

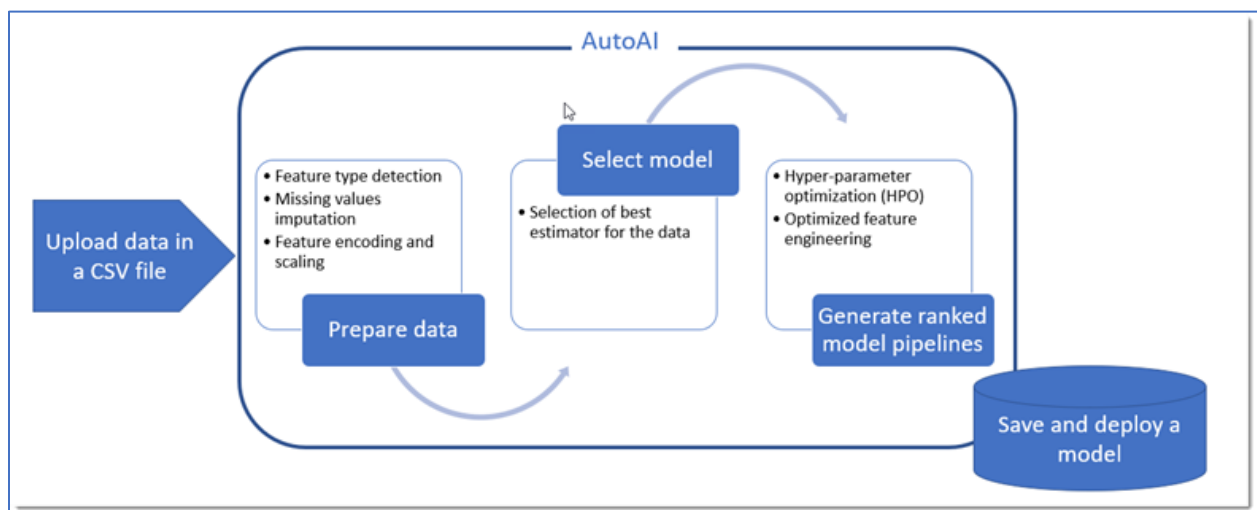
# AutoAI + DevOps

## 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 (optional) 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](#)

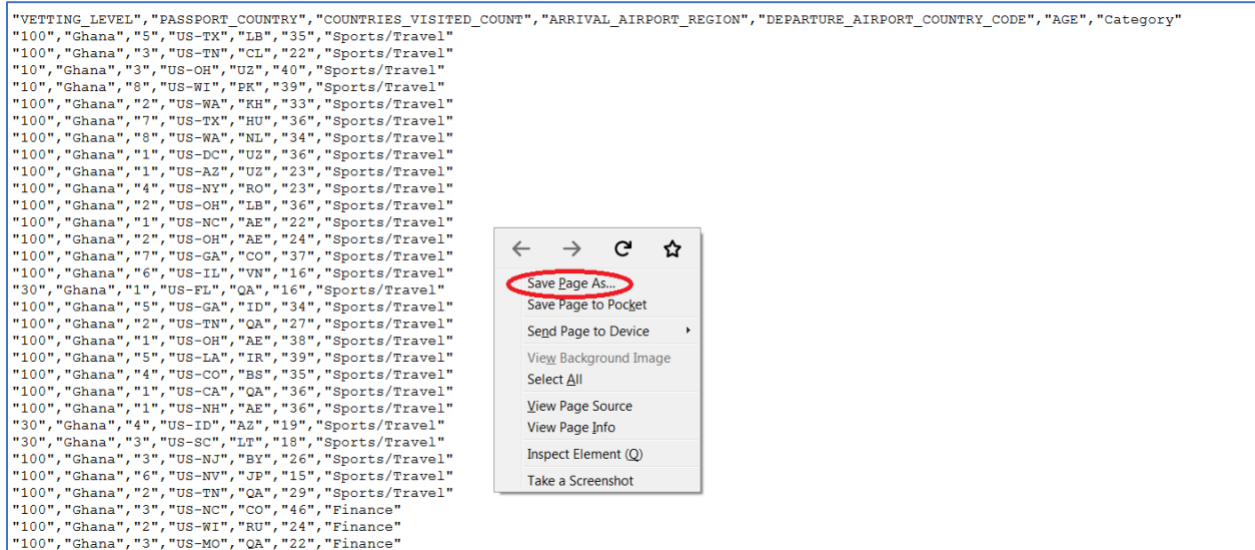
We will perform the following steps in this lab:

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

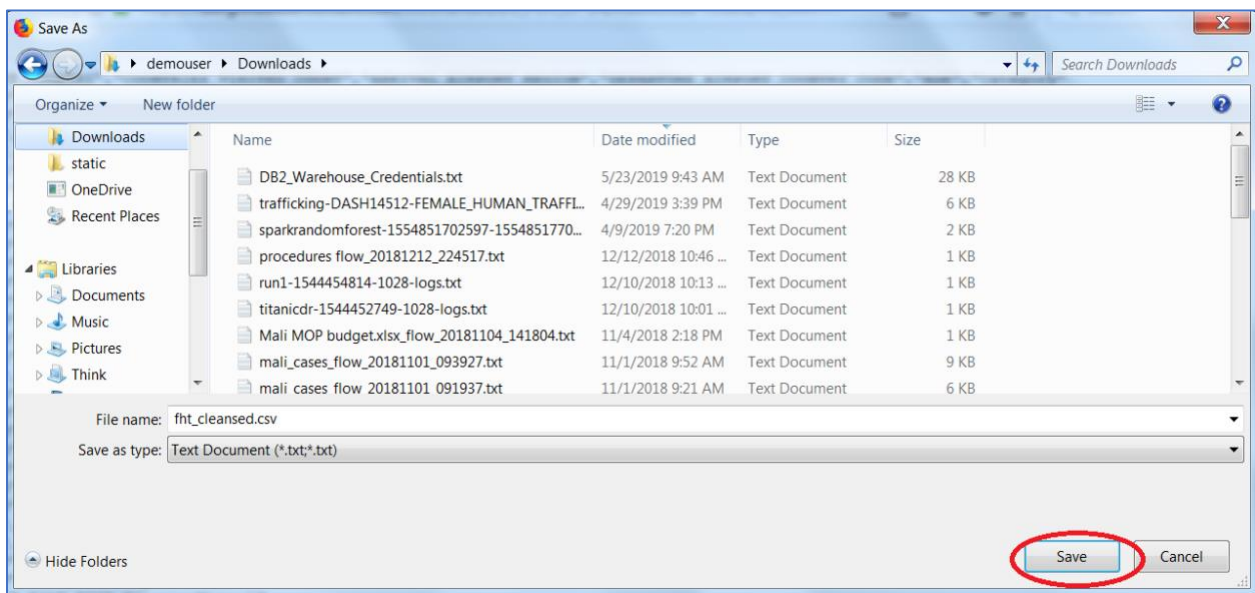
4. Test the model
5. Deploy a simple web front-end application and connecting it to the deployed model using an IBM Cloud DevOps toolchain.

## 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...**



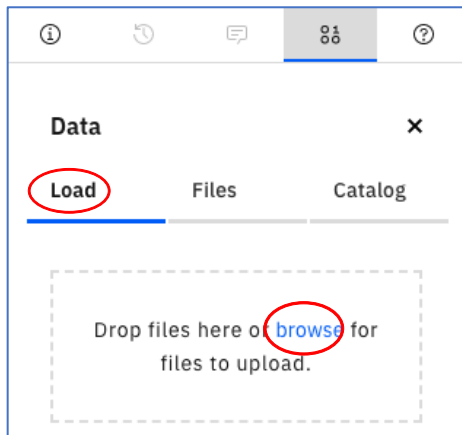
3. Click on **Save**.



