5 Python Libraries Every Data Scientist Should Know About

Artem Shelamanov · Follow
Published in Python in Plain English · 7 min read · Jan 16, 2024

972 5

This article is perfect for you if you are a beginner or intermediate machine learning engineer or data scientist. You've already selected your preferred machine learning library like PyTorch or TensorFlow, and mastered choosing the right architecture for your model. You can also train a model and tackle real-world problems. But what's next?

In this article, I'll reveal five libraries that I believe every machine learning engineer and data scientist should be familiar with. These will be a valuable addition to your skill set, making you a more competitive candidate and simplifying the machine learning development process.

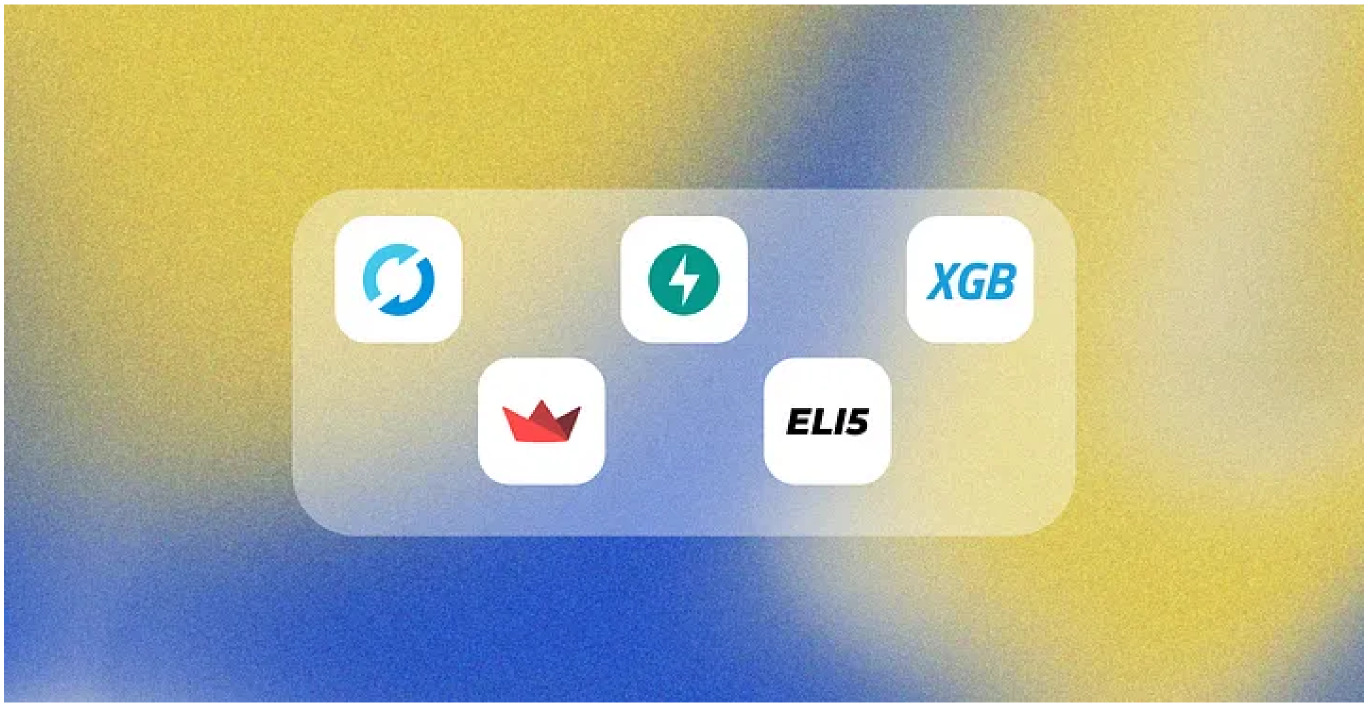


Image by author

1. MLFlow — experiment and model tracking

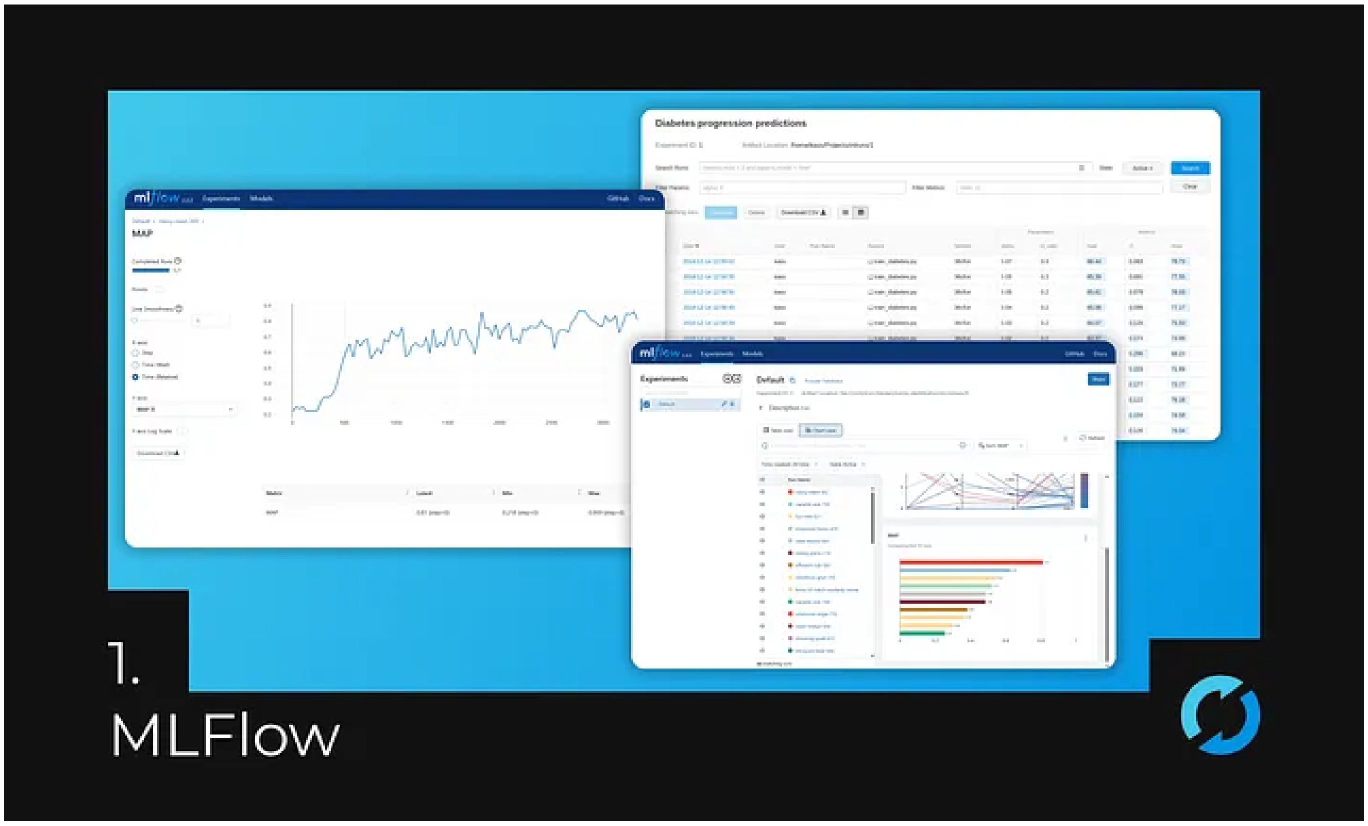


Image by author, examples from https://mlflow.org

Imagine you're an ML developer working on a project to build a model that predicts customer churn. You start by exploring your data using Jupyter notebooks, experimenting with different algorithms and hyperparameters. As you progress, your notebooks become increasingly cluttered with code, results, and visualizations. It becomes **difficult** to **track your progress**, **identify what worked and what didn't**, and **replicate your** best **results**.

