

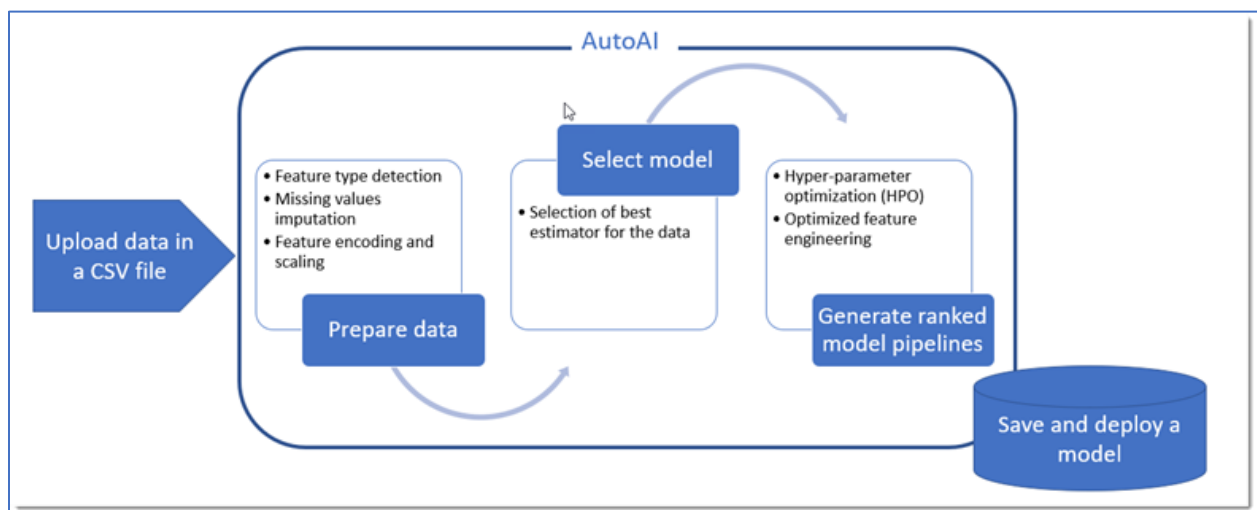
AutoAI + DevOps

Introduction

This lab consists of two parts. The first part will demonstrate the new and exciting AutoAI capability to build and deploy an optimized model based on the Female Human Trafficking datasets. The second part will deploy an application using the IBM Cloud DevOps toolchain that will invoke the deployed model to predict the trafficking risk.

AutoAI in Watson Studio automatically analyzes your data and generates candidate model pipelines customized for your predictive modeling problem. AutoAI algorithms analyze your dataset to discover data transformations, estimator algorithms, and parameter settings that work best for your problem setting. Results are displayed on a leaderboard, showing the automatically generated model pipelines ranked according to your problem optimization objective.

Using AutoAI, you can build and deploy a machine learning model with sophisticated training features and no coding. The tool does most of the work for you.



The AutoAI process follows this sequence to build candidate pipelines:

- [Data pre-processing](#)
- [Automated model selection](#)
- [Automated feature engineering](#)
- [Hyperparameter optimization](#)

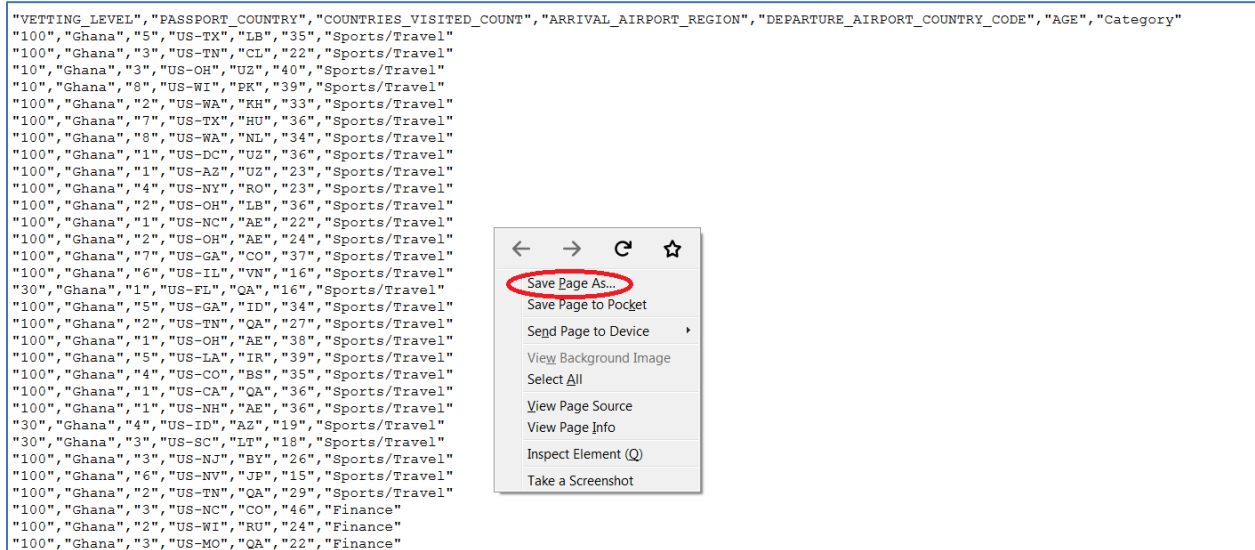
We will perform the following steps in this lab:

1. Download a female human trafficking cleansed dataset from the github repo
2. Add an Auto AI Experiment to create a model to predict the trafficking risk
3. Save and Deploy the model

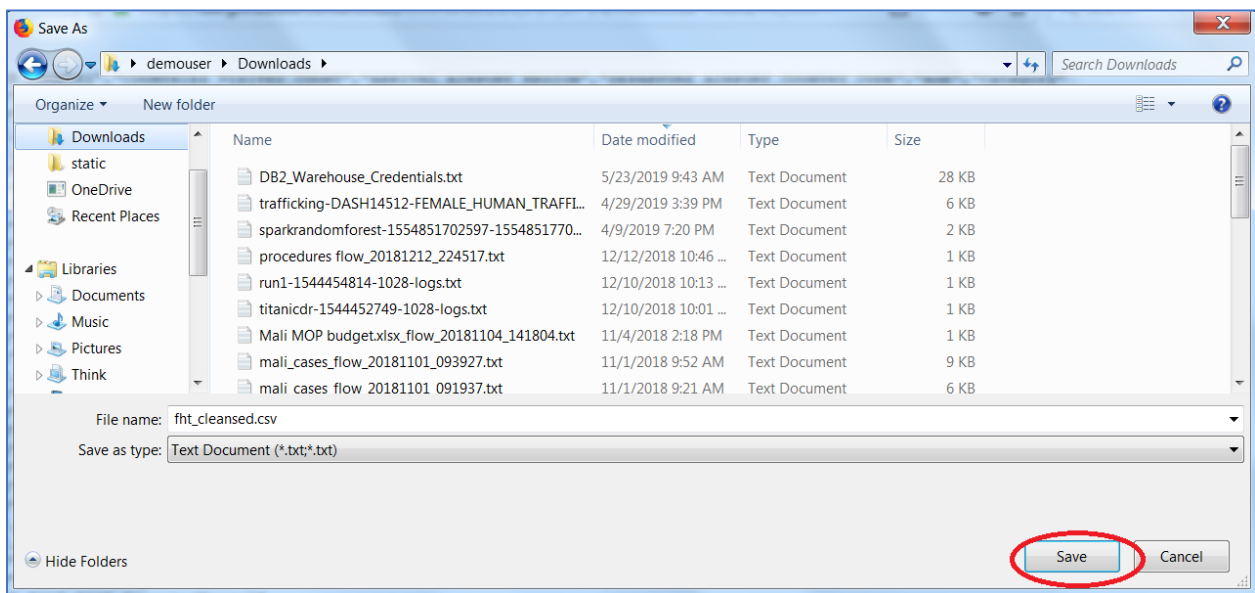
4. Test the model
5. Deploy a simple web front-end application and connecting it to the deployed model using an IBM Cloud DevOps toolchain.

Step 1: Download a female human trafficking cleansed dataset

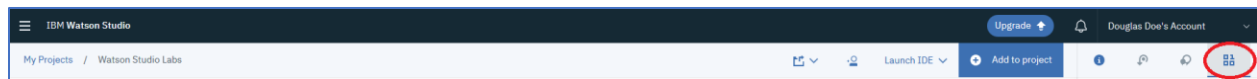
1. Download the fht_cleansed.csv data file from the following location by clicking [here](#)
2. Right-click on the window, and click **Save Page As...**



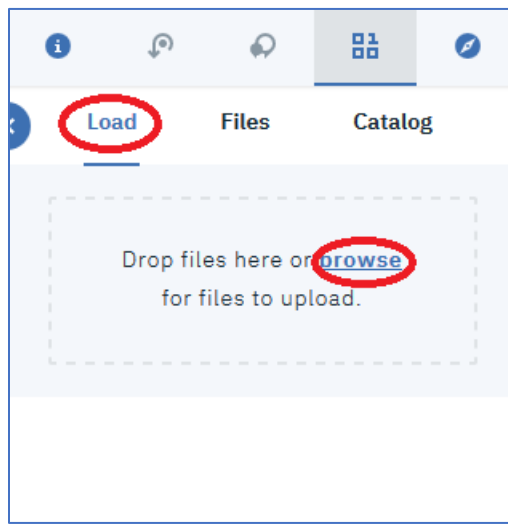
3. Click on **Save**.



