**ModelOps - Model Monitoring on Cloud Pak for Data**

Eric Martens, Digital Technical Engagement, Data Science and AI

emartens@us.ibm.com

[emartens@us.ibm.com](mailto:emartens@us.ibm.com)

A close up of a logo

Description automatically generated

**Table of contents**

Contents

[Overview 1](#_Toc60836908)

[Required software, access, and files 1](#_Toc60836909)

[Required skills 1](#_Toc60836910)

[ModelOps Overview 2](#_Toc60836911)

[Step 1: Deploy models 2](#_Toc60836912)

[Step 2: Configure Watson OpenScale 6](#_Toc60836913)

[Step 3: Configure model monitoring 8](#_Toc60836914)

[Step 4: Evaluate and compare models 18](#_Toc60836915)

[Conclusion and Next Steps 20](#_Toc60836916)

# Overview

In this lab you will learn how to complete the following ModelOps tasks in **Cloud Pak for Data:**

* Deployment of candidate models
* Configuring OpenScale monitoring
* Evaluating and comparing models with test data
* Generating detailed explanations of predictions

# Required software, access, and files

1. To complete this lab, you will need a **Cloud Pak for Data as a Service** (**CPDaaS**) account: <https://dataplatform.cloud.ibm.com>

* If you don’t have a CPDaaS account, use the same URL to sign up for a free trial. The account will be activated in approximately 5 minutes.

1. If you already have an **IBM Cloud** account, make sure that you provisioned the required services – **Watson Studio, Watson Machine Learning (WML), and Watson OpenScale.**

* Navigate to your *Resource list* in your **IBM Cloud** dashboard: <https://cloud.ibm.com/resources>
* Check if the mentioned services are displayed under **Services**. If not, search for the services in the **Catalog** and add them.

Table

Description automatically generated

1. Download the [project zip file](https://github.com/ericmartens/fs2021/blob/main/fast_start_2021.zip?raw=true) to your machine.

# Required skills

We recommend that users who work through this lab:

* Understand data science lifecycle
* Have at least beginner knowledge of different methods for creating models

# ModelOps Overview

**ModelOps** is a process of developing and deploying data science assets to production. An important focus of ModelOps is automation of deployment, monitoring, and governance.

In this lab we will cover the **Test** and **Monitor** phases of ModelOps. We focus on these steps to show how **Watson OpenScale** can be used to evaluate and compare candidate models so that users can choose the best one for production.

A picture containing diagram

Description automatically generated

# Step 1: Deploy models

1. Sign into [Watson Studio](https://dataplatform.cloud.ibm.com/). From the quick navigation on the left, select **Projects**.

Graphical user interface, application

Description automatically generated

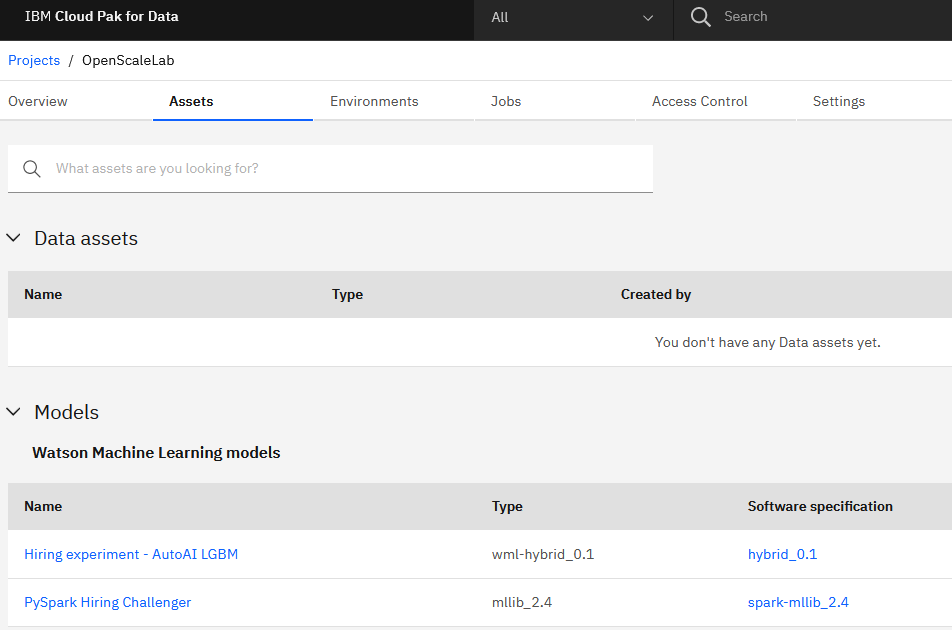
1. Click the **New project** link at the top left, then select **Create a project from a sample or file**.

Graphical user interface, text, application

Description automatically generated

1. Drag the *project zip* file (downloaded from Git) from your machine to the **Upload file** section. Give your project a name, select a Cloud Object Storage service from the dropdown list, and click **Create**.

The project contains two candidate models that predict whether a candidate will be hired by a company.



1. In your project, click the **Settings** tab. Scroll down to the **Associated services** section. Click **Add service** and select **Watson** from the list.

Graphical user interface, application

Description automatically generated

1. Select your **Watson Machine Learning** service from the list and click **Associate service**. When the service has been associated, close the popup window.

Graphical user interface

Description automatically generated with medium confidence

1. In the project **Assets** tab, navigate to the **Models** section. Click the **More** icon for the AutoAI model and select **Promote**.

Graphical user interface, application

Description automatically generated

1. Click **New space** to create a new deployment space for your models. Give your new space a name and select a storage option and WML instance from the dropdowns, and then click **Create**.
2. Once the space has been created, click **Close** to return to the model screen. Ensure that the new space you just created is selected in the **Target space** dropdown and click **Promote**. After your model has been promoted, you will return to the project assets screen. Follow similar steps to promote the PySpark model to the same deployment space.
3. Click the menu button at the top left of the screen, click the **Deployments** section to expand it, and select **View all spaces**.

