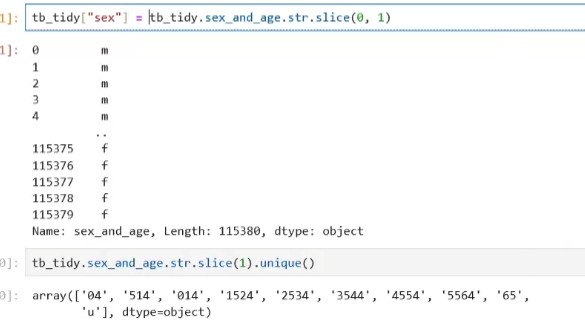
# **Data Tidying and Cleaning**

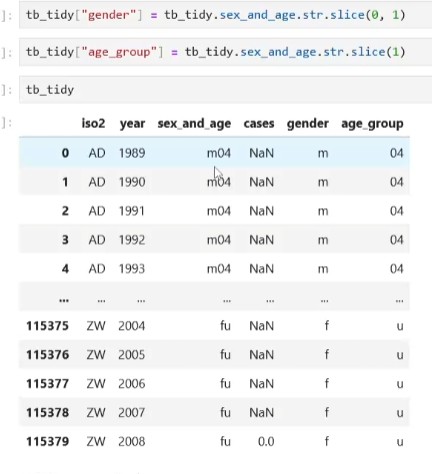
**Melt**

****

**Slice**

****

**Slices**

****

**Drop column**

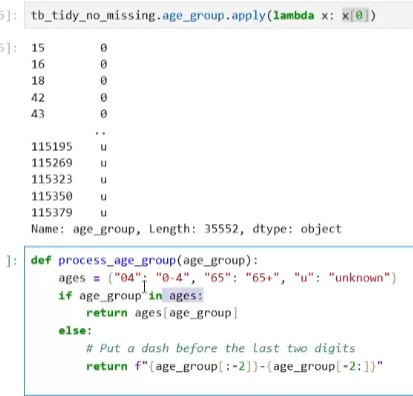
****

**Mean or whatever by condition**

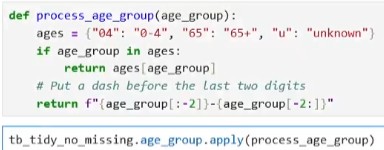
****

**Values count**

**Apply and function**

****

**Apply function**

****

**Split and expand**

**To category**

****

**Rearrange columns**

**Sort by iso2 and then by year**

****

****

**Reset index – преподреждане**

**Презаписване в ново csv**

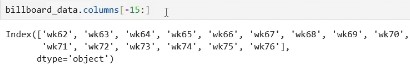
**Дава нова стойност на 2-ри ред, 23 колона**

**Melt, slice, dropna и оставям само тези, които ми трябват**

**Pivot**

****

**Which are in week 50? - notna**

**Last 15 columns**

**Melt more examples**

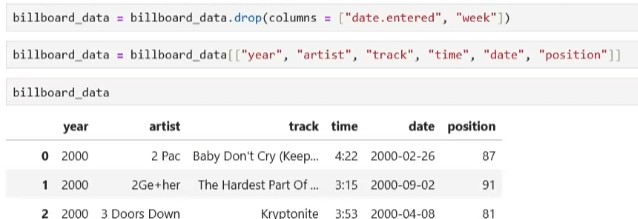
**Slice and astype**

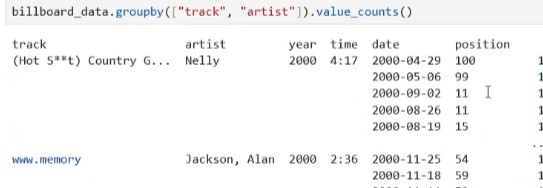
**Data datetime**

**Add week of the year**

****

**Drop and rearange columns**

****

**Group by**

**Condition if**

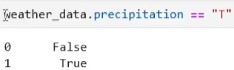
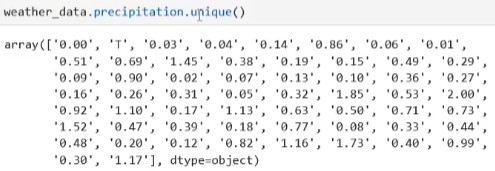
****

**Read csv from link – processing, basic steps**

**Convert column names**

**To datetime**

**Dtypes**

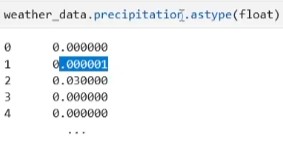
**Object column analysis**

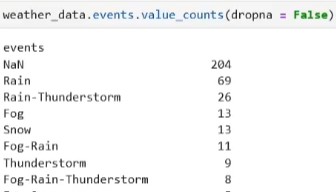
**How many are with ‘T’**

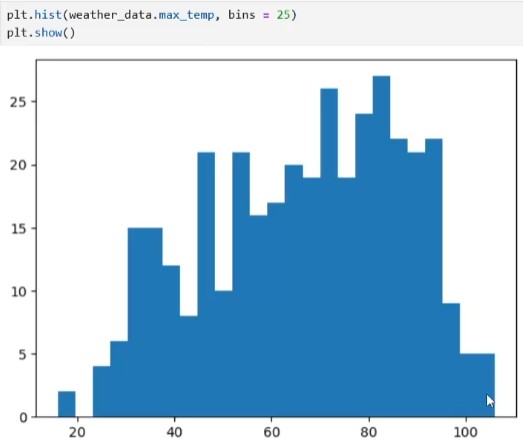
**Replace column value ‘T’ with a very small number**