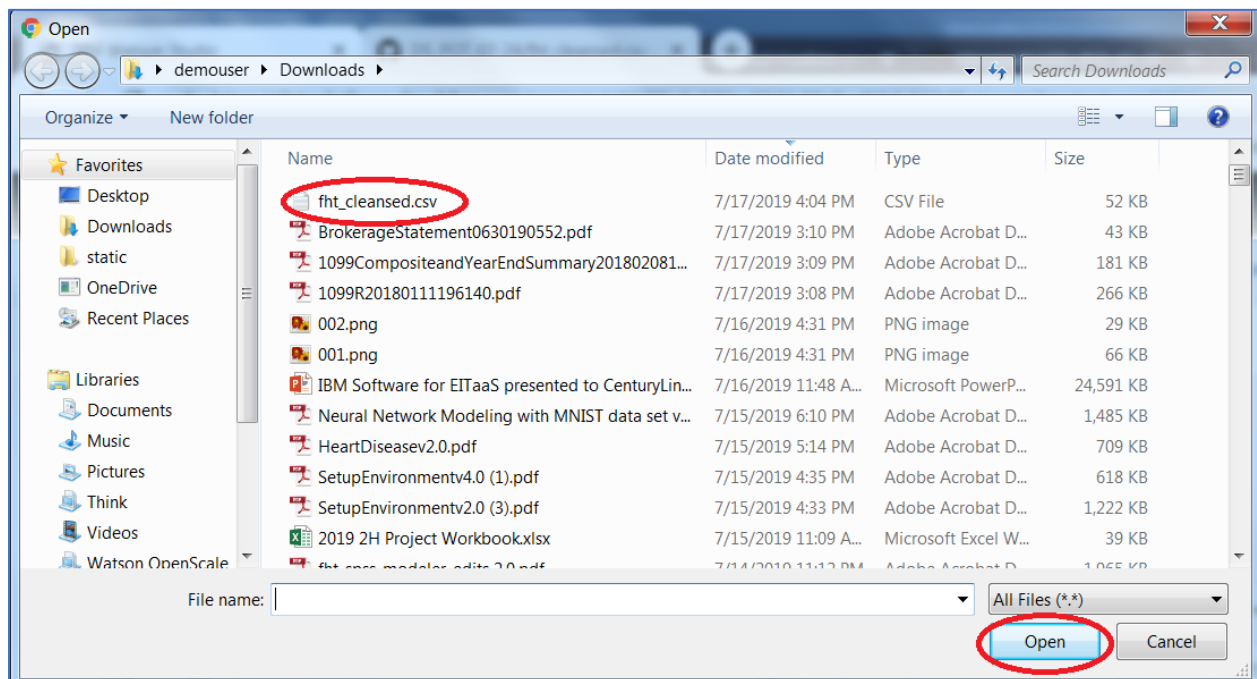
4. Go back to your Watson Studio Labs project. Click on the  icon.



- Click on the **Load** tab and then click on **browse**. If you don't see the **Load** tab, click on the  icon again.



- Go to the folder where the fht_cleansed.csv file is stored. Select the fht_cleansed.csv file and then click **Open**.



- The fht_cleansed.csv file is now added as a Data Asset.

My Projects / Watson Studio Labs

Overview **Assets** Environments Jobs Bookmarks Deployments Access Control Settings

What assets are you looking for?

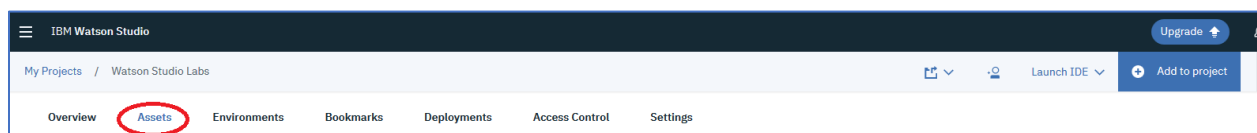
▼ Data assets

0 asset selected.

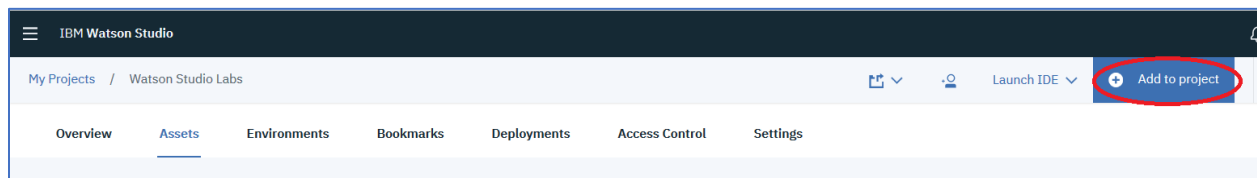
<input type="checkbox"/>	NAME	TYPE	CREATED BY	LAST MODIFIED ▼	ACTIONS
<input type="checkbox"/>	CSV fht_cleansed.csv	Data Asset	Robert Doe	17 Jul 2019, 4:07:36 pm	

Step 2: Add an AutoAI Experiment.

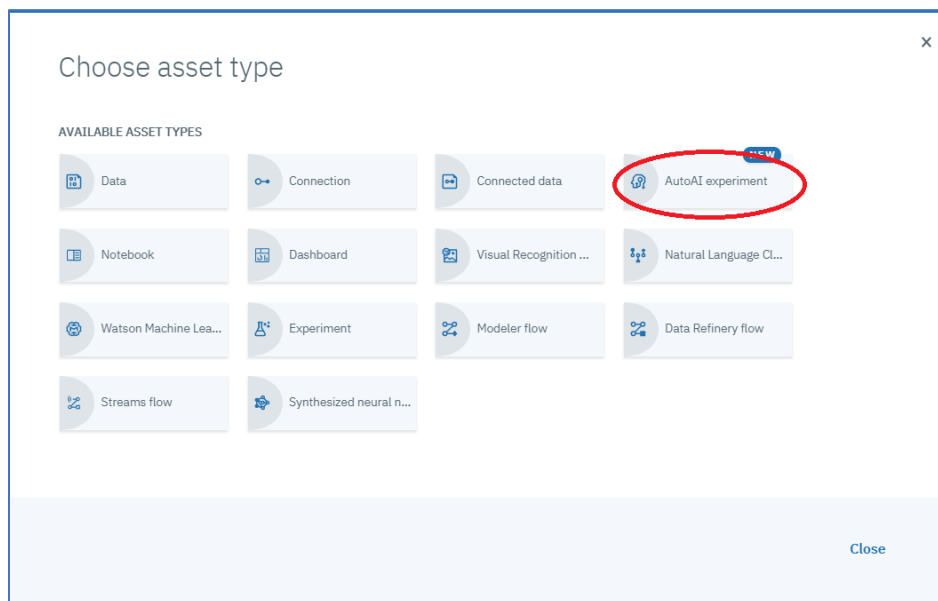
1. If not on the **Assets** page, click on the **Assets** Tab



2. Click on **Add to project**.



3. Click on **AutoAI Experiment**.



4. Enter an **Asset name**, leave the defaults for the **Watson Machine Learning** and **Compute configuration** and click on **Create**.

Create AutoAI experiment

Use this no-code approach to generate a prediction based on data you provide. AutoAI automatically finds the best algorithm for your data and use case. [Learn more](#)

Define AutoAI details

Create AutoAI experiment type

☒ From blank ☐ From sample

Asset name *

FHT_AutoAI

Description

Description of AutoAI experiment

Associated services

Watson Machine Learning Service Instance *

WatsonMachineLearning

Compute configuration * [i](#)

8 vCPU and 32 GB RAM

This compute configuration consumes 20 capacity units per hour. [Learn more](#) about capacity unit hours and Watson Machine Learning pricing plans.

Cancel Create


5. Click on **Select from project**.

My Projects / Watson Studio Labs / FHT_Auto_AI

Configure AutoAI experiment

FHT_Auto_AI

Add data source

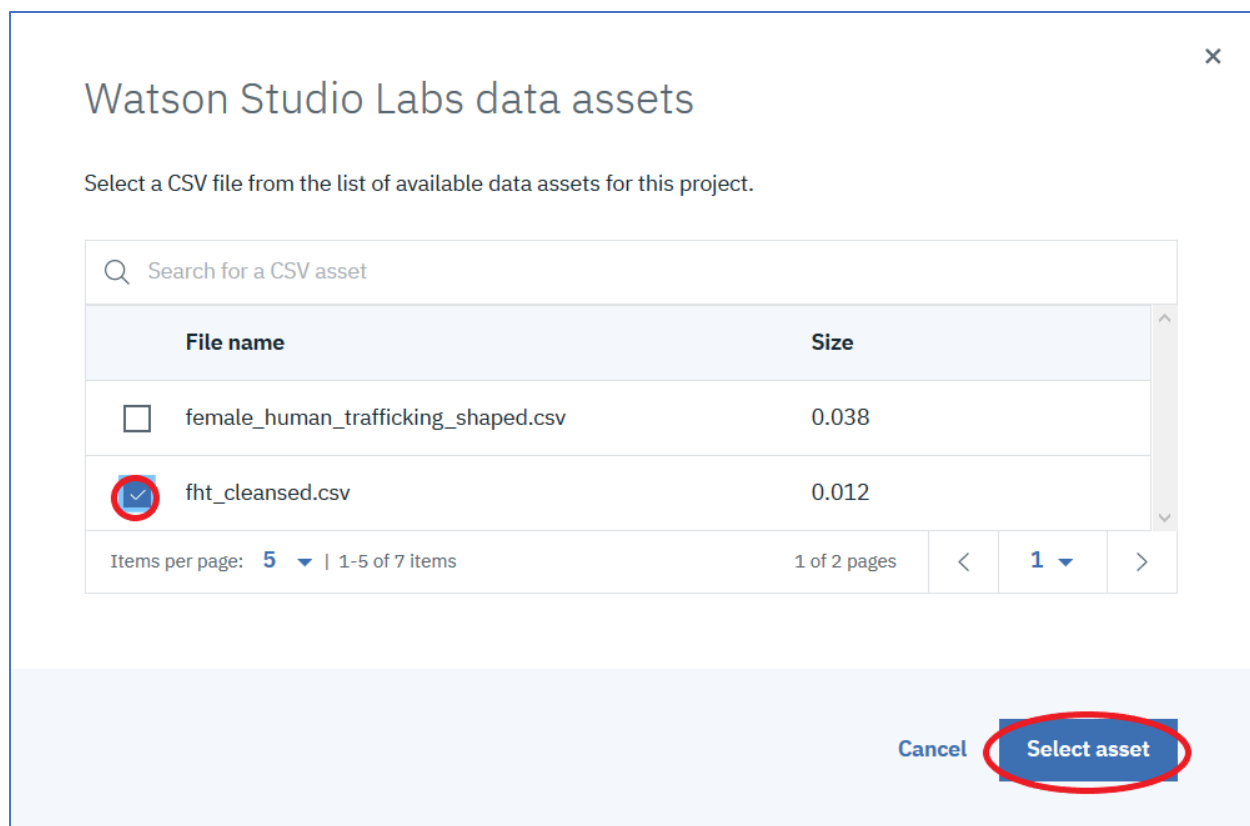


Drop a .csv file here or [browse](#) for a file to upload. Maximum file size is 100 MB.

