Assignment-2

END-TO-END MACHINE LEARNING WORKFLOW
WITH
KIZENML, XAI, AND CLOUD DEPLOYMENT

Group 47

- Vikas Pawar (2022ac05748@wilp.bits-pilani.ac.in)
- Arun Pachauri (2022ac05728@wilp.bits-pilani.ac.in)
- Sourava pani (2022ac05206@wilp.bits-pilani.ac.in)
- Satya Prakash Pandit (2022ac05040@wilp.bits-pilani.ac.in)
- Shivabharti (2022ac05723@wilp.bits-pilani.ac.in)

Table of Contents

1. Objective

For this assignment, we need to create a full end-to-end machine learning workflow using KizenML and Explainable AI (XAI) tools. We can also use other tools or frameworks that fit the workflow, including cloud services like AWS or Azure for deployment. The final submission should include a screen recording that explains my process, along with detailed documentation.

2. Workflow

2.1 Data Collection and Preprocessing

We have chosen Adult Income data set and the objective to build model is to classify the adults into 2 classes based on the features.

- <=50k annual income
- >50k annual income

url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data'

Auto EDA:

EDA report was generated using Dataprep. PFA report.

DataPrep Report.pdf

Based on the insights following preprocessing steps were carried out

Dropped columns that were not influencing the model outcome:

- capital_gain
- capital_loss
- relationship
- marital_status

Transformed numerical and categorical features using StandarScaler, Imputer and One hot encoder

(detail explanation can be found in code files)

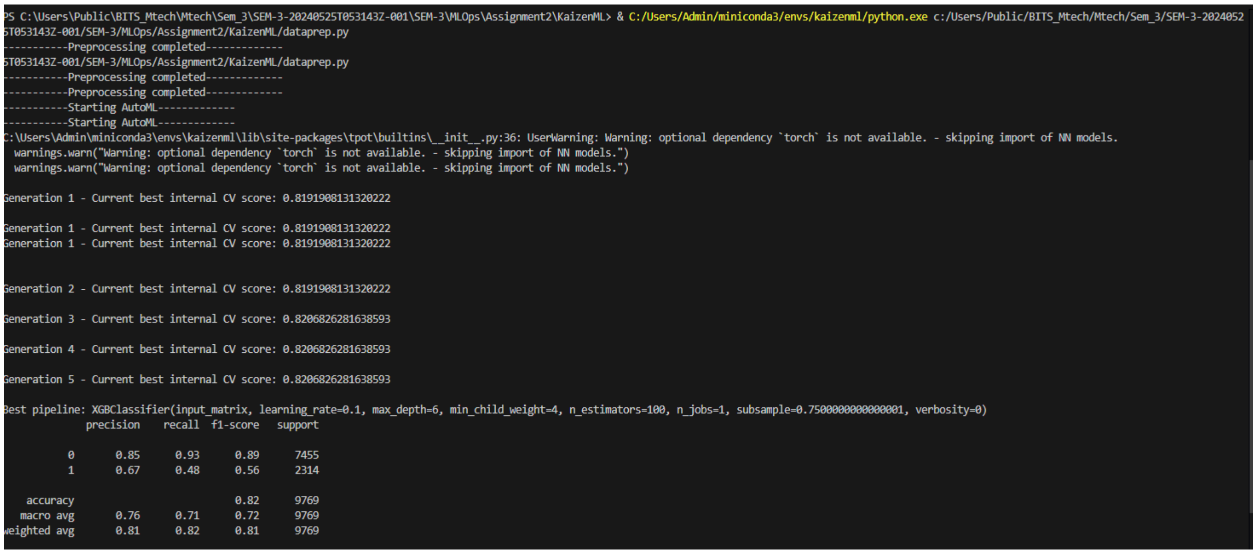
```
numeric_features = ['age', 'fnlwgt', 'education_num','hours_per_week']

categorical_features = ['workclass', 'education', 'occupation', 'race',
'sex', 'native_country']
```

2.2 Model Selection, Training, and Hyperparameter Tuning

We have used TPOT for selection of best algorithm, training and hyperparameter tuning.

We execute TPOT for 5 generations and based or data it suggested to use

```
XGBClassifier(learning_rate=0.1, max_depth=6, min_child_weight=4,
n_estimators=100, n_jobs=1, subsample=0.7500000000000001,
verbosity=0)
```

