**Watson OpenScale**

**Session ID: EDA20T028SH**

Eric Martens, Digital Technical Engagement

A close up of a logo

Description automatically generated

*December 15, 2019*

*Author: Eric Martens emartens@us.ibm.com*

**Table of contents**

Contents

[Overview 1](#_Toc29991664)

[Required software, access, and files 1](#_Toc29991665)

[Part 1: Watson OpenScale Sample Overview 2](#_Toc29991666)

[Run the OpenScale Auto Setup 2](#_Toc29991667)

[Explore the fairness and quality monitors 3](#_Toc29991668)

[Explore the application and drift monitors 8](#_Toc29991669)

[Explain an individual prediction 13](#_Toc29991670)

[Part 2: Deploy and Configure models for monitoring in OpenScale 15](#_Toc29991671)

[Run the notebook to deploy the model 15](#_Toc29991672)

[Prepare your model for monitoring 17](#_Toc29991673)

[Configure the OpenScale monitors 24](#_Toc29991674)

[Explore monitors and upload feedback 30](#_Toc29991675)

# Overview

In this lab you will learn how to use **Watson OpenScale** to monitor production machine learning models for fairness, quality and drift, and to explain how the models arrive at their predictions. For more information about **Watson OpenScale**, see the product page: <https://www.ibm.com/cloud/watson-openscale>

# Required software, access, and files

To complete this lab, you will need a **Cloud Pak for Data** cluster with **Watson OpenScale** installed, or a **Watson Studio Cloud** account and a lite version of Watson OpenScale. The two versions of OpenScale are functionally identical, and either can be used. These instructions will cover the Cloud version, which is significantly easier to provision.

* If you have not already, sign up for a free Watson Studio Cloud account: <https://dataplatform.cloud.ibm.com>
* You will also need a free lite version of Watson OpenScale, which can be created here: <https://cloud.ibm.com/catalog/services/watson-openscale>

Ensure that the Lite version is selected, give your service a name, and click **Create**:

A screenshot of a social media post

Description automatically generated

* You will also need a free lite version of Watson Machine Learning, which can be created here: <https://cloud.ibm.com/catalog/services/machine-learning>

Ensure that the Lite version is selected, give your service a name, and click **Create.**

* Finally, OpenScale provides a free lite version of PostgreSQL to store monitoring data. Optionally, you can provision and provide credentials for paid [PostgreSQL](https://cloud.ibm.com/catalog/services/databases-for-postgresql) or [Db2](https://cloud.ibm.com/catalog/services/db2-warehouse) instances.

# Part 1: Watson OpenScale Sample Overview

## Run the OpenScale Auto Setup

**Watson OpenScale** provides a quick setup utility that will automatically set up a database, create a model, and record seven days’ worth of measurements into the OpenScale monitors.

1. Launch the OpenScale application from your list of deployed services on Cloud Pak for Data, or via the URL for the public cloud version: <https://aiopenscale.cloud.ibm.com/aiopenscale/>
2. If you have already provisioned OpenScale and run the auto setup, but have not configured any other models for monitoring, you can click the **Help** button and then select **Reset auto setup**, and then proceed to step 4.  
     
   A screenshot of a cell phone

   Description automatically generated  
     
   If you have provisioned OpenScale and configured other monitors, you can delete and re-provision OpenScale to remove all data. If you would prefer to keep your configured monitors, you can simply read the descriptions of the available monitors in Section 1, or proceed to Section 2 of the lab to create and configure monitors for the Mortgage Default model.
3. On the Welcome screen, click **Auto setup**

A screenshot of a cell phone

Description automatically generated

The auto setup will provision a free lite version of **Watson Machine Learning** if necessary, and takes 5-10 minutes to run, during which time you can read about the OpenScale monitors and the scenario.  
  
For this demo, we will be monitoring a model that attempts to predict credit risk based on demographic data, as well as credit history, residence information, age, employment status, and more. The scenario and model use synthetic data based on the [UCI German Credit dataset](https://archive.ics.uci.edu/ml/datasets/Statlog+(German+Credit+Data)).

1. When the auto-setup has completed, click **Explore on my own**.

A screenshot of a social media post

Description automatically generated

## Explore the fairness and quality monitors

Watson OpenScale provides two types of monitors: application monitors, and model monitors. We will begin with the model monitors. The model monitors section of the **Insights Dashboard** provides an overview of all the models being monitored by OpenScale.

