

ML DEPLOYMENT AWS SAGEMAKER



SAKSHI WADHWA & ANISHA JOSHI



SO MANY OPTIONS

WHY AWS SAGE MAKER

Build, train and deploy

Scalable & less complex

One-Click deployment

End-to-end fully managed service

TO DEPLOY ML MODEL IN HOSTED ENVIRONMENT

ML MODEL DEPLOYMENT USING SAGEMAKER

What AWS SageMaker Does?



Select & Prepare Training Data

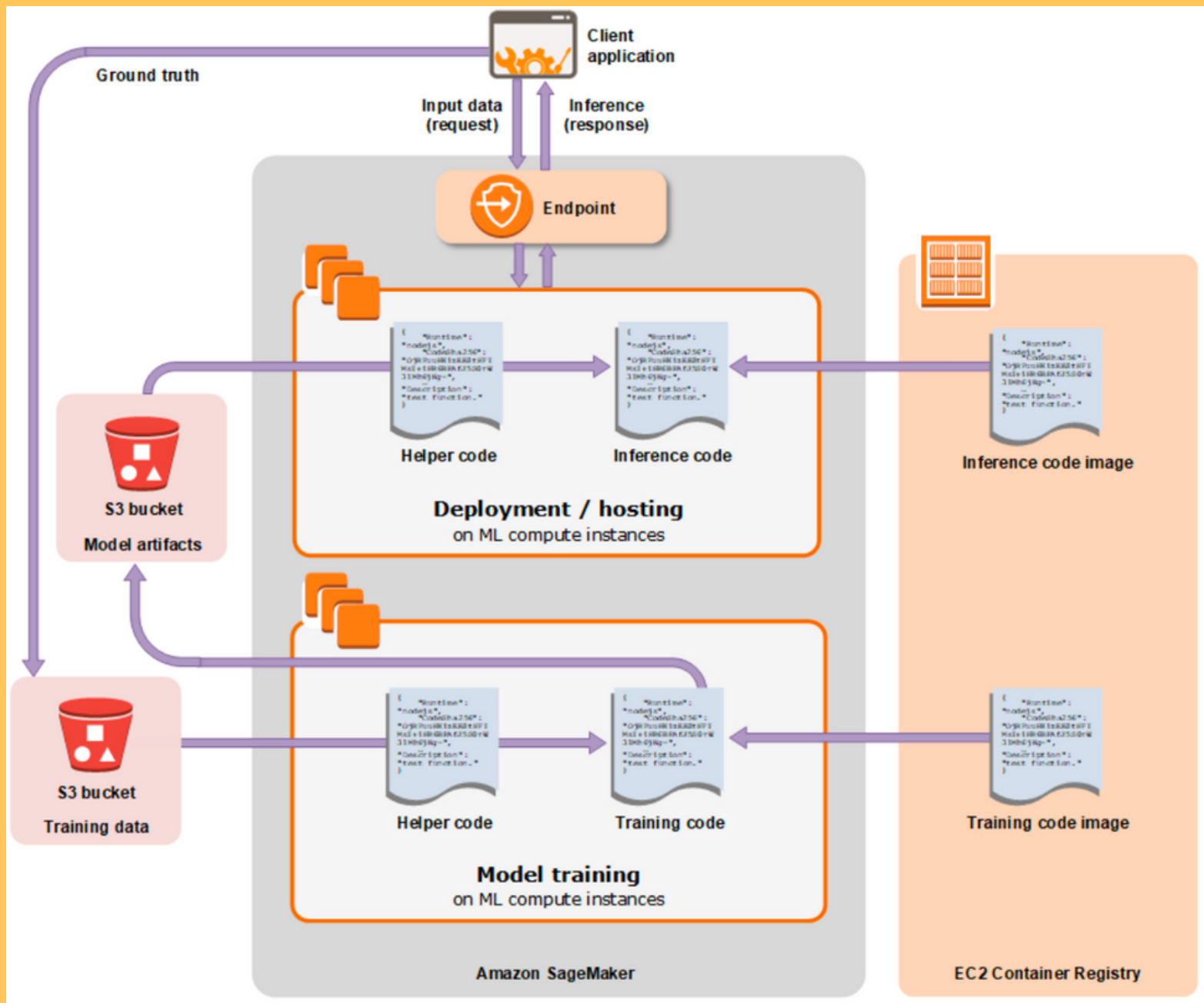
Choose & Optimize Your ML Model

Setup & Manage Environment For Training

Train & Tune Model (Trail & Error)

Deploy Your ML Model To Production

Scale & Manage Production Environment



5 STEPS OF DEPLOYMENT



1. Create EC2 Instance
2. Create s3 buckets to store data and also transfer the train & test data to s3 bucket created
3. Train validation split
4. Model Training Process
5. Model Deployment and Endpoint creation

We used the InBuilt XGBoost classifier in the training phase. In SageMaker each algorithm is stored as containers in Elastic Container Registry(ECR). The ECRs are separately maintained for each region.