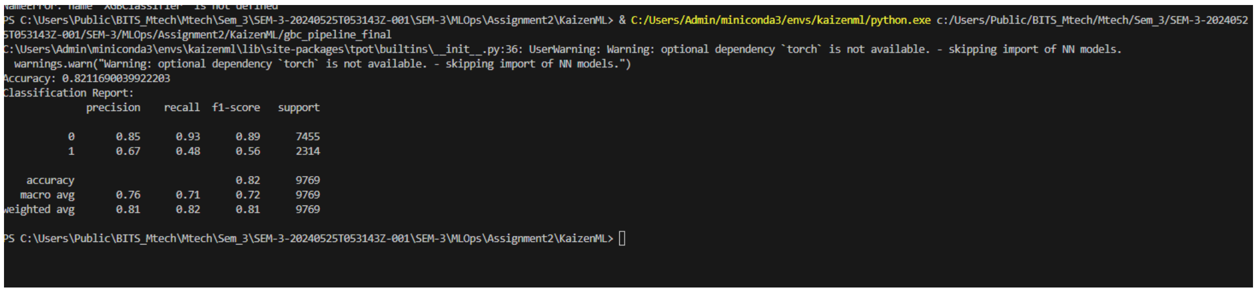


```
PS C:\Users\Public\BITS_Mtech\Mtech\Sem_3\SEM-3-20240525T053143Z-001\SEM-3\MLOps\Assignment2\KaizenML> & C:/Users/Admin/miniconda3/envs/kaizenml/python.exe c:/Users/Public/BITS_Mtech/Mtech/Sem_3/SEM-3-2024052
5T053143Z-001/SEM-3/MLOps/Assignment2/KaizenML/dataprep.py
----------Preprocessing completed------------
5T053143Z-001/SEM-3/MLOps/Assignment2/KaizenML/dataprep.py
----------Preprocessing completed------------
----------Preprocessing completed------------
----------Starting AutoML------------
----------Starting AutoML------------
C:\Users\Admin\miniconda3\envs\kaizenml\lib\site-packages\tpot\builtins\__init__.py:36: UserWarning: Warning: optional dependency `torch` is not available. - skipping import of NN models.
  warnings.warn("Warning: optional dependency `torch` is not available. - skipping import of NN models.")
  warnings.warn("Warning: optional dependency `torch` is not available. - skipping import of NN models.")

Generation 1 - Current best internal CV score: 0.8191908131320222

Generation 1 - Current best internal CV score: 0.8191908131320222
Generation 1 - Current best internal CV score: 0.8191908131320222


Generation 2 - Current best internal CV score: 0.8191908131320222

Generation 3 - Current best internal CV score: 0.8206826281638593

Generation 4 - Current best internal CV score: 0.8206826281638593

Generation 5 - Current best internal CV score: 0.8206826281638593

Best pipeline: XGBClassifier(input_matrix, learning_rate=0.1, max_depth=6, min_child_weight=4, n_estimators=100, n_jobs=1, subsample=0.7500000000000001, verbosity=0)
              precision    recall  f1-score   support

           0       0.85      0.93      0.89      7455
           1       0.67      0.48      0.56      2314

    accuracy                           0.82      9769
   macro avg       0.76      0.71      0.72      9769
weighted avg       0.81      0.82      0.81      9769
```

It generated best_pipeline_5gen.py pipeline which we further trained again on our data set.

Idea behind this is once AutoML recommends a algorithm we can fine tune I further if we need to.

Thus we incorporated the best_pipeline_5gen.py into gbc_pipeline_final.py for training and checking the accuracy



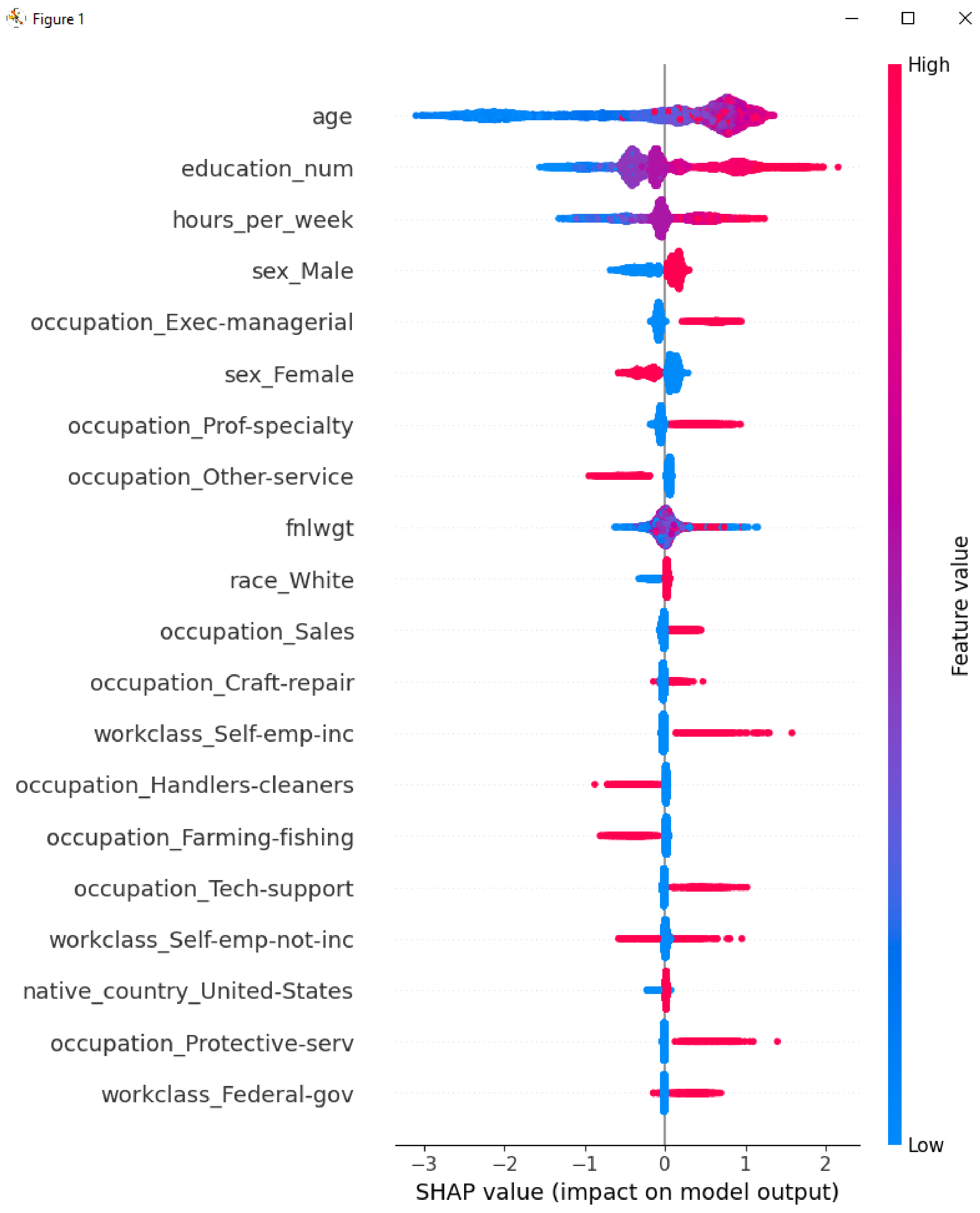
```
PS C:\Users\Public\BITS_Mtech\Mtech\Sem_3\SEM-3-20240525T053143Z-001\SEM-3\MLOps\Assignment2\KaizenML> & C:/Users/Admin/miniconda3/envs/kaizenml/python.exe c:/Users/Public/BITS_Mtech/Mtech/Sem_3/SEM-3-2024052
5T053143Z-001/SEM-3/MLOps/Assignment2/KaizenML/gbc_pipeline_final
C:\Users\Admin\miniconda3\envs\kaizenml\lib\site-packages\tpot\builtins\__init__.py:36: UserWarning: Warning: optional dependency `torch` is not available. - skipping import of NN models.
  warnings.warn("Warning: optional dependency `torch` is not available. - skipping import of NN models.")
Accuracy: 0.8211690039922203
Classification Report:
              precision    recall  f1-score   support

           0       0.85      0.93      0.89      7455
           1       0.67      0.48      0.56      2314

    accuracy                           0.82      9769
   macro avg       0.76      0.71      0.72      9769
weighted avg       0.81      0.82      0.81      9769

PS C:\Users\Public\BITS_Mtech\Mtech\Sem_3\SEM-3-20240525T053143Z-001\SEM-3\MLOps\Assignment2\KaizenML> 
```

