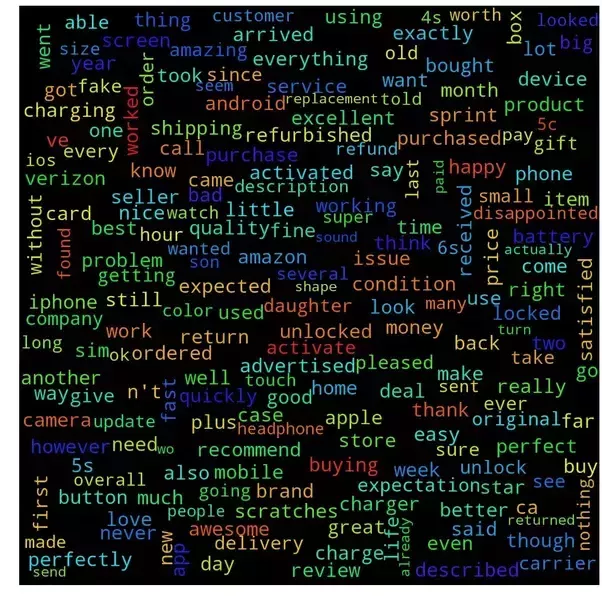
This is yet another somewhat recycled and modified answer. To do this, go slow in the beginning, and learn things systematically. When you have learned some, pick up more and get going. Set some milestones and reach them bit by bit. Be patient, and google whatever you cannot achieve easily. Most of the time all the solutions are available online. All the best learning Python!

* **I would recommend following pathway, which somewhat worked for me, although I keep jumping to and fro these days wherever I get stuck:**
* **First things first: Installation:**I suggest beginners to use ~ Installation from [Download Anaconda Now!](https://www.continuum.io/downloads" \t "https://www.quora.com/_blank). I started with Python 2.7, and I am happy with it. I have used Python 3.5 too, but then I kept coming back to 2.7. I think it is the most supported branch, and it works for all practical purposes.
* **Where to program:**To program, one can use Spyder, or Jupyter notebooks. Complex programming platforms exist such as those given in the below:
* **Where to program:**Sublime Text: [Sublime Text - Download](https://www.sublimetext.com/3" \t "https://www.quora.com/_blank),
* **Where to program:**Atom: [A hackable text editor for the 21st Century](https://atom.io/" \t "https://www.quora.com/_blank). This was somewhat harder because it doesn’t really allow installing themes and packages so easily. I actually stopped using it. It did have some amazing features though ~ For example ‘code complete’.
* **Where to program:**Pycharm: [PyCharm](https://www.jetbrains.com/pycharm/" \t "https://www.quora.com/_blank). As you grow in Python programming as profession, move on to using PyCharm. That is what most hardcore programmers use. But the licence for this can be costly.
* **Now the basic learning:**
* **Which website to go to: You should really begin here:**Go to [Python Programming Tutorials](http://pythonprogramming.net/" \t "https://www.quora.com/_blank) and learn the basics. The programmer is very nice and teaches everything in a very simple way. My idea about learning anything and everything is to get the basics out of the way, and this website will achieve the purpose.
* **Which website to go to:**The second source should be: Learn Python The Hard Way: [Learn Python the Hard Way](https://learnpythonthehardway.org/book/" \t "https://www.quora.com/_blank). I spend a decent amount of time on this, though I moved on to next thing whenever I thought I had gotten a reasonable solution.
* **Which website to go to:**Similarly there are other programmers: Such as [Python for Developers](http://ricardoduarte.github.io/python-for-developers/" \t "https://www.quora.com/_blank).
* **Which website to go to:**One really good source I found is ~ [Automate the Boring Stuff with Python](https://automatetheboringstuff.com/" \t "https://www.quora.com/_blank). You would find detailed answers to a lot of questions on this website/book, and you should be able to get some good value out of it.
* **Which website to not go to:**I would actually recommend that people stay away from Udemy, and Coursera, because they can confuse themselves initially. It is not that those courses are not good, but my opinion is that they aren’t made to be interesting and practical.
* **Which website to go to:**A few of data-scientists I know suggested Google’s Python class: [Google's Python Class | Python Education  
   | Google Developers](https://developers.google.com/edu/python/" \t "https://www.quora.com/_blank). I spent some time here as well, but I felt like I knew a few things already, so I quickly moved on.
* **How to go about loading up packages:**Packages can be loaded up from here: [vinta/awesome-python](https://github.com/vinta/awesome-python" \t "https://www.quora.com/_blank)
* **What packages to learn:**Quickly as you have learned the basics, dive into learning the packages, and begin with Pandas, NumPy, SciPy, Matplotlib, Seaborn, and Scikit Learn.
* **What packages to learn:**Learn pandas from here: [10 Minutes to pandas](http://pandas.pydata.org/pandas-docs/stable/10min.html" \t "https://www.quora.com/_blank), and detailed synopsis from [http://pandas.pydata.org/pandas-...](http://pandas.pydata.org/pandas-docs/stable/pandas.pdf" \t "https://www.quora.com/_blank), or [pandas: powerful Python data analysis toolkit](http://pandas.pydata.org/pandas-docs/stable/index.html" \t "https://www.quora.com/_blank)
* **What packages to learn:**Learn machine learning from here: [scikit-learn: machine learning in Python](http://scikit-learn.org/stable/" \t "https://www.quora.com/_blank)
* **How to learn from others:**Follow other people and how they write codes on Kaggle: [Your Home for Data Science](https://www.kaggle.com/" \t "https://www.quora.com/_blank)
* **For plotting, follow this website: [Seaborn: statistical data visualization](http://seaborn.pydata.org/" \t "https://www.quora.com/_blank)**
* **How to learn from others:**Get inspiration from how to come up with beautiful notebooks from here: [Notebook Gallery](http://nb.bianp.net/sort/views/" \t "https://www.quora.com/_blank)
* **Is there one resource for everything I need to know? You can probably make this your homepage or something:**[vinta/awesome-python](https://github.com/vinta/awesome-python" \t "https://www.quora.com/_blank). It gives a long curated list of Python packages that can keep someone busy for several months.
* How can I program for GUI development: WxGlade seems to be one of the best resources. [FrontPage - wxPyWiki](https://wiki.wxpython.org/FrontPage" \t "https://www.quora.com/_blank)
* Kivy [Kivy: Cross-platform Python Framework for NUI](https://kivy.org/" \l "home" \t "https://www.quora.com/_blank) also seems to be a very nice GUI development tool.

