3

Working and Learning with Kaggle Notebooks

Kaggle Notebooks – which until recently were calledKernels – are Jupyter notebooks in the browser that can run free of charge. This means you can execute your experiments from any device with an internet connection, although something bigger than a mobile phone is probably a good idea. The technical specification of the environment (as of the time of this writing) is quoted below from the Kaggle website; the most recent version can be verified at https://www.kaggle.com/docs/notebooks:

* 9 hours execution time
* 20 Gigabytes of auto-saved disk space (/kaggle/working)
* Additional scratchpad disk space (outside /kaggle/working) that will not be saved outside of the current session

CPU Specifications

* 4 CPU cores
* 16 Gigabytes of RAM

GPU Specifications

* 2 CPU cores
* 13 Gigabytes of RAM

TPU Specifications

* 4 CPU cores
* 16 Gigabytes of RAM

In this chapter, we will cover the following topics:

* Setting up a Notebook
* Running your Notebook
* Getting the best out of Notebooks
* Upgrading to Google Cloud Platform (GCP)
* One step beyond: using Notebooks in your data science portfolio

Kaggle Learn courses

Without further ado, let us jump into it. The first thing we need to do is figure out how to set up a Notebook.

# Setting up a Notebook

There are two primary methods of creating a Notebook: from the front page or from a dataset.

To proceed with the first method, go to the Code section of the menu on the left-hand side of the landing page at <https://www.kaggle.com/> and click the + New Notebook button. This is the preferred method if you are planning an experiment that involves uploading your own dataset:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Figure 3.1: Creating a new Notebook from the Code page

Alternatively, you can go to the page of the dataset you are interested in and click the New Notebook button there, as we saw in the previous chapter:

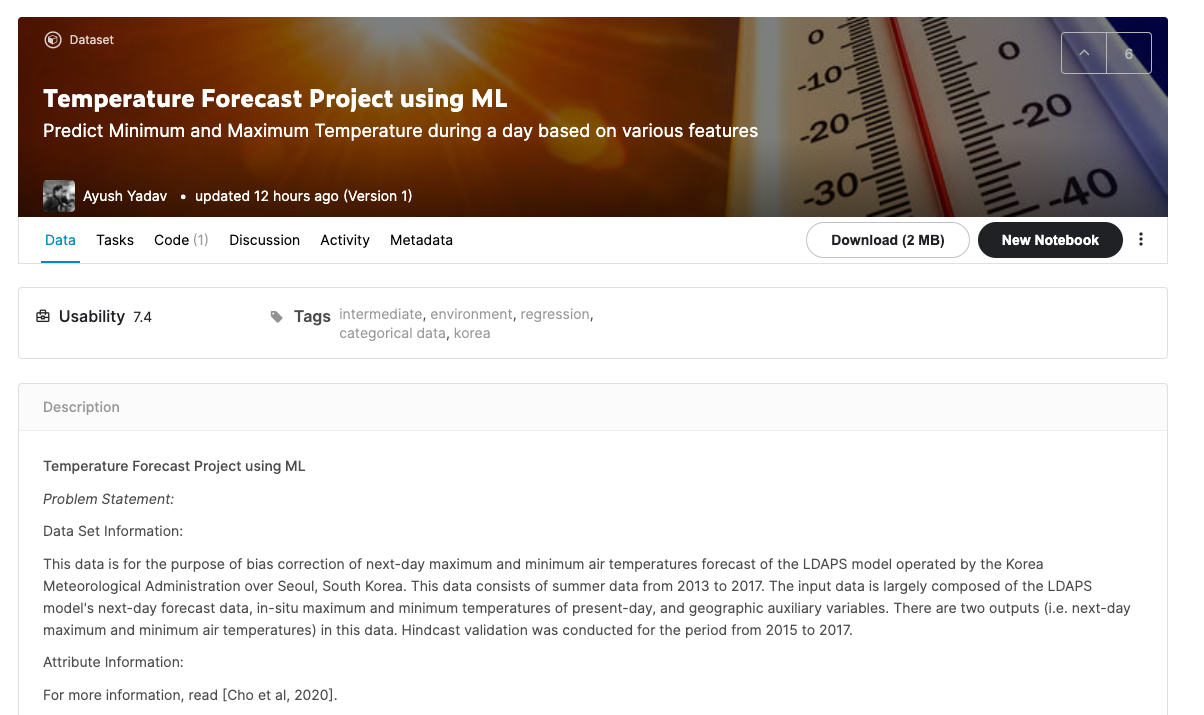


Figure 3.2: Creating a new Notebook from a dataset page

Whichever method you choose, after clicking New Notebook, you will be taken to your Notebook page:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Figure 3.3: The Notebook page

On the right-hand side of the new Notebook page shown above, we have a number of settings that can be adjusted:

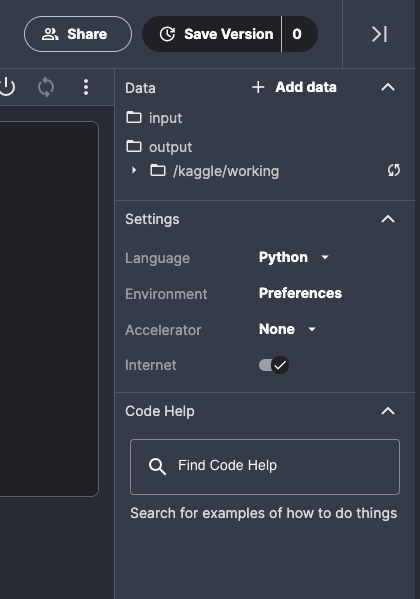


Figure 3.4: Notebook options

We will discuss the settings briefly. First, there is the coding Language. As of the time of this writing, the Kaggle environment only allows Python and R as available options for coding your Notebooks. By default, a new Notebook is initialized with the language set to Python – if you want to use R instead, click on the dropdown and select R.

