Kubeflow Fairing

Release v0.7.0

Nov 08, 2019

Contents:

Main documentation: https://www.kubeflow.org/docs/fairing/

Source code: https://github.com/kubeflow/fairing/

CHAPTER 1

kubeflow.fairing package

1.1 Subpackages

1.1.1 kubeflow.fairing.backends package

Submodules

kubeflow.fairing.backends.backends module

class kubeflow.fairing.backends.backends.**AWSBackend**(*namespace=None*, *build_context_source=None*)

Bases: *kubeflow.fairing.backends.backends.KubernetesBackend*

Use to create a builder instance and create a deployer to be used with a traing job or a serving job for the AWS backend.

get_builder(*preprocessor*, *base_image*, *registry*, *needs_deps_installation=True*, *pod_spec_mutators=None*)

Creates a builder instance with right config for AWS

Parameters

- **preprocessor** – Preprocessor to use to modify inputs
- **base_image** – Base image to use for this job
- **registry** – Registry to push image to. Example: gcr.io/kubeflow-images
- **needs_deps_installation** – need depends on installation(Default value = True)
- **pod_spec_mutators** – list of functions that is used to mutate the podsspec. e.g. fairing.cloud.gcp.add_gcp_credentials_if_exists This can used to set things like volumes and security context. (Default value =None)

get_serving_deployer(*model_class*, *service_type='ClusterIP'*, *pod_spec_mutators=None*)

Creates a deployer to be used with a serving job for AWS

Parameters

- **model_class** – the name of the class that holds the predict function.
- **service_type** – service type (Default value = 'ClusterIP')
- **pod_spec_mutators** – list of functions that is used to mutate the podsspec. (Default value = None)

get_training_deployer(*pod_spec_mutators=None*)
Creates a deployer to be used with a training job for AWS

Parameters **pod_spec_mutators** – list of functions that is used to mutate the podsspec. (Default value = None)

Returns job for handle all the k8s' template building for a training

class kubeflow.fairing.backends.backends.**AzureBackend**(*namespace=None*, *build_context_source=None*)
Bases: *kubeflow.fairing.backends.backends.KubernetesBackend*

Use to create a builder instance and create a deployer to be used with a traing job or a serving job for the Azure backend.

get_builder(*preprocessor*, *base_image*, *registry*, *needs_deps_installation=True*, *pod_spec_mutators=None*)
Creates a builder instance with right config for Azure

Parameters

- **preprocessor** – Preprocessor to use to modify inputs
- **base_image** – Base image to use for this job
- **registry** – Registry to push image to. Example: gcr.io/kubeflow-images
- **needs_deps_installation** – need depends on installation(Default value = True)
- **pod_spec_mutators** – list of functions that is used to mutate the podsspec. e.g. fairing.cloud.gcp.add_gcp_credentials_if_exists This can used to set things like volumes and security context. (Default value =None)

class kubeflow.fairing.backends.backends.**BackendInterface**
Bases: object

Backend interface. Creating a builder instance or a deployer to be used with a traing job or a serving job for the given backend. And get the approriate base container or docker registry for the current environment.

get_base_contanier()
Returns the approriate base container for the current environment

Returns base image

get_builder(*preprocessor*, *base_image*, *registry*)
Creates a builder instance with right config for the given backend

Parameters

- **preprocessor** – Preprocessor to use to modify inputs
- **base_image** – Base image to use for this builder
- **registry** – Registry to push image to. Example: gcr.io/kubeflow-images

Raises **NotImplementedError** – not implemented exception

get_docker_registry()
Returns the approriate docker registry for the current environment

Returns None

get_serving_deployer(*model_class*)
Creates a deployer to be used with a serving job

Parameters **model_class** – the name of the class that holds the predict function.

Raises **NotImplementedError** – not implemented exception

get_training_deployer(*pod_spec_mutators=None*)
Creates a deployer to be used with a training job

Parameters **pod_spec_mutators** – list of functions that is used to mutate the podsspec. e.g. fairing.cloud.gcp.add_gcp_credentials_if_exists This can used to set things like volumes and security context. (Default value = None)

Raises **NotImplementedError** – not implemented exception

class kubeflow.fairing.backends.backends.**GCPManagedBackend**(*project_id=None*, *region=None*, *training_scale_tier=None*)
Bases: *kubeflow.fairing.backends.backends.BackendInterface*

Use to create a builder instance and create a deployer to be used with a traing job or a serving job for the GCP.

get_builder(*preprocessor*, *base_image*, *registry*, *needs_deps_installation=True*, *pod_spec_mutators=None*)
Creates a builder instance with right config for GCP

Parameters

- **preprocessor** – Preprocessor to use to modify inputs
- **base_image** – Base image to use for this job
- **registry** – Registry to push image to. Example: gcr.io/kubeflow-images
- **needs_deps_installation** – need depends on installation(Default value = True)
- **pod_spec_mutators** – list of functions that is used to mutate the podsspec. e.g. fairing.cloud.gcp.add_gcp_credentials_if_exists This can used to set things like volumes and security context. (Default value =None)

get_docker_registry()
Returns the approriate docker registry for GCP

Returns docker registry

get_serving_deployer(*model_class*, *pod_spec_mutators=None*)
Creates a deployer to be used with a serving job for GCP

Parameters

- **model_class** – the name of the class that holds the predict function.
- **service_type** – service type (Default value = 'ClusterIP')
- **pod_spec_mutators** – list of functions that is used to mutate the podsspec. (Default value = None)

get_training_deployer(*pod_spec_mutators=None*)
Creates a deployer to be used with a training job for GCP

Parameters **pod_spec_mutators** – list of functions that is used to mutate the podsspec. (Default value = None)

Returns job for handle all the k8s' template building for a training

class kubeflow.fairing.backends.backends.**GKEBackend**(*namespace=None*, *build_context_source=None*)