You can monitor models hosted with Watson Machine Learning on public or private clouds, Microsoft Azure, Amazon SageMaker, or custom models that provide JSON prediction output.

1. Click on the **Model monitors** tab.

The dashboard shows how many deployments are currently being monitored, as well as an overview of the alerts from those models. Below, each model is represented by a tile showing the machine learning provider and alerts for that model.

A screenshot of a cell phone

Description automatically generated

1. Click on the **GermanCreditRiskModel** tile.

The left side of the screen shows all of the active monitors for the model, divided into sections for **Fairness**, **Quality**, and **Drift**. In the **Fairness** section, you can see we have chosen to model two features, *Age* and *Sex*, for fairness. Additionally, you can see that we have an alert for the *Sex* feature.  
  
A screenshot of a cell phone

Description automatically generated

1. Click on the **Sex** fairness monitor.

Note the time frame above the graph. We are looking at hourly data from the past week, but the time frame can be adjusted as necessary. The graph shows the fairness score for females as a light blue line. The threshold we have set for an alert is shown by the red line. As you can see, the fairness score has dropped beneath the threshold consistently over the past week, alerting us to a potential unfair bias issue with the model. We can use OpenScale to investigate scores for specific time periods by moving the mouse over the chart.  
  
  
A screenshot of a cell phone

Description automatically generated

1. Click on the chart at the point where fairness was at its lowest (84%).

This screen shows us fairness details for this particular time period. OpenScale calculates its fairness score using a combination of actual predictions (payload data) and perturbed data, generated when the prediction probability is close to 50%. OpenScale will flip the monitored feature to see how it affects the prediction outcome.  
  
The fairness score is reached by dividing the percentage of positive outcomes for the monitored group (females, 67%) by the percentage of positive outcomes for the reference group (males, 80%).  
  
  
A screenshot of a cell phone

Description automatically generated  
  
  
The graph shows the breakdown of positive and negative predictions for our two groups. You can use the radio button at the top to view payload and perturbed data, actual payload (prediction) data, and training data.  
  
A screenshot of a cell phone

Description automatically generated

1. Click **Training** data set to view the training data breakdown.  
     
   As you can see, our training data had significantly more records for males than for females, which may be a potential source of our unfair bias against females.  
     
     
     
   A screenshot of a cell phone

   Description automatically generated
2. Click the **Debiased** data set.  
     
   OpenScale can create a sort of “corrective lens” to reduce or remove unfair model bias. It does this by training another model to predict when an outcome of the production model is likely to be unfairly biased, and flipping the feature value from the monitored group (female) to the reference group (male) and returning this prediction. On this screen, you can see how using this model will affect the fairness scores for other features.
3. Click the **back arrow** to return to the model dashboard.  
     
   A close up of a logo

   Description automatically generated
4. Click the **Area under ROC** monitor in the **Quality** section.  
     
   A screenshot of a cell phone

   Description automatically generated  
     
   OpenScale provides several different quality measurements. For our binary classification model, *Area under ROC* provides the best standard for model quality. These scores are generated by providing ground truth feedback data to the model, either via CSV upload or using a RESTful endpoint provided by OpenScale.  
     
   As with the fairness monitor, the chart in the middle of the screen shows model performance over an adjustable time window, with the relevant measurement shown as the light blue line and the alert threshold represented by the red line. As you can see, our model quality has consistently been above the threshold until the most recent measurement, represented by the right-most portion of the chart.
5. Click on the right-most portion of the chart, where Area under ROC drops to 0.69.

Here we can see a further breakdown of the feedback data and the various accuracy scores, and the number of feedback records evaluated.  
  
A screenshot of a cell phone

Description automatically generated

1. Click on the **monitor icon** to return to the **Insights Dashboard**.

A screenshot of a cell phone

Description automatically generated

## Explore the application and drift monitors

Using key performance indicator data from your applications, OpenScale can look for correlations between model performance and business results. That data can be uploaded via CSV file, or sent directly to OpenScale via API.