Graphical user interface, application

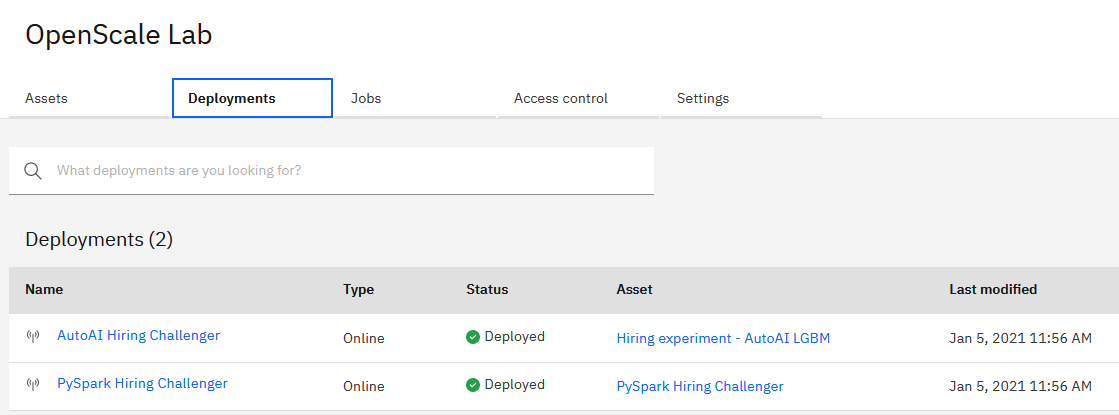
Description automatically generated

1. Click the space you just created. From the **Assets** tab, click the **Deploy** icon for the PySpark model.

Graphical user interface, application

Description automatically generated

1. Select the **Online** deployment type and name your model **PySpark Hiring Challenger** to make it easy to identify. Click **Create**.
2. Repeat the two steps above to create a deployment for the AutoAI model, using **AutoAI Hiring Challenger** as the name.
3. Click on the **Deployments** tab and make sure that both models have been successfully deployed.



# Step 2: Configure Watson OpenScale

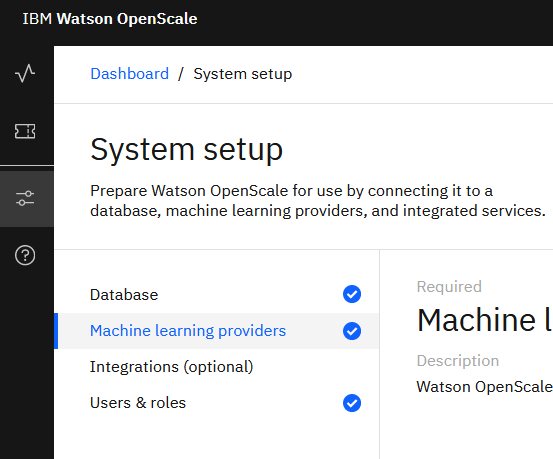
You have successfully deployed two models. We will now configure OpenScale to monitor these models for fairness, quality, and drift, and compare the results.

1. In a new browser tab, navigate to [Watson OpenScale](https://aiopenscale.cloud.ibm.com/aiopenscale/) and log in.

Next, we will check if OpenScale has been configured. If you used *Automatic Setup* after you provisioned the OpenScale service, then OpenScale has been configured with default values. It was configured to use the *Lite* database plan and it created 2 WML instances – *preproduction* and *production*.

Your models have been deployed to a different WML instance, and we need to add it to OpenScale so that you can select models to monitor.

Click on **Settings**. We will use this view to review/modify OpenScale configuration.



1. If the Database icon is checked, you can skip this step.

Click the pencil icon on the right of the screen to configure your database. Select the **Free lite plan** database from the **Database type** dropdown, unless you would like to use a paid database service you have already provisioned. Click **Save**.

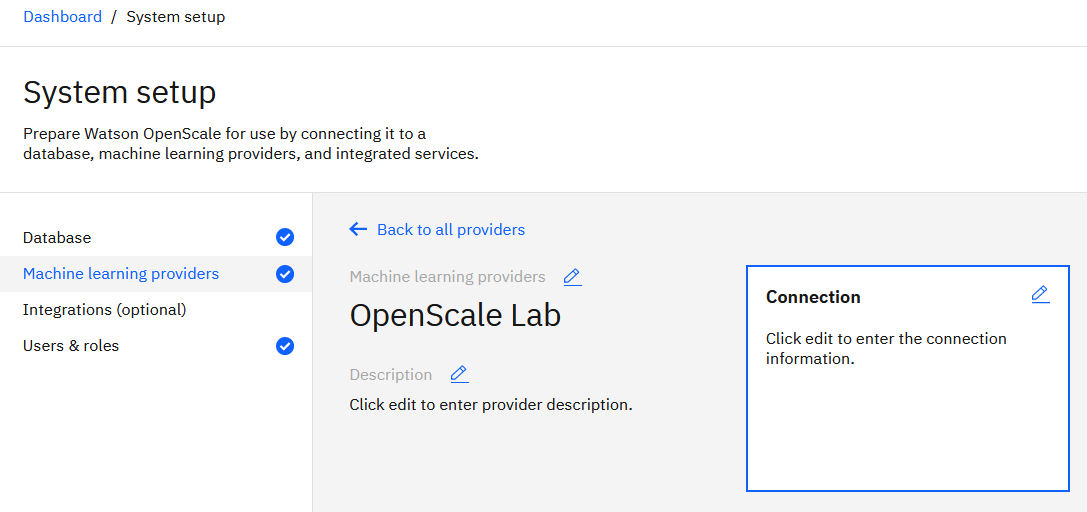
1. Select **Machine learning providers** from the list at the left of the screen.