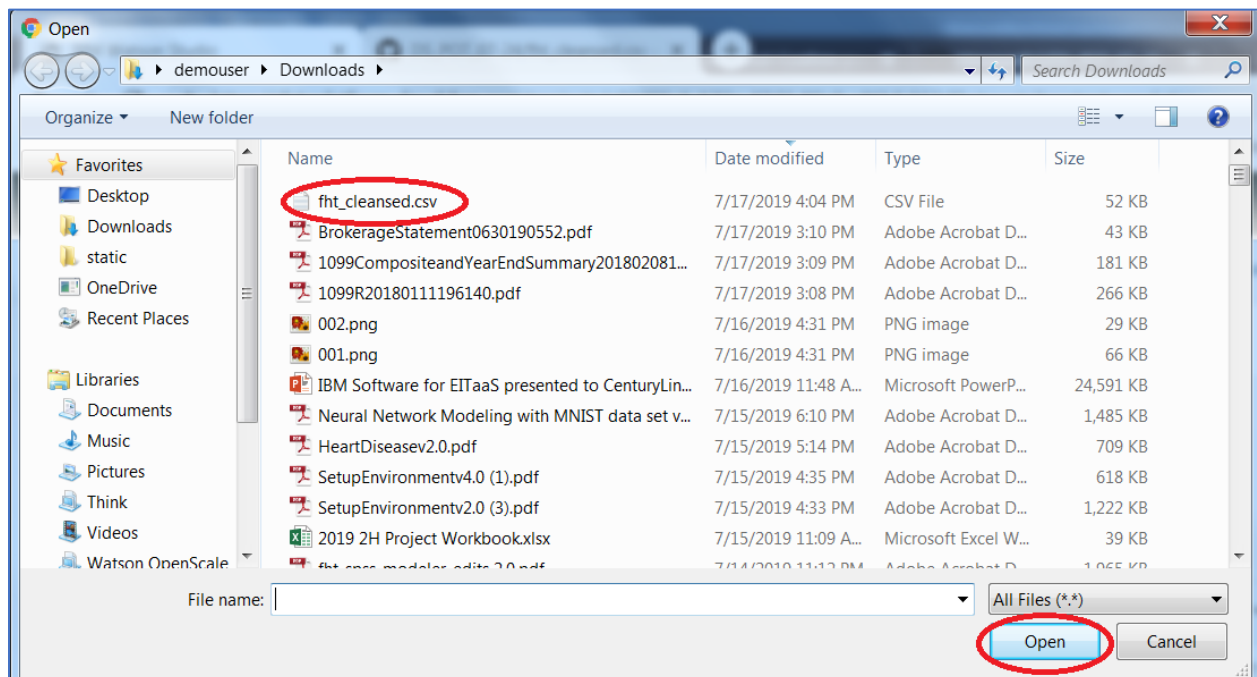
4. Go back to your Watson Studio Labs project. Click on the  icon.



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



6. Go to the folder where the fht\_cleansed.csv file is stored. Select the fht\_cleansed.csv file and then click **Open**.



7. The fht\_cleansed.csv file is now added as a Data Asset.

▼ Data assets				
0 assets selected.				
<input type="checkbox"/>	Name	Type	Created by	Last modified ↓
<input type="checkbox"/>	CSV fht_cleansed.csv	Data Asset	FCTO Labs	Aug 03, 2020, 10:18 AM

## Step 2: Add an AutoAI Experiment.

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

My projects / Watson Studio Labs					Launch IDE ▼	Add to project +
Overview	Assets	Environments	Jobs	Deployments	Access Control	Settings

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

My projects / Watson Studio Labs					Launch IDE ▼	Add to project +
Overview	Assets	Environments	Jobs	Deployments	Access Control	Settings

3. Click on **AutoAI Experiment**.

### Choose asset type

Available asset types

Data	Connection	Connected data	AutoAI experiment
Notebook	Dashboard	Visual Recognition ...	Natural Language Cl...
Watson Machine Lea...	Deep learning experi...	Data Replication	Modeler flow
Data Refinery flow	Streams flow	Decision Optimizatio...	