Next comes Environment: this toggle allows you to decide whether to always use the latest Docker environment (the risky option; fast to get updates but dependencies might break with future updates) or pin the Notebook to the original version of the environment provided by Kaggle (the safe choice). The latter option is the default one, and unless you are conducting very active development work, there is no real reason to tinker with it.

Accelerator allows a user to choose how to run the code: on CPU (no acceleration), GPU (necessary for pretty much any serious application involving deep learning), or TPU. Keep in mind that moving from CPU to (a single) GPU requires only minimal changes to the code and can be handled via system device detection. Migrating your code to TPU requires more elaborate rewriting, starting with data processing. An important point to keep in mind is that you can switch between CPU/GPU/TPU when you are working on your Notebook, but each time you do, the environment is restarted and you will need to run all your code from the beginning.

Finally, we have the Internet toggle which enables or disables online access. If you are connected and need to, for example, install an extra package, the download and installation of dependencies will take place automatically in the background. The most common situation in which you need to explicitly disable internet access is for submission to a competition that explicitly prohibits online access at submission time.

An important aspect of using Notebooks is that you can always take an existing one (created by yourself or another Kaggler) and clone it to modify and adjust to your needs. This can be achieved by clicking the Copy and Edit button at the top right of the Notebook page. In Kaggle parlance, the process is referred to as forking:

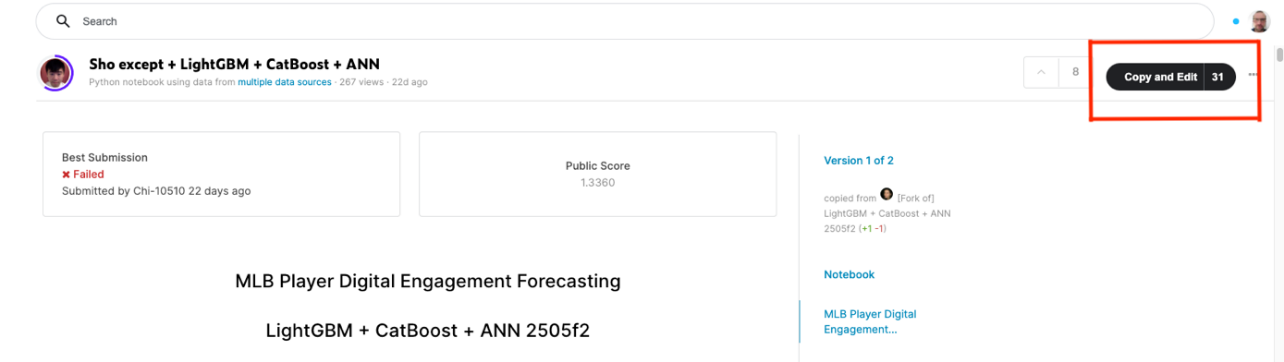


Figure 3.5: Forking an existing Notebook

A note on etiquette: If you have participated in a Kaggle competition before, you will probably have noticed that the leaderboard is flooded with forks of forks of well-scoring Notebooks. There is nothing wrong with building on somebody else’s work – but if you do, remember to upvote the original author and give explicit credit to the creator of the reference work.

A Notebook you create is private (only visible to you) by default. If you want to make it available to others, you can choose between adding collaborators, so that only the users explicitly added to the list will be able to view or edit the content, or making the Notebook public, in which case everybody can see it.

# Running your Notebook

All the coding is finished, the Notebook seems to be working fine, and you are ready to execute. To do that, go to the upper-right corner of your Notebook page and click Save Version.