Graphical user interface, text, application, chat or text message

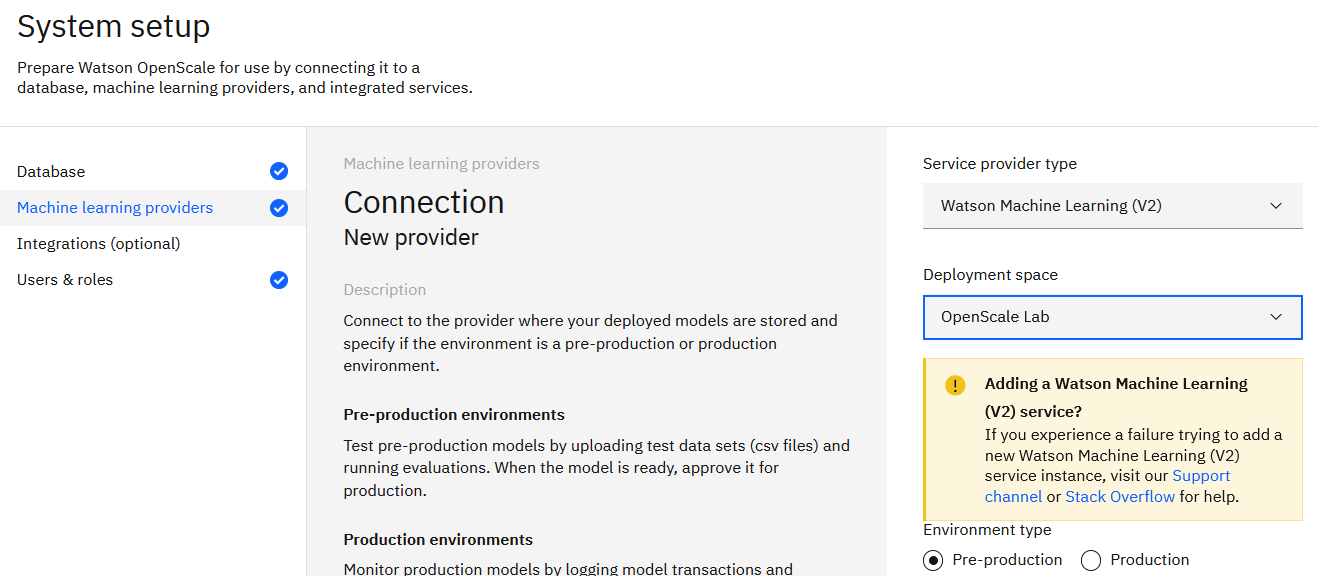
Description automatically generated

1. Click **Add machine learning provider**.

Provide the name and click on the edit icon in the **Connection** tile.



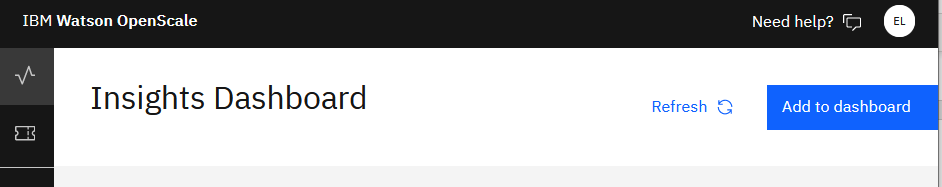
1. Select **Watson Machine Learning (V2)** as the Service provider type. Select the deployment space you have been working with from the **Deployment space** dropdown. Choose **Pre-production** as the environment type, which will simplify running tests on the models. Click **Save**.



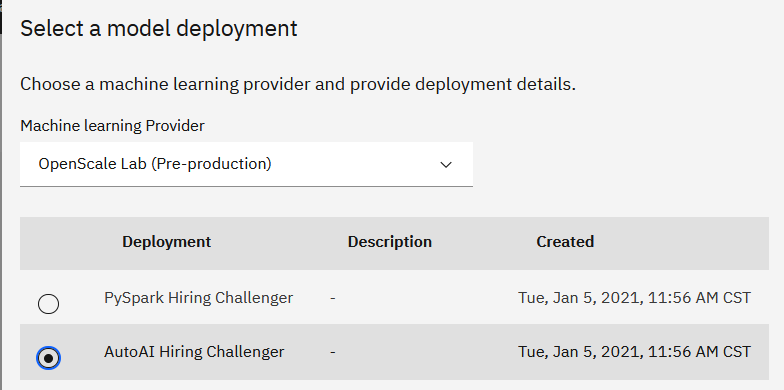
# Step 3: Configure model monitoring

You will now configure OpenScale to monitor your deployed models, starting with the AutoAI model. You will then repeat these steps for the second model.

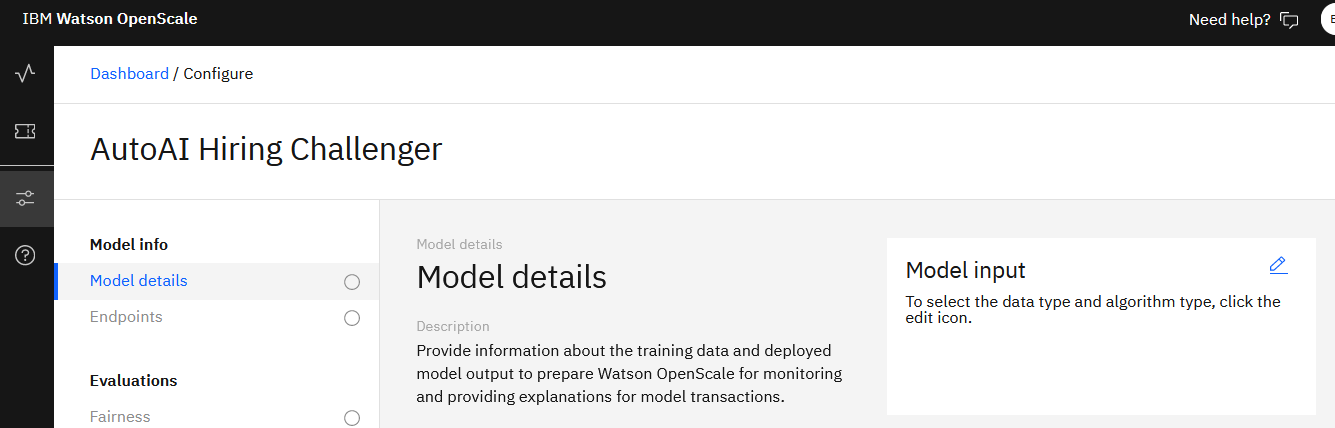
1. Click the dashboard link at the top left to return to the **Insights Dashboard**, then click **Add to dashboard**.



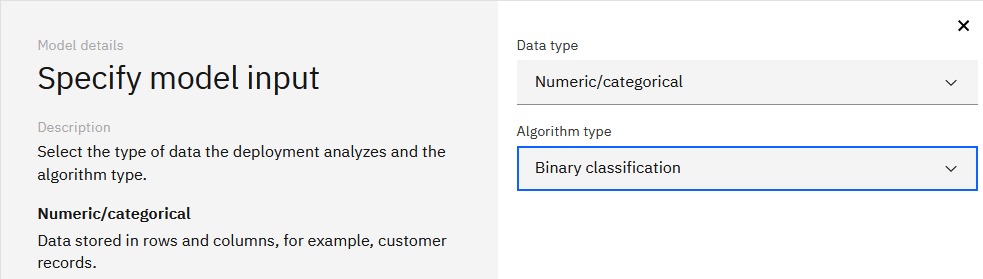
1. Select the provider you just added from the list, and then select the deployed AutoAI model and click **Configure**.



1. When the model has finished saving, click **Configure monitors**. Click the pencil icon in the **Model input** tile.



1. Set the data type to **Numeric/categorical** and the algorithm type to **Binary classification**. Click **Save and continue**.

