assignment4

June 15, 2022

1 Assignment 4

1.1 Description

In this assignment you must read in a file of metropolitan regions and associated sports teams from `assets/wikipedia_data.html` and answer some questions about each metropolitan region. Each of these regions may have one or more teams from the "Big 4": NFL (football, in `assets/nfl.csv`), MLB (baseball, in `assets/mlb.csv`), NBA (basketball, in `assets/nba.csv` or NHL (hockey, in `assets/nhl.csv`). Please keep in mind that all questions are from the perspective of the metropolitan region, and that this file is the "source of authority" for the location of a given sports team. Thus teams which are commonly known by a different area (e.g. "Oakland Raiders") need to be mapped into the metropolitan region given (e.g. San Francisco Bay Area). This will require some human data understanding outside of the data you've been given (e.g. you will have to hand-code some names, and might need to google to find out where teams are)!

For each sport I would like you to answer the question: **what is the win/loss ratio's correlation with the population of the city it is in?** Win/Loss ratio refers to the number of wins over the number of wins plus the number of losses. Remember that to calculate the correlation with `pearsonr`, so you are going to send in two ordered lists of values, the populations from the wikipedia_data.html file and the win/loss ratio for a given sport in the same order. Average the win/loss ratios for those cities which have multiple teams of a single sport. Each sport is worth an equal amount in this assignment (20%*4=80%) of the grade for this assignment. You should only use data **from year 2018** for your analysis – this is important!

1.2 Notes

1. Do not include data about the MLS or CFL in any of the work you are doing, we're only interested in the Big 4 in this assignment.
2. I highly suggest that you first tackle the four correlation questions in order, as they are all similar and worth the majority of grades for this assignment. This is by design!
3. It's fair game to talk with peers about high level strategy as well as the relationship between metropolitan areas and sports teams. However, do not post code solving aspects of the assignment (including such as dictionaries mapping areas to teams, or regexes which will clean up names).
4. There may be more teams than the assert statements test, remember to collapse multiple teams in one city into a single value!

1.3 Question 1

For this question, calculate the win/loss ratio's correlation with the population of the city it is in for the **NHL** using **2018** data.

[3]:
```python
import pandas as pd
import numpy as np
import scipy.stats as stats
import re

nhl_df=pd.read_csv("assets/nhl.csv")
cities=pd.read_html("assets/wikipedia_data.html")[1]
cities=cities.iloc[:-1,[0,3,5,6,7,8]]

def nhl_correlation():
    global cities
    global nhl_df

    # Limpio cities dataframe
    cities["NHL"] = cities["NHL"].apply(lambda x: re.sub(r"\[.+\]", "", x))
    cities["NHL"] = cities["NHL"].apply(lambda x: re.findall('[A-Z][a-z]+', x))
    cities = cities.explode("NHL")

    # Limpio nhl_df dataframe
    nhl_df = nhl_df[nhl_df["year"] == 2018]
    nhl_df["team"] = nhl_df["team"].apply(lambda x: x.replace("*", ""))
    nhl_df["team"] = nhl_df["team"].apply(lambda x: x.split(" ")[-1])

    # Hago inner merge de los dataframes
    merge_df = pd.merge(cities, nhl_df, how='inner', left_on="NHL",␣
 ↪right_on="team")
    merge_df['WL_ratio'] = merge_df["W"].astype("float")/(merge_df["W"].
 ↪astype("int") + merge_df["L"].astype("int"))
    merge_df['Population (2016 est.)[8]'] = merge_df['Population (2016 est.
 ↪)[8]'].astype(int)
    group_df = merge_df.groupby('Metropolitan area').agg({'WL_ratio':np.mean,␣
 ↪'Population (2016 est.)[8]': np.mean})


    # Correlación de pearson
    population_by_region = group_df['Population (2016 est.)[8]']
    win_loss_by_region = group_df['WL_ratio']

    assert len(population_by_region) == len(win_loss_by_region), "Q1: Your␣
 ↪lists must be the same length"
    assert len(population_by_region) == 28, "Q1: There should be 28 teams being␣
 ↪analysed for NHL"
```

```
    result = stats.pearsonr(population_by_region, win_loss_by_region)
    return result

nhl = nhl_correlation()
print(f'Coeficiente de correlación = {nhl[0]:.1} y su p-valor= {nhl[1]:.2}')
```

```
Coeficiente de correlación = 0.01 y su p-valor= 0.95
```

[ ]:

```

```

1.4 Question 2

For this question, calculate the win/loss ratio's correlation with the population of the city it is in for the **NBA** using **2018** data.