Further, enhance your learning by exploring more. Check out some of the best answers on what people are doing all over the world using Python. This is straight from Quora, and I occasionally visit this question just to see new additions. [What are the best Python scripts you've ever written?](https://www.quora.com/What-are-the-best-Python-scripts-youve-ever-written). A lot of people from all over the world have given their Python programs which you should be able to download and use easily.

**Here is what is amazing about Python in my opinion:**

* It is a very light, and an extremely logical language.
* The Anaconda suite has provided some amazing capabilities for various OSes.
* Of all the programming languages I have ever used, it has the cleanest and simplest syntax, as if it was written for the beauty of the syntax.
* It has an enormous library and packages maintained by a very large community. These include Pandas, NumPy, SciPy and so on. These libraries carry millions of lines of codes, which you can simply borrow and use. So you would be building on the top of some of the world’s best programmers.
* There are just too many Python enthusiasts all over the world, so no matter where you are, you would have no trouble in finding someone who knows more than you. Stack Overflow now overflows with Python support.
* Implementing machine learning algorithms is much simpler because many of them are available as packages and pretty much very minimal programming has to be done to get up and running.
* There are libraries available for web development. I usually don’t do web development, but some of my friends do, and they love it.
* One can process human readable language with packages such as NLTK. And you can be up and running in very little time. Think for example about this project ~ You have data given by Amazon on how much people like Apple phones and all that carries human text. You can create beautiful word clouds in very little time.



* The speed is really good. For all practical purposes, I always weigh the amount of time needed to code something, and the speed, and I always end up picking Python over C. In fact, in last ~ 2 to 3 years, I have pretty much moved everything to Python.
* *Now, I do admit that I did PhD in molecular simulations, and if I chose Python to do molecular simulations, probably that would not have worked.*
* *C/C++/Fortran are hardcore scientific computing languages, and Python cannot really come close to them when the things we want to solve are fluid simulations, molecular simulations, and aerodynamic simulations.*
* *Some of my friends who work in options trading world tell me that their language of use is C++. C++ is object oriented, and lightening quick when it comes to speed. Coding is certainly much heavier, but folks in those businesses have to do with speed over functionality, because they are all competing for money.*
* People working in industries like Intel, Exxon, BP, all kinds of FMCG industries, and industries like Nike, Adidas, and so on, can benefit the most from Python. I am more specifically talking about non-software folks, who have not exactly learned the computer science fundamentals, and cannot really spend 4–6 year of their lives learning programming languages and software development.
* Python will take 2–3 months only for you to make a significant improvement.
* I have friends who are biologists who were able to learn Python for their practical purposes and could derive a large value out of it in very little time spent.
* Check out what one can do with just one to two lines of code in Python: This answer of mine had gotten a ton of hits when I published it. I was actually shocked that it got so many upvotes. [Rohit Malshe's answer to What are some of the most elegant, greatest Python one liners?](https://www.quora.com/What-are-some-of-the-most-elegant-greatest-Python-one-liners/answer/Rohit-Malshe)

