International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 09 Issue 11 ǁ 2021 ǁ PP. 36-53

A Review on Python Libraries and Ides for Data Science

AL. Sayeth Saabith[1], T. Vinothraj[2], MMM.Fareez[3]
[*1] *Centre for Information Communication Technology, Faculty of Science, Eastern University, Sri Lanka*
[2] *Centre for Information Communication Technology, Faculty of Science, Eastern University, Sri Lanka*
[3] *Finance Department Eastern University, Sri Lanka*

Abstract
A Python Integrated Development Environment (IDE) gives every one of the fundamental apparatuses required for software development with Python language. For researchers, an IDE is basically a software application that packs all the common developer tools into an exclusive user-friendly Graphical User interface. For the most part, it incorporates a source code editor to compose software programs and local build automation to develop a local build of the software-such as compiling computer source code. Amazing advantages of using one of the best python IDEs are The Integrated Development Environment helps you deal with a huge codebase and accomplish fast deployment, Developers and software engineers can utilize these editors to create PC or web-based applications, DevOps engineers can also employ IDEs to perform continuous integration, and it assists with task automation and improves the prolificacy and the overall proficiency of the software engineer. Python IDEs are power-packed with features such as build automation, code linting, testing, and debugging that can altogether accelerate your work. In this paper we first analyze you to Applications of Data Science, and Python Libraries Data Science. Moreover, this paper deeply analyzes the Python IDEs which are using in Data Science.

Keywords: *Data Science (DS), Machine Learning (ML), IDE, GUI, Python*

Date of Submission: 02-11-2021 Date of acceptance: 16-11-2021

I. INTRODUCTION

1.1 DATA SCIENCE:

Data science is a deep study of the massive amount of data, which involves extracting meaningful insights from raw, structured, and unstructured data that is processed using the scientific method, different technologies, and algorithms.

It is a multidisciplinary field that uses tools and techniques to manipulate the data so that you can find something new and meaningful. Data science uses the most powerful hardware, programming systems, and most efficient algorithms to solve the data related problems. It is the future of artificial intelligence.

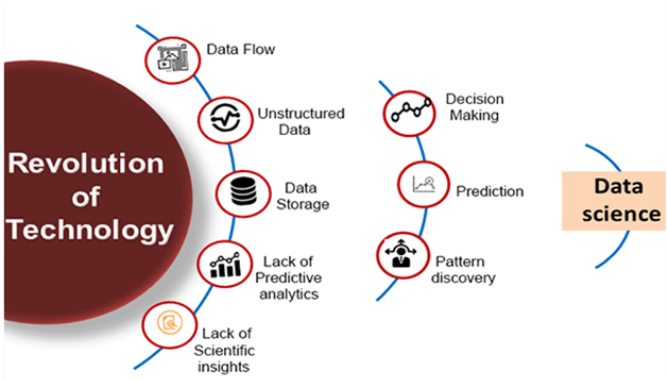

Figure1: Revolution of Data Science

1.1.1 Data Science Components:

The main components of Data Science are given below:

1. Statistics: Statistics is one of the most important components of data science. Statistics is a way to collect and analyze the numerical data in a large amount and finding meaningful insights from it.

2. Domain Expertise: In data science, domain expertise binds data science together. Domain expertise means specialized knowledge or skills of a particular area. In data science, there are various areas for which we need domain experts.
3. Data engineering: Data engineering is a part of data science, which involves acquiring, storing, retrieving, and transforming the data. Data engineering also includes metadata (data about data) to the data.
4. Visualization: Data visualization is meant by representing data in a visual context so that people can easily understand the significance of data. Data visualization makes it easy to access the huge amount of data in visuals.
5. Advanced computing: Heavy lifting of data science is advanced computing. Advanced computing involves designing, writing, debugging, and maintaining the source code of computer programs.

6. Mathematics: Mathematics is the critical part of data science. Mathematics involves the study of quantity, structure, space, and changes. For a data scientist, knowledge of good mathematics is essential.
7. Machine learning: Machine learning is backbone of data science. Machine learning is all about to provide training to a machine so that it can act as a human brain. In data science, we use various machine learning algorithms to solve the problems.

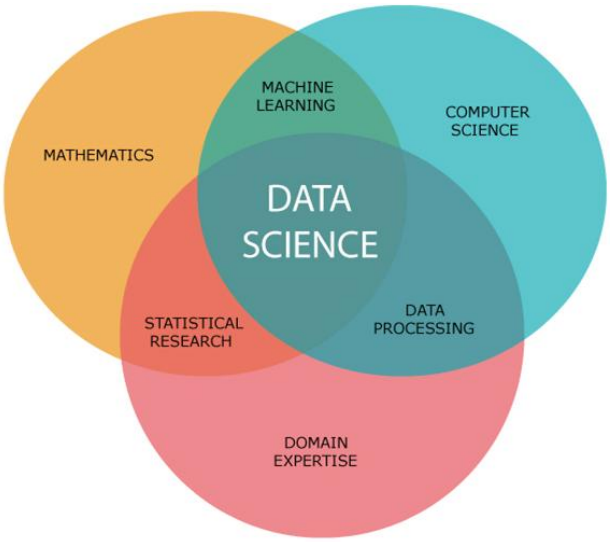

Figure2: Components of Data Science

1.1.2 Data Science Life Cycle

It is an iterative process that aims to produce insights and make predictions to achieve business goals. Various steps are involved in the Data Science life cycle such as business understanding, data preparation, data cleaning, visualization, modeling, and deployment. Let's go through these steps in detail
Business understanding: Before processing data, it is important to understand what the problem is or the objectives the business wants to achieve. For example, if a business wants to reduce credit loss, then it needs to find out the factors that affect it. For this, we need to understand our data by its structure, sources, relevance, and its type.
Data preparation: It is the most important step in the Data Science life cycle that involves data extraction, merging different data sources, cleaning, and dealing with missing values. Although it takes a lot of time to clean and transform the data, it is a crucial step to create a good model.
Exploratory data analysis: Before building the actual model, we have to gather information about the possible solutions and the affecting factors. We have to find the best possible solution that provides suitable results after processing the data.
Data modeling: The prepared data is fed to the data model, which provides the desired output. After selecting the model, we need to select the algorithm that provides the perfect results. To achieve the desired results, we can also use hyperparameters while maintaining a balance between generalization and performance.
Model evaluation: After the model is trained and modified based on the requirements, it is tested by unused datasets and evaluation metrics. If the desired results are not achieved, we must re-iterate the model until it gets it right.
Model deployment: Model deployment is the final step in the Data Science life cycle, where the model is deployed in the desired channel and format. After rigorous evaluation and modifications, the data model will become ready to provide the results in real time.

Now that we have understood what exactly is Data Science and looked at its sub-domains, let's go through some of its applications.

1.2 THE DATA SCIENCE APPLICATION

Data Science has created a strong foothold in several industries such as medicine, banking, manufacturing, transportation etc. It has immense applications and has variety of uses. Some of the following applications of Data Science are:

1. Banking

Banking is one of the biggest applications of Data Science. Big Data and Data Science have enabled banks to keep up with the competition.

With Data Science, banks can manage their resources efficiently, furthermore, banks can make smarter decisions through fraud detection, management of customer data, risk modeling, real-time predictive analytics, customer segmentation, etc.

Banks also assess the customer lifetime value that allows them to monitor the number of customers that they have. It provides them with several predictions that the business bank will derive through their customers.

In case of fraud detection, banks allow the companies to detect frauds that involve a credit card, insurance, and accounting. Banks are also able to analyze investment patterns and cycles of customers and suggest you several offers that suit you accordingly.

Furthermore, banks have the ability to risk modeling through data science through which they can assess their overall performance. With Data Science, banks are able to tailor personalized marketing that suits the needs of their clients.

In real-time and predictive analytics, banks use machine learning algorithms to improve their analytics strategy. Furthermore, banks use real-time analytics to understand underlying problems that impede their performance.

