HW3-dataModel-Romain_Blanchot

February 11, 2025

1 Homework 3: Data Preparation

CPE232 Data Models

1.1 Project setup

[16]:

```
# !pip install matplotlib
```

[17]:

```
import pandas as pd

df = pd.read_csv('bike_sharing_demand.csv')
```

[18]:

```
df.head()
```

[18]:

```
   season  year  month  hour  holiday  weekday  workingday weather  temp  \
0  spring     0      1     0    False        6       False   clear  9.84
1  spring     0      1     1    False        6       False   clear  9.02
2  spring     0      1     2    False        6       False   clear  9.02
3  spring     0      1     3    False        6       False   clear  9.84
4  spring     0      1     4    False        6       False   clear  9.84

   feel_temp  humidity  windspeed  count
0     14.395      0.81        0.0     16
1     13.635       NaN        0.0     40
2     13.635      0.80        0.0     32
3     14.395      0.75        0.0     13
4     14.395      0.75        0.0      1
```

[19]:

```
url = "https://kmutt.me/"
```

1.2 The Secret URL Challenge!

Welcome, brave explorer! Your mission, should you choose to accept it, is to uncover a hidden phrase scattered across the questions below. Each question holds a vital clue—a word or phrase—that will bring you closer to unlocking the **Secret URL**!

Once you have gathered all the hidden words, combine them **in order** and attach them to this URL:
`https://kmutt.me/[your_combined_phrase]`

For example, if you discover the words `['quest', 'begin']`, your final URL will be:
`https://kmutt.me/questbegin`

Are you ready to solve the mystery and reveal the secret link? Let the adventure begin!

[20]:

```
df.describe()
```

[20]:

```
        year  month       hour    weekday       temp  feel_temp  \
count  200.0  200.0  200.000000  200.000000  200.000000  200.000000
mean     0.0    1.0   11.455000    3.160000    9.389000   11.689600
std      0.0    0.0    6.832377    2.235933    3.713618    4.580663
min      0.0    1.0    0.000000    0.000000    3.280000    3.030000
25%      0.0    1.0    6.000000    1.000000    6.560000    9.090000
50%      0.0    1.0   11.000000    3.000000    8.200000   10.985000
75%      0.0    1.0   17.000000    5.000000   10.660000   13.635000
max      0.0    1.0   23.000000    6.000000   18.860000   22.725000

         humidity   windspeed       count
count  170.000000  200.000000  200.000000
mean     0.559059   13.745452   53.950000
std      0.176368    8.637962   48.931472
min      0.280000    0.000000    1.000000
25%      0.422500    7.001500   12.000000
50%      0.510000   12.998000   47.000000
75%      0.690000   19.250775   76.000000
max      1.000000   36.997400  219.000000
```

1.2.1 Clue 1: A Note from the Keeper of the Winds

"Traveler, the first clue hides in the mist! To uncover it, follow these steps carefully:"

1. Find the moment when the wind was strongest during misty weather.
2. Look at that row and gather the numbers hidden in the hour and count columns.
3. Add 65 to each number and turn them into letters. but divide count by 3.
4. Arrange them in the order given by hour and count to reveal the hidden phrase!

"Solve this mystery, and you will take the first step toward unlocking the secret URL!"

Monkey Mode Activated!

1. Ooo ooo! Find rows where weather is 'mist'!

2. Pick the row with the BIGGEST windspeed!

3. Grab hour and count columns and divide count by 3!

4. Add 65 to each number! 6 5

5. Turn those numbers into LETTERS!

Ooo OOO! Secret phrase unlocked!