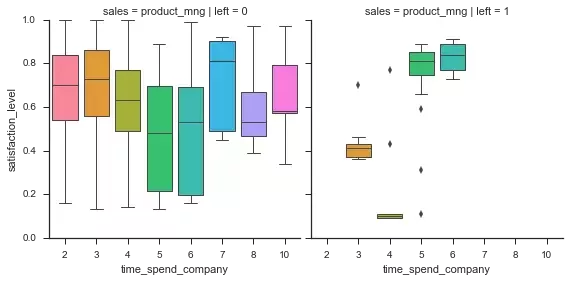
**What have I not liked about Python? (Some people’s opinions may differ)**

*Perhaps the only thing I want to point out is that if someone wanted to develop GUIs, their GUIs may not look as good as some of those developed in Visual Studio. I have personally liked Visual Studio GUIs better, but I may be wrong. Libraries like PyQT, Tkinter, etc. are available. I have just stayed away from them.*

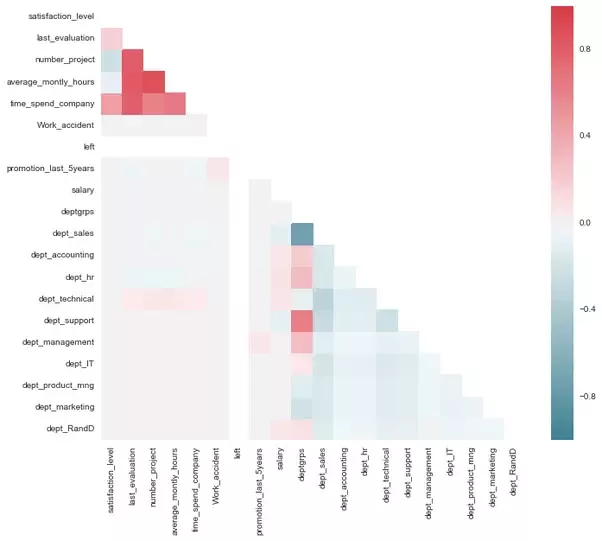
*Some Java programmers, or C# enthusiasts will never have any trouble whatsoever when they would want to program GUIs, but Python people will not shine here so easily.*

For anyone with just an year of experience in Python, one can make significant achievements.

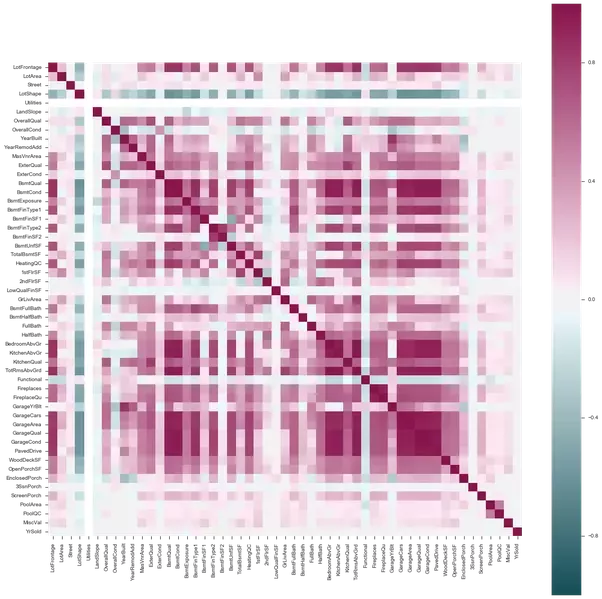
* **Data analysis:**
* One can create all sorts of data reports by getting data from various sources. Usually in industry setting, SQL becomes an important part, but Python lets one run SQL very easily.
* I highly recommend the readers of this answer to also learn SQL if they need it, and learn it systematically. Their data science skills will take a step jump. [Rohit Malshe's answer to Why is Python considered a good language for AI and Machine Learning?](https://www.quora.com/Why-is-Python-considered-a-good-language-for-AI-and-Machine-Learning/answer/Rohit-Malshe)
* One can then use iPython notebooks, that look very pretty for presenting lots of data, plots, and so on, and things can be automated to run on a daily basis.
* This is an enormous and vast field. One can do all sorts of statistical analysis, and compete with someone using JMP, R, SAS and so on.
* I answered some time back, if Python is better than R. And I have to tell you that I use Python although I know how to use R. [Rohit Malshe's answer to Is Python better than R?](https://www.quora.com/Is-Python-better-than-R/answer/Rohit-Malshe)
* Python programmers will usually have an edge over all of the above.
* Read my answer on data science world: [Rohit Malshe's answer to How can I become a data scientist?](https://www.quora.com/How-can-I-become-a-data-scientist-1/answer/Rohit-Malshe)
* Machine learning in itself is a vast field, but using Python’s pre-existing packages such as Sci-kit learn, one can become a machine learning engineer in very little time. At least you can easily begin working with datasets on Kaggle, and derive value out of it.
* Checkout out the type of plots one can easily make using the Seaborn library in Python.