PROJECT WALK THROUGH

1. Creating a SageMaker Notebook Instance

The screenshot shows the Amazon SageMaker console's 'Notebook instances' page. At the top, there's a breadcrumb 'Amazon SageMaker > Notebook instances'. Below this, the title 'Notebook instances' is followed by a refresh button, an 'Actions' dropdown, and a prominent orange 'Create notebook instance' button. A search bar labeled 'Search notebook instances' is present. Below the search bar is a table with columns: Name, Instance, Creation time, Status, and Actions. One instance is listed: 'IDS594Project' with instance type 'ml.t2.medium', created on 'Feb 28, 2022 04:57 UTC', and status 'InService'. The Actions column for this instance contains links for 'Open Jupyter' and 'Open JupyterLab'.

Name	Instance	Creation time	Status	Actions
IDS594Project	ml.t2.medium	Feb 28, 2022 04:57 UTC	InService	Open Jupyter Open JupyterLab

2. Creating s3 bucket ~ scalable cloud storage

The screenshot shows the AWS S3 console's 'Buckets' page. The title is 'Buckets (2)' with an 'Info' link. A descriptive text states 'Buckets are containers for data stored in S3.' followed by a 'Learn more' link. Below this, there are buttons for refresh, 'Copy ARN', 'Empty', 'Delete', and a prominent orange 'Create bucket' button. A search bar labeled 'Find buckets by name' is present. Below the search bar is a table with columns: Name, AWS Region, Access, and Creation date. One bucket is listed: 's3bucketmlops' in the 'US West (N. California) us-west-1' region, with 'Bucket and objects not public' access and a creation date of 'February 27, 2022, 18:13:58 (UTC-06:00)'.

Name	AWS Region	Access	Creation date
s3bucketmlops	US West (N. California) us-west-1	Bucket and objects not public	February 27, 2022, 18:13:58 (UTC-06:00)

PROJECT WALK THROUGH

3. Upload the dataset into s3 buckets

Train Test split

```
import numpy as np
train_data, test_data = np.split(model_data.sample(frac=1, random_state=1729), [int(0.7 * len(model_data))])
print(train_data.shape, test_data.shape)
```

```
(28831, 61) (12357, 61)
```

Saving Train And Test Into Buckets

We start with Train Data

```
import os
pd.concat([train_data['y_yes'], train_data.drop(['y_no', 'y_yes'],
                                                axis=1)],
          axis=1).to_csv('train.csv', index=False, header=False)
boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.join(prefix, 'train/train.csv')).upload_file('train.csv')
s3_input_train = sagemaker.inputs.TrainingInput(s3_data='s3://{}/{}/train'.format(bucket_name, prefix), content_type='csv')
```

Test Data Into Buckets

```
pd.concat([test_data['y_yes'], test_data.drop(['y_no', 'y_yes'], axis=1)], axis=1).to_csv('test.csv', index=False, header=False)
boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.join(prefix, 'test/test.csv')).upload_file('test.csv')
s3_input_test = sagemaker.inputs.TrainingInput(s3_data='s3://{}/{}/test'.format(bucket_name, prefix), content_type='csv')
```

This helps in retrieving the training & test data

PROJECT WALK THROUGH

4. Train the Model

```
# Here it is automatically building an URI Container
container = get_image_uri(boto3.Session().region_name,
                           'xgboost',
                           repo_version='1.0-1')
```

```
# Now we are initialize hyperparameters
hyperparameters = {
    "max_depth": "5",
    "eta": "0.2",
    "gamma": "4",
    "min_child_weight": "6",
    "subsample": "0.7",
    "objective": "binary:logistic",
    "num_round": 50
}
```

5. Construct the SageMaker Estimator

[illegible]

