User

I have List of Lists in python as Lst=[[1,2],[3,4],[5,6]]. Can you give me code that separates these lists inside list individually and multiplies each list by 5?

Bing

Sure, here is a simple Python code snippet that does what you asked for:

Lst = [[1,2],[3,4],[5,6]]

# Separate lists and multiply each by 5

new\_Lst = [[element\*5 for element in sub\_list] for sub\_list in Lst]

print(new\_Lst)

This code uses list comprehension to iterate over each sublist in Lst, and then over each element in the sublist. It multiplies each element by 5 and stores the results in a new list of lists, new\_Lst. The print statement will output the new list.

User

That produces output like [[5, 10], [15, 20], [25, 30]]. I need output like [5,10], [15,20] & [25,30] - 3separate lists.

Bing

I see, you want to print each list separately. Here's how you can do it:

Lst = [[1,2],[3,4],[5,6]]

# Separate lists and multiply each by 5

new\_Lst = [[element\*5 for element in sub\_list] for sub\_list in Lst]

# Print each list separately

for i in new\_Lst:

print(i)

This code will print each list on a new line. If you want to print them on the same line with a comma and space between them, you can modify the print statement like this:

print(', '.join(map(str, new\_Lst)))

This will give you the output: [5, 10], [15, 20], [25, 30]. Each list is separated by a comma and a space. Please note that the lists are still part of a larger list. If you want to work with them individually, you should refer to them by their index in new\_Lst. For example, new\_Lst[0] would give you [5, 10].

User

I do not want to print them. I want to store them like Lst1, Lst2, Lst3, etc.

Bing

I understand. However, in Python, it's not a good practice to dynamically create variables like Lst1, Lst2, etc. It can make the code less readable and more error-prone.

Instead, you can store the lists in a dictionary where the keys are the names of the lists. Here's how you can do it:

Lst = [[1,2],[3,4],[5,6]]

# Separate lists and multiply each by 5

new\_Lst = [[element\*5 for element in sub\_list] for sub\_list in Lst]

# Store lists in a dictionary

dict\_of\_lists = {}

for i in range(len(new\_Lst)):

dict\_of\_lists['Lst'+str(i+1)] = new\_Lst[i]

# Now you can access the lists as dict\_of\_lists['Lst1'], dict\_of\_lists['Lst2'], etc.

In this code, dict\_of\_lists['Lst1'] will give you the first list [5, 10], dict\_of\_lists['Lst2'] will give you the second list [15, 20], and so on. This way, you can easily access each list individually. It's a more Pythonic way to handle this situation. I hope this helps! 😊

User

Can you define UDF in python which takes input arguments as Lst = [[1,2],[3,4],[5,6]]

# Separate lists and multiply each by 5

new\_Lst = [[element\*5 for element in sub\_list] for sub\_list in Lst]

Bing

Sure, I can help you define a User Defined Function (UDF) in Python that takes a list of lists as an argument and returns a new list where each element is multiplied by 5. Here's how you can do it:

def multiply\_elements(Lst):

# Separate lists and multiply each by 5

new\_Lst = [[element\*5 for element in sub\_list] for sub\_list in Lst]

return new\_Lst

# Usage:

Lst = [[1,2],[3,4],[5,6]]

result = multiply\_elements(Lst)

print(result) # prints: [[5, 10], [15, 20], [25, 30]]

In this code, multiply\_elements is a function that takes one argument, Lst. It returns a new list of lists where each element is multiplied by 5. You can call this function with your list as an argument to get the desired result. The print statement is used to display the result. I hope this helps! 😊

User

I have defined UDF in file xlwing\_trial.py as under:

import xlwings as xw
@xw.func
@xw.arg('Lst', ndim=2)
def multiply\_elements(Lst):
# Separate lists and multiply each by 5
new\_Lst = [[element\*5 for element in sub\_list] for sub\_list in Lst]
return new\_Lst

Next, I went to excel sheet and in cells A1 to B4 inserted values as:

