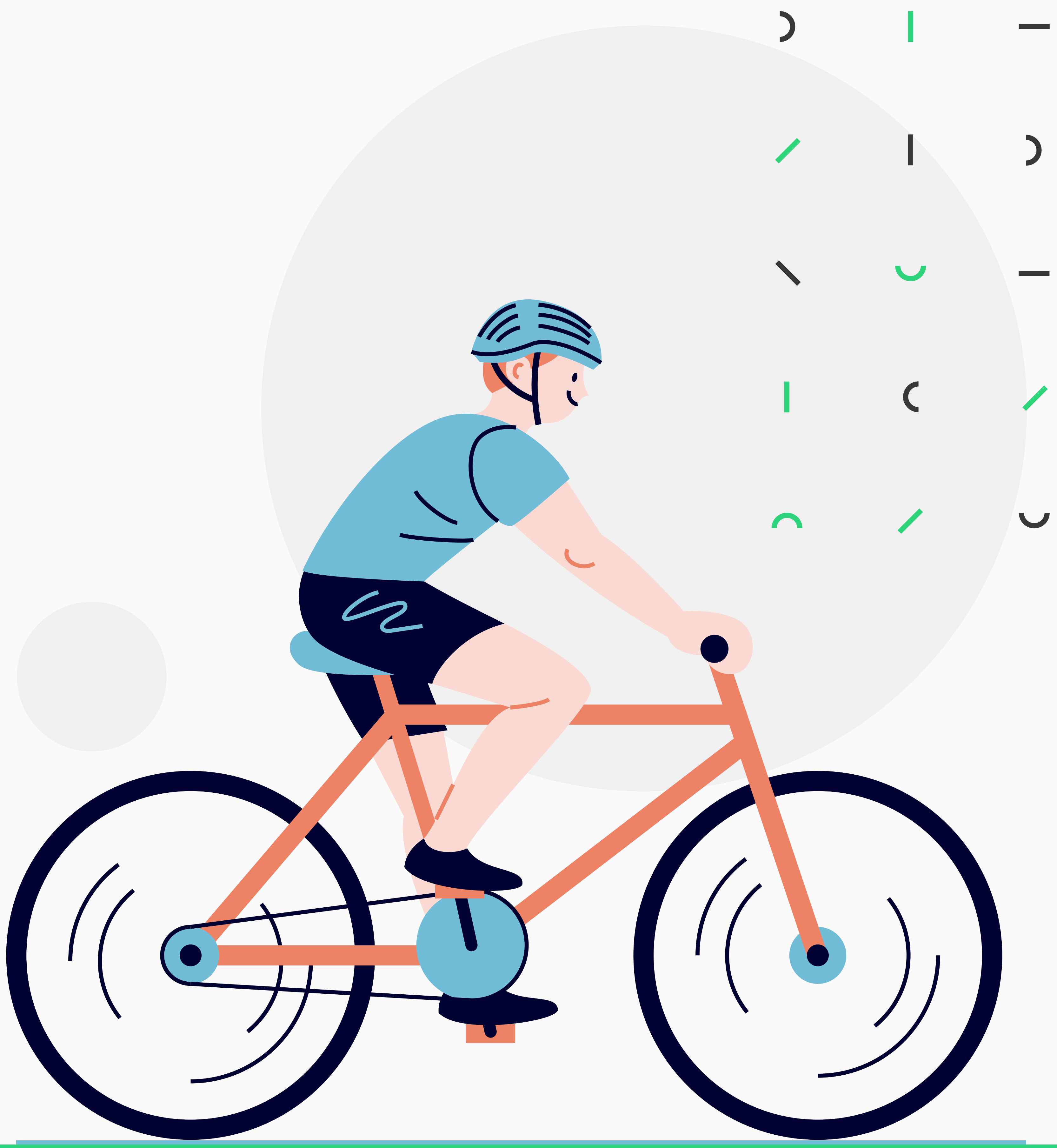
Seoul Bike Sharing Demand Analysis



EMMA KILBERTUS
ANOUK LEYRIS
ANNA PLAIDEAU

Dataset Information

→ Prediction of hourly bike attendance required for the stable supply of rental bikes

- Rental bikes are introduced in many urban cities for the enhancement of mobility comfort
- It is important to make the rental bike available and accessible to the public at the right time as it lessens the waiting time

Attribute Information

DATE

Year-month-day

RENTED BIKE COUNT

Count of bikes rented at each hour

HOUR

Hour of the day

TEMPERATURE

Temperature in Celsius

HUMIDITY

%

WINDSPEED

m/s

VISIBILITY

10m

DEW POINT TEMPERATURE

Celsius

SOLAR RADIATION

MJ/m2

RAINFALL

mm

SNOWFALL

cm

SEASONS

Winter, Spring, Summer, Autumn

HOLIDAY

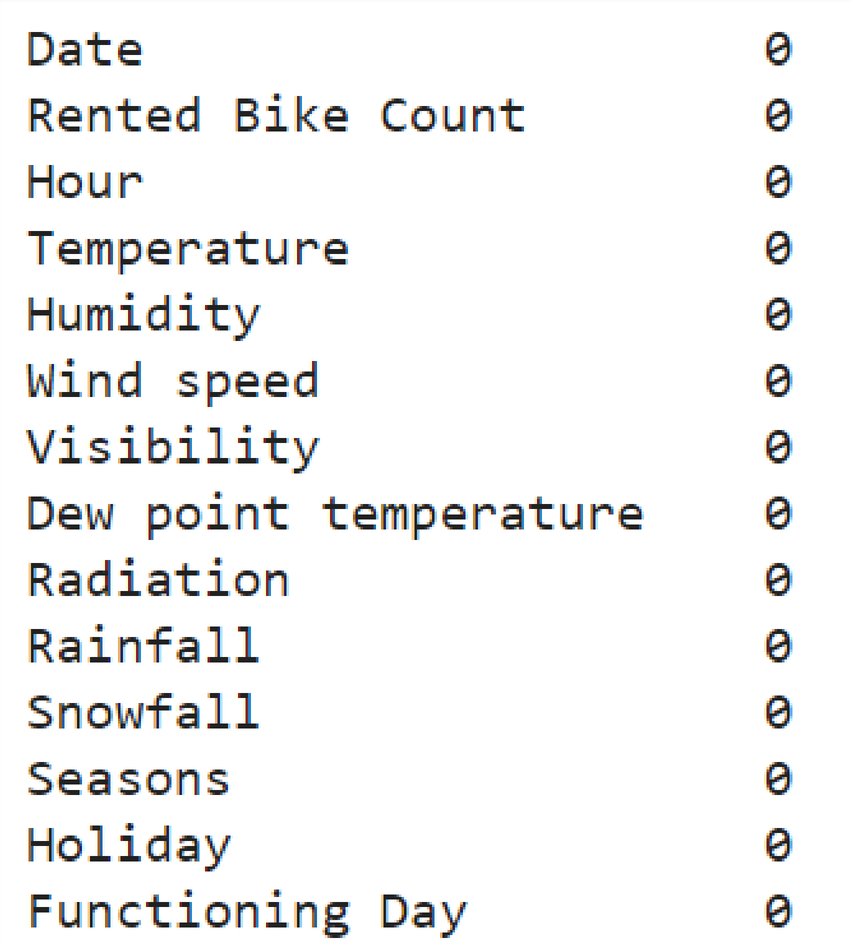
Holiday/No holiday

FUNCTIONAL DAY

NoFunc(Non Functional Hours), Fun(Functional hours)

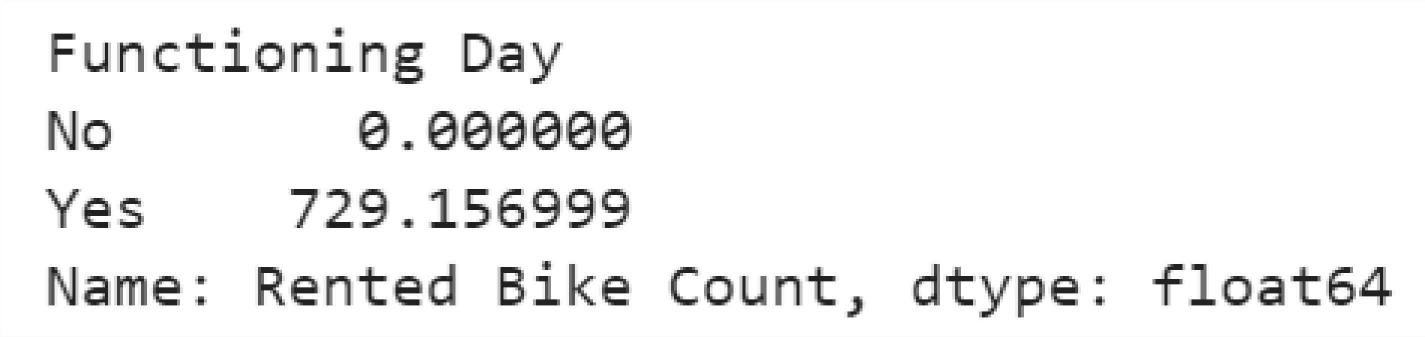
Data Cleaning

Checking for missing values



```
Date                     0
Rented Bike Count        0
Hour                     0
Temperature              0
Humidity                 0
Wind speed               0
Visibility               0
Dew point temperature    0
Radiation                0
Rainfall                 0
Snowfall                 0
Seasons                  0
Holiday                  0
Functioning Day          0
```

Functioning Day = No → no bike is rented



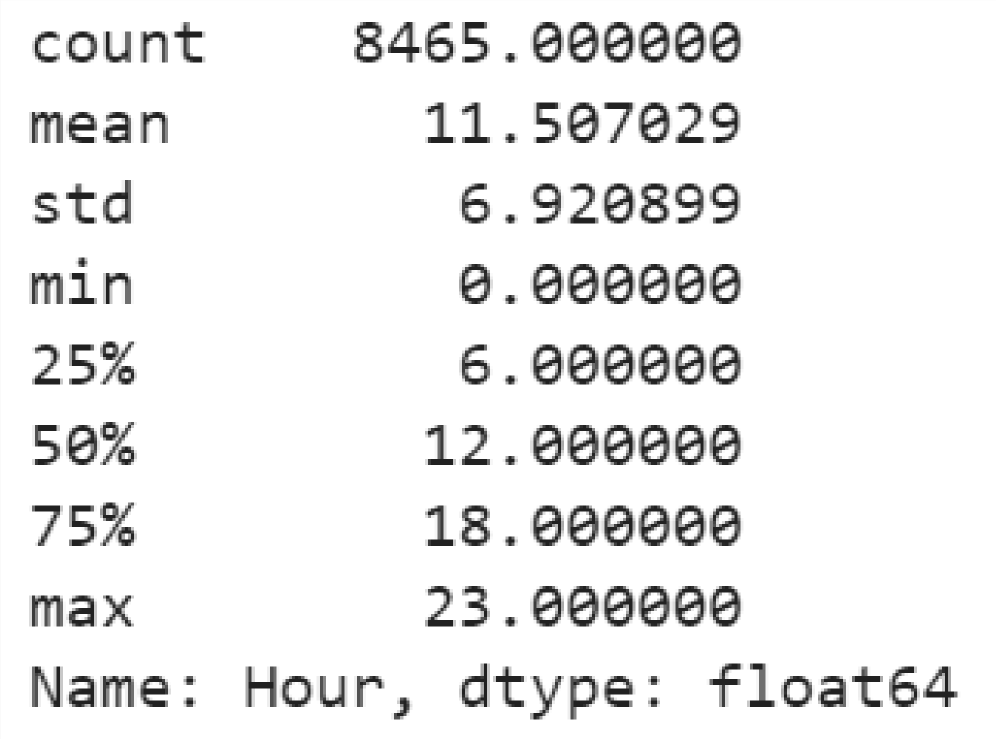
```
Functioning Day
No       0.000000
Yes    729.156999
Name: Rented Bike Count, dtype: float64
```