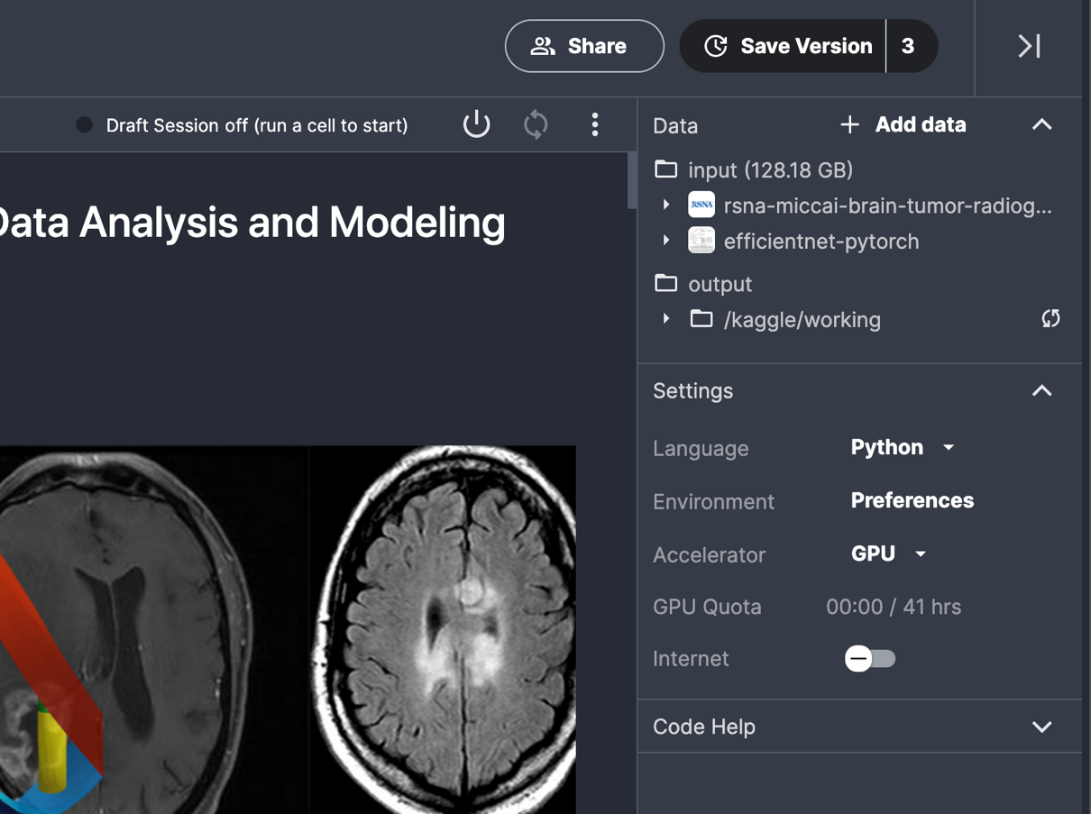


Figure 3.6: Submitting your script

Save & Run All is usually used to execute the script, but there is also a Quick Save option, which can be used to save an intermediate version of the script before it is ready for submission:

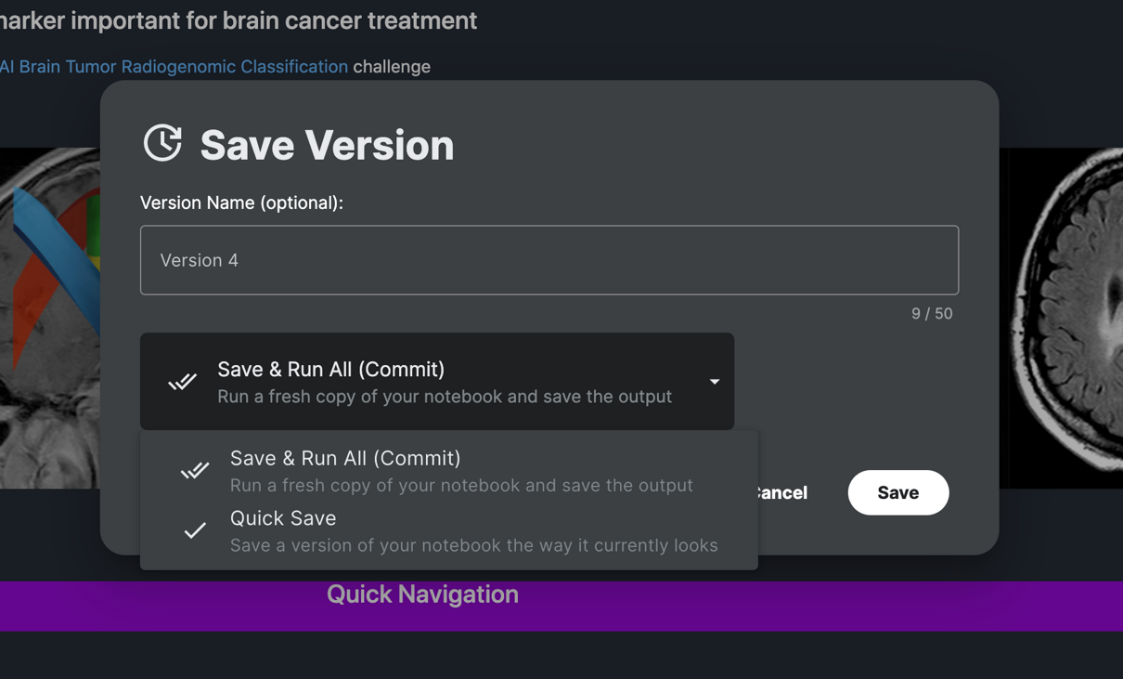


Figure 3.7: Different options for Save Version

Once you have launched your script(s), you can head to the lower-left corner and click on Active Events:

Obraz zawierający tekst, zrzut ekranu, monitor, ekran

Opis wygenerowany automatycznie

Figure 3.8: Monitoring active events

In this manner, you can monitor the behavior of your Notebooks. Normal execution is associated with the message Running, otherwise, it is displayed as Failed. Should you decide that you want to kill a running session for whatever reason (for instance, you realize you forgot to use the most recent data), you can do it by clicking on the three dots on the right-hand side of your script entry under Active Events and you will receive a pop-up like the one shown in the figure below:

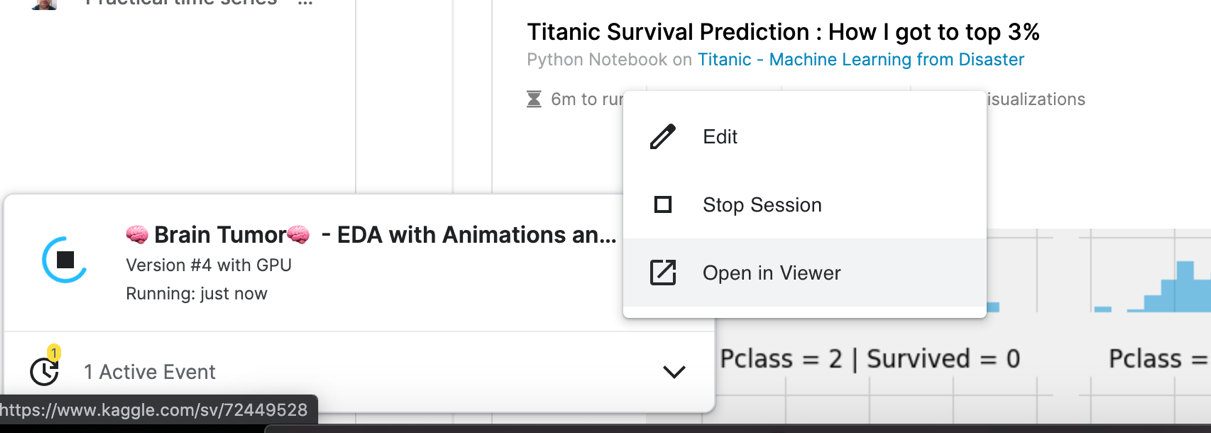


Figure 3.9: Canceling Notebook execution

# Saving Notebooks to Github

A recently introduced feature (see: <https://www.kaggle.com/product-feedback/295170>) allows you to store your code or your notebook to the version control repository Github (<https://github.com/>). You can store your work both to public and private repositories and that happens automatically as you save a version of your code. Such a feature can prove quite useful to share your work with your Kaggle teamates as well as showcasing your work to a wider public.

In order to abilitate this feature, you need to edit your notebook and on the File menu, choose the Link to Github option.

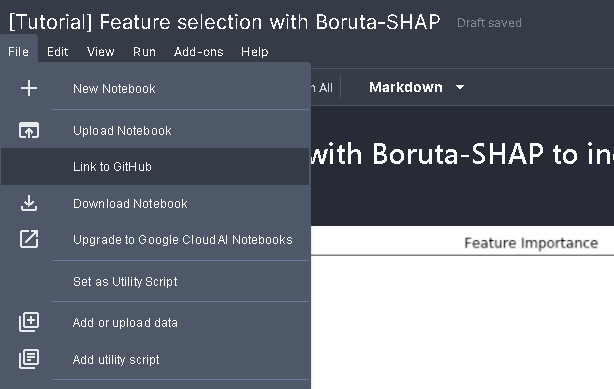


Figure 3.10: Where to abilitate the Github feature

After having choosen the option, you will have to link your Github account to the notebook. You will be explicitly asked for linking permissions only the first time you opt for it. For any successive link to any new notebook of yours, the operations will be automatic.

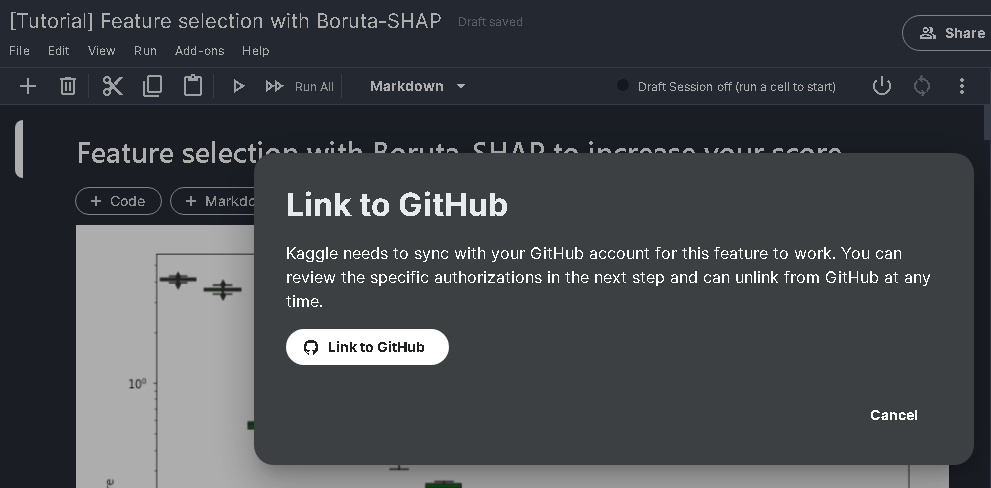


Figure 3.11: Linking to Github

Only after linking your notebook you will be allowed the synching of your work to a repository of your choice when you decide to save your work.

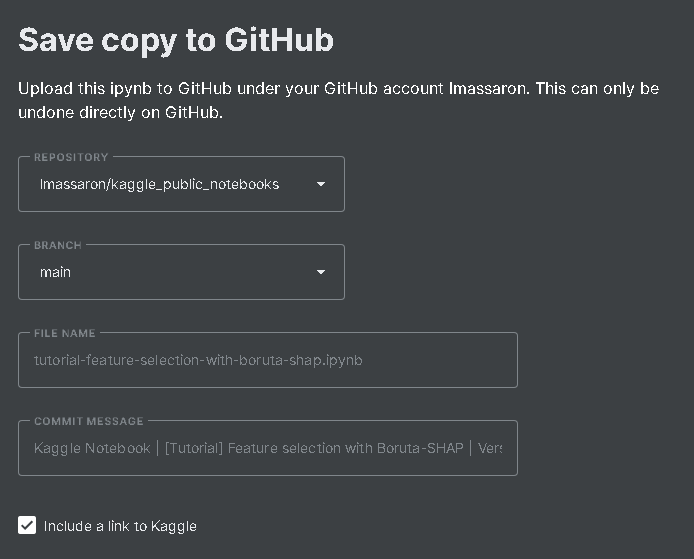


Figure 3.12: Committing your work to Github

After having decided for a repository and for a branch (thus allowing you to store different developments of your work), you can change the name of the file you are going to push to the repository and modify the commit message.

If by chance you decide not to synch anymore a notebook on Github, all you have to do is to get back to the File menù and select Unlink from Github. Finally, if you definitely want Kaggle to stop connecting with your GitHub repository, you can unlink your accounts from either your Kaggle account page under My linked accounts or from GitHub’s settings pages(https://github.com/settings/applications).