The **Application Monitors** tab of the **Insights Dashboard** provides an overview of the applications being monitored. In this case, we a credit assessment application that is using our risk model, and we are monitoring it for number of loan applications accepted per day and total amount of credit extended per day. As with the model monitors, we can set alert thresholds for these KPIs that will trigger alerts if they drop below certain levels.  
  
A screenshot of a cell phone

Description automatically generated

1. Click on the tile for the credit risk model application.  
     
   Here, we can see the two KPIs we are monitoring, performance trends, and information about how they might correlate to model performance.
2. Click on the **Performance** tab.  
     
   This graph shows how the KPI has performed over an adjustable time window as compared to the threshold we have set for its performance.

A screenshot of a cell phone

Description automatically generated

1. Click on the **Correlations** tab.  
     
   This screen shows how two metrics, expected drop in model accuracy due to drift and drop in data consistency, correlate to our KPIs. OpenScale has calculated that as model accuracy falls due to drift, our KPI falls as well.   
     
   A screenshot of a cell phone

   Description automatically generated
2. Click the **View** link beneath **Influence on KPI** to see the strength of the correlation.
3. Click the **Close** button to return to the previous screen.  
     
   A screenshot of a social media post

   Description automatically generated  
     
   We can use OpenScale’s drift monitor to investigate what might be affecting our model and causing drops in our KPIs.
4. Click on **Go to Drift monitor**.  
     
   OpenScale’s drift monitor is a separate linear regression drift model, trained to determine which types of data the production model struggles to correctly predict. This drift model allows OpenScale to forecast potentially costly drops in model accuracy without requiring additional feedback data.   
     
   Additionally, the drift monitor compares incoming prediction requests with the training data to identify changes in data consistency that also may affect model output.  
     
   These two measurements are shown on the drift monitor screen. Estimated drop in accuracy is represented by the dark blue line, drop in data consistency by the light blue line, and alert threshold by the red line.  
     
   A screenshot of a cell phone

   Description automatically generated
5. Click the chart where the drop in accuracy is at its greatest (8%).  
     
   A screenshot of a map

   Description automatically generated  
     
   Here, we can get a detailed view of the transactions responsible for estimated drops in accuracy, data consistency, or both.
6. Click on **Transactions responsible for drop in accuracy**.  
     
   A screenshot of a social media post

   Description automatically generated  
     
   OpenScale divides transactions (predictions) that are affecting model accuracy into groups that share feature characteristics, providing a view of which feature values are causing our drift and how much influence each is having.
7. Click on one of the tiles for a transaction grouping.  
     
   A screenshot of a cell phone

   Description automatically generated  
     
   Here, OpenScale provides a detailed summary of how the values are affecting the model, as well as recommendations for how to address the issues with corrected training data. Finally, OpenScale lists the predictions and feature values for this grouping.

## Explain an individual prediction

Using a variety of open source algorithms, OpenScale can provide highly-detailed explanations of the predictions your model has made.

1. Click the **Explain prediction** link in the **Actions** column of the transactions table.  
     
   A screenshot of a cell phone

   Description automatically generated  
     
   OpenScale’s explanation features work by slightly perturbing the feature values from the original prediction, sending these values to the production model, and measuring the impact the changes have on the outcome. By sending thousands of perturbed requests, OpenScale can gain a detailed picture of feature importance for not only relatively simple models like linear regression or decision tree classifiers, but also complex neural networks and image recognition models.  
     
   A screenshot of a cell phone

   Description automatically generated  
     
   The upper portion of the screen shows information on the model and original prediction, as well as IBM’s contrastive explanation technology. The **Minimum changes for No Risk outcome** show the Pertinent Negative values, or the least amount the feature values for the transaction can be changed to get a different outcome.  
     
   The **Maximum changes allowed for the same outcome** show the Pertinent Positive values, or how much the feature values can change and still have the model make the same prediction.
2. Scroll down to see the factors that influenced the prediction.  
     
   Here, you can see the confidence the model has in its prediction (59%) as well as a quick summary of the most important factors that led the model to make a prediction that this particular loan application represents a risky one.  
     
   The chart at the bottom of the screen shows the feature values for this prediction, whether they contributed to a No Risk or Risk prediction, and how much they influenced the model.  
     
   A screenshot of a cell phone

   Description automatically generated  
     
   This detailed information allows you to ensure that your models are making predictions grounded in reality, as well as providing full explainability for predictions in case of an external audit or internal review of the model.  
     