We delete the rows where "Functioning Day = No" and the column Functioning Day

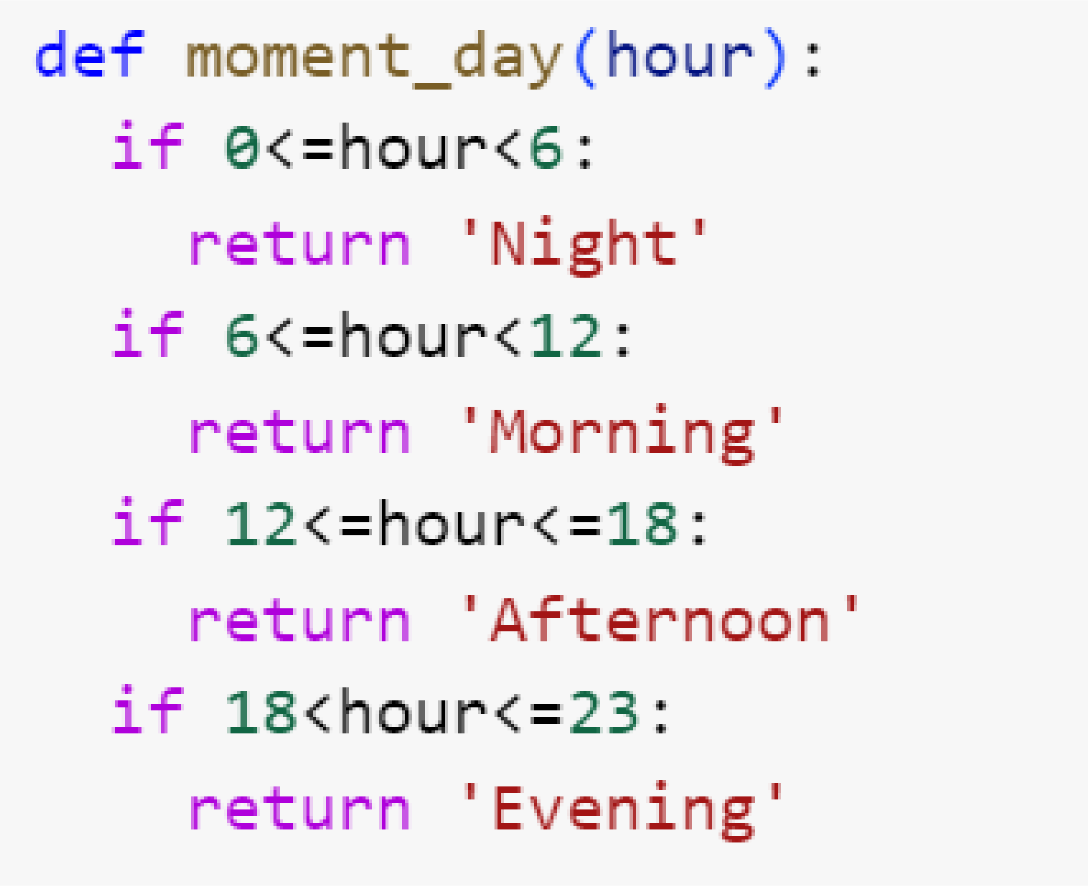
Feature Engineering

- **Separate date into 3 columns → associate each date with a day of the week**

- **Create a column Moment Day**

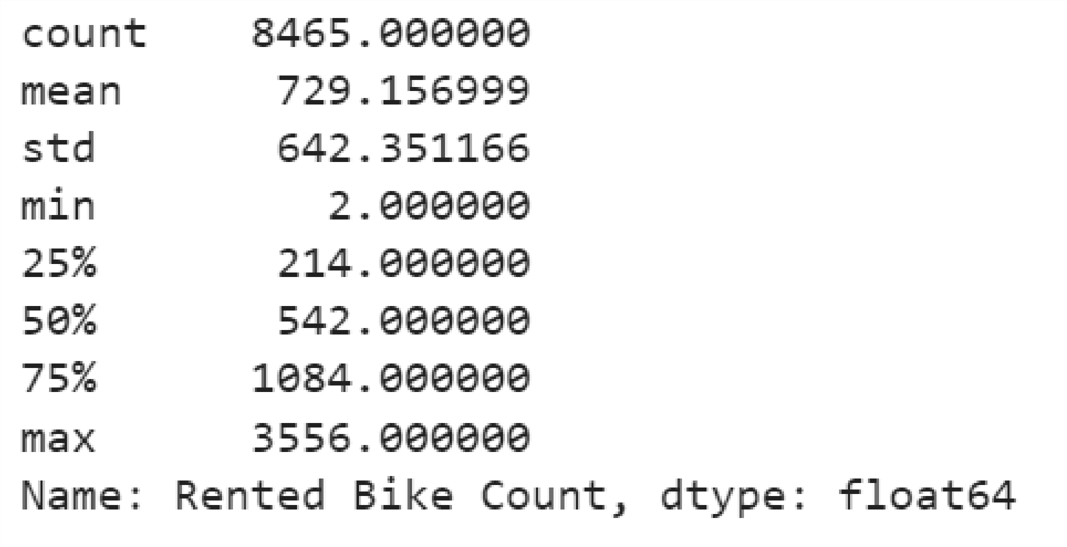


```
count    8465.000000
mean       11.507029
std         6.920899
min         0.000000
25%         6.000000
50%        12.000000
75%        18.000000
max        23.000000
Name: Hour, dtype: float64
```

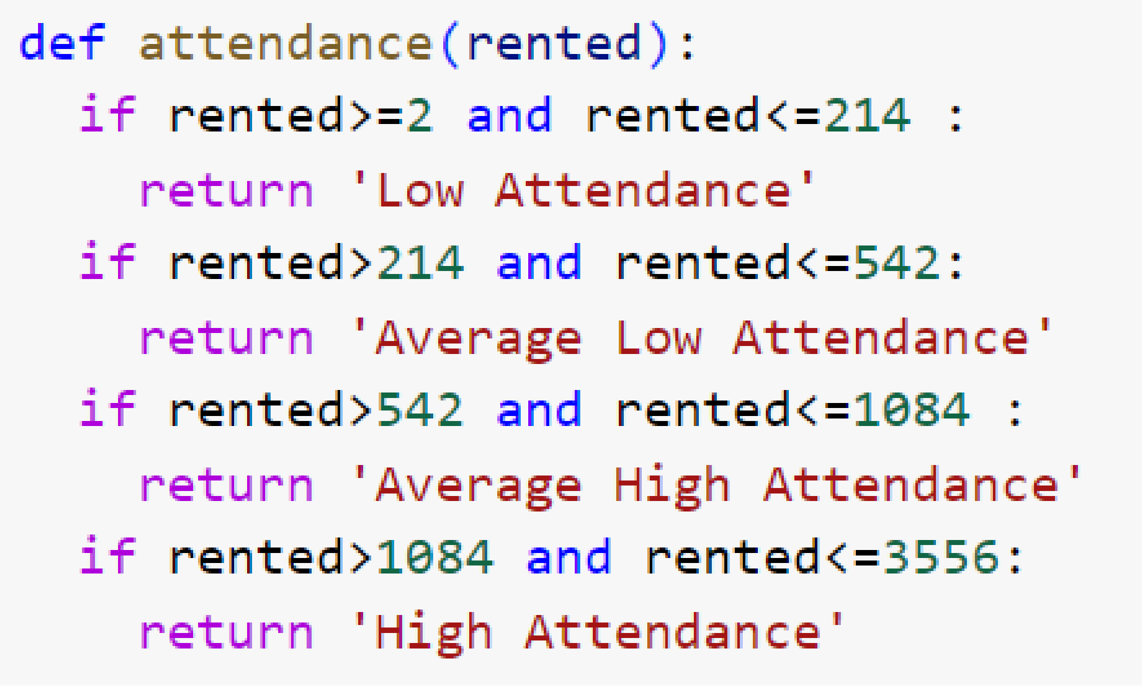


```
def moment_day(hour):
  if 0<=hour<6:
    return 'Night'
  if 6<=hour<12:
    return 'Morning'
  if 12<=hour<=18:
    return 'Afternoon'
  if 18<hour<=23:
    return 'Evening'
```

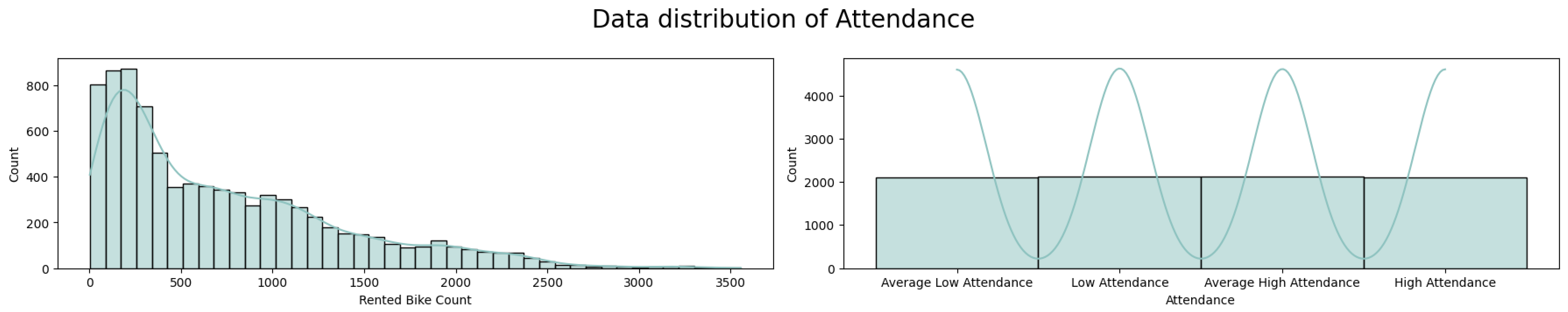
Attendance classes made by statistics



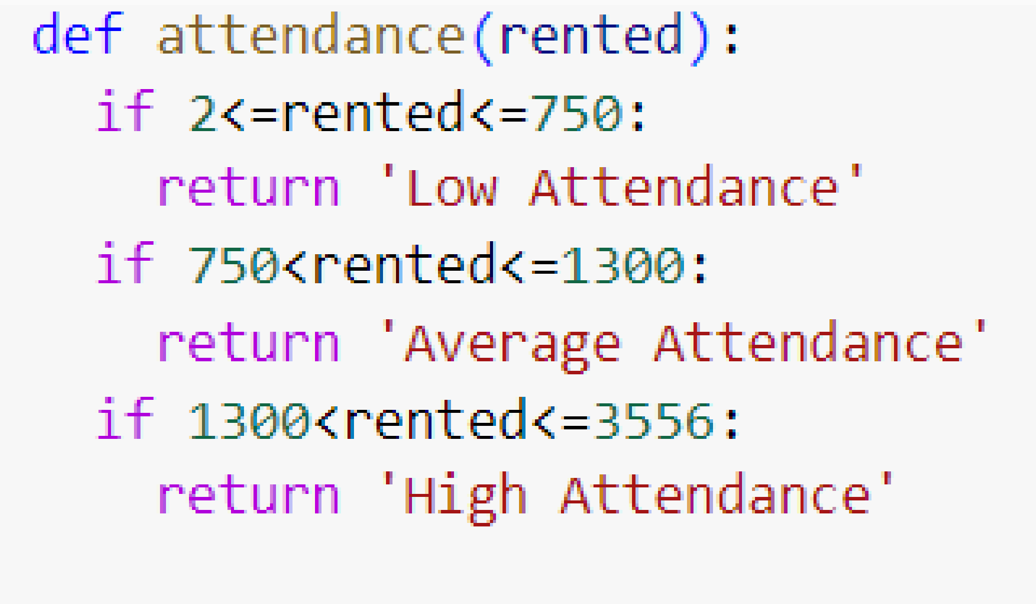
```
count     8465.000000
mean       729.156999
std        642.351166
min          2.000000
25%        214.000000
50%        542.000000
75%       1084.000000
max       3556.000000
Name: Rented Bike Count, dtype: float64
```



