In [ ]:
```
#1.Use pandas to get some insights into the data
```

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn import preprocessing
from sklearn.preprocessing import MinMaxScaler
from sklearn import metrics
import matplotlib as plt
%matplotlib inline
```

In [3]:
```
url = "https://raw.githubusercontent.com/edyoda/data-science-complete-tutorial/master/Data/house_rental_data.csv.txt"
df1 = pd.read_csv(url)
df1.head()
```

Out[3]:

	Unnamed: 0	Sqft	Floor	TotalFloor	Bedroom	Living.Room	Bathroom	Price
0	1	1177.698	2	7	2	2	2	62000
1	2	2134.800	5	7	4	2	2	78000
2	3	1138.560	5	7	2	2	1	58000
3	4	1458.780	2	7	3	2	2	45000
4	5	967.776	11	14	3	2	2	45000

In [6]:
```
df1=df1.drop(["Unnamed: 0"], axis=1)
df1.head()
```

Out[6]:

	Sqft	Floor	TotalFloor	Bedroom	Living.Room	Bathroom	Price
0	1177.698	2	7	2	2	2	62000
1	2134.800	5	7	4	2	2	78000
2	1138.560	5	7	2	2	1	58000
3	1458.780	2	7	3	2	2	45000
4	967.776	11	14	3	2	2	45000

In [7]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 645 entries, 0 to 644
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Sqft         645 non-null    float64
 1   Floor        645 non-null    int64
 2   TotalFloor   645 non-null    int64
 3   Bedroom      645 non-null    int64
 4   Living.Room  645 non-null    int64
 5   Bathroom     645 non-null    int64
 6   Price        645 non-null    int64
dtypes: float64(1), int64(6)
memory usage: 35.4 KB
```

In [12]:
```
df1=df1.rename(columns = {"Living.Room" : "Living Room","TotalFloor":"Total Floor"})
df1.head()
```

Out[12]:

	Sqft	Floor	Total Floor	Bedroom	Living Room	Bathroom	Price
0	1177.698	2	7	2	2	2	62000
1	2134.800	5	7	4	2	2	78000
2	1138.560	5	7	2	2	1	58000
3	1458.780	2	7	3	2	2	45000
4	967.776	11	14	3	2	2	45000

In [13]:
```
df1.describe()
```

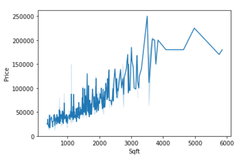
Out[13]:

	Sqft	Floor	Total Floor	Bedroom	Living Room	Bathroom	Price
count	645.000000	645.000000	645.000000	645.000000	645.000000	645.000000	645.000000
mean	1527.656260	5.939535	10.855814	2.837209	1.813953	1.810853	61986.823256
std	767.386531	3.884721	4.996208	1.010740	0.462364	0.683574	35635.091007
min	359.358000	1.000000	1.000000	1.000000	0.000000	0.000000	6100.000000
25%	925.080000	3.000000	7.000000	2.000000	2.000000	1.000000	39000.000000
50%	1423.200000	5.000000	12.000000	3.000000	2.000000	2.000000	50000.000000
75%	1892.856000	8.000000	14.000000	4.000000	2.000000	2.000000	75000.000000
max	5856.468000	22.000000	38.000000	7.000000	4.000000	5.000000	250000.000000

In [ ]:
```
#2.Show some interesting visualization of the data
```

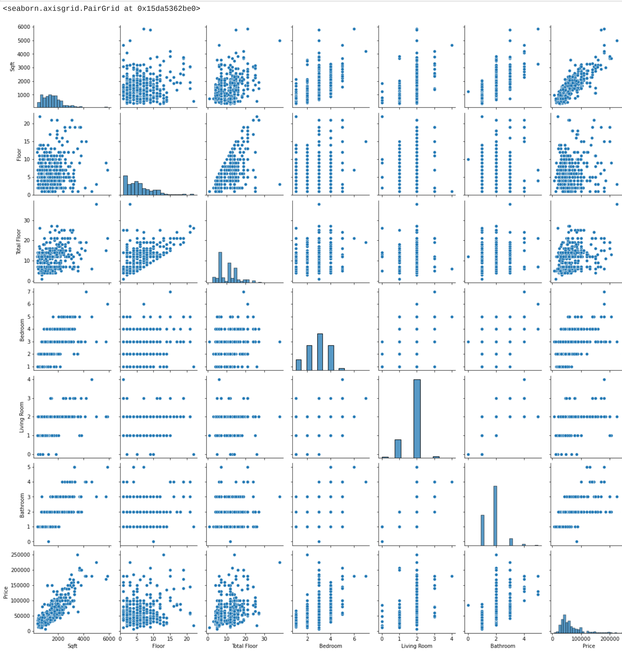
In [15]:
```
sns.lineplot(x = df1["Sqft"], y = df1["Price"])
```

