
pyforest - feel the bliss of automated imports

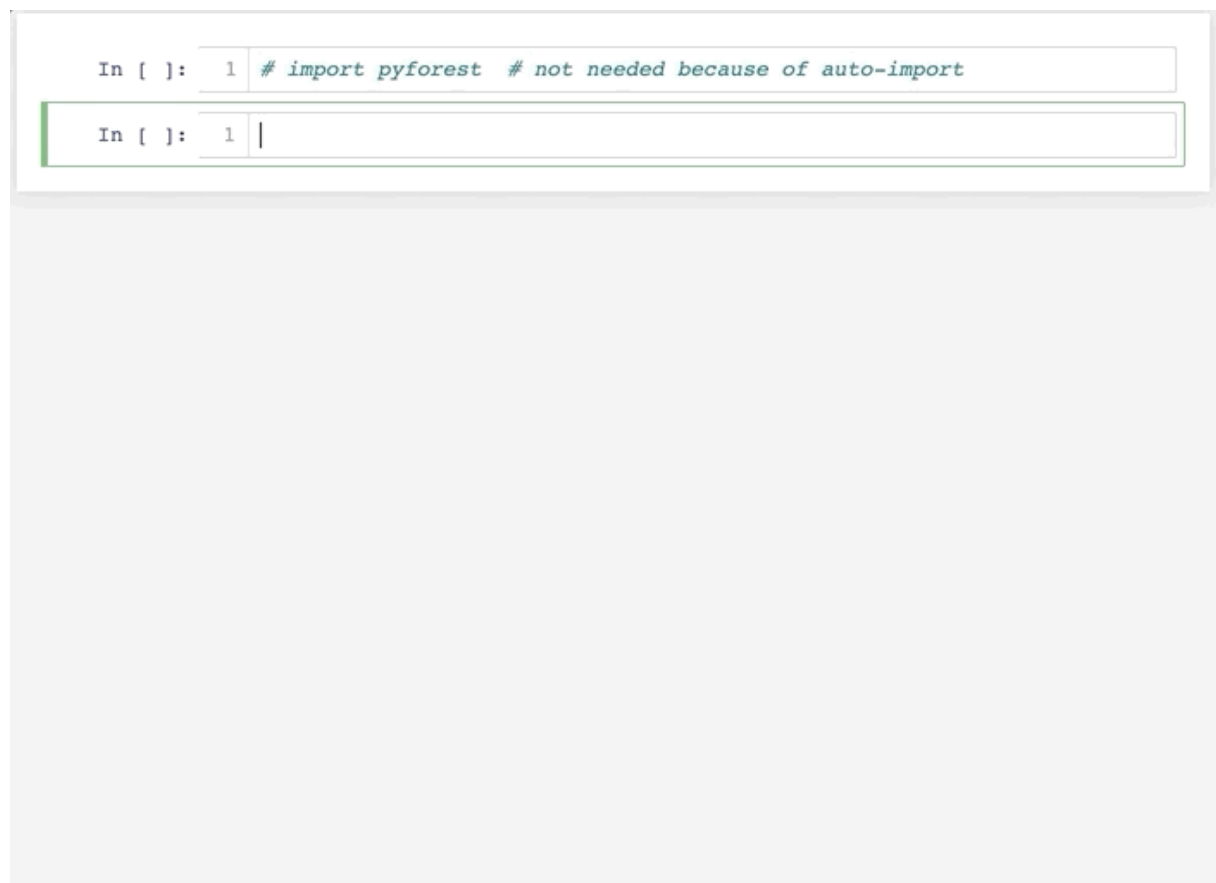
From the makers of bamboolib

Writing the same imports over and over again is below your capacity. Let pyforest do the job for you.

With pyforest you can use all your favorite Python libraries without importing them before. If you use a package that is not imported yet, pyforest imports the package for you and adds the code to the first Jupyter cell. If you don't use a library, it won't be imported.

- [Demo in Jupyter Notebook](#)
- [Scenario](#)
- [Using pyforest](#)
- [Installation](#)
- [FAQs](#)
- [Contributing](#)
- [About](#)

Demo in Jupyter Notebook



```
In [ ]: 1 # import pyforest # not needed because of auto-import

In [ ]: 1 |
```

Scenario

You are a Data Scientist who works with Python. Every day you start multiple new Jupyter notebooks because you want to explore some data or validate a hypothesis.

During your work, you use many different libraries like `pandas`, `matplotlib`, `seaborn`, `numpy` or `sklearn`. However, before you can start with the actual work, you always need to import your libraries.

