980
2000
2020
2040
YEAR



[61]:
```
sns.lmplot(x='YEAR', y='RAIN', data=rain_per_year)
```

[61]:
```
<seaborn.axisgrid.FacetGrid at 0x2b722957340>
```



[62]:

```
import statsmodels.formula.api as sm
```

[63]:

```
result = sm.ols(formula='RAIN ~ YEAR',data=rain_per_year).fit()
print(result.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                   RAIN   R-squared:                       0.022
Model:                            OLS   Adj. R-squared:                  0.007
Method:                 Least Squares   F-statistic:                     1.520
Date:                Fri, 31 May 2024   Prob (F-statistic):              0.222
Time:                        18:34:13   Log-Likelihood:                -289.74
No. Observations:                  70   AIC:                             583.5
Df Residuals:                      68   BIC:                             588.0
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
```

```
------------------------------------------------------------------------------
Intercept    378.4702    180.659      2.095      0.040      17.970     738.970
YEAR          -0.1124      0.091     -1.233      0.222      -0.294       0.069
==============================================================================
Omnibus:                        0.144   Durbin-Watson:                   1.891
Prob(Omnibus):                  0.931   Jarque-Bera (JB):                0.196
Skew:                           0.102   Prob(JB):                        0.907
Kurtosis:                       2.839   Cond. No.                     1.95e+05
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 1.95e+05. This might indicate that there are
strong multicollinearity or other numerical problems.
```

[64]:

```
result = sm.ols(formula='RAIN ~ YEAR',data=rain_per_year).fit()
print(result.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                   RAIN   R-squared:                       0.022
Model:                            OLS   Adj. R-squared:                  0.007
Method:                 Least Squares   F-statistic:                     1.520
Date:                Fri, 31 May 2024   Prob (F-statistic):              0.222
Time:                        18:34:17   Log-Likelihood:                -289.74
No. Observations:                  70   AIC:                             583.5
Df Residuals:                      68   BIC:                             588.0
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept    378.4702    180.659      2.095      0.040      17.970     738.970
YEAR          -0.1124      0.091     -1.233      0.222      -0.294       0.069
==============================================================================
Omnibus:                        0.144   Durbin-Watson:                   1.891
Prob(Omnibus):                  0.931   Jarque-Bera (JB):                0.196
Skew:                           0.102   Prob(JB):                        0.907
Kurtosis:                       2.839   Cond. No.                     1.95e+05
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 1.95e+05. This might indicate that there are
strong multicollinearity or other numerical problems.
```

[65]:

```
result.pvalues
```

[65]:

```
Intercept    0.039904
YEAR         0.221791
dtype: float64
```

[ ]:

```

```