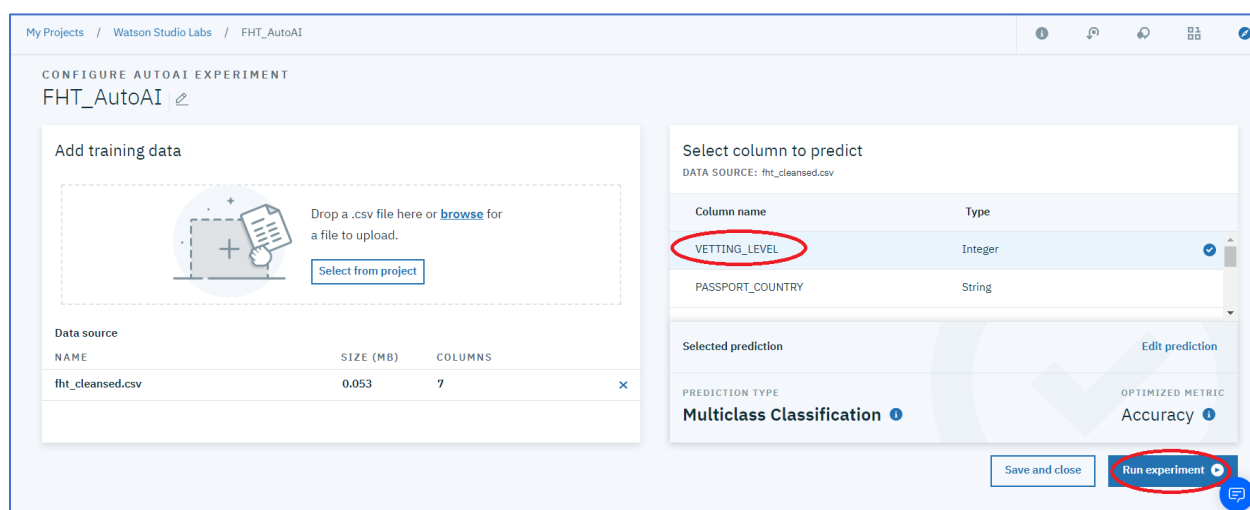
— OR —

Select from project

6. Select the **fht_cleansed.csv** dataset and click on **Select asset**.

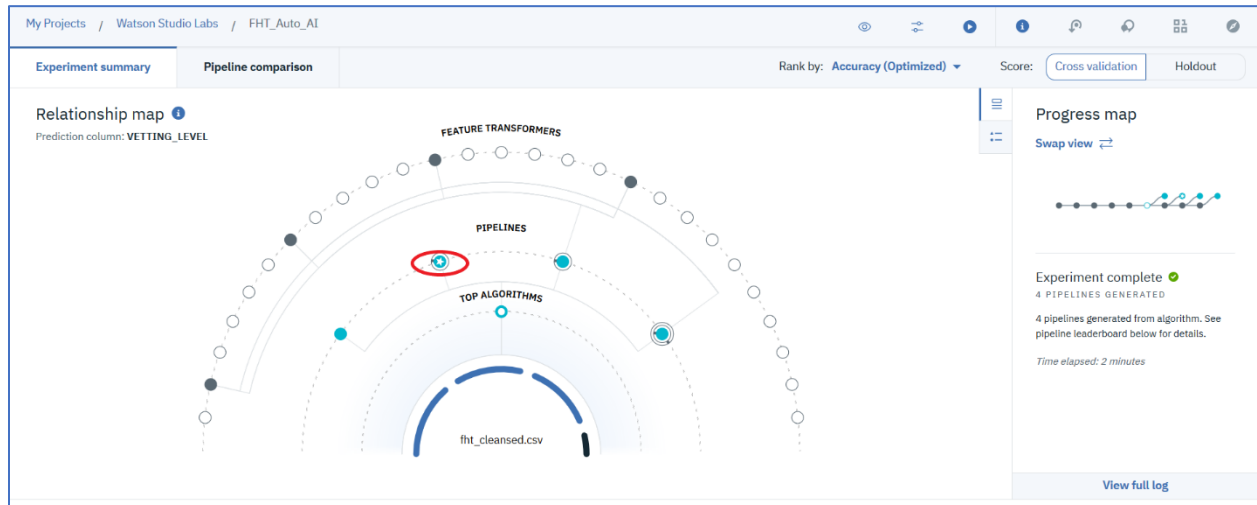


7. Select **VETTING_LEVEL** as the column to predict, leave the default for **OPTIMIZED METRIC** and click on **Run Experiment**. Note the system scanned the **VETTING_LEVEL** values to determine that a **Multiclass Classification** was the **PREDICTION TYPE**.



8. Each pipeline is displayed as the processing is completed for the pipeline. Four pipelines will be created. The first pipeline picks the best algorithm. In this case, it is Random Forest. The second pipeline performs a hyperparameter optimization to see if tuning the algorithm parameters will improve the performance metric. The third pipeline will derive

new features (i.e. feature engineering) to improve the performance metric. Finally, the fourth pipeline will do another hyperparameter optimization including the newly derived features. The results are shown visually below. The second pipeline performs the best based on the cross-validation accuracy.

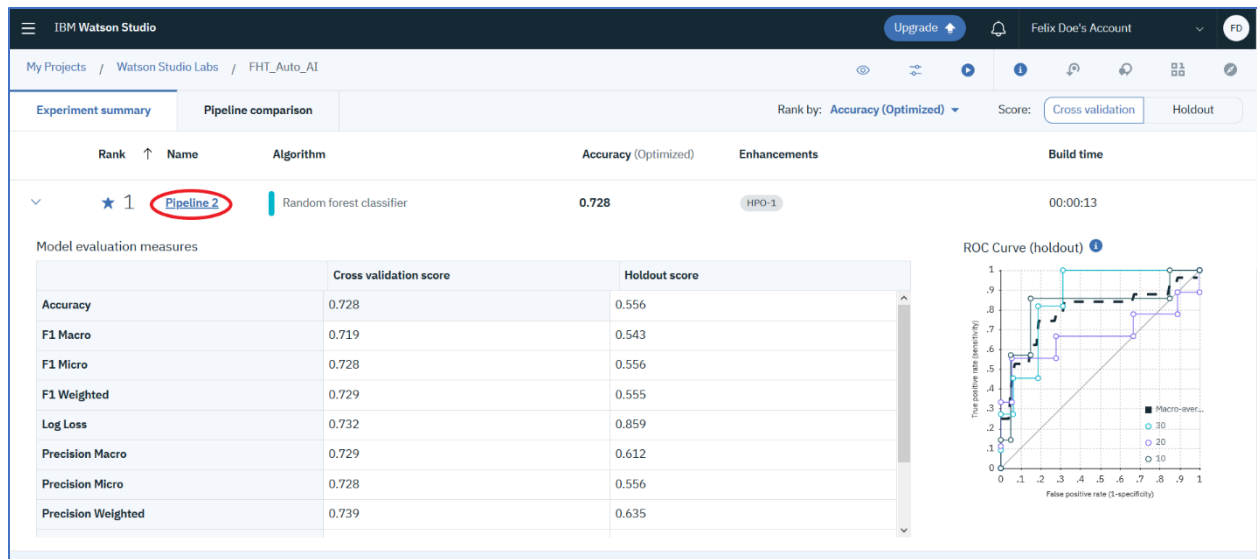


9. Scroll down to view the Pipeline leaderboard. Click on the right caret next to Pipeline 2.

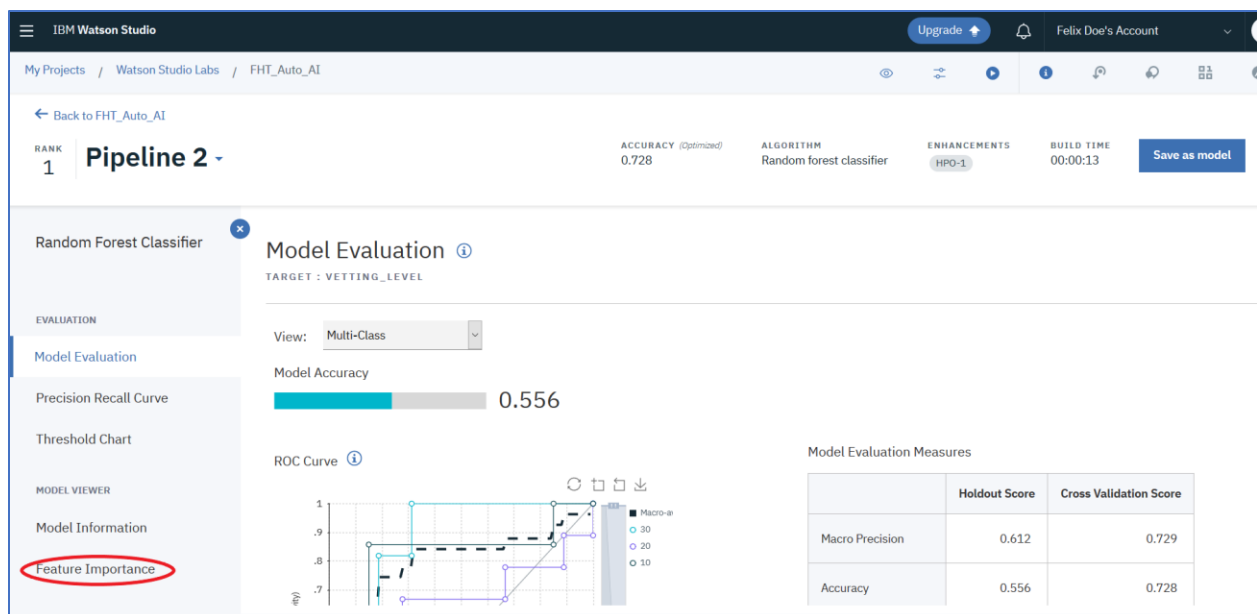
The screenshot shows the Watson Studio interface with the 'Pipeline leaderboard' expanded. The leaderboard displays four pipelines, all using a 'Random forest classifier' algorithm. Pipeline 2 is highlighted as the top performer with an accuracy of 0.728. The 'Enhancements' column shows the specific optimizations applied to each pipeline.

