

 On branch **default**

[1]

# Load Dataset of Billionaires Statistics Dataset

[2] LOAD DATASET billionaires AS csv FROM Billionaires Statistics Dataset.csv @ artifact file 69

billionaires (2640 rows) 

Views  

	rank (short)	finalWorth (int)	category (string)	personName (string)	age (short)	country (string)	city (string)	source (string)	industri
0	1	211000	Fashion & Retail	Bernard Arnault & family	74	France	Paris	LVMH	Fashion
1	2	180000	Automotive	Elon Musk	51	United States	Austin	Tesla, SpaceX	Automoti
2	3	114000	Technology	Jeff Bezos	59	United States	Medina	Amazon	Technolo
3	4	107000	Technology	Larry Ellison	78	United States	Lanai	Oracle	Technolo
4	5	106000	Finance & Investments	Warren Buffett	92	United States	Omaha	Berkshire Hathaway	Finance
5	6	104000	Technology	Bill Gates	67	United States	Medina	Microsoft	Technolo
6	7	94500	Media & Entertainment	Michael Bloomberg	81	United States	New York	Bloomberg LP	Media &
7	8	93000	Telecom	Carlos Slim Helu & family	83	Mexico	Mexico City	Telecom	Telecom
8	9	83400	Diversified	Mukesh Ambani	65	India	Mumbai	Diversified	Diversifie
9	10	80700	Technology	Steve Ballmer	67	United States	Hunts Point	Microsoft	Technolo
10	11	80500	Fashion & Retail	Francoise Bettencourt Meyers & family	69	France	Paris	L'Oréal	Fashion
11	12	79200	Technology	Larry Page	50	United States	Palo Alto	Google	Technolo
12	13	77300	Fashion & Retail	Amancio Ortega	87	Spain	La Coruna	Zara	Fashion
13	14	76000	Technology	Sergey Brin	49	United States	Los Altos	Google	Technolo
14	15	68000	Food & Beverage	Zhong Shanshan	68	China	Hangzhou	Beverages, pharmaceuticals	Food & E
15	16	64400	Technology	Mark Zuckerberg	38	United States	Palo Alto	Facebook	Technolo
16	17	59000	Diversified	Charles Koch & family	87	United States	Wichita	Koch Industries	Diversifie
17	17	59000	Diversified	Julia Koch & family	60	United States	New York	Koch Industries	Diversifie
18	19	58800	Fashion & Retail	Jim Walton	74	United States	Bentonville	Walmart	Fashion
19	20	57600	Fashion & Retail	Rob Walton & family	78	United States	Bentonville	Walmart	Fashion

[3]

# Data Information

[4]


```
# Get read-only pandas dataframe object for given dataset.
df = vizierdb.get_data_frame('billionaires');

#How Data Looks Like
print(df.head())

#Checking the Number of Rows and Columns
print(df.shape)

#Check the datatype of the dataset
print(df.info())

#summary of dataset
print(df.describe())
```

 Data Curation Project

Projects ▾

Branches ▾

 Notebook

Dataset ▾

Caveats ▾

Settings ▾

0	1	211000	...	46.227638	2.213749
1	2	180000	...	37.090240	-95.712891
2	3	114000	...	37.090240	-95.712891
3	4	107000	...	37.090240	-95.712891
4	5	106000	...	37.090240	-95.712891

[5 rows x 35 columns]

(2640, 35)

