Capstone Project Guidelines

AI/Machine Learning Career Track

Summary

As you think about your capstone project, keep in mind that it's better to pick a relatively straightforward, "boring" project that you can deliver well than to pick a very complex, shiny idea that'll give you trouble. Here are the things that will make your capstone project a success:

- Focus your project around a realistic client and problem.
- Use your mentor as a resource, a sounding board, and a filter.
- Use the course TAs and the community for feedback throughout the course.

Springboard

TABLE OF CONTENTS

Springboard

Introduction

How to Pick a Good Capstone Project

When you are working as a machine learning engineer in the industry, you have to deliver a "good enough" solution that's ready for production in a limited amount of time. Unlike a typical course project, you don't have the luxury of taking the time to find the optimal or best approach, and you have to ensure that your work is production-ready. It's very important to have a sense of the various tradeoffs between different approaches and pick one that's well-suited to the problem and resources you have.

How do you pick a capstone project that reflects this mindset? Here are some general guidelines:

- **Is this a real problem that someone would care about?**
    - Ideally, your project could be a real application that others can try out or something to add to your portfolio to highlight your skills.
- **Is there a real data set available?**
    - You want to work on a project that has real-world data, not an academic/teaching dataset.
- **Is the data set easy enough to acquire and clean?**
    - Pick a dataset that's relatively clean. As a rule of thumb, if you have to spend more than two weeks cleaning your data, you may want to select another dataset.

To paraphrase Einstein, *keep it as simple as possible, but no simpler.* :-) Your mentor will help you decide if your project idea meets these guidelines.

Some students choose to work with a project topic that leverages specific domain expertise they have from any prior work experience, while others want to showcase their skills in an industry or domain they would like to enter. In both cases, work closely with your mentor to choose projects that are the right balance of challenging and attainable given your current skill set.

We recommend reading through this entire document before starting on your capstone project.

Good luck!!!

Springboard

Project Milestones

We have broken down the capstone project into two phases with several milestones. There are prompts at appropriate points in the curriculum to work on them. Here's a quick overview of what you'll do for the capstone project:

Phase One: Build a working prototype. Estimate Time: 55-74.5 Hours

1. **Step One:** Pick your Initial project ideas. You'll pick up to 3 project ideas to propose to your mentor and the Springboard community.
2. **Step Two:** Write your project proposal. This will help your mentor better understand your idea.
3. **Step Three:** Collect your data. After your mentor approves of your proposal, you'll want to collect the data.
4. **Step Four:** Data wrangling and exploration. You'll apply data wrangling techniques to your project to help you analyze it.
5. **Step Five:** Machine learning or deep learning prototype.
6. **Step Six:** Scale your prototype.

Phase Two: Deploy your prototype to production. Estimate Time: 25-30 Hours

1. **Step One:** Create a deployment architecture
2. **Step Two:** Run your code end-to-end with testing
3. **Step Three:** Deploy your application to production
    a. Implement your data pipeline
    b. Build a web service with an API for your application
    c. Package your application as a container
    d. Test and document your API and application
4. **Extra Credit Step:** Build a web interface to your application

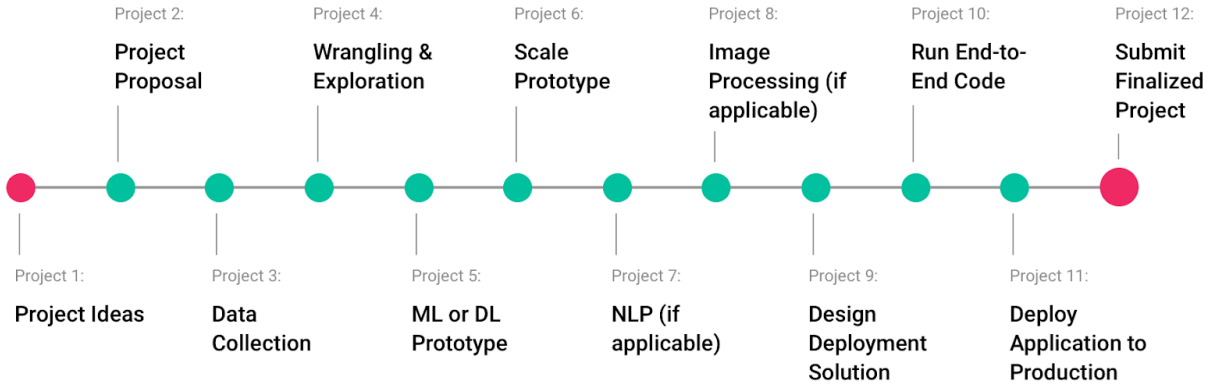
Phase 1:

Step no.	What you will do	Unit(s) no. & name of deliverable (what you will submits)	Estimated time
1	Pick your Initial project ideas. You'll pick up to 3 project ideas to propose to your mentor and the Springboard community.	Unit 2 Overview of Artificial Intelligence & Machine Learning	1-1.5 Hours
2	Write your project proposal. This will help your mentor better understand your chosen capstone project idea.	Unit 3	2-4 Hours
3	Collect your data. After your mentor approves of your capstone project proposal, you'll want to collect the data you need.	Unit 5.1	2-4 Hours
4	Data wrangling and exploration. You'll apply the data wrangling techniques to your project to help you get ready to analyze it.	Unit 5.3 - 5.9	15-20 Hours
5	Protype your ML/DL application. You'll try out various ML/DL techniques to identify the one that's the most viable for your problem.	Units 8, 9 , 11	25-30 Hours
6	Scale your ML/DL prototype. You'll move your prototype out of Jupyter notebooks into a self-contained application and ensure that it works with large data sets.		10-15 Hours
7	IF APPLICABLE, you'll apply advanced NLP techniques to your Capstone Project	Unit 13 Natural Language Processing	10 - 20 Hours
	IF APPLICABLE, you'll apply advanced image processing techniques to your Capstone Project	Unit 14 & 15 Image Processing & Computer Vision	10 - 20 Hours

Springboard

Phase 2:

Step no.	What you will do	Unit(s) no. & name of deliverable (what you will submits)	Estimated time
1	Create a deployment architecture	Unit 17.1 - 17.3	3-5 Hours
2	Run your code end-to-end with testing	Unit 17.4 - 17.5	5-10 Hours
3	Deploy your application to production • Implement your data pipeline • Build a web service with an API for your application • Package your application as a container • Test and document your API and application	Unit 17.6	12-15 Hours



Phase 1: Build a Working Prototype

Step 1. Pick Your Initial Project Ideas

Think of three project ideas that meet the guidelines presented above and that you're excited to work on. Here are a few great sources for large datasets that are appropriate for this course:

- fast.ai research datasets collection

- Google dataset search
- AWS open datasets repository
- Uber Movement
- Yelp dataset

Consider searching Kaggle, one of the most popular sources for datasets. Additionally, Quandl, US Government Open Data, Data is Plural, and UCI Machine Learning Repository are all great places to locate data as long as they meet the requirements below. If you have any questions, reach out to your mentor. Once you have picked datasets, your goal is to narrow them down to the ONE idea that you'll be working on.

Ideally, your dataset should have at least 15K-20K samples **at the bare minimum**. In this course, since we'd like to see you build large-scale applications, we encourage you to go for larger datasets: something that's at least 8GB in size or has at least 1 million samples.

For your initial project ideas, you'll want to:

- Include a short blurb for each of your ideas.
    - The blurb should, at a high level, describe the problem and the data you'll be using to solve it. At this point, there's no need to talk about specific methods and techniques.
- Post your idea (title and blurb) on the community and solicit feedback from both mentors and other students. Pick one idea to work on based on the feedback you get. Discuss the idea with your mentor to ensure that they're on board.

Please note: The goal of a project is NOT to do something novel — it's to demonstrate your competence as a machine learning engineer. It's perfectly acceptable to work on a dataset that's been worked on before and even answer a question that's been answered before, as long as the work is your own.

See an example of the project ideas by one of our alumni, Siri Surab.