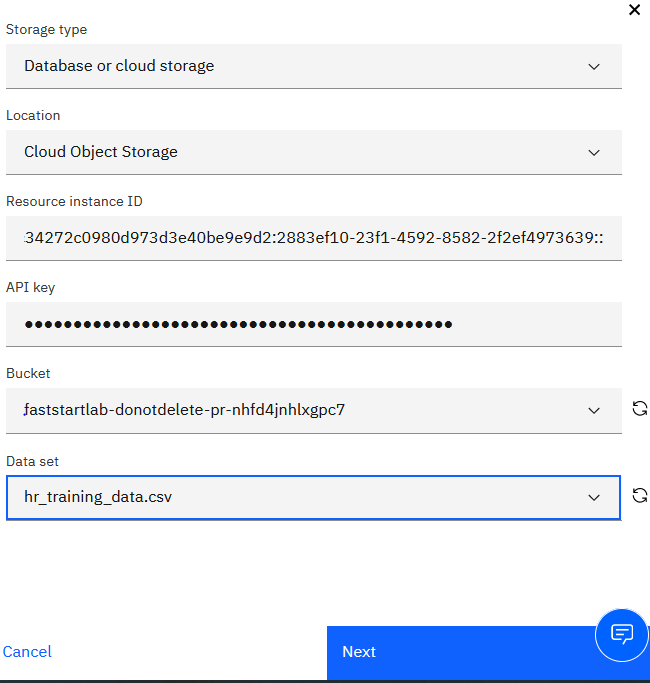


In the next section, you will connect to the instructor’s **Cloud Object Storage (COS)** to reference the training dataset, *hr\_training\_data.csv*. This dataset is included in the imported project, and if you would like to connect to data in your project, follow instructions in **Appendix A: Connecting to Training Data.**

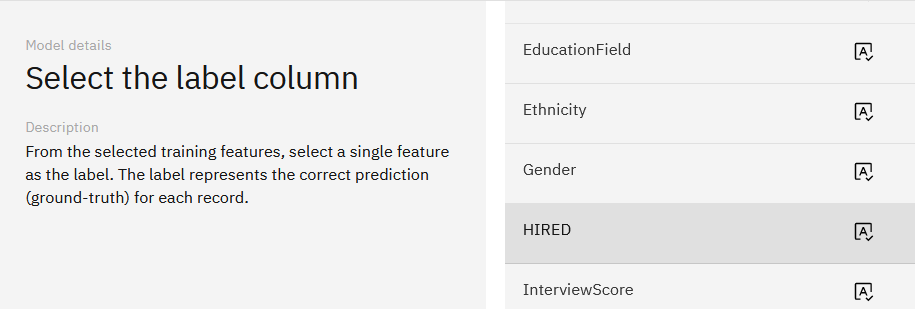
1. Click the pencil icon in the **Training data** tile. Set the location to **Cloud Object Storage** and enter the following values into the corresponding fields and click **Connect**:

|  |  |
| --- | --- |
| **Field** | **Value** |
| Resource instance ID | crn:v1:bluemix:public:cloud-object-storage:global:a/7d8b3c34272c0980d973d3e40be9e9d2:2883ef10-23f1-4592-8582-2f2ef4973639:: |
| API key | yqcPbWZ0AQPHleHVerrR4Wx5e9pymBdMgydbEra5zCif |

1. Select **faststartlab-donotdelete…** as the bucket, and **hr\_training\_data.csv** as the data set. Click **Next**.



1. Select **HIRED** as the label column, and click **Next**.



1. Check the box to the left of **Features** to select all columns as training features, then scroll down and uncheck the boxes for **Gender** and **Ethnicity**.

Gender and Ethnicity were not used for training the model. These features are included as OpenScale input so that OpenScale can find indirect bias.

Click **Next**.

Table

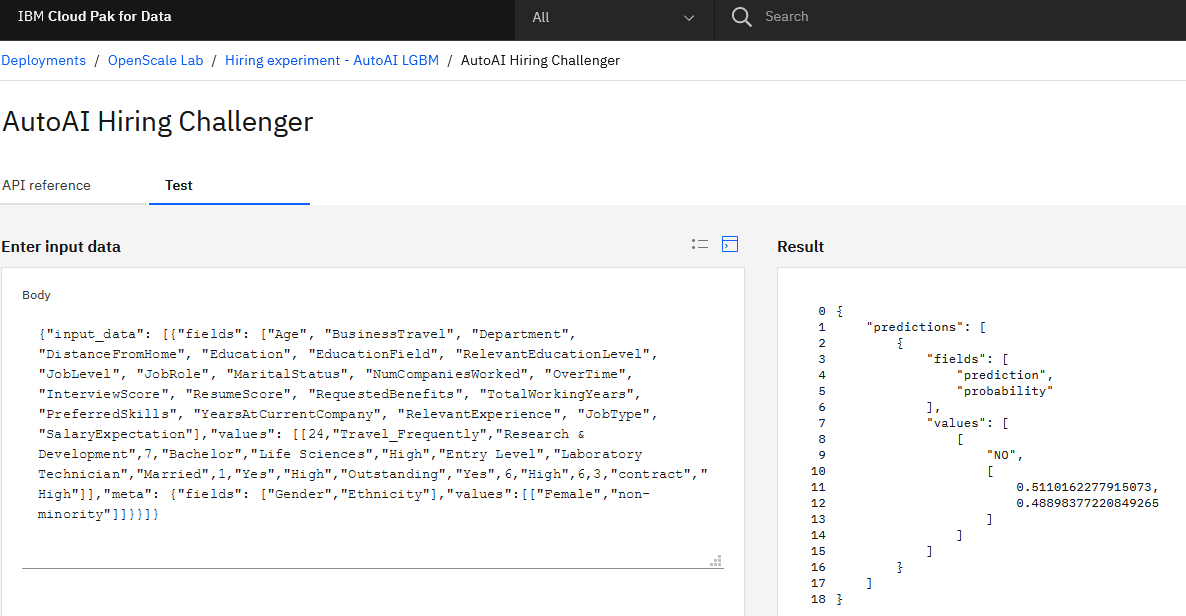
Description automatically generated with medium confidence

Now, we’ll need to submit data to our model so OpenScale can construct the correct payload log in the datamart.

1. Keep OpenScale open in one tab and return to the browser tab with your deployment space. Click on the **Deployments** tab and select the deployed AutoAI model. From the **Test** tab, click the icon to provide input data as JSON.