# Getting the most out of Notebooks

Kaggle does provide a certain amount of resources for free, with the quotas resetting weekly. You get a certain number of hours to use with both GPU and TPU; it is 30 hours for TPU, but for GPU the numbers can vary from week to week (you can find the official statement describing the new "floating" quotas policy here: <https://www.kaggle.com/product-feedback/173129>). You can always monitor your usage in your own profile:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Figure 3.13: Current status for accelerator quotas

While the amounts might seem large at first glance, this initial impression can be deceptive; it is actually fairly easy to use your quota very quickly. Some practical suggestions that can help you control the usage of the resources:

* The counter for the quota (measuring how long you have been using your chosen accelerator, GPU or TPU) starts running the moment you initialize your Notebook.
* This means that you should always start by checking that GPU is disabled under settings (see Figure 3.5 above). Write the boilerplate first, check your syntax, and enable/disable GPU for when you add the parts of the code that actually depend on GPU initialization. A reminder: the Notebook will restart when you change the accelerator.

It is usually a good idea to run the code end-to-end on a small subset of data to get a feel for the execution time. All Notebooks, irrespective of the chosen accelerator, have a hard limit of 9 hours of execution time. This way, you minimize the risk that your code will crash due to exceeding this limit.

Sometimes the resources provided freely by Kaggle are not sufficient for the task at hand, and you need to move to a beefier machine. A good example is a recent tumor classification competition:

<https://www.kaggle.com/c/rsna-miccai-brain-tumor-radiogenomic-classification/data>

If your raw data is over 100GB, you need to either resize/downsample your images (which is likely to have an adverse impact on your model performance) or train a model in an environment capable of handling high-resolution images. You can set up the whole environment yourself (an example of this setup is the section Using Kaggle datasets in Google Colab in Chapter 2), or you can stay within the framework of Notebooks but swap the underlying machine. This is where Google Cloud AI Notebooks come in.

## Upgrading to Google Cloud Platform (GCP)

The obvious benefit to upgrading to GCP is getting access to more powerful hardware: a Tesla P100 GPU (provided free by Kaggle) is decent for many applications, but not top of the line in terms of performance, and 16GB RAM can also be quite limiting, especially in resource-intensive applications like large NLP models or high-resolution image processing. While the improvement in execution time is obvious, leading to faster iteration through the development cycle, it comes at a cost: you need to decide how much you are prepared to spend. For a powerful machine crunching the numbers, time is quite literally money.

In order to migrate your Notebook to the GCP environment, go to the sideline menu on the right-hand side and click on Upgrade to Google Cloud AI Notebooks:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Figure 3.14: Upgrading to the Google Cloud AI Notebooks option

You will be greeted by the following prompt:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Figure 3.15: Upgrade to Google Cloud AI Platform Notebooks prompt

When you click Continue, you will be redirected to the Google Cloud Platform console, where you need to configure your billing options. As a reminder: GCP is not free. If it is your first time, you will need to complete a tutorial guiding you through the necessary steps.

## One step beyond

As mentioned earlier in this chapter, Kaggle Notebooks are a fantastic tool for education and participating in competitions; but they also serve another extremely useful purpose, namely as a component of a portfolio you can use to demonstrate your data science skills.

There are many potential criteria to consider when building your data science portfolio (branding, audience reach, enabling a pitch to your potential employer, and so on) but none of them matter if nobody can find it. Because Kaggle is part of Google, the Notebooks are indexed by the most popular search engine in the world; so if someone is looking for a topic related to your code, it will show up in their search results.   
  
Below, I show a personal example: a few years ago, I wrote a Notebook for a competition. The problem I wanted to tackle was adversarial validation (for those unfamiliar with the topic: a fairly easy way to see if your training and test sets have a similar distribution is to build a binary classifier trained to tell them apart; the concept is covered in more detail in Chapter 6, Designing Good Validation). When writing this chapter, I tried to search for the Notebook and, lo and behold, it shows up high up in the search results (notice the fact that I did not mention Kaggle or any personal details like my name in my query):

Obraz zawierający tekst

Opis wygenerowany automatycznie

Figure 3.16: Konrad’s Notebook showing up on Google

Moving on to other benefits of using Notebooks to demonstrate your skillset: just like Competitions, Datasets, and Discussions, Notebooks can be awarded votes/medals and thus position you in the progression system and ranking. You can stay away from the competitions track and become an Expert, Master, or Grandmaster purely by focusing on high-quality code the community appreciates. The most up-to-date version of the progression requirements can be found at <https://www.kaggle.com/progression>; below we give a snapshot relevant to Notebooks:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Figure 3.17: Tier progression requirements

Progressing in the Notebooks category can be a challenging experience; while easier than Competitions, it is definitely harder than Discussions. The most popular Notebooks are those linked to a specific competition: exploratory data analysis, end-to-end proof of concept solutions, as well as leaderboard chasing; it is an unfortunately common practice that people clone the highest-scoring public Notebook, tweak some parameters to boost the score, and release it to wide acclaim (if upvotes can be considered a measure of sentiment). This is not meant to discourage the reader from publishing quality work on Kaggle – a majority of Kagglers do appreciate novel work and quality does prevail in the long term – but a realistic adjustment of expectations is in order.