PROJECT WALK THROUGH

6. Deploy the Model & Create Endpoints

Deploy Machine Learning Model As Endpoints

```
In [38]: xgb_predictor = estimator.deploy(initial_instance_count=1,instance_type='ml.m4.xlarge')
-----!

In [40]: xgb_predictor.__dict__.keys()

Out[40]: dict_keys(['endpoint_name', 'sagemaker_session', 'serializer', 'deserializer', '_endpoint_config_name', '_model_names', '_context'])
```

Endpoint Creation in SageMaker

The screenshot shows the AWS SageMaker console interface. The left sidebar contains navigation links for Notebook instances, Lifecycle configurations, Git repositories, Processing, Training, Inference (with sub-links for Compilation jobs, Model packages, Models, Endpoint configurations, Endpoints, and Batch transform jobs), Edge Manager, Augmented AI, and AWS Marketplace. The main content area is titled 'Amazon SageMaker > Endpoints'. It features a search bar, a 'Create endpoint' button, and a table of endpoints.

	Name	ARN	Creation time	Status	Last updated
<input type="radio"/>	sagemaker-xgboost-2022-02-27-06-04-41-176	arn:aws:sagemaker:us-east-1:016262909703:endpoint/sagemaker-xgboost-2022-02-27-06-04-41-176	Feb 27, 2022 06:04 UTC	InService	Feb 27, 2022 06:09 UTC
<input type="radio"/>	demo-endpoint-for-ensemble-modelling-2022-02-26-19-56-16	arn:aws:sagemaker:us-east-1:016262909703:endpoint/demo-endpoint-for-ensemble-modelling-2022-02-26-19-56-16	Feb 26, 2022 19:56 UTC	InService	Feb 26, 2022 20:04 UTC



LESSONS LEARNT

- The target column has to be the first column in the data frame and there should be no headers included in the CSV when uploading to S3.
- S3 bucket should be in the same region as our training job
- The building model is inbuilt and to use the correct model we use `get_image_uri()`
- Dedicate one EC2 instance to model training while defining the estimator
- Delete the endpoints and bucket objects at the end

WHAT AFTER DEPLOYMENT

Overall Classification Rate: 89.7%

Predicted Observed	No Purchase	Purchase
No Purchase	91% (10785)	34% (151)
Purchase	9% (1124)	66% (297)

