Image Credit: Tiia Monto[1]

DSTA Coursework 1: Dataset dimensionality analysis

Montel Moore | MSc Data Science | 08/02/2021

PHASE 1: INTRODUCTION

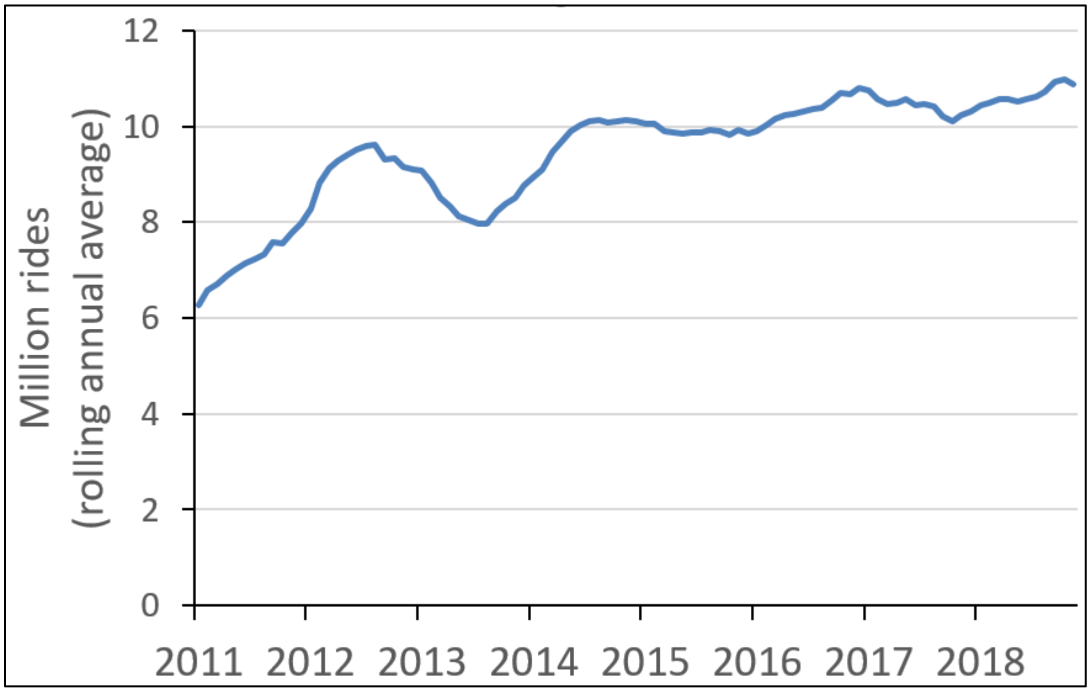


Figure 1: Bike sharing in London over time.[2]

Active travel is becoming increasingly relevant as we shift to a more health and planet conscious world. In 2010, a cycle hire scheme was implemented in London, and it saw significant uptake over the decade.

This report analyses a Kaggle dataset built around the hourly bike hires in London, its data management side and the dimensionalities that it involves.

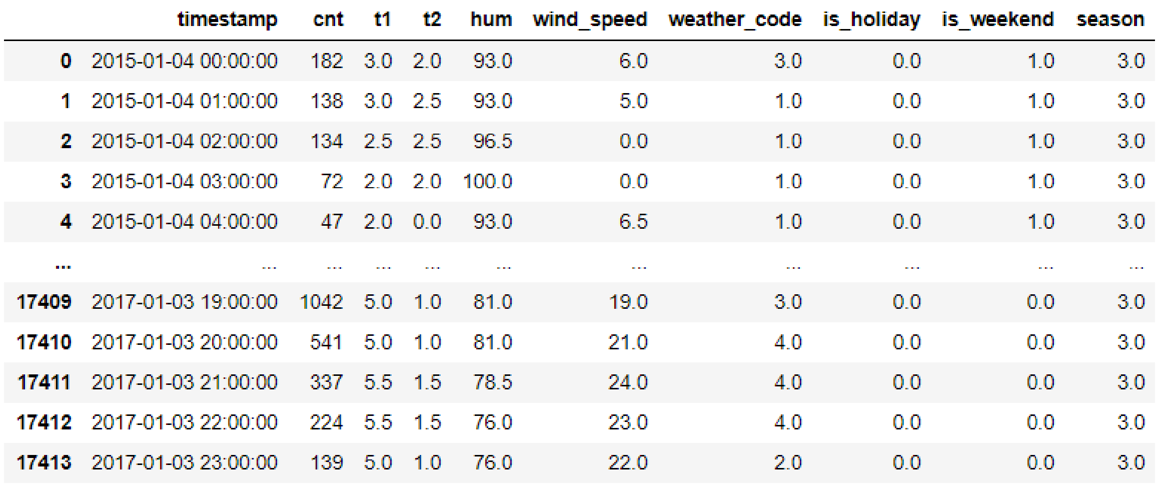
Phase 2

2.1 DESCRIPTION AND SOURCE OF DIMENSIONS

This dataset is a time series, focused on the first two dimensions: *timestamp* and count (*cnt*). Each row is an instance of data corresponding to the timestamp, and the dataset consists of hourly timestamps from the dates of 01/04/15 – 01/03/17. The count is the number of bike rentals or "shares", of London's Santander cycles per hour, collected from Transport for London [3].

The dataset also contains weather data dimensions (temperature (*t1*), temperature feels like (*t2*), humidity percentage (*hum*), *wind_speed* and *weather_code*) from freemeteo.com [4]. Bank holidays were identified from the UK government website [5], and this Boolean dimension *is_holiday* was included.

Finally, *season* and the Boolean dimension *is_weekend* were added. *Season* and *is_weekend* were likely to have been inferred based on the compiler's knowledge of the UK's seasonal patterns and the Georgian calendar. Figure 2 shows a preview of the dataset as a pandas DataFrame.



	timestamp	cnt	t1	t2	hum	wind_speed	weather_code	is_holiday	is_weekend	season
0	2015-01-04 00:00:00	182	3.0	2.0	93.0	6.0	3.0	0.0	1.0	3.0
1	2015-01-04 01:00:00	138	3.0	2.5	93.0	5.0	1.0	0.0	1.0	3.0
2	2015-01-04 02:00:00	134	2.5	2.5	96.5	0.0	1.0	0.0	1.0	3.0
3	2015-01-04 03:00:00	72	2.0	2.0	100.0	0.0	1.0	0.0	1.0	3.0
4	2015-01-04 04:00:00	47	2.0	0.0	93.0	6.5	1.0	0.0	1.0	3.0
...	...	...	...	...	...	...	...	...	...	...
17409	2017-01-03 19:00:00	1042	5.0	1.0	81.0	19.0	3.0	0.0	0.0	3.0
17410	2017-01-03 20:00:00	541	5.0	1.0	81.0	21.0	4.0	0.0	0.0	3.0
17411	2017-01-03 21:00:00	337	5.5	1.5	78.5	24.0	4.0	0.0	0.0	3.0
17412	2017-01-03 22:00:00	224	5.5	1.5	76.0	23.0	4.0	0.0	0.0	3.0
17413	2017-01-03 23:00:00	139	5.0	1.0	76.0	22.0	2.0	0.0	0.0	3.0

Figure 2: First five and last five rows from original Kaggle dataset

The Kaggle challenge is "to predict the future bike shares". As of 08/02/2021, 38 notebooks have been submitted for predicting the future demand of bike shares.

Phase 3

Dimensional analysis: write down the main aggregate measures of the dataset: number of data points, number of dimensions. Select a small number of dimensions that you consider the key to understanding how data is distributed. Describe and comment those dimensions (e.g., range of the dimension, quality of the data, possible data quality/integrity issues) in your essay.

3.1 DATA CLEANING/PRE-PROCESSING

The original dataset contains two years of sequential hourly data (from 04/01/2015 to 01/03/2017), organised in 17414 rows and 10 columns/dimensions as described in Phase 2. Pre-processing techniques outlined below were performed in order to make the provide more information and to make the dataset suitable for statistical modelling and analysis.

3.1.1 Addition of variables

The *Timestamp* variable was further discretised into variables representing the *day-of-week* and *hour* (out of 23). Another dimension (*observation*) was also produced. *Observation* is the number of hours from the starting time (e.g. 04/01/2015 00:00:00 = 0, 04/01/2015 01:00:00 = 1); this makes it convenient to model the effect of time on bike shares.

The *weather_code*, *season, day-of-week* and *hour* variables were one-hot dummy encoded to enable a correlation analysis.

An additional dimension, *precipitation(mm)* was added to the dataset. This was deemed necessary because rainfall can affect ridership [6]. The "weather code" categories 7 and 10 describe rainfall but provide ambiguity regarding the level of rainfall. The precipitation data was gathered from CEDA.

3.1.2 Missing data

The dataset was expected to have 17544 rows worth of data (731 days * 24 hours per day), however it only contained 17414 rows of data. A list of the full 17544 hours was generated separately from the dataset, and the dataset was compared against this list in order to select a sample of missing hours. The "missing hours" were searched for in the original weather and cycle count data sources; it was found that the weather data had some missing observations, which were the likely cause for the "missing hours".

Figure 3 shows the number of data points per hourly interval. Some hours have more data points than others because the dataset author chose to omit the timestamps with missing weather data. Due to the large volume of data and the relatively even distribution of hours, it is unlikely that this will have a significant biasing effect on predictive models built from the data.

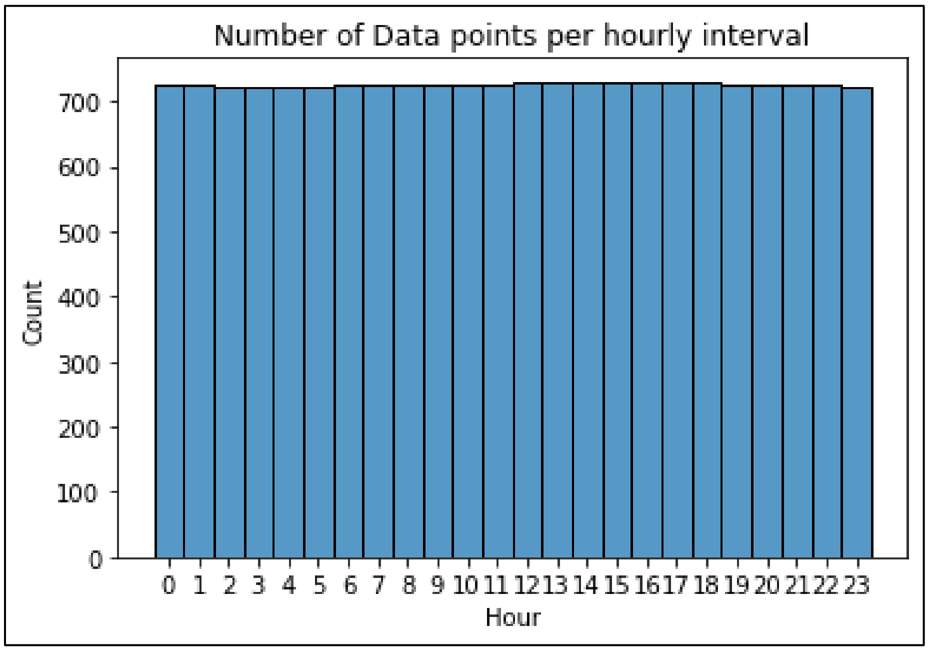
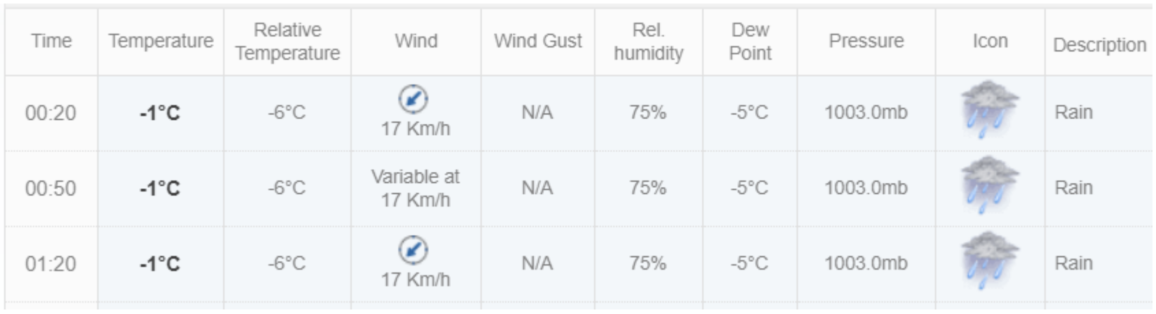


Figure 3: Total number of data points per hourly interval

3.1.3 Asynchronous time intervals

The historical weather data from freemeteo.com is recorded at the 20th and 50th minutes per hour, as shown in the "time" column of Figure 4. This does not align with the timestamps in the dataset, which were recorded every 0th minute of the hour. This means that the data set author interpolated the weather data match to every 0th minute of the hour. This was likely done by either taking the 20th or 50th minute data alone, or by averaging both values for each hour. Weather is unlikely to change dramatically within the space of 30 minutes, therefore this discrepancy is unlikely to have a significant biasing effect on predictive models built from the data.



Time	Temperature	Relative Temperature	Wind	Wind Gust	Rel. humidity	Dew Point	Pressure	Icon	Description
00:20	**-1°C**	-6°C	17 Km/h	N/A	75%	-5°C	1003.0mb		Rain
00:50	**-1°C**	-6°C	Variable at 17 Km/h	N/A	75%	-5°C	1003.0mb		Rain
01:20	**-1°C**	-6°C	17 Km/h	N/A	75%	-5°C	1003.0mb		Rain

Figure 4: Sample of historical weather data captured from freemeteo.com

3.2 Analysis of critical features

3.2.1 Feature importance and correlation scoring

Two methods were used to evaluate the importance each feature with respect to the dependent variable, bike shares: Pearson's correlation coefficient and MSE-based feature importance's extracted from a random-trees regressor model. The Pearson's correlation coefficient is the linear correlation between the dependent variable and the independent variable. The random-forest regressor was used in addition to Pearson's because the Pearson's correlation coefficient is weak at modelling non-linear relationships. In addition, random-forests models remove some multi-collinearity, while Pearson's does not take it into account. There was a high degree of collinearity between independent predictors t1 and t2 (r = 0.988344), therefore t2 was removed from the feature analysis. Though the random-forest model takes collinearity into account, it still ranked t1 and t2 similarly because there each individual tree is would place a roughly equal weighting to t1 or t2 when deciding to split on a variable.