   Finally, the data from this explanation is stored in OpenScale’s database, where it, along with all other metrics, can be retrieved via API and surfaced to business users or even customers via dashboards or other applications.

# Part 2: Deploy and Configure models for monitoring in OpenScale

## Run the notebook to deploy the model

To get started, we need a model for **Watson OpenScale** to monitor, which we will create using Watson Studio’s Python notebook feature. In this use case, we will try and predict whether an applicant will default on their mortgage based on features such as the list price of the home, applicant debt levels, and more. All the data files and notebooks are located in this github repository:

<https://github.com/emartensibm/mortgage-default>

1. Navigate to [Watson Studio](https://dataplatform.cloud.ibm.com/projects/) and click **New project**.

A screenshot of a cell phone

Description automatically generated

Click the **Create an empty project** tile. Give your project a name, an optional description, and follow the prompts to assign or create storage, and then click **Create**.

1. Once the project has been created, click the blue **Add to project** button and then select **Notebook**.

A screenshot of a social media post

Description automatically generated

1. Select the **From URL** option and give your notebook a name.  
     
   A screenshot of a cell phone

   Description automatically generated
2. Scroll down and copy and paste the following into the **Notebook URL** field:  
   <https://raw.githubusercontent.com/emartensibm/mortgage-default/master/mortage_default_model_creation.ipynb>
3. Click **Create Notebook**.
4. Follow the instructions in the notebook to provision and create service credentials for Watson Machine Learning. Subsequent steps in this lab will use these same credentials, so it will be helpful to paste them into a text editor for later use. If you choose to change your model’s name or deployment name, note the new names as well.  
     
   After completing the notebook, you will have created, saved and deployed a machine learning model. Proceed to the next section.

## Prepare your model for monitoring

Watson OpenScale’s monitors can be configured via web user interface, or via Python client. In this section, we will use the web UI, but a Python notebook that achieves the same configuration is available [here](https://github.com/emartensibm/mortgage-default/blob/master/mortgage_openscale_config.ipynb).

1. You will be returning to your Watson Studio project shortly, so leave that browser tab open. In a new browser tab, sign in to your instance of [Watson OpenScale](https://aiopenscale.cloud.ibm.com/aiopenscale/). If you have already worked with OpenScale in the past and have set up your instance of Watson Machine Learning as a provider, you may skip to step 6. If not, continue on.
2. If you are signing into OpenScale for the first time, you will see a Welcome screen that offers to perform an auto-setup. This auto-setup takes about five minutes to run. It will (if necessary) provision an instance of Watson Machine Learning, set up a free version of PostgreSQL for OpenScale data, create and deploy a model, and populate OpenScale with one week’s worth of monitor readings. If this is your first time using OpenScale, we recommend running the auto-setup, but it is not necessary for this lab. You can always skip the auto-setup, then later remove and recreate your instance of OpenScale and run it then. If you choose to run it, wait for it to finish and take as much of the subsequent tour as you like, and then skip to step 6.
3. If you skip the auto-setup, you will be asked to configure a database for OpenScale. If you have a paid PostgreSQL or Db2 instance, you can click the **Use existing or purchase a new database** tile. You can also click **Use the free Lite plan database**, which will allow you to complete the lab. However, no credentials are supplied for this database, so you will not be able to access it directly without using the OpenScale APIs or clients later. Select an option and click **Save**. Once the database is saved, click **Select Provider**.  
     
   A screenshot of a cell phone

   Description automatically generated
4. Next, you will need to identify a machine learning provider. In addition to models hosted on IBM’s machine learning services, OpenScale can also monitor models on Micosoft Azure, Amazon SageMaker, or custom environments. Click **Add machine learning provider**. Then, from the **Select a provider** screen, select **Watson Machine Learning** and click **Next**.
5. Select your WML instance from the dropdown list, give it a name that will be used in OpenScale, and click **Save**. Once the configuration is finished saving, click **Go to Dashboard**.  
     
   A screenshot of a cell phone

   Description automatically generated
6. From **Model Monitors** tab on the **Insights Dashboard**, click the **Add** button. If you already have models in your dashboard, click the blue **Add to dashboard** button.  
     