2. Finance

Data Science has played a key role in automating various financial tasks. Just like how banks have automated risk analytics, finance industries have also used data science for this task. Financial industries need to automate risk analytics in order to carry out strategic decisions for the company.

Using machine learning, they identify, monitor and prioritize the risks. These machine learning algorithms enhance cost efficiency and model sustainability through training on the massively available customer data.

Similarly, financial institutions use machine learning for predictive analytics. It allows the companies to predict customer lifetime value and their stock market moves.

Data Science also plays a key role in algorithmic trading. Through rigorous analysis of data, financial institutions are able to make data-driven decisions. It is also playing an important role in making the customer experiences better for the users.

Through extensive analysis of client experience and modification of preferences, financial institutions are able to create a personalized relationship with their customers.

This is further boosted by the real-time analytics of customers which increases the personalization. Through various customer sentiment analysis techniques and machine learning algorithms, we can boost the social media interaction, boost their feedback and analyze customer reviews.

Also, the additional machine learning techniques like natural language processing and data mining have contributed to the transformation of information for smarter governance that helps to increase the profitability of businesses.

3. Manufacturing

In the 21st century, Data Scientists are the new factory workers. That means that data scientists have acquired a key position in the manufacturing industries. Data Science is being extensively used in manufacturing industries for optimizing production, reducing costs and boosting the profits.

Furthermore, with the addition of technologies like the Internet of Things (IoT), data science has enabled the companies to predict potential problems, monitor systems and analyze the continuous stream of data.

Furthermore, with data science, industries can monitor their energy costs and can also optimize their production hours.

With a thorough analysis of customer reviews, data scientists can help the industries to make better decisions and improve the quality of their products. Another important aspect of data science in industries is Automation.

With the help of historical and real-time data, industries are able to develop autonomous systems that are helpful in boosting the production of manufacturing lines. It has taken away the redundant jobs and introduced powerful machines that use machine learning technologies like reinforcement learning.

4. Transport

Another important application of data science is transport. In the transportation sector, Data Science is actively making its mark in making safer driving environments for the drivers. It is also playing a key role in optimizing vehicle performance and adding greater autonomy to the drivers.

Furthermore, in the transport sector, Data Science has actively increased its manifold with the introduction of self-driving cars.

Through extensive analysis of fuel consumption patterns, driver behavior and active vehicle monitoring, data science has created a strong foothold in the transport industry. The self-driving cars the most trending topics in the world today.

With the introduction of autonomy to vehicles through reinforcement learning, vehicle manufacturers are able to create intelligent automobiles. Furthermore, industries can create better logistical routes with the help of data science.

Using a variety of variables like consumer profile, location, economic indicators, and logistics, vendors can optimize delivery routes and provide a proper allocation of resources.

Also, various transportation companies like Uber are using data science for price optimization and providing better experiences to their customers. Using powerful predictive tools, they accurately predict the price based on parameters like a weather pattern, availability of transport, customers, etc.

5. Healthcare

In the health-care industry, data science is making great leaps. The various industries in health-care making use of data science are

- **Medical Image Analysis**

In the medical image analysis, data science has created a strong sphere of influence for analyzing medical images such as X-rays, MRIs, CT-Scans, etc. Previously, doctors and medical examiners would have to manually search for clues in the medical images.

However, with the advancements in computing technologies and surge in data, it is possible to create machines that can automatically detect flaws in the imagery.

- **Genomic Data Science**

Genomic Data Science applies the statistical techniques to genomic sequences, allowing the bioinformaticians and geneticists to understand the defects in genetic structures. It is also helpful in classifying diseases that are genetic in nature. With data science, we can analyze how genes react to varying kinds of medicines. Also, several big data technologies like MapReduce have significantly reduced the processing time for genome sequencing.

- **Drug Discovery**

Another important field making use of data science is drug discovery. In drug discovery, new candidate medicines are formulated. Drug Discovery is a tedious and often complex process.

Data Science can help us to simplify this process and provide us with an early insight into the success rate of the newly discovered drug. With Machine Learning, we can also analyze several combinations of drugs and their effect on different gene structure to predict the outcome.

- **Predictive Modeling for Diagnosis**

With the advancements in predictive modeling, data scientists can help to predict the outcome of disease given the historical data of the patients. Data Science has enabled practitioners to analyze the data, make correlations between the variables of the data and also provide insights to doctors and medical practitioners.

- **Natural Language Processing**

Natural Language Processing is a technology of data science that is focused on the analysis of textual information. Using NLP, we can create intelligent bots that answer to user queries. The application of this can be extended to the healthcare sector where we can create bots that answer questions of patients and provide them with proper diagnostic guidelines. Curious to know more about Data Science? Have a look at Data Science Future

6. E-Commerce

E-commerce and retail industries have been hugely benefitted by data science. Some of the ways in which data science has transformed the e-commerce industries are-

For identifying a potential customer base, data science is being heavily utilized, Usage of predictive analytics for forecasting the goods and services, Data Science is also used for identifying styles of popular products and predicting their trends, and with data science, companies are optimizing their pricing structures for their consumers.

Data Science is also being heavily used in collaborative filtering, where it forms the backbone of advanced recommendation system. Using this technique, the e-commerce platforms are able to provide insights to the

customers based on their historical purchases and purchases made by people of the same style. This type of hybrid recommendation systems, consisting of both collaborative and content-based filtering are helping the industries to provide better services to their customers.
Also, companies are making use of sentiment analysis to analyze the feedbacks provided by the customers. This makes use of natural language processing to analyze texts and online surveys.
Fraud Detection, which is the central role of machine learning in industries is tailored for finding fraud merchants and frauds in wire-transfers.

1.3 What is an IDE?

IDE, or Integrated Development Environment, brings all the different aspects of writing code under a single umbrella – code editor, compiler/interpreter, and debugger. IDEs make it easier to start programming new applications quickly without setting up different utilities and learning other tools to run a program. The debugger tool inside IDEs is a boon that helps us examine variables and inspect code. IDEs helps to isolate the error that is bothering our otherwise brilliant code. Some IDEs also give us the capability to unit test our code to ensure it runs in every scenario. IDEs also have intelligent auto-code completion recommendations to anticipate what we will type next. Although this can make us lazy programmers, it inevitably saves us time while writing Python programs. So, with that backdrop, let's start exploring the various Python IDEs and unravel the capabilities of each of them!

II. PYTHON LIBRARIES FOR DATA SCIENCE

Python is the most widely used programming language today. When it comes to solving data science tasks and challenges, Python never ceases to surprise its users. Most data scientists are already leveraging the power of Python programming every day. Python is an easy-to-learn, easy-to-debug, widely used, object-oriented, opensource, high-performance language, and there are many more benefits to Python programming. Python has been built with extraordinary Python libraries that are used by programmers every day in solving problems.

TensorFlow: TensorFlow is a library for high-performance numerical computations with around 35,000 comments and a vibrant community of about 1,500 contributors. It's used across various scientific fields. TensorFlow is a framework for defining and running computations that involve tensors, which are partially defined computational objects that eventually produce a value. Key features of TensorFlow are Better computational graph visualizations, reduces error by 50 to 60 percent in neural machine learning, Parallel computing to execute complex models, Seamless library management backed by Google, and Quicker updates and frequent new releases to provide you with the latest features, TensorFlow is particularly useful for Speech and image recognition, Text-based applications, Time-series analysis, and Video detection.
NumPy: NumPy (Numerical Python) is the fundamental package for numerical computation in Python; it contains a powerful N-dimensional array object. It has around 18,000 comments on GitHub and an active community of 700 contributors. It's a general-purpose array-processing package that provides high-performance multidimensional objects called arrays and tools for working with them. NumPy also addresses the slowness problem partly by providing these multidimensional arrays as well as providing functions and operators that operate efficiently on these arrays. Key features of NumPy are Provides fast, precompiled functions for numerical routines, Array-oriented computing for better efficiency, Supports an object-oriented approach, and Compact and faster computations with vectorization. NumPy is particularly useful for Extensively used in data analysis, creates a powerful N-dimensional array, Forms the base of other libraries, such as SciPy and scikit-learn, and Replacement of MATLAB when used with SciPy and matplotlib
SciPy: SciPy (Scientific Python) is another free and open-source Python library extensively used in data science for high-level computations. SciPy has around 19,000 comments on GitHub and an active community of about 600 contributors. It's widely used for scientific and technical computations because it extends NumPy and provides many user-friendly and efficient routines for scientific calculations. Key features of SciPy are Collection of algorithms and functions built on the NumPy extension of Python, High-level commands for data manipulation and visualization, Multidimensional image processing with the SciPy.ndimage submodule, and includes built-in functions for solving differential equations. SciPy is particularly useful for Multidimensional image operations, Solving differential equations and the Fourier transform, Optimization algorithms, and Linear algebra
Pandas: Pandas (Python data analysis) is a must in the data science life cycle. It is the most popular and widely used Python library for data science, along with NumPy in matplotlib. With around 17,00 comments on GitHub and an active community of 1,200 contributors, it is heavily used for data analysis and cleaning. Pandas provide fast, flexible data structures, such as data frame CDs, which are designed to work with structured data very quickly and intuitively. Key features of Pandas are Eloquent syntax and rich functionalities that gives you the