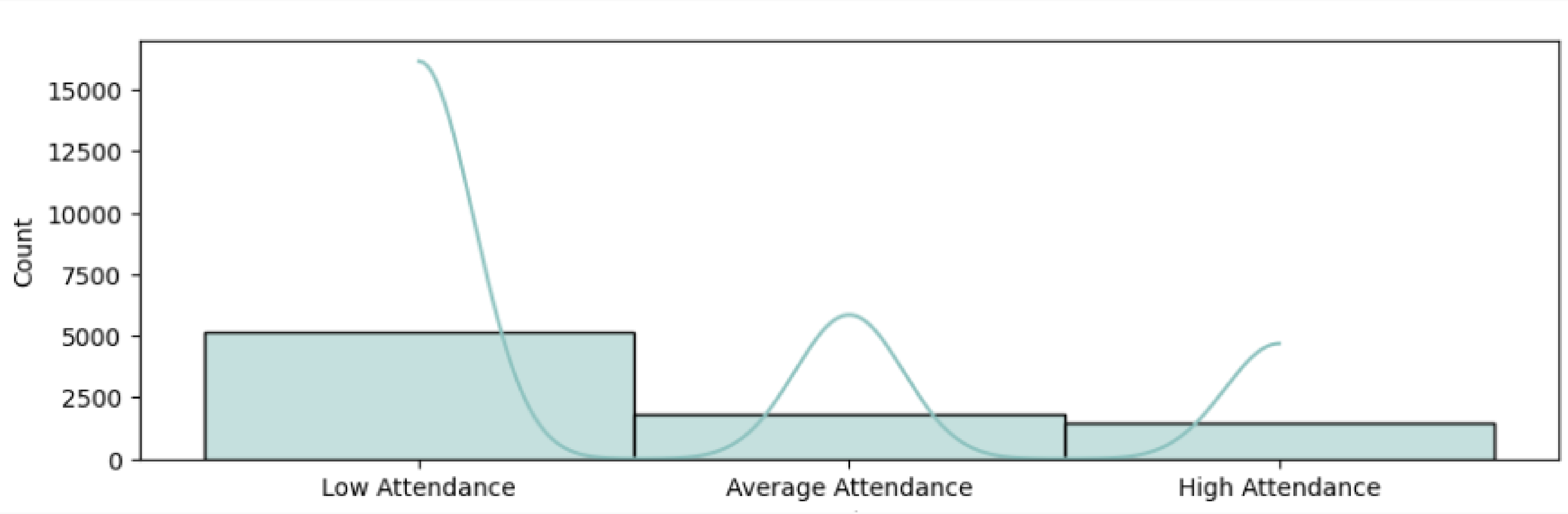
```python
def attendance(rented):
  if rented>=2 and rented<=214 :
    return 'Low Attendance'
  if rented>214 and rented<=542:
    return 'Average Low Attendance'
  if rented>542 and rented<=1084 :
    return 'Average High Attendance'
  if rented>1084 and rented<=3556:
    return 'High Attendance'
```

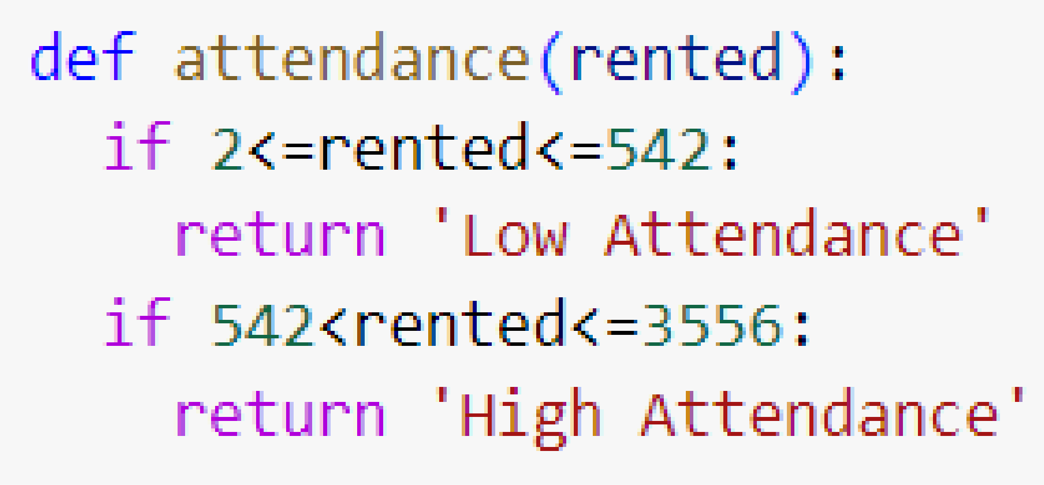


Arbitrary Attendance and Binary Attendance

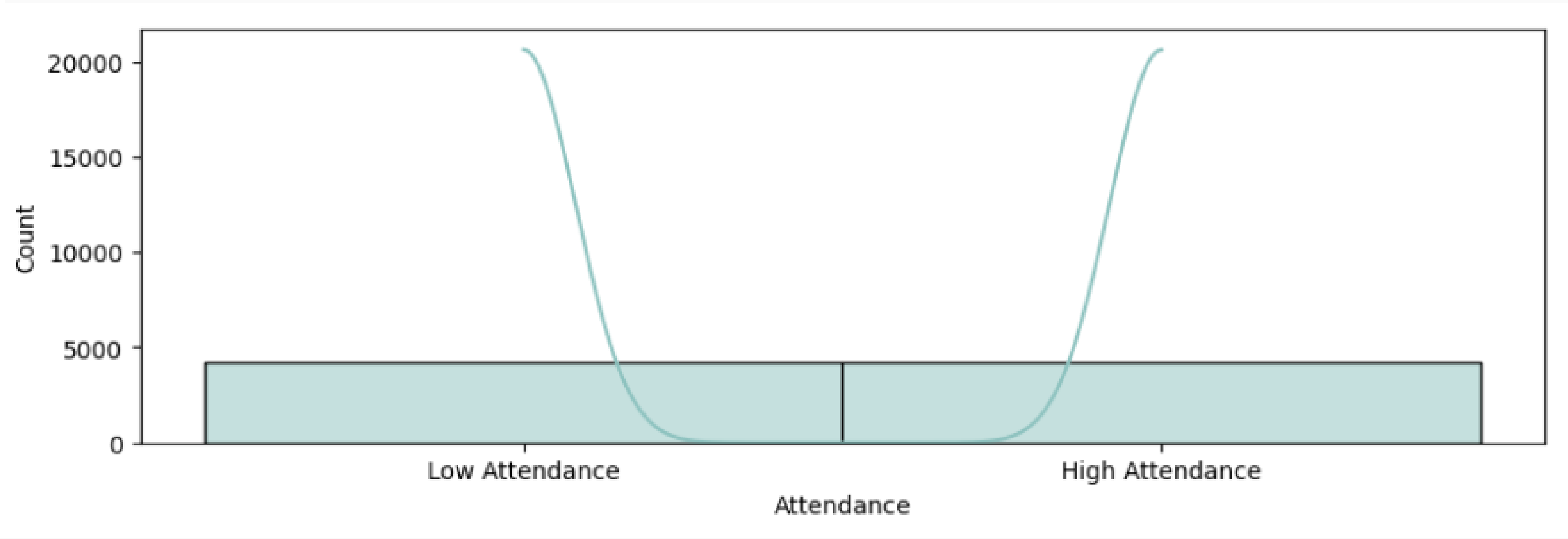


```python
def attendance(rented):
  if 2<=rented<=750:
    return 'Low Attendance'
  if 750<rented<=1300:
    return 'Average Attendance'
  if 1300<rented<=3556:
    return 'High Attendance'
```



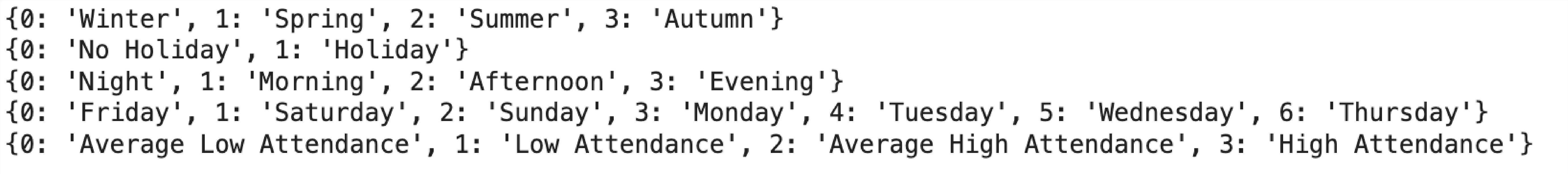


```python
def attendance(rented):
  if 2<=rented<=542:
    return 'Low Attendance'
  if 542<rented<=3556:
    return 'High Attendance'
```



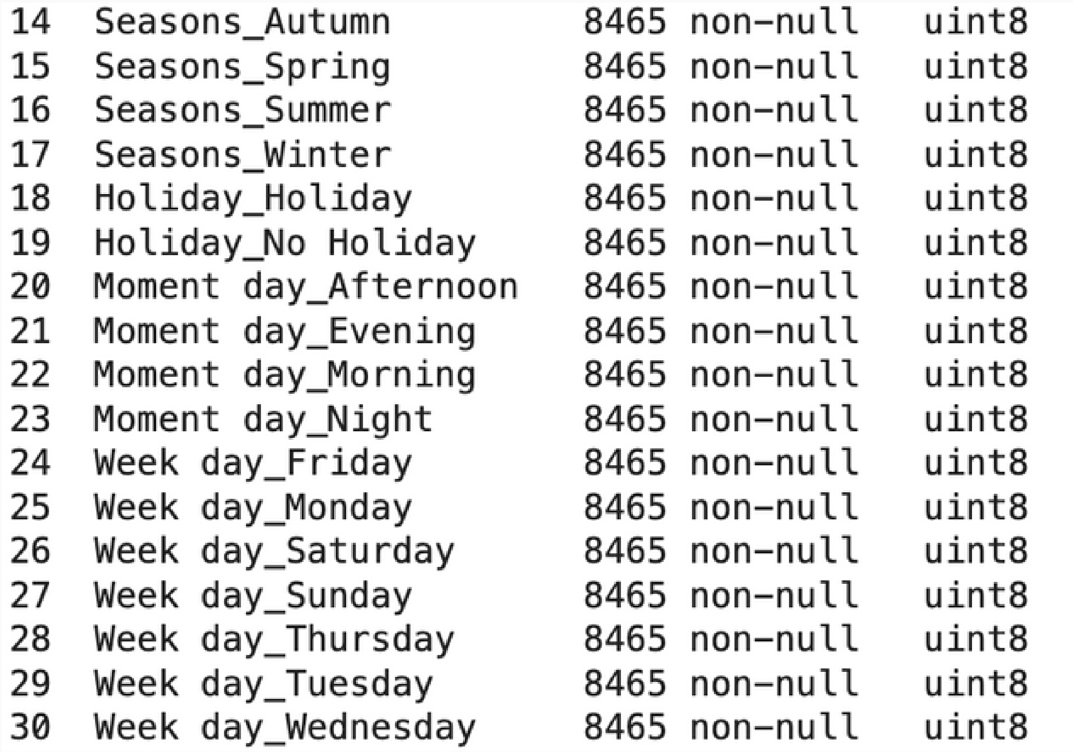
Data Encoding

With OrdinalEncoder



```
{0: 'Winter', 1: 'Spring', 2: 'Summer', 3: 'Autumn'}
{0: 'No Holiday', 1: 'Holiday'}
{0: 'Night', 1: 'Morning', 2: 'Afternoon', 3: 'Evening'}
{0: 'Friday', 1: 'Saturday', 2: 'Sunday', 3: 'Monday', 4: 'Tuesday', 5: 'Wednesday', 6: 'Thursday'}
{0: 'Average Low Attendance', 1: 'Low Attendance', 2: 'Average High Attendance', 3: 'High Attendance'}
```