1 4
8 3
7 0
2 6

Next, I type in Cell D1 formula as =multiply\_elements(A1:B4)

Then I clicked on import function button on xlwings tab, I saw error in excel as vba runtime error 1004 method vbporject of object \_workbook failed. Can you resolve?
I tried to import UDF by clicking on that import function button on xlwings tab, but there is a vba error dialogue box in excel

Bing

The error you're encountering, Run-time error '1004': Method 'VBProject' of object '\_Workbook' failed, is often due to Excel's security settings. Specifically, Excel needs to be set to "Trust access to the VBA project object model". Here's how you can do this:

1. Open Excel and go to `File > Options > Trust Center`.
2. Click on `Trust Center Settings`.
3. In the Trust Center dialog box, click on `Macro Settings` in the left pane.
4. Check the box for `Trust access to the VBA project object model`.
5. Click `OK` and then `OK` again to save the settings.

After doing this, try importing your UDF again.

If you're still encountering issues, there might be a problem with your xlwings setup or your Python environment. Make sure that your Python environment is properly set up and that xlwings is correctly installed and configured.

Please note that modifying these settings changes Excel's security level and could potentially make your computer more vulnerable. Only change these settings if you understand the risks involved.

I hope this helps! If you have any other questions or need further clarification, feel free to ask. 😊

User

I did all the above steps. Now that error disappeared and in console also I can see "Imported functions from the following modules: xlwings\_trial" but in excel cell D1 where I typed =multiply\_elements(A1:B4) instead of result array on application of function, I see message "object required"

Bing

The "Object Required" error in Excel usually occurs when a referenced object is not complete, is not valid, or is not specified correctly. This might happen if Excel is not able to find the function multiply\_elements in the Python module xlwings\_trial that you mentioned.

Here are a few things you can check:

1. Make sure that the Python file containing your function (`xlwings\_trial.py`) is in the same directory as your Excel file.
2. Ensure that the Python function is correctly defined with the `@xw.func` decorator and that it's correctly handling the input array.
3. Check if there's any issue with the data types. The function should be able to handle the data type of the cell values it's receiving from Excel.
4. Make sure that you have trusted access to the VBA project object model.
5. You need to add a reference to xlwings in VBA. Open up the Developer console (Alt+F11), click on Tools -> References and select xlwings.

If you've checked all of these and you're still having issues, there might be a problem with your xlwings setup or your Python environment. Make sure that your Python environment is properly set up and that xlwings is correctly installed and configured.

If none of these solutions work, please provide more details or consider reaching out to the xlwings community or support for more specific assistance. I hope this helps! 😊

User

Last step helped me:

You need to add a reference to xlwings in VBA1. Open up the Developer console (Alt+F11), click on Tools -> References and select xlwings1.

Now I can see the results of UDF as a dynamic array! That was really an issue which was reolved.👌

Bing

I'm glad to hear that the issue was resolved and you're now able to see the results of your User Defined Function as a dynamic array in Excel! It's great that you were able to successfully integrate Python with Excel using xlwings. If you have any more questions or need further assistance, feel free to ask. Happy coding! 😊

User

I want ot use another UDF. It is defined as:
import xlwings as xw
@xw.func
@xw.arg('Days', ndim=2)
def aging\_buckets(Days):
aging\_dict = {
range(-sys.maxsize, 0): "Advance",
range(0, 31): "Not Due",
range(31, 61): "31-60 Days",
range(61, 91): "61-90 Days",
range(91, 181): "91-180 Days",
range(181, 366): "6 Months to 1 year",
range(366, sys.maxsize): "More than 1 year"
}
for r, value in aging\_dict.items():
if Days in r:
return value

Issue is

Bing

It seems like your message got cut off before you could explain the issue you're facing with your User Defined Function (UDF). Could you please provide more details about the problem? For example, are you receiving any error messages? Or is the function not behaving as expected? The more information you can provide, the better I can assist you. 😊