[8]:

```
import pandas as pd
import numpy as np
import scipy.stats as stats
import re

nba_df=pd.read_csv("assets/nba.csv")
cities=pd.read_html("assets/wikipedia_data.html")[1]
cities=cities.iloc[:-1,[0,3,5,6,7,8]]

def nba_correlation():

    # Asigno los dataframes globales para que estén disponibles en el namespace
    # de esta función

    global cities
    global nba_df

    #Limpio cities dataframe
    cities["NBA"] = cities["NBA"].apply(lambda x: re.sub("\[.+\]|\", "", x))
    cities["NBA"] = cities["NBA"].apply(lambda x: re.
 ↪findall('[A-Z][a-z]+|[\d]+[a-z]+', x))
    cities = cities.explode("NBA")

    #Limpio nhl_df dataframe
    nba_df = nba_df[nba_df["year"] == 2018]
    nba_df["team"] = nba_df["team"].apply(lambda x: re.sub("\(\d+\)", "", x))
    nba_df["team"] = nba_df["team"].apply(lambda x: x.replace('*', ""))
    nba_df["team"] = nba_df["team"].apply(lambda x: x.split(" ")[-1].strip())

    #Hago inner merge de los dataframes
    merge_df = pd.merge(cities, nba_df, how='inner', left_on="NBA",␣
 ↪right_on="team")
```

```
    merge_df['WL_ratio'] = merge_df["W"].astype("float")/(merge_df["W"].
↪astype("int") + merge_df["L"].astype("int"))
    merge_df['Population (2016 est.)[8]'] = merge_df['Population (2016 est.
↪)[8]'].astype(int)

    # Agrupo a los equipos según su ciudad y calculo el promedio de WL_ratio
    # de la ciudad
    group_df = merge_df.groupby('Metropolitan area').agg({'WL_ratio':np.mean,␣
↪'Population (2016 est.)[8]': np.mean})
    group_df.rename(inplace= True, columns = {'WL_ratio':'WL_ratio_mean',␣
↪'Population (2016 est.)[8]':'Population'})

    #Correlación de pearson
    population_by_region = group_df['Population']
    win_loss_by_region = group_df['WL_ratio_mean']

    assert len(population_by_region) == len(win_loss_by_region), "Q2: Your␣
↪lists must be the same length"
    assert len(population_by_region) == 28, "Q2: There should be 28 teams being␣
↪analysed for NBA"
    5

    assert len(population_by_region) == len(win_loss_by_region), "Q2: Your␣
↪lists must be the same length"
    assert len(population_by_region) == 28, "Q2: There should be 28 teams being␣
↪analysed for NBA"

    return stats.pearsonr(population_by_region, win_loss_by_region)

nba = nba_correlation()
print(f'Coeficiente de correlación = {nba[0]:.1} y su p-valor= {nba[1]:.2}')
```

```
Coeficiente de correlación = -0.2 y su p-valor= 0.37
```

[ ]:

1.5 Question 3

For this question, calculate the win/loss ratio's correlation with the population of the city it is in for the **MLB** using **2018** data.

