

NOEL CHING - github.com/nolll77

Machine Learning Engineer with Microsoft Azure

Azure Python SDK ML Samples

Chapter 1 Welcome to Machine Learning Engineer with Microsoft Azure

Lesson 1 Machine Learning Engineer Program Introduction

1. Welcome!

<https://photos.app.goo.gl/yRBbnUzgsft6yp8Y9>

Our mission is to power careers through tech education. We're honored that you've chosen us to help in your career journey. Whether embarking on a new path. Curious about a new field. Or advancing in your own field. I want you to know, you're in the right place. We are here to help you accomplish your goals.

I can't wait to see where this journey takes you!

2. The Udacity Experience

<https://photos.app.goo.gl/us3PpnidMjSZ4YKH9>

If this is your first Nanodegree program, welcome. If you have taken a Nanodegree program with us before, you already know a little bit of what to expect. Udacity is always improving our learning experience for students. So you can watch this to see what is changed or skip ahead. You can also read this in the FAQ located in the resources tab.

Let's get started.

We truly believe you couldn't have chosen a better place to learn. A lot of people have worked tirelessly to create the course material and platform features, as we all want you to get the most of your learning journey. Our pledge is to provide you with resources and services that will further enable you to succeed in your Nanodegree program. As you go through the lessons, you may find some to be difficult and may feel discouraged. If this happens, we encourage you to take advantage of your community channels and mentor services to help clarify the concepts, or just to share your doubts. Our mentors are there to help you! You will get advice and guidance to ensure you are on track to complete the projects. We WANT you to succeed and graduate with a Udacity Nanodegree certificate.



Machine Learning Engineer for Microsoft Azure

We value the trust you have put in us by making this investment in your future career. We take it seriously and we are here to help you achieve whatever you came here to accomplish.

3. Welcome to Machine Learning Engineer Program

Welcome to the Machine Learning Engineer with Microsoft Azure!

How will you benefit from this Nanodegree program?

There has been a critical shortage of engineers capable of building production machine learning models in the cloud. Strong software engineering skills mixed with a foundational knowledge of both cloud computing and machine learning is an effective template for machine learning engineering.

This Nanodegree is a followup on the partnership with Microsoft, where they have provided the initial content for a free course of Machine Learning with Azure, which is focused on giving an overview of the Azure Machine learning feature set to build and deploy models. While this Nanodegree focuses on the practitioner level of skill set required for students.

What you will learn

The goal of the Machine Learning Engineer Nanodegree program is to equip learners with the ability to build, deploy, and debug production machine learning models. In addition, it teaches critical skills in operationalizing Machine Learning models and building pipelines.

A graduate of this program will become a practitioner in this field with the following skills:

- Use the Azure Machine Learning platform
- Operationalize Machine Learning on Azure
- Correctly select the appropriate Machine Learning service and optimize it

In this program, students will enhance their skills by building and deploying sophisticated ML solutions using popular open source tools and frameworks such as scikit-learn. Using Azure Machine Learning's MLOps capabilities, students will gain experience in understanding their ML models, protecting people and their data, and controlling the end-to-end ML lifecycle at scale.

- Course 1 - students will be able to use Azure Machine Learning Studio to configure workspaces, run experiments, create, and run models.
 - Create an Azure Machine Learning workspace
 - Manage experiment and compute contexts
 - Create and manage datastores and datasets
 - Use Azure Machine learning Designer to create models and training pipelines.
 - Use Azure Machine Learning SDK to run experiments and create pipelines

- Create, run and monitor a pipeline
 - Automate and monitor experiments
 - Use AutoML
 - Manage, monitor, and explain models
-
- Course 2 - students will be able to operationalize a machine learning model, including key activities such as logging, debugging, load-testing, elastic configuration, picking the hardware configuration, and exposing an API.
 - Determine the appropriate cloud architecture for production deployment (i.e. GPU, elastic endpoints, etc)
 - Deploy machine learning model to Azure
 - Debug production issues with the machine learning model
 - Build logging for the machine learning model
 - Apply continuous delivery to the machine learning system with a batch inference pipeline
 - Make recommendations based on load testing outcomes
 - Build APIs for the machine learning model
 - Produce a web service through a pipeline, applying ML techniques, and measuring performance to identify optimizations

What the program offers

Throughout the program, you will work with:

- Interactive lesson content
- Exercises
- Hands-on project where you will apply and practice your new skills.
- Practical experience by using the built-in Azure labs accessible inside the Udacity classroom and run complex machine learning tasks for no additional cost.
- Professional review of your project by our project reviewers who will provide personalized feedback on how you can improve your work.

These projects will serve as a portfolio you can use to demonstrate your skills. After you complete the program, you will be ready for positions like Full Stack Software Engineer, Software Architect, DevOps

Engineer, Sr. Software Engineer, or Machine Learning Engineer or bring these crucial skills to your current role.

We are excited and honored that you are taking this step in your journey with us!

4. Prerequisites

A well-prepared learner will meet the following prerequisites:

- Programming knowledge in Python
- Experience with basic Python programming (e.g., ability to read and write simple Python scripts; understanding of introductory concepts like variables, loops, modules, conditionals, data types, and functions).
- A background in beginning level statistics would also be helpful to understand and deploy the ML models.
- Some experience with fundamental statistics and algebra, including an understanding of data distributions (e.g., normal distribution) measures of central tendency and variability (e.g., mean and standard deviation) and basic linear equations.
- Udacity also recommends basic familiarity with fundamental machine learning concepts (such as feature engineering and supervised vs. unsupervised learning) and classic machine learning algorithms (such as linear regression and k-means clustering).
- An understanding of the basics of Azure and Docker/Container experience
- If you'd like to prepare for this Nanodegree program, check out our [Introduction to Machine Learning and AI Programming with Python](#) courses.

5. Meet your instructors

Meet Your Instructors

We have a great group of instructors for each of the courses of the Nanodegree program. Each of them brings expertise from working in the field and experience teaching others. Below is a short bio of each instructor, and you'll learn more about them during the courses they teach.

Course 1 - Using Azure Machine Learning Platform



Erick Galinkin



Machine Learning Engineer for Microsoft Azure

Principal AI Researcher at Rapid7

Erick Galinkin is a hacker and scientist specializing in Applying Artificial Intelligence to Cybersecurity. He also conducts academic research on machine learning theory and the interplay between algorithmic game theory and information security.



Noah Gift

Founder of Pragmatic AI Labs

 Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

Noah Gift teaches and consults at top universities and companies globally, including the Duke and Northwestern. His areas of expertise are machine learning, MLOps, A.I., Data Science, Machine Learning, and Cloud Architecture. Noah has authored several bestselling books, including Pragmatic AI, Python for DevOps, and Cloud Computing for Data Analysis. You can reach him at noahgift.com

Course 2: Machine Learning Operations



Alfredo Deza

Alfredo Deza is a passionate software engineer, speaker, author, and former Olympic athlete. He has given several lectures around the world about Open Source Software, personal development, and professional sports. Alfredo has written several books about DevOps and Python, and continues to share his knowledge about resilient infrastructure, testing, and robust development practices in courses, books, and presentations around the world.

Capstone - Azure Machine Learning Engineer



Soham Chatterjee

Graduate Student at The Nanyang Technological University

Soham is an Intel® Software Innovator and a former Deep Learning Researcher at Saama Technologies. He is currently a Masters by Research student at NTU, Singapore. His research is on Edge Computing, IoT, and Neuromorphic Hardware.

6. Career chat with your instructors

<https://photos.app.goo.gl/CnN14sWWCLeSj9mq9>

7. How to Succeed

<https://photos.app.goo.gl/8rEcupV6h7Eiyh518>

Lesson 2 Mentor Help, Peer Chat, and Careers

1. What It Takes

Completing a Udacity Nanodegree program takes perseverance and dedication, but the rewards outweigh the challenges. Throughout your Nanodegree program, you will develop and demonstrate specific skills that will serve you for a lifetime. Congratulations on taking the first step towards developing the skills you need to power your career through tech education!

The videos, text lessons, and quizzes you encounter in the classroom are optional but recommended. In order to graduate, you need to pass every project. Projects will test your ability to apply the skills and strategies you have learned in the classroom to real-world problems. They will also provide tangible outputs you can use to demonstrate your skills for current and future employers.

Projects are designed to be challenging. Many students initially struggle, but with a little grit they are able to learn from their mistakes and build their skills. Data from nearly 100,000 Udacity graduates show that commitment and persistence are the highest predictors of whether or not a student will graduate.

At some point, nearly every student will get stuck on a new concept or skill, and doubt may set in. Don't panic. Don't quit. Be patient, and work through the problem. Remember that you are not alone and the problem that you are encountering is likely one that many others have experienced as well. Whether you are stuck or simply looking for encouragement, you'll find Udacity Mentors and students there to help.

2. Getting Help

Getting Help

As questions come up during your Nanodegree program, click on the Help button on the left sidebar of your classroom. You'll see four options, each for a separate type of support:

- Technical Mentor Help: is available on Knowledge, our expert Q&A platform, by clicking on "Mentor Help". You can search for answers to questions similar to yours or post new questions related to your project or lessons. Udacity's expert technical mentors answer all new questions.

- **Peer Chat:** Join [study lounge](#), your "virtual coffeehouse" where every Udacity student can hang — Drop in to say hello and connect with others to share insights on your Nanodegree program.
- **Career Support:** is available for all students and alumni including expert feedback on your application materials profiles, and lessons on "How to Network Successfully" to "Rehearsing for Interviews" and more. Click on "Career Services" to get started.
- **Udacity Support Community:** Receive peer support and find answers to your non-technical questions quickly through [Udacity Support Community](#). Receive peer support from our global community of lifelong learners that help each other succeed by sharing their experience and expertise. Start a discussion [here](#).
- **General Account Help:** This is where you can get details on non-technical issues such as 3rd party tools, billing, deadlines, and more. You can even find additional help here via. live chat. Simply click on Account Help or visit our [Help Center](#) to find answers. .

3. Mentor Help

Mentor Help

Technical Mentor Support: is provided through Knowledge, our expert Q&A platform. You can search for answers to questions similar to yours or post new questions related to your project or lessons. Udacity's expert technical mentors answer all-new questions.

How to Use Knowledge

To ensure you're getting the quality and prompt support you need, it's helpful to follow these best practices and guidelines for Knowledge.

The screenshot shows the Udacity Knowledge platform interface. At the top right, there is a button labeled "Ask a new question". Below it, a blue header bar contains the word "Knowledge" and two links: "ASK A MENTOR" and "ACTIVITY". On the left side, there is a search bar with the placeholder "Search for your question..." and a "FILTER" section. The "FILTER" section includes dropdown menus for "Program" (set to "Business Analytics N...") and "Project" (set to "Analyze NYSE Data"), along with a checkbox for "Unanswered". To the left of the search bar, there is a callout text "Search for related questions" with an arrow pointing to the search bar. To the left of the "FILTER" section, there is another callout text "Filter for project-related questions" with an arrow pointing to the "Project" dropdown menu.

- **Search for Similar Existing Questions:** with tens of thousands of students enrolled in Nanodegree programs, many of your questions may have already been asked and answered. To look for existing answers to similar questions that may provide the immediate support you need, use the filter on the left side of your screen to select your Nanodegree and related project or write in the key terms related to your question in the search bar at the top of the page.
- **Ask a New Question:** if you want to ask a new question, select “Ask a Mentor” at the top of the page. Kindly remember that Knowledge is for technical questions only; for other types of support and feedback, please write to support at support@udacity.com.

When you ask a new question, the platform immediately assigns it to one of our expert mentors spread across the globe to ensure prompt replies. Of note, when a mentor answers your question, you will see “Mentor” next to their name to differentiate their support from comments your fellow learners may also provide.

Ask a Technical Mentor

Question

What are the differences between apples and oranges?

Character limit: 0/150

Details

B I | ⚡ ‘’ „ „ | ☰ ☱ ☲ ☳ | ☴ ☵ ☶ ☷

Please include all information that a technical mentor would need to answer your question, such as a code snippet and Github link. If you are encountering an error, please also include the output of the error message.

Github or Project Link (Optional)

<https://>

Program

Select...

Project ⓘ

Select...

Select your Nanodegree first.

SUBMIT QUESTION

If you don't see your question, simply create a new post. You are likely to get an answer within 24 hours and you'll be helping future students who may encounter the same problem.

How to Ask a Good Question?

Students that follow these tips typically receive the strongest initial support and avoid back-and-forth with mentors:

- **Ask Specific Questions:** if you have closely related questions that form part of a general theme, it's okay to ask them all together. But consider using bullet points to separately list each of the questions in your post. Keeping your questions organized helps ensure mentors



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

provide clear answers to each specific question. If your questions are less closely connected, it's best to submit new, separate questions for each one.

- **Provide Details and Links:** explaining what (if anything) you've already attempted to solve the problem helps mentors know where to start when answering your question. Likewise, if your question is not related specifically to a project, but rather to an exercise, quiz, or lesson, it's helpful to include information such as lesson or quiz name, screenshots, and classroom links.

Overall the key to asking a good question is to imagine yourself trying to answer your own question. Imagine you were coming to it without any prior knowledge. Does the question make complete sense? Or are there gaps around the context?

- **Start with a Clear Question Title:** attempt to summarize your entire question in one sentence. You may even write the title at the end, just before posting the question. This will help you summarize the issue before you include details in the question itself.
- **Share Code Correctly:** by using the “Code Block” option to properly format your code. If your question concerns a piece of external code, include a link to the file on Github.

In fact, Github lets you create a link to a specific line in a file. To do so, just click to the left of the line number, and then select copy permalink in the ellipsis that appears in the margin. Paste the permalink right into the Github box on your question submission form.

- **Figuring out Errors:** if your question is about an error message or stack trace, include the entire error message, by either formatting the error message using the “Code Block” option or creating a Gist or a Paste on Pastebin, and including a link to it in the description.
- **Avoid Screenshots of Code or Error Messages:** do not use screenshots of code or error messages. They are hard to read and the text cannot be copied to debug it.

If you receive a helpful answer, kindly select it as the “accepted answer.” For questions from other students, if you see other helpful answers kindly select the “upvote” option. Conversely, if you don't think an answer strongly answers a question, select the “downvote” option.

If a response fully answers your question, click this to indicate your question is resolved.

1 Answer

Set as Accepted Answer

Upvote for good answers; downvote for unhelpful answers

Getting Additional Support

At times, students want support from a different mentor. As everyone learns differently, we want to make this a simple process for learners like you.

If you receive an answer that you are not satisfied with and want a different mentor to chime in, kindly select the option at the top of the page in Knowledge. If you reply directly in the comments section, without clicking on the link at the top of the page, your question will not be answered by another mentor.

The screenshot shows the 'Knowledge' page with a blue header bar containing 'Knowledge', a search bar with 'Search for your question...', and a 'ASK A MENTOR' button. Below the header, a message box displays: 'A technical mentor answered your question. If you need more help, [request a new answer from another mentor](#)'. An arrow points from the text 'Select this option at the top of the page to get further support from a different mentor' to the 'ASK A MENTOR' button.

4. Submitting Classroom Feedback

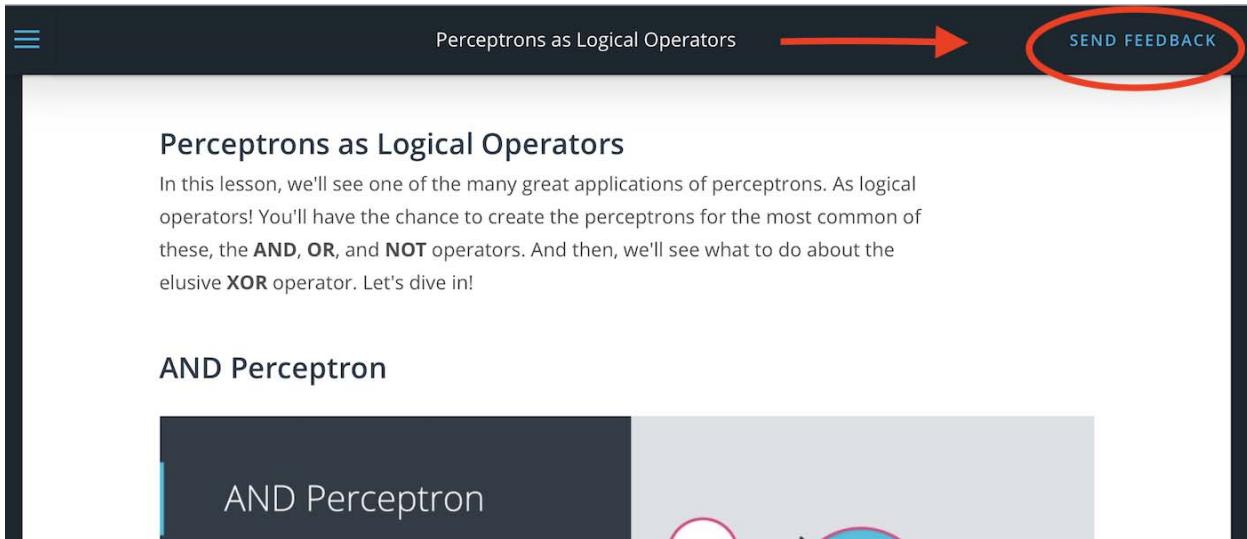
If you would like to share feedback on your learning experience or need to report any errors in the classroom content, click the 'Submit Feedback' button on the upper-right corner of the classroom. This feature is available in all of your lessons and allows you to provide comments on specific sections of the lesson you're on.

NOTE: When you report feedback through the classroom, you won't receive a direct response, but don't worry! Your feedback will be documented and shared with the appropriate team for review. We

strive to create the best learning experience for each unique individual and we'll take your feedback into account as we continue to improve the program!

How Classroom Feedback Works

Click the 'Submit Feedback' button on the upper-right corner of the classroom:



This will highlight different sections on the current concept that you're in. Click 'Select' on the section you would like to provide feedback on:

☰ Perceptrons as Logical Operators CANCEL

Perceptrons as Logical Operators

In this lesson, we'll see one of the many great applications of perceptrons. As logical operators! You'll have the chance to create the perceptrons for the most common of these, the **AND**, **OR**, and **NOT** operators. And **SELECT** see what to do about the elusive **XOR** operator. Let's dive in!

SELECT

AND Perceptron

AND Perceptron

IN IN OUT

| | | |
|---|---|---|
| ✓ | ✓ | ✓ |
| ✓ | ✗ | ✗ |
| ✗ | ✓ | ✗ |
| ✗ | ✗ | ✗ |

IN IN OUT

| | | |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |

Once you select a section, you will see a feedback box appear on the bottom-right corner of the classroom where you can fill in additional information that you would like to share with us. Once you click 'Send,' you will see a confirmation message that your feedback has been sent and received.

AND

| IN | IN | OUT |
|----|----|-----|
| ✓ | ✓ | ✓ |
| ✓ | ✗ | ✗ |
| ✗ | ✓ | ✗ |
| ✗ | ✗ | ✗ |

Send feedback

What's your feedback?

Select a category

- Content contains outdated information
- Content is confusing
- Content needs more detail
- Error in content
- Other

What are the weights and bias for the AND p

Set the weights (weight1, weight2) and bias [bias] to

SEND



What are the weights and bias for the AND gate?

Set the weights (weight1, weight2) and bias [bias] to calculate AND operation as shown above.

Feedback sent, thank you!

And that's it! As a reminder, you won't receive a direct response for classroom feedback submission, but please know that your comments have been shared with the appropriate team for review.

5. Help Center

Get help with your account, browse categories of FAQs, and more!

Get Account help

Read detailed help articles on account-related topics like billing, third-party tools, project deadlines, graduation, and more.

Choose the type of help you require

For your specific questions, look through categories of Frequently Asked Questions and find answers quickly.

Get in touch with us

Reach out to our Support team directly via [Live Chat](#). Click on Account Help in the Classroom or visit our [Help Center](#) to get started.

6. Udacity Support Community



Welcome to Udacity's Support Community!

Become a part of our global community of lifelong learners that help each other succeed by sharing their experience and expertise. Receive peer support and find answers to your non-technical questions quickly through Udacity Support Community.

Find answers to your questions

Search Udacity Support Community for answers to commonly asked questions.

Receive peer support

Ask the community by posting a question and following discussions.

Share your expertise

Get recognized for sharing your insights, experiences, and tips with fellow learners.

Checkout status of your Support tickets

Manage and streamline your view of the support tickets, community conversations, and people you follow in the 'My Profile' section.

Solved, helpful, and recommended posts

Let other learners know which posts solved your problem, highlight helpful responses, and get the answers you want faster with recommended posts.

Build your reputation in the community

Explore and participate in Support Community programs to earn points, levels, and badges. Have fun and unlock rewards along the way.

Join the [Support Community](#) now

7. Career Services

You're probably in this Nanodegree program to advance your career - whether looking for a new job, learning new skills for your current job, or just exploring.

If you want career support during your program, career coaches provide personalized feedback on your LinkedIn profile and GitHub as optional projects.

If you are actively looking for a new role, you can also submit your resume and cover letter in the sidebar under Help > Career Services.

You'll get honest, concise feedback from career coaches, including industry professionals.

The Udacity Careers team is here to help you figure out what you're looking for, plan for it, and work with you to move forward in your career. You can always contact us at career-support@udacity.com.

8. Plagiarism

Plagiarism and Academic Dishonesty

Overview of Plagiarism at Udacity

Defining Plagiarism

Any act claiming or implying another person's work as your own.

Examples:

- Submitting a project you didn't create to copying code into a program without citation.
- Any action in which you misleadingly claim an idea or piece of work as your own, when it is not.

Copying and Combining Code

Using another person's work in your own work

Examples:

- Following a guide that someone may have created for completing a Udacity project, whether from a video or website.

- Taking a part of someone else's project and changing some variable and function names, - regardless of whether you give the source credit or not.
- If you have not done the work yourself and attempt to mask it.

Using Someone Else's Code to Inform your Solution

You should never knowingly view someone else's work until you have completed the project yourself - nor should you share your project with someone who has not yet completed theirs. However, once you have completed your project, you are encouraged to see how other people have approached the same challenge in a different way. This allows you to compare your different strategies and ways of thinking.

Submitting Identical Works Post Collaboration

If you decide to work together with another student on a project, you are then expected to write your code separately. Submitting identical projects or submissions with identical portions is considered plagiarism as described in the Udacity [Honor Code](#).

Seeking Help without violating the Honor Code

To be clear, you are encouraged to seek help by talking to Mentors on Knowledge, other students, and alumni. You're also welcome to use frameworks and libraries to assist you, as long as they aren't removing important goals from the project that you should know how to do yourself.

If you are struggling and need help, we encourage you to ask technical questions on Knowledge to explore why your approach is not the correct approach. Mentors typically provide answers within an hour of questions being posted, so you can expect quick guidance on your question.

If you're ever in doubt as to whether or not something would be considered plagiarism, ask a Mentor on Knowledge. Our goal is for every student to graduate armed with a toolbox of skills that they can apply across a variety of concepts and problems,. The best way to achieve this is to ensure every student does their own work and is able to demonstrate their skills in each project submission.

9. Quiz: Plagiarism at Udacity

QUESTION 1 OF 2

Select the statements that should be classified as plagiarism:

Two or more students work on a project together and end up with the identical code submission, or significant portions of their submissions show duplicated code.

Two or more students who discuss a project together to get a general idea of implementation, then separate to each write their own code individually.

Copying code or using code that has been provided for you and approved for use in your project by Udacity without attribution.

Using or adapting a code from someone else's project and then properly attributing the source code.

Using or adapting and then properly attributing (give date and URL) a small piece of helper code. The helper code must not be directly relevant to the concepts being assessed in your assignment.

QUESTION 2 OF 2

Select the statements that represent the correct choice when you need help on your projects:

Seek help from mentors by asking queries on knowledge

Look at examples of completed projects on the internet

Ask someone else to write the code where you are blocked

Chapter 2 Using Azure Machine Learning

Lesson 1 Introduction to Azure ML

1. Meet Your Instructor

This course is taught by Noah Gift. In this next video, you can meet Noah and learn a little about his background.

<https://photos.app.goo.gl/PxTz8Eo7z4rYKpMk7>

Noah Gift is the founder of [Pragmatic A.I. Labs](#). He teaches machine learning, MLOps, A.I., and Data Science courses at MSDS, Northwestern University, the Duke MIDS Graduate Data Science Program, the Graduate Data Science program at UC Berkeley, the UC Davis Graduate School of Management MSBA program, and the UNC Charlotte Data Science Initiative. He is also the author of several books, including:

- [Pragmatic A.I.: An introduction to Cloud-Based Machine Learning \(Pearson, 2018\)](#)
- [Python for DevOps \(O'Reilly, 2020\).](#)
- [Cloud Computing for Data Analysis, 2020](#)
- [Testing in Python, 2020](#)
- [Python Command Line Tools, 2020](#)
- [Minimal Python, 2020](#)

2. Course Overview

Course Overview

<https://photos.app.goo.gl/Bjuf1aNQpFfxfc6W7>

Course Outline

Below is an outline showing the structure of the course, for your reference.

Lesson 1: Introduction to Azure Machine Learning

- **What You'll Build.** We'll have a look at a preview of what you'll build on your final project.

- **Why Do ML in the Cloud?** We'll go over some key reasons why you would want to do Machine Learning (ML) in the cloud. We'll look into some of the limitations of performing ML locally on your machine, as well as how the cloud addresses these limitations.
- **When to Do ML in the Cloud.** Most of the time the cloud is your best option for ML. But we'll discuss a few edge cases in which using the cloud may not be the best route.
- **Customers of ML.** We'll look at the different customers of machine learning, including both internal and external customers.

Lesson 2: Workspaces and AzureML Studio

- **The Azure ML Platform.** We'll talk about the core features of the Microsoft Azure ML platform and how they enable you to be more productive as a data scientist or machine learning engineer.
- **Workspaces and Notebooks.** Workspaces and notebooks are critical components of Microsoft Azure. We'll learn how these tools enable you to be a more effective data scientist—including the use of Jupyter to build and deploy machine learning models.

Lesson 3: Datastores and Datasets

- **Datastores and Datasets.** Datastores and Datasets are a critical component of cloud computing. We'll learn how Azure allows you to easily integrate third party datasets and open datasets into our ML pipeline to quickly develop working solutions.

Lesson 4: Training models in Azure

- **Managing pipelines.** We'll cover how to create a pipeline that you manage yourself using the console, which will allow you to make very small changes that can be run repeatedly.
- **Hyperparameters in experiments.** We'll learn how to use hyperparameters in experiments, including how we can automate the creation of hyperparameters and make very small changes that create huge value in terms of prediction accuracy.

Lesson 5: Azure ML SDK

- **Creating pipelines.** We'll talk about how the Azure ML SDK allows us to programmatically create pipelines. By automating the creation of pipelines, we can create more pipelines; also, our pipeline creation becomes a repeatable process that other members of our team can follow.
- **Managing experiments.** We'll learn how to use the SDK to manage experiments programmatically with Python, allowing us to easily rerun our processes using Python scripts.

Lesson 6: Automated Machine Learning and Hyperparameter Tuning

- **Automated ML and SDK.** We'll discuss how we can use the Azure ML SDK to do Automated ML and create models more quickly and accurately.

- **Model Interpretation.** We'll explore how model interpretation allows us to build solutions using Automated ML that we (and others) can more easily interpret, making visible the core features that are driving the model.

Project: Creating and Optimizing an ML Pipeline

In the project at the end of this course, you'll have the opportunity to create and optimize an ML pipeline. You'll do this both using HyperDrive and also Automated ML, so that you can compare the results of the two methods.

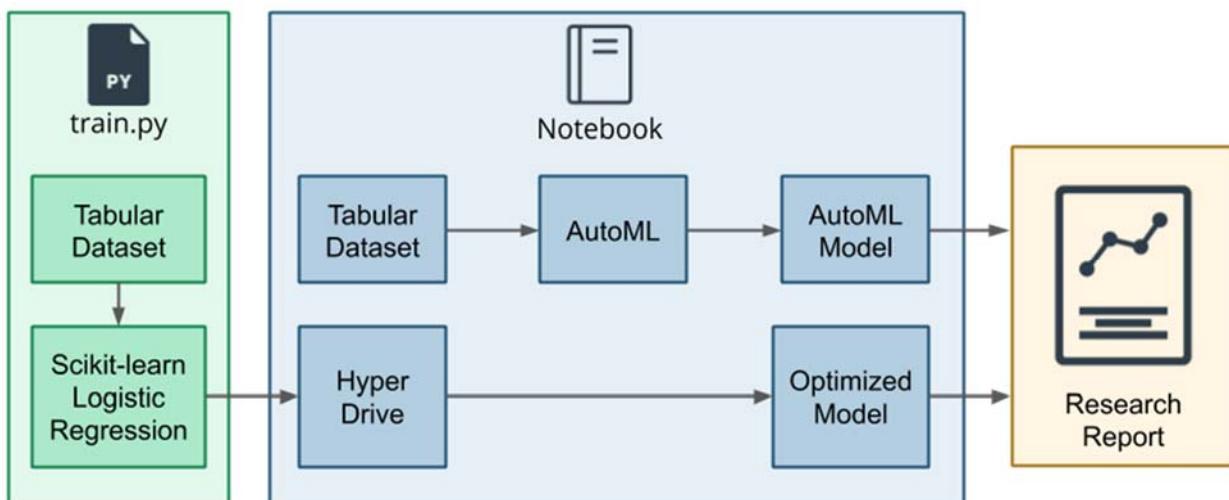
3. What You'll Build

What You'll Build

<https://photos.app.goo.gl/Xef4RmyG8ZFwapSAA>

In the project at the end of this course, you'll have the opportunity to create and optimize an ML pipeline. First, you'll be provided a custom-coded model—a standard Scikit-learn Logistic Regression—the hyperparameters of which you will optimize using HyperDrive. Then, you'll find an optimal model for the same dataset using automated machine learning. This will allow you to compare the results of the two methods in a final research report.

You can see the main steps that you'll be taking in the diagram below:



Additional Resources

- [Microsoft Documentation: What is automated machine learning?](#)

4. Why Do Machine Learning in the Cloud?

Why Do Machine Learning in the Cloud?

<https://photos.app.goo.gl/DrHdhbtzVm57xD9K9>

Although you can certainly do machine learning locally on your own machine, but this will involve limitations imposed by your machine's resources. In contrast, the cloud provides a variety of platform tools that are constantly being developed and improved by some of the best engineers in the world. In the Azure cloud, these include:

- *Data lakes*, which provide essentially infinite storage that is also very secure.
- *Machine Learning Studio*, which makes it easy to build a full, end-to-end ML solution, leveraging great ready-made resources like *open datasets*.
- *DevOps build systems*, allowing you to integrate DevOps pipelines with your machine learning workflows.
- *Serverless technology*, in which Azure runs the server and handles the provisioning of machine resources for you automatically, allowing you to more easily solve complex orchestration problems.

QUIZ QUESTION

QUIZ QUESTION

A critical component of cloud native computing is the **ability to automatically scale infrastructure resources**—such as storage, compute, or disk I/O—up and down in response to the demands of the job.

What term is commonly used to refer to this ability?

Plasticity

Dynamic responsiveness

Proportional responsiveness

Elasticity

Additional Resources

- If serverless technology interests you, you may want to check out [Azure Functions](#).
- For a more detailed definition and explanation of elastic computing, you may want to read the article, [What is elastic computing or cloud elasticity?](#)

5. When to Do ML in the Cloud

When to Do ML in the Cloud

<https://photos.app.goo.gl/89k31EdsAJ655Avj7>

Most of the time the cloud is your best option for ML. Some reasons to favor cloud computing include:

- Data governance. Cloud solutions offer additional security to help prevent problems like data breaches and piracy.
- Better ability to leverage new services. A cloud-native approach allows you to leverage new services as they're developed. You're *betting on the future*, knowing that the cloud platform will get a constant set of new features (in contrast to you needing to build those features yourself).

Even so, there are a few edge cases in which using the cloud may not be the best route. These include:

- Legal/government requirements. If your company handles sensitive data, such as medical records or government files, it may not be legally possible to move to the cloud.
- Legacy applications. If you have a legacy application that is currently working, but that is unlikely to be updated, it may not make sense to take the risk of moving it to the cloud.
- Specialized HPC. In some cases, you may already have a specialized and very expensive High Performance Computing (HPC) cluster. In such a case, it may be more cost effective to keep what you have and only use the cloud for some parts of your workflow.

Additional Resources

- [Microsoft Documentation: Comparison between Azure data storage and on-premises storage](#)

6. Customers of Machine Learning

<https://photos.app.goo.gl/mRttK2yWRcTPSGf99>

QUIZ QUESTION

QUIZ QUESTION

As an ML engineer, you will have many stakeholders and customers of the pipelines you produce.

Which of the following could be one of your customers or stakeholders as an ML engineer?

- An executive at your company, who wants to ensure that your ML solutions translate to good ROI (Return On Investment) for the company.
- Paying users, who generate revenue for your company.
- Your teammates, who may need internal products or metrics for data-driven solutions.
- Vulnerable groups, who may be harmed by your models (e.g., your model may discriminate against certain groups).

Additional Resources

If you'd like to read more about bias and discrimination, you may want to check out:

- The blog post, [Tackling Bias in Machine Learning](#), by Suraj Acharya, which includes a description of several techniques that can be used to mitigate bias.
- The paper, [A Survey on Bias and Fairness in Machine Learning](#), which includes real-world examples of algorithmic discrimination, as well as an extensive list of different types of bias that can occur.
- How can your company [perform a digital transformation?](#)

7. Lesson Review

Lesson Review

<https://photos.app.goo.gl/ErU8L5HANNRxxb7e7>

Here are the topics we covered in this lesson:

- What You'll Build. We had a look at a preview of what you'll build on your final project.

- **Why Do ML in the Cloud?** We went over some key reasons why you would want to do Machine Learning (ML) in the cloud. We looked into some of the limitations of performing ML locally on your machine, as well as how the cloud addresses these limitations.
- **When to Do ML in the Cloud.** Most of the time the cloud is your best option for ML. But we discussed a few edge cases in which using the cloud may not be the best route.
- **Customers of ML.** We looked at the different customers of machine learning, including both internal and external customers.

Lesson 2 Workspaces and the AzureML Studio

1. Lesson Overview

<https://photos.app.goo.gl/CaLuBi4DHKkXAidy6>

Lesson Outline: Workspaces and AzureML Studio

Here's what we'll be covering in this lesson:

- **The Azure ML Platform.** We'll talk about the core features of the Microsoft Azure ML platform and how they enable us to be more productive as data scientists or machine learning engineers—including how elastic resources give us the power to do ML at scale.
- **Managing and choosing compute resources.** We'll explore the role of compute instances, compute clusters, inference clusters, and attached compute. We'll also consider the tradeoffs involved in choosing resources, such as GPU vs CPU, VM size, cluster size, and dedicated vs low-priority instances.
- **Workspaces and Notebooks.** Workspaces and notebooks are critical components of Microsoft Azure. We'll learn how these tools enable you to be a more effective data scientist—including the use of Jupyter to build and deploy machine learning models.

2. Intro to Azure ML Platform

<https://photos.app.goo.gl/qFMTQfiu4UiMMihY8>

The Azure ML platform is composed of two main components: The *Azure Cloud* and *Azure ML Studio*.

Key features of Azure Cloud include:

- Scalable, on-demand compute instances.
- Data storage and connectivity.

Key features of Azure ML studio include:



- The ability to easily orchestrate machine learning workflows.
- Model registration and management (or [MLOps](#)). This allows you to register models or track them and control the [lineage](#).
- Metrics and monitoring, giving you the ability to look at the compute, the storage, or a training job, and make sure that it's running effectively.
- Model deployment; you can create an inference end point and deploy your model so that it runs 24/7 and is elastically scaled to meet demand.

QUESTION 1 OF 2

Which of the following components of Azure ML Studio provides a drag-and-drop, low-code/no-code way to create machine learning models?

Notebooks

Azure ML Designer

ML Control Center

Data labeling

Azure ML Controller



Microsoft Azure
Noel Ching

QUESTION 2 OF 2

Below are some of the components provided by the Azure platform. Can you match each one with either the **Azure Cloud** or **Azure ML Studio**?

| Azure Cloud | COMPONENT | CLOUD OR ML STUDIO? |
|-------------|-----------------------------------|---------------------|
| | Model registration and management | Azure ML Studio |
| | Scalable, on-demand compute | Azure Cloud |
| | Data storage and connectivity | Azure Cloud |
| | ML workflow orchestration | Azure ML Studio |
| | Model deployment | Azure ML Studio |

Additional Resources

- If you'd like to read through a deeper overview, feel free to check out [Microsoft's official documentation on the Azure ML Platform](#).

3. Accessing Azure Machine Learning

Accessing Azure Machine Learning

On the next page, we will start the first hands-on exercise for this course. You will find many similar exercises throughout this Nanodegree. For all of these exercises, you will need access to [Azure Machine Learning](#). If you have (or want to purchase) your own Azure subscription, you are welcome to use it to complete the exercises—however, please be aware that using your own account can involve a significant cost.

For that reason, we've provided access to an Azure account through virtual machines (VMs) that you can access right here in the Udacity classroom. You'll find one on the next page. The layout looks like this, with the instructions on the left and the VM on the right:

Instructions

Virtual Machine

Exercise: Creating Notebooks

New Compute Instance

Compute name:

Region:

Virtual machine type:

Storage:

Access Lab

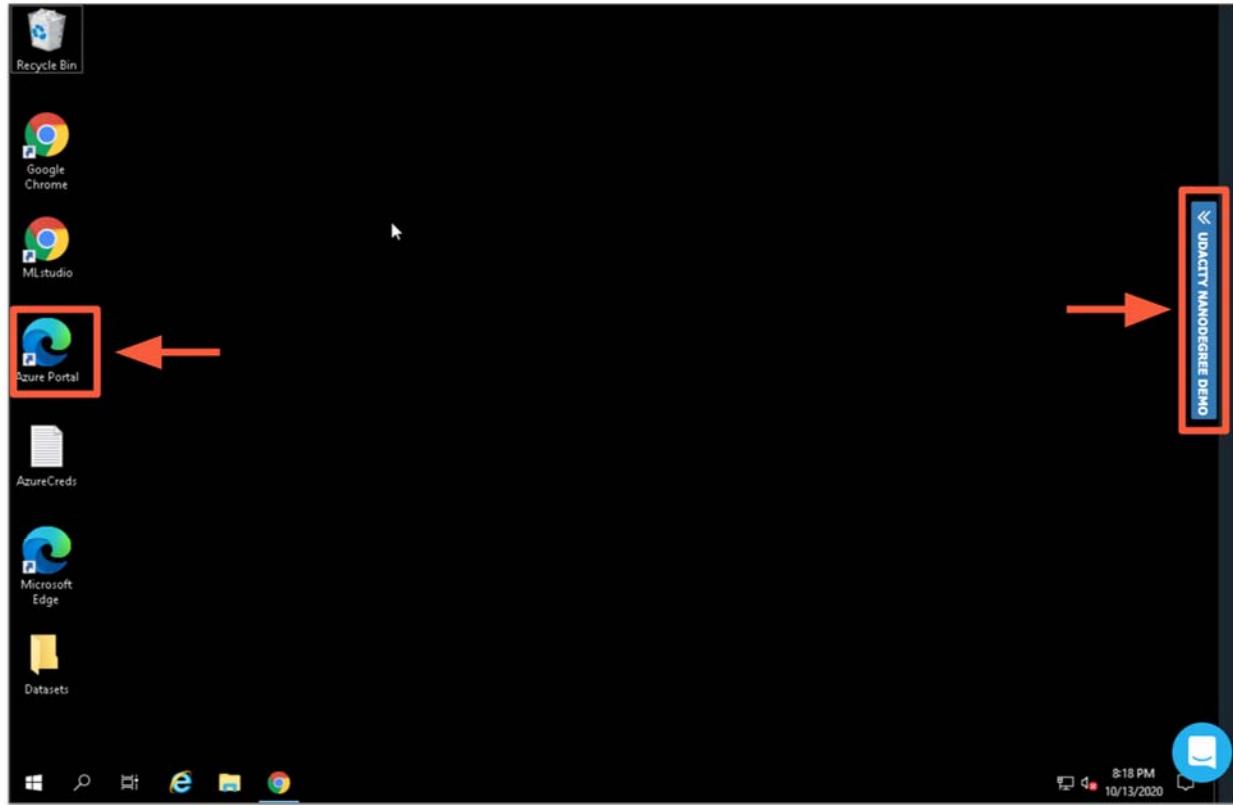
If you are having issues with the lab, please contact support at udacity-labsupport@udacity.com.

We'll be using Jupyter Notebooks multiple times in this course, so make sure you've followed along and can create one in Azure ML Studio:

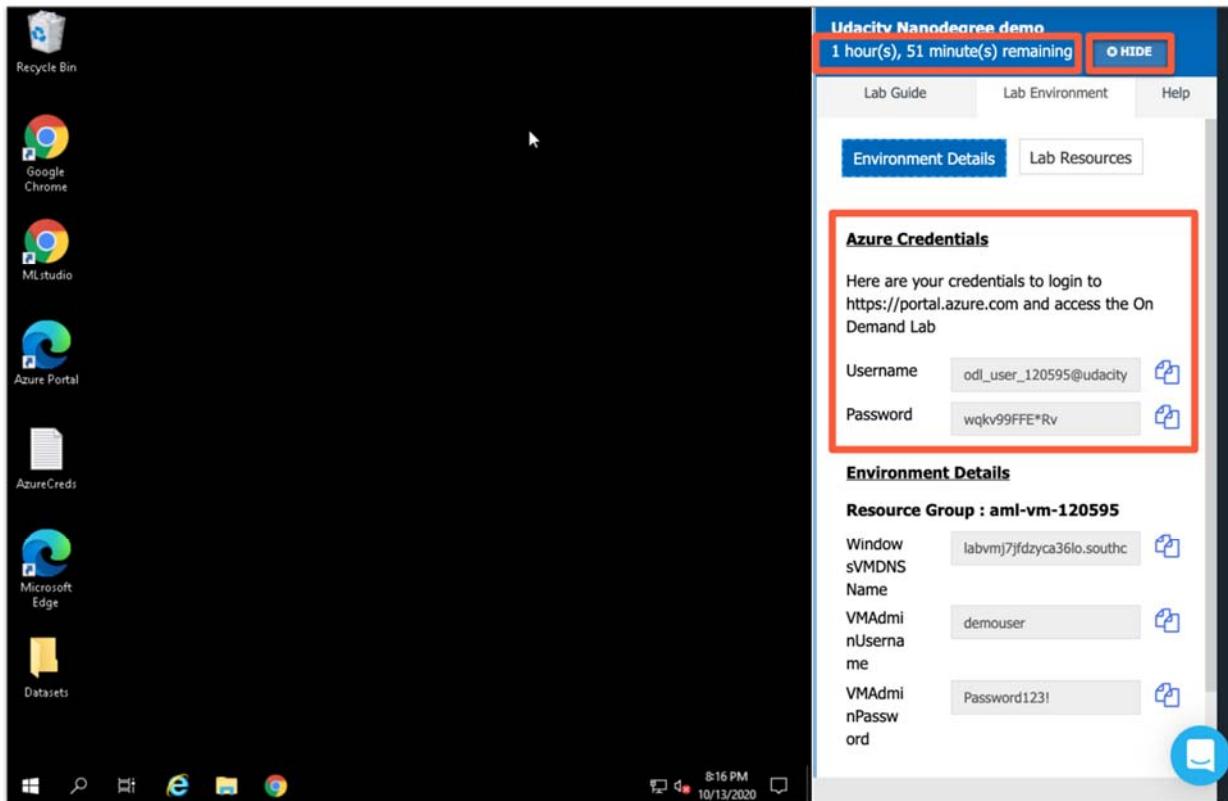
- Create a new compute instance
- Name the notebook
- Select between CPU or GPU
- Open the notebook interface

ACCESS LAB

To open the VM, you'll simply click the ACCESS LAB button. In general, we have programmed the VMs to complete as much setup as possible for you, so that you can dive directly into the exercise once it starts. However, please be aware that this sometimes means it will take a while to load (usually around 5–10 minutes), since it is setting up all of the resources and configurations you need. Once the VM loads, it will look like this:



On the VM desktop, you'll find a shortcut to the Azure Portal. You can access your login credentials by clicking on the blue tab on the right.



You'll also see that this indicates the time you have remaining to access the VM. Because running the VM and Azure ML requires significant resources, each exercise has a limited time for completion—but if you run out of time, you can extend your VM session by an additional 15 minutes (and you can extend your session 4 times). Please note that there is also a one hour timeout—so if you walk away from the computer and are not accessing the VM at all for over an hour, it will quit automatically.

Browser Type and Cookies

Generally the best browser to use for these exercises is [Chrome](#). If you are using a different browser and experience any issues, please try switching to Chrome to see if this resolves the problem.

Also, if you have not already, be sure to [enable cookies](#) in your browser. In some cases, failing to enable cookies will result in the error shown below.



Getting Help

If you have a technical problem while using your VM, you can send an email to Udacity Support at udacity-labsupport@udacity.com. Please be sure to send your message from the email you used to enroll in the Nanodegree program.

Please copy the below text into your email and fill out the following information:

- **Subject Line:** Error in Lab name

- **Timestamp and Timezone:**
- **Brief description of what you were doing when the issue occurred:**
- **Screenshots:**

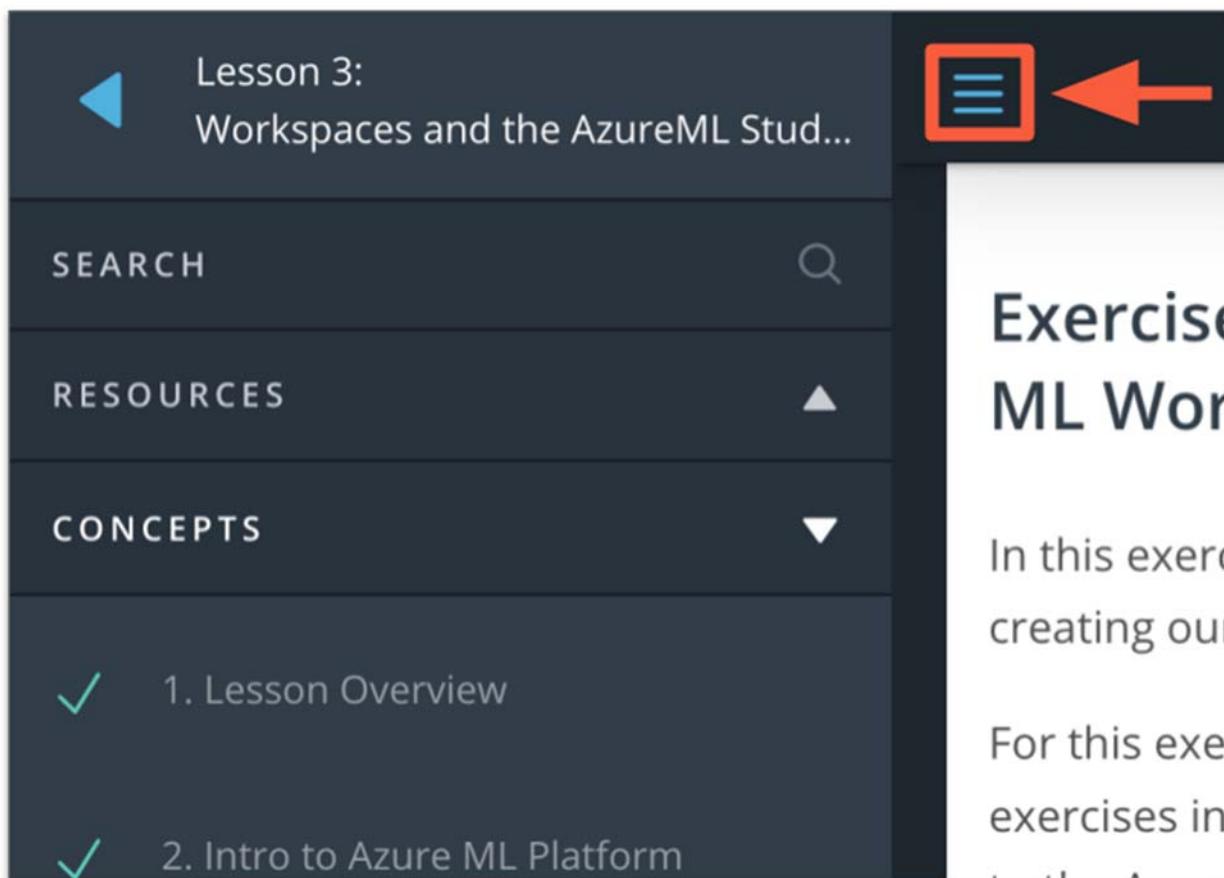
4. Exercise: Creating an Azure ML Workspace

In this exercise, we'll get started by creating our first Azure ML Workspace.

How to Use the VM

To access the VM and get started, click the Access Lab button. Note that it can sometimes take several minutes for your VM to load. This is a good time to get up and take a short break.

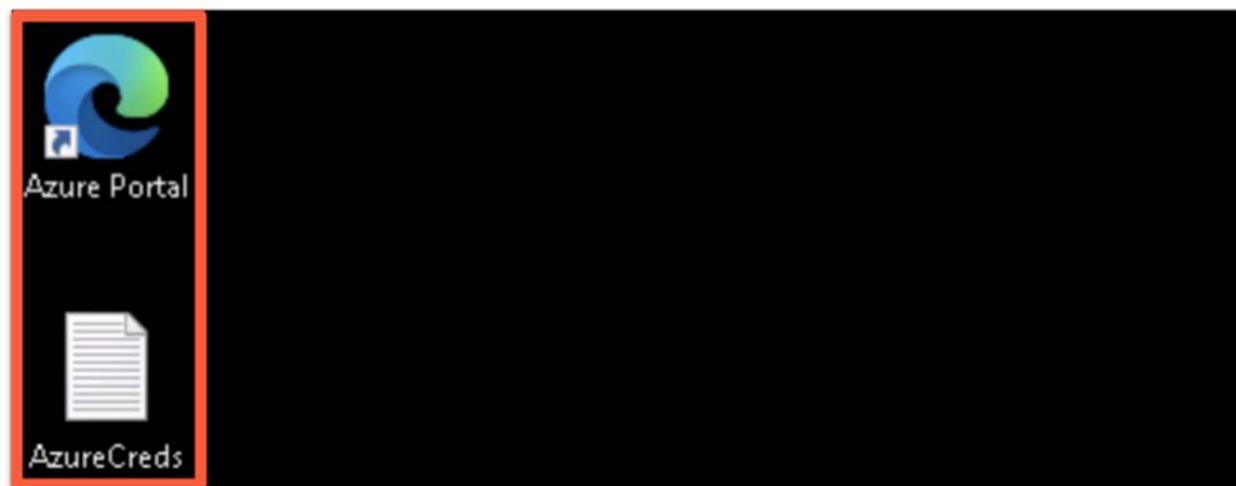
Tip: If you are working on a small screen, you can make more room for the lab by hiding the Udacity menu. Just click the hamburger menu in the upper left:



You can also resize these instructions as needed, by clicking and dragging on the divider:

The screenshot shows a dark-themed user interface window titled "Exercise: Creating an Azure ML Workspace". The window contains a description: "In this exercise, we'll get started by creating our first Azure ML Workspace." To the right of the text, there is a red-bordered rectangular area containing a horizontal double-headed arrow icon, indicating a step or action related to the exercise.

Once your VM loads, you can find a shortcut for the Azure Portal on the desktop:



Once you are in the portal, you will want to head to Azure Machine Learning. If you don't see it as shown in the video below, simply enter Machine Learning in the search box.

Before going further, be sure to do the following:

- Collapse the left-hand Udacity classroom menu to make more space for the exercise.
- Adjust the divider so that the lab and instructions take up the amount of space you like on your monitor.
-  Click on **ACCESS LAB** and wait for the VM to load. This may take several minutes to spin up.
- Go to **Azure ML Studio**. This should load automatically; if not, use the `AzureCreds.txt` file and **Azure Portal** shortcut on the desktop to get logged in.

Creating Your Workspace

Now let's try out the basic process of creating an Azure ML Workspace. Follow along and try it for yourself!

Note: The video shows you how to set the workspace edition, but in your VM this is done automatically and you will not see the option to set it. Additionally, the subscription and resource groups may be named differently from the video—you can simply select the available options (there will only be one option for each of these).

<https://photos.app.goo.gl/PjGns2odf8Dgt2qh7>

⚠️ Important: In some browsers, you may find that if you pause/play the instructional video, you are then unable to type anything in the VM. This happens because the YouTube video player sometimes does not return the keyboard focus. If this happens, simply click again anywhere else in these instructions (such as on this text) and you should find you're again able to type within the VM.

Here are the main steps you should be sure to complete before moving on:

-  From inside Azure Machine Learning Studio, click the **+ Add** icon.
-  On the **Basics** tab, select **Subscription** and create (if not already created)
-  Set the **Resource group**. In your VM, there will only be one option available (note that the name may be different from what is shown in the video).
-  Enter a **Workspace name**
-  Select the **Region**

QUESTION 1 OF 4

Which **Workspace Edition** is used for all of the exercises in this course?

(Note that your VM should have the edition set automatically, but you should be aware of the different editions for when you use Azure ML Studio in the future on your own.)

Basic

Pro

Advanced

Enterprise

QUESTION 2 OF 4

Why is the Workspace edition choice important?

It adds the ability to use additional machine learning features, including AutoML

It adds Active Directory support

You can get faster access to resources

It is cheaper

Now you know the basic process for setting up a workspace. This is something we'll do many times throughout this course.

Configuring Workspace Settings

<https://photos.app.goo.gl/nr2sjzrphu4jbKBw6>

QUESTION 3 OF 4

If you need to configure your workspace which icon would you select?

Happy Face Icon

Gear Icon

Question Mark Icon

QUESTION 4 OF 4

Which of these settings can you change in the workspace configuration?

(Select all that apply.)

Language

Font Size

Color Theme (e.g., Light/Dark)

Icon Size

As you can see, Azure ML studio has a lot of features, and we've only touched on a few of them here. Don't worry, you don't need to understand all of these right now—we will learn more about other features and how to use them throughout the course.

Additional Resources

- As an additional reference, you may want to have a look at Microsoft's documentation on how to [Create an Azure Machine Learning Workspace](#).

Getting Started with Lab

1. Once the environment is provisioned, a virtual machine (JumpVM) and lab guide will get loaded in your browser. Use this virtual machine throughout the workshop to perform the lab.
2. To get the lab environment details, you can select Lab Environment tab.



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

Udacity Nanodegree demo
0 hour(s), 56 minute(s) remaining × HIDE

Lab Guide Lab Environment Help

Environment Details Lab Resources

Azure Credentials

Here are your credentials to login to
<https://portal.azure.com> and access the On Demand
Lab

Username

Password

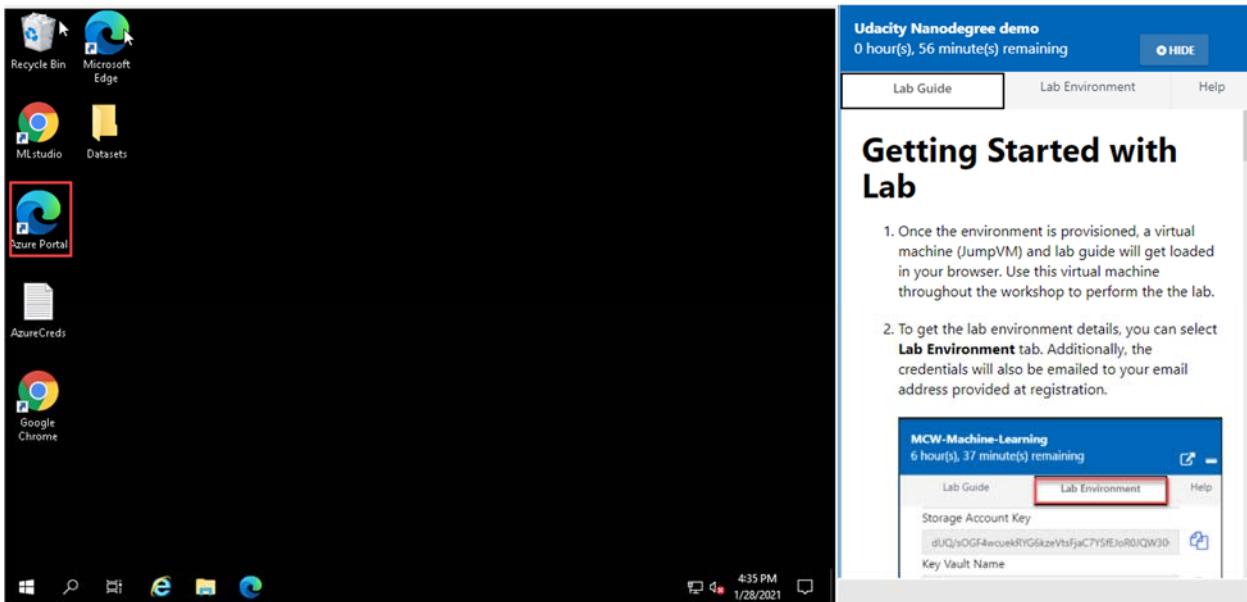
Environment Details

Resource Group : aml-vm-136385

Windows
VMDNS Name

Login to Azure Portal

1. In the JumpVM, click on Azure portal shortcut of Microsoft Edge browser which is created on desktop.



2. When you click on Azure portal, edge browser welcome screen will come up, select Get started.

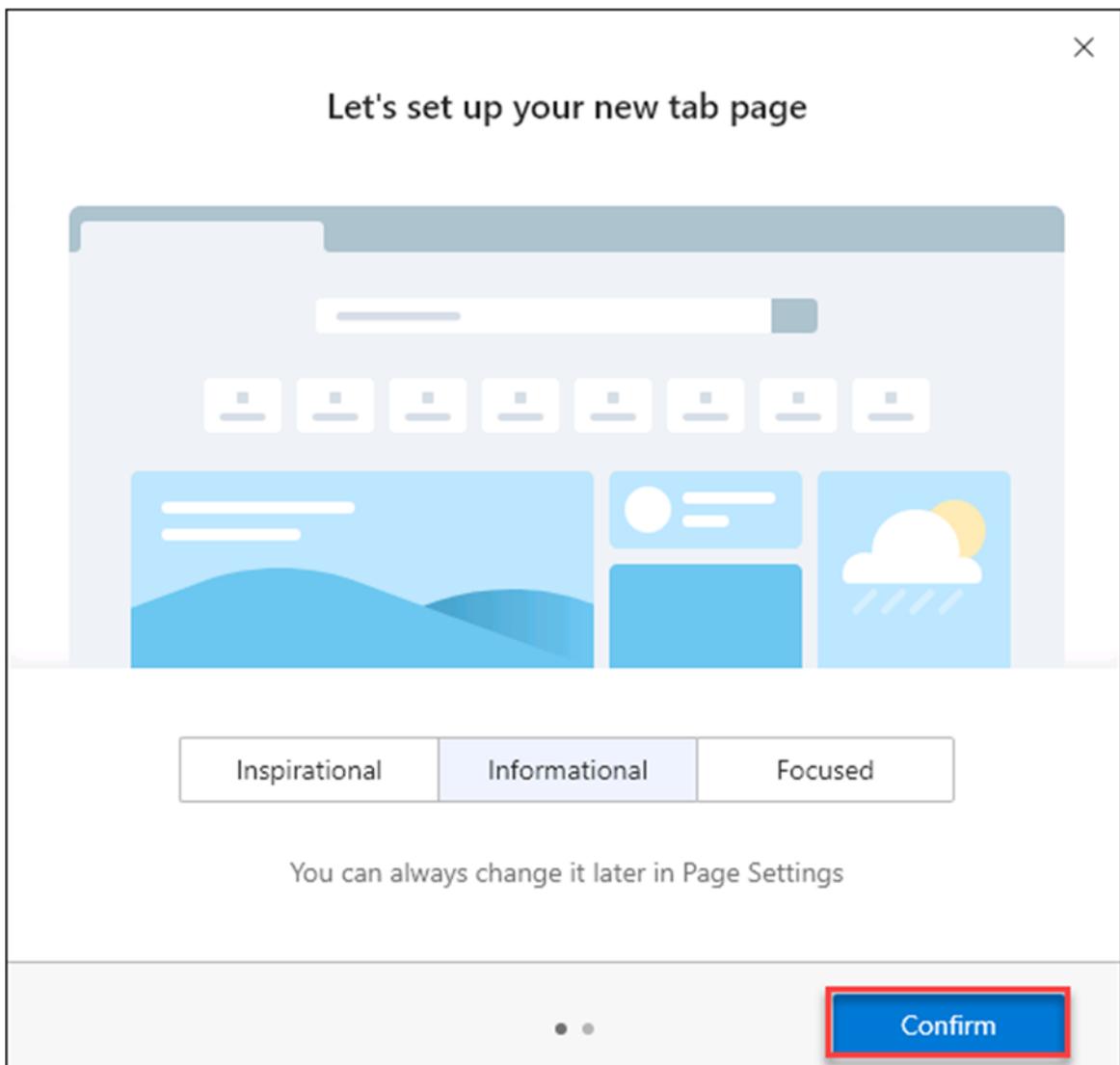


Welcome to the new Microsoft Edge

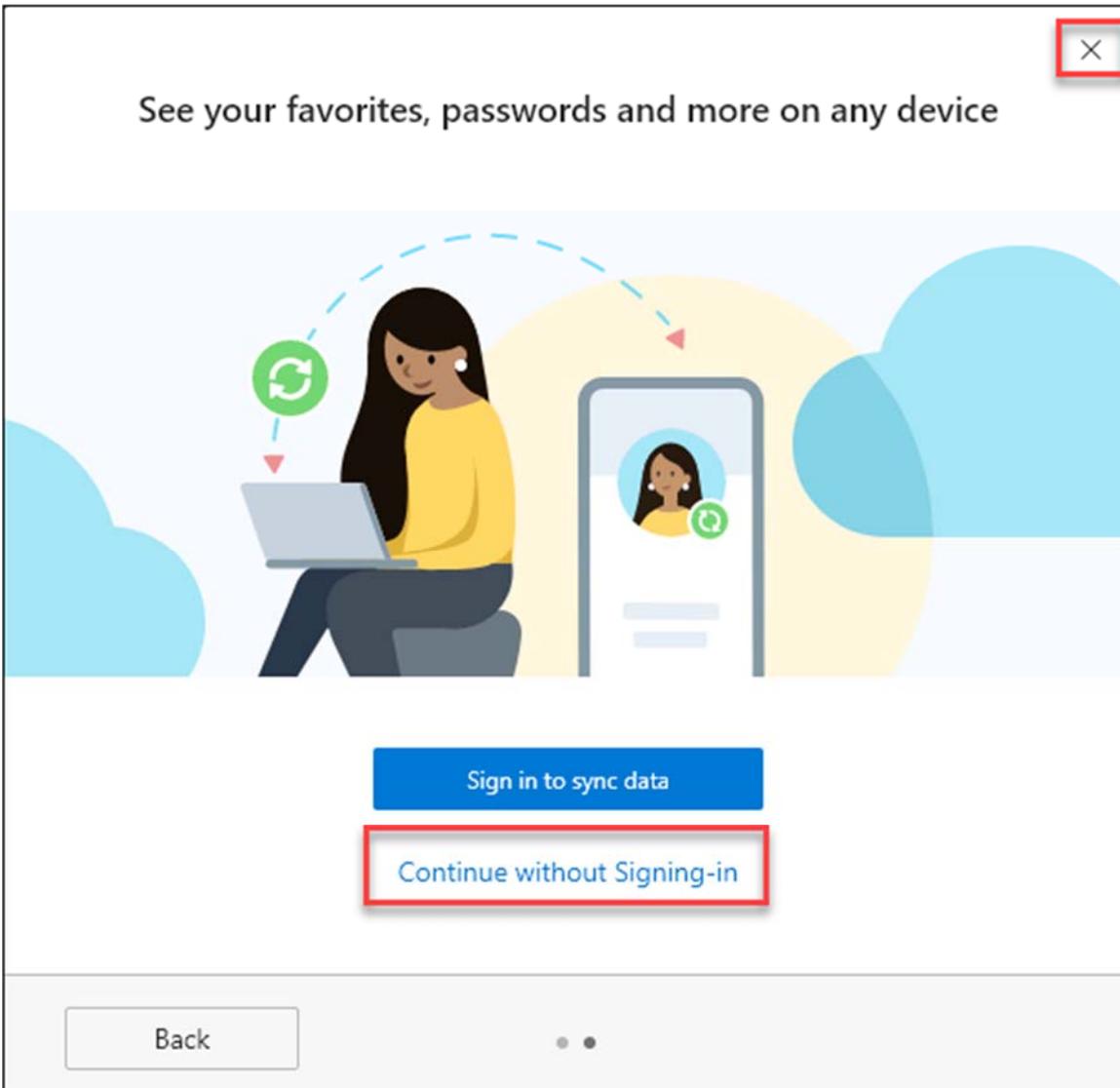
A fast, safe, and productive web browser that works for
you

[Get started](#)

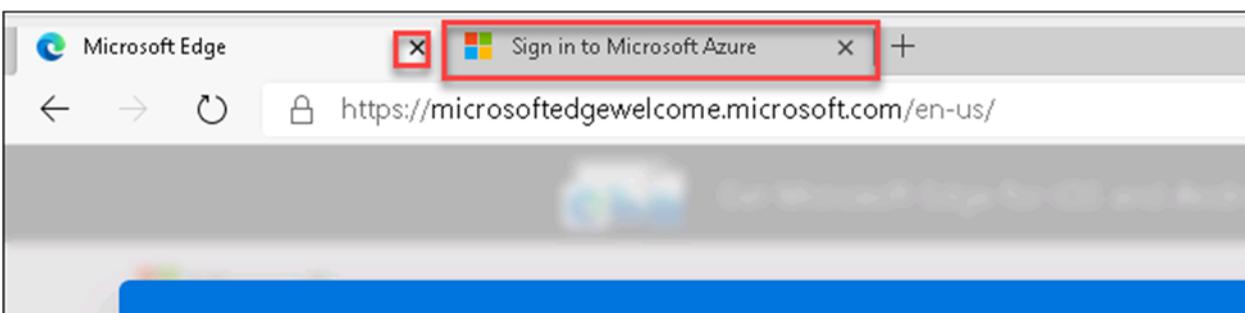
3. On next window, click on Confirm.



4. Now, you can close the popup which is coming up.

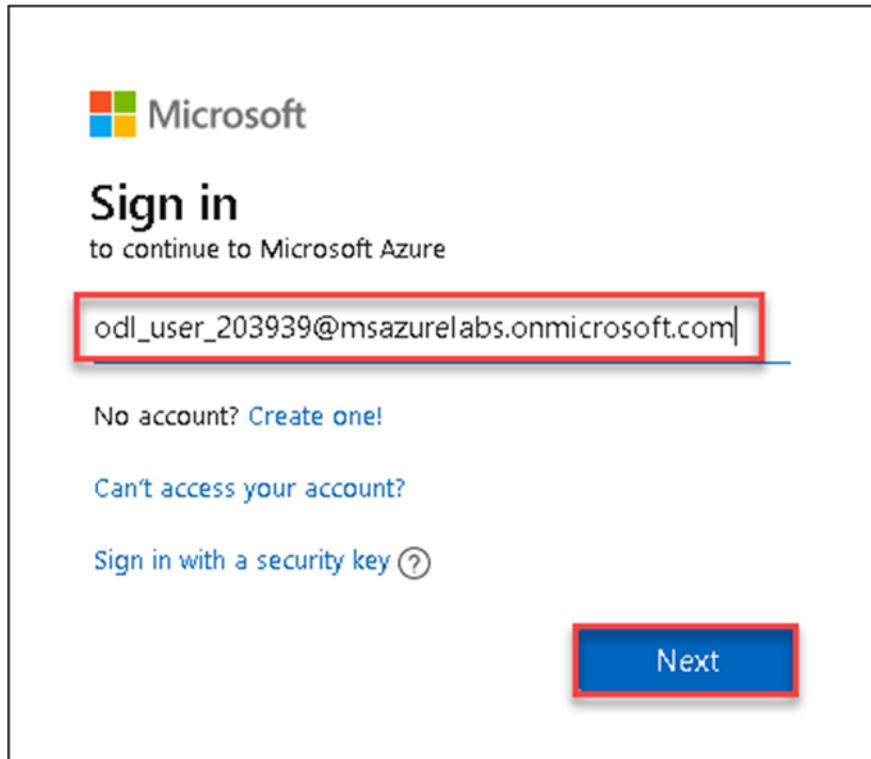


5. Now, you will see two tabs in edge browser, close first tab named with Microsoft Edge.



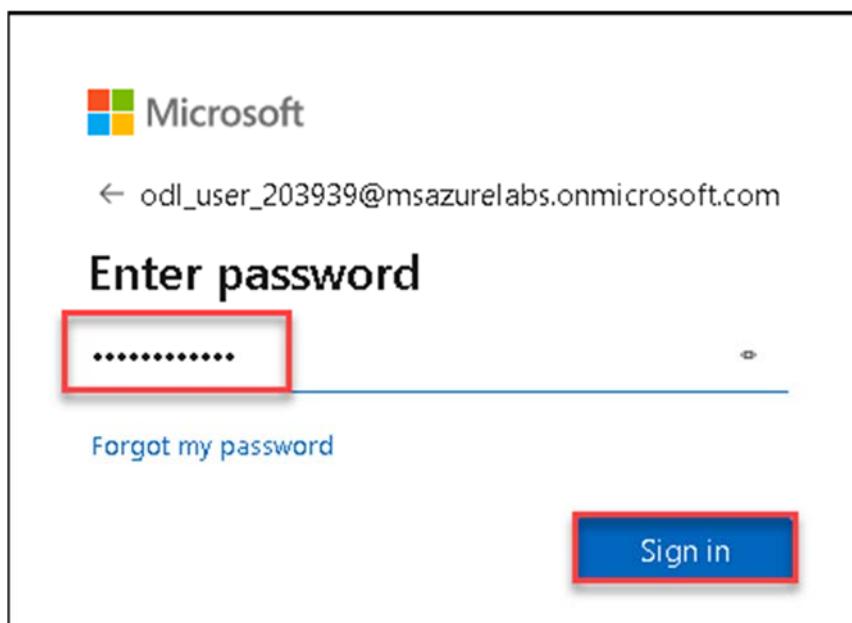
6. On Sign in to Microsoft Azure tab you will see login screen, in that enter following email/username and then click on Next.

- o Email/Username: odl_user_142274@udacitylabs.onmicrosoft.com



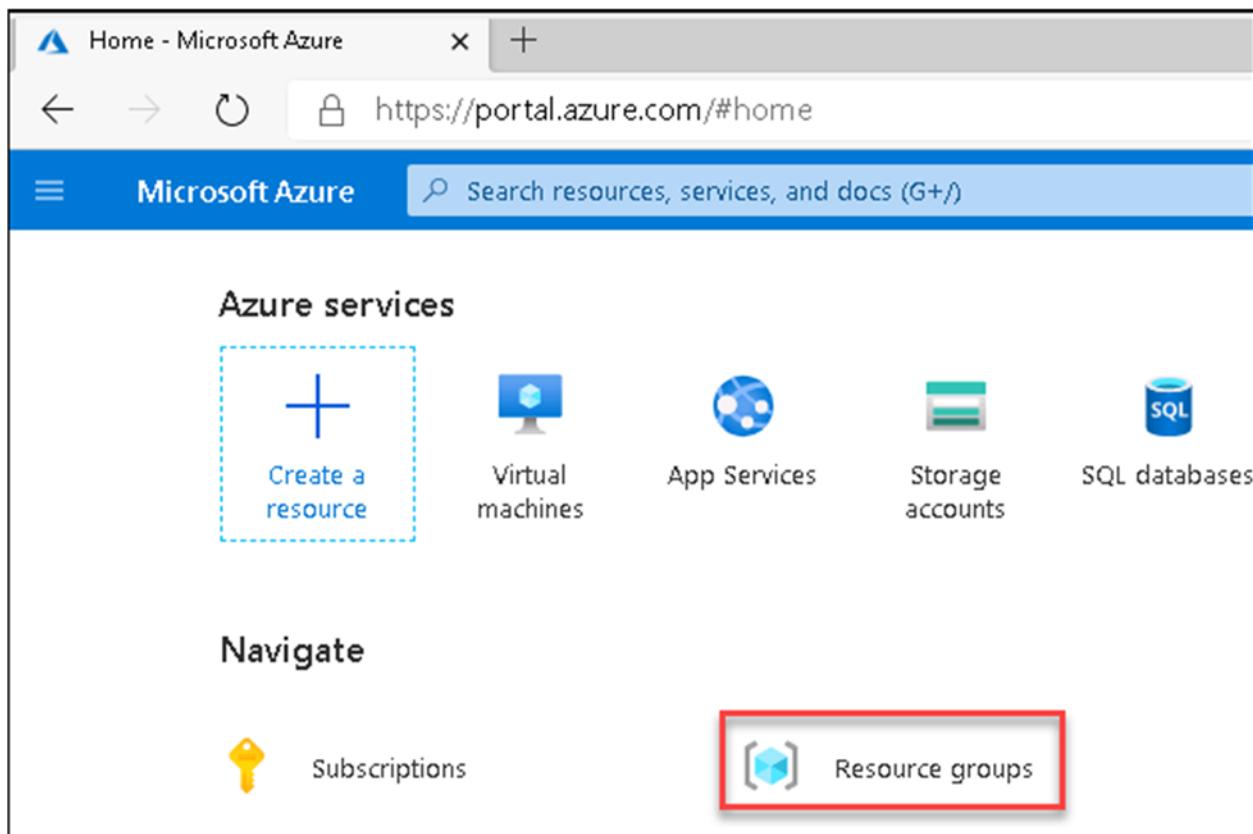
7. Now enter the following password and click on Sign in.

- o Password: jeda92MQS*Qz



8. If you see the pop-up Stay Signed in?, click No
9. If you see the pop-up You have free Azure Advisor recommendations!, close the window to continue the lab.

10. If a Welcome to Microsoft Azure popup window appears, click Maybe Later to skip the tour.
11. Now you will see Azure Portal Dashboard, click on Resource groups to see the resource groups.



12. Confirm you have all resource group are present as shown below.

The screenshot shows the Microsoft Azure portal interface. The title bar says 'Resource groups - Microsoft Azure'. The address bar shows the URL 'https://portal.azure.com/#blade/HubsExtension/BrowseResourceGroups'. The top navigation bar includes 'Microsoft Azure', a search bar, and user information 'odl_user_136385@udacity.com'. Below the search bar are buttons for 'New', 'Manage view', 'Refresh', 'Export to CSV', 'Open query', 'Assign tags', and 'Feedback'. A filter bar allows filtering by 'Subscription == all', 'Location == all', and 'Add filter'. The main content area displays 'Showing 1 to 1 of 1 records.' with one item listed: 'aml-quickstarts-136385' (Subscription: 'Udacity CloudLabs Sub - 04', Location: 'South Central US'). There are sorting options for 'Name ↑', 'Subscription ↑↓', and 'Location ↑↓'. At the bottom, there are buttons for '< Previous', 'Page 1 of 1', and 'Next >'.

13. Now, click on the Resource group and select your machine learning workspace

Here are your credentials to login to <https://portal.azure.com> and access the On Demand Lab

Username : odl_user_142392@udacitylabs.onmicrosoft.com

Password : psyy04EFX*GV

odl_user_142417@udacitylabs.onmicrosoft.com

gkbb95LBM*dO

Environment Details

Resource Group : aml-vm-142392

WindowsVMName : labvmn3ce4qq745wlq.southcentralus.cloudapp.azure.com

VMAAdminUsername : demouser

VMAdminPassword : Password123!

5. Managing Compute Resources

<https://photos.app.goo.gl/ihgKKZKcCe9U5ETo6>

Managing compute resources effectively involves several main components:

Compute instances

A central component of managing compute resources involves the ability to spin up a *Virtual Machine (VM)*, provision the correct resources for that VM, and then launch a *notebook*—which you can then use to control other components of your project, such as by calling out to a compute cluster.

Compute clusters

A *compute cluster* is a specialized cluster of Virtual Machines used to train ML models at scale, adjusting elastically to the requirements. Compute clusters can also employ specialized GPU or CPU resources.

Inference clusters

Inference clusters, as the name suggests, are used to establish a production ML endpoint and serve out predictions. Like compute clusters, inference clusters are also groups of Virtual Machines—and like compute clusters, they too can scale up and down in response to demand, thus ensuring that your customers always have a good response time. Inference clusters can be configured to do monitoring and logging, and set up to ensure that the model is running 24/7.

Attached compute

Attached compute, or "unmanaged compute," allows you to do integration with third party services. Those third party services could be anything from your own VM to something like a data bricks cluster that does managed Spark. If you have experience with a tool from, say, a previous job, you can integrate that tool with Azure ML Studio and thus use that pre-existing skill in your pipeline.

QUESTION 1 OF 2

Which of these compute resources are *elastic*?

(Select all that apply.)

Inference cluster

Compute instance

Attached compute

Compute cluster



Microsoft Azure
Noel Ching

QUESTION 2 OF 2

Below are the components we just discussed. Can you match each one with the correct description?

Submit to check your answer choices!

| DESCRIPTION | COMPONENT |
|--|-------------------|
| Used for integration with third party services. | Attached compute |
| A cluster of VMs used for model training. | Compute cluster |
| A cluster of VMs used for serving predictions. | Inference cluster |
| A VM you use as a starting point; used to launch a notebook that can call out to other components. | Compute instance |

Additional Resources

- [Microsoft Documentation: Create Compute Resources](#)

6. Exercise: Managing Compute Resources

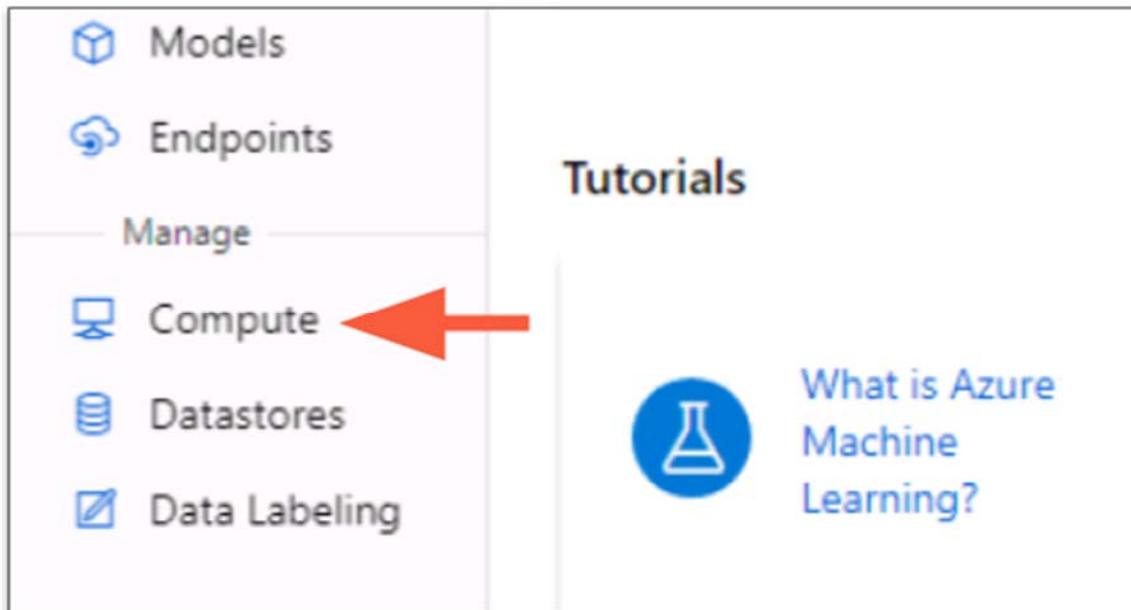
Exercise: Managing Compute Resources

As we just discussed, there are several types of compute resources:

- Compute instances
- Compute clusters
- Inference clusters

- Attached compute

In this exercise, we'll walk through creating the first three of these. To get started, click the **Access Lab** button and then, once you are in Azure ML Studio, go to the **Compute** section:



Now you're ready to follow along with the video below and try creating different types of compute resources.

Note that in the video, you will sometimes see Noah click the **New** button to create his compute resources. This button will be available *if you already have another compute resource of that type*. When creating your first compute resource of a given type, you'll instead see a **Create** button. These buttons do the same thing.

<https://photos.app.goo.gl/eWGjkBrW4ebC6Xhh6>

Before moving on, be sure to try creating some different compute resources:

- Create a **compute instance**
- Create a **compute cluster**
- Create an **inference cluster**

It may take a little while for each compute resource to be created. Once a resource has been created, you'll be able to see it available in the appropriate section. For example, if you go to the **Compute Instances** tab, you should be able to see that the instance you created is now **Running**:

The screenshot shows the 'Compute' blade in the Azure portal. The 'Compute instances' tab is selected. A message at the top states: 'In the wake of COVID-19, we are prioritizing maintaining service availability for first responders, health and ...'. Below the message are buttons for 'New', 'Refresh', 'Start', 'Stop', and a search bar. The main table lists one compute instance:

| Name | Status | Application URI |
|--------------|---------|--------------------------------|
| jupyter-demo | Running | JupyterLab Jupyter RStudio SSH |

Note that once you've created a compute instance, you'll be able to click the **Jupyter** link under **Application URI** in order to create and run Jupyter notebooks on the compute instance. This is something we'll be doing repeatedly in later lessons, particularly when we look at how to use the Azure ML SDK to write Python code to programmatically control our ML pipelines. For now, just take note of the option.

QUESTION 1 OF 2

What is the use of selecting a minimum and maximum number of nodes?

- Azure will provision the minimum number of nodes when it doesn't have enough systems available. It will provision more nodes as they become available.
- The number of nodes you can provision is dependent on your workspace edition (*Basic* or *Enterprise*).

The minimum number of nodes is for when you have fewer tasks running, and the maximum is for when you have all your tasks running at the same time.

QUESTION 2 OF 2

What should be the minimum number of nodes to ensure automated cost savings?

0

1

Selecting Nodes does not ensure cost savings

-1

7. Creating Notebooks

<https://photos.app.goo.gl/jkuk7aCDC7UMfCGR6>

Creating a compute instance and then spinning up a notebook is one of the most critical tasks in an Azure ML pipeline, because the notebook is used as an interface for conducting MLOps and controlling the pipeline. For example, the notebook is used to:

- Conduct Exploratory Data Analysis (EDA)
- Call out to a compute cluster to train models
- Call out to an inference cluster to deploy an endpoint

QUESTION 1 OF 2

The notebook itself can involve many application URIs, including Jupyter, JupyterLab, and RStudio.

Can you match each with its description?

Submit to check your answer choices!

| DESCRIPTION | URI |
|---|------------|
| Used for standalone workflows in Python. | Jupyter |
| Used for advanced configurations like copying data and installing packages. | SSH |
| Used for team-based workflows. | JupyterLab |
| Used to control R projects; more commonly used in academic and statistics. | RStudio |

QUESTION 2 OF 2

What best describes the role of a notebook in an Azure Machine Learning pipeline?

- Notebooks are usually run on compute clusters and are the primary tool for model training.
- Notebooks are usually run on inference clusters and are the primary tool for serving predictions.
- Notebooks are usually run on a compute instance and are the primary tool for calling out to, and controlling, other parts of the pipeline.

Additional Resources

Here are some additional Microsoft resources about notebooks that you may find helpful as a reference:

- [Explore Azure Machine Learning with Jupyter notebooks](#)
- [How to run Jupyter Notebooks in your workspace](#)

8. Exercise: Creating Notebooks

<https://photos.app.goo.gl/UWe7EdHRVK6YE7DB7>

We'll be using Jupyter Notebooks multiple times in this course, so make sure you've followed along and can create one in Azure ML Studio:

- Create a new compute instance
- Name the notebook
- Select between CPU or GPU
- Open the notebook interface using Application URI
- Create a new notebook and try running some Python code in it

QUESTION 1 OF 2

Which of these compute types would you use to create a jupyter notebook?

- Compute Instance
- Compute Cluster
- Inference Cluster
- Attached Compute



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

QUESTION 2 OF 2

What is a good reason to create a CPU notebook instead of a GPU?

You are doing general data science and want to keep costs under control

There is no significant cost difference, but CPU performs better under certain circumstances.

You should always use a GPU

You should always use a CPU

9. Choosing the Compute Resource for Your Needs

When choosing the compute resources for your needs, some key factors to consider are **GPU vs CPU**, **cluster size**, **VM size**, and **dedicated vs low-priority instances**. Let's have a look at the factors involved in each of these choices.

<https://photos.app.goo.gl/xfGyCqh254XQ6kwk6>

CPU vs GPU

Characteristics of CPUs:

- Less expensive
- Lower level of concurrency (a CPU isn't designed for parallel computing)
- Can be a lot slower in training deep learning models
- If time really isn't a critical factor, it could be a good option to save

Characteristics of GPUs

- More expensive
- Higher level of concurrency (GPUs are designed for parallel computing)
- Optimal for deep learning
- Used in both inference and training; the power level for these may be different

Cluster Size

The primary consideration with cluster size is response time:

- If you have a cluster that starts at zero nodes, that means it will have to spin up a node when it is needed for a task, which could lead to slower response times (but will save on costs).
- A larger starting cluster size will result in better responsiveness, but will also be more expensive.

In sum, if you have more time, but less money, you may want to go with a smaller cluster size. Conversely, if you have less time and more money, you may want to go with a larger cluster size.

VM Size

The "size" of a VM can vary in terms of a number of features, including:

- RAM
- Disk size
- Number of cores
- Clock speed

Getting a VM with more RAM, larger disk size, a greater number of cores, and higher clock speed, will result in better performance, but will also be more expensive. The choice depends on the training and inference needs of the problem you're trying to solve, as well as your time and budgetary constraints.

Dedicated Instances vs Low-Priority Instances

- When an instance is *low-priority* this means it is *interruptible*—essentially, Microsoft Azure can take those resources and assign them to another task, thus interrupting a job.
- When an instance is *dedicated*, or *non-interruptible*, this means that the job will never be terminated without your permission.

This is another consideration of time vs money, since interruptible instances are less expensive than dedicated ones.

QUESTION 1 OF 2

One choice in selecting a compute resource is whether you need a VM with just a CPU or whether you also need a GPU.

Can you match each of these descriptions with whether it better describes CPU or GPU?

CPU

| DESCRIPTION | CPU OR GPU? |
|--|-------------|
| More expensive | GPU |
| Optimal for deep learning | GPU |
| Higher level of concurrency; designed for parallel computing | GPU |
| Less expensive; if time isn't a critical factor, it could be a good option to save money | CPU |



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

QUESTION 2 OF 2

You are running a deep learning task that needs to have a quick response time and be highly reliable—and you have a large budget to pay for the resources.

Which of the following combinations is best suited to your needs in this case?

A small cluster size running dedicated, high-performance VMs with GPUs

A large cluster size running interruptible, high-performance VMs with GPUs

A large cluster size running dedicated, high-performance VMs with GPUs

A large cluster size running dedicated, high-performance VMs with CPUs only

10. Lesson Review

<https://photos.app.goo.gl/DA7CAKcaFXz5AW9v7>

Lesson Outline: Workspaces and AzureML Studio

Here's what we covered in this lesson:

- **The Azure ML Platform.** We talked about the core features of the Microsoft Azure ML platform and how they enable us to be more productive as data scientists or machine learning engineers—including how elastic resources give us the power to do ML at scale.
- **Managing and choosing compute resources.** We explored the role of compute instances, compute clusters, inference clusters, and attached compute. We also explored the tradeoffs involved in choosing resources, such as GPU vs CPU, VM size, cluster size, and dedicated vs low-priority instances.
- **Workspaces and Notebooks.** Workspaces and notebooks are critical components of Microsoft Azure. We learned how these tools enable you to be a more effective data scientist—including the use of Jupyter to build and deploy machine learning models.

Lesson 3 Datastore and Datasets

1. Lesson Overview

<https://photos.app.goo.gl/mkHjnAk2L22zFVs5>

Lesson Outline: Datastores and Datasets

This lesson is all about managing data by using **Datastores** and **Datasets**. Here are the main topics we'll cover:

- **Managing data.** We'll talk about some of the MLOps workflows that are enabled by managing data within the Azure ML studio platform.
- **Working with Datasets.** Working with datasets is critical for sharing data and having a high performance pipeline that doesn't require copying the data back and forth.
- **Dataset monitoring and data drift.** If you collect your data, train a model, and then you get some new customers, those new customers may not be the correct target for the model that you created previously—you may need to recreate the model to get a better fit. With Dataset monitoring, you can detect those changes.
- **Using Datasets in notebooks.** We'll cover the workflow for using Datasets in notebook, and we'll show you how you can leverage the open Datasets that are available in Azure.
- **Dealing with sensitive data.** One of the most important topics in ML is how to handle personally identifiable information and protect it using encryption.

2. Managing Data in Azure Machine Learning

<https://photos.app.goo.gl/Hpewpa6qKKTppc398>

The Problem: Managing Data Without the Cloud

To understand why we need Datastores—and cloud-based data storage more generally—it helps to first consider the alternative. If you were trying to handle the data on your own local machine, you would be faced with a number of challenges, including:

- **Data governance.** If you have secret or sensitive data that you need to keep protected, *data governance* becomes a concern.
- **Do-It-Yourself (DIY) interface to data storage types.** You will have to write a bunch of extra code to connect different storage types (such as SQL or Databricks).
- **Hardware constraints.** Your machine's resources (e.g., CPU, disk IO, storage) are limited.
- **Third party integration.** If you want to use an off-the-shelf, third party tool, this could pose integration problems—and you will have to handle the integration problems on your own.

The Solution: The Azure Cloud and Datastores

Datastores have a number of key properties that address the above challenges:

1. **Easy-to-use interface for many storage types.** A Datastore is essentially an abstraction that provides easy methods to interact with many different tools and with many different complex data storage types. This removes the extra work of struggling with third-party integrations and writing do-it-yourself interfaces for different data storage types.
2. **Secure, centralized control of the data.** This means you don't need to have separate systems that move your data to different locations—which results in *better efficiency*. You don't have to copy the data back and forth; instead, your ML projects can simply use copies derived from that centralized data. This centralized control also includes built-in encryption and thus results in *better security*.
3. **Scalability.** Datastores leverage cloud-native elastic storage, meaning that they can scale in response to demand. Obviously, this is something your hard drive cannot do.

QUIZ QUESTION

Again, Datastores address a number of problems that you would otherwise run into if you weren't using the cloud.

Can you match each problem with the feature of Datastores that addresses it?

Submit to check your answer choices!

| PROBLEM | DATASTORES FEATURE |
|---|--|
| Your machine's resources (e.g., CPU, disk IO, storage) are limited. | Scalability / Elasticity |
| You have to write a bunch of extra code to connect different storage types. | Easy-to-use interface for many storage types |
| You need to make many copies of the same data. | Centralized control |
| You have sensitive data that needs to be protected. | Centralized control |

Additional Resources

- If you'd like to learn more about ingesting data on Azure, have a look at [the Microsoft documentation on describing data ingestion and processing](#).

3. Exercise: Managing Data in Azure Machine Learning

In this exercise we'll create a new dataset by uploading a CSV file, and then profile and explore it. You can find the dataset in the `Datasets` folder on the desktop of the virtual machine—or, if you are using your own Azure account, you can [download the dataset here](#).

<https://drive.google.com/file/d/1pSGcRldGngoJMyyfUaJ41EEW67vheDLz/view?usp=sharing>

Create the Dataset

<https://photos.app.goo.gl/MModF7TaAeC3JDNt9>

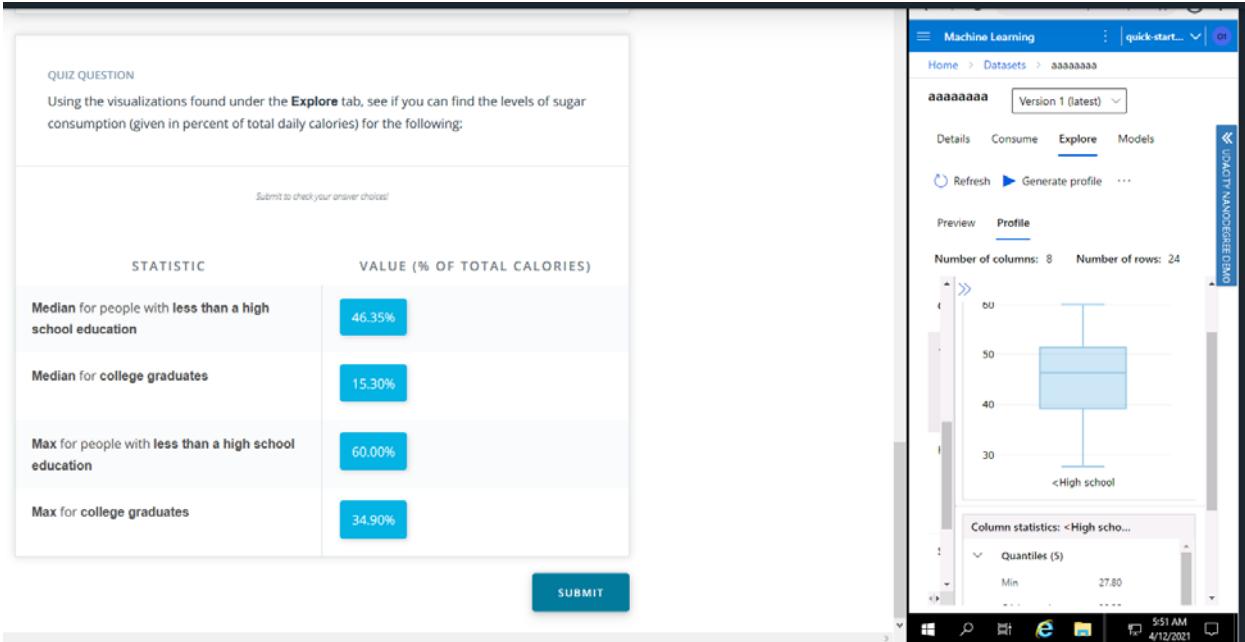
Let's upload the CSV file and use it to create a dataset.

- Go to **Datasets** and click **Create dataset > From local files**.
- Give the dataset a **name**, set the type to **tabular** and enter a **description**.
- Select a **datastore**. The *Azure Blob Storage* default is fine.
-  **Select the file** for the dataset by uploading the `sugar.csv` file (you can find it in the `Datasets` folder on the desktop of the VM).
- Under **Settings and preview**, be sure to select **Column headers > Use headers from the first file**.
- Create** the dataset and then switch to the **Explore** tab—we'll look at this next!

Explore the Dataset

<https://photos.app.goo.gl/sPqEdGJnvRtwZUcF8>

-  Click on the **Explore** tab and then switch the view from **Preview** to **Profile**
-  Click the **<<** symbol on the right and play around with viewing the **histogram** and **boxplot** visualizations for some different columns



The screenshot displays two windows side-by-side. On the left is a 'QUIZ QUESTION' window with the following text:

Using the visualizations found under the **Explore** tab, see if you can find the levels of sugar consumption (given in percent of total daily calories) for the following:

Submit to check your answer choices!

| STATISTIC | VALUE (% OF TOTAL CALORIES) |
|--|-----------------------------|
| Median for people with less than a high school education | 46.35% |
| Median for college graduates | 15.30% |
| Max for people with less than a high school education | 60.00% |
| Max for college graduates | 34.90% |

A 'SUBMIT' button is at the bottom.

On the right is a Microsoft Machine Learning studio interface window titled 'Machine Learning' with the path 'Home > Datasets > aaaaaaaaa'. It shows a dataset named 'Version 1 (latest)'. The 'Explore' tab is selected. A boxplot visualization for the column '<High school' is shown, with the following statistics:

- Min: 27.80
- Quantiles (5):

4. Working with Datasets

We'll get into an exercise shortly in which we get some hands-on practice working with datasets. But first, let's make sure we are clear on the relationship between Datastores and Datasets:

- A **Datastore** is an abstraction that gives you an easy way to interact with different data storage types.
- A **Dataset** is a pointer to the location of a data file stored in a Datastore.

In this next video we'll explore what that means in more detail.

<https://photos.app.goo.gl/iEY17a5Pq5SskwUEA>

QUESTION 1 OF 2

Which of the following are correct statements about **Datasets**?

(Select all that apply.)

- They are abstractions that help you interface with different data storage types.
- They are simply data files that are inside Datastores.
- They are pointers to data files that are inside Datastores.
- You can create multiple datasets that all point to the same data file.

QUESTION 2 OF 2

Which of the following are **benefits of Datasets**?

(Select all that apply.)

- You can create many different Datasets that all refer to that same file, but that can have different characteristics or be in different projects.
- Datasets can simplify the access path to the data's location.
- Datasets can make it easier to share and collaborate.
- They can improve performance
- They allow lazy evaluation
- There are many open Datasets easily available to you

5. Dataset Monitor and Data Drift

<https://photos.app.goo.gl/zv67CACLMki1RsfM7>

A common problem is that you may train a model on some data and achieve a good fit—but then, over time, new data comes in and the model fits less and less well. This is called **data drift**. You can use **Dataset monitor** to address this problem. Essentially, the process is:

1. **Register a baseline Dataset.** This would be a Dataset that has a good fit with your model (commonly you would use the Dataset with which you trained the model).
2. **Register a target Dataset.** This might be your model input data that you are receiving over time.
3. **Check for a delta.** If there's a specified delta (i.e., a given magnitude of difference) between the baseline and target Datasets, the system will trigger an alert (such as an email) and/or some form of automatic, corrective action (such as re-training the model).

QUESTION 1 OF 2

What can happen if there is a large data drift in your dataset?

- Nothing, since data drift does not affect the accuracy of the model
- Data drift can reduce how well the model fits with new data
- Data drift can affect existing data by duplicating data points
- Data drift increases the number of data points and increases model training time

A feedback card with a red 'X' icon in the top right corner. In the center is a circular icon containing a document with checkmarks and a smiling person icon. Below the icon is a red 'Try Again' button.

Remember that data drift happens when new data is added to an existing dataset, and the model—which was trained on the original data—performs less well on the new data.

TRY AGAIN

QUESTION 2 OF 2

What are some ways to counter data drift?

(Select all that apply.)

- Keep using the same model
- Train a new machine learning model that takes into account the new data
- Remove outlier data or find the root cause of the drift and fix it
- Create a new dataset and discard the previous data

Data drift can affect existing data by duplicating data points.



Try Again

Remember that one way to account for data drift is to train a new model that fits better. Alternatively, you can investigate the cause for the data drift and remove spurious data points.

TRY AGAIN

✓ Remove outlier data or find the root cause of the drift and fix it

Additional Resources

- You can read more about the process of detecting data drift and data monitoring in Microsoft's article, [Detect data drift \(preview\) on datasets](#).

6. Using Datasets in Notebooks

As we have mentioned previously, notebooks provide the main control point for interfacing with the Azure ML platform. You can use them to control things like model deployments and distributed training, and they also provide a convenient way to share with other people on your team. In this section, we'll get into how you can use Datasets in your notebooks.

<https://photos.app.goo.gl/i8XheKPQKWqZsvKn6>

There are two **Dataset types** you should take note of:

- **FileDataset.** This is a generic Dataset type. This is useful for things like computer vision or anything where you need to have a lot of flexibility.
- **TabularDataset.** As the name implies, this type is for *tabular* (i.e., table-based) data. This type allows you to handle formats like JSON, CSV, or Parquet.

As you're getting started using Datasets in Azure notebooks, you may find it very helpful to make use of [Azure Open Datasets](#). These are curated, ready-to-use, public Datasets that you can use to get things running and do ML without needing to worry about collecting your own data.

QUESTION 2 OF 2

Can you match the dataset type with its use case?

Submit to check your answer choices!

| DATASET TYPE | USE CASE |
|--|-----------------------|
| Generic dataset type; can be used for image files | FileDataset |
| Can be used for CSV and Parquet files | TabularDataset |
| Previously collected and curated data present in the Azure Ecosystem | Open Datasets |

X


[Try Again](#)

Remember that **FileDataset** can be used for generic data sources like images, which is useful for computer vision applications or text files for NLP.

Remember that **TabularDatasets** can be used for (as the name implies) tabular data, such as CSV or parquet files.

Remember that Azure provides **Open Datasets** that are cleaned, openly available datasets.

[TRY AGAIN](#)



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

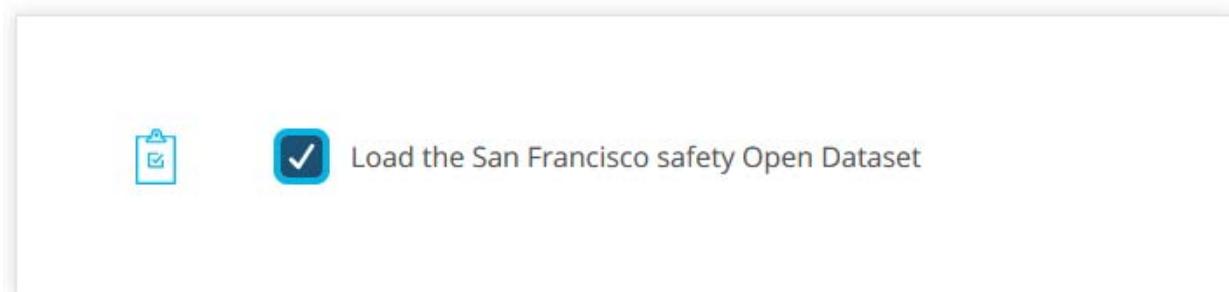
7. Exercise: Using Datasets in Notebooks

In this exercise, we'll first see how we can load an Open Dataset and then use it from a Jupyter Notebook.

Loading an Open Dataset

First, we'll walk through how to load an Open Dataset. Follow along and load this same dataset yourself so that you can use it in the next part of the exercise.

<https://photos.app.goo.gl/y3fQBk5FL2Dg2wpf6>



Using the Dataset in a Notebook

One of the ways you can query data in Azure Machine Learning studio is by generating sample queries for Jupyter Notebooks. Let's explore that workflow.

Note: When you query the data as shown in the video, be aware that it can take some time to pull all the data into the dataframe and display the results. In the meantime, it may appear that the notebook is not doing anything.

<https://photos.app.goo.gl/Hb3mtCNtxd2ZE6oF9>

Now it's time to explore the dataset in a Jupyter notebook. Here are the main steps:



- Find the dataset
- Copy the sample usage code
- Go to a Jupyter Notebook
- Run the query that displays the data (*this may take a few minutes to display the results*)

Exploring the Data

Now that we have loaded the data, we can explore it with any number of our standard data analysis tools.

For example, you could assign the data to a Pandas dataframe:

```
df = dataset.to_pandas_dataframe()
```

And then, say, get a count all of the incident reports in one query:

```
df.groupby('category').count().sort_values(  
    by="status", ascending=False)
```

At the time we ran this, these were the top reports:

Street and Sidewalk Cleaning: 19,451
Potentially Life-Threatening: 10,668
Graffiti: 6,359
Non Life-threatening: 5,493

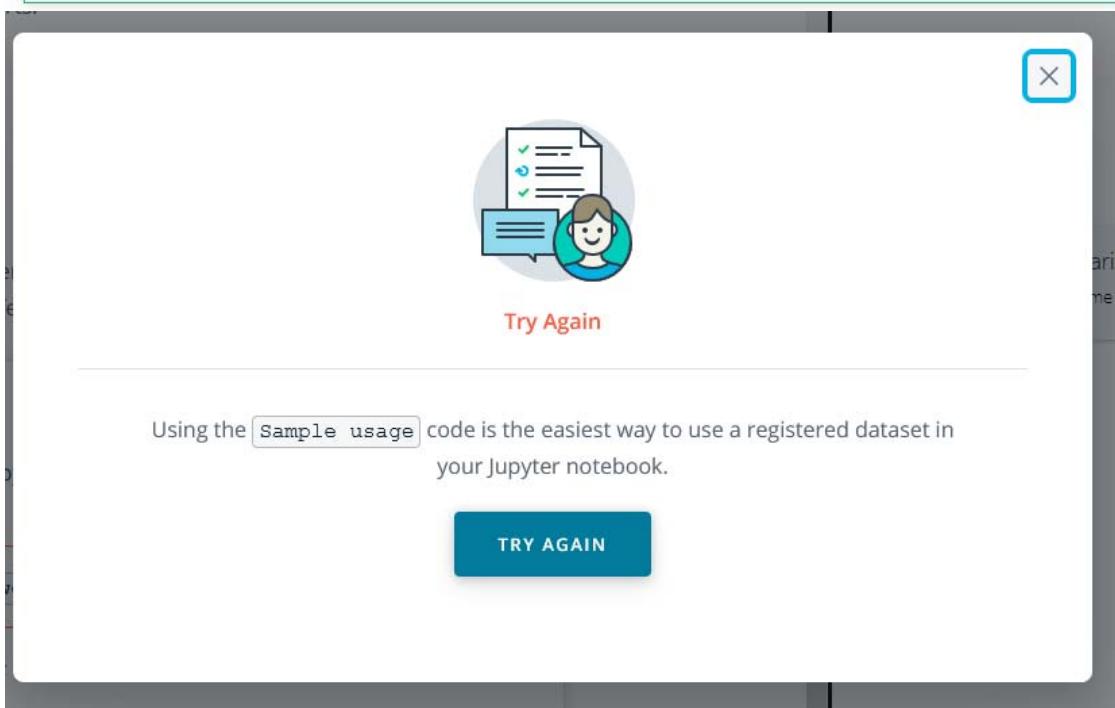
Feel free to try this for yourself or play with other features of the data. Note that your numbers might be a bit different from ours if you selected a different date range.

QUIZ QUESTION

What is the easiest way to use data in a Jupyter notebook?

Copy the link to the dataset and use `wget` in your Jupyter notebook

Copy the `sample_usage` code after clicking the consume button and paste it in your notebook



8. Dealing with Sensitive Data

<https://photos.app.goo.gl/k9caFCkevfimrF967>

The Principle of Least Privilege (PoLP)

In the video, Noah mentioned a fundamental security principle known as the [Principle of Least Privilege](#) or **PoLP**, which says:

We should give users access only to what they need and nothing more.

Essentially, the more privileges a user has, the more potential there is for them to misuse those privileges. By restricting a user's access rights to only those features they need, we can reduce this potential of misuse.

You don't need to have a deep understanding of this principle for this course, but if you're curious to learn more about it, you might enjoy [this article](#) from the Cybersecurity and Infrastructure Security Agency (CISA) (part of the US Department of Homeland Security). It has a number of examples that are interesting to read and will help make the principle more concrete.

Security Principles and Practices

To keep your data from being compromised, you'll want your system to follow some core security principles and practices. Let's briefly review some of the main ones:

- **Authentication.** This is all about *who can log in*. You want to restrict access using the PoLP principle, so that people who don't need to log in can't log in.
- **Authorization.** This is all about *what a user can access* once they have logged in. Again, you want to restrict access using the PoLP to make sure that users are only able to access, for example, the folders that they really need to be able to access. A user can only access what they need to access.
- **Securing compute targets and data.** You want to lock down both physical access to the data storage and also online access (e.g., applying best practices, such as firewall rules, to lock down access to open ports).
- **Network security.** This is all about protecting data from interception. Are you using a VPN to limit access to the network resources? And are you also encrypting the data over the network?
- **Data encryption.** Data encryption itself encompasses both data at *rest* and *in transit*. Obviously, you want to use secure protocols to prevent others from accessing the data. But by making sure the data is encrypted at all times, you ensure that even if the data *is* somehow breached, it still cannot be used.

Azure Encryption Services

To help you ensure your data is secure, Azure has several encryption services that are deeply integrated with the platform:

- **Increased control over keys and passwords**, making it much easier to implement security best practices as described above.
- **The ability to create and import encryption keys in minutes**, which ensures that you're not having performance bottlenecks by using encryption.
- **Applications have no direct access to encryption keys**. Each service only accesses what it needs, so that If a machine is compromised, this breach won't then open the door to a further attack on your system.

QUIZ QUESTION

Below are some of the security principles and practices we just discussed. Can you match each of them with the description that best fits?

Submit to check your answer choices!

| DESCRIPTION | SECURITY PRINCIPLE / PRACTICE |
|---|-------------------------------------|
| People who don't need to log in can't log in. | Authentication |
| Network security | Data is protected from interception |
| Even if the data is somehow breached, it still cannot be used. | At-rest data encryption |
| Once logged in, users can only access what they need to access. | Authorization |
| Lock down both physical and online access to the data storage. | Securing compute targets and data |

9. Lesson Review

<https://photos.app.goo.gl/vwsbjFxiTqjTMvw48>

Lesson Outline: Datastores and Datasets

This lesson was all about managing data by using **Datastores** and **Datasets**. Here are the main topics we covered:

- **Managing data.** We talked about some of the MLOps workflows that are enabled by managing data within the Azure ML studio platform.
- **Working with Datasets.** Working with datasets is critical for sharing data and having a high performance pipeline that doesn't require copying the data back and forth.
- **Dataset monitoring and data drift.** If you collect your data, train a model, and then you get some new customers, those new customers may not be the correct target for the model that you created previously—you may need to recreate the model to get a better fit. With Dataset monitoring, you can detect those changes.
- **Using Datasets in notebooks.** We covered the workflow for using Datasets in notebook, and we showed you how you can leverage the open Datasets that are available in Azure.
- **Dealing with sensitive data.** One of the most important topics in ML is how to handle personally identifiable information and protect it using encryption.

Lesson 4 : Training models in Azure ML

1. Lesson Overview

<https://photos.app.goo.gl/LjXEYdGvcAWxWjsE9>

Lesson Outline: Training models in Azure

This lesson is all about **training models**. Here's what we'll be covering in this lesson:

- **Introduction to Designer.** First, we'll learn how we can quickly build our models using the no code/low code, drag-and-drop interface of the Designer.
- **Managing pipelines.** We'll cover how to create a pipeline that we manage ourselves using the console, which will allow us to make very small changes that can be run repeatedly. Our goal in managing our pipelines is to do *MLOps*—that is, to apply a DevOps-based workflow to ML.
- **Running pipelines.** We'll learn how to run pipelines and experiments, as well as how to look at the output and tune our models.
- **Hyperparameters in experiments.** Finally, we'll learn how to use hyperparameters in experiments, including how we can automate the creation of hyperparameters and make very small changes that create huge value in terms of prediction accuracy.
- **Monitoring models with application insights.** Finally, we'll talk about how we can leverage Azure's *Application Insights* infrastructure to monitor the performance of our models in production.

2. Introduction to Designer

<https://photos.app.goo.gl/S7FfuaPg4u2kbRhy7>

The Designer is a no code/low code interface that allows you to organize resources using a simple drag-and-drop interface. You can use the Designer to build, test, and deploy your ML models. The Designer can be used to organize and configure a variety of resources, including:

- [Pipelines](#)
- [Datasets](#)
- [Compute resources](#)
- [Registered models](#)
- [Published pipelines](#)
- [Real-time endpoints](#)

Note that the Designer has many sample projects, which provide a great way to get started and learn the interface.

QUIZ QUESTION

Which of the following are true statements about the Designer?

(Select all that apply.)

It can be used to build ML models.

It uses a Command Line Interface to programmatically control your ML pipeline using BASH commands.

It can be used to deploy ML models.

It can be used to test ML models.

It uses a low code/no code drag-and-drop interface to organize your resources.

Additional Resources

- For a great overview of the designer, also see Microsoft's article, [What is Azure Machine Learning designer?](#)

3. Managing Pipelines

<https://photos.app.goo.gl/fZHqNWh9LiBMzYF6A>

What is Azure Pipelines?

Azure Pipelines is a cloud service that incorporates the DevOps approach of [Continuous Integration \(CI\)](#) and [Continuous Delivery \(CD\)](#) to automate your ML workflow. With Azure Pipelines, you can:

- Automatically build and test your code project
- Share your project with others

Ways to Run Pipelines

There are several different tools you can use to control a pipeline in Azure:

- You can use the **Designer** to set up your pipeline using a low code/no code, drag-and-drops interface.
- You can use the **console** to go through and adjust an existing pipeline.
- You can use the **Azure ML SDK** to programmatically control your pipeline with Python.
- You can use a **notebook** to spin up a pipeline, configure it, and rerun it whenever needed.

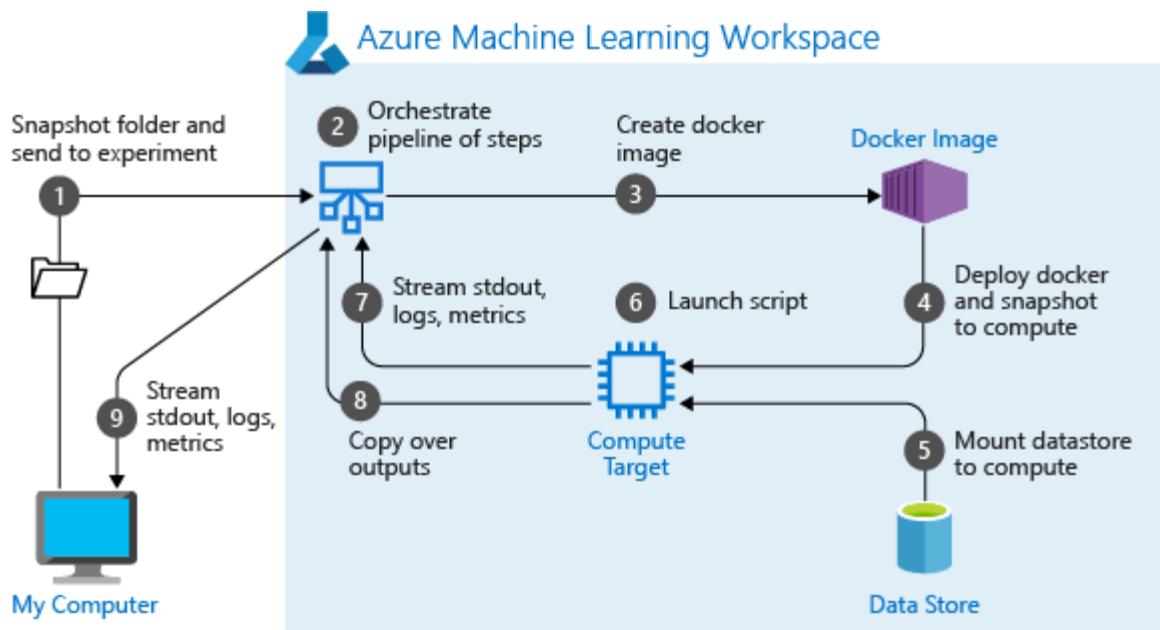
Components of an AzureML Pipeline

An AzureML pipeline has several main components:

1. **Data preparation.** This involves steps like importing, validating, cleaning, wrangling (or "munging"), transforming, normalizing, and staging your data. This step tends to be a large proportion of the work in most ML projects.
2. **Training configuration.** A typical training configuration includes steps like parameterization, file paths, logging, and reporting.
3. **Training validation.** Training validation involves repeatedly running through your experiment, picking different hardware, compute resources, doing distributed computing, and also monitoring your progress.
4. **Model deployment.** The final step is to deploy the model. This typically involves actions like versioning, scaling, provisioning, and configuring access control.

The Azure Machine Learning Workspace

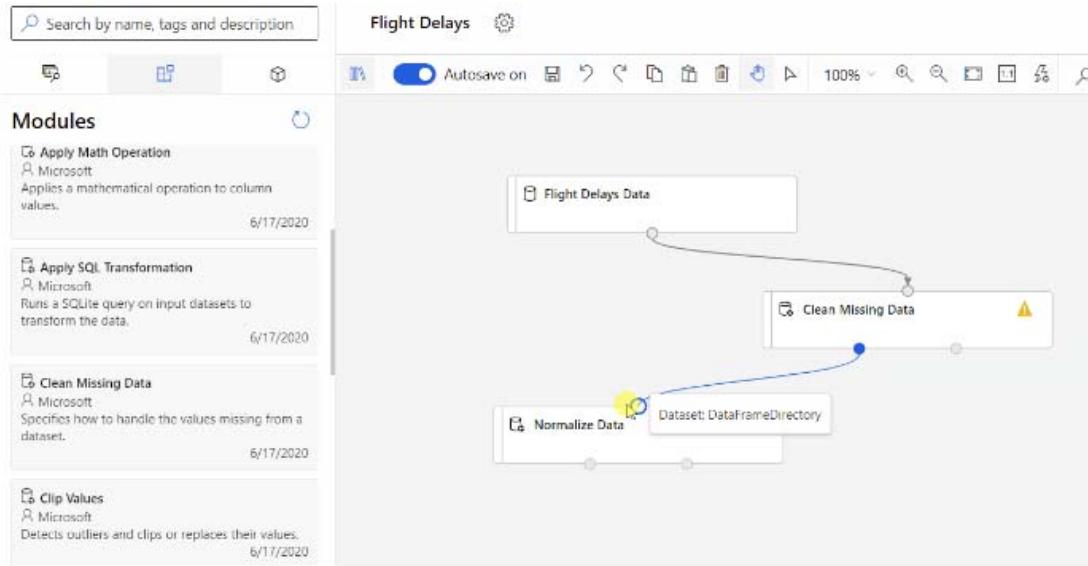
A lot of the complicated inner workings of a pipeline are automated for you behind the scenes by the Azure Machine Learning Workspace. You can see this in the diagram below:



You don't need to understand or remember all of the details shown here, but one key idea to understand is that this system supports an MLOps approach by providing a traceable [data lineage](#)—meaning, you have visibility into the process that the data has undergone, including the origin of the data and the root cause of any errors that arise.

QUESTION 1 OF 2

What tool is being used to create a pipeline in the graphic below?



- The Azure ML SDK
- The Console
- The Designer
- A notebook

QUESTION 2 OF 2

Below are the main components of an AzureML Pipeline. Can you match each with the typical actions it involves?

Submit to check your answer choices!

| TYPICAL ACTIONS | PIPELINE COMPONENT |
|---|------------------------|
| Parameterization, file paths, logging, and reporting. | Training configuration |
| Importing, validating, cleaning, transforming, normalizing, and staging data. | Data preparation |
| Picking hardware and compute resources, doing distributed computing, and monitoring progress. | Training validation |
| Versioning, scaling, provisioning, and configuring access control. | Model deployment |

Additional Resources

If you'd like to learn more about the underlying concepts of Azure ML Pipelines, we recommend you read the Microsoft Learn article, [What are Azure Machine Learning pipelines?](#)

4. Exercise: Introduction to Designer

Let's get some practice with the *Designer* by trying out one of the sample pipelines.

<https://photos.app.goo.gl/48QQUuEkHZN3mfrM9>

Try It!

The goal of this exercise is simply for you to get comfortable with the designer. You can play with any of the samples in the designer, but we recommend that you at least follow along with the video and try out the sample pipeline demonstrated there:

-  Go to **Designer** > **Show more samples** > **Regression - Automobile Price Prediction**.
-  Try clicking on different modules in the pipeline (such as **Decision Forest Regression**).
-  Set up the run by clicking **Submit**, setting a **compute target**, and setting the datastore (blob storage is fine) under **default output settings**.
-  Click **Submit** and under the **Set up pipeline run** options that pop up, select **Create new**.
-  Once the pipeline is running, switch to the **Compute** section and find the pipeline that you just started running.
-  Once the run has finished, you can click on it and explore the **Graph** and **Steps** tabs.

Running an Existing Pipeline

One of the great features available in the Designer is the ability to easily re-run an existing pipeline. Let's give this a try with the sample pipeline we just ran.

<https://photos.app.goo.gl/wSZ7rT14WAdBZEA27>

Try it!

-  Go to the **Pipelines** section and find the run you just did (above) with the **Automobile Price Prediction** pipeline.
-  If you like, you can try changing the settings or steps in the pipeline. When you're ready, simply click **Resubmit** and the pipeline will run again.



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

QUIZ QUESTION

Which of the following are true statements about the designer?

(Select all that apply.)

- It can be used to train models
- It gives a visual interface for you to see the different steps in your model training task
- It is a good way to re-run previous experiments or change parameters of an existing experiment and run it again

The Automobile Price Prediction pipeline.

The screenshot shows the Microsoft Azure Machine Learning Studio interface. At the top, there is a navigation bar with various options like 'Home', 'Pipelines', 'Datasets', etc. Below the navigation, the title 'Automobile Price Prediction' is visible. The main area displays a flowchart of the pipeline, which includes several components such as 'Get Data', 'Preprocess Data', 'Train Model', and 'Score Model'. A large circular icon in the center represents the 'Designer' component, which is highlighted in the question. Below the flowchart, there is a message: 'Remember that you can also use the Designer to get a visual depiction of the steps involved in the model training task.' Further down, another message says: 'Remember that the Designer can also be used to re-run previous experiments.' At the bottom of the pipeline view, there is a button labeled 'TRY AGAIN'.



Try Again

Remember that you can also use the Designer to **train models**.

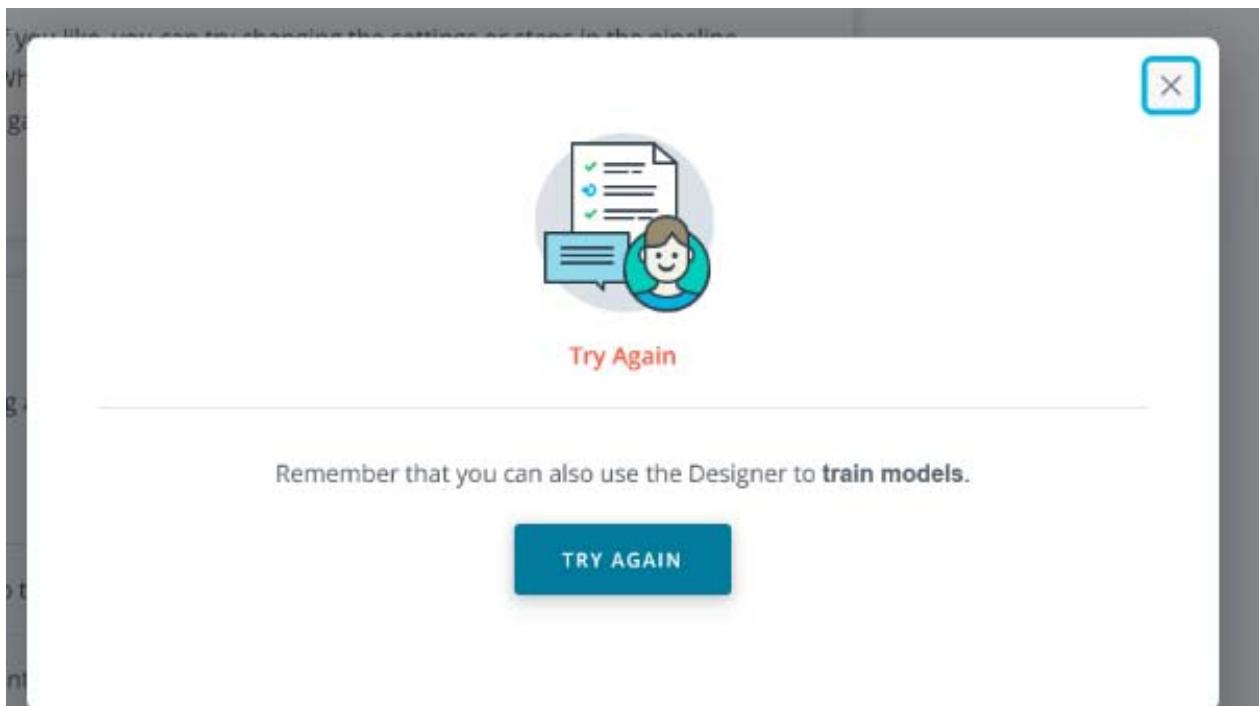
Remember that the Designer can also be used to **re-run previous experiments**.

TRY AGAIN



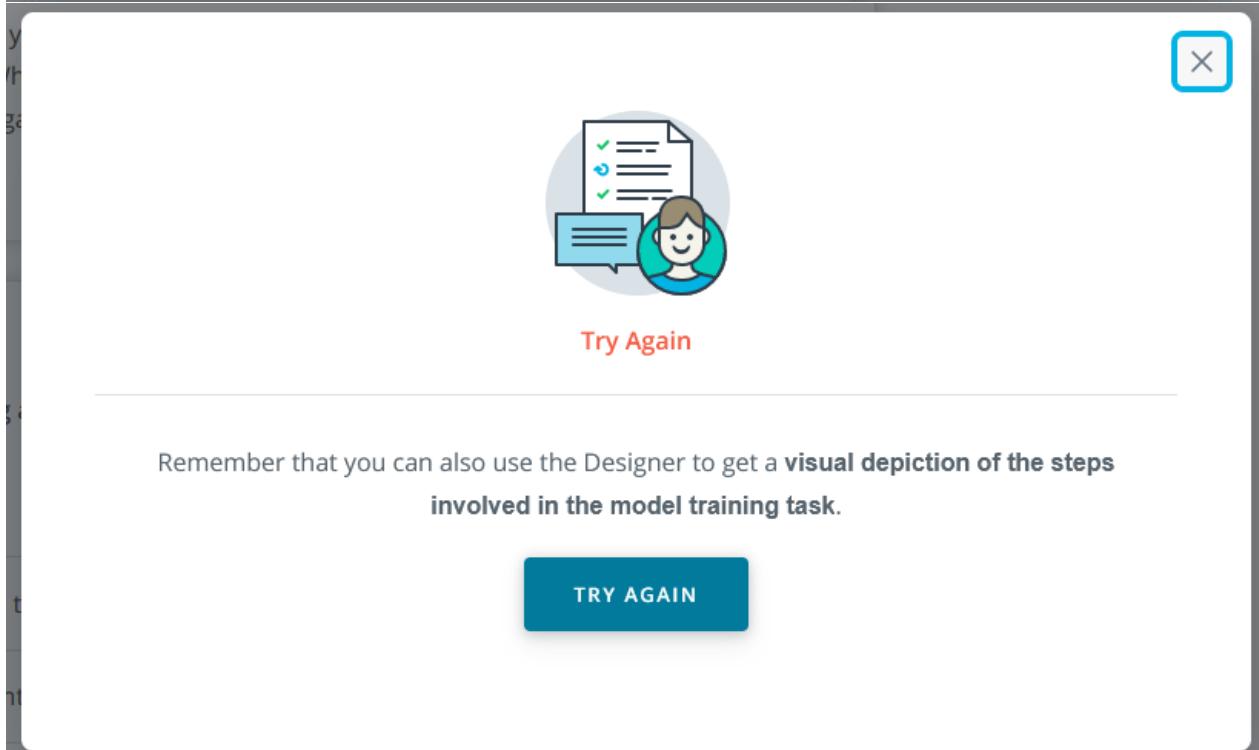
Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure



Remember that you can also use the Designer to **train models**.

TRY AGAIN



Remember that you can also use the Designer to get a **visual depiction of the steps involved in the model training task**.

TRY AGAIN

5. Hyperparameters in Experiments

<https://photos.app.goo.gl/Z7xuTLnDAUa7savb8>

Key Tasks for Using Hyperparameters in Experiments

Using hyperparameters in experiments involves the following key tasks:

1. **Define** the parameter search space.
2. **Optimize** a chosen metric.
3. **Specify termination**—meaning, define the criteria that say when the process should be terminated.
4. **Allocate resources**, such as selecting the compute clusters you'll use.
5. **Launch an experiment** using an Azure pipeline.
6. **Visualize** the results to see what you actually created.

On the next page, we'll get some hands-on practice with this process, including how to choose runs that have the best hyperparameter tuning.

QUIZ QUESTION

Below are some of the terms related to Hyperparameter tuning that we just discussed. Can you match each term with the correct description?

Submit to check your answer choices!

| DESCRIPTION | TERM |
|--|---------------------------|
| A parameter that has a fixed value. | Discrete hyperparameter |
| A parameter that can take any value in a range of values. | Continuous hyperparameter |
| A way to randomly select hyperparameters in a search space. | Random sampling |
| A way to select discrete values over a search space. | Grid sampling |
| A way to select values based on how previous values improved the training performance. | Bayesian sampling |



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure



Try Again

Remember that **Random Sampling** is a way to randomly select either discrete or continuous hyperparameters.

Remember that **Grid Sampling** can only select discrete values from a list of values.

TRY AGAIN

A way to select discrete values over a search space. Random sampling.

Random sampling.



Try Again

Remember that **Discrete Hyperparameters** can only take distinct values in a set of values.

Remember that **Continuous Hyperparameters** can be any value in a continuous distribution of values.

Remember that **Grid Sampling** can only select discrete values from a list of values.

Remember that **Bayesian Sampling** is a way to optimally select hyperparameters based on how your previous hyperparameters performed.

TRY AGAIN

A way to select values based on how previous ones performed.

Bayesian sampling.



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

6. Exercise: Hyperparameters in Experiments

<https://photos.app.goo.gl/sajxehc7E4F57Jq37>

Try It!

For this exercise, you will try to run an existing pipeline and then perform a hyperparameter search using Designer. Here are the steps you need to perform for this task:

-  Go to **Designer** and select the sample project **Tune Parameters for Binary Classification - Adult Income**.
-  Under **Settings** choose a **compute target**.
-  Select the **Tune Model Hyperparameters**. You can select **Random sweep**.
-   **Submit** the run and, when it's done, view the results under **Experiments**.
-  Change the model hyperparameters to use **Entire grid** and re-run the pipeline.

QUESTION 1 OF 2

In terms of time taken to run the pipeline, what is the distinction between using **entire grid** and **random sweep**?

- A random sweep is exhaustive, but more time-consuming. In contrast, a grid sweep can get good results without taking as much time.
- A grid sweep is exhaustive, but more time-consuming. In contrast, a random sweep can get good results without taking as much time.
- This hyperparameter has no impact on time taken to run the pipeline.



Try Again

A random sweep is not exhaustive—as the name implies, it randomly selects a subset of possible parameter values.

TRY AGAIN

QUESTION 2 OF 2

In this particular case, how was the accuracy different in running a random sweep versus the entire grid?

(If you don't want to take the time to run it both ways yourself, you can probably get by with just running the *grid sweep* and comparing it with the accuracy I got on a random sweep, which was about `.85`.)

- Using the **entire grid** resulted in a significantly (over 10%) **better** accuracy than using random sweep
- Using the **entire grid** resulted in a significantly (over 10%) **worse** accuracy than using random sweep
- There was little or no difference in the accuracy



Thanks for completing that!

Correct!

When I ran the model I had an accuracy of about .85 with the entire grid and a very similar accuracy with a random sweep. Your numbers might be a little different, but you should see a similar pattern. The take-home here is that, in many cases, the random sweep will perform essentially as well as if you had used the entire grid—even though it takes much less time.

[CONTINUE](#)



Try Again

When I ran the model I had an accuracy of about .85 with the entire grid and a very similar accuracy with a random sweep. Your numbers might be a little different, but you should see a similar pattern.

TRY AGAIN



Try Again

When I ran the model I had an accuracy of about .85 with the entire grid and a very similar accuracy with a random sweep. Your numbers might be a little different, but you should see a similar pattern.

TRY AGAIN

Additional Resources

- You can read more about the features we've explore here—including how grid sweep and random sweep work—in the [Microsoft documentation on tuning model hyperparameters](#).

7. Monitoring Models with Application Insights

<https://photos.app.goo.gl/qdWhSoHHxjR5dnVA>

Once you have deployed an ML model, you need to monitor the model's performance—which you can do using Microsoft Azure's *Application Insights*. **Application Insights** is an Application Performance Management (APM) service that is available as a feature of *Azure Monitor*.

Metrics for Monitoring Models

There are a variety of metrics you can use when monitoring your ML models with *Application Insights*. Some examples are:

- **Request rate.** The request rate allows you to figure out how many times your model is being called. If it's being called quite frequently, you may want to have a larger cluster.
- **Response time.** For example, if your application is getting very long response times and it's doing computer vision, you might need to switch to a GPU inference end point.
- **Failure rate.** If your application is failing more than it should (e.g., it has a 20% failure rate) this is something that you may be able to address with an infrastructure change or by using a different model.
- **Exceptions.** You may train a new mode and put it into production, but then find it doesn't fit with the data structure of what your request is expecting—a situation like this will generate an *exception*, which you can then address.

QUESTION 1 OF 2

Can you match the **metric name** to **what it measures**?

Submit to check your answer choices!

| METRIC | WHAT IT MEASURES |
|---|------------------|
| Number of times requests are being sent to the model | Request Rate |
| Time it takes for a request to get a response | Response Time |
| Number of times the application fails to provide a result | Failure Rates |
| Incorrect data types being fed to the model | Exceptions |

Can you match the metric name to what it measures?



[Try Again](#)

Remember that **Request Rate** is the number of requests being sent to the model.

Remember that **Response Time** is the time that a user has to wait to get a response, once a request has been sent.

Remember that **Failure Rate** is the number of times a model fails to give a prediction.

Remember that **Exceptions** happen when an incorrectly formatted input is fed to the model.

[TRY AGAIN](#)

QUESTION 5



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

QUESTION 2 OF 2

And what are some ways to improve these metrics?

Submit to check your answer choices!

| IMPROVEMENT METHOD | METRIC |
|--|---------------|
| A different cluster configuration can improve this. | Request Rate |
| A more powerful machine, or multiple machines can help improve this | Response Time |
| Changing an incorrect model or the current deployment infrastructure can help improve this | Failure Rate |
| Changing the way data is fed to the model can help improve this | Exceptions |



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure



Try Again

Remember that a low **Response Rate** can be fixed by changing the cluster configuration in which the model is deployed.

Remember that a low **Response Time** can be fixed by increasing the computing power of the cluster or by having more compute clusters.

Remember that a high **Failure Rate** can be fixed by changing the model or the cluster configuration.

Remember that **Exceptions** can be fixed by changing the way data is fed into the model.

TRY AGAIN

8. Lesson Review

<https://photos.app.goo.gl/uBaNoE4VYBT1VMot5>

Lesson Outline: Training models in Azure

This lesson was all about **training models**. Here's what we covered in this lesson:

- **Introduction to Designer.** First, we learned how we can quickly build our models using the no code/low code, drag-and-drop interface of the Designer.
- **Managing pipelines.** We covered how to create a pipeline that we manage ourselves using the console, which allows us to make very small changes that can be run

repeatedly. We saw that our goal in managing our pipelines is to do *MLOps*—that is, to apply a DevOps-based workflow to ML.

- **Running pipelines.** We learned how to run pipelines and experiments, as well as how to look at the output and tune our models.
- **Hyperparameters in experiments.** Finally, we learned how to use hyperparameters in experiments, including how we can automate the creation of hyperparameters and make very small changes that create huge value in terms of prediction accuracy.
- **Monitoring models with application insights.** Finally, we talked about how we can leverage Azure's *Application Insights* infrastructure to monitor the performance of our models in production.

Lesson 5 : The Azure ML SDK

1. Lesson Overview

<https://photos.app.goo.gl/GjwsxHcNT2YHXJz99>

Lesson Outline: Azure ML SDK

This lesson is all about the **Azure ML SDK**—and in particular, how to use the SDK to programmatically control and automate our machine learning pipelines. Here are the main topics:

- **Managing data with the SDK.** We'll get into how to manage our cloud resources with the SDK, including how to do monitoring and logging.
- **Creating pipelines.** We'll talk about how the Azure ML SDK allows us to programmatically explore, create, and manage pipelines. By automating the creation of pipelines, we can create much larger numbers of pipelines; also, our pipeline creation becomes a repeatable process that other members of our team can follow.
- **Managing experiments.** We'll learn how to use the SDK to manage experiments programmatically with Python, allowing us to easily rerun our processes using Python scripts.

2. The Azure ML SDK

This lesson is all about the **Azure ML Software Development Kit (SDK)**. The SDK is especially useful to us because it essentially connects two other incredibly useful tools:

- The **Azure ML platform**, which gives us the ability to easily **automate** our machine learning pipelines.
- The **Python ecosystem**, which has a rich open source library and allows you to greatly **customize** your applications.

The SDK allows you to combine the *automation* of the Azure ML platform with the *customizability* of the Python ecosystem to greatly increase the effectiveness, efficiency, and flexibility of your ML applications. Because the SDK is Python-based, it allows us to tap into a rich ecosystem and build a huge variety of tools, from web applications to command-line tools.

<https://photos.app.goo.gl/Lvg6jjBRFCAmYoSPA>

Azure ML Key Capabilities

- **Dataset lifecycle.** You can use the SDK to manage your dataset lifecycle. This includes organizing, monitoring, and logging your ML experiments and resources.
- **Train models.** The SDK can be used to train a model, either locally or with cloud resources (e.g., to do GPU-accelerated model training).
- **AutoML.** Using the SDK, you can leverage the power of the cloud platform to automatically iterate through algorithms and hyper parameter tunings to find the best model for running predictions.
- **Web services.** Finally, you can deploy your web service easily by converting your trained model into a RESTful service that can be consumed by any application.

Azure ML Modes

- **Stable.** The *stable* mode uses an established version to ensure you have a stable environment that you can count on. This is recommended for most use cases.
- **Experimental.** If you're an early adopter, you can use *experimental* mode, which is for advanced users who want to try out new features as they become available and are comfortable tolerating some potential bugs.

QUIZ QUESTION

Which of the following tasks can you do with the AzureML SDK?

(Select all that apply)

- Create and Manage Datasets
- Create and Manage Cloud Resources
- Train AutoML Models
- Build Pipelines
- Deploy Webservices

Additional Ressources

- For your reference, you may want to bookmark the [Azure ML SDK documentation](#).
- For more details on getting the Python SDK installed, see Microsoft's documentation on how to [Install the Azure Machine Learning SDK for Python](#).

3. Managing Data with the SDK

The Azure ML SDK allows you to automate your machine learning pipeline, and a key part of this automation is using the SDK to *manage data*. As we've mentioned, the data preparation step in an ML pipeline often makes up a large proportion (not uncommonly about 80%) of the work, so automating the management of this part of the pipeline can make it dramatically more efficient.

Note: Earlier we mentioned that a dataset can reference data in a Datastore. In addition, a dataset object can also reference a dataset using **public web URLs**. We'll see an example of this in the following video.

<https://photos.app.goo.gl/aA8rvencaeXEJKcjJ9>

Interfacing with Datasets

You can interface with Datasets via the API. How you do so depends in part on the type of Dataset you are using. If you recall from an earlier lesson, there are two **Dataset types**:

- **FileDataset.** This is a generic Dataset type. This is useful for things like computer vision or anything where you need to have a lot of flexibility.
- **TabularDataset.** As the name implies, this type is for *tabular* (i.e., table-based) data. This type allows you to handle formats like JSON, CSV, or Parquet.

We'll get some practice managing Datasets in an exercise shortly, but to get an initial idea of how this might work, here's an example of a typical piece of code for interfacing with a FileDataset:

```
from azureml.core.dataset import Dataset

url_paths = [
    'http://yann.lecun.com/exdb/mnist/train-images-idx3-ubyte.gz',
    'http://yann.lecun.com/exdb/mnist/train-labels-idx1-ubyte.gz',
    'http://yann.lecun.com/exdb/mnist/t10k-images-idx3-ubyte.gz',
    'http://yann.lecun.com/exdb/mnist/t10k-labels-idx1-ubyte.gz'
]

dataset = Dataset.File.from_files(path=url_paths)
```

This small block of Python allows us to simplify this part of the process considerably via the straightforward act of placing all the URL paths in a list and assigning them to a single variable.

Then, by instantiating the `dataset` object, we have the power to manipulate the data using built-in methods that come with the API. For example we may want to load the data into a Pandas DataFrame, which we can easily do using the `to_pandas_dataframe` method:

```
df = dataset.to_pandas_dataframe()
```

QUIZ QUESTION

Which of the following data sources are best paired with **TabularDatasets**?

(Select all that apply.)

JSON

MNIST Images

CSV files

Parquet Files

WHICH WE CAN EASILY DO USING THE TO_PANDAS DATATFRAME METHOD.



Try Again

Remember that TabularDataset is for *tabular* (table-based) data. This type allows you to handle formats like JSON, CSV, or Parquet.

TRY AGAIN

Additional Resources

- For additional details on how to use the SDK to manage your datasets, you may want to look at [the documentation for the dataset module](#).

4. [DEPRECATED] Exercise: Managing Data with the SDK

<https://photos.app.goo.gl/QE3v3GfGsYRATq8L9>

The screenshot shows a Jupyter Notebook interface with the following components:

- File Edit View Run Kernel Tabs Settings Help**: The top navigation bar.
- Guide**: A sidebar containing a title and a script.
- apiversion.py**: A code editor window containing the following Python code:

```
1 import azureml.core
```
- Terminal**: A terminal window showing a root shell prompt on a Linux system.
- Toolbar**: A vertical toolbar on the left side of the interface.
- Bottom Bar**: A dark bar with "Menu" and "Expand" buttons.

Guide Content:

Install and use the API

In this lab you will install the api and run a script that prints the API VERSION.

1. install the `azureml-sdk`

```
pip install --upgrade azureml-sdk
```

2. run the script in your workspace
 `python apiversion.py`
 It should run silently without any output.

3. add a print statement that uses the `VERSION` object that is attached to `azureml.core`

5. Exercise: Managing Data With the SDK

For this exercise, your goal will be to use the SDK to upload and register a dataset, and then retrieve some simple data from it.

In the video below, Noah gets the Dataset from Kaggle. For convenience, we've loaded it into your VM so that you don't need to do this—the dataset is available in the `Datasets` folder on the Desktop of your VM. If you are working on this in your own Azure account, you can also [download the data here](#).

<https://drive.google.com/file/d/1n-aMlU6M0rP-kNM4-ltYgHWwXhwMaC6P/view?usp=sharing>

[**https://photos.app.goo.gl/vCXu28k6QjYJW8w28**](https://photos.app.goo.gl/vCXu28k6QjYJW8w28)

The image shows two vertically stacked mobile quiz interfaces. Both screens have a light gray header bar at the top. The first screen is titled "QUESTION 1 OF 2" and asks, "What is the **median age** of all NBA players?" Below the question, there is a list of four options: "26" (selected), "23", "28", and "24". The second screen is titled "QUESTION 2 OF 2" and asks, "What is the **median salary** of all NBA players?" Below the question, there is a list of four options: "5.37M" (selected), "5.49M", "2.58M", and "7.92M". Each option is preceded by a small circular button for selection.

Supporting Materials

[Nba 2017 Players With Salary Wiki Twitter](#)

<https://drive.google.com/file/d/1XctkQBqgdJdOe4IpMyFshV2ghSbItBdI/view?usp=sharing>

6. Using the SDK in Azure Cloud Shell (Optional)

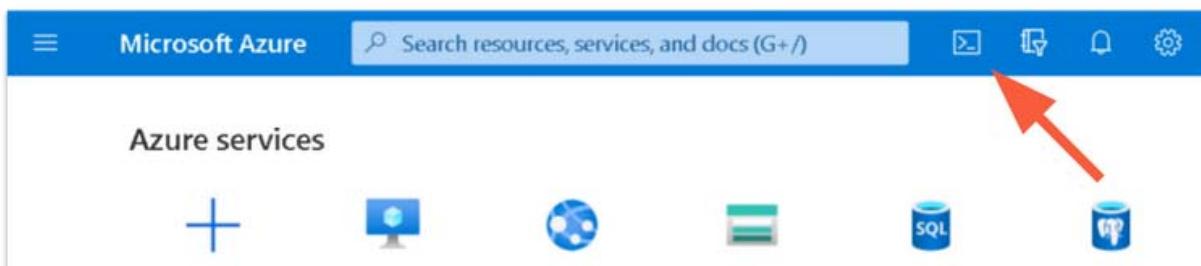
As we saw in the previous exercise, one of the most common ways to use the SDK is via a Jupyter notebook. Another approach that can also be quite useful is to use the SDK via the **Azure Cloud Shell**.

As a developer, you may want to explore SDK functionality in a Command-Line Interface where you can try out commands in the interactive prompt. By using IPython in the Azure Cloud Shell, you can easily experiment with different SDK features. An approach that some developers use is to play with a feature initially in the Cloud Shell, and then later implement it in a Jupyter Notebook.

Using the SDK in the Cloud Shell is optional for this course. If you'd like to explore it, we'll walk through the basic setup on this page. If you prefer not to explore it at this time, you can simply skip the remainder of this page and come back to it another time.

Enable Cloud Shell

The first time you use the Cloud Shell in your Azure account, you'll need to enable it. To do so, click on the **shell icon**:



Then click on **Bash > Create storage**. Once the process has completed, you should see the Command-Line Interface for Cloud Shell.

Create a Virtual Environment and Install IPython

Once you have the Cloud Shell running, the first thing you'll want to do is **create a virtual environment** for the SDK. You can do that using `venv` as follows:

```
python -m venv ~/.azure-ml-sdk
```

Then **activate the virtual environment** and **install IPython**:

```
source ~/.azure-ml-sdk/bin/activate && pip install ipython
```

Import the SDK

Now you're ready to import and access the SDK from the Cloud Shell, as demonstrated in the video below.

<https://photos.app.goo.gl/oH8ErXL93EhDorhK8>

Example: Using the SDK to Control Experiments

Now that you know the basics, you can see an example of how to use it to communicate with a section of the Azure SDK—in this case, it is used to interface with and control Experiments.

<https://photos.app.goo.gl/Q6s54Eua1JNuWJ5i6>

QUIZ QUESTION

What is one of the main reasons why a developer might want to use the SDK via the Azure Cloud Shell?

- To build complex applications only in IPython
- To quickly try out new SDK functionality
- To control experiments using a GUI

QUIZ QUESTION

Try Again

Remember that the goal here is to try out features of the SDK in an interactive prompt. Anything that involves a UX/UI or a large project isn't a good fit.

TRY AGAIN

Installing and Using the SDK with a Makefile

One of our main goals in using the Azure ML SDK is to add automation to our Python projects. One way we can do that is to use the SDK is to create an installation step with a [Makefile](#) and incorporate the Azure ML SDK with your Continuous Integration workflow.

A *Makefile* can include a command to install and upgrade packages in a `requirements.txt` file. Here's an example in Azure Cloud Shell:

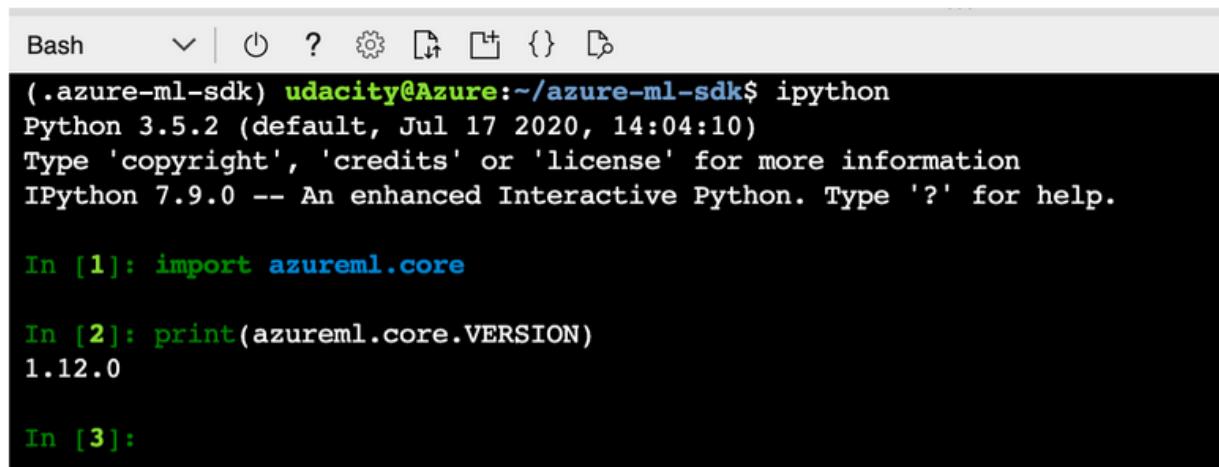
```
(.azure-ml-sdk) udacity@Azure:~/azure-ml-sdk$ cat Makefile
install:
    pip install --upgrade pip && \
        pip install --upgrade -r requirements.txt
```

The `requirements.txt` file will look like this:

```
(.azure-ml-sdk) udacity@Azure:~/azure-ml-sdk$ cat requirements.txt
ipython
azureml-sdk
```

And then, to install, you can simply run `make install` in the shell.

To test it, you can launch it from IPython and simply run `import azureml.core`, as shown in the screenshot below.



A screenshot of a terminal window titled "Bash". The window shows the following text:

```
(.azure-ml-sdk) udacity@Azure:~/azure-ml-sdk$ ipython
Python 3.5.2 (default, Jul 17 2020, 14:04:10)
Type 'copyright', 'credits' or 'license' for more information
IPython 7.9.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: import azureml.core

In [2]: print(azureml.core.VERSION)
1.12.0

In [3]:
```

7. Creating a Pipeline with the SDK

<https://photos.app.goo.gl/7v9N731UsTnWeF2WA>

We've seen previously how to set up a pipeline use the Designer—now we'll see how to do it programmatically using the SDK. Specifically, we'll look here at some example code for the *data preparation*, *training configuration*, and *training validation* steps of the pipeline.

You can find the code examples we went over below, for your reference. On the next page, we'll go through an exercise so you can get some hands-on practice.

Data Preparation

```
ws = Workspace.from_config()
blob_store = Datastore(ws, "workspaceblobstore")
compute_target = ws.compute_targets["STANDARD_NC6"]
experiment = Experiment(ws, 'MyExperiment')

input_data = Dataset.File.from_files(
    DataPath(datastore, '20newsgroups/20news.pkl'))
```

Training Configuration

```
output_data = PipelineData("output_data", datastore=blob_store)

input_named = input_data.as_named_input('input')

steps = [ PythonScriptStep(
    script_name="train.py",
    arguments=["--input", input_named.as_download(),
               "--output", output_data],
    inputs=[input_data],
    outputs=[output_data],
    compute_target=compute_target,
    source_directory="myfolder"
) ]

pipeline = Pipeline(workspace=ws, steps=steps)
```

Training Validation

```
pipeline_run = experiment.submit(pipeline)
pipeline_run.wait_for_completion()
```

QUESTION 1 OF 3

Can you identify what the function of the following code snippet is?

```
Dataset.File.from_files(DataPath(datastore, '20newsgroups/20news.pkl'))
```

```
<          >
```

- It is used to select a compute target to run our experiment in
- It is used to initialise a `FileDataset` from a datastore.
- This is the way to submit an experiment
- This is a series of actions to take in our pipeline job

pipeline.run.wait_for_completion()



Try Again

Remember that `Dataset.File.from_files()` is used to create a `FileDataset` object that can later be used for training our models

TRY AGAIN

```
input_data = Dataset.File.from_files(DataPath(datastore,  
'20newsgroups/20news.pkl'))
```

QUESTION 2 OF 3

Can you identify what the function of the following code snippet is?

```
pipeline_run = experiment.submit(pipeline)
```

- It is used to select a compute target to run our experiment in
- It is used to initialise a `FileDataset` from a datastore.
- This is the way to submit an experiment
- This is a series of actions to take in our pipeline job



Try Again

Remember that we can use the `submit` method of our `experiment` object to submit a pipeline to run.

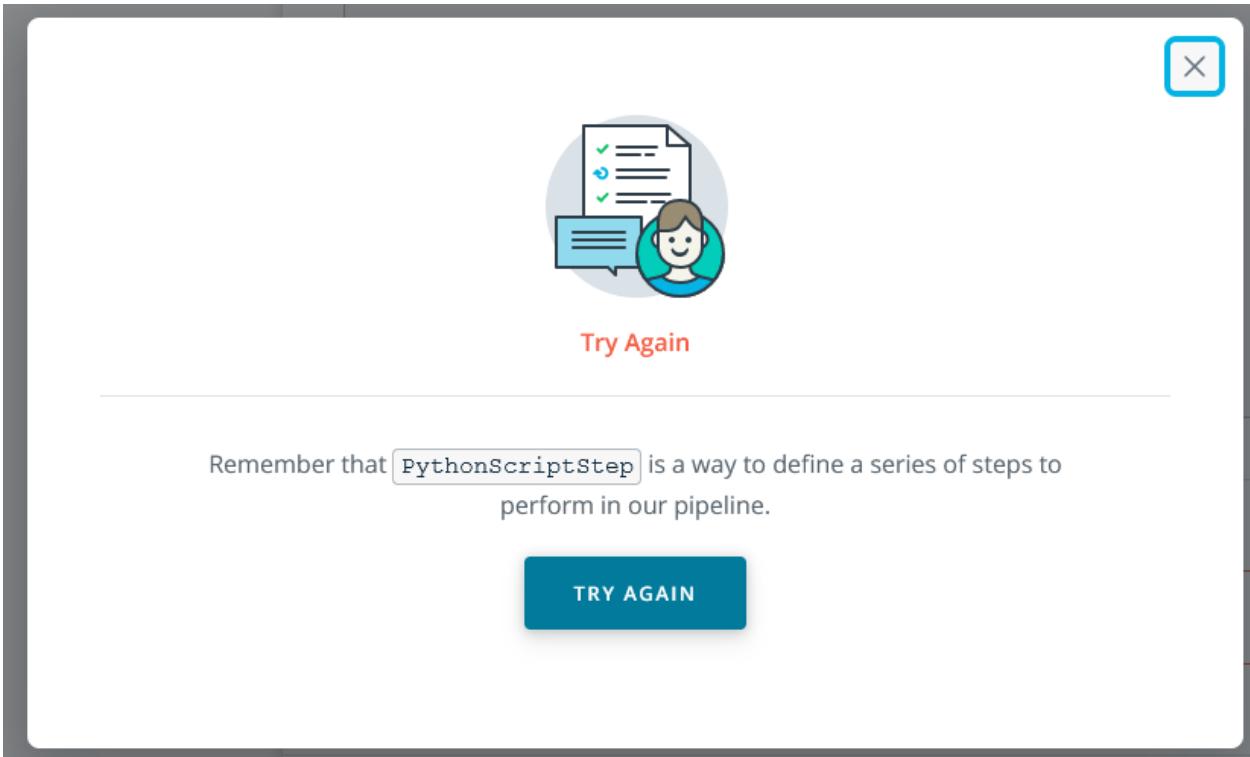
TRY AGAIN

QUESTION 3 OF 3

Can you identify what the function of the following code snippet is?

```
steps = [ PythonScriptStep(  
    script_name="train.py",  
    arguments=["--input", input_named.as_download(),  
              "--output", output_data],  
    inputs=[input_data],  
    outputs=[output_data],  
    compute_target=compute_target,  
    source_directory="myfolder"  
) ]
```

- This is a series of actions to take in our pipeline job
- This is the way to submit an experiment
- It is used to initialise a `FileDataset` from a datastore.
- It is used to select a compute target to run our experiment in



8. Managing Experiments with the SDK

Earlier, you learned how to use the Designer to view the results of your previous runs. If we manage our experiments with the SDK, then we can also access this same data—but in a way that gives us greater control and increased functionality. This involves two main tasks: **Listing past experiments** and **submitting new experiments**.

1. Listing Past Experiments

To do this, we need to pass in a workspace object, and then list the collection of trials. This will give us data very similar to the list we've accessed previously using the Designer. Here's the example code we looked at:

```
from azureml.core.experiment import Experiment

# First, we pass in the workspace object `ws` to the `list` method
# and it returns a Python list of `Experiment` objects.
list_experiments = Experiment.list(ws)

# If we print the contents of the variable,
# we can see the list of all the experiments
# that have been run in that workspace.
print(list_experiments)
[Experiment(Name: dataset_profile,
    Workspace: Azure-ML-Workspace), Experiment(Name: binary-classification,
    Workspace: Azure-ML-Workspace), Experiment(Name: regression,
    Workspace: Azure-ML-Workspace), Experiment(Name: sfautoml]
```

2. Submitting New Experiments

To submit a new experiment (such as if we wanted to try a different model type or a different algorithm), we would again pass in a workspace object and then submit the experiment. Here's the example code:

```
from azureml.core.experiment import Experiment

experiment = Experiment(ws, "automl_test_experiment")
run = experiment.submit(config=automl_config, show_output=True)
```

QUIZ QUESTION

Which of the following are true statements about the SDK?

(Select all that apply)

- You can list both running and previously run experiments
- You can use the SDK to manage experiments and even run new or previously run experiments
- You can use the SDK to change details of the user who ran an experiment
- You can use the SDK to get details of previously run experiments like the accuracy and model hyperparameters



Try Again

Remember that the SDK can be used to re-run previous experiments, or even create new experiment runs

It is not possible to change the details of the person who ran the experiment using the SDK

Remember that the SDK can be used to get view details of previous experiments. These can include data like who ran the experiment and how long it ran as well as details about the model accuracy and parameters

TRY AGAIN

9. Exercise: Managing Experiments with the SDK

In this exercise, you'll use the SDK in a Jupyter notebook to manage experiments. You will do this by first connecting the Azure workspace, and then passing that into the Experiments API.

<https://photos.app.goo.gl/GgtNZdwsXNrpsgc59>

For this exercise, you can do everything in the default Jupyter notebook provided by Azure. To open it, go to **Compute** and, under **Application URI** click on **Jupyter**.

The screenshot shows the Azure Compute blade. At the top, there are tabs for 'Compute instances', 'Compute clusters', 'Inference clusters', and 'Attached compute'. Below these are buttons for '+ New', 'Refresh', 'Start', 'Stop', 'Restart', 'Delete', 'View quota', and a toggle for 'Show created by me only'. A red arrow points to the 'Application URI' column in the table below. The table has columns for 'Name', 'Status', 'Application URI', and 'Virtual machine size'. One row is selected, showing 'jupyter-demo' with 'Running' status, and 'JupyterLab', 'Jupyter' (which is highlighted in blue), and 'RStudio' under 'Application URI', and 'STANDARD' under 'Virtual machine size'.

| Name | Status | Application URI | Virtual machine size |
|--------------|---------|----------------------------|----------------------|
| jupyter-demo | Running | JupyterLab Jupyter RStudio | STANDARD |

Then, go to the **AzureML Samples** tab, and open **How to Use AzureML > Machine Learning Pipelines > Intro to Pipelines**. This is a sample project that we will **Clone** and look through to see how we can use a notebook to manage an experiment.

<https://photos.app.goo.gl/Y1thCNZ5h8UNRZw96>

In changing the number of cross-validations from 6 to 5, you'll likely see very little difference. Cross-validation is a helpful technique to prevent overfitting, but there may be marginal or no benefits to changing the number of folds by one increment up or down.

Try it!

Here are the *key tasks* you should follow along with in the video:



- import the Workspace: `from azureml.core import Workspace`.
- Make a `ws` object through `Workspace.get()`. Remember to pass in the `name`, `subscription_id`, and `resource_group`.
- List your experiments through `Experiment.list`.
- List all the active experiments.
- Retrieve previous runs of an experiment.

QUESTION 1 OF 3

How can you use the AzureML SDK to view your current experiments?

`Experiment.show(ws)`

`Experiment.print_active_experiments(ws)`

`Experiment.list(ws)`

`Experiment.active(ws)`



Try Again

Remember that you can use `Experiment.list(ws)` to list all your active experiments, where `ws` is an instance of your workspace.

TRY AGAIN

QUESTION 2 OF 3

What can you do with the list of your current experiments?

(Select all that apply.)

- Change or manipulate the experiment settings
- Resubmit the experiments
- See the name of the experiments
- See a list of previous runs of the experiment



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

Thanks for completing that!

Correct!

By listing all the current experiments, you can manipulate the experiment settings, resubmit them, and see previous runs of the experiment. You can see an extensive documentation of all that you can do [here](#)

CONTINUE

Managing Experiment Runs in Notebooks

In this next part of the exercise, we'll look at how we can manage and edit experiment runs using the SDK in a Jupyter notebook.

<https://photos.app.goo.gl/H4hBLsB8tUSeTPsk6>

QUESTION 3 OF 3

What are some of the advantages of using the SDK?

(Select all that apply)

- It makes it easier to programmatically edit runs
- It can be used to add and query for tags of experiments
- It can be used to view the status of all experiment runs (past and previous) as well as see who ran them
- It can be used to automate multiple tasks and runs



Try Again

That is correct, but there is at least one more advantage you haven't yet selected.

Remember that the SDK is a powerful tool that can be used to do many tasks that may be tedious or difficult using the GUI. These include programmatically editing runs, adding and querying for tags, viewing statuses of experiments, and automating tasks.

TRY AGAIN

Additional Resources

- If you'd like to read more, you can see [an extensive documentation of all the things Experiments can do here](#).

10. Edge Cases: SDK Versioning

<https://photos.app.goo.gl/wGMHrixxdEhiCA1V7>

One potential issue or "gotcha" that can come up when you're using tools like Jupyter Notebooks or the Azure Cloud Shell is that you may not really know what versions your code is using. Things may run fine for a while and then suddenly stop working because of an issue with the version of one of the libraries being used by your code.

One way you can address this problem is by pinning your dependencies so that your code always uses a specific known version. For example:

- Jupyter notebooks can do this by using a pip package that has a specific version
- Azure Cloud Shell can do this by using a requirements file

For example, here's what that might look like in a `requirements.txt` file for a Python project:

```
Flask==1.0.2
pandas==0.24.2
scikit-learn==0.20.3
pylint
```

The first three packages are specifically pinned to a version number. The last package, pylint, isn't.

QUIZ QUESTION

What is a key reason to use `pip` package version numbers?

It locks your project to a known working version of Python packages

It will auto-update versions

All conflicts get resolved behind the scenes automatically

`pandas==0.24.2`



[Try Again](#)

Remember that there is nothing magic about version numbers. It locks your project to a specific revision of the library and prevents auto-updating of versions.

[TRY AGAIN](#)

Additional Resources

If you're not too familiar with pinning dependencies, you might want to check out the helpful blogpost by Dustin Ingram, [Pin all dependencies \(& let pip sort 'em out\)](#).

11. Lesson Review

<https://photos.app.goo.gl/wzNxLm3U7MXWJhGM8>

Lesson Outline: Azure ML SDK

This lesson was all about the **Azure ML SDK**—and in particular, how to use the SDK to programmatically control and automate our machine learning pipelines. Here are the main topics we covered:

- **Managing data with the SDK.** We got into how to manage our cloud resources with the SDK, including how to do monitoring and logging.
- **Creating pipelines.** We talked about how the Azure ML SDK allows us to programmatically explore, create, and manage pipelines. By automating the creation of pipelines, we can create much larger numbers of pipelines; also, our pipeline creation becomes a repeatable process that other members of our team can follow.
- **Managing experiments.** We learned how to use the SDK to manage experiments programmatically with Python, allowing us to easily rerun our processes using Python scripts.

Lesson 6 : Automated ML and Hyperparameter tuning

1. Lesson Overview

<https://photos.app.goo.gl/BbraxscVCbv2dEbq8>

Lesson Outline: AutoML and Hyperparameter Tuning

In this lesson, we'll see how we can use **Automated ML and Hyperparameter tuning** to dramatically speed up the development of our models. Here's what we'll be covering:

- **Automated ML and SDK.** We'll discuss how we can use the Azure ML SDK with AutoML to operationalize and automate model creation—allowing you to create models more quickly and accurately.
- **Model Interpretation.** We'll explore how model interpretation allows us to build solutions using AutoML that we (and others) can more easily interpret, making visible the core features that are driving the model.
- **Exporting Models with ONNX.** ONNX is a portability platform for models that was created by Microsoft and that allows you to convert models from one framework to another, or even to deploy models to a device (such as an iOS or Android mobile device).

2. Hyperparameter Tuning with HyperDrive

In this section, we'll look at how we can do **hyperparameter tuning** using **HyperDrive**.

Hyperparameters are simply adjustable parameters you choose for model training.

HyperDrive is a package that helps you automate choosing parameters.

Let's have a look.

<https://photos.app.goo.gl/cyL8bcNRv1pcsDU46>

Main Steps for Tuning with HyperDrive

- **Define the parameter search space.** This could be a *discrete/categorical* variable (e.g., apple, banana, pair) or it can be a *continuous* value (e.g., a time series value).
- **Define the sampling method over the search space.** This is a question of the method you want to use to find the values. For example, you can use a *random*, *grid*, or *Bayesian* search strategy.
- **Specify the primary metric to optimize.** For example, the Area Under the Curve (AUC) is a common optimization metric.
- **Define an early termination policy.** An early termination policy specifies that if you have a certain number of failures, HyperDrive will stop looking for the answer.

Note that to use HyperDrive, you must have a custom-coded machine learning model. Otherwise, HyperDrive won't know what model to optimize the parameters for!

Controlling HyperDrive with the SDK

You can control HyperDrive with the SDK. Here is the example code we looked at in the video:

```
from azureml.train.hyperdrive import BayesianParameterSampling
from azureml.train.hyperdrive import uniform, choice
param_sampling = BayesianParameterSampling( {
    "learning_rate": uniform(0.05, 0.1),
    "batch_size": choice(16, 32, 64, 128)
})
```

We also saw that we can specify whether we are tuning a *discrete* or *continuous* variable.

Discrete example:

```
{  
    "batch_size": choice(16, 32, 64, 128)  
    "number_of_hidden_layers": choice(range(1,5))  
}
```

Continuous example:

```
{  
    "learning_rate": normal(10, 3),  
    "keep_probability": uniform(0.05, 0.1)  
}
```

And finally, we will need to find the best model parameters. Here's an example:

```
best_run = hyperdrive_run.get_best_run_by_primary_metric()  
best_run_metrics = best_run.get_metrics()  
parameter_values = best_run.get_details()['runDefinition']['Arguments']  
  
print('Best Run Id: ', best_run.id)  
print('\n Accuracy:', best_run_metrics['accuracy'])  
print('\n learning rate:', parameter_values[3])  
print('\n keep probability:', parameter_values[5])  
print('\n batch size:', parameter_values[7])
```

The ultimate result is that we are able to choose the best tuning and use it in our final machine learning model.

QUESTION 1 OF 2

What is one of the main reasons to perform hyperparameter tuning?

- It can be used to create multiple different models that can be deployed separately.
- It can be used to automate the process of training multiple models with different parameters to find the best model.
- It requires less compute time and compute resources to train a model.
- Hyperparameter tuning can find the best model for any dataset.

model.



Try Again

Remember that hyperparameter tuning only makes the job of training multiple models with different parameters easier and faster (and it does it in a smart way). It is not a way to create multiple models (even though multiple models are trained) and it does not find the best parameters for a task.

TRY AGAIN

QUESTION 2 OF 2

Which of the following are the different aspects of HyperDrive you have to consider when using it to perform hyperparameter tuning?

(Select all that apply)

Define a search space consisting of continuous and/or discrete parameters

Define a sampling method

Specify the different model types

Specify a primary metric to optimize

Specify an early termination policy



Try Again

Remember that when we perform hyperparameter tuning, we need to specify a **search space** which includes all the different parameters we want to perform our search over. Parameters can be continuous values or discrete values.

A **primary metric to optimize** needs to be selected when training the model.

An **early termination policy** stops the training process if a certain condition is met.

TRY AGAIN

Additional Resources

- For more details and code examples see the article, [Tune hyperparameters for your model with Azure Machine Learning](#).

3. Exercise: Hyperparameter Tuning with HyperDrive

On this page, we'll get some hands-on practice with hyperparameter tuning. We'll do this first in the Designer, and then again using the SDK.

To find the sample project used in this next video, go to **Designer > Show more samples > Tune Parameter for Binary Classification - Adult Income**.

<https://photos.app.goo.gl/avcroBbxhbLEcm5J7>

QUESTION 1 OF 2

When I ran this using random sweep, I got an accuracy of about .

How does this compare to the run you did using the entire grid?

- Using the **entire grid** resulted in a significantly (over 10%) **better** accuracy than using random sweep
- Using the **entire grid** resulted in a significantly (over 10%) **worse** accuracy than using random sweep
- There was little or no difference in the accuracy



Thanks for completing that!

Correct!

When I ran mine, I got about the same accuracy (around .85) with both models. Your numbers might be a little different from mine, but you should see a similar pattern—as we saw when we did something similar in an earlier lesson, we get very little difference between an exhaustive search and a random sweep.

CONTINUE

You will have to train the model again using a different set of



Try Again

When I ran mine, I got about the same accuracy (around .85) with both models. Your numbers might be a little different from mine, but you should see a similar pattern—as we saw when we did something similar in an earlier lesson, we get very little difference between an exhaustive search and a random sweep.

TRY AGAIN



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

Try It!

For this exercise, you will have to train the model again using a different set of hyperparameters and compare the accuracy. Here are the steps you need to complete for this task:



- Train the model again using "entire grid"
- Select a new accuracy metric AUC
- How does this change the accuracy of the model?
- Experiment with different hyperparameters. How does it affect the model accuracy?

Hyperparameter Tuning with the SDK

In the above exercise, we performed our tuning using the designer. We can also do this using a notebook and the Azure ML SDK, which we'll demonstrate in this next video.

If you'd like to follow along, you can open the sample project used in the video. To do so, go to **Notebooks > Samples > 1.21.0 > tutorials** and open the tutorial-1st-experiment.ipynb notebook. Then click the **Clone this notebook** button.

Notebooks

Files Sample notebooks

Search to filter notebooks

Samples

1.21.0

how-to-use-azureml

tutorials

create-first-ml-experiment

imgs

tutorial-1st-experiment...

Want to start editing? **Clone this notebook**

Copyright (c) Microsoft Corporation. All rights reserved.

Tutorial: Train your first model

This tutorial is **part two of a two-part tutorial series**. In the previous tutorial, you created a workspace and chose a development environment. In this tutorial, you learn the foundational design patterns in Azure Machine Learning service, and train a simple scikit-learn model based on the diabetes data set. After completing this tutorial, you will have the practical knowledge of the SDK to scale up to developing more-complex experiments and workflows.

In this tutorial, you learn the following tasks:

“

- Connect your workspace and create an experiment
- Load data and train a scikit-learn model
- View training results in the studio
- Retrieve the best model

<https://photos.app.goo.gl/jP18aMfqdAfWejS8>

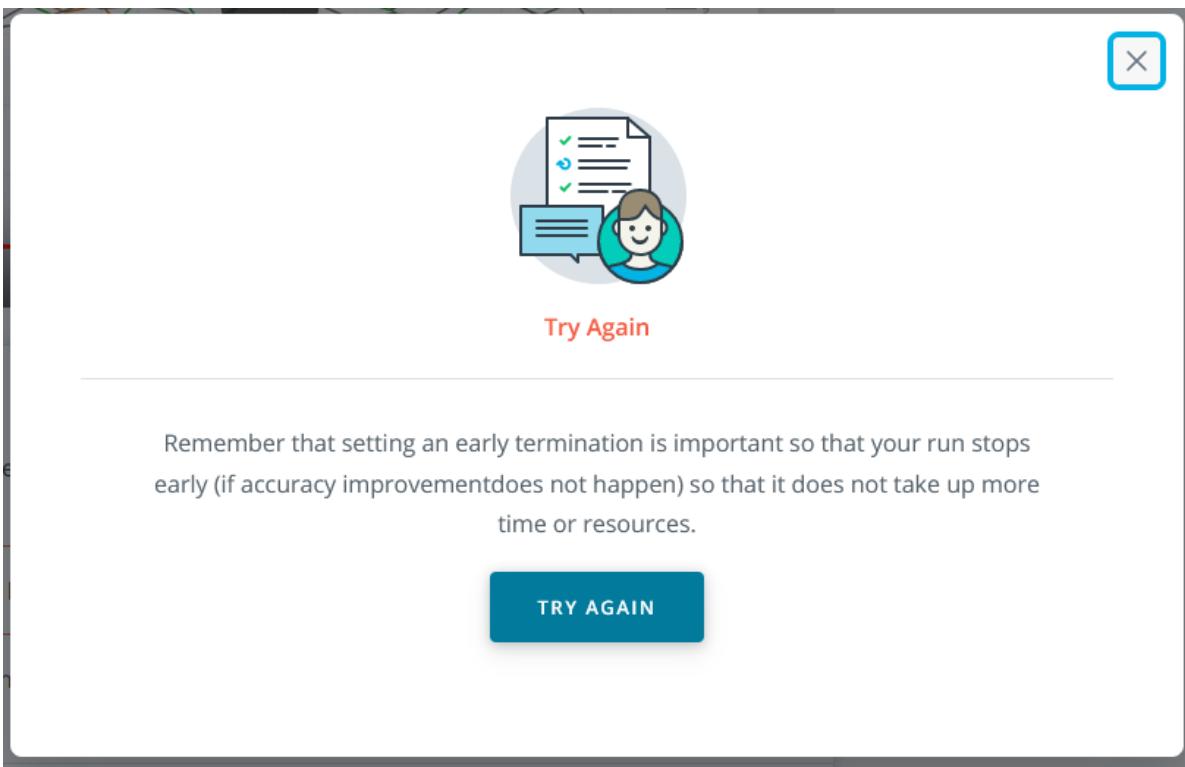
QUESTION 2 OF 2

Why do you need to select an early termination policy?

Early termination helps improve accuracy.

Early termination prevents experiments from running for a long time and using up resources.

Early termination is a way to create a checkpoint and stop the training (and then continue from the checkpoint at a later time).



4. Introducing Automated Machine Learning

In this lesson, we'll get into **Automated Machine Learning** (which we will sometimes refer to as *AutoML*). We'll start by first considering some of the issues that come up in traditional machine learning, so that we can then discuss how automated machine learning addresses these issues.

<https://photos.app.goo.gl/Y5zz9z3mzEXbV1767>

Traditional ML

To understand why Automated ML is a useful tool, it helps to first understand some of the challenges we face with traditional ML. These include:

- **Focus on technical details vs the business problem.** The code and technical details can consume large amounts of the available resources, distracting our focus from the business problem we want to use the ML to solve.
- **Lack of automation.** With traditional ML, we have to do many things manually, even though they could easily be automated with tools like Azure ML Studio.

- **Too much HiPPO influence.** The Highest Paid Person's Opinion (HiPPO) can have an unduly large influence on decisions about the output of the model, even though this decision might be better made automatically.
- **Feature engineering.** What are the features that I need to get the best accuracy? What are the columns I should select? This can be a huge task that requires a lot of human effort.
- **Hyperparameter selection.** For example, with a clustering model, what number of clusters will give the best results? There can be a lot of trial and error and many false starts.
- **Training and Tuning.** What are the different parameters you're using when training your model? What machines and resources should you use? How should you best tune the parameters? In traditional ML, these questions require a human to supervise the process.

Automated ML

Automated ML can help with all of the above problems. Essentially, AutoML involves the application of DevOps principles to machine learning, in order to automate all aspects of the process. For example, we can automate feature engineering, hyperparameter selection, model training, and tuning. With AutoML, we can:

- Create hundreds of models a day
- Get better model accuracy
- Deploy models faster

This creates a quicker feedback loop and allows us to bring ideas to market much sooner. Overall, it reduces the time that we have to spend on technical details, allowing for more effort to be put into solving the underlying business problems.

QUIZ QUESTION

Have a look at [this page](#), which describes how Microsoft is applying AutoML to a variety of problems.

To which of the following problems is Microsoft AutoML being applied?

(Select all that apply.)

Model compression

Feature engineering

Hyperparameter tuning

Model selection

Neural architecture search

Additional Resources

- If you are interested in doing a much deeper dive on the types of problems solved by AutoML, you may want to check out [this video from Microsoft Research](#).

5. Exercise: Introducing AutoML

<https://photos.app.goo.gl/3Ps5bQ79ZbZJwVZd8>

For this exercise, your task is to use AutoML to train a model on the NBA dataset. Here are the steps that you need to do:

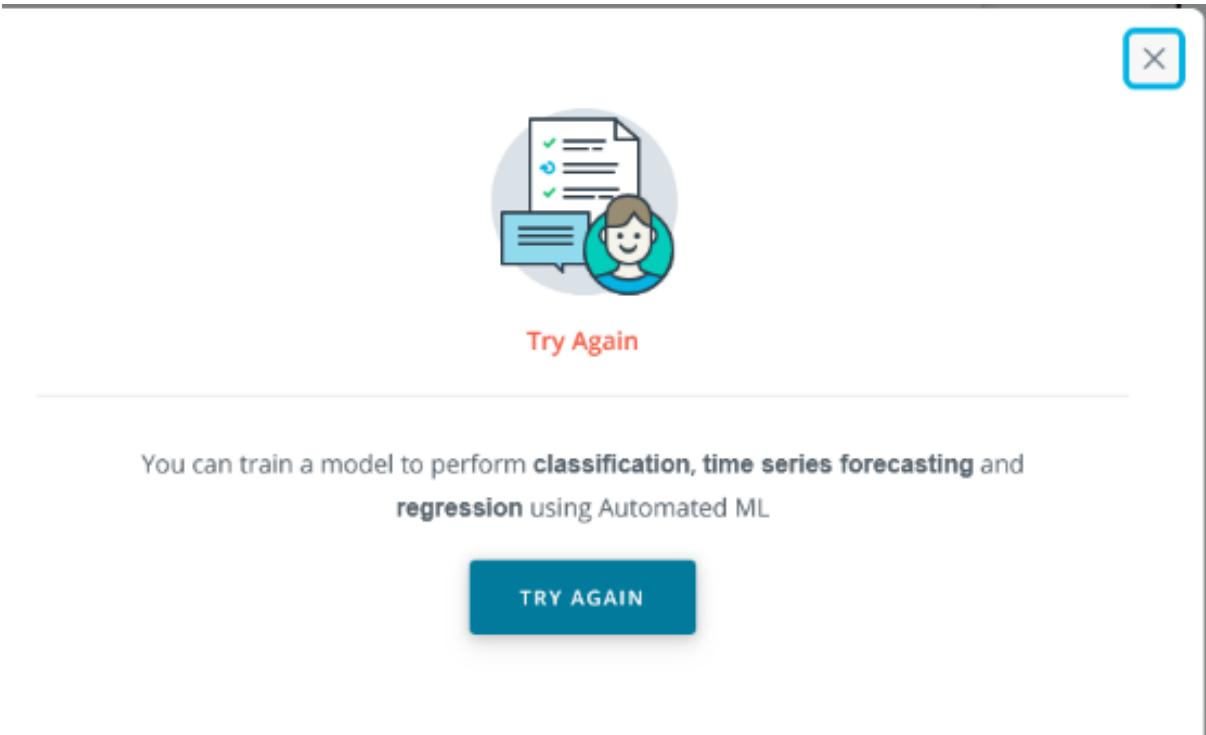
-  Go to the **Datasets** section and open the pre-existing **nba-dataset**.
-  Create a new Automated ML Run.
-  Choose the NBA dataset.
-  Run the training job.

QUIZ QUESTION

Which of the following are the different tasks supported by Automated ML?

(Select all that apply.)

- Classification
- Natural Language Generation
- Classification using Deep Learning
- Time Series Forecasting
- Regression
- Image generation



6. AutoML and the SDK

<https://photos.app.goo.gl/pnUfdJHQ89uj78U97>

Configuring AutoML from the SDK

We can easily leverage AutoML from the SDK to automate many aspects of our pipeline, including:

- Task type
- Algorithm iterations
- Accuracy metric to optimize
- Algorithms to blacklist/whitelist
- Number of cross-validations
- Compute targets
- Training data

To do this, we first use the `AutoMLConfig` class. In the code example below, you can see that we are creating an `automl_config` object and setting many of the parameters listed above:

```
from azureml.train.automl import AutoMLConfig

automl_config = AutoMLConfig(task="classification",
                             X=your_training_features,
                             y=your_training_labels,
                             iterations=30,
                             iteration_timeout_minutes=5,
                             primary_metric="AUC_weighted",
                             n_cross_validations=5
                            )
```

Running AutoML from the SDK

Once we have completed our configuration, we can then run it using the SDK. Here's a typical example of what that would look like:

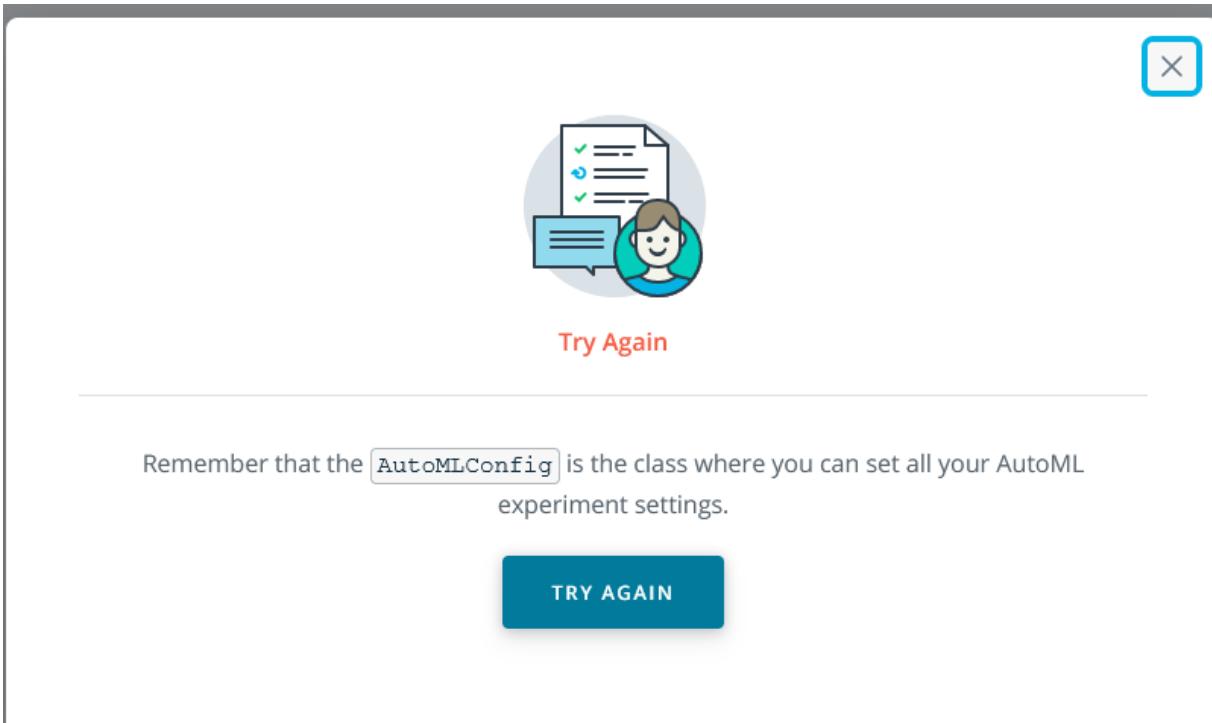
```
from azureml.core.experiment import Experiment

experiment = Experiment(ws, "automl_test_experiment")
run = experiment.submit(config=automl_config, show_output=True)
```

QUIZ QUESTION

What is the reason for creating an `AutoMLConfig`?

- It is the place where you have to specify your training and testing data
- It provides a single place where you can provide all the details of your AutoML experiment run
- It is the place where you have to specify the types of models you want to train



Additional Resources

- For more details on using the SDK with AutoML, see the [Microsoft documentation on configuring automated ML experiments in Python](#).

7. Model Interpretation in Azure ML

<https://photos.app.goo.gl/sksqoGGqzpm2kStE6>

The Azure ML platform has some core tools that make it easy to understand model behavior by looking at a previously run experiment. For example, we can easily get visualizations that help us interpret why a model is making the prediction it is making, or identify and select for the best features.

QUESTION 1 OF 2

Which of the following are reasons you might want to do model interpretation?

(Select all that apply.)

- To understand why the model made certain predictions
- It can be used to select the best model for a task
- It can be used to select the best features for a task
- It can be used to get an idea of how important a feature is for a task



Try Again

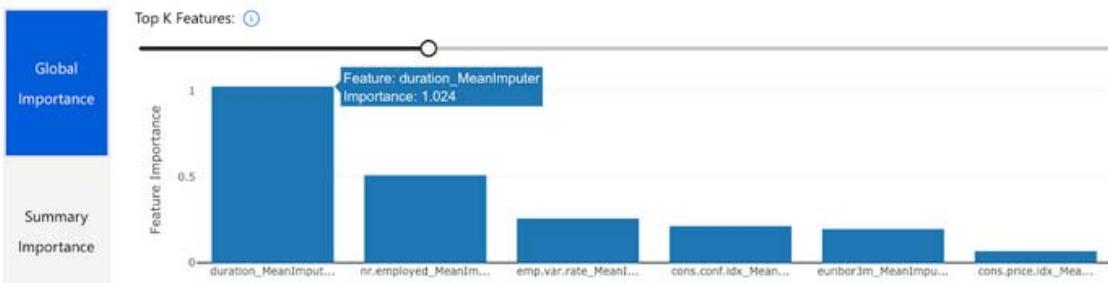
Remember that model interpretability is useful because it can help us understand why a model **made a certain prediction** as well as help us get an idea of the **importance of individual features** for tasks.

It is *not* really applicable for **selecting the best model for a task**.

TRY AGAIN

QUESTION 2 OF 2

What is this visualization showing?



- The importance of each of the top features that contributed to the best model.
- The importance of each of the bottom features that contributed to the best model.
- The importance of each of the features, including how much each feature contributed to negative vs. positive classifications.

This visualization shows the *top* (i.e., most important) contributors, not the *bottom* contributors.

TRY AGAIN



Try Again

That's close—but this visualization doesn't show the negative and positive classification contributions. You can get that information under the *Summary Importance* tab.

TRY AGAIN

Additional Resources

- For more details on model interpretability using the Azure ML SDK, you may want to read, [Use the interpretability package to explain ML models & predictions in Python.](#)

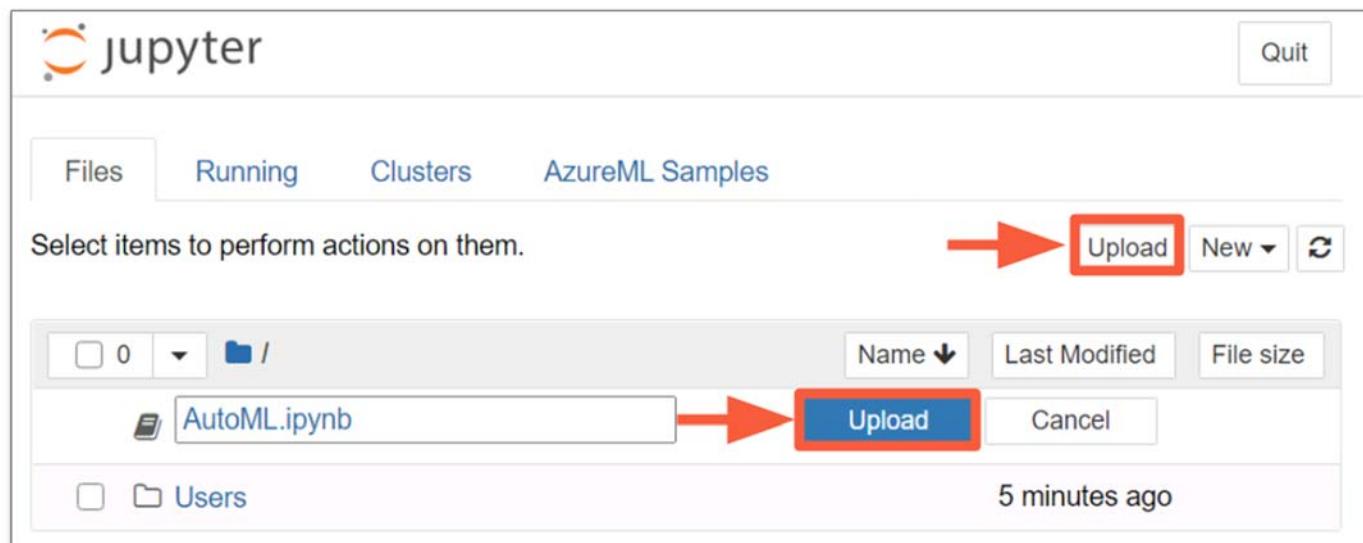
8. Exercise: AutoML and the SDK

Typical AutoML Workflow

In this exercise, we'll create an AutoML run using the SDK in a Jupyter Notebook. We've provided a notebook for you to play with, which you can upload from the VM's Desktop. This exercise also uses the NBA dataset we used earlier, which has been preloaded into the lab.

To get the Jupyter notebook uploaded, first go to **Compute** and select **Compute Instances**. You should see a compute instance already present, though Azure ML may still be creating it (this can take a few minutes). Once the compute instance is running, under **Application URI**, select **Jupyter**.

Finally, from the Jupyter interface, select **Upload > Desktop > Notebooks > AutoML.ipynb**. Then click the **Upload** button next to the notebook's name.



Once you have the notebook open, you're ready to follow along with Noah in the video below.

⚠ Note that in the first cell of the notebook, you will need to enter your own information for `subscription_id`, `resource_group`, and `workspace_name` (or you will get an error when running the cell). The easiest way to get this info is to go to the dataset (**Datasets > Nba_Dataset**) and then click on **Consume**. From here, you can copy all of the info you need to import the dataset.

After you start the run in the video below, it may take a while for it to complete before you can view the results (this is normal). When it is done, the status will change from *Running* to *Completed*.

<https://photos.app.goo.gl/CVxwhiEkDwc2EbUe8>

Before going further, be sure you've followed along and tried doing the following things from the notebook:

- Upload and open the AutoML.ipynb notebook
- Edit the first cell to enter your own information for `subscription_id`, `resource_group`, and `workspace_name`.
- In each cell, read through the code and then run the cell; you'll see that we are doing many of the same things we did when we ran AutoML earlier, but now controlling these functions programmatically.

QUESTION 1 OF 3

Which of the following metrics are supported by Automated ML for a classification task?

(Select all that apply.)

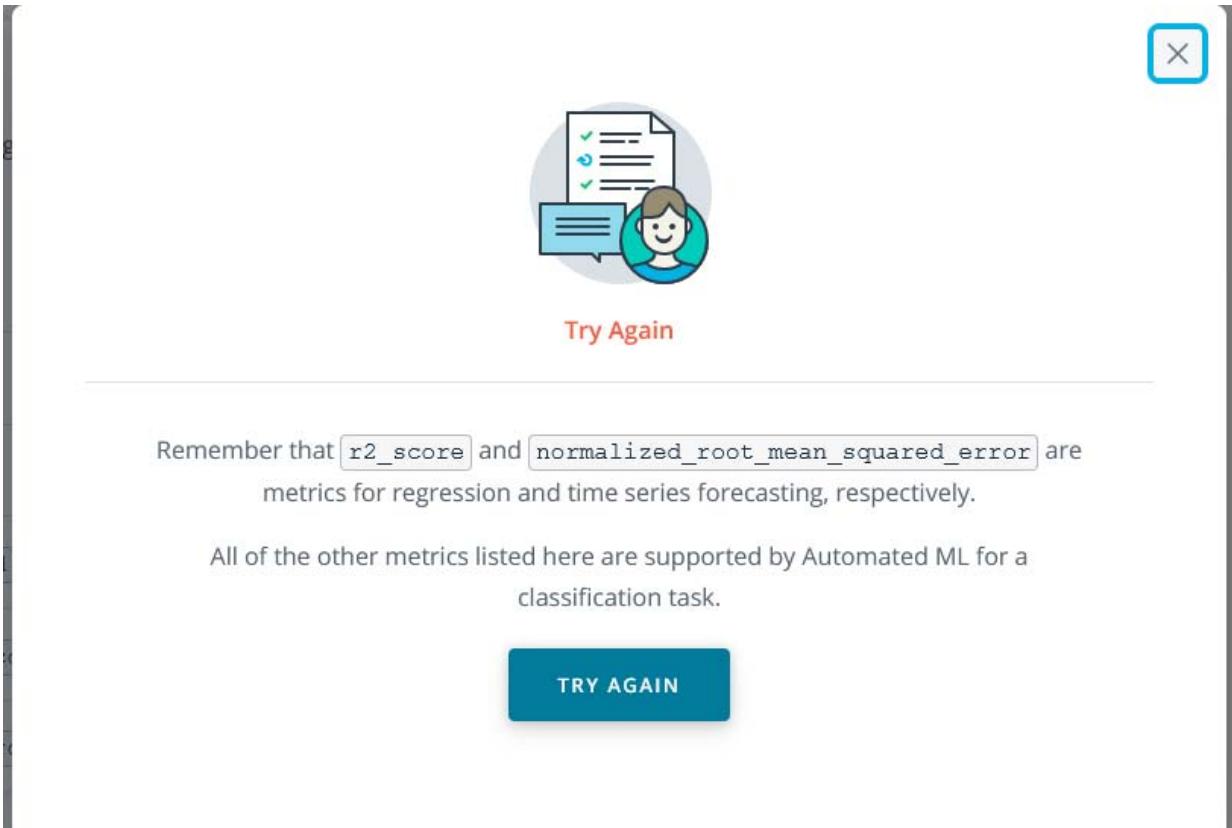
`accuracy`

`r2_score`

`AUC_weighted`

`precision_score_weighted`

`normalized_root_mean_squared_error`



Model Interpretation in Azure ML

Now we will use the run we just performed to explore the model interpretation features of AzureML.

<https://photos.app.goo.gl/sVcCPUaD4oXvzFcH7>

Note that it can take several minutes to run the model explanation, during which time you may see the message No explanations to display. You can check on the status of the model-explanation run by going to the **Child runs** tab. You can also start a model-explanation run yourself at any time by going to **Explain model** and creating a new one.

Once the child run is complete, if you don't see the visual Noah demos in the video, make sure that the **View previous dashboard experience** toggle is switched on.

Try it!

For this exercise, you will explore the interpretability of an AutoML experiment. To complete this exercise:



- Open the previous AutoML experiment we did earlier in this page
- Survey the models trained
- View the explanation for the `votingEnsemble`
- Explore the the "top K features"
- Explore the metrics of the selected model

QUESTION 2 OF 3

Which of the following is true about the correlation between a feature and a class?

- Different features can have different effects for each class
- Each feature has the same effect for all classes
- Different features have the same effect for each class
- It is not possible to know the effect of features on a class



Try Again

Remember that each feature can have different effects on the predictions of each class. For instance, the height of an NBA player can have different effects for what position the player plays at.

TRY AGAIN

QUESTION 3 OF 3

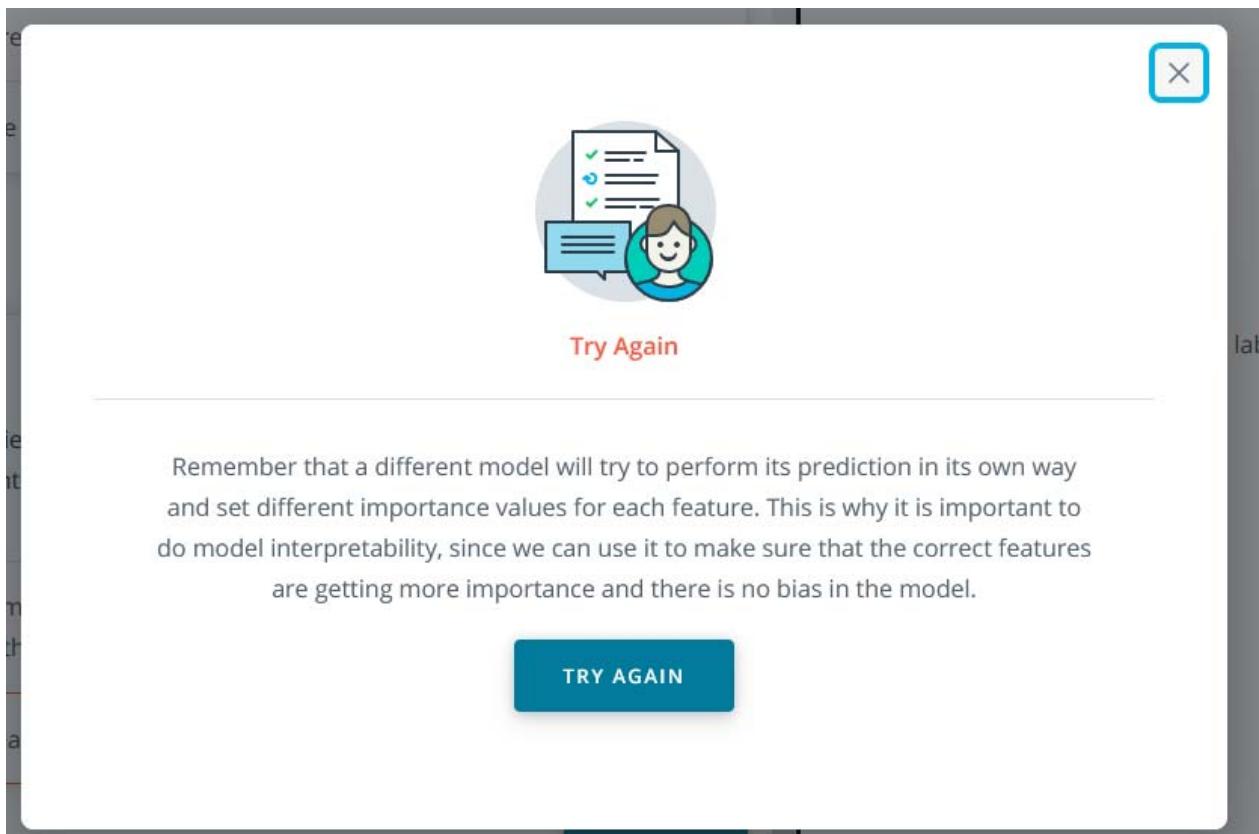
Now see if you can view the explanations for a different model. Does the other model give a different importance for the same feature?

- Yes—different models can potentially give a different importance for each feature during the training process
- No—since the features are the same, the importance will also be the same



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure



9. Exercise: AutoML—from CSV to Custom Metrics

In this exercise, we will again create an AutoML run using the NBA data. Then we will create some custom metrics and try to predict the number of pageviews a player will get. This brings together many of the components we've done separately in different parts of the lesson.

As we did earlier, we'll start by importing the NBA CSV. Note that the video shows the CSV being downloaded from Kaggle, but—as before—you can find the CSV in the *Datasets* folder on the VM desktop (or [download the CSV from here if you are doing this exercise in your own Azure subscription](#)).

<https://photos.app.goo.gl/DR7VnDGv2EgCnTY49>

Try It!

For this exercise, you will have to create an AutoML run and try to predict the number of pageviews a player will get.

Here are the steps you need to do:



- Download the dataset
- Upload the dataset to your Azure Workspace
- Create a new AutoML experiment and perform regression to predict the pageviews
- See the explanations tab and find out which feature has the most impact on a player's pageviews

QUESTION 1 OF 2

Which of these features was the most important for predicting the pageviews of a player?

Age

Twitter Favourite Count

Twitter Retweet Count

Salary



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure



Try Again

A player's Twitter favourite count is the most important feature for predicting their pageviews. You can see how to get the same result in the solution video below.

TRY AGAIN

Solution

<https://photos.app.goo.gl/WVGVPgQyYYiZgb3a6>

Try It!

Can you try to find another position that has equal performance to centre position?



I have found the other position

<https://photos.app.goo.gl/VWzLh7azx5xMVY3X9>



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

Try It!

Can you try to find the actual probability using the confusion matrix?



I have found it!

<https://photos.app.goo.gl/QYmvTZVL8Utb42Ad8>

QUESTION 2 OF 2

What is the advantage of using custom metrics?

- Some standard metrics may not be the best way to measure the performance of the model. Custom metrics are almost always better alternatives.
 - It can be used to represent the performance of the model in a different way.
 - It can help us delve deeper into the performance of the model and interpret that performance.
-



Try Again

Remember that custom metrics are a good way to explore the performance of the model in more detail and they can also give us insights into how the model performs for certain classes and why.

TRY AGAIN

10. Exporting Models with ONNX

The [Open Neural Network Exchange \(ONNX\)](#) is an open-sources portability platform for models that allows you to convert models from one framework to another, or even to deploy models to a device (such as an iOS or Android mobile device).

<https://photos.app.goo.gl/QgYdujfEs8non3nd8>

Getting Models

To get models with ONNX, there are several options. You can:

- Train the model in Azure ML
- Convert an existing model
- Get a pre-trained ONNX model
- Create an ONNX model from an AutoML service, such as [Azure Custom Vision service](#)

Using the Runtime

Installation can be done with a simple pip install, for either CPU or GPU builds:

```
pip install onnxruntime      # CPU build  
pip install onnxruntime-gpu  # GPU build
```

Then we can simply import the runtime and then instantiate an InferenceSession object, passing in the path to the model:

```
import onnxruntime  
session = onnxruntime.InferenceSession("path to model")
```

Practicing with ONNX

In this next video, we'll walk through some of the basics of using ONNX. You'll find a notebook below that you can use to follow along and try it for yourself.

<https://photos.app.goo.gl/ATvq7KHCVmKafw2R8>

The screenshot shows a Jupyter Notebook interface with the title bar "File Edit View Run Kernel Tabs Settings Help" and the notebook name "Onnx.ipynb". The toolbar includes icons for file operations, a search bar, and tabs for "Markdown". The Python version is listed as "Python 3".

Practicing with ONNX

First we download the CoreML model. We use the CoreML model from [Matthijs Hollemans's tutorial](#).

```
[ ]: import urllib.request  
coreml_model_url = "https://github.com/hollance/YOLO-CoreML-MPSNNGraph/raw/master/TinyYOLO.mlmodel"  
urllib.request.urlretrieve(coreml_model_url, filename="TinyYOLO.mlmodel")
```

Then we use ONNXMLTools to convert the model.

```
[ ]: !pip install onnxmltools==1.0.0.0  
!pip install coremltools
```

```
[ ]: import onnxmltools  
import coremltools
```

Load a CoreML model

```
[ ]:
```

[HIDE SOLUTION](#)

```
coreml_model = coremltools.utils.load_spec('TinyYOLO.mlmodel')
```

The taskbar at the bottom of the screen shows various application icons, including a browser, a terminal, and Microsoft Office applications.

File Edit View Run Kernel Tabs Settings Help

Onnx.ipynb X Python 3

Load a CoreML model

[]:

HIDE SOLUTION

```
coreml_model = coremltools.utils.load_spec('TinyYOLO.mlmodel')
```

Convert from CoreML into ONNX

[]:

HIDE SOLUTION

```
onnx_model = onnxmltools.convert_coreml(coreml_model,
'TinyYOLov2')
```

Save ONNX model

[]:

HIDE SOLUTION

```
onnxmltools.utils.save_model(onnx_model, 'tinyyolov2.onnx')
```

Get the size

The screenshot shows a Jupyter Notebook interface with the following content:

- Cell 1:** `coreml_model = coremltools.utils.load_spec('TinyYOLO.mlmodel')`
- Section Header:**

Convert from CoreML into ONNX
- Cell 2:** `[]:` (empty cell)
- Section Header:**

HIDE SOLUTION
- Cell 3:** `onnx_model = onnxmltools.convert_coreml(coreml_model, 'TinyYOLOv2')`
- Section Header:**

Save ONNX model
- Cell 4:** `[]:` (empty cell)
- Section Header:**

HIDE SOLUTION
- Cell 5:** `onnxmltools.utils.save_model(onnx_model, 'tinyyolov2.onnx')`
- Section Header:**

Get the size
- Cell 6:** `[]:` (empty cell)
- Section Header:**

HIDE SOLUTION
- Cell 7:** `import os
print(os.path.getsize('tinyyolov2.onnx'))`

11. Lesson Review

<https://photos.app.goo.gl/hgChQzMcoDhER9Sx5>

Lesson Outline: AutoML and Hyperparameter Tuning

In this lesson, we saw how we can use **AutoML** and **Hyperparameter tuning** to dramatically speed up the development of our models. Here's what we covered:

- **AutoML and SDK.** We discussed how we can use the Azure ML SDK with AutoML to operationalize and automate model creation—allowing you to create models more quickly and accurately.
- **Model Interpretation.** We explored how model interpretation allows us to build solutions using AutoML that we (and others) can more easily interpret, making visible the core features that are driving the model.
- **Exporting Models with ONNX.** ONNX is a portability platform for models that was created by Microsoft and that allows you to convert models from one framework to another, or even to deploy models to a device (such as an iOS or Android mobile device).

12. Course Review

<https://photos.app.goo.gl/F21m6N8Gw7DdjGjB8>

<https://photos.app.goo.gl/pxVwEbQciGiJu8B67>

Congratulations!

Course Outline

We've covered a lot in this course! Here's an outline of everything we've done, for your reference:

Lesson 1: Introduction to Azure Machine Learning

- **What You'll Build.** We'll have a look at a preview of what you'll build on your final project.
- **Why Do ML in the Cloud?** We'll go over some key reasons why you would want to do Machine Learning (ML) in the cloud. We'll look into some of the limitations of performing ML locally on your machine, as well as how the cloud addresses these limitations.
- **When to Do ML in the Cloud.** Most of the time the cloud is your best option for ML. But we'll discuss a few edge cases in which using the cloud may not be the best route.
- **Customers of ML.** We'll look at the different customers of machine learning, including both internal and external customers.

Lesson 2: Workspaces and AzureML Studio

- **The Azure ML Platform.** We talked about the core features of the Microsoft Azure ML platform and how they enable you to be more productive as a data scientist or machine learning engineer.
- **Workspaces and Notebooks.** Workspaces and notebooks are critical components of Microsoft Azure. We learned how these tools enable you to be a more effective data scientist—including the use of Jupyter to build and deploy machine learning models.

Lesson 3: Datastores and Datasets

- **Datastores and Datasets.** Datastores and Datasets are a critical component of cloud computing. We learned how Azure allows you to easily integrate third party datasets and open datasets into our ML pipeline to quickly develop working solutions.

Lesson 4: Training models in Azure

Here's what we covered in this lesson:

- **Managing pipelines.** We covered how to create a pipeline that you manage yourself using the console, which allows you to make very small changes that can be run repeatedly.
- **Hyperparameters in experiments.** We learned how to use hyperparameters in experiments, including how we can automate the creation of hyperparameters and make very small changes that create huge value in terms of prediction accuracy.

Lesson 5: Azure ML SDK

Here's what we covered in this lesson:

- **Creating pipelines.** We talked about how the Azure ML SDK allows us to programmatically create pipelines. By automating the creation of pipelines, we can create more pipelines; also, our pipeline creation becomes a repeatable process that other members of our team can follow.
- **Managing experiments.** We learned how to use the SDK to manage experiments programmatically with Python, allowing us to easily rerun our processes using Python scripts.

Lesson 6: AutoML and Hyperparameter Tuning

Here's what we covered in this lesson:

- **AutoML and SDK.** We discussed how we can use the Azure ML SDK to do AutoML and create models more quickly and accurately.
- **Model Interpretation.** We explored how model interpretation allows us to build solutions using AutoML that we (and others) can more easily interpret, making visible the core features that are driving the model.

Project: Creating and Optimizing an ML Pipeline

In the project, you'll have the opportunity to create and optimize an ML pipeline. You'll do this both using HyperDrive and also AutoML, so that you can compare the results of the two methods.

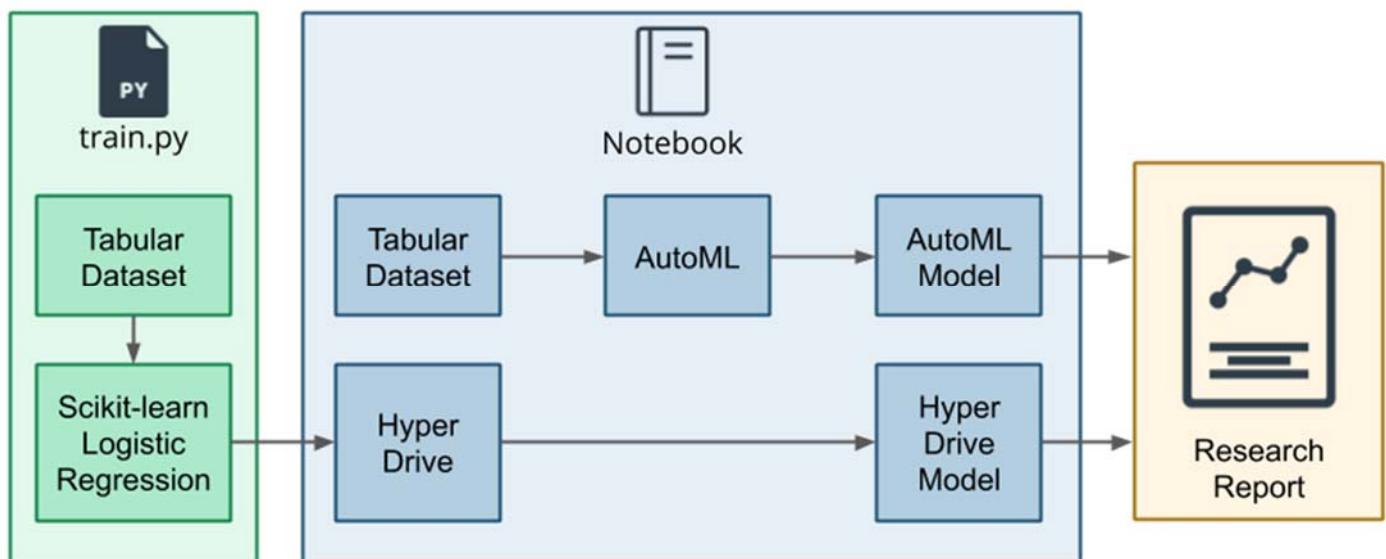
13. Project : Optimizing an ML Pipeline in Azure

1. Project Overview

<https://photos.app.goo.gl/5CtRA6W7wNDuB8zF9>

In this project, you'll have the opportunity to create and optimize an ML pipeline. You'll be provided a custom-coded model—a standard Scikit-learn Logistic Regression—the hyperparameters of which you will optimize using HyperDrive. You'll also use AutoML to build and optimize a model on the same dataset, so that you can compare the results of the two methods.

You can see the main steps that you'll be taking in the diagram below:



2. Getting Started

Getting the Starter Code

You can find all of the starter code for this project at [this Github repo](#).

Keep in mind that you only have 120 minutes before your virtual machine times out. If you are approaching that mark, make sure you commit your work to GitHub using the instructions on the "Check Your Work" page.

- Clone the repository on your own GitHub account, so that you can work on this copy.
- Open the "Azure Workspace" page in the classroom in a new tab or window, so that you can have the instructions open while you work.
- In the virtual machine, click "Access Lab" to start the lab. (Note: the lab may take a few minutes to load)
- Once the virtual machine has loaded, open a browser, navigate to Github and log in.

- On your cloned GitHub repo, click the green "Code" button to download a .zip of your files.
- Double click the "Azure Portal" button and access the Azure portal using the credentials on the virtual machine desktop.
- Once logged into the Azure portal, navigate to "ml.azure.com"
- Access the Azure Machine Learning Workspace
- Navigate to the "Notebooks" page
- Click "Upload files" and upload your notebook and the `train.py` script to Azure.

3. Setting up the Training Script

Setting up the Training Script

As part of our pipeline, we'll need a training script. For this part of the project, we've provided starter code which is marked with `# TODO` sections where you'll need to add your code. This script will be used in the next part of the project.

Review and Modify the `train.py` script

Explore the functionality of the provided code and modify the `train.py` script to run in your pipeline.

- Read through the training script to ensure you understand what is being done
- Check for parameters you may need to pass to the script later
- Import data from the specified URL using `TabularDatasetFactory`
- Split data into *train* and *test* sets

4. HyperDrive Pipeline

HyperDrive Pipeline

In this step, we'll build our training pipeline. You'll see `### YOUR CODE HERE ###` prompts where you should add your code. Note that some of the cells have pre-filled code that you should *not* modify.

You'll start by getting the `workspace` and `experiment` objects running. Then, you'll build your pipeline—from creating a compute cluster, to HyperDrive, to running the `train.py` script containing the custom-coded Logistic Regression with a HyperDrive run.

Review and modify the `'project.ipynb'` notebook

Create a compute cluster and a HyperDriveConfig, then run your Hyperdrive experiment.

- Create a compute cluster to run the experiment on, using a `vm_size` or `"Standard_D2_V2"`
- Specify a parameter sampler
- Specify a policy for early stopping
- Create an estimator for the `train.py` script
- Create a HyperDriveConfig
- Submit the HyperDriveConfig to run the experiment
- Use the `RunDetails` widget to see the progress of your run
- Use the `.get_best_run_by_primary_metric()` method of the run to select the best hyperparameters for your model
- Save the best model

5. AutoML Run

AutoML Run

Once you've completed your HyperDrive run (or while it is running!), you can set up your AutoMLConfig in the same Notebook. In this section, you'll import your data from the specified URL again, clean the data, and pass the cleaned data to an AutoMLConfig that you create.

But be careful with your configuration: **Your AutoML instances will time out after 30 minutes!**

Once you've successfully run both experiments, you can move on to creating documentation.

Create an AutoML run in the Notebook

Import and clean the data, then submit an AutoML run.

- Create a dataset from the provided URL using TabularDatasetFactory in the notebook
- Split data into train and test sets
- Modify the provided AutoML config
- Submit your AutoML run
- Save the best model



6. Modifying the README

Modifying the README

Now it is time to report on your results (from both of your experiments) in the form of a README.

This README should consist of a summary of your problem statement, your architectures, how the problem was solved, and the changes you would make. The repo you cloned has a starter README with more detailed prompts on what information you need to include in order for your project to meet specifications.

Optionally, you do not need to complete the README in your Azure workspace and can modify it locally on your own machine or in GitHub itself after your code has been uploaded.

Modify the README

Write your research summary in the README

- A summary of the problem statement
- How the problem was solved
- Explanation of the architecture, data, hyperparameters, and classification algorithm
- Explanation of the rationale for choosing a particular parameter sampler
- Explanation of the rationale for choosing a particular early stopping policy
- Description of the model and hyperparameters produced by AutoML
- Comparison of the models and their performance
- Identification of areas to improve the outcome
- (Optional) Proof of cluster cleanup if not included in the notebook

7. Check Your Work

Once you submit your project, we'll review your work and give you feedback if there's anything that you need to work on. If you'd like to see the exact points that your reviewer will check for when looking at your work, you can have a look over the project [rubric](#).

| PROJECT SPECIFICATION | |
|--|---|
| Optimizing an ML Pipeline in Azure | |
| Documentation | |
| CRITERIA | MEETS SPECIFICATIONS |
| Explain the pipeline architecture. | <p>The README contains an explanation of:</p> <ul style="list-style-type: none">• The pipeline architecture, including data, hyperparameter tuning, and classification algorithm.• The benefits of the chosen parameter sampler.• The benefits of the chosen early stopping policy. |
| Compare a provided model with one generated by AutoML. | <p>The README contains:</p> <ul style="list-style-type: none">• One or more sentences describing the model and parameters generated by AutoML.• Two or more sentences comparing the two models and their performance. |
| Explain and justify ways to improve models. | <p>The README contains two or more sentences explaining potential improvements for a future experiment and why these improvements might improve the model.</p> |

Training Pipeline and AutoML

| CRITERIA | MEETS SPECIFICATIONS |
|--|---|
| Use HyperDrive to automatically find optimal parameters. | A hyperdrive config is used and includes: <ul style="list-style-type: none"> • A parameter sampler • A policy for early stopping |
| Pass parameters to training scripts. | All specifiable parameters of the training script are specified in the hyperdrive config. |
| Retrieve the best run using <code>.get_best_run_by_primary_metric()</code> . | <code>.get_best_run_by_primary_metric()</code> is used on the hyperdrive run to retrieve the best run. |
| Use the <i>RunDetails</i> widget to explore run metrics. | The hyperdrive run is passed to the <i>RunDetails</i> widget. |
| Create an AutoMLConfig for training. | The solution notebook includes an AutoML config, which contains the following parameters: <ul style="list-style-type: none"> • <code>task</code> • <code>primary_metric</code> • <code>experiment_timeout_minutes</code> • <code>training_data</code> • <code>label_column_name</code> • <code>n_cross_validations</code> |

Infrastructure

| CRITERIA | MEETS SPECIFICATIONS |
|---|--|
| Create a compute cluster using the SDK. | A compute cluster is created using the Azure SDK and the <code>ComputeTarget</code> and <code>AmlCompute</code> objects. |
| Import data to a Dataset using the SDK. | A <code>TabularDatasetFactory</code> is used to create a dataset from the provided link. |
| Clean up deployed resources. | The <code>delete</code> method of the <code>AmlCompute</code> object is used to remove the cluster following training. OR An image of the compute cluster being selected for deletion is included in the README. |

Suggestions to Make Your Project Stand Out!

1. Include a diagram of your pipeline architecture.
2. Export your model and run it in Cloud Shell.
3. Extend your AutoML config to include more parameters.
4. Have your code check for existing compute clusters before creating a new one.

Once your `train.py` script and notebook - and optionally, your README file - are completed, you'll need to upload them to GitHub.

Note: You must upload your work to GitHub to submit your project for this course. Your progress in the virtual machine WILL NOT BE SAVED.

Follow the instructions below to make sure your work gets committed correctly. If you chose to modify your README in the virtual machine, you will also need to upload that file. Otherwise, if you chose to wait until your code is uploaded to GitHub, do not forget to go back and modify the README before submitting.

Committing files from Azure to GitHub

Follow the steps below to download your files and commit them to GitHub:

-  Under user files, click the three dots on the right side of the file name for your `train.py` script and click download to save it to the virtual machine.
-  Under user files, click the three dots on the right side of the file name for your notebook and click download to save it to the virtual machine.
-  Navigate to your project's GitHub repo
-  In the repo, click the "Add file" dropdown and click "upload files"
-  Upload your completed files to the repository and commit them.

Before you submit your project, make sure that the following files are complete and included in your Github repository:

-  README.md
-  train.py
-  A Jupyter Notebook

8. Azure Workspace

Azure Credentials

Getting Started with Lab

1. Once the environment is provisioned, a virtual machine (JumpVM) and lab guide will get loaded in your browser. Use this virtual machine throughout the workshop to perform the lab.
2. To get the lab environment details, you can select **Lab Environment** tab.

Udacity Nanodegree demo
0 hour(s), 56 minute(s) remaining HIDE

Lab Guide Lab Environment Help

Environment Details Lab Resources

Azure Credentials

Here are your credentials to login to
<https://portal.azure.com> and access the On Demand
Lab

Username 

Password 

Environment Details

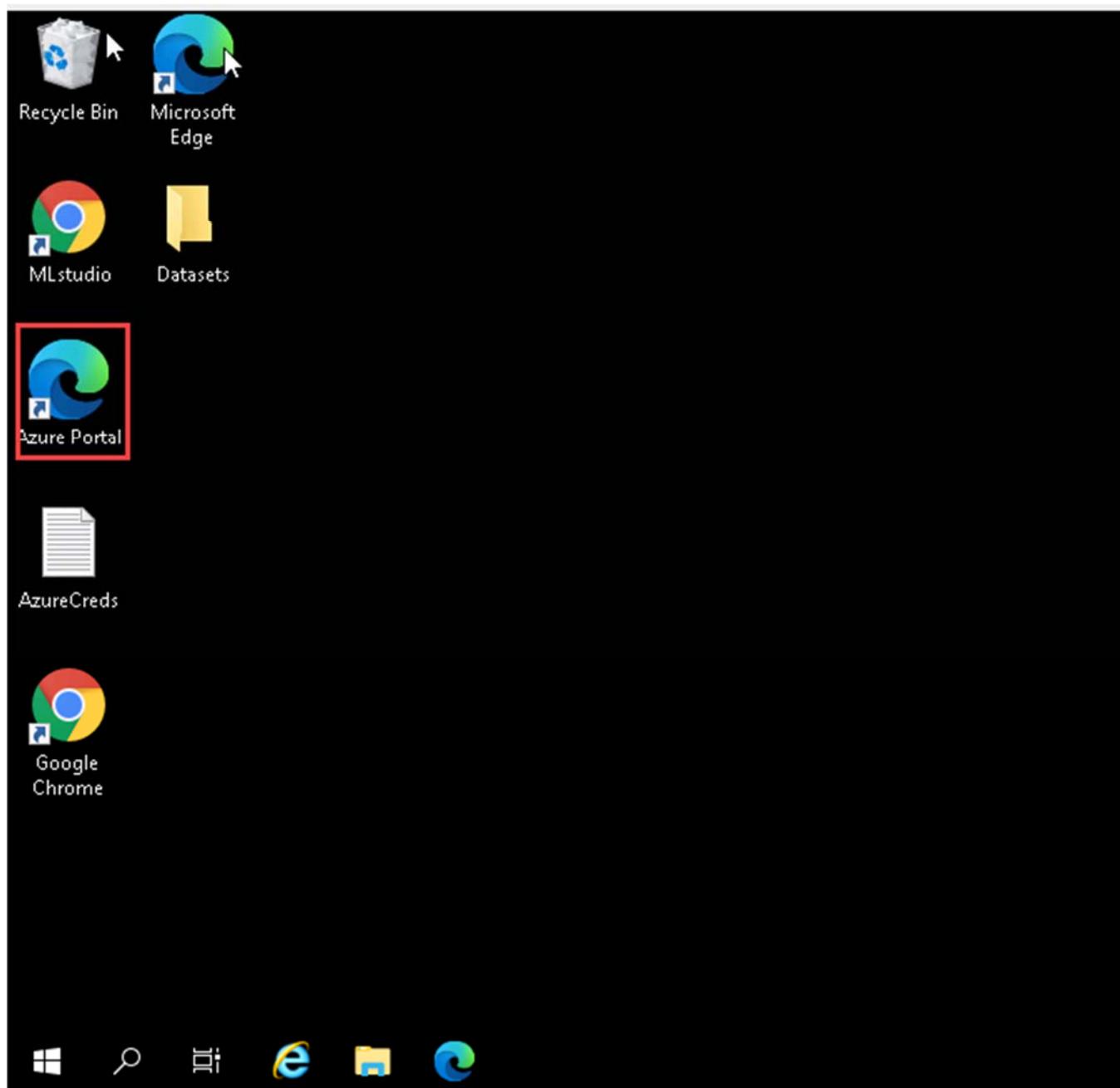
Resource Group : aml-vm-136385

Windows 
VMDNSN
ame

VMAdmin 

Login to Azure Portal

1. In the JumpVM, click on Azure portal shortcut of Microsoft Edge browser which is created on desktop.



2. When you click on Azure portal, edge browser welcome screen will come up, select **Get started**.

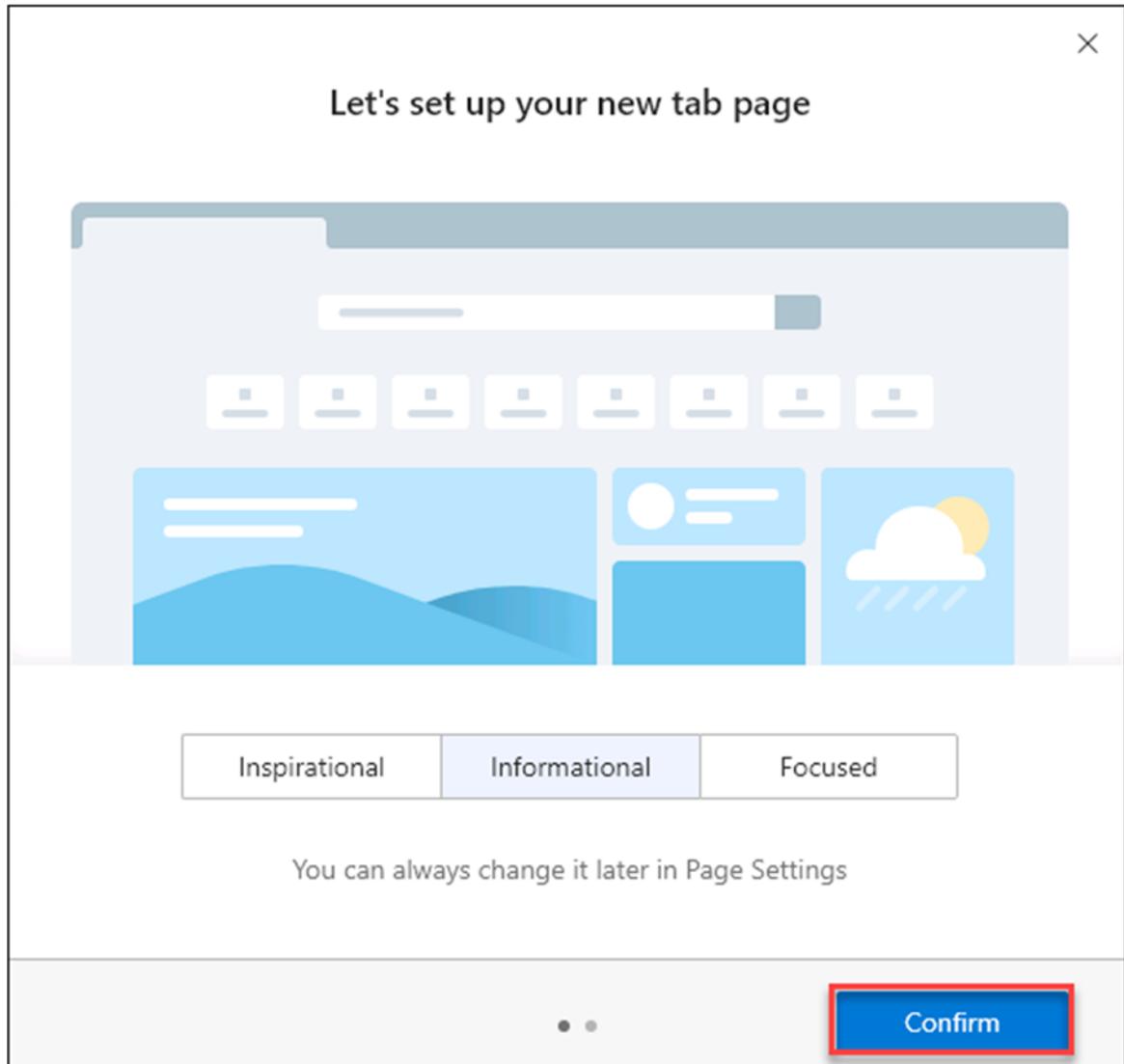


Welcome to the new Microsoft Edge

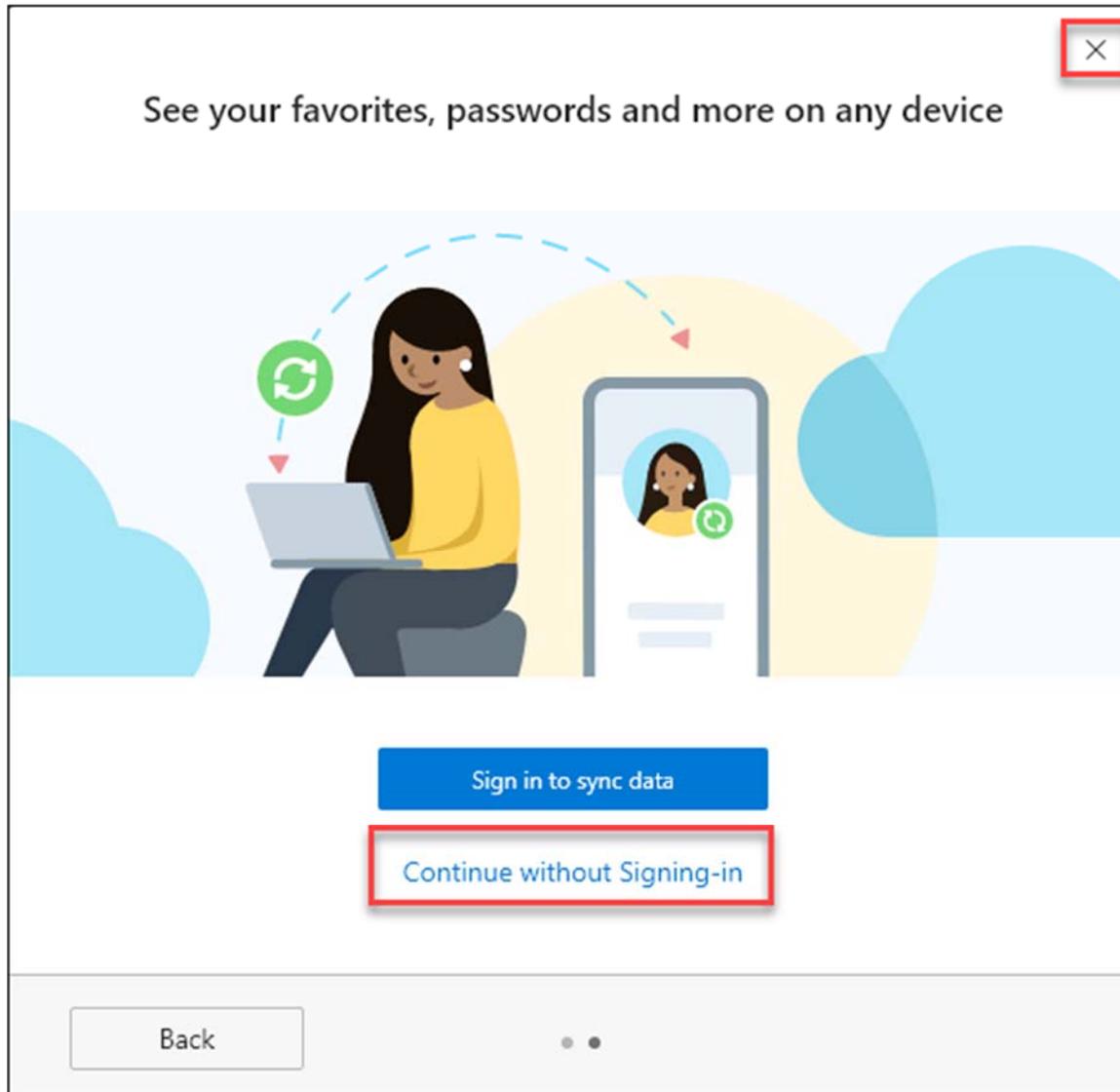
A fast, safe, and productive web browser that works for
you

[Get started](#)

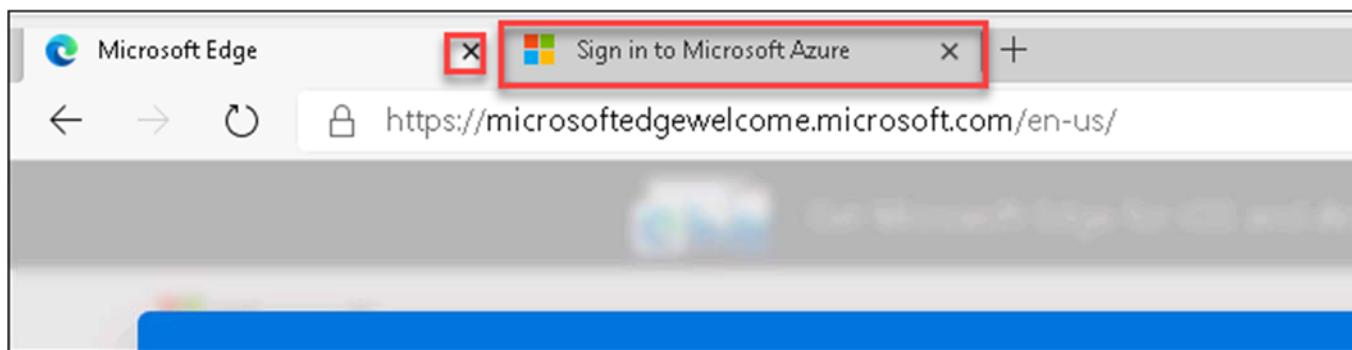
3. On next window, click on **Confirm**.



4. Now, you can close the popup which is coming up.

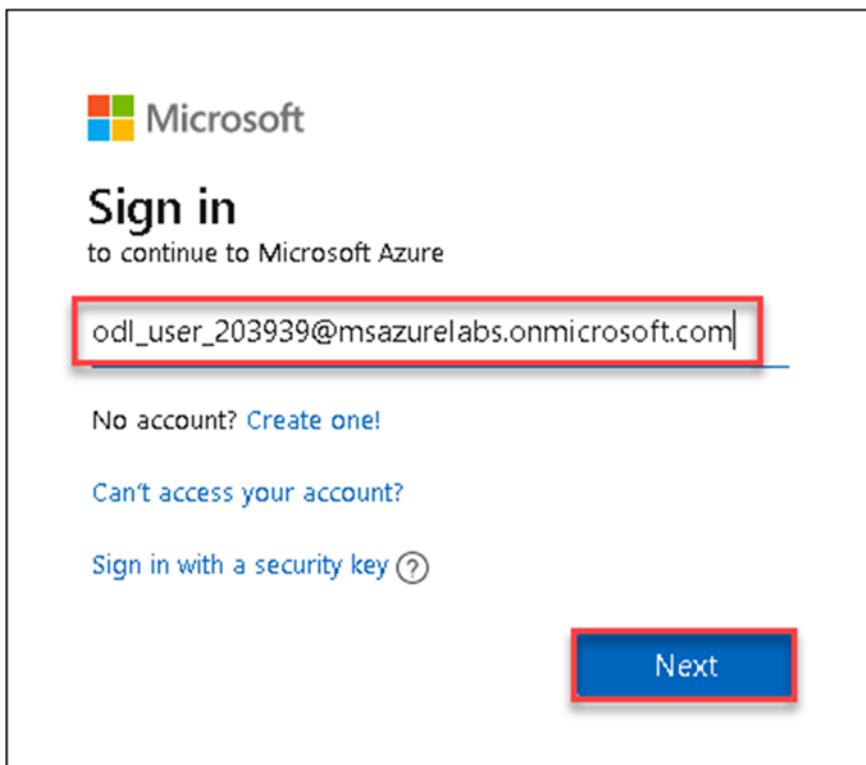


- Now, you will see two tabs in edge browser, close first tab named with **Microsoft Edge**.



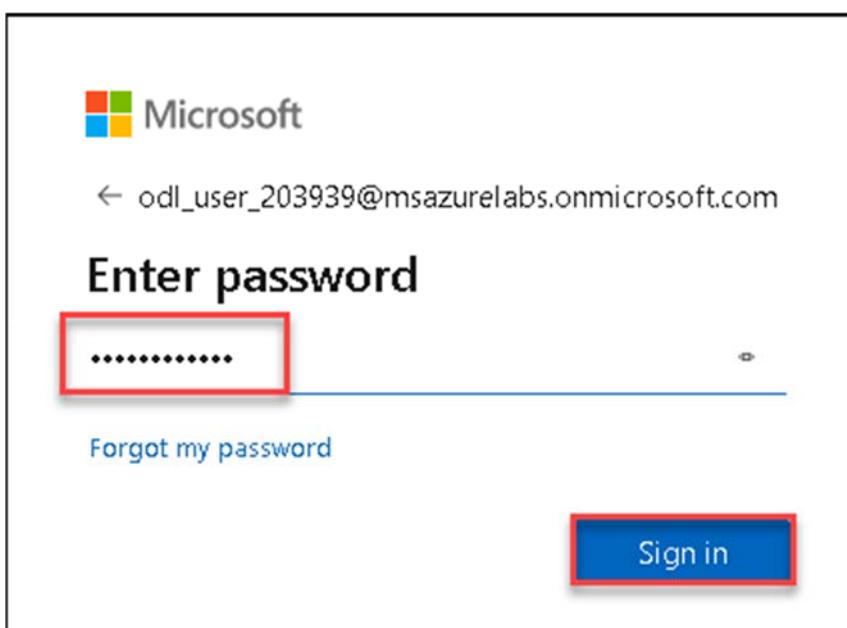
- On **Sign in to Microsoft Azure** tab you will see login screen, in that enter following email/username and then click on **Next**.

- Email/Username: odl_user_143116@udacitylabs.onmicrosoft.com



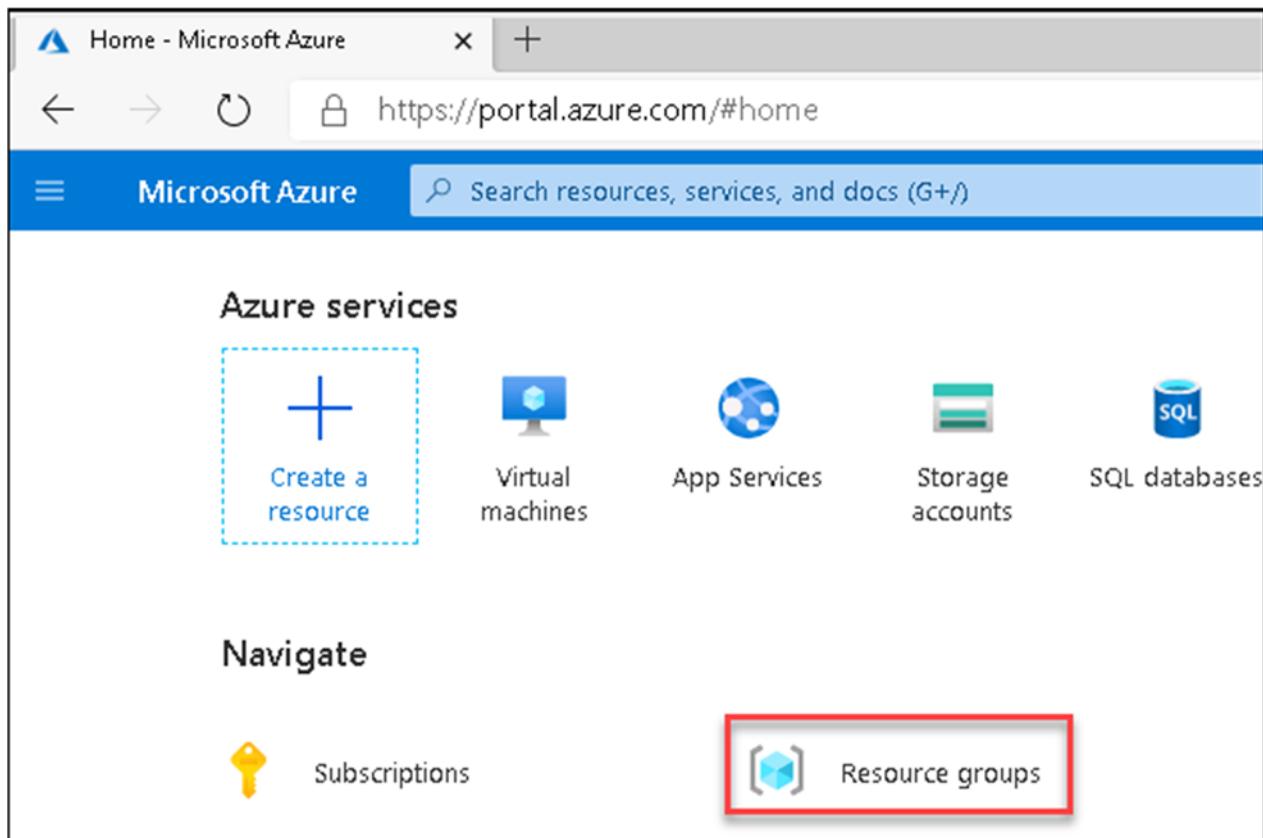
7. Now enter the following password and click on **Sign in**.

- Password: uvvv05VED*kH



8. If you see the pop-up **Stay Signed in?**, click No

9. If you see the pop-up **You have free Azure Advisor recommendations!**, close the window to continue the lab.
10. If a **Welcome to Microsoft Azure** popup window appears, click **Maybe Later** to skip the tour.
11. Now you will see Azure Portal Dashboard, click on **Resource groups** to see the resource groups.



12. Confirm you have all resource group are present as shown below.

A screenshot of the Microsoft Azure portal showing the 'Resource groups' blade. The URL in the browser bar is https://portal.azure.com/#blade/HubsExtension/BrowseResourceGroups. The page title is 'Resource groups'. A breadcrumb navigation shows 'Home > Resource groups'. The main content area displays a single resource group named 'aml-quickstarts-136385'. The table includes columns for Name, Subscription, and Location. The 'Subscription' column shows 'Udacity CloudLabs Sub - 04' and the 'Location' column shows 'South Central US'. The table has a header row with sorting icons for Name, Subscription, and Location. At the bottom, there is a pagination control showing 'Page 1 of 1'.

| Name | Subscription | Location |
|------------------------|----------------------------|------------------|
| aml-quickstarts-136385 | Udacity CloudLabs Sub - 04 | South Central US |

13. Now, click on the **Resource group** and select your machine learning workspace

Lab Environment

Here are your credentials to login to <https://portal.azure.com> and access the On Demand Lab

Username

odl_user_143116@udacitylabs.onmicrosoft.com

Password

uvvv05VED*kH

Environment Details

Resource Group : aml-vm-143116

WindowsVMDNSName

labvmmmbuhllbyijpb6.southcentralus.cloudapp.azure.com

VMAdminUsername

demouser

VMAdminPassword

Password123!

9. Optimizing an ML Pipeline in Azure

The screenshot shows a project submission interface. At the top, there's a dark header bar with the title "Optimizing an ML Pipeline in Azure" and a "SUBMIT PROJECT" button. Below the header, the page title "Project Submission" is displayed. On the left, there's a card showing project details: "DUE DATE" (Nov 24), "STATUS" (Unsubmitted, with a note "Project past due"), and a "SUBMIT PROJECT" button. To the right of the card is a dark callout box containing the text "Have project questions? Ask a technical mentor or search for existing answers!" and a "ASK A MENTOR" button. At the bottom of the page, a note says: "When you're ready to submit, go ahead and click the SUBMIT PROJECT button. You'll then have the option to provide a link to your GitHub repository."

Chapter 3 : 3. Machine Learning Operations

Lesson 1 : Introduction to Azure ML

1. Meet Your Instructor

<https://photos.app.goo.gl/VvjeTeEPpZf875pMA>

Summary

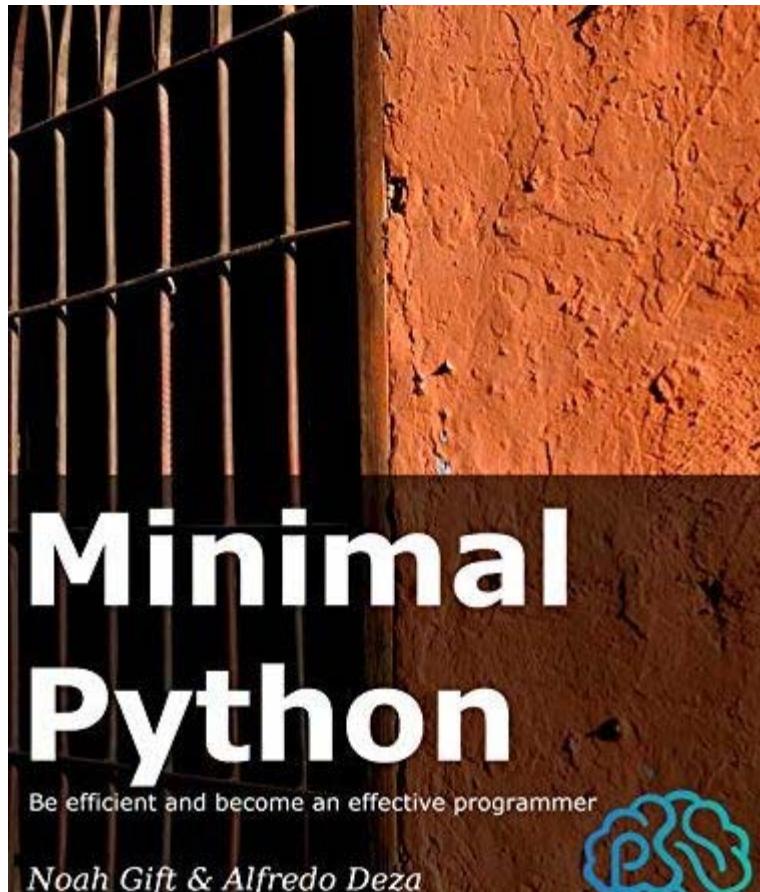
My name is Alfredo Deza and I'll be your instructor for this course. I'm very excited to show you everything about Operationalizing Machine Learning in Azure.

2. Prerequisites

<https://photos.app.goo.gl/R7q9aXQB4VWRvvRK9>

Recommended books

These are some books that I highly recommend about DevOps and cloud computing.



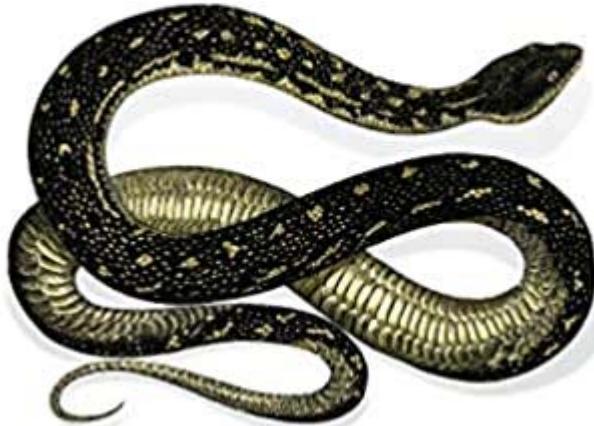
Minimal Python

The [Minimal Python Book](#) is a great resource to get started in Python and find good insights into the basics which we will apply later in this course.

O'REILLY®

Python for DevOps

Learn Ruthlessly Effective Automation



Noah Gift, Kennedy Behrman,
Alfredo Deza & Grig Gheorghiu

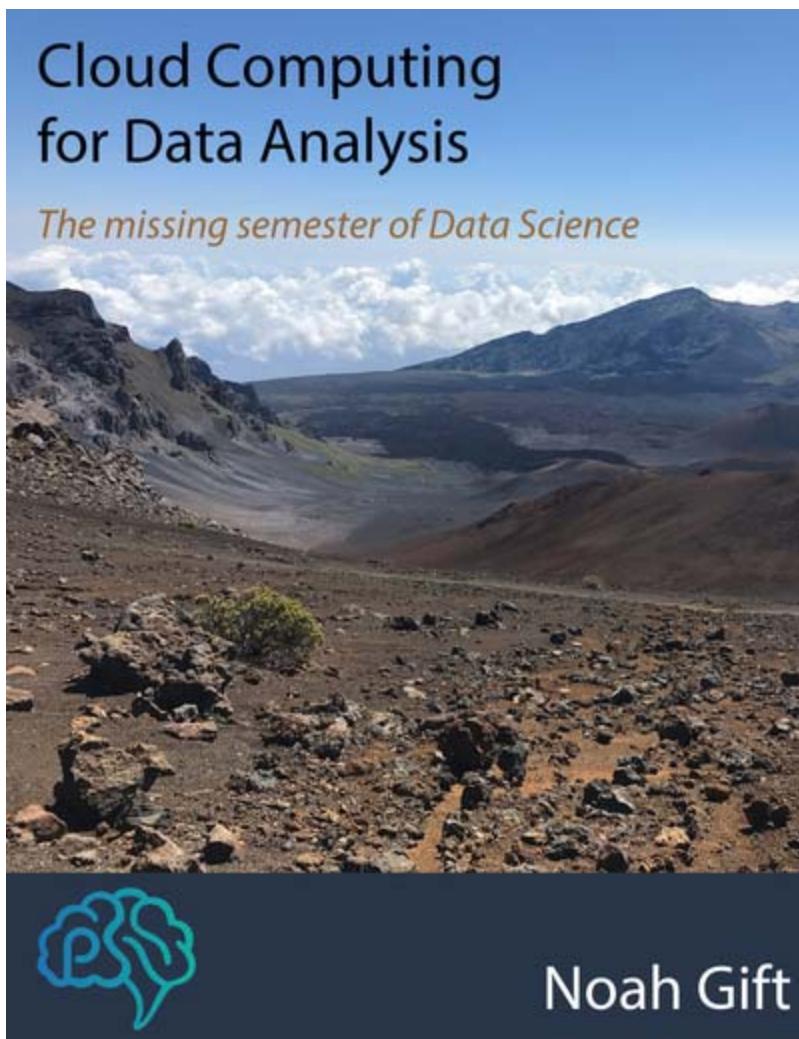
Python For DevOps

The [Python for DevOps book](#) has lots of key facts that can be directly applied to Machine Learning Operations, including logging, benchmarking, and other Docker.



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

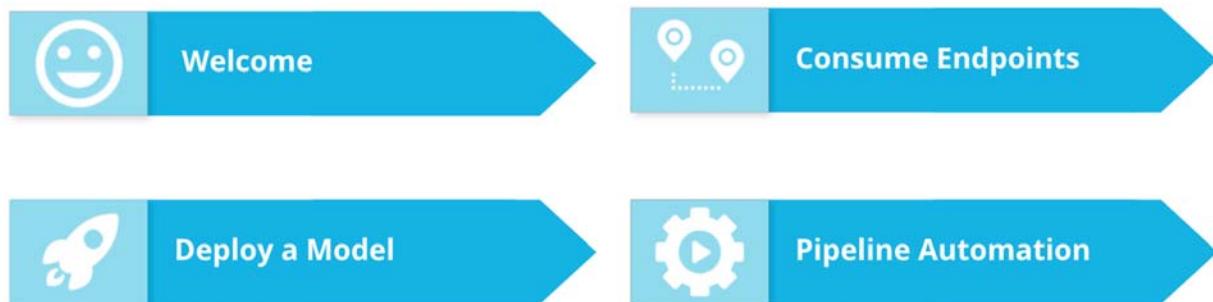


Cloud Computing for Data Analysis

Finally, the [Cloud Computing for Data Analysis book](#) has several relevant chapters related to this course like Managed Machine Learning Systems, Platforms, and AutoML.

3. Course Overview

<https://photos.app.goo.gl/PhZSvf7aTG68CSYQ9>



Summary

In this course, we will talk about three main points:

- Deploying a Model
How to create cluster configuration and compute targets and how to get models into production.
- Consuming Endpoints
Consume endpoints to interact with a deployed model in Azure ML Studio.
- Pipeline Automation
Some important aspects of pipelines: how to publish one and consume one. Machine Learning operations can greatly benefit directly from automation efforts.

Machine Learning Operations is related to applying principles of DevOps to Machine Learning, and that will be the focus throughout this course.

4. Lesson Overview

<https://photos.app.goo.gl/PVFBr6nVXjQ8TN1R6>



Summary

This lesson will cover the core concepts of what MLOps is and who are the people you will interact with. You will also be introduced to some of the tools and dependencies, and finally, you will be given a high-level introduction of the final project and its main steps.

5. Introduction to Azure ML Ops

<https://photos.app.goo.gl/8t2tZdhGkf1xCajQA>

Summary

Since at least 2015, there have been concerns about challenges using **Machine Learning in applications**. But now that industry best-practices like DevOps principles are being applied, this is starting to change.

One of the most important things is that "**DevOps**" refers to the collection of principles that allow getting software into production, or in this case, trained models.

MLOps (Machine Learning Operations) is about applying DevOps best-practices and principles to machine learning operations.

The book [Python for DevOps](#) has several key concepts that demonstrate what can be achieved when DevOps principles are applied. Another useful resource is [Azure's resources on Machine Learning Operations](#), which covers important details about the mix of DevOps and Machine Learning.

QUIZ QUESTION

What are some of the core principles of DevOps?

Automation

Designing Websites

Meetings with managers

Designing Robust pipelines

6. Stakeholders

<https://photos.app.goo.gl/gXEpN9dstbaczcvUA>

Summary

As a Machine Learning Operations specialist, you will interact with numerous people interested in getting a model into production, from the business side (like product managers) to the technical side (like data scientists and engineers). They all want to interact with you so that they can ensure the process of getting models into production and make sure their needs are incorporated into the process.

7. Tools & Dependencies

<https://photos.app.goo.gl/LFDh3NWAsCkPTRTt9>

Summary

Although you will interact primarily with Azure Machine Learning Studio, the Python SDK will be used as well. Some other tools that will be used in this course are:

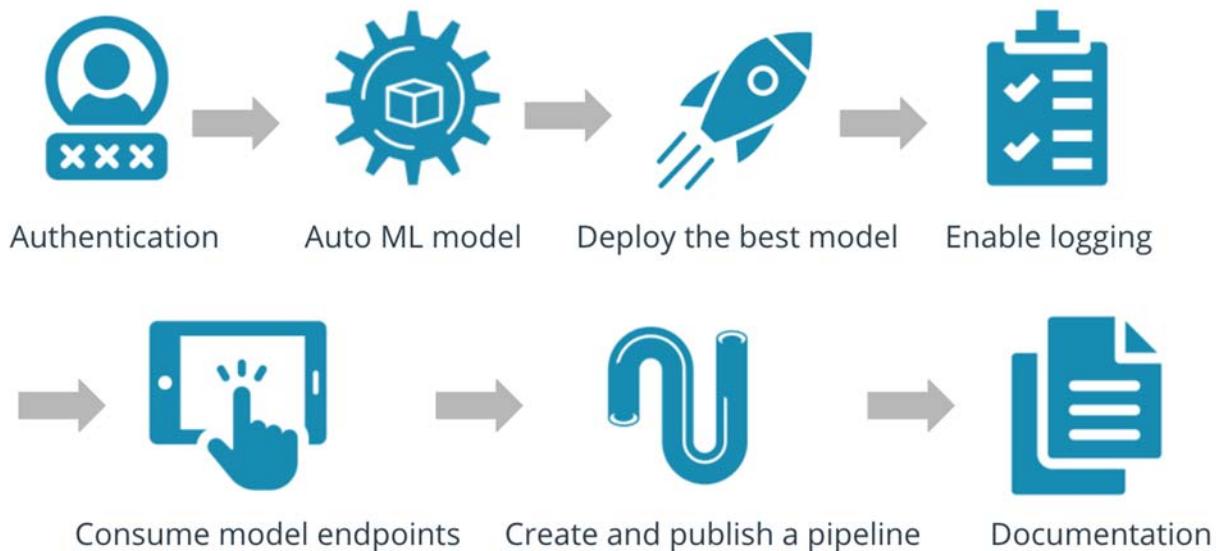
- [Azure ML studio](#)
- [Azure Python SDK](#)
- [Jupyter Notebooks](#)
- [Swagger](#)
- [Apache Benchmark](#)
- Terminal

Note, the instructor is using a Mac terminal, so some of the commands or operations shown by the instructor in this course may be somewhat different if you are using a Windows System and a PowerShell as the terminal.

This course will also provide you a Github repo that contains starter codes for exercises and the project. Please clone the [Github repo](#) to your account.

8. Final Project

<https://photos.app.goo.gl/hYeqGo93WLigVgSY9>



Summary

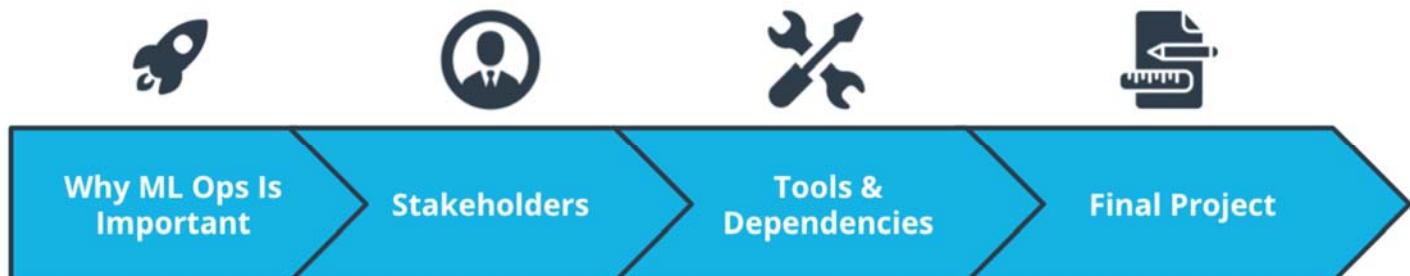
Both the Azure ML Studio and the Python SDK will be used in this project. You will start with authentication and then run an Automated ML experiment to deploy the best model.

Next, you will enable Application Insight to review important information about the service when consuming the model.

And finally, you will create, publish, and interact with a pipeline. Once all of that is complete, you will create a short screencast and a README to demonstrate all your hard work.

9. Lesson Review

<https://photos.app.goo.gl/9m29rxfCYPVF3J136>



Summary

This lesson has covered key facts about MLOps and who you will potentially interact with. Then, you learned about the tools you will use in this course. And finally, we covered the major steps in the final project.

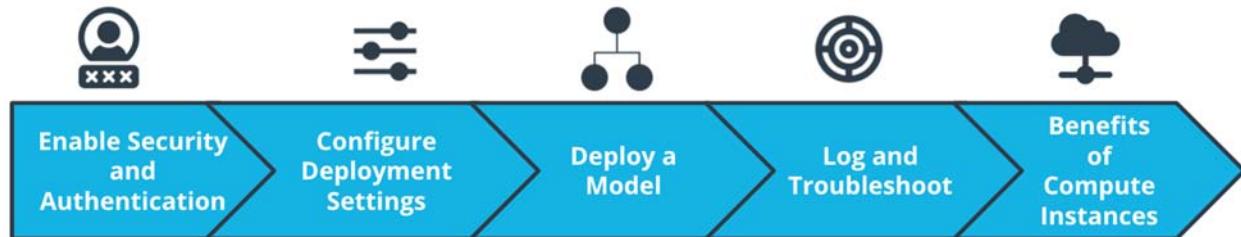
10. Good Luck!

<https://photos.app.goo.gl/BepVfyp6fyvth5cQA>

Lesson 2 : Deploy a model

1. Introduction

<https://photos.app.goo.gl/MTkzEWCa76pUWDoj8>



Summary

In this lesson we will talk about model deployments, and what are all the important details you want to be aware of when shipping models into production.

Deploying a model is a crucial step towards a robust workflow. To get a model into production, several things need to take place. We will cover these topics in this lesson.

- Enabling Security and Authentication
- Configuring Deployment Settings
- Deploying a Model
- Troubleshooting
- Benefits of compute instances

2. Best Practices of DevOps Professionals

<https://photos.app.goo.gl/Z2Rj8j4CKf4ZwSNZA>

Summary

Some of the most important best-practices of DevOps professionals are introduced in this section. Constant evaluation and monitoring, along with a robust deployment is a good way to survive disruptive changes.

New terms

- **DevOps:** A set of best practices that helps provide continuous delivery of software at the highest quality with a constant feedback loop.

Further reading

[Chapter 7 of Python For DevOps](#) has an excellent section about logging and troubleshooting.

3. Enable Security and Authentication

<https://photos.app.goo.gl/ZtFszhh64aggV2uEA>

Summary

Authentication is crucial for the continuous flow of operations. Continuous Integration and Delivery system (CI/CD) rely on uninterrupted flows. When authentication is not set properly, it requires human interaction and thus, the flow is interrupted. An ideal scenario is that the system doesn't stop waiting for a user to input a password. So whenever possible, it's good to use authentication with automation.

Authentication types

Key- based

- Azure Kubernetes service enabled by default
- Azure Container Instances service disabled by default

Token- based

- Azure Kubernetes service disabled by default
- Not support Azure Container Instances

Interactive

- Used by local deployment and experimentation (e.g. using Jupyter notebook)

Service Principal

A “Service Principal” is a user role with controlled permissions to access specific resources. Using a service principal is a great way to allow authentication while reducing the scope of permissions, which enhances security.

New terms

- **CI/CD:** Continuous Integration and Continuous Delivery platform. Jenkins, CircleCI, and Github Actions are a few examples.

Further reading

Both the [Jenkins](#) and [Github Actions](#) websites have good information about their CI/CD platforms and why they are compelling the CI/CD platform.

4. Accessing Azure ML

Accessing Azure Machine Learning

On the next page, we will start the first hands-on exercise for this course. You will find many similar exercises throughout this Nanodegree. For all of these exercises, you will need access to [Azure Machine Learning](#). If you have (or want to purchase) your own Azure subscription, you are welcome to use it to complete the exercises—however, please be aware that using your own account can involve a significant cost.

For that reason, we've provided access to an Azure account through virtual machines (VMs) that you can access right here in the Udacity classroom. You'll find one on the next page. The layout looks like this, with the instructions on the left and the VM on the right:

Instructions

Virtual Machine

The screenshot shows the Azure ML Studio interface. On the left, there's a sidebar with options like 'Compute', 'Datasets', 'Compute instances', 'Model registry', and 'Metrics'. The main area is titled 'Exercise: Creating Notebooks'. It shows a 'Compute' instance named 'mlcompute' with a status of 'Running'. A 'New compute instance' dialog box is open, prompting for a name ('Compute name') and location ('Region'). The 'Compute type' dropdown is set to 'D2s (Large)'. Below the dialog, there's a note: 'We'll be using Jupyter Notebooks multiple times in this course, so make sure you've followed along and can create one in Azure ML Studio:' followed by a list of four tasks with checkboxes. A blue 'Download template for workspace' button is at the bottom.

We'll be using Jupyter Notebooks multiple times in this course, so make sure you've followed along and can create one in Azure ML Studio:

- Create a new compute instance
- Name the notebook
- Select between CPU or GPU
- Open the notebook interface

ACCESS LAB

If you are having issues with the lab, please contact support at udacity-labsupport@udacity.com.

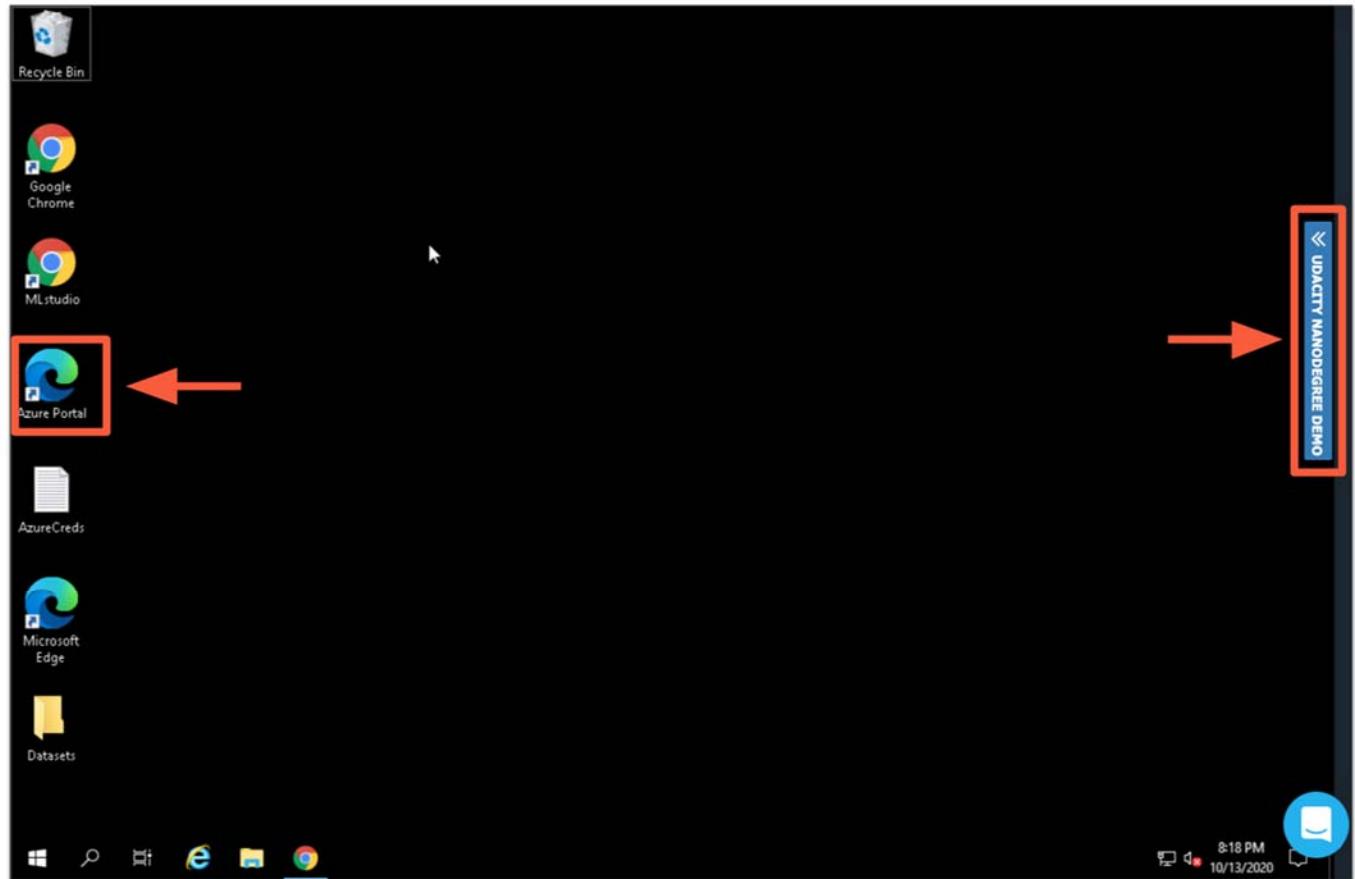
To open the VM, you'll simply click the **ACCESS LAB** button. In general, we have programmed the VMs to complete as much setup as possible for you, so that you can dive directly into the exercise once it starts. However, please be aware that this sometimes means it will take a while to load since it is setting up all of the resources and configurations you need. Most of the labs will be ready around 5–10 minutes, but a few may take 30 minutes because we are *pre-loading a trained model* for you.

Labs that take very long to load are:

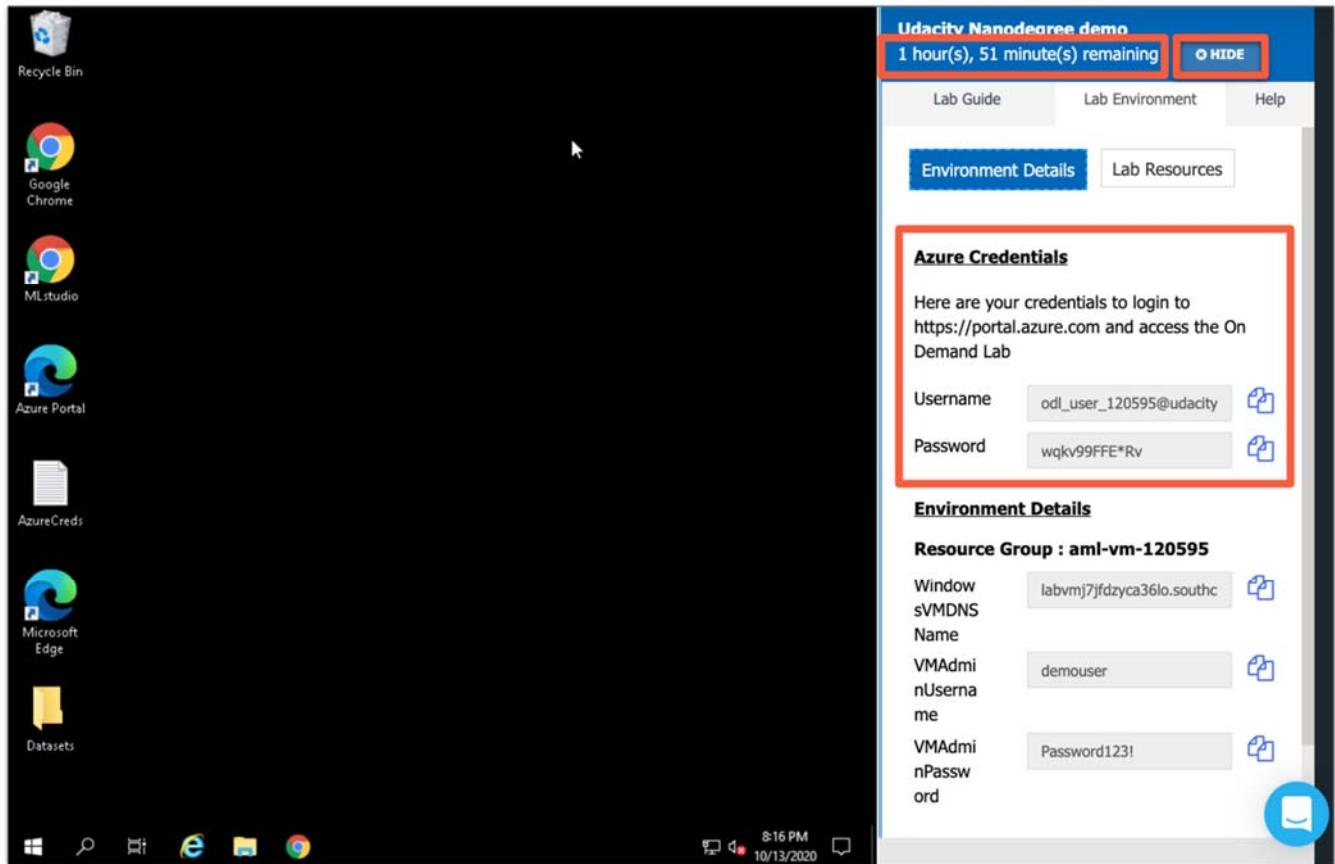
- Lesson 1: Exercise: Enable Application Insights
- All the labs in Lesson 2
- Lesson 3: Publish and Consume a Pipeline

Feel free to load these labs beforehand.

Once the VM loads, it will look like this:



On the VM desktop, you'll find a shortcut to the Azure Portal. You can access your login credentials by clicking on the blue tab on the right.



You'll also see that this indicates the time you have remaining to access the VM. Because running the VM and Azure ML requires significant resources, each exercise has a limited time for completion—but if you run out of time, you can extend your VM session by an additional 15 minutes (and you can extend your session 4 times). Please note that there is also a one hour timeout—so if you walk away from the computer and are not accessing the VM at all for over an hour, it will quit automatically.

Browser Type and Cookies

Generally the best browser to use for these exercises is [Chrome](#). If you are using a different browser and experience any issues, please try switching to Chrome to see if this resolves the problem.

Also, if you have not already, be sure to [enable cookies](#) in your browser. In some cases, failing to enable cookies will result in the error shown below.

 {{ 'RDP_LAB_ENVIRONMENT_STATUS' | translate}}

Getting Help

If you have a technical problem while using your VM, you can send an email to Udacity Support at udacity-labsupport@udacity.com. Please be sure to send your message from the email you used to enroll in the Nanodegree program.

Please copy the below text into your email and fill out the following information:

- *Subject Line:* Error in Lab name
- *Timestamp and Timezone:*
- *Brief description of what you were doing when the issue occurred:*
- *Screenshots:*

5. Exercise: Enable Security and Authentication

How to Use This Workspace (lab)

For this exercise (and most of the exercises in this course) you need access to the Azure Portal. If you have your own Azure subscription, you are welcome to use it to complete the exercises—however, please be aware that using your own account can involve a significant cost that you would then need to pay yourself.

For that reason, we've provided access to an Azure account through virtual machines (VMs) that you can load in workspaces, like this one, at no additional cost. To access the VM and get started, click the **Access Lab** button. Note that it can sometimes take several minutes for your VM to load.

Tip: If you are working on a small screen, you can make more room for the lab by hiding the Udacity menu. Just click the hamburger menu in the upper left:

The screenshot shows the Azure Machine Learning Studio interface. On the left, there's a sidebar with a back arrow, the title "Lesson 3: Workspaces and the AzureML Stud...", a search bar, and sections for "RESOURCES" and "CONCEPTS". Below these are two completed items: "1. Lesson Overview" and "2. Intro to Azure ML Platform", each preceded by a green checkmark. On the right, a white sidebar titled "Exercise ML Wor..." contains text about creating an ML workspace and instructions for exercises. A red box highlights the "HIDE" button at the top right of this sidebar.

On the right, you will see a VM lab guidance. One piece of information that you will see on this guidance is the lab *time remaining*. It is very important to track the remaining time because the lab will expire and your progress will be lost when the time remaining reaches 0. You can hide the VM lab guidance by clicking the "Hide" button.

The screenshot shows the header of a VM lab guidance window. It features a blue header bar with the text "Udacity Nanodegree demo" and "0 hour(s), 58 minute(s) remaining". To the right of the time remaining is a red-bordered "HIDE" button. Below the header is a navigation bar with three tabs: "Lab Guide", "Lab Environment", and "Help". The "Lab Guide" tab is currently active.

Then it will collapse into a narrow blue sidebar on the right side of the screen. You can click this sidebar to show the guidance.



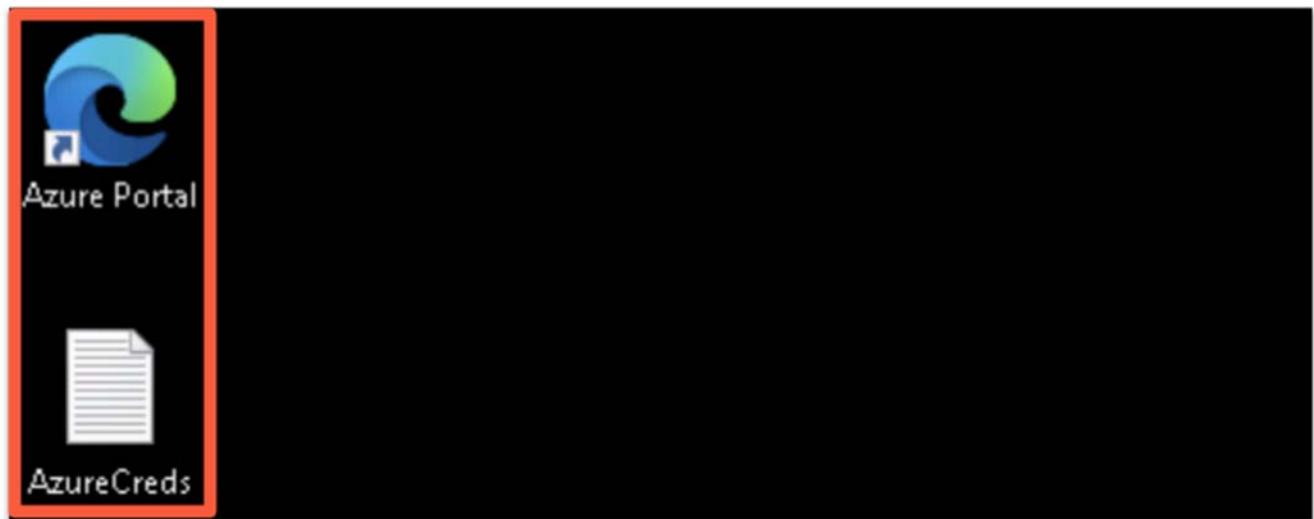
You can also resize these instructions as needed, by clicking and dragging on the divider:

Exercise: Creating an Azure ML Workspace

In this exercise, we'll get started by creating our first Azure ML Workspace.

The slide has a red double-headed arrow icon in the top right corner.

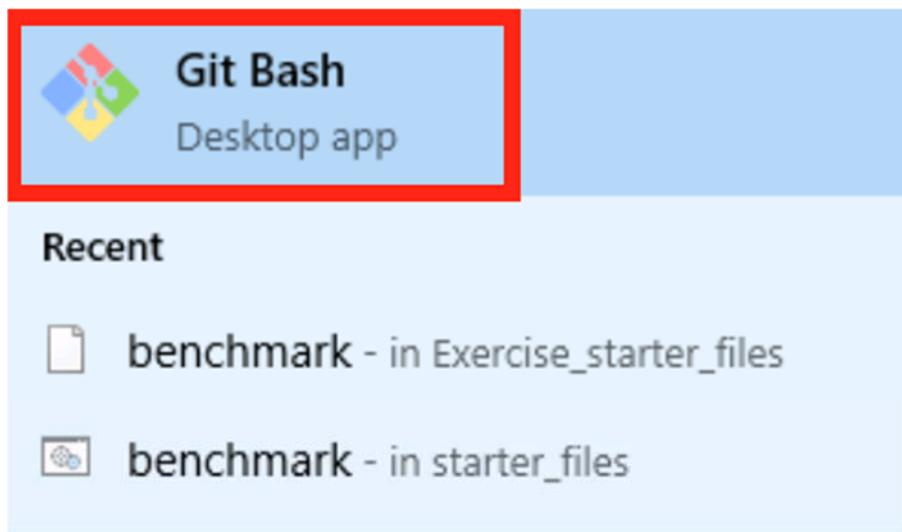
Once your VM loads, go to the Azure Portal by double-clicking the shortcut on the virtual machine's desktop. You can find the login information in the `AzureCreds.txt` file that is also on the desktop:



Once you are in the portal, you will want to head to **Azure Machine Learning**. You can enter Machine Learning in the search box and choose it.

If you are using the lab provided to you, you can use the "Git bash" as the terminal.

You can click the magnifying lens icon in the bottom left corner and type "git bash" in the search bar. Then click on the Git Bash application under "Apps".



Command

git bash



A git bash command window will show up. You can then use it to run several commands. For example, you can use the `cd` command to set the path. You can use `sh` to run the `.sh` script.

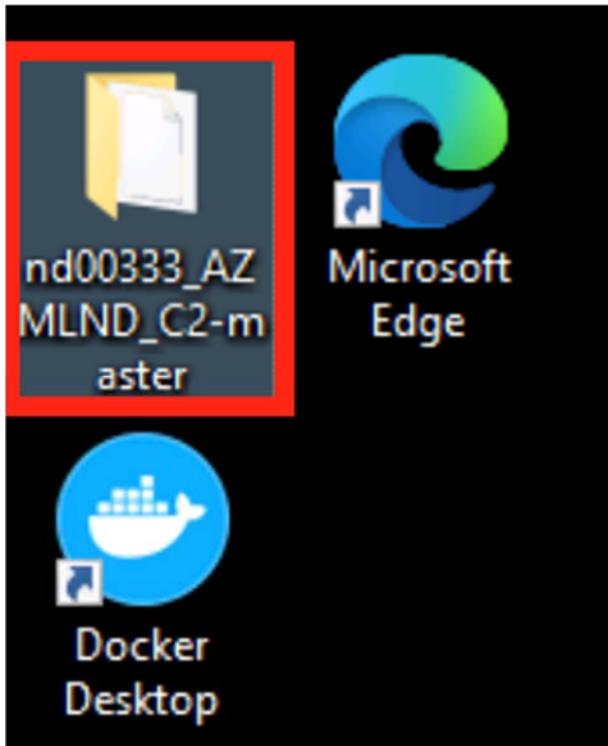


```
MINGW64:/c/Users/demouser/Desktop
demouser@labvm MINGW64 ~
$ cd desktop
demouser@labvm MINGW64 ~/desktop
$ sh filename.sh
```

Another important thing to note is that some of the files used in the walkthroughs and exercises are stored in the `Exercise_starter_files` in the GitHub repo as well as the **"nd00333_AZMLND_C2-master" folder on the VM desktop.**

| | | |
|--|----------------------|--------------|
|  Exercise_starter_files | Add files via upload | 13 days ago |
|  starter_files | Update README.md | 9 days ago |
|  README.md | Create README.md | 2 months ago |

GitHub repo



Folder on the desktop

Now let's start learning!

<https://photos.app.goo.gl/oZHVAh8v3QrPSHPG6>

⚠️ Important: In some browsers, you may find that if you pause/play the instructional video, you are then unable to type anything in the VM. This happens because the YouTube video player sometimes does not return the keyboard focus. If this happens, simply click again anywhere else in these instructions (such as on this text) and you should find you're again able to type within the VM.

Summary

In this demo, the Service Principal gets enabled and the many steps of getting to create one are performed successfully.

Note: you won't be able to create a Service Principal if you are using the **lab provided** to you, an error like this `ValidationError: Insufficient privileges to complete the operation` will appear. It is because you are **not authorized** to create a Service Principal on the Udaicty account. But this **will not** be a blocker for other exercises in the course. There should not be any issues if you are using **your own account**. We still encourage you to follow the demo and type in the command and see the information and output come from Azure.

Several steps are needed to create the Service Principal:

- Ensure the `az` command-line tool is installed along with the `ml` extension

The Azure Machine Learning extension allows you to interact with Azure Machine Learning Studio, part of the `az` command. Ensure it is installed with the following command:

```
az extension add -n azure-cli-ml
```

- Create the Service Principal with `az` after login in

```
az ad sp create-for-rbac --sdk-auth --name ml-auth
```

- Capture the `"objectId"` using the `clientId`:

```
az ad sp show --id xxxxxxxx-3af0-4065-8e14-xxxxxxxxxxxx
```

- Assign the role to the new Service Principal for the given Workspace, Resource Group and User `objectId`

```
$ az ml workspace share -w Demo -g demo --user xxxxxxxx-cbdb-4cf0-089f-xxxxxxxxxxx --role owner
```

Exercice: Create a Service Principal Authentication method

In this exercise, you will create a Service Principal which is a good way to authenticate to Azure ML services.

Now create the Service Principal (SP). Remember you can use whatever name you want, although in the examples used `ml-auth`.

Note: you won't be able to create a Service Principal if you are using the **lab provided** to you, an error like this `ValidationError: Insufficient privileges to complete the operation` will appear. It is because you are **not authorized** to create a Service Principal on the Udaicty account. But this **will not** be a blocker for other exercises in the course. There should not be any issues if you are using **your own account**. We still encourage you to follow the demo and type in the command and see the information and output come from Azure.

Note: You will need to match your workspace, subscription, and ID in the command line below.

```
$ az ml workspace share -w Demo -g demo --user xxxxxxxx-cbdb-4cf0-089f-xxxxxxxxxxx --role owner
```

Please make sure the following is completed:

-  Make sure `az` is installed and enabled it in the terminal
-  `az login` has succeeded
-  Make sure the Azure Machine Learning extension is installed
-  Create the Service Principal (SP) and allow the SP access to your specific workspace
-  `az ml workspace share` completed without errors

6. Solution: Enable Security and Authentication

1. You are able to log in with `az`

```
bash-3.2$ az login
You have logged in. Now let us find all the subscriptions to which you have access...
[{"cloudName": "AzureCloud",
"homeTenantId": "660b3398-b80e-49d2-bc5b-ac1dc93b5254",
"id": "a59af59a-605a-4b0e-968f-a30ad6bb7ad5",
"isDefault": true,
"managedByTenants": [],
"name": "Azure Sponsorship - Udacity-Testing",
"state": "Enabled",
"tenantId": "660b3398-b80e-49d2-bc5b-ac1dc93b5254",
"user": {
    "name": "udacityinsructor2@udacitylabs.onmicrosoft.com",
    "type": "user"
}}
]
bash-3.2$
```

2. The azure-cli-ml extension is added or already installed.

```
bash-3.2$ az extension add -n azure-cli-ml
Extension 'azure-cli-ml' is already installed.
bash-3.2$
```

3. After running az ad sp create-for-rbac --sdk-auth --name ml-auth, Azure responds with output similar to this:

Note: you can use names other than ml-auth.

```
Changing "ml-auth" to a valid URI of "http://ml-auth", which is the required fo
Creating a role assignment under the scope of "/subscriptions/xxxxxxxx-2cb7-4cc5-90b4-eb1d0c6c24c6"
  Retrying role assignment creation: 1/36
  Retrying role assignment creation: 2/36
{
  "clientId": "xxxxxxxx-3af0-4065-8e14-xxxxxxxxxxxx",
  "clientSecret": "xxxxxxxxxxxxxx.IPgqLjBH2.Uj6VCo1hk3",
  "subscriptionId": "39b85eca-2cb7-4cc5-90b4-eb1d0c6c24c6",
  "tenantId": "xxxxxxxx-cbdb-4c04-89fc-xxxxxxxxxxxx",
  "activeDirectoryEndpointUrl": "https://login.microsoftonline.com",
  "resourceManagerEndpointUrl": "https://management.azure.com/",
  "activeDirectoryGraphResourceId": "https://graph.windows.net/",
  "sqlManagementEndpointUrl": "https://management.core.windows.net:8443/",
  "galleryEndpointUrl": "https://gallery.azure.com/",
  "managementEndpointUrl": "https://management.core.windows.net/"
}
```

Changing "ml-auth" to a valid URI of "http://ml-auth", which is the required format used for service principal names

Creating a role assignment under the scope of "/subscriptions/xxxxxxxx-2cb7-4cc5-90b4-xxxxxxxx24c6"

Retrying role assignment creation: 1/36

Retrying role assignment creation: 2/36

```
{  
  "clientId": "xxxxxxxxx-3af0-4065-8e14-xxxxxxxxxxxx",  
  "clientSecret": "xxxxxxxxxxxxx.IPgqLjBH2.Uj6VCo1hk3",  
  "subscriptionId": "39b85eca-2cb7-4cc5-90b4-eb1d0c6c24c6",  
  "tenantId": "xxxxxxxx-cbdb-4c04-89fc-xxxxxxxxxxxx",  
  "activeDirectoryEndpointUrl": "https://login.microsoftonline.com",  
  "resourceManagerEndpointUrl": "https://management.azure.com/",  
  "activeDirectoryGraphResourceId": "https://graph.windows.net/",  
  "sqlManagementEndpointUrl": "https://management.core.windows.net:8443/",  
  "galleryEndpointUrl": "https://gallery.azure.com/",  
  "managementEndpointUrl": "https://management.core.windows.net/"  
}
```

4. Capture the "objectId" using the ClientID:

```
$ az ad sp show --id xxxxxxxx-3af0-4065-8e14-xxxxxxxxxxxx
```

This step will output some information and you will find the objectId to assign the role.

5. Finally, allow the Service Principal access to the workspace. You will need to change the code to match your workspace, subscription, and the objectId value retrieved from the previous step.

```
$ az ml workspace share -w Demo -g demo --user xxxxxxxx-cbdb-4cf0-089f-xxxxxx
```

```
$ az ml workspace share -w Demo -g demo --user xxxxxxxx-cbdb-4cf0-089f-xxxxxxxxxxx --role owner
```

Note: This command should complete without any output

7. Quiz: Enable Security and Authentication

QUESTION 1 OF 3

What are some good practices to survive disruptive changes?

Constant Evaluation

Website redesign

Robust Deployment

Monitoring

Meeting with stakeholders

QUESTION 2 OF 3

Which ONE of the following is the reason for NOT having human interaction in a CI/CD environment?

To enhance security

To prevent unauthorized access

Because double checking increases confidence in the deployment

Automated authentication is a better choice because it enables continuous flow.



Try Again

Double checking will increase confidence in deployments, but not at the expense of having to interact with a CI/CD environment.

TRY AGAIN



Try Again

Preventing unauthorized access is a good idea, but it isn't necessary to prompt for user input in automation

TRY AGAIN



Try Again

Although having authentication is great, it isn't a good idea to halt automation most of the time requesting a password prompt

TRY AGAIN

QUESTION 3 OF 3

Match the right authorization constraint to the Azure service

Submit to check your answer choices!

AZURE SERVICE

Azure Kubernetes Service (AKS)

KEY-BASED AUTHORIZATION

Enabled by default

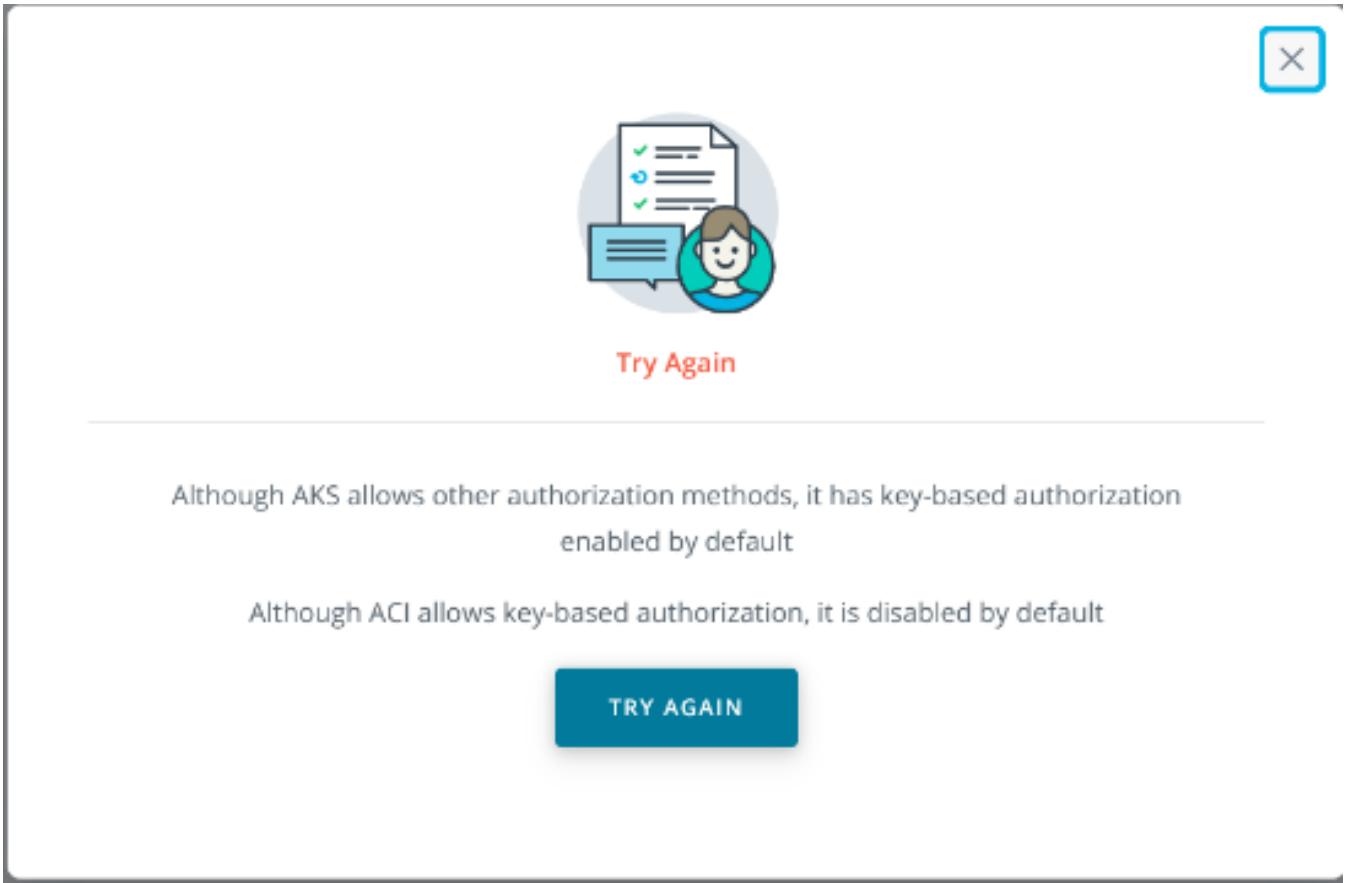
Azure Container Instance (ACI)

Disabled by default



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure



Although AKS allows other authorization methods, it has key-based authorization enabled by default

Although ACI allows key-based authorization, it is disabled by default

TRY AGAIN

8. Configure Deployment Settings

<https://photos.app.goo.gl/AQXwqaAXL5wRrLmZ6>

Summary

Deployment is about delivering a trained model into production so that it can be consumed by others. **Configuring deployment settings** means making choices on cluster settings and other types of interaction with a deployment. Having a good grasp on configuring production environments in Azure ML Studio and the Python SDK is the key to get robust deployments.

ACI and AKS

Both [ACI](#) and [AKS](#) are available in the Azure ML platform as deployment options for models.

ACI is a container offering from Azure, which uses container technology to quickly deploy compute instances. The flexibility of ACI is reduced as to what AKS offers, but it is far simpler to use.

AKS, on the other hand, is a Kubernetes offering. The Kubernetes service is a cluster that can expand and contract given on demand, and it does take more effort than the container instance to configure and setup.

New terms

- **ACI:** Azure Container Instance
- **AKS:** Azure Kubernetes Service
- **Deployment:** A way to deliver work into production
- **Concurrent Operations:** Also referred to as "*concurrency*", it is the number of operations to run at the same time

Further reading

Azure's [tutorial for how to setup authentication](#) goes into some key details of the deployment configuration.

9. Deploy an Azure Machine Learning model

<https://photos.app.goo.gl/DUdxQSP9ziR3Rfu59>

Summary

The primary task as a Machine Learning engineer is to ship models into production. Constant evaluation allows identifying potential issues and creating a baseline so that adapting or updating is possible.

Some key steps to deploy a model are:

- A previously trained model
- Complete the deployment form
- Enable authentication
- Select or create a new compute cluster

New terms

- **Constant Evaluation:** A feedback loop that is necessary to detect potential issues

10. Exercise: Deploy an Azure Machine learning Model

Part 1: Configure Deployment Settings

<https://photos.app.goo.gl/AHiMYtjpu4vSBnrg9>

Summary

In this demo, we covered how to configure a few deployment settings and explored the flexibility of Azure ML studio. Although you didn't deploy anything yet, this is a good foundation for the next steps in this lesson. The time required to run the experiment varies depending on the configurations you choose.

Remember that the *maximum nodes* configured in your cluster must be greater than the number of *concurrent operations* in your experiment.

Part 2: Deploy an Azure Machine Learning Model

After the experiment is completed, you will have many models available. Now, we can deploy some of the models so others can make use of them too.

<https://photos.app.goo.gl/NAV9iGbyFkDr65Rj9>

Summary

This demo showed you how to deploy an Azure Machine learning model that was previously trained. You need to fill out the deployed model form, enable authentication, and select the Azure container instance to compute type.

Exercise: Deploy an Azure ML model

Configuring deployment settings is a useful way to explore the flexibility of a cloud provider like Azure. In this exercise, you will first configure a deployment in Automated ML. After the experiment run is completed, you will deploy a model so it can be consumed.

Part 1

In Azure ML Studio, train a model using the [bike-no.csv](#) dataset with Automated ML, create a new compute cluster, and run the AutoML experiment. The dataset can be found on the lab desktop or in the course GitHub repo.

Hint: There are many ways to upload the data. You can copy the link above and paste it to the Azure ML studio and the data will be loaded from this link. Or you can upload the CSV file directly to Azure ML studio.

Note: Be careful with your configuration if you use the lab provided to you. **Your lab will timeout after a certain amount of time. Please keep track of the time remaining. It can be found in the lab guidance sidebar on the right side of the lab screen!**

Please make sure you complete the following



- Create a New Automated ML run
- Create and configure a new compute cluster and select 1 as the number of minimum nodes
- Upload and select the bike dataset
- Select `cnt` as the target column
- Configure Auto ML run using the newly created cluster
- Successfully run the Auto ML experiment
- Find the best model

Part 2:

After the `bike-no.csv` dataset has been trained with Automated ML, the deployment of the model happens next. This allows for consuming this model.

In this exercise, please deploy a model into a production environment using **Azure Container Instance (ACI)**. You will find trained models in the *Assets* section under *Models* in Azure ML Studio. Select one that is readily available.

Note: Make sure you enable *Authentication*, to prevent unauthorized access.

Please complete the following items to deploy a model:



- Select any model for deployment
- Deploy the model and enable "Authentication"
- Deploy the model using Azure Container Instance (ACI)
- Verify that the model is deployed as the "Deploy status" is shown as succeed or healthy

11. Solution: Deploy an Azure Machine Learning Model

Part 1: Configure deployment settings

1. Create a new Automated ML run

The screenshot shows the Microsoft Azure Machine Learning studio interface. On the left is a navigation sidebar with options like 'New', 'Home', 'Notebooks', 'Automated ML (preview)', 'Designer (preview)', 'Datasets', 'Experiments', 'Pipelines', 'Models', 'Endpoints', 'Compute', 'Datastores', and 'Data Labeling'. The main area is titled 'Welcome to the studio!' and features several cards: 'Create new' (Notebook, Automated ML run (preview), Pipeline (preview), Dataset), 'Notebooks' (Start now), 'Automated ML (preview)' (Start now), 'Designer (preview)' (Start now), 'Tutorials' (Compute instance, Training cluster, Datastore, Data labeling project), 'Train your first ML model with Notebook', 'Create, explore and deploy Automated ML experiments', 'What are compute targets in Azure Machine Learning?', and 'Deploy models with Azure Machine Learning'. At the bottom, there are 'Links' for 'Blog' and 'Documentation'.

2. Next, make sure you have the dataset uploaded. If you don't, upload and select it. This solution uses the [bike-no.csv](#) dataset.

Create dataset from local files

Basic info

Datastore and file selection (selected)

Settings and preview

Schema

Confirm details

Datastore and file selection

Select or create a datastore *

Currently selected datastore: workspaceblobstore (Azure Blob Storage) (Default)

Previously created datastore

Create new datastore

Select files for your dataset *

After dataset creation, these files will be uploaded to your default Blob storage and made available in your workspace. Supported file types include: delimited (i.e. csv, tsv), Parquet, JSON Lines, and plain text.

Browse 1 files selected. Total size 3.958 MiB. 0/1 files uploaded

| File name | Size (MiB) | Upload % | Status |
|-------------------------|------------|----------|--------|
| bankmarketing_train.csv | 3.958 | | |

Upload path

UI Files will be uploaded to '\$(Upload path)/07-25-2020_124429_UTC'

Skip data validation ⓘ

3. Create and configure your new compute cluster.

Create a new Automated ML run

Select dataset

Configure run (selected)

Task type and settings

Configure run

bike-no ([View dataset](#))

Experiment name *

Create new

New experiment name

demo-experiment

Target column * ⓘ

Column14

Select compute cluster * ⓘ

Select a Compute...

Compute cluster is required

[Create a new compute](#) ⏪ Refresh compute

Create a new Automated ML run

Configure run

Configure the experiment. Select from experiment [View dataset](#)

Dataset
bike-no ([View dataset](#))

Experiment name *
 Create new
New experiment name
demo-experiment

Target column * Column14

Select compute cluster * demo-cluster

Select a Compute...
Compute cluster is required
[Create a new compute](#) [Refresh](#)

[Back](#) [Next](#)

New compute cluster [View](#)

Virtual machine type * CPU (Central Processing Unit)

Virtual machine priority * [Dedicated](#) [Low priority](#)

Virtual machine size * Standard_DS3_v2 4 Cores, 14 GB (RAM), 28 GB (Disk)

Minimum number of nodes * 1 [Creating...](#)

To avoid charges when no jobs are running, set the minimum nodes to 0. This setting allows Azure Machine Learning to de-allocate the compute nodes when idle. Any higher value will result in charges for the number of nodes allocated.

Maximum number of nodes * 5

Idle seconds before scale down * 120

[Advanced settings](#)

[Download a template for automation](#) [Create](#) [Cancel](#)

- Once the new compute cluster is successfully created, use this cluster to run the autoML experiment. Make sure you fill the name and target column.

Create a new Automated ML run

Configure run

bike-no ([View dataset](#))

Experiment name *
 Create new
New experiment name
demo-experiment

Target column * Column14

Select compute cluster * demo-cluster

Select a Compute...
Compute cluster is required
[Create a new compute](#) [Refresh](#)

[Back](#) [Next](#) [Cancel](#)

- You will see the experiment in the experiment section and a new model is created.

Experiments

| Experiment | Latest run | Last submitted ↓ | Created |
|-----------------|------------|---------------------|---------------------|
| demo-experiment | 3 | Sep 3, 2020 9:43 AM | Sep 3, 2020 9:40 AM |

Part 2: Deploy an Azure ML model

1. Go to the *Automated ML* section and find the recent experiment with a completed status. Click on it.

The screenshot shows the Microsoft Azure Machine Learning studio interface. On the left, there is a navigation sidebar with various options like New, Home, Notebooks, Automated ML (preview), Designer (preview), Datasets, Experiments, Pipelines, Models, Endpoints, Compute, Datastores, and Data Labeling. The 'Automated ML (preview)' option is currently selected. The main content area displays the details of a specific experiment run. The top navigation bar shows the path: course-2 > Automated ML (preview) > ml-experiment2 > Run 1. Below this, the title 'Run 1' is followed by a green checkmark and the word 'Completed'. There are refresh and cancel buttons. The main content is divided into tabs: Details (which is selected), Data guardrails, Models, Outputs + Logs, Child runs, and Snapshot. The 'Properties' section contains the following information:

- Status: Completed
- Created: Jul 29, 2020 8:53 AM
- Duration: 15m 48.21s
- Compute target: auto-ml
- Run ID: AutoML_56d2d734-96df-4a95-af6b-4bac1bca375e
- Run number: 1
- Script name: --
- Created by: Udacity Instructor 2
- Input datasets: Input name: input_data, ID: a4308302-82f2-4d4a-bb71-1090e1ebfc2
- Output datasets: None
- Arguments: None

At the bottom of the properties section, there are links for 'See all properties' and 'Raw JSON'.

2. Go to the "Model" tab and select a model from the list and click it. Above it, a triangle button (or Play button) will show with the "Deploy" word. Click on it.

Then

1) Fill out the form with a meaningful name and description. For Compute Type use **Azure Container Instance (ACI)**

2) Enable Authentication

3) Do not change anything in the Advanced section.

3. Deployment takes a few seconds. After a successful deployment, a green checkmark will appear on the "Run" tab and the "Deploy status" will show as succeed.

12. Quiz: Configure Deployment Settings

QUESTION 1 OF 3

What exactly is a deployment?

Delivers a trained model into production

Defines the CPU and memory requirements

Creates different ways to consume a model

Enables authentication



[Try Again](#)

Selecting CPU sizes or RAM requirements are more related to configuring a deployment

A deployment itself will not end up enabling authentication, which is a configuration or a setting, rather than a deployment itself

[TRY AGAIN](#)



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure



Try Again

Selecting CPU sizes or RAM requirements are more related to configuring a deployment

Sometimes you can deploy locally, while other times, a deployment goes into production, creating different ways to consume a model.

TRY AGAIN

QUESTION 2 OF 3

Match the deployment-related events to identify which happen before deployment and which happen afterward.

Submit to check your answer choices!

| DEPLOYMENT EVENT | WHEN IT HAPPENS |
|---------------------------|-----------------|
| Selecting machine types | Before |
| Defining cluster settings | Before |
| Authenticate and interact | After |
| Enable Authentication | Before |
| Retrieve logs | After |



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

New compute cluster ⓘ

Compute name * ⓘ

Region * ⓘ

Virtual machine type *

Virtual machine priority * ⓘ

Dedicated Low priority

Virtual machine size * ⓘ

Minimum number of nodes * ⓘ

Maximum number of nodes * ⓘ

Idle seconds before scale down * ⓘ

› Advanced settings

Maximum Number of Nodes in a new compute cluster

QUESTION 3 OF 3

What should the maximum number of nodes be when creating a cluster if the number of concurrent operations (also referred to as "concurrency") of the experiment is 5?

5

6

1

0



Try Again

0 wouldn't be enough to run any jobs at all, and the form wouldn't allow creating a cluster with no nodes.

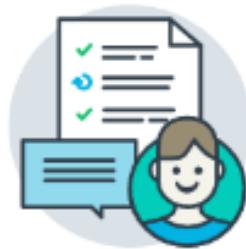
TRY AGAIN



Try Again

Since concurrency is set to 5, it needs more than 1 node to run on.

TRY AGAIN



Try Again

5 nodes wouldn't be enough and would produce an error, even though the concurrency is the same value (5).

TRY AGAIN

13. Quiz: Deploy an Azure Machine Learning model

Train Deploy a New Model



Try Again

Training a model is good, but it happens long before deploying a model. Thus, it's not the objective of deploying a model.

TRY AGAIN

A screenshot of the Azure Machine Learning studio interface. On the left, there is a sidebar with two main sections: 'Author' and 'Assets'. Under 'Author', there are three items: 'Notebooks', 'Automated ML (preview)', and 'Designer (preview)'. Under 'Assets', there are five items: 'Datasets', 'Experiments', 'Pipelines', 'Models', and 'Endpoints'. Each item has a corresponding blue icon next to it.

- Author
 - Notebooks
 - Automated ML (preview)
 - Designer (preview)
- Assets
 - Datasets
 - Experiments
 - Pipelines
 - Models
 - Endpoints

Finding a deployed Model

QUESTION 2 OF 3

In Azure Machine Learning Studio, after deploying a model, in details of the deployed model?

Endpoints

Experiments

Automated ML

Models



Try Again

The Models section display all models available for deployment, but not deployed models

[TRY AGAIN](#)



Try Again

The Automated ML section will have details of recent automated ML runs, but not about deployed models

TRY AGAIN

QUESTION 3 OF 3

What would the end result be after deploying a model?

The most efficient model from Auto ML

An HTTP endpoint

A configured cluster

A group of powerful algorithms



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure



Try Again

Auto ML can provide you a list of the best algorithms for a model. This doesn't happen by deploying a model

TRY AGAIN



Try Again

Before deploying a model, the most efficient model is already known!

TRY AGAIN



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

14. Enable Application Insights

<https://photos.app.goo.gl/wA1Wp1uw7D7ai22Y7>

Summary

In this section, we discussed Application Insights that is a very useful tool to detect anomalies, visualize performance. It can be enabled before or after a deployment. To enable Application Insights after a model is deployed, you can use the below command with the python SDK. In the next section, you will learn how to do it.

```
# enable application insight
service.update(enable_app_insights=True)
```

New terms

- **Logging:** Informational output produced by the software, usually in the form of text
- **Application Insights:** A special Azure service which provides key facts about an application
- **Webservice:** One of the most used Python classes from Azure's Python SDK

Further reading

Azure's documentation has a whole section on monitoring and collecting data with [Application Insights](#)

15. Exercise: Enable Application Insights

Note: This lab will pre-load a trained model and deploy it for you so you don't have to repeat what you did in the previous lab. You probably have noticed that it takes you more than 30 minutes to train models using Automated ML. Similarly, this lab will take about 30 minutes to load because it trains and deploys the model behind the scene. While the lab is loading, leave the page open and take a break. Come back in 30 minutes and the lab will be ready for you.

You may see some scripts running in the terminal (PowerShell). This is used for model training and deployment. **DO NOT** close the terminal. It will close automatically once the deployment is done. And once it is done, you can see an endpoint that is **NotNull**.

<https://photos.app.goo.gl/yDgkG65XbQtSUV5z5>

Summary

In this demo, you learned how to enable Application Insights from a deployed model and then produced logging output with the Python SDK. The following command will enable the logging for you. You can find a `logs.py` that contains part of the codes in the `nd00333_AZMLND_C2-master/Exercise_starter_files` in the Github Repo or in the lab desktop folder.

```
from azureml.core import Workspace
from azureml.core.webservice import Webservice

# Requires the config to be downloaded first to the current working directory
ws = Workspace.from_config()

# Set with the deployment name
name = "demo-model-deploy"

# load existing web service
service = Webservice(name=name, workspace=ws)

# enable application insight
service.update(enable_app_insights=True)

logs = service.get_logs()

for line in logs.split('\n'):
    print(line)
```

The line where the variable `name` is defined will need to match the model to be deployed. Also, ensure that `enable_app_insights` is set to `True` so that the service can have this enabled in Azure Machine Learning Studio.

The line where the variable `name` is defined will need to match the model to be deployed. Also, ensure that `enable_app_insights` is set to `True` so that the service can have this enabled in Azure Machine Learning Studio.

Exercise: Enable Application Insights

In this exercise, you will enable Application Insights and retrieve logs from a deployed model. Enabling *Application Insights* is vital to debug problems in production environments.

Find a deployed model endpoint, verify that the Application Insights is false, then use the **Python SDK** to enable Application Insights, which should in turn enable logging.

You are recommended to write the code by yourself but if you stuck, you can find a `logs.py` in the exercise starter_files in the [Github Repo](#) or in a folder on the lab desktop.

Please complete the following using Azure Python SDK



- ✓ Find a deployed model in ML studio and verify the Application Insight is False
- ✓ Download `config.json` from ML Studio
- ✓ Write or update code to enable logging for a deployed model
- ✓ Run the code to view the logs and enable Application Insights
- ✓ "Application Insights" is enabled in the Details tab of the endpoint.

16. Solution: Enable Application Insights

Solution: Enable Application Insights

1. Download the `config.json` file from the top left menu in the Azure portal. Put this file in the same directory of other files needed for this exercise.

The screenshot shows the Azure Cloud Shell interface with the following configuration settings:

- Current directory:** Udacity
- Current subscription:** Azure Sponsorship - Udacity-Testing
- Current workspace:** course-2
- Resource Group:** cloud-shell-storage-eastus
- Location:** eastus
- Download config file:** A link to download the configuration file from the Azure Portal.

2. Find the previously deployed model to verify its name. It is needed in the SDK to select it for enabling logging. In this example, `exercise-deployment-1` is the name of the service. This information is available from the *Endpoints* section.

Note: In the *Details* tab, at the very end, "*Application Insights enabled*" should be `false`. The SDK will get it enabled next.

Microsoft Azure Machine Learning

course-2 > Endpoints > exercise-deployment-1

exercise-deployment-1

Details Consume

| Attributes | |
|----------------------------------|--|
| Service ID | exercise-deployment-1 |
| Description | -- |
| Deployment state | Healthy ⓘ |
| Compute type | ACI |
| Created by | Udacity Instructor 2 |
| Model ID | AutoML5d2d734944:1 |
| Created on | 8/16/2020 10:09:36 AM |
| Last updated on | 8/16/2020 10:09:36 AM |
| Image ID | -- |
| REST endpoint | <input type="text" value="http://666fac5d-5231-4e8a-adc2-825e673032c3.eastus.azurecontainer.io/score"/> Copy |
| Key-based authentication enabled | true |
| Swagger URI | http://666fac5d-5231-4e8a-adc2-825e673032c3.eastus.azurecontainer.io/swagger.json |
| CPU | 1 |
| Memory | 2 GB |
| Application Insights enabled | false |

3. In the `starter_code` there is a `logs.py` file which should allow you to:

- 1) Dynamically authenticate to Azure
- 2) Enable Application Insights
- 3) Display logs

It should be similar to this:

You need to change the name to match your experiment, save it, and run it. The log output should display some output.

```
from azureml.core import Workspace
from azureml.core.webservice import Webservice

ws = Workspace.from_config()

# load existing web service
service = Webservice(name="exercise-deployment-1", workspace=ws)
logs = service.get_logs()

for line in logs.split('\n'):
    print(line)
```

```
from azureml.core import Workspace
from azureml.core.webservice import Webservice

ws = Workspace.from_config()

# load existing web service
service = Webservice(name="exercise-deployment-1", workspace=ws)
logs = service.get_logs()

for line in logs.split('\n'):
    print(line)
```

4. To enable insights, the following line needs to be added to the script:

```
service.update(enable_app_insights=True)
```

```
service.update(enable_app_insights=True)
```

Run it once again to enable insights. The Attributes page should show Insights is now enabled for the deployed model.

Microsoft Azure Machine Learning

course-2 > Endpoints > exercise-deployment-1

exercise-deployment-1

Attributes

Service ID
exercise-deployment-1

Description
--

Deployment state
Healthy ⓘ

Compute type
ACI

Created by
Udacity Instructor 2

Model ID
[AutoML56d2d734944:1](#)

Created on
8/16/2020 10:09:36 AM

Last updated on
8/16/2020 11:02:07 AM

Image ID
--

REST endpoint
<http://666fac5d-5231-4e8a-adc2-825e673032c3.eastus.azurecontainer.io/score> 

Key-based authentication enabled
true

Swagger URI
<http://666fac5d-5231-4e8a-adc2-825e673032c3.eastus.azurecontainer.io/swagger.json>

CPU
1

Memory
2 GB

Application Insights enabled
true

Application Insights url
<https://portal.azure.com#@microsoft.onmicrosoft.com/resource/subscriptions/a59af59a-605a-4b0e-968f-a30ad6bb7ad5/resourcegroups/cloud-shell-storage-eastus/providers/microsoft.insights/components/course23255986363>

17. Quiz: Enable Application Insights

QUESTION 1 OF 3

What are some important things about logging?

- It helps detect anomalies
- Shows error conditions as they happen
- Identifies the best algorithm
- Informs about different pipelines available

[Try Again](#)

Logging can help detect anomalies by looking at behavior that wasn't common before.

When an error condition happens, like an unhandled exception, logging will show it immediately.

Logs provide information about anomalies and uncommon behavior. To get an overview of different pipelines, Azure ML Studio has a "Pipelines" section.

[TRY AGAIN](#)



Try Again

Logging can help detect anomalies by looking at behavior that wasn't common before.

When an error condition happens, like an unhandled exception, logging will show it immediately.

Identifying the best algorithm is a job for AutoML or a pipeline.

TRY AGAIN



Try Again

Logging can help detect anomalies by looking at behavior that wasn't common before.

TRY AGAIN



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure



Try Again

When an error condition happens, like an unhandled exception, logging will show it immediately.

TRY AGAIN

```
1 from azureml.core import Workspace
2 from azureml.core.pipelines import Pipeline
3
4 ws = Workspace.from_config()
5
6 service = Pipeline(name="exercise-deployment-1", workspace=ws)
7 service.update(enable_logging=True)
```

A

```
1 from azureml.core import Workspace
2 from azureml.core.webservice import Webservice
3
4 ws = Workspace.from_config()
5
6 service = Webservice(name="exercise-deployment-1", workspace=ws)
7 service.update(enable_logging=True)
```

B

```
1 from azureml.core import Workspace
2 from azureml.core.webservice import Webservice
3
4 ws = Workspace.from_config()
5
6 service = Webservice(name="exercise-deployment-1", workspace=ws)
7 service.update(enable_app_insights=True)
```

C

QUESTION 2 OF 3

What is the right code example to enable logging for the `"exercise-deployment-1"` deployed service?

A

B

C



Try Again



Try Again

Image B doesn't enable Application Insights which is needed for logging

[TRY AGAIN](#)

Image A does not seem to enabled Application Insights to get logging working

[TRY AGAIN](#)

QUESTION 3 OF 3

What are some important features of Application Insights?

Detects performance Anomalies

Detects performant Algorithms

Includes Analytics Tools

Includes Pipeline tools



Try Again

Application Insights can detect anomalies and uncommon behavior.

Application Insights contains many Analytic tools to detect anomalies and uncommon behavior.

Application Insights helps detect anomalies and uncommon behavior. But pipeline tools can not be used to detect anomalies.



Try Again

Application Insights can help detect anomalies and uncommon behavior. A performant algorithm is not an anomaly or uncommon behavior.

Application Insights helps detect anomalies and uncommon behavior. But pipeline tools can not be used to detect anomalies.

TRY AGAIN



Try Again

Application Insights contains many Analytic tools to detect anomalies and uncommon behavior.

TRY AGAIN

18. Troubleshoot Deployment Issues

<https://photos.app.goo.gl/1GkohVjTQ4Ymb9v56>

Summary

In this section, we covered different techniques and diagnosis that you can use to identify potential issues like unhandled exceptions from a deployed service. Using local deployment is a special technique, which makes it easier to identify some of these potential issues.

Common HTTP errors:

- 502: the application crashes because of an unhandled exception.
- 503: there are large spikes in requests and the system is not able to cope with all of them.
- 504: request timed out.

Deploy Locally

To deploy locally using the Python SDK you will need to use the `LocalWebService` class and configure it for a local deployment

```
from azureml.core.webservice import LocalWebService

deployment_config = LocalWebService.deploy_configuration(port=9001)
# Deploy the service
service = Model.deploy(ws, "local-service", [model], inference_config, deployment_config)

service.reload()
print(service.run(input_data=json_data))
```

```
from azureml.core.webservice import LocalWebService

deployment_config = LocalWebService.deploy_configuration(port=9001)
# Deploy the service
service = Model.deploy(ws, "local-service", [model], inference_config,
deployment_config)

service.reload()
print(service.run(input_data=json_data))
```

Deploying locally has some benefits. First, it is easier and faster to verify unhandled exceptions from the scoring script since you don't have to wait for deployment in Azure. Also, many people or teams can debug at the same time.

New terms

- **HTTP Status code:** A number that represents a status when an HTTP server responds. Error conditions in the server side start at 500

Further reading

The [troubleshoot deployment of models section](#) of Azure's documentation covers some of these troubleshooting concepts in detail.

19. Exercise: Troubleshoot Deployment Issues

Being able to troubleshoot is useful, but knowing what to expect in certain error conditions is even more important.

Reflect

Having an idea of what the problem may be ahead of time is crucial. For this exercise, describe 2 things you can expect to go wrong, and 1 thing you can do to debug a deployed container service.

aaa

SUBMIT

20. Solution: Troubleshoot Deployment Issues

There are multiple things you can expect to go wrong. When you submit HTTP requests to a deployed model, there are three HTTP codes that you may encounter:

- **HTTP STATUS 502:** After a deployment, the application crashes because of an unhandled exception.
- **HTTP STATUS 503:** When there are large spikes in requests, the system may not be able to cope with all of them and some clients may see this code.
- **HTTP STATUS 504:** The request timed out. In Azure, the requests time out after 1 minute. If the `score.py` script is taking longer than a minute, this error code will be produced.

When an error code shows up, one thing you can do is retrieving the logs output. Logs output is *always* useful to debug problems in deployed containers. Showing below is an extract of what you should see in a successful response to a scoring request.

```
Validation Request Content-Type
Received input: {'data': [{'instant': 1, 'date': '2011-01-01 00:00:00,000000',
Headers passed in (total 12):
Host: localhost:5001
X-Real-Ip: 127.0.0.1
X-Forwarded-For: 127.0.0.1
X-Forwarded-Proto: http
Connection: close
Content-Length: 812
User-Agent: ApacheBench/2.3
Accept: */*
Authorization: Bearer q8szMDbCoNlxDZCpiGI8tnqaxtClyDiy
Content-Type: application/json
X-Ms-Request-Id: 7cb6f8b9-e511-43b7-982f-e413d6e3239d
Accept-Encoding: gzip
Scoring Timer is set to 60.0 seconds
200
```

```
Validation Request Content-Type
Received input: {'data': [{instant': 1, 'date': '2011-01-01
00:00:00,000000', 'season': 1, 'yr': 0, 'mnth': 1, 'weekday': 6,
'weathersit': 2, 'temp': 0.344167, 'atemp': 0.363625, 'hum': 0.805833,
'windspeed': 0.160446, 'casual': 331, 'registered': 654 }]}
Headers passed in (total 12):
Host: localhost:5001
X-Real-Ip: 127.0.0.1
X-Forwarded-For: 127.0.0.1
X-Forwarded-Proto: http
Connection: close
Content-Length: 812
User-Agent: ApacheBench/2.3
Accept: */*
Authorization: Bearer q8szMDbCoNlxDZCpiGI8tnqaxtClyDiy
Content-Type: application/json
X-Ms-Request-Id: 7cb6f8b9-e511-43b7-982f-e413d6e3239d
Accept-Encoding: gzip
Scoring Timer is set to 60.0 seconds
200
```

21. Quiz: Troubleshoot Deployment Issues

| System is unhealthy | | |
|---------------------|--------------------------------------|--|
| HTTP RESPONSE CODE | WHY IS THIS RESPONSE HAPPENING? | |
| 502 | Unhandled Exception | |
| 503 | System is unable to cope with demand | |
| 504 | Request time out | |

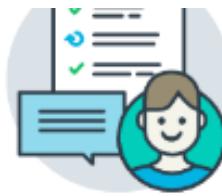


Try Again

502 will happen when the application is having exceptions

503 is returned when the system is having trouble coping with the demand

TRY AGAIN



Try Again

504 happens when the request is taking more than 60 seconds, the Azure ML limit

TRY AGAIN



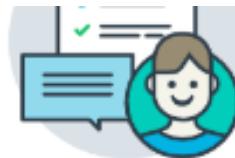
Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

QUESTION 2 OF 3

What are the key features of deploying locally that can help troubleshoot?

- Easier to verify unhandled exceptions
- Identifies the best algorithm
- It is faster to deploy than waiting for Azure
- Only one person can debug at the time



[Try Again](#)

By deploying locally, an unhandled exception will be immediately observed, as the application will not start at all.

One advantage of deploying locally is distributing the load of troubleshooting problems with the application to many people.

[TRY AGAIN](#)



[Try Again](#)

By deploying locally, an unhandled exception will be immediately observed, as the application will not start at all.

[TRY AGAIN](#)

[Try Again](#)

By deploying locally, an unhandled exception will be immediately observed, as the application will not start at all.

A local deployment deploys a trained model. The best algorithm is identified when a model is being trained.

When deploying locally, all the resources are ready and allocated for the application so you don't have to wait.

[TRY AGAIN](#)



Microsoft Azure
Noel Ching

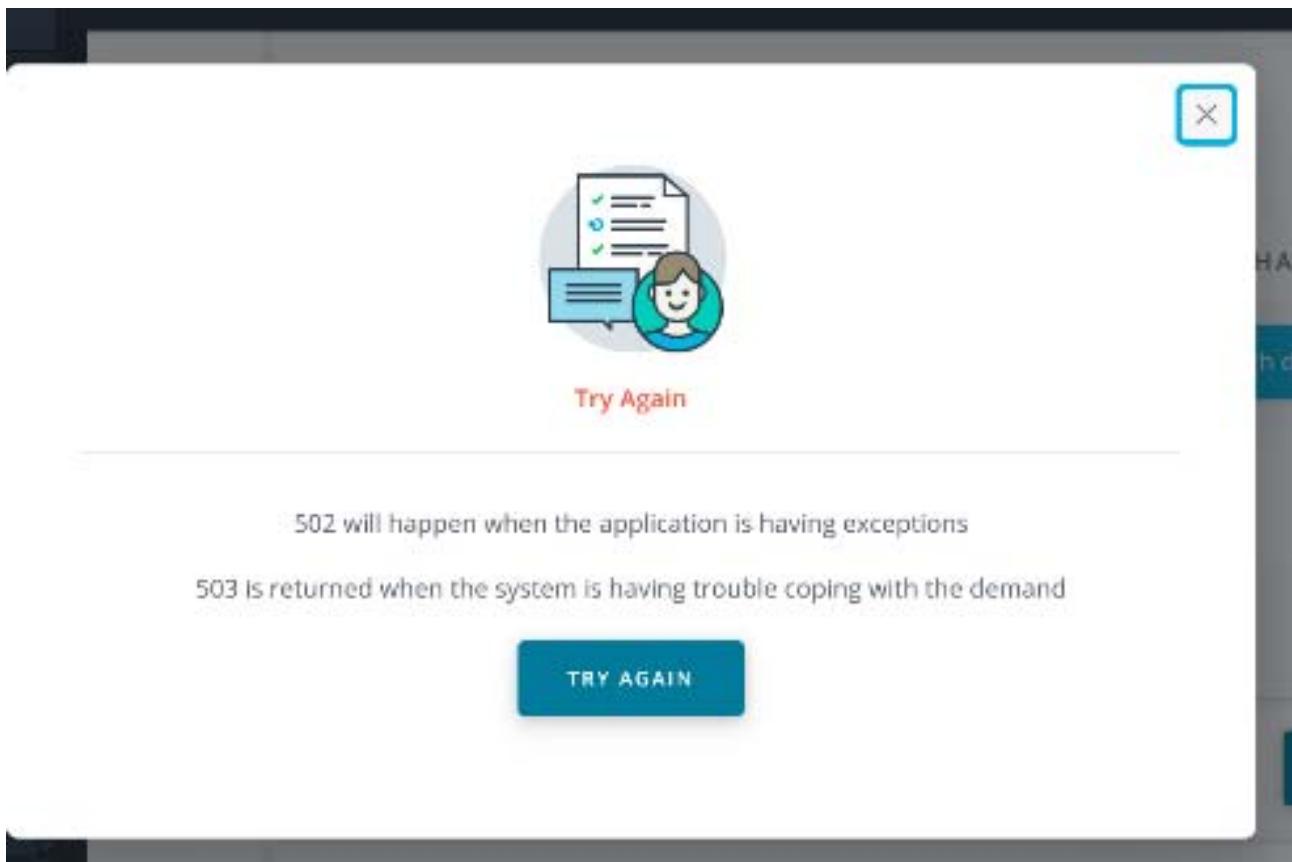
Machine Learning Engineer for Microsoft Azure



Try Again

When deploying locally, all the resources are ready and allocated for the application so you don't have to wait.

TRY AGAIN



502 will happen when the application is having exceptions

503 is returned when the system is having trouble coping with the demand

TRY AGAIN



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

QUESTION 3 OF 3

Which option will be the right one to apply for a local deployment using the Python SDK?



```
deployment_config =  
Webservice.deploy_configuration(local_port=8890)
```



```
deployment_config = LocalWebservice.deploy_configuration(port=8890)
```



```
deployment_config =  
LocalPipeline.deploy_configuration(local_port=8890)
```



Try Again

A `Webservice` class cannot do local deployments

TRY AGAIN



Try Again

There are no `LocalPipelines()` available that can do local deployments

TRY AGAIN

22. Key Benefits of Compute Instances

<https://photos.app.goo.gl/npd42dfYJqpHsv569>

Summary

Compute Instances are a managed, cloud-based workstation. It can be used in Azure Machine Learning Studio. This is a great way to take advantage of preloaded machine dependencies and not being responsible for their management.

New terms

- **Compute Instance:** A distinct type of a compute offering from Azure
- **Cloud-based workstation:** Sometimes, compute instances are referred to as a cloud-based workstation because it is ready to start developing

Further reading

Azure's documentation on "[What is a compute instance](#)" defines some other key features of their offering.

23. Exercise: Key Benefits of Compute Instances

Reflect

Being able to define the key benefits of different computing instances is important so that you know how to choose the right one use later. In this exercise, describe 3 benefits of using compute instances.

aaa

SUBMIT

24. Quiz: Key Benefits of Compute Instances

QUESTION 1 OF 3

Choose some of the benefits of compute instances

Fully managed

Useful for HTTP endpoints

Security and compliance are taken care of

Elastic storage

Compute instances are managed by Azure, no need to worry about any maintenance.

Compute instances are cloud-based workstations. Its benefits do not come from deployment or consuming endpoints.

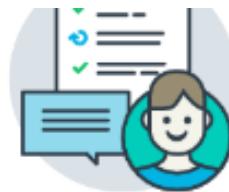
Security and compliance are closely checked by Azure on compute instances.

Although you can select different storage sizes, there is no elastic storage provided with compute instances.

Try Again

Compute instances are managed by Azure, no need to worry about any maintenance.

TRY AGAIN



Try Again

Compute instances are managed by Azure, no need to worry about any maintenance.

Security and compliance are closely checked by Azure on compute instances.

Although you can select different storage sizes, there is no elastic storage provided with compute instances.

TRY AGAIN



Try Again

Compute instances are managed by Azure, no need to worry about any maintenance.

Compute instances are cloud-based workstations. Its benefits do not come from deployment or consuming endpoints.

Security and compliance are closely checked by Azure on compute instances.

TRY AGAIN



Try Again

Security and compliance are closely checked by Azure on compute instances.

TRY AGAIN



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

QUESTION 2 OF 3

Match benefits to their corresponding target

Compute Cluster

| BENEFITS | TARGET TYPE |
|-------------------------|------------------|
| Share and Collaborate | Compute Instance |
| Deploy to production | Compute Cluster |
| Pre-installed Libraries | Compute Instance |
| Ready-to-try examples | Compute Instance |



Try Again

Typically a workstation will provide a lot of pre-installed libraries.

TRY AGAIN



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

[Try Again](#)

Sharing and collaborating is a good benefit of a **cloud-based** tool

Deploying to production requires a cluster that is ready to run.

Typically a workstation will offer many ready-to-try examples

[TRY AGAIN](#)

QUESTION 3 OF 3

What is a good use case for a compute instance?

- HTTP Endpoint
- Jupyter Notebook
- Batch inference Pipeline
- Deploying a local container



Try Again

A local container does not need a compute instance.

TRY AGAIN



Try Again

A batch inference pipeline is provided by the pipeline class in Azure

TRY AGAIN

Try Again

An HTTP endpoint is going to be provided by Azure when deploying a model to production

TRY AGAIN

25. Glossary

- **ACI:** Azure Container Instance
- **AKS:** Azure Kubernetes Service
- **Application Insights:** A special Azure service which provides key facts about an application
- **CI/CD:** Continuous Integration and Continuous Delivery platform. Jenkins, CircleCI, and Github Actions, are a few examples
- **Cloud-based workstation:** Sometimes, compute instances are referred to as a cloud-based workstation, because it is ready to start developing
- **Compute Instance:** A distinct type of a compute offering from Azure
- **DevOps:** A set of best practices that helps provide continuous delivery of software at the highest quality with a constant feedback loop
- **Deployment:** A way to deliver work into production
- **Endpoint:** A part of an HTTP API. Either a full URL or a partial URL identifying a part
- **HTTP API:** A URL that exposes logic to interact with software, in this case, a trained model
- **HTTP Status code:** A number that represents a status when an HTTP server responds. Error conditions in the server side start at 500
- **Logging:** Informational output produced by software, usually in the form of text
- **Shipping into production:** The most important aspect of a Machine Learning specialist
- **Webservice:** One of the most used Python classes from Azure's Python SDK

26. Further Reading

These are some interesting links related to DevOps, CI/CD, and configuration in Azure ML Studio:

- [Chapter 7 of Python For DevOps](#) has an excellent section about logging and troubleshooting
- Both the [Jenkins](#) and [Github Actions](#) websites have good information about their CI/CD platforms and why they are compelling
- Azure's [tutorial for how to setup authentication](#) goes into some key details of the deployment configuration
- Azure's documentation has a whole section on monitoring and collecting data with [Application Insights](#)
- The [troubleshoot deployment of models section](#) of Azure's documentation covers some of these troubleshooting concepts in detail
- Azure's documentation on "[What is a compute instance](#)" defines some other key features of their offering

27. Lesson Review

<https://photos.app.goo.gl/QhH3uAEFXeWiNAWi9>

Summary

In this lesson, we went through a lot of useful aspects of getting a model into production. Remember that getting a model into production is one of the most important tasks as an ML specialist while taking care of security and configuration.

We also covered different troubleshooting techniques, including deploying locally and using log output to identify issues.

Lesson 3: Consume Endpoints

1. Introduction

<https://photos.app.goo.gl/BVZzkNzJQGRdBbiP6>



Summary

This is the lesson about Consuming Endpoints. These endpoints allow other services to interact with deployed models. And in this lesson, you will learn all the key facts about interacting with them.

There are some interesting details you need to be aware of when trying to use HTTP and you will go through each of these:

- Swagger
- Consuming deployed services
- Benchmarking

New terms

- **Swagger:** A tool that eases the documentation efforts of HTTP APIs
- **Benchmarking:** being able to create a baseline of acceptable performance so that it can be compared to day-to-day behavior

2. Pre-launch the Labs

In this lesson, you will have three labs: *Swagger Documentation*, *Consume Deployed Services*, and *Benchmark The Endpoint*. Just like the *Enable Application Insights* lab you had before, these three labs will train and deploy a model for you so that you can focus on learning new material without training and deploying a model again.

✓ **5. Exercise: Swagger Documentation**

✓ 6. Solution: Swagger Documentation

✓ 7. Quiz: Swagger Documentation

✓ 8. Consume Deployed Service

✓ **9. Exercise: Consume Deployed Serv...**

✓ 10. Solution: Consume Deployed Ser...

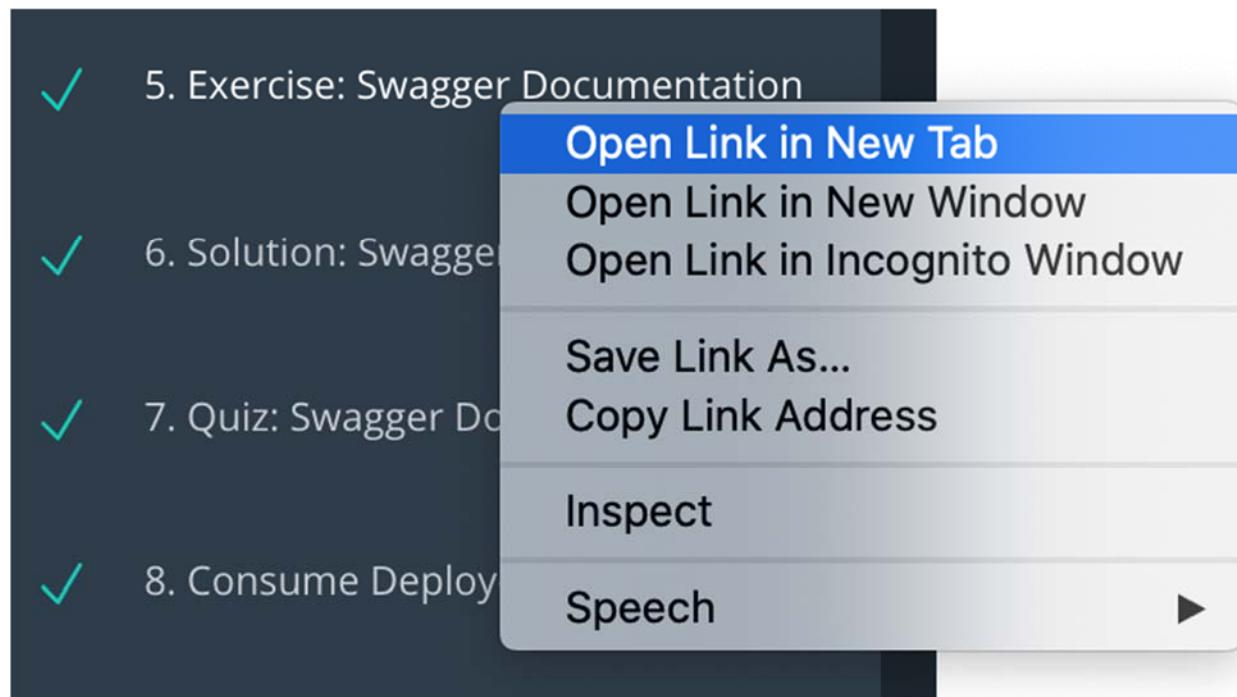
✓ 11. Quiz: Consume Deployed Service

✓ 12. Benchmark the Endpoint

✓ **13. Exercise: Benchmark the Endpoint**

Each of the three labs will take about 30 minutes to load and deploy the model. As mentioned before, you can take a break while the lab is loading and come back in 30 minutes. Or you can open the exercise page in a new tab and launch it beforehand.

Once again, you may see some scripts running in the terminal (PowerShell). This is used for model training and deployment. **DO NOT** close the terminal. It will close automatically once the deployment is done. And once it is done, you can see an endpoint that is **Not Null**.



Open the exercise page in a new tab and launch it beforehand

3. Creating a Benchmark

<https://photos.app.goo.gl/ZgaDC9KPWWkH2hkQA>

Summary

In this lesson, you will interact with a deployed service and create a benchmark. Interacting and creating a benchmark is critical for enhancing performance and detecting anomalies.

Further reading

[Chapter 4 of Python For DevOps](#) covers benchmarking and other useful Linux utilities that can be useful.

4. Swagger Documentation

<https://photos.app.goo.gl/gmYm8gmR3SiRL7vc6>

Summary

Swagger is a tool that helps build, document, and consume RESTful web services like the ones you are deploying in Azure ML Studio. It further explains what types of HTTP requests that an API can consume, like POST and GET.

Azure provides a `swagger.json` that is used to create a web site that documents the HTTP endpoint for a deployed model.

The screenshot shows the Swagger UI interface for a POST request to the endpoint `/score`. The request body is a JSON object named `serviceInputPayload`, which is described as "The input payload for executing the real-time machine learning service". An example value is provided as a JSON snippet:

```
{  
  "data": [  
    {  
      "instant": 1,  
      "date": "2011-01-01 00:00:00,000000",  
      "session": 1,  
      "yr": 0,  
      "mth": 1,  
      "weekday": 6,  
      "weathersit": 2,  
      "temp": 0.344167,  
      "atemp": 0.363625,  
      "hum": 0.865833,  
      "windspeed": 0.160446,  
      "casual": 331,  
      "registered": 654  
    }  
  ]  
}
```

The "Parameter content type" dropdown is set to `application/json`.

Swagger example

New terms

- **RESTful:** A style for building HTTP endpoints that emphasizes separation of concerns

Further reading

[The Swagger homepage](#) has in-depth examples and usage to dig further into other details.

5. Exercise: Swagger Documentation

Note: This lab takes about 30 min to load. You may see some scripts running behind the scene to deploy a model for you. **DO NOT** close the terminal (PowerShell) that runs the scripts. It will close automatically once the deployment is done. Once the deployment is done, you can see an endpoint that is **NotNull**.

Now, let's look at how to use Swagger Documentation.

<https://photos.app.goo.gl/wNEGF5iTxDK1JC8>

Note: in the video, the instructor used `localhost` port 80 to display the **Swagger page**. It may not work for everyone. If `localhost` doesn't work for you, check if you can use a different port other than 80, for example, port 9000. Ensure that the updated port is used when trying to reach the swagger instance by `localhost`, for example `localhost:9000`.

If you see code 400, message bad request version in the Python script, it means that you are using `https` instead of `http` in the Swagger page. Remember to use `http` only when accessing these URLs.

Summary

In the `Swagger.sh` file, it has a command line:

```
docker run -p 80:8080 swaggerapi/swagger-ui
```

After the Swagger UI container is running, you can access the website on `http://localhost:9000`.

Running `serve.py` is crucial so that the contents of `swagger.json` can be consumed locally by Swagger. If `swagger.json` is not present, or if the local server is not running, then Swagger will not be able to produce the docs.

To give you more information: running `serve.py` is needed because Azure protects against CORS (Cross Origin Resource Sharing) and the server that hosts `swagger.json` needs to be allowed to happen. This is done in the script with the following method:

```
def end_headers(self):
    self.send_header("Access-Control-Allow-Origin", "*")
    SimpleHTTPRequestHandler.end_headers(self)
```

Note: the above information is not closely related to the course and will not be tested in the final project. It is just for those who are interested in the commands in `serve.py`.

By default, the `serve.py` script will run and serve contents on `localhost:8000` - this is an important detail because it is required as input in the Swagger UI page. The value that is required in the Swagger UI is `http://localhost:8000/swagger.json`. Please notice that you should use `http` instead of `https`.

Both `serve.py` and `swagger.sh` can be found in the `nd00333_AZMLND_C2-master/Exercise_starter_files` directory in the course Github repo or on the lab desktop.

Exercise: Swagger documentation

In this exercise, run both `swagger.sh` and `serve.py` to get Docker running locally serving Swagger so that you can interact with the deployed model Documentation. Use the `http://localhost/` and `http://localhost:8000/swagger.json`, to look at your swagger document and specifics of your model.

Tips: if you change the Swagger UI port to something other than 80, `http://localhost/` will not load the Swagger page. You need to include the new port to load the Swagger page. For example, use `http://localhost:9000` if the new port is set to 9000.

Please complete the following:



- Download the swagger.json file
- Run the `swagger.sh` and `serve.py` scripts
- Use localhost to interact with the swagger instance running with the documentation for the HTTP API of the model
- Use `http://localhost:8000/swagger.json` to display the contents of the API for the model
- HTTP API methods and responses for the model are shown in the Swagger UI

Solution: Swagger Documentation

1. Ensure that Docker is installed on your computer.
2. Azure provides a Swagger JSON file for deployed models. Head to the Endpoints section, and find your deployed model there.
3. Click on the name of the model, and details will open that contains a Swagger URI section. Download the file locally to your computer and put it in the same directory with `serve.py` and `swagger.sh`.
4. Run two scripts:

`serve.py` will start a Python server on port 8000. This script needs to be right next to the downloaded `swagger.json` file. NOTE: this will not work if `swagger.json` is not on the same directory.

`swagger.sh` which will download the latest Swagger container, and it will run it on port 80. If you don't have permissions for port 80 on your computer, update the script to a higher number (above 9000 is a good idea).

5. Open the browser and go to <http://localhost:8000> where `serve.py` should list the contents of the directory. `swagger.json` must show. If it doesn't, it needs to be downloaded from the deployed model endpoint.



Directory listing for /

- [serve.py](#)
- [swagger.json](#)
- [swagger.sh](#)

6. Go to <http://localhost/> which should have Swagger running from the container (as defined in swagger.sh). If you changed the port number, use that new port number to reach the local Swagger service (for example, <http://localhost:9000> if port 9000 is used).

localhost

https://petstore.swagger.io/v2/swagger.json

Explore

Swagger Petstore 0.5

[Base URL: petstore.swagger.io/v2]
https://petstore.swagger.io/v2/swagger.json

This is a sample server Petstore server. You can find out more about Swagger at <http://swagger.io> or on [irc.freenode.net](#). #swagger. For this sample, you can use the api key `special-key` to test the authorization filters.

Terms of service
Contact the developer
Apache 2.0
Find out more about Swagger

Schemes

HTTPS

Authorize

pet Everything about your Pets

Find out more

POST /pet/{petId}/uploadImage uploads an image

POST /pet Add a new pet to the store

PUT /pet Update an existing pet

7. On the top bar, where `petstore.swagger.io` shows, change it to <http://localhost:8000/swagger.json>, then hit the Explore button. It should now display the contents of the API for the model
8. Look around at the different HTTP requests that are supported for your model, including the example.

The screenshot shows the Swagger UI interface for an API named 'demo-model-deploy'. At the top, it displays the URL `http://localhost:8000/swagger.json`. Below the title, it says 'API specification for the Azure Machine Learning service demo-model-deploy'. A dropdown menu for 'Schemes' is set to 'HTTPS'. The 'default' endpoint is shown with two operations: a 'GET' method for '/' and a 'POST' method for '/score'. Under the 'Models' section, there is a 'ServiceInput' model. The UI has a clean, modern design with a light gray background and blue and green highlights for different sections.

7. Quiz: Swagger Documentation

QUESTION 1 OF 2

What is the main reason for using the `serve.py` script?

- Azure does not host the Swagger UI
- Python is a requirement of Swagger
- It enables CORS so that the Swagger can build the documentation locally



Try Again

Swagger is independent of Python and it is not a dependency or requirement.

TRY AGAIN

Try Again

It is true that Azure doesn't host the Swagger UI, but it is not the reason why using `serve.py` is crucial.

TRY AGAIN

QUESTION 2 OF 2

What are some reasons you may want to use the Swagger UI with the `swagger.json` file that Azure provides?

Shows what are the endpoints available

Provides log output of a Pipeline

Explains how the model is trained

Because it has demo inputs for endpoints

A crucial aspect of using Swagger is that it will contain sample input for endpoints

TRY AGAIN

 Try Again

Swagger will not show any log output related to a deployed endpoint

A crucial aspect of using Swagger is that it will contain sample input for endpoints

TRY AGAIN

 Try Again

Swagger will not show any log output related to a deployed endpoint

Swagger will not explain how the model is trained

A crucial aspect of using Swagger is that it will contain sample input for endpoints

TRY AGAIN

8. Consume Deployed Service



<https://photos.app.goo.gl/5Bs2Q25cJr3puJGG9>

Summary

You can consume a deployed service via an HTTP API. An HTTP API is a URL that is exposed over the network so that interaction with a trained model can happen via HTTP requests.

Users can initiate an input request, usually via an HTTP **POST** request. HTTP POST is a request method that is used to **submit data**. The HTTP **GET** is another commonly used request method. HTTP GET is used to **retrieve information** from a URL. The allowed requests methods and the different URLs exposed by Azure create a bi-directional flow of information.

The APIs exposed by Azure ML will use JSON (JavaScript Object Notation) to accept data and submit responses. It served as a bridge language among different environments.

New terms

- **JSON:** JavaScript Object Notation, also referred to as a "*bridge language*" used to make communication possible between two groups who do not share a native dialect
- **GET request method:** GET is a request method supported by HTTP. This method should only be used to retrieve data from a web server
- **POST request method:** POST is a request method supported by HTTP. This method requests that a web server accepts the data enclosed in the body of the request message

Further reading

The ["How to consume a web service"](#) Azure documentation has good examples of further interactions with an endpoint.

9. Exercise: Consume Deployed Service

Note: This lab takes about 30 min to load. You may see some scripts running behind the scene to deploy a model for you. **DO NOT** close the terminal (PowerShell) that runs the scripts. It will close automatically once the deployment is done. Once the deployment is done, you can see an endpoint that is **Not Null**.

<https://photos.app.goo.gl/dxGsLuoP9sfSHtxz6>

Note: the instructor used a different target column in the demo other than `cnt`. So the result displayed in the demo would be different from yours if you use `cnt` as the target column. Your display should be similar to `{"result": [2553]}`

Summary

To interact with the endpoint, the provided `endpoint.py` script has everything you need, except for the full URL to the endpoint and the key to authenticate. You can find this file either in the course Github repo `Exercise_starter_files` directory or on the lab VM desktop `nd00333_AZMLND_C2-master/Exercise_starter_files` directory. It is *crucial* to make the update to the correct URL and authentication key, otherwise, the script will not be able to produce log output.

The authentication is passed in the request as part of the headers, as shown in this snippet:

```
# Set the content type
headers = {"Content-Type": "application/json"}

# If authentication is enabled, set the authorization header
headers["Authorization"] = f"Bearer {key}"
```

Exercise: Consume deployed service

After deployment of a model, the next big item is consuming service to retrieve data. If there is anything incorrect in the deployed model, consuming the data from the service would allow identifying problems. It is also a great way to validate (test) outputs are working as expected.

In this exercise, make an authenticated HTTP request to a deployed model service in Azure Container Services to retrieve data.

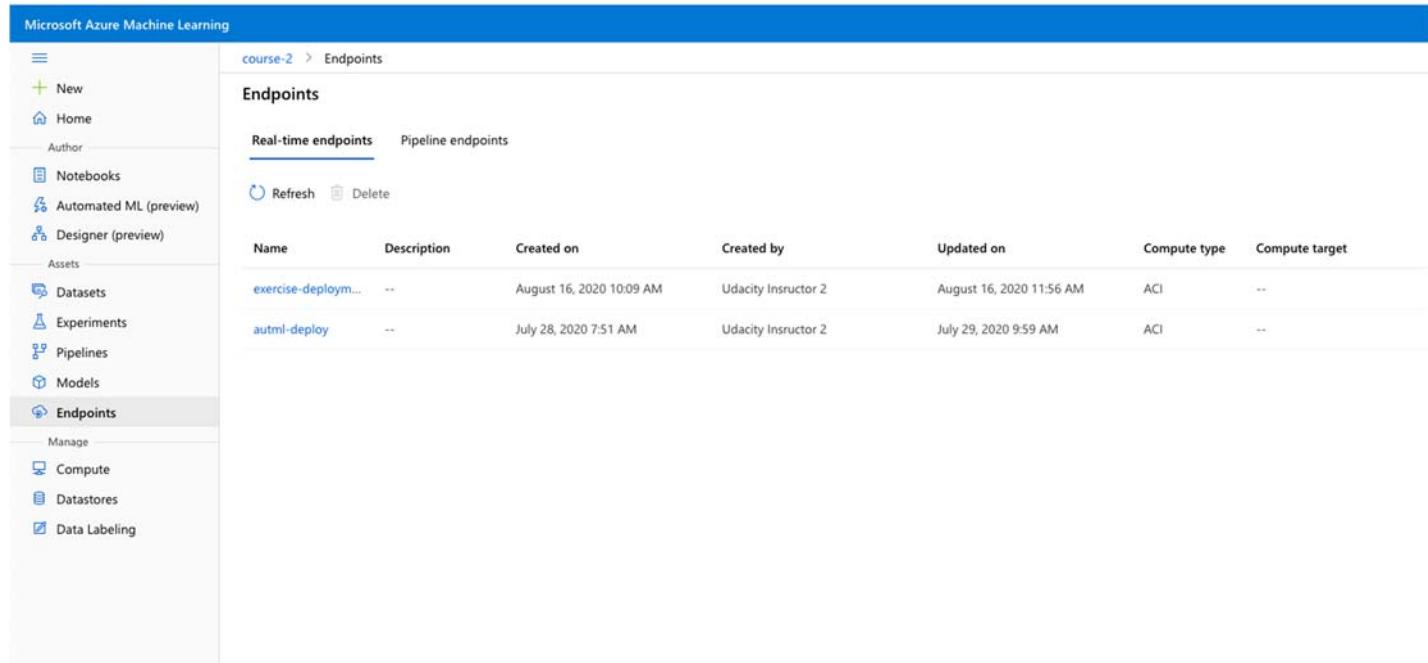
You are recommended to write the code by yourself and change the piece of data passed to the model. If you forget the code, you can take a look at the `endpoint.py` script. This script can be found in the course Github repo `Exercise_starter_files` directory and on the lab VM desktop `nd00333_AZMLND_C2-master/Exercise_starter_files` directory.

Please complete the following:

- Modify both the `scoring_uri` and the `key` to match the key for your service and the URI that was generated after deployment
- Modify the data being passed to the model
- Execute your version of `endpoint.py` file
- Log output is produced from the deployed model
- A `data.json` file is produced

10. Solution: Consume Deployed Service

1. In Azure ML Studio, head over to the "Endpoints" section and find a previously deployed model. The compute type should be ACI (Azure Container Instance).



The screenshot shows the Microsoft Azure Machine Learning Studio interface. The left sidebar has a navigation menu with options like New, Home, Author, Notebooks, Automated ML (preview), Designer (preview), Assets, Datasets, Experiments, Pipelines, Models, and Endpoints. The 'Endpoints' option is currently selected and highlighted in grey. The main content area is titled 'Endpoints' and shows two entries in a table:

| Name | Description | Created on | Created by | Updated on | Compute type | Compute target |
|---------------------|-------------|--------------------------|----------------------|--------------------------|--------------|----------------|
| exercise-deploym... | -- | August 16, 2020 10:09 AM | Udacity Instructor 2 | August 16, 2020 11:56 AM | ACI | -- |
| autml-deploy | -- | July 28, 2020 7:51 AM | Udacity Instructor 2 | July 29, 2020 9:59 AM | ACI | -- |

2. In the "*Consume*" tab, of the endpoint, a "*Basic consumption info*" will show the endpoint URL and the authentication types. Take note of the URL and the "*Primary Key*" authentication type.

The screenshot shows the Microsoft Azure Machine Learning interface. On the left, there's a sidebar with various options like New, Home, Notebooks, Automated ML (preview), Designer (preview), Datasets, Experiments, Pipelines, Models, Endpoints (which is currently selected), Compute, Datastores, and Data Labeling. The main area shows the path course-2 > Endpoints > exercise-deployment-1. Below this, the title "exercise-deployment-1" is displayed. There are two tabs: "Details" and "Consume", with "Consume" being active. Under "Basic consumption info", it shows a REST endpoint URL: <http://666fac5d-5231-4e8a-adc2-825e673032c3.eastus.azurecontainer.io ...>. It also lists authentication types: "Using key" (selected) and "Using token". Below that, it shows two keys: Primary key (q8szMDbCoNlxDZCpiGl8tnqaxtC1yDiy) with a "Regenerate" button, and Secondary key (Rz5MtuLhR0e4UTJw8pWVsK8VbG0SGfb) with a "Regenerate" button.

3. Using the provided `endpoint.py` replace the `scoring_uri` and `key` to match the REST endpoint and primary key respectively. The script issues a POST request to the deployed model and gets a JSON response that gets printed to the terminal.

Running it should produce similar results to this:

```
$ python endpoint.py  
{"result": [2553]}
```

```
$ python endpoint.py  
{"result": [2553]}
```

A data.json file will appear after you run endpoint.py

Note: the instructor used a different target column in the demo other than cnt. So the result displayed in the demo would be different from yours if you use cnt as the target column. Your display should be similar to {"result": [2553]}

11. Quiz: Consume Deployed Service

QUESTION 1 OF 3

What are the HTTP request methods you will end up using with a deployed model?

GET

DELETE

HEAD

POST

[TRY AGAIN](#)

`DELETE` is not exposed in the HTTP API endpoints

`POST` is one of the key request methods that you will use to interact with an endpoint, sending a JSON payload

[TRY AGAIN](#)

[Try Again](#)

`DELETE` is not exposed in the HTTP API endpoints

`HEAD` is not available as a supported request method for the HTTP endpoint

`POST` is one of the key request methods that you will use to interact with an endpoint, sending a JSON payload

[TRY AGAIN](#)

[Try Again](#)

`DELETE` is not exposed in the HTTP API endpoints

`HEAD` is not available as a supported request method for the HTTP endpoint

[TRY AGAIN](#)

POST is one of the key request methods that you will use to interact with an endpoint, sending a JSON payload

TRY AGAIN

QUESTION 2 OF 3

Match what you can do with each HTTP request method

DELETE

POST

ACTION

HTTP REQUEST METHOD

Submit data

POST

Retrieve data

GET

Find about the health status of the API

GET



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

Try Again

Find the health status of an API is considered as retrieving data from a web server.

TRY AGAIN

QUESTION 3 OF 3

What are some characteristics of **JSON** ?

It is used to store and transmit information

It is not used as HTTP API output files in Azure

It is used as HTTP API input files in Azure

It is a bridge language

JSON helps normalizing communications between services and programming languages.

TRY AGAIN

JSON helps normalizing communications between services and programming languages.

TRY AGAIN

JSON offers commonality between programming languages and APIs. It is used to connect endpoints and customers.

JSON helps normalizing communications between services and programming languages.

TRY AGAIN

12. Benchmark the Endpoint

<https://photos.app.goo.gl/nrrKdAfrs619u3fj9>

Summary

A benchmark is used to create a baseline or acceptable performance measure. Benchmarking HTTP APIs is used to find the average response time for a deployed model.

One of the most significant metrics is the *response time* since Azure will timeout if the response times are longer than sixty seconds.

Apache Benchmark is an easy and popular tool for benchmarking HTTP services. You will learn about it on the next page.

New terms

- **Response Time:** The time in seconds (or milliseconds) that service takes to produce a response
- **Timeout:** When a request is sent, this is an error when the server cannot produce a response in a given amount of time

Further reading

The documentation website for the [Apache Benchmark Tool](#) goes deep into all the options needed to benchmark almost any URL.

13. Exercise: Benchmark the Endpoint

Note: This lab takes about 30 min to load. You may see some scripts running behind the scene to deploy a model for you. **DO NOT** close the terminal (PowerShell) that runs the scripts. It will close automatically once the deployment is done. Once the deployment is done, you can see an endpoint that is **Not Null**.

Now, let's learn how to run the benchmark!

<https://photos.app.goo.gl/rtKZV4oMWrjvWX3b8>

Summary

Benchmarking services is an interesting topic. Everyone likes to have a highly performant endpoint, but the answer to what it takes to have a performant endpoint varies. It is useful to create a baseline for benchmarks so that comparing subsequent results is meaningful.

The `benchmark.sh` script doesn't have much code at all in it. It does include the `ab` command that runs against the selected endpoint using the `data.json` file created by the same `endpoint.py` file you used in the previous exercise. The `ab` command looks like this:

```
ab -n 10 -v 4 -p data.json -T 'application/json' -H 'Authorization: Bearer SECRET'
```

```
ab -n 10 -v 4 -p data.json -T 'application/json' -H 'Authorization: Bearer SECRET' http://URL.azurecontainer.io/score
```

After running the `benchmark.sh` or simply after running the above command, you will see the output of requests sent to and responses from the endpoint, and at the end, a summary with key information to determine response performance.

You can find the provided `endpoint.py` and `benchmark.sh` script in the `Exercise_starter_files` directory in the GitHub repo or on the lab desktop `nd00333_AZMLND_C2-master/Exercise_starter_files` directory.

Exercise: Benchmark the endpoint

In this exercise use the Apache Benchmark command-line tool (`ab`) to generate lots of HTTP POST requests to get performance metrics out of the Azure Container Instance.

Make sure you have the Apache Benchmark command-line tool installed and available in your path:

```
$ which ab  
/usr/bin/ab
```

```
$ ab --help  
Usage: ab [options] [http[s]://]hostname[:port]/path  
Options are:  
...  
...
```

You can use the provided `endpoint.py` and `benchmark.sh` script in the `Exercise_starter_files` directory to generate the benchmark. Make sure you modify it to match the URI and Keys.

Note: Be careful with your configuration if you use the lab provided to you. **Your experiment will timeout after 30 minutes!**

Please complete the following:



- ✓ Make sure you have the Apache Benchmark command-line tool installed and available in your path
- ✓ In the `bechmark.sh`, replace the key and URI again
- ✓ Update `endpoint.py` with the key and URL. Run `endpoint.py`. A `data.json` file should appear
- ✓ Run the `benchmark.sh` file or the `ab` command
- ✓ Apache Benchmark (ab) runs against the HTTP API using authentication keys to retrieve performance results

14. Solution: Benchmark the Endpoint

1. Make sure you have the Apache Benchmark command-line tool installed and available in your path:

```
$ which ab  
/usr/bin/ab
```

```
$ ab --help  
Usage: ab [options] [http[s]://]hostname[:port]/path  
Options are:  
...  
...
```

```
$ which ab  
/usr/bin/ab  
$ ab --help  
Usage: ab [options] [http[s]://]hostname[:port]/path  
Options are:  
...  
...
```

2. Run the `endpoint.py`. Just like before, it is important to use the right URI and Key to communicate with the deployed endpoint. A `data.json` should be present. This is required for the next step where the JSON file is used to HTTP POST to the endpoint.
3. In the provided started code, there is a `benchmark.sh` script with a call to `ab` similar to this:

```
ab -n 10 -v 4 -p data.json -T 'application/json' -H 'Authorization: Bearer Agb...
```

```
ab -n 10 -v 4 -p data.json -T 'application/json' -H 'Authorization: Bearer Agb...
```

Once you verify the file exists locally, run the `benchmark.sh` script. A highly verbose output should appear similar to this:

```
$ bash benchmark.sh
This is ApacheBench, Version 2.3 <$Revision: 1843412 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 8530a665-66f3-49c8-a953-b82a2d312917.eastus.azurecontainer.io (be patient)
---
POST /score HTTP/1.0
Content-length: 812
Content-type: application/json
Authorization: Bearer Agb3D23IyggpjVet4z8Y5oYGEvsnPzCC
Host: 8530a665-66f3-49c8-a953-b82a2d312917.eastus.azurecontainer.io
User-Agent: ApacheBench/2.3
Accept: */*

---
LOG: header received:
HTTP/1.0 200 OK
Content-Length: 33
Content-Type: application/json
Date: Thu, 30 Jul 2020 12:33:34 GMT
Server: nginx/1.10.3 (Ubuntu)
X-Ms-Request-Id: babfc511-a0f0-4ecb-a243-b3010a76b8b9
X-Ms-Run-Function-Failed: False

{"forecast": [985.0001475440272], "index": [{"date": 1356998400000, "_automl_dummy_0": 985.0001475440272}], "version": "1.0", "status": "OK", "error": null}
LOG: Response code = 200
LOG: header received:
HTTP/1.0 200 OK
Content-Length: 33
Content-Type: application/json
Date: Thu, 30 Jul 2020 12:33:34 GMT
Server: nginx/1.10.3 (Ubuntu)
X-Ms-Request-Id: b48dd8da-0b4e-44fd-ale5-04043bfa77f1
X-Ms-Run-Function-Failed: False

{"forecast": [985.0001475440272], "index": [{"date": 1356998400000, "_automl_dummy_0": 985.0001475440272}], "version": "1.0", "status": "OK", "error": null}
LOG: Response code = 200
..done
```

```

$ bash benchmark.sh
This is ApacheBench, Version 2.3 <$Revision: 1843412 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 8530a665-66f3-49c8-a953-b82a2d312917.eastus.azurecontainer.io
(be patient)...INFO: POST header ==
---
POST /score HTTP/1.0
Content-length: 812
Content-type: application/json
Authorization: Bearer Agb3D23IygqpjVet4z8Y5oYGEvsnPzCC
Host: 8530a665-66f3-49c8-a953-b82a2d312917.eastus.azurecontainer.io
User-Agent: ApacheBench/2.3
Accept: */*

---
LOG: header received:
HTTP/1.0 200 OK
Content-Length: 33
Content-Type: application/json
Date: Thu, 30 Jul 2020 12:33:34 GMT
Server: nginx/1.10.3 (Ubuntu)
X-Ms-Request-Id: babfc511-a0f0-4ecb-a243-b3010a76b8b9
X-Ms-Run-Function-Failed: False

"[{"forecast": [985.0001475440272], "index": [{"date": 1356998400000, "_automl_dummy_grain_col": "_automl_dummy_grain_col"}]}]"
LOG: Response code = 200
LOG: header received:
HTTP/1.0 200 OK
Content-Length: 33
Content-Type: application/json
Date: Thu, 30 Jul 2020 12:33:34 GMT
Server: nginx/1.10.3 (Ubuntu)
X-Ms-Request-Id: b48dd8da-0b4e-44fd-a1e5-04043bfa77f1
X-Ms-Run-Function-Failed: False

"[{"forecast": [985.0001475440272], "index": [{"date": 1356998400000, "_automl_dummy_grain_col": "_automl_dummy_grain_col"}]}]"
LOG: Response code = 200
..done

```

15. Quiz: Benchmark the Endpoint

QUESTION 1 OF 2

What is a good definition of benchmarking?

- Is a debugging technique
 - A group of HTTP request methods
 - An acceptable performance measure
 - A workflow for training models
-

Benchmarking set baselines or acceptable performance measures. Benchmarking HTTP APIs is used to find the average response time for a deployed model.

[TRY AGAIN](#)

Try Again

Although a benchmark can help debug, benchmarking is used to set benchmarks and acceptable performance measures.

TRY AGAIN

Benchmarking set baselines or acceptable performance measures. So it is not related to a workflow or training models.

TRY AGAIN

QUESTION 2 OF 2

In the following output from Apache Benchmark, which item would hint at a possible abnormality?

| | |
|-----------------------|-----------------------|
| Concurrency Level: | 40 |
| Time taken for tests: | 23.942 seconds |
| Complete requests: | 10000 |
| Failed requests: | 38 |
| Total transferred: | 1660000 bytes |
| HTML transferred: | 130000 bytes |
| Requests per second: | 417.68 [#/sec] (mean) |
| Time per request: | 95.767 [ms] (mean) |

Failed Requests because anything more than 0 indicates a service failure

Requests per second because 417 requests per second is not fast enough

Time per request because 95ms is too slow

[Try Again](#)

Unless a baseline indicated otherwise, 95ms is a very acceptable value

[TRY AGAIN](#)



Try Again

This is not meaningful enough to be an abnormality. Unless a baseline indicated many more were handled before.

TRY AGAIN

16. Curating Data Input

<https://photos.app.goo.gl/RRqFEQdeipYtrXwk6>

Summary

There are some key items to ensure when sending data to a deployed endpoint. You need to make sure that the keys and values are following the constraints. For example, if one field is repeated, this could potentially cause an error response, or if the format needs a date and time as a string, rather than an integer.

Remember, using values that the service doesn't expect would produce an error response.

QUIZ QUESTION

In the following JSON payload, which ONE field will FOR SURE cause an error?

```
{  
    "data": [  
        {  
            "instant": 1,  
            "date": "2013-01-01 00:00:00,000000",  
            "season": 1,  
            "year": 1963,  
            "month": 17,  
            "weekday": 6,  
            "weathersit": 2,  
        },  
    ]  
}
```

weekday

year

month

Try Again

weekday appears to be correct, with a possible value between 1 and 7 for the days of the week

TRY AGAIN

`year` appears to be correct, with a value that makes sense.

TRY AGAIN

17. Glossary

- **Benchmarking:** being able to create a baseline of acceptable performance so that it can be compared to day-to-day behavior
- **GET request method:** GET is a request method supported by HTTP. This method should only be used to retrieve data from a web server
- **JSON:** JavaScript Object Notation, also referred to as a "*bridge language*" used to make communication possible between two groups who do not share a native dialect
- **POST request method:** POST is a request method supported by HTTP. This method requests that a web server accepts the data enclosed in the body of the request message
- **RESTful:** A style for building HTTP endpoints that emphasizes separation of concerns
- **Response Time:** The time in seconds (or milliseconds) that service takes to produce a response
- **Swagger:** A tool that eases the documentation efforts of HTTP APIs
- **Timeout:** When a request is sent, this is an error when the server cannot produce a response in a given amount of time

Further Reading

These suggested articles and book chapters are related to concepts in this lesson:

- [Chapter 4 of Python For DevOps](#) covers benchmarking and other useful Linux utilities that can be useful.
- [The Swagger homepage](#) has in-depth examples and usage to dig further into other details.

- The "[How to consume a web service](#)" Azure documentation has good examples of further interactions with an endpoint.
- The documentation website for the [Apache Benchmark Tool](#) goes deep into all the options needed to benchmark almost any URL.

19. Lesson Review

<https://photos.app.goo.gl/LKf5bR7Y3wuby8zu8>



Summary

In this lesson, we went through a critical step after deploying a model: consuming information from it.

This lesson covered a lot of material related to a deployed model: starting with Swagger to better understand the API, to consume a deployed service, and all the way to create a baseline for finding potential problems.

Lesson 4: Pipeline Automation

1. Introduction

<https://photos.app.goo.gl/kYP6b4AcA1fJwAGK8>



Summary

In this lesson, we will go into more detail about automation. Automation is a core pillar of DevOps applicable to Machine Learning operations.

A good feature of Azure is Pipelines, and these are closely related to automation. Some key factors covered about pipelines are:

- Creating a pipeline (include batch inference pipeline)
- Publishing a pipeline
- Interacting with a pipeline via an HTTP API endpoint

New terms

- **Automation:** A core pillar of DevOps which is applicable to Machine Learning

Further reading

["What are Machine Learning Pipelines"](#) has good information about Pipelines and how they related to Machine Learning.

2. Pre-launch the Labs

In this lesson, you will have two labs: *Create a Pipeline* and *Publish and Consume a Pipeline*. The first lab takes about 5 min to load. However, the second lab "Publish and Consume a Pipeline" takes about 30 min to load because the lab creates a pipeline for you behind the scene, so you do not need to create a pipeline again. It normally takes 30 min to create a pipeline using the Azure Python SDK and the lab takes about the same time to launch.

✓ 4. Exercise: Create a Pipeline

✓ 5. Quiz: Create a Pipeline

✓ 6. Solution: Create a Pipeline

✓ 7. Publish a Pipeline

✓ 8. Consume Pipeline Endpoint (API)

✓ 9. Exercise: Publish and Consume a ...

As mentioned before, you can take a break while the lab is loading and come back in 30 minutes. Or you can open the exercise page in a new tab and launch it beforehand.

You may see some scripts running in the terminal (PowerShell). **DO NOT** close the terminal. It will close automatically once the deployment is done. And once it is done, you can see an endpoint that is **Not Null**.

3. Automation

<https://photos.app.goo.gl/ouEot6AqXyD2p4k19>

Summary

A great way to automate workflows is via Pipelines. This lesson covers publishing as well as interacting with them. Published pipelines allow external services to interact with them so that they can do work more efficiently.

4. Create a Pipeline

<https://photos.app.goo.gl/2pvfSNeavxwPXdUv5>

Summary

The most common SDK class is the Pipeline class. You will use this when creating a Pipeline. Pipelines can take configuration and different steps, like AutoML for example.

Different steps can have different arguments and parameters. Parameters are just like variables in a Python script.

There are areas you can play with when creating a pipeline and we covered two:

- Use pipeline parameters
- Recurring Scheduled Pipelines
- Batch Inference Pipelines

Create a Pipeline

This is the most common Python SDK class you will see when dealing with Pipelines. Aside from accepting a workspace and allowing multiple steps to be passed in, it uses a description that is useful to identify it later.

```
from azureml.pipeline.core import Pipeline

pipeline = Pipeline(
    description="pipeline_with_automlstep",
    workspace=ws,
    steps=[automl_step])
```

Using Pipeline Parameters

Pipeline parameters are also available as a class. You configure this class with the various different parameters needed so that they can later be used.

In this example, the `avg_rate_param` is used in the `arguments` attribute of the `PythonScriptStep`.

```
from azureml.pipeline.steps import PythonScriptStep
from azureml.pipeline.core import PipelineParameter

avg_rate_param = PipelineParameter(name="avg_rate", default_value=0.5)
train_step = PythonScriptStep(script_name="train.py",
                             arguments=["--input", avg_rate_param],
                             target=compute_target,
                             source_directory=project_folder)
```

Scheduling a recurring Pipeline

To schedule a Pipeline, you must use the `ScheduleRecurrence` class which has the information necessary to set the interval.

Once that has been created, it has to be passed into the `create()` method of the `Schedule` class as a `recurrence` value.

```
from azureml.pipeline.core.schedule import ScheduleRecurrence, Schedule

hourly = ScheduleRecurrence(frequency="Hourly", interval=4)
pipeline_schedule = Schedule.create(ws, name="RecurringSchedule",
                                     description="Trains model every few hours",
                                     pipeline_id=pipeline_id,
                                     experiment_name="Recurring_Pipeline_name",
                                     recurrence=hourly)
```

Batch Inference Pipeline

One of the core responsibilities of a batch inference pipeline is to run in parallel. For this to happen, you must use the `ParallelRunConfig` class which helps define the configuration needed to run in parallel.

Some important aspects of this are the script that will do the work (`entry_script`), how many failures it should tolerate (`error_threshold`), and the number of nodes/batches needed to run (`mini_batch_size`, 5 in this example).

```

from azureml.pipeline.steps import ParallelRunConfig

parallel_run_config = ParallelRunConfig(
    source_directory='scripts',
    entry_script="scoring.py",
    mini_batch_size="5",
    error_threshold=4,
    output_action="append_row",
    environment=batch_env,
    compute_target=aml_target,
    node_count=5)

parallelrun_step = ParallelRunStep(
    name="batch-score",
    parallel_run_config=parallel_run_config,
    inputs=[batch_data_set.as_named_input('batch_data')],
    output=output_dir,
    arguments=[],
    allow_reuse=True
)

# create the pipeline
pipeline = Pipeline(workspace=ws, steps=[parallelrun_step])

```

New terms

- **Batch inference:** The process of doing predictions using parallelism. In a pipeline, it will usually be on a recurring schedule
- **Recurring schedule:** A way to schedule pipelines to run at a given interval
- **Pipeline parameters:** Like variables in a Python script, these can be passed into a script argument

Further reading

The [Batch inference announcement](#) from Microsoft has very good insight as to what can these types of Pipelines do.

5. Exercise: Create a Pipeline

In the demo, the instructor will use a Jupyter notebook `aml-pipelines-with-automated-machine-learning-step.ipynb`. You can find this notebook in the `Exercise_starter_file` in the course Github or in the `nd00333_AZMLND_C2-master > Exercise_starter_file` folder on the lab desktop.

You will need to upload the above notebook to the **Notebooks** section of the Azure Machine Learning Workspace and then access it via the **Compute Instance** section.

⚠️ IMPORTANT: Make sure you are using this notebook via the **compute instance** (i.e., do *not* run it in the local Python environment).

Part 1 - Set up the workspace

<https://photos.app.goo.gl/Trk4hSubq17VcU6i8>

Part 2 - Create a pipeline with AutoML steps

<https://photos.app.goo.gl/3Mo6LZcY1oCgLFcp9>

Summary

Pipelines are very useful and are a foundation of automation and operations in general. Being able to create a Pipeline allows for easier interaction with model deployments.

This demo showed you how to use the Python SDK to create a pipeline with AutoML steps. Feel free to use the provided Jupyter notebook in the `Exercise_starter_file` directory to explore different features.

Exercise: Create a pipeline

For this exercise, you will create a pipeline using the python SDK.

You have already seen the `aml-pipelines-with-automated-machine-learning-step.ipynb` notebook (up to *Examine Results* section) provided in the exercise starter files. Now, review the

code and try to write your own code to create and run the pipelines. If you get stuck, check the notebook provided to you.

First, create a pipeline using the Python SDK. (This is the part that up until the *Examine Results* section in the provided notebook)

It is optional, you can copy and run cells in *Examine Results* section to test the pipeline and retrieve the best model. This step involves running an Automated ML experiment so it will take about 30 min to complete. Please keep track of the remaining time before you run these cells.

Tips: Make sure you update cells to match your dataset and other variables. These are noted in comments like this:

```
# Choose a name for the run history container in the workspace.  
# NOTE: update these to match your existing experiment name  
experiment_name = 'ml-experiment-1'  
project_folder = './pipeline-project'
```

Free free to modify the code to explore the different pipeline features and parameters. To speed up and shorten the total amount to train the model, you can change the "experiment_timeout_minutes" value from 20 to 10. These settings are in the Python Notebook, which are currently set to 20 minutes:

```
automl_settings = {  
    "experiment_timeout_minutes": 20,  
    "max_concurrent_iterations": 4,  
    "primary_metric" : 'normalized_root_mean_squared_error',  
    "n_cross_validations": 5  
}
```

Note: Be careful with your configuration if you use the lab provided to you. **Your experiment will timeout after a certain amount of time, please keep track of the time remaining for the lab!**

Create and run the pipelines using the Python SDK.



- Write your own code to create a pipeline
- Verify the pipeline has been created and shows in Azure ML studio, in the *Pipelines* section
- (Optional) Copy and run the cells in **Examine Results** in the provided notebook to test the pipeline

Part 2: Publish a pipeline

In this part, you need to publish a pipeline using the **both** Azure ML studio and the Python SDK. Please re-use the Pipeline created in the previous part.

You are recommended to write your own code to publish the pipeline. If you get stuck, review the first a few cells in the " Publish and run from REST endpoint" section in the provided notebook.

Note: Your experiment will timeout after a certain amount of time, please keep track of the time remaining for the lab!

Please complete the following tasks:

-
- In Pipelines section choose a pipeline available that has a status of Completed.
 - Publish an existing pipeline in ML studio, which creates an endpoint
 - Verify the pipeline has been published
 - Write your own code to publish the pipeline
 - Verify the pipeline has been published again

6. Quiz: Create a Pipeline

QUESTION 1 OF 3

What is the most common call you will see when creating pipelines with the Python SDK?



```
PythonScriptStep(script_name="train.py", inputs=[input_data], outputs=[output_data])
```



```
Pipeline(workspace=ws, steps=steps)
```



```
Experiment(workspace=ws, name="pipeline-experiment")
```

An `Experiment` does not produce a Pipeline

[TRY AGAIN](#)

A `PythonScriptStep` is one of the many steps that can be defined within a Pipeline

[TRY AGAIN](#)

-
- Pipeline parameters allows you to define values that get passed onto pipelines.

 - They are inputs that can alter how the pipeline behaves

 - They are much like variables in a Python program.

 - They are commonly used with Python scripts that can accept arguments
-

Pipeline parameters allow you to define values that can then be passed onto pipelines

Parameters can be used as inputs that alter a pipeline behavior

[TRY AGAIN](#)

Parameters are just like variables in Python

Some scripts use arguments and Pipeline parameters are a great fit for them

TRY AGAIN

Parameters can be used as inputs that alter a pipeline behavior

Parameters are just like variables in Python

Some scripts use arguments and Pipeline parameters are a great fit for them

TRY AGAIN

QUESTION 3 OF 3

What is the problem with the following Python SDK call?

```
pipeline_param = PipelineParameter(name="param1", default_value="10"
train_step = PythonScriptStep(script_name="train.py",
                                arguments=["--param1", "100"],
                                target=compute_target,
                                source_directory=project_folder)
```

There is nothing wrong with the SDK call

`pipeline_param` is not used as an argument

`PythonScriptStep` is the incorrect class for a Pipeline step

`PythonScriptStep` is a valid class for a Pipeline step

There is actually one thing that is not being assigned in the `PythonScriptStep`,
take another look!

[TRY AGAIN](#)

[TRY AGAIN](#)

7. Solution: Create a Pipeline

1. Open up the Jupyter Notebook and make sure you replace all of the URIs, Keys, and experiment names to match your own. Anywhere noted, ensure that the right components are replaced as shown in the next screenshot.

```
In [ ]: # Choose a name for the run history container in the workspace.  
# NOTE: update these to match your existing experiment name  
experiment_name = 'ml-experiment-1'  
project_folder = './pipeline-project'  
  
experiment = Experiment(ws, experiment_name)  
experiment
```

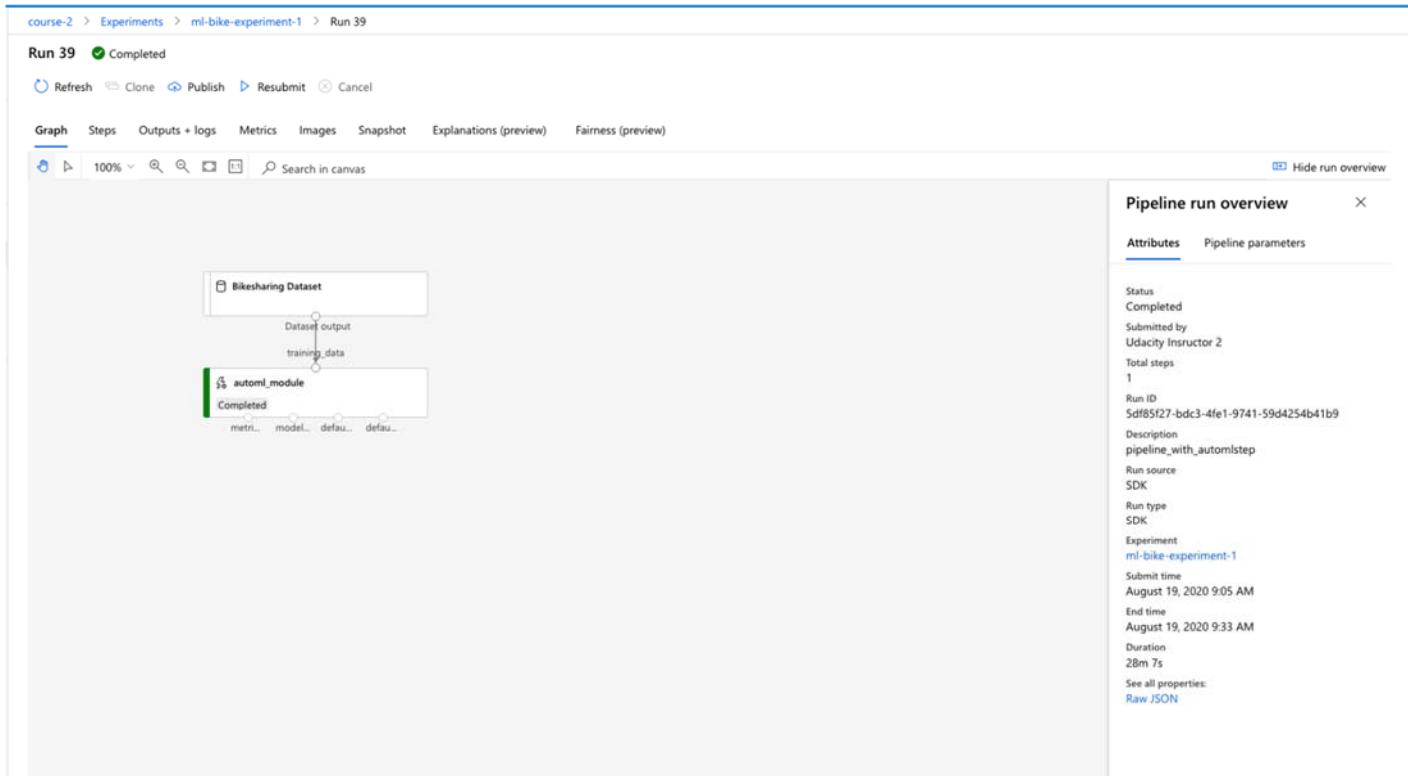
Create or Attach an AmlCompute cluster

You will need to create a [compute target](#) for your AutoML run. In this tutorial, you get the default [AmlCompute](#) cluster.

Udacity Note There is no need to create a new compute target, it can re-use the previous cluster.

```
In [ ]: from azureml.core.compute import AmlCompute  
from azureml.core.compute import ComputeTarget  
from azureml.core.compute_target import ComputeTargetException  
  
# NOTE: update the cluster name to match the existing cluster  
# Choose a name for your CPU cluster
```

2. Run the Jupyter Notebook all the way up until the `Examine Results` section. In the end, the pipeline should be available in Azure ML Studio in the `Pipelines` section.



3. Clicking on the Pipeline should take you to the experiment that demonstrates the graph using the Bikesharing Dataset and the Auto ML Model.
4. (Optional) You can keep running through the cells in the Examine Results section to retrieve metrics and the best model

8. Publish a Pipeline

<https://photos.app.goo.gl/cBrPfPYAR1SA1opb7>

Summary

Publish pipelines

Publishing a pipeline is the process of making a pipeline publicly available. You can publish pipelines in Azure Machine Learning Studio, but you can also do this with the Python SDK.

When a Pipeline is published, a public HTTP endpoint becomes available, allowing other services, including external ones, to interact with an Azure Pipeline.

Automation with pipelines

Pipelines are all about Automation. Automation connects different services and actions together to be part of a new workflow that wasn't possible before.

There are some good examples of how different services can communicate to the pipeline endpoint to enable automation.

- A hosted project using version control: when a new change gets merged, a trigger is created to send an HTTP request to the endpoint and train the model.
- A newer dataset gets uploaded to a storage system that triggers an HTTP request to the endpoint to re-train the model.
- Several teams that want to use AutoML with datasets that are hosted externally can configure the external cloud provider to trigger an HTTP request when a new dataset gets saved.
- A CI /CD platform like Jenkins, with a job that submits an HTTP request to Azure when it completes without error.

New terms

- **HTTP Trigger:** With configuration, a service can create an HTTP request based on certain conditions
- **Publishing a Pipeline:** Allowing external access to a Pipeline over an HTTP endpoint

Further reading

The Azure documentation on [how to create and run ML Pipelines](#) is a good introduction to Pipelines in general, but also on publishing them.

9. Consume Pipeline Endpoint (API)

<https://photos.app.goo.gl/jNxoSimGeA8zk1jB7>

Summary

Pipeline endpoints can be consumed via HTTP, but it is also possible to do so via the Python SDK. Since there are different ways to interact with published Pipelines, this makes the whole pipeline environment very flexible.

It is key to find and use the correct HTTP endpoint to interact with a published pipeline. Sending a request over HTTP to a pipeline endpoint will require authentication in the request headers. We will talk more about it later.

New terms

- **Pipeline endpoint:** The URL of the published Pipeline
- **HTTP Headers:** Part of the HTTP specification, where a request can attach extra information, like authentication

Further reading

The documentation on the [Pipeline Endpoint class](#) has some good information about methods that allow interacting with a pipeline.

10. Exercise: Publish and Consume a Pipeline

Exercise: Publish and Consume a Pipeline

Note: Since this course focuses on MLOps, we will pre-load a pipeline for you so you don't need to create it by yourself. You probably noticed that it takes you more than 30 minutes to create a pipeline, and similarly, this lab will take about 30 minutes to load because it loads a pipeline behind the scene. While the lab is loading, leave the page open and take a break. Come back in 30 minutes and the lab will be ready for you.

You may see some scripts running in the terminal (PowerShell). This is used for loading the pipeline. **DO NOT** close the terminal. It will close automatically once the process is done.

In the demo, the instructor will continue using `aml-pipelines-with-automated-machine-learning-step.ipynb`. You can find this notebook in the `Exercise_starter_file` in the course Github or in the `nd00333_AZMLND_C2-master > Exercise_starter_file` folder on the lab desktop.

Part 1: Publish a pipeline

<https://photos.app.goo.gl/7D3PsmhjBELGzTAp8>

Summary

Once a Pipeline is created and is considered as a good candidate to use it, it needs to be *published*. Publishing a Pipeline will allow interacting with it over HTTP. In this demo, you have learned how to publish a pipeline using both the Azure ML Studio and Python SDK. Feel free to play around with the Jupyter notebook provided to you in the course Github repo to explore more features of the pipeline.

There are some key code cells related to publishing that was **NOT** covered in the video. They are covered here:

A key part of publishing a pipeline is the `pipeline_run` object. To publish a pipeline, you need a `pipeline_run` object!

In the demo, we first created the pipeline and the `pipeline_run` object when we submit an experiment (in part 2 video):

```
pipeline run = experiment.submit(pipeline)
```

Then we published the pipeline using the `pipeline_run` object:

```
published_pipeline = pipeline_run.publish_pipeline(name="Bikesharing Train", de
```

```
published_pipeline = pipeline_run.publish_pipeline(name="Bikesharing Train",  
description="Training bikesharing pipeline", version="1.0")
```

The `name` attribute will help identify this Pipeline in Azure ML Studio later after it is published.

However, if in a situation where you don't run through the cells to create the pipeline, for example, the pipeline has already created, there will be no `pipeline_run` object. The above code will cause an error as a result of no `pipeline_run` object. What do you do to publish an existing pipeline?

To publish an *existing pipeline* without submitting an experiment to create one, you need to populate the experiment name from an existing experiment. In my case, I used the experiment `ml-bike-experiment-1`:

```
# NOTE: update these to match your existing experiment name and a previous experiment
experiment_name = 'ml-bike-experiment-1'
project_folder = './pipeline-bike-project'

experiment = Experiment(ws, experiment_name)
```

```
# NOTE: update these to match your existing experiment name and a previous
experiment_name = 'ml-bike-experiment-1'
project_folder = './pipeline-bike-project'

experiment = Experiment(ws, experiment_name)
```

Then you can create the `pipeline_run` object, but for this case, you must have a `run_id` of an existing experiment run. A `run_id` is available in the *Pipelines* section in Azure ML studio.

```
from azureml.pipeline.core import PipelineRun

run_id = "78e729c3-4746-fffff-aaaaa-abe970f4966f"
pipeline_run = PipelineRun(experiment, run_id)
```

After the `pipeline_run` object is created, you can use it to publish the pipeline:

```
published_pipeline = pipeline_run.publish_pipeline(name="Bikesharing Train", de
< >
```

```
published_pipeline = pipeline_run.publish_pipeline(name="Bikesharing Train",
description="Training bikesharing pipeline", version="1.0")
```

The screenshot shows the Microsoft Azure Machine Learning Pipelines interface. On the left, there's a sidebar with navigation links like 'New', 'Home', 'Notebooks', 'Automated ML (preview)', 'Designer (preview)', 'Datasets', 'Experiments', 'Pipelines' (which is selected), 'Models', 'Endpoints', 'Compute', 'Datastores', and 'Data Labeling'. The main area has a breadcrumb 'course-2 > Pipelines' and a title 'Pipelines'. It features three tabs: 'Pipeline runs' (selected), 'Pipeline endpoints', and 'Pipeline drafts'. Below the tabs is a button '+ New pipeline' and a 'Refresh' button. A table lists pipeline runs with columns: Run, Run ID, Experiment, and Status. The data is as follows:

| Run | Run ID | Experiment | Status |
|--------|--------------------------------------|-----------------------------|-----------|
| Run 84 | 4ac84556-7bdb-4122-b735-ef1c346d4333 | pipeline-bike-rest-endpoint | Completed |
| Run 82 | 78e729c3-4746-417f-ad9a-abe970f4966f | pipeline-bike-rest-endpoint | Completed |
| Run 76 | 4f96fd2e-3c2c-40a7-9267-281431ba8f1b | pipeline-bike-rest-endpoint | Completed |
| Run 72 | cfb09aa3-7a4d-47ae-8922-a363616961eb | pipeline-bike-rest-endpoint | Completed |
| Run 71 | a3be0c38-b129-42fc-8bcb-d328e1b8ebbe | pipeline-bike-rest-endpoint | Completed |
| Run 54 | fe217c7c-d7df-4000-92e4-19a5af2ea07a | pipeline-bike-rest-endpoint | Completed |
| Run 95 | 48e13764-c92c-4d06-a155-65d363332dd1 | ml-bike-experiment-1 | Completed |

Find the Run ID in the Pipelines section

Part 2: Consume a pipeline endpoint

<https://photos.app.goo.gl/tQYzrYuGv4uQ2dRG7>

Summary

Although a Pipeline Endpoint can be consumed via HTTP directly and by any tool that can send and consume HTTP requests, it is useful to know there are other options like the Azure SDK.

In this demo, you have learned how to send a post request to the pipeline using SDK. Feel free to modify the code provided to you in the course GitHub repo to explore more requests options to interact with pipelines.

There are some important aspects of consuming a pipeline with the SDK. One of them is authentication, which is sent over HTTP headers. The `requests` library sends an HTTP request and the Authentication has to be part of it. The authentication is retrieved with:

```
from azureml.core.authentication import InteractiveLoginAuthentication

interactive_auth = InteractiveLoginAuthentication()
auth_header = interactive_auth.get_authentication_header()
```

Then send a request with the `post()` helper:

```
import requests

rest_endpoint = published_pipeline.endpoint
response = requests.post(rest_endpoint,
                          headers=auth_header,
                          json={"ExperimentName": "pipeline-bike-rest-endpoint"})

```

```
import requests

rest_endpoint = published_pipeline.endpoint
response = requests.post(rest_endpoint,
                          headers=auth_header,
                          json={"ExperimentName": "pipeline-bike-rest-
endpoint"})

```

Exercise: Publish and consume a pipeline

In this exercise, you will publish a pipeline that already created for you. This pipeline only has one step: an Automated ML experiment. After the pipeline is published, you will consume the pipeline endpoint to run an experiment.

You can re-use the provided Jupyter Notebook, `aml-exercise-pipelines-with-automated-machine-learning-step.ipynb`. You do **not** need to run all of the cells in the notebook since a pipeline should have been created for you. The pipeline can be found in the *Pipelines* in ML Studio. You only need to modify and review the codes in *Publish and run from the REST endpoint*.

Part 1

In this part, you need to publish a pipeline using the **both** Azure ML studio and the Python SDK. Please use the Pipeline that has created for you.

You are recommended to write your own code when using the Python SDK to publish the pipeline. If you get stuck, review the first a few cells in the " Publish and run from REST endpoint" section in the provided notebook.

Note: Your experiment will timeout after a certain amount of time, please keep track of the time remaining for the lab!

Please complete the following tasks:

- In Pipelines section choose a pipeline available that has a status of Completed.
- Publish an existing pipeline in ML studio, which creates an endpoint
- Verify the pipeline has been published
- Write your own code to publish the pipeline
- Verify the pipeline has been published again

Part 2: Consume a pipeline endpoint

In the previous part, you published a pipeline. In this part, you will consume a pipeline using the Azure Python SDK. Please write your own code to send a post request to the endpoint. The post request will trigger the pipeline and schedule an Automated ML experiment.

At end of the notebook, it shows you how to monitor the status of the pipeline run using RunDetails Widget. It is an optional step, but you can watch the full output using RunDetails. It may take 20-30 min to complete the run.

Note: Your experiment will timeout after a certain amount of time, please keep track of the time remaining for the lab!

Please complete the following tasks:

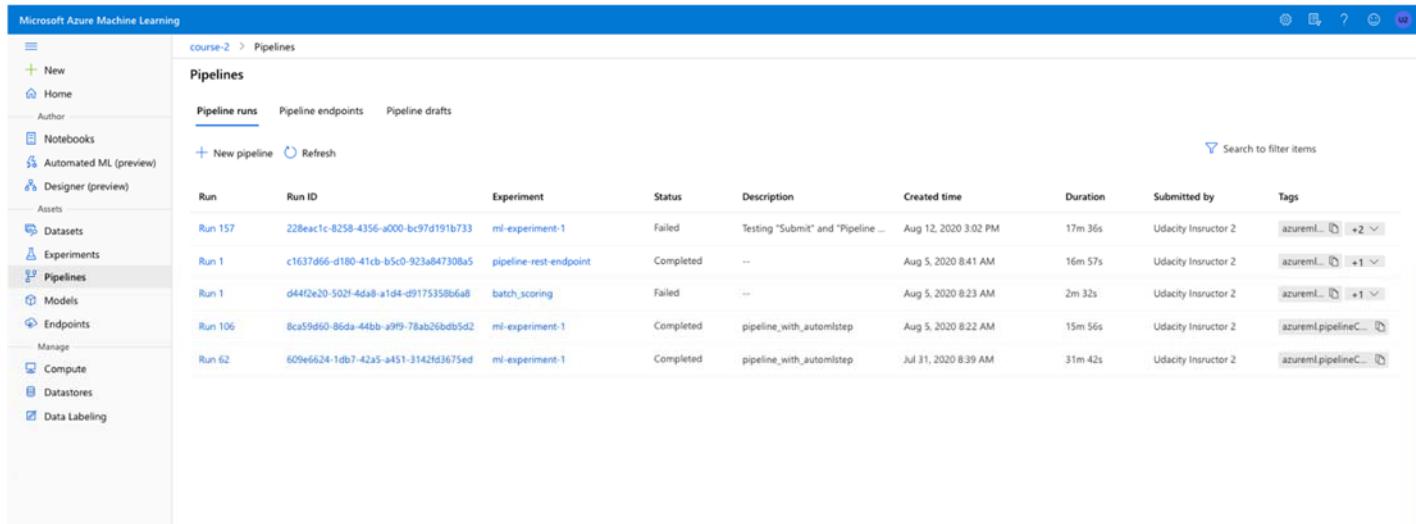
- In the "Pipeline endpoints", the "Published Pipeline overview" showing a REST endpoint and a status of ACTIVE
- Write codes that send a post request to the pipeline endpoint to trigger the pipeline
- In "Experiment" in ML Studio, it shows the run is scheduled or running
- (Optional) In Jupyter notebook the "Use RunDetails Widget" shows the step runs

11. Solution: Publish and Consume a Pipeline

Part 1: Publish a Pipeline

ML Studio

1. In Azure ML Studio, under the *Pipelines* section, you will get to a list of all the pipelines available. Click on a *Run ID* that has a status of *Completed*.



The screenshot shows the Microsoft Azure Machine Learning studio interface. The left sidebar is a navigation menu with sections like New, Home, Author, Notebooks, Automated ML (preview), Designer (preview), Assets, Datasets, Experiments, Pipelines (which is currently selected and highlighted in grey), Models, Endpoints, Manage, Compute, Datastores, and Data Labeling. The main content area is titled 'course-2 > Pipelines' and contains a table titled 'Pipelines'. The table has columns for Run, Run ID, Experiment, Status, Description, Created time, Duration, Submitted by, and Tags. There are five rows in the table:

| Run | Run ID | Experiment | Status | Description | Created time | Duration | Submitted by | Tags |
|---------|--------------------------------------|------------------------|-----------|------------------------------------|----------------------|----------|----------------------|----------------------|
| Run 157 | 228eac1c-8258-4356-a000-bc97d191b733 | ml-experiment-1 | Failed | Testing "Submit" and "Pipeline ... | Aug 12, 2020 3:02 PM | 17m 36s | Udacity Instructor 2 | azureml... +2 |
| Run 1 | c1637d66-d180-41cb-b5c0-923a847308a5 | pipeline-rest-endpoint | Completed | -- | Aug 5, 2020 8:41 AM | 16m 57s | Udacity Instructor 2 | azureml... +1 |
| Run 1 | d44f2e20-502f-4da0-a1d4-d9175350b6a8 | batch_scoring | Failed | -- | Aug 5, 2020 8:23 AM | 2m 32s | Udacity Instructor 2 | azureml... +1 |
| Run 106 | bca59d60-86da-44bb-a9f9-78ab26bdb5d2 | ml-experiment-1 | Completed | pipeline_with_automlstep | Aug 5, 2020 8:22 AM | 15m 56s | Udacity Instructor 2 | azureml.pipelineC... |
| Run 62 | 609e6624-1db7-42a5-a451-3142fd3675ed | ml-experiment-1 | Completed | pipeline_with_automlstep | Jul 31, 2020 8:39 AM | 31m 42s | Udacity Instructor 2 | azureml.pipelineC... |

- Click on the *Publish* button so that the overlay menu shows up, and fill it with something descriptive. You can either re-use an endpoint, or create a new one.

The screenshot shows the Microsoft Azure Machine Learning interface. On the left, there's a sidebar with navigation links like 'New', 'Home', 'Author', 'Notebooks', 'Automated ML (preview)', 'Designer (preview)', 'Assets', 'Datasets', 'Experiments' (which is selected), 'Pipelines', 'Models', 'Endpoints', 'Manage', 'Compute', 'Datastores', and 'Data Labeling'. The main area shows a pipeline graph titled 'Run 39' which is 'Completed'. The graph consists of a 'Bikesharing Dataset' node connected to a 'automl_module' node. The 'automl_module' node has several outputs: 'training_data', 'metr...', 'model...', 'defau...', and 'defau...'. Below the graph, there are tabs for 'Graph', 'Steps', 'Outputs + logs', 'Metrics', 'Images', 'Snapshot', 'Explanations (preview)', and 'Fairness (preview)'. A search bar at the top says 'Search in canvas'. To the right of the graph, an overlay window titled 'Set up published pipeline' is open. It contains a section for 'PipelineEndpoint' where 'Create new' is selected and the name 'bikesharing-pipeline' is entered. Below this, a note says 'This completes the publishing pipeline exercise'. Under 'Published pipeline', there are two checked checkboxes: 'Set as default pipeline for this endpoint.' and 'Continue on failure step'. At the bottom of the overlay are 'Publish' and 'Cancel' buttons.

Python SDK

- Create experiment and `pipeline_run`

The experiment and `run_id` of that experiment are crucial. Update the experiment name, `project_folder`:

```
# NOTE: update these to match your existing experiment name and a previous experiment
experiment_name = 'ml-bike-experiment-1'
project_folder = './pipeline-bike-project'

experiment = Experiment(ws, experiment_name)
```

```
# NOTE: update these to match your existing experiment name and a previous experiment
experiment_name = 'ml-bike-experiment-1'
project_folder = './pipeline-bike-project'

experiment = Experiment(ws, experiment_name)
```

And for the run_id:

```
from azureml.pipeline.core import PipelineRun

run_id = "78e729c3-4746-fffff-aaaaa-abe970f4966f"
pipeline_run = PipelineRun(experiment, run_id)
```

The screenshot shows the Microsoft Azure Machine Learning studio interface. The left sidebar has a navigation menu with items like 'New', 'Home', 'Notebooks', 'Automated ML (preview)', 'Designer (preview)', 'Datasets', 'Experiments', 'Pipelines' (which is selected and highlighted in grey), 'Models', 'Endpoints', 'Compute', 'Datastores', and 'Data Labeling'. The main content area is titled 'course-2 > Pipelines' and shows a table of pipeline runs. The table has columns for 'Run', 'Run ID', 'Experiment', 'Status', and 'Description'. There are 9 rows listed, each corresponding to a completed pipeline run with a unique ID and experiment name.

| Run | Run ID | Experiment | Status | Description |
|--------|--------------------------------------|-----------------------------|-----------|--------------------------|
| Run 84 | 4ac84556-7bdb-4122-b735-ef1c346d4333 | pipeline-bike-rest-endpoint | Completed | -- |
| Run 82 | 78e729c3-4746-417f-ad9a-abe970f4966f | pipeline-bike-rest-endpoint | Completed | -- |
| Run 76 | 4f96fd2e-3c2c-40a7-9267-281431ba8f1b | pipeline-bike-rest-endpoint | Completed | -- |
| Run 72 | cfb09aa3-7a4d-47ae-8922-a363616961eb | pipeline-bike-rest-endpoint | Completed | -- |
| Run 71 | a3be0c38-b129-42fc-8bcb-d328e1b8ebbe | pipeline-bike-rest-endpoint | Completed | -- |
| Run 54 | fe217c7c-d7df-4000-92e4-19a5af2ea07a | pipeline-bike-rest-endpoint | Completed | -- |
| Run 95 | 48e13764-c92c-4d06-a155-65d363332dd1 | ml-bike-experiment-1 | Completed | pipeline_with_automlstep |

Find the Run ID in the Pipelines section in Azure ML Studio

- Once the `pipeline_run` object is created, you can publish the pipeline

```
published_pipeline = pipeline_run.publish_pipeline(  
    name="Bikesharing Train", description="Training bikesharing pipeline", versi  
< >
```

```
published_pipeline = pipeline_run.publish_pipeline(  
    name="Bikesharing Train", description="Training bikesharing pipeline",  
version="1.0")
```

Part 2: Consume a pipeline endpoint

- Once the pipeline is published, you can authenticate:

```
from azureml.core.authentication import InteractiveLoginAuthentication  
  
interactive_auth = InteractiveLoginAuthentication()  
auth_header = interactive_auth.get_authentication_header()
```

```
from azureml.core.authentication import InteractiveLoginAuthentication  
  
interactive_auth = InteractiveLoginAuthentication()  
auth_header = interactive_auth.get_authentication_header()
```

- The published pipeline will be used to retrieve the endpoint. This endpoint is the URI that the SDK will use to communicate with it over HTTP. The relevant code that uses the endpoint to make the HTTP request looks like this:

```

import requests

rest_endpoint = published_pipeline.endpoint
response = requests.post(rest_endpoint,
                         headers=auth_header,
                         json={"ExperimentName": "pipeline-bike-rest-endpoint"
                               })

```

```

import requests

rest_endpoint = published_pipeline.endpoint
response = requests.post(rest_endpoint,
                         headers=auth_header,
                         json={"ExperimentName": "pipeline-bike-rest-
endpoint"})

```

- Once the Jupyter Notebook completes all of its steps, the Pipeline will be triggered and available in Azure ML Studio.

| Run status | experiment_status_description | experiment_status |
|------------|-------------------------------|-------------------|
| 1 Running | 0 Completed | No data |
| 0 Failed | 0 Other | No data |

12. Quiz: Publish a Pipeline

QUESTION 1 OF 3

Match which action can be handled by Azure ML studio, the Python SDK, or both

Python SDK

FEATURE

ENVIRONMENT

Create a Pipeline

Both SDK and Studio

Publish a Pipeline

Both SDK and Studio

Review a Pipeline

Azure ML Studio





Try Again

Creating a pipeline is possible with Azure ML Studio and the Python SDK

You can review a Pipeline before publishing in Azure ML studio

TRY AGAIN

QUESTION 2 OF 3

What are the true facts after a pipeline is published?

- Any service outside of Azure can interact with the Pipeline
 -
 - An HTTP endpoint is produced, that accepts requests from anywhere
 -
 - A model is trained using AutoML and other steps
 -
 - An experiment is modified to reflect it is published via HTTP
-

An HTTP endpoint is the result of publishing a Pipeline

TRY AGAIN

Once it is published, any service can interact with the Pipeline via HTTP

An HTTP endpoint is the result of publishing a Pipeline

An experiment doesn't change when a Pipeline gets published

TRY AGAIN

Once it is published, any service can interact with the Pipeline via HTTP

TRY AGAIN

```
1 publish(name="My_New_Pipeline",
2         description="My New Pipeline Description",
3         version="1.0")
```

A

```
1 publish_pipeline(name="My_New_Pipeline",
2                   description="My New Pipeline Description",
3                   version="1.0")
```

B

```
1 pipeline_run.publish_pipeline(name="My_New_Pipeline",
2                               description="My New Pipeline Description",
3                               version="1.0")
```

C

QUESTION 3 OF 3

Which of the code examples is correct to publish a pipeline using the Python SDK?

A

B

C

`publish` is not a separate function available in the SDK to call it in this way

TRY AGAIN

`publish_pipeline` is not a separate function available in the SDK to call it in this way

TRY AGAIN



Microsoft Azure
Noel Ching

13. Quiz: Automation With Pipeline Endpoints

QUESTION 1 OF 3

What are some true facts about Automation in general?

-
- There is always a way to automate
 - Automation is always a straightforward process
 - Automating as much as you can in day-to-day operations is the goal
 - Automation enables whole environments to be more productive, resilient, and scale infinitely
-

Automation is sometimes elusive, but there is always a way to automate

When things are clear, automation is straightforward, but this isn't always the case. Careful planning and execution will make it easier though

Automating as much as possible is a great goal to have

By automating processes, whole environments can be more productive and resilient

TRY AGAIN

TRY AGAIN

Automation is sometimes elusive, but there is always a way to automate

By automating processes, whole environments can be more productive and
resilient

TRY AGAIN

Automation is sometimes elusive, but there is always a way to automate

Automating as much as possible is a great goal to have

TRY AGAIN

Automating as much as possible is a great goal to have

By automating processes, whole environments can be more productive and
resilient

TRY AGAIN



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

QUESTION 2 OF 3

Which of these are good examples of pipeline automation?

- A project using Git, upon receiving new changes it triggers an HTTP request to the endpoint and train the model.
- A website redesign gets published
- A new file in a storage system in a separate cloud provider is created, which sends an HTTP request to the endpoint
- A new model is trained

A separate project that can trigger an HTTP request is a good example of two separate workflows working together with automation

A website redesign is not a good example of automation, as it is not a collection of workflows working together

A separate storage system that upon a specific action like a new file, that then creates a request to an endpoint is a good example of automation

Although a new model can be trained by a Pipeline, automation is about connecting services together eliminating manual processes

[TRY AGAIN](#)



Microsoft Azure
Noel Ching

Machine Learning Engineer for Microsoft Azure

A separate project that can trigger an HTTP request is a good example of two separate workflows working together with automation

A separate storage system that upon an specific action like a new file, that then creates a request to an endpoint is a good example of automation

Although a new model can be trained by a Pipeline, automation is about connecting services together eliminating manual processes

TRY AGAIN

TRY AGAIN

A separate project that can trigger an HTTP request is a good example of two separate workflows working together with automation

A website redesign is not a good example of automation, as it is not a collection of workflows working together

TRY AGAIN

TRY AGAIN

A separate storage system that upon an specific action like a new file, that then creates a request to an endpoint is a good example of automation

TRY AGAIN

TRY AGAIN

QUESTION 3 OF 3

What is one of the most important aspects of having a publicly accessible HTTP endpoint to trigger a Pipeline?

- Allows Pipelines to run more efficiently
- Allows external services to take advantage of Azure with ease
- Allows Models to take advantage of Azure ML Studio

TRY AGAIN

The efficiency of pipelines is not going to get affected by exposing an HTTP endpoint to them.

TRY AGAIN

A publicly accessible HTTP endpoint is probably not going to allow models to
related differently to Azure ML Studio

TRY AGAIN

14. Quiz: Consume Pipeline Endpoint (API)

QUESTION 1 OF 3

Why is using the Python SDK a powerful option to consume a Pipeline endpoint?

Because it is faster than HTTP

Because by using code, this can be reviewed and enhanced by others

Because it is more efficient



Microsoft Azure
Noel Ching

The Python SDK will not be faster than HTTP, because in the end, it will send an HTTP request still.

[TRY AGAIN](#)

There is not much more efficiency in using the Python SDK, although there are ways to make it easier to consume

[TRY AGAIN](#)

QUESTION 2 OF 3

Which is the correct Python SDK call to find the Pipeline HTTP endpoint?



endpoint.published_pipeline



published_pipeline.endpoint



published_pipline.http_endpoint

A published_pipeline is not a method that is available

[TRY AGAIN](#)

[Try Again](#)

There is no http_endpoint method for published pipeline objects

[TRY AGAIN](#)



Microsoft Azure
Noel Ching

```
1 from azureml.core.authentication import InteractiveLoginAuthentication  
2  
3 interactive_auth = InteractiveLoginAuthentication()  
4 auth_header = interactive_auth.authentication()
```

A

```
1 from azureml.core.authentication import InteractiveLoginAuthentication  
2  
3 interactive_auth = InteractiveLoginAuthentication()  
4 auth_header = interactive_auth.get_authentication()
```

B

```
1 from azureml.core.authentication import InteractiveLoginAuthentication  
2  
3 interactive_auth = InteractiveLoginAuthentication()  
4 auth_header = interactive_auth.get_authentication_header()
```

C

QUESTION 3 OF 3

To authenticate to a Pipeline endpoint, authentication headers need to be created. From the images above, what is the correct way to get the headers necessary to send an HTTP request?

A

B

C

`get_authentication()` is not a valid method call in the SDK

TRY AGAIN

Iry Again

`authentication()` is not a valid method call in the SDK

TRY AGAIN

15. Pipelines - Different Tasks

<https://photos.app.goo.gl/hYUiuhM7sP1tecDn7>

Summary

Pipelines can perform several other tasks aside from training a model. Some of these tasks, or steps are:

- Data Preparation
- Validation
- Deployment
- Combined tasks

16. Glossary

- **Automation:** A core pillar of DevOps which is applicable to Machine Learning
- **Batch inference:** The process of doing predictions using parallelism. In a pipeline, it will usually be on a recurring schedule
- **HTTP trigger:** With configuration, a service can create an HTTP request based on certain conditions
- **Pipeline parameters:** Like variables in a Python script, these can be passed into a script argument
- **Publishing a Pipeline:** Allowing external access to a Pipeline over an HTTP endpoint
- **Recurring schedule:** A way to schedule pipelines to run at a given interval

17. Further reading

Here are a few links to further explain important aspects of Pipelines:

- "[What are Machine Learning Pipelines](#)" has good information about Pipelines and how they relate to Machine Learning.
- The Azure documentation on [how to create and run ML Pipelines](#) is a good introduction to Pipelines in general, but also on publishing them.
- The documentation on the [Pipeline Endpoint class](#) has some good information about methods that allow interacting with a pipeline.

18. Lesson Review

<https://photos.app.goo.gl/em42pc3g4vawj7Ft8>

Summary

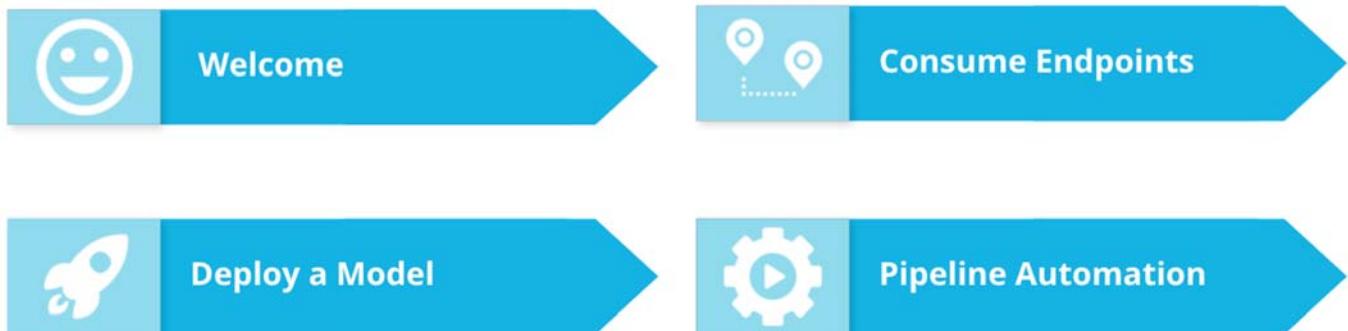
This lesson covered a few key aspects of Machine Learning Pipelines:

- Creating a Machine Learning Pipeline using Python SDK
- Publish a pipeline using both ML Studio and Python SDK
- Consume a published pipeline using Python SDK

All of these concepts are a solid foundation to start exploring pipelines more and taking advantage of their flexibility.

19. Course Review

<https://photos.app.goo.gl/o2MdDBMMuzD6rgGD8>



Summary

In this course, we started by defining MLOps and who are the people you will interact with.

Then, I covered how to configure clusters and compute targets to get models into production.

Next, we learned about how to interact with a deployed model and get useful information from it.

And finally, we went over pipelines: from creating them to publishing and consuming them, emphasizing automation all throughout.

Lesson 5 : Project: Operationalizing Machine Learning

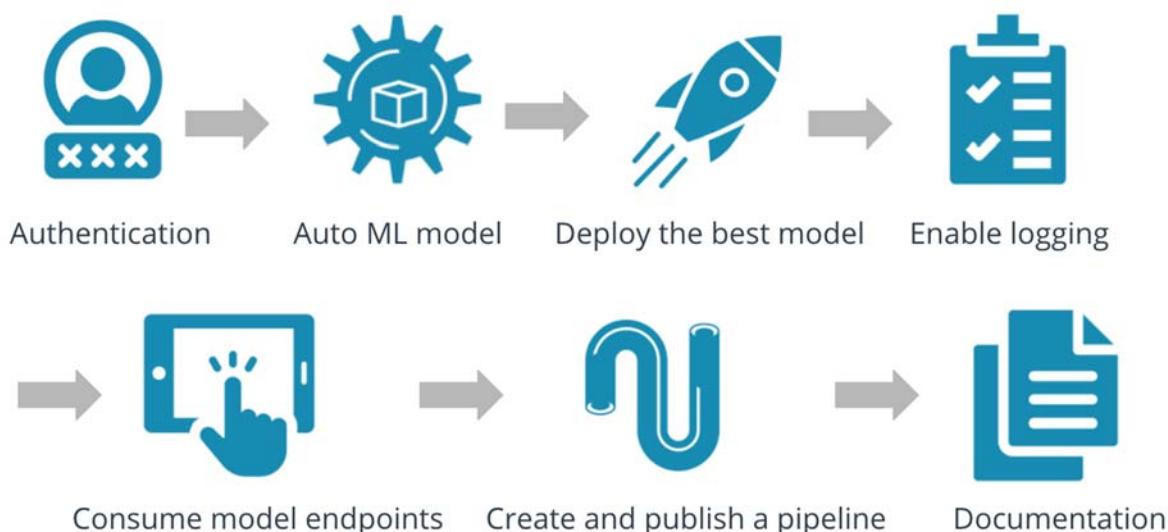
1. Project Overview

In this project, you will continue to work with the [Bank Marketing dataset](#). You will use Azure to configure a cloud-based machine learning production model, deploy it, and consume it. You will also create, publish, and consume a pipeline. In the end, you will demonstrate all of your work by creating a README file and a screencast video.

Project main steps

In this project, you will follow the below steps:

1. Authentication
2. Automated ML Experiment
3. Deploy the best model
4. Enable logging
5. Swagger Documentation
6. Consume model endpoints
7. Create and publish a pipeline
8. Documentation



Project submissions

At the end of the project, you will submit a README file that describes the main components of the project and a screencast that shows the entire process of the working ML application.

2. Project Setup

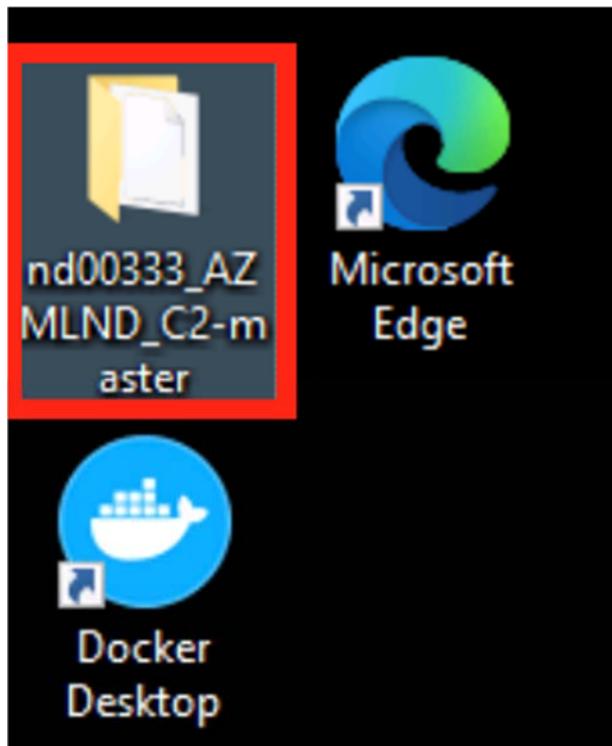
<https://photos.app.goo.gl/tody3mN5W29XSxgm6>

Working environment

You can use either the Project [lab](#) provided by Udacity or your own Azure account to complete the project.

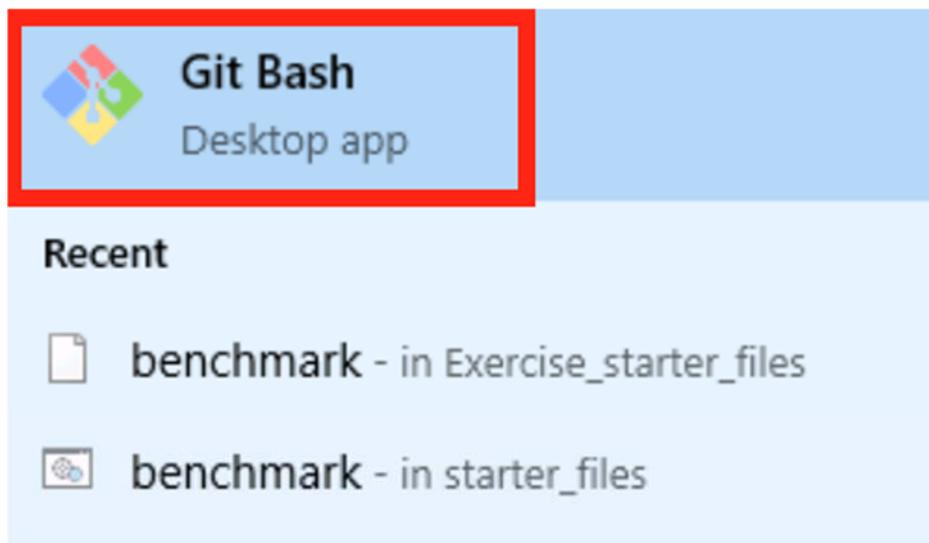
Some notes for you if you are using the lab provided

1. All the starter files are ready for you on the lab desktop. You can find them in
nd00333_AZMLND_C2-master > Starter files folder



The folder contains all the files for this course

2. You can use the "Git bash" as the terminal. Search "git bash" and click the Git bash application



Command

git bash



3. You will have a maximum of **10 chances** to load and work in the provided lab. You will not be able to access the lab after 10 trials. So please only load the lab when you plan to do work in it.
4. Be careful with your configuration. **Your lab will timeout after 4 hours! All your work saved locally in the lab will be lost once it times out.** Always **SAVE** your work in **Github!!!**
5. A window will pop up 15 minutes before the lab times out and it will give you a chance to extend the lab for another **15 min**. You will have **4 chances** to extend the lab, which means you can maximumly extend the lab for one hour.

Getting started

Please complete the following:

- Clone the repo to your Github account so that you can work on this copy
You can find all of the starter code for this project at this [Github repo](#).
- If you use your own account, make sure Azure ML Studio has been upgraded from classic to enterprise to enable AutoML

https://github.com/udacity/nd00333_AZMLND_C2/tree/master/starter_files

3. Step 1: Authentication

NOTE: If you are using your own Azure account, please complete this step. If you are using the lab Udacity provided to you, you can skip this step since you are not authorized to create a security principal. You will NOT be graded on this item and skipping this step will NOT affect other steps in the project.

In this step, you will need to install the Azure Machine Learning Extension which allows you to interact with Azure Machine Learning Studio, part of the `az` command. After having the Azure machine Learning Extension, you will create a Service Principal account and associate it with your specific workspace. You will also be instructed to take screenshots to document your work.

Install and login

Please complete the following if you use your own account



- Install `az` and enable it in the terminal
- `az login` has succeeded
- Install the Azure Machine Learning extension
- Create the Service Principal (SP) and allow the SP access to your specific workspace
- Take a screenshot showing that a "**Service Principal**" has been created
- `az ml workspace share` completed without errors
- Take a screenshot showing that the `az ml workspace share` command has been run successfully, with no errors or tracebacks.

4. Step 2: Automated ML Experiment

At this point, security is enabled and authentication is completed. In this step, you will create an experiment using **Automated ML**, configure a compute cluster, and use that cluster to run the experiment.

Note: Be careful with your configuration if you use the lab provided to you. **Your experiment will timeout after a certain amount of time! All your work saved locally in the lab will be lost once it times out.** Always **SAVE** your work in **Github!!!**

You will use the same Bankmarketing dataset with course 1.

Copy the link to a new browser window to download the data:

https://automl-samplenotebookdata.blob.core.windows.net/automl-sample-notebook-data/bankmarketing_train.csv

https://drive.google.com/file/d/1HCSBooAEelrgxCffpyf6WN-zYI_e18FY/view?usp=sharing

Upload the bankmarketing_train.csv to Azure Machine Learning Studio so that it can be used when training the model.

Create and run Auto ML Experiment

Please complete the following:

Note: the goal of these steps is not to create a great model but to provide you a model so you can deploy it and consume it. So don't be too harsh on yourself in creating a highly accurate model.

-
- Create a New Automated ML run
 - Select and upload the Bankmarketing dataset
 - Create a new Automated ML experiment
 - Configure a new compute cluster
Select Standard_DS12_v2 for the Virtual Machine Size and select 1 as the number of minimum nodes.
 - Run the experiment using *Classification*, ensure *Explain best model* is checked.
On Exit criterion, reduce the default (3 hours) to 1 and reduce the Concurrency from default to 5 (this number should always be less than the number of the compute cluster)

Note: This process takes about 15 minutes and it runs about 5 minutes per iteration

Take screenshots to show your work

Please complete the following to show your work:

-
- Take a screenshot of “Registered Datasets” in ML Studio showing that Bankmarketing dataset available
 - Take a screenshot showing that the experiment is shown as completed
 - Take a screenshot of the best model after the experiment completes
-

5. Step 3: Deploy the Best Model

After the experiment run completes, a summary of all the models and their metrics are shown, including explanations. The *Best Model* will be shown in the *Details* tab. In the *Models* tab, it will come up first (at the top). Make sure you select the best model for deployment.

Deploying the Best Model will allow to interact with the HTTP API service and interact with the model by sending data over POST requests.

Deploy the model

Please complete the following items to deploy the **best** model.

-  Select the **best** model for deployment
-  Deploy the model and enable "Authentication"
-  Deploy the model using Azure Container Instance (ACI)

6. Step 4: Enable Application Insights

Now that the *Best Model* is deployed, enable Application Insights and retrieve logs. Although this is configurable at deploy time with a check-box, it is useful to be able to run code that will enable it for you.

Enable Application Insights

Please complete the following using Azure Python SDK:

- Ensure `az` is installed, as well as the Python SDK for Azure
- Create a new virtual environment with Python3
- Write and run code to enable Application Insights
- Use the provided code `logs.py` to view the logs
- Take a screenshot showing that "Application Insights" is enabled in the Details tab of the endpoint.
- Take a screenshot showing logs by running the provided `logs.py` script

7. Step 5: Swagger Documentation

In this step, you will consume the deployed model using Swagger.

Azure provides a [Swagger JSON file](#) for deployed models. Head to the *Endpoints* section, and find your deployed model there, it should be the first one on the list.

A few things you need to pay attention to:

1. `swagger.sh` will download the latest Swagger container, and it will run it on port 80. If you don't have permissions for port 80 on your computer, update the script to a higher number (above 9000 is a good idea).
2. `serve.py` will start a Python server on port 8000. This script needs to be right next to the downloaded `swagger.json` file. **NOTE:** this will not work if `swagger.json` is not on the same directory.

Consume model using Swagger

Please complete the following:

- Download the swagger.json file
- Run the `swagger.sh` and `serve.py`
- Interact with the swagger instance running with the documentation for the HTTP API of the model.
- Display the contents of the API for the model
- Take a screenshot showing that swagger runs on localhost showing the HTTP API methods and responses for the model

8. Step 6: Consume Model Endpoints

Once the model is deployed, use the `endpoint.py` script provided to interact with the trained model. In this step, you need to run the script, modifying both the `scoring_uri` and the `key` to match the key for your service and the URI that was generated after deployment.

Hint: This URI can be found in the *Details* tab, above the *Swagger URI*.

Model endpoints

Please complete the following:



- ✓ Modifying both the `scoring_uri` and the `key` to match the key for your service and the URI that was generated after deployment
- ✓ Execute the `endpoint.py` file, the output should be similar to the following:

```
{"result": ["yes", "no"]}
```
- ✓ Take a screenshot showing that the `endpoint.py` script runs against the API producing JSON output from the model.

The following is an optional step to benchmark the endpoint using Apache bench. You will not be graded on it but I encourage you to try it out.

Benchmark

It is an optional step. If you want to load-test your model, please complete the following



- ✓ Make sure you have the Apache Benchmark command-line tool installed and available in your path
- ✓ In the `endpoint.py`, replace the key and URI again
- ✓ Run `endpoint.py`. A `data.json` file should appear
- ✓ Run the `benchmark.sh` file. The output should look similar to the text below
- ✓ Take a screenshot showing that Apache Benchmark (ab) runs against the HTTP API using authentication keys to retrieve performance results



Microsoft Azure
Noel Ching

```
This is ApacheBench, Version 2.3 <$Revision: 1843412 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 8530a665-66f3-49c8-a953-b82a2d312917.eastus.azurecontainer.io (be
---
POST /score HTTP/1.0
Content-length: 812
Content-type: application/json
Authorization: Bearer Agb3D23IygXXXXXXXXXXXXXXXXXXXXXX
Host: 8530a665-66f3-49c8-a953-b82a2d312917.eastus.azurecontainer.io
User-Agent: ApacheBench/2.3
Accept: */*

---
LOG: header received:
HTTP/1.0 200 OK
Content-Length: 33
Content-Type: application/json
Date: Thu, 30 Jul 2020 12:33:34 GMT
Server: nginx/1.10.3 (Ubuntu)
X-Ms-Request-Id: babfc511-a0f0-4ecb-a243-b3010a76b8b9
X-Ms-Run-Function-Failed: False

"{"result": ["yes", "no"]}"
LOG: Response code = 200
LOG: header received:
HTTP/1.0 200 OK
Content-Length: 33
Content-Type: application/json
Date: Thu, 30 Jul 2020 12:33:34 GMT
Server: nginx/1.10.3 (Ubuntu)
X-Ms-Request-Id: b48dd8da-0b4e-44fd-a1e5-04043bfa77f1
X-Ms-Run-Function-Failed: False
```

This is ApacheBench, Version 2.3 <\$Revision: 1843412 \$>

```
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/
```

```
Benchmarking 8530a665-66f3-49c8-a953-b82a2d312917.eastus.azurecontainer.io
(be patient)...INFO: POST header ==
---
POST /score HTTP/1.0
Content-length: 812
Content-type: application/json
Authorization: Bearer Agb3D23IygXXXXXXXXXXXXXXXXXXXXXX
Host: 8530a665-66f3-49c8-a953-b82a2d312917.eastus.azurecontainer.io
User-Agent: ApacheBench/2.3
Accept: */*
```



```
---
LOG: header received:
HTTP/1.0 200 OK
Content-Length: 33
Content-Type: application/json
Date: Thu, 30 Jul 2020 12:33:34 GMT
Server: nginx/1.10.3 (Ubuntu)
X-Ms-Request-Id: babfc511-a0f0-4ecb-a243-b3010a76b8b9
X-Ms-Run-Function-Failed: False

"{"result": ["yes", "no"]}"
LOG: Response code = 200
LOG: header received:
HTTP/1.0 200 OK
Content-Length: 33
Content-Type: application/json
Date: Thu, 30 Jul 2020 12:33:34 GMT
Server: nginx/1.10.3 (Ubuntu)
X-Ms-Request-Id: b48dd8da-0b4e-44fd-a1e5-04043bfa77f1
X-Ms-Run-Function-Failed: False
```

9. Step 7: Create, Publish and Consume a Pipeline

For this part of the project, you will use the Jupyter Notebook provided in the starter files. You must make sure to update the notebook to have the same keys, URI, dataset, cluster, and model names already created. The notebook has several notes that look similar to this:

```
# NOTE: update these to match your existing experiment name
experiment_name = 'ml-experiment-1'
```

This is an indication to make updates or change the code on your own.



Complete the Jupyter Notebook

Run through all the cells in the provided Notebook which uses the Python SDK.

- ✓ Upload the Jupyter Notebook

```
aml-pipelines-with-automated-machine-learning-  
step.ipynb
```

to the Azure ML studio

- ✓ Update all the variables that are noted to match your environment



- ✓ Make sure a `config.json` has been downloaded and is available in the current working directory

- ✓ Run through the cells

- ✓ Verify the pipeline has been created and shows in Azure ML studio, in the *Pipelines* section

- ✓ Verify that the pipeline has been scheduled to run or is running



Microsoft Azure
Noel Ching

Screenshots

Please take the following screenshots to show your work:

- The pipeline section of Azure ML studio, showing that the pipeline has been created
- The pipelines section in Azure ML Studio, showing the Pipeline Endpoint
- The Bankmarketing dataset with the AutoML module
- The “Published Pipeline overview”, showing a REST endpoint and a status of ACTIVE
- In Jupyter Notebook, showing that the “Use RunDetails Widget” shows the step runs
- In ML studio showing the scheduled run

10. Step 8: Documentation

Screencast

In this project, you need to record a screencast that shows the entire process of the working ML application. The screencast should meet the following criteria:



- Screencast is 1-5 minutes in length
 - Audio is clear and understandable
 - Video is 1080P or higher with 16:9 aspect ratio
 - Text is readable
-

Screencast content requirement

A screencast is a useful way to share your project with others. In this project, you need to record a screencast that shows the entire process of the working ML application.

The screencast should include a demonstration of the following area:



- Working deployed ML model endpoint
- Deployed Pipeline
- Available AutoML Model
- Successful API requests to the endpoint with a JSON payload

README

An important part of your project submissions is a README file that describes the project and documents the main steps. Please use the README.md template provided to you as a start. The README should include the following areas:



- An overview of the project
- An Architectural Diagram
- A short description of how to improve the project in the future
- All the screenshots required in the project main steps with short descriptions
- A link to the screencast video on YouTube (or a similar alternative streaming service)

11. Project Lab

The screenshot shows the Microsoft Azure portal interface. The URL in the address bar is <https://portal.azure.com/#@udacitylabs.onmicrosoft.com/resource/subscriptions/81cefad3-d2c9-4f77-a466-99a7f541c7bb/resourceGroups/ml-quickstarts-143133/providers/Microsoft.MachineLearningServices/workspaces/quick-starts-ws-143133>. The page title is "quick-starts-ws-143133".

Essentials section details:

| Setting | Value |
|----------------------|---|
| Resource group | aml-quickstarts-143133 |
| Location | South Central US |
| Subscription | Udacity CloudLabs Sub - 12 |
| Subscription ID | 81cefad3-d2c9-4f77-a466-99a7f541c7bb |
| Studio web URL | https://ml.azure.com/?tid=660b3398-b80e-49d2-bc5b-ac1dc93b5254&wsid=/subscriptions/81cefad3-d2c9-4f77-a466-99a7f541c7bb/resourceGroups/ml-quickstarts-143133/providers/Microsoft.MachineLearningServices/workspaces/quick-starts-ws-143133 |
| Storage | mlstrg143133 |
| Registry | — |
| Key Vault | mikeyvault143133 |
| Application Insights | mlappinsight143133 |

Manage your machine learning lifecycle
Use the Azure Machine Learning studio to build, train, evaluate, and deploy machine learning models. Learn more [»](#)

[Launch studio](#)

Getting Started with Lab

- Once the environment is provisioned, a virtual machine (JumpVM) and lab guide will get loaded in your browser. Use this virtual machine throughout the workshop to perform the lab.
- To get the lab environment details, you can select **Lab Environment** tab.

The screenshot shows a blue header bar with the text "Udacity Nanodegree demo" and "0 hour(s), 56 minute(s) remaining". There is a "HIDE" button with a hide icon. Below the header are three tabs: "Lab Guide" (disabled), "Lab Environment" (selected and highlighted in black border), and "Help". At the bottom are two buttons: "Environment Details" (selected and highlighted in blue border) and "Lab Resources".

Azure Credentials

Here are your credentials to login to

<https://portal.azure.com> and access the On Demand
Lab

Username

odl_user_136385@udacitylabs.o



Password

usbs37PTM*tS



Environment Details

Resource Group : aml-vm-136385

Windows

labvmt7we3jifmlix6.southcentral



VMDNSN

ame

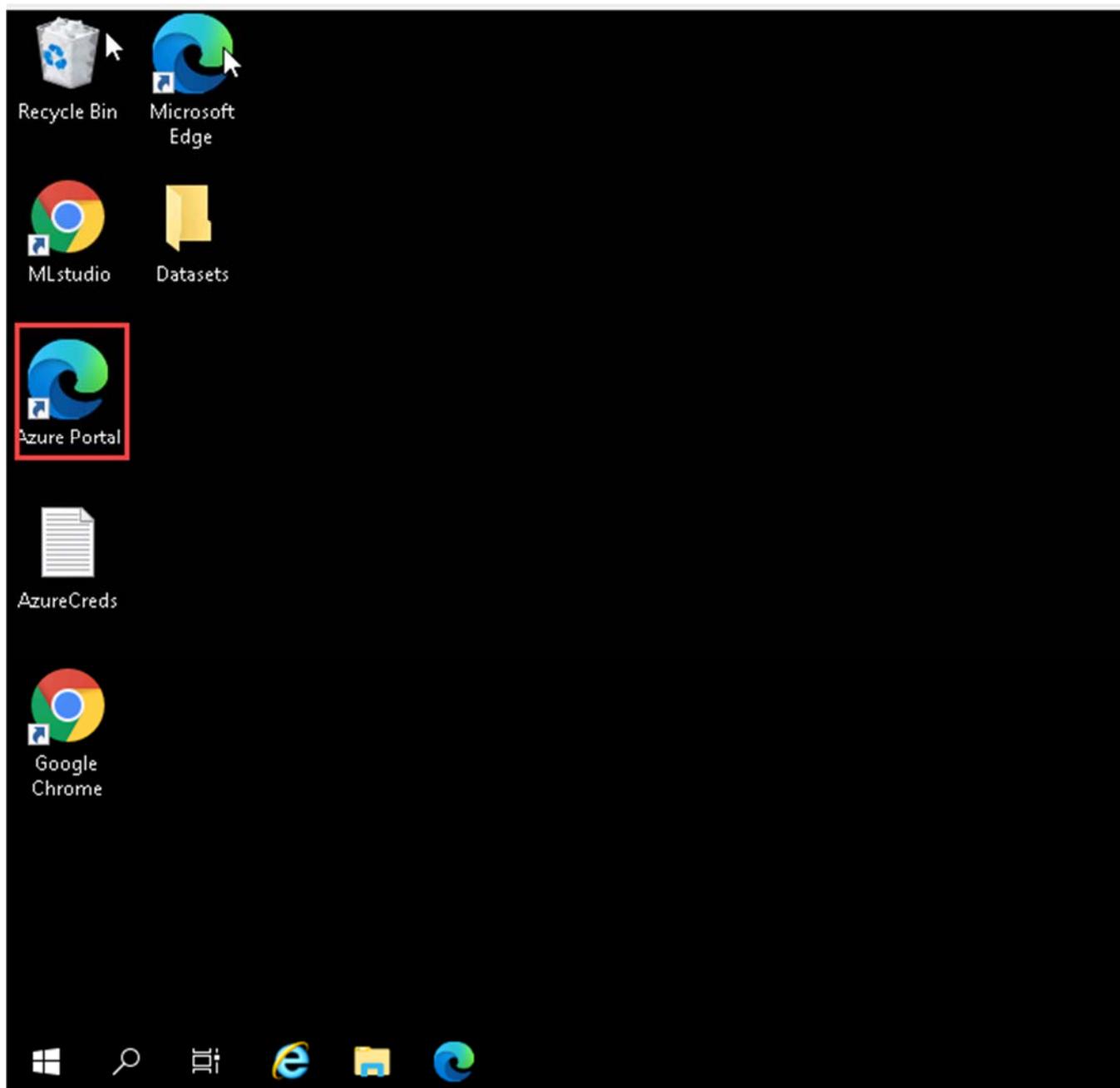
VMAdmin

demouser



[Login to Azure Portal](#)

1. In the JumpVM, click on Azure portal shortcut of Microsoft Edge browser which is created on desktop.



2. When you click on Azure portal, edge browser welcome screen will come up, select **Get started**.

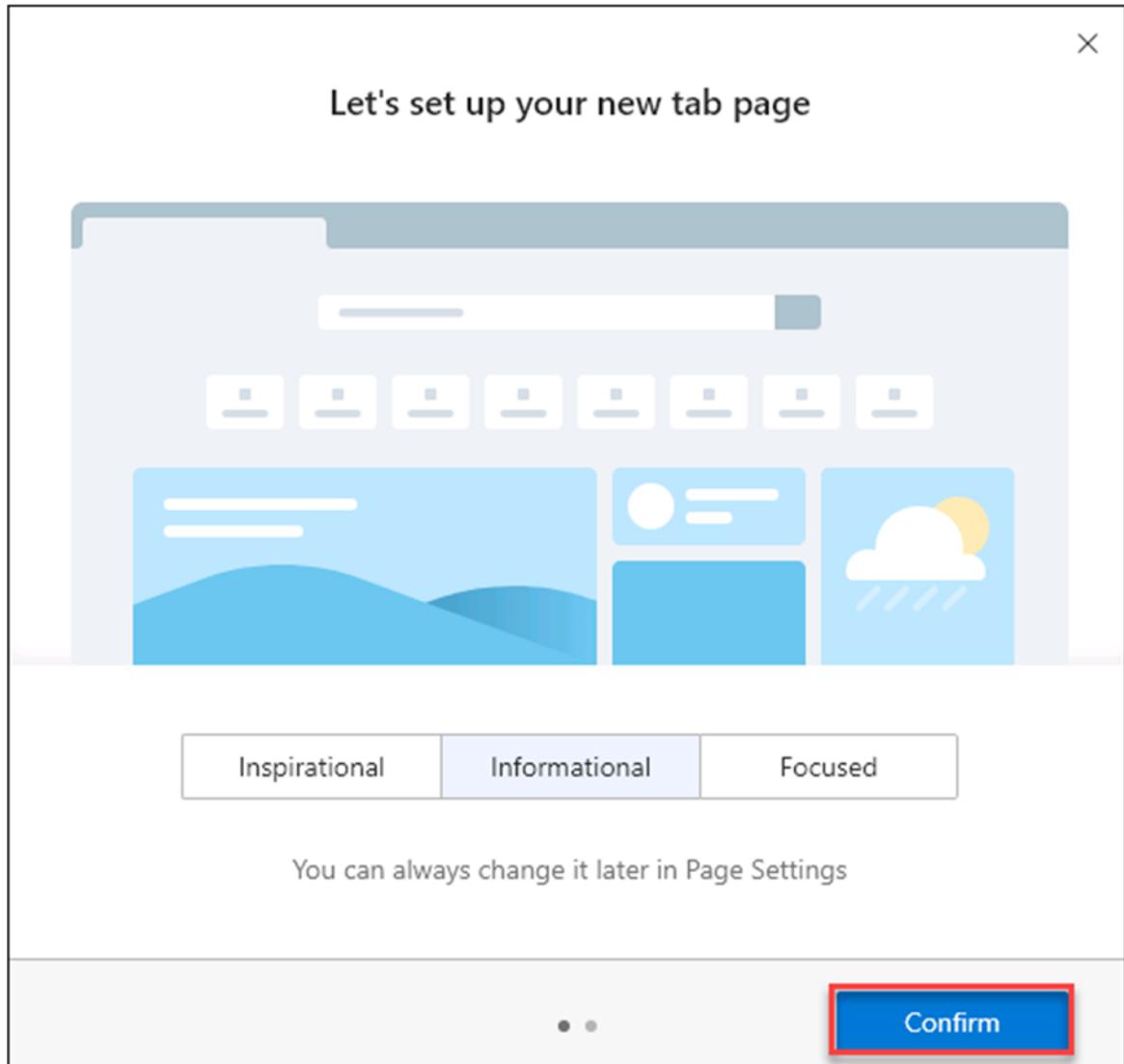


Welcome to the new Microsoft Edge

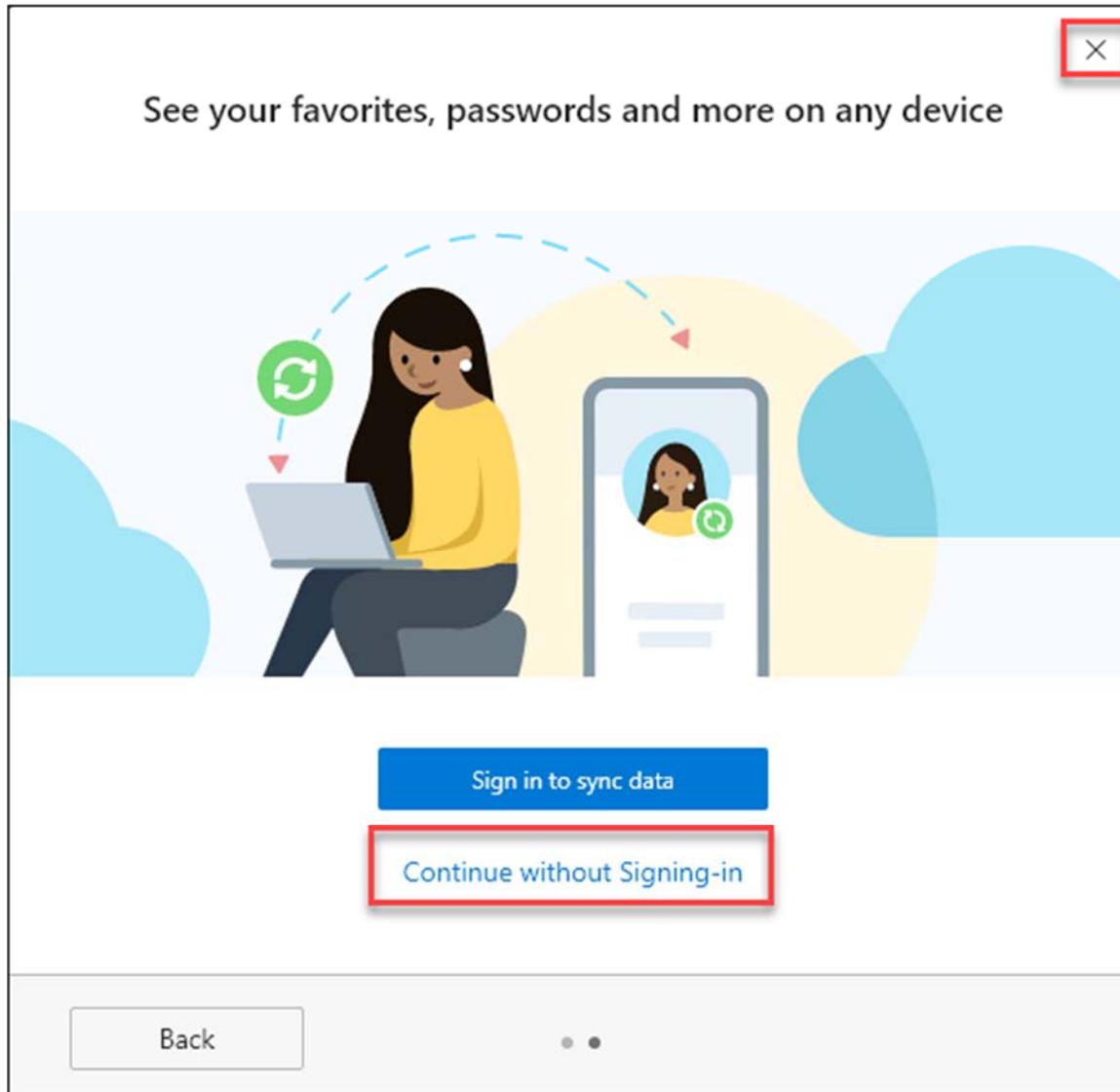
A fast, safe, and productive web browser that works for
you

[Get started](#)

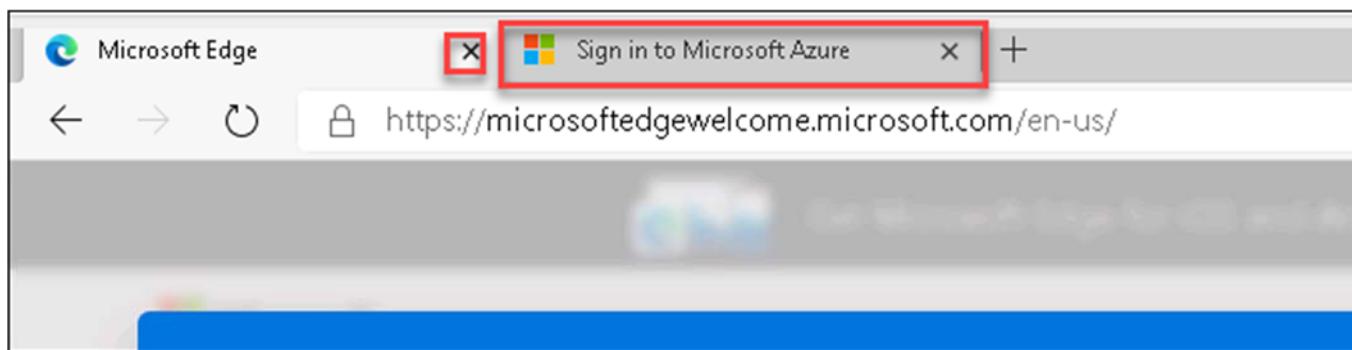
3. On next window, click on **Confirm**.



4. Now, you can close the popup which is coming up.

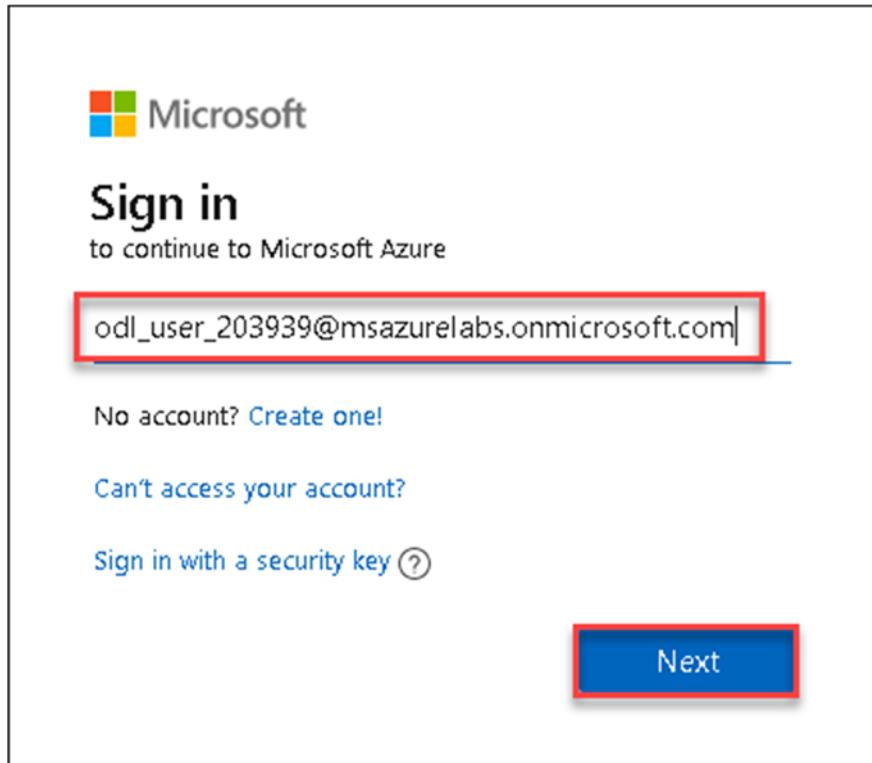


- Now, you will see two tabs in edge browser, close first tab named with **Microsoft Edge**.

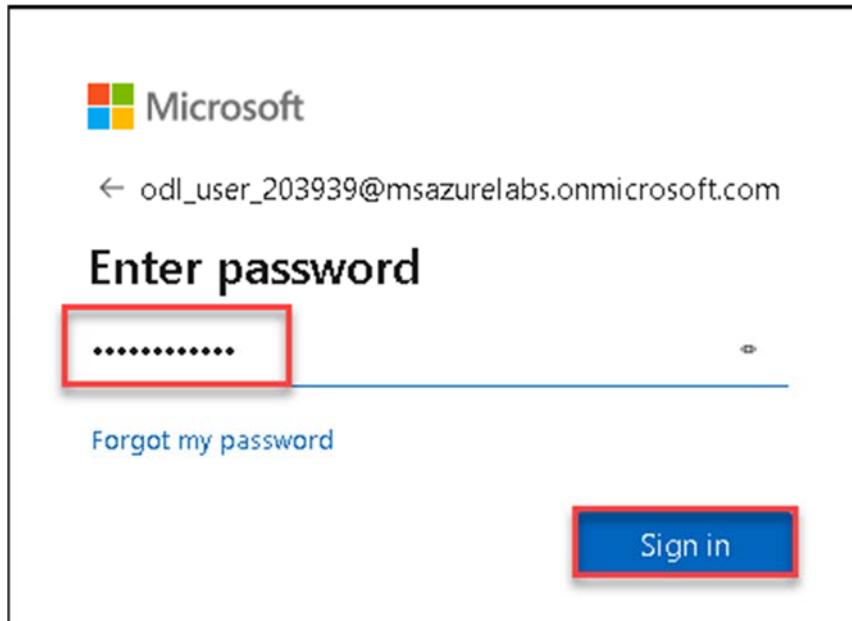


- On **Sign in to Microsoft Azure** tab you will see login screen, in that enter following email/username and then click on **Next**.

- Email/Username: odl_user_143133@udacitylabs.onmicrosoft.com



- Now enter the following password and click on **Sign in**.
 - Password: onzp40KVM*0A



- If you see the pop-up **Stay Signed in?**, click No
- If you see the pop-up **You have free Azure Advisor recommendations!**, close the window to continue the lab.
- If a **Welcome to Microsoft Azure** popup window appears, click **Maybe Later** to skip the tour.
- Now you will see Azure Portal Dashboard, click on **Resource groups** to see the resource groups.

The screenshot shows the Microsoft Azure portal homepage. At the top, there's a navigation bar with a back arrow, forward arrow, refresh icon, and a search bar containing the URL <https://portal.azure.com/#home>. Below the navigation bar is a blue header bar with the text "Microsoft Azure" and a search input field labeled "Search resources, services, and docs (G+)".

The main content area is titled "Azure services". It features a "Create a resource" button with a plus sign inside a dashed blue square, followed by icons for "Virtual machines", "App Services", "Storage accounts", and "SQL databases".

Below this section is a "Navigate" panel. It includes a "Subscriptions" link with a key icon and a "Resource groups" link with a cube icon. The "Resource groups" link is highlighted with a red rectangular box.

- Confirm you have all resource group are present as shown below.

The screenshot shows the Microsoft Azure Resource Groups page. At the top, there's a navigation bar with a back arrow, forward arrow, refresh button, and a search bar containing the URL 'https://portal.azure.com/#blade/HubsExtension/BrowseResourceGroups'. Below the search bar is a user profile for 'odl_user_136385@udacity.com'. The main content area is titled 'Resource groups' and shows one record: 'aml-quickstarts-136385'. The details for this group are: Subscription 'Udacity CloudLabs Sub - 04', Location 'South Central US', and a timestamp 'Created 10 hours ago'. There are filter options at the top, sorting by 'Name ↑', and a pagination control at the bottom showing 'Page 1 of 1'.

- Now, click on the **Resource group** and select your machine learning workspace

Azure Credentials

Here are your credentials to login to <https://portal.azure.com> and access the On Demand Lab

Username : odl_user_143133@udacitylabs.onmicrosoft.com



Password : onzp40KVM*0A

Environment Details

Resource Group : aml-vm-143133

WindowsVMDNSName : labvm4imzig5p55au6.southcentralus.cloudapp.azure.com

VMAdminUsername : demouser

VMAdminPassword : Password.1!!

12. Project: Operationalizing Machine Learning

The screenshot shows a project submission interface. At the top, a dark header bar displays the project title "Project: Operationalizing Machine Learning" and a "SUBMIT PROJECT" button. Below the header, the main section is titled "Project Submission". On the left, there's a summary card with the following details:

- DUE DATE:** Jan 4
- STATUS:** Unsubmitted (indicated by a yellow circle)
- Notes:** Project past due

To the right of the summary card is a dark callout box containing the text: "Have project questions? Ask a technical mentor or search for existing answers!" with a "ASK A MENTOR" button below it.

Submission

Please make sure to check over the [Project Rubric](#) to ensure you have met all of the project specifications. This rubric shows you the exact points reviewers will check for when looking at your work.

Before you submit, check if the following have been completed:

- Everything in the [Rubric](#) is met.
- A README file is completed
- A screencast is recorded and access is enabled
- The actual screencast video is **not** included in the submission

Ready to Submit?

Once you have all the items above completed, you can click the submit button. You'll then have the option to either provide a link to your GitHub repository or upload a zip file.

The screenshot shows a web page with the Udacity logo at the top left and a 'Logout' link at the top right. The main title 'Operationalizing Machine Learning' is centered at the top. Below it, the question 'How would you like to submit your project?' is displayed. Two options are presented: 'UPLOAD ZIP FILE' with a folder icon and 'SELECT GITHUB REPO' with a GitHub icon. A note below the GitHub icon states: 'If you're unfamiliar with GitHub, choose "Upload zip file".'

Chapter 4 : Capstone Azure Machine Learning



Lesson 1: Project

1. Project Overview

This project gives you the opportunity to use the knowledge you have obtained from this Nanodegree to solve an interesting problem. In this project, you will create two models: one using Automated ML (denoted as AutoML from now on) and one customized model whose hyperparameters are tuned using HyperDrive. You will then compare the performance of both the models and deploy the best performing model.

This project will demonstrate your ability to use an external dataset in your workspace, train a model using the different tools available in the AzureML framework as well as your ability to deploy the model as a web service.

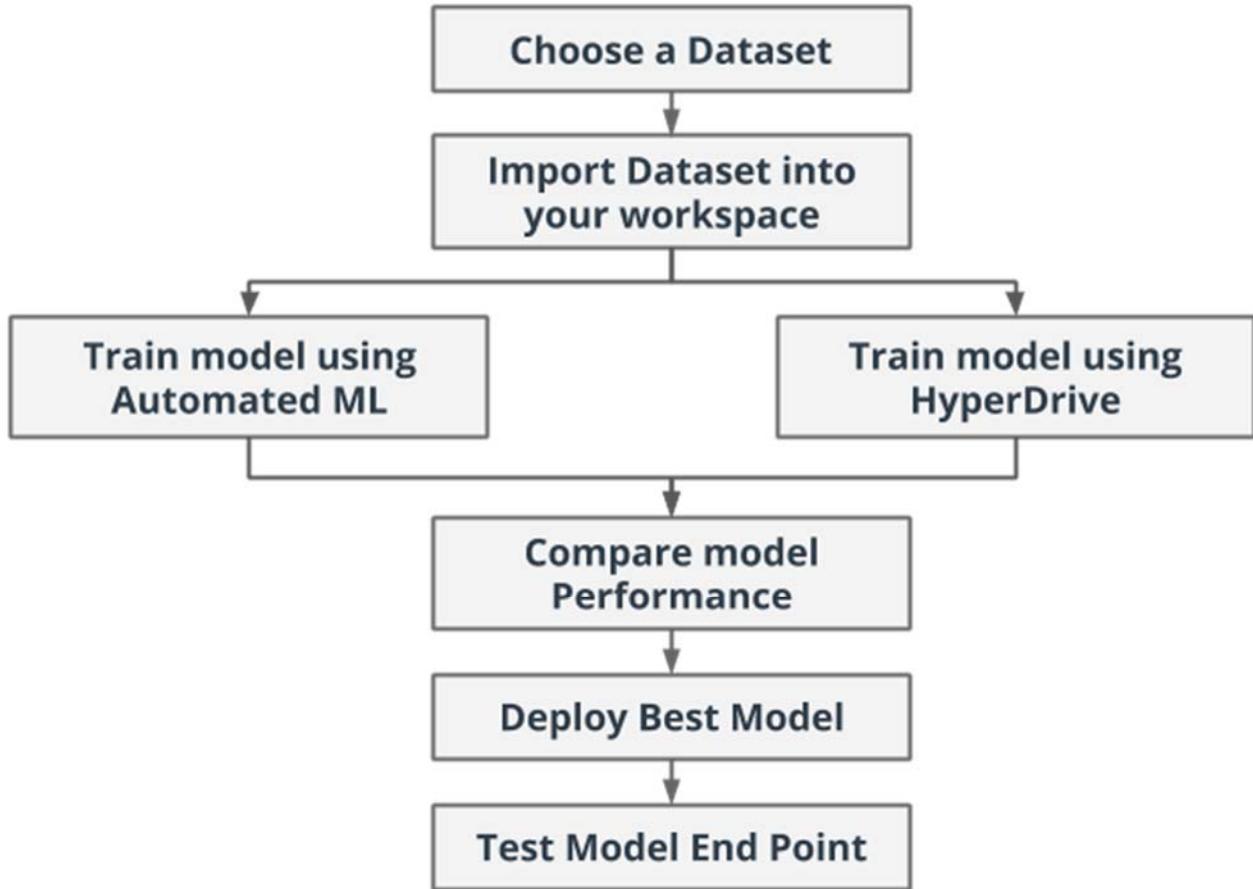
<https://photos.app.goo.gl/jhLtX9TjrcqMnnAT6>

How it works

You will be using both the `hyperdrive` and `automl` API from `azureml` to build this project. You can choose the model you want to train, and the data you want to use. However, the data you use needs to be external and not available in the Azure's ecosystem. For instance, you can use the [Heart Failure Prediction](#) dataset from Kaggle to build a classification model.

Project Workflow

To complete this project, you will have to perform these overall tasks, in the order given below:



We will go over these tasks in more detail in the later pages. After you have deployed the model and tested the web service, you will need to submit a README file with details about the data, the model, its performance, and how you deployed it. Finally, make sure that you check the project rubrics to make sure that your project has all the requirements the reviewer will be looking for in your submission.

Useful Links

You may find these links useful when building your project

- [Automated ML Overview](#)
- [HyperDrive Overview](#)
- [Deploy and Consume your model](#)
- [Hyperparameter tuning for ML models example](#)

2. Step 1: Project Setup

By the end of this task, you should have your workspace set up and your data accessible in your workspace.

You can use the [lab](#) provided by Udacity or your own Azure account to complete the project.

If you plan to use the lab provided to you, the capstone GitHub repo address and files are downloaded and put on the desktop for your convenience.

Note: You will have a maximum of **10 chances** to load and work in the lab. You will not be able to access the lab after 10 trials. Please load the lab wisely. Be careful with your configuration if you use the lab provided to you. **Your experiment will timeout after a certain amount of time! All your work saved locally in the lab will be lost once it times out.** Always **SAVE** your work in **Github!!!**

Set up your Azure Development Environment

Before you start working on your project, make sure that your Azure ML Studio Workspace is set up. Here are the main things that you need to do:

-
-  If you haven't already, create a new workspace in Azure ML Studio
 -  If you do not have a VM set up, then create a compute instance VM to run your jupyter notebooks
-

Download Starter Files

Some starter files have been provided for you. These files have some boilerplate code and TODOs that will help you start working on your project. You can find them here.

Note: After you have downloaded the starter files, you should use the same files to do your work in. The reviewers will be looking for the same files/filenames to judge your submission. It will also help ensure that reviewers can successfully run your project and don't have problems getting your code working because of missing files or incorrect file paths.

-  Download and unzip starter files in this [repo](#)
-  Optional: You can also create a separate folder in your workspace to keep all your project files together.
-  Uploaded files to your workspace

https://github.com/udacity/nd00333-capstone/tree/master/starter_file

Dataset

For this project, you can use any dataset you want and choose any task for that dataset. However, the dataset needs to be **external** to the Azure ML ecosystem. For instance, you can use a dataset from Kaggle, UCI ML repository, or any other open-source data repositories. Before choosing a task and dataset, make sure that it is supported by Azure ML's `automl` API



Choose a dataset

Make sure you can access the dataset in your workspace

4. Step 2: Model Training and Deployment

At the end of this task, you will have finished training models on your dataset and deployed the best model.

Note: Be careful with your configuration if you use the lab provided to you. **Your experiment will timeout after 2 hours! All your work saved locally in the lab will be lost once it times out.** Always **SAVE** your work in **Github!!!**

Train a model using Automated ML

The `automl.ipynb` file contains a starter code to help you train a model using Automated ML. The file contains some TODOs that you need to finish.



- I have finished all the TODOs in the `automl.ipynb` file
- Take a screenshot of the RunDetails widget that shows the progress of the training runs of the different experiments
- Take a screenshot of the best model with its run id

Train a model with HyperDrive

The `hyperparameter_tuning.ipynb` file contains some starter codes to help you train a model and perform hyperparameter tuning using HyperDrive.

The type of model you use is not important. You can use ML models through Scikit-learn or Deep Learning models like ANNs and CNNs through Keras, TensorFlow, or PyTorch for this part of the project.

-
- I have finished all the TODOs in the `hyperparameter_tuning.ipynb` file
 - I have used one type of sampling method: grid sampling, random sampling or Bayesian sampling
 - Specify an early termination policy (not required in case of Bayesian sampling)
 - I have tuned at least 2 hyperparameters in my model
 - I have visualized the progress and performance of my hyperparameter runs (Logs metrics during the training process)
 - I have saved the best model from all the hyperparameter runs
 - Take a screenshot of the RunDetails widget that shows the progress of the training runs of the different experiments
 - Take a screenshot of the best model with its run id and the different hyperparameters that were tuned



Model Deployment

After you have trained your models. You will have to deploy your best model as a webservice and test the model endpoint



- I have deployed the model with the best accuracy as a webservice
- I have tested the webservice by sending a request to the model endpoint
- I have deleted webservice and shut down all the computes that I have used
- Take a screenshot showing the model endpoint as active

4. Step 3: Complete the README

By the end of this task, you will have a detailed and professional README file of your completed project.

Note: Be careful with your configuration if you use the lab provided to you. **Your experiment will timeout after 2 hours! All your work saved locally in the lab will be lost once it times out.** Always **SAVE** your work in **Github!!!**

An important part of your project is creating a README file that describes the project, explains how to set up and run the code, and describes your benchmarks and results. We've included a template in the starter files (that you downloaded earlier), with TODOs for each of the things you should include.

Note: You'll see some optional TODOs for "standout suggestions". We'll go over these on the next page.



I have completed all of the required TODOs in the README file

5. Step 4: Standout suggestions

If you've been having a good time with this project and want to take it further, here are some suggestions for a whole bunch of additional things you can do with it. If you want the additional practice or want to turn this project into a nice portfolio piece, we recommend taking a shot at these—but note that these are all optional, so you can skip any (or all) of them and go straight to the project submission page.

Note: Be careful with your configuration if you use the lab provided to you. **Your experiment will timeout after 2 hours! All your work saved locally in the lab will be lost once it times out.** Always **SAVE** your work in **Github!!!**

Here are a list of the standout suggestions for your reference:

-  **Convert your model to ONNX format.** Your jupyter notebook contains documentation and code for converting the model to the ONNX format.
-  **Deploy your model to the Edge using Azure IoT Edge.** Your submission includes an extra jupyter notebook containing the code for deploying the model to the edge. The README file contains documentation about the deployment as well as screenshots showing the data arriving at your IoT hub. You can use [this tutorial](#) for reference
-  **Enable logging in your deployed web app.** Can you log useful data about the requests being sent to the webapp? This may include, inference time, the time at which the request arrived and other details. You can find out more about logging [here](#). Your submission should contain code to perform logging in the deployed web app. Your `README` file also contains information about the details logged and a screenshot of the logs and metrics collected.

Update your `README` file



If you have done any of the standout suggestions, make sure that you have documented it in your `README` file

6. Step 5: Create a Screencast

In the last step, you will need to create a screencast to demonstrate your work and the deployed model.

Screencast

In this project, you need to record a screencast that shows the entire process of the working ML application. The screencast should meet the following criteria:



- Screencast is 1-5 minutes in length
- Audio is clear and understandable
- Video is 1080P or higher with 16:9 aspect ratio
- Text is readable

Screencast content requirement

A screencast is a useful way to share your project with others. In this project, you need to record a screencast that shows the entire process of the working ML application.

The screencast should include a demonstration of the following area:

- A working model
 - Demo of the deployed model
 - Demo of a sample request sent to the endpoint and its response
 - Demo of any additional feature of your model
 - Add the screencast link to the README file
-

7. Step 6: Check your work

Before you submit

Your reviewer will use the instructions you provide to run and test your project, so it's important to make sure your instructions are clear, that your code works, and that all necessary dependencies are included.

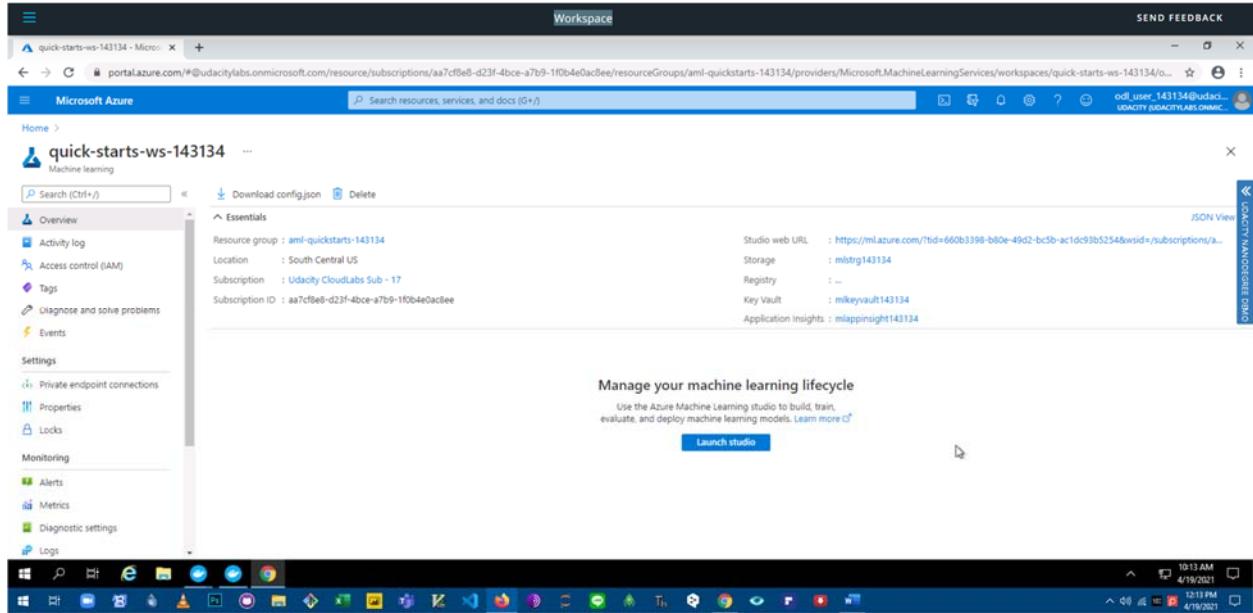
Note: The models themselves are large files and you do not need to include them in your submission. However, do remember to include all the scripts and notebooks that you used in your project.

Before submitting, be sure you've done the following:



- Make sure that you have provided all the files in your submission
- Make sure that any special instructions for running your project are mentioned in your `README`
- Test your instructions to make sure that your project runs
- The actual screencast video is **not** included in the submission
- Zip the entire directory containing your code and all required files (except the model files). Make sure you've included everything that your reviewer will need to run your project.
- To reduce the size of your submission, make sure the zip file does *not* include the model files. The final zip file should be no more than a few megabytes in size.
- Make sure that you have included all the screenshots and video links in your submission
- If you'd like to be especially thorough, you can have a look at the project [rubric](#). This describes the exact requirements that your reviewer will be looking at when they go over your submission.

8. Workspace



Getting Started with Lab

1. Once the environment is provisioned, a virtual machine (JumpVM) and lab guide will get loaded in your browser. Use this virtual machine throughout the workshop to perform the the lab.
2. To get the lab environment details, you can select Lab Environment tab.

Udacity Nanodegree demo
0 hour(s), 56 minute(s) remaining × HIDE

Lab Guide Lab Environment Help

Environment Details Lab Resources

Azure Credentials

Here are your credentials to login to
<https://portal.azure.com> and access the On Demand
Lab

Username 

Password 

Environment Details

Resource Group : aml-vm-136385

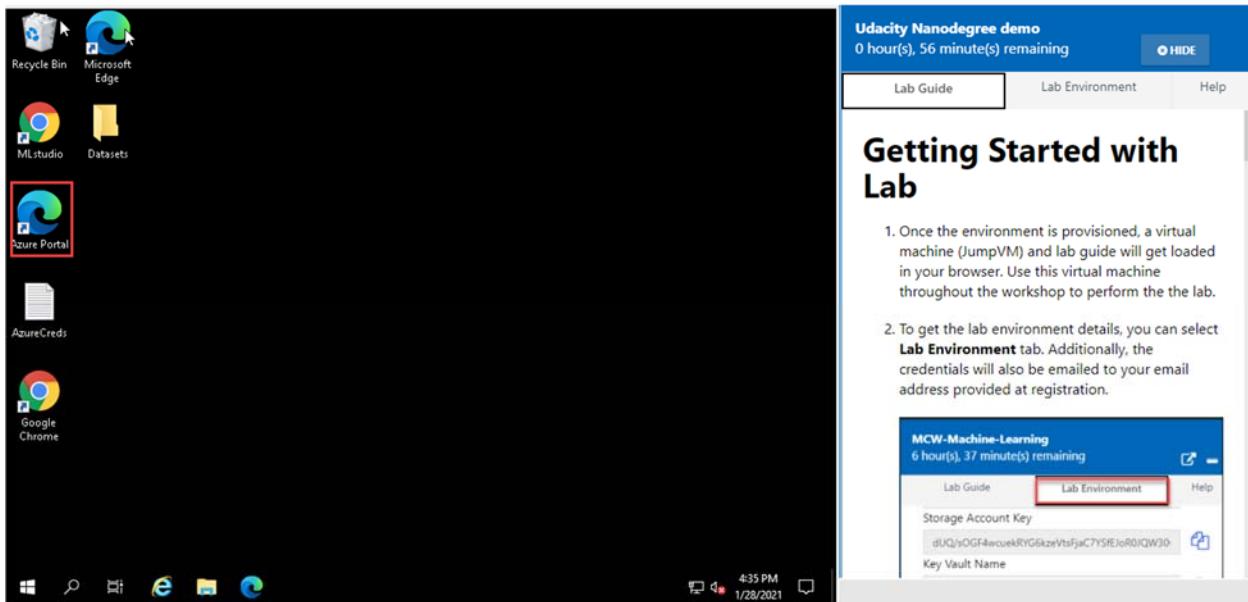
Windows 

VMDNS Name

VMAdmin 

Login to Azure Portal

1. In the JumpVM, click on Azure portal shortcut of Microsoft Edge browser which is created on desktop.



- When you click on Azure portal, edge browser welcome screen will come up, select Get started.

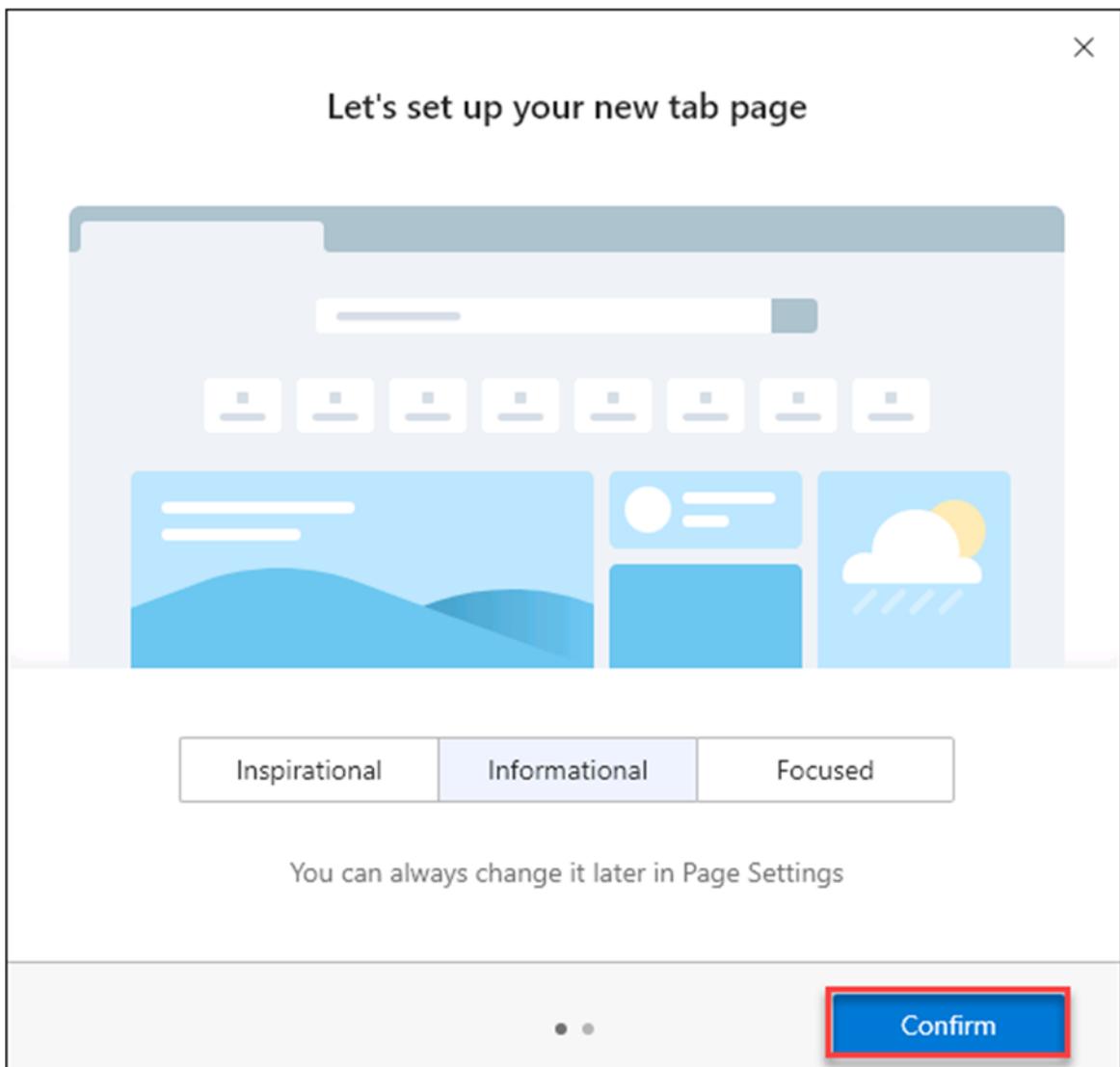


Welcome to the new Microsoft Edge

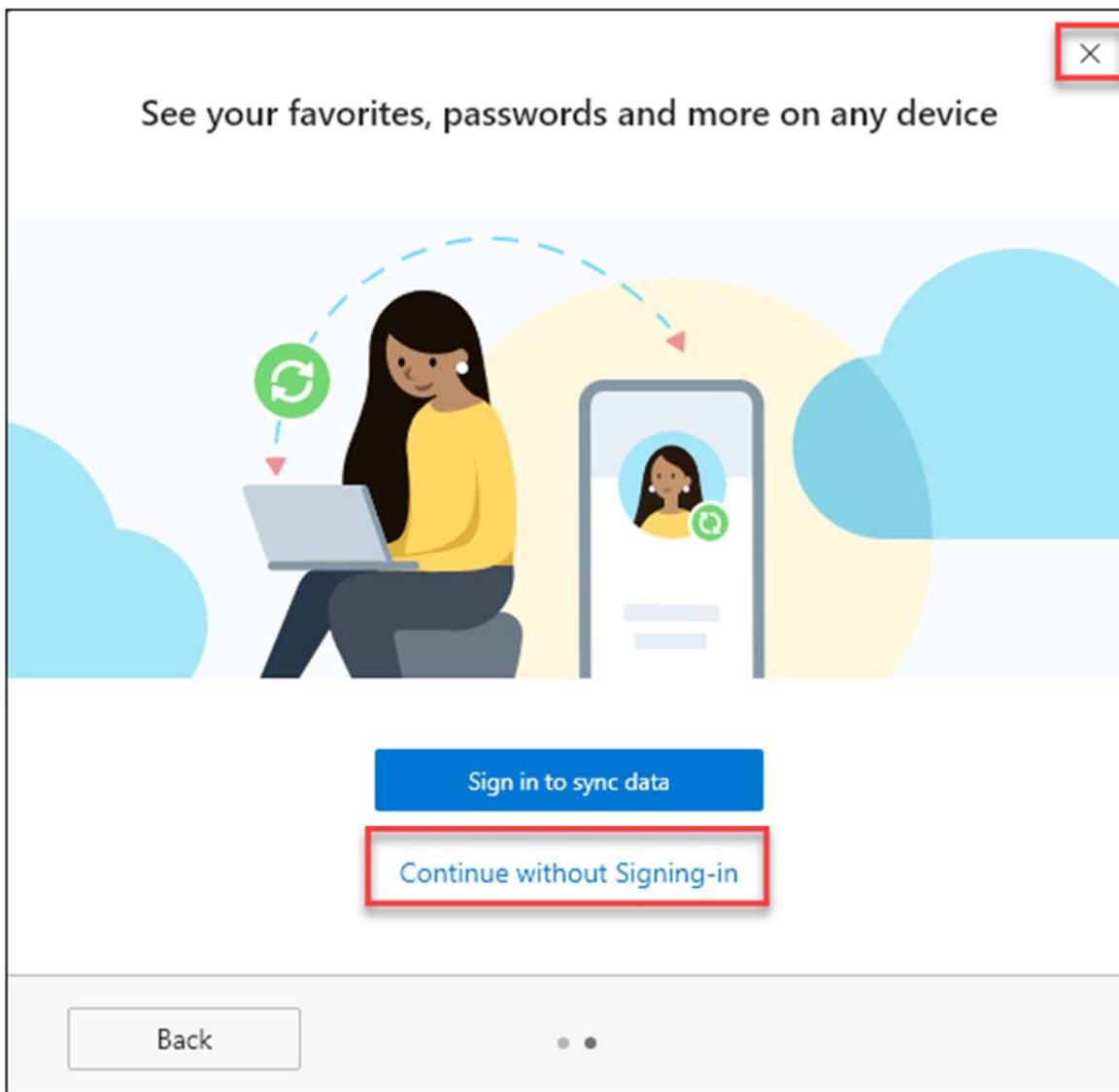
A fast, safe, and productive web browser that works for
you

[Get started](#)

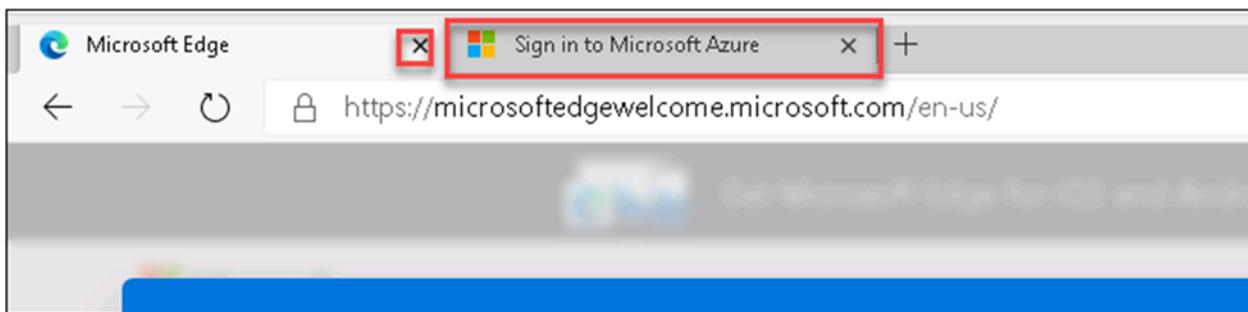
3. On next window, click on Confirm.



4. Now, you can close the popup which is coming up.

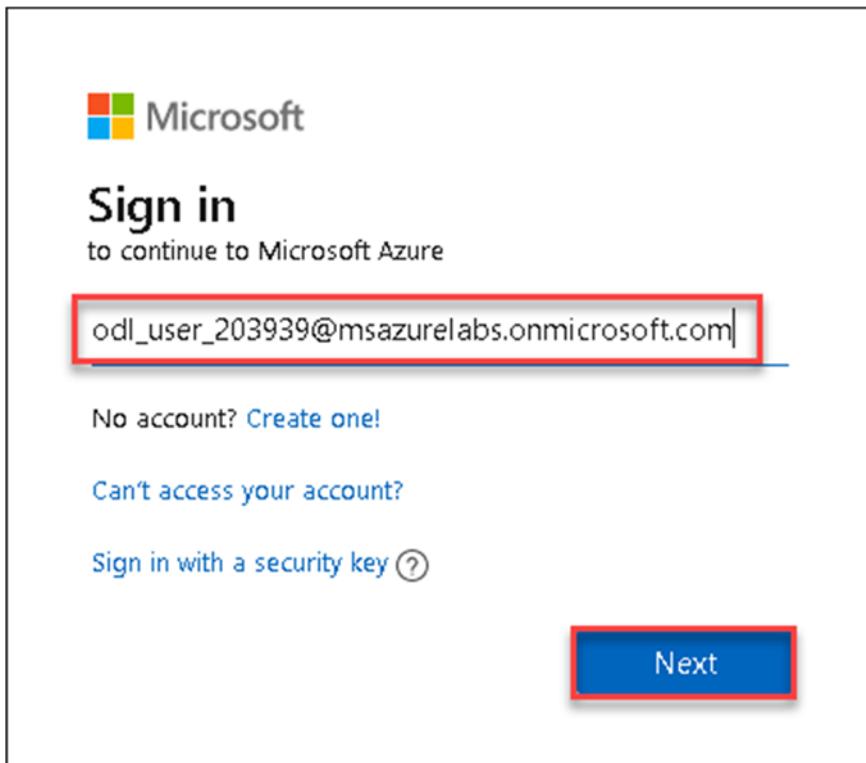


5. Now, you will see two tabs in edge browser, close first tab named with Microsoft Edge.



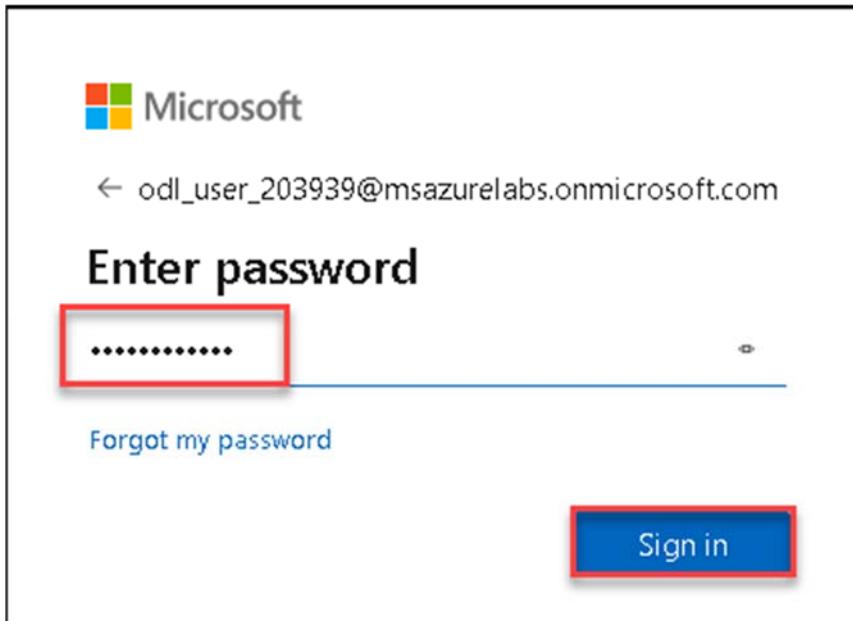
6. On Sign in to Microsoft Azure tab you will see login screen, in that enter following email/username and then click on Next.

- Email/Username: odl_user_143134@udacitylabs.onmicrosoft.com

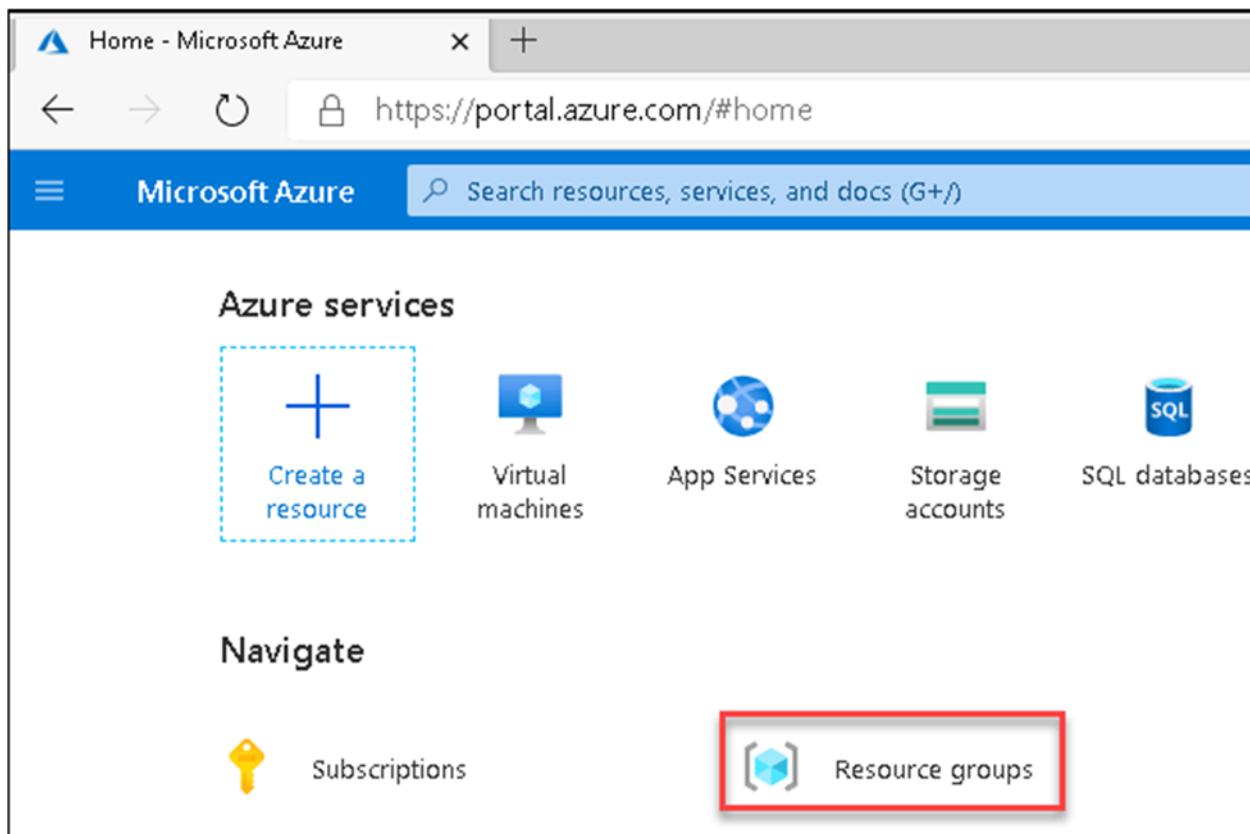


□ Now enter the following password and click on Sign in.

- Password: aqku38FTN*8U



- If you see the pop-up Stay Signed in?, click No
- If you see the pop-up You have free Azure Advisor recommendations!, close the window to continue the lab.
- If a Welcome to Microsoft Azure popup window appears, click Maybe Later to skip the tour.
- Now you will see Azure Portal Dashboard, click on Resource groups to see the resource groups.



- Confirm you have all resource group are present as shown below.

The screenshot shows the Microsoft Azure portal interface. The title bar says 'Resource groups - Microsoft Azure'. The address bar shows the URL 'https://portal.azure.com/#blade/HubsExtension/BrowseResourceGroups'. The top navigation bar includes 'Microsoft Azure', a search bar, and user information 'odl_user_136385@udacity.com'. Below the search bar are buttons for 'New', 'Manage view', 'Refresh', 'Export to CSV', 'Open query', 'Assign tags', and 'Feedback'. A filter bar allows filtering by 'Subscription == all', 'Location == all', and 'Add filter'. The main content area displays 'Showing 1 to 1 of 1 records.' with one item listed: 'aml-quickstarts-136385' under 'Subscription' 'Udacity CloudLabs Sub - 04' and 'Location' 'South Central US'. There are also 'Name', 'Subscription', and 'Location' sorting options. At the bottom, there are navigation links for '< Previous', 'Page 1 of 1', and 'Next >'.

- Now, click on the Resource group and select your machine learning workspace

Azure Credentials

Here are your credentials to login to <https://portal.azure.com> and access the On Demand Lab

Username : odl_user_143134@udacitylabs.onmicrosoft.com

Password : aqku38FTN*8U

Environment Details

Resource Group : aml-vm-143134

WindowsVMDNSName : labvm6xk3zqhzc5gc.southcentralus.cloudapp.azure.com

VMAdminUsername : demouser

VMAdminPassword: Password.1!!

9. Capstone Project - Azure Machine Learning Engineer

The screenshot shows a project submission interface. At the top, it says "Capstone Project - Azure Machine Learning Engineer" and has a "SUBMIT PROJECT" button. Below that is a section titled "Project Submission". It displays two cards: one for "DUE DATE" (Feb 14) and one for "STATUS" (Unsubmitted, Project past due). To the right is a dark blue callout bubble with white text asking about project questions and an "ASK A MENTOR" button.

Submission
Please double-check if the [Project Rubric](#) to ensure you have met all of the project specifications. This rubric shows you the exact points reviewers will check for when looking at your work.

Ready to Submit?
Once you have all the items above completed, you can click the submit button. You'll then have the option to either provide a link to your GitHub repository or upload a zip file.

Capstone - Azure Machine Learning Engineer

How would you like to submit your project?



UPLOAD ZIP FILE



SELECT GITHUB REPO

If you're unfamiliar with
GitHub, choose "Upload zip file".

Chapter 5: Career Services

Lesson 1: Take 30 Min TO Improve Your LinkedIn

1. Get Opportunities with LinkedIn

<https://photos.app.goo.gl/jRF5fpV2pJ1NR8K36>

Why Use LinkedIn for Networking?

Employers Flock to LinkedIn

In the U.S. alone,

- 149 million workers have LinkedIn profiles.
- 20,000+ companies recruit on LinkedIn.
- Over 3 *million* jobs are posted *a month*.

Using LinkedIn is a smarter way to build your professional network. Did you know, [40% of jobs are hired through referrals](#)?! Ensure you're connecting with others across your field using LinkedIn - they'll help you find your next job and grow professionally.

No Resubmission Required

Unlike other Nanodegree projects, you submit to a Career Service once to get feedback and do not need to meet specifications to continue on in your program. We highly encourage you to take advantage of this guidance; most Alumni who report jobs and other opportunities (such as collaboration) attribute their success to networking.

Let's Get Started

Submit your LinkedIn profile and receive in-depth and honest feedback from a Career Reviewer. They'll ensure you're presenting your best self to your network.

Resources

We recommend reviewing these resources before submitting for feedback to ensure your LinkedIn profile is ready for review.

1. [Rubric](#). *Your project will be reviewed by a Udacity Career Reviewer against this rubric.*
2. [Checklist](#). *Based on the project rubric, this is a handy checklist to use during your LinkedIn building.*
3. [Career Resource Center](#). *Find additional tips and guides on developing your LinkedIn profile.*

If you're looking for more guidance, continue on to the lesson to learn more about networking, personal branding, and LinkedIn profile best practices.

Career Review Requirement

Your Nanodegree program was designed to ensure your long-term success in the field, and to pursue a successful career you must showcase the skills you've learned to employers effectively. Therefore, this is a requirement for graduation.

Opting Out

If improving your LinkedIn profile does not currently align with your Nanodegree goals, you may opt-out of this graduation requirement.

If you would like to opt-out, please click [here](#).

2. Use Your Story to Stand Out

How to Stand Out

Imagine: You're a hiring manager and need to pick 5 people to interview for a role. You get 50 applications, and everyone seems pretty qualified. How do you compare job candidates? You'll probably pick the candidates that stand out the most to you - likely the people who communicate their motivation and dedication for the job.

Personal Stories

A job candidate's personal story is always unique to them. Employers aren't just looking for someone with the skills, but for someone who can help drive the company's mission forward. That's why they need to know your work ethic and what drives you.

Personal stories are memorable. Connect with your potential boss or colleague by telling your personal story. You want employers to know how you solve problems, overcome challenges, and achieve results. You want employers to know what excites you, what motivates you, what drives you forward.

All of this can be achieved through effective storytelling and building a personal brand.

In this next video, meet **Chris Saden**, a software engineer at Udacity, who will share with us how he redefined his personal brand during a career change from admissions counselor to full stack engineer.

<https://photos.app.goo.gl/rqx7yPRvWzVvNwxT9>

3. Why Use an Elevator Pitch

<https://photos.app.goo.gl/tacQrQMAxLs3NKKq6>

4. Create Your Elevator Pitch

<https://photos.app.goo.gl/ninXrutWzfXAEVas8>

Use Your Elevator Pitch with Recruiters

<https://photos.app.goo.gl/7BdxgXdFUP1VSyGe8>

5. Use Your Elevator Pitch on LinkedIn

Communicate Your Elevator Pitch in your LinkedIn Summary

Let's take another look at Chris's elevator pitch:



Hi, I'm Chris, a Full Stack Software Engineer who loves building education products. I recently developed a web app using AngularJS that lets teachers share student writing samples anonymously. I'd love to combine my passion for learning and teaching with my software development skills to continue building personalized learning products for people.

Use the same talking points for your LinkedIn summary, which may be the first thing people read on your profile. When recruiters (or anyone else) search on LinkedIn, keywords from your summary contribute to your rank in search results.

LinkedIn Summary Guidelines

- Keep the summary to 3-5 points maximum.
- Use numbers where possible.
- Always use active instead of passive language.
- Include a call to action so that people are more likely to contact you for more information.

See how Chris uses the same elevator pitch to craft his LinkedIn summary:

Chris Saden • 1st
Software Engineer at Udacity
San Francisco, California

[Message](#)

[More...](#)

 Udacity

 Georgia Institute of
Technology

 See contact info

 See connections (500+)

I'm a teacher turned software engineer who loves building personalized learning apps. My latest web app connects teachers around the world to collaborate on lesson plans. Right now, I'm working on improving Udacity's Career Portal, a customized path to help Nanodegree graduates get hired in tech.
Skills: Python, AngularJS, Node.js, React, SQL

Time to prepare your pitch!

Take 30 min to develop your elevator pitch and check off the tasks below as you go.



-  Articulate 3 main points.
-  Say your pitch out loud to yourself (in front of the mirror, to your cat, your rubber duck, your baby...).
-  Say your pitch to someone else (your roommate, partner, friend, bus driver...).

6. Create Your Profile With SEO In Mind

Before we continue:

If you already have a LinkedIn profile - fantastic! Use this section to investigate, revise, and improve the highlighted sections.

If you don't have a LinkedIn profile - you will soon! Open up [LinkedIn](#) in another browser tab and create an account. You can expect to spend about 10 minutes going through LinkedIn's "Welcome" flow.

Keywords and SEO

As there are millions of LinkedIn users, you may be wondering how you stand out or are surfaced to recruiters. Below, we dive into the sections of your profile which are essential to SEO (search engine optimization - or optimizing your profile for searches on LinkedIn).

On your LinkedIn profile, more-so than on a resume or other application materials, it is helpful to use keywords specific to the job you are targeting. This is especially true of the words you include in your headline, summary, and current job title. Think of the words that a recruiter might type to search for candidates of your target job and check to ensure they're listed in your profile.

For example, if you were looking for a job developing mobile augmented reality apps, you might want to include words like engineer, iOS, Android, Unity, or 3D.

Pro Tip: To find keywords to add, read job descriptions of positions you are targeting.

For example, when you search "Udacity Software Engineer" on LinkedIn, you may find similar results to the image below. What do you notice about the search results?



P • 2nd

Platform Engineer at Udacity
San Francisco Bay Area

Summary: ...as a Internal Product Engineer then transitioned to be a software engineer. She joined Udacity's...



15 shared connections



O • 2nd

Sr. Software Engineer at Udacity
San Francisco Bay Area

Current: Back End Engineering Lead at Udacity



31 shared connections



V • 2nd

Software Engineer at Udacity
San Francisco Bay Area

Past: Software Engineer at .



5 shared connections

LinkedIn search results for "Udacity software engineer" (with some redacted information)

You may have noticed that:

1. All have profile pictures.
2. All have the keyword "software engineer" in their Headline, Summary, and Past Experience sections.
3. All are second connections with the searcher. The people you're 1st, 2nd, and 3rd connections to will be prioritized in your search results. If you want to be surfaced in search results, you'll need to connect with others in your industry to boost your visibility.

Up next, let's take a closer look at your profile picture, name, headline, and summary. We'll cover your experiences and connections later in the lesson.

7. Profile Essentials

Name

Here you will need to use your real name, not a pseudonym. No Codergirl42, no HireMe McJoberson, just your real first and last name.

Pro Tip: You can add former names to your Profile. See [instructions](#) from LinkedIn.

Profile Picture

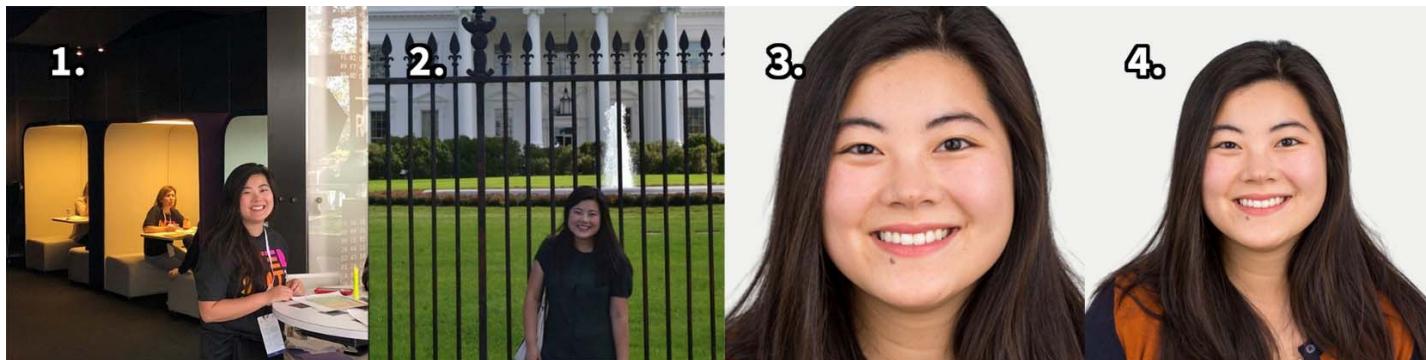
Do

- Upload a picture cropped to your head and shoulders.
- Aim for a high resolution photo.

Don't

- No inappropriate or unprofessional items or environments.
- Don't include other people - this is *your* profile after all!
- No selfies. Someone is willing to take a photo for you, we promise!

Quiz: Professional Profile Picture



QUESTION 1 OF 2

Of the four options above, which is the best LinkedIn profile picture?

1

2

3

4

Try Again

Look at all the photos individually. Which one is the one least distracting and most clear?

TRY AGAIN

Look at all the photos individually. Which one is the one least distracting and most clear?

TRY AGAIN



Microsoft Azure
Noel Ching

Look at all the photos individually. Which one is the one least distracting and most clear?

TRY AGAIN

Correct! This photo is the most professional. It's a high quality photo set against a neutral background.

Picture 1, even though it's set in a work location, it includes other people and is messy (see the items sprawled on the table).

Picture 2, even though it's set at a cool location (the White House), it's low quality. The viewer wouldn't even be able to discern the person's face from this photo.

Picture 3, while the same photo as picture four, it's less professional due to the crop - the close zoom is a bit unsettling and looks like a floating head.

CONTINUE

Headline

Your headline should be one of the following options:

- **Your Current Job Title.** This is the simplest but most effective way to optimize SEO; LinkedIn even defaults to suggesting you do this.
- **Your Education.** If you do not yet have a job in your new industry, add your education such as "Data Analyst Nanodegree Graduate." You still include the key term "data analyst".
- **Your Target Job.** Remember when Chris said his job search improved when he finally called himself a software engineer? If you're a Nanodegree graduate with a portfolio of real-world projects to share, feel confident and list your target job title.

Summary

You should already have your summary, crafted from your elevator pitch, from the exercise with Chris. If you need a refresher, just go back to "Use Your Elevator Pitch on LinkedIn." Your summary should:

- Be written in first person, with a professional but conversational tone.
- Include your key abilities and contributions.
- Include 5 key skills (programming languages, software, etc.) for SEO purposes.

For reference, here's Chris's summary:

Chris Saden • 1st

Software Engineer at Udacity
San Francisco, California

[Message](#)

[More...](#)



Udacity



Georgia Institute of
Technology



[See contact info](#)



[See connections \(500+\)](#)

I'm a teacher turned software engineer who loves building personalized learning apps. My latest web app connects teachers around the world to collaborate on lesson plans. Right now, I'm working on improving Udacity's Career Portal, a customized path to help Nanodegree graduates get hired in tech. Skills: Python, AngularJS, Node.js, React, SQL

Exercise: What are your keywords?

Write down 3-5 keywords that you would like to feature prominently and repeatedly in your LinkedIn profile. Now, go read 2-3 job descriptions for your target positions. See if you can find two keywords that you hadn't thought to include before.

Exercise: What are your keywords?

Write down 3-5 keywords that you would like to feature prominently and repeatedly in your LinkedIn profile. Now, go read 2-3 job descriptions for your target positions. See if you can find two keywords that you hadn't thought to include before.

aaa

SUBMIT

Next up, we'll go through optimizing keywords in your work experiences and other accomplishments.

8. Work Experiences & Accomplishments

Showcase Successful Work Experiences

The Experience section of your LinkedIn profile should mirror your resume. Ensuring your resume and LinkedIn are consistent will help to build your personal brand.

In our [free course on resume writing](#), we provide detailed guidance on how to describe your work experiences in a way that showcases success. Here are some guidelines to keep in mind for both resumes and LinkedIn profiles:

Be concise.

- Include only relevant experience.
- Keep each description to 3 bullet points or less.
- At least 1 of these bullet points should demonstrate an individual contribution.
- At least 1 of these bullet points should communicate a project result (success metrics, findings).

Always convey

Action, Numbers, Success.

- Action - Use active verbs to describe what you did.
- Numbers - Quantify your accomplishments.
- Success - Define each experience in terms of what you learned or achieved.

Order matters.

Make sure that your most relevant and most recent experience appears at the top of your profile.

Accomplishments

You can adapt the "Accomplishments" section of your LinkedIn profile to a wide variety of work experiences and achievements. For example, the "Accomplishments" section for "Projects" is an excellent place to **showcase your Udacity projects!** You can also list the Udacity courses you have taken in the "Accomplishments" section for "Courses."

When describing your projects be sure to include the specific technical skills you learned relevant to your target field. Think about the parts of your Udacity projects that you are most proud of and frame them as successes in your project description. Here are a couple good examples from Udacity alums:

[Kristin, Virtual Reality Developer Nanodegree Program Alum](#) - Notice how Kristin uses the project section to showcase the technologies she has worked with (i.e. Unity, Oculus Rift). She also provides links to visuals for each project, so that it's easy for recruiters and connections to dive deeper into a project that piques their curiosity.

[Michael, Self Driving Car Nanodegree Program Alum](#) - Notice how Michael uses the "Projects" section to include more details about his experience with autonomous vehicles, and highlights his use of particular models like convolutional neural networks.

9. Build and Strengthen Your Network

Find More Connections

In order to use your LinkedIn profile effectively, it's important to have as many connections as possible. In general, 500+ connections are needed to fully optimize your profile.

This may seem like a lot, but once you get rolling, you'll get to that number fast. After you actively start engaging on LinkedIn, by joining groups and going to networking events, your number of connections will climb. You are more likely to show up in search results on LinkedIn if you have more connections, which means you'll be more visible to recruiters.

Here are some examples of the types of connections you could add:

- current and former colleagues
- people you attended school with (high school, university, graduate school, Udacity)
- your favorite teachers and professors
- people you meet at professional events, like conferences and meetups
- volunteering companions
- friends and relatives
- roommates and neighbors

Need some incentive to get started? No time like the present. Finish the tasks below and check them off as you go!

Kickstart Your Network



- ✓ Send connection requests to 3 people you attended school with.
- ✓ Send connection requests to 3 people you have worked with in the past.
- ✓ Send connection requests to 3 people you know in a nonprofessional context, but respect as professionals.

Join Groups

Joining groups allows you to see content from and post content to people who share your interests. Join groups which relate to your industry, of professionals who live near you, or of people who attended the same alma mater. For example, it is common for people to join location-based professional groups, like the Chicago Young Professionals, or for Udacity students to join the Udacity Alumni Network. You might want to join groups associated with specific technologies, like Swift Developers, User Interface Design, or Self-Driving Cars.

Joining groups vastly increases the number of people in your network, increasing the visibility of your profile and the number of opportunities that come your way.

Quick group exercise

Sign in to your profile and join a group associated with...

- a school
- a technical field
- a location

10. Reaching Out on LinkedIn

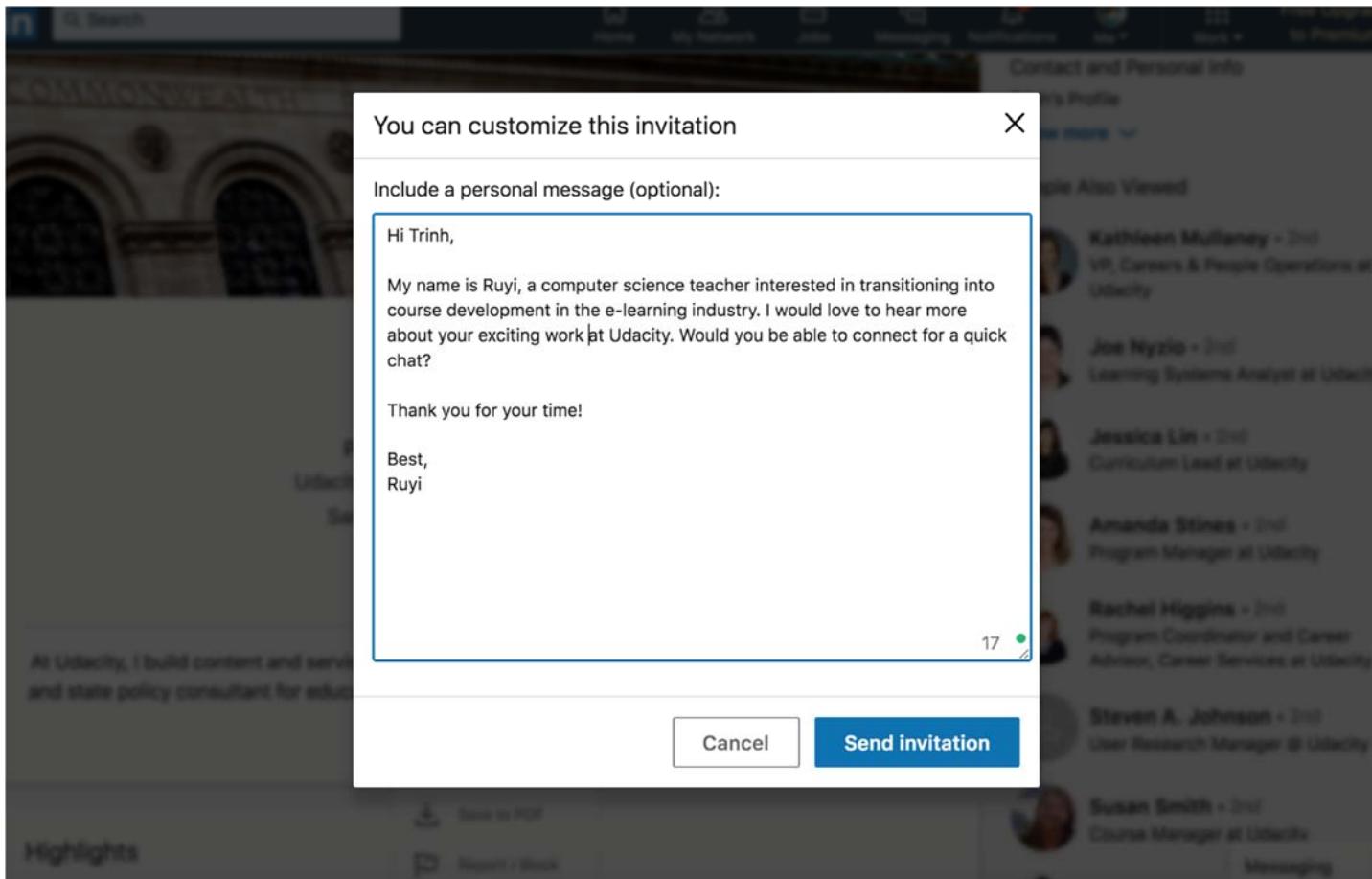
Optimizing your LinkedIn Profile is important for showcasing yourself to any recruiter or employer. However, you can do more than passively wait for someone else to reach out.

After improving your LinkedIn Professional Profile, connect with other industry professionals via LinkedIn to set up an informational interview, learn about a position, or build new professional relationships.

Example LinkedIn Conversation

Platforms like LinkedIn make it easy for people to find and connect with each other. **When you connect with someone, make sure to write a note in the request.** You only have 300

characters in a connection request to make a first great impression. Below is an example of how you might request an informational interview.



Your invitation should always be polite, professional, and friendly!

Once they've accepted your invitation to connect, you can move forward with sending your request to meet for an informational interview.

Trinh Nguyen

Program Manager at Udacity

...

Ruyi

5:01 PM

Trinh Nguyen is now a connection



Hey Ruyi, thanks for connecting and telling me a bit about yourself. I'm happy to chat. What would you like to learn about Udacity?

5:03 PM

Hi Trinh

Thanks

Hey Trinh

Hi Trinh, thank you for accepting my invitation to connect! I noticed your exciting work guiding content updates in the Intro to Programming Nanodegree Program at Udacity. I wanted to ask if we could meet for a short informational interview. I began exploring online courses, which inspired my interest in pursuing a career in course development and management. I hope to eventually enter into a new role at an e-learning company to create engaging courses.

Would you be available to meet in two weeks at a time that works for you? I would be happy to take you out for coffee and hear about your experiences. Thank you for your time, and I look forward to hearing



Send

...



Microsoft Azure
Noel Ching

404

Machine Learning Engineer for Microsoft Azure

Connection established! Now it's up to you to schedule a time to meet.

Don't forget: It's expected that one will receive connection requests through LinkedIn.

There's no reason to think you're inconveniencing the other person. Like you, other people are interested in improving professionally, which can revolve around sharing their own experiences and knowledge or discussing current topics of interest in the industry.

Not sure how to start making connections? Review our [advice in the Career Resource Center](#) on how to successfully request and conduct an informational interview.

You may feel nervous or unsure at first, but with practice and frequent networking interactions, you will soon become a networking pro.

11. Boost Your Visibility

You've optimized your profile. What's next?

Creating a killer profile is a key first step toward building a strong LinkedIn presence, but there is a lot more you can do to nurture your connections and make your profile easy to find. Here are some of the avenues LinkedIn provides to engage with your professional community.

Follow

Follow the influential companies in your target field to stay informed about new discoveries, trends, and current events in your professional sphere. Identify companies where you can imagine working, and follow those too; you'll see job posts sooner and you'll be more likely to show up in their recruiters' searches.

Post, Like, and Comment

Authoring original posts is a fantastic way to boost your visibility, but if you are a little shy about posting at first, don't worry. Plenty of people get meaningful value out of LinkedIn without authoring posts. You can also engage by liking and commenting; these small actions can go a long way toward boosting your visibility.

Endorse and Recommend

There are so many good reasons to endorse and recommend on LinkedIn. **Effective team players recognize the contributions of others.** By endorsing and recommending current and former colleagues you can show future coworkers that you are capable of acknowledging your teammates. This acknowledgement will also deepen your relationship with the colleagues and collaborators you endorse, not to mention it will increase the likelihood of someone acknowledging you in return.

Here's a profile from [Mike, a former Udacity employee](#). Take a look at his "Recommendations" section. Notice that he has received and given multiple recommendations. Not only does this give us a window into working with Mike, but it shows that he's a colorful and clear communicator.

12. Up Next



Up Next

By now, you know how to:

- Market yourself effectively with your elevator pitch.
- Optimize the essential LinkedIn profile components.
- Effectively describe your work experience and projects.
- Create a strong network by adding connections and joining groups.
- Raise your LinkedIn visibility.

Being confident in this will help you network naturally, whether on LinkedIn or at an event in-person.

If you are seeking additional advice on how to enhance your LinkedIn profile, check out the following resources:

- [How to Stand Out on LinkedIn](#)
- [Tips for building a great LinkedIn Profile](#)
- [Land a Great Job with LinkedIn](#)
- [31 Tips for LinkedIn](#)

Move on to the LinkedIn Profile Career Service Review and get personalized feedback on your online presence!

13. Improve Your LinkedIn Profile

The screenshot shows a project submission page. At the top, there's a dark header with the title "Improve Your LinkedIn Profile" and a "SUBMIT PROJECT" button. Below the header, the main title "Project Submission" is displayed. On the left, there are two status cards: one for "DUE DATE" showing "Feb 16" and another for "STATUS" showing "Unsubmitted" with a note "Project past due". To the right, a dark callout box contains the text "Have project questions? Ask a technical mentor or search for existing answers!" followed by a blue "ASK A MENTOR" button.

Have job opportunities come to you by optimizing your LinkedIn profile! Complete this simple process to get personalized tips from a career mentor.

Here's how to submit your LinkedIn profile for feedback now:

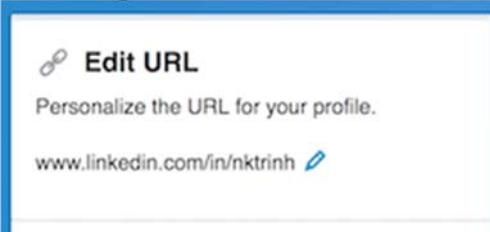
- Check out this [checklist](#) for tips on optimizing your LinkedIn profile (optional).
- **Make sure your profile is in English.** If you do not have an English profile yet, check out LinkedIn's [advice](#) on how to create a secondary language profile.
- Make your profile publicly visible. Scroll down and set the profile section toggles to Show under the “Edit Visibility” section of [your public profile edit page](#).

Edit Visibility

You control your profile's appearance for viewers who are not logged-in members. Limits you set here affect how your profile appears on search engines, profile badges, and permitted services like Outlook.

[Learn more](#)

Your profile's public visibility On 

- Create a custom profile URL in the “Edit URL” section of [your public profile edit page](#).


 **Edit URL**
Personalize the URL for your profile.
www.linkedin.com/in/nktrinh 
- Submit your public profile URL (such as www.linkedin.com/in/udacitystudent) for guidance from a career mentor by clicking “Submit Project” below.



Submit Link

Project link
Check that your link is publicly accessible

Desired Role and Industry (Required)
What kind of role are you looking for, and in which industry? This will help us give you targeted advice. (Examples: "I'm looking for a data analyst role in finance or business", "I am looking for a digital marketing role at a retail company")

Additional Details (Required)
What else would you like your reviewer to know? The more detail you can give us, the better we can support you. Do you have links to specific job postings you are applying to? What are your career goals?

Udacity Honor Code

I understand that my submitted materials will be visible to the reviewer, including any personal information I choose to include.

I understand that Udacity actively checks submissions for plagiarism and that failure to adhere to the [Udacity Honor Code](#) may result in the cancellation of my enrollment.

Provide valid project submission
Specify desired role and industry
Specify additional career details
Honor code not accepted

SUBMIT

Lesson 2 : Optimize Your GitHub Profile

1. Prove Your Skills With GitHub

The Tech World Has Adopted GitHub

In 2017, [GitHub saw](#):

- 1.5 million organizations using the platform
- 24 million users across 200 countries, from entrepreneurs to open source contributors to company employees
- 25.3 million active repositories
- 47 million pull requests, which means people and organizations are working together

Software engineers, developers, and data scientists at most companies use GitHub daily to house their work, read each other's documentation, and collaborate across teams and organizations.

Recruiters, hiring managers, and senior engineers search through GitHub for potential job candidates. Your GitHub profile - activity, repos, commits, documentation - provides evidence that you're a skilled engineer.

Udacity graduate Ryan Waite was [hired](#) based on the strength of his GitHub profile.

No Resubmission Required

Unlike other Nanodegree projects, you submit to a Career Service once to get feedback and do not need to meet specifications to continue on in your program. We highly encourage you to take advantage of this guidance; most Alumni who report jobs and other opportunities (such as collaboration) attribute their success to networking.

Career Review Requirement

Your Nanodegree program was designed to ensure your long-term success in the field, and to pursue a successful career you must showcase the skills you've learned to employers effectively. Therefore, this is a requirement for graduation, unless you opt-out.

Opting Out

If improving your GitHub profile does not currently align with your Nanodegree goals, **you may opt-out of this graduation requirement**.

If you would like to opt-out, please click [here](#).

Let's Get Started

Submit your GitHub profile and receive in-depth and honest feedback from a Career Reviewer. They'll ensure you're presenting your best self to the community.



Submit Link

Project link
Check that your link is publicly accessible

Desired Role and Industry (Required)
What kind of role are you looking for, and in which industry? This will help us give you targeted advice. (Examples: "I'm looking for a data analyst role in finance or business", "I am looking for a digital marketing role at a retail company")

Additional Details (Required)
What else would you like your reviewer to know? The more detail you can give us, the better we can support you. Do you have links to specific job postings you are applying to? What are your career goals?

Udacity Honor Code

I understand that my submitted materials will be visible to the reviewer, including any personal information I choose to include.
 I understand that Udacity actively checks submissions for plagiarism and that failure to adhere to the [Udacity Honor Code](#) may result in the cancellation of my enrollment.

Provide valid project submission
Specify desired role and industry
Specify additional career details
Honor code not accepted

SUBMIT

2. Introduction

[Version Control with Git](#)

3. GitHub profile important items

<https://photos.app.goo.gl/FodbD5Vymmt7Fabc8>

4. Good GitHub repository

<https://photos.app.goo.gl/mphog1e3qPcbSW867>

5. Interview with Art - Part 1

<https://photos.app.goo.gl/e1YSzmZHM3VgcSUh6>

6. Identify fixes for example “bad” profile

<https://photos.app.goo.gl/x37KcJS4TGTohgpPA>

Search GitHub

Pull requests Issues Gist

Follow

Popular repositories

- ContentProvider
- frontend-nanodegree-arcade-g...
- frontend-nanodegree-arcade... (partial)
- PROJECT_ZERO_AppPortfolio
- Sunshine-Version-2
The official repository for Developing Android Apps

Public contributions

Summary of pull requests, issues opened, and commits. Learn how we count contributions.

Less More

| | | |
|--|--|---|
| Contributions in the last year 3 total Jan 13, 2015 – Jan 13, 2016 | Longest streak 1 day June 2 – June 2 | Current streak 0 days Last contributed 5 months ago |
|--|--|---|

Contribution activity

Period: 1 week

View Intro

VIEW ANSWER

SUBMIT ANSWER

<https://github.com/acorn47>

<https://photos.app.goo.gl/ckps1EePWodgSEsC6>

7. Quick Fixes #1

<https://photos.app.goo.gl/hqMQDebAL47mgawU8>

8. Quick Fixes #2

<https://photos.app.goo.gl/PKmLtcC5b5NPiGUW8>

[Start Quiz](#)

Here's the [checklist](#) for keeping track of improvements.



Professional GitHub Profile Checklist

General

- I included at least three projects on my GitHub account.
- My GitHub account demonstrates my knowledge of how to make incremental commits.
- My commit graph shows many green squares, at minimum for the last two weeks. (This indicates that commits have been pushed regularly).

Hiring Perspective: The first thing that a recruiter or hiring manager will look at is your commit graph and the frequency of contribution. Next, they will usually dig deeper into projects to get a sense of the kind of engineer you are. **Regular activity and major contributions indicate a passion for coding, and companies want to hire people that love to code because it correlates highly with success in the workplace.**

Tip: Try to boost your activity by building on starter code or contributing to open source. Get started with the open source projects listed [here](#).

Personal Profile

- My GitHub username is professional.
- My profile picture is a professional image.
- My profile includes at least one up-to-date link for 'URL' and/or 'Company' fields and/or 'Contact Email'.
- My profile includes my current location.

Hiring Perspective: Recruiters will use links to your personal online profiles or social media to learn about your personal interests, so be sure to showcase any relevant passions.

Recruiters and companies will also pay attention to your personal details to judge whether you're presenting yourself professionally to the world.

Tip: If you have an online persona or currently feature your "brand" on GitHub (Example: your Gravatar image is your own company's logo instead of your face), you may need to evaluate if this presentation is professional.

Projects

- My last commit message matches the [Udacity Git Commit Message Style Guide](#). If it does not, I am following another style guide.
- My projects are forked appropriately.
- My projects have meaningful names and descriptions.
- My most recent three projects have a completed README practices.

Hiring Perspective: Recruiters and companies will check whether a candidate's commit messages are properly formatted. **Companies want to know how well you're able to record for yourself and others information essential for properly maintaining code over time.**

Tip: Not all forked repos/changes require modifications to the README, especially when contributing upstream to another's project (would need to be a substantial change). All changes should be documented with commit messages to explain the changes. Technical recruiters may or may not be programmers, so they may not be able to understand code or run code. Even if they do, they may not want to invest time in seeing what your project looks like. READMEs are a way to immediately tell the recruiter how you can produce a great product or project. A poor README is a poor reflection on the effort you'd put into a work project.

Udaciousness

- I have contributed to an open source project.
- I have starred at least one repository I'd like to keep track of.

Hiring Perspective: When you contribute to open source projects, it's a valuable opportunity to interact with other engineers. **Recruiters and companies review your public activity to look for evidence of strong collaboration and teamwork.**

Tip: If you're only just beginning, try finding some documentation (even the ReadMe), cloning the project, and clarifying or updating it based on the problems you personally encounter. Next, you can submit a pull request. Once you've built up some confidence, you'll be able to move on to fixing bugs and contributing to features.



Professional GitHub Profile Checklist

General

- I included at least three projects on my GitHub account.
- My GitHub account demonstrates my knowledge of how to make incremental commits.
- My commit graph shows many green squares, at minimum for the last two weeks. (This indicates that commits have been pushed regularly).

Hiring Perspective: The first thing that a recruiter or hiring manager will look at is your commit graph and the frequency of contribution. Next, they will usually dig deeper into projects to get a sense of the kind of engineer you are. Regular activity and major contributions indicate a passion for coding, and companies want to hire people that love to code because it correlates highly with success in the workplace.

Tip: Try to boost your activity by building on starter code or contributing to open source. Get started with the open source projects listed [here](#).

Personal Profile

- My GitHub username is professional.
- My profile picture is a professional image.
- My profile includes at least one up-to-date link for 'URL' and/or 'Company' fields and/or 'Contact Email'.
- My profile includes my current location.

Hiring Perspective: Recruiters will use links to your personal online profiles or social media to learn about your personal interests, so be sure to showcase any relevant passions.

Recruiters and companies will also pay attention to your personal details to judge whether you're presenting yourself professionally to the world.

Tip: If you have an online persona or currently feature your "brand" on GitHub (Example: your Gravatar image is your own company's logo instead of your face), you may need to evaluate if this presentation is professional.

Projects

- My last commit message matches the [Udacity Git Commit Message Style Guide](#). If it does not, I am following another style guide.
- My projects are forked appropriately.
- My projects have meaningful names and descriptions.
- My most recent three projects have a completed README practices.

Hiring Perspective: Recruiters and companies will check whether a candidate's commit messages are properly formatted. Companies want to know how well you're able to record for yourself and others information essential for properly maintaining code over time.

Tip: Not all forked repos/changes require modifications to the README, especially when contributing upstream to another's project (would need to be a substantial change). All changes should be documented with commit messages to explain the changes. Technical recruiters may or may not be programmers, so they may not be able to understand code or run code. Even if they do, they may not want to invest time in seeing what your project looks like. READMEs are a way to immediately tell the recruiter how you can produce a great product or project. A poor README is a poor reflection on the effort you'd put into a work project.

Udaciousness

- I have contributed to an open source project.
- I have starred at least one repository I'd like to keep track of.

Hiring Perspective: When you contribute to open source projects, it's a valuable opportunity to interact with other engineers. Recruiters and companies review your public activity to look for evidence of strong collaboration and teamwork.

Tip: If you're only just beginning, try finding some documentation (even the ReadMe), cloning the project, and clarifying or updating it based on the problems you personally encounter. Next, you can submit a pull request. Once you've built up some confidence, you'll be able to move on to fixing bugs and contributing to features.

9. Writing READMEs with Walter

<https://photos.app.goo.gl/wuZBuX3fcEvP1fFA>

10. Interview with Art - Part 2

<https://photos.app.goo.gl/9J8FYqNvzxpHj9hD9>

11. Commit messages best practices

As Art said, commit messages are essential for communicating why your code was changed. This is for your coworkers or collaborators, and also for your future self. Let's go through some best practices for writing commit messages.

Here is the message format we use here at Udacity. You can find it [here](#) for future reference.

```
type: subject
```

```
body
```

```
footer
```

The first line is the subject. This should be a short description of what changed. Nothing like “fixed it” or “did something,” these need to be clear and informative, and try to avoid profanity. The subject should be 50 characters or less, with the first letter capitalized, and end without a period. At Udacity, we also include a short annotation about the type of the commit, if it is a bug fix, a feature, change to the documentation, etc.

The body is next, this is where you give a more detailed description of why you made the change. The body should typically have around 72 characters per line. This is to ensure that the message fits into a terminal window when using git on the command line. You'll also need to make sure there is a blank line between the subject line and the body. You can also add bullet points, using asterisks or dashes, when you need to make a list.

Some commits don't require a body in the message. If you fix a typo for example, it's okay to only have a subject line.

You can also include a footer, typically this will be used to indicate which issues or bugs the commit addresses.

A more fleshed out example looks like this:

```
feat: Summarize changes in around 50 characters or less

More detailed explanatory text, if necessary. Wrap it to about 72
characters or so. In some contexts, the first line is treated as the
subject of the commit and the rest of the text as the body. The
blank line separating the summary from the body is critical (unless
you omit the body entirely); various tools like `log`, `shortlog`
and `rebase` can get confused if you run the two together.
```

```
Explain the problem that this commit is solving. Focus on why you
are making this change as opposed to how (the code explains that).
Are there side effects or other unintuitive consequences of this
change? Here's the place to explain them.
```

Further paragraphs come after blank lines.

- Bullet points are okay, too
- Typically a hyphen or asterisk is used for the bullet, preceded
 by a single space, with blank lines in between, but conventions
 vary here

```
If you use an issue tracker, put references to them at the bottom,
like this:
```

Resolves: #123

See also: #456, #789

This does come with an exception of course. If you are working on an open source project, be sure to follow the message format for that project. This will make the maintainers happy and increase the chance your pull request is accepted.

12. Reflect on your commit messages

<https://photos.app.goo.gl/qNqwqnpj5y9oA5eZA>

13. Participating in open source projects

<https://photos.app.goo.gl/AZfAEJS6PxpTeWs5A>

[Get Started With Open Source Projects](#)

14. Interview with Art - Part 3

<https://photos.app.goo.gl/BT2eR1rYgC5292QdA>

15. Participating in open source projects 2

<https://photos.app.goo.gl/kRDPYfBRYdhMi9Nu6>

16. Starring interesting repositories

<https://photos.app.goo.gl/Gue3seoGpecMkBsp9>

<https://photos.app.goo.gl/qZMLvcLtSfgK7VeBA>

17. Next Steps



Apply these best GitHub practices moving forward and attract the attention of recruiters and senior software engineers.

You'll want to:

- Commit every week, at minimum
- Get 2-4 weeks streaks

A personal side-project may grow into something bigger. You can't always predict what will catch the eye of a recruiter or senior software engineer, but when it does, you want to showcase your best work.

Continue building your knowledge of using GitHub and improving your profile, be sure to check out the "GitHub Professional Profile Resources" found in the Classroom folder under Lesson Resources.

Once you've implemented the necessary changes to make your profile more professional, remember to maintain activity by regularly making commits on GitHub. Good luck!

18. Project: Optimize Your GitHub Profile

The screenshot shows a project submission interface. At the top, there's a dark header bar with the title 'Optimize Your GitHub Profile' on the left and a 'SUBMIT PROJECT' button on the right. Below the header, the main area has a light gray background. It features a large, dark blue speech bubble-like callout on the right side containing text about asking mentors. On the left, there's a card with project details: 'DUE DATE' (Feb 17), 'STATUS' (Unsubmitted, with a note 'Project past due'), and a button labeled 'ASK A MENTOR'. The overall design is clean and modern.

GitHub Profiles are a key piece of "evidence" to an employer that you'd be a good job candidate, because they can see the details of your work. Recruiters use GitHub as a way to find job candidates, and many Nanodegree alumni have received work opportunities from their activity on GitHub. In addition, using GitHub is a way for you to collaborate on projects with other programmers - this will show that you are able to work well with others on an engineering team on the job.

Submission Instructions

1. Submit this project after you've done some continuous coding for two weeks, so the reviewer will be able to comment on trends.
2. Submit your GitHub profile URL as your project.
3. If you'd like to **opt-out** of this project, please read the instructions underneath the divider line, below.

Submission Instructions

1. Submit this project after you've done some continuous coding for two weeks, so the reviewer will be able to comment on trends.
2. Submit your GitHub profile URL as your project.
3. If you'd like to **opt-out** of this project, please read the instructions underneath the divider line, below.

Project Resources

1. [Project Rubric](#). *Your project will be reviewed by a Udacity Career Reviewer against this rubric.*
2. [Project checklist](#). *Based on the project rubric, this is a handy checklist on GitHub best practices.*
3. [Career Resource Center](#). *Find additional tips and guides on developing your GitHub Profile.*

Up Next

Continue developing on GitHub! For project ideas, consider [contributing to open source](#).

Career Review Requirement

Your Nanodegree program was designed to ensure your long-term success in the field, and to pursue a successful career you must showcase the skills you've learned to employers effectively. Therefore, this is a requirement for graduation, unless you opt-out.

Opting Out

If improving your GitHub profile does not align with your Nanodegree goals, **you may opt-out of this graduation requirement**.

When you opt-out:

- *You will no longer be able to submit the GitHub Review from this Nanodegree program and all other Nanodegree programs.*
- *Your decision to opt-out is final and future access to GitHub Review will need to be purchased separately.*

If you would like to opt-out, please click [here](#).

Project link

Check that your link is publicly accessible

Desired Role and Industry (Required)

What kind of role are you looking for, and in which industry? This will help us give you targeted advice. (Examples: "I'm looking for a data analyst role in finance or business", "I am looking for a digital marketing role at a retail company")

Additional Details (Required)

What else would you like your reviewer to know? The more detail you can give us, the better we can support you. Do you have links to specific job postings you are applying to? What are your career goals?

Udacity Honor Code

- I understand that my submitted materials will be visible to the reviewer, including any personal information I choose to include.
- I understand that Udacity actively checks submissions for plagiarism and that failure to adhere to the [Udacity Honor Code](#) may result in the cancellation of my enrollment.

Provide valid project submission
Specify desired role and industry
Specify additional career details
Honor code not accepted

SUBMIT



Microsoft Azure
Noel Ching

422

Machine Learning Engineer for Microsoft Azure