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2640 entries, 0 to 2639
Data columns (total 35 columns):
#   Column                                     Non-Null Count  Dtype
---  -
0   rank                                       2640 non-null   int16
1   finalWorth                               2640 non-null   int32
2   category                                 2640 non-null   object
3   personName                              2640 non-null   object
4   age                                       2575 non-null   float64
5   country                                  2602 non-null   object
6   city                                     2568 non-null   object
7   source                                   2640 non-null   object
8   industries                               2640 non-null   object
9   countryOfCitizenship                    2640 non-null   object
10  organization                             325 non-null    object
11  selfMade                                 2640 non-null   bool
12  status                                  2640 non-null   object
13  gender                                  2640 non-null   object
14  birthDate                               2564 non-null   object
15  lastName                                2640 non-null   object
16  firstName                               2637 non-null   object
17  title                                   339 non-null    object
18  date                                    2640 non-null   object
19  state                                   753 non-null    object
20  residenceStateRegion                    747 non-null    object
21  birthYear                              2564 non-null   object
22  birthMonth                             2564 non-null   float64
23  birthDay                               2564 non-null   float64
24  cpi_country                             2456 non-null   float32
25  cpi_change_country                      2456 non-null   float32
26  gdp_country                             2476 non-null   object
27  gross_tertiary_education_enrollment     2458 non-null   float32
28  gross_primary_education_enrollment_country 2459 non-null   float32
29  life_expectancy_country                  2458 non-null   float32
30  tax_revenue_country_country              2457 non-null   float32
31  total_tax_rate_country                   2458 non-null   float32
32  population_country                       2476 non-null   float64
33  latitude_country                         2476 non-null   float32
34  longitude_country                       2476 non-null   float32
dtypes: bool(1), float32(9), float64(4), int16(1), int32(1), object(19)
memory usage: 585.4+ KB
None
rank      finalWorth  ...  latitude_country  longitude_country
count  2640.000000  2640.000000  ...      2476.000000      2476.000000
mean    1289.159091  4623.787879  ...      34.903595      12.583156
std      739.693726  9834.240939  ...      17.003498      86.762993
min        1.000000  1000.000000  ...     -40.900558     -106.346771
25%      659.000000  1500.000000  ...      35.861660     -95.712891
50%     1312.000000  2300.000000  ...      37.090240      10.451526
75%     1905.000000  4200.000000  ...      40.463669     104.195396
max     2540.000000 211000.000000  ...      61.924110     174.885971
```

[8 rows x 15 columns]

[5]

 Console ▾

Timing

Datasets ▾

 Charts ▾



## Indentifying Issues with data visualization

Here below sql queries return values where we can see that there are missing values in the dataset in some of columns. Some of Location is missing such state , country and residence of country.

[6]

```
Age,
COUNT(*) AS Billionaire_Count
FROM
billionaires
GROUP BY
age
ORDER BY
Billionaire_Count DESC
```

agevise\_billionaries (80 rows) 

Views

	Age (short)	Billionaire_Count (long)
0	60	88
1	58	83
2	59	82
3	57	75
4	66	74
5	68	73
6	67	73
7	61	72
8	55	72
9	73	71
10	65	69
11		65
12	72	65
13	70	65
14	56	64
15	71	64
16	63	63
17	77	63
18	53	62
19	69	62

[7]