Table 1 and table 2 show the top three predictive variables based on correlation coefficients and random forest feature importance's. These were calculated for two datasets: one with continuous variables only, and another with mixed and continuous variables. The continuous dataset variables will be used for PCA, and the mixed variables dataset will be used for a (FAMD) factor analysis of mixed data.

Top 3 Correlation coefficients			
Continuous variables only		Mixed variables	
hum	-0.463	hum	-0.463
t1	0.389	t1	0.389
hour	0.324	08:00	0.334

Table 2: Top 3 correlation coefficients for the dataset with continuous variables only, and both mixed and continuous variables.

Top 3 Random forest feature imporances			
Continuous variables only model		Mixed variables model	
hour	0.623	hour	0.552
t1	0.125	is_weekend	0.150
observation #	0.100	t1	0.086

Table 1: Top 3 features ranked on mse-based feature importance, for the dataset with continuous variables only, and mixed and continuous variables.

3.2.2 Descriptive statistics

The following section will detail the range and quality of the most important dimensions and the appended dimension, *precipitation(mm)*.

Hour

The hour of the day is a strong predictor of the bike shares, being the most important variable for the random forest models. It appears to have a x^7 polynomial relationship to the total bike shares, and a general trend of more bike users in the later hours of the day, though there is a morning peak at 8:00 am, which was the most important hour category when hours were encoded as dummy variable. As mentioned in section 3.1.2, there is a roughly even spread of data points for each hour, which makes hour a reliable predictor.

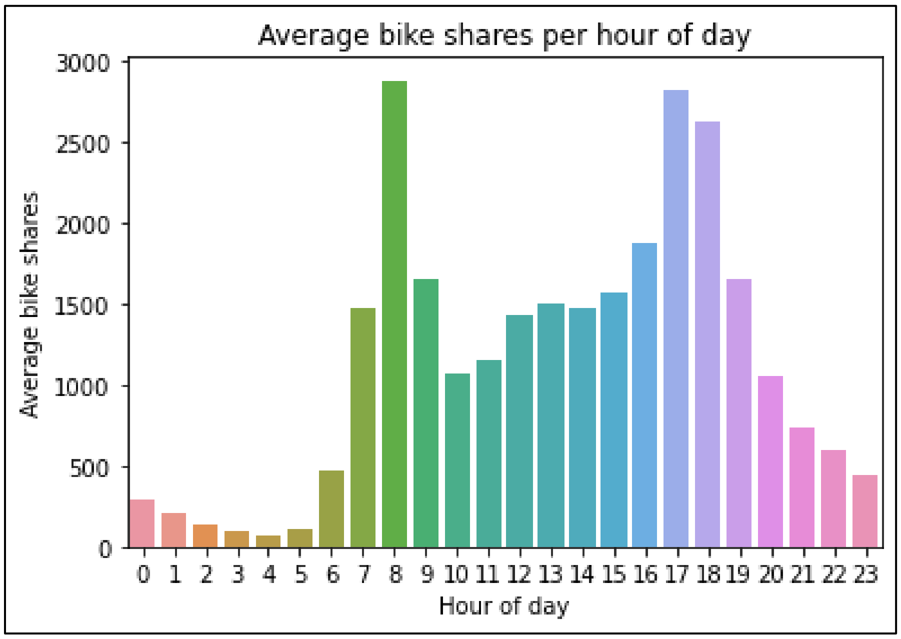


Figure 5: Average bike shares per hour of day

Temperature

Temperature had a significant positive correlation with the number of shares, which is to be expected because cyclists are less likely to cycle in cold weather[7]. This is especially evident once the temperature exceeds 25 degrees, where it becomes unlikely for there to be less than 1000 cycle hires per hour. Temperatures ranged from -1.5 to 34, however temperatures over 28 were relatively uncommon.

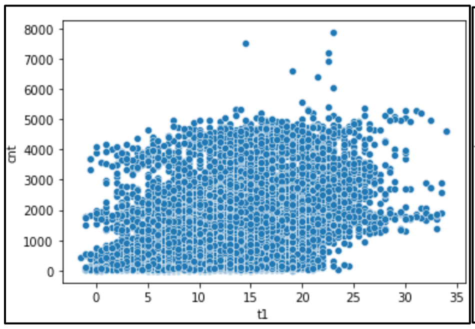


Figure 8a: Scatter plot of temperature (t1) against bike shares per hour

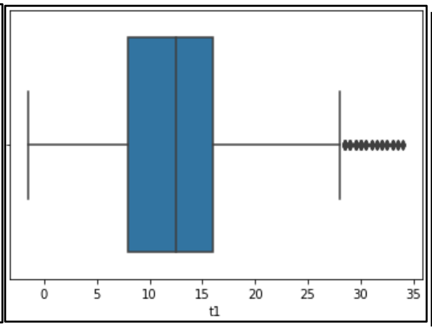


Figure 8b: Distribution of temperature

	temperature
count	17414
mean	12.468091
std	5.571818
min	-1.5
25%	8
50%	12.5
75%	16
max	34

Figure 8c: Temperature descriptive statistics

Humidity

Humidity was the most important variable according to the Pearson's correlation coefficient, however it was ranked as a less important feature for the random forest model. The correlation coefficient was negative which means that cycles were less likely to be hired as the humidity increased towards 100%. Humidity ranged from 20.5% to 100%, however humidity values below 31% were relatively uncommon.

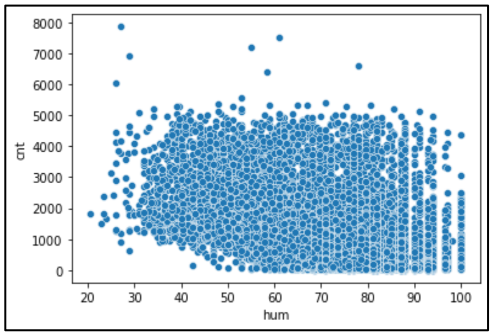


Figure 11a: Scatter plot of humidity (hum) against bike shares per hour

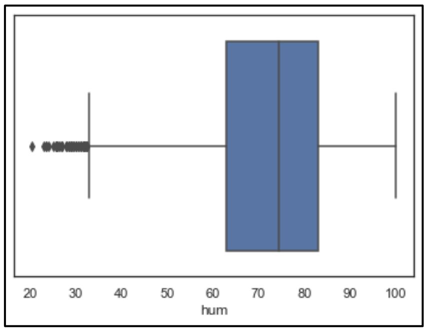
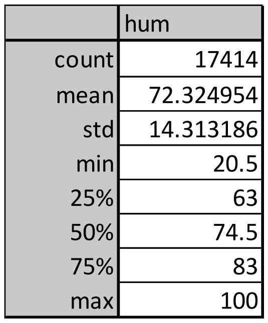


Figure 11b: Distribution of humidity (hum)



	hum
count	17414
mean	72.324954
std	14.313186
min	20.5
25%	63
50%	74.5
75%	83
max	100

Figure 11c: Humidity descriptive statistics

Timestamp/observation number

The timestamp/observation number has a small, but significant and positive correlation with the number of bike shares ($p < 0.05$, $r = 0.04$). From Figure 12 it is evident that the number of bike shares follows a yearly cyclical pattern, with the number of bike shares being slightly higher in 2016 than in 2015.

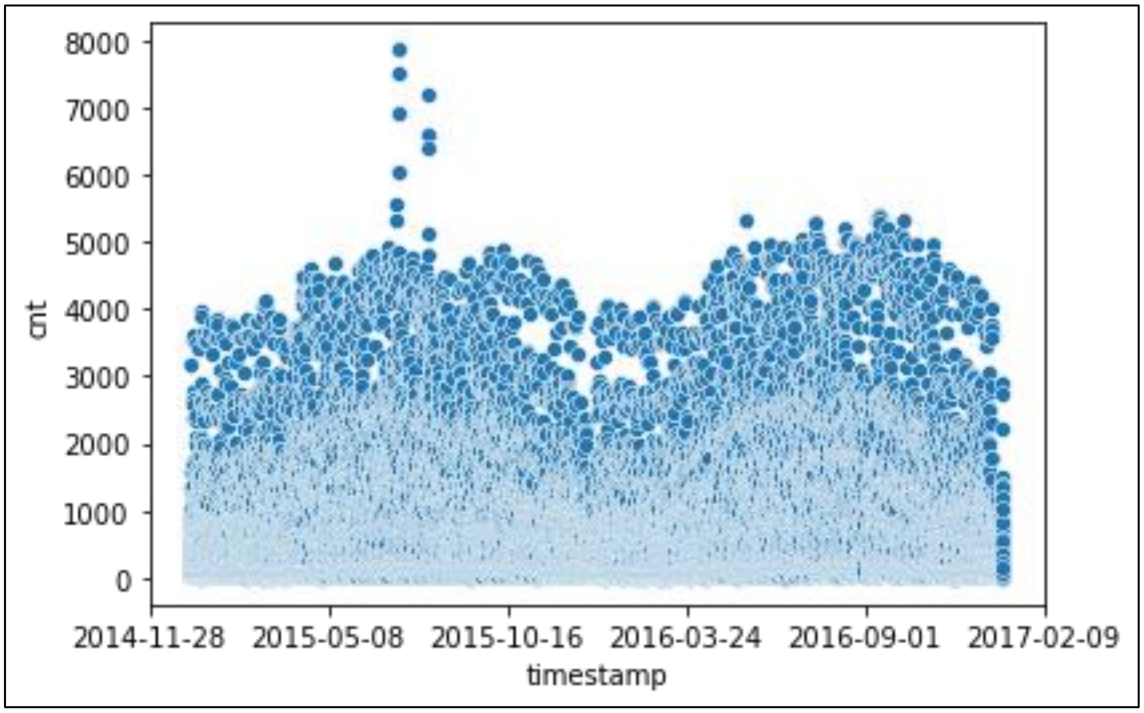


Figure 12: Scatter plot of timestamp against bike shares per hour

Weekend/Weekday

Is_weekend had a significant negative correlation with the number of bike shares. This is likely due to a higher commuter volume on weekdays vs weekends. Figure 13 shows the average bike shares for the weekday vs weekend.

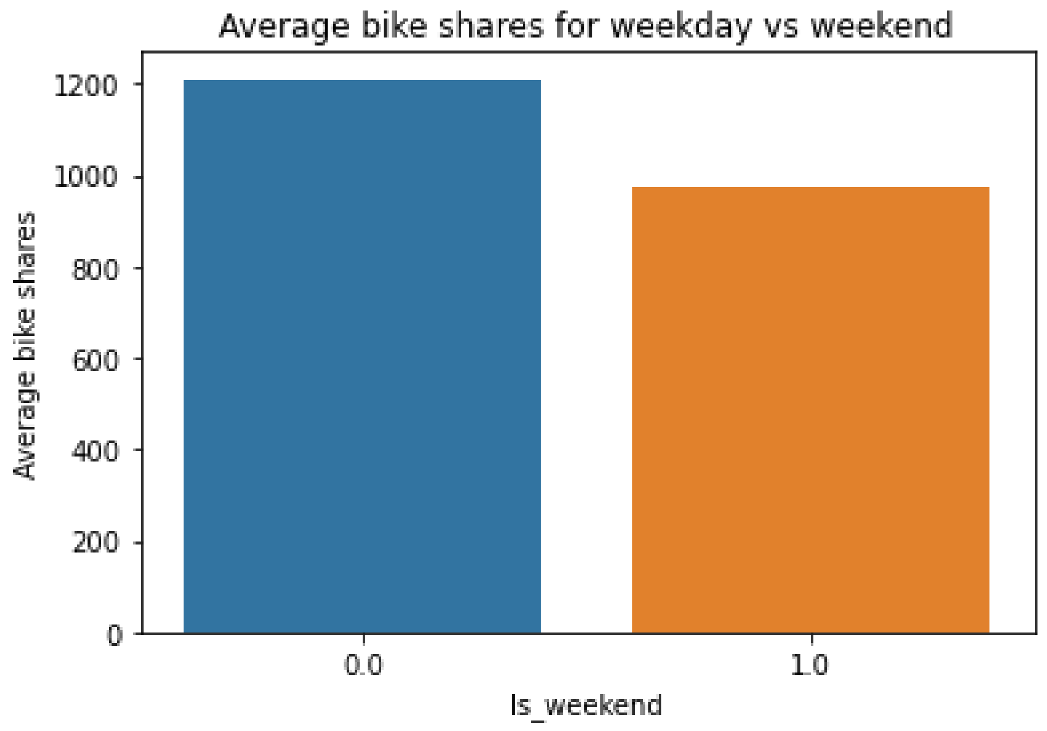


Figure 13: Average hourly bike shares for weekend days vs weekdays. 0.0 represents weekdays, 1.0 represents weekends

Precipitation/rainfall

The appended variable (precipitation) did not appear to have a significant positive or negative correlation with the number of bike shares per hour. Though Figure 16 shows a negative correlation between precipitation and bike shares, there were not enough recordings of precipitation > 0 to make a significant difference. In addition, there appears to be a non-linear relationship between precipitation and bike shares which cannot be captured with the Pearson's correlation coefficient.

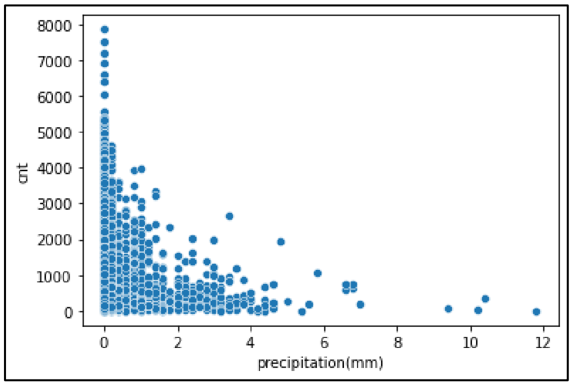


Figure 15a: Scatter plot of precipitation against bike shares per hour

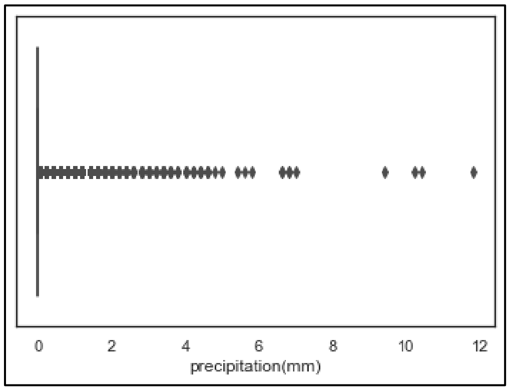


Figure 15b: Distribution of precipitation.