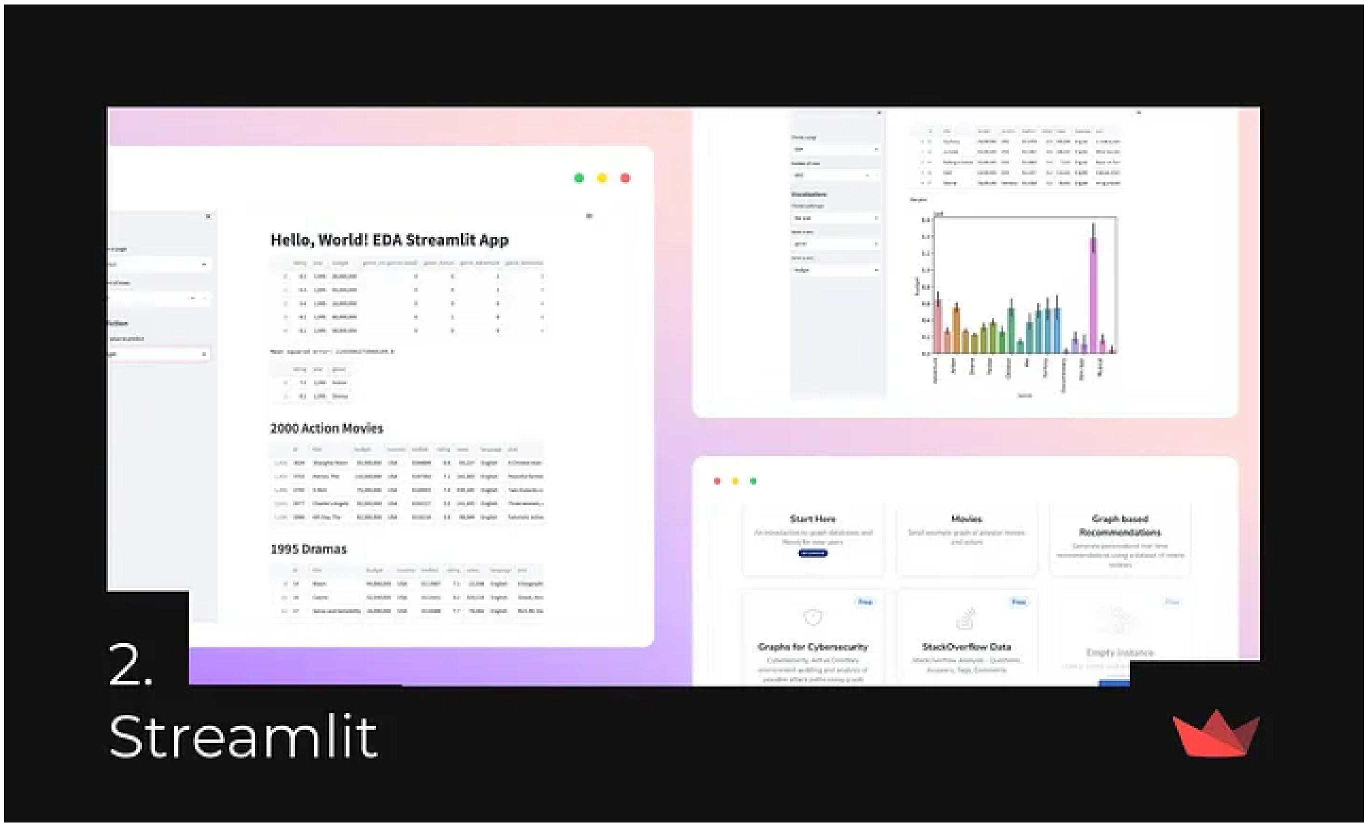
This is where **MLflow** comes in. MLflow is a platform that helps you manage your ML experiments from start to finish, ensuring **traceability** and **reproducibility**. It provides a centralized repository for storing your code, data, and model artifacts, along with a tracking system that records all your experiments, including hyperparameters, metrics, and outputs.