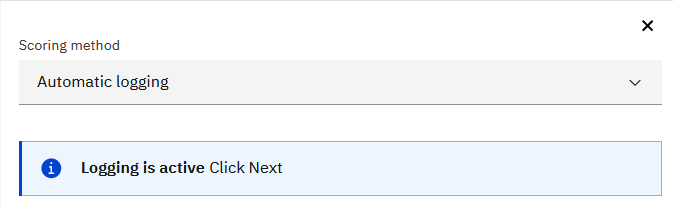
Copy and paste the JSON below into the field, and then click **Predict**:

{"input\_data": [{"fields": ["Age", "BusinessTravel", "Department", "DistanceFromHome", "Education", "EducationField", "RelevantEducationLevel", "JobLevel", "JobRole", "MaritalStatus", "NumCompaniesWorked", "OverTime", "InterviewScore", "ResumeScore", "RequestedBenefits", "TotalWorkingYears", "PreferredSkills", "YearsAtCurrentCompany", "RelevantExperience", "JobType", "SalaryExpectation"],"values": [[24,"Travel\_Frequently","Research & Development",7,"Bachelor","Life Sciences","High","Entry Level","Laboratory Technician","Married",1,"Yes","High","Outstanding","Yes",6,"High",6,3,"contract","High"]],"meta": {"fields": ["Gender","Ethnicity"],"values":[["Female","non-minority"]]}}]}

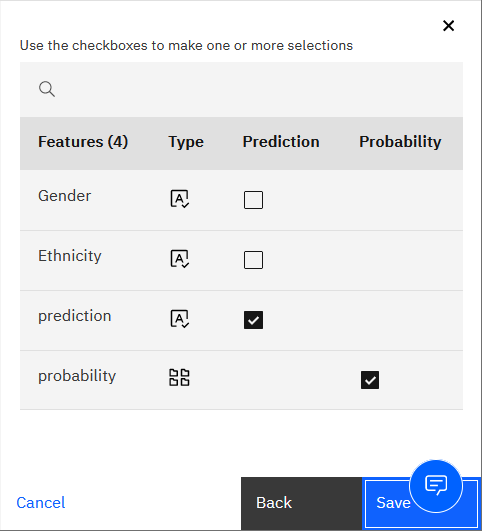


1. Return to OpenScale tab.

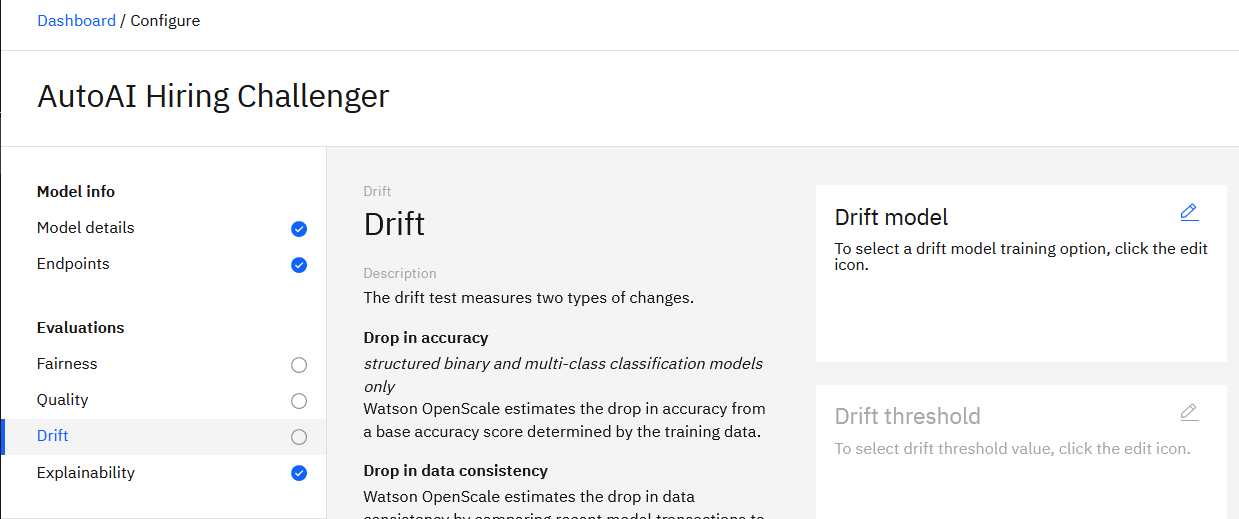
Verify that the scoring method is set to **Automatic logging** and click **Check now**. You should receive a message that logging is active. If you do not receive this message, wait 30 seconds and check again. Once logging is active, click **Next**



1. You do not need to make any changes on the last page of the configuration wizard. Click **Save**.



1. Click on the **Drift** monitor in the **Evaluations** section on the left side of the screen. Click the pencil icon in the **Drift model** tile.



1. Select the **Train in Watson OpenScale** option. Click **Next**.

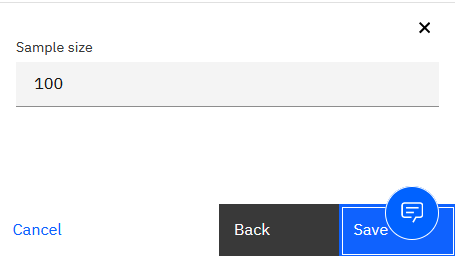
Graphical user interface, text, application

Description automatically generated

1. Click **Next** to accept the default drift threshold.

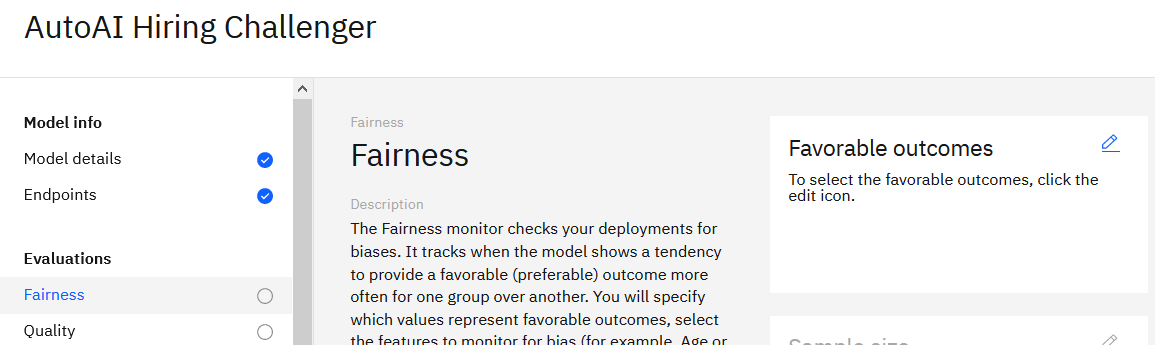


1. Set the sample size to 100 and click **Save**.



OpenScale will begin training a drift monitor, which will take a few minutes. You can continue configuring the other monitors while this training occurs.

1. Click the **Fairness** monitor in the **Evaluations** section. Click the pencil icon in the **Favorable outcomes** tile.