freedom to deal with missing data, enables you to create your function and run it across a series of data, High-level abstraction, and contains high-level data structures and manipulation tools. SciPy is particularly useful for general data wrangling and cleaning, ETL (extract, transform, load) jobs for data transformation and data storage, as it has excellent support for loading CSV files into its data frame format, used in a variety of academic and commercial areas, including statistics, finance, and neuroscience, Time-series-specific functionality, such as date range generation, moving window, linear regression, and date shifting.

Matplotlib: Matplotlib has powerful yet beautiful visualizations. It's a plotting library for Python with around 26,000 comments on GitHub and a very vibrant community of about 700 contributors. Because of the graphs and plots that it produces, it's extensively used for data visualization. It also provides an object-oriented API, which can be used to embed those plots into applications. Key features of Matplotlib are Usable as a MATLAB replacement, with the advantage of being free and open-source, supports dozens of backends and output types, which means you can use it regardless of which operating system you're using or which output format you wish to use, Pandas itself can be used as wrappers around MATLAB API to drive MATLAB like a cleaner, and Low memory consumption and better runtime behavior. Matplotlib is particularly useful for Correlation analysis of variables, visualize 95 percent confidence intervals of the models, Outlier detection using a scatter plot, and visualize the distribution of data to gain instant insights.

III. PYTHON IDE FOR DATA SCIENCE

Python is the most versatile language in the programming world, and it applicable in almost every domain of software development and as well as Data Science operations. It helps us in taking care of our current programming task as well as lets us focus on the core functionality of Python programming languages. Python makes its presence in every emerging field. It is the fastest-growing programming language and can develop any application. [1,3, 4, 5]. Python provides various libraries like matplotlib, seaborn, TensorFlow, scikit-learn and other important tools required for data science processing. Furthermore, it provides other tools like Flask, support for SQLite and other functionalities that can lead to a comprehensive data product. Python IDEs for data science that make data analysis and machine learning easier! Let's also see which are the most popular Python IDE and text editors. It's important to note that the below statistics are from December 2018 and should give a good understanding of what is popular today.

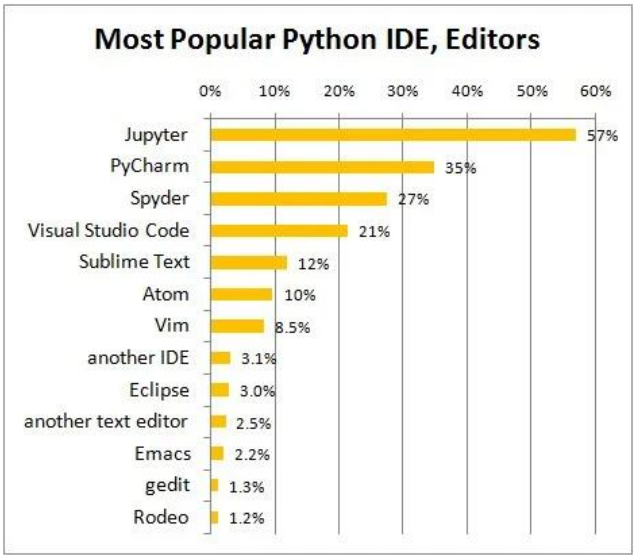

Figure3: Popular Python IDEs

With a running course of almost 3 decades, Python has garnered enormous popularity among the programming community. Using IDLE or the Python Shell for writing down Python code is effectual for smaller projects, but not practical while working on full-fledged machine learning or data science projects. In such a case, you need to use an IDE (Integrated Development Environment) or a dedicated code editor. As Python is one of the leading programming languages, there is a multitude of IDEs available. So, the question is, "Which is the best IDE for Python?"

Seemingly, there is no single IDE or code editor for Python that can be crowned with "THE BEST" label. This is because each of them has their own strengths and weaknesses. Furthermore, choosing among the vast number of IDEs might be time-consuming. Worry not though, as we've reviewed. In order to help you pick

the right one, we've analyzed some of the prominent IDEs for Python, specifically created for working with data science projects.

3.1 Jupyter Lab:

Jupyter was introduced in 2014 and is a successor to iPython. It is a web application based on a server-client structure which is free, open-source, and easy to use. Its name is a reference to three core programming languages supported by Jupyter – Julia, Python, and R. But Jupyter supports over 40 programming languages!

Most data scientists have worked with Jupyter notebooks at some point or another in their lives because of the functionalities and ease of use it offers. But the classic Jupyter notebooks are getting a make-over with the next generation JupyterLab launched in 2018. It is a web-based IDE for data science that serves as a great starting place for data science beginners.

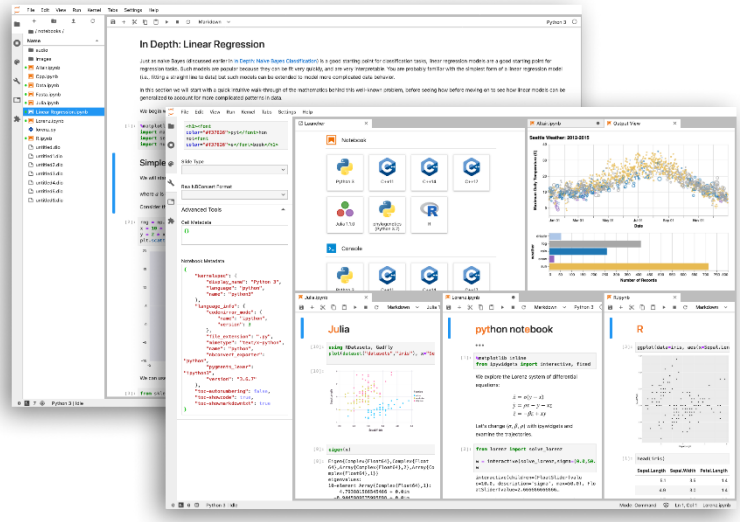

Figure4: JupyterLab Interface

Jupyter provides an interactive output which means you can write your code and test it there and then. This becomes extremely useful when you are a beginner and just starting out your journey in machine learning and data science. In addition to writing your Python code, you can create great notebooks that contain visualizations and text explanations using the markdown editor which even enables you to write even Latex equations!

It brings the terminal, text editor, console, and file directory view all under the same roof in a single work area with a flexible layout. Using the various magic commands and notebook extensions, you can really augment the functionalities of Jupyter. You can add features like auto-formatting, debugging, autosave, auto code completion, and much, much more. There is even a zen mode extension to minimize distractions and maximize productivity! The notebooks that you create with JupyterLab can be downloaded in a variety of formats ranging from pdf to .py files and even as slides for presentations! JupyterLab comes bundled with the Anaconda distribution. This makes it quite easy to install JupyterLab and other IDEs t. It is available for Windows, Linux, and macOS s platforms.