Final trained model and preprocessor pipeline were exported to the pkl file

```
joblib.dump(preprocessor, 'preprocessor.pkl')

joblib.dump(exported_pipeline, 'trained_model.pkl')
```

2.3 Explainable AI (XAI) Implementation

We used SHAP tool for XAI implementation.



Components of the SHAP Summary Plot:

X-axis (SHAP value):

The x-axis represents the SHAP values, which indicate how much each feature contributes to pushing a prediction from the base value (expected output across the dataset) either higher (positive SHAP value) or lower (negative SHAP value). Positive SHAP values push the model's prediction towards the positive class (for binary classification), while negative SHAP values push it towards the negative class.

Y-axis (Features):

The y-axis lists all the features in the dataset, ordered by importance. The most important features (those with the largest impact on predictions) are at the top, and the least important features are at the bottom. Color of the Points:

Each point represents a single instance (row of data) in the dataset. The color of the points indicates the feature value for that instance. Typically, red means a high feature value and blue means a low feature value. For example, for the feature "age", red points represent older individuals, and blue points represent younger individuals. Spread of the Points:

The spread along the x-axis shows how much the SHAP value varies across different instances for that feature. A wide spread indicates that the feature has a high impact for some data points but little for others, which is common in complex models with nonlinear relationships.

How to Interpret the Plot:

Feature Importance: The topmost features have the greatest impact on the model's predictions across all instances in the dataset. These are the features your model relies on the most.

SHAP Value Direction:

For each feature, the direction in which the points are spread indicates how changes in that feature value influence the prediction. Positive SHAP value (right side of x-axis): For a given feature, these values push the prediction toward a higher outcome (e.g., income >50K). Negative SHAP value (left side of x-axis): These values push the prediction toward a lower outcome (e.g., income <=50K).

Color Gradient:

Look at the color gradient to understand the feature's contribution. If red points (high feature values) are mainly on the right side (positive SHAP values), it means high values of that feature increase the prediction (e.g., older age increases the probability of earning more than $50K). Conversely, if blue points (low feature values) are on the left side (negative SHAP values), low values of that feature decrease the probability of a positive outcome.

Interpretation:

If "age" is a top feature and the red points are on the right with positive SHAP values, it indicates that older individuals are more likely to have a higher income.

If "hours_per_week" shows a wide spread of SHAP values and most red points (higher working hours) are on the right side, it suggests that people who work more hours tend to have higher incomes, but for some instances, the effect might not be as pronounced.

Observations:

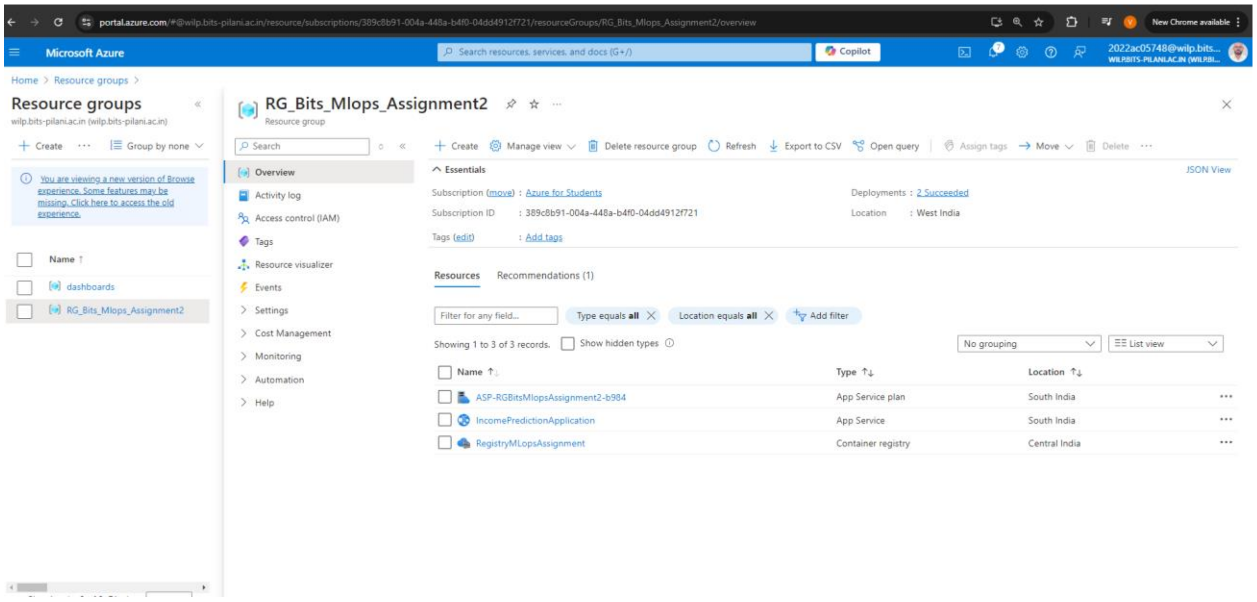
Which Features Matter: The higher up the feature is in the plot, the more important it is. Complex Relationships: The spread of SHAP values shows how a feature impacts the prediction differently across instances, revealing nonlinear relationships. This plot helps you understand how each feature contributes to individual predictions and offers insights into the global behavior of the model.

2.4 Model Deployment Using Cloud Services

We have used **Azure** for our model deployment.

Following steps were carried out:

1) Subscribed to Azure student services plan
2) Created a resource group – which holds all our services



Resource groups
wilp.bits-pilani.ac.in (wilp.bits-pilani.ac.in)

You are viewing a new version of Browse experience. Some features may be missing. Click here to access the old experience.

Name ↑
dashboards
RG_Bits_Mlops_Assignment2

RG_Bits_Mlops_Assignment2
Resource group

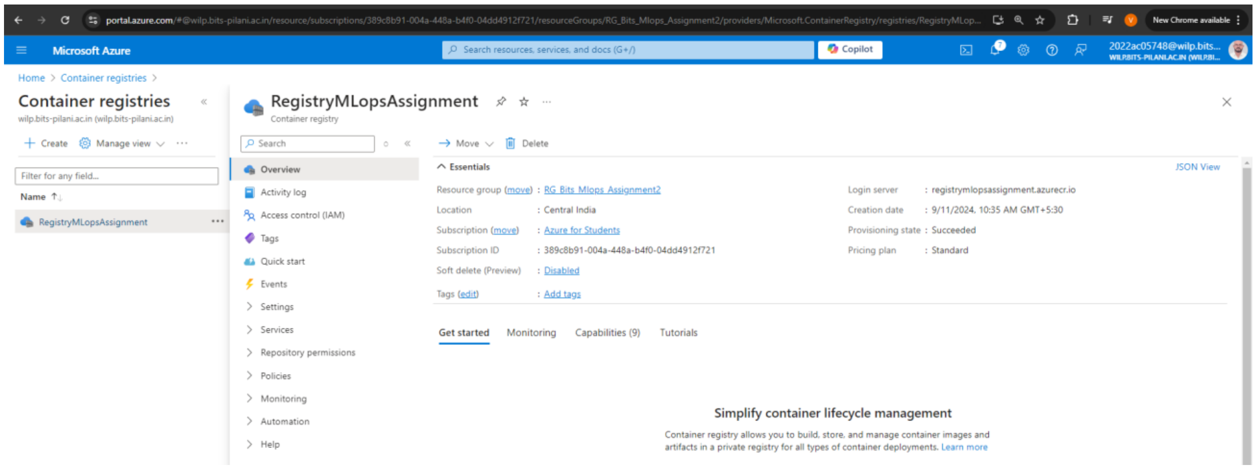
Essentials

Subscription (move) : Azure for Students
Subscription ID : 389c8b91-004a-448a-b4f0-04dd4912f721
Tags (edit) : Add tags
Deployments : 2 Succeeded
Location : West India

Showing 1 to 3 of 3 records.

Name ↑↓	Type ↑↓	Location ↑↓
ASP-RGBitsMlopsAssignment2-b984	App Service plan	South India
IncomePredictionApplication	App Service	South India
RegistryMLopsAssignment	Container registry	Central India

3) We created docker container registry – to hold our docker images



Home > Container registries >

Container registries
wilp.bits-pilani.ac.in (wilp.bits-pilani.ac.in)