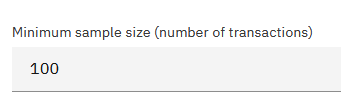


1. Use the checkboxes to set *YES* as the favorable value and *NO* as the unfavorable value, then click **Next**.

Graphical user interface, application

Description automatically generated

1. Set the minimum sample size to 100 records, then click **Next**.



1. For fields to monitor, check the boxes next to **Gender** and **Ethnicity** and click **Next**.

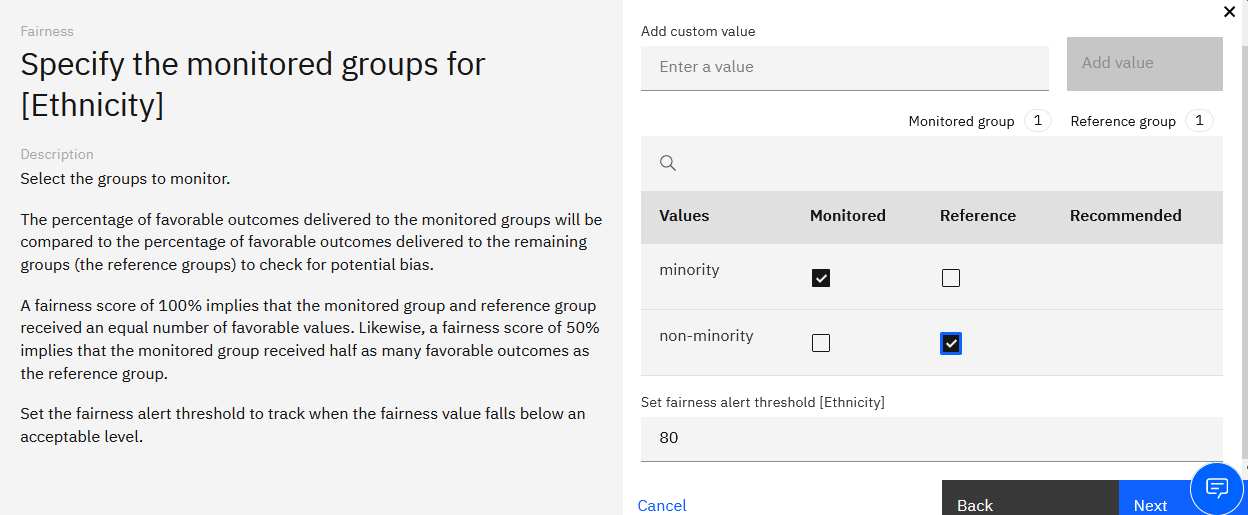
Graphical user interface, application, table

Description automatically generated

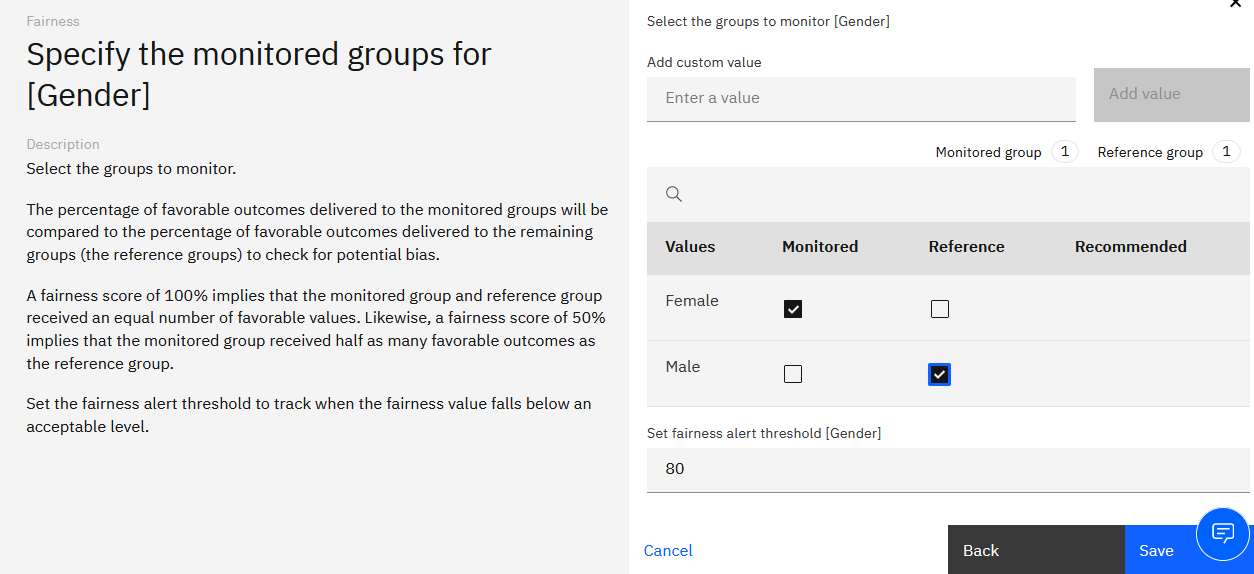
Next, we will configure minority and non-minority groups for monitoring bias and the threshold percentage for flagging bias. The explanations in OpenScale UI provide details of how fairness is measured.

1. For *Ethnicity*, set the *minority* group to **Monitored**, and the *non-minority* group to **Reference**. We will keep the default value (80) for fairness alert threshold. Click **Next**.

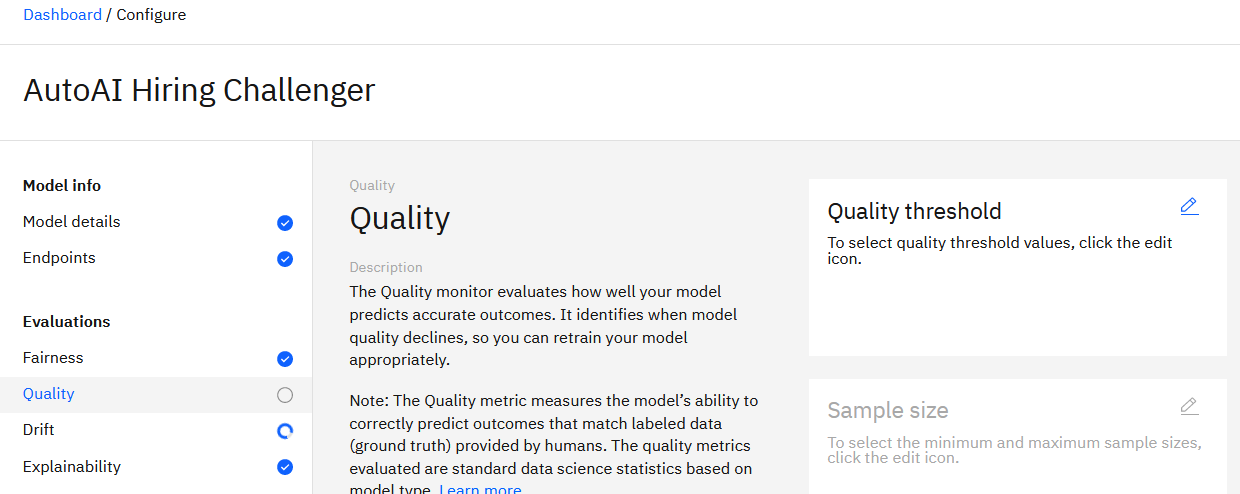
This setting means that when less than 80% of minority group is hired (compared to non-minority), OpenScale will issue an alert.