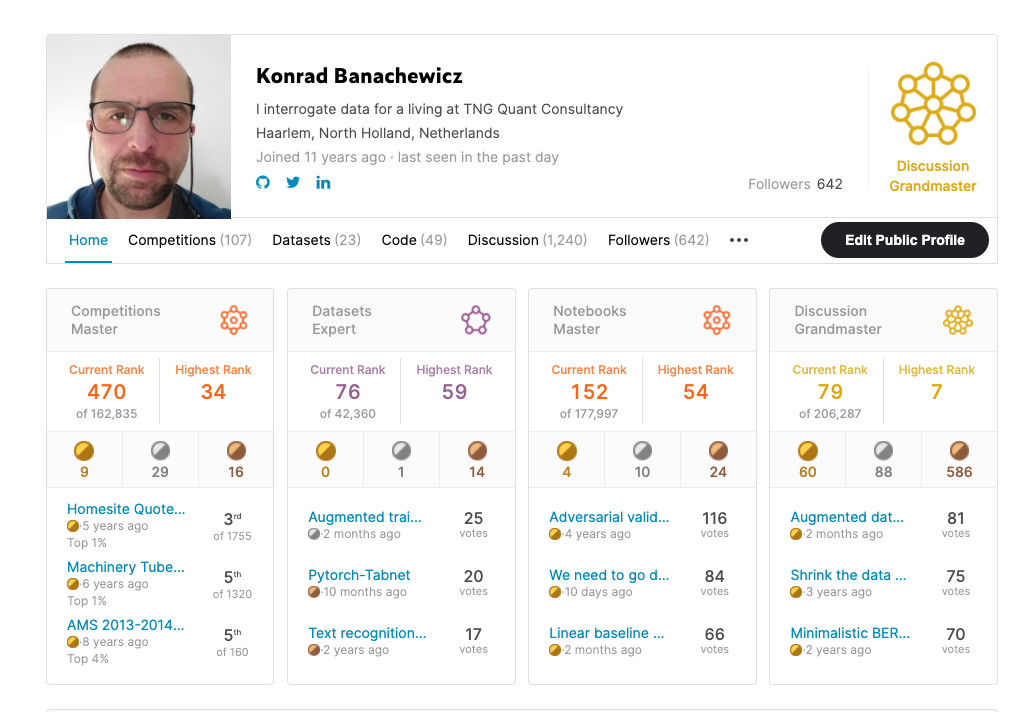
Your Kaggle profile comes with followers and gives you the possibility of linking other professional networks like LinkedIn or GitHub, so you can leverage the connections you gain inside the community:  


Figure 3.18: Konrad’s Kaggle profile

In this day and age, it is easy to be skeptical about claims of “community building”, but in the case of Kaggle, it happens to actually be true. Their brand recognition in the data science universe is second to none, both among practitioners and among recruiters who actually do their homework. In practice, this means that a (decent enough) Kaggle profile can get you through the door already; which, as we all know, is frequently the hardest step.

We had the pleasure of speaking to Martin Henze, aka Heads or Tails (<https://www.kaggle.com/headsortails>), a Kaggle Grandmaster in Notebooks and Discussion and a Data Scientist at Edison Software. Martin is also the author of Notebooks of the Week: Hidden Gems, a weekly collection of the very best Notebooks that have escaped public notice. You can get noticed about the new releases of his Hiddn Gems by simply following his Kaggle profile or his accounts on Twitter and LinkedIN.

What's your favourite kind of competition and why? In terms of techniques, solving approaches, what is your speciality on Kaggle?

For a long time, my focus was on EDA notebooks rather than leaderboard predictions themselves. Most of my experience prior to Kaggle had been with tabular data, and the majority of my EDA notebooks deal with extracting intricate insights from newly launched tabular challenges. I still consider this my specialty on Kaggle, and I have spent a significant amount of time crafting the structure, data visualizations, and storytelling of my notebooks.

How do you approach a Kaggle competition? If you work in data science, how different is this approach to what you do in your day-to-day work?

Even as Kaggle has shifted away from tabular competitions, I strongly believe that the data themselves are the most important aspect of any challenge. It is easy to focus too early on model architectures and hyperparameter tuning. But in many competitions, the key to success remains a data-centric approach that is built on detailed knowledge of the dataset and its quirks and peculiarities. This is true for image data, NLP, time series, and any other data structures you can think of. Therefore, I always start with an extensive EDA before building a simple baseline model, a CV framework, and then slowly iterating the complexity of this pipeline.

The main difference compared to my data science day job is probably that the kind of baseline models that most experienced people can build within the first week of a new challenge would be considered sufficient to put into production. In many cases, after those first few days we’re more than 80% on the way to the ultimate winner’s solution, in terms of scoring metric. Of course, the fun and the challenge of Kaggle are to find creative ways to get those last few percent of, say, accuracy. But in an industry job, your time is often more efficiently spent in tackling a new project instead.

Has Kaggle helped you in your career? If so, how?

Kaggle has shaped and supported my career tremendously. The great experience in the Kaggle community motivated me to transition from academia to industry. Today, I’m working as a data scientist in a tech startup and I’m continuously growing and honing my skills through Kaggle challenges.

In my case, my focus on constructing extensive Kaggle notebooks helped me a lot, since I could easily use those as my portfolio. I don’t know how often a hiring manager would actually look at those resources, but I frequently got the impression that my Grandmaster title might have opened more doors than my PhD did. Or maybe it was a combination of the two. In any case, I can much recommend having a portfolio of public notebooks. Moreover, during my job search, I used the strategies I learned on Kaggle for various take-home assignments and they served me well.

In your experience, what do inexperienced Kagglers often overlook? What do you know now that you wish you'd known when you first started?