Same way, making correlations plots is easy too:



Or this:



* **Automating boring stuff:**
* One can use PyAutoGUI module to automate a large number of screen tasks.
* This is extremely simple to use, and one can learn this in just 3 to 4 hours and can build substantial automation in their work.
* To learn this, Google ~ Automate boring stuff with Python.
* **Web development:**
* In addition to this, one can create and maintain websites ~ for which they would have to learn Django and other frameworks, but the basics remain the same.
* This is an elaborate field and would take some time to be good at it, but there is just no limit to what you cannot do.
* [25 of the Most Popular Python and Django Websites](https://www.shuup.com/en/blog/25-of-the-most-popular-python-and-django-websites/" \t "https://www.quora.com/_blank): You would be amazed to see the websites people have designed using Python.
* **Web scraping and API usage:**
* Imagine this project: Download S&P500, Nasdaq, volatility, interest rates, and so on ~ data from federal reserve bank, Yahoo or Google Finance, and derive value and conclusions out of it. Python will crack this in no time. There are APIs available for this and some already existing packages that you only have to import to start doing this.
* To develop a program that can plot S&P500, and every possible index from Federal reserve bank website, it took me merely 2 to 3 hours.
* **Constantly updating schedules:**
* By using Raspberry Pi, one can automate things like ~ “Constantly show me the timings of upcoming buses, or trains”. So if you have a screen, a Raspberry Pi, and some Python programming skills, you can build an automatic schedule display in no time, and at a very low price.
* A friend of mine who lives in Chicago travels to most places by some or the other public transport, and he has reverse engineered the code to tell him when to make a move from his home to catch the bus. Sure, this can be done easily by a cell phone app, but why not do it yourself for the fun and love of it.
* **Robotics:**
* One can control robotics using Raspberry Pi, and Python programming.
* This science is also quite elaborate, but once you know how to use a Raspberry Pi, it is not tremendously hard.
* **Game development:**
* Probably I am not the best person to comment on this, but a lot of programmers use Python for game development. Python is quick enough, and fast to code for such purposes.
* **GUI development:**
* I have to say, there may be better programming languages for this. Microsoft support for many other programming languages has been better than that for Python.
* My wish will come true, when Python and Visual Basic are combined one day to be fully integrated. Then one can enjoy the graphics of visual basic, and programming capabilities of Python.
* **AI type of solutions:**
* You can use Python’s libraries to talk to the computer, and get some enhancements from some of the other already existing software that does this task. The backend can then be built in Python.
* Check out this answer on how you can talk to your computer: [Rohit Malshe's answer to What should I do after reading "Building Machine Learning Systems With Python" if I want to be able to make a system that can speak like a person?](https://www.quora.com/What-should-I-do-after-reading-Building-Machine-Learning-Systems-With-Python-if-I-want-to-be-able-to-make-a-system-that-can-speak-like-a-person/answer/Rohit-Malshe)
* **Image processing:**
* My job requires image processing, and some scientific computing such as calculating Fourier transforms, and deriving conclusions out of them. This can be done by Python programming easily enough compared to some other programming languages. Probably some people who have used MATLAB would be delighted to know that Python can crack this problem easily enough.
* **Automating emails and reports:**
* One can setup automatic reports to send you information on one or the other thing. Imagine this ~ What are the prices of 100 stocks at the open ~ send mails to you on this on a daily basis ~ Python will achieve this very easily.
* Download reports from your organizations SQL server, process everything according to a particular sequence and logic and send you a report every morning ~ Python will achieve it a day or less.
* **Machine learning:**
* A few days back, I wrote about how can one go about learning machine learning. Some readers may find this helpful, hence I am somewhat interlinking my answers. [Rohit Malshe's answer to How do I learn machine learning?](https://www.quora.com/How-do-I-learn-machine-learning-1/answer/Rohit-Malshe)

Python is amazing. *Stay blessed and stay inspired!*