Bases: *kubeflow.fairing.backends.backends.KubernetesBackend*

Use to create a builder instance and create a deployer to be used with a traing job or a serving job for the GKE backend. And get the approriate docker registry for GKE.

get_builder(*preprocessor*, *base_image*, *registry*, *needs_deps_installation=True*, *pod_spec_mutators=None*)

Creates a builder instance with right config for GKE

Parameters

- **preprocessor** – Preprocessor to use to modify inputs
- **base_image** – Base image to use for this job
- **registry** – Registry to push image to. Example: gcr.io/kubeflow-images
- **needs_deps_installation** – need depends on installation(Default value = True)
- **pod_spec_mutators** – list of functions that is used to mutate the podsspec. e.g. fairing.cloud.gcp.add_gcp_credentials_if_exists This can used to set things like volumes and security context. (Default value =None)

get_docker_registry()

Returns the approriate docker registry for GKE

Returns docker registry

get_serving_deployer(*model_class*, *service_type='ClusterIP'*, *pod_spec_mutators=None*)

Creates a deployer to be used with a serving job for GKE

Parameters

- **model_class** – the name of the class that holds the predict function.
- **service_type** – service type (Default value = 'ClusterIP')
- **pod_spec_mutators** – list of functions that is used to mutate the podsspec. (Default value = None)

get_training_deployer(*pod_spec_mutators=None*)

Creates a deployer to be used with a training job for GKE

Parameters **pod_spec_mutators** – list of functions that is used to mutate the podsspec. (Default value = None)

Returns job for handle all the k8s' template building for a training

class kubeflow.fairing.backends.backends.**KubeflowAWSBackend**(*namespace=None*, *build_context_source=None*)

Bases: *kubeflow.fairing.backends.backends.AWSBackend*

Kubeflow for AWS backend refer to AWSBackend

class kubeflow.fairing.backends.backends.**KubeflowAzureBackend**(*namespace=None*, *build_context_source=None*)

Bases: *kubeflow.fairing.backends.backends.AzureBackend*

Kubeflow for Azure backend refer to AzureBackend

class kubeflow.fairing.backends.backends.**KubeflowBackend**(*namespace=None*, *build_context_source=None*)

Bases: *kubeflow.fairing.backends.backends.KubernetesBackend*

Kubeflow backend refer to KubernetesBackend

class kubeflow.fairing.backends.backends.**KubeflowGKEBackend**(*namespace=None*, *build_context_source=None*)

Bases: *kubeflow.fairing.backends.backends.GKEBackend*

Kubeflow for GKE backend refer to GKEBackend

class kubeflow.fairing.backends.backends.**KubernetesBackend**(*namespace=None*, *build_context_source=None*)

Bases: *kubeflow.fairing.backends.backends.BackendInterface*

Use to create a builder instance and create a deployer to be used with a traing job or a serving job for the Kubernetes.

get_builder(*preprocessor*, *base_image*, *registry*, *needs_deps_installation=True*, *pod_spec_mutators=None*)

Creates a builder instance with right config for the given Kubernetes

Parameters

- **preprocessor** – Preprocessor to use to modify inputs
- **base_image** – Base image to use for this job
- **registry** – Registry to push image to. Example: gcr.io/kubeflow-images
- **needs_deps_installation** – need depends on installation(Default value = True)
- **pod_spec_mutators** – list of functions that is used to mutate the podsspec. e.g. fairing.cloud.gcp.add_gcp_credentials_if_exists This can used to set things like volumes and security context. (Default value =None)

get_serving_deployer(*model_class*, *service_type='ClusterIP'*, *pod_spec_mutators=None*)

Creates a deployer to be used with a serving job for the Kubernetes

Parameters

- **model_class** – the name of the class that holds the predict function.
- **service_type** – service type (Default value = 'ClusterIP')
- **pod_spec_mutators** – list of functions that is used to mutate the podsspec. (Default value = None)

get_training_deployer(*pod_spec_mutators=None*)

Creates a deployer to be used with a training job for the Kubernetes

Parameters **pod_spec_mutators** – list of functions that is used to mutate the podsspec. (Default value = None)

Returns job for handle all the k8s' template building for a training

Module contents

1.1.2 kubeflow.fairing.builders package

Subpackages

kubeflow.fairing.builders.append package

Submodules

kubeflow.fairing.builders.append.append module

class kubeflow.fairing.builders.append.append.**AppendBuilder**(*registry=None*, *image_name='fairing-job'*, *base_image='gcr.io/kubeflow-images-public/fairing:dev'*, *push=True*, *preprocessor=None*)

Bases: *kubeflow.fairing.builders.base_builder.BaseBuilder*

Builds a docker image by appending a new layer tarball to an existing base image. Does not require docker and runs in userspace.

Parameters

- **base_image** – Base image to use for the build (default: {constants.DEFAULT_BASE_IMAGE})
- **image_name** – image name to use for the new image(default: {constants.DEFAULT_IMAGE_NAME})
- **preprocessor** – Preprocessor{BasePreProcessor} to use to modify inputs before sending them to docker build
- **push** – Whether or not to push the image to the registry

build()

Will be called when the build needs to start

timed_push(*transport*, *src*, *img*, *dst*)

Push image to the registry and log the time spent to the log

Parameters

- **transport** – the http transport to use for sending requests
- **src** – repo from which to mount blobs
- **img** – the image to be pushed
- **dst** – the fully-qualified name of the tag to push

Module contents

kubeflow.fairing.builders.cluster package

Submodules

kubeflow.fairing.builders.cluster.azurestorage_context module

class kubeflow.fairing.builders.cluster.azurestorage_context.**StorageContextSource**(*namespace=None*, *region=None*, *resource_group=None*, *storage_account=None*)

Bases: *kubeflow.fairing.builders.cluster.context_source.ContextSourceInterface*

Azure storage context source

cleanup()
Cleans up the context after the build is complete

generate_pod_spec(*image_name*, *push*)
Generates a pod spec for building the image in the cluster, pointing to the prepared build context