[21]:

```
print("Valeurs uniques dans weather:", df["weather"].unique())


max_wind_speed_in_misty_weather = df[df["weather"] == "misty"]["windspeed"].
  ↪max()
target_row = df[(df["weather"] == "misty") & (df["windspeed"] ==␣
  ↪max_wind_speed_in_misty_weather)]

print("Nombre de lignes trouvées:", len(target_row))

# get the hour and count of the target row
hour, count = target_row["hour"].values[0] + 65, target_row["count"].values[0]//
  ↪3 + 65

# just change the hour and count to the corresponding ascii character
result = str(chr(hour)) + str(chr(count))

# concatenate the result to the url
url = url + result
print("your current url is: ", url)
```

```
Valeurs uniques dans weather: ['clear' 'misty' 'rain']
Nombre de lignes trouvées: 1
your current url is:  https://kmutt.me/LU
```

1.2.2 Clue 2: The Hidden Words in the Weather

The next piece of the puzzle lies in the unique weathers that were observed! To find the clue:

1. Look at all the different weather conditions recorded in the dataset.
2. Take the last two word of each unique weather type you find.
3. The combination of these words will lead you to the next step in your adventure!
4. Unravel this mystery, and you'll be one step closer to the secret URL!

Monkey Mode

1. Ooo ooo! Find all the different weather types!
2. Get the LAST TWO word of each one!
3. Combine the words to move closer to the secret!

Monkey magic will lead you to the next clue!

[23]:

```
# get the unique values of the target column
unique_values = df["weather"].unique()

# get the last two characters of each unique value
last_two_character = [value[-2:] for value in unique_values]

# join all the last two characters
result = ''.join(last_two_character)

# concatenate the result to the url
url = url + result

print("your current url is: ", url)
```

```
your current url is:  https://kmutt.me/LUartyin
```

1.3 Clue 3: The missing Humidity

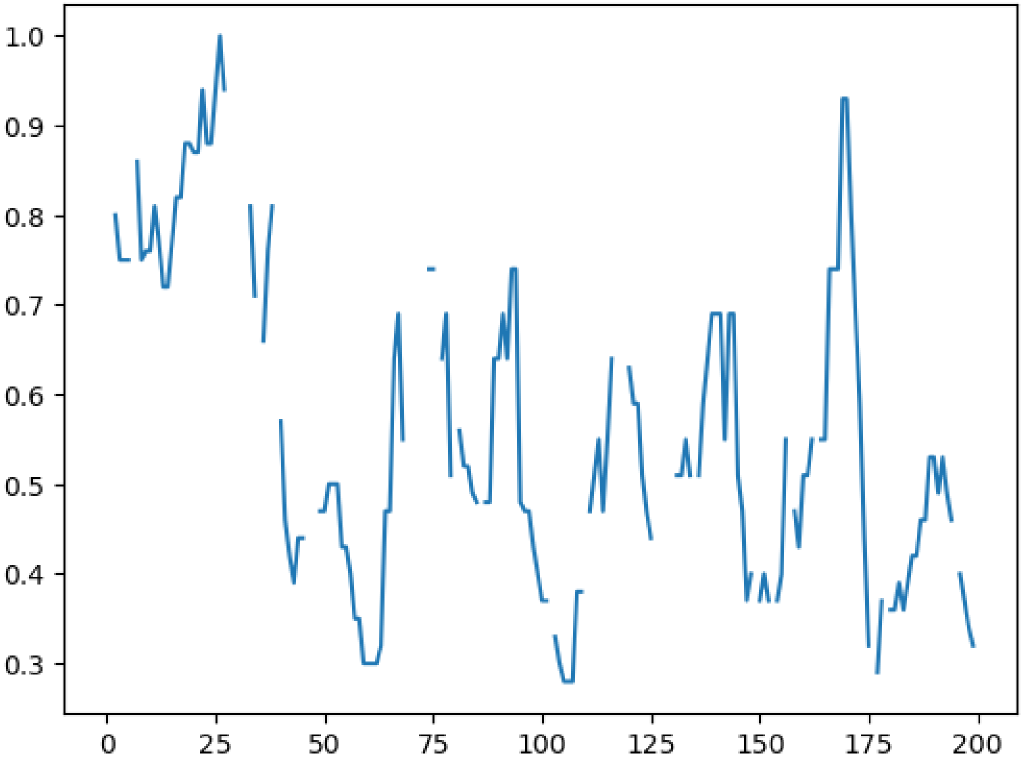
Someone tried to hide a secret message in the humidity levels! you need to see this!!