With Get Dummies

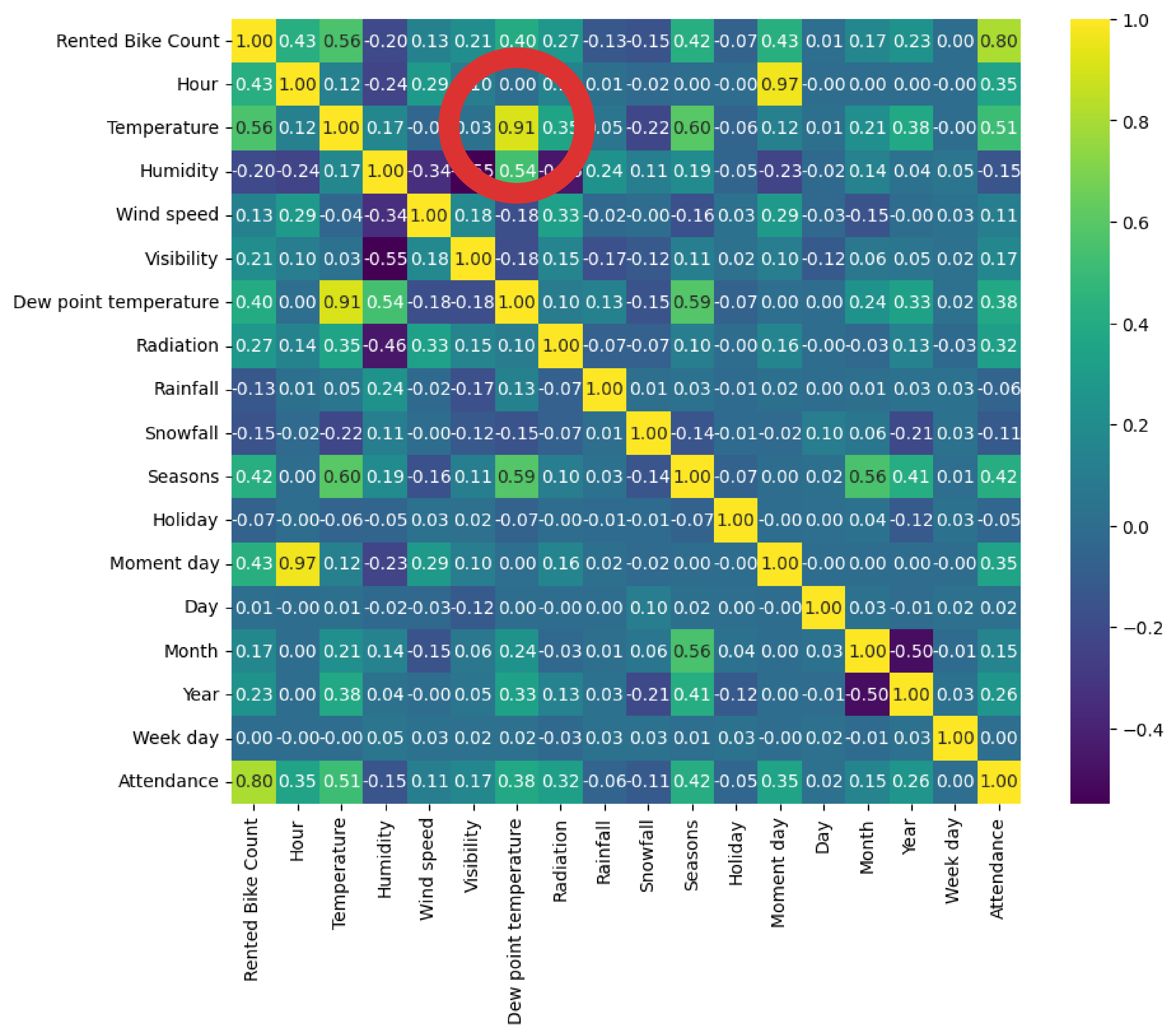


```
14  Seasons_Autumn          8465 non-null   uint8
15  Seasons_Spring          8465 non-null   uint8
16  Seasons_Summer          8465 non-null   uint8
17  Seasons_Winter          8465 non-null   uint8
18  Holiday_Holiday         8465 non-null   uint8
19  Holiday_No Holiday      8465 non-null   uint8
20  Moment day_Afternoon    8465 non-null   uint8
21  Moment day_Evening      8465 non-null   uint8
22  Moment day_Morning      8465 non-null   uint8
23  Moment day_Night        8465 non-null   uint8
24  Week day_Friday         8465 non-null   uint8
25  Week day_Monday         8465 non-null   uint8
26  Week day_Saturday       8465 non-null   uint8
27  Week day_Sunday         8465 non-null   uint8
28  Week day_Thursday       8465 non-null   uint8
29  Week day_Tuesday        8465 non-null   uint8
30  Week day_Wednesday      8465 non-null   uint8
```

Creation of new columns with name of classes

They are filled with 0s and 1s

Data Visualization - Correlation Matrix

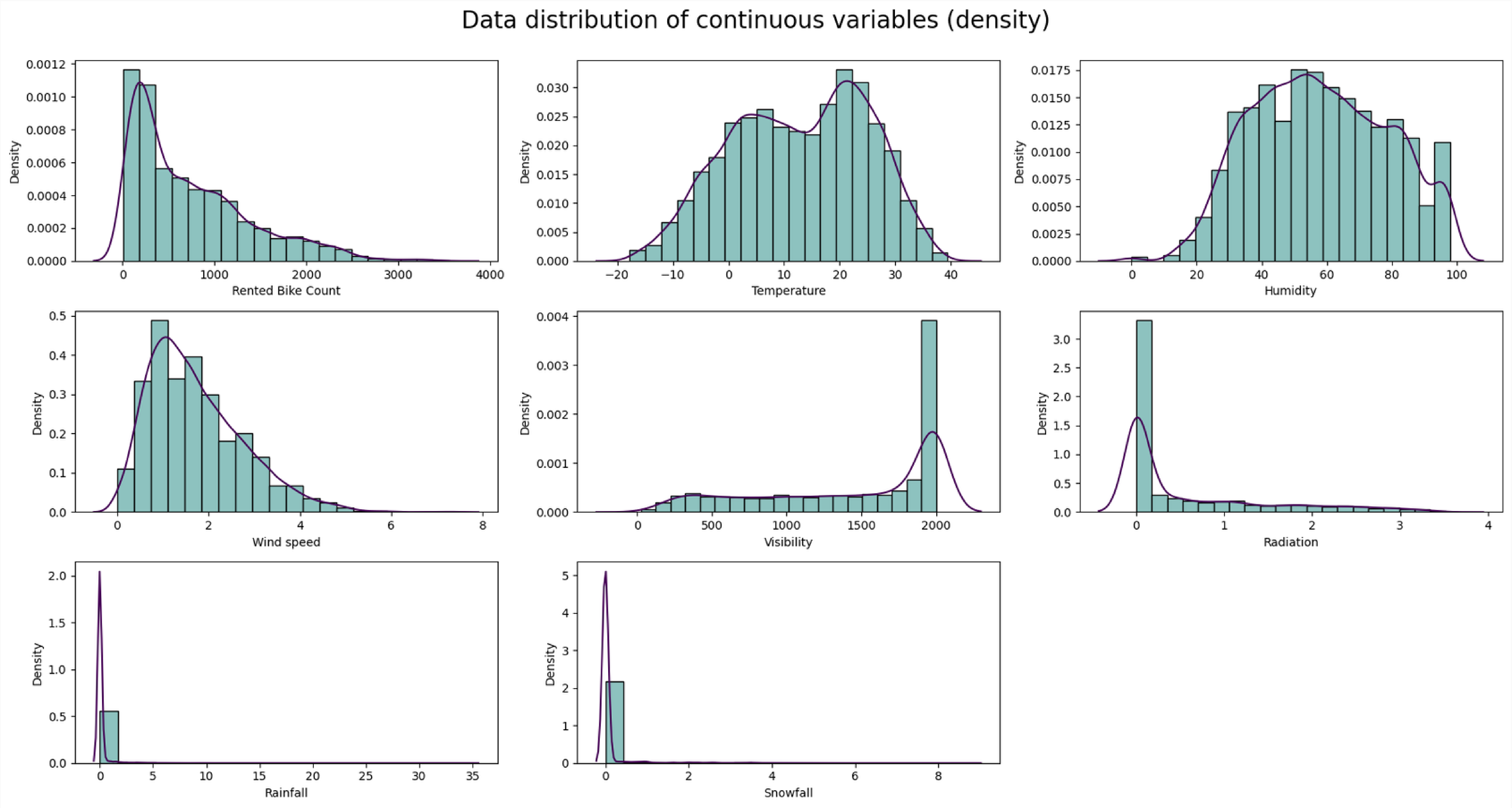


	Rented Bike Count	Hour	Temperature	Humidity	Wind speed	Visibility	Dew point temperature	Radiation	Rainfall	Snowfall	Seasons	Holiday	Moment day	Day	Month	Year	Week day	Attendance
Rented Bike Count	1.00	0.43	0.56	-0.20	0.13	0.21	0.40	0.27	-0.13	-0.15	0.42	-0.07	0.43	0.01	0.17	0.23	0.00	0.80
Hour	0.43	1.00	0.12	-0.24	0.29	0.10	0.00	[illegible]	0.01	-0.02	0.00	-0.00	0.97	-0.00	0.00	0.00	-0.00	0.35
Temperature	0.56	0.12	1.00	0.17	[illegible]	0.03	0.91	0.35	0.05	-0.22	0.60	-0.06	0.12	0.01	0.21	0.38	-0.00	0.51
Humidity	-0.20	-0.24	0.17	1.00	-0.34	[illegible]	0.54	[illegible]	0.24	0.11	0.19	-0.05	-0.23	-0.02	0.14	0.04	0.05	-0.15
Wind speed	0.13	0.29	-0.04	-0.34	1.00	0.18	-0.18	0.33	-0.02	-0.00	-0.16	0.03	0.29	-0.03	-0.15	-0.00	0.03	0.11
Visibility	0.21	0.10	0.03	-0.55	0.18	1.00	-0.18	0.15	-0.17	-0.12	0.11	0.02	0.10	-0.12	0.06	0.05	0.02	0.17
Dew point temperature	0.40	0.00	0.91	0.54	-0.18	-0.18	1.00	0.10	0.13	-0.15	0.59	-0.07	0.00	0.00	0.24	0.33	0.02	0.38
Radiation	0.27	0.14	0.35	-0.46	0.33	0.15	0.10	1.00	-0.07	-0.07	0.10	-0.00	0.16	-0.00	-0.03	0.13	-0.03	0.32
Rainfall	-0.13	0.01	0.05	0.24	-0.02	-0.17	0.13	-0.07	1.00	0.01	0.03	-0.01	0.02	0.00	0.01	0.03	0.03	-0.06
Snowfall	-0.15	-0.02	-0.22	0.11	-0.00	-0.12	-0.15	-0.07	0.01	1.00	-0.14	-0.01	-0.02	0.10	0.06	-0.21	0.03	-0.11
Seasons	0.42	0.00	0.60	0.19	-0.16	0.11	0.59	0.10	0.03	-0.14	1.00	-0.07	0.00	0.02	0.56	0.41	0.01	0.42
Holiday	-0.07	-0.00	-0.06	-0.05	0.03	0.02	-0.07	-0.00	-0.01	-0.01	-0.07	1.00	-0.00	0.00	0.04	-0.12	0.03	-0.05
Moment day	0.43	0.97	0.12	-0.23	0.29	0.10	0.00	0.16	0.02	-0.02	0.00	-0.00	1.00	-0.00	0.00	0.00	-0.00	0.35
Day	0.01	-0.00	0.01	-0.02	-0.03	-0.12	0.00	-0.00	0.00	0.10	0.02	0.00	-0.00	1.00	0.03	-0.01	0.02	0.02
Month	0.17	0.00	0.21	0.14	-0.15	0.06	0.24	-0.03	0.01	0.06	0.56	0.04	0.00	0.03	1.00	-0.50	-0.01	0.15
Year	0.23	0.00	0.38	0.04	-0.00	0.05	0.33	0.13	0.03	-0.21	0.41	-0.12	0.00	-0.01	-0.50	1.00	0.03	0.26
Week day	0.00	-0.00	-0.00	0.05	0.03	0.02	0.02	-0.03	0.03	0.03	0.01	0.03	-0.00	0.02	-0.01	0.03	1.00	0.00
Attendance	0.80	0.35	0.51	-0.15	0.11	0.17	0.38	0.32	-0.06	-0.11	0.42	-0.05	0.35	0.02	0.15	0.26	0.00	1.00

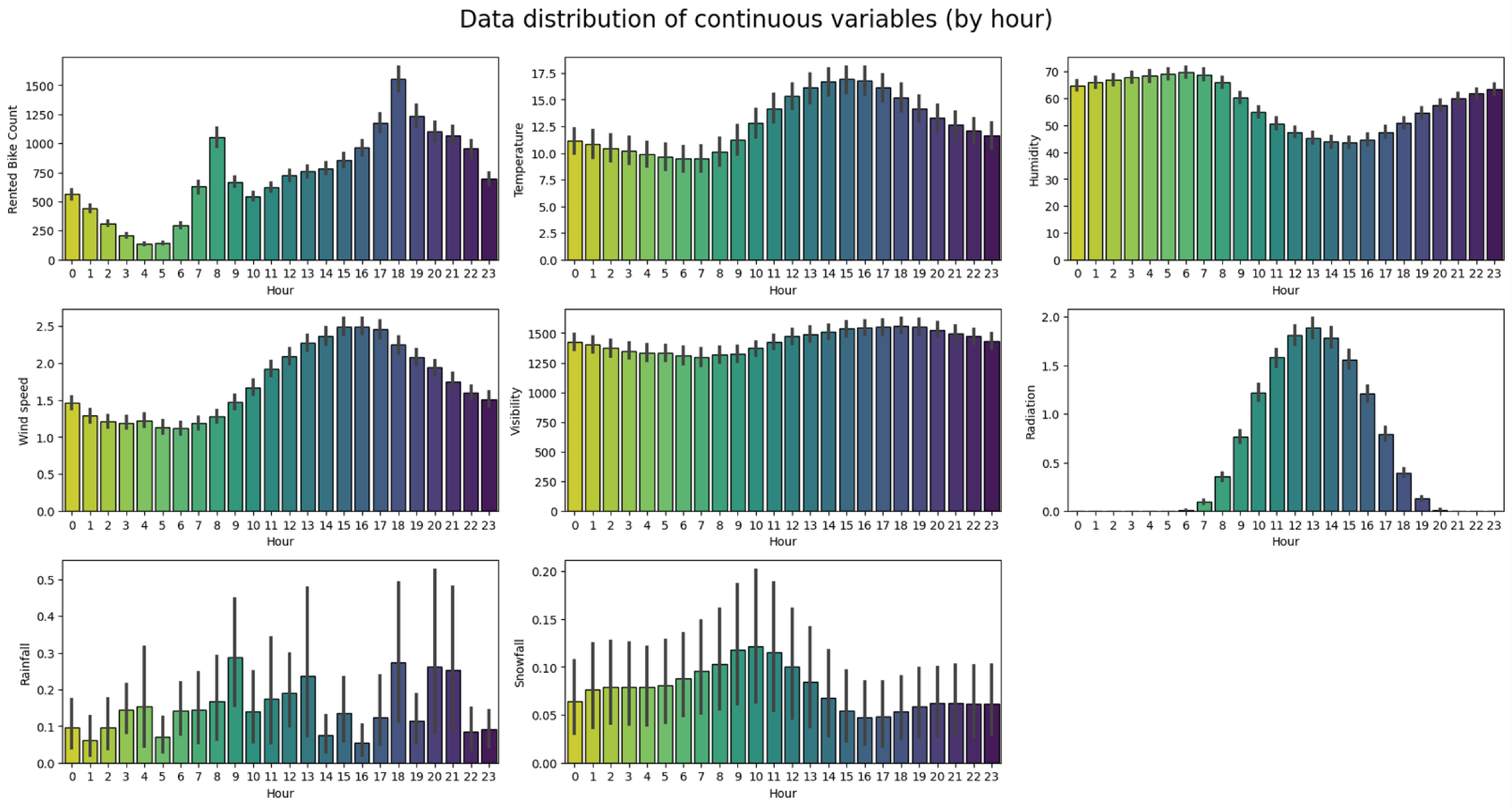
Strong correlation between Temperature and Dew Point Temperature