   A screenshot of a cell phone

   Description automatically generated
7. OpenScale will read the deployed models from your WML instance and populate the dropdown list. Select the mortgage default list from the model and click **Configure**. Once your selection has saved, click the **Configure monitors** button.  
     
   A screenshot of a social media post

   Description automatically generated
8. You will need to provide some details and data on your model. Select **Numeric/categorical** as the Data type, and **Binary classification** as the algorithm type, then click **Save**.   
     
   A screenshot of a cell phone

   Description automatically generated
9. Next, you will be prompted to send requests to your model so OpenScale can create the correct database logging schema. There are a variety of ways to send requests to the model, but we will use a Python notebook that can be scheduled to send requests periodically. Return to the browser tab with the Watson Studio project and click the blue **Add to project** button and select **Notebook**.
10. Select the **From URL** tab and give your model a name. In the **Notebook URL** field, paste the following URL and click **Create Notebook**:  
    https://raw.githubusercontent.com/emartensibm/mortgage-default/master/mortgage\_model\_feed.ipynb  
      
    A screenshot of a social media post

    Description automatically generated
11. Follow the instructions and run the notebook to send scoring requests to the model. It will use the Watson Machine Learning service credentials you created in the first section. Ensure that the model and deployment names match what you used in the previous notebook if you chose to change them. When you have successfully run the notebook, continue to the next step.
12. Return to the browser tab with OpenScale and click the **I’m finished** button. It may take some time for the logging table to be created, so if it fails at first, wait a minute or two and try again.  
      
    A screenshot of a cell phone

    Description automatically generated
13. When the configuration has finished saving, click **Model details** in the left pane, and then click the **Begin** button.  
      
    A screenshot of a social media post

    Description automatically generated
14. We will be manually configuring the monitors. Click **Next**.
15. OpenScale requires access to your model’s training data for the explainability and drift detection services. Set the **Location** field to **Cloud Object Storage**.  
      
    A screenshot of a cell phone

    Description automatically generated  
      
    Paste the following values into the form fields, click **Test**, and then click **Next**.

|  |  |
| --- | --- |
| Field | Value |
| Login URL | https://s3.us.cloud-object-storage.appdomain.cloud |
| Resource Instance ID | crn:v1:bluemix:public:cloud-object-storage:global:a/7d8b3c34272c0980d973d3e40be9e9d2:2883ef10-23f1-4592-8582-2f2ef4973639:: |
| API Key | yqcPbWZ0AQPHleHVerrR4Wx5e9pymBdMgydbEra5zCif |

1. Select the only available bucket from the **Bucket** dropdown, and then select *Mortgage\_Full\_Records.csv* from the **Data set** dropdown. Click **Next**.  
     
   A screenshot of a cell phone

   Description automatically generated
2. Identify the label column in our training data. OpenScale has selected **MortgageDefault**, which is correct. Click **Next**.  
     
   A screenshot of a cell phone

   Description automatically generated
3. Select the features used to train the model. Click **Select all** to highlight all features, and then click the **ID** tile to deselect it. Then click **Next**.  
     
   A screenshot of a cell phone

   Description automatically generated
4. OpenScale has automatically detected the categorical features. Click **Next** to advance. On the next screen, click the **prediction** tile to identify it as the prediction field, and click **Next**. On the next screen, click the **probability** tile to identify it as the probability field, and click **Next** again. Finally, click Next to skip the step to identify a transaction ID column, and then click **Save** to save the model details.  
     
   Your model is now prepared for monitoring. You can proceed to the next section.

## Configure the OpenScale monitors

Now that the model has been set up, we can proceed to configuring the quality, fairness, and drift monitors.

1. Click on the **Quality** section in the left pane, and then click **Begin**.  
     
   A screenshot of a cell phone

   Description automatically generated
2. On this screen, we can set the threshold for quality alerts. If the quality measurement drops below the value we set here, we will see an alert on our Insights dashboard. Use the slider to set the threshold to **70%** and click **Next**.  
     
   A screenshot of a cell phone

   Description automatically generated
3. Next, set the minimum and maximum sample sizes for the quality monitor. We will use 100 for the minimum sample size for this model so we can quickly see results in the monitor. Using a such small value here can cause drastic fluctuations in the measurement. For real-world production systems, you should use a value closer to the default 1000. Use the slider or the text field to set the minimum sample size to **100** and click **Next**, then click **Save**.  
     
