AWS SageMaker STEPS

- Create S3 bucket
  - Upload data to bucket
- Create SageMaker notebook instance
- Create Jupyter notebook
  - Set up data
  - Create training job to train model
  - Fit model to data
  - Deploy model for inference
  - Perform inference using deployed model
  - Evaluate model
- Clean up

CREATE BUCKET

- New bucket
  - Do this if you want to create a new bucket
- Services -> S3
- Create bucket
  - Enter bucket name
    - Bucket name must be unique across all S3 buckets
    - Cannot contain uppercase characters or underscores
    - Must start with lowercase letter or number
    - More on bucket naming
      - https://docs.aws.amazon.com/AmazonS3/latest/dev//BucketRestrictions.html#bucketnamingrules
  - Select US West for region (default)
  - Click Create

CREATE FOLDER

- Existing bucket
  - If using an existing bucket, then just create folder within bucket.
- Select bucket
  - Services -> S3
  - Click on bucket to use
- Create folder
  - Select bucket
  - Select ‘Create folder’ for bucket
  - Enter name for folder
  - Click ‘Save’

UPLOAD DATA TO S3 BUCKET

- Services -> S3
- Create folder (if needed)
- Click into folder
- Click ‘Upload’
- Select files
  - Drag & drop or select files to upload
  - Click ‘Upload’

AMAZON SAGEMAKER NOTEBOOK INSTANCE

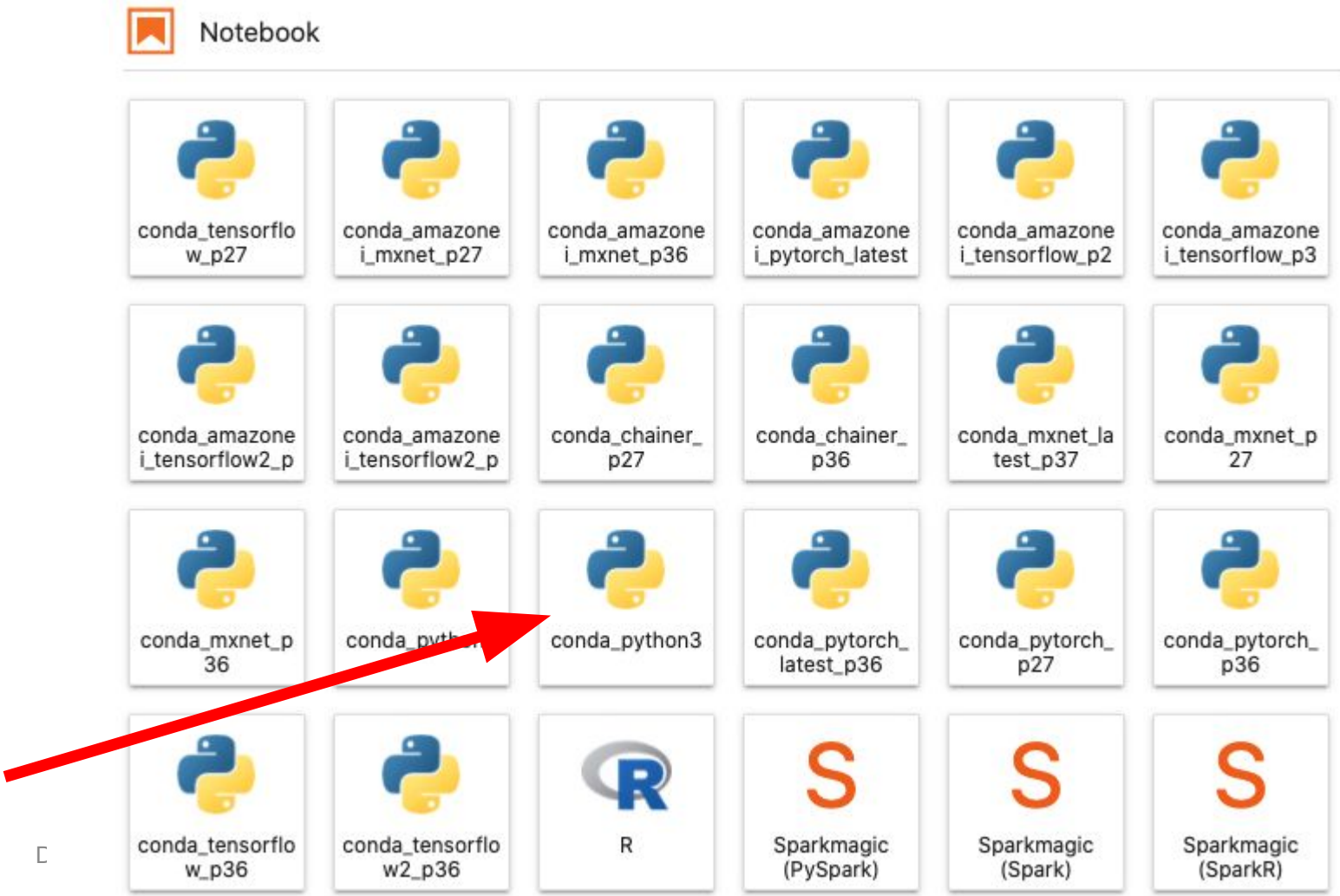
- Services -> Amazon SageMaker (under Machine Learning)
- Click on Dashboard in the left menu
- Click on Notebook Instances
- Click on ‘Create notebook instance’
- Notebook instance name
  - Enter notebook instance name
- Notebook instance type
  - **ml.t3.medium**
- IAM Role
  - Select ‘Create a new role’
  - Specific S3 bucket → Enter your bucket name
  - Click ‘Create role’