There are several **problems** with this. Admittedly, they are small but they add up over time. - It is boring because the imports are mostly the same. This is below your capacity. - Missing imports disrupt the natural flow of your work. - Sometimes, you may even need to look up the exact import statements. For example, `import matplotlib.pyplot as plt` or `from sklearn.ensemble import GradientBoostingRegressor`

What if you could just focus on using the libraries?

pyforest offers the following **solution**: - You can use all your libraries like you usually do. If a library is not imported yet, pyforest will import it and add the import statement to the first Jupyter cell. - If a library is not used, it won't be imported. - Your notebooks stay reproducible and sharable without you wasting a thought on imports.

Using pyforest

After you installed pyforest and its Jupyter extension, you can **use your favorite Python Data Science commands like you normally would - just without writing imports**.

For example, if you want to read a CSV with pandas:

```
1 df = pd.read_csv("titanic.csv")
```

pyforest will automatically import pandas for you and add the import statement to the first cell:

```
1 import pandas as pd
```

Which libraries are available? - We aim to add all popular Python Data Science libraries which should account for >99% of your daily imports. For example, we already added `pandas` as `pd`, `numpy` as `np`, `seaborn` as `sns`, `matplotlib.pyplot` as `plt`, or `OneHotEncoder` from `sklearn` and many more. In addition, there are also helper modules like `os`, `re`, `tqdm`, or `Path` from `pathlib`. - You can see an overview of all currently available imports here - If you are missing an import, you can either **add the import to your user specific pyforest imports** as described in the FAQs or you can open a pull request for the official pyforest imports

In order to gather all the most important names, we need your help. Please open a pull request and add the imports that we are still missing.

Installation

You need Python 3.6 or above because we love f-strings.

From the terminal (or Anaconda prompt in Windows), enter:

```
1 pip install --upgrade pyforest
2 python -m pyforest install_extensions
```

Please make sure to restart any running Jupyter server so that the javascript extension can be loaded properly.

Also, please note that this will add pyforest to your IPython default startup settings. If you do not want this, you can disable the `auto_import` as described in the FAQs below.

Frequently Asked Questions

- **“How to add my own import statements without adding them to the package source code?”**
 - pyforest creates a file in your home directory at `~/ .pyforest/user_imports.py` in which you can type any **explicit** import statements you want (e.g. `import pandas as pd`). Your own custom imports take precedence over any other pyforest imports. **Please note:** implicit imports (e.g. `from pandas import *`) won't work.
- **“Doesn't this slow down my Jupyter or Python startup process?”**
 - No, because the libraries will only be imported when you actually use them. Until you use them, the variables like `pd` are only pyforest placeholders.
- **“Why can't I just use the typical IPython import?”**
 - If you were to add all the libraries that pyforest includes, your startup time might take more than 30s.
- **“I don't have and don't need tensorflow. What will happen when I use pyforest?”**
 - Tensorflow is included in pyforest but pyforest does not install any dependencies. You need to install your libraries separately from pyforest. Afterwards, you can access the libraries via pyforest if they are included in the pyforest imports.
- **“Will the pyforest variables interfere with my own local variables?”**
 - No, never. pyforest will never mask or overwrite any of your local variables. You can use your variables like you would without pyforest. The worst thing that can happen is that you overwrite a pyforest placeholder and thus cannot use the placeholder any more (duh).
- **“What about auto-completion on lazily imported modules?”**
 - It works :) As soon as you start the auto-completion, pyforest will import the module and return the available symbols to your auto-completer.
- **“How to (temporarily) deactivate the auto_import in IPython and Jupyter?”**
 - Go to the directory `~/ .ipython/profile_default/startup` and adjust or delete the `pyforest_autoimport.py` file. You will find further instructions in the file. If you don't use the auto_import, you will need to import pyforest at the beginning of your notebook via `import pyforest`
- **“How to (re)activate the pyforest auto_import?”**

-
- Execute the following Python command in Jupyter, IPython or Python: `from pyforest .auto_import import setup; setup()`. Please note that the `auto_import` only works for Jupyter and IPython.

- **“Can I use pyforest outside of the Jupyter Notebook or Lab?”**

- Technically, yes. However, this is not the intended use case. pyforest is aimed primarily for the use in a Jupyter Notebook or Lab. If you want to use pyforest in IPython or a Python script etc, please import it as follows `import pyforest`. Afterwards, you can get the currently active imports via `pyforest.active_imports()`

- **“Why is the project called pyforest?”**

- pyforest is created to be the home for all Data Science packages - including pandas. And in which ecosystems do pandas live? :)

Contributing

If you'd like to contribute, a great place to look is the issues marked with help-wanted.

In order to gather all the most important names, we need your help. Please open a pull request and add the imports that we are still missing to the pyforest imports. You can also find the guidelines in the pyforest imports file

About

pyforest is developed by 8080 Labs. Our goal is to make Python Data Scientists 10x faster. If you like the speedup to your workflow, you might also be interested in our other project bamboolib