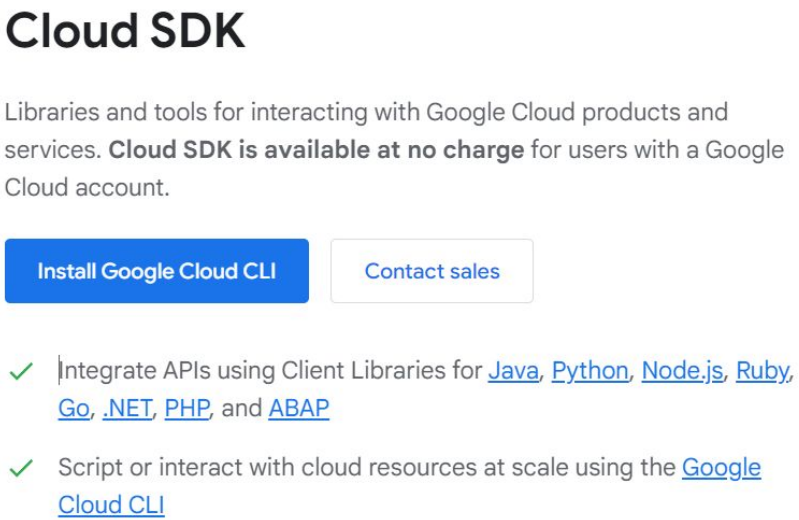
Week 2, Cloud Application Development

Azamat Serek, PhD, Assist.Prof.

Google Cloud SDK

https://cloud.google.com/sdk?gad_source=1&gclid=CjwKCAjwoJa2BhBPEiwA0l0ImIuk-5UWi03vGHQ8sfXTgi-h_gbmZ1xp4GbxWu8V1KsOGY2u8BPtWhoCs2QQAvD_BwE&gclsrc=aw.ds&hl=ru



Python Cloud Client Libraries

https://cloud.google.com/python/docs/reference?_gl=1*12gcqw1*_up*MQ..&gclid=CjwKCAjwoJa2BhBPEiwA0l0ImIuk-5UWi03vGHQ8sfXTgi-h_gbmZ1xp4GbxWu8V1KsOGY2u8BPtWhoCs2QQAvD_BwE&gclsrc=aw.ds

The Cloud Client Libraries are the recommended way to access Google Cloud APIs programmatically. The Cloud Client Libraries support accessing Google Cloud services in a way that significantly reduces the boilerplate code you have to write.

Vertex AI

Vertex AI: Google Vertex AI is an integrated suite of machine learning tools and services for building and using ML models with AutoML or custom code. It offers both novices and experts the best workbench for the entire machine learning development lifecycle.

https://cloud.google.com/python/docs/reference/aiplatform/latest?_gl=1*r36exu*_up*MQ..&gclid=CjwKCAjwoJa2BhBPEiwA0l0ImIuk-5UWi03vGHQ8sfXTgi-h_gbmZ1xp4GbxWu8V1KsOGY2u8BPtWhoCs2QQAvD_BwE&gclsrc=aw.ds

Installation

Install this library in a virtualenv using pip. virtualenv is a tool to create isolated Python environments. The basic problem it addresses is one of dependencies and versions, and indirectly permissions.

With virtualenv, it's possible to install this library without needing system install permissions, and without clashing with the installed system dependencies.

Mac/Linux

```
pip install virtualenv
virtualenv <your-env>
source <your-env>/bin/activate
<your-env>/bin/pip install google-cloud-aiplatform
```

Windows

```
pip install virtualenv
virtualenv <your-env>
<your-env>\Scripts\activate
<your-env>\Scripts\pip.exe install google-cloud-aiplatform
```

Supported Python Versions

Python >= 3.8

Deprecated Python Versions

Python <= 3.7.

Overview

All publicly available SDK features can be found in the `google/cloud/aiplatform` directory. Under the hood, Vertex SDK builds on top of GAPIC, which stands for Google API CodeGen. The GAPIC library code sits in `google/cloud/aiplatform_v1` and `google/cloud/aiplatform_v1beta1`, and it is auto-generated from Google's service proto files.

For most developers' programmatic needs, they can follow these steps to figure out which libraries to import:

1. Look through `google/cloud/aiplatform` first – Vertex SDK's APIs will almost always be easier to use and more concise comparing with GAPIC
2. If the feature that you are looking for cannot be found there, look through `aiplatform_v1` to see if it's available in GAPIC
3. If it is still in beta phase, it will be available in `aiplatform_v1beta1`

Importing

Vertex AI SDK resource based functionality can be used by importing the following namespace:

```
from google.cloud import aiplatform
```

Vertex AI SDK preview functionality can be used by importing the following namespace:

```
from vertexai import preview
```

Vertex AI SDK general availability (GA) functionality can be used by importing the following namespace:

```
import vertexai
```

```
aiplatform.init(
    # your Google Cloud Project ID or number
    # environment default used is not set
    project='my-project',

    # the Vertex AI region you will use
    # defaults to us-central1
    location='us-central1',

    # Google Cloud Storage bucket in same region as location
    # used to stage artifacts
    staging_bucket='gs://my_staging_bucket',

    # custom google.auth.credentials.Credentials
    # environment default credentials used if not set
    credentials=my_credentials,

    # customer managed encryption key resource name
    # will be applied to all Vertex AI resources if set
    encryption_spec_key_name=my_encryption_key_name,

    # the name of the experiment to use to track
    # logged metrics and parameters
    experiment='my-experiment',

    # description of the experiment above
    experiment_description='my experiment description'
)
```

Datasets

Vertex AI provides managed tabular, text, image, and video datasets. In the SDK, datasets can be used downstream to train models.

To create a tabular dataset:

```
my_dataset = aiplatform.TabularDataset.create(
    display_name="my-dataset", gcs_source=['gs://path/to/my/dataset.csv'])
```

You can also create and import a dataset in separate steps:

```
from google.cloud import aiplatform

my_dataset = aiplatform.TextDataset.create(
    display_name="my-dataset")

my_dataset.import(
    gcs_source=['gs://path/to/my/dataset.csv']
    import_schema_uri=aiplatform.schema.dataset.ioformat.text.multi_label_classification
)
```

Training

The Vertex AI SDK for Python allows you train Custom and AutoML Models.

You can train custom models using a custom Python script, custom Python package, or container.

Preparing Your Custom Code

Vertex AI custom training enables you to train on Vertex AI datasets and produce Vertex AI models. To do so your script must adhere to the following contract:

It must read datasets from the environment variables populated by the training service:

```
os.environ['AIP_DATA_FORMAT']  # provides format of data
os.environ['AIP_TRAINING_DATA_URI']  # uri to training split
os.environ['AIP_VALIDATION_DATA_URI']  # uri to validation split
os.environ['AIP_TEST_DATA_URI']  # uri to test split
```

Running Training

```
job = aiplatform.CustomTrainingJob(
    display_name="my-training-job",
    script_path="training_script.py",
    container_uri="us-docker.pkg.dev/vertex-ai/training/tf-cpu.2-2:latest",
    requirements=["gcsfs==0.7.1"],
    model_serving_container_image_uri="us-docker.pkg.dev/vertex-ai/prediction/tf2-cpu.2-2:latest",

)

model = job.run(my_dataset,
                replica_count=1,
                machine_type="n1-standard-4",
                accelerator_type='NVIDIA_TESLA_K80',
                accelerator_count=1)
```

AutoML

```
dataset = aiplatform.TabularDataset('projects/my-project/location/us-central1/datasets/{DATASET_ID

job = aiplatform.AutoMLTabularTrainingJob(
  display_name="train-automl",
  optimization_prediction_type="regression",
  optimization_objective="minimize-rmse",
)

model = job.run(
    dataset=dataset,
    target_column="target_column_name",
    training_fraction_split=0.6,
    validation_fraction_split=0.2,
    test_fraction_split=0.2,
    budget_milli_node_hours=1000,
    model_display_name="my-automl-model",
    disable_early_stopping=False,
)
```

Models

To get a model:

```
model = aiplatform.Model('/projects/my-project/locations/us-central1/models/{MODEL_ID}')
```

To upload a model:

```
model = aiplatform.Model.upload(
    display_name='my-model',
    artifact_uri="gs://python/to/my/model/dir",
    serving_container_image_uri="us-docker.pkg.dev/vertex-ai/prediction/tf2-cpu.2-2:latest",
)
```

Deploying model

To deploy a model:

```
endpoint = model.deploy(machine_type="n1-standard-4",
                        min_replica_count=1,
                        max_replica_count=5
                        machine_type='n1-standard-4',
                        accelerator_type='NVIDIA_TESLA_K80',
                        accelerator_count=1)
```

Model evaluation

To list all model evaluations for a model:

```
model = aiplatform.Model('projects/my-project/locations/us-central1/models/{MODEL_ID}')

evaluations = model.list_model_evaluations()
```

To get the model evaluation resource for a given model:

```
model = aiplatform.Model('projects/my-project/locations/us-central1/models/{MODEL_ID}')

# returns the first evaluation with no arguments, you can also pass the evaluation ID
evaluation = model.get_model_evaluation()

eval_metrics = evaluation.metrics
```

Batch Prediction

To create a batch prediction job:

```
model = aiplatform.Model('/projects/my-project/locations/us-central1/models/{MODEL_ID}')

batch_prediction_job = model.batch_predict(
  job_display_name='my-batch-prediction-job',
  instances_format='csv',
  machine_type='n1-standard-4',
  gcs_source=['gs://path/to/my/file.csv'],
  gcs_destination_prefix='gs://path/to/my/batch_prediction/results/',
  service_account='my-sa@my-project.iam.gserviceaccount.com'
)
```

Exercise (will not be given grade)

Go through this
https://cloud.google.com/python/docs/reference/aiplatform/latest?_gl=1*r36exu*_up*MQ..&gclid=CjwKCAjwoJa2BhBPEiwA0l0ImIuk-5UWi03vGHQ8sfXTgi-h_gbmZ1xp4GbxWu8V1KsOGY2u8BPtWhoCs2QQAvD_BwE&gclsrc=aw.ds

Execute and write report