Some examples for inspiration

Here are a few ideas that might spark inspiration for your capstone project. Many of these ideas come from natural language processing or computer vision, since they're two of the hottest fields within AI right now, but your idea can definitely be unrelated to natural language processing and computer vision.

- **Inventory tracking and compliance using object recognition:** A company wants to automatically track inventory in its warehouses using a camera with an object

recognition algorithm. You can think of this as a home application: a smart fridge recognizes what's in it based on pictures.

- **Language translation:** Also called machine translation, this technique uses AI to translate one human language to another, whether in text or speech. You can also work between the two formats e.g. speech-to-text transcription or text-to-speech generation.
- **QA systems and chatbots:** Increasingly, companies are using automated chatbots to address their customer service workloads. These bots can produce human-like responses to questions (within limits) and are getting better every day.
- **Text summarization:** Imagine an application that can digest the daily news and produce a coherent summary for a consumer. You can apply summarization to different domains, such as an application that can automatically produce a personalized summary for a student who's trying to research a large amount of material.
- **Fraud/spam detection:** Detect "bad" transactions or items in a dataset. This could take the form of detecting fraudulent credit card transactions, fake news on social media, spam in email, doctored images or video, or abusive behavior on Twitter. Depending on the problem, you can use a variety of techniques, ranging from "traditional" machine learning to the latest in deep learning.

An example of an actual Capstone Project

This is a project by one of our alumni, Siri Surab. She used the Quora Duplicate Questions dataset from Kaggle, and applied NLP techniques from both "old-school" Machine Learning as well as Deep Learning to identify duplicate questions on Quora.

- Blog Post
- GitHub repository

We don't expect you to understand all of these techniques at the beginning of the course, but we've presented this here as an example of what your final project will look like. You'll go much deeper into this specific project in a later unit on NLP.

Step 2. Write Your Project Proposal

Once you've picked your final capstone project idea, you will write a proposal. The project proposal is a short (1-2 page) document that answers the following questions:

1. What is the problem you want to solve? Why is it an interesting problem for anyone?
2. What data are you going to use for this? How will you acquire this data?
3. Briefly outline your approach to solving this problem (knowing that you may not know everything in advance and methods might change later). This might include:
    a. Is this a supervised or unsupervised problem?
        i. If supervised, is it a classification or regression problem?
    b. What are you trying to predict?
    c. What will you use as predictors?
    d. Will you try a more "traditional" machine learning approach, a deep learning one, or both?
4. What will be your final deliverable? This is typically an application deployed as a web service with an API or (for extra credit) a web/mobile app.
5. What computational resources would you need at a minimum to do this project? *You may not have a very clear sense now, but work with your mentor to come to an estimate on this. In real industry applications, you'll often be called upon to provide resource estimates at the beginning of a project.*
    a. Processing power (CPU)
    b. Memory
    c. Specialized hardware such as GPUs

The proposal will be part of a GitHub repository for your project. All code and further documentation you write will be added to this repository.

Once your mentor has approved your proposal, please share the GitHub repository URL on the community and ask for feedback.

Step 3. Collect Your Data

The first step in your capstone project is to collect data. In some cases, it can be as simple as downloading a dataset in a zip file or a tarball. Or, it can require extracting data using a publicly available API or scraping a website. We urge you to work closely with your mentor to ensure that the data collection process is not too challenging. Also, **if your data collection requires you to write code, START EARLY.**

At the end of this step, you'll submit a link to your GitHub repository that contains the following:

1. Code for how you collected the data, if applicable

2. The actual dataset: If your dataset is small enough to fit in a CSV, then include it in the repository. If it's a big dataset or has a lot of binary files (graphics, audio), consider using the Git Large File Storage extension.

Step 4. Clean and Wrangle Your Data

In the course, you'll apply some of the data wrangling techniques you have learned to your capstone dataset. As you're working in your Jupyter notebook, document the steps you undertook to clean your dataset.