User

Sure. Again reproducing the message:

I want ot use another UDF. It is defined as:
import xlwings as xw
@xw.func
@xw.arg('Days', ndim=2)
def aging\_buckets(Days):
aging\_dict = {
range(-sys.maxsize, 0): "Advance",
range(0, 31): "Not Due",
range(31, 61): "31-60 Days",
range(61, 91): "61-90 Days",
range(91, 181): "91-180 Days",
range(181, 366): "6 Months to 1 year",
range(366, sys.maxsize): "More than 1 year"
}
for r, value in aging\_dict.items():
if Days in r:
return value

The issue is my input is an array (ndim=2) It is one column with several rows in excel. However, I think the function runs on one element at a time. Can you redefine above function which takes array as argument (Lists of List - since it is an excel range) and returns array?

Bing

Sure, I understand your requirement. You want the function to take a 2D array (list of lists) as input and return a 2D array as output. Here's how you can modify your function to achieve this:

import xlwings as xw

import sys

@xw.func

@xw.arg('Days', ndim=2)

def aging\_buckets(Days):

aging\_dict = {

range(-sys.maxsize, 0): "Advance",

range(0, 31): "Not Due",

range(31, 61): "31-60 Days",

range(61, 91): "61-90 Days",

range(91, 181): "91-180 Days",

range(181, 366): "6 Months to 1 year",

range(366, sys.maxsize): "More than 1 year"

}

result = []

for day\_list in Days:

result.append([next(value for r, value in aging\_dict.items() if day in r) for day in day\_list])

return result

In this modified function, Days is a list of lists (2D array). The function iterates over each sublist in Days, and for each day in the sublist, it finds the corresponding value in aging\_dict and appends it to the result list. The result list is also a list of lists (2D array), so it should work with your Excel range.

Please note that this function assumes that all elements in Days are integers. If there are non-integer values, you might need to add error handling code.

I hope this helps! If you have any other questions or need further assistance, feel free to ask. 😊

User

Actual my python code file where the function defined is like this:

import xlwings as xw
import sys
@xw.func
@xw.arg('Lst', ndim=2)
def multiply\_elements(Lst):
# Separate lists and multiply each by 5
new\_Lst = [[(element\*5)+3 for element in sub\_list] for sub\_list in Lst]
return new\_Lst

@xw.func
@xw.arg('Days', ndim=2)
def aging\_buckets(Days):
aging\_dict = {
range(-sys.maxsize, 0): "Advance",
range(0, 31): "Not Due",
range(31, 61): "31-60 Days",
range(61, 91): "61-90 Days",
range(91, 181): "91-180 Days",
range(181, 366): "6 Months to 1 year",
range(366, sys.maxsize): "More than 1 year"
}
result = []
for day\_list in Days:
result.append([next(value for r, value in aging\_dict.items() if day in r) for day in day\_list])
return result

As you can see, there are two functions. In cell B2 I am giving formula as =aging\_buckets(A2:A1012)

It goes into some loop and never comes out.

Bing

I see, it seems like the function might be stuck in an infinite loop or taking a long time to process due to the large input size. Here are a few things you could try:

1. \*\*Check your input data\*\*: Make sure the data in the range A2:A1012 is as expected and there are no unexpected values that could cause the function to behave unexpectedly.
2. \*\*Debug your function\*\*: You can add print statements in your function to see where it might be getting stuck. This can help you identify if there's a specific part of your function that's causing the issue.
3. \*\*Optimize your function\*\*: If the function is taking a long time due to the large input size, you might need to optimize your function. For example, you could try using a more efficient data structure for `aging\_dict` that allows faster lookups.
4. \*\*Use array formulas\*\*: According to the [xlwings documentation](^2^), calling one big array formula in Excel is much more efficient than calling many single-cell formulas. You can pass an Excel Range as a function argument, as opposed to a single cell and it will show up in Python as list of lists.
5. \*\*Restart UDF Server\*\*: If you want to make sure everything is in a fresh state, click Restart UDF Server.