```
SELECT
Country,
COUNT(*) AS Billionaire_Count
FROM
billionaires
GROUP BY
Country
ORDER BY
Billionaire_Count DESC
```

countryvise\_billionaries (79 rows) 

Views

	Country (string)	Billionaire_Count (long)
0	United States	754
1	China	523
2	India	157
3	Germany	102
4	United Kingdom	82
5	Russia	79
6	Switzerland	78
7	Hong Kong	68

Data Curation Project

Projects ▾

Branches ▾

Notebook

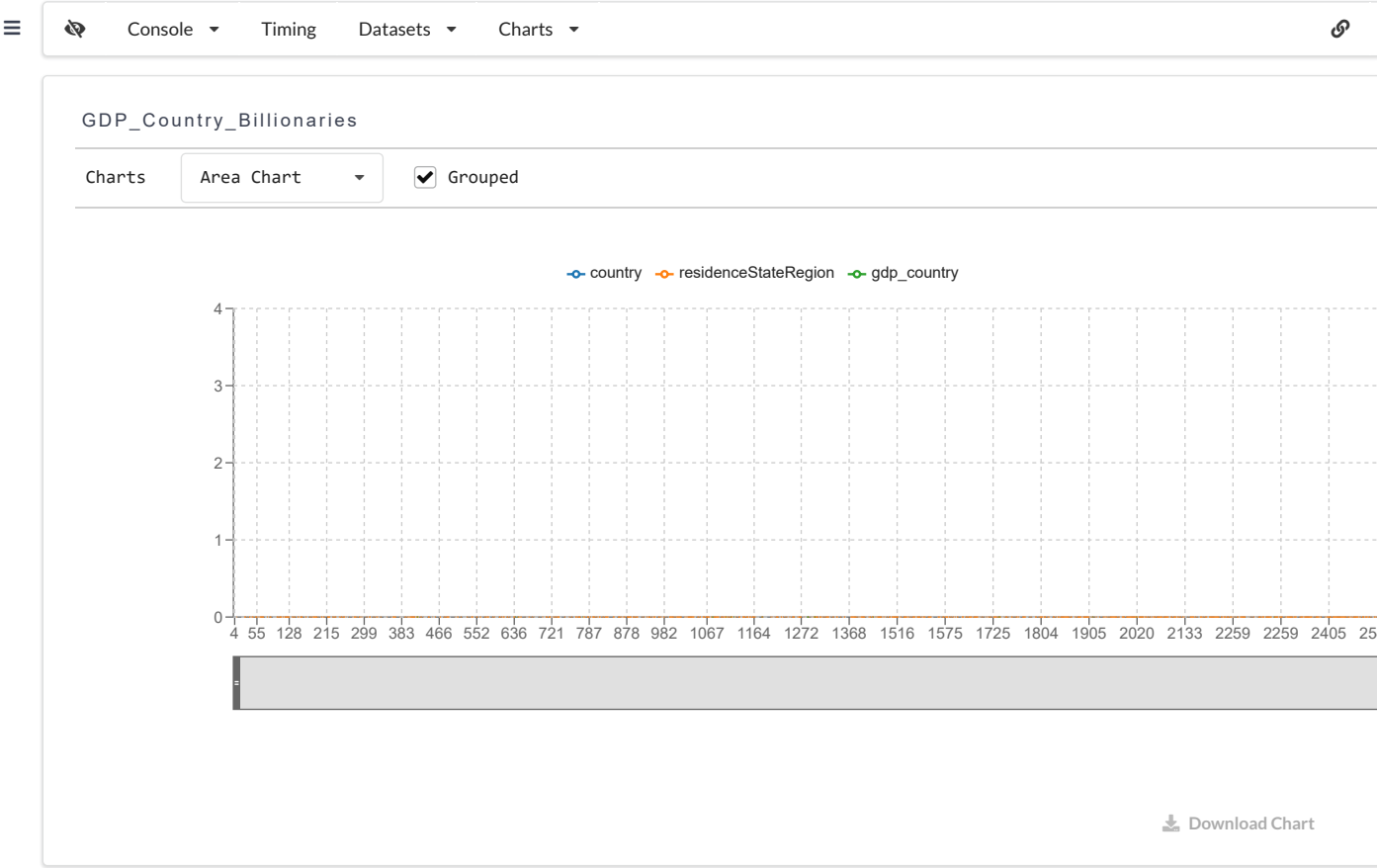
Dataset ▾

Caveats ▾

Settings ▾

9	Singapore	46
10	Brazil	44
11	Taiwan	43
12	Australia	43
13	Canada	42
14		38
15	Japan	38
16	France	35
17	South Korea	29
18	Thailand	28
19	Sweden	26

[8] CREATE Area Chart GDP\_Country\_Billionaires FOR "billionaires"



Console ▾

Timing

Datasets ▾

Charts ▾

## Data Cleaning

[10]

```
import pandas as pd
# Get read-only pandas dataframe object for given dataset.
df = vizierdb.get_data_frame('billionaires');

#Dealing with null values
def missing(df):
    missing_values = df.isnull().sum()
    missing_percent = (missing_values / len(df)) * 100
    print(pd.concat([missing_values, missing_percent], axis=1, keys=['Missing_Number', 'Missing_Percent']))

missing(df)
```



[Console](#) ▾
 [Timing](#)
[Datasets](#) ▾
 [Charts](#) ▾
 

[illegible]

Console ▼ Timing Datasets ▼ Charts ▼


Views

5/10

Data Curation Project

Projects ▾

Branches ▾

 Notebook

Dataset ▾


Caveats ▾

Settings ▾

5	6	104000	Technology	Bill Gates	67	United States	Medina	Microsoft	Techno
6	7	94500	Media & Entertainment	Michael Bloomberg	81	United States	New York	Bloomberg LP	Media &
7	8	93000	Telecom	Carlos Slim Helu & family	83	Mexico	Mexico City	Telecom	Telecor
8	9	83400	Diversified	Mukesh Ambani	65	India	Mumbai	Diversified	Diversif
9	10	80700	Technology	Steve Ballmer	67	United States	Hunts Point	Microsoft	Techno
10	11	80500	Fashion & Retail	Francoise Bettencourt Meyers & family	69	France	Paris	L'Oréal	Fashior
11	12	79200	Technology	Larry Page	50	United States	Palo Alto	Google	Techno
12	13	77300	Fashion & Retail	Amancio Ortega	87	Spain	La Coruna	Zara	Fashior
13	14	76000	Technology	Sergey Brin	49	United States	Los Altos	Google	Techno
14	15	68000	Food & Beverage	Zhong Shanshan	68	China	Hangzhou	Beverages, pharmaceuticals	Food &
15	16	64400	Technology	Mark Zuckerberg	38	United States	Palo Alto	Facebook	Techno
16	17	59000	Diversified	Charles Koch & family	87	United States	Wichita	Koch Industries	Diversif
17	17	59000	Diversified	Julia Koch & family	60	United States	New York	Koch Industries	Diversif
18	19	58800	Fashion & Retail	Jim Walton	74	United States	Bentonville	Walmart	Fashior
19	20	57600	Fashion & Retail	Rob Walton & family	78	United States	Bentonville	Walmart	Fashior

[18]




Console ▾

Timing

Datasets ▾


Charts ▾



Null Values

[19]




Console ▾

Timing

Datasets ▾

Charts ▾



rank

0

finalWorth

0

category

0

personName

0

age

65

country

38

city

72

source

0

industries

0

countryOfCitizenship

0

organization

2315

selfMade

0

status

0

gender

0

birthDate

76

title

2301

date

0

state

1887

residenceStateRegion

1893

cpi\_country

184

cpi\_change\_country

184

gdp\_country

164

gross\_tertiary\_education\_enrollment

182

gross\_primary\_education\_enrollment\_country

181

life\_expectancy\_country

182

tax\_revenue\_country\_country

183

total\_tax\_rate\_country

182

population\_country

164

latitude\_country

164

longitude\_country

164

dtype: int64

[20]



Console ▾

Timing

Datasets ▾

Charts ▾



## Remove time from birthDate column

[21]