We delete the column Dew Point Temperature as Temperature is more relevent

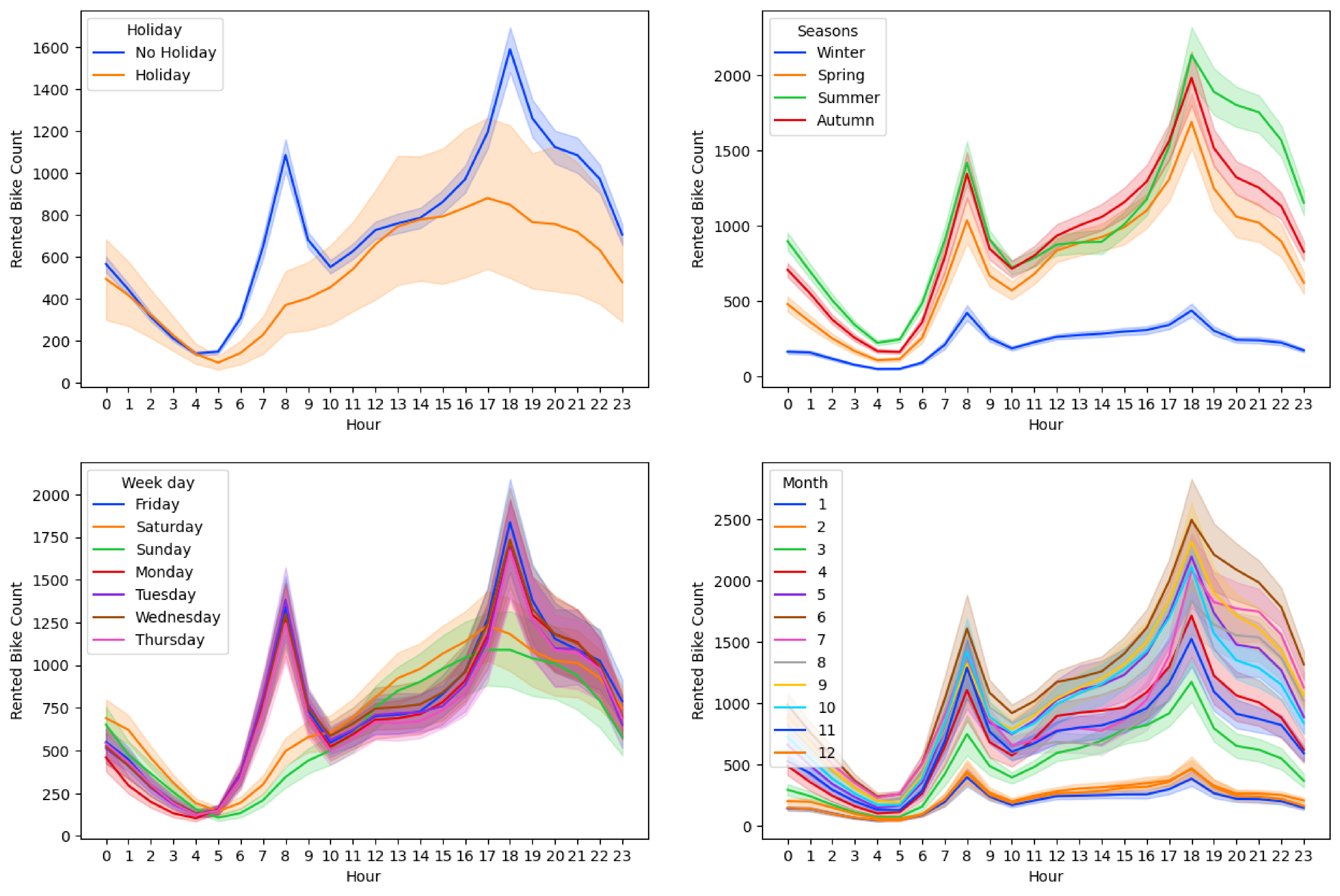
Data Visualization - Data Distribution



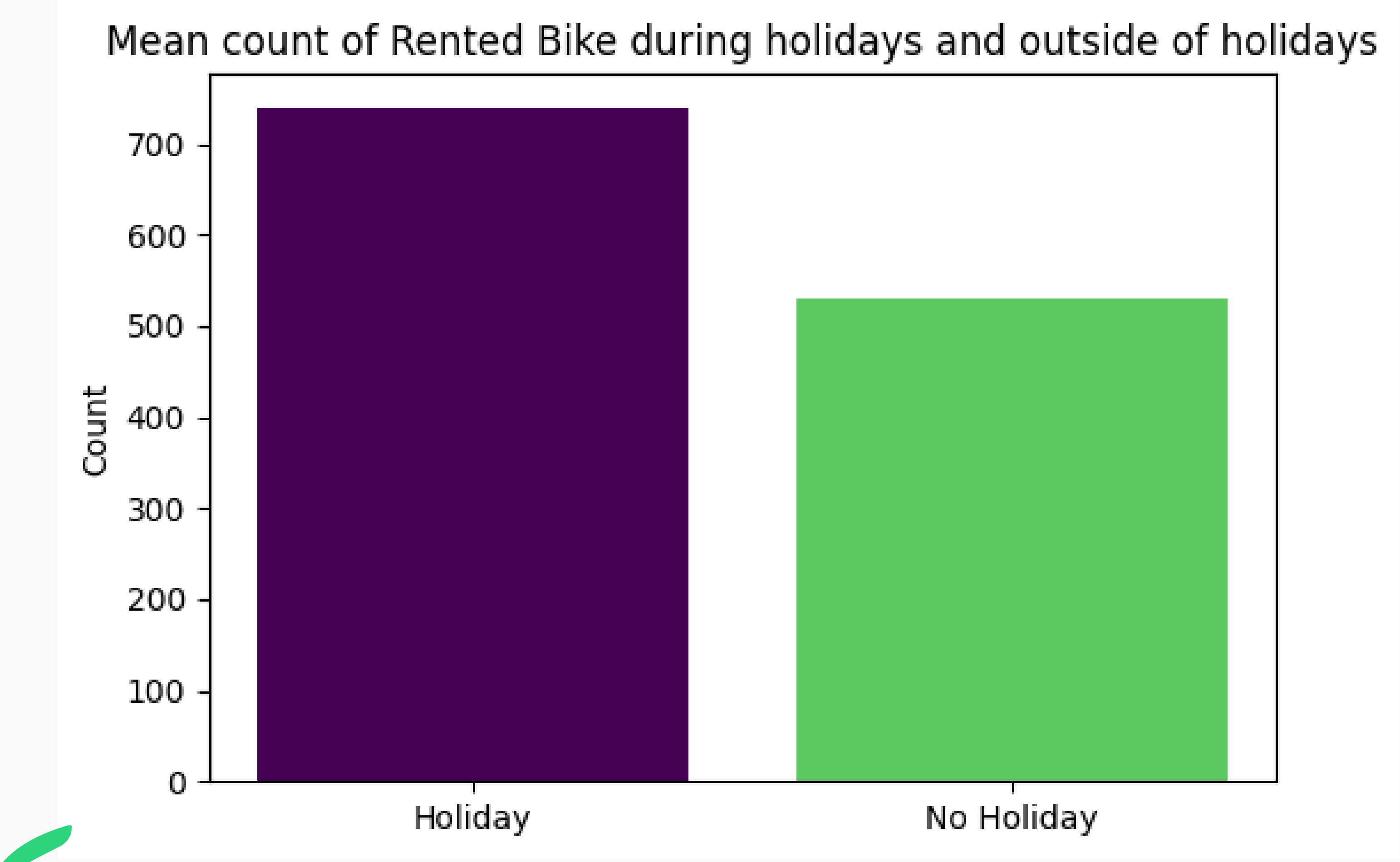
Data Visualization - Data Distribution



Data Visualization - Data Trends

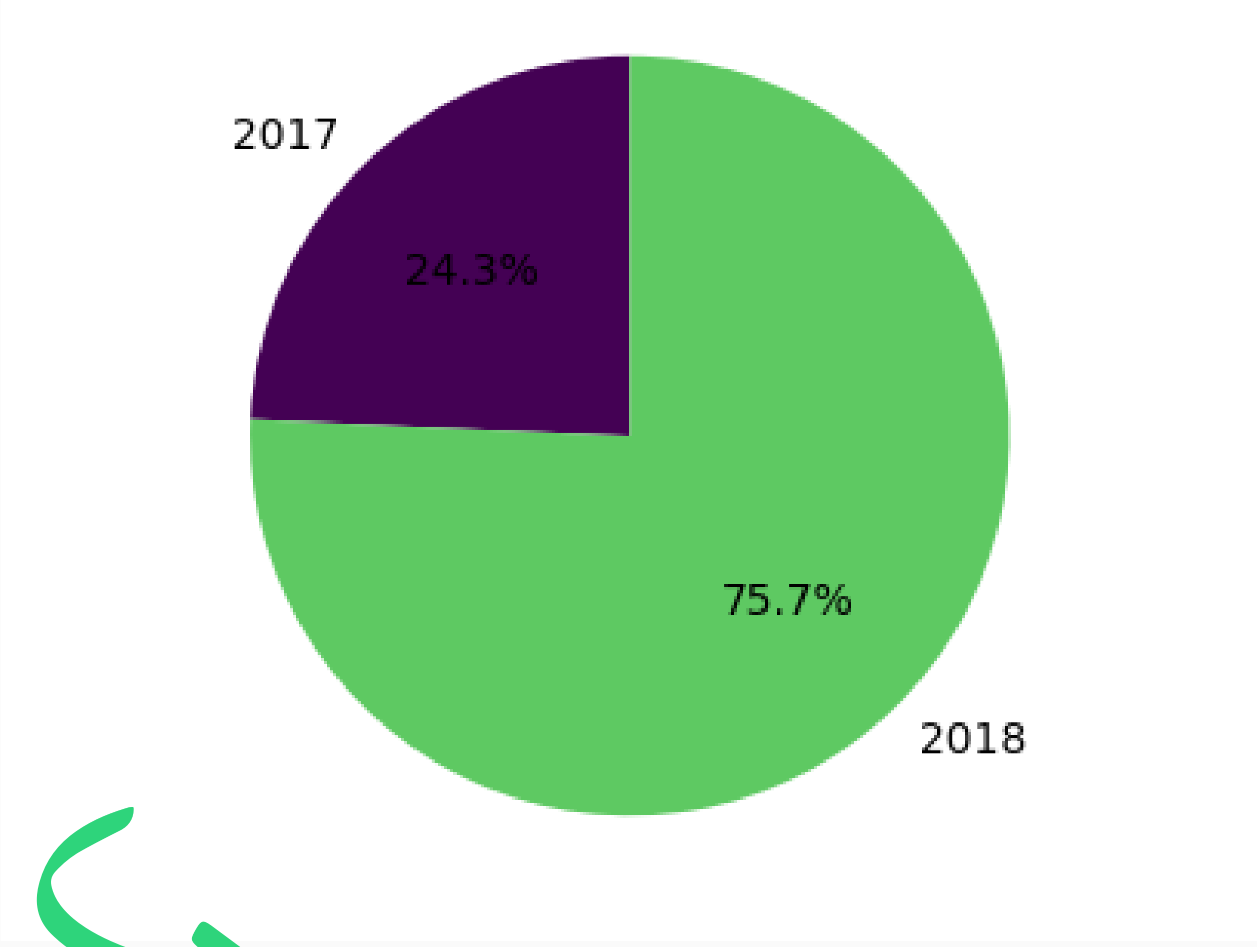


Data Visualization - Data Trends



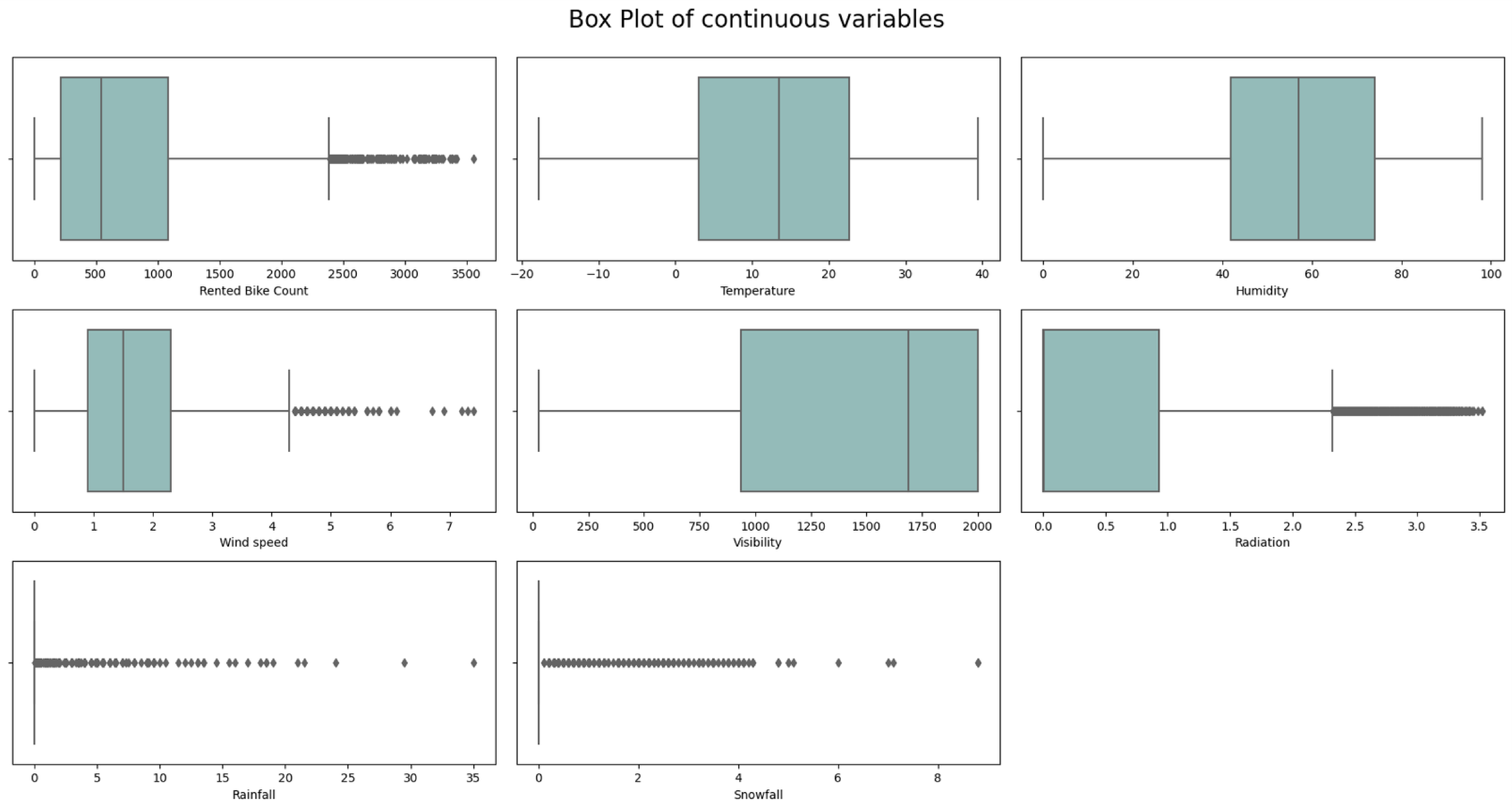
The bikes are used almost as much for relaxing during vacations as for getting to work

Average number of bikes rented per year



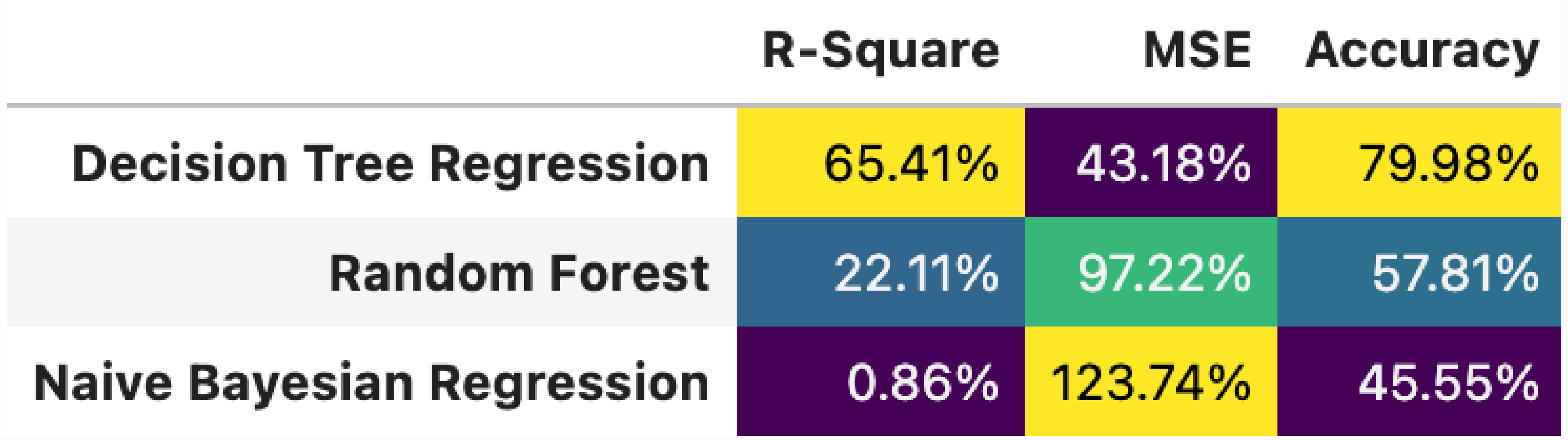
We delete the column Year