Rank	Name	Algorithm	Accuracy (Optimized)	Enhancements	Build time
1	Pipeline 2	Random forest classifier	0.728	HPO-1	00:00:13
2	Pipeline 4	Random forest classifier	0.720	HPO-1 FE HPO-2	00:00:25
3	Pipeline 1	Random forest classifier	0.702	None	00:00:01
4	Pipeline 3	Random forest classifier	0.699	HPO-1 FE	00:01:01

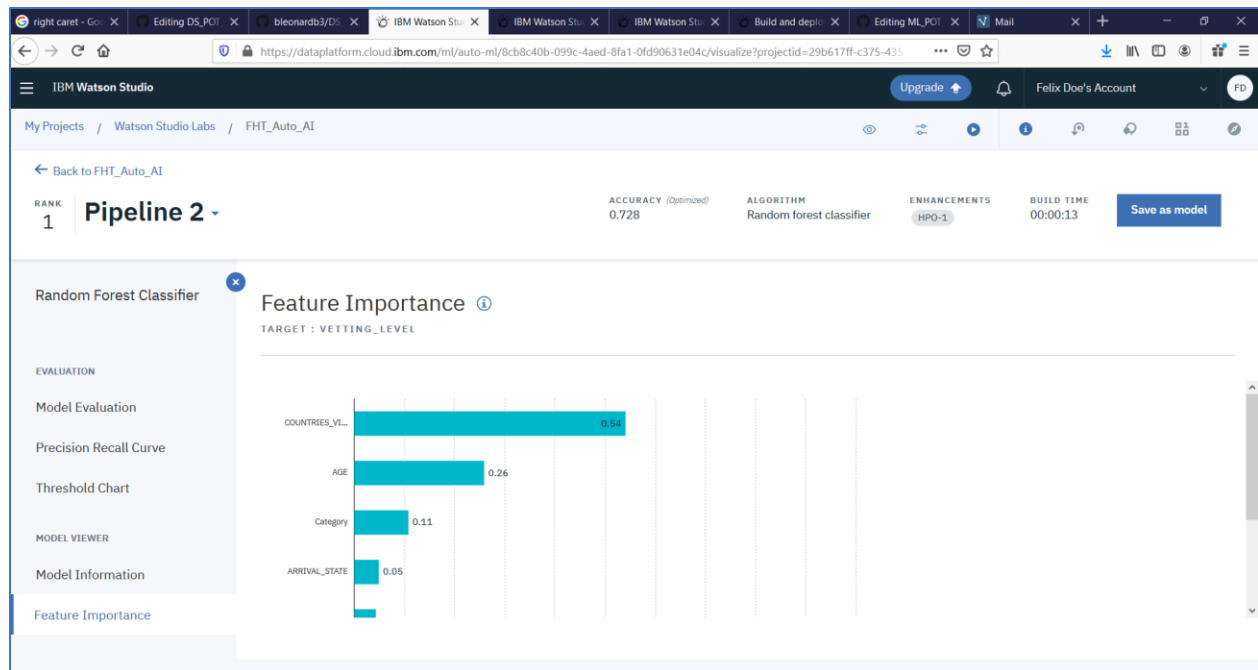
10. Scores are displayed for different metrics for both the training sample and the holdout sample. The holdout sample is 10%. Click on **Pipeline 2**.



11. The model evaluation metric for the holdout sample is displayed. On the left are options for additional information. Click on **Feature Importance**.

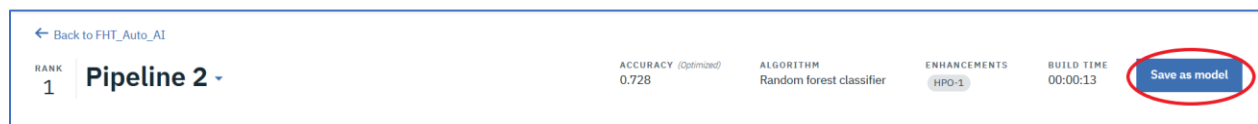


12. According to the **Feature Importance**, the **COUNTRIES_VISITED_COUNT** feature is the most important, followed by the **Age** feature and the **Category** feature.



Step 3: Save and Deploy the Model

1. Click on **Save as model**



2. Optionally change the default name and click **Save**.

×

Save as model

Save this model as a project asset so you can deploy, train, and test it.

Model name

FHT_Auto_AI - P2 RandomForestClassifierEstimator

Description (optional)

Description of model

Associated project

Cancel Save

3. A message is displayed showing the model was successfully saved. Click on **View in project**.

IBM Watson Studio

Upgrade

Felix Doe's Account

FD

My Projects / Watson Studio Labs / FHT_Auto_AI

Back to FHT_Auto_AI

RANK 1 Pipeline 2

ACCURACY (Optimized) 0.728

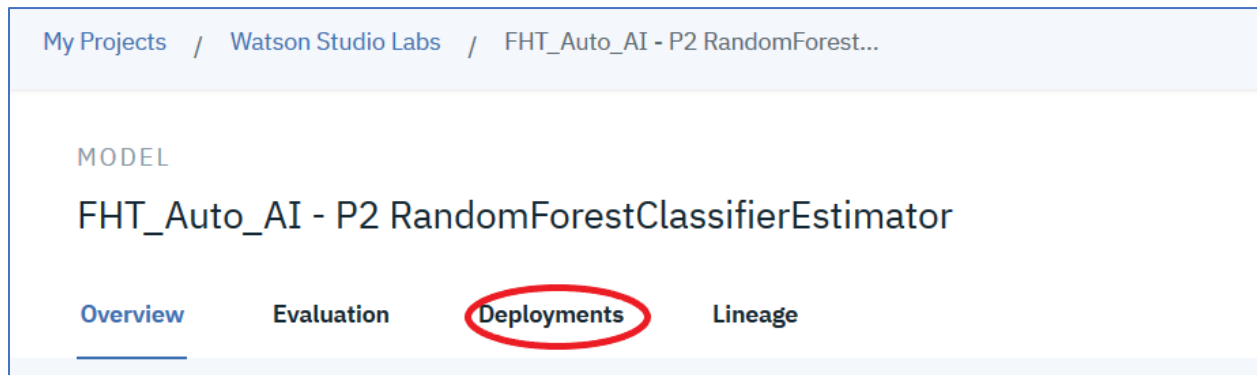
ALGORITHM Random forest classifier

ENHANCEMENT HPO-1

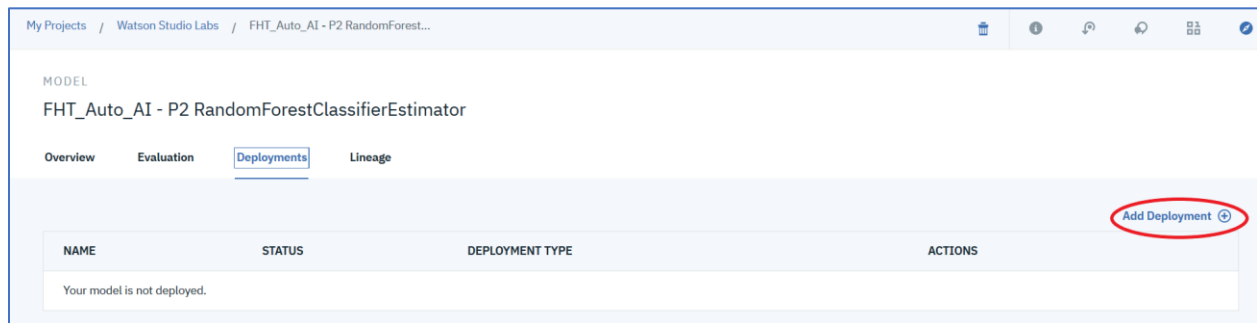
FHT_Auto_AI - P2 RandomForestClassif...
view in project

Random Forest Classifier

4. Click on **Deployments**.



5. Click on **Add Deployment**



6. Enter a **Name**, optionally a **Description** and click **Save**.

The screenshot shows the 'Create Deployment' form in the Watson Studio interface. The breadcrumb navigation is 'My Projects / Watson Studio Labs / FHT_Auto_AI - P2 RandomForest...'. The form has a title 'Create Deployment' and a section 'Define deployment details'. It contains three input fields: 'Name' (with the value 'FHT_AutoAI_Deployed' entered and circled in red), 'Description' (with a placeholder 'Deployment description'), and 'Deployment type' (with 'Web service' selected and circled in red). At the bottom right of the form, there are two buttons: 'Cancel' and 'Save', with the 'Save' button circled in red.