RegistryMLopsAssignment
Container registry

Essentials

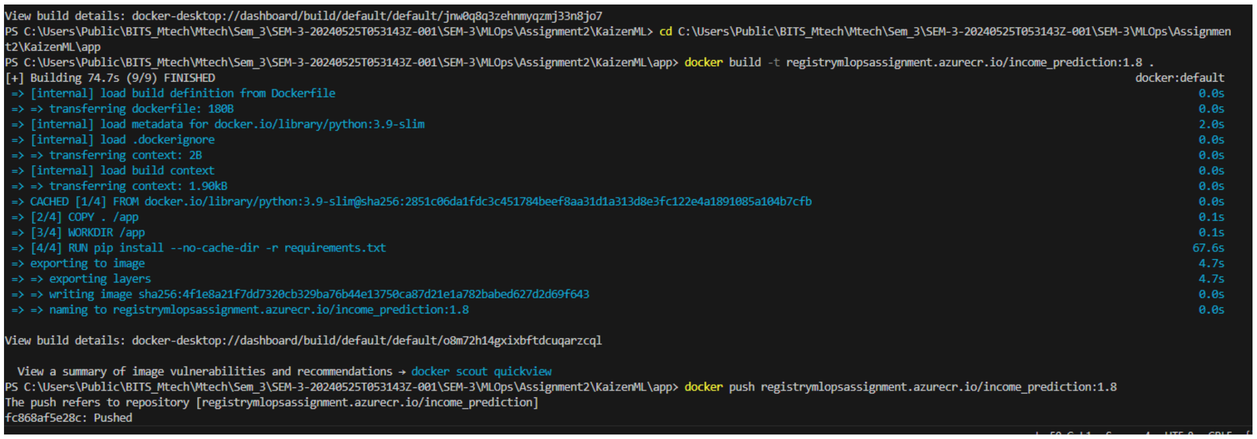
Resource group (move) : RG Bits Mlops Assignment2
Location : Central India
Subscription (move) : Azure for Students
Subscription ID : 389c8b91-004a-448a-b4f0-04dd4912f721
Soft delete (Preview) : Disabled
Tags (edit) : Add tags

Login server : registrymlopsassignment.azurecr.io
Creation date : 9/11/2024, 10:35 AM GMT+5:30
Provisioning state : Succeeded
Pricing plan : Standard

Simplify container lifecycle management

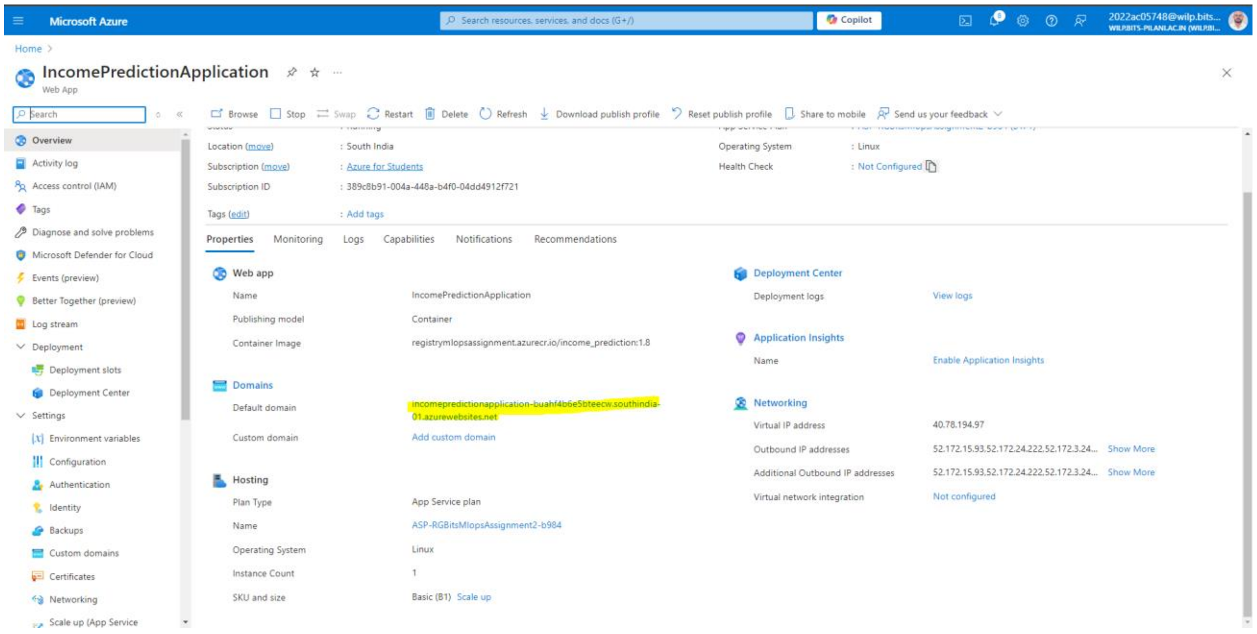
Container registry allows you to build, store, and manage container images and artifacts in a private registry for all types of container deployments. Learn more