SUMMARY/FUTURE GOALS

The Kaggle "London" bike sharing dataset is a reliable and usable dataset which involves carefully picked dimensions. In part two of this report, Principal Component Analysis (PCA) will be investigated to reduce the dimensions into a set of eigenvectors which can be used as a predictor of the total number of bike shares. A feasibility study will be undertaken on a 2nd dimensionality reduction method, Factor Analysis of Mixed Data (FAMD).

REFERENCES

1. Monto, T., 2018. *Santander bicycles on the Exhibition Road in London.*. [image] Available at: <https://commons.wikimedia.org/wiki/File:Santander_Cycles.jpg> [Accessed 19 February 2021].
2. Number of hires of Santander bikes from June 2011 to May 2019 (annual rolling average).. [image] Available at: <https://commons.wikimedia.org/wiki/File:Santander_bikes.png> [Accessed 19 February 2021].
3. *cycling.data.tfl.gov.uk*. [online] Available at: <https://cycling.data.tfl.gov.uk/> [Accessed 19 February 2021].
4. Freemeteo.co.uk. n.d. *The Weather*. [online] Available at: <https://freemeteo.co.uk/> [Accessed 19 February 2021].
5. GOV.UK. n.d. *UK bank holidays*. [online] Available at: <https://www.gov.uk/bank-holidays> [Accessed 19 February 2021].
6. Ahmed, Farhana & Rose, G. & Jacob, C.. (2010). *Impact of weather on commuter cyclist behaviour and implications for climate change adaptation*. ATRF 2010: 33rd Australasian Transport Research Forum.
7. Tin Tin, S., Woodward, A., Robinson, E. and Ameratunga, S., 2012. Temporal, seasonal and weather effects on cycle volume: an ecological study. *Environmental Health*, 11(1).

APPENDIX

Code for DSTA CW1

February 20, 2021

1 Phase 1

We have chosen to select the "London Bike Sharing" dataset

[1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.stats import ttest_ind
import statsmodels.api as sm
import matplotlib.ticker as plticker
import datetime
import locale
locale.setlocale(locale.LC_ALL, '')
```

[1]:
```
'English_United Kingdom.1252'
```

[2]:
```
bike_shares_data = pd.read_csv('data\london_merged.csv', sep = ",")
```

[3]:
```
bike_shares_data
```

[3]:
```
                 timestamp   cnt   t1   t2    hum  wind_speed  weather_code  \
0      2015-01-04 00:00:00   182  3.0  2.0   93.0         6.0           3.0
1      2015-01-04 01:00:00   138  3.0  2.5   93.0         5.0           1.0
2      2015-01-04 02:00:00   134  2.5  2.5   96.5         0.0           1.0
3      2015-01-04 03:00:00    72  2.0  2.0  100.0         0.0           1.0
4      2015-01-04 04:00:00    47  2.0  0.0   93.0         6.5           1.0
…                        …     …    …    …      …           …             …
17409  2017-01-03 19:00:00  1042  5.0  1.0   81.0        19.0           3.0
17410  2017-01-03 20:00:00   541  5.0  1.0   81.0        21.0           4.0
17411  2017-01-03 21:00:00   337  5.5  1.5   78.5        24.0           4.0
17412  2017-01-03 22:00:00   224  5.5  1.5   76.0        23.0           4.0
17413  2017-01-03 23:00:00   139  5.0  1.0   76.0        22.0           2.0

       is_holiday  is_weekend  season
0             0.0         1.0     3.0
1             0.0         1.0     3.0
2             0.0         1.0     3.0
```

```
3              0.0          1.0     3.0
4              0.0          1.0     3.0
…            …          …      …
17409          0.0          0.0     3.0
17410          0.0          0.0     3.0
17411          0.0          0.0     3.0
17412          0.0          0.0     3.0
17413          0.0          0.0     3.0

[17414 rows x 10 columns]
```

[4]:
```
bike_shares_data.reset_index(level=0, inplace=True)
```

[5]:
```
bike_shares_data = bike_shares_data.rename({"index":"observation"}, axis = 1)
```

[6]:
```
bike_shares_data
```

[6]:
```
       observation           timestamp   cnt   t1   t2    hum  wind_speed  \
0                0 2015-01-04 00:00:00   182  3.0  2.0   93.0         6.0
1                1 2015-01-04 01:00:00   138  3.0  2.5   93.0         5.0
2                2 2015-01-04 02:00:00   134  2.5  2.5   96.5         0.0
3                3 2015-01-04 03:00:00    72  2.0  2.0  100.0         0.0
4                4 2015-01-04 04:00:00    47  2.0  0.0   93.0         6.5
…              …                   …    …    …   …     …
17409        17409 2017-01-03 19:00:00  1042  5.0  1.0   81.0        19.0
17410        17410 2017-01-03 20:00:00   541  5.0  1.0   81.0        21.0
17411        17411 2017-01-03 21:00:00   337  5.5  1.5   78.5        24.0
17412        17412 2017-01-03 22:00:00   224  5.5  1.5   76.0        23.0
17413        17413 2017-01-03 23:00:00   139  5.0  1.0   76.0        22.0

       weather_code  is_holiday  is_weekend  season
0               3.0         0.0         1.0     3.0
1               1.0         0.0         1.0     3.0
2               1.0         0.0         1.0     3.0
3               1.0         0.0         1.0     3.0
4               1.0         0.0         1.0     3.0
…             …          …          …      …
17409           3.0         0.0         0.0     3.0
17410           4.0         0.0         0.0     3.0
17411           4.0         0.0         0.0     3.0
17412           4.0         0.0         0.0     3.0
17413           2.0         0.0         0.0     3.0

[17414 rows x 11 columns]
```

[7]:
```
bike_shares_data['timestamp'] = pd.to_datetime(bike_shares_data['timestamp'])
```

```
[8]: bike_shares_data["season"].replace(to_replace = 0.0, value = "spring", inplace␣
      ↪= True)
     bike_shares_data["season"].replace(to_replace = 1.0, value = "summer", inplace␣
      ↪= True)
     bike_shares_data["season"].replace(to_replace = 2.0, value = "fall", inplace =␣
      ↪True)
     bike_shares_data["season"].replace(to_replace = 3.0, value = "winter", inplace␣
      ↪= True)
```

```
[9]: l = bike_shares_data.shape[0]
     bike_shares_data.insert(2,"hour", [datetime.datetime.
      ↪strftime(bike_shares_data["timestamp"][i], '%H') for i in range(l)])
     bike_shares_data.insert(2, "day of week", [datetime.datetime.
      ↪strftime(bike_shares_data["timestamp"][i], '%A') for i in range(l)])
     bike_shares_data["hour"] = bike_shares_data["hour"].astype("int64")
```

```
[10]: rainfall_data = pd.read_csv("data\Rainfall_2015.csv", sep = ",").append(
          pd.read_csv("data\Rainfall_2016.csv", sep = ",").append(pd.
       ↪read_csv("data\Rainfall_2017.csv", sep = ","),ignore_index =␣
       ↪True),ignore_index = True)
      rainfall_data
```

```
[10]:             ob_end_time  prcp_amt
      0      04/01/2015 00:00       0.0
      1      04/01/2015 01:00       0.0
      2      04/01/2015 02:00       0.0
      3      04/01/2015 03:00       0.0
      4      04/01/2015 04:00       0.0
      …                   …        …
      17537  03/01/2017 19:00       0.0
      17538  03/01/2017 20:00       0.0
      17539  03/01/2017 21:00       0.0
      17540  03/01/2017 22:00       0.0
      17541  03/01/2017 23:00       0.0

      [17542 rows x 2 columns]
```

```
[11]: for i in range(rainfall_data.shape[0]):
          rainfall_data.iloc[i,0] = datetime.datetime.strptime(rainfall_data.
       ↪iloc[i,0], "%d/%m/%Y %H:%M")
```

```
[12]: rainfall_data = rainfall_data.rename(columns={"ob_end_time": "timestamp",␣
       ↪"prcp_amt": "precipitation(mm)"})
```

```
[13]: bike_shares_data["precipitation(mm)"] = 0.0
```

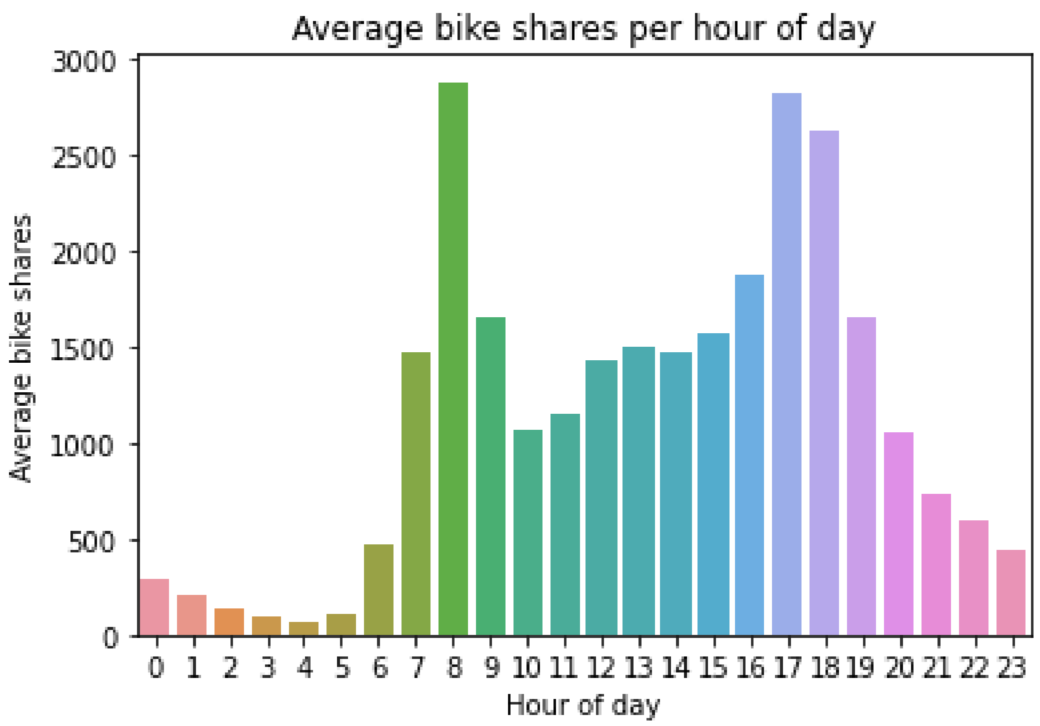
[14]:
```
for i in range(bike_shares_data["timestamp"].shape[0]):
    a = rainfall_data[rainfall_data["timestamp"] ==␣
 ↪bike_shares_data["timestamp"].iat[i]]["precipitation(mm)"]
    if a.size == 1:
        bike_shares_data["precipitation(mm)"].iat[i] = a
```

2 Plot average, or total share count per hour - average likely better

[15]:
```
hours = [str(i) + ":00" for i in bike_shares_data["hour"].unique()]
col = ["cnt"] + hours
hour_of_day_dummy = pd.DataFrame(data = bike_shares_data["cnt"], columns = col)
hour_of_day_dummy = hour_of_day_dummy.fillna(0)
for i in range(bike_shares_data.shape[0]):
    hour = bike_shares_data["hour"][i]
    hour_of_day_dummy[str(hour) + ":00"][i] = 1
hour_of_day_dummy
```

[15]:
```
        cnt  0:00  1:00  2:00  3:00  4:00  5:00  6:00  7:00  8:00  …  14:00  \
0       182     1     0     0     0     0     0     0     0     0  …      0
1       138     0     1     0     0     0     0     0     0     0  …      0
2       134     0     0     1     0     0     0     0     0     0  …      0
3        72     0     0     0     1     0     0     0     0     0  …      0
4        47     0     0     0     0     1     0     0     0     0  …      0
…       …    …    …    …    …    …    …    …    …    …   …    …     …
17409  1042     0     0     0     0     0     0     0     0     0  …      0
17410   541     0     0     0     0     0     0     0     0     0  …      0
17411   337     0     0     0     0     0     0     0     0     0  …      0
17412   224     0     0     0     0     0     0     0     0     0  …      0
17413   139     0     0     0     0     0     0     0     0     0  …      0

       15:00  16:00  17:00  18:00  19:00  20:00  21:00  22:00  23:00
0          0      0      0      0      0      0      0      0      0
1          0      0      0      0      0      0      0      0      0
2          0      0      0      0      0      0      0      0      0
3          0      0      0      0      0      0      0      0      0
4          0      0      0      0      0      0      0      0      0
…       …    …    …    …    …    …    …    …    …    …
17409      0      0      0      0      1      0      0      0      0
17410      0      0      0      0      0      1      0      0      0
17411      0      0      0      0      0      0      1      0      0
17412      0      0      0      0      0      0      0      1      0
17413      0      0      0      0      0      0      0      0      1

[17414 rows x 25 columns]
```

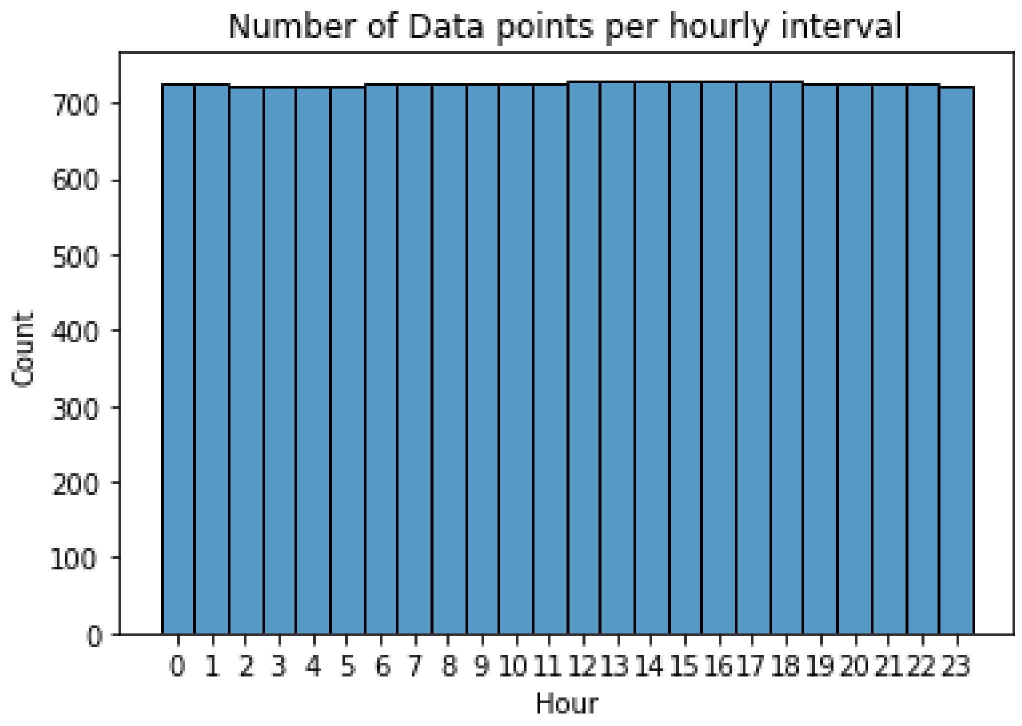
```
[16]: df1 = bike_shares_data.groupby("hour").mean()
      ax = sns.barplot(x = df1.index, y = "cnt", data = df1)
      ax.set(xlabel = "Hour of day", ylabel = "Average bike shares", title = "Average␣
       ↪bike shares per hour of day")

      plt.show()