Pros	Cons
• Visually intuitive organization of code. • Static (but changeable) display of function outputs. • Easy replication of notebooks or into new notebooks, or into PDFs. • Code presentation- with Jupyter Notebook you can deploy codes and markdowns which makes the code easy to read and understand. • User interface- the user interface of the Jupyter Notebook is exceptionally smooth, there are a lot of easy shortcuts as well the icons to make our work easier • Server hosting- with Jupyter Notebook	• no code style correction- it does not have any functionality to auto-correct the code style such as spaces. • Most of the time jupyter notebook kernel do not have consistency, it unexpectedly stops, and it need to restart it several times. • no third-party app integration • No IDE integration/linting • No testing integrations • Difficult to view changes in GitHub • Notebook harder to productionize than scripts

server hosting is extremely easy which adds to the security feature • Exploratory data analysis/viewing code in-line • Data exploration/visualization • Switch between different coding languages • Simple and elegant code writing ability. Easier to understand the code that way. • The ability to see the output after each step. • The ability to use ton of library functions in Python. • Jupyter has an easy navigation platform compared to others. • Jupyter makes python programming because of some compelling features like viewing details of bash executions. • We can use it as a notebook and share the slide and publish it online through GitHub. • Attractive programming environment. • Developing code snippets for big or small projects • Highly recommended data analysis presentation	• Should work on Configuration setup, it takes a lot of time. • Work on code styling correction, sometimes it makes a major difference. • There is no IDE integration. • Should include more programming language. • Need more Hotkeys for creating a beautiful notebook. Sometimes we need to download other plugins which messes its default settings. • Once the data exploration, experimentation and prototyping are done, you usually need to move the analytics 'to production,' which is software development and should result in scalable, maintainable, robust code. You simply do not want to do this with a notebook, but develop a properly structured set of source files with an IDE like Spyder or PyCharm that supports things like version control, running automated tests etc.

3.2 PyCharm

PyCharm, like the name suggests, is a charming Python IDE created by JetBrains, the company behind the popular IntelliJ IDEA IDE for Java. It is a great IDE to try out if you are looking to work on a project containing multiple scripts interacting with each other. PyCharm is suited for any developer who wishes to create software applications in Python, be it web applications, data science applications, or even just a simple Python script. PyCharm lets you get your work done quickly and efficiently!

PyCharm has two versions – a free Community version and a paid Professional version that is available for a free 30-day trial, giving you the opportunity to try out whether you want it as your new Python IDE. Both of these can be downloaded from this page for either Windows, Linux, or macOS. To compare the two versions, you can have a look at the image below which will give you an overview of the features that are missing in the free version:

Luckily, if you are a student or are teaching at an education facility, you can apply to get access to all JetBrains IDEs for free. All you have to do is apply for it on this webpage and you will get free access to all of JetBrains's IDEs. PyCharm clearly has a lot to offer and will surely be able to tackle all the development work related to Python, from web development to data science applications. PyCharm can be a resource-intensive IDE requiring plenty of memory and storage space.

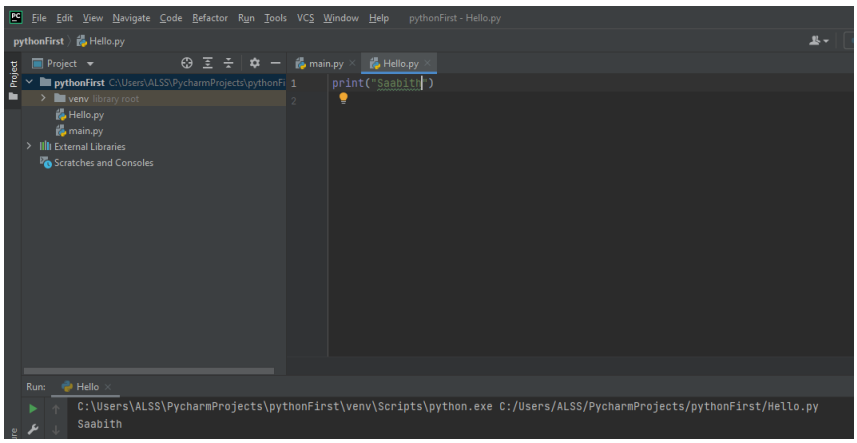

Figure5: PyCharm Interface

Let's list some of PyCharm's key features that make it such a popular IDE among developers:
•PyCharm's code editor is second to none. It has syntax and error highlighting, code analysis, and quick fixes for instantly improving the code. Additional features like auto-code generation, auto-indentation, code folder, etc. also make it incredibly comfortable to code in PyCharm
•When starting a new project in PyCharm, you can choose from different environments like Virtualenv, Pipenv, or Conda, which help keep dependencies required by different projects separate by creating isolated Python environments for them

- PyCharm provides easy navigation capability. You can search for anything and PyCharm will find it for you. It also allows you to locate any usage for your symbol in the entire project. These features are incredibly useful if you are working on a big project, especially a web development project where there are multiple scripts within the same project
- PyCharm's bookmarks and TODO capabilities allow you to leave remarks in your code that serve as a reminder to make the necessary amends the next time you navigate through them
- It provides you with refactoring capability to safely restructure your code. This includes renaming, extracting method, in lining local variable, changing method signature, and much more
- PyCharm has a powerful debugger with a graphical interface which makes debugging an easy task
- It has integrated unit testing and you can observe the results in a graphical manner. By default, PyCharm uses unit test as the test runner but it supports other frameworks too
- PyCharm has an integrated Version Control System to keep track of changes made to files and applications. It provides a unified user interface for CVS, Git, Mercurial, Perforce, and Subversion
- You can use plugins to add extra features to PyCharm like adding a new theme in addition to the default dark and light themes already present.

Pros	Cons
• Data science scripting. • Frontend development based on Python. • Very intuitive and Simple IDE for developers. • So many plugins are available and are free to use which improves the productivity. • Inbuilt virtual environment creation makes dependency management very easy. • Terminal is inbuilt and again improves the productivity. • Source control management is baked in to easily resolve any code conflicts. • It's really intuitive and easy to use • It's easy to switch between versions of Python, which is something pretty useful in this language • The syntax highlights help us when writing code, making our time more efficient • Even though it is meant for Python, it works really well with other languages, which makes it a very versatile tool. • Autocompletion and auto-suggestions • Code comparison, Spelling Correction, Language Injection • Connectivity with various databases, to perform queries in the IDE itself. • Integration with major version controllers. Performing all commands in the IDE itself. • Quickly and conveniently install, update and remove plugins in the IDE itself using the repositories. • Real-time coding verification to warn of misuse of methods and/or functions. • Process of searching for files by name or by source code snippets. • Access to files that refer to their encoding as a native file that contains one class declaration being used in another by clicking on the class name, for example. • Default indentation picks up, which helps developers a lot as most of them (who are new to Python) makes an error while creating a method,	• Community edition is, while quite usable, far behind the paid alternative. At least a basic database support would be nice. • Sometime addons don't work as expected. • It takes up lot of memory when 2 or more projects are open at the same time. • Setting up proxies is not straight forward. • Most of the features in paid version are now available in VS code for free. • It can be a little slow to startup • It's expensive for amateur work, so if you are a freelancer DevOp this probably won't be affordable • Improvements in debugging console • More default themes • Fresh look and feel • The process of upgrading IDE versions can be improved for the Linux environment as we need to download a new version and use it. This does not happen when it works on Windows. • Failure to learn to learn all the features of the IDE, making the most of all its functions. At least I didn't find this description. • The value of the full version is very expensive, thinking about the location I am in. • These new versions use a lot of machine feature. Java consuming a lot of memory. • Copy-paste does not work correctly. The default printing is in the form of Insert. I always need to press the enter button to come out of the insert mode and also have to right-click and select copy or paste instead of default CNTRL c and CNTRL V. It is working good for some others in the team, and I am not able to fix this. • Too much of underlining the code considering negligible errors make the code look bad. • Auto suggestions sometimes does not work as expected. • Abstracting commands into UI dialogs is nice. However, due to this the language changes, making it non-obvious to use sometimes. Tooltips for text boxes can eliminate the need for additional googling • Memory intensive - only a beefy machine can host this IDE in its full glory. Modular and plugin-based approach like VS Code can only load those features into