If none of these solutions work, please provide more details or consider reaching out to the xlwings community or support for more specific assistance. I hope this helps! 😊

User

Meanwhile, as I was watching your above message, I saw error message as "Automation Error - Remote Procedure Call Failed"

Bing

I'm sorry to hear that you're encountering this error. The "Automation Error - Remote Procedure Call Failed" is a common issue that can occur due to various reasons. Here are a few potential solutions:

1. \*\*Restart the RPC service\*\*: Press Windows + R to open a Run dialog. Type `services.msc` in the box that appears. Find and right-click on the "Remote Procedure Call" service in the Services window and select the "Refresh" option.
2. \*\*Check your Excel and Python setup\*\*: Make sure that your Excel file (macro-enabled) and Python module have different names. Also, ensure that you have added the default xlwings package from Anaconda to Excel Add-ins and added it to references in VBA. You might need to restart the UDF server.
3. \*\*Check your code\*\*: In some cases, this error can occur if your code is trying to perform too many operations at once or if it's working with a large amount of data. You might need to optimize your code or break down your operations into smaller parts.
4. \*\*Check your system's resources\*\*: If your system is running low on resources, it might not be able to handle the operations you're trying to perform. Try closing any unnecessary applications or processes and then try again.

If none of these solutions work, please provide more details or consider reaching out to the xlwings community or support for more specific assistance. I hope this helps! 😊

User

To test my code, I tried as under:
import sys
Days=[236,161,165,407,-97,52,85]
def aging\_buckets(Days):
aging\_dict = {
range(-sys.maxsize, 0): "Advance",
range(0, 31): "Not Due",
range(31, 61): "31-60 Days",
range(61, 91): "61-90 Days",
range(91, 181): "91-180 Days",
range(181, 366): "6 Months to 1 year",
range(366, sys.maxsize): "More than 1 year"
}
result = []
for day\_list in Days:
result.append([next(value for r, value in aging\_dict.items() if day in r) for day in day\_list])
return result
res=aging\_buckets(Days)

It returned error:

Traceback (most recent call last):
File "<stdin>", line 1, in <module>
File "<stdin>", line 13, in aging\_buckets
TypeError: 'int' object is not iterable

Bing

The error message is indicating that you're trying to iterate over an integer, which is not possible in Python. In your function aging\_buckets, you're treating each day in Days as if it's a list, but Days is actually a list of integers, not a list of lists.

Here's how you can modify your function to work with a list of integers:

import sys

def aging\_buckets(Days):

aging\_dict = {

range(-sys.maxsize, 0): "Advance",

range(0, 31): "Not Due",

range(31, 61): "31-60 Days",

range(61, 91): "61-90 Days",

range(91, 181): "91-180 Days",

range(181, 366): "6 Months to 1 year",

range(366, sys.maxsize): "More than 1 year"

}

result = [next(value for r, value in aging\_dict.items() if day in r) for day in Days]

return result

Days=[236,161,165,407,-97,52,85]

res=aging\_buckets(Days)

print(res)

In this modified function, the list comprehension directly iterates over Days, treating each day as an integer. The function should now work as expected with your input. I hope this helps! 😊

User

That one run successfully in python. So What I did was to change my xlwings py code as under:
@xw.func
@xw.arg('Days', ndim=2)
def aging\_buckets(Days):
aging\_dict = {
range(-sys.maxsize, 0): "Advance",
range(0, 31): "Not Due",
range(31, 61): "31-60 Days",
range(61, 91): "61-90 Days",
range(91, 181): "91-180 Days",
range(181, 366): "6 Months to 1 year",
range(366, sys.maxsize): "More than 1 year"
}
result = [next(value for r, value in aging\_dict.items() if day in r) for day in Days]
return result

But still when I tried this on a smaller range of cells =aging\_buckets(A2:A10), still it went into loop. Does xlwings plug-in allow only one python udf in py file which is linked in excel. As I stated above there are two functions in that xlwings py file.