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

New AutoAI experiment

**Define details**

**From blank** **From sample**

Name \*

FHT AutoAI

Description

Description of AutoAI experiment

**Associate services**

Watson Machine Learning Service Instance \*

pm-20-aa

Compute configuration \* ⓘ

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 AutoAI

Configure AutoAI experiment

FHT AutoAI

**Add data source**

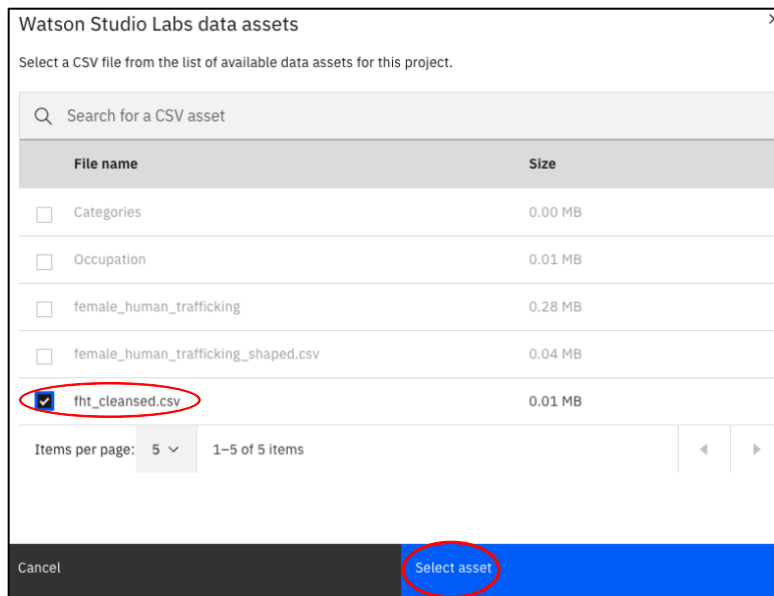


**Add data source**  
Drop a .csv file here or [browse](#) for a file to upload.

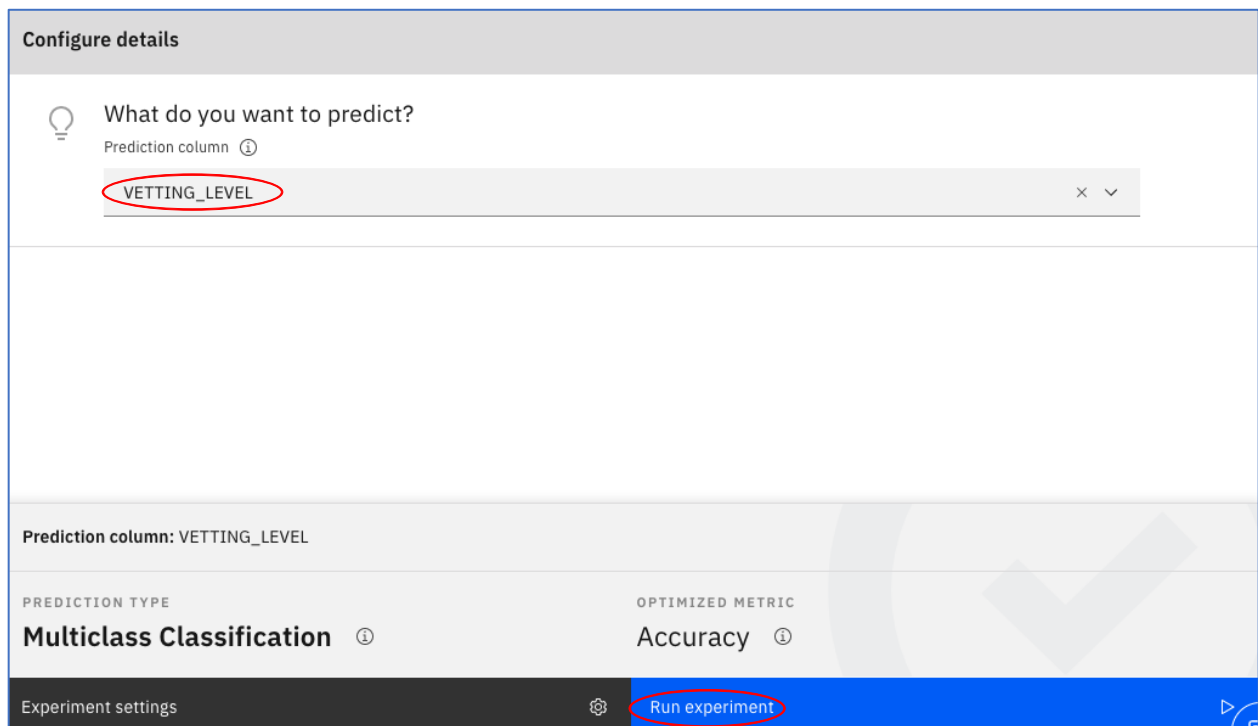
— OR —

Select from project

6. Select the **fht\_cleansed.csv** dataset and click on **Select asset**.



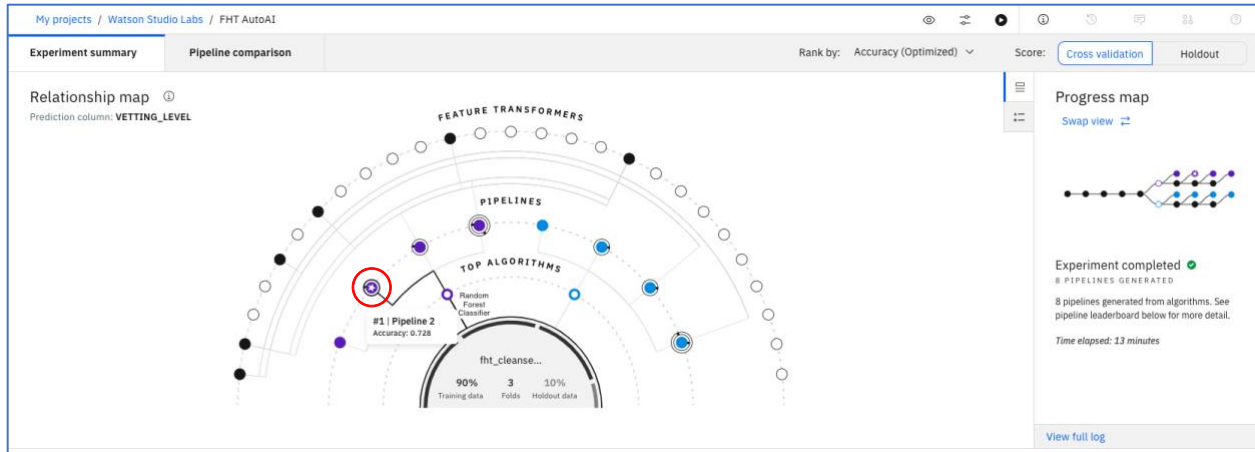
7. Select **VETTING\_LEVEL** as the column to predict, leave the default for **OPTIMIZED METRIC** and click on **Run Experiment**. Note the system scanned the **VETTING\_LEVEL** values to determine that a **Multiclass Classification** was the **PREDICTION TYPE**.



8. It will take several minutes for the eight alternative pipelines to be analyzed. 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 try to improve the performance metric. The fourth pipeline will do another hyperparameter optimization including the newly derived features. The next 4 pipelines do the same thing for the second best algorithm. Note, you can move ahead at any point after the first pipeline has been created. We are going to proceed as if Pipeline 2 is the best ranked pipeline. Yours could be different.

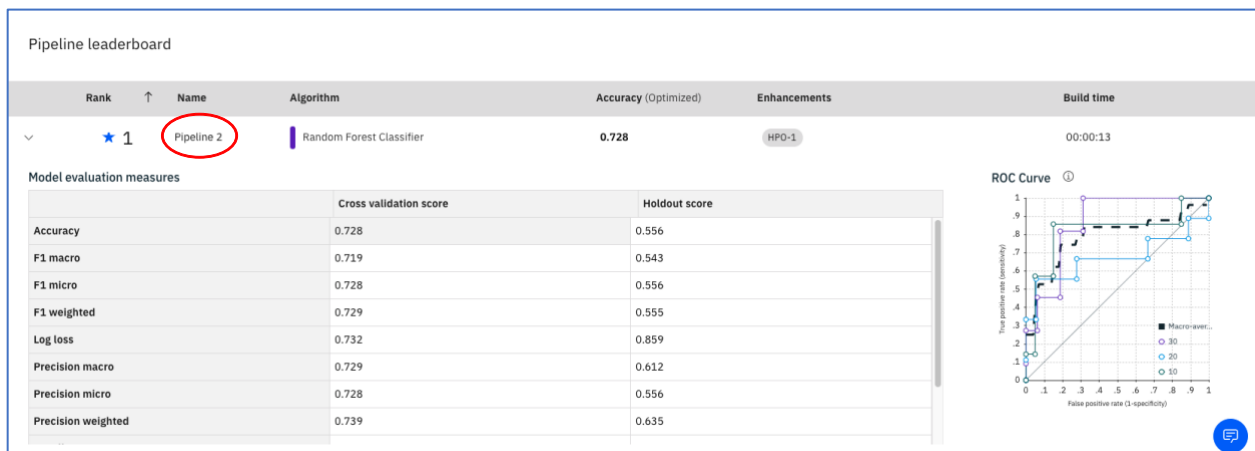
9. Scroll down to view the Pipeline leaderboard. Click on the right caret next to Pipeline 2.



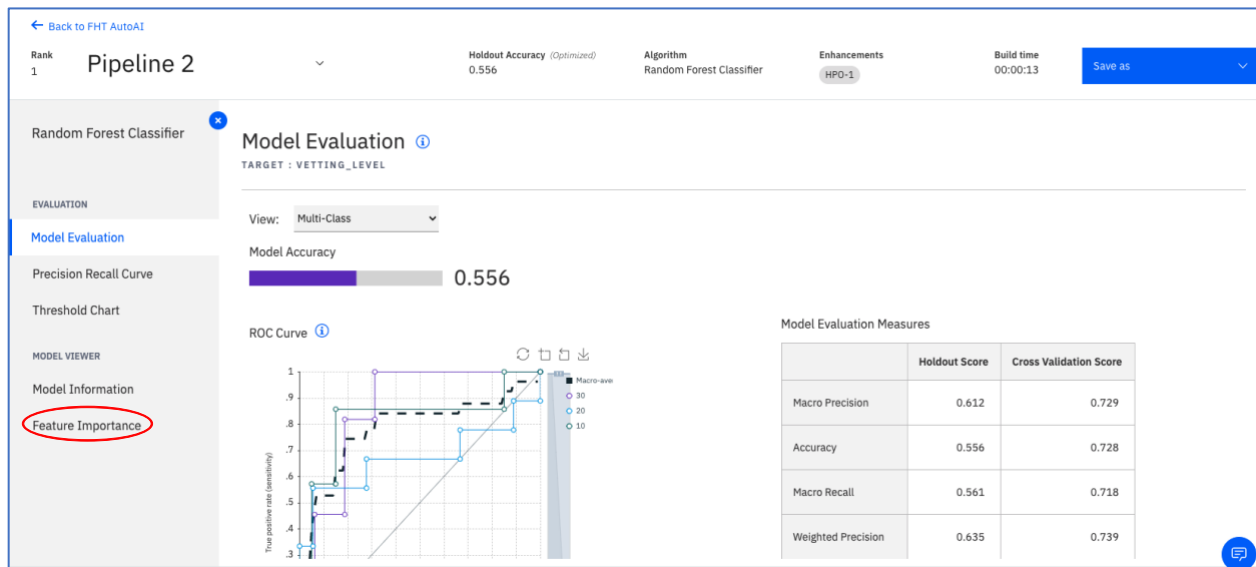
Pipeline leaderboard

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 8	XGB Classifier	0.716	HPO-1 FE HPO-2	00:02:56
>	4	Pipeline 6	XGB Classifier	0.711	HPO-1	00:01:00