Here's how MLflow helps you avoid the pitfalls of using Jupyter notebooks alone:

1. **Centralized Repository**: MLflow keeps your code, data, and model artifacts organized and easily accessible. You can quickly find the resources you need without getting lost in a maze of notebooks.
2. **Experiment Tracking:** MLflow records every experiment, including the exact code, data, and hyperparameters used. This allows you to easily compare different experiments and identify what led to the best results.
3. **Reproducibility:** MLflow makes it possible to reproduce your best models with the exact same code, data, and environment. This is crucial for ensuring the consistency and reliability of your results.

So, if you're serious about building effective machine learning models, ditch the Jupyter notebook chaos and embrace the power of MLflow.

2. Streamlit — quick and pretty web applications



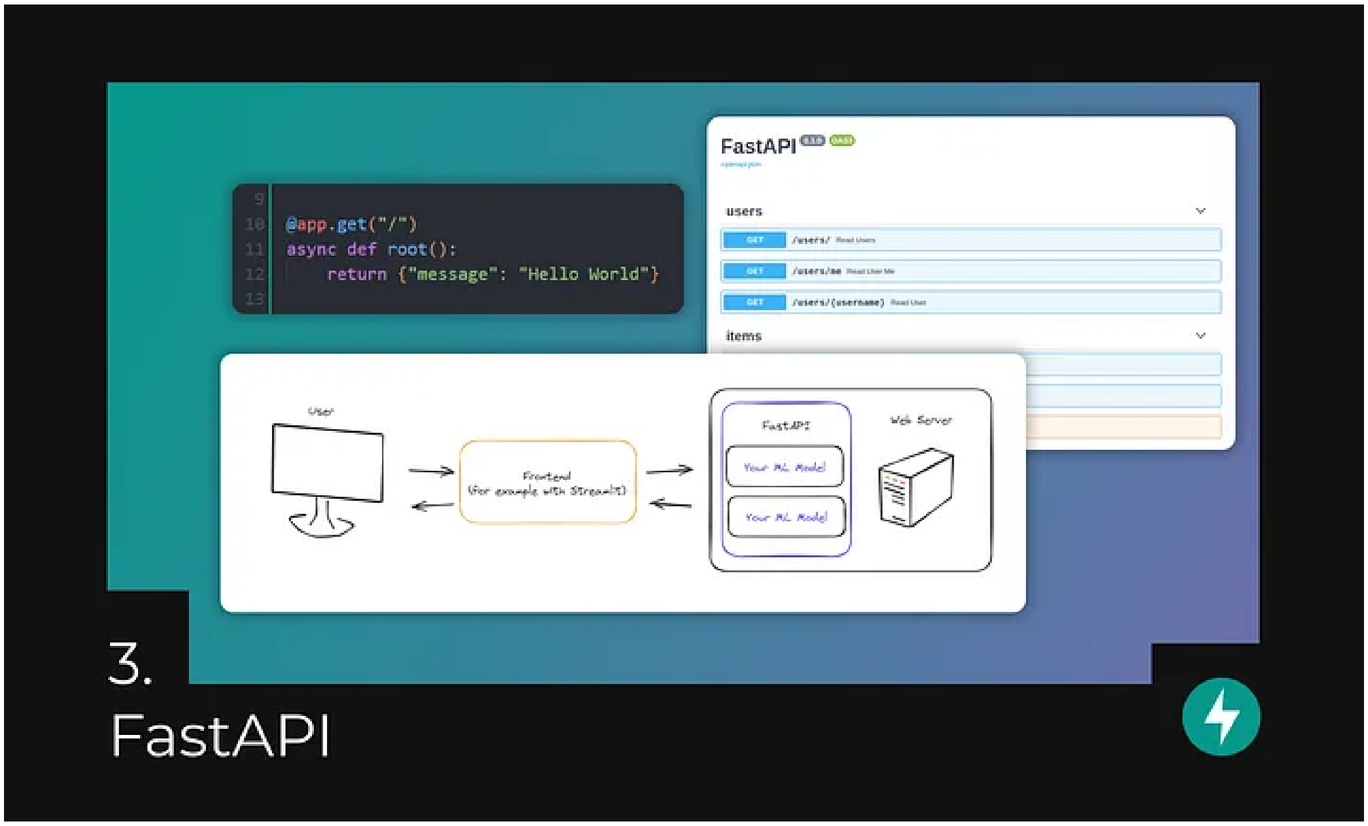
Streamlit is the most popular Frontend framework for Data Science. It is an open-source Python framework. It allows users to create interactive data apps rapidly and with ease, making it particularly beneficial for data scientists and machine learning engineers who may not have extensive web development knowledge.