I think that we are all constantly growing in experience. And we’re all wiser now than we were ten years, five years, or even one year ago. With that out of the way, one crucial aspect that is often overlooked is that you want to have a plan for what you’re doing, and to execute and document that plan. And that’s an entirely understandable mistake to make for new Kagglers, since everything is novel and complex and at least somewhat confusing. I know that Kaggle was confusing for me when I first joined. So many things you can do: forums, datasets, challenges, courses. And the competitions can be downright intimidating: Neuronal Cell Instance Segmentation; Stock Market Volatility Prediction. What even are those things? But the competitions are also the best place to start.

Because when a competition launches, nobody really has a clue about it. Yeah, maybe there is a person who has done their PhD on almost the same topic. But those are rare. Everyone else, we’re all pretty much starting from zero. Digging into the data, playing with loss functions, running some simple starter models. When you join a competition at the beginning, you go through all that learning curve in an accelerated way, as a member of a community. And you learn alongside others who will provide you with tons of ideas. But you still need a plan.

And that plan is important, because it’s easy to just blindly run some experiments and see all that GPU RAM being used and feel good about it. But then you forget which version of your model was doing best, and is there a correlation between local validation and LB? Did I already test this combination of parameters? So write down what you are going to do and then log the results. There are more and more tools that do the logging for you, but this is also easily done through a custom script. Machine learning is still mostly an experimental science, and the key to efficient experiments is to plan them well and to write down all of the results so you can compare and analyse them.What mistakes have you made in competitions in the past?

I have made lots of mistakes and I hope that I managed to learn from them. Not having a robust cross-validation framework was one of them. Not accounting for differences between train and test. Doing too much EDA and neglecting the model building – that one was probably my signature mistake in my first few competitions. Doing not enough EDA and missing something important – yep, done that too. Not selecting my final two submissions. (Ended up making not much of a difference, but I still won’t forget it again.)

The point about mistakes, though, is similar to my earlier point about experiments and having a plan. Mistakes are fine if you learn from them and if they help you grow and evolve. You still want to avoid making easy mistakes that could be avoided by foresight. But in machine learning (and science!) failure is pretty much part of the process. Not everything will always work. And that’s fine. But you don’t want to keep making the same mistakes over and over again. So the only real mistake is not to learn from your mistakes. This is true for Kaggle competitions and in life.

Are there any particular tools or libraries that you would recommend using for data analysis or machine learning?

I know that we increasingly live in a Python world, but when it comes to tabular wrangling and data visualization I still prefer R and its tidyverse: dplyr, ggplot2, lubridate, etc. The new tidymodels framework is a serious contender to sklearn. Even if you’re a die-hard Python aficionado, it pays off to have a look beyond pandas and friends every once in a while. Different tools often lead to different viewpoints and more creativity. In terms of deep learning, I find Pytorch most intuitive alongside its FastAI interface. And, of course, everyone loves huggingface nowadays; and for very good reasons.

What's the most important thing someone should keep in mind or do when they're entering a competition?

The most important thing is to remember to have fun and to learn something. So much valuable insight and wisdom is being shared both during and after a competition that it would be a shame not to take it in and grow from it. Even if the only thing you care for is winning, you can only accomplish that by learning and experimenting and standing on the shoulders of this community. But there is so much more to Kaggle than the leaderboards, and once you start contributing and giving back to the community you will grow in a much more holistic way. I guarantee it.

# Kaggle Learn courses

A great many things about Kaggle are about acquiring knowledge. Whether it be the things you learn in a competition, datasets you manage to find in the ever-growing repository, or demonstration of a hitherto unknown model class, there is always something new to find out. The newest addition to that collection is the courses gathered under the Kaggle Learn label: <https://www.kaggle.com/learn>. These are micro-courses marketed by Kaggle as “the single fastest way to gain the skills you’ll need to do independent data science projects,” the core unifying theme being a crash course introduction across a variety of topics. Each course is divided into small chapters, followed by coding practice questions. The courses are delivered using Notebooks, where portions of the necessary theory and exposition are intermingled with the bits you are expected to code and implement yourself.

Below, we provide a short overview of the most useful ones:

* **Intro to ML / Intermediate ML**: https://www.kaggle.com/learn/intro-to-machine-learning and https://www.kaggle.com/learn/intermediate-machine-learning

These two courses are best viewed as a two-parter: the first one introduces different classes of models used in machine learning, followed by a discussion of topics common to different models like under/overfitting or model validation. The second one goes deeper into feature engineering, dealing with missing values and handling categorical variables. Useful for people beginning their ML journey.

* **pandas**: <https://www.kaggle.com/learn/pandas>

This course provides a crash-course introduction to one of the most fundamental tools used in modern data science. You first learn how to create, read, and write data, and then move on to data cleaning (indexing, selecting, combining, grouping, and so on). Useful for both beginners (pandas functionality can be overwhelming at times) and practitioners alike (as a refresher/reference).

* **Game AI**:<https://www.kaggle.com/learn/intro-to-game-ai-and-reinforcement-learning>

This course is a great wrap-up of the tech-focused part of the curriculum introduced by Kaggle in the learning modules. You will write a game-playing agent, tinker with its performance, and use the minimax algorithm. This one is probably best viewed as a practice-oriented introduction to reinforcement learning.

* **Machine Learning Explainability**: <https://www.kaggle.com/learn/machine-learning-explainability>

Building models is fun, but in the real world not everybody is a data scientist, so you might find yourself in a position where you need to explain what you have done to others. This is where this mini-course on model explainability comes in: you will learn to assess how relevant your features are with three different methods: permutation importance, SHAP, and partial dependence plots. Extremely useful to anybody working with ML in a commercial setting, where projects live or die on how well the message is conveyed.