Data Visualization - Box Plots



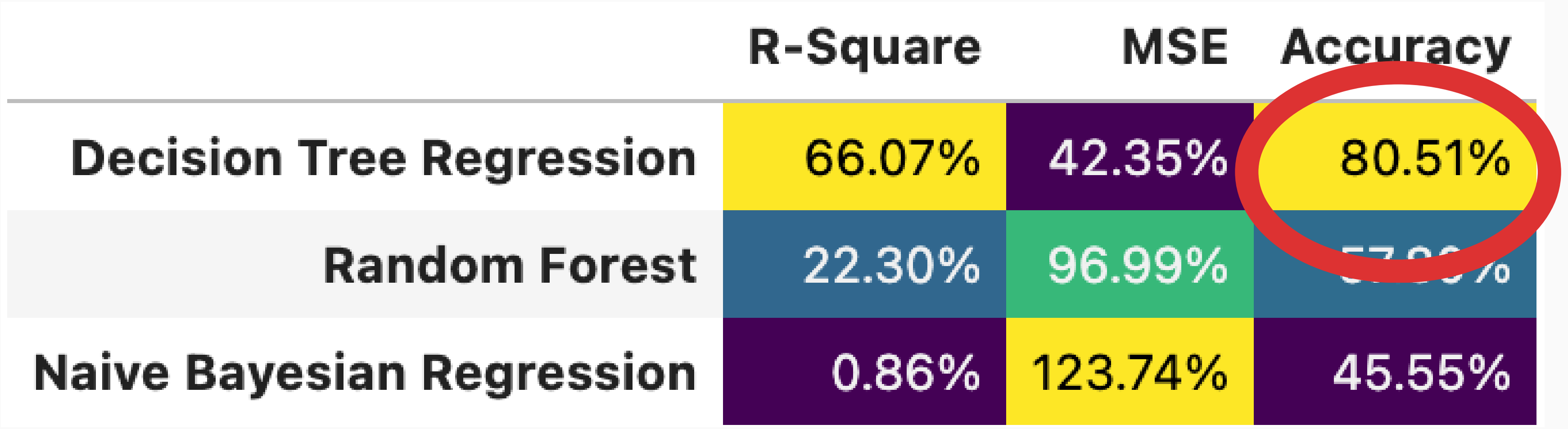
Get Dummies : Why did we choose this encoding?

With OrdinalEncoder



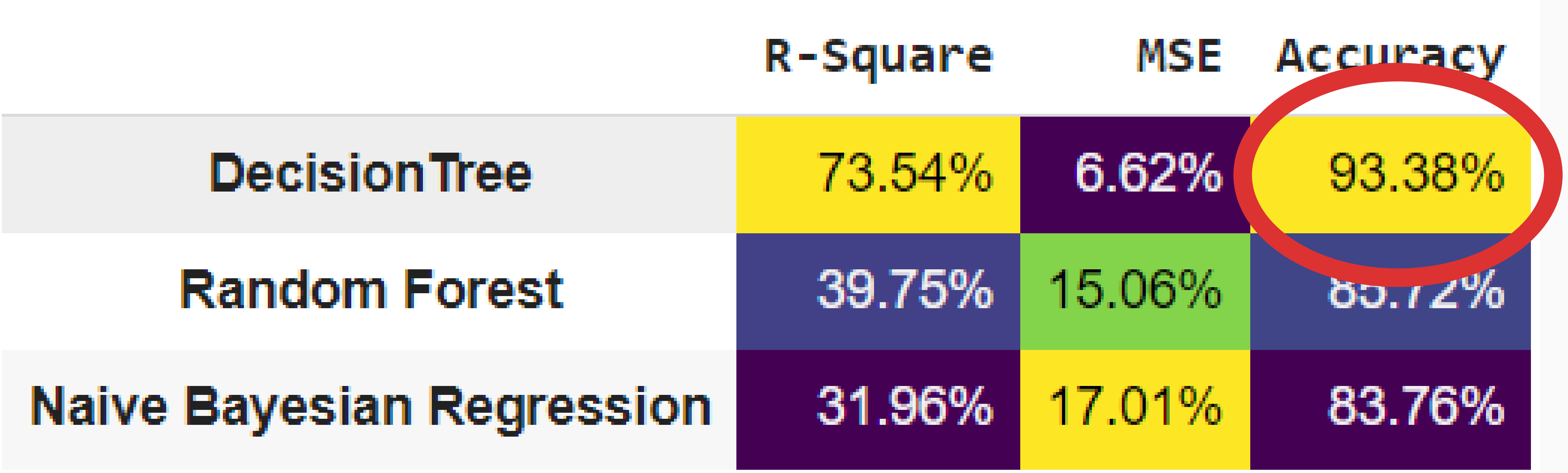
	R-Square	MSE	Accuracy
Decision Tree Regression	65.41%	43.18%	79.98%
Random Forest	22.11%	97.22%	57.81%
Naive Bayesian Regression	0.86%	123.74%	45.55%

With Get Dummies



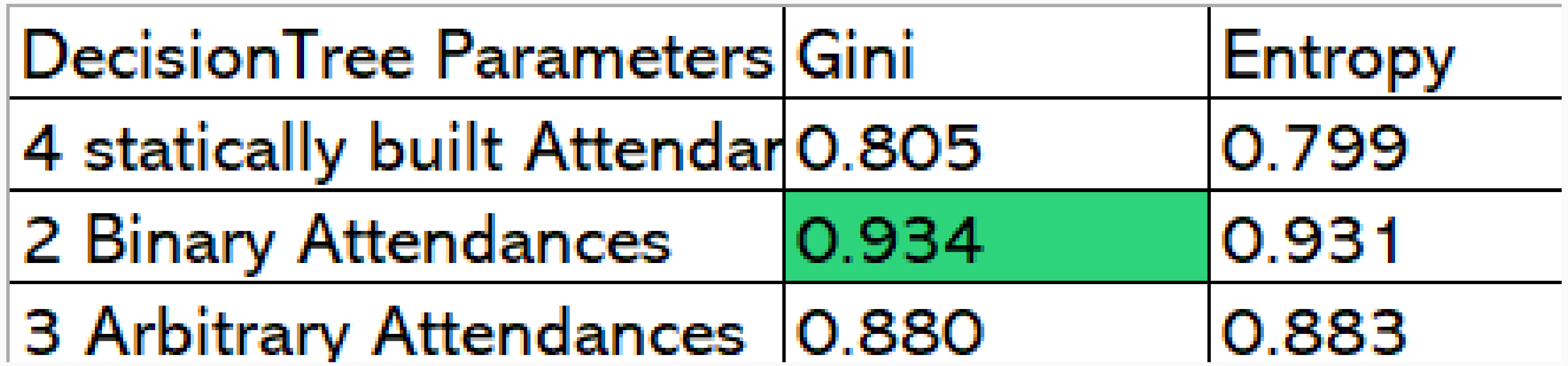
	R-Square	MSE	Accuracy
Decision Tree Regression	66.07%	42.35%	80.51%
Random Forest	22.30%	96.99%	57.90%
Naive Bayesian Regression	0.86%	123.74%	45.55%

Decision Tree Regression: Why did we choose this algorithm?