using a loop or such thing. • Switch between one project to the other is done very easily using PyCharm when compared with other tools. • Syntax highlighting and printing different colors for a method, variable, code, and comment also helps in concluding what is what. • One can create a java file, can connect to DB, can connect to different servers with Unix, and can also create an automation framework using a robotic framework, which in turn makes it an RPA tool. • Light and dark theme also helps a lot while working at late nights and during the daytime. • Git integration is really essential as it allows anyone to visually see the local and remote changes, compare revisions without the need for complex commands. • Complex debugging tools are basked into the IDE. Controls like break on exception are sometimes very helpful to identify errors quickly. • Multiple runtimes - Python, Flask, Django, Docker are native them to IDE. This makes development and debugging and even more seamless. • Integrates with Jupyter and Markdown files as well. Side by side rendering and editing makes it simple to develop such files. • Customizable interface: layout, color scheme, hot keys, etc. may all be individually tailored to a user. • Platform intelligence: debugging, code analysis, dependency resolution, and auto completion make the PyCharm IDE efficient and a pleasure to use • Support / tutorials guide the user through learning the different capabilities (this was a big deal for me when I switched to using Python / PyCharm after coding in a text editor and R Studio) • Syntax Highlighting: helps us read-write better code and understand what is going on with a simple glimpse • Version Control: being able to see a history of the file(s) I am working on lets me know straight away who I should contact in case I need it • Virtual Environments: don't need to get back to the console, I can do whatever I need within my IDE • Cross Product compatibility: being based on IntelliJ IDEA it's very easy to move from PyCharm to IntelliJ or WebStorm	memory as per need • The biggest complaint I have about PyCharm is that it can use a lot of RAMS which slows down the computer / IDE. I use the paid version, and have otherwise found nothing to complain about the interface, utility, and capabilities. • Startup speed: could be faster • Runtime environment: needs to change the runtime every time I want to switch from code to test cases • Updates: on Ubuntu 14.04, updating is a little bit troublesome: must be done manually, cannot be updated automatically •

3.3 Spyder

Spyder is a scientific environment for Python, built for scientists, engineers and data analysts. It combines advanced editing, analysis, debugging, and profiling functionality of a comprehensive development tool with data exploration, interactive execution, deep inspection, and visualization capabilities similar to a scientific package. Spyder is sponsored by open-source supporters QuanSight, and NumFOCUS, as well as individual donors. This is a lightweight, free, and open-source Python IDE. It is completely written in Python and designed exclusively for data scientists and analysts. Its interface is very basic when you compare it to other IDEs, but it has all the necessary components we look for in a coding environment. It consists of a text editor, file explorer, variable explorer, and IPython console all in a single window. It has built-in integration with many popular scientific packages including NumPy, SciPy, Pandas, IPython, and others.

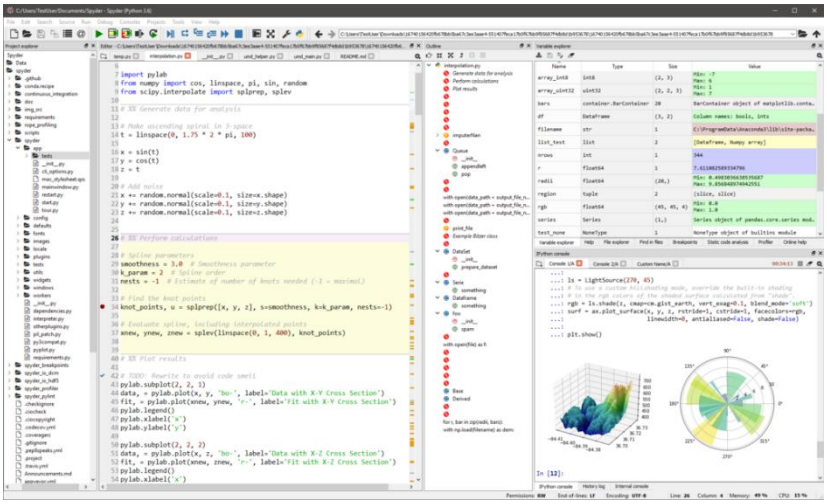

Figure6: Spyder Interface

Here are some of the key features Spyder offers:

- A pretty decent code editor with features like syntax highlighting, code completion, and real-time code analysis, which highlights the potential problem or a syntax error in your code
- An interactive code execution that allows you to execute your code by line, by cell block, or run the entire file in a single go, leaving the choice entirely up to you!
- An IPython console if you just want to test out a few lines of code without wanting to disrupt your primary session
- A Variable editor showing the variables, functions, modules, etc. of the currently selected IPython Console session. It also provides several built-in supports for editing objects like lists, strings, tuples, etc. along with a really awesome feature to display some of them as an image or even a plot!
- The Static code analysis feature detects style issues, bad practices, potential bugs, and other quality problems in your code, without even having to actually execute it. This is done using the very popular PyLint analyzer
- A Debugger for times when you are stuck scratching your head over an error you cannot solve. It allows breakpoints and the execution flow to be viewed and controlled right from the Spyder GUI
- A Profiler to determine the statements in your code that need optimization to improve the performance of your code, (because no one is a born programmer)
- A basic Git version control system to commit or browse files, directory, or the entire repository
- A History log pane that records all commands introduced in the editor and IPython console
- A Help pane gives a detailed description of any object. It offers documentation about modules, classes, functions, and methods. This can be accessed from the editor as well as from the IPython console

Pros	Cons
• Well formatted comments in the code. • It's Free and Open source to use any library in python • Spyder is best suited for Python only with data analysis and reporting generation operations. • firstly, it's free of cost • Attractive interface and UI • Similar to RStudio and other IDE's • Debugging of your existing code • Generates figures very quickly as part of a figures tab which lets users understand results quickly • Different layouts are available for the software which will give the users freedom to decide what layout works best for them	• Well formatted comments in the code. • It's Free and Open source to use any library in python • Spyder is best suited for Python only with data analysis and reporting generation operations. • Suitable for Python only. • Difficult to setup its env. • Not suitable for collaborative work. • The results tab needs to improved. • The software requires a bit of a learning curve. Tutorials about how the software can be used should be added.

3.4 PyDev

The PyDev IDE is a Python IDE for Eclipse. It was developed in 2003 and was made open source in 2009. It can be used in Python, Jython, and IronPython development. There are many more features that PyDev has to offer which you will come across if you choose to work with this IDE. If you already have experience working with Java in Eclipse, you will find a lot of familiarity in using the PyDev IDE for Python development purposes. There are a couple of ways to install PyDev on your local machine. If you are already working with Java in Eclipse, then you would find it easier to install it as a plugin. But the recommended way of using PyDev is by installing LiClipse which bundles PyDev along with a lightweight editor. You can download it for Windows, Linux, or macOS.