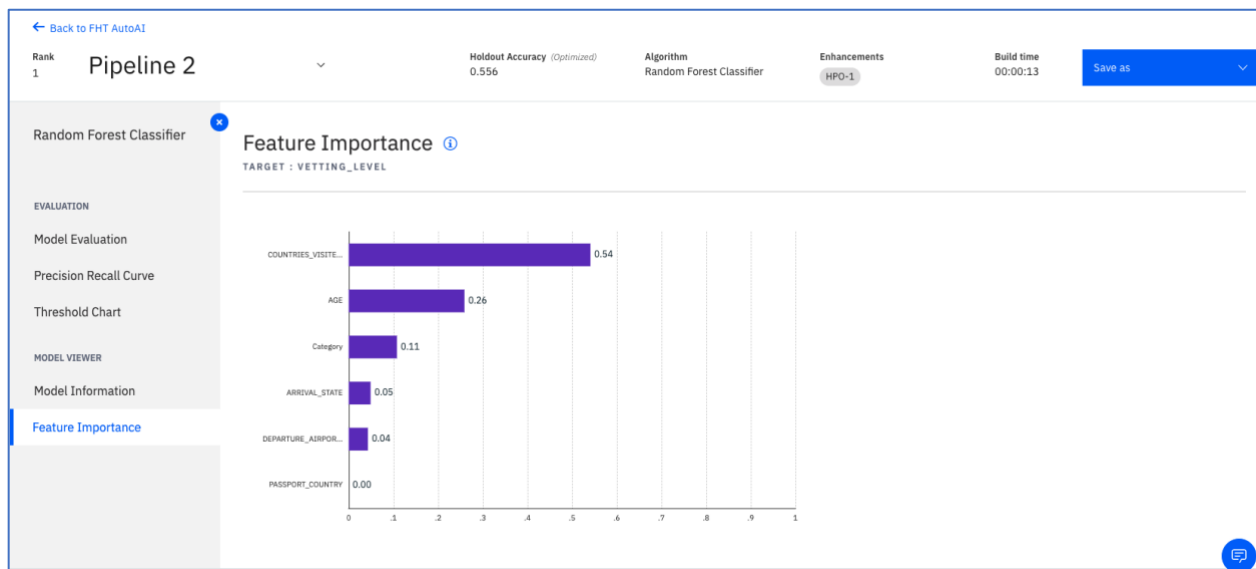
10. Scores are displayed for different metrics for both the training sample and the holdout sample. The holdout sample is 10%. Click on **Pipeline 2**.



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



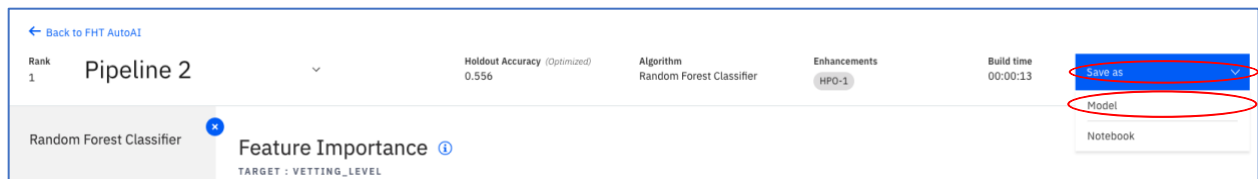
12. According to the **Feature Importance**, the **COUNTRIES\_VISITED\_COUNT** feature is the most important, followed by the **Age** feature and the **Category** feature.



### Step 3: Save and Deploy the Model

1. **Click on Save as.** AutoAI can generate both a machine learning model as well as a script in the form of a Jupyter Notebook to build the model. Feel free to explore the Notebook. When ready, **click on 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 AutoAI - P2 RandomForestClassifierEstimator

Description (optional)

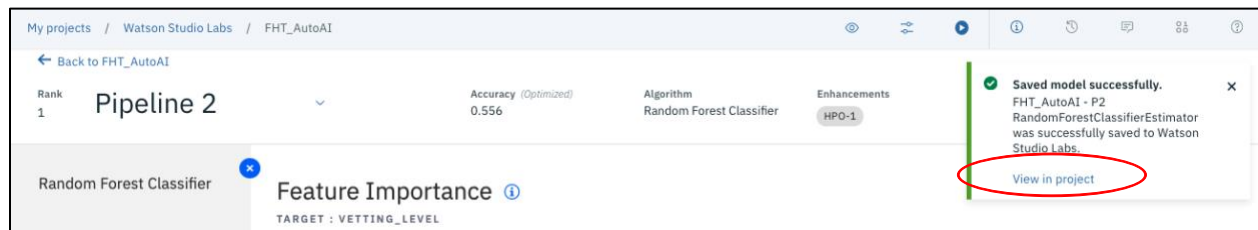
Description of model

Associated project

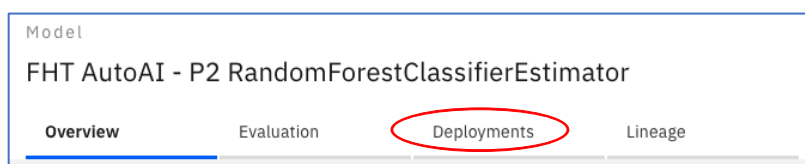
Watson Studio Labs

Cancel Save

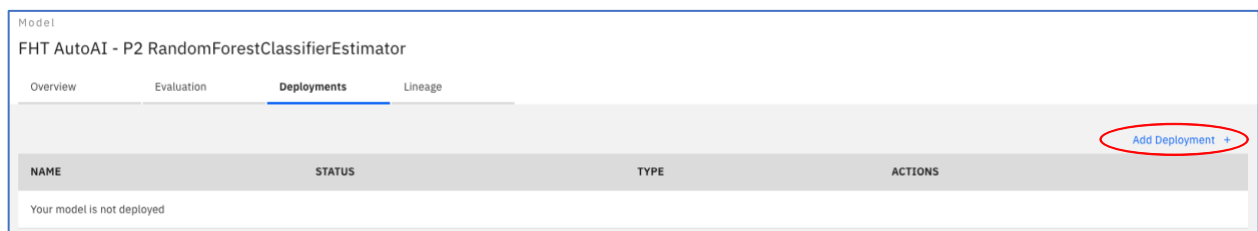
3. A message is displayed showing the model was successfully saved. Click on **View in project**.



4. Click on **Deployments**.



5. Click on **Add Deployment**



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

Create Deployment

Define deployment details

Name  
FHT\_AutoAI\_Deployment

Description  
Deployment description

Deployment type  
☒ Web service

Cancel Save

7. Once the status is “Ready,” the model is successfully deployed in the IBM Cloud.

Model

FHT AutoAI - P2 RandomForestClassifierEstimator

Overview Evaluation Deployments Lineage

Add Deployment +

NAME	STATUS	TYPE	ACTIONS
FHT_AutoAI_Deployment	Ready	Web Service	:

8. Click on **FHT\_AutoAI\_Deployment**.

Model

FHT AutoAI - P2 RandomForestClassifierEstimator

Overview Evaluation Deployments Lineage

Add Deployment +

NAME	STATUS	TYPE	ACTIONS
FHT_AutoAI_Deployment	Ready	Web Service	:

9. Click on **Test**.

FHT\_AutoAI\_Deployment

Overview Implementation Test

Deployment

Name FHT\_AutoAI\_Deployment

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**

Click **Predict**.

**FHT\_AutoAI\_Deployment**


Overview Implementation **Test**

**Enter input data**  

PASSPORT\_COUNTRY

COUNTRIES\_VISITED\_COUNT

ARRIVAL\_STATE

DEPARTURE\_AIRPORT\_COUNTRY\_CODE  
 

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

**FHT\_AutoAI\_Deployment**

Overview Implementation **Test**

**Enter input data**

PASSPORT\_COUNTRY  
Ghana

COUNTRIES\_VISITED\_COUNT  
3

ARRIVAL\_STATE  
OH

DEPARTURE\_AIRPORT\_COUNTRY\_CODE  
RU

**Predict**

```
{
  "predictions": [
    {
      "fields": [
        "prediction",
        "probability"
      ],
      "values": [
        [
          30,
          [
            0.33125122193970735,
            0.14794541431068214,
            0.5208033637496103
          ]
        ]
      ]
    }
  ]
}
```