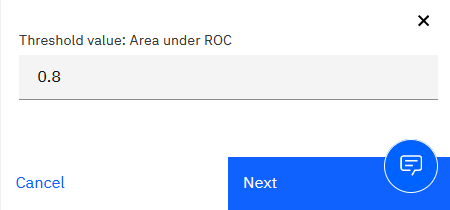
1. For Gender, set *Female* to **Monitored** and *Male* to **Reference**. We will keep the default value (80) for fairness alert threshold. Click **Save**.



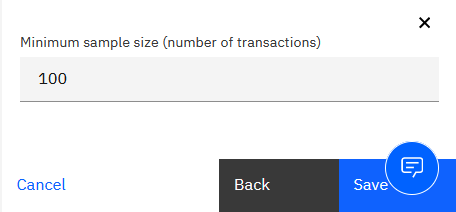
1. In the **Evaluations** section, select the **Quality** monitor. Click on the pencil icon in the **Quality** threshold tile.



1. Click **Next** to accept the default value of 0.8.



1. Set the minimum sample size to 100 and click **Save**.



**You have finished configuring the first model.**

If you would like to compare the 2 models, finish the rest of this section. If you would like to run evaluation on the model that you just finished, you can continue to **Step 4: Evaluate and Compare models**.

Next, we need to configure the PySpark model.

1. Click on the **Dashboard** link at the top left of the screen to return to the **Insights Dashboard**. Click the **Add to dashboard button**. Once again, select the machine learning provider you set up earlier, but this time, choose the *PySpark* model and click **Configure**.

Graphical user interface, application

Description automatically generated

1. Return to the beginning of this section and repeat the same steps for the PySpark model, using the exact same values as with the AutoAI model. However, during the **Specify model ouput details** step, set the **Prediction** value to **predictedLabel**.

Graphical user interface, application

Description automatically generated

**You have finished configuring the second model.**

# Step 4: Evaluate and compare models

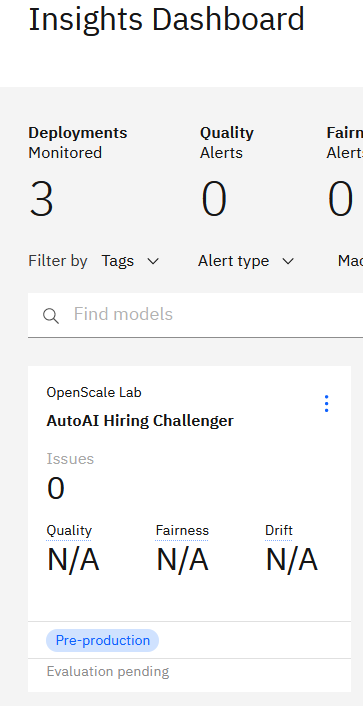
You will now compare the one or two pre-production models to see how they perform on a set of test data.

1. Download this file to your machine: <https://raw.githubusercontent.com/ericmartens/fs2021/main/evaluation_data.csv>

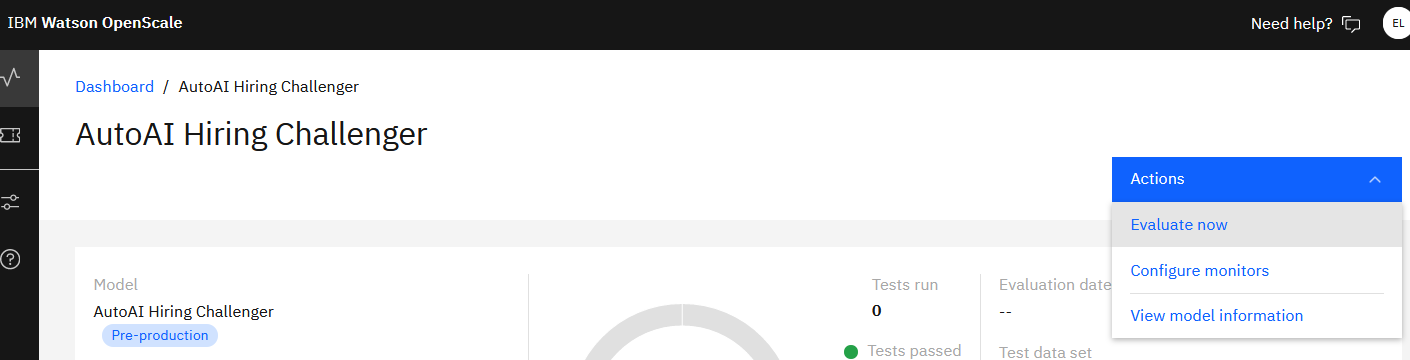
Similar to the training dataset, this file contains the actual result for an applicant (the applicant was hired or not hired). We are using this file in the context of a test, but this file can also represent actual outcomes that are used for evaluation of a model in production. The file can also be found in your Watson Studio project.

In this lab we will load the file and invoke evaluation manually. OpenScale can be configured to get data automatically from a data source or with an API.

1. From the [OpenScale Insights dashboard](https://aiopenscale.cloud.ibm.com/aiopenscale/insights), click on the tile for the AutoAI model.



1. From the **Actions** menu, select **Evaluate now**.



1. Set the **Import** option to **from CSV file**. Drag the file you downloaded from your machine to the upload window, and click **Upload and evaluate**.

Graphical user interface, text, application

Description automatically generated

1. The evaluation will take a few minutes to run. When it is finished, return to the Insights dashboard and repeat the steps above for the PySpark model (if you configured the 2nd model).

When the tests complete, the dashboard for the PySpark model shows the evaluation scores for model fairness, quality, and drift. If the score is better than the alert threshold, then the test will register as having passed.

In the screenshot below, the fairness score for ethnicity is 42% fair, which is significantly lower than our alert threshold of 80%, so the fairness test has failed.

Because our model is a binary classification algorithm, OpenScale has selected Area under ROC as the preferred quality metric. The model has scored 0.87, which is above the 0.80 threshold we set, so the quality test has passed.