4) We built a docker image of our application and then pushed it to Azure container registry.



```
View build details: docker-desktop://dashboard/build/default/default/jnw0q8q3zehnmyqzmj33n8jo7
PS C:\Users\Public\BITS_Mtech\Mtech\Sem_3\SEM-3-20240525T053143Z-001\SEM-3\MLOps\Assignment2\KaizenML> cd C:\Users\Public\BITS_Mtech\Mtech\Sem_3\SEM-3-20240525T053143Z-001\SEM-3\MLOps\Assignmen
t2\KaizenML\app
PS C:\Users\Public\BITS_Mtech\Mtech\Sem_3\SEM-3-20240525T053143Z-001\SEM-3\MLOps\Assignment2\KaizenML\app> docker build -t registrymlopsassignment.azurecr.io/income_prediction:1.8 .
[+] Building 74.7s (9/9) FINISHED                                                                                                    docker:default
 => [internal] load build definition from Dockerfile                                                                                           0.0s
 => => transferring dockerfile: 180B                                                                                                           0.0s
 => [internal] load metadata for docker.io/library/python:3.9-slim                                                                             2.0s
 => [internal] load .dockerignore                                                                                                              0.0s
 => => transferring context: 2B                                                                                                                0.0s
 => [internal] load build context                                                                                                              0.0s
 => => transferring context: 1.90kB                                                                                                            0.0s
 => CACHED [1/4] FROM docker.io/library/python:3.9-slim@sha256:2851c06da1fdc3c451784beef8aa31d1a313d8e3fc122e4a1891085a104b7cfb                0.0s
 => [2/4] COPY . /app                                                                                                                          0.1s
 => [3/4] WORKDIR /app                                                                                                                         0.1s
 => [4/4] RUN pip install --no-cache-dir -r requirements.txt                                                                                  67.6s
 => exporting to image                                                                                                                         4.7s
 => => exporting layers                                                                                                                        4.7s
 => => writing image sha256:4f1e8a21f7dd7320cb329ba76b44e13750ca87d21e1a782babed627d2d69f643                                                   0.0s
 => => naming to registrymlopsassignment.azurecr.io/income_prediction:1.8                                                                      0.0s

View build details: docker-desktop://dashboard/build/default/default/o8m72h14gxixbftdcuqarzcql

  View a summary of image vulnerabilities and recommendations → docker scout quickview
PS C:\Users\Public\BITS_Mtech\Mtech\Sem_3\SEM-3-20240525T053143Z-001\SEM-3\MLOps\Assignment2\KaizenML\app> docker push registrymlopsassignment.azurecr.io/income_prediction:1.8
The push refers to repository [registrymlopsassignment.azurecr.io/income_prediction]
fc868af5e28c: Pushed
```

5) We then created an Azure App service to expose our app as API from the docker image that we had pushed into Azure container registry.



IncomePredictionApplication

Location (move)	: South India
Subscription (move)	: Azure for Students
Subscription ID	: 389c8b91-004a-448a-b4f0-04dd4912f721
Tags (edit)	: Add tags
Operating System	: Linux
Health Check	: Not Configured

Web app

Name	IncomePredictionApplication
Publishing model	Container
Container Image	registrymlopsassignment.azurecr.io/income_prediction:1.8

Domains

Default domain	incomepredictionapplication-buahf4b6e5bteecw.southindia-01.azurewebsites.net
Custom domain	Add custom domain

Hosting

Plan Type	App Service plan
Name	ASP-RGBitsMlopsAssignment2-b984
Operating System	Linux
Instance Count	1
SKU and size	Basic (B1) Scale up

Deployment Center

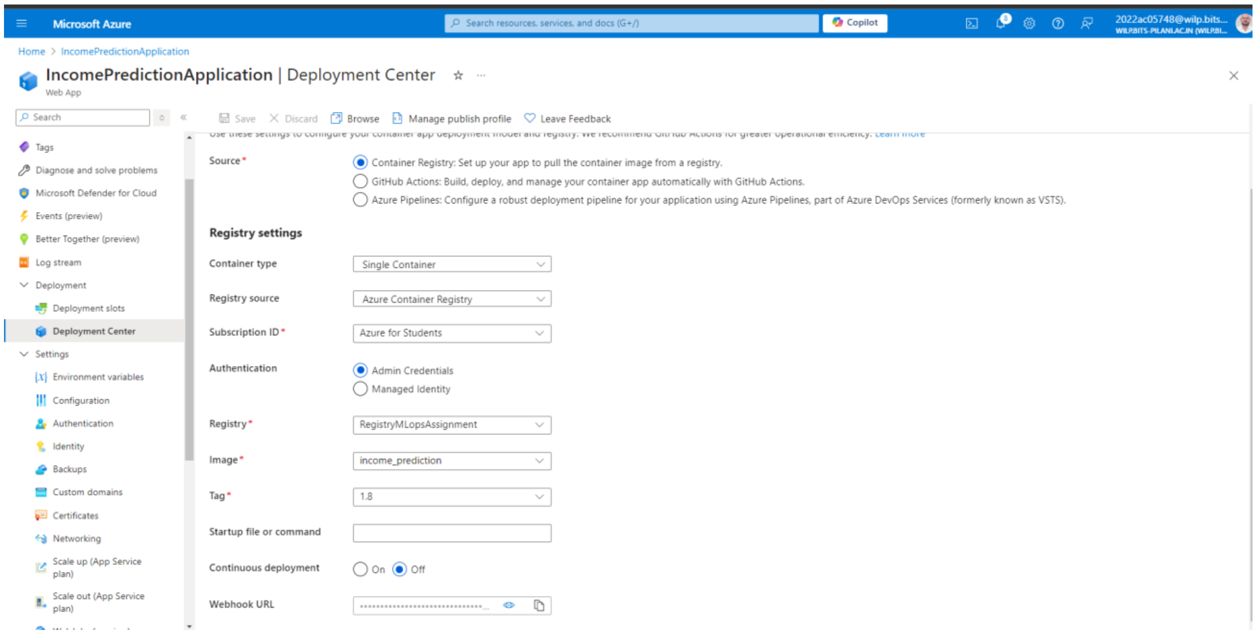
Deployment logs	View logs

Application Insights

Name	Enable Application Insights

Networking

Virtual IP address	40.78.194.97
Outbound IP addresses	52.172.15.93,52.172.24.222,52.172.3.24... Show More
Additional Outbound IP addresses	52.172.15.93,52.172.24.222,52.172.3.24... Show More
Virtual network integration	Not configured



IncomePredictionApplication | Deployment Center

Use these settings to configure your container app deployment model and registry. We recommend GitHub Actions for greater operational efficiency. Learn more

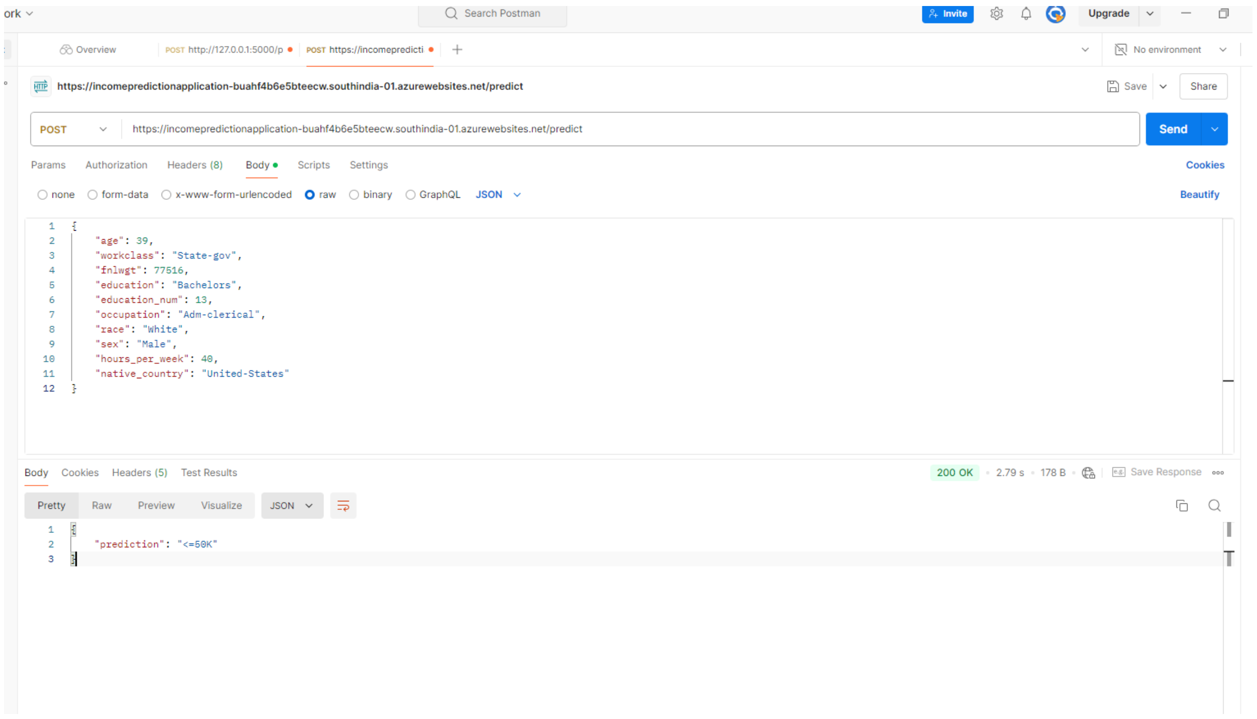
Source *
[✓ Container Registry: Set up your app to pull the container image from a registry.]
GitHub Actions: Build, deploy, and manage your container app automatically with GitHub Actions.
Azure Pipelines: Configure a robust deployment pipeline for your application using Azure Pipelines, part of Azure DevOps Services (formerly known as VSTS).

Registry settings

Container type	Single Container
Registry source	Azure Container Registry
Subscription ID *	Azure for Students
Authentication	[✓ Admin Credentials] Managed Identity
Registry *	RegistryMLopsAssignment
Image *	income_prediction
Tag *	1.8
Startup file or command	
Continuous deployment	On [✓ Off]
Webhook URL	••••••••••••••••••••••••••••••

6) We then started the APP service and then through **POSTMAN** sent the request and got the model prediction. Below are the sample predictions from each class.

Income <= 50k



POST https://incomepredictionapplication-buahf4b6e5bteecw.southindia-01.azurewebsites.net/predict