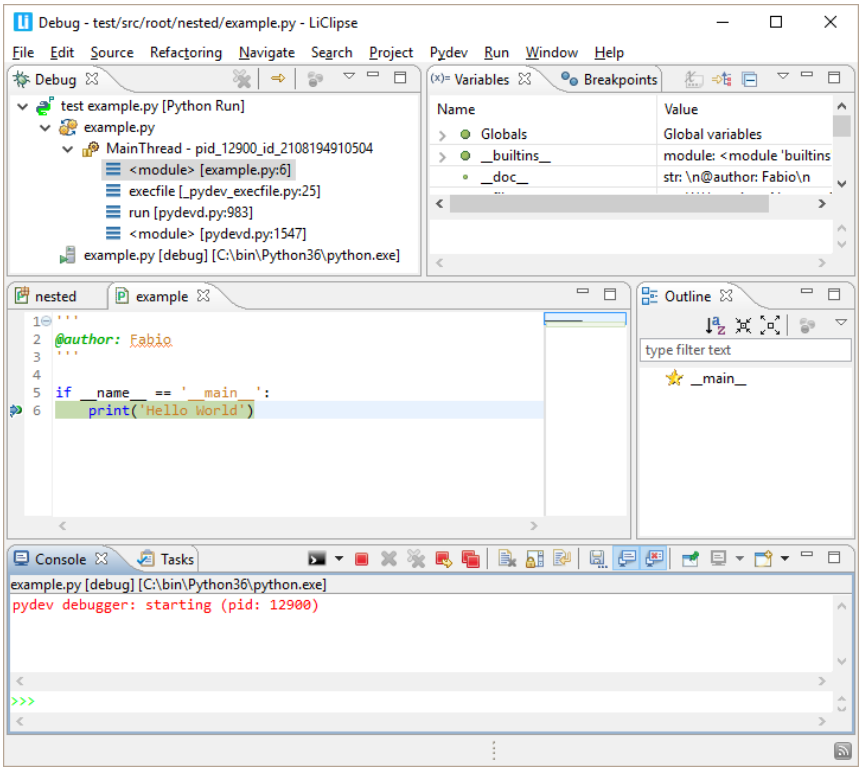

Figure7: PyDev Interface

Here are some of favorite features from PyDev:

- Provides code completion capability to complete tokens and automatically import its module
- On-the-fly code analysis that finds common errors like undefined variables, unused variables and imports, duplicate signatures, bad indentation, and much more
- Refactoring capability like renaming variables, methods, classes, and attributes, extracting methods and variables, and also in lining variables
- The debugger offers rich capabilities like conditional breakpoints, expression evaluation and the ability to view the variables in the current stack. It even has the remote debugging capability to debug external programs
- No IDE is complete without a Python console and PyDev is no different. It has an interactive console for Python, Jython, and IronPython depending on the interpreter being used
- PyDev provides unit testing capability through the unittest, nosetest or pytest formats available
- Basic syntax highlighting and code folding for better access to the code area
- Django comes pre-installed with PyDev so you will have a smooth experience developing Django based web-applications.

Pros	Cons
• integration with Iron Python, CPython, Django and Jython	multiple plugins tend to slow down performance significantly.
• efficient syntax highlighting and multi-language editor	

• an interactive console, smart indent, content assistants and tabs preferences.	

3.5 Visual Studio

Visual Studio or VS is another great IDE for Python developed by Microsoft, but it is only available for Windows and macOS users. It has a free Community version and paid Professional and Enterprise versions. It is lightweight and comes with its own marketplace for extensions. VS provides support for building Python web applications using Django and Flask, and Data Science applications with built-in Conda and IPython support. VS allows you to work with a plethora of other tools, like SQL, Unity, .NET, Node.js, and much more. So, it should come as no surprise that Visual Studio will be great for anyone who wishes to create great applications for devices, the cloud, or anything in between.

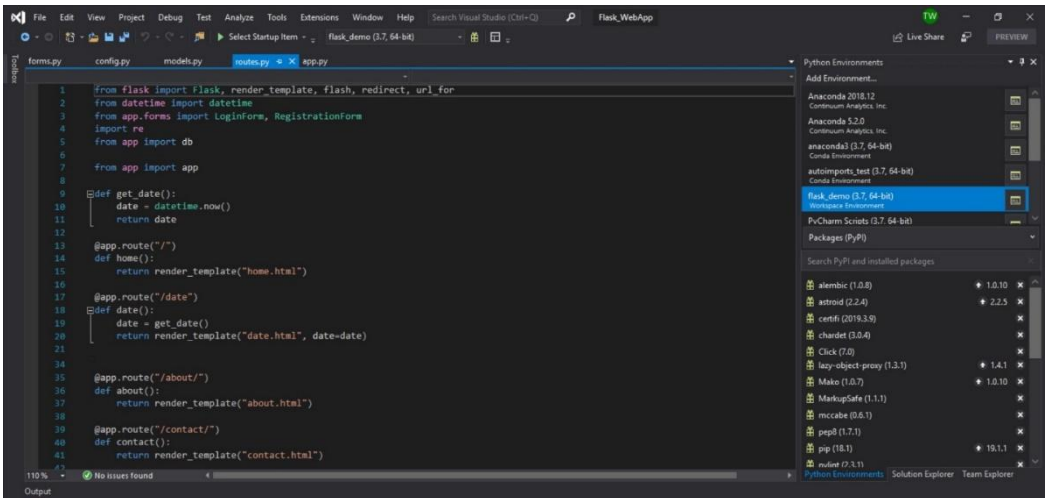

Figure8: VS Interface

Visual Studio for Python has its own features that set it apart from other IDEs:

- Visual Studio's code editor is guided by the IntelliSense syntax which provides auto-completion capability, type hints for functions and classes, signature help, quick info tooltip, and code coloring
- It has code snippets to insert code fragments into your file through shortcuts.
- VS has a host of pre-configured formatting options in addition to the default PEP-8 style formatting.
- The refactor in VS is also pretty neat, providing you with options like renaming, extracting methods, adding imports, and removing unused imports.
- Microsoft has integrated PyLint into Visual Studio that checks for errors in Python code and encourages good Python coding patterns. This will definitely improve your coding standard.
- Visual Studio provides an interactive read-evaluate-print loop (REPL) window which lets you enter arbitrary Python code and see immediate results. This will be useful when experimenting with a new API or library.
- In addition to adding breakpoints in code, the debugger allows you to inspect and modify variable values or insert arbitrary Python expressions and view its result. You can also use the Python Debug Interactive window, which is richer as it provides an interactive REPL experience for debugging.
- It also has the capability to allow unit tests via the unit test or the pytest frameworks.
- It is a great IDE if you want to build web applications using Flask or Django. Downloading these libraries and other dependencies is super easy with Visual Studio's virtual environment support without having to write a single code line.
- Visual Studio provides integration with local Git repositories and remote repositories on GitHub and Azure Repos. You can clone a repo, commit changes, and manage branches with these integrations.

Pros	Cons
• smart code completion with the IntelliSense feature • Git integration and multi-language support • highly customizable code editor.	• search feature is limited; • Lowish performance as reported by many users.

3.6 Thonny

Thonny is an excellent Python IDE that will run on Windows, Linux, and Mac. The debugger of Thonny helps in debugging codes line by line, this process helps a lot for beginners who are learning to code. The excellent GUI of Thonny makes the installation of third-party packages much easier. Thonny autocompletes code according to its prediction and inspects the code for bracket mismatching and highlights the error which is a great feature for beginners. It is completely free to download. When you call a function in Thonny, it will be done in a separate window which makes the user understand the local variables & call stack of the function better. The package manager of Thonny helps you in downloading them and increasing the functionality of python.

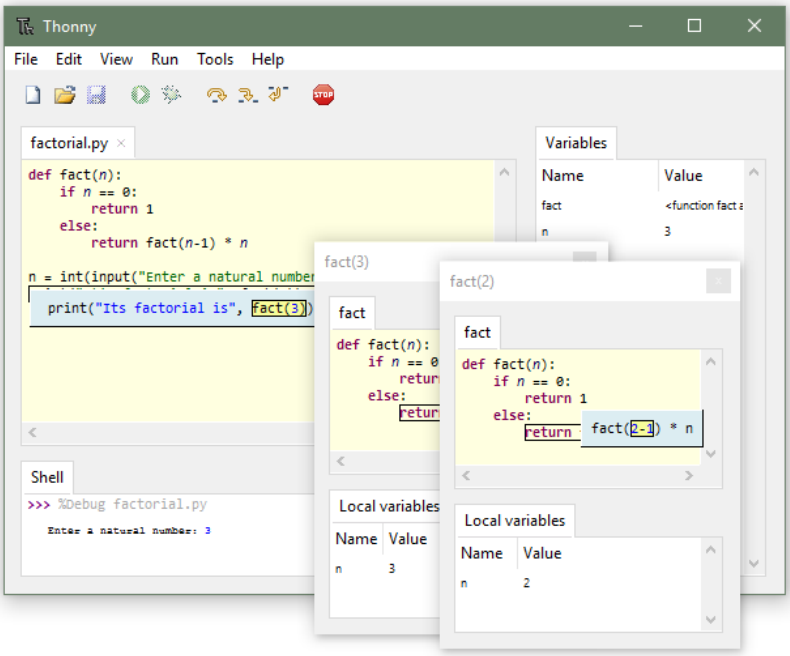

Figure9: Thonny Interface

The following are some of the primary features of Thonny:

- It permits the software developers to check the effect of code and shell commands on python variables and also includes a simple debugger. Not to forget how the code editor highlights the syntax errors.
- It also includes backing for CPython and MicroPython and shows steps through Expression Evaluation.
- A portion of the extra significant provisions of Thonny is – statement stepping without breakpoints, live variables during the process of debugging, ability to access a new window with a separate local variables table and code pointer, availability of separate windows for performing function calls, along with a simple and clean pip GUI.
- Compatible with all major platforms (Windows, macOS, Linux) and comes with the mode for explaining references.