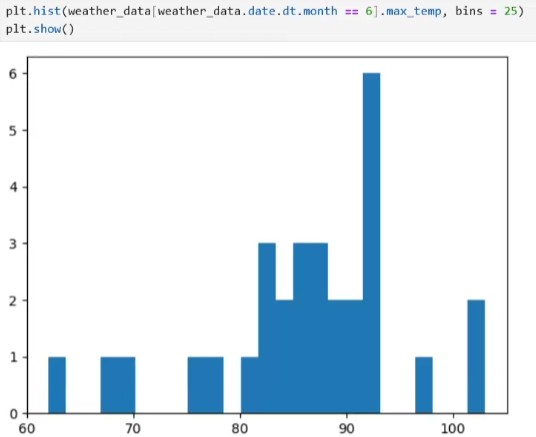
****

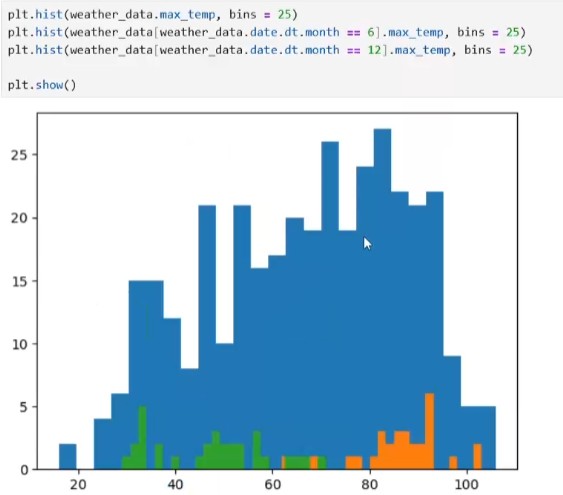
**Astype float**

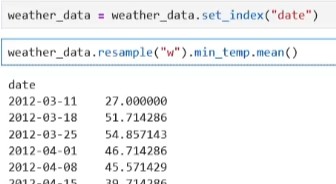
****

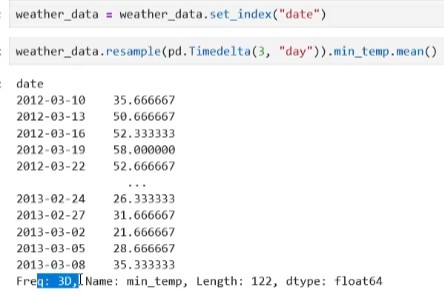
**Count events including NAN**

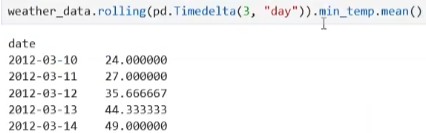
**Hist**

**Hist max temperatures for june – 6**

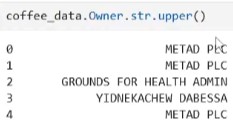
**Plot all together**

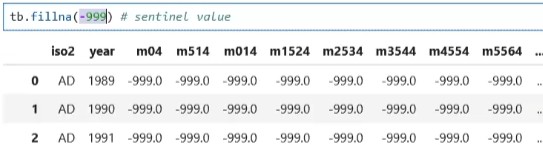
**Average temperature for every week**

**Min temp for every 3 days – by 3 days – 10.03/13.03/16.03**

**Or rolling – there is overlap – every day 10.03/11.03/12.03**

**Str transformations**

****

**We could replace the missing data with a dummy value**

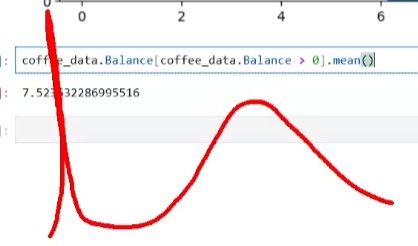
**Median Conclusion:**

* **Use the mean when your data is normally distributed without outliers.**
* **Use the median when your data is skewed or contains outliers.**