      #constant added for intercept,
      linear_model=sm.OLS(hour_of_day_dummy["cnt"], sm.add_constant(hour_of_day_dummy.
       ↪drop(labels = ["cnt", "0:00"], axis = 1)))
      result=linear_model.fit()
      print(result.summary())
```



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.605
Model:                            OLS   Adj. R-squared:                  0.604
Method:                 Least Squares   F-statistic:                     1156.
Date:                Sat, 20 Feb 2021   Prob (F-statistic):               0.00
Time:                        00:24:05   Log-Likelihood:            -1.3834e+05
No. Observations:               17414   AIC:                         2.767e+05
Df Residuals:                   17390   BIC:                         2.769e+05
Df Model:                          23
Covariance Type:            nonrobust
```

```
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const        290.6091     25.373     11.453      0.000     240.875     340.343
1:00         -89.9779     35.883     -2.508      0.012    -160.313     -19.643
2:00        -154.3054     35.921     -4.296      0.000    -224.713     -83.898
3:00        -196.3636     35.921     -5.467      0.000    -266.771    -125.956
4:00        -217.2957     35.921     -6.049      0.000    -287.704    -146.888
5:00        -179.9018     35.921     -5.008      0.000    -250.310    -109.494
6:00         176.0176     35.859      4.909      0.000     105.731     246.304
7:00        1178.1361     35.859     32.855      0.000    1107.850    1248.422
8:00        2592.2141     35.883     72.240      0.000    2521.879    2662.549
9:00        1362.4101     35.846     38.007      0.000    1292.148    1432.672
10:00        774.0516     35.871     21.579      0.000     703.741     844.362
11:00        860.6096     35.846     24.008      0.000     790.347     930.872
12:00       1143.1083     35.822     31.911      0.000    1072.894    1213.322
13:00       1215.1299     35.834     33.910      0.000    1144.892    1285.368
14:00       1181.3950     35.834     32.969      0.000    1111.157    1251.633
15:00       1274.0123     35.822     35.565      0.000    1203.798    1344.226
16:00       1579.1923     35.809     44.100      0.000    1509.002    1649.382
17:00       2538.9760     35.834     70.854      0.000    2468.738    2609.214
18:00       2338.4348     35.834     65.258      0.000    2268.197    2408.673
19:00       1360.9012     35.846     37.965      0.000    1290.639    1431.163
20:00        769.1722     35.846     21.458      0.000     698.910     839.434
21:00        450.6402     35.859     12.567      0.000     380.354     520.926
22:00        301.8005     35.871      8.414      0.000     231.490     372.111
23:00        149.0432     35.908      4.151      0.000      78.660     219.427
==============================================================================
Omnibus:                     1579.612   Durbin-Watson:                   0.420
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            10454.999
Skew:                          -0.141   Prob(JB):                         0.00
Kurtosis:                       6.785   Cond. No.                         25.0
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

[17]:
```
df1 = bike_shares_data.groupby("hour").mean()
ax = sns.histplot(x = bike_shares_data["hour"], bins=np.arange(25)-0.5)

ax.set(xlabel = "Hour", ylabel = "Count", title = "Number of Data points per␣
 ↪hourly interval")
ax.set(xticks=range(0, 24), xticklabels=list(range(0,24)))

plt.show()
```



[18]:
```
pd.DataFrame(bike_shares_data["hour"].value_counts().sort_index())
```

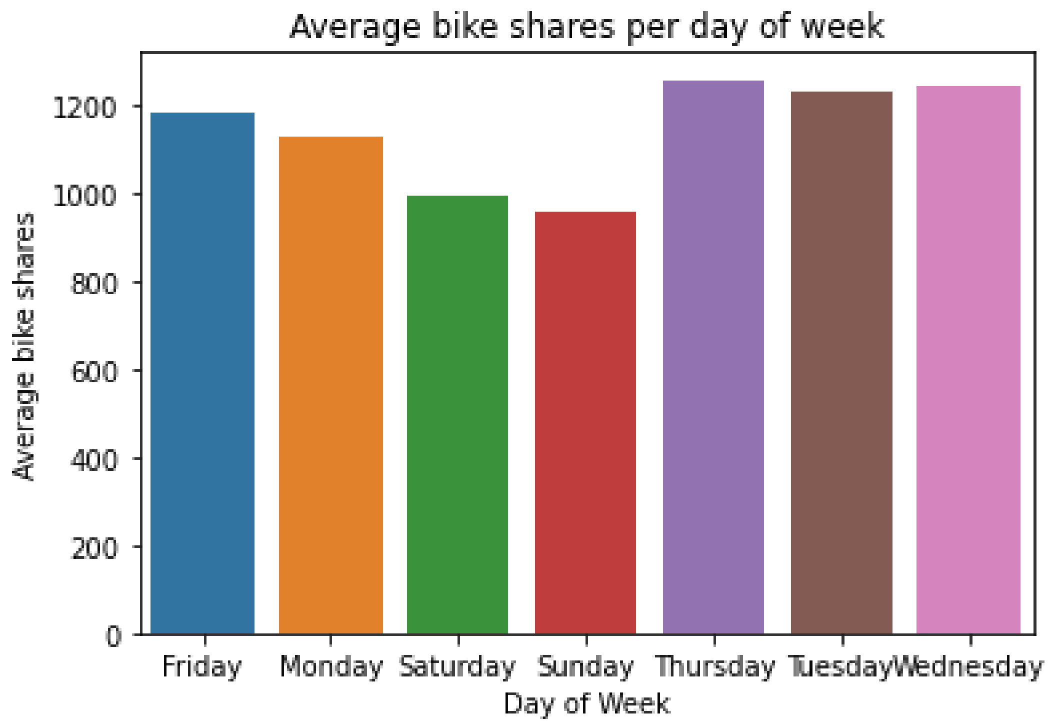
[18]:
```
    hour
0    724
1    724
2    721
3    721
4    721
5    721
6    726
7    726
8    724
9    727
10   725
11   727
12   729
13   728
14   728
15   729
16   730
17   728
18   728
19   727
```

```
20    727
21    726
22    725
23    722
```

3 Plot average share count per day-of-week in histogram

[19]:
```
days = [i for i in bike_shares_data["day of week"].unique()]
col = ["cnt"] + days
day_of_week_dummy = pd.DataFrame(data = bike_shares_data["cnt"], columns = col)
day_of_week_dummy = day_of_week_dummy.fillna(0)
for i in range(bike_shares_data.shape[0]):
    day = bike_shares_data["day of week"][i]
    day_of_week_dummy[day][i] = 1
```

[20]:
```
df1 = bike_shares_data.groupby("day of week").mean()
ax = sns.barplot(x = df1.index, y = "cnt", data = df1)
ax.set(xlabel = "Day of Week", ylabel = "Average bike shares", title = "Average␣
 ↪bike shares per day of week")

plt.show()

#constant added for intercept,
linear_model=sm.OLS(day_of_week_dummy["cnt"], sm.add_constant(day_of_week_dummy.
 ↪drop(labels = ["cnt", "Friday"], axis = 1)))
result=linear_model.fit()
print(result.summary())
```



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.011
Model:                            OLS   Adj. R-squared:                  0.010
Method:                 Least Squares   F-statistic:                     31.49
Date:                Sat, 20 Feb 2021   Prob (F-statistic):           6.88e-38
Time:                        00:24:07   Log-Likelihood:            -1.4633e+05
No. Observations:               17414   AIC:                         2.927e+05
Df Residuals:                   17407   BIC:                         2.927e+05
Df Model:                           6
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       1182.7727     21.808     54.235      0.000    1140.026    1225.519
Sunday      -223.2054     30.672     -7.277      0.000    -283.325    -163.086
Monday       -52.5019     30.663     -1.712      0.087    -112.604       7.600
Tuesday       47.3327     30.672      1.543      0.123     -12.787     107.452
Wednesday     61.6363     30.720      2.006      0.045       1.421     121.851
Thursday      76.0379     30.711      2.476      0.013      15.841     136.235
Saturday    -187.2189     30.795     -6.080      0.000    -247.579    -126.859
==============================================================================
Omnibus:                     3519.550   Durbin-Watson:                   0.442
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             6330.073
```

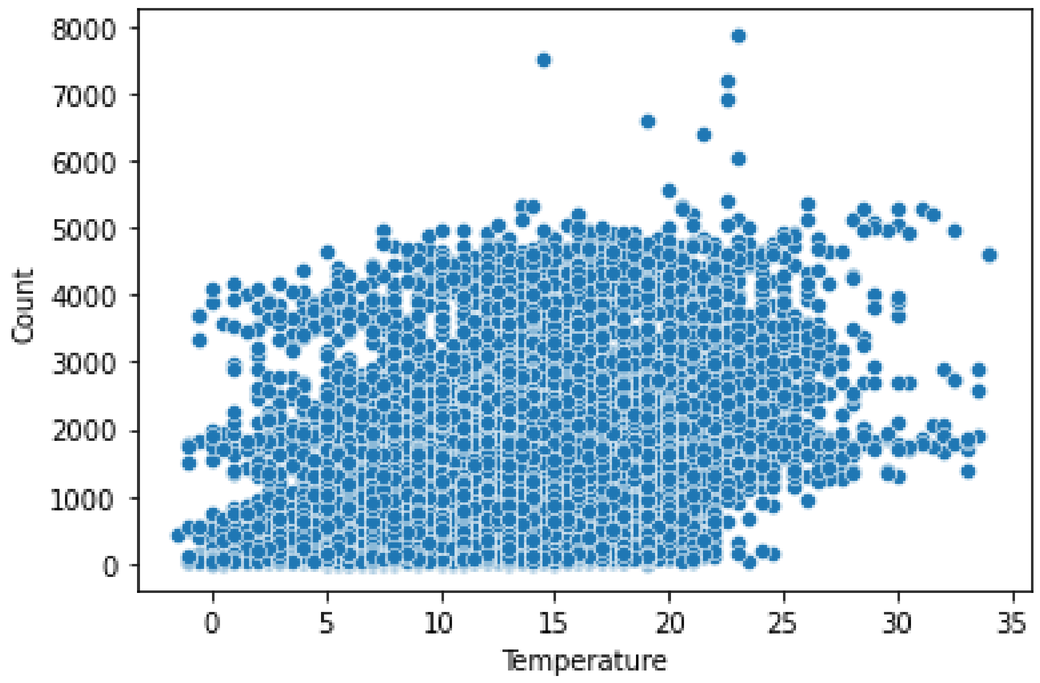
```
Skew:                           1.292   Prob(JB):                         0.00
Kurtosis:                       4.431   Cond. No.                         7.93
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

[21]:
```
np.asarray(bike_shares_data["cnt"])
```

[21]:
```
array([182, 138, 134, …, 337, 224, 139], dtype=int64)
```

4 Plot t1 "actual temperature" against count

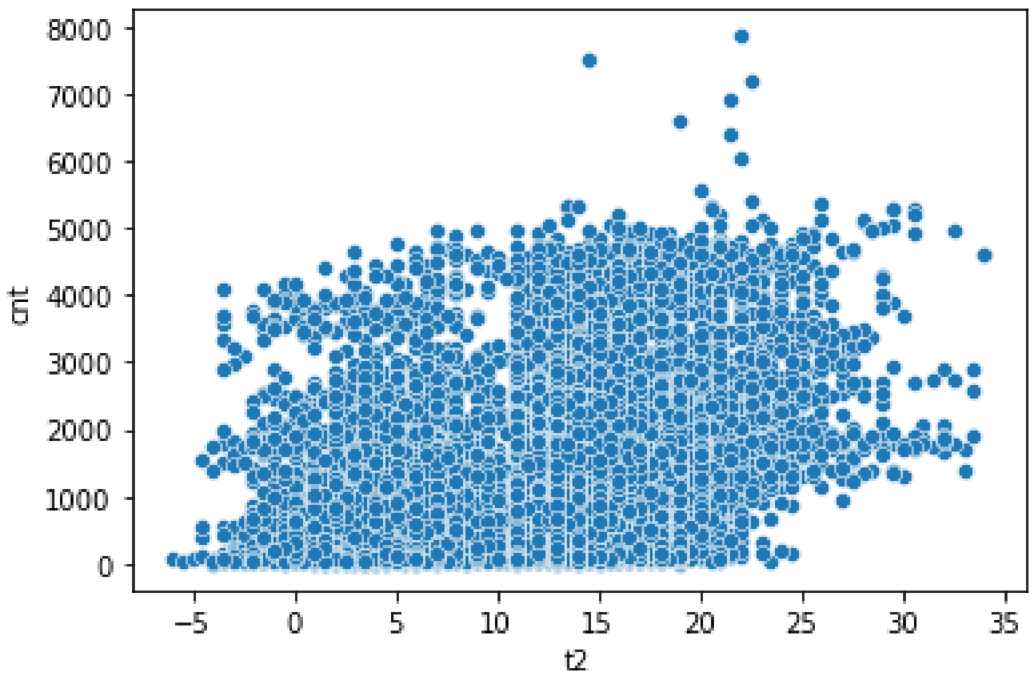
[22]:
```
ax = sns.scatterplot(x = bike_shares_data["t1"], y = bike_shares_data["cnt"])

ax.set(xlabel = "Temperature", ylabel = "Count")

plt.show()