Pros	Cons
• It has an easy-to-use user interface. • The user interface does not contain any distractions for beginners. • a simple debugger • Thonny goes an extra mile to make the interpreter function simpler and error-free • all must-have features for Python development including syntax highlighting, function calls, a robust editor and more	• the functionality is quite basic. For advanced features and visual editor interface, developers must resort to other advanced IDEs (i.e., PyCharm) • Users may encounter some issues for which a quick fix is not available.

3.7 Atom

Atom is an excellent IDE for ML & DS professionals which supports many other languages besides python like C, C++, HTML, JavaScript, etc. You can use it on Windows, Linux, and Mac. Atom supports MySQL, PostgreSQL, Microsoft SQL Server which helps you in writing and executing SQL queries/commands.

There are many useful packages in Atom like the atom-beautify package which beautifies your code and makes it more accurate. The outline view feature of Atom lets you see a tree-based view of your code and you can cross-check your classes, functions, etc. easily. Atom will provide you many themes and templates from GitHub to choose from. ML & DS professionals also prefer Atom because of its ability for cross-platform editing. It is one of the best open-source free IDEs to use currently.

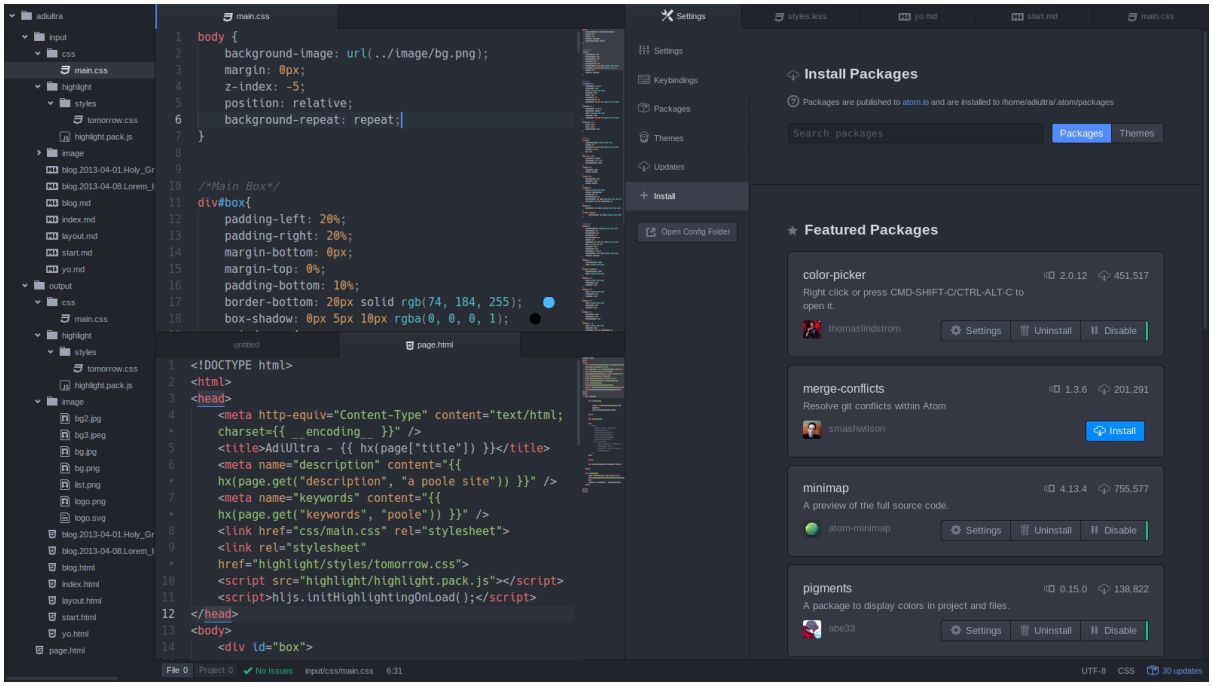

Figure10: Atom Interface

Pros	Cons
• Active community support • Awesome integration with Git • Provides support for managing multiple projects	• Might experience performance issues on older CPUs • Suffers from migration issues

3.8 Rodeo

The logo with the orange hints at the fact that this Python IDE is developed especially for carrying out data analysis. If you have some experience with RStudio, then you will know that Rodeo shares many of its traits with it. For those of you unaware of RStudio, it is the most popular integrated development environment for the R language. Like RStudio, Rodeo's window is divided into four divisions, namely text editor, console, environment for variable visualization, and plot/libraries/files. Amazingly, both Rodeo and RStudio shares a great degree of resemblance with MATLAB.

What's best about Rodeo is that it offers the same level of convenience to both beginners and veterans. As the Python IDE allows you to see and explore while creating simultaneously, Rodeo is undoubtedly one of the best IDEs for those starting out with data science using Python. The IDE also boasts built-in tutorials and comes with helper material.

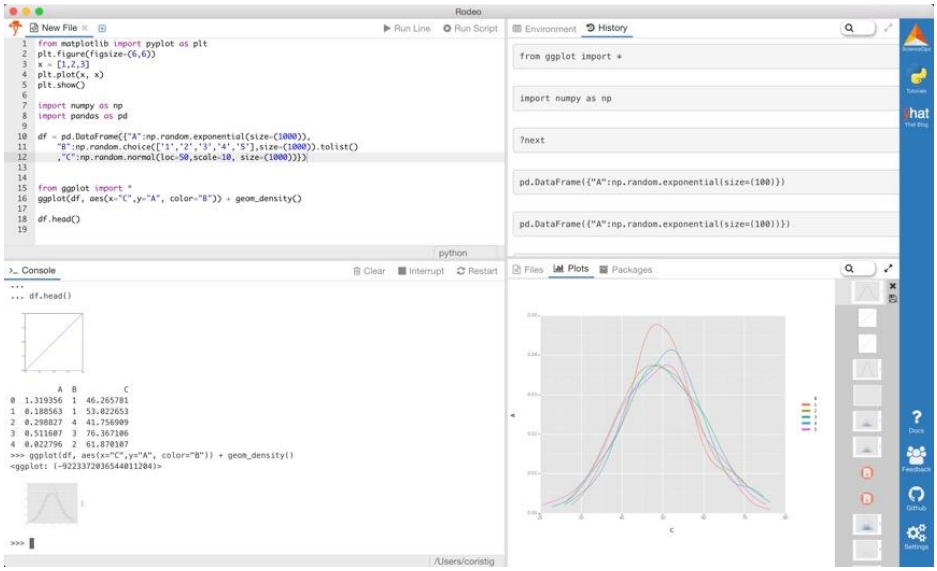

Figure11: Rodeo Interface

Pros	Cons
• a lightweight and flexible development architecture. • block or single execution of code. • built-in Python documentation; • iPython integration, powerful debugger, code autocompletion, visual file navigator and more	• memory issues could happen all too often; • multiple bugs as reported by developers.

3.9 AWS Cloud9

The recently released AWS Cloud9, a cloud-based integrated design environment (IDE) from Amazon, is designed for collaborative programming. It is equipped with numerous features, and on-board connectivity. The AWS web service provides a software platform and the tools to programme in over 40 dynamic programming languages.