Bing

The xlwings library does allow multiple User Defined Functions (UDFs) in the same Python file. The issue you're experiencing might be due to other factors. Here are a few things you could try:

1. \*\*Check your function definitions\*\*: Make sure that both of your functions are correctly defined with the `@xw.func` decorator and that they're correctly handling the input array.
2. \*\*Check your Excel and Python setup\*\*: Make sure that your Excel file and Python module have different names. Also, ensure that you have added the default xlwings package from Anaconda to Excel Add-ins and added it to references in VBA.
3. \*\*Restart UDF Server\*\*: If you want to make sure everything is in a fresh state, click Restart UDF Server.
4. \*\*Debug your function\*\*: You can add print statements in your function to see where it might be getting stuck. This can help you identify if there's a specific part of your function that's causing the issue.

If none of these solutions work, please provide more details or consider reaching out to the xlwings community or support for more specific assistance. I hope this helps! 😊

User

Can changing this:
result = [next(value for r, value in aging\_dict.items() if day in r) for day in Days]
will make difference? I read that excel range input is taken as list of list in xlwings python plug-in for UDF.

Bing

Yes, you're correct. In xlwings, a range of cells from Excel is passed to Python as a list of lists (2D array), where each list represents a row of cells. So if you're passing a column of cells from Excel to your UDF, it will be received as a list of lists in Python, where each list contains one element.

Given this, you should modify your function to correctly handle this input format. Here's how you can do it:

import xlwings as xw

import sys

@xw.func

@xw.arg('Days', ndim=2)

def aging\_buckets(Days):

aging\_dict = {

range(-sys.maxsize, 0): "Advance",

range(0, 31): "Not Due",

range(31, 61): "31-60 Days",

range(61, 91): "61-90 Days",

range(91, 181): "91-180 Days",

range(181, 366): "6 Months to 1 year",

range(366, sys.maxsize): "More than 1 year"

}

result = [[next(value for r, value in aging\_dict.items() if day[0] in r)] for day in Days]

return result

In this modified function, day[0] is used to access the first (and only) element of each list in Days. The function should now work correctly with your Excel range. I hope this helps! 😊

User

I tried again with above code giving function in excel cell B2 as =aging\_buckets(A2:A10) and then pressed enter but it is stuck and no ouput comes out. Does xlwings handles python dictionaries correctly because above UDF contain that. Can you see whether I have made any mistake in giving function command as above. I also retried restarting UDF server.

Bing

The xlwings library does handle Python dictionaries correctly, and your function seems to be defined correctly as well. However, there could be several reasons why your UDF is not working as expected. Here are a few things you could try:

1. \*\*Check your Excel and Python setup\*\*: Make sure that your Excel file and Python module have different names. Also, ensure that you have added the default xlwings package from Anaconda to Excel Add-ins and added it to references in VBA.
2. \*\*Debug your function\*\*: You can add print statements in your function to see where it might be getting stuck. This can help you identify if there's a specific part of your function that's causing the issue.
3. \*\*Check your system's resources\*\*: If your system is running low on resources, it might not be able to handle the operations you're trying to perform. Try closing any unnecessary applications or processes and then try again.
4. \*\*Reinstall Python and xlwings\*\*: Some users have reported that uninstalling their Python version and reinstalling it using the Anaconda distribution resolved their issues. After reinstalling, you can install the xlwings add-in from the Anaconda prompt by typing `xlwings addin install`.

If none of these solutions work, please provide more details or consider reaching out to the xlwings community or support for more specific assistance. I hope this helps! 😊

User

When I give a range like this:

236
161
165
407
486
442
642
361
-91
13

In xlwings how it is treated as list of lists? Suppose that above range represents cell A2:A11.

Bing

In xlwings, when you pass a range of cells from Excel to a Python function, it's passed as a list of lists (2D array), where each list represents a row of cells.