   A screenshot of a cell phone

   Description automatically generated
4. We can now configure the fairness monitor. Select **Fairness** from the left pane and then click **Begin**.  
     
   A screenshot of a cell phone

   Description automatically generated
5. We must specify positive and negative model outcomes. This is a binary classification model, so there are only two values. For multiclass classification models, we would could select from all the different possible outcomes. For regression models, we would specify a positive range and a negative range.  
     
   For this model, a “No” outcome means the model does not predict that the applicant will default on their mortgage, so the loan should be offered. A “Yes” outcome means the loan should not be offered without further investigation, as the model believes the applicant is likely to default.  
     
   Click and drag the **No** icon to the Favorable values section. Drag the **Yes** icon to the Unfavorable values section. Then click **Next**.  
     
   A screenshot of a cell phone

   Description automatically generated
6. Next, we will specify the features to monitor. We can choose any feature, not just traditional features that might include societal bias such as gender, age, or ethnicity. In this case, our model’s training data was gathered before applying for mortgages online was common. Therefore, online applications are under-represented in our training data and may be a source of potential bias. Click the **AppliedOnline** tile to select that feature, and then click **Next**.  
     
   A screenshot of a cell phone

   Description automatically generated
7. Identify the group that applied online by dragging the **Yes** icon to the **Monitored group** section, and the **No** icon to the **Reference group** section. Click **Next**.  
     
   A screenshot of a cell phone

   Description automatically generated
8. Like the quality threshold we set in a previous step, OpenScale allows us to specify a fairness score at which we will see an alert on the Insights dashboard. Use the slider to set the fairness alert threshold to **90%** and click **Next**.
9. As before, using a relatively small value for sample size could result in drastic swings in the fairness score, but will allow us to quickly see results in our monitor without having to feed as much data into the model. Use the slider or the text entry to set the minimum sample size to **100** and click **Next**, then click **Save**.  
     
   A screenshot of a social media post

   Description automatically generated
10. The final monitor to configure is the drift monitor, which can detect when your model is receiving requests it struggles to correctly predict and identify potential accuracy drops without additional feedback data. Click **Drift** in the left pane and click **Begin**.  
      
    A screenshot of a cell phone

    Description automatically generated
11. The drift monitor works by training a drift detection monitor from your training data, which we provided via Object Storage in earlier steps. OpenScale can automatically train this model for us. Click the **Analyze and train in Watson OpenScale** tile, then click **Next**.  
      
    A screenshot of a cell phone

    Description automatically generated
12. As with the fairness and quality monitors, we need to set an alert threshold for drift. Use the slider to select **5%** and click **Next**.
13. You will also need to specify a sample size. Use the slider or the text entry to set the minimum sample size for drift to **100** and click **Next**, then click **Save**. OpenScale will begin training the drift monitor, which may take up to ten minutes. You can continue to the next section while the drift model trains.

A screenshot of a social media post

Description automatically generated

## Explore monitors and upload feedback

Now that you have successfully configured OpenScale to monitor your models, you can begin exploring the Insights dashboard and begin explaining model predictions.

1. Click the icon for the dashboard in the upper left of the screen.  
     
   
2. Click on the **Model Monitors** tab, and then click on the tile for the **Mortgage Default** model you have created.  
     
   A screenshot of a cell phone

   Description automatically generated
3. First, we will look at model quality, which requires new “ground truth” feedback data.   
     
   Download the following file to your machine, taking care to save it with the **.csv** extension and not a .txt extension:  
   <https://raw.githubusercontent.com/emartensibm/mortgage-default/master/mortgage_feedback.csv>
4. From the list of monitors on the left side of the screen, click on **Area under ROC** in the **Quality** section. Then click on the **Add feedback data** link in the lower right corner.  
     
   A picture containing screenshot

   Description automatically generated
5. This screen shows code snippets for providing feedback data via the OpenScale Python client or REST endpoint, allowing you to automate the feedback process by sending data directly from your business applications. For this lab, we will use direct CSV upload. Click the **Add Feedback Data** button and navigate to the file you downloaded in a previous step. Choose **Comma (,)** as the field delimiter and click **Select**.  
     