[6]:
```
import pandas as pd
import numpy as np
import scipy.stats as stats
import re

mlb_df=pd.read_csv("assets/mlb.csv")
```

```
cities=pd.read_html("assets/wikipedia_data.html")[1]
cities=cities.iloc[:-1,[0,3,5,6,7,8]]

def mlb_correlation():
    global cities
    global mlb_df

    # Limpio cities dataframe
    cities["MLB"] = cities["MLB"].str.replace("\[.+\]|\", "", regex = True)
    cities["MLB"] = cities["MLB"].apply(lambda x : x.strip())
    cities["MLB"] = cities["MLB"].apply(lambda x: re.findall('[A-Z][a-z]+[␣
↪][A-Z][a-z]+|[A-Z][a-z]+|[\d]+[a-z]+', x))
    cities = cities.explode("MLB")

    # Limpio mlb_df dataframe
    mlb_df = mlb_df[mlb_df["year"] == 2018]

    def extraer_equipos(row):
        ''' Función que saca la ciudad que antecede al nombre de los equipos'''
        # creo lista de equipos a partir de cities.MLB sacando los datos nan
        equipos = [x.strip() for x in cities.MLB if type(x) == str]
        for equipo in equipos:
            if equipo in row['team']:
                row['team'] = equipo
                return row
            else:
                continue

    mlb_df = mlb_df.apply(extraer_equipos, axis = 'columns')

    # Hago inner merge de los dataframes
    merge_df = pd.merge(cities, mlb_df, how='inner', left_on="MLB",␣
↪right_on="team")
    merge_df['WL_ratio'] = merge_df["W"].astype("float")/(merge_df["W"].
↪astype("int") + merge_df["L"].astype("int"))
    merge_df['Population (2016 est.)[8]'] = merge_df['Population (2016 est.
↪)[8]'].astype(int)
    group_df = merge_df.groupby('Metropolitan area').agg({'WL_ratio':np.mean,␣
↪'Population (2016 est.)[8]': np.mean})
    group_df.rename(inplace= True, columns = {'WL_ratio':'WL_ratio_mean',␣
↪'Population (2016 est.)[8]':'Population'})

    # Correlación de pearson
    population_by_region = group_df['Population']
    win_loss_by_region = group_df['WL_ratio_mean']
    assert len(population_by_region) == len(win_loss_by_region), "Q3: Your␣
↪lists must be the same length"
    assert len(population_by_region) == 26, "Q3: There should be 26 teams being␣
↪analysed for MLB"

    result = stats.pearsonr(population_by_region, win_loss_by_region)

    return result

result = mlb_correlation()

print(f'Coeficiente de correlación = {result[0]:.1} y su p-valor= {result[1]:.
↪2}')
```

```
Coeficiente de correlación = 0.2 y su p-valor= 0.46
```

[ ]:

```
```

1.6 Question 4

For this question, calculate the win/loss ratio's correlation with the population of the city it is in for the **NFL** using **2018** data.

[1]:

```
import pandas as pd
import numpy as np
import scipy.stats as stats
import re

nfl_df=pd.read_csv("assets/nfl.csv")
cities=pd.read_html("assets/wikipedia_data.html")[1]
cities=cities.iloc[:-1,[0,3,5,6,7,8]]

def nfl_correlation():

    global cities
    global nfl_df

    # Limpio cities dataframe
    cities["NFL"] = cities["NFL"].str.replace("\[.+\]|\", "", regex = True)
    cities["NFL"] = cities["NFL"].apply(lambda x : x.strip())
    cities["NFL"] = cities["NFL"].apply(lambda x: re.findall('[A-Z][a-z]+[␣
↪][A-Z][a-z]+|[A-Z][a-z]+|[\d]+[a-z]+', x))
    cities = cities.explode("NFL")

    # Limpio mlb_df dataframe
    nfl_df = nfl_df[nfl_df["year"] == 2018]
    nfl_df["team"] = nfl_df["team"].replace("\W", "", regex=True)
    nfl_df["team"] = nfl_df["team"].str.findall('[A-Z][a-z]+$|[\d]+[a-z]+$')
    nfl_df["team"] = nfl_df["team"].apply(lambda x : x[0])

    # Hago inner merge de los dataframes
    merge_df = pd.merge(cities, nfl_df, how='inner', left_on="NFL",␣
 ↪right_on="team")
    merge_df['WL_ratio'] = merge_df["W"].astype("float")/(merge_df["W"].
 ↪astype("int") + merge_df["L"].astype("int"))
    merge_df['Population (2016 est.)[8]'] = merge_df['Population (2016 est.
 ↪)[8]'].astype(int)
    group_df = merge_df.groupby('Metropolitan area').agg({'WL_ratio':np.mean,␣
 ↪'Population (2016 est.)[8]': np.mean})
    group_df.rename(inplace= True, columns = {'WL_ratio':'WL_ratio_mean',␣
 ↪'Population (2016 est.)[8]':'Population'})

    # Correlación de pearson
    population_by_region = group_df['Population']
    win_loss_by_region = group_df['WL_ratio_mean']

    assert len(population_by_region) == len(win_loss_by_region), "Q4: Your␣
 ↪lists must be the same length"
    assert len(population_by_region) == 29, "Q4: There should be 29 teams being␣
 ↪analysed for NFL"

    return stats.pearsonr(population_by_region, win_loss_by_region)

result = nfl_correlation()
print(f'Coeficiente de correlación = {result[0]:.1} y su p-valor= {result[1]:.
 ↪2}')
```

```
Coeficiente de correlación = 0.005 y su p-valor= 0.98
```