So, if you're passing a column of cells from Excel (like A2:A11 in your example), it will be received in Python as a list of lists, where each list contains one element.

For your specific example, the range A2:A11 would be represented in Python as:

[[236], [161], [165], [407], [486], [442], [642], [361], [-91], [13]]

Each inner list corresponds to a row in the Excel range. For example, [236] corresponds to cell A2, [161] corresponds to cell A3, and so on. This is why we need to use day[0] in the function aging\_buckets to access the first (and only) element of each list. I hope this clarifies your doubt! 😊

User

Does: [[236], [161], [165], [407], [486], [442], [642], [361], [-91], [13]] mean one dimensional or two dimensional array?

Bing

The list [[236], [161], [165], [407], [486], [442], [642], [361], [-91], [13]] is a two-dimensional array.

In Python, a one-dimensional array would look like this: [236, 161, 165, 407, 486, 442, 642, 361, -91, 13].

However, the array you provided contains ten lists (each with one element), making it a two-dimensional array. The outer list contains ten elements (which are themselves lists), and each inner list contains one element. I hope this helps! 😊

User

I have a single column csv file like this:
236
161
165
407
486
442
642
361
-91
13
214
37

Can you give me python code which reads the file and convert into list of list like [[161],[165],[361]] ...

Bing

Sure, I can help you with that. Here's a simple Python code snippet that reads a single column CSV file and converts it into a list of lists:

import csv

# Open the CSV file

with open('your\_file.csv', 'r') as f:

reader = csv.reader(f)

# Convert each row to a list and store in a new list

result = [[int(row[0])] for row in reader]

print(result)

In this code, replace 'your\_file.csv' with the actual path to your CSV file. This code reads the file, converts each row to a list (with one element), and stores these lists in the result list. The print statement will output the result list.

Please note that this code assumes that all elements in your CSV file are integers. If there are non-integer values, you might need to add error handling code. I hope this helps! 😊

User

bing ai

Bing

Never waste time searching again – our new engine gives you the answers you need! We started with Microsoft Bing's rich results and made them even better with AI technology.

User

I think in xlwings, to use multiple functions (UDF) from same python .py file, we need to load a particular function as a module in a UDF modules option. How to do it?

Bing

Yes, you're correct. In xlwings, you can specify a particular function to be loaded as a module in the UDF Modules option. Here's how you can do it:

1. Open your Excel workbook.
2. Click on the xlwings tab in the Excel ribbon.
3. In the UDF Modules field, enter the name of the Python module (without the .py extension) that contains your function.
4. Click on "Import Python UDFs" to import the functions from the specified module into Excel.

Please note that the Python file (module) should be in the same directory as your Excel file or in a directory that's included in your PYTHONPATH. Also, make sure that you've installed the xlwings add-in and enabled "Trust access to the VBA project object model" in Excel's Trust Center settings.

I hope this helps! If you have any other questions or need further assistance, feel free to ask. 😊

User

When I want to restart udf server, I get vba error 1000 - no command specified in configuration, cannot autostart server

Bing

The error "No command specified in the configuration, cannot autostart server" typically occurs when the Python interpreter path is not correctly specified in the xlwings configuration.

Here are a few steps you can follow to resolve this issue:

1. \*\*Specify the Python interpreter path in the xlwings configuration file\*\*: The xlwings configuration file (xlwings.conf) should be located in the same directory as your Excel workbook. Open this file and make sure that the path to your Python interpreter is correctly specified under the "INTERPRETER" setting.
2. \*\*Check your Excel and Python setup\*\*: Make sure that your Excel file and Python module have different names. Also, ensure that you have added the default xlwings package from Anaconda to Excel Add-ins and added it to references in VBA.
3. \*\*Restart UDF Server\*\*: If you want to make sure everything is in a fresh state, click Restart UDF Server.