**AI Ethics**: <https://www.kaggle.com/learn/intro-to-ai-ethics>

This last course is a very interesting addition to the proposition: it discusses the practical tools to guide the moral design of AI systems. You will learn how to identify the bias in AI models, examine the concept of AI fairness, and find out how to increase transparency by communicating ML model information. Very useful for practitioners, as “responsible AI” is a phrase we will be hearing more and more of.

Apart from the original content created by Kaggle, there are other learning opportunities available on the platform through user-created Notebooks; the reader is encouraged to explore them on their own.

Andrada Olteanu (<https://www.kaggle.com/andradaolteanu>) is one Kaggle Notebooks Grandmaster who very much encourages learning from Notebooks. Andrada is a Z by HP Global Data Science Ambassador, Data Scientist at Endava, and Dev Expert at Weights & Biases. We caught up with Andrada about Notebook competitions, her career, and more.

What's your favourite kind of competition and why? In terms of techniques and solving approaches, what is your speciality on Kaggle?

I would say my specialty on Kaggle leans more towards Data Visualization, as it enables me to combine art and creativity with data.

I would not say I have a favorite type of competition, but I would rather say I like to switch it up occasionally and choose whatever I feel is interesting. The beauty of Kaggle is that one can learn multiple areas of Data Science (computer vision, NLP, exploratory data analysis and statistics, time series, and so on) while also becoming familiar and comfortable with many topics (like sports, the medical field, finance and cryptocurrencies, worldwide events, etc.)

Another great thing is that, for example, if one wants to become more proficient in working with text data, there is almost always a Kaggle Competition that requires NLP. Or, if one wants to learn how to preprocess and model audio files, there are competitions that enable that skill as well.

Tell us about a particularly challenging competition you entered, and what insights you used to tackle the task.

The most challenging “competition” I have ever entered was the “Kaggle Data Science and Machine Learning Annual Survey”. I know this is not a “real” competition – with a leaderboard and heavy-duty machine learning involved – however for me it was one of the competitions I have “sweated” during and learned the most.

This is a Notebook competition, where the users have to become creative in order to win one of the 5 prizes Kaggle puts on the table. I have participated in it 2 years in a row. In the first year (2020), it challenged my more “basic” visualization skills and forced me to think outside the box (I took 3rd place); in the second year (2021), I prepared for it for around 4 months by learning D3, in an attempt to get to a whole another level on my Data Visualization skills (still in review; so far, I have won the “Early Notebook Award” prize). The best insights I can give here are:

• First, do not get lost within the data and try to create graphs that are as accurate as possible; if necessary, build double verification methods to be sure that what you are representing is clear and concise. Nothing is worse than a beautiful graph that showcases inaccurate insights.

• Try to find inspiration around you: from nature, from movies, from your work. You can draw amazing themes and interesting ways to spruce up your visualization.

Has Kaggle helped you in your career? If so, how?

Yes. Tremendously. I believe I owe a big part of where I am now in my career to Kaggle, and for this I am forever grateful. Through Kaggle I have became a Z by HP Ambassador; I have also discovered Weights & Biases, which is an amazing machine learning experiment platform and now I am a proud Dev Expert for them. Last but not least, through this platform I connected with my now Lead Data Scientist at Endava, who recruited me, and I have been working with him since. In short, my position at Endava and the connection I have with 2 huge companies (HP and Weights & Biases) are a direct result of my activity on the Kaggle platform.

I believe the most overlooked aspect of Kaggle is the community. Kaggle has the biggest pool of people, all gathered in one convenient place, from which one could connect, interact, and learn from.

The best way to leverage this is to take, for example, the first 100 people from each Kaggle section (Competitions, Datasets, Notebooks – and if you want, Discussions), and follow on Twitter/LinkedIn everybody that has this information shared on their profile. This way, you can start interacting on a regular basis with these amazing people, who are so rich in insights and knowledge.

What mistakes have you made in competitions in the past?

The biggest mistake I have made in competitions in the past is to not participate in them. I believe this is the biggest, most fundamental mistake beginners make when they enter onto the platform.

Out of fear (and I am talking from personal experience), they believe they are not ready, or they just don’t know how to start. Fortunately, if you follow a simple system, it will become very easy to enter any competition:

• Enter any competition you like or sounds interesting

• Explore the description page and the data

• If you have no idea how to start, no worries! Just enter the “Code” section and look around for notebooks that have a lot of upvotes, or are made by experienced people, like Grandmasters. Start doing a “code along” notebook, where you look at what others have done and “copy” it, researching and trying to improve it yourself. This is, in my opinion, the best way to learn – you never get stuck, and you learn by doing in a specific project.

What's the most important thing someone should keep in mind or do when they're entering a competition?

They should keep in mind that it is OK to fail, as usually it is the best way to learn.

What they should also keep in mind is to always learn from the Competition Grandmasters, because they are usually the ones who share and explain machine learning techniques that one may never think of. The best way of learning something is to look at others that "have already made it," so your road to success will not be as bumpy, but rather much more painless, smooth, and quick. Take 2-3 Grandmasters that you really admire and make them your teachers; study their notebooks, code along, and learn as much as possible.

Do you use other competition platforms? How do they compare to Kaggle?

I have never used any other competition platform – simply because I feel like Kaggle has it all.

# Summary

In this chapter, we have discussed Kaggle Notebooks, multi-purpose, open coding environments that can be used for education and experimentation, as well as for promoting your data science project portfolio. You are now in a position to create your own Notebook, efficiently utilize the available resources, and use the results for competitions or your individual projects.

In the next chapter, we will introduce discussion forums, the primary form of exchanging ideas and opinions on Kaggle.