In [1]:

```
import pandas as pd
```

In [2]:

```
import numpy as np
```

In [22]:

```
import matplotlib.pyplot as plt
```

In [4]:

```
import seaborn as sns
```

In [5]:

```
df=pd.read_csv(r'C:\Users\Rutu\Documents\New folder\weather.csv')
```

In [6]:

```
df.head()
```

Out[6]:

	Year	Month	Day	High Temp (F)	Avg Temp (F)	Low Temp (F)	High Dew Point (F)	Avg Dew Point (F)	Low Dew Point (F)	High Humidity (%)	...	Low Sea Level Press (in)	High Visibility (mi)	Avg Visibility (mi)	Lo Visibilit (m
0	2008	1	1	40	34	28	37	31	22	92	...	29.56	10	8	
1	2008	1	2	33	22	10	28	18	-3	82	...	29.55	10	10	
2	2008	1	3	14	11	7	-3	-7	-9	60	...	30.22	10	10	1
3	2008	1	4	32	20	8	13	5	-8	63	...	30.37	10	10	1
4	2008	1	5	42	35	27	26	16	12	64	...	30.17	10	10	1

5 rows × 24 columns

In [7]:

```
df.tail()
```

Out[7]:

	Year	Month	Day	High Temp (F)	Avg Temp (F)	Low Temp (F)	High Dew Point (F)	Avg Dew Point (F)	Low Dew Point (F)	High Humidity (%)	...	Low Sea Level Press (in)	High Visibility (mi)	Avg Visibility (mi)	Visi
3744	2018	4	4	58	49	39	56	42	19	100	...	29.29	10	4	
3745	2018	4	5	43	37	30	21	9	4	48	...	29.65	10	10	
3746	2018	4	6	43	36	29	38	27	9	100	...	29.71	10	8	
3747	2018	4	7	47	41	35	38	26	16	92	...	29.69	10	10	
3748	2018	4	8	42	37	32	21	17	11	52	...	29.76	10	10	

5 rows × 24 columns

In [8]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3749 entries, 0 to 3748
Data columns (total 24 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Year                      3749 non-null   int64  
 1   Month                     3749 non-null   int64  
 2   Day                       3749 non-null   int64  
 3   High Temp (F)             3749 non-null   int64  
 4   Avg Temp (F)              3749 non-null   int64
 5   Low Temp (F)              3749 non-null   int64
 6   High Dew Point (F)        3749 non-null   int64
 7   Avg Dew Point (F)         3749 non-null   int64
 8   Low Dew Point (F)         3749 non-null   int64
 9   High Humidity (%)         3749 non-null   int64
 10  Avg Humidity (%)          3749 non-null   int64
 11  Low Humidity (%)          3749 non-null   int64
 12  High Sea Level Press (in)  3749 non-null   float64
 13  Avg Sea Level Press (in)   3749 non-null   float64
 14  Low Sea Level Press (in)   3749 non-null   float64
 15  High Visibility (mi)      3749 non-null   int64
 16  Avg Visibility (mi)       3749 non-null   int64
 17  Low Visibility (mi)       3749 non-null   int64
 18  High Wind (mph)           3749 non-null   int64
 19  Avg Wind (mph)            3749 non-null   int64
 20  High Wind Gust (mph)      3749 non-null   int64
 21  Snowfall (in)             3749 non-null   float64
 22  Precip (in)               3749 non-null   float64
 23  Events                    3749 non-null   object
dtypes: float64(5), int64(18), object(1)
memory usage: 703.1+ KB
```

In [11]:

```
df['Events'].dropna
```

Out[11]:

```
<bound method Series.dropna of 0       Both
1       Snow
2       None
3       None
4       None
        ...
3744    Rain
3745    None
3746    Both
3747    Rain
3748    None
Name: Events, Length: 3749, dtype: object>
```

In [13]:

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

Out[13]:

```
Year                         0
Month                        0
Day                          0
High Temp (F)                0
Avg Temp (F)                 0
Low Temp (F)                 0
High Dew Point (F)           0
Avg Dew Point (F)            0
Low Dew Point (F)            0
High Humidity (%)            0
Avg Humidity (%)             0
Low Humidity (%)             0
High Sea Level Press (in)    0
Avg Sea Level Press (in)     0
Low Sea Level Press (in)     0
High Visibility (mi)         0
Avg Visibility (mi)          0
Low Visibility (mi)          0
High Wind (mph)              0
Avg Wind (mph)               0
High Wind Gust (mph)         0
Snowfall (in)                0
Precip (in)                  0
Events                       0
dtype: int64
```

In [15]:

```
df.dropna(inplace=True)
```

In [17]:

```
df.shape
```

Out[17]:

```
(3749, 24)
```

In [18]:

```
df.describe()
```

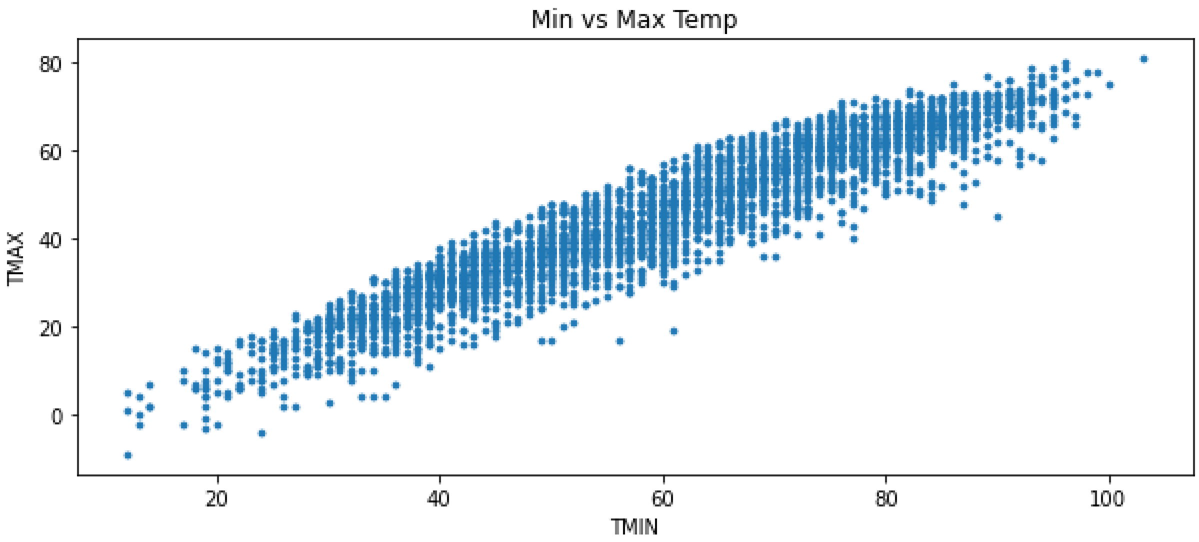
Out[18]:

	Year	Month	Day	High Temp (F)	Avg Temp (F)	Low Temp (F)	High Dew Point (F)	Avg Dew Point (F)
count	3749.000000	3749.000000	3749.000000	3749.000000	3749.000000	3749.000000	3749.000000	3749.000000
mean	2012.640437	6.410243	15.699653	59.537477	52.370766	44.706055	45.703654	39.735663
std	2.966161	3.477825	8.807769	18.352603	17.361271	16.835002	17.590199	18.614174
min	2008.000000	1.000000	1.000000	12.000000	2.000000	-9.000000	-13.000000	-18.000000
25%	2010.000000	3.000000	8.000000	44.000000	39.000000	32.000000	33.000000	26.000000
50%	2013.000000	6.000000	16.000000	60.000000	53.000000	45.000000	47.000000	41.000000
75%	2015.000000	9.000000	23.000000	75.000000	67.000000	59.000000	61.000000	55.000000
max	2018.000000	12.000000	31.000000	103.000000	92.000000	81.000000	78.000000	74.000000

8 rows × 23 columns

In [43]:

```
fig,(ax1) = plt.subplots(1, figsize = (10,4))
x=df['High Temp (F)']
y=df['Low Temp (F)']
ax1.scatter(x,y,s=8)
plt.title ('Min vs Max Temp')
plt.xlabel('TMIN')
plt.ylabel('TMAX')
plt.show()
```



In [45]:

```
X = df['High Temp (F)'].values.reshape(-1,1).astype('float32')
y = df['Low Temp (F)'].values.reshape(-1,1).astype('float32')
```

In [46]:

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [47]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

In [48]:

```
h = LinearRegression()
h.fit(X_train,y_train)
print(h.intercept_) # to retrieve theta_0
print(h.coef_) # to retrieve theta_1
```

```
[-6.8372574]
[[0.8671528]]
```

In [50]:

```
y_pred = h.predict(X_test)
compare = pd.DataFrame({'Actual': y_test.flatten(), 'Predicted': y_pred.flatten()})
compare
```

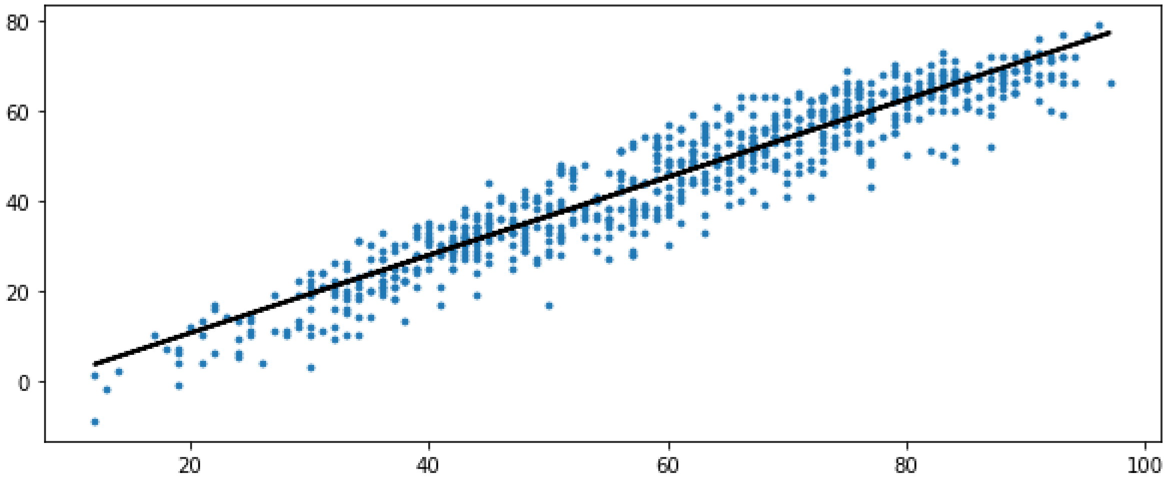
Out[50]:

	Actual	Predicted
0	12.0	19.177326
1	58.0	57.332054
2	48.0	48.660522
3	60.0	59.933510
4	40.0	33.051773
...	...	...
745	34.0	28.716007
746	56.0	59.066353
747	33.0	31.317467
748	49.0	46.059063
749	51.0	64.269272

750 rows × 2 columns

In [52]:

```
fig,(ax1) = plt.subplots(1, figsize = (10,4))
ax1.scatter (X_test, y_test, s = 8)
plt.plot(X_test,y_pred, color = 'black', linewidth = 2)
plt.show()
```


80
60
40
20
0
20
40
60
80
100