7. The model is successfully deployed in the IBM Cloud.

MODEL			
FHT_AutoAI - P3 XGBClassifierEstimator			
Overview Evaluation Deployments Lineage			
Add Deployment			
NAME	STATUS	DEPLOYMENT TYPE	ACTIONS
FHT_AutoAI_Deployed	ready	Web Service	

8. Click on **FHT_AutoAI_Deployed**.

MODEL			
FHT_AutoAI - P3 XGBClassifierEstimator			
Overview Evaluation Deployments Lineage			
Add Deployment			
NAME	STATUS	DEPLOYMENT TYPE	ACTIONS
FHT_AutoAI_Deployed	ready	Web Service	

9. Click on **Test**.

FHT_AutoAI_Deployed	
Overview	Implementation Test
Deployment	
Name	FHT_AutoAI_Deployed
Type	Web Service
Deployment ID	48ea36dd-0780-49c1-a0c8-f73f20546784
Status	ready
Asset type	model
Asset name	FHT_AutoAI - P2 XGBClassifierEstimator
Machine learning service	WatsonMachineLearning
Created	20 Jul 2019 09:34am

10. Enter the following values:

PASSPORT_COUNTRY – **Ghana**

COUNTRIES_VISITED_COUNT – **3**

ARRIVAL_STATE – **OH**

DEPARTURE_AIRPORT_COUNTRY_CODE – **RU**

AGE – **20**

Category – Sports/Travel

Overview

Implementation

Test

Enter input data

PASSPORT_COUNTRY

Ghana

COUNTRIES_VISITED_COUNT

3

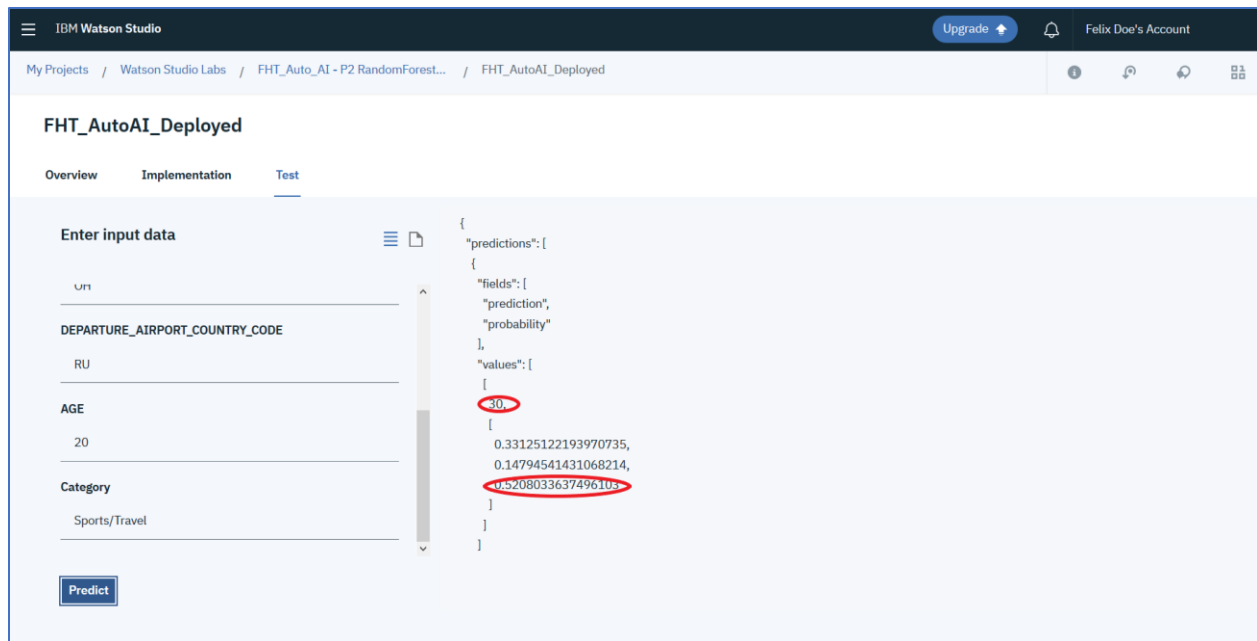
ARRIVAL_STATE

OH

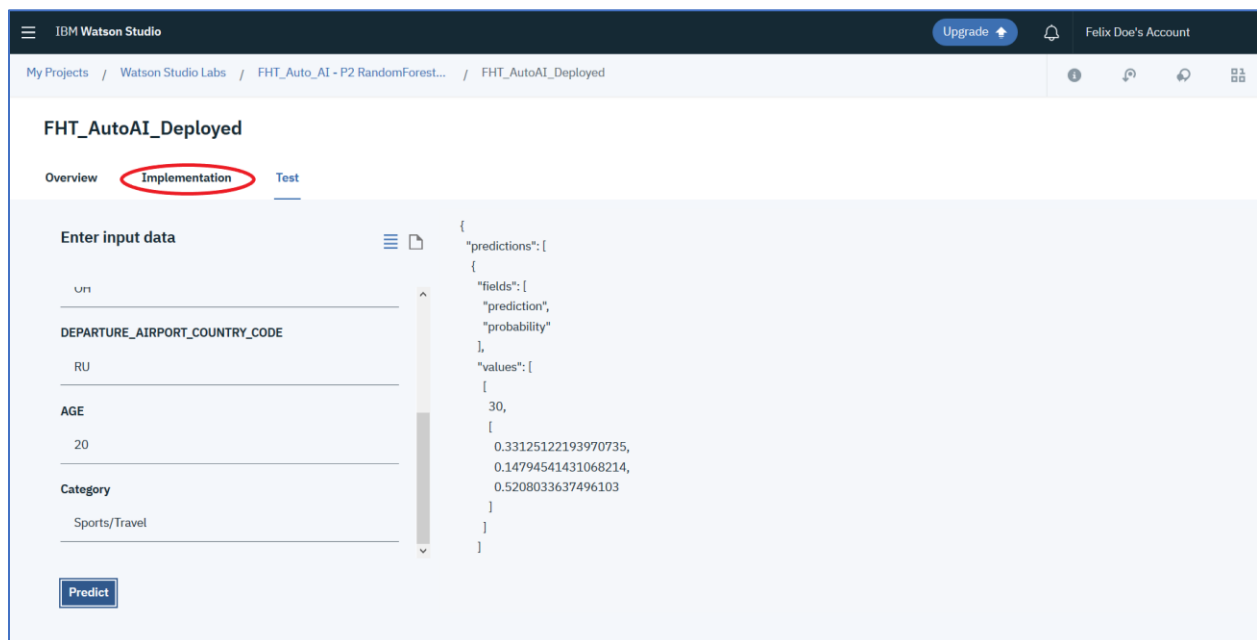
DEPARTURE_AIRPORT_COUNTRY_CODE

Predict

11. The prediction is **30** (low risk) of trafficking with 52% confidence.



12. Click on Implementation



13. The Implementation panel provides information for the application developers to invoke the deployed model. It includes sample code in various programming languages and the scoring endpoint to be used when invoking the web service. Open Windows Notepad to copy and paste the scoring endpoint, or just leave this panel available to cut and paste the scoring endpoint. We will need the scoring endpoint in the next section.

FHT_AutoAI_Deployed

Overview **Implementation** Test

Implementation View API Specification

Scoring End-point	https://us-south.ml.cloud.ibm.com/v4/deployments/e020c2c0-7c2a-43fe-9204-58f6398990bb/predictions
Authorization: Bearer <token>	Review the WML authentication documentation for details about generating IAM tokens.
ML-Instance-ID	The "ML-Instance-ID" HTTP header must be populated with the WML instance id, which can be obtained as described here .
Content-type: application/json	Required if the request body is sent in JSON format.

Code Snippets

cURL Java JavaScript Python Scala

```
# TODO: manually define and pass values to be scored below
curl -X POST --header 'Content-Type: application/json' --header 'Accept: application/json' --header 'Authorization: Bearer $IAM_TOKEN' --header 'ML-Instance-ID: $ML_INSTANCE_ID' -d '{"input_data": [{"fields": [ARRAY_OF_FEA
```

Step 4: Deploy a simple web front-end to invoke the Watson Machine Learning service

This section provides an example of a simple Python Flask web front-end application that invokes the Female Human Trafficking risk prediction deployed model, demonstrating embedding machine learning in a web app. You will click on a link below that will deploy the sample Python web application into your IBM Cloud account. A toolchain will be set up for continuous delivery of the application. The application code will be cloned from a public Git repository into a private Git repo in your account that will be set up as part of the toolchain. Each time you commit changes to the repo, the app will be built and deployed.

The toolchain uses tools that are part of the Continuous Delivery service. If an instance of that service isn't already in your account, when you click **Deploy**, it is automatically added with the free [Lite](#) plan selected.

The steps below guide you in configuring the application to connect to your Watson Machine Learning service, and to update the application with the deployed model's scoring endpoint.




1. Click on the **Deploy to IBM Cloud** link below to deploy a sample Python Flask web application into your IBM Cloud account. Note you may get this message – “*An IBM Cloud account is required. To get started, click Log In or Sign Up at the top of this page*”. If you get this message, click on **Log In**.

[Deploy to IBM Cloud](#)

2. In the **Select Region**, make sure that the region is Dallas. If not, change it to Dallas, and wait for the screen to refresh.

Toolchains / Create a toolchain /

Deploy to IBM Cloud: FHT app Cancel Create

Tools:   




Template Info:
GIT URL
<https://github.com/open-toolchain/default>
GIT BRANCH

Toolchain Name:
FHT-20190830004555115

Select Region:
Dallas

Select a resource group:
Default
[Select a CF Organization \(deprecated\)](#)

Tool Integrations




 Git Repos and Issue Tracking
 Delivery Pipeline **Required**
 More tools

3. Click **Create**.

IBM Cloud Search resources and offerings...