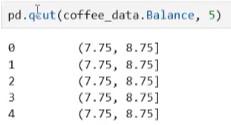
****

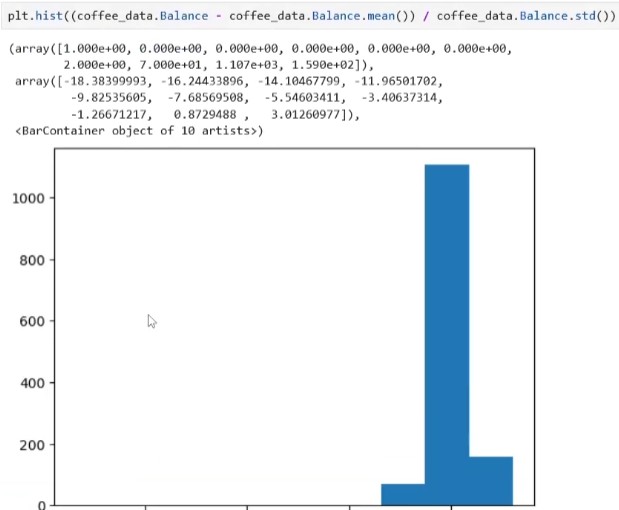
**Mean of only positive example - when we have outliers DATASET TRANSFORMATIONS**

**<https://scikit-learn.org/stable/modules/preprocessing.html#non-linear-transformation>**

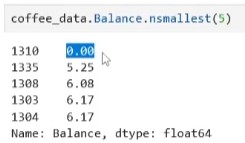
****

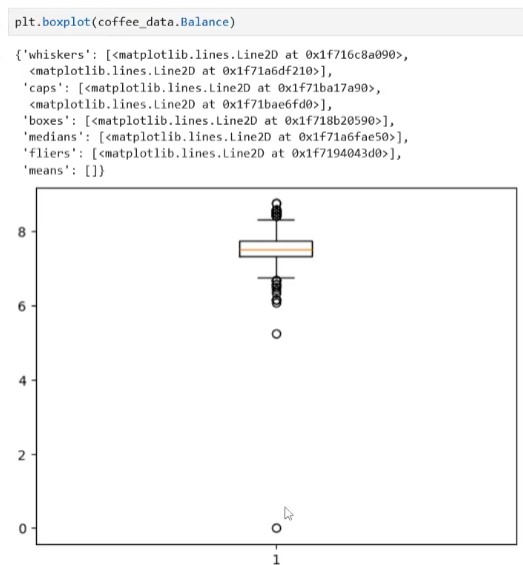
**QCUT**

****

****

**Find smallest 5**

****

**Find outliers**

****

**Median Conclusion:**

* Use the mean when your data is normally distributed without outliers.
* Use the median when your data is skewed or contains outliers.

**Mean of only positive example - when we have outliers DATASET TRANSFORMATIONS**

**<https://scikit-learn.org/stable/modules/preprocessing.html#non-linear-transformation>**

Pandas api request with authentication ChatGPT

To make an API request with authentication and load the data into a pandas DataFrame, you can use Python's requests library. Below is a step-by-step guide for handling API requests that require authentication, such as using an API key or basic authentication.

1. Install Required Libraries

If you haven't installed pandas or requests, you can install them using pip: bash

Копиране на код

pip install pandas requests

1. Import Libraries

Start by importing the necessary libraries:

python Копиране на код

import pandas as pd import requests

1. Set Up the Authentication

Depending on the type of authentication the API requires, you can set it up as follows: API Key in Headers

If the API uses an API key in the headers: python

Копиране на код

url = "[https://api.example.com/data"](https://api.example.com/data%22) headers = {

"Authorization": "Bearer YOUR\_API\_KEY", "Content-Type": "application/json"

}

response = requests.get(url, headers=headers) Basic Authentication

If the API uses basic authentication (username and password): python

Копиране на код

url = "[https://api.example.com/data"](https://api.example.com/data%22)

auth = ("your\_username", "your\_password")

response = requests.get(url, auth=auth)

1. Convert the JSON Response to a DataFrame

Assuming the API returns data in JSON format, convert the response to a pandas DataFrame:

python Копиране на код

data = response.json() df = pd.DataFrame(data)

# **Display the DataFrame**

print(df.head()) Example Workflow

Here’s an example that combines all the steps: python

Копиране на код import pandas as pd import requests

# **Example API URL**

url = "[https://api.example.com/data"](https://api.example.com/data%22)

# **Headers for API Key Authentication**

headers = {

"Authorization": "Bearer YOUR\_API\_KEY", "Content-Type": "application/json"

}

# **Make the GET request with headers**

response = requests.get(url, headers=headers)

# **Check if the request was successful**

if response.status\_code == 200:

# Convert the JSON response to a DataFrame data = response.json()

df = pd.DataFrame(data)

# Display the DataFrame print(df.head())

else:

print(f"Failed to retrieve data: {response.status\_code}") Notes:

* Handling Errors: It's good practice to check the response status code before attempting to parse the JSON.
* Pagination: If the API paginates the results, you may need to loop through pages and combine the data.
* Rate Limits: Be mindful of API rate limits, and handle them appropriately (e.g., by adding delays or checking headers for rate limit information).

This method enables you to securely access API data using authentication and load it into a pandas DataFrame for further analysis.