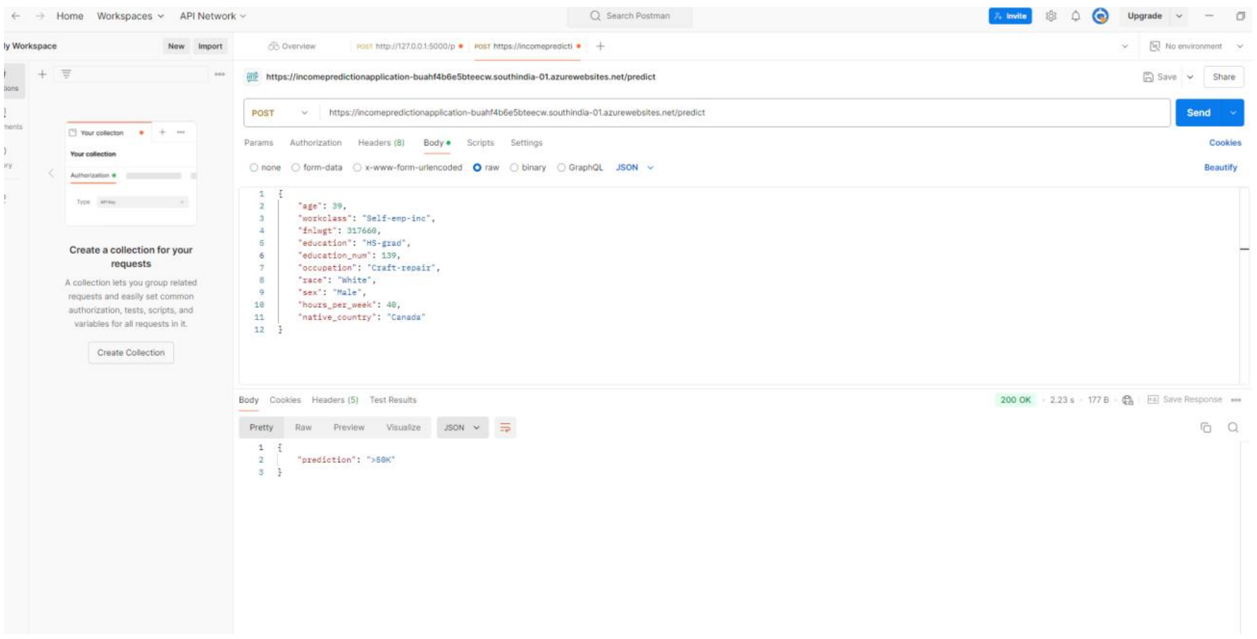
Body: raw, JSON

```
{
    "age": 39,
    "workclass": "State-gov",
    "fnlwgt": 77516,
    "education": "Bachelors",
    "education_num": 13,
    "occupation": "Adm-clerical",
    "race": "White",
    "sex": "Male",
    "hours_per_week": 40,
    "native_country": "United-States"
}
```

200 OK · 2.79 s · 178 B

```
{
    "prediction": "<=50K"
}
```

Income > 50k



POST https://incomepredictionapplication-buahf4b6e5bteecw.southindia-01.azurewebsites.net/predict

Body: raw, JSON

```
{
    "age": 39,
    "workclass": "Self-emp-inc",
    "fnlwgt": 317660,
    "education": "HS-grad",
    "education_num": 139,
    "occupation": "Craft-repair",
    "race": "White",
    "sex": "Male",
    "hours_per_week": 40,
    "native_country": "Canada"
}
```

200 OK · 2.23 s · 177 B

```
{
    "prediction": ">50K"
}
```

3. Challenges and Solutions

We encountered below challenges

1) Initially we tried to use AutoSklearn library to perform Auto ML . We faced lot of environmental issues and therefore had to switch to TPOT.
2) TPOT autoML was taking too long to train on local desktop. So we reduced the generations from 10 to 5.
3) After building the docker image locally, at the time of testing we were unable to connect with Docker image. The issue occurred as we had not done the port mapping. We resolved it using below mapping.

   ```
   docker run -d -p 5000:80 0e6cdc868afd
   ```
4) Later when the docker image was pushed to Azure container registry and when we tried to access it vias **POSTMAN** we encountered

   ```
   <!doctype html>
   <html lang=en>
    <title>405 Method Not Allowed</title>
    <h1>Method Not Allowed</h1>
    <p>The method is not allowed for the requested URL.</p>
   ```
   It was resolved after entering correct URL
5) Finally we encountered below error

   ```
   { "error": "\"None of [Index(['age', 'fnlwgt', 'education_num',
   'hours_per_week', 'workclass',\\n 'education', 'occupation', 'race',
   'sex', 'native_country'],\\n dtype='object')] are in the [columns]\"" }
   ```

After trying various options we figure out that this was because we were passing parameters in a column format as below

```
{
    "data": [
        {
            "age": 39,
            "workclass": "State-gov",
            "fnlwgt": 77516,
            "education": "Bachelors",
            "education_num": 13,
            "occupation": "Adm-clerical",
            "race": "White",
            "sex": "Male",
            "hours_per_week": 40,
            "native_country": "United-States"
        }
    ]
}
```

We then rectified the format as below and finally our model started predicting

```
{
    "age": 39,
    "workclass": "State-gov",
    "fnlwgt": 77516,
    "education": "Bachelors",
    "education_num": 13,
    "occupation": "Adm-clerical",
    "race": "White",
    "sex": "Male",
    "hours_per_week": 40,
    "native_country": "United-States"
}
```

4. Conclusion

In this assignment we learnt about KaizenML concept and its implementation via TPOT library.

We also got acquainted with Azure platform and managed to deploy our model as an API through docker image.

Repository link:

`https://github.com/VP8145/mlops_automl.git`

Model API: (POSTMAN)

https://incomepredictionapplication-buahf4b6e5bteecw.southindia-01.azurewebsites.net/predict

Model parameters for getting prediction

```
{
    "age": 39,
    "workclass": "State-gov",
    "fnlwgt": 77516,
    "education": "Bachelors",
    "education_num": 13,
    "occupation": "Adm-clerical",
    "race": "White",
    "sex": "Male",
    "hours_per_week": 40,
    "native_country": "United-States"
}

{
    "age": 39,
    "workclass": "Self-emp-inc",
    "fnlwgt": 317660,
    "education": "HS-grad",
    "education_num": 139,
    "occupation": "Craft-repair",
    "race": "White",
    "sex": "Male",
    "hours_per_week": 40,
    "native_country": "Canada"
}
```

5. References / URLs

Dataset: https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data

Code Repository (GitHub): https://github.com/arunpachauri/MLOPs

Postman URL (Azure): https://incomepredictionapplication-buahf4b6e5bteecw.southindia-01.azurewebsites.net/predict