Toolchains / Create a toolchain /

Deploy to IBM Cloud: FHT app Cancel Create

Tools:   




Template Info:
GIT URL
<https://github.com/open-toolchain/default>
GIT BRANCH

Toolchain Name:
FHT-20200114043102623

Select Region:
Dallas

Select a resource group:
Default
[Select a CF Organization \(deprecated\)](#)

Tool Integrations




 Git Repos and Issue Tracking
 Delivery Pipeline **Required**
 More tools




FEEDBACK
ASK A QUESTION

4. Scroll down and click **New+** to create an API key.

Toolchains / Create a toolchain /

Deploy to IBM Cloud: FHT app Cancel Create

Tools:   

 Git Repos and Issue Tracking  Delivery Pipeline **Required**  More tools

The Delivery Pipeline automates continuous deployment.

App name:

IBM Cloud API key: New

The value is required.

5. Click **OK**.

Create a new API key with full access

Warning: This will create a new API key that allows anyone who has it the ability to do anything you could do. You can improve your security posture by using the [IAM UI to create a service ID API key](#) that limits access to only what your pipeline requires, and then pasting that into the template UI instead.

For more information on API keys and access see the [IAM documentation](#).




Key will be called: API Key for FHT-20200114043611885




☐ Save this key in a secrets store for reuse

Cancel OK

6. Please wait until the Region, Organization, and Space are filled in. **Note that these must match the corresponding Region (Dallas), Organization, and Space where the FHT_WML_Model_Deployed is deployed.** If this is not the case, try a different Region. Click on **Create**.

Toolchains / Create a toolchain / **Deploy to IBM Cloud: FHT app** Cancel **Create**

Tools:   

 Git Repos and Issue Tracking  **Delivery Pipeline**  More tools


The Delivery Pipeline automates continuous deployment.

App name: ⓘ

IBM Cloud API key: ⓘ Now +

Region: ⓘ Organization: ⓘ Space: ⓘ


7. Your app is being created! To watch the pipeline deploy your app, click **Delivery Pipeline**.


 **FHT-20190516022609200**


Resource Group: Default Location: Dallas [Add tags](#)


✓ **Your app is being created! Quick start:** To watch the pipeline deploy your app, click **Delivery Pipeline**. After the app is created, click **Delivery Pipeline**.

THINK **CODE** **DELIVER**

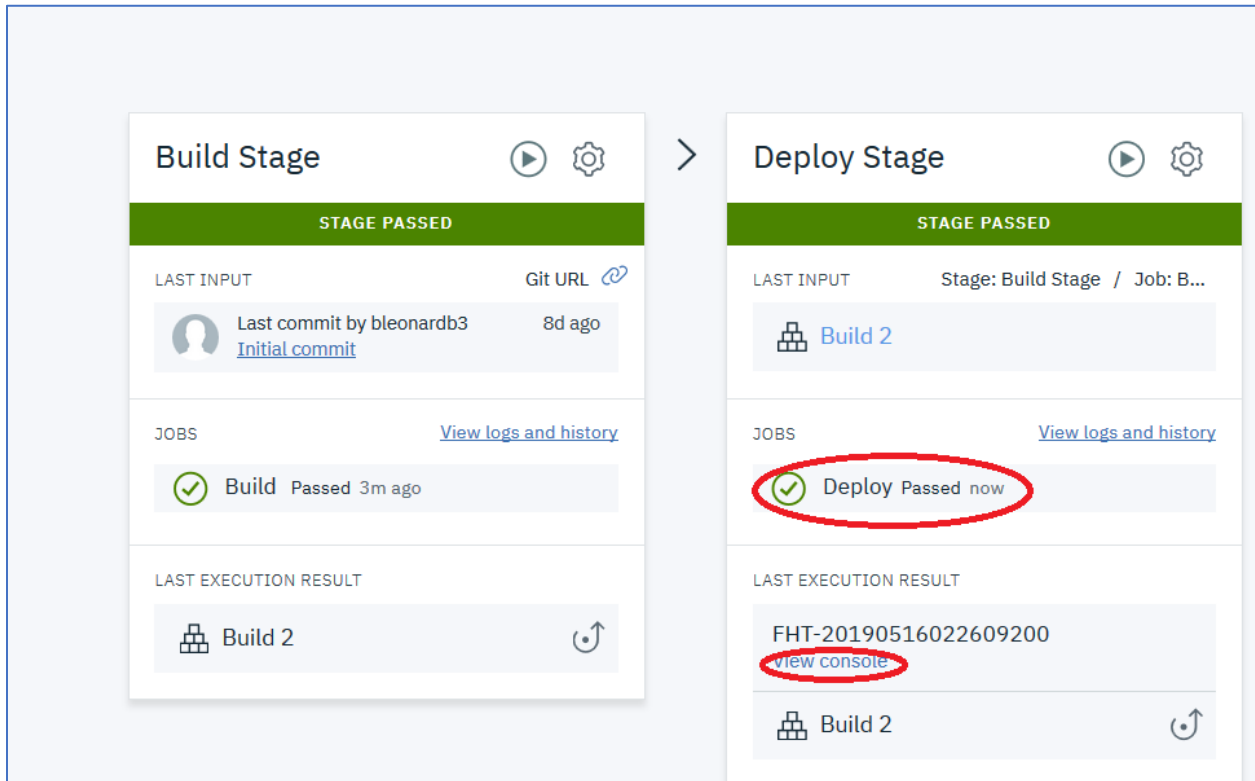
 **Issues**
FHT-2019051602260...
✓ Configured

 **Git**
FHT-2019051602260...
✓ Configured

 **Delivery Pipeline**
FHT-2019051602260...
✓ Configured

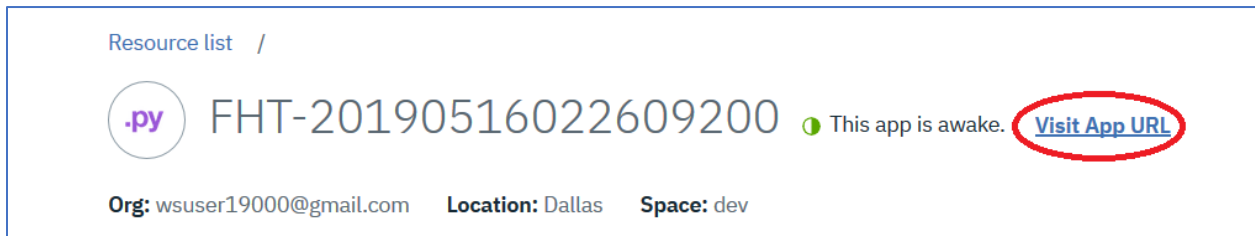
 **Eclipse Orion Web IDE**
✓ Configured

8. After the app is deployed successfully (should say Deploy Passed in the Deploy stage-may take about 2 minutes), view the running app by clicking on **View Console**



The screenshot shows two side-by-side panels for the 'Build Stage' and 'Deploy Stage'. Both panels have a green header indicating 'STAGE PASSED'. The 'Build Stage' panel shows the last input as 'Last commit by bleonardb3' and a job 'Build' that passed 3 minutes ago. The 'Deploy Stage' panel shows the last input as 'Build 2' and a job 'Deploy' that passed 'now'. In the 'Deploy Stage' panel, the 'Deploy Passed now' status is circled in red. Below the jobs, the 'LAST EXECUTION RESULT' for 'Build 2' is shown, with a 'view console' link circled in red.

9. Click on **Visit App URL**



The screenshot shows a resource list for a .py application. The application ID is 'FHT-20190516022609200'. The status is 'This app is awake.' and the 'Visit App URL' link is circled in red. Below the application ID, the organization is 'wsuser19000@gmail.com', the location is 'Dallas', and the space is 'dev'.

10. The web form collecting the FHT data should appear. Note that the application is not functional until we connect it to the Watson Machine Learning service so if you Submit you will get an error! Close the FHT Prediction browser tab.

FHT Prediction

To determine the trafficking risk prediction, please enter the following:

Categories

Sports/Travel

Age:

Number of countries visited:

Passport Country

☐ Ghana

☐ Brazil

☐ Pakistan

☐ Bangladesh

☐ Haiti

☐ India

Arrival State:

Departure Country:

Submit

11. We are now going to connect the application to the Watson Machine Learning service that was created earlier. Scroll down until you see the Connections panel. Click on **Create Connection**.

Connections

No services are connected to this app

You can bind a service:

Create connection

12. You should see at least 2 services listed, a Cloud Object Storage service, and a Watson Machine Learning service. Point the cursor on the **Machine Learning** service for your application, and then click on **Connect**.

Connect Existing Compatible Service

Search compatible services

Default

10Items per page | 1-2 of 2 items

1 of 1 pages < 1 >

SERVICES	RESOURCE GROUP	PLAN	SERVICE OFFERING
cloud-object-storage-ue	Default	Lite	Cloud Object Storage
WatsonMachineLearning	Default	Lite	Machine Learning

Connect

13. A **Connect IAM-enabled service** pop up will appear. Click **Connect**.

Connect IAM-Enabled Service

To connect, you can customize the ServiceID and access role used for this binding. Restaging your app is required to connect this service and may result in application downtime.