- The platform run on a managed EC2 instance, features a browser-based editor which is fully customizable to your programming style. It is configured to allow the code to be written, run and debugged within the browser editor.

- With support for multiple programming languages, defining the run configurations for a specific project language can be set by default. For more control, the development environment allows custom configurations to be setup. Environmental variables, command line options and file naming conventions can all be controlled.
- The cloud-based platform provides a range of tools for the development of serverless applications and for testing Lambda functions. The environment has the necessary tools, SDK's and libraries pre-packaged, to develop serverless applications. With support for the Serverless Application Model (SAM), resource definition and management are simplified with SAM templates. AWS Lambda functions can be edited and debugged within the development environment.
- For businesses running their own Linux servers, the Cloud9 development environment has SSH connectivity. This will allow the IDE to run on any Linux server, whether on-premises, with Amazon AWS or another cloud-based provider.
- To control the running EC2 instance, the IDE includes a built-in terminal with full sudo privileges. The terminal also provides a way to perform Git push and pull functions, to compile code, to run commands or to display server output.
- The AWS Cloud9 IDE allows multiple developers to collaborate in real-time, allowing for pair or team programming. Within the IDE, a chat client enables online team members to chat with each other, without having to leave the development environment. The editor identifies the code attributed to each programmer.

- Cloud9 integrates with other Amazon applications to provide quick deployment of applications. The Codestar application provides quick end-to-end deployment through the CodeCommit, CodeBuild, Code Pipeline and CodeDeploy applications.

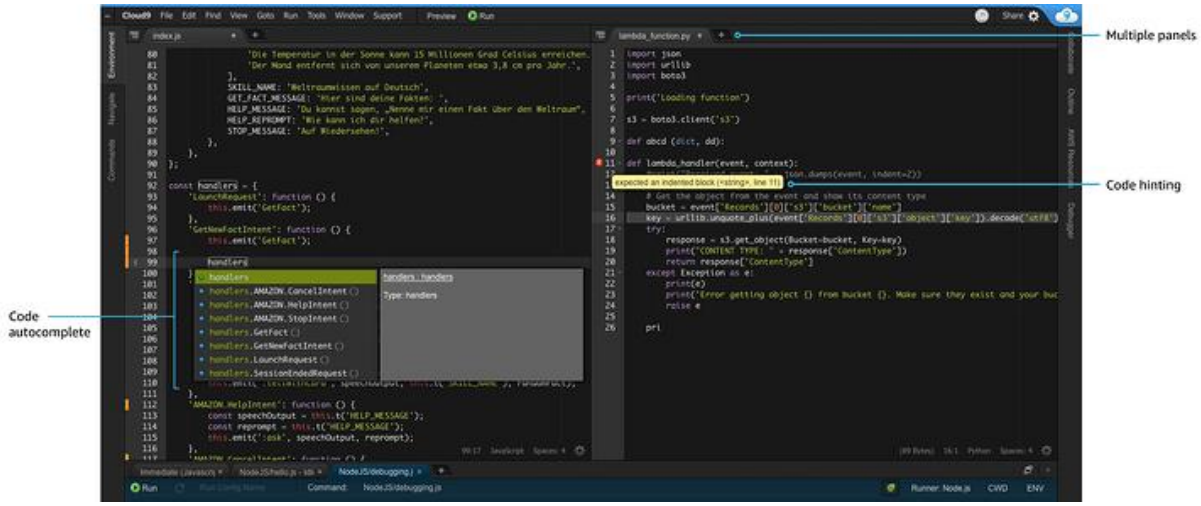

Figure12: AWS Cloud 9 Interface

Pros	Cons
• web-based, no need to install a local IDE; • powerful team collaboration in real-time; • great to create, manage and debug serverless applications; • multiple cursors and a drag-and-drop interface.	• lack of shortcuts; • the documentation leaves wanting more; • complicated setup.

IV. CONCLUSION

Which is the Best IDE?, The answer to this is subjective, but it would still be a good idea to start with an IDE that is less convoluted if you are at a beginner level. Once you get a good handle on coding, you should then switch to a better IDE that has a lot of the built-in features that will allow you to code more efficiently. Features of a Good IDE

Integrated Debugger: A debugger is used to debug program errors and bugs. This feature enables developers to step through their code while executing it. In case if something goes wrong during execution, it would help them find where exactly the error occurred. For example, suppose I am trying to execute my code "print " and instead of printing 'Hello world', it throws an exception saying "SyntaxError: invalid syntax". If I don't know what went wrong, I might try stepping into the code and see how far it got before throwing up an error. With a debugger, I could easily figure out why the code didn't work. In most IDEs, the debugger escalates through your code and applies breakpoints for executing the code partially.

Highlighting of syntaxes: Having the option to spot keywords, variables, and symbols in your code makes perusing and understanding code a lot simpler.

Automated formatting of Code: A rather intriguing element, the code indents itself as you use functions, and loops while you compose your program.

Automatically save and reload source code: A developer's schedule is tightly packed indeed, and hence an IDE should save your work and resume everything later - in a similar state it was in when you left to save you time later on.

Execution from within the environment: It ought to have an inherent compiler to execute your code, if not, then it's merely a code text editor.

REFERENCES

[1]. K. R. Srinath, "Python – The Fastest Growing Programming Language," International Research Journal of Engineering and Technology (IRJET), vol. 4, Issue 12 pp. 354–357, December2017.
[2]. A.L.Sayeth Saabith, T.Vinothraj, MMM.Fareez, "PYTHON CURRENT TREND APPLICATIONS- AN
[3]. OVERVIEW", International Journal of Advance Engineering and Research Development, Volume 6, Issue 10, October 2019, E-ISSN (O): 2348-4470
[4]. Kalyani Adawadkar, "Python Programming-Applications and Future," International Journal of Advance Engineering and Research Development(IJAERD), Special Issue SIEICON-2017,pp. 1–4, April -2017.
[5]. QuantInsti, Python Basics With Illustrations from the Financial Markets, QuantInsti Quantitative Learning Pvt. Ltd.

[6]. Pinky Sodhi, Naman Awasthi, Vishal Sharma, "Introduction to Machine Learning and Its Basic Application in Python", Proceedings of 10th International Conference on Digital Strategies for Organizational Success, January, 2019
[7]. Dhar, V. (2013). "Data science and prediction". Communications of the ACM. 56(12): 64–73. doi:10.1145/2500499
[8]. Jeff Leek (12 December 2013). "The key word in "Data Science" is not Data, it is Science". Simply Statistics. [3] Hayashi, Chikio (1 January 1998). "What is Data Science? Fundamental Concepts and a Heuristic Example". In Hayashi, Chikio; Yajima, Keiji; Bock, Hans-Hermann; Ohsumi, Noboru; Tanaka, Yutaka; Baba, Yasumasa (eds.). Data Science, Classification, and Related Methods. Studies in Classification, Data Analysis, and Knowledge Organization. Springer Japan. pp. 40–51. doi:10.1007/978-4-431- 65950-1_3. ISBN 9784431702085.
[9]. https://www.edureka.co/blog/what-is-data-science/
[10]. Davenport, Thomas H. (January 1, 2006). "Competing on Analytics". Harvard Business Review
[11]. Glossary of Terms. Machine Learning - Special issue on applications of machine learning and the knowledge discovery process archive. Volume 30 Issue 2-3, Feb/March, 1998. Pages 271-274
[12]. https://www.geeksforgeeks.org/history-of-python/
[13]. https://www.python.org/doc/essays/blurb/
[14]. https://docs.python.org/3/library/
[15]. https://insights.stackoverflow.com/survey/2019
[16]. https://insights.stackoverflow.com/survey/2020
[17]. https://trends.google.com/trends/
[18]. https://towardsdatascience.com/
[19]. https://analyticsindiamag.com/10-best-python-libraries-for-computer-vision/
[20]. https://www.tiobe.com/tiobe-index/