With Streamlit, developers can build and share attractive user interfaces and deploy models without requiring in-depth front-end experience or knowledge. The framework is free, all-Python, and open-source, enabling the creation of shareable web apps in minutes.

If you have some pet project involving machine learning, a good idea would be adding an interface to it using Streamlit. It takes no time to start working with it, there are many templates ready, and you can finish your frontend in minutes. It's also extremely easy to share it, meaning it will definitely look good in your resume.

If you want to look into other frontend libraries in Python, be sure to check out my article Top-5 Python Frontend Libraries for Data Science.

3. FastAPI — deploy your models easily and quickly



```
@app.get("/")
async def root():
    return {"message": "Hello World"}
```

Image by author

Once you've trained and validated your model, you need to deploy it so that it can be used by other applications. This is where FastAPI comes in.

FastAPI is a high-performance web framework for building RESTful APIs. It's known for its simplicity, ease of use, and speed. This makes it an ideal choice for deploying machine learning models to production.

Here are some of the reasons why ML engineers and data scientists should learn FastAPI:

- **Speed**: FastAPI is incredibly fast. It uses a modern asynchronous programming model that makes it efficient at handling multiple requests simultaneously. This is essential for deploying machine learning models that need to process large amounts of data.
- **Simplicity**: FastAPI is easy to learn and use. It has a clear and concise syntax that makes it easy to write clean and maintainable code. This is important for ML engineers and data scientists who are not necessarily experienced web developers.
- **Ease of use**: FastAPI provides a lot of features that make it easy to build and deploy APIs. For example, it has built-in support for automatic documentation, data validation, and error handling. This saves time and effort, allowing ML engineers to focus on their core work of building and deploying models.
- **Production-ready**: FastAPI is designed for production use. It has features like support for multiple backends, security, and deployment tools. This makes it a reliable choice for deploying critical machine learning models.

In conclusion, FastAPI is a powerful and versatile tool that can be used to deploy machine learning models to production. Its ease of use, speed, and production-readiness make it an ideal choice for ML engineers and data scientists who want to make their models accessible to others.

4. XGBoost — faster and better way to predict tabular data

Image by author, source 1 and source 2

XGBoost is a powerful machine learning algorithm that is known for its accuracy, speed, and scalability. It is based on the gradient boosting framework, which combines multiple weak learners into a strong learner. In simple words, you use multiple small models like random forest, you combine them into a big model, and in the end you get a faster model (compared to, for example, a neural net), but at the same time it's scalable and is less prone to overfitting.