linear_model=sm.OLS(bike_shares_data["cnt"], sm.
 ↪add_constant(bike_shares_data["t1"]))
result=linear_model.fit()
print(result.summary())
```



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.151
Model:                            OLS   Adj. R-squared:                  0.151
Method:                 Least Squares   F-statistic:                     3101.
Date:                Sat, 20 Feb 2021   Prob (F-statistic):               0.00
Time:                        00:24:07   Log-Likelihood:            -1.4500e+05
No. Observations:               17414   AIC:                         2.900e+05
Df Residuals:                   17412   BIC:                         2.900e+05
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const        199.0395     18.569     10.719      0.000     162.641     235.437
t1            75.7183      1.360     55.685      0.000      73.053      78.384
==============================================================================
Omnibus:                     3716.877   Durbin-Watson:                   0.508
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             7309.695
Skew:                           1.294   Prob(JB):                         0.00
Kurtosis:                       4.837   Cond. No.                         33.6
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

5 Plot t2 “feels like temperature” against count

[23]:
```
sns.scatterplot(x = bike_shares_data["t2"], y = bike_shares_data["cnt"])
plt.show()

linear_model=sm.OLS(bike_shares_data["cnt"], sm.
 ↪add_constant(bike_shares_data["t2"]))
result=linear_model.fit()
print(result.summary())
```

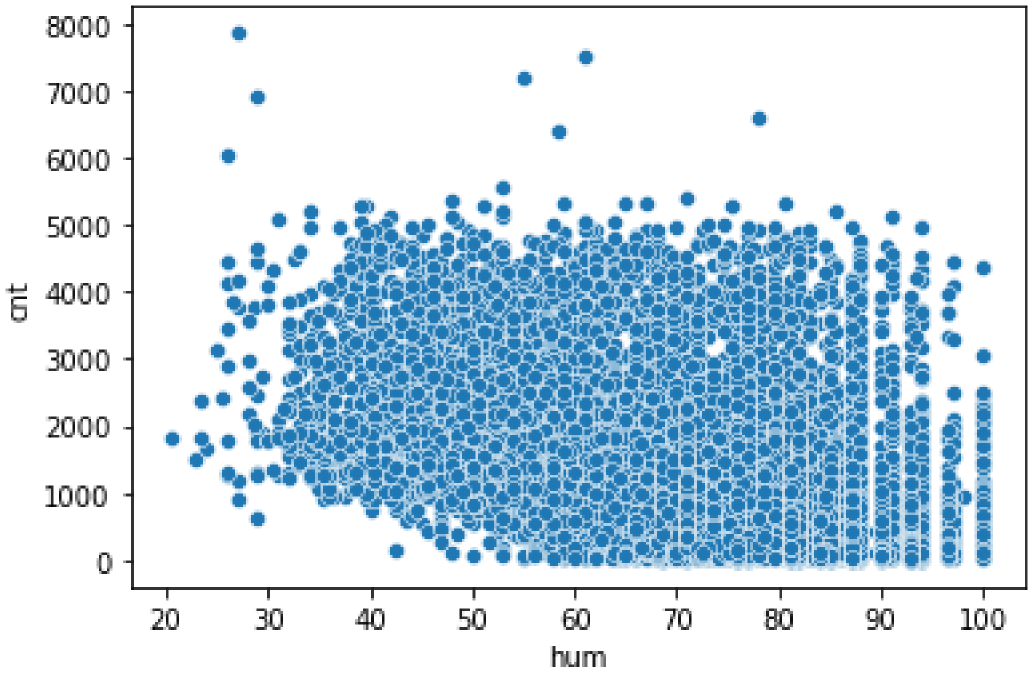


```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.136
Model:                            OLS   Adj. R-squared:                  0.136
Method:                 Least Squares   F-statistic:                     2745.
Date:                Sat, 20 Feb 2021   Prob (F-statistic):               0.00
Time:                        00:24:07   Log-Likelihood:            -1.4515e+05
No. Observations:               17414   AIC:                         2.903e+05
Df Residuals:                   17412   BIC:                         2.903e+05
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const        445.6968     15.349     29.038      0.000     415.612     475.782
t2            60.5342      1.155     52.394      0.000      58.270      62.799
==============================================================================
Omnibus:                     3625.855   Durbin-Watson:                   0.501
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             6962.646
Skew:                           1.278   Prob(JB):                         0.00
Kurtosis:                       4.750   Cond. No.                         26.8
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
```

```
specified.
```

6 Plot humidity against count

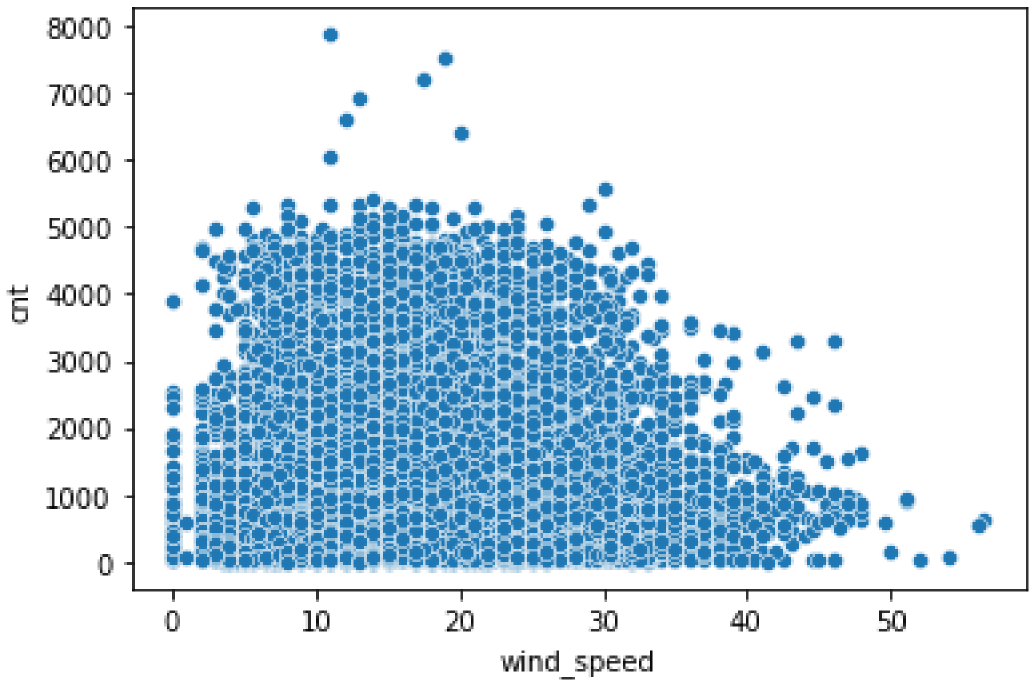
[24]:
```
sns.scatterplot(x = bike_shares_data["hum"], y = bike_shares_data["cnt"])
plt.show()

linear_model=sm.OLS(bike_shares_data["cnt"], sm.
 ↪add_constant(bike_shares_data["hum"]))
result=linear_model.fit()
print(result.summary())
```



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.214
Model:                            OLS   Adj. R-squared:                  0.214
Method:                 Least Squares   F-statistic:                     4748.
Date:                Sat, 20 Feb 2021   Prob (F-statistic):               0.00
Time:                        00:24:07   Log-Likelihood:            -1.4432e+05
No. Observations:               17414   AIC:                         2.887e+05
Df Residuals:                   17412   BIC:                         2.887e+05
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       3681.2262     37.547     98.043      0.000    3607.630    3754.822
hum          -35.0933      0.509    -68.909      0.000     -36.092     -34.095
==============================================================================
Omnibus:                     4870.089   Durbin-Watson:                   0.555
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            12270.925
Skew:                           1.539   Prob(JB):                         0.00
Kurtosis:                       5.727   Cond. No.                         380.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

7 Plot wind speed against count

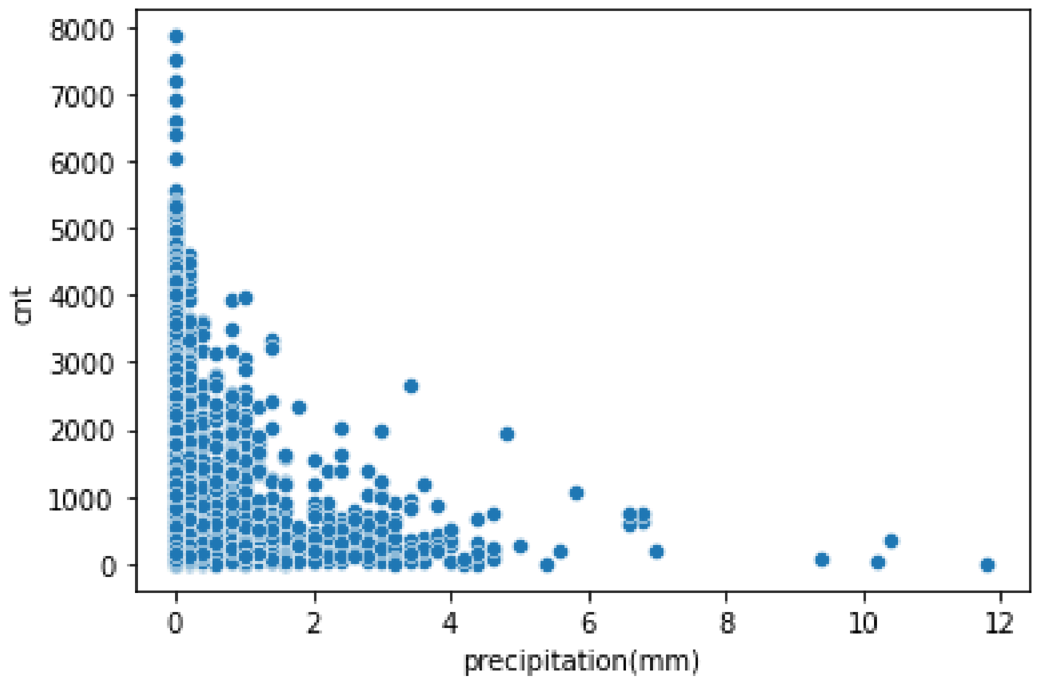
[25]:
```
sns.scatterplot(x = bike_shares_data["wind_speed"], y = bike_shares_data["cnt"])
plt.show()

linear_model=sm.OLS(bike_shares_data["cnt"], sm.
 ↪add_constant(bike_shares_data["wind_speed"]))
result=linear_model.fit()
print(result.summary())
```



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.014
Model:                            OLS   Adj. R-squared:                  0.013
Method:                 Least Squares   F-statistic:                     238.7
Date:                Sat, 20 Feb 2021   Prob (F-statistic):           1.70e-53
Time:                        00:24:08   Log-Likelihood:            -1.4630e+05
No. Observations:               17414   AIC:                         2.926e+05
Df Residuals:                   17412   BIC:                         2.926e+05
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const        888.7350     18.378     48.359      0.000     852.713     924.757
wind_speed    15.9848      1.035     15.451      0.000      13.957      18.013
==============================================================================
Omnibus:                     3811.201   Durbin-Watson:                   0.443
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             7293.776
Skew:                           1.350   Prob(JB):                         0.00
Kurtosis:                       4.662   Cond. No.                         40.1
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

8 Plot precipitation(mm) against count

[26]:
```
sns.scatterplot(x = bike_shares_data["precipitation(mm)"], y =␣
 ↪bike_shares_data["cnt"])
plt.show()
linear_model=sm.OLS(bike_shares_data["cnt"], sm.
 ↪add_constant(bike_shares_data["precipitation(mm)"]))
result=linear_model.fit()
print(result.summary())
```



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.009
Model:                            OLS   Adj. R-squared:                  0.009
Method:                 Least Squares   F-statistic:                     153.4
Date:                Sat, 20 Feb 2021   Prob (F-statistic):           4.32e-35
Time:                        00:24:08   Log-Likelihood:            -1.4635e+05
No. Observations:               17414   AIC:                         2.927e+05
Df Residuals:                   17412   BIC:                         2.927e+05
Df Model:                           1
Covariance Type:            nonrobust
================================================================================
=====
                       coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
-----
const             1160.6244      8.308    139.692      0.000    1144.339
1176.910
precipitation(mm) -263.7346     21.291    -12.387      0.000    -305.467
-222.002
==============================================================================
Omnibus:                     3653.241   Durbin-Watson:                   0.445
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             6764.134
Skew:                           1.318   Prob(JB):                         0.00
```

```
Kurtosis:                       4.542   Cond. No.                         2.61
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

[27]:
```
bike_shares_data
```

[27]:
```
       observation           timestamp day of week  hour   cnt   t1   t2  \
0                0 2015-01-04 00:00:00      Sunday     0   182  3.0  2.0
1                1 2015-01-04 01:00:00      Sunday     1   138  3.0  2.5
2                2 2015-01-04 02:00:00      Sunday     2   134  2.5  2.5
3                3 2015-01-04 03:00:00      Sunday     3    72  2.0  2.0
4                4 2015-01-04 04:00:00      Sunday     4    47  2.0  0.0
…              …                   …          …    …    …    …    …
17409        17409 2017-01-03 19:00:00     Tuesday    19  1042  5.0  1.0
17410        17410 2017-01-03 20:00:00     Tuesday    20   541  5.0  1.0
17411        17411 2017-01-03 21:00:00     Tuesday    21   337  5.5  1.5
17412        17412 2017-01-03 22:00:00     Tuesday    22   224  5.5  1.5
17413        17413 2017-01-03 23:00:00     Tuesday    23   139  5.0  1.0

         hum  wind_speed  weather_code  is_holiday  is_weekend  season  \
0       93.0         6.0           3.0         0.0         1.0  winter
1       93.0         5.0           1.0         0.0         1.0  winter
2       96.5         0.0           1.0         0.0         1.0  winter
3      100.0         0.0           1.0         0.0         1.0  winter
4       93.0         6.5           1.0         0.0         1.0  winter
…        …          …            …          …          …       …
17409   81.0        19.0           3.0         0.0         0.0  winter
17410   81.0        21.0           4.0         0.0         0.0  winter
17411   78.5        24.0           4.0         0.0         0.0  winter
17412   76.0        23.0           4.0         0.0         0.0  winter
17413   76.0        22.0           2.0         0.0         0.0  winter

       precipitation(mm)
0                    0.0
1                    0.0
2                    0.0
3                    0.0
4                    0.0
…                    …
17409                0.0
17410                0.0
17411                0.0
17412                0.0
17413                0.0
```