The predicted drop in accuracy due to drift is at 5%, which is below the alert threshold of 10%, so the drift test has passed.

Clicking on the scores will allow you to dig deeper into any of these metrics, inspect individual transactions, and more. Take a few minutes to explore the evaluation results, then return to this dashboard and proceed.

Graphical user interface, application, Teams

Description automatically generated

1. From the PySpark results dashboard, click **Actions** and select **Compare**.

Graphical user interface, application

Description automatically generated

1. From the **Compare Model** dropdown, select the *AutoAI Hiring Challenger* model.

Graphical user interface, text, application

Description automatically generated

1. OpenScale generates a table to allow a side-by-side comparison between the most recent evaluation results for the two models, with better scores marked by the green vertical bar. In the screenshot below, you can see that the PySpark model performed better on the fairness test, but the AutoAI model had better results on the quality and drift tests.

Graphical user interface, application

Description automatically generated

1. OpenScale can also generate a PDF report of the test results, including detailed information on the model, and explanations on how the test scores are calculated. This capability can save data scientists the time and effort required to create and format reporting for business users.

To download the report, click on **Actions** and select **Download report PDF**.

Graphical user interface, text, application

Description automatically generated

You have finished comparing the two candidate models. We will now use OpenScale’s explanation capabilities to learn more about how a model arrives at an individual prediction.

# Step 5: Explain a prediction

You will now use OpenScale’s explanation feature to take an in-depth look at a single prediction. When you run a test on the model, OpenScale generates detailed explanations for two randomly selected predictions.

1. From one of the model dashboards, locate the **Number of explanations** section of the test details and click the number beneath it.

Graphical user interface, application

Description automatically generated

1. Click the **Explain** action link for one of the two predictions. The service may take a few minutes to run, but when it finishes, OpenScale will present a detailed look at how the model arrived at the prediction using the open source LIME algorithm. Note that all of the information from the explanation service can also be obtained using the Python SDK or the REST API, allowing developers to include it in business applications. For a simple example of how that may be done, please see this [Github repository](https://github.com/emartensibm/openscale_insurance).
2. The top section contains an overview of the prediction outcome, confidence, and primary features influencing the decision:

A picture containing text

Description automatically generated

1. The middle section provides more information on the influence of the individual features, and whether they made the model more or less confident in the final prediction. Mousing over the bars on the chart will show the value of the corresponding feature, and the influence in the prediction.

Chart, bar chart

Description automatically generated

1. Scroll to the top of the screen and click on the **Inspect** tab.
2. OpenScale’s contrastive explanation analysis can provide further information on the prediction, telling you to alter the feature values to obtain a different prediction. Click on the **Run analysis** button.

Graphical user interface, text, application

Description automatically generated

1. When the analysis has loaded, you can see which values would need to be altered to change the model’s prediction. You can try changing the values on the left side of the table and then clicking the **Score new values** button at the bottom to see how your changes impact the prediction and confidence of the model.

Graphical user interface, application, Teams

Description automatically generated

# Conclusion and Next Steps

In this lab you learned how to complete the following ModelOps tasks in Cloud Pak for Data:

* Promote and deploy models
* Configure OpenScale datamarts and machine learning providers
* Configure model monitors with OpenScale
* Evaluate and compare models using test data
* Explain individual predictions and analyze how changing feature values affect that prediction

# Appendix A: Connecting to Training Data

OpenScale requires access to the model training data in order to train the drift detection model, calculate feature distributions, find correlations between features and data used for indirect bias detection, and more. If you are unable to provide OpenScale access to the training data, you can use provided resources to create and run a Python notebook to generate the necessary information. In this lab, you connected to the data in an existing Cloud Object Storage bucket. If you would like to connect to the data file in your Watson Studio project, you will need the corresponding credentials for the file. To obtain the credentials, follow the steps below.

1. Navigate to the [Cloud Resources Dashboard](https://cloud.ibm.com/resources).
2. Scroll down to and expand the **Storage** section.

Graphical user interface, application

Description automatically generated

1. Click on the Object Storage instance you associated with your project.
2. From the menu at the left of the screen, click on **Service Credentials**.

Graphical user interface, text, application

Description automatically generated

1. Click on the **New credential** button.

Graphical user interface

Description automatically generated with medium confidence

1. Give your credentials a name and access level. For OpenScale, *Reader* permissions are sufficient. Click **Add**.

Graphical user interface, application

Description automatically generated

1. Locate the new credentials from the list. It will likely be at the bottom, so you may need to scroll forward to the last page if you have created other credential sets.
2. Click on the arrow icon to expand the credential set. You can use the *apikey* and *resource\_instance\_id* values to connect OpenScale to the object storage instance during the configuration step.

Graphical user interface, text, application, email

Description automatically generated

1. When connecting OpenScale to this object storage instance, you will find the training data file in a bucket associated with your project. The bucket will be named with the format *projectnamenospaces-<unique-identifier>*, so if your project is entitled “Fast Start 2021” your bucket should be called *faststart2021-<unique-identifier>*. Connecting to the bucket will allow you to select the *hr\_training\_data.csv* file from a list of your project assets. Make sure to use this file when configuring OpenScale, and not *hr\_training\_data\_autoai.csv*. The differences in these two files are discussed in Appendix B: Training data sets.

# Appendix B: Training data sets

During the lab, you may have noticed that there are two different sets of training data. One, *hr\_training\_data.csv*, contains the protected Gender and Ethnicity fields, while the other, *hr\_training\_data\_autoai.csv*, does not.

Gender and ethnicity of the job applicant were not included as features in the training of the models, which is why they are absent from the data used for the AutoAI experiment. However, in order to monitor the models for bias, OpenScale needs access to the full data set to find correlations between included features and protected features.

During model evaluations or at runtime, the protected gender and ethnicity features would be submitted as metadata. This allows them to be stored in the OpenScale datamart and used to calculate model bias, but not be used by the model itself for prediction. For an example of how this is done, see the notebooks here: https://github.com/ericmartens/indirect-bias

If you are familiar with creating AutoAI models, you may be wondering why we cannot use the full data set and simply deselect the protected features when setting up the experiment. At the current time, AutoAI pipelines require all training data fields to be submitted, even if the values are discarded before the model makes a prediction. Therefore, if gender and ethnicity are present in the training data, they must be present in the prediction request. Attempting to remove them from the request or designate them as metadata instead of features will result in an error. Additionally, submitting them as normal features will cause OpenScale to assume they are being used by the model. OpenScale will not attempt to find correlated proxy features and will not be able to accurately monitor them for bias.