[24]:

```
df["humidity"].plot()
```

[24]:

```
<Axes: >
```



[25]:

```
df["humidity"].mean()
```

[25]:

```
np.float64(0.5590588235294117)
```

Missing value in the **humidity** column make their average weird.
Find the missing numbers and combine them to reveal the next part of the secret URL!

Monkey Mode

1. Ooo ooo! Find the missing numbers in the humidity column!
2. Combine the missing numbers to reveal the next part of the secret URL!

This is too easy for us. You too you also can do it!

[26]:

```
missing_values = df['humidity'].isna().sum()

url = url + str(missing_values)

print("your current url is: ", url)
```

```
your current url is:  https://kmutt.me/LUartyin30
```

1.3.1 Clue 4: Make the Hum(idity)an back!

Yes! we got a number of missing humidity from the previous clue. Now, we need to make it back to the original data. This is too hard? Don't worry about it you can do it without my help.

[27]:

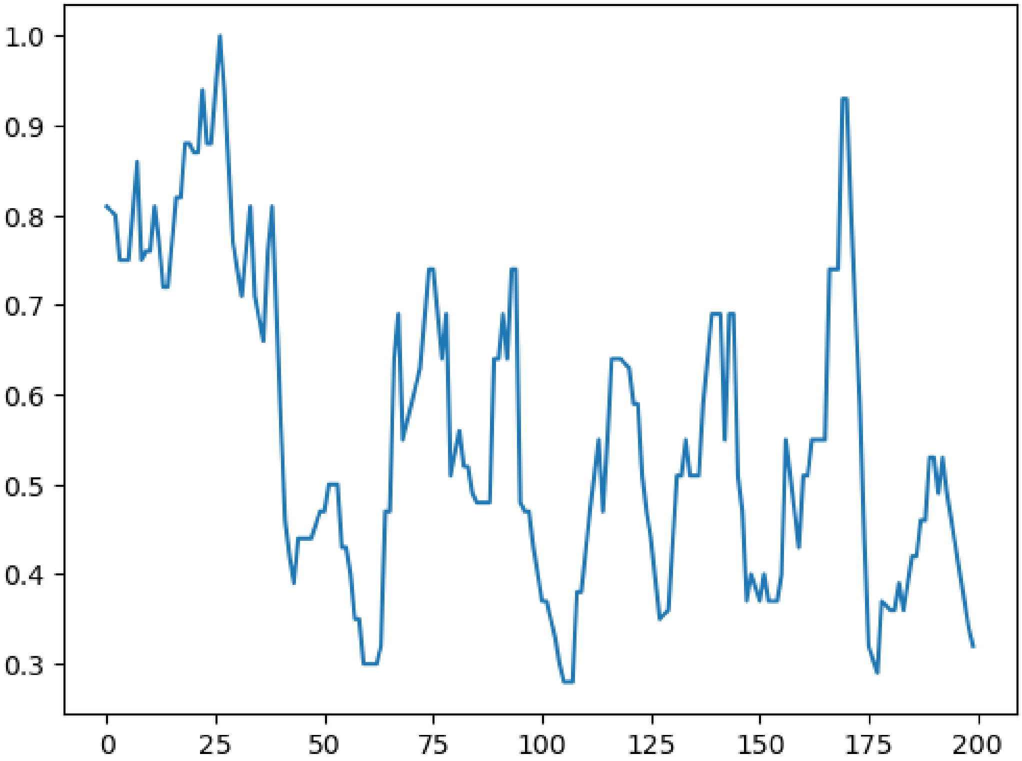
```
for i in range(len(df['humidity'])):


    if pd.isnull(df['humidity'][i]):

        prev_val = df['humidity'][:i].dropna().iloc[-1] if i > 0 else None

        next_val = df['humidity'][i+1:].dropna().iloc[0] if i < len(df)-1 else␣
 ↪None


        if prev_val is not None and next_val is not None:
            df.loc[i,'humidity'] = (prev_val + next_val) / 2
        elif prev_val is not None:
            df.loc[i,'humidity'] = prev_val
        elif next_val is not None:
            df.loc[i,'humidity'] = next_val
```

[28]:

```
df["humidity"].plot()
```

[28]:

```
<Axes: >
```



now, find the average of the humidity column and add it to the missing value. Then, you will find the next part of the secret URL!