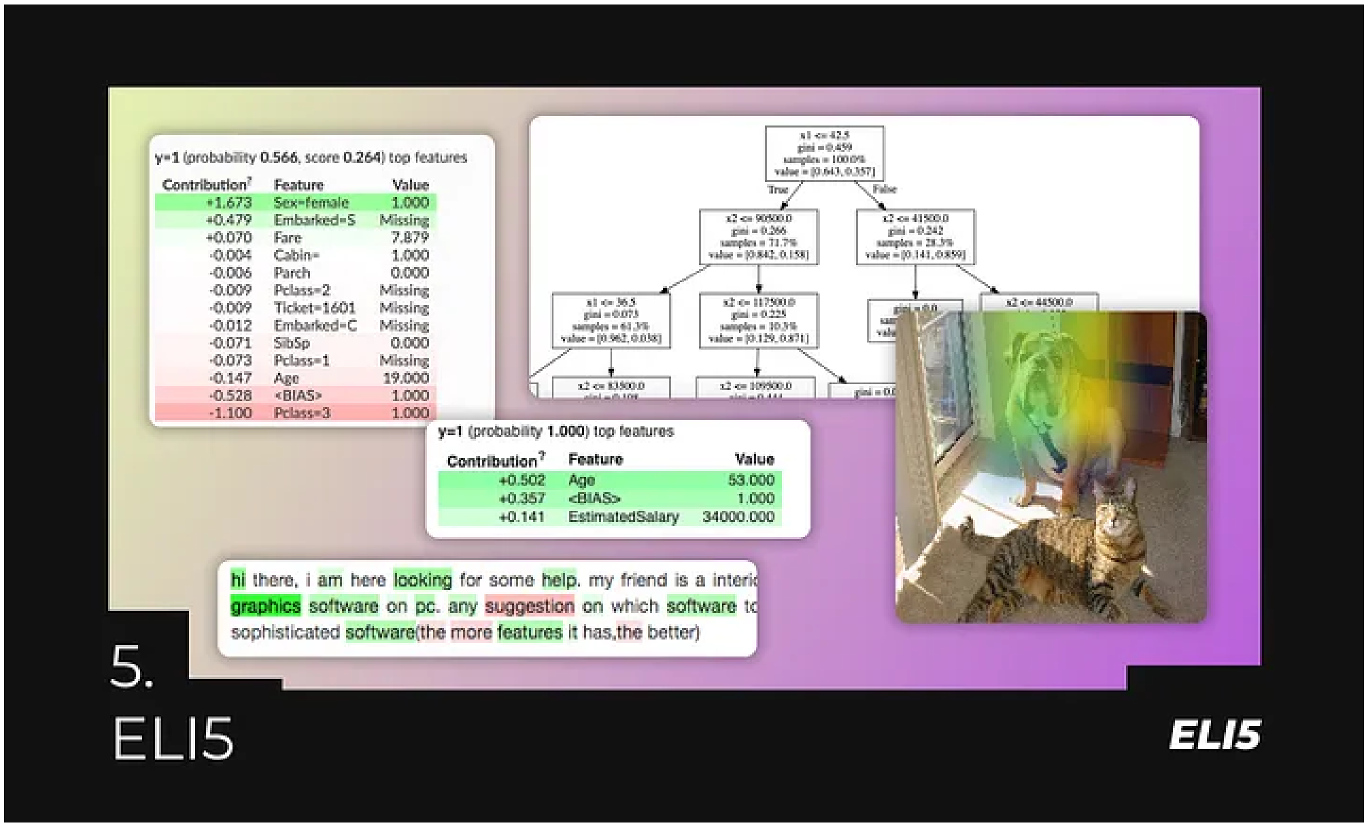
Here are some of the reasons why ML engineers and data scientists should learn XGBoost:

- **Accuracy**: XGBoost is one of the most accurate machine learning algorithms available. It has been used to win many machine learning competitions, and it is consistently ranked among the top algorithms for various tasks.

- **Speed:** XGBoost is also very fast. It is able to train and make predictions on large datasets quickly and efficiently. This makes it a good choice for applications where speed is important, such as real-time fraud detection or financial modeling.
- **Scalability:** XGBoost is highly scalable. It can handle large datasets and complex models without sacrificing accuracy. This makes it a good choice for applications where data volume or model complexity is a concern.

If you have a task with tabular data (like predicting a price of a house based on the rooms it has, or calculating the likelihood of customer buying the product based on last purchases/account data), XGBoost is something you should always try first, before resorting to neural nets with Keras or PyTorch.

5. ELI5 — make your models more interpretable and transparent



y=1 (probability 0.566, score 0.264) top features

Contribution?	Feature	Value
+1.673	Sex=female	1.000
+0.479	Embarked=S	Missing
+0.070	Fare	7.879
-0.004	Cabin=	1.000
-0.006	Parch	0.000
-0.009	Pclass=2	Missing
-0.009	Ticket=1601	Missing
-0.012	Embarked=C	Missing
-0.071	SibSp	0.000
-0.073	Pclass=1	Missing
-0.147	Age	19.000
-0.528	<BIAS>	1.000
-1.100	Pclass=3	1.000

y=1 (probability 1.000) top features

Contribution?	Feature	Value
+0.502	Age	53.000
+0.357	<BIAS>	1.000
+0.141	EstimatedSalary	34000.000

hi there, i am here looking for some help. my friend is a interi
graphics software on pc. any suggestion on which software to
sophisticated software(the more features it has,the better)

Image by author and from source 1, source 2

After you trained your model, you can just deploy it and use it. But at this point the model will be more of a "black box" — you plug stuff in, you get stuff out. How does it work? Nobody knows. Numbers go here, numbers go there, and in the end you get an answer.