```
#Remove time from birthDate column

import pandas as pd
df = vizierdb.get_data_frame('billionaires')

df['birthDate'] = pd.to_datetime(df['birthDate'])
# Extract date component only
df['birthDate'] = df['birthDate'].dt.date

print(df['birthDate'])

print(df.isnull().sum())
```

```
0      1949-03-05
1      1971-06-28
2      1964-01-12
3      1944-08-17
4      1930-08-30
...
2635   1971-12-14
2636   1943-03-10
2637   1962-12-18
2638   1951-08-21
2639   1956-11-01
Name: birthDate, Length: 2640, dtype: object

rank                                0
finalWorth                          0
category                            0
personName                          0
age                                  65
country                             38
city                                 72
source                              0
industries                          0
countryOfCitizenship                 0
organization                         2315
selfMade                             0
status                               0
gender                               0
birthDate                            76
title                               2301
date                                 0
state                               1887
residenceStateRegion                 1893
cpi_country                          184
cpi_change_country                   184
gdp_country                          164
gross_tertiary_education_enrollment  182
gross_primary_education_enrollment_country  181
life_expectancy_country               182
tax_revenue_country_country           183
total_tax_rate_country                 182
population_country                    164
latitude_country                      164
longitude_country                     164
dtype: int64
```

[22]



## Data cleaning for Null Values

[23]