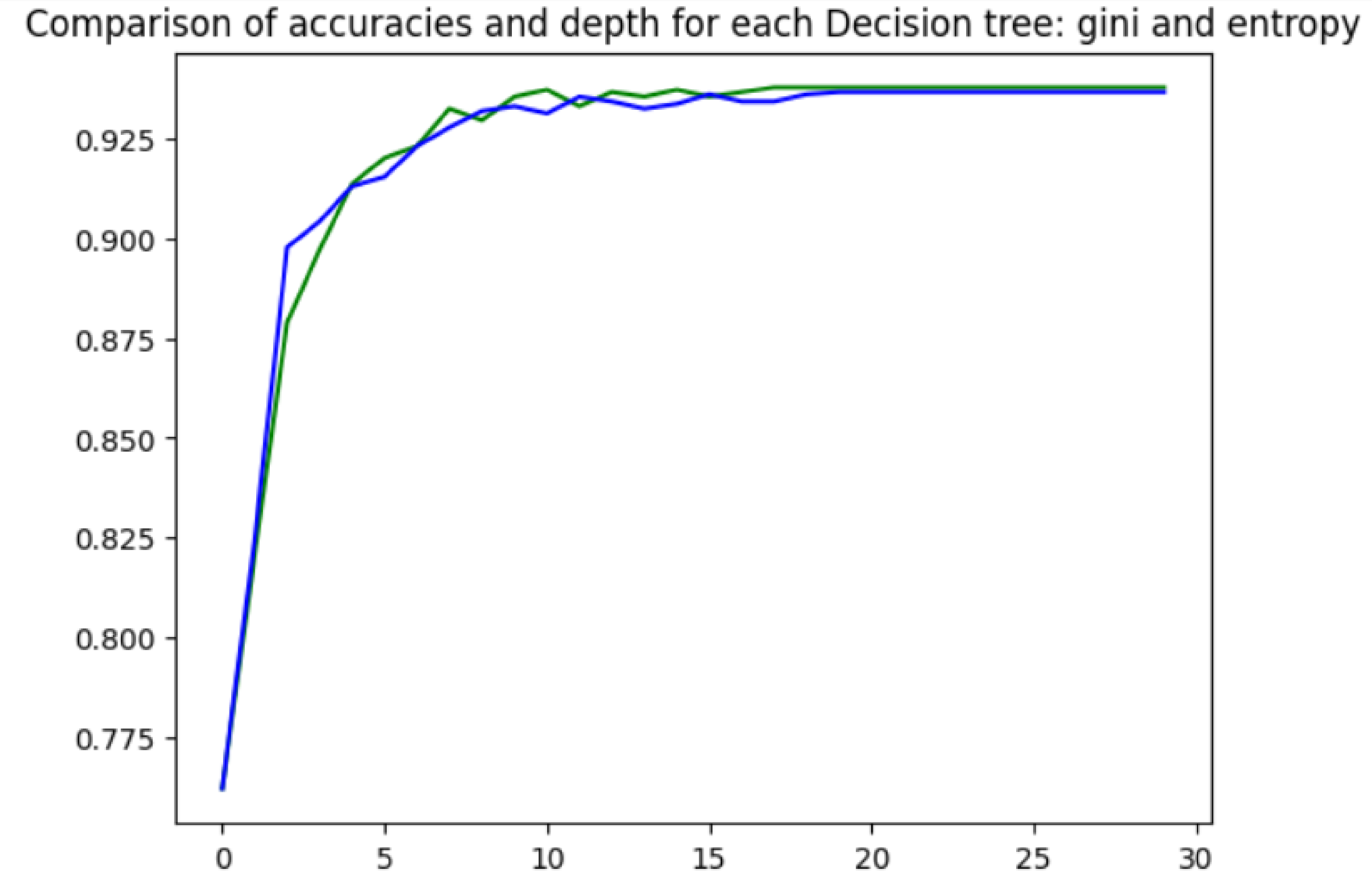


	R-Square	MSE	Accuracy
DecisionTree	73.54%	6.62%	93.38%
Random Forest	39.75%	15.06%	85.72%
Naive Bayesian Regression	31.96%	17.01%	83.76%

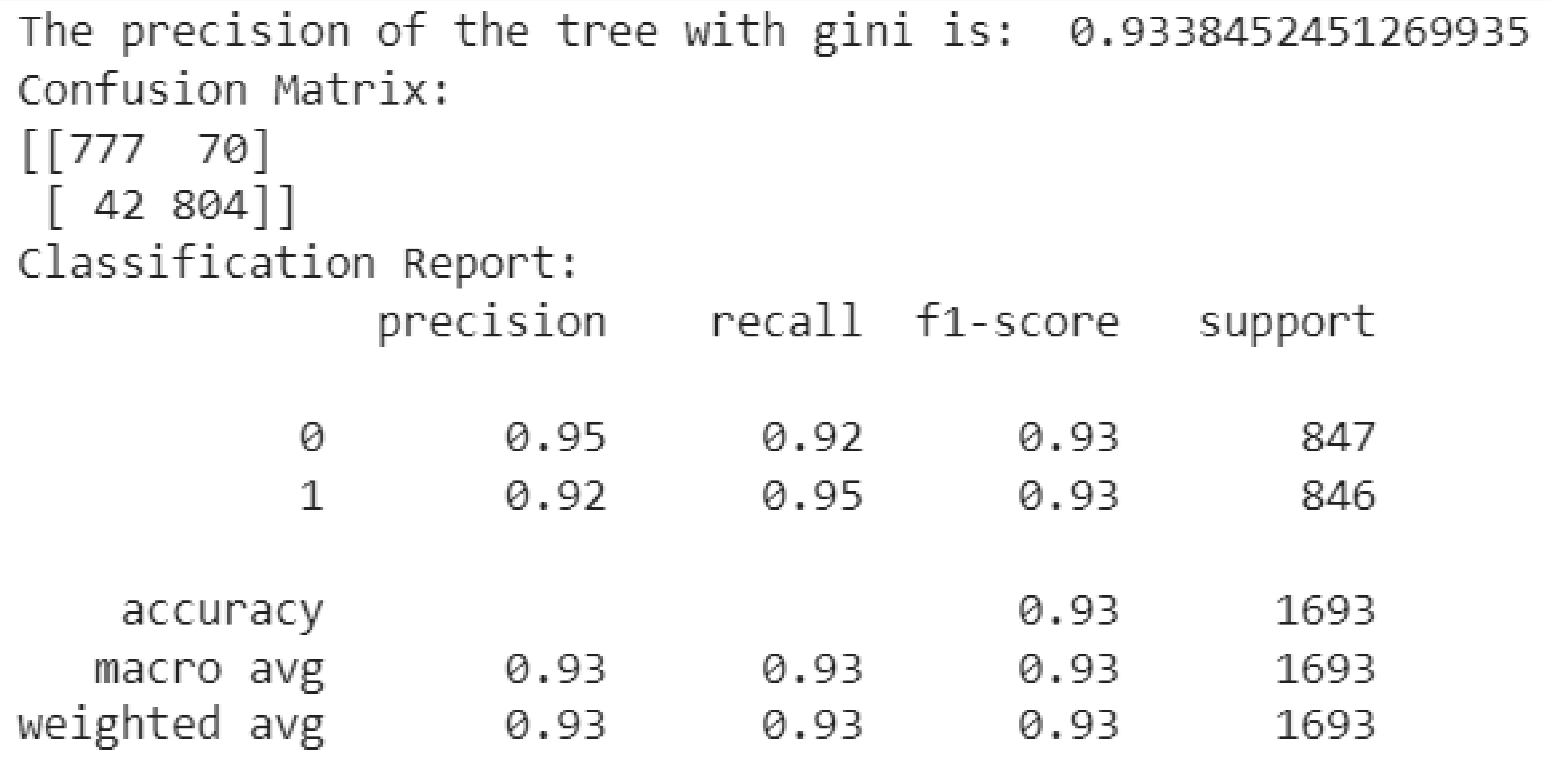
Modeling with Decision Tree : Parameters choices



DecisionTree Parameters	Gini	Entropy
4 statically built Attendar	0.805	0.799
2 Binary Attendances	0.934	0.931
3 Arbitrary Attendances	0.880	0.883



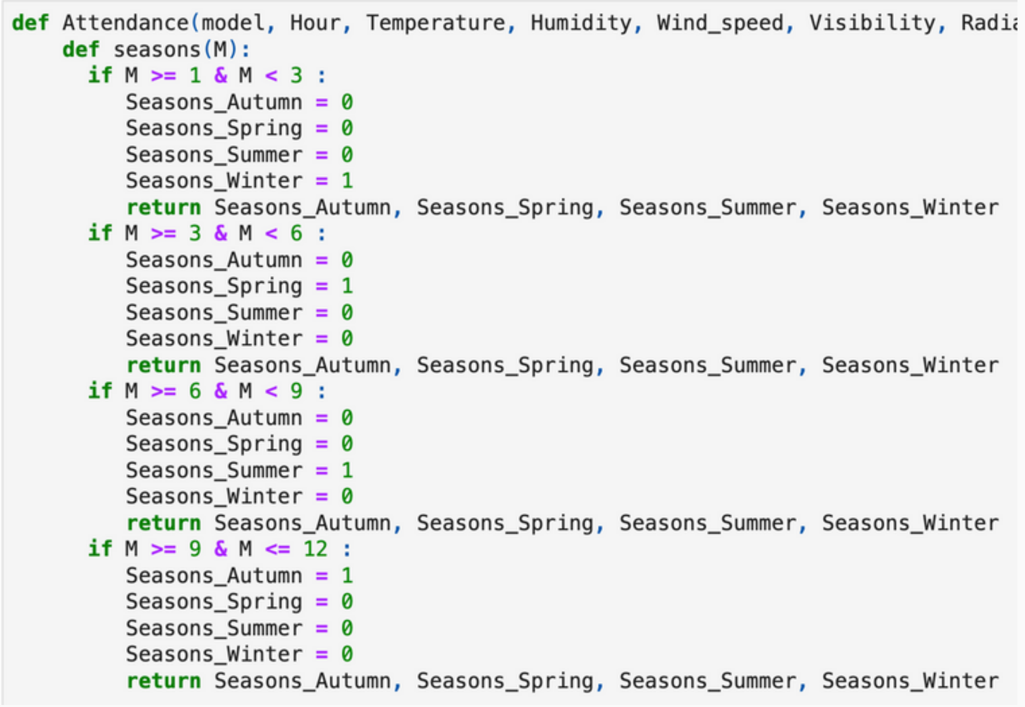
Modeling with Decision Tree “gini” : Results