Consider the following as you take notes:

- What kind of cleaning steps did you perform?
- How did you deal with missing values, if any?
- Were there outliers, and how did you decide to handle them?
- If your dataset is too large to work with, does it make sense to build your prototype on a smaller subset of the data?

Optional: Explore Your Data

After you have obtained and wrangled your dataset, you will perform preliminary exploration. This exploratory data analysis (EDA) uses a combination of inferential statistics and data visualization to find interesting trends and significant features. For example:

- Are there variables that are particularly significant in terms of explaining the answer to your project question?
- Are there strong correlations between pairs of independent variables or between an independent and a dependent variable?

At the end of this step, you'll submit a link to your Jupyter notebook in your GitHub repository that shows how you cleaned, wrangled, and (optionally) explored the data.

Step 5. Build Your Machine Learning (or Deep Learning) Prototype

The goal of this step is to find a machine learning or deep learning approach that works for your problem, and show that it's a viable one. Since the application has not been deployed to production yet, we'll call it a *prototype*.

You'll build your prototype in a Jupyter notebook. Depending on your problem, you'll be using a more "traditional" machine learning (ML) technique or a deep learning (DL) one. Your goal is to come up with a working implementation in a Jupyter notebook. This prototype could work on a

subset of the data, but demonstrates that your approach to solving the problem is a viable one based on the following criteria:

- The data has been reasonably split into training, validation, and test sets
- You have used the correct metric(s) to evaluate the performance of your algorithm
- The performance of your algorithm is “good enough” as determined by your mentor

At this point, you’ll submit a link to your Jupyter notebook with the ML/DL algorithm coded and your results well-documented in a way that your mentor (or a potential employer) can easily follow.

Step 6. Scale Your Prototype with Large-Scale Data

In this step, your goal is to ensure that your ML/DL approach, which has proved to be viable, can work with large volumes of data. You can work with your mentor to determine what that means for your problem.

Using scikit-learn, SparkML, Keras, TensorFlow, PyTorch or another technology you have learned, implement your prototype at scale.

In case your earlier prototype was working with a subset, ensure that this scaled-up prototype can handle your complete dataset.

Think about what your capstone problem would look like in the real world.

- How much data would you need to handle?
- Can you scale your prototype to handle that volume of data using the approach and tools you have selected?

In a Jupyter notebook, implement the scaled version of your prototype and document what trade-offs and implementation decisions you have to make to scale your algorithm. Submit the GitHub link to this notebook.

Phase 2: Deploy Your Prototype to Production

Congratulations on creating your machine learning solution prototype! As a machine learning engineer, your work does not end here. It’s your job to also deploy this solution to production.

In this section, we’ve outlined a few steps you’re likely to follow as you deploy your application, based on input from MLEs who work in the industry.

Please note: You may not always follow these steps in the order stated, and you're welcome to work with your mentor to figure out the best deployment strategy for your project.

Step 1. Design Your Deployment Solution Architecture

The first step in deploying a prototype is to determine what the deployment would look like in production. What are the various pieces of the deployment, and how do they fit in? Here are some specific questions to think about:

- What are the major components of your system? What are the inputs and outputs?
- Where and how will the data be stored?
- How will data get from one component of the system to another?
- What is the lifecycle of your ML/DL model?
    - How frequently do you need to retrain your model? Is it at fixed intervals when you collect a certain amount of new data or when some other conditions are met?
    - What kind of data do you need for retraining? How will you store and manage it?
    - How do you know if the retrained model is good enough to deploy?
    - How will the retrained model be deployed?
    - How will the retrained model be stored as an artifact?
- How will the system be monitored? How will you debug it if there are problems?
- How will your system respond to unexpected errors or outages?
- What are the specific tools/technologies you'll use to build this system?
- What is the estimated implementation cost in terms of resources, time, and money as applicable?

Submit a link to a document (Google Docs or PDF) containing the following:

1. A sketch or diagram of what your deployment architecture would look like. You're welcome to draw it by hand and take a picture to upload to your document.
2. A 1-2 page write-up explaining your design decisions, covering why you chose to design your system in a certain way and why you have chosen to use certain technologies. Also, include an estimate of the cost of the system (e.g. computing resources, time, money).