```
1 import pandas as pd
```

```

4 #State Imputing missing values in the 'state' column based on the 'country' column and using the mode of the state within eac
5
6
7 # Get read-only pandas dataframe object for dataset with given name.
8 df = vizierdb.get_data_frame('billionaires')
9
10 #drop longitude and latitude if they are null
11 df.dropna(subset=['latitude_country', 'longitude_country'], inplace=True)
12
13 #age
14 #fill mean value in age
15 df['age'].fillna(df['age'].mean(), inplace=True)
16
17 #country
18 # Function to get missing country values using geopy
19 df.dropna(subset=['latitude_country', 'longitude_country'], inplace=True)
20
21 #age
22 #fill mean value in age
23 df['age'].fillna(df['age'].mean(), inplace=True)
24
25 #country missing value
26 def fill_missing_country(row):
27     if pd.notnull(row['latitude_country']) and pd.notnull(row['longitude_country']):
28         geolocator = Nominatim(user_agent="my_geocoder")
29         location = geolocator.reverse((row['latitude_country'], row['longitude_country']), language="en")
30
31         address = location.raw.get('address', {})
32         state = address.get('country', None)
33
34         return state
35     else:
36         return None
37
38 # Apply the function to fill in missing state values
39 df['country'] = df.apply(fill_missing_country, axis=1)
40 def get_state_from_coordinates(latitude, longitude):
41     geolocator = Nominatim(user_agent="your_app_name") # Replace "your_app_name" with a unique user agent
42
43     location = geolocator.reverse((latitude, longitude), language='en')
44     address = location.raw.get('address', {})
45
46     state = address.get('state', None)
47     return state
48
49 # Load your dataset, assuming it has 'latitude' and 'longitude' columns
50 # and 'state' column with missing values
51 # Iterate through the rows with missing state values
52 for index, row in df[df['state'].isnull()].iterrows():
53     latitude = row['latitude_country']
54     longitude = row['longitude_country']
55
56     # Check if both latitude and longitude are available
57     if not pd.isnull(latitude) and not pd.isnull(longitude):
58         # Get the state from coordinates
59         state = get_state_from_coordinates(latitude, longitude)
60
61         # Update the 'state' column in the dataframe
62         df.at[index, 'state'] = state
63
64 #state
65 #Imputing missing values in the 'state' column based on the 'country' column and using the mode of the city within each speci
66 df['state'] = df.groupby('country')['state'].transform(lambda x: x.fillna(x.mode().iloc[0] if not x.mode().empty else np.nan)
67 df['state'].fillna(df['state'].mode()[0], inplace=True)
68
69 #city
70 #Imputing missing values in the 'city' column based on the 'country' column and using the mode of the city within each specif
71 df['city'] = df.groupby('country')['city'].transform(lambda x: x.fillna(x.mode().iloc[0] if not x.mode().empty else np.nan))
72 df['city'].fillna(df['city'].mode()[0], inplace=True)
73
74
75 # Assuming your DataFrame is named df and the columns are 'country', 'state', and 'residenceStateRegion'
76 df['residenceStateRegion'] = df.groupby(['country', 'state'])['residenceStateRegion'].transform(lambda x: x.fillna(x.mode().i
77 df['residenceStateRegion'].fillna(df['residenceStateRegion'].mode()[0], inplace=True)
78
79 #residenceStateRegion

```



Data Curation Project

Projects ▾

Branches ▾

Notebook

Dataset ▾

Caveats ▾

Settings ▾

```
82 df.loc[(df.country == 'United States') & (df.residenceStateRegion.isna()==True)][ 'state']
83
84 # gdp_country column has object data type because the values start with $ so I will convert it to float
85 df['gdp_country'] = df['gdp_country'].str.replace('$','').replace(',','', regex=True).astype(float)
86 df['gdp_country'] = df.groupby('country')['gdp_country'].transform(lambda x: x.fillna(x.mean()))
87
88 #cpi missing value
89 df['cpi_country'] = df.groupby('country')['cpi_country'].transform(lambda x: x.fillna(x.mean()))
90
91 #organization is based on source so combine that data
92 df['organization'] = df['organization'].combine_first(df['source'])
93
94
95 #Other Columns
96 #We will be imputing values of below columns using the specific country as reference, as these values remain same for that sp
97 columns_to_impute = [
98     'gross_tertiary_education_enrollment',
99     'gross_primary_education_enrollment_country',
100     'life_expectancy_country',
101     'tax_revenue_country_country',
102     'total_tax_rate_country',
103     'population_country',
104     'latitude_country',
105     'longitude_country',
106     'cpi_country',
107     'cpi_change_country'
108 ]
109
110 # Impute missing values for each specified column based on 'country'
111 df[columns_to_impute] = df.groupby('country')[columns_to_impute].transform(lambda x: x.fillna(x.mean() if not x.mode().empty
112
113 # If there are still missing values after group-wise imputation, fill them with the overall mean
114 df[columns_to_impute] = df[columns_to_impute].apply(lambda x: x.fillna(x.mean() if not x.mode().empty else np.nan))
115
116 # Save the updated dataframe
117 df.to_csv('cleaned_data.csv', index=False)
118
119
120 print(df.isnull().sum())
```

i

Show Code Examples

✎

Change Command

✕

Dismiss

🔗

Subn

🔍

Console ▾


Timing

Datasets ▾

Charts ▾


🔗

rank	0
finalWorth	0
category	0
personName	0
age	0
country	0
city	0
source	0
industries	0
countryOfCitizenship	0
organization	0
selfMade	0
status	0
gender	0
birthDate	59
title	2141
date	0
state	0
residenceStateRegion	0
cpi_country	0
cpi_change_country	0
gdp_country	0
gross_tertiary_education_enrollment	0
gross_primary_education_enrollment_country	0
life_expectancy_country	0
tax_revenue_country_country	0
total_tax_rate_country	0
population_country	0
latitude_country	0

 **Data Curation Project**

Projects ▾


Branches ▾

 Notebook

Dataset ▾

Caveats ▾

Settings ▾



Connected to vizier @ <http://localhost:5001/vizier-db/api/v1/> 