CREATE SageMaker NOTEBOOK INSTANCE

- Create notebook instance
  - Click on ‘Create notebook instance’
  - Wait until notebook status changes to InService.
    - May take a few minutes. Click on arrow to refresh.
  - Click ‘Open JupyterLab’
  - Upload existing notebook
    - Click on up arrow to upload notebook
    - Select **conda_python3** kernel
  - Create new notebook
    - File -> New -> Notebook
    - For ‘Select Kernel’, choose conda_python3
    - Edit notebook

CREATE JUPYTER NOTEBOOK

- Starting new notebook from within JupyterLab
- For this class, use conda-python3 as kernel



SPECIFY DATA LOCATION ON S3

```
bucket = “<bucket_name>”
prefix = “<folder_name>”

fname = “<file_name>”
data_fname = "s3://{}/{}/{}".format(bucket,prefix,fname)
df = pd.read_csv(data_fname)
```

SET UP DATA AND OUTPUT ON S3 FOR MODEL

```
bucket = “<bucket_name>”
prefix = “<folder_name>”

train_path =
    “{}/{}/tmodel_data/{}”.format(bucket, prefix,”train_data.csv”)
train_df.to_csv(train_path, index=False, header=False)

s3_input_train =
      sagemaker.inputs.TrainingInput(s3_data=train_path,
                                     content_type='csv')

output_location = "s3://{}/{}/model".format(bucket,prefix)
```

TRAIN MODEL

```
sess = sagemaker.Session()

from sagemaker.amazon.amazon_estimator import image_uris
xgb_image = image_uris.retrieve(framework="xgboost",
                                region=my_region, version='latest')

xgb_model = sagemaker.estimator.Estimator(
                    xgb_image, iam_role,
                    train_instance_count=1,
                    train_instance_type='ml.m5.xlarge’,
                    output_path=<output_location>,
                    sagemaker_session=sess)
xgb_model.fit({'train': <s3_train_loc>, 'validation': <s3_val_loc>})
```

REAL-TIME INFERENCE

```
xgb_predictor = xgb_model.deploy (
    initial_instance_count=1,
    serializer = sagemaker.serializers.CSVSerializer(),
    instance_type='ml.t2.medium')
```

BATCH INFERENCE

```
xgb_transformer =
        xgb_model.transformer(
            instance_count=1,
            instance_type='ml.m5.large’,
            output_path=<output_location>)
```

HYPERPARAMETER TUNING

```
# Specify tuning job parameters
hyperparameter_ranges = {
    'max_depth': IntegerParameter(1, 10)}

# Create tuning job
Optimizer = sagemaker.tuner.HyperparameterTuner(
    estimator=xgb_model,
    hyperparameter_ranges=hyperparameter_ranges,
    base_tuning_job_name='XGBoost-Tuner',
    objective_type='Minimize',
    # objective_metric_name='validation:accuracy',
    objective_metric_name='validation:merror',
    max_jobs=10,
    max_parallel_jobs=5)

# Launch tuning job
Optimizer.fit({'train': s3_input_train, 'validation': s3_input_val})
```

CLEANING UP AFTER SESSION

- At end of each session
- SageMaker Dashboard
  - Stop notebook instances
  - Delete training jobs
  - Delete hyperparameter tuning jobs
  - Delete inference models
  - Delete batch transform jobs
  - i.e., nothing in green
- S3
  - Download results file if haven’t done so

WORK ON EXISTING NOTEBOOK

- Go to SageMaker Dashboard
  - Services -> Amazon SageMaker (under Machine Learning)
- Click on ‘Notebook instances’
- Select existing notebook
  - Click on ‘Start’ under Actions
- Wait until Status changes to InService
- Click on ‘Open in JupyterLab’
  - Click on circular arrow to refresh

CLEANING UP AFTER ASSIGNMENT

- SageMaker
  - Delete notebook instances, models, endpoints
- S3
  - Download results file if haven't done so

CLEANING UP AT END OF COURSE

- Delete everything when done
  - Any running service, even if idle, will incur changes!
- SageMaker
  - Delete notebook instances, models, endpoints
- S3
  - Download results file if haven't done so
  - Select folder and select Actions -> Delete
  - If done, also delete S3 bucket
    - **Note: All contents will be lost!**
- All services
  - Delete all other services