```
[17414 rows x 14 columns]
```

9 Histogram of weather code against count

Weather code likely not the best sort of predictor

[28]:
```
weather = ["weather code " + str(int(i)) for i in np.
 ↪sort(bike_shares_data["weather_code"].unique())]

col = ["cnt"] + weather

col
```

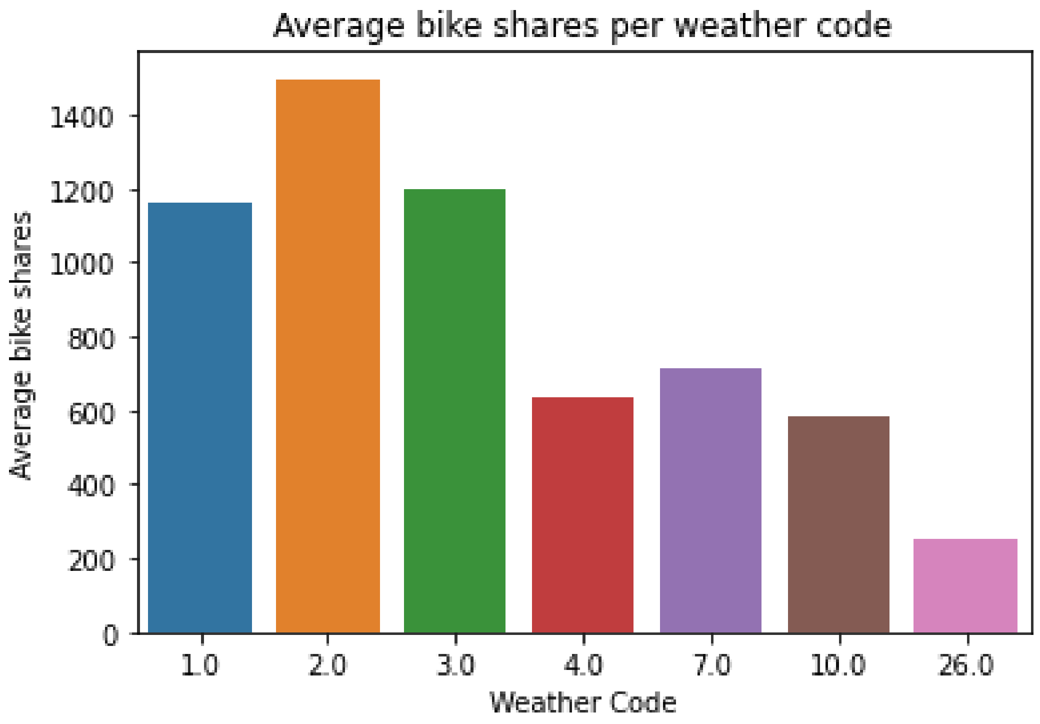
[28]:
```
['cnt',
 'weather code 1',
 'weather code 2',
 'weather code 3',
 'weather code 4',
 'weather code 7',
 'weather code 10',
 'weather code 26']
```

[29]:
```
weather_code_dummy = pd.DataFrame(data = bike_shares_data["cnt"], columns = col)
weather_code_dummy = weather_code_dummy.fillna(0)
weather_code_dummy

for i in range(bike_shares_data.shape[0]):
    code = "weather code " + str(int(bike_shares_data["weather_code"][i]))
    weather_code_dummy[code][i] = 1

weather_code_dummy
```

[29]:
```
        cnt  weather code 1  weather code 2  weather code 3  weather code 4  \
0       182               0               0               1               0
1       138               1               0               0               0
2       134               1               0               0               0
3        72               1               0               0               0
4        47               1               0               0               0
…     …              …              …              …              …
17409  1042               0               0               1               0
17410   541               0               0               0               1
17411   337               0               0               0               1
17412   224               0               0               0               1
17413   139               0               1               0               0

       weather code 7  weather code 10  weather code 26
```

```
0                    0                0                0
1                    0                0                0
2                    0                0                0
3                    0                0                0
4                    0                0                0
…               …               …               …
17409                0                0                0
17410                0                0                0
17411                0                0                0
17412                0                0                0
17413                0                0                0

[17414 rows x 8 columns]
```

[30]:
```
df1 = bike_shares_data.groupby("weather_code").mean()
ax = sns.barplot(x = df1.index, y = "cnt", data = df1)
ax.set(xlabel = "Weather Code", ylabel = "Average bike shares", title =␣
 ↪"Average bike shares per weather code")

plt.show()

linear_model=sm.OLS(weather_code_dummy["cnt"], sm.
 ↪add_constant(weather_code_dummy.drop(labels = ["cnt", "weather code 2"],␣
 ↪axis = 1)))
result=linear_model.fit()
print(result.summary())
```

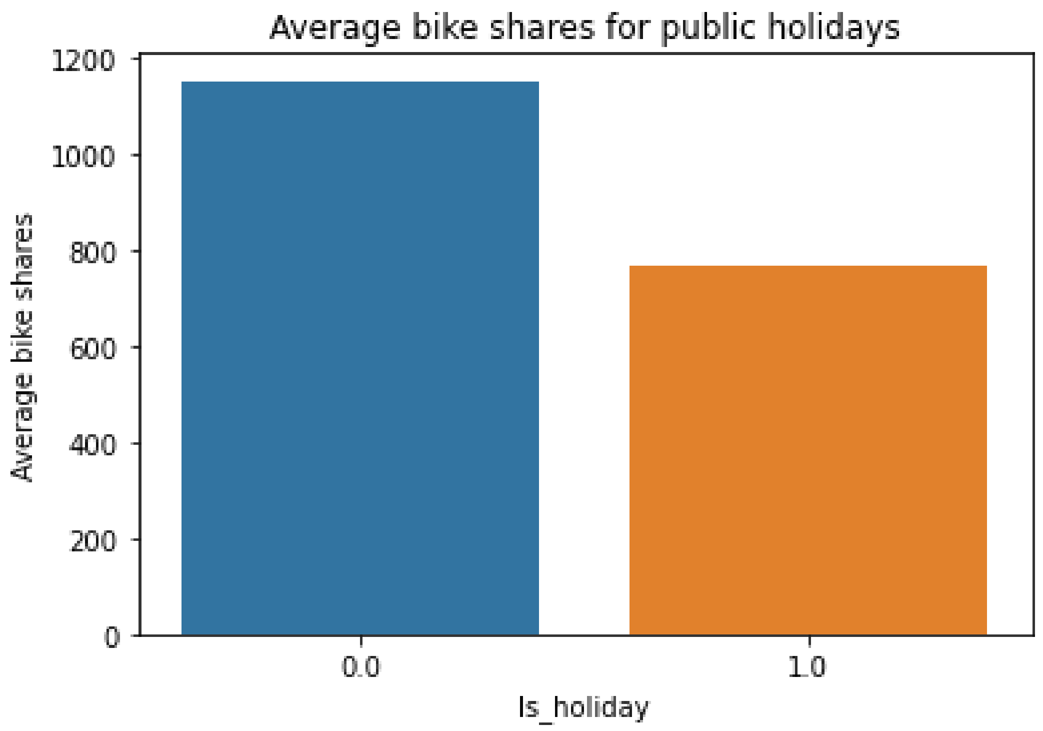


```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.065
Model:                            OLS   Adj. R-squared:                  0.065
Method:                 Least Squares   F-statistic:                     203.0
Date:                Sat, 20 Feb 2021   Prob (F-statistic):          4.09e-251
Time:                        00:24:09   Log-Likelihood:            -1.4583e+05
No. Observations:               17414   AIC:                         2.917e+05
Df Residuals:                   17407   BIC:                         2.917e+05
Df Model:                           6
Covariance Type:            nonrobust
================================================================================
===
                      coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
---
const            1496.1775     16.519     90.571      0.000    1463.798
1528.557
weather code 1   -334.0885     21.258    -15.716      0.000    -375.756
-292.421
weather code 3   -301.0530     24.143    -12.469      0.000    -348.377
-253.730
weather code 4   -860.9466     32.013    -26.894      0.000    -923.696
-798.198
weather code 7   -783.2111     28.055    -27.917      0.000    -838.201
-728.221
weather code 10  -912.7489    280.900     -3.249      0.001   -1463.342
-362.156
weather code 26 -1245.3275    136.457     -9.126      0.000   -1512.796
-977.859
==============================================================================
Omnibus:                     3749.423   Durbin-Watson:                   0.510
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             7230.494
Skew:                           1.320   Prob(JB):                         0.00
Kurtosis:                       4.730   Cond. No.                         38.8
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

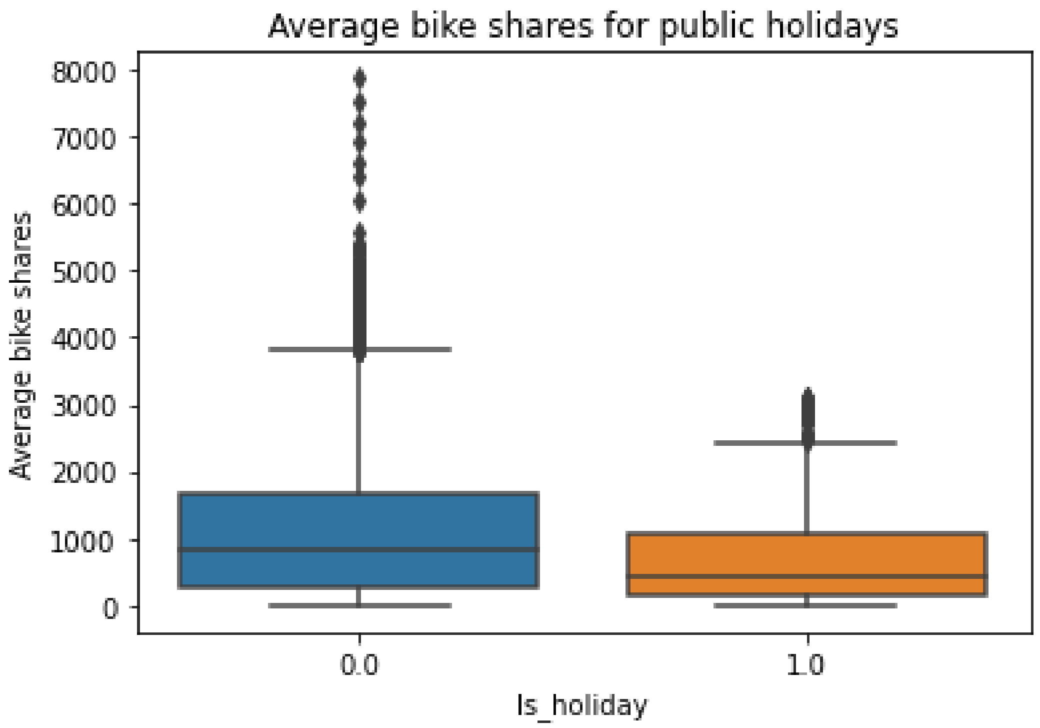
10 Average count for "is holiday = TRUE" vs average count for "is holiday = False"

May need to control for other variables as holidays are few and far between

[31]:
```
df1 = bike_shares_data.groupby("is_holiday").mean()
ax = sns.barplot(x = df1.index, y = "cnt", data = df1)
ax.set(xlabel = "Is_holiday", ylabel = "Average bike shares", title = "Average␣
 ↪bike shares for public holidays")

plt.show()

linear_model=sm.OLS(bike_shares_data["cnt"], sm.
 ↪add_constant(bike_shares_data["is_holiday"]))
result=linear_model.fit()
print(result.summary())

df2 = bike_shares_data.filter(items = ["is_holiday", "cnt"], axis = 1)
ax = sns.boxplot(x = "is_holiday", y = "cnt", data = df2)
ax.set(xlabel = "Is_holiday", ylabel = "Average bike shares", title = "Average␣
 ↪bike shares for public holidays")

plt.show()
```

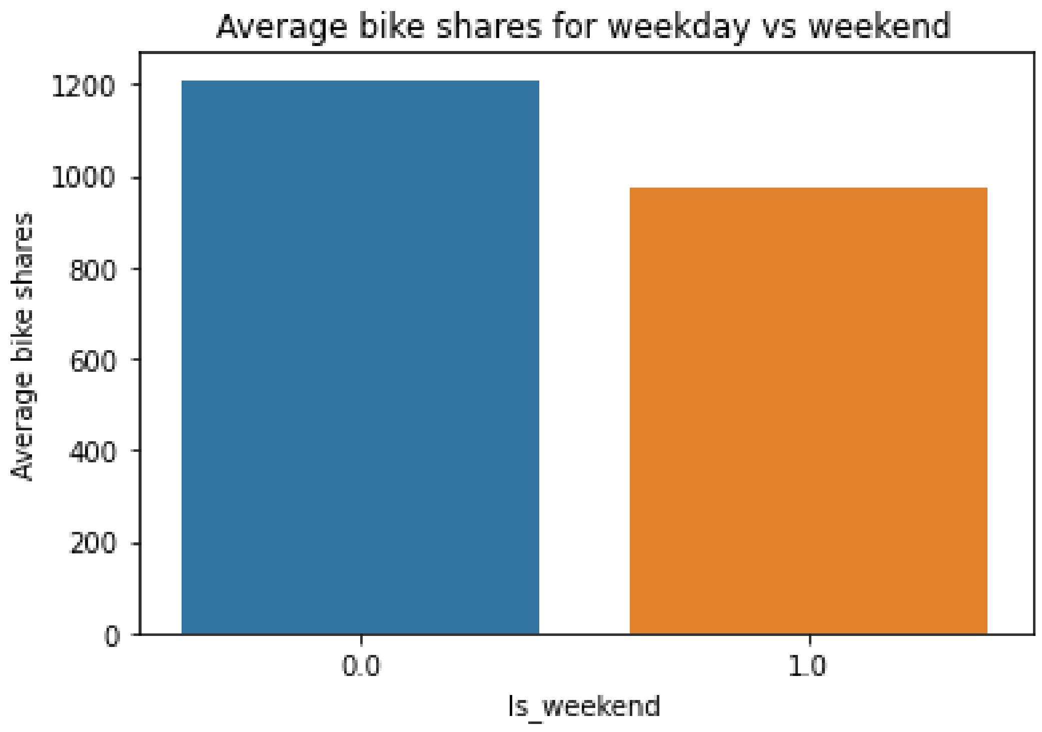


```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.003
Model:                            OLS   Adj. R-squared:                  0.003
Method:                 Least Squares   F-statistic:                     46.66
Date:                Sat, 20 Feb 2021   Prob (F-statistic):           8.71e-12
Time:                        00:24:10   Log-Likelihood:            -1.4640e+05
No. Observations:               17414   AIC:                         2.928e+05
Df Residuals:                   17412   BIC:                         2.928e+05
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       1151.5252      8.304    138.668      0.000    1135.248    1167.802
is_holiday  -381.9991     55.922     -6.831      0.000    -491.611    -272.387
==============================================================================
Omnibus:                     3661.265   Durbin-Watson:                   0.438
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             6776.708
Skew:                           1.321   Prob(JB):                         0.00
Kurtosis:                       4.535   Cond. No.                         6.81
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```



11 Average count for “is weekend = TRUE” vs average count for “is weekend = False”

```
[32]: df1 = bike_shares_data.groupby("is_weekend").mean()
      ax = sns.barplot(x = df1.index, y = "cnt", data = df1)
      ax.set(xlabel = "Is_weekend", ylabel = "Average bike shares", title = "Average␣
       ↪bike shares for weekday vs weekend")

      plt.show()

      linear_model=sm.OLS(bike_shares_data["cnt"], sm.
       ↪add_constant(bike_shares_data["is_weekend"]))
      result=linear_model.fit()
      print(result.summary())
```



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.009
Model:                            OLS   Adj. R-squared:                  0.009
Method:                 Least Squares   F-statistic:                     163.7
```

```
Date:                Sat, 20 Feb 2021   Prob (F-statistic):          2.63e-37
Time:                        00:24:10   Log-Likelihood:           -1.4634e+05
No. Observations:               17414   AIC:                        2.927e+05
Df Residuals:                   17412   BIC:                        2.927e+05
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       1209.2748      9.682    124.897      0.000    1190.297    1228.253
is_weekend  -231.8591     18.124    -12.793      0.000    -267.383    -196.335
==============================================================================
Omnibus:                     3542.728   Durbin-Watson:                   0.442
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             6404.084
Skew:                           1.296   Prob(JB):                         0.00
Kurtosis:                       4.451   Cond. No.                         2.44
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

12 Average count for season

[33]:
```python
season = [str(i) for i in np.sort(bike_shares_data["season"].unique())]

col = ["cnt"] + season

col
```

[33]:
```
['cnt', 'fall', 'spring', 'summer', 'winter']
```

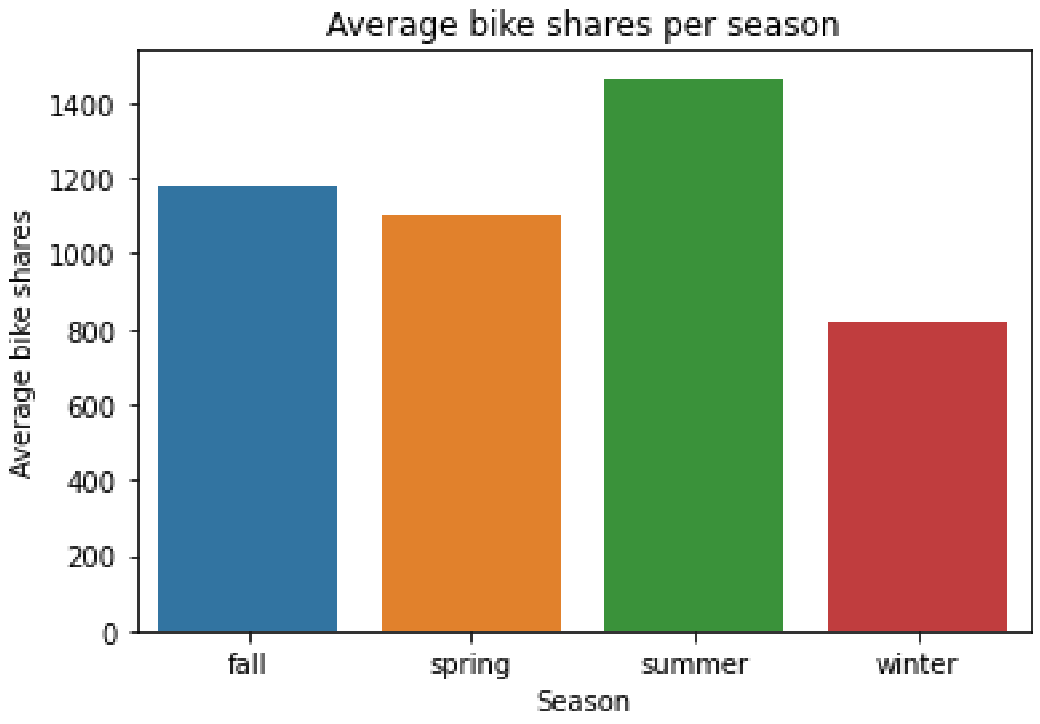
[34]:
```python
bike_shares_data["season"]
```

[34]:
```
0        winter
1        winter
2        winter
3        winter
4        winter
          …
17409    winter
17410    winter
17411    winter
17412    winter
17413    winter
Name: season, Length: 17414, dtype: object
```

[35]:
```
season_dummy = pd.DataFrame(data = bike_shares_data["cnt"], columns = col)
season_dummy = season_dummy.fillna(0)
season_dummy

for i in range(bike_shares_data.shape[0]):
    code = bike_shares_data["season"][i]
    season_dummy[code][i] = 1

season_dummy
```

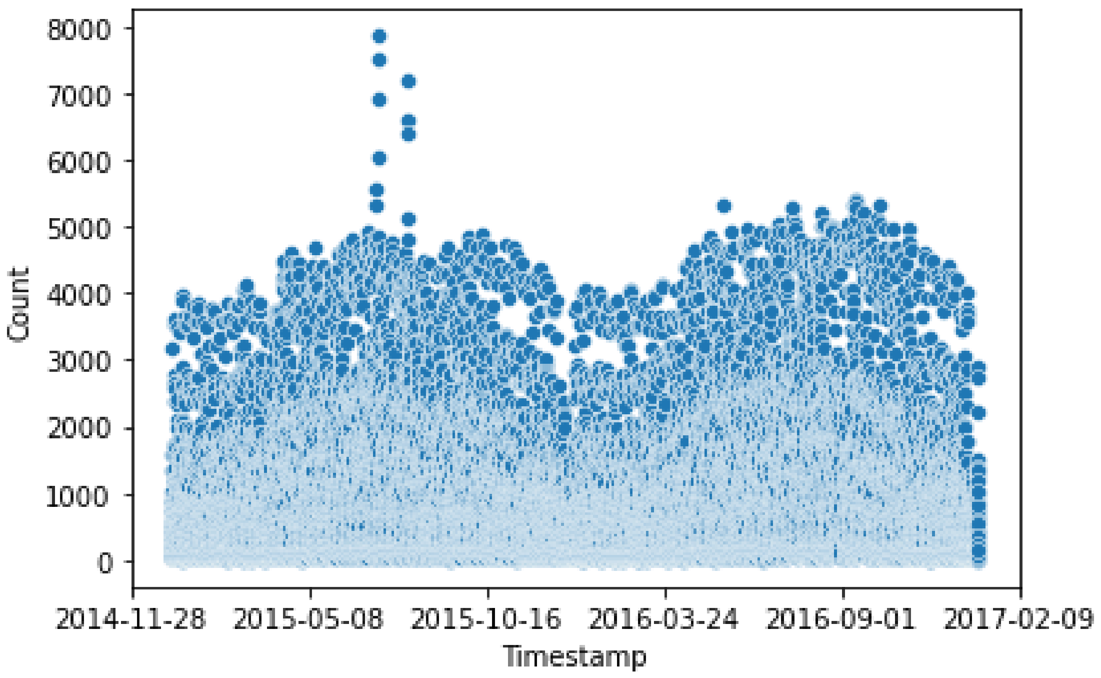
[35]:
```
        cnt  fall  spring  summer  winter
0       182     0       0       0       1
1       138     0       0       0       1
2       134     0       0       0       1
3        72     0       0       0       1
4        47     0       0       0       1
…      …    …      …      …      …
17409  1042     0       0       0       1
17410   541     0       0       0       1
17411   337     0       0       0       1
17412   224     0       0       0       1
17413   139     0       0       0       1

[17414 rows x 5 columns]
```

[36]:
```
df1 = bike_shares_data.groupby("season").mean()
ax = sns.barplot(x = df1.index, y = "cnt", data = df1)
ax.set(xlabel = "Season", ylabel = "Average bike shares", title = "Average bike␣
 ↪shares per season")

plt.show()

linear_model=sm.OLS(season_dummy["cnt"], sm.add_constant(season_dummy.
 ↪drop(labels = ["cnt", "summer"], axis = 1)))
result=linear_model.fit()
print(result.summary())
```



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.045
Model:                            OLS   Adj. R-squared:                  0.044
Method:                 Least Squares   F-statistic:                     270.3
Date:                Sat, 20 Feb 2021   Prob (F-statistic):         1.66e-171
Time:                        00:24:11   Log-Likelihood:           -1.4603e+05
No. Observations:               17414   AIC:                         2.921e+05
Df Residuals:                   17410   BIC:                         2.921e+05
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       1464.4652     16.015     91.441      0.000    1433.073    1495.857
fall        -285.5110     22.760    -12.545      0.000    -330.122    -240.900
spring      -360.6336     22.640    -15.929      0.000    -405.011    -316.256
winter      -642.7361     22.724    -28.285      0.000    -687.277    -598.195
==============================================================================
Omnibus:                     3321.878   Durbin-Watson:                   0.457
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             5850.871
Skew:                           1.232   Prob(JB):                         0.00
Kurtosis:                       4.412   Cond. No.                         4.78
==============================================================================
Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

[37]:
```
import matplotlib.ticker as ticker
ax = sns.scatterplot(x = bike_shares_data["timestamp"], y =␣
 ↪bike_shares_data["cnt"])
ax.set(xlabel = "Timestamp", ylabel = "Count")
ax.xaxis.set_major_locator(ticker.LinearLocator(6))
plt.show()
linear_model=sm.OLS(bike_shares_data["cnt"], sm.
 ↪add_constant(bike_shares_data["observation"]))
result=linear_model.fit()
print(result.summary())
```



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.002
Model:                            OLS   Adj. R-squared:                  0.002
Method:                 Least Squares   F-statistic:                     28.02
Date:                Sat, 20 Feb 2021   Prob (F-statistic):           1.21e-07
Time:                        00:24:11   Log-Likelihood:            -1.4641e+05
No. Observations:               17414   AIC:                         2.928e+05
Df Residuals:                   17412   BIC:                         2.928e+05
Df Model:                           1
Covariance Type:            nonrobust
===============================================================================
                  coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------
const        1067.7680     16.432     64.980      0.000    1035.559    1099.977
observation     0.0087      0.002      5.294      0.000       0.005       0.012
==============================================================================
Omnibus:                     3658.931   Durbin-Watson:                   0.438
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             6768.183
Skew:                           1.321   Prob(JB):                         0.00
Kurtosis:                       4.533   Cond. No.                     2.01e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 2.01e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

[38]:
```
corr_matrix_continous = bike_shares_data.drop(labels = ["timestamp","season",␣
↪"day of week", "is_weekend", "is_holiday", "weather_code"], axis = 1).corr()
```

[39]:
```
corr_matrix_appended = pd.concat((bike_shares_data, day_of_week_dummy.
↪drop(labels = "cnt", axis = 1),
          weather_code_dummy.drop(labels = "cnt", axis = 1),
          season_dummy.drop(labels = "cnt", axis = 1),
          hour_of_day_dummy.drop(labels = "cnt", axis = 1)), axis = 1, join =␣
↪"outer").corr()
count_corr = corr_matrix_appended.filter(items = ["cnt"], axis = 1).abs().
↪sort_values(by = "cnt", axis = 0, ascending = False)
```

[40]:
```
corr_matrix_continous
```

[40]:
```
                   observation      hour       cnt        t1        t2  \
observation           1.000000  0.001678  0.040086  0.132239  0.143846
hour                  0.001678  1.000000  0.324423  0.168708  0.153956
cnt                   0.040086  0.324423  1.000000  0.388798  0.369035
t1                    0.132239  0.168708  0.388798  1.000000  0.988344
t2                    0.143846  0.153956  0.369035  0.988344  1.000000
hum                   0.119287 -0.295653 -0.462901 -0.447781 -0.403495
wind_speed           -0.126083  0.141792  0.116295  0.145471  0.088409
precipitation(mm)     0.025859 -0.004743 -0.093463 -0.004250 -0.007846

                        hum  wind_speed  precipitation(mm)
observation        0.119287   -0.126083           0.025859
hour              -0.295653    0.141792          -0.004743
```

```
cnt                -0.462901    0.116295          -0.093463
t1                 -0.447781    0.145471          -0.004250
t2                 -0.403495    0.088409          -0.007846
hum                 1.000000   -0.287789           0.153130
wind_speed         -0.287789    1.000000           0.043441
precipitation(mm)   0.153130    0.043441           1.000000
```

[41]:
```
corr_matrix_continous.filter(items = ["cnt"], axis = 1).abs().sort_values(by =␣
 ↪"cnt", axis = 0, ascending = False)
```

[41]:
```
                        cnt
cnt                1.000000
hum                0.462901
t1                 0.388798
t2                 0.369035
hour               0.324423
wind_speed         0.116295
precipitation(mm)  0.093463
observation        0.040086
```

[42]:
```
count_corr
```

[42]:
```
                        cnt
cnt                1.000000
hum                0.462901
t1                 0.388798
t2                 0.369035
8:00               0.333934
17:00              0.324647
hour               0.324423
18:00              0.286043
4:00               0.204898
3:00               0.200889
5:00               0.197736
2:00               0.192833
1:00               0.180904
weather code 2     0.178668
summer             0.171869
winter             0.170381
weather_code       0.166633
0:00               0.163633
weather code 7     0.148419
weather code 4     0.141802
16:00              0.140090
23:00              0.134830
6:00               0.130034
wind_speed         0.116295
```

```
22:00                0.105780
9:00                 0.098088
19:00                0.097798
is_weekend           0.096499
precipitation(mm)    0.093463
15:00                0.081200
21:00                0.077245
13:00                0.069807
Sunday               0.069332
14:00                0.063313
7:00                 0.062596
12:00                0.055983
Saturday             0.055217
is_holiday           0.051698
weather code 26      0.048351
Thursday             0.043578
observation          0.040086
Wednesday            0.038127
Tuesday              0.032867
weather code 3       0.024265
spring               0.021024
fall                 0.018929
20:00                0.016028
10:00                0.015067
Friday               0.014794
weather code 10      0.014631
weather code 1       0.012930
Monday               0.004850
11:00                0.001561
```

[43]:
```
from sklearn.ensemble import RandomForestRegressor
a = pd.concat((bike_shares_data, day_of_week_dummy.drop(labels = "cnt", axis =␣
 ↪1),
           weather_code_dummy.drop(labels = "cnt", axis = 1),
           season_dummy.drop(labels = "cnt", axis = 1),
           hour_of_day_dummy.drop(labels = "cnt", axis = 1)), axis = 1, join =␣
 ↪"outer").drop(labels = ["timestamp","cnt","season", "day of week", "t2"],␣
 ↪axis = 1).values
b = pd.concat((bike_shares_data, day_of_week_dummy.drop(labels = "cnt", axis =␣
 ↪1),
           weather_code_dummy.drop(labels = "cnt", axis = 1),
           season_dummy.drop(labels = "cnt", axis = 1),
           hour_of_day_dummy.drop(labels = "cnt", axis = 1)), axis = 1, join =␣
 ↪"outer")["cnt"].values
c = RandomForestRegressor(n_estimators = 100, max_features = "auto").fit(a,b)
```

```
[44]: e = pd.concat((bike_shares_data, day_of_week_dummy.drop(labels = "cnt", axis =␣
       ↪1),
                 weather_code_dummy.drop(labels = "cnt", axis = 1),
                 season_dummy.drop(labels = "cnt", axis = 1),
                 hour_of_day_dummy.drop(labels = "cnt", axis = 1)), axis = 1, join =␣
       ↪"outer").drop(labels = ["timestamp","cnt","season", "day of week", "t2"],␣
       ↪axis = 1).columns
      d = zip(e, c.feature_importances_)
```

```
[45]: f = list(d)
```

```
[46]: pd.DataFrame(f, columns = ["feature", "importance"]).sort_values(by =␣
       ↪"importance", axis = 0, ascending = False)
```

```
[46]:               feature  importance
      1                hour    0.552914
      7          is_weekend    0.148761
      2                  t1    0.086558
      35               8:00    0.058625
      3                 hum    0.029340
      6          is_holiday    0.018605
      0         observation    0.018096
      46              19:00    0.011654
      5        weather_code    0.011287
      26             winter    0.010028
      8   precipitation(mm)    0.009198
      4          wind_speed    0.008393
      45              18:00    0.003050
      44              17:00    0.002884
      14             Friday    0.002861
      43              16:00    0.002743
      47              20:00    0.002708
      37              10:00    0.002180
      20     weather code 7    0.002126
      34               7:00    0.001802
      38              11:00    0.001539
      13           Thursday    0.001397
      36               9:00    0.001362
      9              Sunday    0.001251
      15           Saturday    0.001244
      33               6:00    0.001243
      10             Monday    0.000834
      23               fall    0.000812
      18     weather code 3    0.000735
      16     weather code 1    0.000671
      24             spring    0.000667
      11            Tuesday    0.000636
```

```
17     weather code 2    0.000620
25             summer    0.000616
12          Wednesday    0.000514
48              21:00    0.000455
50              23:00    0.000321
19     weather code 4    0.000207
39              12:00    0.000190
27               0:00    0.000175
42              15:00    0.000164
41              14:00    0.000154
40              13:00    0.000135
49              22:00    0.000121
28               1:00    0.000048
32               5:00    0.000027
29               2:00    0.000027
30               3:00    0.000013
22    weather code 26    0.000004
21    weather code 10    0.000004
31               4:00    0.000002
```

[47]:
```
from sklearn.ensemble import RandomForestRegressor
a2 = bike_shares_data.drop(labels = ["timestamp","cnt","season", "day of week",␣
 ↪"is_weekend", "is_holiday", "weather_code", "t2"], axis = 1).values
b2 = bike_shares_data["cnt"].values
c2 = RandomForestRegressor(n_estimators = 5, verbose = 2).fit(a2,b2)
```

```
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.0s remaining:    0.0s

building tree 1 of 5
building tree 2 of 5
building tree 3 of 5
building tree 4 of 5
building tree 5 of 5

[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:    0.2s finished
```

[48]:
```
e2 = bike_shares_data.drop(labels = ["timestamp","cnt","season", "day of week",␣
 ↪"is_weekend", "is_holiday", "weather_code", "t2"], axis = 1).columns
d2 = zip(e2, c2.feature_importances_)
```

[49]:
```
f2 = list(d2)
```

[50]:
```
pd.DataFrame(f2, columns = ["feature", "importance"]).sort_values(by =␣
 ↪"importance", axis = 0, ascending = False)
```

[50]:
```
             feature  importance
1               hour    0.614407
2                 t1    0.133248
```

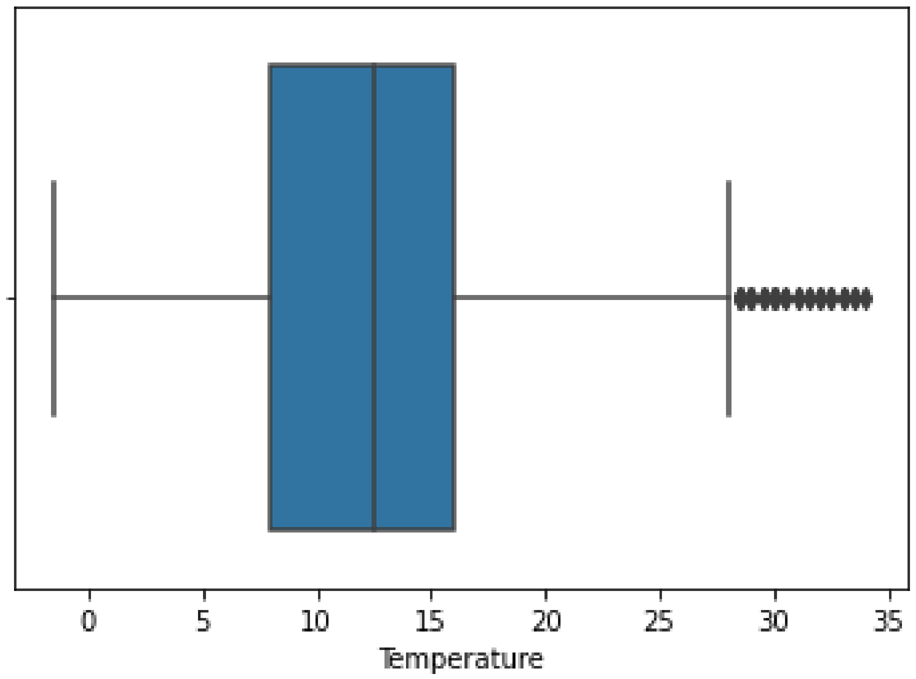
```
0        observation    0.096788
3                hum    0.076231
4         wind_speed    0.062931
5  precipitation(mm)    0.016395
```

[51]:
```
bike_shares_data.filter(items = ["hour", "t1", "hum", "is_weekend",␣
 ↪"precipitation(mm)"], axis = 1).describe()
```

[51]:
```
               hour            t1           hum    is_weekend  \
count  17414.000000  17414.000000  17414.000000  17414.000000
mean      11.513265     12.468091     72.324954      0.285403
std        6.915893      5.571818     14.313186      0.451619
min        0.000000     -1.500000     20.500000      0.000000
25%        6.000000      8.000000     63.000000      0.000000
50%       12.000000     12.500000     74.500000      0.000000
75%       18.000000     16.000000     83.000000      1.000000
max       23.000000     34.000000    100.000000      1.000000

       precipitation(mm)
count       17414.000000
mean            0.066441
std             0.384543
min             0.000000
25%             0.000000
50%             0.000000
75%             0.000000
max            11.800000
```

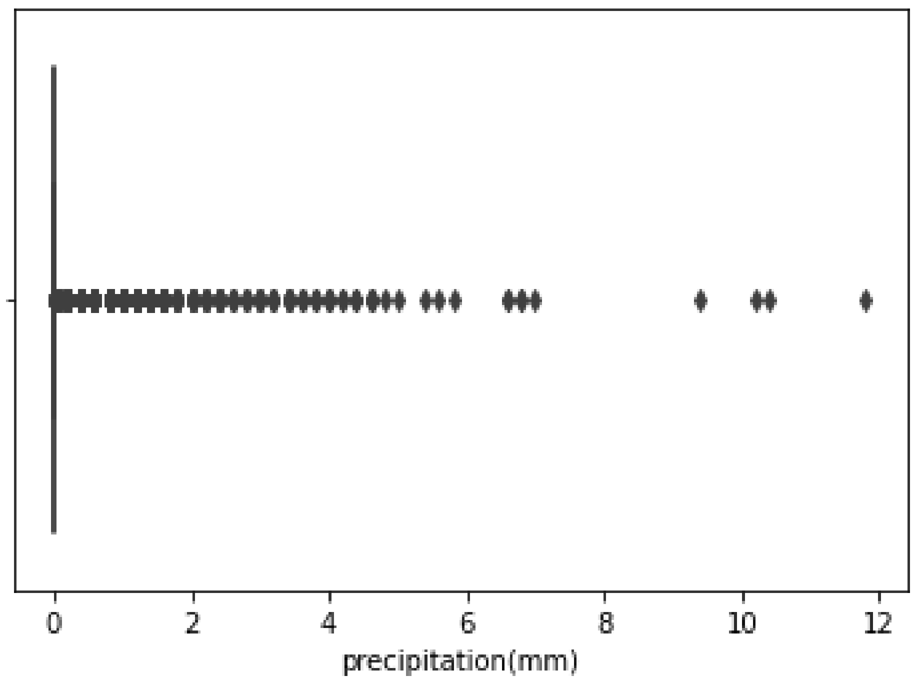
[52]:
```
ax = sns.boxplot(bike_shares_data["t1"])

ax.set(xlabel = "Temperature")

plt.show()
```

```
C:\Users\Montel\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. 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(
```



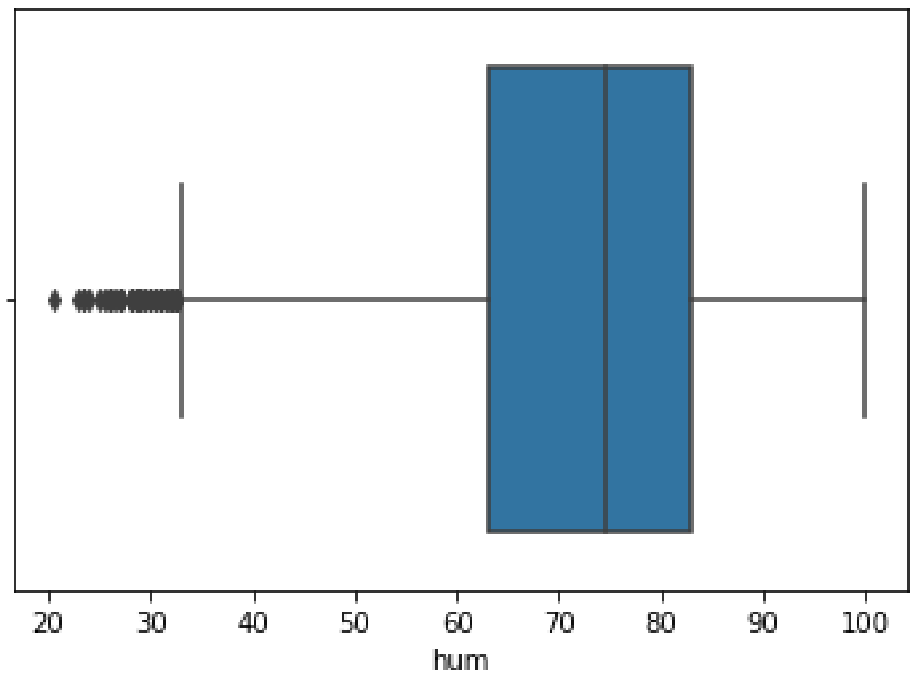
```
[53]: sns.boxplot(bike_shares_data["precipitation(mm)"])

      plt.show()
```

```
C:\Users\Montel\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. 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(
```

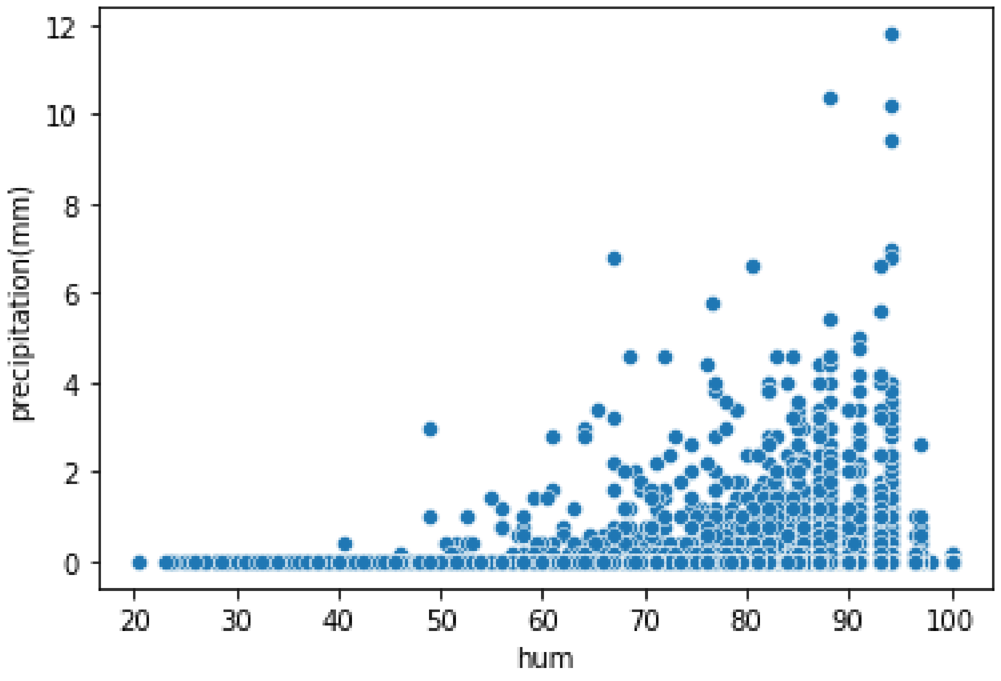


[54]:
```
sns.boxplot(bike_shares_data["hum"])

plt.show()
```

```
C:\Users\Montel\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. 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(
```



[55]:
```
sns.scatterplot(x = bike_shares_data["hum"], y =␣
 ↪bike_shares_data["precipitation(mm)"])

plt.show()
```


precipitation(mm)
hum