Parameters **pod_spec** – pod spec

prepare(*context_filename*)
Makes the context somehow available for use in the pod spec

upload_context(*context_filename*)

kubeflow.fairing.builders.cluster.cluster module

class kubeflow.fairing.builders.cluster.cluster.**ClusterBuilder**(*registry=None*, *image_name='fairing-job'*, *context_source=None*, *preprocessor=None*, *push=True*, *base_image='gcr.io/kubeflow-images-public/fairing:dev'*, *pod_spec_mutators=None*, *namespace=None*, *docker_file_path=None*, *cleanup=False*)

Bases: *kubeflow.fairing.builders.base_builder.BaseBuilder*

Builds a docker image in a Kubernetes cluster.

build()
Runs the build

kubeflow.fairing.builders.cluster.context_source module

class kubeflow.fairing.builders.cluster.context_source.**ContextSourceInterface**
Bases: object

Interface that provides the build context to the in cluster builder

cleanup()
Cleans up the context after the build is complete

generate_pod_spec(*pod_spec*)
Generates a pod spec for building the image in the cluster, pointing to the prepared build context

Parameters **pod_spec** – pod spec

prepare()
Makes the context somehow available for use in the pod spec

kubeflow.fairing.builders.cluster.gcs_context module

class kubeflow.fairing.builders.cluster.gcs_context.**GCSContextSource**(*gcp_project=None*, *credentials_file=None*, *namespace='default'*)

Bases: *kubeflow.fairing.builders.cluster.context_source.ContextSourceInterface*

Google cloud storage context for docker builder

cleanup()
Cleans up the context after the build is complete

generate_pod_spec(*image_name*, *push*)
Generates a pod spec for building the image in the cluster, pointing to the prepared build context

Parameters **pod_spec** – pod spec

prepare(*context_filename*)
Makes the context somehow available for use in the pod spec

upload_context(*context_filename*)

kubeflow.fairing.builders.cluster.s3_context module

class kubeflow.fairing.builders.cluster.s3_context.**S3ContextSource**(*aws_account=None*, *region=None*, *bucket_name=None*)

Bases: *kubeflow.fairing.builders.cluster.context_source.ContextSourceInterface*

aws S3 context for docker builder

cleanup()
Cleans up the context after the build is complete

generate_pod_spec(*image_name*, *push*)
Generates a pod spec for building the image in the cluster, pointing to the prepared build context

Parameters **pod_spec** – pod spec

prepare(*context_filename*)

Parameters **context_filename** – context filename

upload_context(*context_filename*)

Parameters **context_filename** – context filename

Module contents

kubeflow.fairing.builders.docker package

Submodules

kubeflow.fairing.builders.docker.docker module

class kubeflow.fairing.builders.docker.docker.**DockerBuilder**(*registry=None*, *image_name='fairing-job'*, *base_image='gcr.io/kubeflow-images-public/fairing:dev'*, *preprocessor=None*, *push=True*, *dockerfile_path=None*)

Bases: *kubeflow.fairing.builders.base_builder.BaseBuilder*

A builder using the local Docker client

build()
Runs the build

publish()
push the docker image to the docker registry

Module contents

Submodules

kubeflow.fairing.builders.base_builder module

class kubeflow.fairing.builders.base_builder.**BaseBuilder**(*registry=None*, *image_name=None*, *base_image='gcr.io/kubeflow-images-public/fairing:dev'*, *push=True*, *preprocessor=None*, *dockerfile_path=None*)

Bases: *kubeflow.fairing.builders.builder.BuilderInterface*

A builder using the local Docker client

build()
Runs the build

full_image_name(*tag*)
Retrun the full image name

Parameters **tag** – the new tag for the image

generate_pod_spec()
This method should return a V1PodSpec with the correct image set. This is also where the builder should set the environment variables and volume/volumeMounts that it may need to work

kubeflow.fairing.builders.builder module

class kubeflow.fairing.builders.builder.**BuilderInterface**
Bases: object

build()
Will be called when the build needs to start

generate_pod_spec()
This method should return a V1PodSpec with the correct image set. This is also where the builder should set the environment variables and volume/volumeMounts that it may need to work

kubeflow.fairing.builders.dockerfile module

kubeflow.fairing.builders.dockerfile.**write_dockerfile**(*docker_command=None*, *destination=None*, *path_prefix='/app/'*, *base_image=None*, *install_reqs_before_copy=False*)
Generate dockerfile accoding to the parameters

Parameters

- **docker_command** – string, CMD of the dockerfile (Default value = None)
- **destination** – string, destination folder for this dockerfile (Default value = None)
- **path_prefix** – string, WORKDIR (Default value = constants.DEFAULT_DEST_PREFIX)
- **base_image** – string, base image, example: gcr.io/kubeflow-image
- **install_reqs_before_copy** – whether to install the prerequisites (Default value = False)

Module contents

1.1.3 kubeflow.fairing.cloud package

Submodules

kubeflow.fairing.cloud.aws module

class kubeflow.fairing.cloud.aws.**S3Uploader**(*region*)
Bases: object

For AWS S3 up load

create_bucket_if_not_exists(*bucket_name*)
Create bucket if this bucket not exists

Parameters **bucket_name** – Bucket name

upload_to_bucket(*blob_name*, *bucket_name*, *file_to_upload*)
Upload a file to an S3 bucket

Parameters

- **blob_name** – S3 object name
- **bucket_name** – Bucket to upload to
- **file_to_upload** – File to upload

kubeflow.fairing.cloud.aws.**add_aws_credentials**(*kube_manager*, *pod_spec*, *namespace*)
add AWS credential

Parameters

- **kube_manager** – kube manager for handles communication with Kubernetes' client
- **pod_spec** – pod spec like volumes and security context
- **namespace** – The custom resource

kubeflow.fairing.cloud.aws.**add_aws_credentials_if_exists**(*kube_manager*, *pod_spec*, *namespace*)
add AWS credential

Parameters

- **kube_manager** – kube manager for handles communication with Kubernetes' client
- **pod_spec** – pod spec like volumes and security context
- **namespace** – The custom resource

kubeflow.fairing.cloud.aws.**add_ecr_config**(*kube_manager*, *pod_spec*, *namespace*)
add secret

Parameters

- **kube_manager** – kube manager for handles communication with Kubernetes' client
- **pod_spec** – pod spec like volumes and security context
- **namespace** – The custom resource

kubeflow.fairing.cloud.aws.**create_ecr_registry**(*registry*, *repository*)
create secret registry

Parameters

- **registry** – registry
- **repository** – repository name

kubeflow.fairing.cloud.aws.**guess_account_id**()
Get account id

kubeflow.fairing.cloud.aws.**is_ecr_registry**(*registry*)
verify secrte registy

Parameters **registry** – registry

kubeflow.fairing.cloud.azure module

class kubeflow.fairing.cloud.azure.**AzureFileUploader**(*namespace*, *credentials=None*, *subscription_id=None*)

Bases: object

create_share_if_not_exists(*share_service*, *share_name*)

create_storage_account_if_not_exists(*region*, *resource_group_name*, *storage_account_name*)

Creates the storage account if it does not exist.

In either case, returns the StorageAccount class that matches the given arguments.

delete_uncompressed_files(*target_dir*)

get_storage_credentials(*resource_group_name*, *storage_account_name*)

uncompress_tar_gz_file(*tar_gz_file*, *target_dir*)

upload_tar_gz_contents(*share_service*, *share_name*, *dir_name*, *tar_gz_file*)

upload_to_share(*region*, *resource_group_name*, *storage_account_name*, *share_name*, *dir_name*, *tar_gz_file_to_upload*)

kubeflow.fairing.cloud.azure.**add_acr_config**(*kube_manager*, *pod_spec*, *namespace*)

kubeflow.fairing.cloud.azure.**add_azure_files**(*kube_manager*, *pod_spec*, *namespace*)

kubeflow.fairing.cloud.azure.**create_storage_creds_secret**(*namespace*, *context_hash*, *storage_account_name*, *storage_key*)

kubeflow.fairing.cloud.azure.**delete_storage_creds_secret**(*namespace*, *context_hash*)

kubeflow.fairing.cloud.azure.**get_azure_credentials**(*namespace*)

kubeflow.fairing.cloud.azure.**get_plain_secret_value**(*secret_data*, *key*)

kubeflow.fairing.cloud.azure.**is_acr_registry**(*registry*)

kubeflow.fairing.cloud.docker module

kubeflow.fairing.cloud.docker.**add_docker_credentials**(*kube_manager*, *pod_spec*, *namespace*)

kubeflow.fairing.cloud.docker.**add_docker_credentials_if_exists**(*kube_manager*, *pod_spec*, *namespace*)

kubeflow.fairing.cloud.docker.**create_docker_secret**(*kube_manager*, *namespace*)

kubeflow.fairing.cloud.docker.**get_docker_secret**()

kubeflow.fairing.cloud.gcp module

class kubeflow.fairing.cloud.gcp.**GCSUploader**(*credentials_file=None*)

Bases: object

get_or_create_bucket(*bucket_name*)

upload_to_bucket(*blob_name*, *bucket_name*, *file_to_upload*)

kubeflow.fairing.cloud.gcp.**add_gcp_credentials**(*kube_manager*, *pod_spec*, *namespace*)

Note: This method will be deprecated soon and will become unavailable by 1.0. All future access will use Workload Identity.

kubeflow.fairing.cloud.gcp.**add_gcp_credentials_if_exists**(*kube_manager*, *pod_spec*, *namespace*)

kubeflow.fairing.cloud.gcp.**get_default_docker_registry**()

kubeflow.fairing.cloud.gcp.**guess_project_name**(*credentials_file=None*)

kubeflow.fairing.cloud.storage module

class kubeflow.fairing.cloud.storage.**GCSStorage**
Bases: *kubeflow.fairing.cloud.storage.Storage*

copy_cmd(*src_url*, *dst_url*, *recursive=True*)
gets a command to copy files from/to remote storage from/to local FS

exists(*url*)
checks if the url exists in the given storage

class kubeflow.fairing.cloud.storage.**Storage**
Bases: object

copy_cmd(*src_url*, *dst_url*, *recursive=True*)
gets a command to copy files from/to remote storage from/to local FS

exists(*url*)
checks if the url exists in the given storage

kubeflow.fairing.cloud.storage.**get_storage_class**(*url*)

kubeflow.fairing.cloud.storage.**lookup_storage_class**(*url*)

Module contents

1.1.4 kubeflow.fairing.constants package

Submodules

kubeflow.fairing.constants.constants module

Module contents

1.1.5 kubeflow.fairing.deployers package

Subpackages

kubeflow.fairing.deployers.gcp package

Submodules

kubeflow.fairing.deployers.gcp.gcp module

class kubeflow.fairing.deployers.gcp.gcp.**GCPJob**(*project_id=None*, *region=None*, *scale_tier=None*, *job_config=None*)
Bases: *kubeflow.fairing.deployers.deployer.DeployerInterface*

Handle submitting training job to GCP.

create_request_dict(*pod_template_spec*)
Return the request to be sent to the ML Engine API.

Parameters **pod_template_spec** – pod spec template of the training job

deploy(*pod_template_spec*)
Deploys the training job

Parameters **pod_template_spec** – pod spec template of the training job

get_logs()
Streams the logs for the training job

kubeflow.fairing.deployers.gcp.gcpserving module

class kubeflow.fairing.deployers.gcp.gcpserving.**GCPServingDeployer**(*model_dir*, *model_name*, *version_name*, *project_id=None*, ***deploy_kwargs*)
Bases: *kubeflow.fairing.deployers.deployer.DeployerInterface*

Handle deploying a trained model to GCP.

deploy(*pod_template_spec*)
Deploys the model to Cloud ML Engine.

Parameters **pod_template_spec** – pod spec template of training job

get_logs()
abstract get log

Module contents

kubeflow.fairing.deployers.job package

Submodules

kubeflow.fairing.deployers.job.job module

class kubeflow.fairing.deployers.job.job.**Job**(*namespace=None*, *runs=1*, *output=None*, *cleanup=True*, *labels=None*, *job_name='fairing-job-'*, *stream_log=True*, *deployer_type='job'*, *pod_spec_mutators=None*, *annotations=None*)
Bases: *kubeflow.fairing.deployers.deployer.DeployerInterface*

Handle all the k8s' template building for a training

create_resource()
create job

deploy(*pod_spec*)
deploy the training job using k8s client lib

Parameters **pod_spec** – pod spec of deployed training job

do_cleanup()
clean up the pods after job finished

generate_deployment_spec(*pod_template_spec*)

Generate a V1Job initialized with correct completion and parallelism (for HP search) and with the provided V1PodTemplateSpec

Parameters **pod_template_spec** – V1PodTemplateSpec

generate_pod_template_spec(*pod_spec*)

Generate a V1PodTemplateSpec initiazlied with correct metadata and with the provided pod_spec

Parameters **pod_spec** – pod spec

get_logs()
get logs from the deployed job

set_anotations(*annotations*)

set_labels(*labels*, *deployer_type*)
set labels for the pods of a deployed job

Parameters

- **labels** – dictionary of labels {label_name:label_value}
- **deployer_type** – deployer type name

Module contents

kubeflow.fairing.deployers.kfserving package

Submodules

kubeflow.fairing.deployers.kfserving.kfserving module

class kubeflow.fairing.deployers.kfserving.kfserving.**KFServing**(*framework*, *default_storage_uri=None*, *canary_storage_uri=None*, *canary_traffic_percent=0*, *namespace=None*, *labels=None*, *annotations=None*, *custom_default_container=None*, *custom_canary_container=None*, *stream_log=False*, *cleanup=False*)

Bases: *kubeflow.fairing.deployers.deployer.DeployerInterface*

Serves a prediction endpoint using Kubeflow KFServing.

deploy(*isvc*)
deploy kfserving endpoint

Parameters **isvc** – InferenceService for deploying.

generate_isvc()
generate InferenceService

generate_predictor_spec(*framework*, *storage_uri=None*, *container=None*)
Generate predictor spec according to framework and default_storage_uri or custom container.

get_logs()
get log from prediction pod

set_labels(*labels*)
set label for deployed prediction

Parameters **labels** – dictionary of labels {label_name:label_value}

Module contents

kubeflow.fairing.deployers.serving package

Submodules

kubeflow.fairing.deployers.serving.serving module

class kubeflow.fairing.deployers.serving.serving.**Serving**(*serving_class*, *namespace=None*, *runs=1*, *labels=None*, *service_type='ClusterIP'*, *pod_spec_mutators=None*)

Bases: *kubeflow.fairing.deployers.job.job.Job*

Serves a prediction endpoint using Kubernetes deployments and services

delete()
delete the deployed service

deploy(*pod_spec*)
deploy a seldon-core REST service

Parameters **pod_spec** – pod spec for the service

generate_deployment_spec(*pod_template_spec*)
generate deployment spec(V1Deployment)

Parameters **pod_template_spec** – pod spec template

generate_service_spec()
generate service spec(V1ServiceSpec)

Module contents

kubeflow.fairing.deployers.tfjob package

Submodules

kubeflow.fairing.deployers.tfjob.tfjob module

class kubeflow.fairing.deployers.tfjob.tfjob.**TfJob**(*namespace=None*, *worker_count=1*, *ps_count=0*, *chief_count=1*, *runs=1*, *job_name='fairing-tfjob-'*, *stream_log=True*, *labels=None*, *pod_spec_mutators=None*, *cleanup=False*, *annotations=None*)

Bases: *kubeflow.fairing.deployers.job.job.Job*

Handle all the k8s' template building to create tensorflow training job using Kubeflow TFOperator

create_resource()
create a tfjob training

generate_deployment_spec(*pod_template_spec*)
Returns a TFJob template

Parameters **pod_template_spec** – template spec for pod

get_logs()
get logs

set_container_name(*pod_template_spec*)

Sets the name of the main container to *tensorflow*. This is required for TfJobs

Parameters **pod_template_spec** – spec for pod template

Module contents

Submodules

kubeflow.fairing.deployers.deployer module

class kubeflow.fairing.deployers.deployer.**DeployerInterface**
Bases: object

Deploys a training job to the cluster

deploy(*pod_template_spec*)
Deploys the training job

Parameters **pod_template_spec** – pod template spec

get_logs()
Streams the logs for the training job

Module contents

1.1.6 kubeflow.fairing.frameworks package

Submodules

kubeflow.fairing.frameworks.lightgbm module

kubeflow.fairing.frameworks.lightgbm.**execute**(*config*, *docker_registry*, *base_image='gcr.io/kubeflow-fairing/lightgbm:latest'*, *namespace=None*, *stream_log=True*, *cores_per_worker=None*, *memory_per_worker=None*, *pod_spec_mutators=None*)
Runs the LightGBM CLI in a single pod in user's Kubeflow cluster. Users can configure it to be a train, predict, and other supported tasks by using the right config. Please refere https://github.com/microsoft/LightGBM/blob/master/docs/Parameters.rst for more information on config options.

Parameters

- **config** – config entries
- **docker_registry** – docker registry name
- **base_image** – base image (Default value = "gcr.io/kubeflow-fairing/lightgbm:latest")
- **namespace** – k8s namespace (Default value = None)
- **stream_log** – should that stream log? (Default value = True)
- **cores_per_worker** – number of cores per worker (Default value = None)
- **memory_per_worker** – memory value per worker (Default value = None)
- **pod_spec_mutators** – pod spec mutators (Default value = None)

kubeflow.fairing.frameworks.lightgbm.**generate_context_files**(*config*, *config_file_name*, *num_machines*)
generate context files

Parameters

- **config** – config entries
- **config_file_name** – config file name
- **num_machines** – number of machines

kubeflow.fairing.frameworks.lightgbm_dist_training_init module

kubeflow.fairing.frameworks.utils module

kubeflow.fairing.frameworks.utils.**get_config_value**(*config*, *field_names*)
get value for a config entry

Parameters

- **config** –
- **field_names** –

kubeflow.fairing.frameworks.utils.**init_lightgbm_env**(*config_file*, *mlist_file*)
initialize env for lightgbm

Parameters

- **config_file** – path to config path
- **mlist_file** – path to file to write ip list

kubeflow.fairing.frameworks.utils.**load_properties_config_file**(*config_file*)
load config from a file

Parameters **config_file** – config file path

kubeflow.fairing.frameworks.utils.**nslookup**(*hostname*, *retries=600*)
Does nslookup for the hostname and returns the IPs for it.

Parameters

- **hostname** – hostname to be looked up
- **retries** – Number of retries before failing. In autoscaled cluster, it might take upto 10mins to create a new node so the default value is set high.(Default value = 600)

kubeflow.fairing.frameworks.utils.**parse_cluster_spec_env**()
parse cluster spec env variables

kubeflow.fairing.frameworks.utils.**save_properties_config_file**(*config*, *file_name=None*)
save config into a file

Parameters

- **config** – dictionary of give configs
- **file_name** – path to config file(Default value = None)

kubeflow.fairing.frameworks.utils.**scrub_fields**(*config*, *filed_names*)
scrub fields in config

Parameters

- **config** – config spec
- **filed_names** – name of fields

kubeflow.fairing.frameworks.utils.**update_config_file**(*file_name*, *field_name*, *new_value*)
update config file

Parameters

- **file_name** – file name
- **field_name** – field name
- **new_value** – new value to be added/updated

kubeflow.fairing.frameworks.utils.**write_ip_list_file**(*file_name*, *ips*, *port=None*)
write list of ips into a file

Parameters

- **file_name** – fine name
- **ips** – list of ips
- **port** – default port(Default value = None)

Module contents

1.1.7 kubeflow.fairing.functions package

Submodules

kubeflow.fairing.functions.function_shim module

class kubeflow.fairing.functions.function_shim.**ObjectType**
Bases: enum.Enum

An enumeration.

CLASS = 2

FUNCTION = 1

NOT_SUPPORTED = 3

kubeflow.fairing.functions.function_shim.**call**(*serialized_fn_file*)
Get the content from serialized function.

Parameters **serialized_fn_file** – the file includes serialized function

Returns object: The content of object.

kubeflow.fairing.functions.function_shim.**compare_version**(*local_python_version*)
Compare the Python major and minor version for local and remote python.

Parameters **local_python_version** – Python version of local environment

Returns None.

kubeflow.fairing.functions.function_shim.**get_execution_obj_type**(*obj*)
Get the execution object type, the object can be a function or class.

Parameters **obj** – The name of object such as the function or class

Returns int: The corresponding object type.

Module contents

1.1.8 kubeflow.fairing.kubernetes package

Submodules

kubeflow.fairing.kubernetes.manager module

class kubeflow.fairing.kubernetes.manager.**KubeManager**
Bases: object

Handles communication with Kubernetes' client.

create_deployment(*namespace*, *deployment*)
Create an V1Deployment in the specified namespace.

Parameters

- **namespace** – The custom resource
- **deployment** – Deployment body to create

Returns object: Created V1Deployments.

create_isvc(*namespace*, *isvc*)
Create the provided InferenceService in the specified namespace.

Parameters

- **namespace** – The custom resource
- **InferenceService** – The InferenceService body

Returns object: Created InferenceService.

create_job(*namespace*, *job*)
Creates a V1Job in the specified namespace.

Parameters

- **namespace** – The resource
- **job** – Job defination as kubernetes

Returns object: Created Job.

create_secret(*namespace*, *secret*)
Create secret in the specified namespace.

Parameters

- **namespace** – The custom resource
- **secret** – The secret body

Returns object: Created secret.

create_tf_job(*namespace*, *job*)
Create the provided TFJob in the specified namespace. The TFJob version is defined in TF_JOB_VERSION in fairing.constants. The version TFJob need to be installed before creating the TFJob.

Parameters

- **namespace** – The custom resource

- **job** – The JSON schema of the Resource to create

Returns object: Created TFJob.

delete_deployment(*name*, *namespace*)
Delete an existing model deployment and relinquish all resources associated.

Parameters

- **name** – The deployment name
- **namespace** – The custom resource

Returns obje deployment.

delete_isvc(*name*, *namespace*)
Delete the provided InferenceService in the specified namespace.

Parameters

- **name** – The custom object
- **namespace** – The custom resource

Returns object: The deleted InferenceService.

delete_job(*name*, *namespace*)
Delete the specified job and related pods.

Parameters

- **name** – The job name
- **namespace** – The resource

Returns object: the deleted job.

delete_tf_job(*name*, *namespace*)
Delete the provided TFJob in the specified namespace.

Parameters

- **name** – The custom object
- **namespace** – The custom resource

Returns object: The deleted TFJob.

get_service_external_endpoint(*name*, *namespace*, *selectors=None*)
Get the service external endpoint as http://ip_or_hostname:5000/predict.

Parameters

- **name** – The sevice name
- **namespace** – The custom resource
- **selectors** – A selector to restrict the list of returned objects by their labels.
- **Defaults** – to everything

Returns str: the service external endpoint.

log(*name*, *namespace*, *selectors=None*, *container=''*, *follow=True*)
Get log of the specified pod.

Parameters

- **name** – The pod name

- **namespace** – The custom resource
- **selectors** – A selector to restrict the list of returned objects by their labels.
- **Defaults** – to everything
- **container** – The container for which to stream logs.
- **if** – there is one container in the pod
- **follow** – True or False (Default value = True)

Returns str: logs of the specified pod.

secret_exists(*name*, *namespace*)
Check if the secret exists in the specified namespace.

Parameters

- **name** – The secret name
- **namespace** – The custom resource.

Returns bool: True if the secret exists, otherwise return False.

kubeflow.fairing.kubernetes.utils module

kubeflow.fairing.kubernetes.utils.**get_resource_mutator**(*cpu=None*, *memory=None*)
The mutator for getting the resource setting for pod spec.

The useful example: https://github.com/kubeflow/fairing/blob/master/examples/train_job_api/main.ipynb

Parameters

- **cpu** – Limits and requests for CPU resources (Default value = None)
- **memory** – Limits and requests for memory (Default value = None)

Returns object: The mutator function for setting cpu and memory in pod spec.

kubeflow.fairing.kubernetes.utils.**mounting_pvc**(*pvc_name*, *pvc_mount_path='/mnt'*)
The function for pod_spec_mutators to mount persistent volume claim.

Parameters

- **pvc_name** – The name of persistent volume claim
- **pvc_mount_path** – Path for the persistent volume claim mounts to.

Returns object: function for mount the pvc to pods.

Module contents

1.1.9 kubeflow.fairing.ml_tasks package

Submodules

kubeflow.fairing.ml_tasks.tasks module

class kubeflow.fairing.ml_tasks.tasks.**BaseTask**(*entry_point*, *base_docker_image=None*, *docker_registry=None*, *input_files=None*, *backend=None*, *pod_spec_mutators=None*)

Bases: object

Base class for handling high level ML tasks.

Parameters

- **entry_point** – An object or reference to the source code that has to be deployed.
- **base_docker_image** – Name of the base docker image that should be used as a base image when building a new docker image as part of an ML task deployment.
- **docker_registry** – Docker registry to store output docker images.
- **input_files** – list of files that needs to be packaged along with the entry point. E.g. local python modules, trained model weigths, etc.

class kubeflow.fairing.ml_tasks.tasks.**PredictionEndpoint**(*model_class*, *base_docker_image=None*, *docker_registry=None*, *input_files=None*, *backend=None*, *service_type='ClusterIP'*, *pod_spec_mutators=None*)

Bases: *kubeflow.fairing.ml_tasks.tasks.BaseTask*

Create a prediction endpoint.

create()
Create prediction endpoint.

delete()
Delete prediction endpoint.

predict_nparray(*data*, *feature_names=None*)
Return the prediction result.

Parameters

- **data** – Data to be predicted.
- **feature_names** – Feature extracted from data (Default value = None)

class kubeflow.fairing.ml_tasks.tasks.**TrainJob**(*entry_point*, *base_docker_image=None*, *docker_registry=None*, *input_files=None*, *backend=None*, *pod_spec_mutators=None*)

Bases: *kubeflow.fairing.ml_tasks.tasks.BaseTask*

Create a train job.

submit()
Submit a train job.

kubeflow.fairing.ml_tasks.utils module

kubeflow.fairing.ml_tasks.utils.**guess_preprocessor**(*entry_point*, *input_files*, *output_map*)

Preprocessor to use to modify inputs before sending them to docker build

Parameters

- **entry_point** – entry_point which to use
- **input_files** – input files
- **output_map** – output

kubeflow.fairing.ml_tasks.utils.**is_docker_daemon_exists**()

To check if docker daemon exists or not.

Module contents

1.1.10 kubeflow.fairing.notebook package

Submodules

kubeflow.fairing.notebook.notebook_util module

kubeflow.fairing.notebook.notebook_util.**get_notebook_name**()

Return the full path of the jupyter notebook.

kubeflow.fairing.notebook.notebook_util.**is_in_notebook**()

To check is in notenook or not.

Module contents

1.1.11 kubeflow.fairing.preprocessors package

Submodules

kubeflow.fairing.preprocessors.base module

class kubeflow.fairing.preprocessors.base.**BasePreProcessor**(*input_files=None*, *command=None*, *executable=None*, *path_prefix='/app/'*, *output_map=None*)

Bases: object

Prepares a context that gets sent to the builder for the docker build and sets the entrypoint :param input_files: the source files to be processed :param executable: the file to execute using command (e.g. main.py) :param output_map: a dict of files to be added without preprocessing :param path_prefix: the prefix of the path where the files will be added in the container :param command: the command to pass to the builder

context_map()

Create context mapping from destination to source to avoid duplicates in context archive

Returns c_map: a context map

context_tar_gz(*output_file=None*)
Creating docker context file and compute a running cyclic redundancy check checksum.

Parameters **output_file** – output file (Default value = None)

Returns output_file,checksum: docker context file and checksum

fairing_runtime_files()
Search the fairing runtime files 'runtime_config.py' :returns: cmd: the execute with absolute path

get_command()
Get the execute with absolute path

Returns cmd: the execute with absolute path

is_requirements_txt_file_present()
Verfiy the requirements txt file if it is present.

Returns res: get the present required files

preprocess()
Preprocess the 'input_files'.

Returns input_files: get the input files

set_default_executable()
Set the default executable file.

Returns executable: get the default executable file if it is not existing, Or None

kubeflow.fairing.preprocessors.base.**reset_tar_mtime**(*tarinfo*)
Reset the mtime on the the tarball for reproducibility.

:param tarinfo: the tarball var :returns: tarinfo: the modified tar ball

kubeflow.fairing.preprocessors.converted_notebook module

class kubeflow.fairing.preprocessors.converted_notebook.**ConvertNotebookPreprocessor**(*notebook_*
note-
book_pre
'kube-
flow.fairi
ex-
e-
cutable=
com-
mand=['
path_pref
out-
put_map=
over-
write=Tr

Bases: *kubeflow.fairing.preprocessors.base.BasePreProcessor*

Convert the notebook preprocessor. :param BasePreProcessor: a context that gets sent to the builder for the docker build and sets the entrypoint.

preprocess()
Preprocessor the Notebook :return:[]: the converted notebook list.

class kubeflow.fairing.preprocessors.converted_notebook.**ConvertNotebookPreprocessorWithFire**

Bases: *kubeflow.fairing.preprocessors.converted_notebook.ConvertNotebookPreprocessor*

Create an entrpoint using pyfire.

preprocess()
Preprocessor the Notebook. :return: results: the preprocessed notebook list.

class kubeflow.fairing.preprocessors.converted_notebook.**FilterIncludeCell**(**kw*)
Bases: nbconvert.preprocessors.base.Preprocessor

Notebook preprocessor that only includes cells that have a comment 'fairing:include-cell'. :param NbPreProcessor: the notebook preprocessor.

filter_include_cell(*src*)
Filter the cell that have a comment 'fairing:include-cell'.

Param src: the source cell.

Returns src: if the source cell matched the filter pattern, or Null.

preprocess_cell(*cell*, *resources*, *index*)
Preprocess the notebook cell.

Parameters

- **cell** – the notebook cell
- **resources** – the code source of the notebook cell.
- **index** – unused argumnet.

Returns cell,resources: the notebook cell and its filtered with magic pattern commands.

class kubeflow.fairing.preprocessors.converted_notebook.**FilterMagicCommands**(**kw*)
Bases: nbconvert.preprocessors.base.Preprocessor

Notebook preprocessor that have a comment which started with '!' or '%'. :param NbPreProcessor: the notebook preprocessor.

filter_magic_commands(*src*)
Filter out the source commands with magic pattern.

Parameters **src** – the source commands.

Returns filtered: the filtered commands list.

preprocess_cell(*cell*, *resources*, *index*)
preprocessor that includes cells

Param cell: the notebook cell.

Param resources: the code source of the notebook cell.

Param index: unused argumnet.

Returns cell,resources: the notebook cell and its filtered with magic pattern commands.

kubeflow.fairing.preprocessors.full_notebook module

class kubeflow.fairing.preprocessors.full_notebook.**FullNotebookPreProcessor**(*notebook_file=None*, *output_file='fairing_out*, *input_files=None*, *command=None*, *path_prefix='/app/'*, *output_map=None*)

Bases: *kubeflow.fairing.preprocessors.base.BasePreProcessor*

The Full notebook preprocess for the context which comes from BasePreProcessor. :param BasePreProcessor: a context that gets sent to the builder for the docker build and sets the entrypoint

set_default_executable()
Ingore the default executable setting for the full_notebook preprocessor.

kubeflow.fairing.preprocessors.function module

class kubeflow.fairing.preprocessors.function.**FunctionPreProcessor**(*function_obj*, *path_prefix='/app/'*, *output_map=None*, *input_files=None*)

Bases: *kubeflow.fairing.preprocessors.base.BasePreProcessor*

FunctionPreProcessor preprocesses a single function. It sets as the command a function_shim that calls the function directly. :param BasePreProcessor: a context that gets sent to the builder for the docker build and sets the entrypoint.

get_command()
Get the execute python command. :returns: command: the command line will be executed

Module contents

1.2 Submodules

1.3 kubeflow.fairing.config module

class kubeflow.fairing.config.**Config**
Bases: object

deploy(*pod_spec*)
deploy the job

Parameters **pod_spec** – pod spec of the job

fn(*fn*)
function

Parameters **fn** – return func that set the preprocessorr and run

get_builder(*preprocessor*)
get the builder

Parameters **preprocessor** – preprocessor function

get_deployer()
get deployer

get_preprocessor()
get the preprocessor

reset()
reset the preprocessor, builder_name and deployer name

run()
run the pipeline for job

set_builder(*name='append'*, ***kwargs*)
set the builder

Parameters **name** – builder name (Default value = DEFAULT_BUILDER)

set_deployer(*name='job'*, ***kwargs*)
set the deployer

Parameters **name** – deployer name (Default value = DEFAULT_DEPLOYER)

set_preprocessor(*name=None*, ***kwargs*)

Parameters **name** – preprocessor name(Default value = None)

1.4 kubeflow.fairing.http_utils module

kubeflow.fairing.http_utils.**configure_http_instance**(*http=None*)
Configure http instance to modify the request headers to append or modify user-agent.

Parameters **http** – Body of googleapiclient (Default value = None)

Returns object: Configurated http contents.

1.5 kubeflow.fairing.runtime_config module

class kubeflow.fairing.runtime_config.**RuntimeConfig**
Bases: object

A passthrough config shim that runs in the fairing runtime

fn(*func*)

get_builder()

get_deployer(***kwargs*)

get_preprocessor()

reset()

run()

set_builder(*name*, ***kwargs*)

set_deployer(*name*, ***kwargs*)

set_preprocessor(*name*, ***kwargs*)

1.6 kubeflow.fairing.utils module

kubeflow.fairing.utils.**crc**(*file_name*)
Compute a running Cyclic Redundancy Check checksum.

Parameters **file_name** – The file name that's for crc checksum.

kubeflow.fairing.utils.**get_current_k8s_namespace**()
Get the current namespace of kubernetes.

kubeflow.fairing.utils.**get_default_target_namespace**()
Get the default target namespace, if running in the kubernetes cluster, will be current namespace, Otherwiase, will be "default".

kubeflow.fairing.utils.**get_image**(*repository*, *name*)
Get the full image name by integrating repository and image name.

Parameters

- **repository** – The name of repository
- **name** – The short image name

Returns str: Full image name, format: repo/name.

kubeflow.fairing.utils.**is_running_in_k8s**()
Check if running in the kubernetes cluster.

kubeflow.fairing.utils.**random_tag**()
Get a random tag.

1.7 Module contents

- genindex
- modindex

Python Module Index

k

Index

D

E

F

G

I

J

K

L

M

N

O

P

R

S

T

U

W