Our model showed an overall classification rate of 89.7%

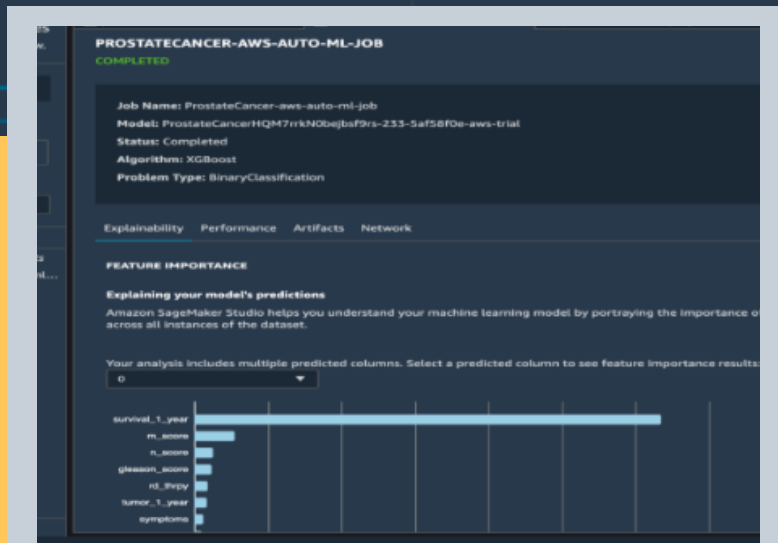
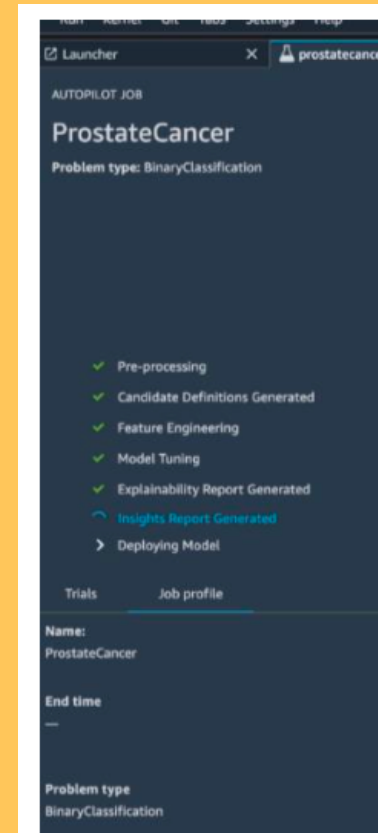
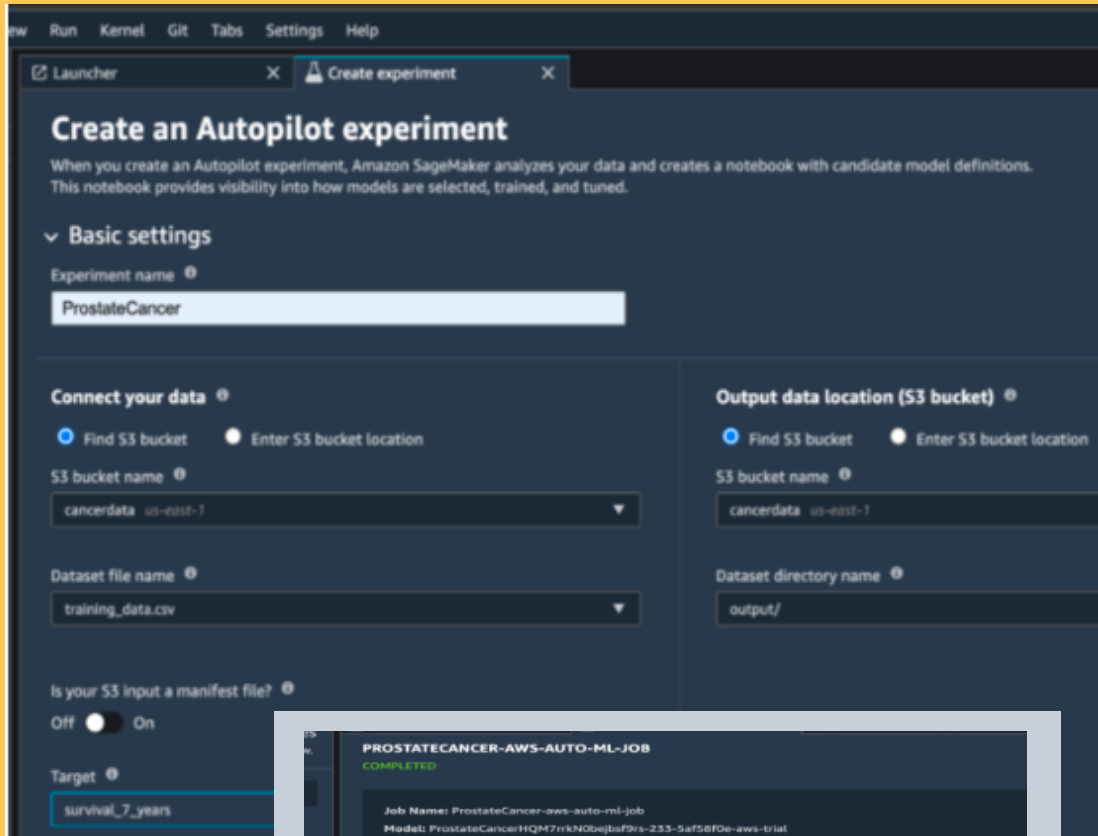
An endpoint is a restful API and Sagemaker automatically creates a restful API for every model that we deploy on Sagemaker. And when we want prediction responses, the service will go and hit the endpoints. These endpoints can be checked in the AWS management console.

Same team members can use these endpoints, directly when need arises, within the enterprise.

AUTOPILOT


Amazon SageMaker Autopilot eliminates the heavy lifting of building ML models

- Provided a prostate cancer dataset for classification problem.
- Got the Best Model Endpoint & Several descriptive reports



There were several stages of the experiment, and we could see them all one by one. The entire exercise took 3 hours and eventually the best model endpoint was provided to us.

The Various reports were also made available – Feature Importance, Accuracy Measures, Explainability Report, Autopilot Candidate definition notebook etc.



*No matter how awesome a model we
have built, it would not give any value
sitting in a jupyter notebook. Hence,
we have to use cloud platforms!*

THANK YOU!

Although this project has ended but it is a start for both of us to explore the scalability of ML models.