12. Click on Implementation

**FHT\_AutoAI\_Deployment**

Overview **Implementation** Test

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\_Deployment**

Overview **Implementation** Test

**Implementation** [View API Specification](#)

Scoring End-point: <https://us-south.ml.cloud.ibm.com/v4/deployments/81cd1af7-7dfc-4e34-855b-b965a149b3df/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": [{"fi
```

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

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 Flash 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

Create

About

Toolchain Name:

FHT-20200803162833229

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 on **Delivery Pipeline**.

Toolchains / Create a toolchain /

## Deploy to IBM Cloud: FHT app

**Create** About


Toolchain Name:  
FHT-20200803162833229


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

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

The Delivery Pipeline automates continuous deployment.

App name:  
FHT-20200426031240254

IBM Cloud API key:  
IBM Cloud API key  **New +**

The value is required.

Region Organization Space

The IBM Cloud CF region The IBM Cloud CF org The IBM Cloud CF space

**Cancel** **Create**

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-20200426031240254

☐ 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**.

The Delivery Pipeline automates continuous deployment.

App name:

FHT-20200426031240254

IBM Cloud API key:

.....

New

+

Region

Dallas

Organization

fctolabs99@gmail.com

Space

dev

Cancel

Create

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

IBM Cloud

Toolchains / FHT-20200803163056950 Add tags

Visit App URL Details Actions...

Overview

Connections Manage

Think

Code

Deliver

Issues FHT-202008031630... ✓ Configured

Git FHT-202008031630... ✓ Configured

Delivery Pipeline FHT-202008031630... Success

Eclipse Orion Web IDE ✓ Configured

FEEDBACK ASK A QUESTION

8. It may take a moment for this section to finish setting up. 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**

Toolchains / FHT-20200803163056950 / FHT-20200803163056950 / FHT-20200803163056950 | Delivery Pipeline

Build Stage

STAGE PASSED

LAST INPUT Git URL

Last commit by bleonardb3 340d ago Add files via upload

JOBS View logs and history

Build Passed 8m ago

LAST EXECUTION RESULT

Build 1

Deploy Stage

STAGE PASSED

LAST INPUT Stage: Build Stage / Job: B...

Build 1

JOBS View logs and history

Deploy Passed 5m ago

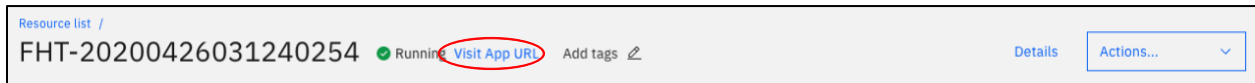
LAST EXECUTION RESULT

FHT-20200803163056950 View console

Build 1



9. Click on **Visit App URL**



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.

A screenshot of the 'FHT Prediction' web form. The form has a dark header bar with the title 'FHT Prediction'. Below the header, the text 'To determine the trafficking risk prediction, please enter the following:' is displayed. The form contains several input fields: a 'Categories' dropdown menu with 'Sports/Travel' selected, an 'Age:' text input field, a 'Number of countries visited:' text input field, a 'Passport Country' section with radio button options for Ghana, Brazil, Pakistan, Bangladesh, Haiti, and India, an 'Arrival State:' text input field, and a 'Departure Country:' text input field. At the bottom of the form is a 'Submit' button.

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**.

IBM Cloud

Resource list / FHT-20200803163056950 Running [Visit App URL](#) [Add tags](#) [Details](#) [Actions...](#)

Getting started

**Overview**

Runtime

Connections

Logs

API Management

Autoscaling

Availability Monitoring

**Instances** [Edit](#)

Health **100%**  
1/1 instance(s) are running

Instances 1

MB memory per instance

0 256 128

**Runtime**

Python

128  
Total MB allocation

128 MB still available  
Used Free

**Runtime cost**  
Current and estimated cost excludes connected services.

**\$0.00**  
Current charges for billing period

**\$0.00**  
Estimated total for billing period  
Aug 1, 2020 - Aug 31, 2020

[Full details](#)

**Connections (0)**

No services are connected to this app

[Create connection](#)

FEEDBACK

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**.

Resource list / FHT-20200803163056950 Running [Visit App URL](#) [Add tags](#) [Details](#) [Actions...](#)

Getting started

Overview

Runtime

**Connections**

Logs


API Management

Autoscaling

Availability Monitoring

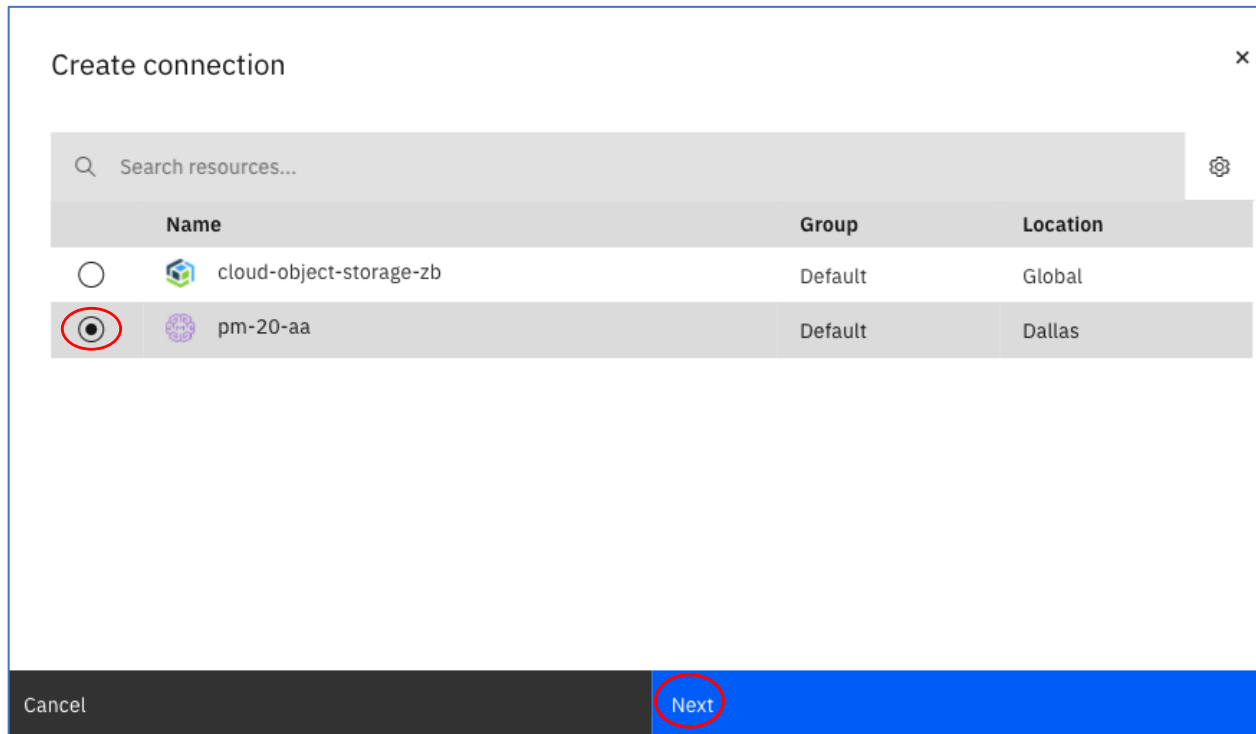
Filter items

[Create connection](#)

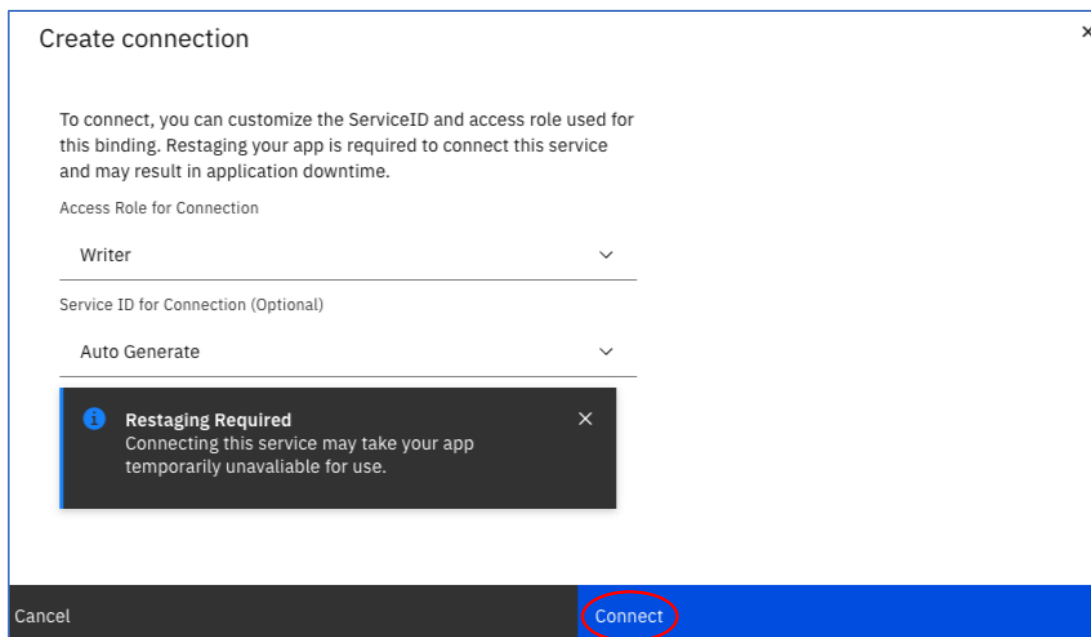
Name	Type
 <p><b>No connections</b></p> <p>No connected resources. Create a connection to bind this app to another resource.</p>	

FEEDBACK

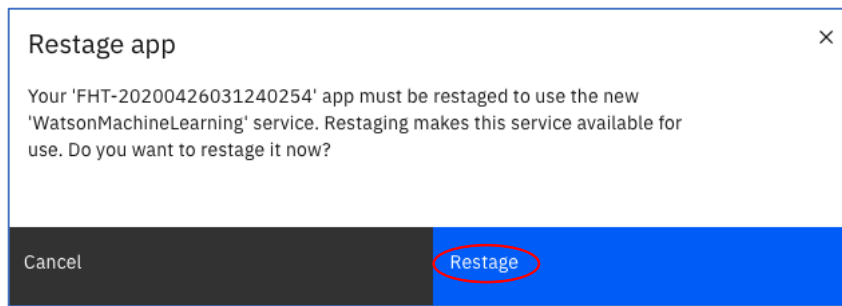
13. You should see at least 2 services listed, a Cloud Object Storage service, and a Watson Machine Learning service. Point the cursor on the **Machine Learning** service (in the screenshot called “pm-20-aa”) for your application for the “Next” button to appear, and then click on **Next**.



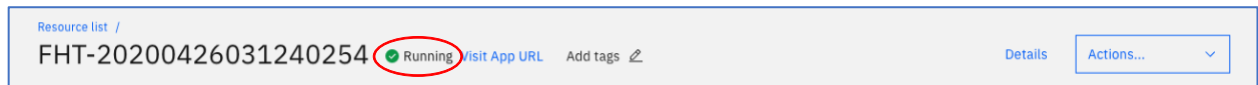
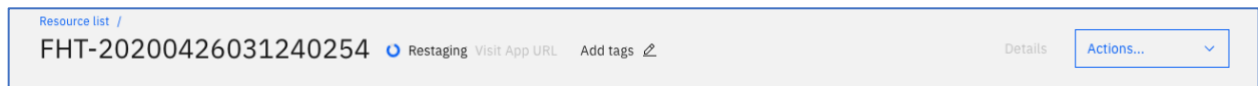
14. A **Connect connection** pop up will appear. We can use this page to set access role for the connection. Click **Connect**.




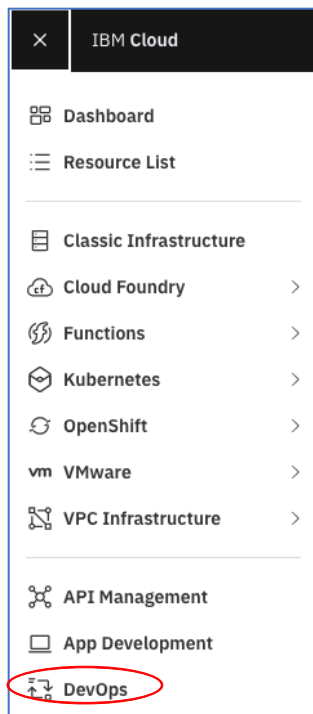
15. A **Restage** app pop up will appear. Click on **Restage**.



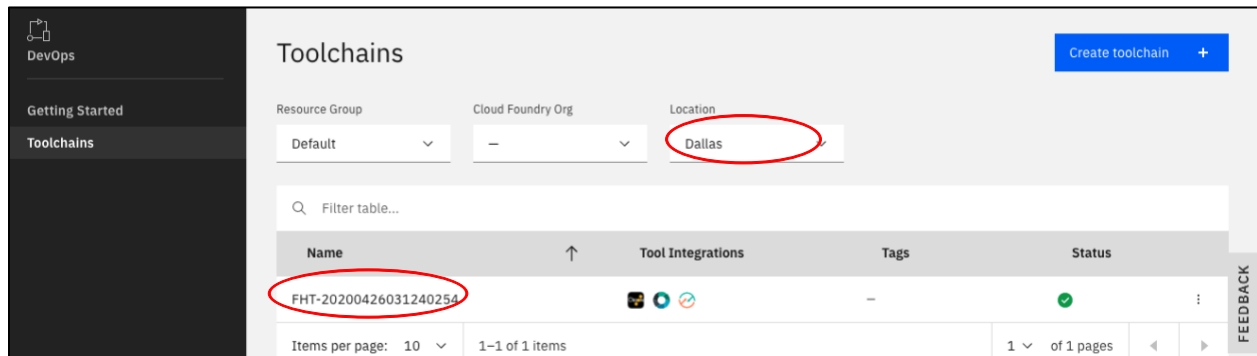
16. Wait for the application status to change from **Restaging** to  **Running**, or something similar.



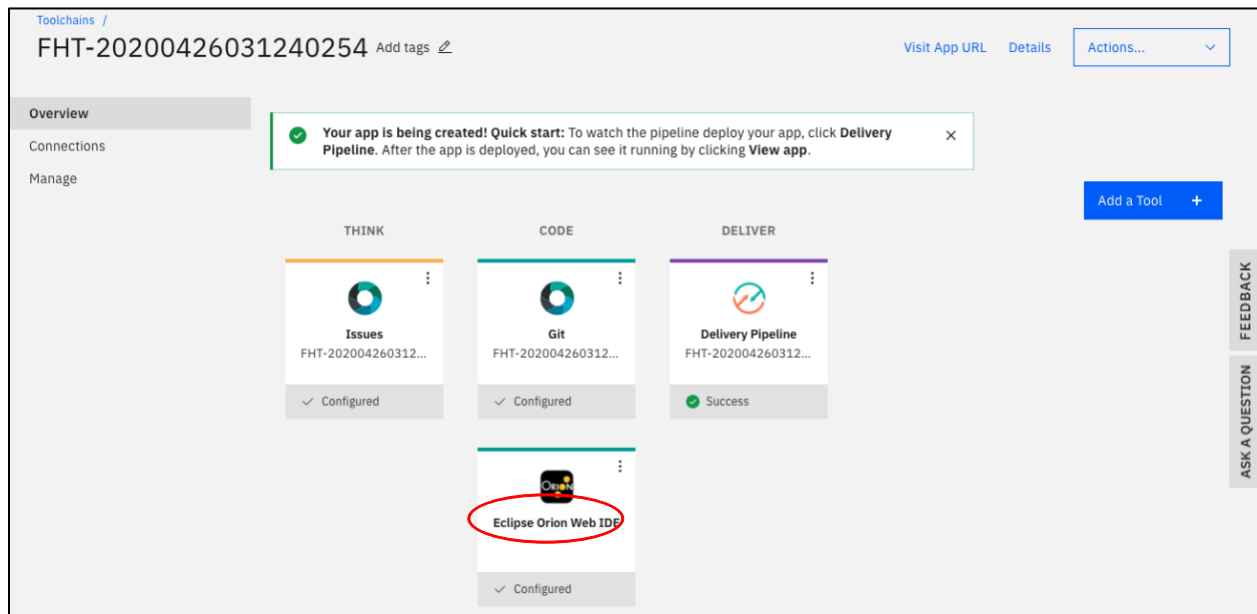
17. We now have tied the web application to the Watson Machine Learning service. Note that the Watson Machine Learning service could have more than one deployed model available to select and then embed in the web application. We now need to copy the scoring endpoint, which we previously copied and pasted 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.



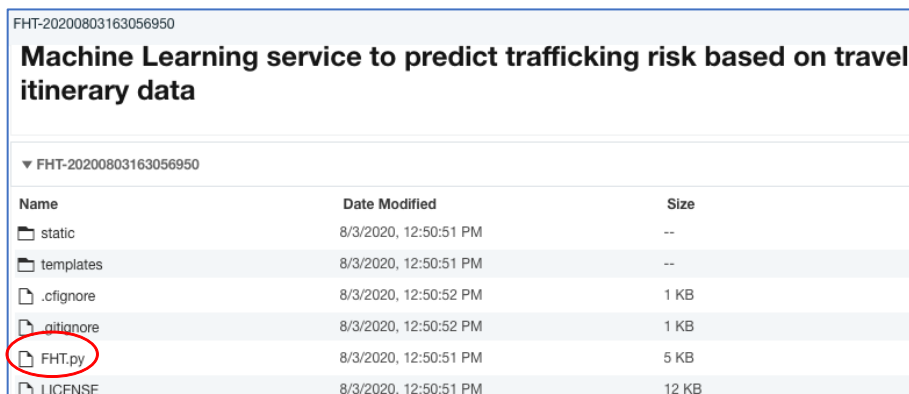
18. We are now going to paste the scoring endpoint into the application code. Click on the Toolchain (FHT-2019xxxxx below). If no toolchain appears, switch the location to Dallas.



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

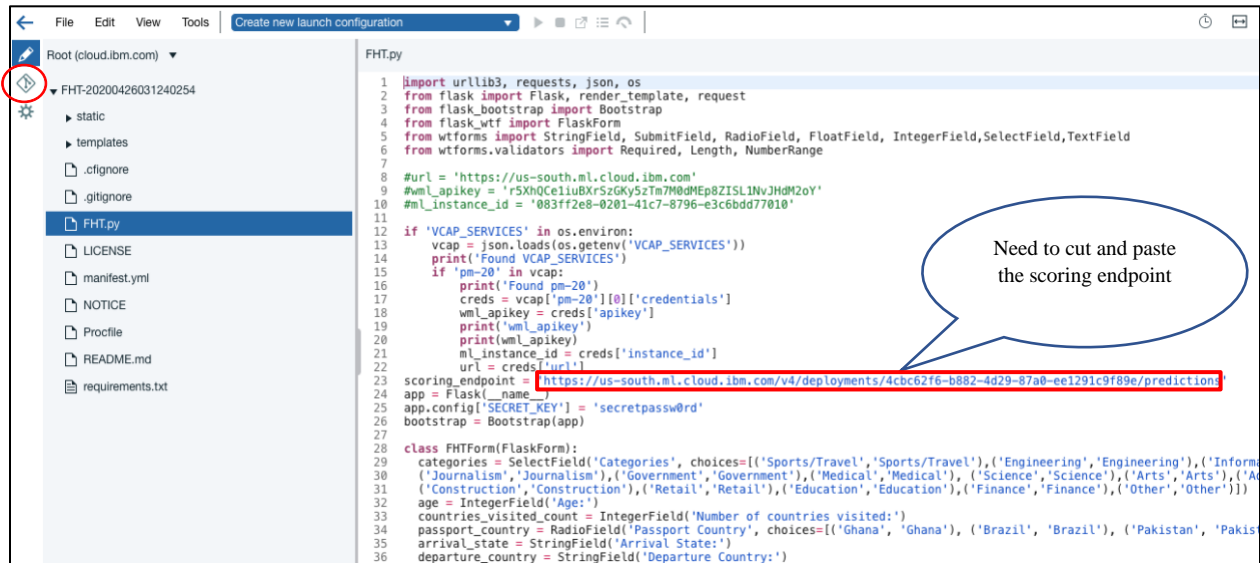


20. Click on the FHT.py file. This is a python source file.



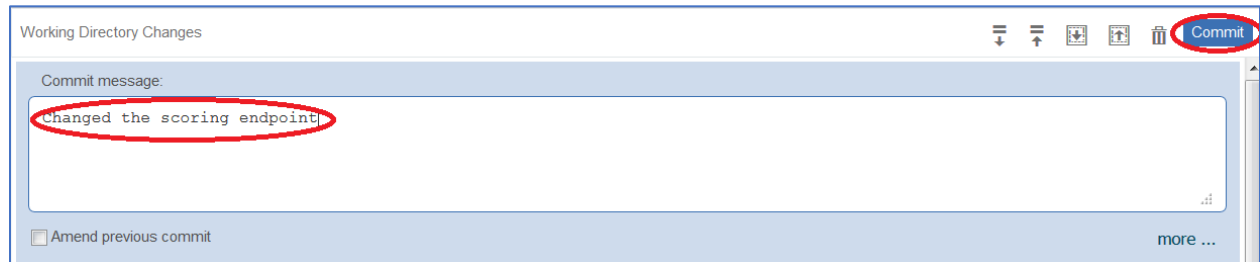
21. Go back to the Notepad file and copy the scoring endpoint to the clipboard. Look around line 23 in the FHT.py file for the “scoring endpoint =”. Select the scoring endpoint value in line 23 (starting with https:// may want to use Shift-End to get to the end of the line, and then back up one space 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, TextField
6 from wtforms.validators import Required, Length, NumberRange
7
8 #url = 'https://us-south.ml.cloud.ibm.com'
9 #wml_apikey = 'r5XhQCelluBxRszGKy5Zm7M0dMEp8ZISL1NvJHdM2oY'
10 #ml_instance_id = '083ff2e8-0201-41c7-8796-e3c6bdd77010'
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         print('Found pm-20')
17         creds = vcap['pm-20'][0]['credentials']
18         wml_apikey = creds['apikey']
19         print('wml_apikey')
20         ml_instance_id = creds['instance_id']
21         url = creds['url']
22
23 scoring_endpoint = 'https://us-south.ml.cloud.ibm.com/v4/deployments/4cbc62f6-b882-4d29-87a0-ee1291c9f89e/predictions'
24 app = Flask(__name__)
25 app.config['SECRET_KEY'] = 'secretpassw@rd'
26 bootstrap = Bootstrap(app)
27
28 class FHTForm(FlaskForm):
29     categories = SelectField('Categories', choices=[('Sports/Travel', 'Sports/Travel'), ('Engineering', 'Engineering'), ('Information', 'Information'), ('Journalism', 'Journalism'), ('Government', 'Government'), ('Medical', 'Medical'), ('Science', 'Science'), ('Arts', 'Arts'), ('Advertising', 'Advertising'), ('Construction', 'Construction'), ('Retail', 'Retail'), ('Education', 'Education'), ('Finance', 'Finance'), ('Other', 'Other')])
30     age = IntegerField('Age:')
31     countries_visited_count = IntegerField('Number of countries visited:')
32     passport_country = RadioField('Passport Country', choices=[('Ghana', 'Ghana'), ('Brazil', 'Brazil'), ('Pakistan', 'Pakistan')])
33     arrival_state = StringField('Arrival State:')
34     departure_country = StringField('Departure Country:')
35
36
```

22. 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

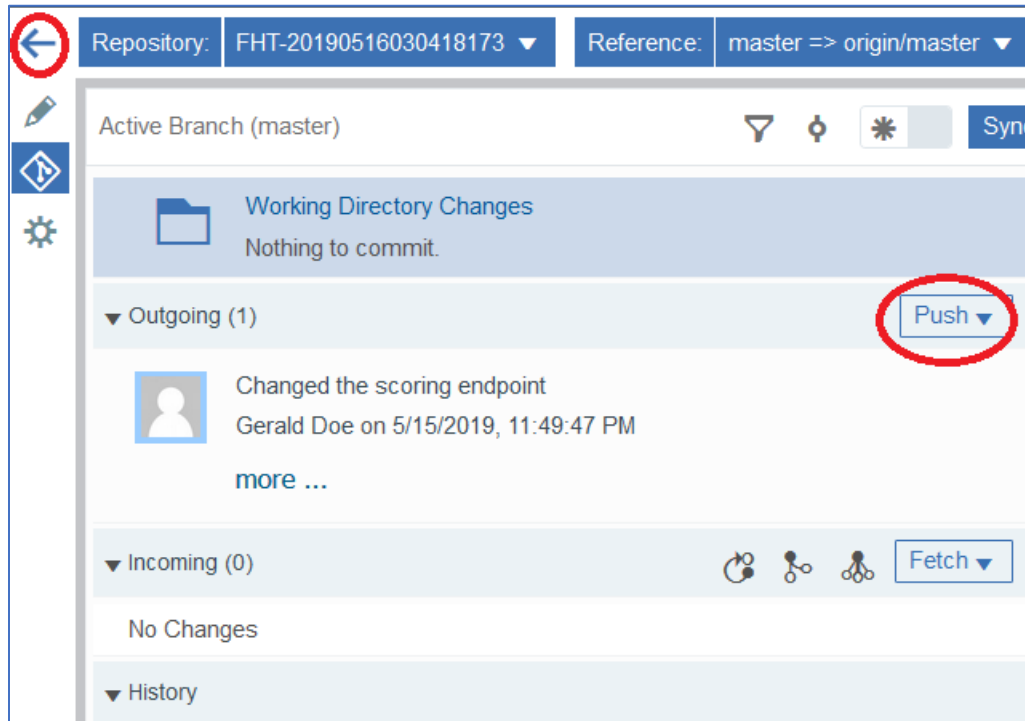
Commit message:

Changed the scoring endpoint

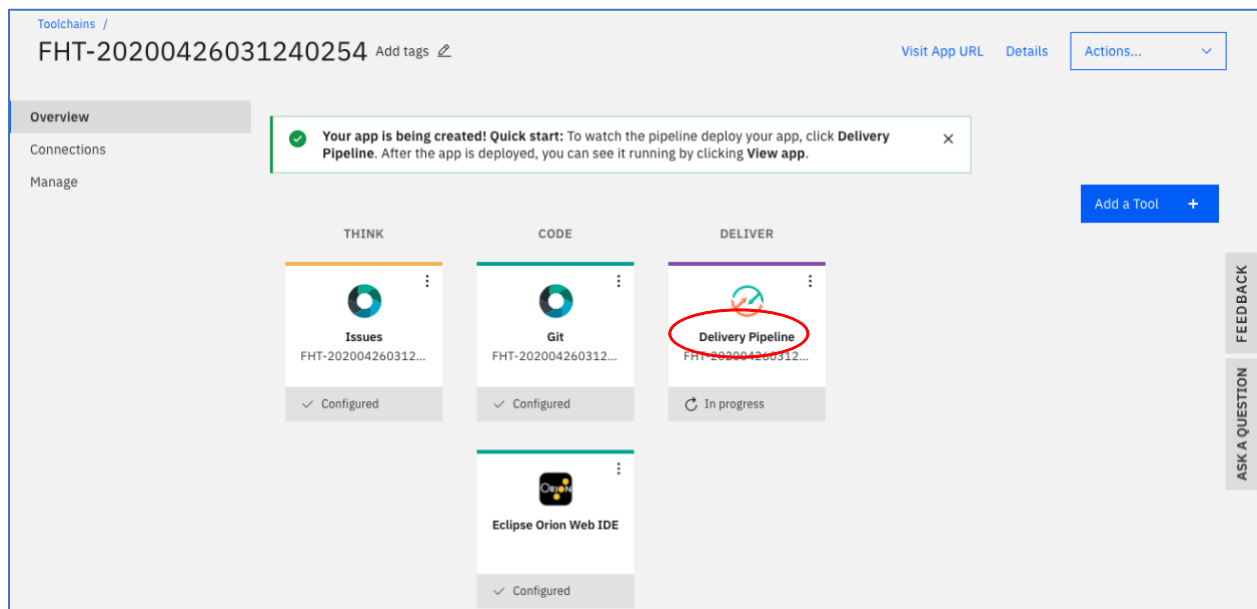
☐ Amend previous commit

**Commit**

23. 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.



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



25. 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**.

The screenshot displays two side-by-side panels for the 'Build Stage' and 'Deploy Stage'. Both panels show a 'STAGE PASSED' status. The 'Build Stage' panel includes a 'LAST INPUT' section with a commit by 'FCTO Labs' and a 'JOBS' section showing a 'Build' job that 'Passed now'. The 'Deploy Stage' panel includes a 'LAST INPUT' section with 'Stage: Build Stage / Job: B...' and a 'JOBS' section showing a 'Deploy' job that 'Passed now'. In the 'LAST EXECUTION RESULT' section of the 'Deploy Stage', the resource ID 'FHT-20200426031240254' is listed, with a 'View console' link circled in red below it.

26. Click on **Visit App URL**.

The screenshot shows a 'Resource list' table with one entry. The entry has the resource ID 'FHT-20200426031240254', a 'Running' status, and a 'Visit App URL' link circled in red. Other links like 'Add tags' and 'Details' are also visible.

Resource list /					
FHT-20200426031240254	Running	Visit App URL	Add tags	Details	Actions...



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

Categories: Sports/Travel

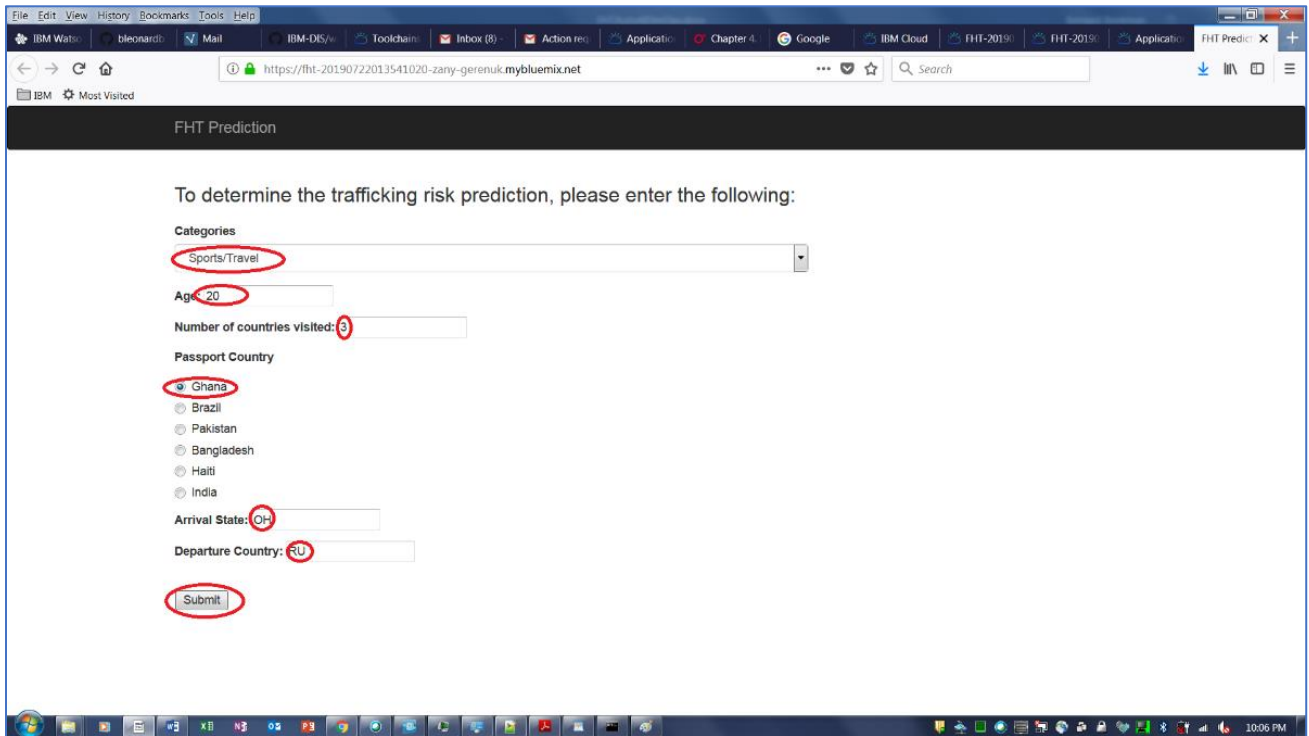
Age: 20

Number of countries visited: 3

Passport Country: Ghana

Arrival State: OH

Departure Country: RU



The screenshot shows a web browser window with the URL <https://fht-20190722013541020-zany-gerenuk.mybluemix.net>. The page title is "FHT Prediction". The form contains the following fields and values, with red circles highlighting the input areas:

- Categories: Sports/Travel
- Age: 20
- Number of countries visited: 3
- Passport Country: Ghana (selected from a dropdown menu)
- Arrival State: OH
- Departure Country: RU
- Submit button

28. 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!**

## **You have successfully completed the Lab!!!**

- ✓ Downloaded a female human trafficking cleansed dataset from the github repo
- ✓ Added an Auto AI Experiment to create a model to predict the trafficking risk
- ✓ Saved and Deployed the model
- ✓ Tested the model
- ✓ Deployed a simple web front-end application and connected it to the deployed model using an IBM Cloud DevOps toolchain.