   A screenshot of a cell phone

   Description automatically generated
6. Click the **blue back arrow** next to your model name to return to the feedback monitor screen, and then click the **Check quality now** link in the lower right corner.  
     
   A picture containing screenshot

   Description automatically generated
7. When the quality check finishes, you will now have a data point at the far right of your graph. The blue dot represents the quality metric score. If you are looking at the Area under ROC metric, you will see a red dot that represents the 70% alert threshold we set for quality during the configuration.  
     
   You can explore the various quality measurements provided by OpenScale, and click on the graph for more details. When you are finished, return to the screen with the list of monitors, and click on the **AppliedOnline** feature in the **Fairness** section.  
     
   A screenshot of a cell phone

   Description automatically generated
8. OpenScale shows the fairness over the past week for the AppliedOnline feature we designated for monitoring in a previous step. We also specified that the monitor needs at least 100 logged predictions to generate a score. The notebook we ran to score the model during the preparation step sent more than 100 prediction requests to the model, so we will be able to get a fairness score.  
     
   The fairness monitor runs hourly, so you may have a fairness score and a data point at the far right of your graph. If not, click the **Check fairness now** link in the bottom right corner of the screen.  
     
   A screenshot of a cell phone

   Description automatically generated
9. Once the fairness monitor has successfully run, you should have a fairness score and data point on the far right of the graph. The small blue dot represents the score, and the red line represents the 90% alert threshold we set during the configuration step. To see more information about the predictions that led to this score, click on the graph data point. Note that because the scored data was selected at random, your scores may vary from the screenshot below.  
     
   A screenshot of a cell phone

   Description automatically generated
10. On this screen, you can explore the a breakdown of the predictions made by your model during this specific time period, including actual predictions (Payload), and actual predictions plus additional predictions OpenScale altered and sent to the model to get a more complete picture of model performance (Payload + Perturbed).  
      
    You can also see the breakdown of your supplied training data, and OpenScale’s automated de-bias feature.  
      
    When you are finished, select **Payload** as your data set, and then click the **View Transactions** button.  
      
    A screenshot of a cell phone

    Description automatically generated
11. This screen shows a summary of the predictions the model made during this time period on the right, and a table of individual predictions on the left. Select a prediction from the table and click the **Explain** link.  
      
    A screenshot of a cell phone

    Description automatically generated
12. The explanation service will take a minute or two to run; OpenScale is using a variety of industry-standard open source algorithms to slightly perturb the data, then sending this altered data to the model to determine how changing feature values affects the prediction. When the service finishes, displayed results tell us which features of that particular record were most influential on the decision made. OpenScale also calculates changes we could make to the record to get a different prediction, and how much we could change the record to still get the same prediction.  
      
    When you are finished exploring, click the **blue back arrow** in the upper left portion of the screen multiple times to return to the list of monitor models.  
      
    A close up of a logo

    Description automatically generated
13. Click on the drift monitor from the list.  
      
    A screenshot of a cell phone

    Description automatically generated
14. OpenScale’s drift monitor runs every three hours, so you may need to wait to complete this portion of the lab. Once the monitor has run, the graph will show a red line that represents the alert threshold for loss of accuracy that we set during the configuration, a dark blue line representing predicted drop in accuracy, and a light blue line for drop in data consistency. Click on the graph for more details.  
      
    A screenshot of a cell phone

    Description automatically generated
15. On this screen, OpenScale shows of transactions it knows the model struggles to correctly predict, which will lead to a drop in model accuracy, as well as transaction records that vary from the training data, which lead to a drop in data consistency. Clicking the transaction sets in the chart show more information about these two groups.  
      
    Click the **Transactions responsible for drop in data consistency** link.  
      
    A screenshot of a cell phone

    Description automatically generated
16. The tiles here show detailed information for which feature values are contributing to the loss in data consistency. Clicking on the tiles will show OpenScale’s recommendations for correcting any issues, as well as listing individual transactions so you can use the explanation service on them if you choose.

**Congratulations!** You have completed the IBM Watson OpenScale mortgage default lab. In this lab, you used a Python notebook in Watson Studio to train, save and deploy a machine learning model. You configured OpenScale to monitor this model, and then added prediction and feedback data to explore the output of these monitors. Finally, you used OpenScale’s explainability service to provide a detailed look at a single transaction.