Out[15]:
```
<AxesSubplot:xlabel='Sqft', ylabel='Price'>
```



In [17]:
```
sns.pairplot(data = df1,palette = 'coolwarm')
```

Out[17]:
```
<seaborn.axisgrid.PairGrid at 0x15da5362be0>
```



In [ ]:
```
#3.Manage data for training & testing
```

In [19]:
```
from scipy import stats
x1 = np.abs(stats.zscore(df1))
print(x1)
```

```
         Sqft     Floor  Total Floor   Bedroom  Living Room  Bathroom  \
0    0.456393  1.014897     0.772347  0.828956     0.402694  0.276918
1    0.791798  0.242042     0.772347  1.151328     0.402694  0.276918
2    0.507434  0.242042     0.772347  0.828956     0.402694  1.187117
3    0.089824  1.014897     0.772347  0.161186     0.402694  0.276918
4    0.730160  1.303669     0.629803  0.161186     0.402694  0.276918
..        ...       ...          ...       ...          ...       ...
640  0.219747  0.273195     0.830110  0.161186     0.402694  0.276918
641  1.500419  0.499660     0.171426  1.819099     1.761785  1.187117
642  1.027127  1.818907     0.629803  1.819099     1.761785  1.187117
643  0.224984  0.757279     0.629803  0.161186     0.402694  0.276918
644  0.196547  0.015577     0.772347  0.161186     0.402694  1.187117

        Price
0    0.000370
1    0.449714
2    0.111966
3    0.477058
4    0.477058
..        ...
640  0.477058
641  1.044355
642  0.477058
643  0.084622
644  0.729814

[645 rows x 7 columns]
```

In [21]:
```
from sklearn.model_selection import train_test_split
X = df1.drop(labels = ['Price'] , axis = 1)
y = df1['Price']
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.2)
```

In [22]:
```
scaler = MinMaxScaler(feature_range = (0,1))
X_train_scaled = scaler.fit_transform(X_train)
X_train = pd.DataFrame(X_train_scaled)

X_test_scaled = scaler.fit_transform(X_test)
X_test = pd.DataFrame(X_test_scaled)
```

In [25]:
```
from sklearn import neighbors
from sklearn.metrics import mean_squared_error
from math import sqrt
from sklearn.neighbors import KNeighborsClassifier
rmse = []
for k in range(1,21):
    knn = neighbors.KNeighborsRegressor(n_neighbors = k)
    knn.fit(X_train, y_train)
    y_pred = knn.predict(X_test)
    err = sqrt(mean_squared_error(y_test,y_pred))
    rmse.append(err)

    print('For k =',k,'RMSE is',err)
```

```
For k = 1 RMSE is 29066.32197181116
For k = 2 RMSE is 27304.537259535537
For k = 3 RMSE is 26189.86948984126
For k = 4 RMSE is 23756.48259568918
For k = 5 RMSE is 23423.75409797009
For k = 6 RMSE is 22118.15439471829
For k = 7 RMSE is 21387.936715551266
For k = 8 RMSE is 21389.431134683848
For k = 9 RMSE is 21978.335174720345
For k = 10 RMSE is 22233.354248362488
For k = 11 RMSE is 22200.593682093313
For k = 12 RMSE is 22119.499252603775
For k = 13 RMSE is 22385.215247190343
For k = 14 RMSE is 22486.11568301487
For k = 15 RMSE is 22582.125990066765
For k = 16 RMSE is 22423.500916278146
For k = 17 RMSE is 22407.072876214013
For k = 18 RMSE is 22180.855996697235
For k = 19 RMSE is 22195.321317544767
For k = 20 RMSE is 22188.890021906976
```

In [30]:
```
print("Best value for k is : ",rmse.index(min(rmse))+1)
```

```
best value for k is :  7
```

In [ ]:
```

```