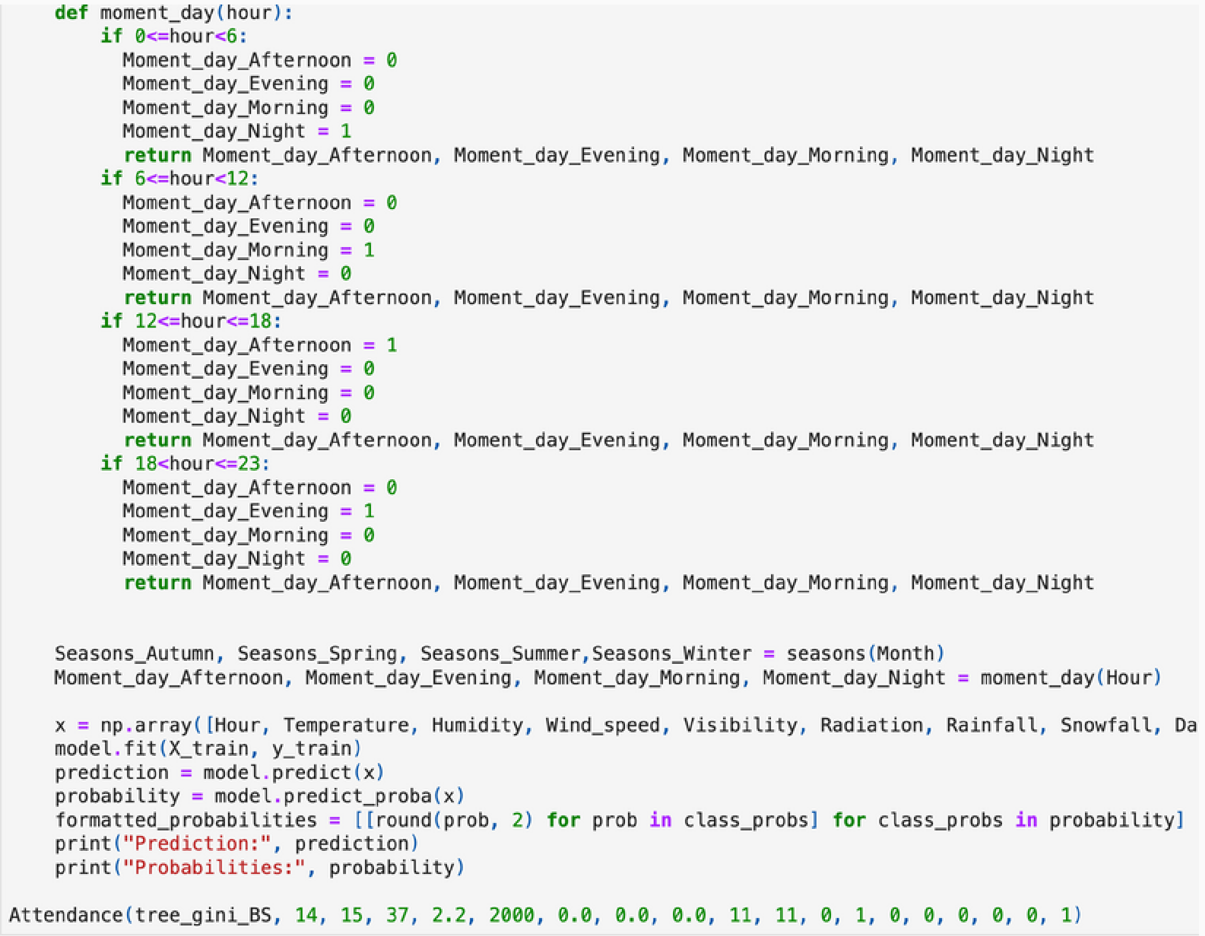
```
The precision of the tree with gini is:  0.9338452451269935
Confusion Matrix:
[[777  70]
 [ 42 804]]
Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.92      0.93       847
           1       0.92      0.95      0.93       846

    accuracy                           0.93      1693
   macro avg       0.93      0.93      0.93      1693
weighted avg       0.93      0.93      0.93      1693
```

- Our choice: “Being sure of our prediction”
- The quality of the information is lower than for 4 types of attendance
- The accuracy is much better

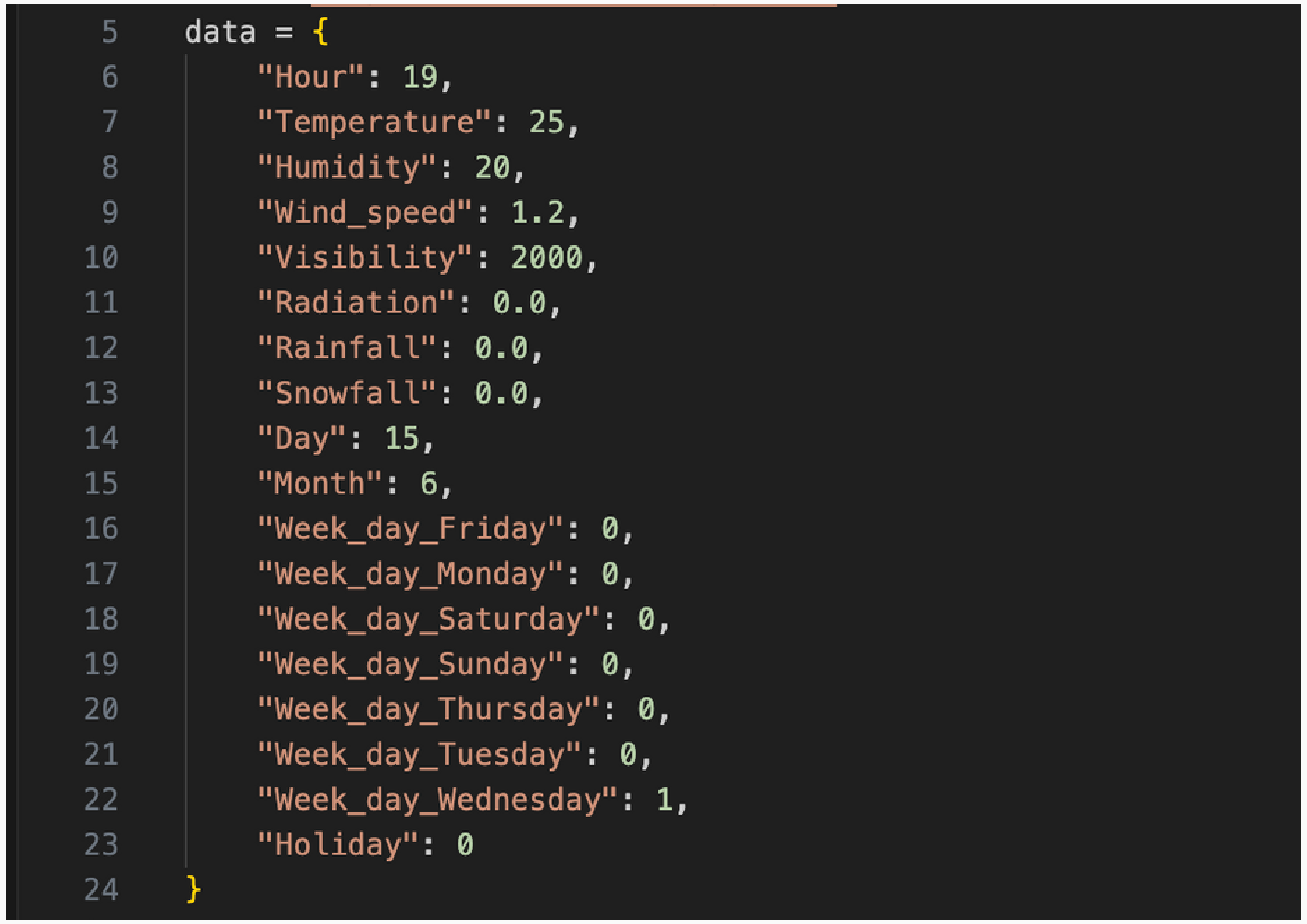
API



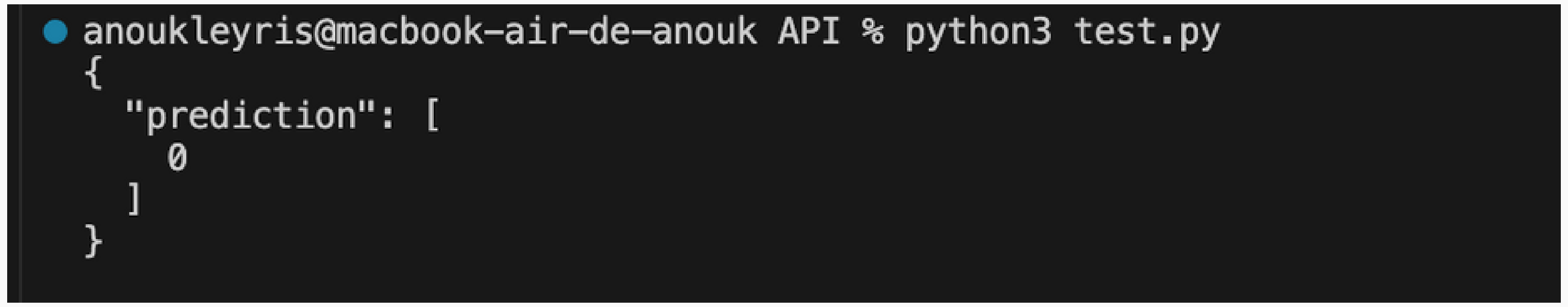
```python
def Attendance(model, Hour, Temperature, Humidity, Wind_speed, Visibility, Radia
    def seasons(M):
      if M >= 1 & M < 3 :
         Seasons_Autumn = 0
         Seasons_Spring = 0
         Seasons_Summer = 0
         Seasons_Winter = 1
         return Seasons_Autumn, Seasons_Spring, Seasons_Summer, Seasons_Winter
      if M >= 3 & M < 6 :
         Seasons_Autumn = 0
         Seasons_Spring = 1
         Seasons_Summer = 0
         Seasons_Winter = 0
         return Seasons_Autumn, Seasons_Spring, Seasons_Summer, Seasons_Winter
      if M >= 6 & M < 9 :
         Seasons_Autumn = 0
         Seasons_Spring = 0
         Seasons_Summer = 1
         Seasons_Winter = 0
         return Seasons_Autumn, Seasons_Spring, Seasons_Summer, Seasons_Winter
      if M >= 9 & M <= 12 :
         Seasons_Autumn = 1
         Seasons_Spring = 0
         Seasons_Summer = 0
         Seasons_Winter = 0
         return Seasons_Autumn, Seasons_Spring, Seasons_Summer, Seasons_Winter
```



```python
    def moment_day(hour):
        if 0<=hour<6:
          Moment_day_Afternoon = 0
          Moment_day_Evening = 0
          Moment_day_Morning = 0
          Moment_day_Night = 1
          return Moment_day_Afternoon, Moment_day_Evening, Moment_day_Morning, Moment_day_Night
        if 6<=hour<12:
          Moment_day_Afternoon = 0
          Moment_day_Evening = 0
          Moment_day_Morning = 1
          Moment_day_Night = 0
          return Moment_day_Afternoon, Moment_day_Evening, Moment_day_Morning, Moment_day_Night
        if 12<=hour<=18:
          Moment_day_Afternoon = 1
          Moment_day_Evening = 0
          Moment_day_Morning = 0
          Moment_day_Night = 0
          return Moment_day_Afternoon, Moment_day_Evening, Moment_day_Morning, Moment_day_Night
        if 18<hour<=23:
          Moment_day_Afternoon = 0
          Moment_day_Evening = 1
          Moment_day_Morning = 0
          Moment_day_Night = 0
          return Moment_day_Afternoon, Moment_day_Evening, Moment_day_Morning, Moment_day_Night


    Seasons_Autumn, Seasons_Spring, Seasons_Summer,Seasons_Winter = seasons(Month)
    Moment_day_Afternoon, Moment_day_Evening, Moment_day_Morning, Moment_day_Night = moment_day(Hour)

    x = np.array([Hour, Temperature, Humidity, Wind_speed, Visibility, Radiation, Rainfall, Snowfall, Da
    model.fit(X_train, y_train)
    prediction = model.predict(x)
    probability = model.predict_proba(x)
    formatted_probabilities = [[round(prob, 2) for prob in class_probs] for class_probs in probability]
    print("Prediction:", prediction)
    print("Probabilities:", probability)

Attendance(tree_gini_BS, 14, 15, 37, 2.2, 2000, 0.0, 0.0, 0.0, 11, 11, 0, 1, 0, 0, 0, 0, 0, 1)
```



```python
data = {
    "Hour": 19,
    "Temperature": 25,
    "Humidity": 20,
    "Wind_speed": 1.2,
    "Visibility": 2000,
    "Radiation": 0.0,
    "Rainfall": 0.0,
    "Snowfall": 0.0,
    "Day": 15,
    "Month": 6,
    "Week_day_Friday": 0,
    "Week_day_Monday": 0,
    "Week_day_Saturday": 0,
    "Week_day_Sunday": 0,
    "Week_day_Thursday": 0,
    "Week_day_Tuesday": 0,
    "Week_day_Wednesday": 1,
    "Holiday": 0
}
```



```
● anoukleyris@macbook-air-de-anouk API % python3 test.py
  {
    "prediction": [
      0
    ]
  }
```

Conclusion

Today, we are able to provide the following information:

Giving the climate measures of a day, the hour and the date, we can say, using DecisionTree with 90% of accuracy, if the attendance of the rented bike system in Seoul is low or high.

With more than 80% of accuracy and the same algorithm, we can even detail the attendance of the rent with 4 degrees based on the stats of rented bike number that we have.