Access Role for Connection

Writer

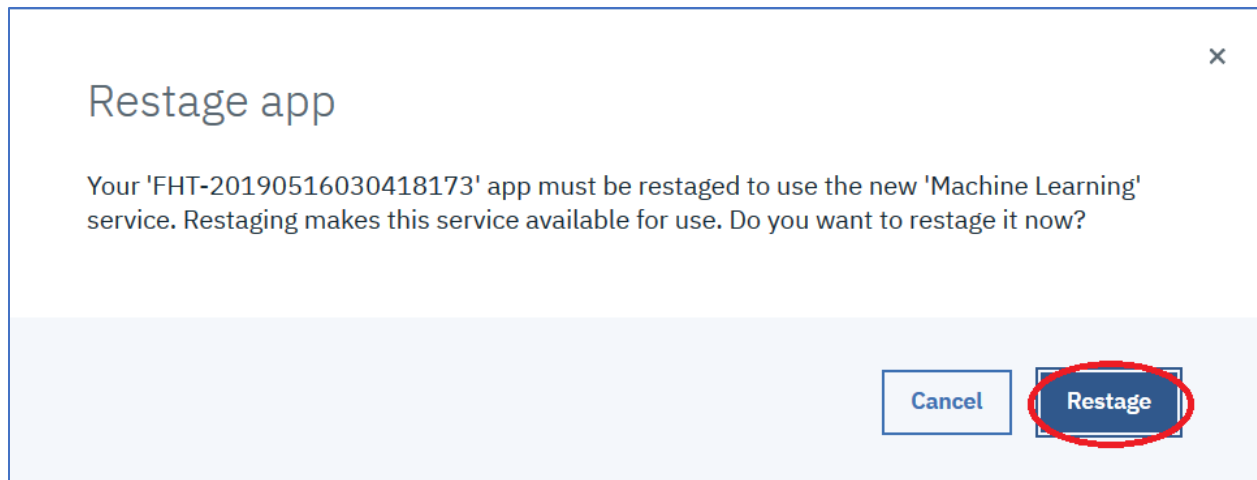
Service ID for Connection (Optional)


Auto Generate

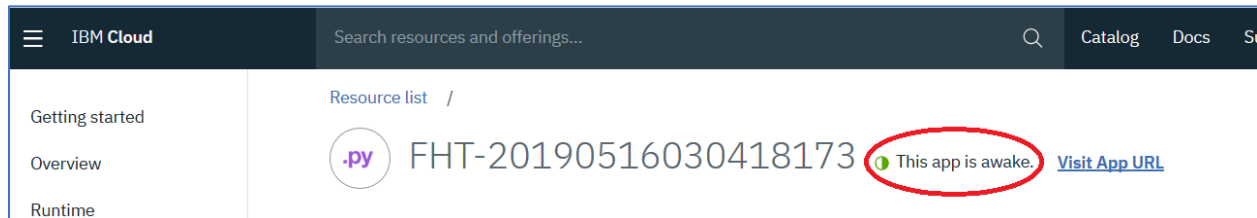
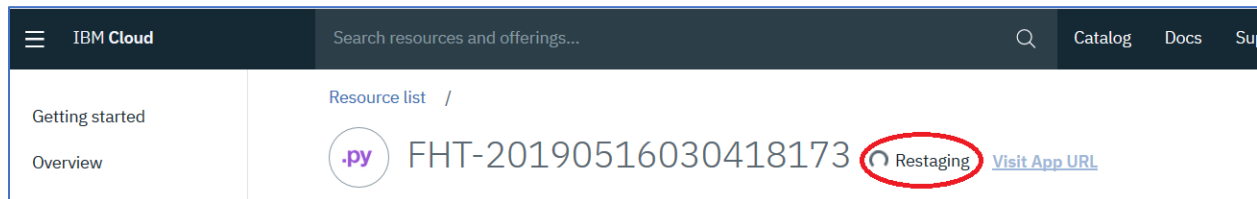
Cancel


Connect

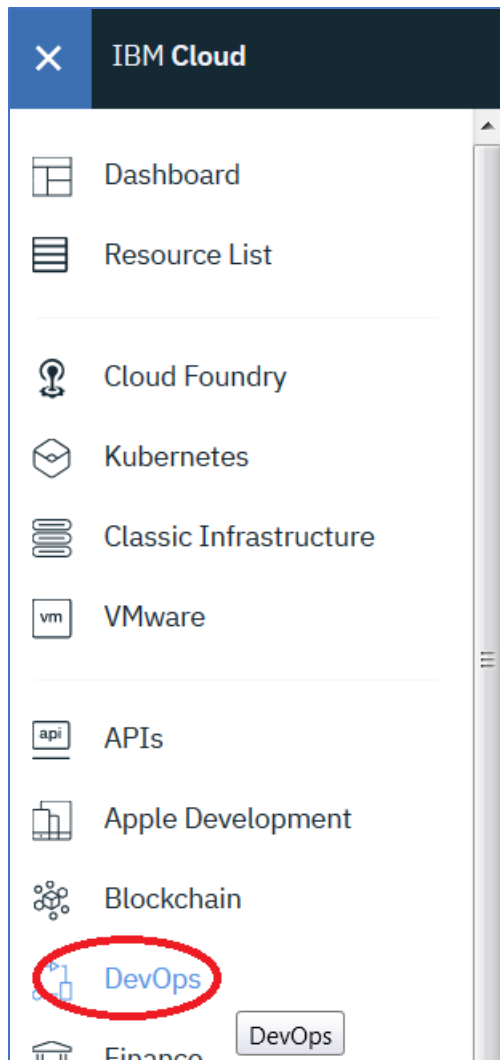
14. A **Restage app** pop up will appear. Click on **Restage**.



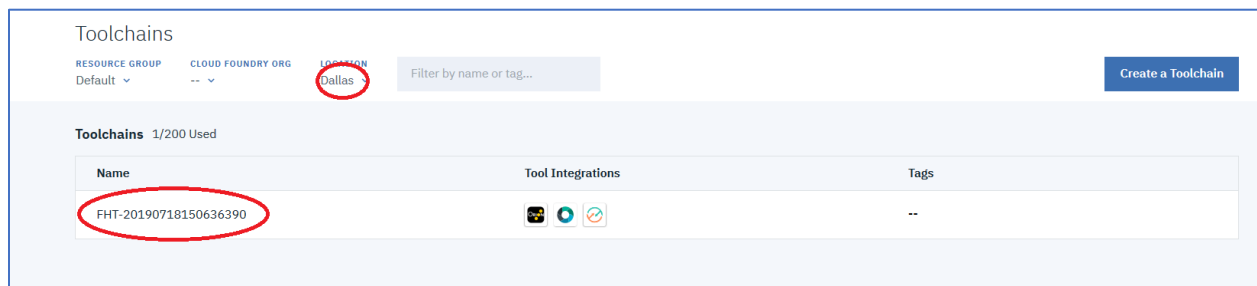
15. Wait for the application status to change from **Restaging** to  This app is awake , or something similar.



16. We now have tied the web application to the Watson Machine Learning service. Note that the Watson Machine Learning service could have more than one deployed model available to select and then embed in the web application. We now need to copy the scoring endpoint, which we previously copied and pasted into Notepad, and paste it in the web application code. Click on the  icon and click on DevOps in the pulldown to navigate to the Toolchain.




17. We are now going to paste the scoring endpoint into the application code. Click on the Toolchain (FHT-2019xxxxx below). If no toolchain appears, switch the location to Dallas.



18. Click on the Eclipse Orion Web IDE. The IDE will enable the editing of the source code to update the scoring endpoint url.


Toolchains /

 FHT-20190516030418173 [Visit App URL](#)


Resource Group: Default Location: Washington DC [Add tags](#)

✓ Your app is being created! Quick start: To watch the pipeline deploy your app, click **Delivery Pipeline**. After the app is deployed, you app.


THINK



Issues
FHT-2019051603041...
✓ Configured

CODE


Git
FHT-2019051603041...
✓ Configured

DELIVER


Delivery Pipeline
FHT-2019051603041...
✓ Configured


Eclipse Orion Web IDE


19. Click on the FHT.py file. This is a python source file.

FHT-20190516030418173

▼ README.md

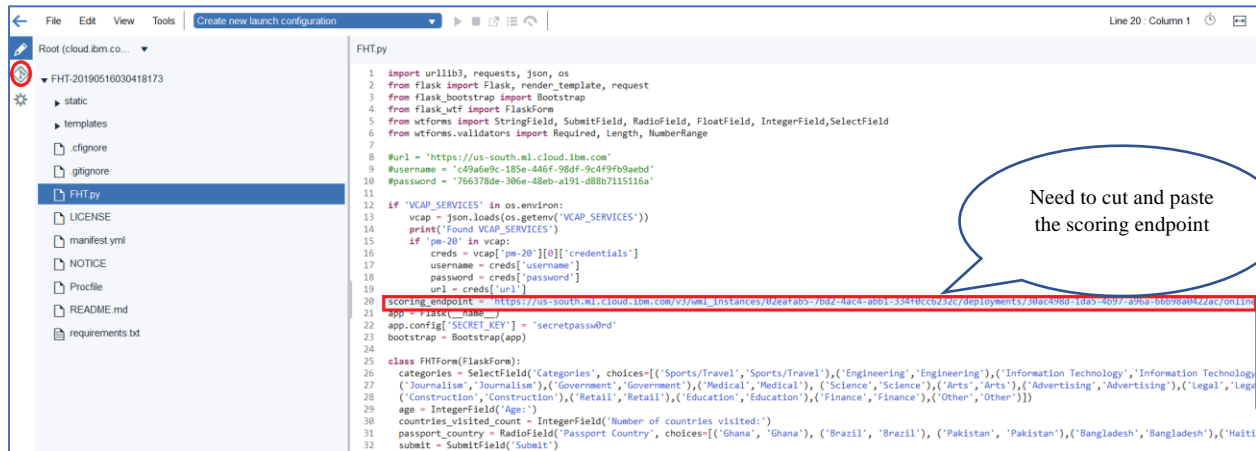
This repository contains a Python Flask program that invokes a Watson Machine Learning service to predict trafficking risk based on travel itinerary data