If none of these solutions work, please provide more details or consider reaching out to the xlwings community or support for more specific assistance. I hope this helps! 😊

User

## I am getting following message when I click on "Import Functions" button in xlwings tab:

## Error

No command specified in the configuration, cannot autostart server 1000

## Press Ctrl+C to copy this message to the clipboard.

## OK

Bing

The error "No command specified in the configuration, cannot autostart server 1000" typically occurs when the Python interpreter path is not correctly specified in the xlwings configuration.

Here are a few steps you can follow to resolve this issue:

1. \*\*Specify the Python interpreter path in the xlwings configuration file\*\*: The xlwings configuration file (xlwings.conf) should be located in the same directory as your Excel workbook. Open this file and make sure that the path to your Python interpreter is correctly specified under the "INTERPRETER" setting.
2. \*\*Check your Excel and Python setup\*\*: Make sure that your Excel file and Python module have different names. Also, ensure that you have added the default xlwings package from Anaconda to Excel Add-ins and added it to references in VBA.
3. \*\*Restart UDF Server\*\*: If you want to make sure everything is in a fresh state, click Restart UDF Server.

If none of these solutions work, please provide more details or consider reaching out to the xlwings community or support for more specific assistance. I hope this helps! 😊

User

As for following link on github: https://github.com/xlwings/xlwings/issues/1785
taking guidance by one user's remarks I did following:

try to reset the configuration by running xlwings config create --force (running this from a Command Prompt where the desired Python installation is the active one).

I ran this command: xlwings config create --force after changing directory to d:\programs\python in Powershell as per above guidance. IS it correct?

Bing

Yes, you're correct. The command xlwings config create --force is used to reset the xlwings configuration. This command creates the user config file (~/.xlwings/xlwings.conf) which is where the settings from the Ribbon add-in are stored. It configures the Python interpreter that you are running this command with. Running this command with the --force flag will overwrite your current configuration.

You should run this command from a Command Prompt or PowerShell where the desired Python installation (the one with xlwings installed) is the active one. Changing the directory to where your Python is installed before running the command, as you did, is a good practice.

Please note that after running this command, you may need to restart Excel or the UDF server for the changes to take effect.

I hope this helps! If you have any other questions or need further assistance, feel free to ask. 😊

User

In windows where ~/.xlwings is located?

Bing

In Windows, the .xlwings directory is located in your user folder. The full path is usually C:\Users\<username>\.xlwings. Here, <username> should be replaced with your actual username on your Windows system. I hope this helps! 😊

User

configuration file xlwings.conf has contents like:

"UDF MODULES",""
"USE UDF SERVER","True"
"SHOW CONSOLE","True"
"INTERPRETER\_WIN","d:\programs\python\python.exe"
"DEBUG UDFS","True"

Is it correctly formatted?

Bing

Yes, your xlwings.conf file appears to be correctly formatted. The settings in the xlwings.conf file are stored as key-value pairs. Here's what each setting does:

* `"UDF MODULES",""`: This specifies the names of Python modules (without .py extension) from which the UDFs are being imported. Separate multiple modules by “;”. If it's empty, it imports a file in the same directory as the Excel spreadsheet with the same name but ending in .py.
* `"USE UDF SERVER","True"`: This setting determines whether to use the same COM Server for RunPython as for UDFs. This will be faster, as the interpreter doesn’t shut down after each call.
* `"SHOW CONSOLE","True"`: If this is set to "True", the Python console will be displayed when running your UDFs.
* `"INTERPRETER\_WIN","d:\programs\python\python.exe"`: This is the path to your Python interpreter.
* `"DEBUG UDFS","True"`: If this is set to "True", you can run the xlwings COM server manually for debugging.

Please note that you may need to restart Excel or the UDF server for changes in the xlwings.conf file to take effect. I hope this helps! 😊

User

What if I change: "INTERPRETER\_WIN","d:\programs\python\python.exe": This is the path to your Python interpreter12.
to
"INTERPRETER","d:\programs\python\python.exe": This is the path to your Python interpreter12.

Bing

The INTERPRETER\_WIN setting in the xlwings.conf file is used to specify the path to the Python interpreter on Windows. If you're using xlwings on a Windows machine, you should use INTERPRETER\_WIN.

The INTERPRETER setting is a more general setting that can be used on both Windows and Mac. However, in the settings, this is stored as INTERPRETER\_WIN or INTERPRETER\_MAC, respectively.

So, if you're using xlwings on a Windows machine, it's recommended to use INTERPRETER\_WIN. If you're using xlwings on a Mac, use INTERPRETER\_MAC. If you want to specify a general setting that applies to both Windows and Mac, you can use INTERPRETER.

Please note that after modifying the xlwings.conf file, you may need to restart Excel or the UDF server for the changes to take effect. I hope this helps! 😊

User

When I restarted as per your guidance above: vba error message is Error 1000 no command specified even though as stated above, configuration file does include interpreter\_win command

Bing

I'm sorry to hear that you're still encountering this issue. The error "No command specified in the configuration, cannot autostart server 1000" typically occurs when the Python interpreter path is not correctly specified in the xlwings configuration.

Here are a few steps you can follow to resolve this issue:

1. \*\*Specify the Python interpreter path in the xlwings configuration file\*\*: The xlwings configuration file (xlwings.conf) should be located in the same directory as your Excel workbook. Open this file and make sure that the path to your Python interpreter is correctly specified under the "INTERPRETER\_WIN" setting.
2. \*\*Check your Excel and Python setup\*\*: Make sure that your Excel file and Python module have different names. Also, ensure that you have added the default xlwings package from Anaconda to Excel Add-ins and added it to references in VBA.
3. \*\*Restart UDF Server\*\*: If you want to make sure everything is in a fresh state, click Restart UDF Server.

If none of these solutions work, please provide more details or consider reaching out to the xlwings community or support for more specific assistance. I hope this helps! 😊

User

In the following python function:

def aging\_buckets(x):
aging\_dict = {
range(-sys.maxsize, 0): "Advance",
range(0, 31): "Not Due",
range(31, 61): "31-60 Days",
range(61, 91): "61-90 Days",
range(91, 181): "91-180 Days",
range(181, 366): "6 Months to 1 year",
range(366, sys.maxsize): "More than 1 year"
}
result = [[next(value for r, value in aging\_dict.items() if day[0] in r)] for day in x]
return result

Instead of using dictionaries like:

range(-sys.maxsize, 0): "Advance",
range(0, 31): "Not Due",
range(31, 61): "31-60 Days",
range(61, 91): "61-90 Days",
range(91, 181): "91-180 Days",
range(181, 366): "6 Months to 1 year",
range(366, sys.maxsize): "More than 1 year"

Can you replace each of the above line with lambda function?

Bing

Lambda functions in Python are anonymous functions that are defined using the lambda keyword. They are used when a small, one-off function is needed. However, in your case, you're using a dictionary to map ranges to labels, which is not something that can be directly replaced with lambda functions.

That being said, you could use a list of tuples where each tuple contains a lambda function (representing the condition) and a label. Here's how you can modify your function to use this approach:

import sys

def aging\_buckets(x):

aging\_list = [

(lambda day: -sys.maxsize <= day < 0, "Advance"),

(lambda day: 0 <= day < 31, "Not Due"),

(lambda day: 31 <= day < 61, "31-60 Days"),

(lambda day: 61 <= day < 91, "61-90 Days"),

(lambda day: 91 <= day < 181, "91-180 Days"),

(lambda day: 181 <= day < 366, "6 Months to 1 year"),

(lambda day: 366 <= day, "More than 1 year")

]

result = [[next(label for condition, label in aging\_list if condition(day[0]))] for day in x]

return result

In this modified function, aging\_list is a list of tuples where each tuple contains a lambda function and a label. The lambda function checks if a day falls within a certain range. The next function is used to find the first label where the lambda function returns True for a given day. I hope this helps! 😊