[ ]:

```
```

1.7 Question 5

In this question I would like you to explore the hypothesis that **given that an area has two sports teams in different sports, those teams will perform the same within their respective sports**. How I would like to see this explored is with a series of paired t-tests (so use `ttest_rel`) between all pairs of sports. Are there any sports where we can reject the null hypothesis? Again, average values where a sport has multiple teams in one region. Remember, you will only be including, for each sport, cities which have teams engaged in that sport, drop others as appropriate. This question is worth 20% of the grade for this assignment.

[2]:

```
import pandas as pd
import numpy as np
import scipy.stats as stats
import re

nhl_df=pd.read_csv("assets/nhl.csv")
nba_df=pd.read_csv("assets/nba.csv")
mlb_df=pd.read_csv("assets/mlb.csv")
nfl_df=pd.read_csv("assets/nfl.csv")
cities=pd.read_html("assets/wikipedia_data.html")[1]
cities=cities.iloc[:-1,[0,3,5,6,7,8]]


# Genero 4 funciones que limpian los dataframes y agrupan a los deportes
# según sus áreas metropolitanas

def NHL_groups_df():
    '''Función que limpia las dataframes cities y nhl_df, luego hace un merge
    entre ambas y agrupa a los equipos según área metropolitana.
    Como resultado devuelve una dataframe que incluye: áreas (grupos como␣
 ↪index,
    y las columnas W/L rate, Population y Nr. of teams para cada grupo'''

    global nhl_df
    global cities

    # Limpio cities['NHL']
    # Hago copia para no modificar la df original que es usada por el resto de␣
 ↪las
    # funciones

    cities1 = cities.copy()
    cities1["NHL"] = cities1["NHL"].apply(lambda x: re.sub("\[.+\]|\", "", x))
    cities1["NHL"] = cities1["NHL"].apply(lambda x: re.findall('[A-Z][a-z]+',␣
 ↪x))
    cities1 = cities1.explode("NHL")

    # Limpio nhl_df dataframe
    nhl_df = nhl_df[nhl_df["year"] == 2018]
    nhl_df["team"] = nhl_df["team"].apply(lambda x: x.replace("*", ""))
    nhl_df["team"] = nhl_df["team"].apply(lambda x: x.split(" ")[-1])

    # Hago inner merge de los dataframes
    merge_df = pd.merge(cities1, nhl_df, how='inner', left_on="NHL",␣
 ↪right_on="team")
    merge_df['WL_ratio'] = merge_df["W"].astype("float")/(merge_df["W"].
 ↪astype("int") + merge_df["L"].astype("int"))
    merge_df['Population (2016 est.)[8]'] = merge_df['Population (2016 est.
 ↪)[8]'].astype(int)
```

```
    # Agrupo a los equipos según su ciudad y calculo: WL_ratio, poblacion y nro.
 ↪ equipos
    group_df = merge_df.groupby('Metropolitan area').agg({'WL_ratio':np.mean,␣
 ↪'Population (2016 est.)[8]': np.mean, 'team':len})
    group_df = group_df. rename(columns= {'WL_ratio':'W/L rate', 'Population␣
 ↪(2016 est.)[8]':'Population', 'team': 'Nr. of teams'})
    group_df['League'] = 'NHL'

    return group_df


def NBA_groups_df():
    '''Función que limpia las dataframes cities y nba_df, luego hace un merge
    entre ambas y agrupa a los equipos según área metropolitana.
    Como resultado devuelve una dataframe que incluye: áreas (grupos como␣
 ↪index,
    y las columnas W/L rate, Population y Nr. of teams para cada grupo'''

    global nba_df
    global cities

    # Limpio cities dataframe
    cities1 = cities.copy()
    cities1["NBA"] = cities1["NBA"].apply(lambda x: re.sub("\[.+\]|\", "", x))
    cities1["NBA"] = cities1["NBA"].apply(lambda x: re.
 ↪findall('[A-Z][a-z]+|[\d]+[a-z]+', x))
    cities1 = cities1.explode("NBA")

    # Limpio nhl_df dataframe