▼ FHT-20190516030418173

Name	Date Modified	Size
static	5/15/2019, 11:37:15 PM	--
templates	5/15/2019, 11:37:15 PM	--
.cfnignore	5/15/2019, 11:37:16 PM	1 KB
.gitignore	5/15/2019, 11:37:16 PM	1 KB
 FHT.py	5/15/2019, 11:37:15 PM	4 KB
LICENSE	5/15/2019, 11:37:15 PM	12 KB
manifest.yml	5/15/2019, 11:37:15 PM	1 KB
NOTICE	5/15/2019, 11:37:15 PM	1 KB
Procfile	5/15/2019, 11:37:15 PM	1 KB
README.md	5/15/2019, 11:37:15 PM	1 KB
requirements.txt	5/15/2019, 11:37:15 PM	1 KB

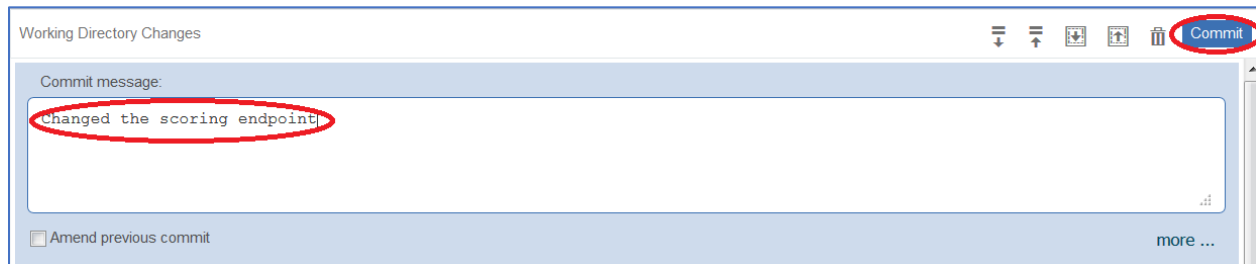
20. Go back to the Notepad file and copy the scoring endpoint to the clipboard. Look around line 20 in the FHT.py file for the “scoring endpoint =”. Select the scoring endpoint value in line 20 (starting with https:// may want to use Shift-End to get to the end of the line, and then back up one space to not select the endpoint quote – if you do just make sure to put it back in). Enter Ctrl-V to paste the new scoring endpoint from your Notepad

file. Enter Ctrl-S or File > Save to save the file. Then click on the  icon on the top left.



```
1 import urllib3, requests, json, os
2 from flask import Flask, render_template, request
3 from flask_bootstrap import Bootstrap
4 from flask_wtf import FlaskForm
5 from wtforms import StringField, SubmitField, RadioField, FloatField, IntegerField, SelectField
6 from wtforms.validators import Required, Length, NumberRange
7
8 #url = 'https://us-south-1.amazonaws.com'
9 #username = 'c49a6e9c-185e-446f-9c4f9f9b9aebd'
10 #password = '766378de-306e-48eb-a191-d88b7115116a'
11
12 if 'VCAP_SERVICES' in os.environ:
13     vcap = json.loads(os.getenv('VCAP_SERVICES'))
14     print('Found VCAP_SERVICES')
15     if 'pm-20' in vcap:
16         creds = vcap['pm-20'][0]['credentials']
17         username = creds['username']
18         password = creds['password']
19         url = creds['url']
20 scoring_endpoint = 'https://us-south-1.amazonaws.com/v2/vm_instances/02e2a7ab5-7bd2-4ac4-b0b1-334f8c6232c/deployments/0a149bd-1das-409f-a9ba-b8b9a9a42aac/online'
21 app = Flask(__name__)
22 app.config['SECRET_KEY'] = 'secretpassw@rd'
23 bootstrap = Bootstrap(app)
24
25 class FHTForm(FlaskForm):
26     categories = SelectField('Categories', choices=[('Sports/Travel', 'Sports/Travel'), ('Engineering', 'Engineering'), ('Information Technology', 'Information Technology'), ('Journalism', 'Journalism'), ('Government', 'Government'), ('Medical', 'Medical'), ('Science', 'Science'), ('Arts', 'Arts'), ('Advertising', 'Advertising'), ('Legal', 'Legal'), ('Construction', 'Construction'), ('Retail', 'Retail'), ('Education', 'Education'), ('Finance', 'Finance'), ('Other', 'Other')])
27     age = IntegerField('Age:')
28     countries_visited_count = IntegerField('Number of countries visited:')
29     passport_country = RadioField('Passport Country', choices=[('Ghana', 'Ghana'), ('Brazil', 'Brazil'), ('Pakistan', 'Pakistan'), ('Bangladesh', 'Bangladesh'), ('Haiti', 'Haiti')])
30     submit = SubmitField('Submit')
```

21. The next step is to commit the change to the git repository. Enter “Changed the Scoring Endpoint” in the Enter Commit Message field, and then click on **Commit**.



Working Directory Changes

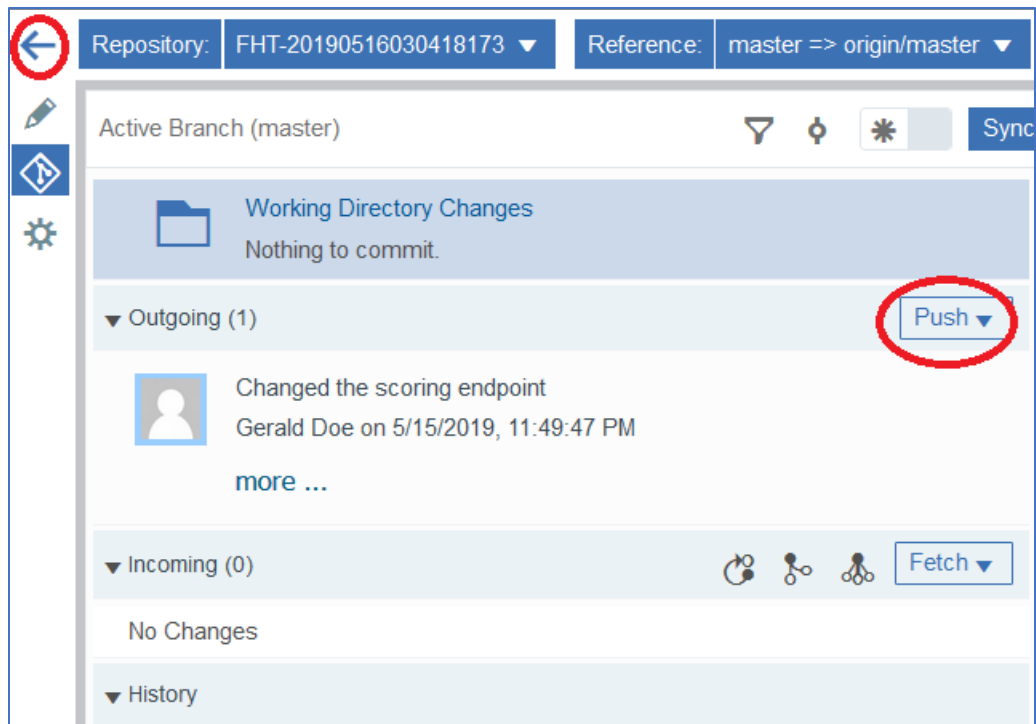
Commit message:

Changed the scoring endpoint


☐ Amend previous commit

Commit

22. Then click on **Push** to push the changes to the central Git repo which will start the build and deploy of the application. Click on the left arrow to return to the Toolchain.




23. Click on the **Delivery Pipeline** to view status of the deployment as before. Refresh the screen if the stage status doesn't change.

 FHT-20190516030418173 [Visit App URL](#)


Resource Group: Default Location: Washington DC [Add tags](#)

✓ Your app is being created! Quick start: To watch the pipeline deploy your app, click **Delivery Pipeline** app.


THINK



Issues
FHT-2019051603041...
✓ Configured

CODE


Git
FHT-2019051603041...
✓ Configured

DELIVER


Delivery Pipeline
FHT-2019051603041...
✓ Configured


Eclipse Orion Web IDE
✓ Configured

24. Once the Deployment status shows **Deploy passed now** it shouldn't take longer than 2 minutes (reload the browser in case the UI didn't update after 2 minutes). Click on **View Console**.


Build Stage

▶ ⚙

STAGE PASSED

LAST INPUT

Git URL [🔗](#)



Last commit by Gerald Doe
[Changed the scoring endpoint](#)

6m ago


JOBSDivView logs and history

✔


Build

Passed 4m ago

LAST EXECUTION RESULT



Build 2



>


Deploy Stage

▶ ⚙

STAGE PASSED

LAST INPUT

Stage: Build Stage / Job: B...



Build 2

JOBSDivView logs and history

✔


Deploy

Passed 2m ago


LAST EXECUTION RESULT

FHT-20190516030418173

[View console](#)




Build 2



25. Click on **Visit App URL**.

Resource list /



FHT-20190516030418173

🟢

This app is awake.

[Visit App URL](#)

Org: wsuser19000@gmail.com Location: Dallas Space: dev

26. The web form should appear. Enter data in all the fields and click on the **Submit** button.

The screenshot shows a web browser window with the URL <https://fht-20190722013541020-zany-gerenuk.mybluemix.net>. The page title is "FHT Prediction". The main content area contains the following form fields and controls:

- A heading: "To determine the trafficking risk prediction, please enter the following:"
- A "Categories" dropdown menu with "Sports/Travel" selected.
- An "Age" input field with the value "20".
- A "Number of countries visited" input field with the value "0".
- A "Passport Country" section with radio buttons for "Ghana", "Brazil", "Pakistan", "Bangladesh", "Haiti", and "India". "Ghana" is selected.
- An "Arrival State" input field with the value "OH".
- A "Departure Country" input field with the value "RU".
- A "Submit" button.

Red circles are drawn around the "Sports/Travel" dropdown, the "Age" field, the "0" in the "Number of countries visited" field, the "Ghana" radio button, the "OH" in the "Arrival State" field, the "RU" in the "Departure Country" field, and the "Submit" button.

27. You should see something similar to the following depending on the values of the input fields that you entered. Click on the **Try Again!**, if you want to experiment with different inputs.

prediction:low risk
probability: 0.52080336375
Try Again!