And what if your customer/boss asks you how did your model come up with some specific answer? Again, you don't know. Or maybe you want to know which parameters are the most important during the training, and which just add noise?

All of these questions can be answered using ELI5. This library will help you to make your model transparent, interpretable and easier to understand. But you may get a lot more information about not only the model, but also the architecture may work better, and what problems does the current model have.

ELI5 already supports libraries like Scikit-Learn, Keras, XGBoost and many others. You can debug your model for classification of images, texts and tabular data.

Conclusion

We've explored five leading data science frameworks. If you grasp every library at least a bit, you will gain multiple advantages:

1. Compared to other data scientists, you will have more chances at landing a job, since you gained multiple skills in different aspects of machine learning.
2. You will be able to work on full-stack projects, since you can not only develop your model, but deploy it with a **FastAPI** backend and let users interact with it in **Streamlit** frontend.
3. You will not get lost in "Jupyter Notebook hell", as all your machine learning experiments will be traceable and reproducible with **MLFlow**, and all models will be versioned correctly.
4. Tabular data is not a problem for you, since you know how to quickly train a scalable, fast and accurate models using **XGBoost.**
5. And most of the models are not "black boxes" for you any more, since you can understand them deeper with **ELI5**, debugging their thinking process and explaining their predictions.

All these libraries will make your life easier, adding many useful and important skills to your arsenal. Happy coding!

References

1. https://mlflow.org
2. https://streamlit.io
3. https://blog.streamlit.io/building-a-streamlit-and-scikit-learn-app-with-chatgpt/
4. https://fastapi.tiangolo.com
5. https://www.geeksforgeeks.org/xgboost/
6. https://xgboost.readthedocs.io/en/stable/
7. https://github.com/TeamHG-Memex/eli5
8. https://eli5.readthedocs.io/en/latest/overview.html#basic-usage
9. https://www.analyticsvidhya.com/blog/2020/11/demystifying-model-interpretation-using-eli5/

Machine Learning · Data Science · Python · Development · AI

Written by Artem Shelamanov

Follow

787 Followers · Writer for Python in Plain English

Data Scientist, Computational Physicist and Game Developer.

More from Artem Shelamanov and Python in Plain English

FAILED

Miner Of Ideas in Python in Plain English

Stop Doing It in Python

Once Taylor Swift said "You will never be able to find happiness if you stay attached to the...

· 7 min read · Jan 25, 2024

1.1K 26

FAILED FAILED

See all from Artem Shelamanov

See all from Python in Plain English

Recommended from Medium

Jason Roell

FAILED

Ultimate Python Cheat Sheet: Practical Python For Everyday…

This Cheat Sheet was born out of necessity. Recently, I was tasked with diving into a new…

33 min read · Jan 30, 2024

1.92K 26

Lists

Predictive Modeling w/ Python

20 stories · 928 saves

Practical Guides to Machine Learning

10 stories · 1091 saves

Coding & Development

11 stories · 458 saves

Natural Language Processing

1223 stories · 702 saves

FAILED

Anmol Tomar in CodeX

Say Goodbye to Loops in Python, and Welcome Vectorization!

Use Vectorization—a super-fast alternative to loops in Python

· 5 min read · Dec 28, 2023

4.3K 49

Liu Zuo Lin in Level Up Coding

7 Things I Wish I Knew Earlier About Python Classes

Day 90 of experimenting with video content:

· 5 min read · 6 days ago

1K 6

Siavash Yasini in Towards Data Science

Python's Most Powerful Decorator

And 5 ways to use it in data science and machine learning

· 11 min read · Feb 2, 2024

2.2K 16

See more recommendations