Your mentor will review your document and discuss it with you in your next call. Based on the feedback from your mentor, you may need to rework and resubmit this architecture design.

Step 2. Run Your Code End-to-End with Logging and Testing

So far, your code has been running in a Jupyter notebook. However, that's not the way production code works. The very first step in deploying your prototype to production is to ensure

that your code runs independently from the command line and is well-tested. So, now is the time for you to apply all of the best practices that you've learned about software engineering!

1. Create a new repository for your production code by cloning the repository where your prototype was created. We will call this new repository the *production repo*.
2. Refactor the code in the production repo. This might include the following steps as applicable:
    a. Get your code out of your Jupyter notebooks into Python files. You'll edit the Python files in a code editor or an IDE such as PyCharm, Eclipse or Atom. Pick your favorite editor or IDE here, no restrictions! Revisit the pre-work to refresh your memory on setting up IDEs.
    b. Create Python modules, classes, and functions, as appropriate, so your code is well-organized
    c. Remove unnecessary print and debugging statements, along with interactive features such as visualizations
    d. Add useful logging statements at critical points of the code.
3. Add unit tests to test your various functions and methods. Aim for at least 60% of code coverage in your tests. Think about how to cover edge cases or erroneous inputs.
4. Ensure that your code runs end-to-end from the command line with appropriate command-line options
5. Create a README at the top level of your repository documenting exactly how to run your code.
6. Submit a link to your production repo containing the refactored and tested code.

Step 3. Deploy Your Application to Production

Now that you've refactored your code and designed an architecture for your system, it's time to actually implement the architecture and deploy it to production.

The exact steps you follow here depends on you and your mentor, and your specific project. However, in general, there are a few common steps that MLEs might follow:

1. **Step one**: Implement your data pipeline. This is where all of the engineering skills you've learned will come in handy.
    a. Implement the right systems for storing your data, loading it for training your model, and making predictions with your model.
    b. Make sure you log necessary and sufficient information for monitoring and debugging your system.
    c. Test each component and subsystem as much as possible as you build it out. It's much easier to find bugs in smaller systems than larger ones

2. **Step two**: Design an API using a tool such as Swagger. If you choose to use a managed ML service (e.g. Domino, AWS SageMaker, Google Cloud ML), they might provide out-of-the-box API calls. That's completely fine!
    a. As you design your API, try to make sure your API follows good design principles.
    b. Use a tool such as Flask to create a simple user interface for your API
3. **Step three**: Package your application as a Docker container. This makes it easy for you or anyone else to "spin up" your application without worrying about installing the right tools or libraries. Some of the platforms you'll use might provide out-of-the-box support for containers; you're welcome to use that.
4. **Step four**: Test your API using tools such as Postman or Swagger. Make sure you test both read and write functions, and various corner cases.
5. **Step five**: Document your API. Using a tool such as Sphinx lets you automatically create beautiful API documentation from annotations in your code. Add your documentation to your GitHub repository.
6. **Step six**: Add instructions at the very beginning of the README for your GitHub repository on how to access and use your application, and submit a link to your GitHub repository

Final Deliverables

The final deliverables for your project, such as your code and application (please see the project evaluation section below), are part of your portfolio, which means that you'll be sharing them with potential employers. We require students to share their project on Piazza to receive feedback from peers and on GitHub to highlight their skills to potential employers.

When you and your mentor agree on a stopping point for the project, you should have the following deliverables ready *before asking your student advisor* to start the completion process.

Mandatory deliverables

- **Code** for your project, well-documented on GitHub. We've provided some guidelines for what a good GitHub should look like below.
- **An application** that is deployed on a cloud platform. This application must be accessible using an API or a web service. Your GitHub README should contain instructions on how to access and use the application.

Optional deliverables

- **A frontend web interface** for your application. This is optional but can make the visualization of your work more compelling. However, if you don't have any web development skills, we recommend you focus on your core MLE skills and not spend time on this task.

- **A blog post** summarizing your project. We love distributing our students work. It helps students get exposure to potential employers and sets you up as a voice in the community. If you write a blog post about your project, our content team will help you polish it up and share it on a variety of platforms.

  Blog posts are a great way to generate awareness of your work and your brand as a machine learning engineer. Here are Springboard's guidelines for blog posts. If you'd like your blog post to be considered for the Springboard blog, please write a draft according to the guidelines and share it with your student advisor. You're also welcome to post on your own blog, Medium, LinkedIn, or other platforms. Please share the link of your post with your student advisor.

Guidelines for Strong Portfolio

Your portfolio consists of all of your projects, including the code and documentation, usually in your GitHub account. Typically, a hiring manager who looks at your portfolio wants to see evidence of both technical and communication skills. It is your responsibility to ensure that your portfolio is clear and easy to navigate.

Tips to Help Your Portfolio Stand Out

1. Every project should ideally be in a separate repository that is clearly named.
2. For each project:
    a. Have a README page:
        i. The README should provide an executive summary of the project (i.e. summarizes the problem, approach, and results), along with instructions, if needed, on how to run and access the application.
        ii. The README should also include a list of the important files that the reader should view. The files should be clearly named and organized.
    b. Clean up your code and document:
        i. Your approach and methodology are clear to any technical reader. You do not need to document every line of your code, but have comments or text explaining important decision points and why you chose them.
    c. Include any other documents that you have created (e.g. a report or slide deck in the same repository as the code).
        i. Make sure the README points them out to the reader.
3. Ensure that your portfolio is not cluttered with "junk" (i.e. repositories or folders that are incomplete, irrelevant, or undocumented).

Overall, try to put yourself in the shoes of an experienced machine learning engineer or hiring manager who has a limited amount of time (5-7 minutes) to look at your portfolio. How can you ensure that you make it easy for them to get a good idea of your skills and abilities? Your course TA, mentor, and the community should also be able to provide good feedback on your portfolio, so please use them as resources.

Project Evaluation

For Springboard to consider your workshop complete and issue a certificate of completion, your mentor needs to approve your final project submissions based on the rubric listed below. If the project is not approved, please discuss the feedback from your mentor and resubmit in case improvements are necessary. Your student advisor will not be able to process your workshop completion until your project is approved by your mentor!

Capstone Project Rubric

We use the following rubric for evaluating final capstone projects. Please take a good look at it and make sure you discuss with your mentor to agree on success criteria.

View the Capstone Project Rubric here.

Your capstone project is evaluated based on two criteria: completion and process/understanding. Within each criterion, specific benchmarks and expectations are listed to help ensure the overall quality and mentor approval of your capstone project.

Because your mentor approves each step and later the entirety of your capstone, it's vital that you work with your mentor throughout the course to determine and agree what the bar is for each criterion and to incorporate timely feedback during the intermediate stages.

Frequently Asked Questions

How complex does a project need to be?

This goal of this course is to give you the skills to not only design and create an ML/DL application, but also deploy it to production using the latest engineering tools and techniques.

ML/DL contains a wide swath of techniques, and not everything in there can apply to the problem you have chosen. The more complex techniques you use, the more attractive your project will be to employers. But, it's important to balance that with a practical approach of producing a real, scalable application. Some questions you can ask yourself:

- What's the technique that best applies to this problem?
- How easily can this technique be deployed to production?
- Are the results of this technique "good enough" to meet the business requirements of the solution?

In the real world, it **can sometimes be better** to use a simple logistic regression approach that's accurate enough, highly performant and can be deployed quickly as a production application, than a really complex deep learning approach that's difficult to deploy or maintain. You'll have to make similar decisions and tradeoffs about your capstone project. When you're interviewing for your MLE role, you'll be expected to justify these decisions and tradeoffs.

Work with your mentor to determine a problem and approach that meets the requirements of this course and makes you both feel confident that you'll be able to do well.

Is it better to stick to particular areas or applications of ML/DL?

You can pick any problem and dataset you like (with mentor approval) that will let you demonstrate your competency with the entire process of creating ML systems. However, we have a few guidelines that we recommend, based on the material we cover in the course and the skills that are in demand.

1. **Deep Learning:** If you decide to do a project involving Deep Learning, we recommend you stick to a problem in **Computer Vision** (also called image processing) or **Natural Language Processing** (NLP). We cover these two areas deeply in the course, with examples, so you'll be able to use them to guide your project. You're welcome to try other areas or other kinds of data (audio, video, etc.), but you might have to figure out a few things on your own. If you're the kind that welcomes a challenge, go for it!

2. **Natural Language Processing (NLP):** Many of our students are excited about doing NLP projects, since it's an extremely high-demand area of ML/DL application. Many companies have vast amounts of data lying around in text documents and communications that they'd love to get insights into.

   NLP has a long history with many techniques to represent and analyze language. However, in the last few years, DL has revolutionized the field. This was solidified by Google moving all of its machine translation tools (e.g. Google Translate) to deep learning. As a result, we strongly recommend that if you choose to do a capstone project

in NLP, you use deep learning based methods in your project.

3. **"Old-school" Machine Learning:** You're also welcome to build a capstone project using traditional ML techniques (regression, SVMs, random forests) rather than deep learning. However, in order for these projects to meet the bar of this course, we encourage you to really focus on the engineering and deployment aspects of these projects, including making sure your project can work with really large datasets, uses the latest tools/techniques (e.g. SparkML or TensorFlow), and has a strong engineering component.

What kinds of cloud resources are available for capstone projects?

As part of your capstone project, you'll be expected to carry out part of your development and deployment on a cloud platform. At Springboard, we're working on providing you with as many cloud resources as possible, but we recommend that you use them cautiously and begin your project by using free options to prototype your work. This reflects the real world: In an actual job, there will be budget and resource constraints, so it helps to use fewer resources earlier on.

We've written up an entire set of guidelines on the available cloud resources and how to use them. Check them out here.

Can I Use a Dataset from Kaggle

You're welcome to use a dataset from Kaggle in your project! Many Kaggle competitions now provide large, complex datasets, including those for computer vision and NLP. If you'd like to use a large Kaggle dataset, make sure your mentor approves.

Just because you use a dataset from Kaggle, it doesn't mean that you have to solve exactly the same problem that they ask. Here are a couple of ways you could use the dataset differently for your capstone project:

- Is there a different problem than the one asked in the competition that the dataset can be used to solve?
- Could you combine it with other datasets to solve a different problem (or even a similar one)?

Your mentor has the final word on whether a Kaggle competition is appropriate for a capstone project, so please make sure to get their explicit approval for the dataset you'd like to use.

Springboard

Can I use a Private Dataset from my Employer or Another Source?

Many students use proprietary data from their employer to work on their capstone projects, which is perfectly fine. **We don't require that you share the raw data** with anyone. However, there are a few things you'll need to consider:

1. **Ensure you have the right permissions:** Your mentor is here to guide you through your project. They can only do that effectively if they can look at your code, summarized results, and charts, even if they don't have access to the actual data.
    a. Springboard still requires that you turn in a project report and a slide deck based on your analysis and place it publicly on GitHub.
    b. If your employer or the people who are providing you the raw data are not comfortable with these requirements, you may need to rethink your project topic. It's your responsibility to ensure that private data is handled appropriately and securely. Please check with the legal team at your employer to see if you need approval in writing in the form of a legal contract or a Non-Disclosure Agreement (NDA).
2. **Start data collection early:** Even if you have the requisite permissions, please make sure to start the data collection process early and have a realistic idea of how soon you can get the data.
    a. Many companies have elaborate processes around data access and extraction, so sometimes, students have become stuck for weeks or months waiting around for their project data to become available.
    b. Ensure that you follow good privacy and security practices. For example, anonymize the data where appropriate. In some cases, you may be legally required to anonymize it (e.g. healthcare data). Please work with the legal and security teams at your employer to ensure you're always in compliance.

If you have any questions about whether or not you can use proprietary data for your capstone project, feel free to email your student advisor!