    nba_df = nba_df[nba_df["year"] == 2018]
    nba_df["team"] = nba_df["team"].apply(lambda x: re.sub("\(\d+\)", "", x))
    nba_df["team"] = nba_df["team"].apply(lambda x: x.replace('*', ""))
    nba_df["team"] = nba_df["team"].apply(lambda x: x.split(" ")[-1].strip())

    # Hago inner merge de los dataframes
    merge_df = pd.merge(cities1, nba_df, how='inner', left_on="NBA",␣
 ↪right_on="team")
    merge_df['WL_ratio'] = merge_df["W"].astype("float")/(merge_df["W"].
 ↪astype("int") + merge_df["L"].astype("int"))
    merge_df['Population (2016 est.)[8]'] = merge_df['Population (2016 est.
 ↪)[8]'].astype(int)

    # Agrupo a los equipos según su ciudad y calculo: WL_ratio, poblacion y nro.
 ↪ equipos
```

```
    group_df = merge_df.groupby('Metropolitan area').agg({'WL_ratio':np.mean,␣
↪'Population (2016 est.)[8]': np.mean, 'team':len})
    group_df = group_df. rename(columns= {'WL_ratio':'W/L rate', 'Population␣
↪(2016 est.)[8]':'Population', 'team': 'Nr. of teams'})
    group_df['League'] = 'NBA'

    return group_df


def MLB_groups_df():
    '''Función que limpia las dataframes cities y mlb_df, luego hace un merge
    entre ambas y agrupa a los equipos según área metropolitana.
    Como resultado devuelve una dataframe que incluye: áreas (grupos como␣
↪index,
    y las columnas W/L rate, Population y Nr. of teams para cada grupo'''

    global mlb_df
    global cities

    #Limpio cities dataframe
    cities1 = cities.copy()
    cities1["MLB"] = cities1["MLB"].str.replace("\[.+\]|\", "", regex = True)
    cities1["MLB"] = cities1["MLB"].apply(lambda x : x.strip())
    cities1["MLB"] = cities1["MLB"].apply(lambda x: re.findall('[A-Z][a-z]+[␣
↪][A-Z][a-z]+|[A-Z][a-z]+|[\d]+[a-z]+', x))
    cities1 = cities1.explode("MLB")

    # Limpio mlb_df dataframe
    mlb_df = mlb_df[mlb_df["year"] == 2018]
    def extraer_equipos(row):
        ''' Función que saca la ciudad que antecede al nombre de los equipos'''
        equipos = [x.strip() for x in cities1.MLB if type(x) == str]
        for equipo in equipos:
            if equipo in row['team']:
                row['team'] = equipo
                return row
            else:
                continue

    mlb_df = mlb_df.apply(extraer_equipos, axis = 'columns')

    # Hago inner merge de los dataframes
    merge_df = pd.merge(cities1, mlb_df, how='inner', left_on="MLB",␣
↪right_on="team")
    merge_df['WL_ratio'] = merge_df["W"].astype("float")/(merge_df["W"].
↪astype("int") + merge_df["L"].astype("int"))
```

```
    merge_df['Population (2016 est.)[8]'] = merge_df['Population (2016 est.
↪)[8]'].astype(int)

    # Agrupo a los equipos según su ciudad y calculo: WL_ratio, poblacion y nro.
↪ equipos
    group_df = merge_df.groupby('Metropolitan area').agg({'WL_ratio':np.mean,␣
↪'Population (2016 est.)[8]': np.mean, 'team':len})
    group_df = group_df. rename(columns= {'WL_ratio':'W/L rate', 'Population␣
↪(2016 est.)[8]':'Population', 'team': 'Nr. of teams'})
    group_df['League'] = 'MLB'

    return group_df


def NFL_groups_df():
    '''Función que limpia las dataframes cities y nfl_df, luego hace un merge
    entre ambas y agrupa a los equipos según área metropolitana.
    Como resultado devuelve una dataframe que incluye: áreas (grupos como␣
↪index,
    y las columnas W/L rate, Population y Nr. of teams para cada grupo'''

    global nfl_df
    global cities

    # Limpio cities dataframe
    cities1 = cities.copy()
    cities1["NFL"] = cities1["NFL"].str.replace("\[.+\]|\", "", regex = True)