[29]:
```
average_humidity = df['humidity'].mean()

average_humidity
```

[29]:
```
np.float64(0.5575249999999999)
```

oh, I forgot to tell you. We only use first 3 decimal places of the average value.

[30]:
```
# get first 3 decimal of the average humidity
result = str(int(average_humidity*1000))

# concatenate the result to the url
url = url + result

print("your current url is: ", url)
```

```
your current url is:  https://kmutt.me/LUartyin30557
```

1.3.2 Clue 5: The Secret Message from the different weathers

We almost there! Find an average of each weather type in the dataset. Then use the ascii number of the sum between `clear` weather and difference of `misty` and `rain` weather to reveal the next part of the secret URL!

Monkey Mode

1. Find the average of each weather type!
2. Use the ASCII number of the sum between `clear` weather and difference of `misty` and `rain` weather!
3. Combine the numbers to reveal the next part of the secret URL!

You're almost there! Keep going!

```
[31]: average_count = df.groupby('weather')['count'].mean()

      clear_avg = average_count['clear']
      misty_avg = average_count['misty']
      rain_avg = average_count['rain']

      groupby_character = chr(int(clear_avg + (misty_avg - rain_avg)))

      url = url + groupby_character

      print("your current url is: ", url)
```

```
your current url is:  https://kmutt.me/LUartyin30557L
```

```
[32]: print("your final url is: ", url)
```

```
your final url is:  https://kmutt.me/LUartyin30557L
```

1.3.3 Clue 6: Fusion!

You've made it this far! Now, You just need to combine the dataframe and and get the standard deviation of `Number of employees` column. then put it in `decode` tools to reveal the final part of the secret URL!

Monkey Mode

1. Combine the dataframe and get the standard deviation of `Number of employees` column!
2. Use the standard deviation as a phrase to unlock the final part of the secret URL!
3. Put the phrase in the `decode` tools to reveal the final part of the secret URL!

Don't be afraid.We will stay with you!

```
[33]: organizations_1 = pd.read_csv('organizations-1.csv')
      organizations_2 = pd.read_csv('organizations-2.csv')
      organizations_3 = pd.read_csv('organizations-3.csv')
```

```
[34]: organizations_1.head()
```

[34]:

```
   Index  Organization Id                     Name  \
0      1  FAB0d41d5b5d22c              Ferrell LLC
1      2  6A7EdDEA9FaDC52  Mckinney, Riley and Day
2      3  0bFED1ADAE4bcC1               Hester Ltd
3      4  2bFC1Be8a4ce42f           Holder-Sellers
4      5  9eE8A6a4Eb96C24              Mayer Group

                          Website           Country  \
0              https://price.net/  Papua New Guinea
1  http://www.hall-buchanan.info/           Finland
2       http://sullivan-reed.com/             China
3             https://becker.com/      Turkmenistan
4          http://www.brewer.com/         Mauritius

                                      Description  Founded  \
0             Horizontal empowering knowledgebase     1990
1             User-centric system-worthy leverage     2015
2                  Switchable scalable moratorium     1971
3  De-engineered systemic artificial intelligence     2004
4              Synchronized needs-based challenge     1991

                      Industry  Number of employees
0                     Plastics                 3498
1  Glass / Ceramics / Concrete                 4952
2                Public Safety                 5287
3                   Automotive                  921
4               Transportation                 7870
```

[35]:

```python
def decode(value: float):
    value = str(int(value))

    return chr(int(value[:2]) + int(value[2:]))
```

[39]:

```python
organizations_combined = pd.concat([organizations_1, organizations_2,␣
 ↪organizations_3])

std_employees = organizations_combined['Number of employees'].std()

print(std_employees)
```

```
2850.8597994927136
```

[41]:

```python
url = url + decode(std_employees)

print("your current url is: ", url)
```

```
your current url is:  https://kmutt.me/LUartyin30557LN
```

1.4 Final Clue: Pokemon configuration

You just need to add a new column call `stat` that will have a condition below:

1. stat calculate from `Attack` + `Defense` + `Speed` + `Sp. Atk` + `Sp. Def` + `HP`
2. If it have type `Normal`, `Grass`, `Fire` or `Water`. Attack will increase by 10%.
3. If it have type `Electric`, `Ice`, `Fighting` or `Poison`. Defense will increase by 10%.
4. If it have type `Ground`, `Flying`, `Psychic` or `Bug`. Speed will increase by 10%.
5. If it have type `Rock`, `Ghost`, `Dragon` or `Dark`. Sp. Atk will increase by 10%.
6. If It have `speed` more than 100. Sp. Def will increase by 50%.
7. If it is a `legendary` pokemon. HP will increase by 100.

Then, group by `Type 1` and find the average of `stat` column. This Clue is **important** you must do it, but I will give you the final part of the secret URL. The final part of the secret URL is `pikachu`.

[42]:
```
pokemon = pd.read_csv("pokemon.csv")
pokemon.head()
```

[42]:
```
   #                   Name Type 1  Type 2  Total  HP  Attack  Defense  \
0  1              Bulbasaur  Grass  Poison    318  45      49       49
1  2                Ivysaur  Grass  Poison    405  60      62       63
2  3               Venusaur  Grass  Poison    525  80      82       83
3  3  VenusaurMega Venusaur  Grass  Poison    625  80     100      123
4  4             Charmander   Fire     NaN    309  39      52       43

   Sp. Atk  Sp. Def  Speed  Generation  Legendary
0       65       65     45           1      False
1       80       80     60           1      False
2      100      100     80           1      False
3      122      120     80           1      False
4       60       50     65           1      False
```

[43]:
```
# Calculate base stat total
pokemon['stat'] = pokemon['Attack'] + pokemon['Defense'] + pokemon['Speed'] + \
                 pokemon['Sp. Atk'] + pokemon['Sp. Def'] + pokemon['HP']

# Type 1 conditions
type1_attack_boost = ['Normal', 'Grass', 'Fire', 'Water']
type1_defense_boost = ['Electric', 'Ice', 'Fighting', 'Poison']
type1_speed_boost = ['Ground', 'Flying', 'Psychic', 'Bug']
type1_spatk_boost = ['Rock', 'Ghost', 'Dragon', 'Dark']

# Apply Type 1 boosts
pokemon.loc[pokemon['Type 1'].isin(type1_attack_boost), 'stat'] += pokemon.
  ↪loc[pokemon['Type 1'].isin(type1_attack_boost), 'Attack'] * 0.1
pokemon.loc[pokemon['Type 1'].isin(type1_defense_boost), 'stat'] += pokemon.
  ↪loc[pokemon['Type 1'].isin(type1_defense_boost), 'Defense'] * 0.1
pokemon.loc[pokemon['Type 1'].isin(type1_speed_boost), 'stat'] += pokemon.
  ↪loc[pokemon['Type 1'].isin(type1_speed_boost), 'Speed'] * 0.1
```

```
pokemon.loc[pokemon['Type 1'].isin(type1_spatk_boost), 'stat'] += pokemon.
 ↪loc[pokemon['Type 1'].isin(type1_spatk_boost), 'Sp. Atk'] * 0.1

# Speed > 100 condition
pokemon.loc[pokemon['Speed'] > 100, 'stat'] += pokemon.loc[pokemon['Speed'] >␣
 ↪100, 'Sp. Def'] * 0.5

# Legendary condition
pokemon.loc[pokemon['Legendary'], 'stat'] += 100

# Group by Type 1 and get mean of stat
type1_stat_means = pokemon.groupby('Type 1')['stat'].mean()
print("\nAverage stats by Type 1:")
print(type1_stat_means)
```

```
Average stats by Type 1:
Type 1
Bug         388.986957
Dark        467.883871
Dragon      609.981250
Electric    470.765909
Fairy       419.058824
Fighting    426.611111
Fire        480.486538
Flying      576.500000
Ghost       458.043750
Grass       437.764286
Ground      458.734375
Ice         453.204167
Normal      416.168367
Poison      407.453571
Psychic     520.771930
Rock        474.118182
Steel       505.888889
Water       443.736607
Name: stat, dtype: float64
```

```
/tmp/ipykernel_41856/855082990.py:12: FutureWarning: Setting an item of
incompatible dtype is deprecated and will raise an error in a future version of
pandas. Value '[322.9 411.2 533.2 635.  314.2 411.4 542.4 647.  644.4 318.8
411.3 538.3
 640.3 255.5 355.  487.  587.  258.6 421.1 268.  451.  303.1 512.6 274.5
 442.  325.  401.5 498.  294.5 447.  325.2 508.2 357.  566.  305.  391.5
 519.5 307.5 399.  500.5 339.  522.  418.5 510.  321.5 497.5 597.5 358.5
 318.5 471.  329.5 482.  311.5 534.5 335.5 488.  329.  529.5 390.5 450.5
 440.5 499.5 602.5 299.  446.5 326.7 459.2 344.5 527.5 504.5 500.  201.
 552.5 655.5 543.5 292.8 330.5 531.5 538.  401.  551.  590.  322.9 411.2
 533.2 314.2 411.4 542.4 320.5 413.  540.5 219.6 422.6 265.  447.  333.8
 465.8 213.  498.  252.  425.  507.5 253.5 344.5 465.5 367.  183.  432.5
 214.5 438.5 497.5 463.  422.  439.5 338.  513.  254.  415.  385.5 306.5
 490.5 469.  549.5 523.  474.5 252.  372.5 498.  541.  591.5 587.5 693.
 314.5 411.5 538.5 641.  316.  413.5 542.  646.  317.  413.5 546.  650.
 243.  427.  223.  345.  487.  224.  347.  490.  275.5 438.5 273.  435.
 299.  473.  286.  448.  686.  245.1 367.1 499.1 192.  264.5 386.5 406.
 314.  472.  574.  407.  509.  311.  470.  572.  478.5 366.  343.5 486.5
 314.  469.5 292.8 475.8 316.  480.  201.5 546.  427.  449.  466.8 351.4
 495.4 493.4 494.  333.  680.  785.  324.8 413.9 535.9 314.8 412.8 544.4
 319.1 411.6 538.6 250.5 347.5 497.  254.5 418.5 283.  522.  336.5 505.5
 278.5 456.  329.8 483.3 492.  356.6 487.6 593.6 315.5 460.2 220.5 417.5
 398.5 464.  334.9 466.9 347.  340.2 503.2 607.2 523.5 545.  549.5 536.
 543.  692.  609.  686.  488.  610.  610.  610.3 732.  312.5 419.  535.5
 314.3 427.3 540.3 313.5 420.5 538.  260.5 428.5 281.  378.  511.  321.3
 507.8 321.3 507.8 321.3 507.8 269.5 365.7 499.5 451.  551.  299.  390.5
 518.5 282.7 486.7 283.5 486.  469.2 324.  494.  543.  469.6 362.8 505.8
 305.  479.5 309.4 481.7 341.  485.  299.5 472.5 339.  486.  477.5 310.
 498.4 501.  358.3 522.3 493.7 589.  587.2 587.2 607.7 612.8 319.1 412.8
 540.7 311.5 414.9 540.9 319.6 411.3 539.5 240.6 428.6 283.  389.3 507.1
 374.  513.8 356.5 541.  480.  335.3 507.3 611. ]' has dtype incompatible with
int64, please explicitly cast to a compatible dtype first.
  pokemon.loc[pokemon['Type 1'].isin(type1_attack_boost), 'stat'] +=
pokemon.loc[pokemon['Type 1'].isin(type1_attack_boost), 'Attack'] * 0.1
```

[44]:

```
url = url + "pikachu"

print("your final url is: ", url)
```

```
your final url is:  https://kmutt.me/LUartyin30557LNpikachu
```

1.4.1 Final Mission (Optional)

Access the secret URL and complete your quest!

Question: What is the final secret URL?
Ans:

Rick Roll Rick Astley - Never Gonna Give You Up (Official Music Video) https://www.youtube.com/watch?v=dQw4w9WgXcQ

https://kmutt.me/LUartyin30557LNpikachu

Enjoy the adventure!