    cities1["NFL"] = cities1["NFL"].apply(lambda x : x.strip())
    cities1["NFL"] = cities1["NFL"].apply(lambda x: re.findall('[A-Z][a-z]+[␣
↪][A-Z][a-z]+|[A-Z][a-z]+|[\d]+[a-z]+', x))
    cities1 = cities1.explode("NFL")

    # Limpio mlb_df dataframe
    nfl_df = nfl_df[nfl_df["year"] == 2018]
    nfl_df["team"] = nfl_df["team"].replace("\W", "", regex=True)
    nfl_df["team"] = nfl_df["team"].str.findall('[A-Z][a-z]+$|[\d]+[a-z]+$')
    nfl_df["team"] = nfl_df["team"].apply(lambda x : x[0])

    # Hago inner merge de los dataframes
    merge_df = pd.merge(cities1, nfl_df, how='inner', left_on="NFL",␣
↪right_on="team")
    merge_df['WL_ratio'] = merge_df["W"].astype("float")/(merge_df["W"].
↪astype("int") + merge_df["L"].astype("int"))
    merge_df['Population (2016 est.)[8]'] = merge_df['Population (2016 est.
↪)[8]'].astype(int)
```

```
    # Agrupo a los equipos según su ciudad y calculo: WL_ratio, poblacion y nro.
 ↪ equipos
    group_df = merge_df.groupby('Metropolitan area').agg({'WL_ratio':np.mean,␣
 ↪'Population (2016 est.)[8]': np.mean, 'team':len})
    group_df = group_df. rename(columns= {'WL_ratio':'W/L rate', 'Population␣
 ↪(2016 est.)[8]':'Population', 'team': 'Nr. of teams'})
    group_df['League'] = 'NFL'
    return group_df


def get_WLrate_Population(deporte):
    '''Función que toma el deporte y selecciona la función adecuada para
    obtener la dataframe con los datos de WL.
    Las posibilidades'''

    if deporte == 'NHL':
        nhl = NHL_groups_df()
        return nhl

    if deporte == 'NBA':
        nba = NBA_groups_df()
        return nba

    if deporte == 'MLB':
        mlb = MLB_groups_df()
        return mlb

    if deporte == 'NFL':
        nfl = NFL_groups_df()
        return nfl

def sports_team_performance():


    #Crea una matriz de correlación vacía y la llena de valones NaN
    sports = ['NFL', 'NBA', 'NHL', 'MLB']
    p_values = pd.DataFrame({k:np.nan for k in sports}, index=sports)

    # Iteración que realiza el ttest para cada par de deportes
    for i in sports:
        for j in sports:
            if i!=j: # evitamos realizar comparaciones de un deportse consigo␣
 ↪mismo
                # Se importan las dataframes con W/L rate y population␣
 ↪calculadas en los ejercicios anteriores
                Mi=get_WLrate_Population(i)
                Mj=get_WLrate_Population(j)
                # Se extraen las columnas W/L rate para cada par de deportes
                Mi=Mi['W/L rate']
                Mj=Mj['W/L rate']
                # Une los deportes según áreas metropolitanas, quedan sólo
                # aquellas áreas que tienen equipos en ambos deportes
                merge=pd.
 ↪merge(Mi,Mj,how='inner',left_index=True,right_index=True)
                # Realiza T-test y agrega el p-valor resultante en su posición
                # dentro de la matriz de correlación
                p_values.loc[i, j]=stats.ttest_rel(merge['W/L rate_x'],merge['W/
 ↪L rate_y'])[1]

    assert abs(p_values.loc["NBA", "NHL"] - 0.02) <= 1e-2, "The NBA-NHL p-value␣
 ↪should be around 0.02"
    assert abs(p_values.loc["MLB", "NFL"] - 0.80) <= 1e-2, "The MLB-NFL p-value␣
 ↪should be around 0.80"

    return p_values

p_values = sports_team_performance()
print(p_values)
```

```
          NFL       NBA       NHL       MLB
NFL       NaN  0.941792  0.030883  0.802069
NBA  0.941792       NaN  0.022297  0.950540
NHL  0.030883  0.022297       NaN  0.000708
MLB  0.802069  0.950540  0.000708       NaN
```

[ ]:

[ ]: