IBM Watson

Build, Deploy and Access the Fairness of Your AI Model

Lab X

Version 2021.1.25

Contents

Preface

Overview

Within a bank's consumer lending department, a customer's application for a loan undergoes a lot of scrutiny before a decision of approval or rejection is made. In that process, a loan agent or customer representative must manually assess the information provided by the applicant, which includes information such as credit history, savings, loan duration, loan amount, and housing information. This information is then compared to the vast amount of historical data of similar applicants and analyzed to see whether there was a risk involved when their loans were approved or rejected. This evaluation process can take a while, which opens the possibility of the bank losing a potential customer to another bank.

To reduce the decision-making time and to increase the accuracy of the decisions being made, an increasing number of banks have begun to use machine learning-based solutions. This modernized approach enables a customer representative to make predictions about a loan application with a click of a button.

In this case study, we show you how to predict the risk of a loan application using the following services:

- Watson Studio
- Watson Machine Learning (using AutoAI)
- Cloud Object Storage
- Watson OpenScale

Watson OpenScale is an open platform designed by IBM Watson that allows businesses to operate, automate, and analyze AI implementations at scale. The OpenScale dashboards help to explain AI outcomes to business users and communicates the status of AI models visually and intuitively. It's a "one-stop-shop" for monitoring fairness within the context of other model attributes (e.g. explainability, accuracy, model health).

OpenScale is an end-to-end product robust enough for data scientists, but approachable enough for use by business owners who might not have engineering backgrounds. There's no coding required.

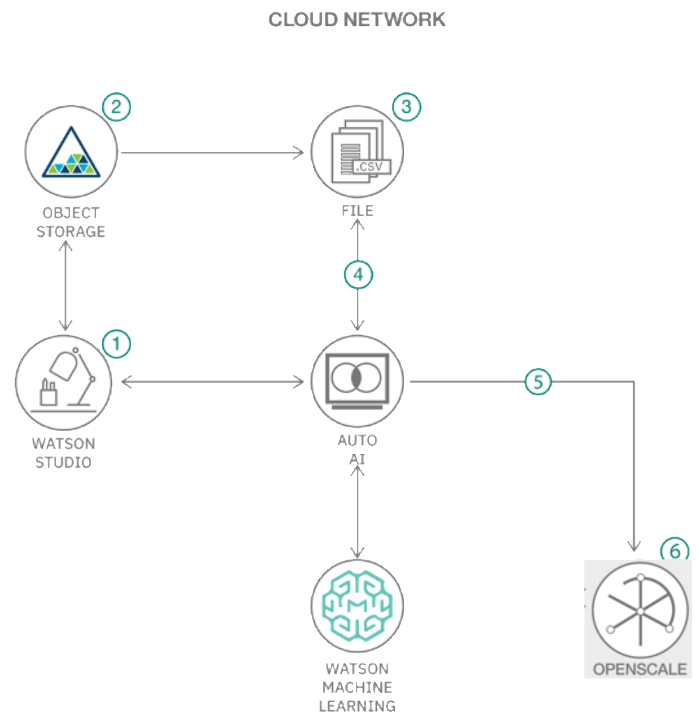
Objectives

Upon completed this tutorial, you will understand how to:

- Provision the necessary service.
- Build and deploy a model in AutoAI.
- Configure the model in OpenScale.
- Analyze: Drift, Fairness, Quality and Explanability.

To complete this lab students must perform the following Milestones:

1. Provision necessary services
2. Build and deploy a model using AutoAI based on the open-source German Credit Score CSV file.
3. Configure the deployed model in OpenScale
4. Analyze model fairness, quality and drift


CLOUD NETWORK
2
OBJECT STORAGE
3
.CSV
FILE
4
1
WATSON STUDIO
AUTO AI
5
6
OPENSCALE
WATSON MACHINE LEARNING

Flow

1. Log into Watson Studio, create a project and initiate an instance of Auto AI & Object Storage
2. Upload the data file in the CSV format to the object storage.
3. Initiate the model building process using Auto AI and create pipelines.
4. Access the deployed model in Watson OpenScale.
5. Analyze, drift, quality (accuracy) based on newly uploaded to test the model
6. Change the Fairness threshold and introduce bias in the data.

Tools

- IBM Watson Studio: Analyze data using RStudio, Jupyter, and Python in a configured, collaborative environment that includes IBM value-adds, such as managed Spark.
- IBM AutoAI: The AutoAI graphical tool in Watson Studio automatically analyzes your data and generates candidate model pipelines customized for your predictive modeling problem.
- IBM Cloud Object Storage: An IBM Cloud service that provides an unstructured cloud data store to build and deliver cost effective apps and services with high reliability and fast speed to market. This code pattern uses Cloud Object Storage.
- Watson OpenScale: Watson OpenScale allows businesses to operate and automate AI at scale - irrespective of how the AI was built and where it runs. Available via the IBM Cloud and IBM Cloud Private, it infuses AI with trust and transparency, explains outcomes, and automatically eliminates bias.

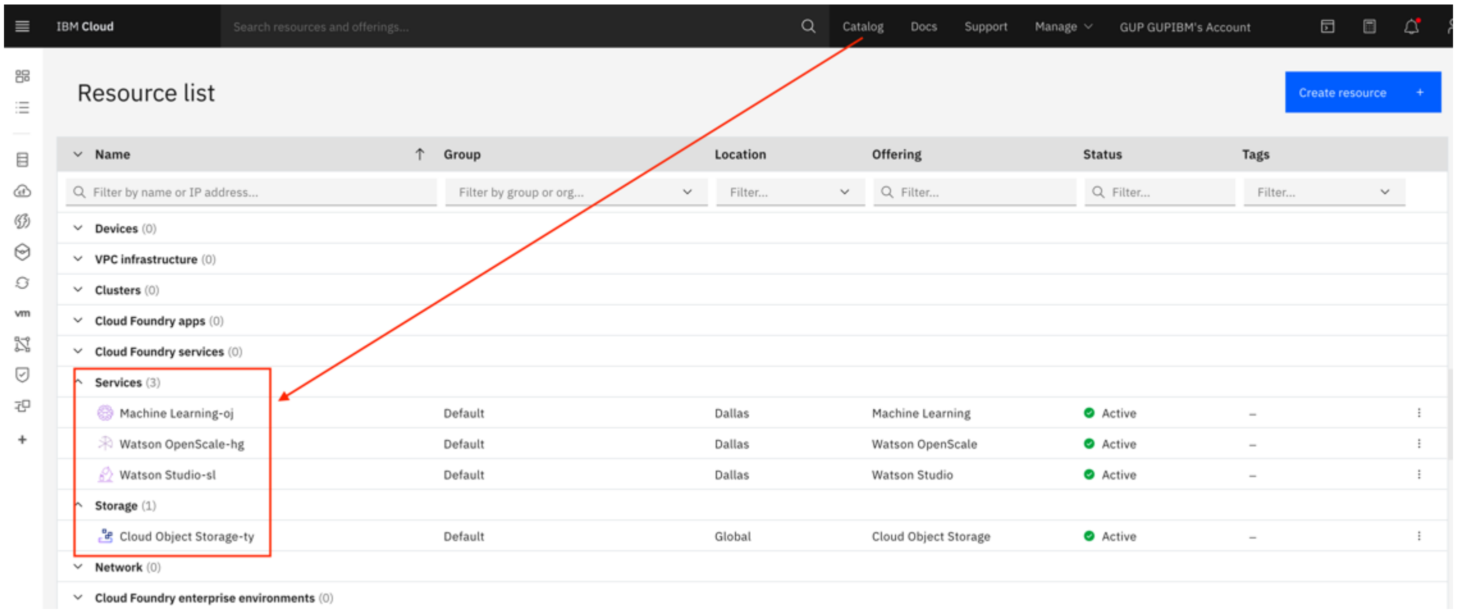
Prerequisites

You have created an account and logged into IBM Cloud platform.

Milestone 1: Create required services

You begin by creating the following four services from the Catalog tab.

1. Log into IBM Cloud.
2. From the **Catalog** menu, search for each of the following services (in no particular order) and click **Create** (Accept all defaults) to provision each of the four services.
    - **Watson Studio** (there is no need to click Get Started) just create the service and on to next.
    - **Machine Learning** (again, no need to view it in Watson Studio, you'll spend time inside each of the services later).
    - **Watson OpenScale**
    - **Cloud Object Storage**
3. Close all tabs in your browser related to those newly created services and keep the IBM Cloud dashboard open.
4. Access the **Resource list** from the hamburger icon in the top left corner.

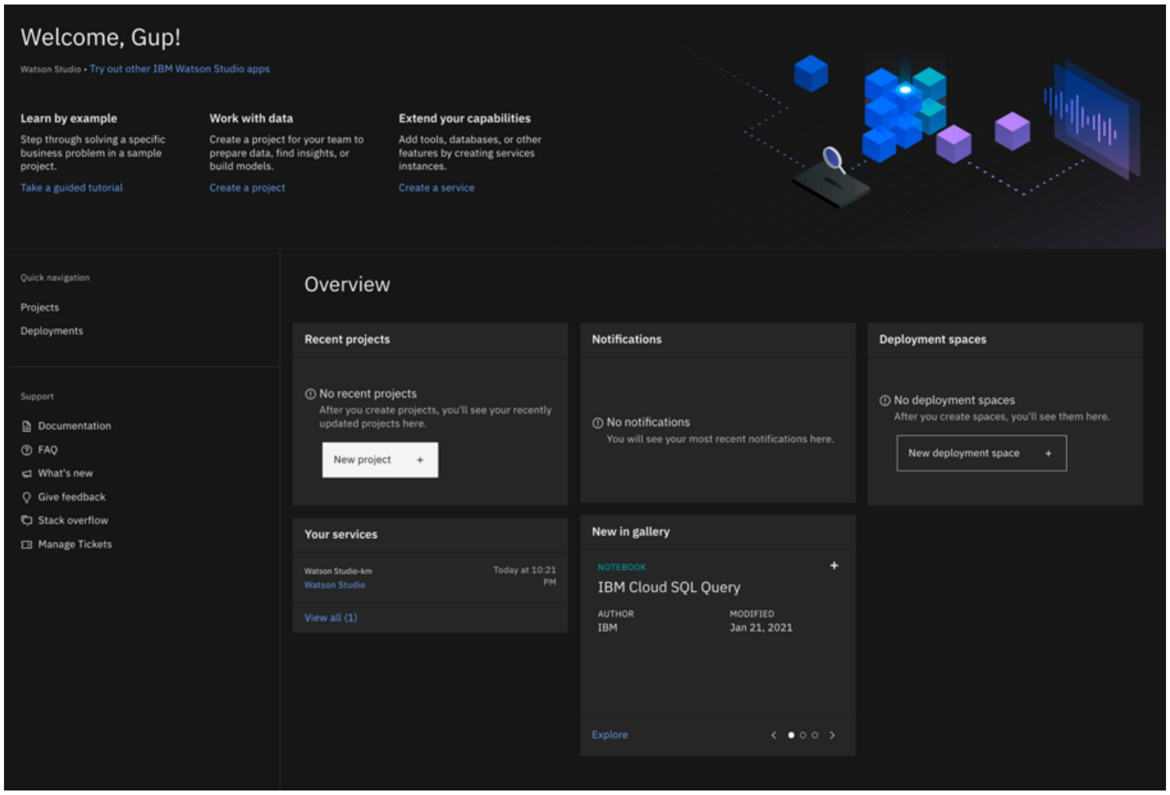


Resource list

Name	Group	Location	Offering	Status	Tags
Filter by name or IP address...	Filter by group or org...	Filter...	Filter...	Filter...	Filter...
Devices (0)					
VPC infrastructure (0)					
Clusters (0)					
Cloud Foundry apps (0)					
Cloud Foundry services (0)					
Services (3)					
Machine Learning-oj	Default	Dallas	Machine Learning	Active	–
Watson OpenScale-hg	Default	Dallas	Watson OpenScale	Active	–
Watson Studio-sl	Default	Dallas	Watson Studio	Active	–
Storage (1)					
Cloud Object Storage-ty	Default	Global	Cloud Object Storage	Active	–
Network (0)					
Cloud Foundry enterprise environments (0)					

Create a new Watson Studio project

1. From the Resources listing, click and open the **Watson Studio** service.
2. Click **Get Started**.
3. Create a **New project**.



Welcome, Gup!

Watson Studio • Try out other IBM Watson Studio apps

Learn by example
Step through solving a specific business problem in a sample project.
Take a guided tutorial

Work with data
Create a project for your team to prepare data, find insights, or build models.
Create a project

Extend your capabilities
Add tools, databases, or other features by creating services instances.
Create a service

Quick navigation
Projects
Deployments

Support
Documentation
FAQ
What's new
Give feedback
Stack overflow
Manage Tickets

Overview

Recent projects
No recent projects
After you create projects, you'll see your recently updated projects here.
New project +

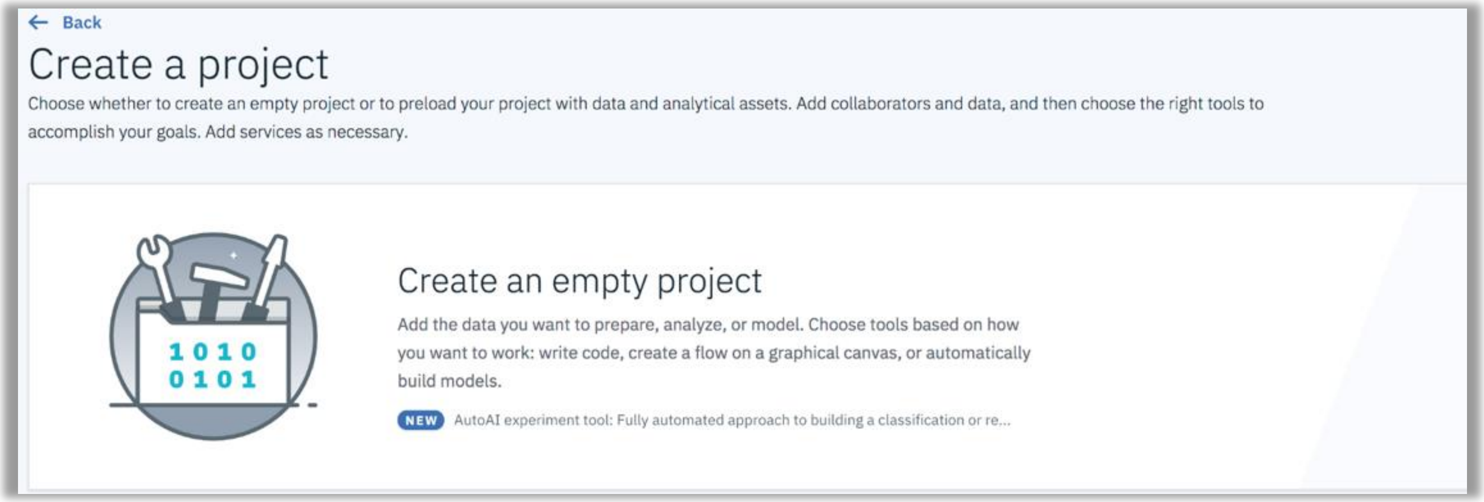
Notifications
No notifications
You will see your most recent notifications here.

Deployment spaces
No deployment spaces
After you create spaces, you'll see them here.
New deployment space +

Your services
Watson Studio-km
Watson Studio
Today at 10:21 PM
View all (1)

New in gallery
NOTEBOOK
IBM Cloud SQL Query
AUTHOR
IBM
MODIFIED
Jan 21, 2021
Explore

4. Select **Create an empty project** per below.



Back

Create a project

Choose whether to create an empty project or to preload your project with data and analytical assets. Add collaborators and data, and then choose the right tools to accomplish your goals. Add services as necessary.

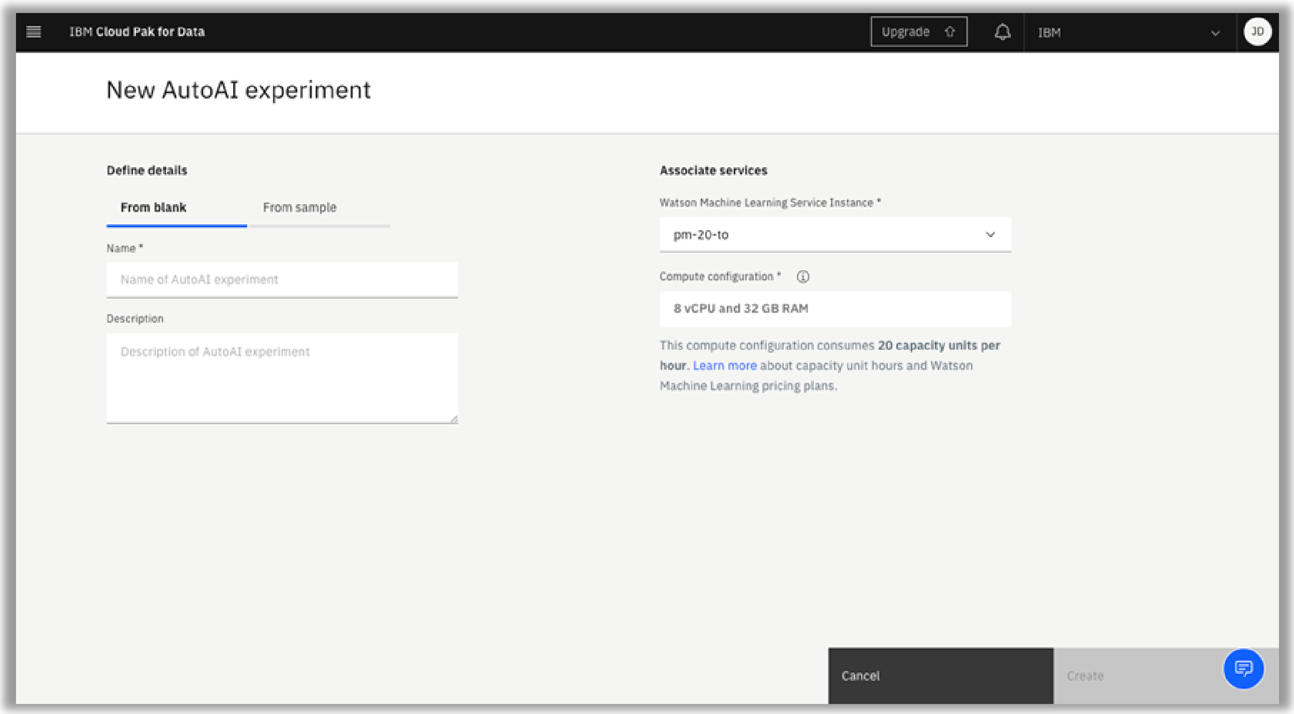
Create an empty project

Add the data you want to prepare, analyze, or model. Choose tools based on how you want to work: write code, create a flow on a graphical canvas, or automatically build models.

NEW AutoAI experiment tool: Fully automated approach to building a classification or re...

Figure 1-1 **Create a project**

5. Define the project by giving a meaningful name.


IBM Cloud Pak for Data

New AutoAI experiment

Define details

From blank | From sample

Name *

Name of AutoAI experiment

Description

Description of AutoAI experiment

Associate services

Watson Machine Learning Service Instance *

pm-20-to

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.

6. From the drop-down list under **Associate services**, find your previously created WML service and click **Associate service**. In the Watson Machine learning radio button.
7. Click **Reload**.
8. Click **Create**.

Create and define experiment

The AutoAI graphical tool in Watson Studio automatically analyzes your data and generates candidate model pipelines customized for your predictive modeling problem. These model pipelines are created iteratively as AutoAI analyzes your dataset and discovers data transformations, 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.

- **Required service**: Watson Machine Learning service
- **Data format**: Tabular: CSV files, with comma (,) delimiter
- **Data size**: Less than 1 GB

9. Click **Add to project** and click **AutoAI experiment**.

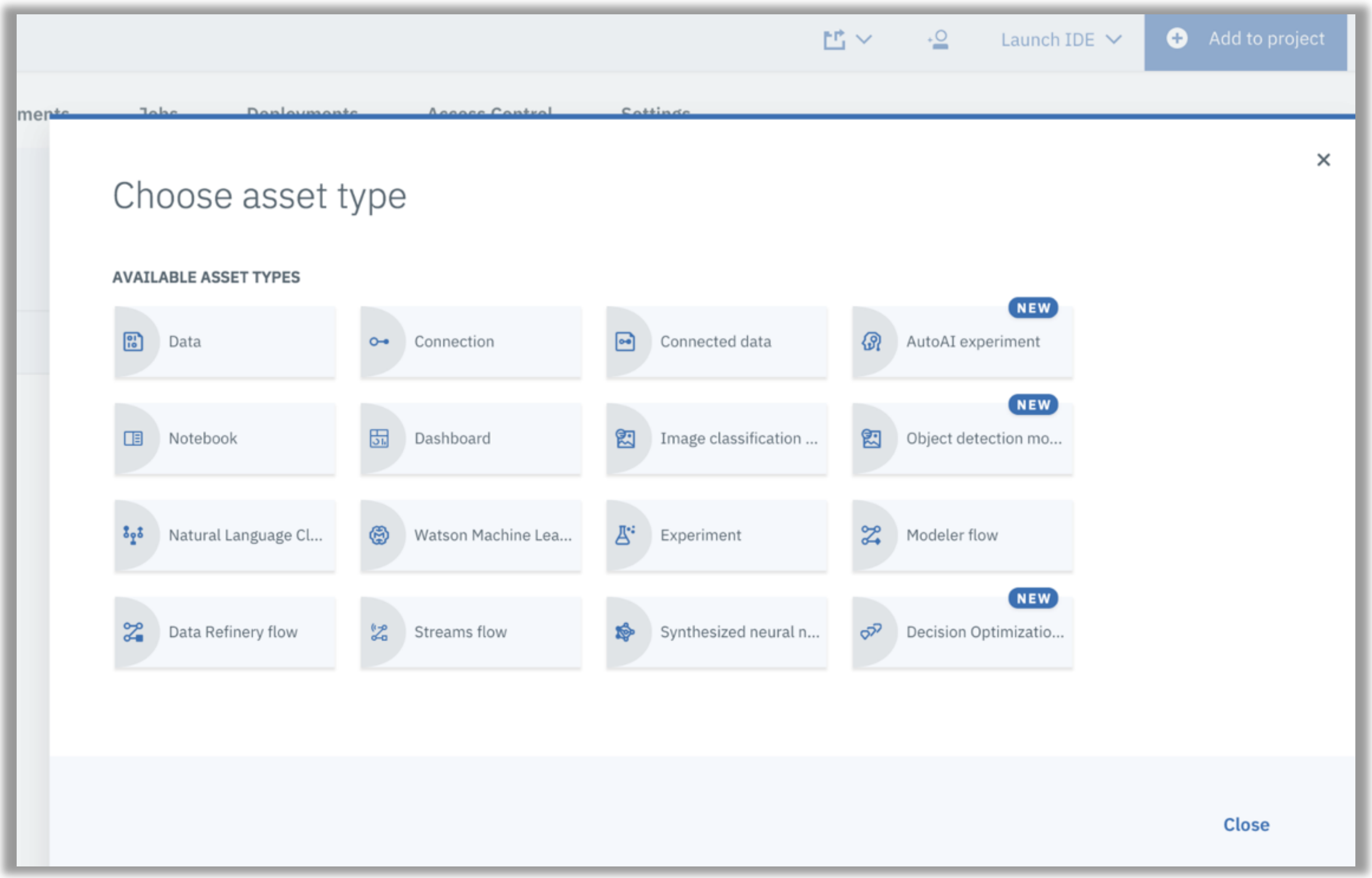


Figure 1-4 **Choose asset type**

Note: The Lite account for IBM Cloud comes with 50 capacity units per month and AutoAI consumes 20 capacity units per hour. This means you could redo this tutorial about 3 to 4 times over in any month and that would consume all 20 CUHs, and so now you have to wait until the month changes (or create another email account).

10. Specify a name for your project, for example “Credit score.”

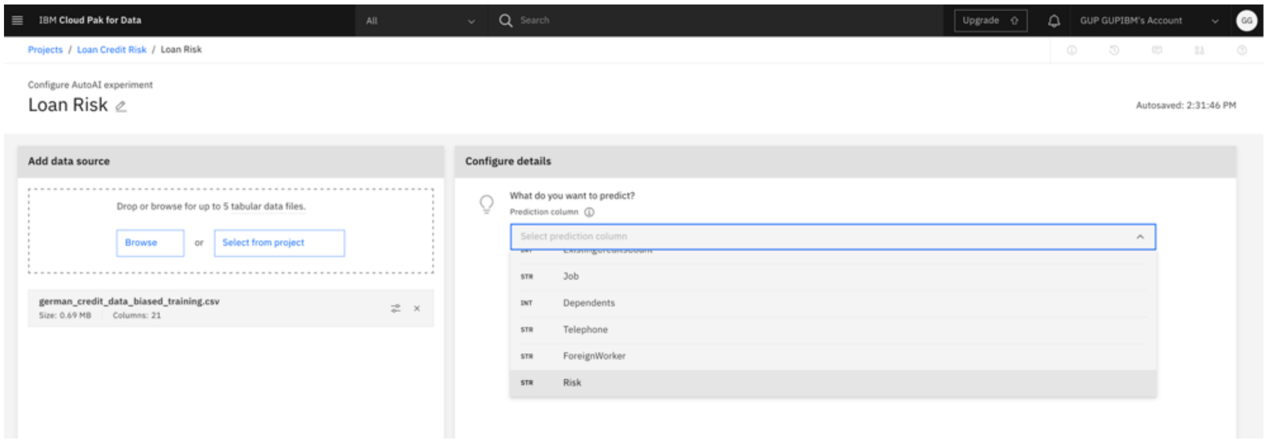
11. Associate a Machine Learning service instance to this project and select the Machine Learning service instance that you provisioned earlier from the drop-down list.

12. Click **Reload** and then click **Create** once the button at the bottom right is in focus.

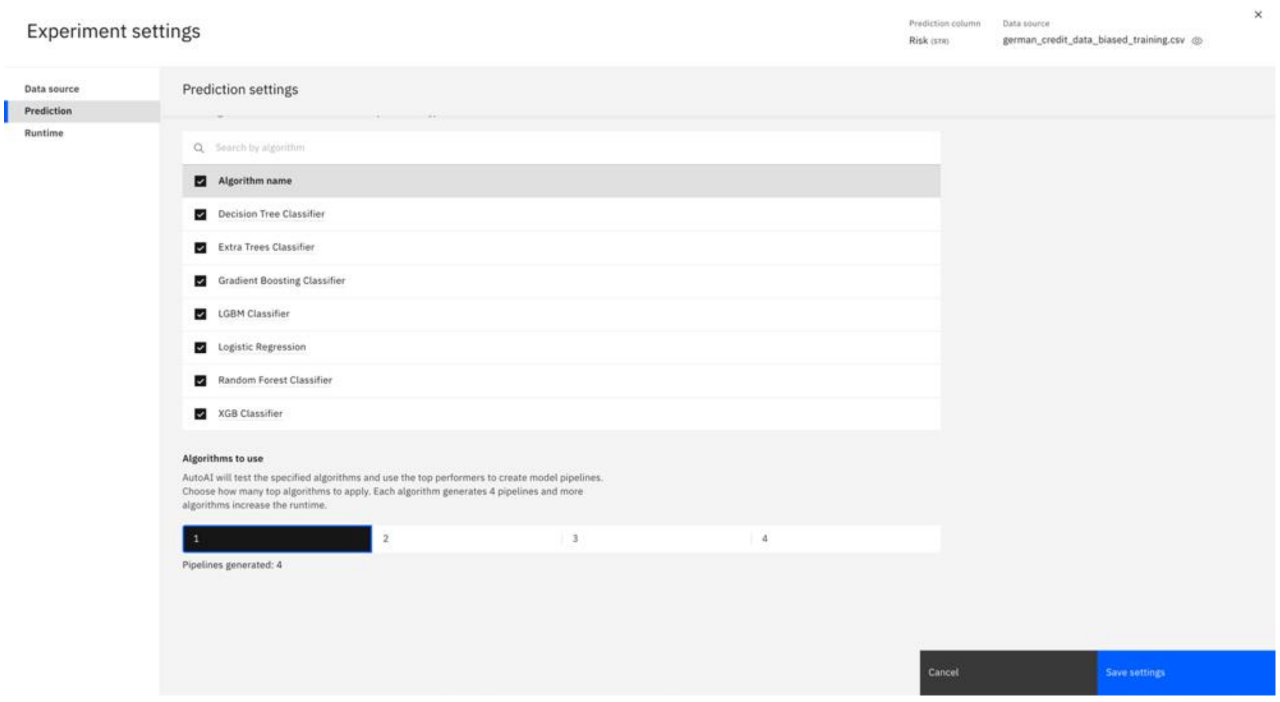
Import the csv file

You begin by importing a CSV file into the experiment. Note that only csv file format is supported in AutoAI.

13. Download this file to your local drive: german_credit_data_biased_training.csv
14. Click **Browse** to choose the downloaded file from previous step.
    Note: This data set is opensource and obtained from here: https://www.kaggle.com/uciml/german-credit.
15. From the drop-down list, scroll to the bottom and select **Risk** (column) as the ground truth, where you will peg other variables (columns) for further analysis.



16. Click the **Experiment settings** tab to adjust pipeline settings.
17. Click the **Prediction** link to the left and scroll down in the page and select **1** (instead of the default 2) pipelines to generate.



AutoAI will test the specified algorithms and use the top performers to create model pipelines. Choose how many top algorithms to apply. Each algorithm generates 4 pipelines and more algorithms increase the runtime.

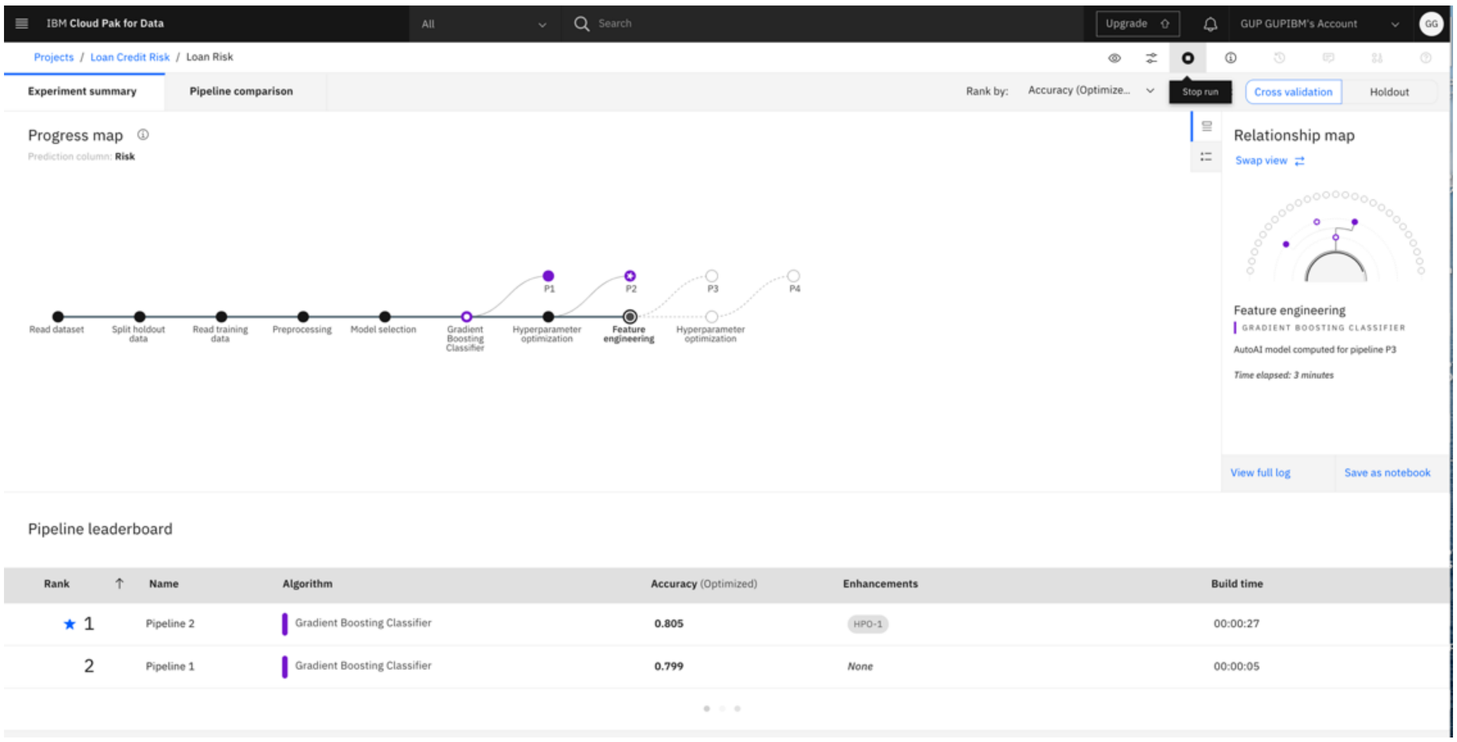
18. Save the settings and now you are ready to run the experiment.

19. Click **Run experiment**.

AutoAI automatically runs the following tasks to build and evaluate candidate model pipelines.

Each pipeline is run with different parameters. All these are done on the fly! Isn't it amazing that we just have to sit and watch while AutoAI takes care of things for us and generates awesome machine learning models!! There's very minimal intervention required to get things going and in no time, we have the generated pipelines to choose from.

1. In order to save your environment from consuming excessive CHUs, once the *second* pipeline has run and a star appears as the reigning model with an accuracy measure, click the **Stop** button in the top right of the menu bar and allow the experiment to stop. The generated pipeline may not have the best accuracy (had you let it run to the end), but it is good enough for this tutorial.



Feature engineering
GRADIENT BOOSTING CLASSIFIER
AutoAI model computed for pipeline P3
Time elapsed: 3 minutes

Pipeline leaderboard

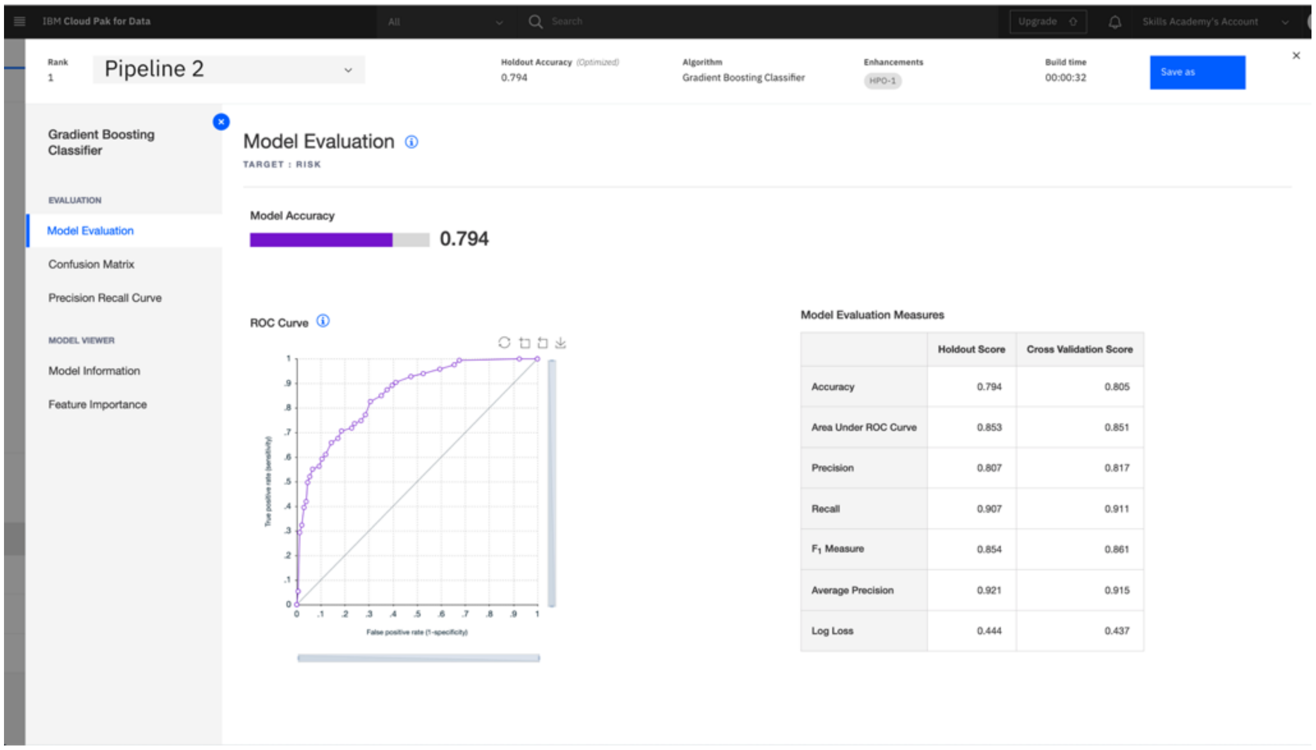
Rank ↑	Name	Algorithm	Accuracy (Optimized)	Enhancements	Build time
★ 1	Pipeline 2	Gradient Boosting Classifier	**0.805**	HPO-1	00:00:27
2	Pipeline 1	Gradient Boosting Classifier	**0.799**	*None*	00:00:05

Figure 1-11 **Analyze results**

2. After the experiment has stopped, click the winning pipeline (with the star), which is ranked 1, to see the evaluation metrics on the left side.

3. Click **Model Evaluation** to review the performance of the model on the hold out sample and

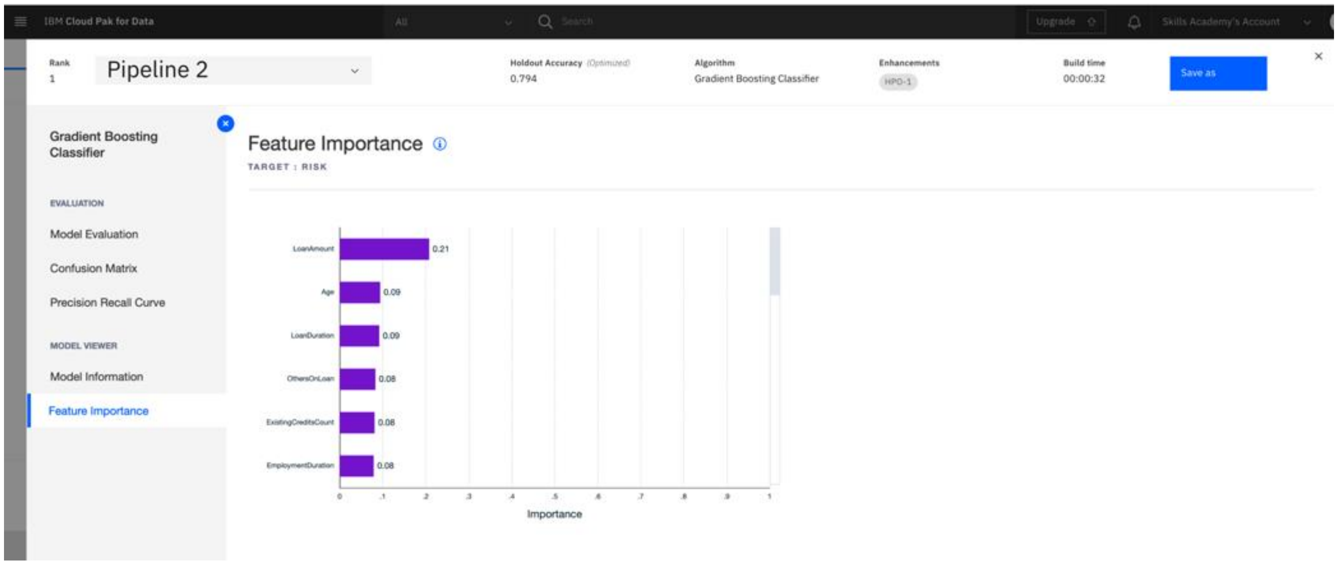
cross validation score.



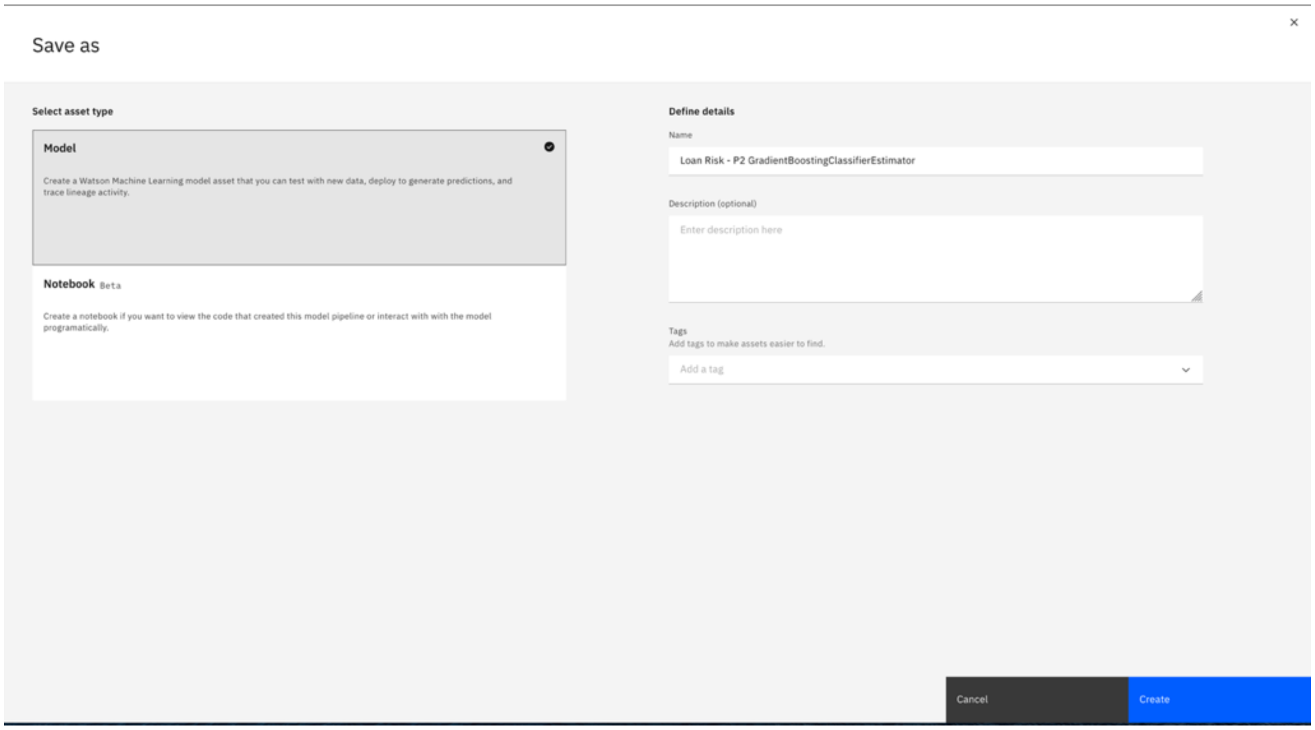
Model Evaluation Measures

	Holdout Score	Cross Validation Score
Accuracy	0.794	0.805
Area Under ROC Curve	0.853	0.851
Precision	0.807	0.817
Recall	0.907	0.911
F_1 Measure	0.854	0.861
Average Precision	0.921	0.915
Log Loss	0.444	0.437

4. Click **Feature Importance** to identify the significant features influencing the outcome. At a first glance, loan amount and age seem to play a large role in potential risk of not being able to pay back the loan. Later in this guide, once you analyze this model in Watson OpenScale, you will examine, Drift, Quality and Fairness, plus Explainability as to why Watson OpenScale made the forthcoming predictions and decisions.



5. Click the **Save as** button to the top right.
6. Select **Model** and click **Create**.


Save as

Select asset type

Model [✓]

Create a Watson Machine Learning model asset that you can test with new data, deploy to generate predictions, and trace lineage activity.

Notebook Beta

Create a notebook if you want to view the code that created this model pipeline or interact with with the model programatically.

Define details

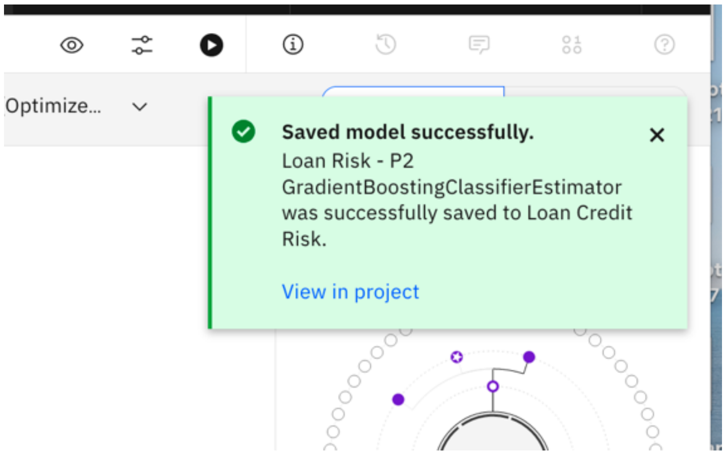
Name: Loan Risk - P2 GradientBoostingClassifierEstimator

Description (optional): Enter description here

Tags
Add tags to make assets easier to find.
Add a tag

Cancel | Create

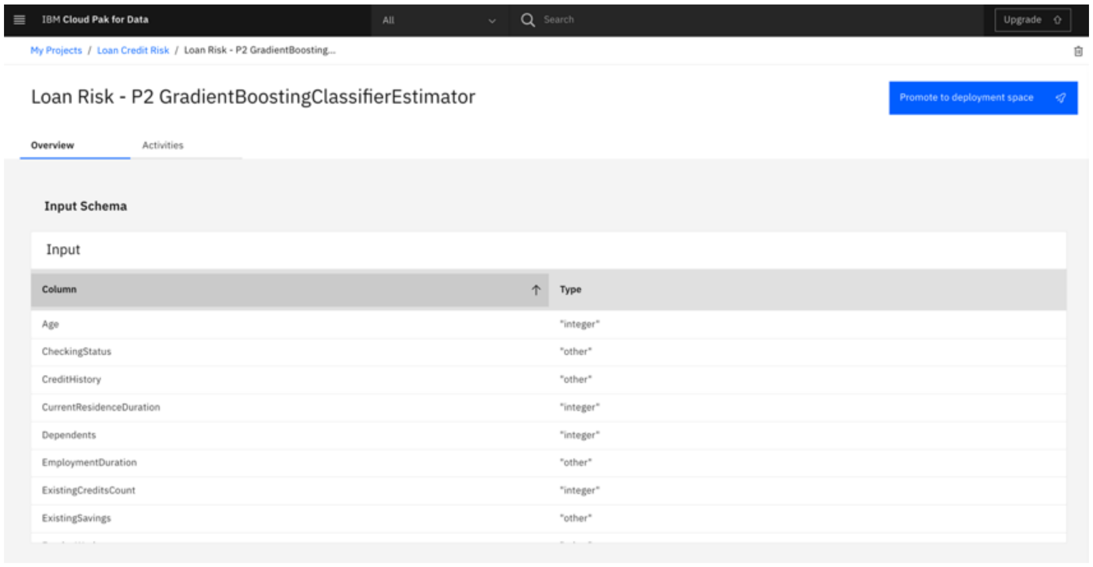
7. Keep an eye for the popup message box alerting you that the model is built and click **View in Projects**. If you miss the popup window, you can find the deployed model in your Watson Studio under the Assets tab.



Saved model successfully.
Loan Risk - P2 GradientBoostingClassifierEstimator was successfully saved to Loan Credit Risk.

View in project

8. Before deploying your model, you must create a deployment space. Click **Promote to deployment space**.


IBM Cloud Pak for Data
All
Search
Upgrade
My Projects / Loan Credit Risk / Loan Risk - P2 GradientBoosting...
Promote to deployment space
Overview
Activities

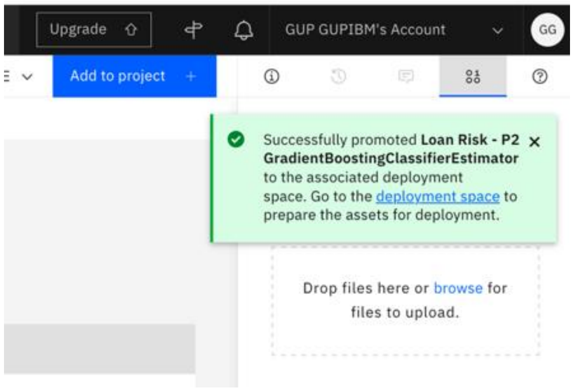
Loan Risk - P2 GradientBoostingClassifierEstimator

Input Schema

Input

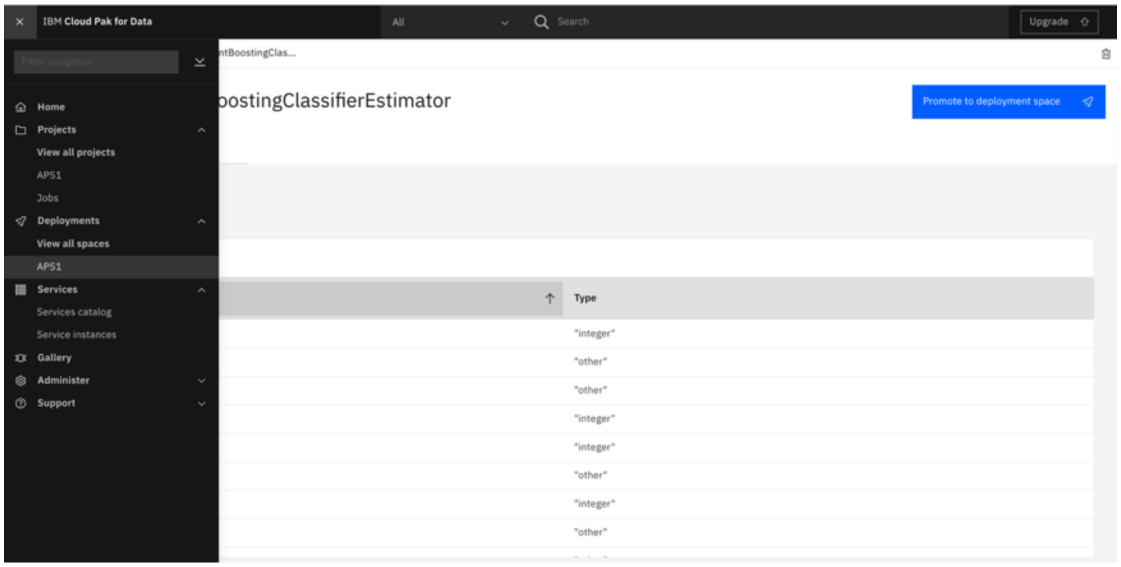
Column	Type
Age	"integer"
CheckingStatus	"other"
CreditHistory	"other"
CurrentResidenceDuration	"integer"
Dependents	"integer"
EmploymentDuration	"other"
ExistingCreditsCount	"integer"
ExistingSavings	"other"

9. Keep an eye on yet another pop-up message and when prompted, click **deployment space** from inside the message.



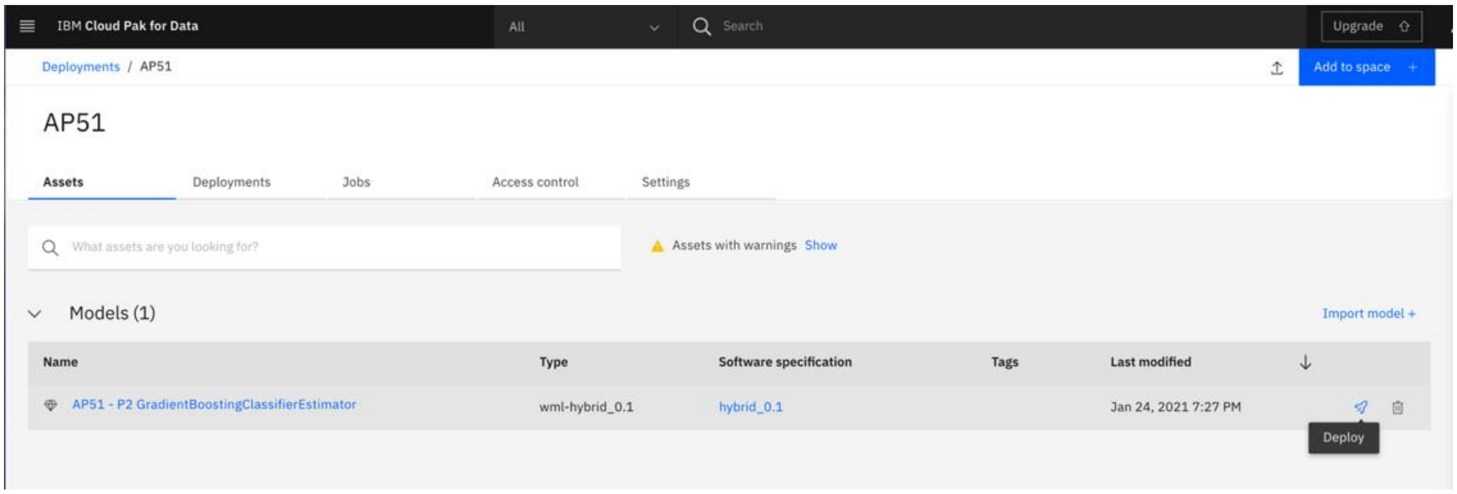
Successfully promoted **Loan Risk - P2 GradientBoostingClassifierEstimator** to the associated deployment space. Go to the deployment space to prepare the assets for deployment.

Note: If you missed the pop-up message, you can find it from the hamburger icon from the top left per below:



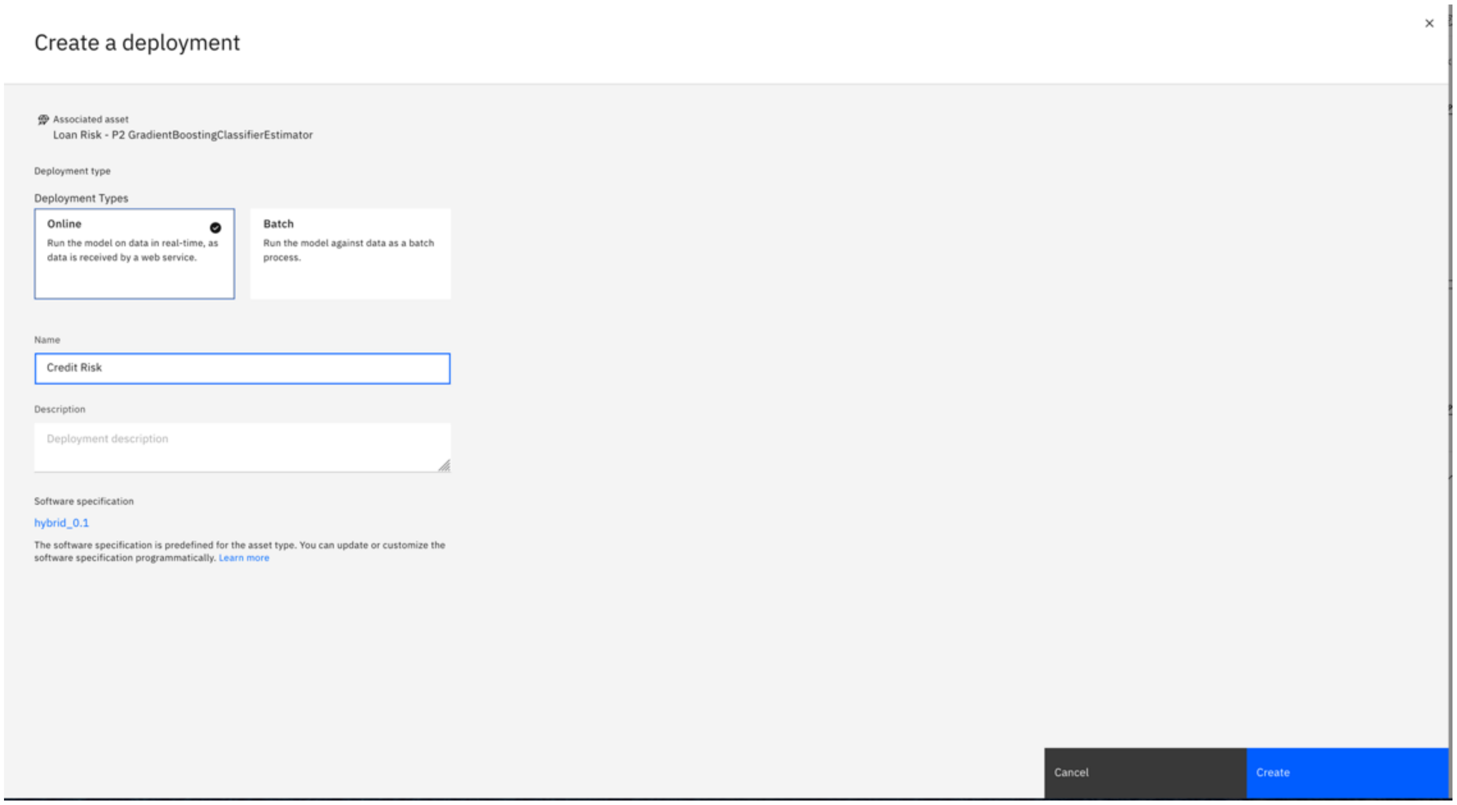
Now that you have created your deployment space, it is time to deploy the model in that container.

10. To the right of the model name, click the **Deploy** icon.



Name	Type	Software specification	Tags	Last modified
AP51 - P2 GradientBoostingClassifierEstimator	wml-hybrid_0.1	hybrid_0.1		Jan 24, 2021 7:27 PM

11. Select **Online**, specify a meaningful name and click **Create**.


Create a deployment

Associated asset
Loan Risk - P2 GradientBoostingClassifierEstimator

Deployment type

Deployment Types

Online
Run the model on data in real-time, as data is received by a web service.

Batch
Run the model against data as a batch process.

Name
Credit Risk

Description
Deployment description

Software specification
hybrid_0.1

The software specification is predefined for the asset type. You can update or customize the software specification programmatically. Learn more

12. Click the **Test** tab and notice that by inserting values (similar to the value type that appear in the cells in the CSV file for each corresponding column); however, that could be a time-consuming event. Instead, you can copy and paste a JSON file and click Predict on that “new” data set, never seen before by the system (not in the training data). More on that later.

You are now ready to configure the Watson OpenScale service.

Milestone 2: Configuring Watson OpenScale

IBM® Watson OpenScale tracks and measures outcomes from your AI models, and helps ensure they remain fair, explainable, and compliant wherever your models were built or are running. Watson OpenScale also detects and helps correct the drift in accuracy when an AI model is in production

Enterprises use IBM® Watson OpenScale to automate and put into service AI lifecycle in business applications. This approach ensures that AI models are free from bias, can be easily explained and understood by business users, and are auditable in business transactions. Watson OpenScale supports AI models built and run in the tools and model serve frameworks of your choice.

Traditional lenders are under pressure to expand their digital portfolio of financial services to a larger and more diverse audience, which requires a new approach to credit risk modeling. Their data science teams currently rely on standard modeling techniques - like decision trees and logistic regression - which work well for moderate data sets and make recommendations that can be easily explained. This approach satisfies regulatory requirements that credit lending decisions must be apparent and explainable.

To provide credit access to a wider and riskier population, applicant credit histories must expand beyond traditional lines of credit. In addition to mortgages and car loans, lenders need alternative credit sources, such as utility and mobile phone plan payment histories, plus education and job titles. These new data sources offer promise, but also introduce risk by increasing the likelihood of unexpected correlations that introduce bias based on an applicant’s age, gender, or other personal traits.

The data science techniques that are most suited to these diverse data sets, such as gradient boosted trees and neural networks, can generate highly accurate risk models, but at a cost. Such models, without explanation of the inner workings, generate opaque predictions that must somehow become apparent. You must ensure regulatory approval. Article 22 of the General Data Protection Regulation (GDPR) requires explainability. The United States Fair Credit Reporting Act (FCRA) that is managed by the Consumer Financial Protection Bureau also requires this level of accountability.

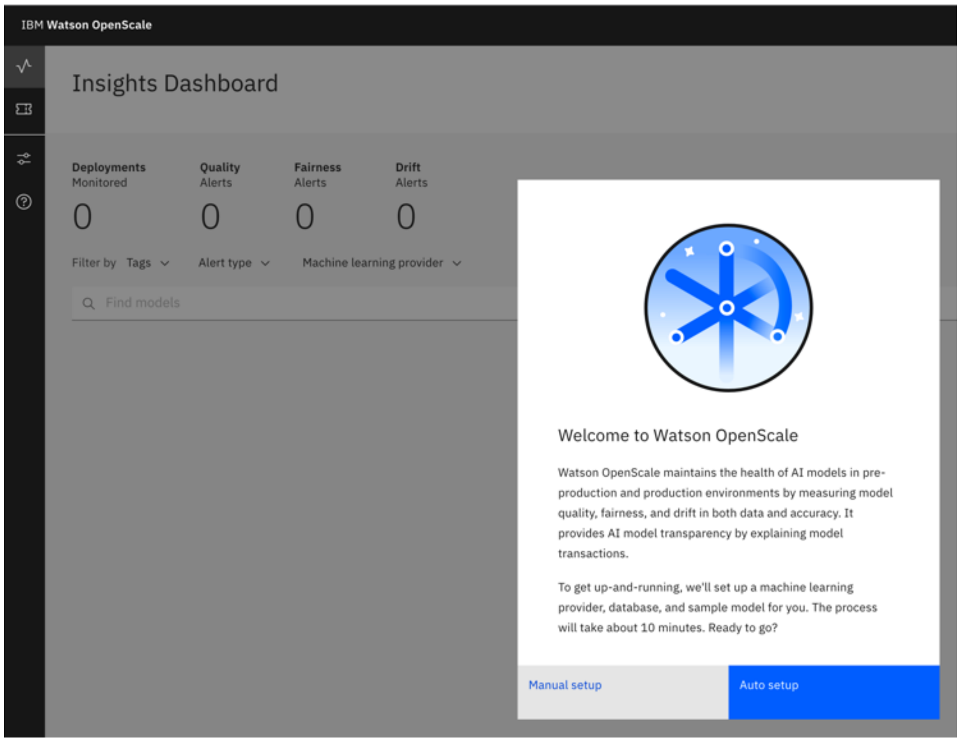
The credit risk model that is provided in this tutorial uses a training data set that contains 20 attributes about each loan applicant. Two of those attributes - age and sex - can be tested for bias. For this tutorial, the focus is on bias against sex and age.

Watson OpenScale monitors the deployed model's propensity for a favorable outcome ("No Risk") for one group (the Reference Group) over another (the Monitored Group). In this tutorial, the Monitored Group for sex is `female`, while the Monitored Group for age is `19 to 25`.

Now that the machine learning model is deployed, you can configure Watson OpenScale to ensure trust and transparency with your models.

Complete the following steps to configure Watson OpenScale.

1. From the Resource list of your IBM Cloud account, click and open the **Watson OpenScale** service.
2. Click **Launch application**.
3. Select **Manual Setup**.

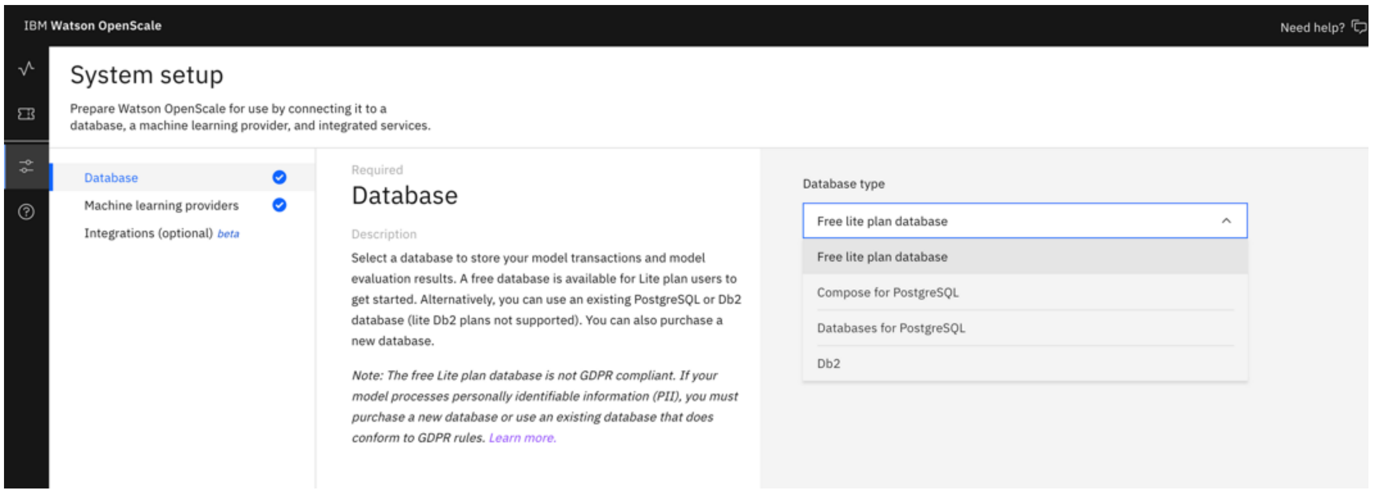


Welcome to Watson OpenScale

Watson OpenScale maintains the health of AI models in pre-production and production environments by measuring model quality, fairness, and drift in both data and accuracy. It provides AI model transparency by explaining model transactions.

To get up-and-running, we'll set up a machine learning provider, database, and sample model for you. The process will take about 10 minutes. Ready to go?

4. Click the third icon, **System setup** and begin with the **Database connectivity** link and select the Free Lite Plan if already not selected by default by Watson OpenScale.

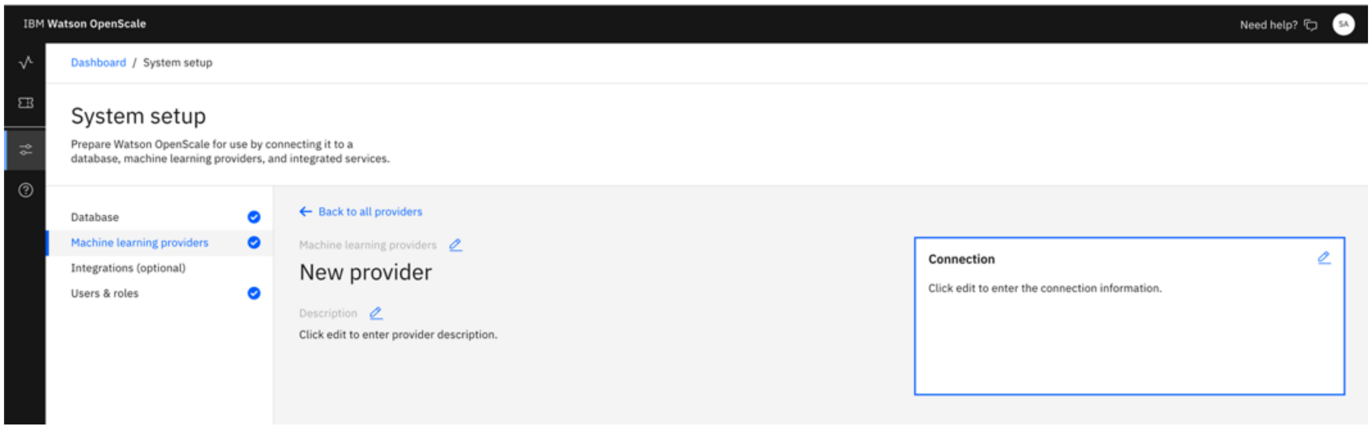


Prepare Watson OpenScale for use by connecting it to a database, a machine learning provider, and integrated services.

Select a database to store your model transactions and model evaluation results. A free database is available for Lite plan users to get started. Alternatively, you can use an existing PostgreSQL or Db2 database (lite Db2 plans not supported). You can also purchase a new database.

Note: The free Lite plan database is not GDPR compliant. If your model processes personally identifiable information (PII), you must purchase a new database or use an existing database that does conform to GDPR rules. Learn more.

5. Click the third menu icon, **System setup** and begin with the **Machine learning providers** link.


IBM Watson OpenScale

Dashboard / System setup

System setup

Prepare Watson OpenScale for use by connecting it to a database, machine learning providers, and integrated services.

Database
Machine learning providers
Integrations (optional)
Users & roles

Back to all providers

Machine learning providers

New provider

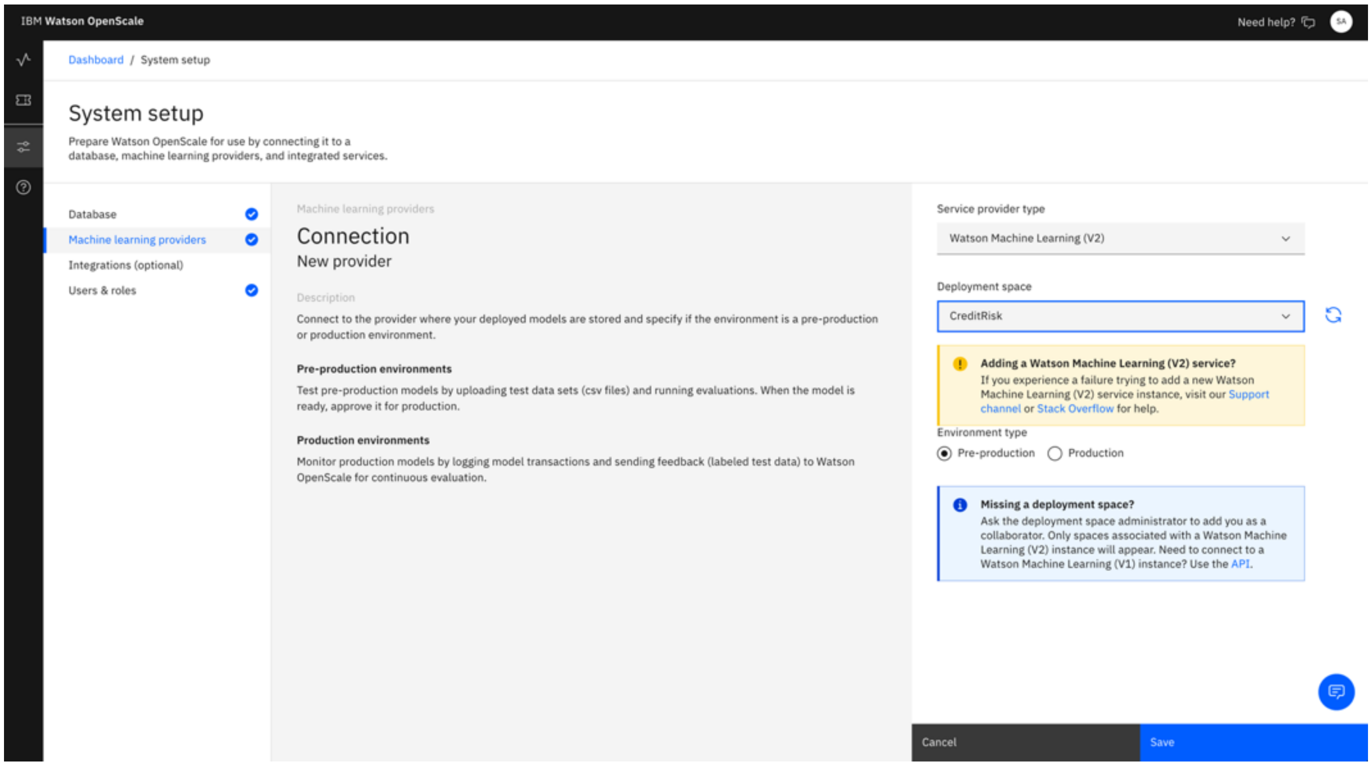
Description

Click edit to enter provider description.

Connection

Click edit to enter the connection information.

6. Use the pencil icon to edit the name of your Machine Learning provider to something meaningful to you (instead of New Provider) and click **Apply**.
7. Click the pencil icon in the Connections box to specify **Watson Machine Learning (V2),** select your deployment space (your space may have different name) from the drop-down and click **Pre-production** and click **Save**.


IBM Watson OpenScale

Dashboard / System setup

System setup

Prepare Watson OpenScale for use by connecting it to a database, machine learning providers, and integrated services.

Database
Machine learning providers
Integrations (optional)
Users & roles

Machine learning providers

Connection

New provider

Description

Connect to the provider where your deployed models are stored and specify if the environment is a pre-production or production environment.

Pre-production environments

Test pre-production models by uploading test data sets (csv files) and running evaluations. When the model is ready, approve it for production.

Production environments

Monitor production models by logging model transactions and sending feedback (labeled test data) to Watson OpenScale for continuous evaluation.

Service provider type: Watson Machine Learning (V2)

Deployment space: CreditRisk

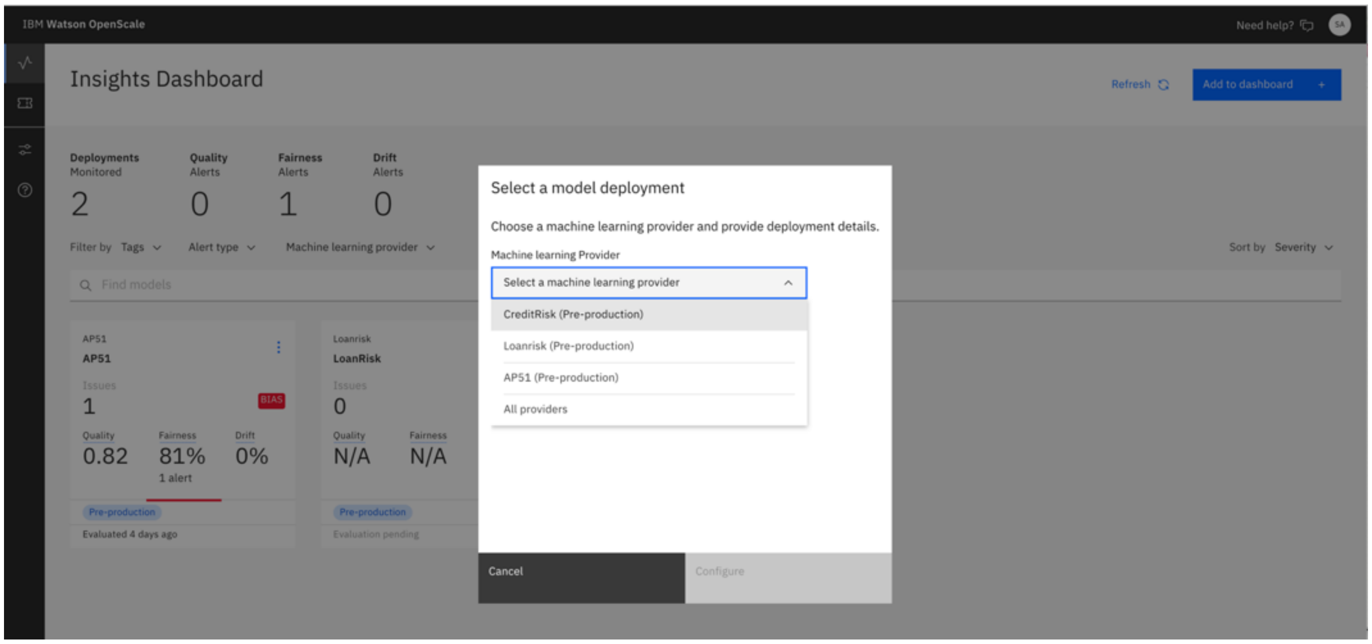
Adding a Watson Machine Learning (V2) service? If you experience a failure trying to add a new Watson Machine Learning (V2) service instance, visit our Support channel or Stack Overflow for help.

Environment type: [✓ Pre-production] Production

Missing a deployment space? Ask the deployment space administrator to add you as a collaborator. Only spaces associated with a Watson Machine Learning (V2) instance will appear. Need to connect to a Watson Machine Learning (V1) instance? Use the API.

Cancel Save

8. Select the machine learning provider you named earlier (the image below is based on numerous exercise…you'll likely have only one model name appearing there).



Select a model deployment

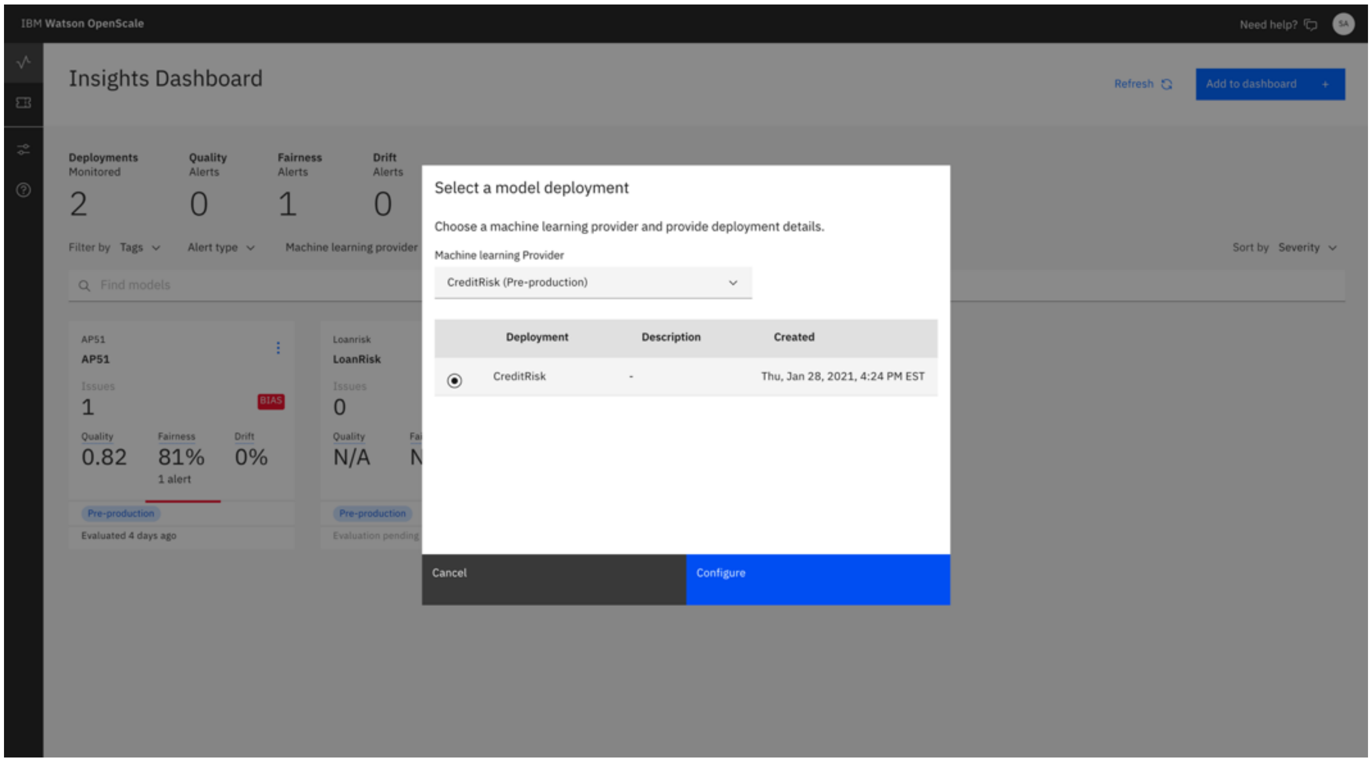
Choose a machine learning provider and provide deployment details.

Machine learning Provider

Select a machine learning provider

- CreditRisk (Pre-production)
- Loanrisk (Pre-production)
- AP51 (Pre-production)
- All providers

9. Select the radio button depicting the ML provider and click **Configure**.



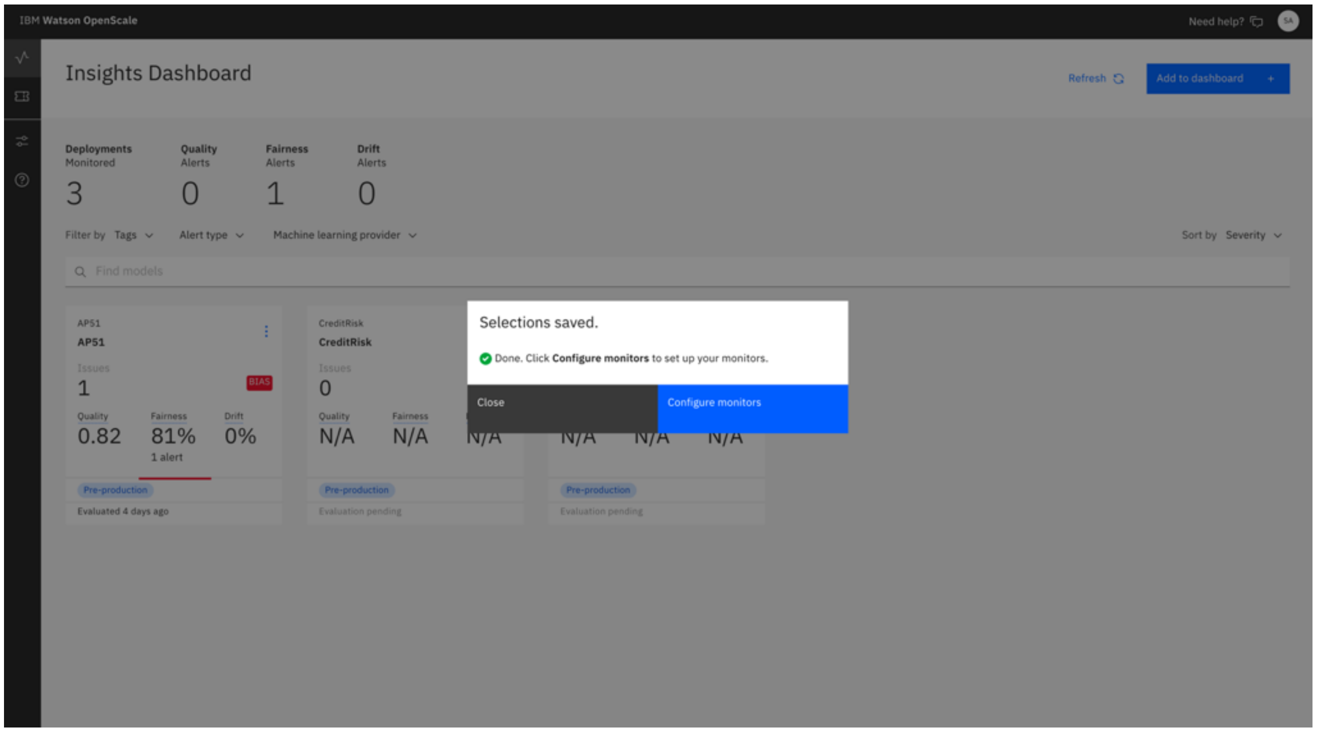
Select a model deployment

Choose a machine learning provider and provide deployment details.

Machine learning Provider: CreditRisk (Pre-production)

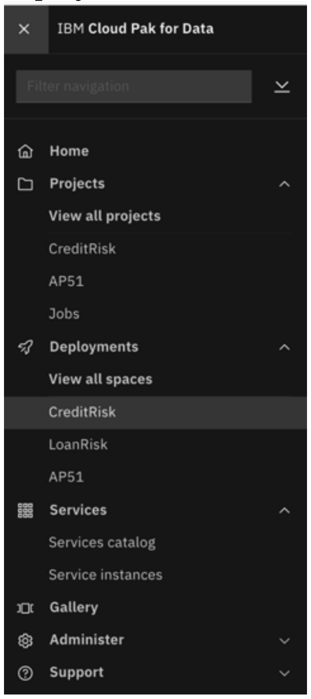
	Deployment	Description	Created
●	CreditRisk	-	Thu, Jan 28, 2021, 4:24 PM EST

10. Click **Configure monitors**.

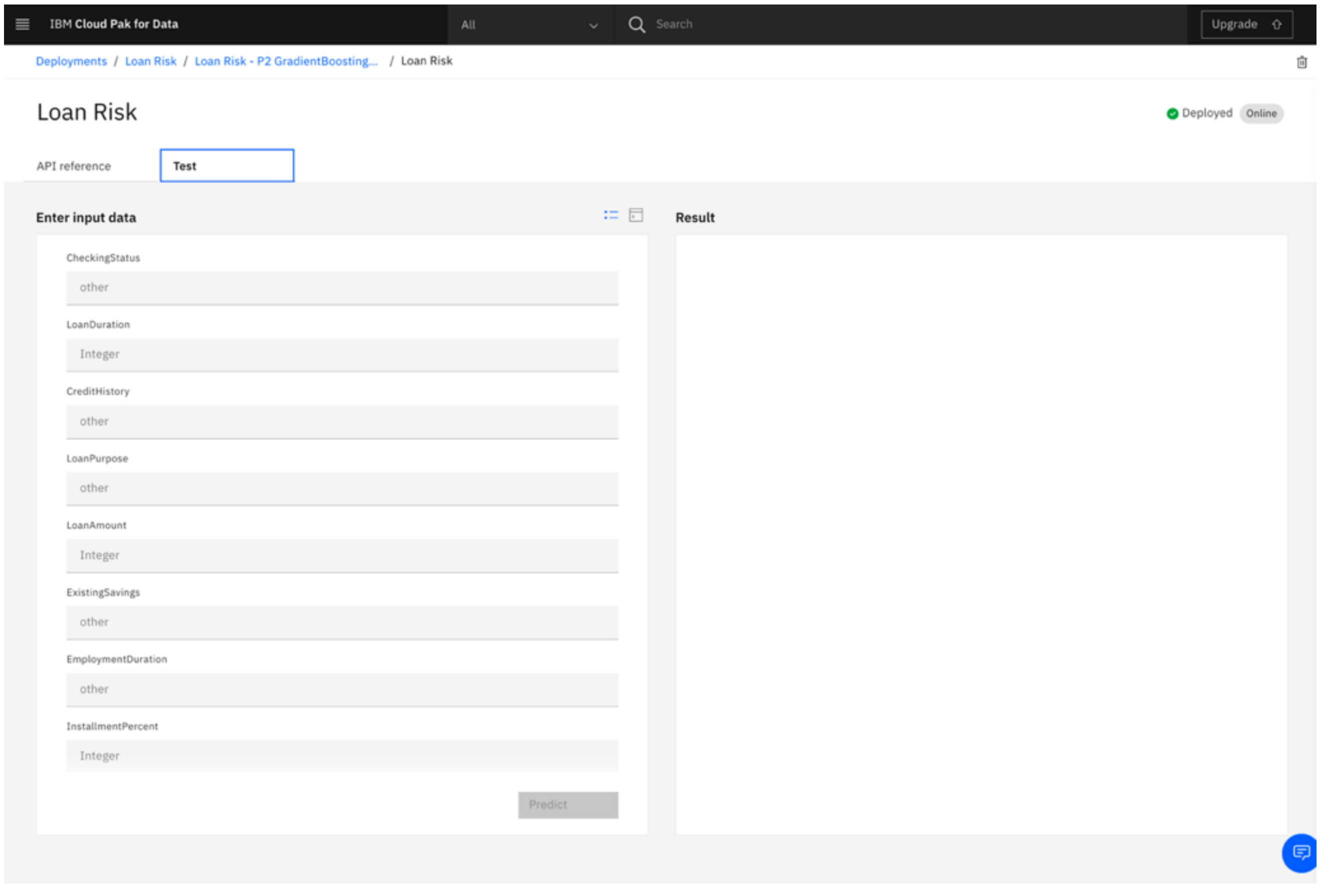


Before you configure your monitors, you can generate a scoring request against your model to test payload logging that the monitors can process. You must provide sample data to Watson Studio in the form of a JSON file to generate a scoring request. Later in the tutorial, repeat the scoring request to provide the actual data to the Watson OpenScale monitors.

11. Go back to Watson Studio and open the **Deployments** tab or from the left panel select deployments and then the name of your deployment.



12. Once you click the deployment name, then click the **Test** tab.


IBM Cloud Pak for Data

Deployments / Loan Risk / Loan Risk - P2 GradientBoosting... / Loan Risk

Loan Risk

Deployed Online

API reference | Test

Enter input data

CheckingStatus: other

LoanDuration: Integer

CreditHistory: other

LoanPurpose: other

LoanAmount: Integer

ExistingSavings: other

EmploymentDuration: other

InstallmentPercent: Integer

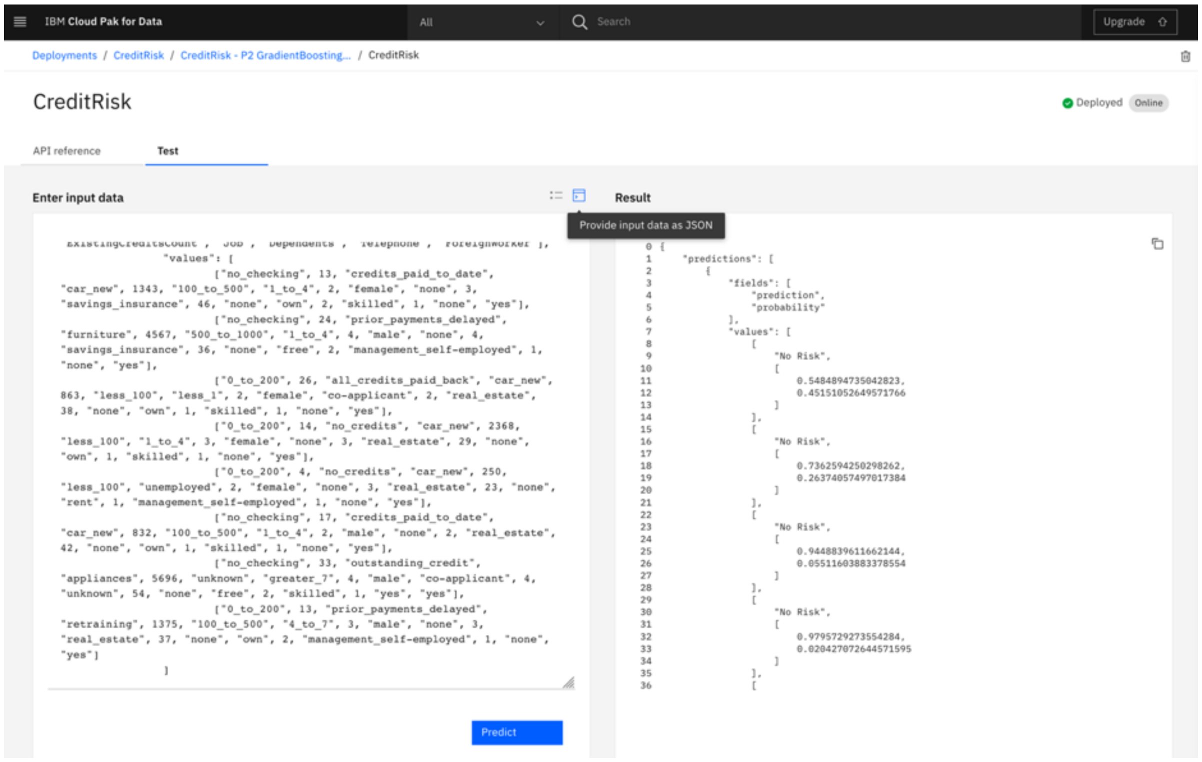
Predict

Result

1. Copy the following JSON snippet:

```
{
      "input_data": [{
                 "fields": ["CheckingStatus", "LoanDuration", "CreditHistory", "LoanPurpose",
"LoanAmount", "ExistingSavings", "EmploymentDuration", "InstallmentPercent", "Sex", "OthersOnLoan",
"CurrentResidenceDuration", "OwnsProperty", "Age", "InstallmentPlans", "Housing",
"ExistingCreditsCount", "Job", "Dependents", "Telephone", "ForeignWorker"],
                 "values": [
                            ["no_checking", 13, "credits_paid_to_date", "car_new", 1343, "100_to_500",
"1_to_4", 2, "female", "none", 3, "savings_insurance", 46, "none", "own", 2, "skilled", 1, "none",
"yes"],
                            ["no_checking", 24, "prior_payments_delayed", "furniture", 4567,
"500_to_1000", "1_to_4", 4, "male", "none", 4, "savings_insurance", 36, "none", "free", 2,
"management_self-employed", 1, "none", "yes"],
                            ["0_to_200", 26, "all_credits_paid_back", "car_new", 863, "less_100",
"less_1", 2, "female", "co-applicant", 2, "real_estate", 38, "none", "own", 1, "skilled", 1, "none",
"yes"],
                            ["0_to_200", 14, "no_credits", "car_new", 2368, "less_100", "1_to_4", 3,
"female", "none", 3, "real_estate", 29, "none", "own", 1, "skilled", 1, "none", "yes"],
                            ["0_to_200", 4, "no_credits", "car_new", 250, "less_100", "unemployed", 2,
"female", "none", 3, "real_estate", 23, "none", "rent", 1, "management_self-employed", 1, "none",
"yes"],
                            ["no_checking", 17, "credits_paid_to_date", "car_new", 832, "100_to_500",
"1_to_4", 2, "male", "none", 2, "real_estate", 42, "none", "own", 1, "skilled", 1, "none", "yes"],
                            ["no_checking", 33, "outstanding_credit", "appliances", 5696, "unknown",
"greater_7", 4, "male", "co-applicant", 4, "unknown", 54, "none", "free", 2, "skilled", 1, "yes",
"yes"],
                            ["0_to_200", 13, "prior_payments_delayed", "retraining", 1375, "100_to_500",
"4_to_7", 3, "male", "none", 3, "real_estate", 37, "none", "own", 2, "management_self-employed", 1,
"none", "yes"]
                ]
      }]
}
```

2. In the **Test** tab select the **Provide input data as JSON** and paste the JSON snippet in the left window.



```
"ExistingCreditsCount", "Job", "Dependents", "Telephone", "ForeignWorker"],
                "values": [
                        ["no_checking", 13, "credits_paid_to_date",
"car_new", 1343, "100_to_500", "1_to_4", 2, "female", "none", 3,
"savings_insurance", 46, "none", "own", 2, "skilled", 1, "none", "yes"],
                        ["no_checking", 24, "prior_payments_delayed",
"furniture", 4567, "500_to_1000", "1_to_4", 4, "male", "none", 4,
"savings_insurance", 36, "none", "free", 2, "management_self-employed", 1,
"none", "yes"],
                        ["0_to_200", 26, "all_credits_paid_back", "car_new",
863, "less_100", "less_1", 2, "female", "co-applicant", 2, "real_estate",
38, "none", "own", 1, "skilled", 1, "none", "yes"],
                        ["0_to_200", 14, "no_credits", "car_new", 2368,
"less_100", "1_to_4", 3, "female", "none", 3, "real_estate", 29, "none",
"own", 1, "skilled", 1, "none", "yes"],
                        ["0_to_200", 4, "no_credits", "car_new", 250,
"less_100", "unemployed", 2, "female", "none", 3, "real_estate", 23, "none",
"rent", 1, "management_self-employed", 1, "none", "yes"],
                        ["no_checking", 17, "credits_paid_to_date",
"car_new", 832, "100_to_500", "1_to_4", 2, "male", "none", 2, "real_estate",
42, "none", "own", 1, "skilled", 1, "none", "yes"],
                        ["no_checking", 33, "outstanding_credit",
"appliances", 5696, "unknown", "greater_7", 4, "male", "co-applicant", 4,
"unknown", 54, "none", "free", 2, "skilled", 1, "yes", "yes"],
                        ["0_to_200", 13, "prior_payments_delayed",
"retraining", 1375, "100_to_500", "4_to_7", 3, "male", "none", 3,
"real_estate", 37, "none", "own", 2, "management_self-employed", 1, "none",
"yes"]
                ]
```

```
{
    "predictions": [
        {
            "fields": [
                "prediction",
                "probability"
            ],
            "values": [
                [
                    "No Risk",
                    [
                        0.5484894735042823,
                        0.45151052649571766
                    ]
                ],
                [
                    "No Risk",
                    [
                        0.7362594250298262,
                        0.26374057497017384
                    ]
                ],
                [
                    "No Risk",
                    [
                        0.9448839611662144,
                        0.05511603883378554
                    ]
                ],
                [
                    "No Risk",
                    [
                        0.9795729273554284,
                        0.020427072644571595
                    ]
                ],
                [
```

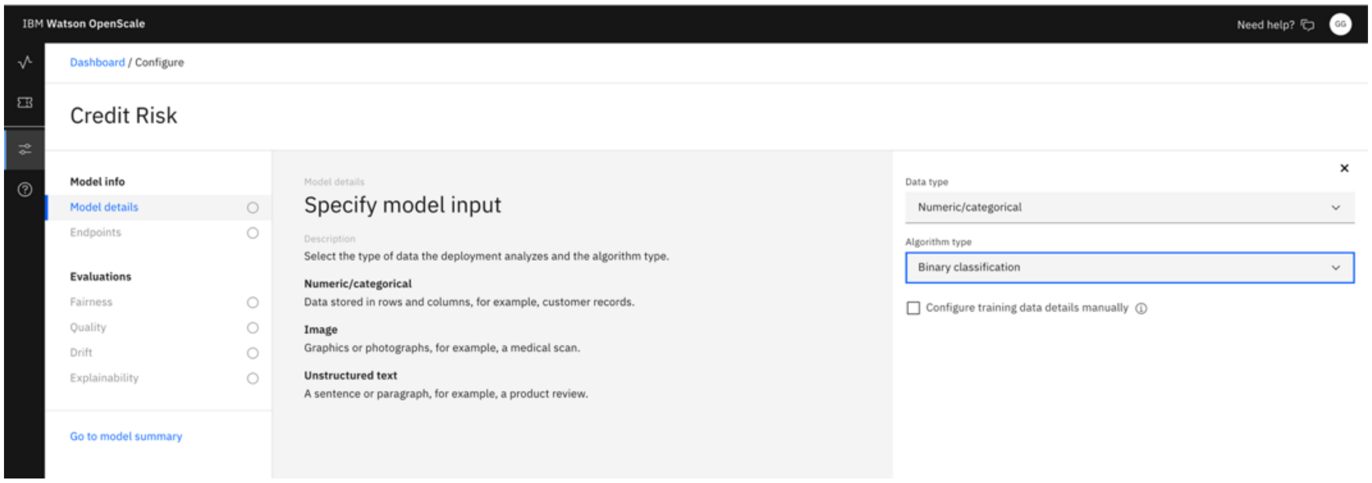
3. Click **Predict**.

In the resulting data, scroll down until you see a Risk assessment. Let's see how Watson OpenScale analyzes new data that may cause drifts in accuracy and which attributes are the culprit.

Configuring Model Details in Watson OpenScale

Now that the machine learning model is deployed, you can configure Watson OpenScale to ensure trust and transparency with your models. Complete the following steps:

1. Refer back to the Watson OpenScale service.
2. Click the top icon, **Dashboard**, and click **Add to Dashboard**.
3. Select your Machine Learning Provider and click **Configure**.
4. Click the three dots in the Pre-Production tile that appears on your dashboard and select **Configure monitors**.
5. From the Model details menu, select: **Numerical/categorical** for Data type (Risk, no-Risk) and **Binary classification** for Algorithm type.



Credit Risk

Model details

Specify model input

Description

Select the type of data the deployment analyzes and the algorithm type.

Numeric/categorical
Data stored in rows and columns, for example, customer records.

Image
Graphics or photographs, for example, a medical scan.

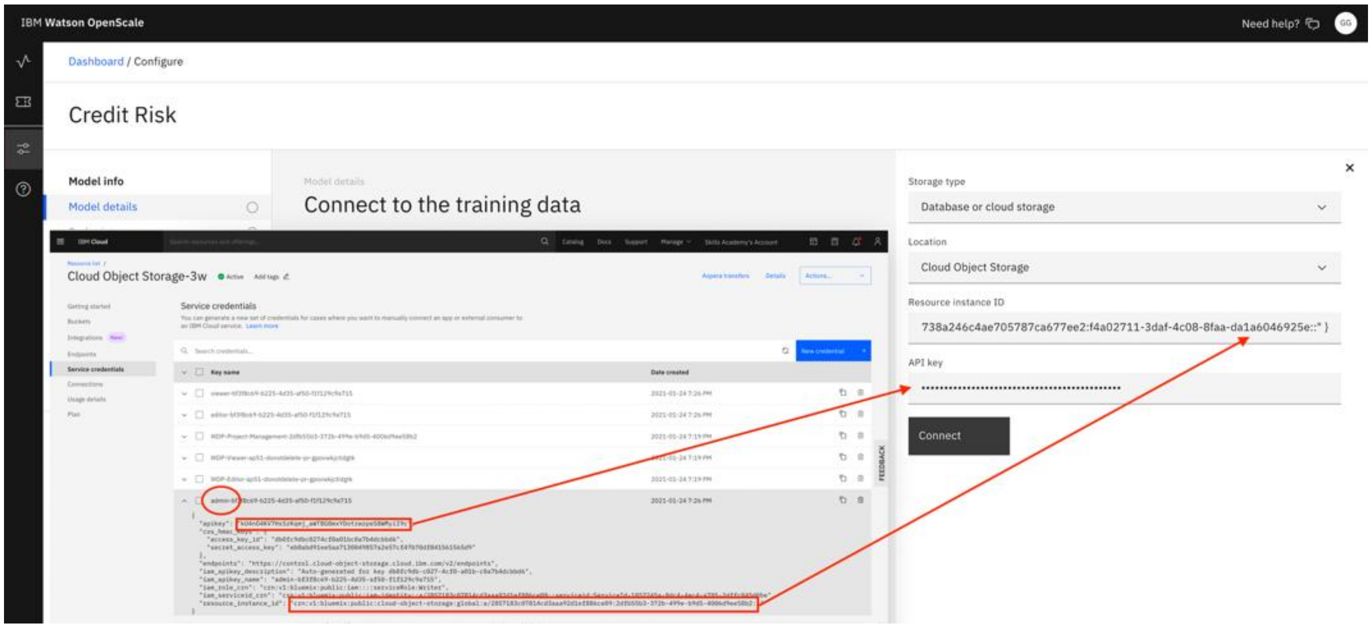
Unstructured text
A sentence or paragraph, for example, a product review.

Data type: Numeric/categorical

Algorithm type: Binary classification

☐ Configure training data details manually

6. You must now provide connectivity details pertaining to the Cloud Object Storage. To do so, first go back to the Resources list and click open the **Cloud Object Storage** service.
7. Open the **Service credentials** link.
8. The final entry (admin) has both the Resource instance ID and the API key. Copy and paste both values in corresponding fields in Watson Open Scale service.


IBM Watson OpenScale — Need help?

Dashboard / Configure

Credit Risk

Model info
Model details

Model details
Connect to the training data

Storage type: Database or cloud storage
Location: Cloud Object Storage
Resource instance ID: 738a246c4ae705787ca677ee2:f4a02711-3daf-4c08-8faa-da1a6046925e::" }
API key: ••
Connect

IBM Cloud — Catalog, Docs, Support, Manage, Skills Academy's Account

Resource list /
Cloud Object Storage-3w ● Active Add tags
Aspera transfers Details Actions...

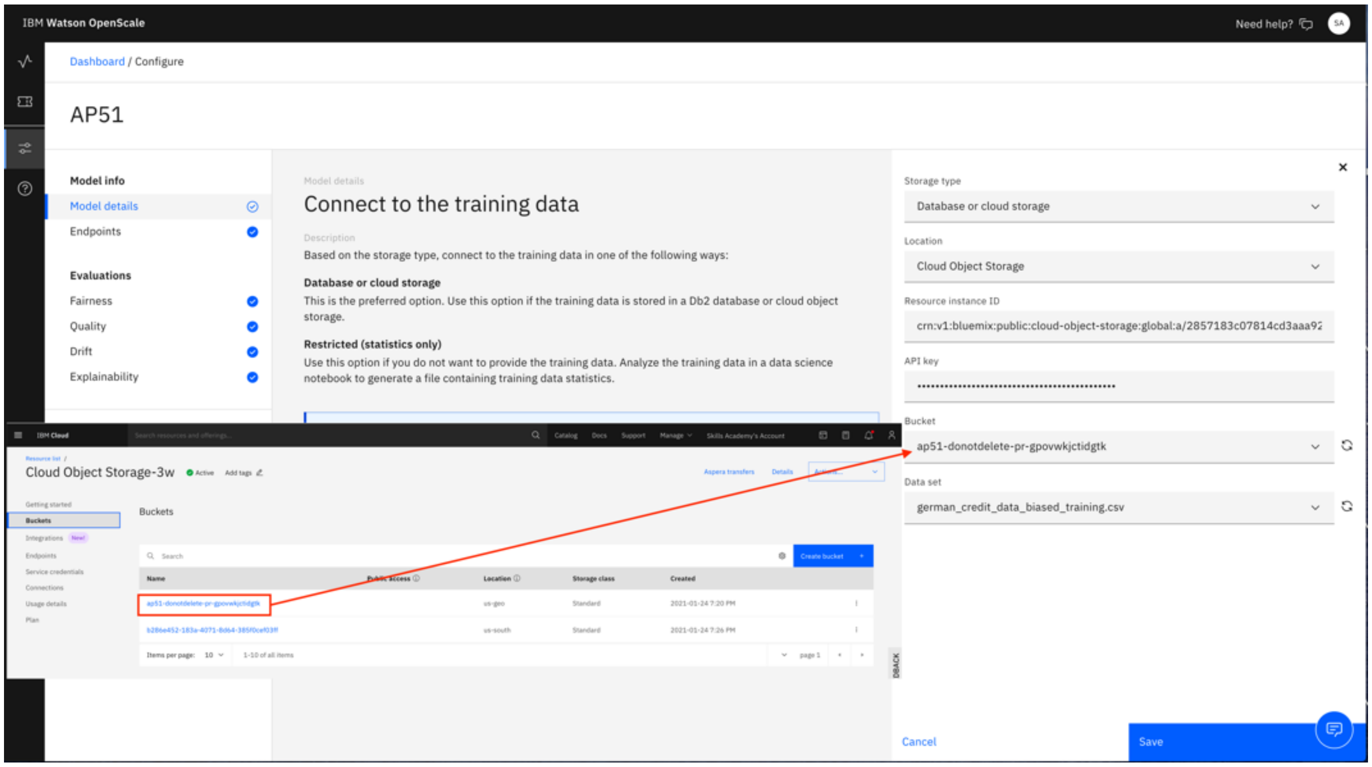
Getting started
Buckets
Integrations New!
Endpoints
Service credentials
Connections
Usage details
Plan

Service credentials
You can generate a new set of credentials for cases where you want to manually connect an app or external consumer to an IBM Cloud service. Learn more

Key name	Date created
viewer-bf3f8c69-b225-4d35-af50-f1f129c9a715	2021-01-24 7:26 PM
editor-bf3f8c69-b225-4d35-af50-f1f129c9a715	2021-01-24 7:26 PM
WDP-Project-Management-2dfb55b3-372b-499e-b9d5-4006d9ee58b2	2021-01-24 7:19 PM
WDP-Viewer-ap51-donotdelete-pr-gpovwkjctidgtk	2021-01-24 7:19 PM
WDP-Editor-ap51-donotdelete-pr-gpovwkjctidgtk	2021-01-24 7:19 PM
admin-bf3f8c69-b225-4d35-af50-f1f129c9a715	2021-01-24 7:26 PM

```
{
  "apikey": "[illegible]",
  "cos_hmac_keys": {
    "access_key_id": "[illegible]",
    "secret_access_key": "[illegible]"
  },
  "endpoints": "https://control.cloud-object-storage.cloud.ibm.com/v2/endpoints",
  "iam_apikey_description": "Auto-generated for key [illegible]",
  "iam_apikey_name": "admin-bf3f8c69-b225-4d35-af50-f1f129c9a715",
  "iam_role_crn": "crn:v1:bluemix:public:iam::::serviceRole:Writer",
  "iam_serviceid_crn": "[illegible]",
  "resource_instance_id": "crn:v1:bluemix:public:cloud-object-storage:global:a/2857183c07814cd3aaa92d1ef886ce09:2dfb55b3-372b-499e-b9d5-4006d9ee58b2::"
}
```

9. Click **Next** and you are now ready to include the Bucket name and the CSV file.
10. The Bucket name is also retrieved from the Cloud Object Storage service. Copy the first entry with the model name followed by `donotdelete`.
11. From the Data set drop down, scroll all the way down and you select the previously downloaded `german_credit_data_biased_training.csv`.
12. Click Save.


IBM Watson OpenScale — Need help?

Dashboard / Configure

AP51

Model info
Model details
Endpoints

Evaluations
Fairness
Quality
Drift
Explainability

Model details
Connect to the training data

Description
Based on the storage type, connect to the training data in one of the following ways:

Database or cloud storage
This is the preferred option. Use this option if the training data is stored in a Db2 database or cloud object storage.

Restricted (statistics only)
Use this option if you do not want to provide the training data. Analyze the training data in a data science notebook to generate a file containing training data statistics.

Storage type: Database or cloud storage
Location: Cloud Object Storage
Resource instance ID: crn:v1:bluemix:public:cloud-object-storage:global:a/2857183c07814cd3aaa92
API key: ••
Bucket: ap51-donotdelete-pr-gpovwkjctidgtk
Data set: german_credit_data_biased_training.csv

Cancel Save

IBM Cloud — Catalog, Docs, Support, Manage, Skills Academy's Account

Resource list /
Cloud Object Storage-3w ● Active Add tags
Aspera transfers Details Actions...

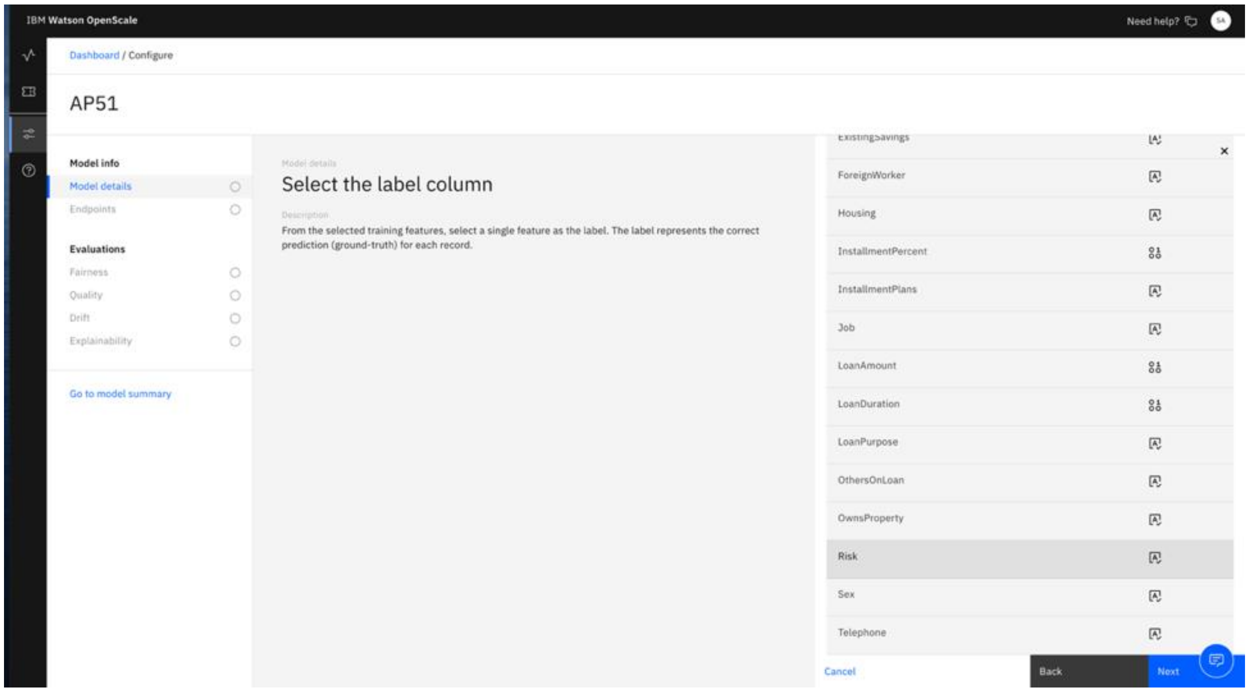
Getting started
Buckets
Integrations New!
Endpoints
Service credentials
Connections
Usage details
Plan

Buckets

Name	Public access	Location	Storage class	Created
ap51-donotdelete-pr-gpovwkjctidgtk		us-geo	Standard	2021-01-24 7:20 PM
b286e452-183a-4071-8d64-385f0cef03ff		us-south	Standard	2021-01-24 7:26 PM

Items per page: 10 1-10 of all items page 1

13. You are now prompted to select the prediction column, the “ground truth” and that happens to be the **Risk** feature. Highlight that feature or column and click **Next**.



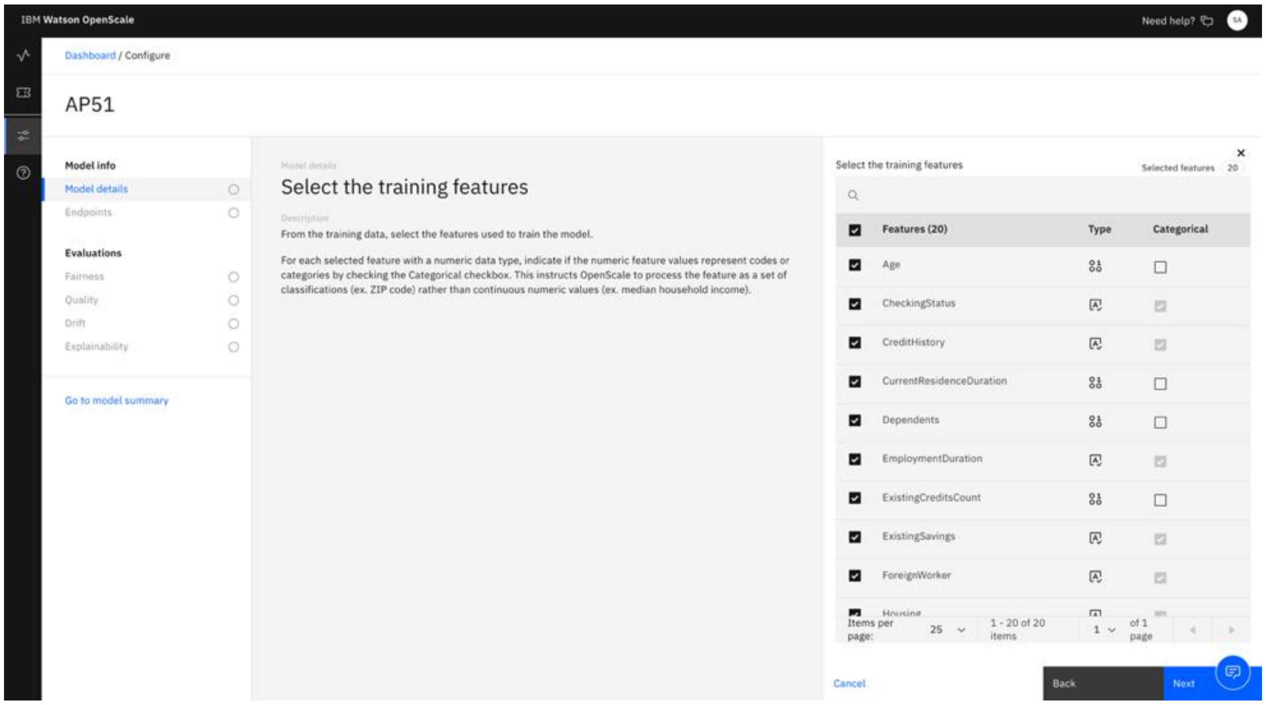
AP51

Select the label column

From the selected training features, select a single feature as the label. The label represents the correct prediction (ground-truth) for each record.

ExistingSavings
ForeignWorker
Housing
InstallmentPercent
InstallmentPlans
Job
LoanAmount
LoanDuration
LoanPurpose
OthersOnLoan
OwnsProperty
Risk
Sex
Telephone

14. Select all 20 features and click **Next**.



AP51

Select the training features

From the training data, select the features used to train the model.

For each selected feature with a numeric data type, indicate if the numeric feature values represent codes or categories by checking the Categorical checkbox. This instructs OpenScale to process the feature as a set of classifications (ex. ZIP code) rather than continuous numeric values (ex. median household income).

Select the training features   Selected features 20

✓	Features (20)	Type	Categorical
✓	Age	01	☐
✓	CheckingStatus	A	✓
✓	CreditHistory	A	✓
✓	CurrentResidenceDuration	01	☐
✓	Dependents	01	☐
✓	EmploymentDuration	A	✓
✓	ExistingCreditsCount	01	☐
✓	ExistingSavings	A	✓
✓	ForeignWorker	A	✓
✓	Housing	A	✓

Items per page: 25   1 - 20 of 20 items   1   of 1 page

Now that you have completed configuring model details, you are ready to begin analyzing and monitoring results. Start with the Drift feature; but first, some verbiage below as to what the Drift measure.

Configure Drift

Watson OpenScale detects both drift in accuracy and drift in data.

- Drop in accuracy

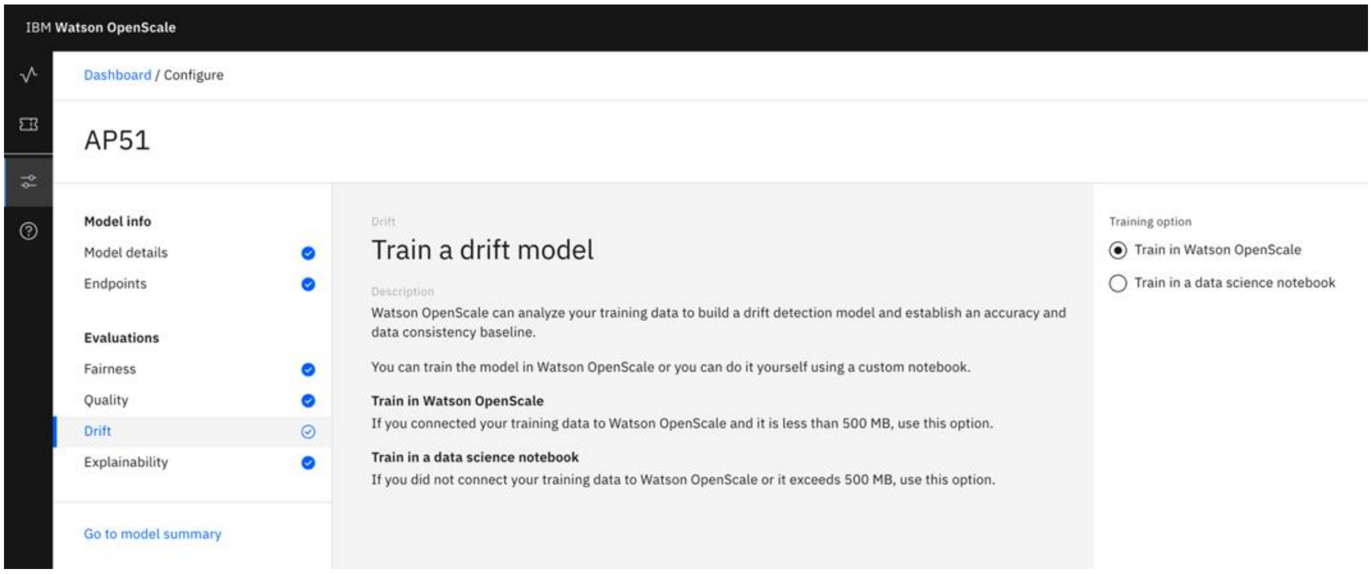
  Estimates the drop in accuracy of the model at runtime. Model accuracy drops if there is an increase in transactions that are similar to those that the model did not evaluate correctly in the training data. This type of drift is calculated for structured binary and multi-class classification models only.

- Drop in data consistency

  Estimates the drop in consistency of the data at runtime as compared to the characteristics of the data at training time.

A drop in either model accuracy or data consistency can lead to a negative impact on the business outcomes that are associated with the model and must be addressed by retraining the model. Now, let's begin with the exercises.

1. Click the **Drift** link in the left panel.



Train a drift model

Description

Watson OpenScale can analyze your training data to build a drift detection model and establish an accuracy and data consistency baseline.

You can train the model in Watson OpenScale or you can do it yourself using a custom notebook.

Train in Watson OpenScale
If you connected your training data to Watson OpenScale and it is less than 500 MB, use this option.

Train in a data science notebook
If you did not connect your training data to Watson OpenScale or it exceeds 500 MB, use this option.

Training option: [✓ Train in Watson OpenScale] Train in a data science notebook

2. Select **Train in Watson Openscale**.
3. Set Drift threshold to **10%.**

Every monitor in Watson OpenScale has a threshold and it is going to tell you if the value of your test is below that threshold.

4. Set Sample size to **100** and click **Save.**

This is the minimum number of transactions that the monitor will use to give you a reading. Eventually what you want to do is to upload feedback data to see how your model is performing. For classroom settings 100 records is ample; for commercial use, you may upload 1000 records and observe the readings.

The monitors run every hour. If your model has had 100 predictions sent to it in the last hour will evaluate over those 100 new records and give you a score for that. Say in a subsequent upload you introduce 10 new records, then it will use 90 from the previous reading, plus the newly added 10 and render the score for that. In another words, every time you see a drift score it is going to use at least 100 predictions to calculate the new score.

This may take upwards of 5 minutes.

The Drift model is a binary classification and is looking at the training data and saying: “what does the model struggle to predict accurately.” The approach with drift addresses the issue that collecting feedback data to further test the efficacy of the model can be time consuming and expensive. What if we could gain insights into how the model may drift from the original threshold? Especially if you are doing credit applications and that may require data on folks who have already paid of their loan. You may not have a regular flow of that information. With the drift model you can access the deviations in real-time without the need of additional feedback data.

Configure Quality

Use quality monitoring to determine how well your model predicts outcomes. When quality monitoring is enabled, it generates a set of metrics every hour by default. You can generate these metrics on demand by clicking the **Check quality now** button or by using the Python client.

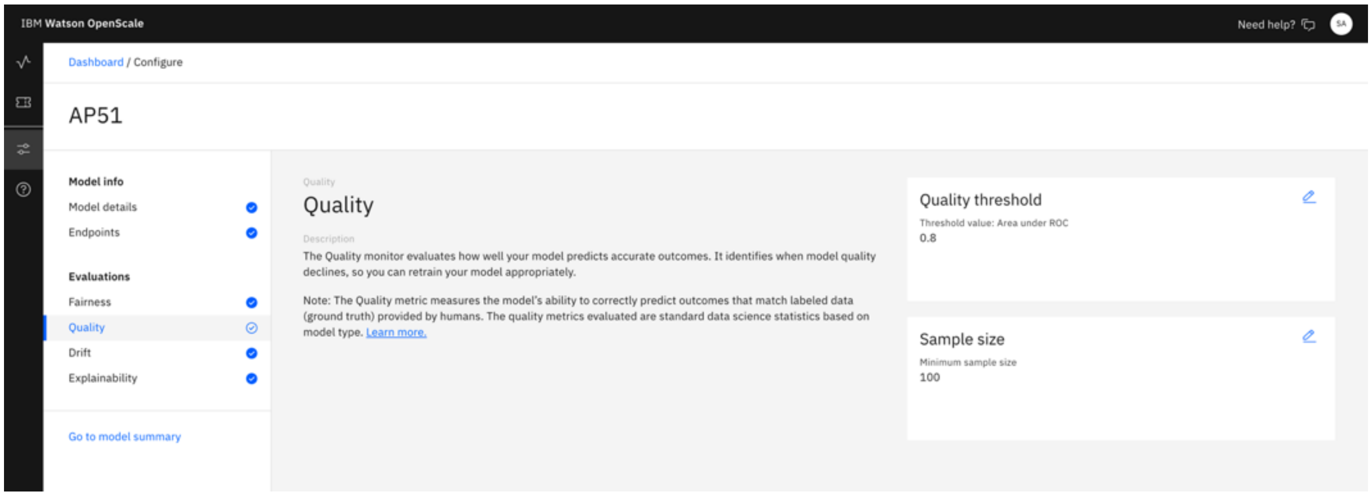
Quality metrics are calculated based on the following information:

- manually labeled feedback data,
- monitored deployment responses for these data.

For proper monitoring, feedback data must be logged to Watson OpenScale on a regular basis. The feedback data can be provided either by using “Add Feedback data” option or using Python client or REST API. For more information, please see the Appendix.

You are now ready to configure Quality settings.

1. Accept the default of **0.8** as Quality (accuracy) threshold
2. Specify **100** for sample size.



AP51

Quality

The Quality monitor evaluates how well your model predicts accurate outcomes. It identifies when model quality declines, so you can retrain your model appropriately.

Note: The Quality metric measures the model’s ability to correctly predict outcomes that match labeled data (ground truth) provided by humans. The quality metrics evaluated are standard data science statistics based on model type. Learn more.

Quality threshold
Threshold value: Area under ROC
0.8

Sample size
Minimum sample size
100

Configure Fairness

Watson OpenScale automatically identifies whether any known protected attributes are present in a model. When Watson OpenScale detects these attributes, it automatically recommends configuring bias monitors for each attribute present, to ensure that bias against these potentially sensitive attributes is tracked in production.
Currently, Watson OpenScale detects and recommends monitors for the following protected attributes:

- sex
- ethnicity
- marital status
- age
- zip code or postal code

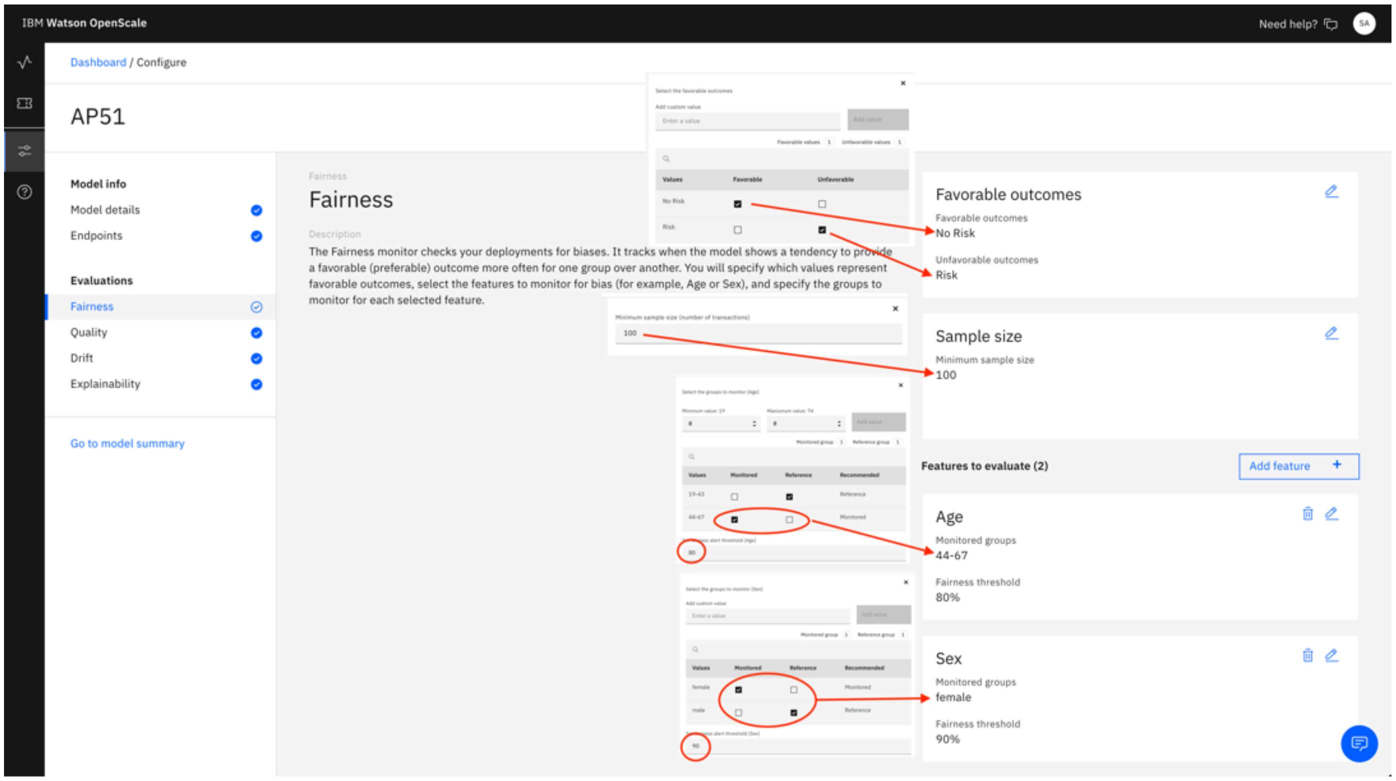
In addition to detecting protected attributes, Watson OpenScale recommends which values within each attribute should be set as the monitored and the reference values. For example, Watson OpenScale recommends that within the `Sex` attribute, the bias monitor be configured such that `Female` and `Non-Binary` are the monitored values, and `Male` is the reference value. If you want to change any of the recommendations, you can edit them via the bias configuration panel.
Recommended bias monitors help to speed up configuration and ensure that you are checking your AI models for fairness against sensitive attributes. As regulators begin to turn a sharper eye on algorithmic bias, it is becoming more critical that organizations have a clear understanding of how their models are performing, and whether they are producing unfair outcomes for certain groups.

1. Let's begin by configuring the Fairness parameters.
    a. In the first configuration box, specify **No Risk** for Favorable outcomes and **Risk** for Unfavorable outcomes.
    The calculation here is called "disparate impact" meaning that we are comparing two things that are essentially different in kind, not allowing comparison. Here we calculate how many good results did males get (favorable) versus how many good results did the females get. Favorable means that we should accept this application for loan, it would not be a risky event.
    b. Set Sample size to **100** (from default of 1000)
    c. OpenScale has selected both **sex** and **age** as prediction columns, accept the default and click Save. Note that with the light account you can select a maximum of two attributes to predict.
    At this stage, Watson Openscale has run the model over the original data, and it had depicted these two features as most likely to have an impact on the fairness of the model.
    d. Select age range of 44-67 as the Monitored group and keep the threshold at **80%**.
    Monitored group is the group that you are worried about having bias against. The reference group is everybody else. For example, the reference group could be the white males.

Select **Female** as the Monitored group and keep the threshold at **80%** (even though it appears as 90 in the image below). Later in this exercise you will change it to 90 % and observe a built-in bias.

When you configure Drift in OpenScale, you have to specify the tolerable accuracy drift magnitude. The drift is measured as the drop in accuracy as compared to the model accuracy at training time. For example, if the model accuracy at training time was 90% and at runtime the estimated accuracy of the model is 80%, then the model is said to have drifted by 10%. Depending on the use case, model owners will be willing to tolerate different amounts of drift. Hence IBM Watson OpenScale allows the user to specify the accuracy drift magnitude (called as Drift alert threshold) for each model being monitored in OpenScale. If the drift for a model drops below the specified threshold, then OpenScale will generate an alert for the user.

2. Save your settings.



AP51

Fairness

Description

The Fairness monitor checks your deployments for biases. It tracks when the model shows a tendency to provide a favorable (preferable) outcome more often for one group over another. You will specify which values represent favorable outcomes, select the features to monitor for bias (for example, Age or Sex), and specify the groups to monitor for each selected feature.

Select the favorable outcomes

Add custom value: Enter a value

Favorable values 1 Unfavorable values 1

Values	Favorable	Unfavorable
No Risk	☑	☐
Risk	☐	☑

Minimum sample size (number of transactions): 100

Select the groups to monitor (Age)

Minimum value: 19 Maximum value: 74

Monitored group 1 Reference group 1

Values	Monitored	Reference	Recommended
19-43	☐	☑	Reference
44-67	☑	☐	Monitored

Set fairness alert threshold (Age): 80

Select the groups to monitor (Sex)

Add custom value: Enter a value

Monitored group 1 Reference group 1

Values	Monitored	Reference	Recommended
female	☑	☐	Monitored
male	☐	☑	Reference

Set fairness alert threshold (Sex): 90

Favorable outcomes

Favorable outcomes: No Risk

Unfavorable outcomes: Risk

Sample size

Minimum sample size: 100

Features to evaluate (2)

Age

Monitored groups: 44-67

Fairness threshold: 80%

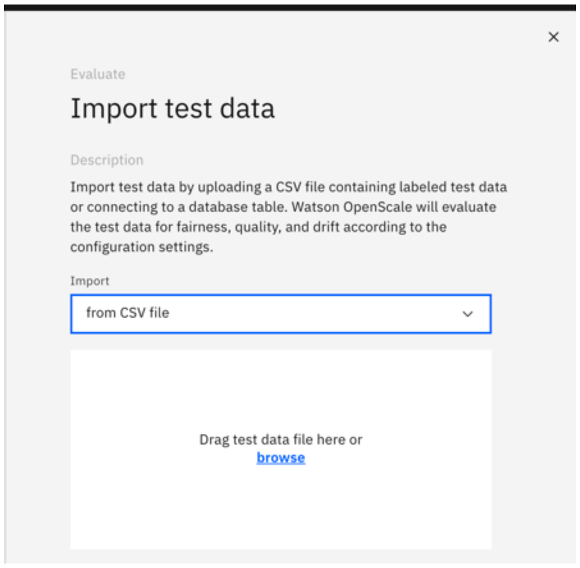
Sex

Monitored groups: female

Fairness threshold: 90%

IBM Watson OpenScale identifies data drift by analyzing the training data and extracting some characteristics of the data. It then compares the data received by the model at periodic intervals with the training data characteristics to identify data drift.

3. Click **Go to model summary**.
4. From the **Actions** drop-down menu, click **Evaluate now**.



Evaluate

Import test data

Description

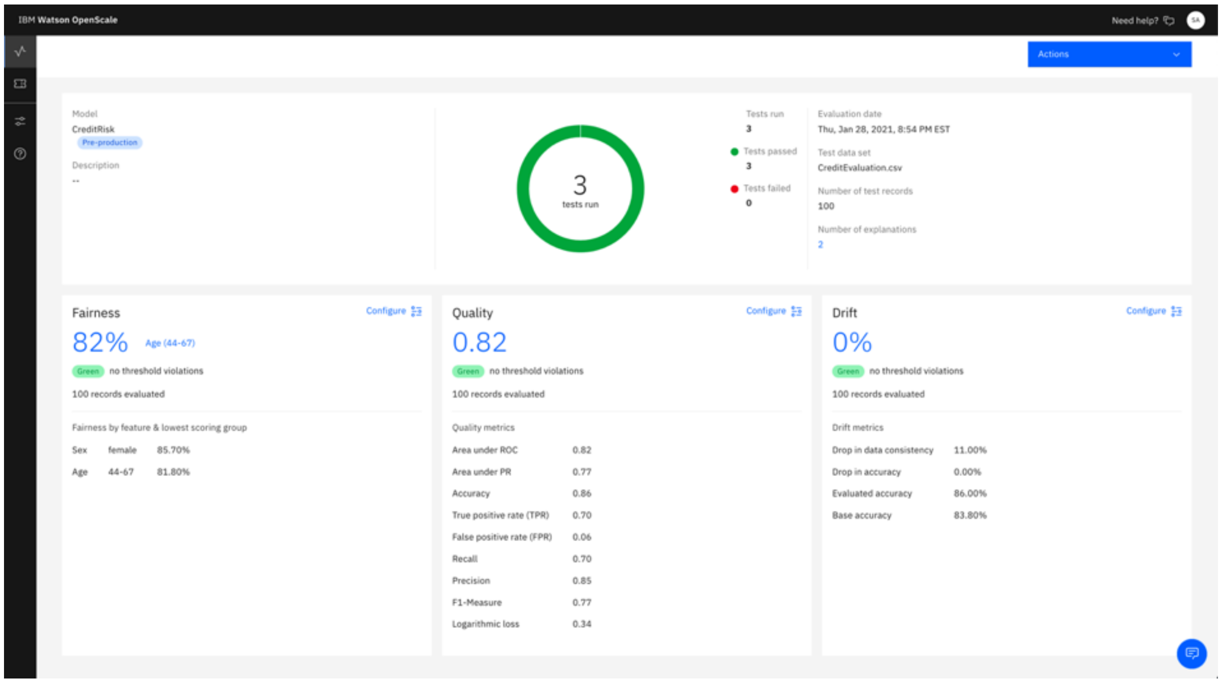
Import test data by uploading a CSV file containing labeled test data or connecting to a database table. Watson OpenScale will evaluate the test data for fairness, quality, and drift according to the configuration settings.

Import

from CSV file

Drag test data file here or browse

5. Point your browser to this file: CreditEvaluation.csv and download it to your local drive.
6. Browse and upload the file and click **Upload and evaluate**. This may take upwards of 7 minutes until the analysis appears per below:



Model
CreditRisk
Pre-production

Description
--

3 tests run

Tests run: 3
Tests passed: 3
Tests failed: 0

Evaluation date: Thu, Jan 28, 2021, 8:54 PM EST
Test data set: CreditEvaluation.csv
Number of test records: 100
Number of explanations: 2

Fairness
82% Age (44-67)
Green no threshold violations
100 records evaluated

Fairness by feature & lowest scoring group

Sex	female	85.70%
Age	44-67	81.80%

Quality
0.82
Green no threshold violations
100 records evaluated

Quality metrics

Area under ROC	0.82
Area under PR	0.77
Accuracy	0.86
True positive rate (TPR)	0.70
False positive rate (FPR)	0.06
Recall	0.70
Precision	0.85
F1-Measure	0.77
Logarithmic loss	0.34

Drift
0%
Green no threshold violations
100 records evaluated

Drift metrics

Drop in data consistency	11.00%
Drop in accuracy	0.00%
Evaluated accuracy	86.00%
Base accuracy	83.80%

Here, **Quality** is using manually scored data. The data that we uploaded using the Risk/No Risk column, it was able to get quality (accuracy) metrics from that data. Since our threshold was set at 80%, then 82% is acceptable and it passed the test, hence the green color.

For the **Fairness** score, it appears (in this example) that the fairness score for both male and female was above threshold of 80%, hence another green indication, meaning that there was no bias as the model fairness.

The Drift model says if our predictions were going to drift by more than 10% as a result of the new (100 rows of data) upload, hence, an alert would've indicated a failed test.

If you were to upload a larger chuck of data, 1000 rows for example, you may then see a 5% drift in data, for example.

In order to identify the accuracy drift, OpenScale needs to understand the behavior of the customers' model on the training and test data. It analyses the customer model behavior and builds its own model (called Drift Detection Model) which predicts if the customers' model is going to generate an accurate prediction for a given data point. Watson OpenScale runs the Drift detection model on the payload data (data received by the model at runtime). Therefore, we now know the number of records in the payload for which the customers' model is likely to have made an error in prediction. We make use of this information to generate the predicted accuracy of the customer's model on the payload data. This value is compared with the accuracy of the model at training time to identify the model accuracy drift.

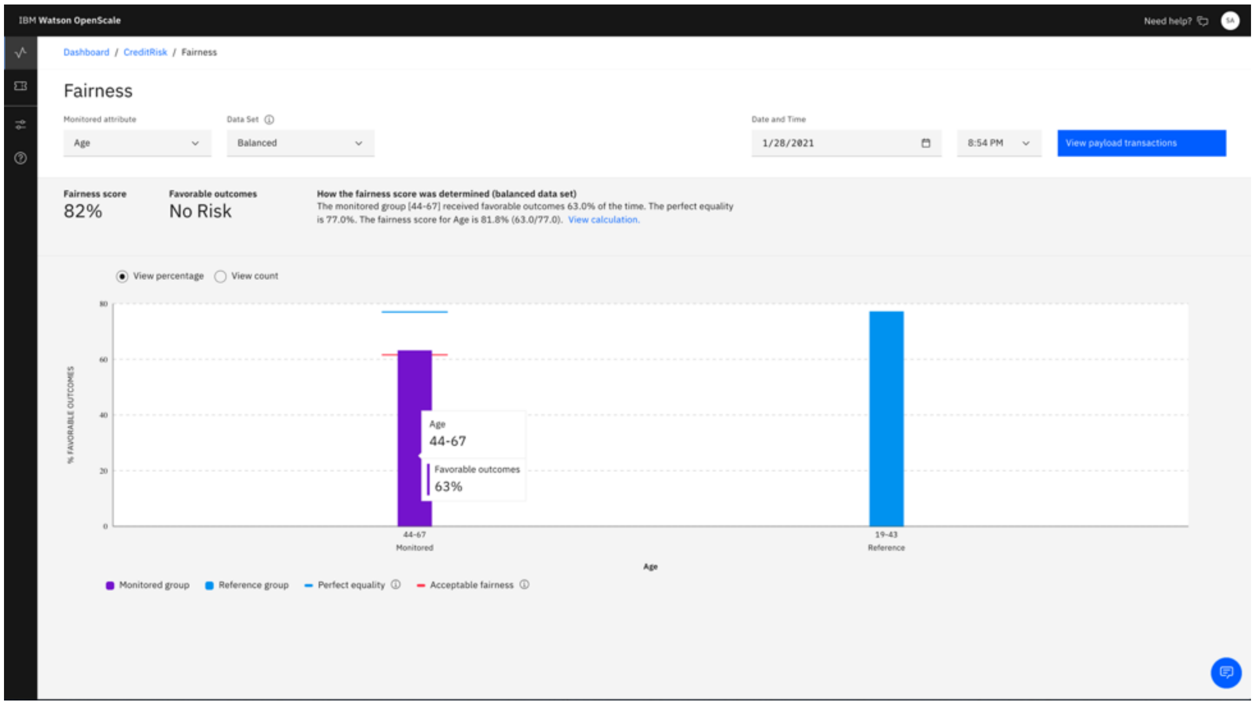
Section 1. Understanding the results

Fairness

Use IBM Watson OpenScale fairness monitoring to determine whether outcomes that are produced by your model are fair or not for monitored group. When fairness monitoring is enabled, it generates a set of metrics every hour by default. You can generate these metrics on demand by clicking the **Check fairness now** button or by using the Python client.

Complete the following steps to gain further insights from Fairness score.

1. Click the value of your Fairness score (in this example, click 82%)



IBM Watson OpenScale

Dashboard / CreditRisk / Fairness

Fairness

Monitored attribute: Age

Data Set: Balanced

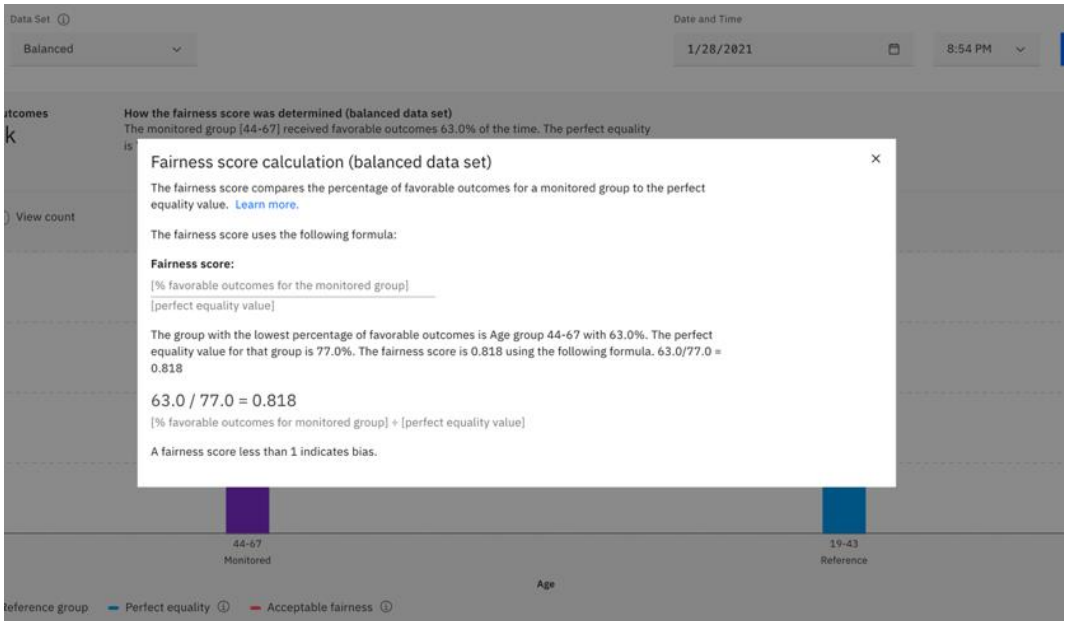
Date and Time: 1/28/2021 8:54 PM

Fairness score: 82%

Favorable outcomes: No Risk

How the fairness score was determined (balanced data set)
The monitored group [44-67] received favorable outcomes 63.0% of the time. The perfect equality is 77.0%. The fairness score for Age is 81.8% (63.0/77.0). View calculation.

Once you hover over the Monitored bar, this is telling you that they got good outcome 63% of the time and if you hover over the Reference group, it will indicate something similar to a good outcome 77% of the time. Therefore, the fairness score is 63/77 = 82%


Data Set

Balanced

Date and Time

1/28/2021

8:54 PM

How the fairness score was determined (balanced data set)

The monitored group [44-67] received favorable outcomes 63.0% of the time. The perfect equality

Fairness score calculation (balanced data set)

The fairness score compares the percentage of favorable outcomes for a monitored group to the perfect equality value. Learn more.

The fairness score uses the following formula:

Fairness score:

$$\frac{\text{[\% favorable outcomes for the monitored group]}}{\text{[perfect equality value]}}$$

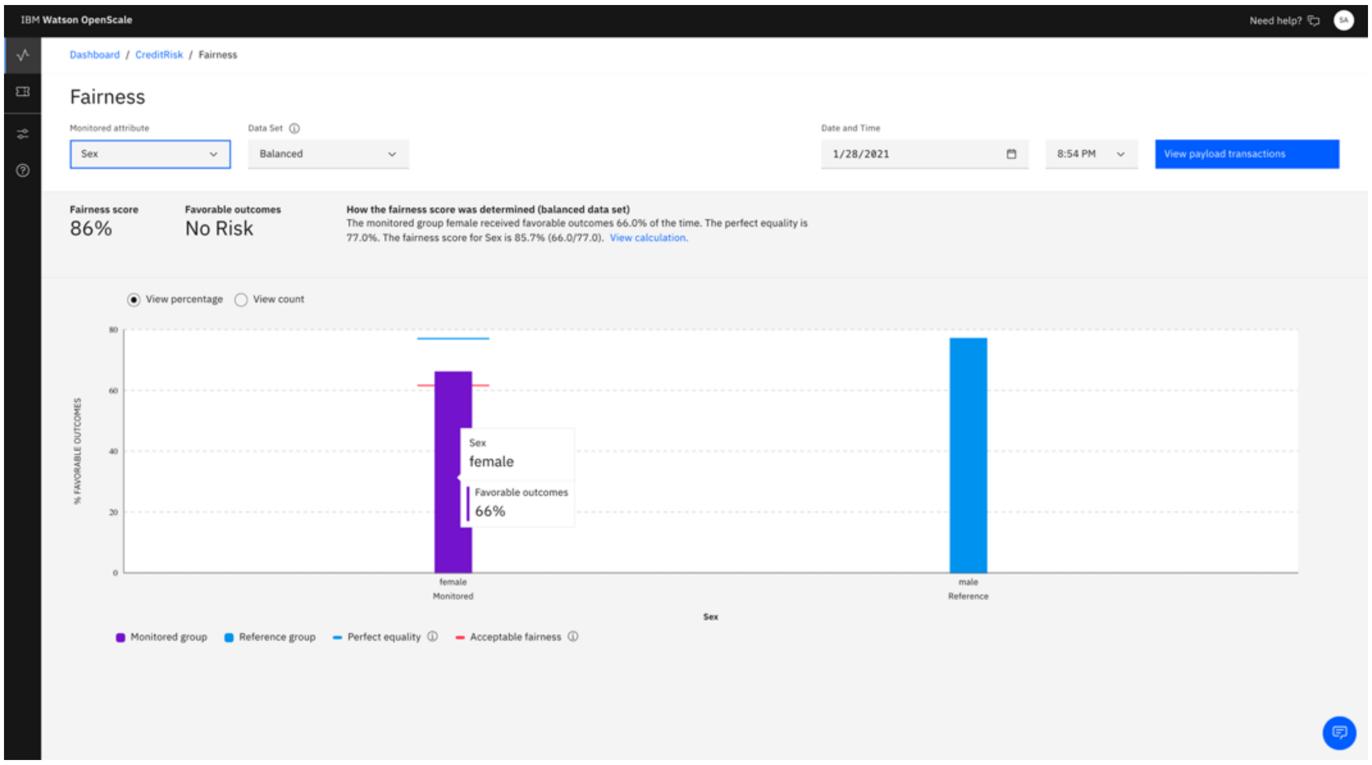
The group with the lowest percentage of favorable outcomes is Age group 44-67 with 63.0%. The perfect equality value for that group is 77.0%. The fairness score is 0.818 using the following formula. 63.0/77.0 = 0.818

63.0 / 77.0 = 0.818

[% favorable outcomes for monitored group] ÷ [perfect equality value]

A fairness score less than 1 indicates bias.

2. Change the Monitored attributes to **Sex** and observe the metrics.


Fairness

Monitored attribute: Sex

Data Set: Balanced

Date and Time: 1/28/2021 8:54 PM

Fairness score	Favorable outcomes
86%	No Risk

How the fairness score was determined (balanced data set)

The monitored group female received favorable outcomes 66.0% of the time. The perfect equality is 77.0%. The fairness score for Sex is 85.7% (66.0/77.0). View calculation.

In the next section, you will change the threshold to 90 and further explanations of the results. At this stage, explore the UI by yourself and click the "i" wherever it appears for self-paced explanations. Later in this document, when you change the threshold for fairness, we will explore in greater detail the Drift Detection Model (a secondary model) that detected bias in the data and the contributing attributes.

Quality

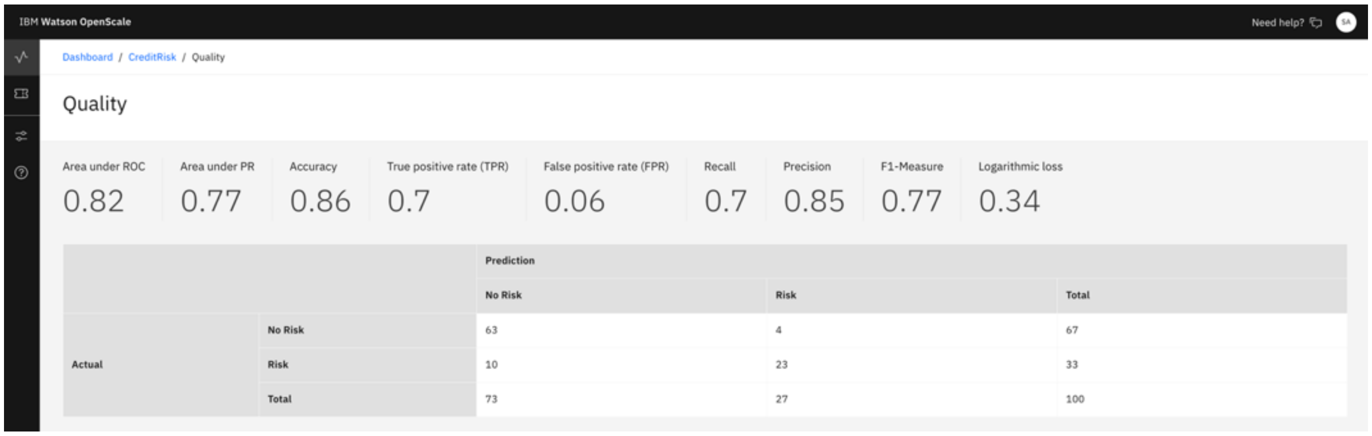
Use quality monitoring to determine how well your model predicts outcomes. When quality monitoring is enabled, it generates a set of metrics every hour by default. You can generate these metrics on demand by clicking the **Check quality now** button or by using the Python client.

Quality metrics are calculated based on the following information:

- manually labeled feedback data,
- monitored deployment responses for these data.

For proper monitoring, feedback data must be logged to Watson OpenScale on a regular basis. The feedback data can be provided either by using "Add Feedback data" option or using Python client or REST API.

1. From the Dashboard, inside the Quality model panel, click the percentage value, in this example: 0.82



Quality

Area under ROC	Area under PR	Accuracy	True positive rate (TPR)	False positive rate (FPR)	Recall	Precision	F1-Measure	Logarithmic loss
0.82	0.77	0.86	0.7	0.06	0.7	0.85	0.77	0.34

		Prediction		
		No Risk	Risk	Total
Actual	No Risk	63	4	67
	Risk	10	23	33
	Total	73	27	100

2. As a class exercise, fill in the table below in the confusion metrics below and provide a verbal explanation as to what the confusion matrix is revealing.

No Risk is *positive class*; **Risk** is *negative* class.

A **true positive** is an outcome where the model *correctly* predicts the positive class. Similarly, a **true negative** is an outcome where the model *correctly* predicts the negative class

A **false positive** is an outcome where the model *incorrectly* predicts the positive class. And a **false negative** is an outcome where the model *incorrectly* predicts the negative class

	No Risk	**Risk**
No Risk	True Positive =	False Positive =
Risk	False Negative =	True Negative =

For machine learning engines other than Watson OpenScale, such as Microsoft Azure ML Studio, Microsoft Azure ML Service, or Amazon Sagemaker ML quality monitoring creates additional scoring requests on the monitored deployment.

Drift

Watson OpenScale detects both drift in accuracy and drift in data.

- Drop in accuracy

  Estimates the drop in accuracy of the model at runtime. Model accuracy drops if there is an increase in transactions that are similar to those that the model did not evaluate correctly in the training data. This type of drift is calculated for structured binary and multi-class classification models only.

- Drop in data consistency

  Estimates the drop in consistency of the data at runtime as compared to the characteristics of the data at training time.

A drop in either model accuracy or data consistency led to a negative impact on the business outcomes that are associated with the model and must be addressed by retraining the model.
Depending on whether you perform drift detection as part of batch processing or not, the drift displays are different.

1.1.1. Drift visualization for non-batch processing data

The drift visualization includes both graphical and numeric statistical data. By clicking the chart, you can display specific transactions that contribute to drift. The top reasons for detected drift display and includes a natural-language description of the observation as well as a list of unexpected values.
Specifically, from the **Select a transaction set from the chart or list below** section, you can choose the following views:

- Transactions responsible for drop in accuracy
- Transactions responsible for drop in accuracy and data consistency
- Transactions responsible for drop in data consistency
- Drift transactions are available in the transaction details screen, where you can click **Explain** to understand how a specific transaction has made it into the drift category (more on Explanations later in this document).

1.1.2. Drift analysis for batch processing data

For the large quantity of data that can be produced by the batch processor, you receive a count of records that contribute to a drop in accuracy, a drop in data consistency, and both. In addition to this summary display, you can run a specialized analysis notebook: Notebook for analyzing payload transactions causing drift.

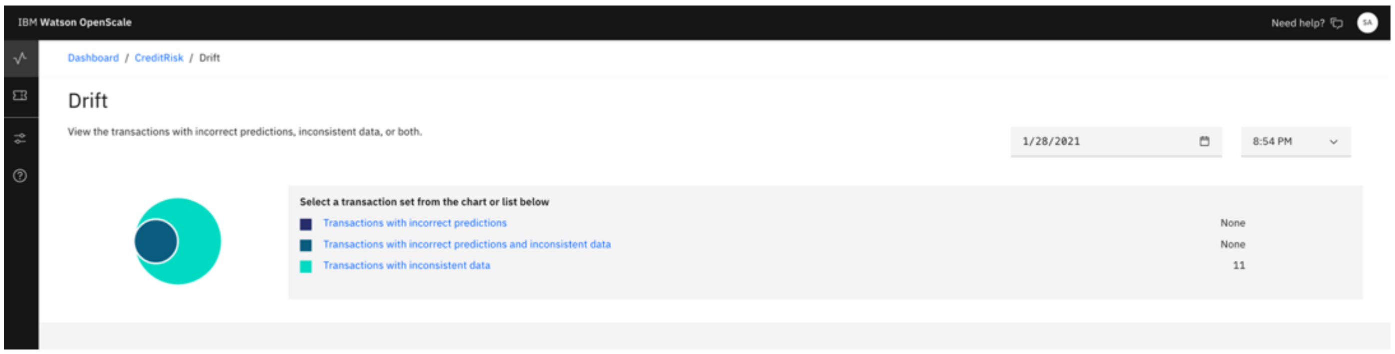
1.1.3. Limitations

The following limitations apply to the drift monitor:

- Drift is supported for structured data only.
- Although classification models support both data and accuracy drift, regression models support only data drift.

- Drift is not supported for Python functions.

1. From the Drift panel in the Dashboard click the Drift percentage, in this case, 0%.



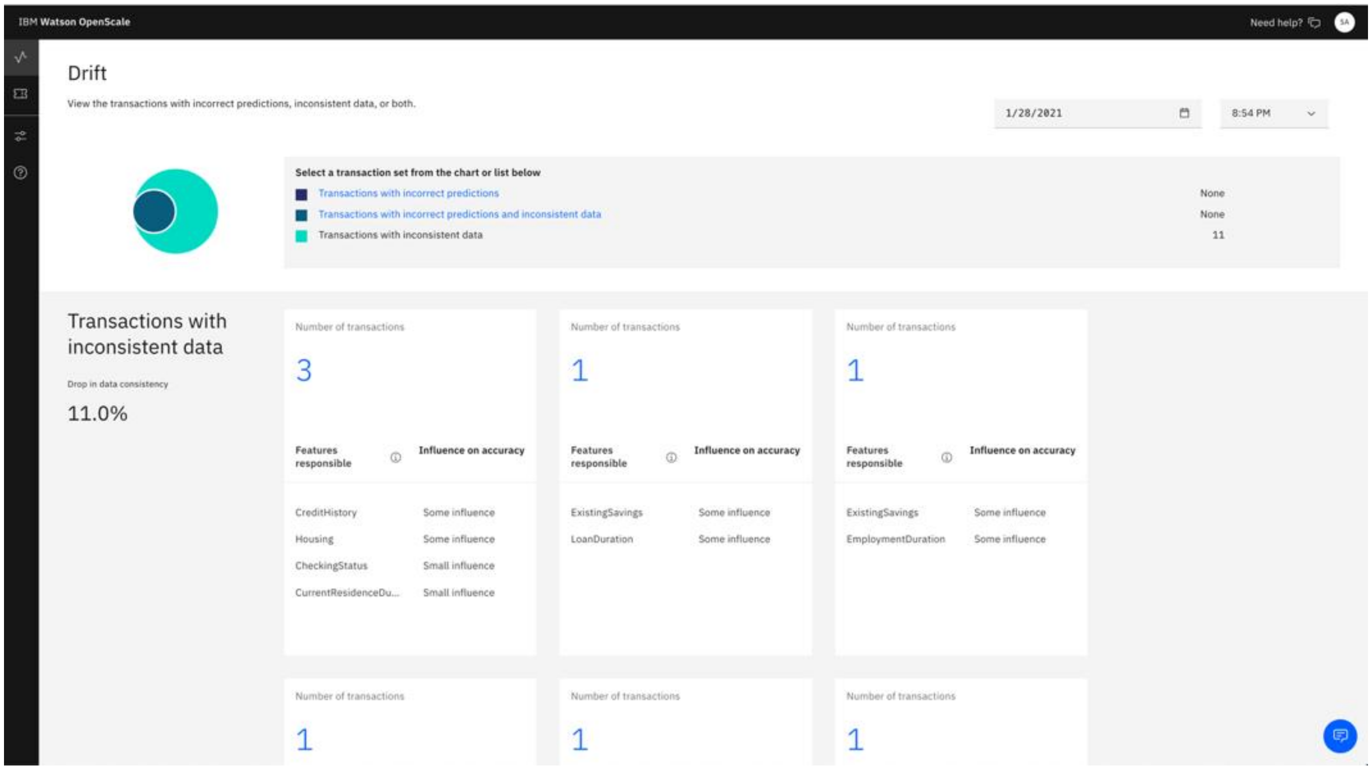
Drift

View the transactions with incorrect predictions, inconsistent data, or both.

Select a transaction set from the chart or list below

■	Transactions with incorrect predictions	None
■	Transactions with incorrect predictions and inconsistent data	None
■	Transactions with inconsistent data	11

The Drift Detection capability in IBM Watson OpenScale monitors the data received by the model in production and (a) estimates the accuracy of the model (***accuracy drift***) and (b) checks if the data is very different than the models' training data (***data drift***). Accuracy Drift is very helpful to enterprises as it helps them to immediately react to a drop in model accuracy before it has any significant impact on the business outcomes. For example, in case of the loan processing model it will ensure that the business does not end up giving loans to a wrong set of customers thereby avoiding bad loans. Data drift on the other hand helps enterprises understand the change in data characteristics at runtime which might point to a change in the business environment. For example, in case of the loan processing model the training data might have had very few applicants with age less than 25. Due to a marketing campaign run to attract younger customers, a lot of young people might be applying for a loan. This will trigger a data drift alert which will point to the fact that the model might not be trained to handle such kind of customers. Hence the business would want to retrain the model to ensure that it makes the right decision for people with age less than 25.



Drift

View the transactions with incorrect predictions, inconsistent data, or both.

1/28/2021

8:54 PM

Select a transaction set from the chart or list below

■ Transactions with incorrect predictions	None
■ Transactions with incorrect predictions and inconsistent data	None
■ Transactions with inconsistent data	11

Transactions with inconsistent data

Drop in data consistency

11.0%

Number of transactions

3

Features responsible	Influence on accuracy
CreditHistory	Some influence
Housing	Some influence
CheckingStatus	Small influence
CurrentResidenceDu...	Small influence

Number of transactions

1

Features responsible	Influence on accuracy
ExistingSavings	Some influence
LoanDuration	Some influence

Number of transactions

1

Features responsible	Influence on accuracy
ExistingSavings	Some influence
EmploymentDuration	Some influence

Number of transactions

1

Number of transactions

1

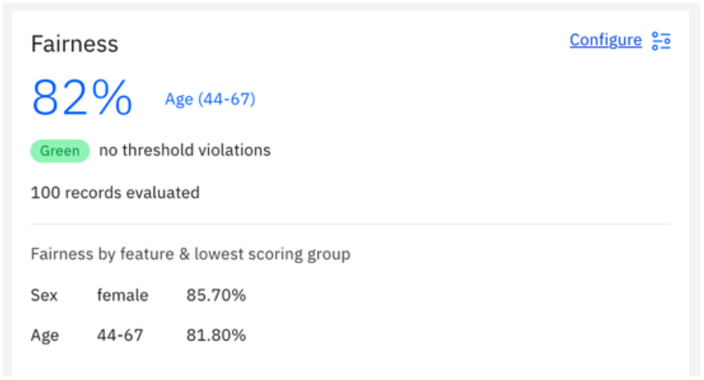
Number of transactions

1

Let’s introduce Bias

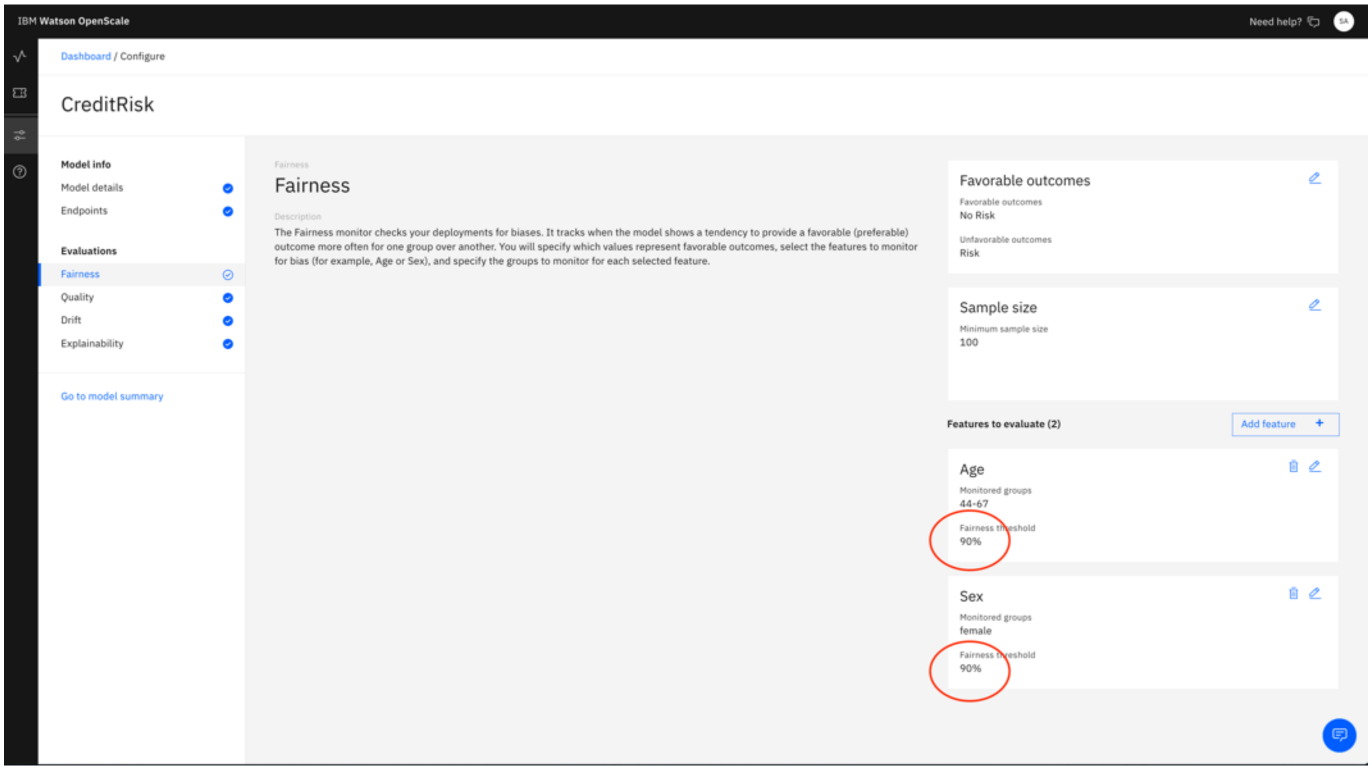
The idea here is to set the threshold well above 80%, for example 90% where your fairness model will display a red color signifying bias detected in your model.

1. Click **Configure**.



Sex	female	85.70%
Age	44-67	81.80%

2. Change the Fairness threshold for both Age and Sex to **90** percent.



Fairness

Description

The Fairness monitor checks your deployments for biases. It tracks when the model shows a tendency to provide a favorable (preferable) outcome more often for one group over another. You will specify which values represent favorable outcomes, select the features to monitor for bias (for example, Age or Sex), and specify the groups to monitor for each selected feature.

Favorable outcomes
Favorable outcomes: No Risk
Unfavorable outcomes: Risk

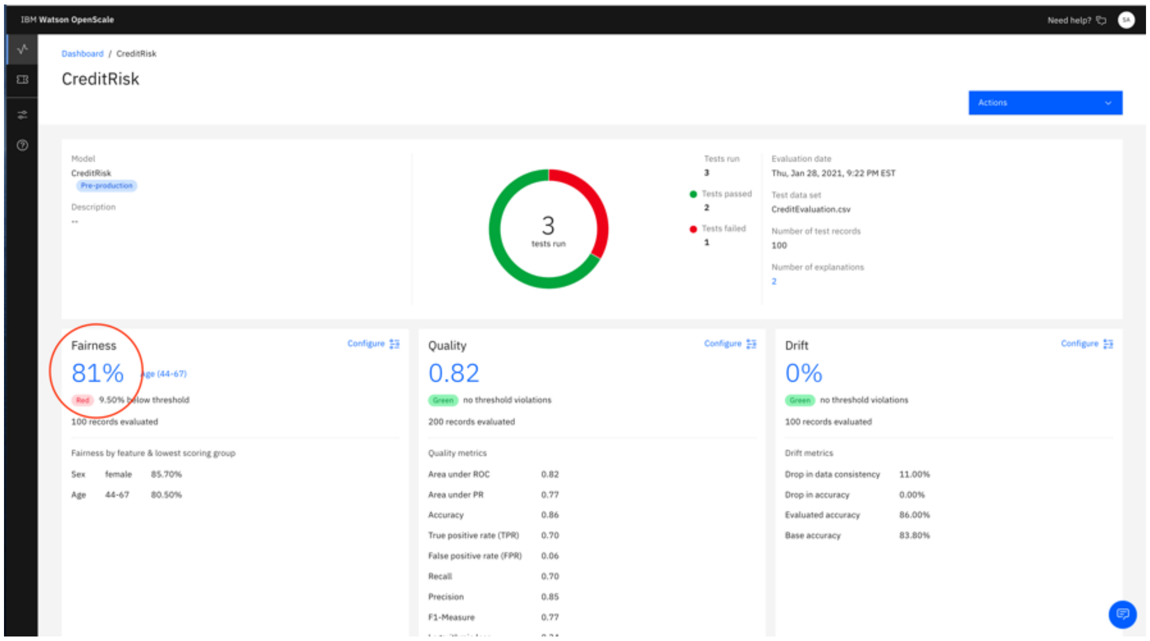
Sample size
Minimum sample size: 100

Features to evaluate (2)

Age
Monitored groups: 44-67
Fairness threshold: 90%

Sex
Monitored groups: female
Fairness threshold: 90%

3. Click **Go to model summary**.
4. From the **Actions** menu select **Evaluate now**.
5. Download the CreditEvaluation CSV file.
6. Browse and select the newly downloaded CSV file with 100 records.
7. Click **Actions** and reupload the 110-record CSV file and **Evaluate now**.



IBM Watson OpenScale

Dashboard / CreditRisk

CreditRisk

Model	CreditRisk (Pre-production)
Description	--
Tests run	3
Tests passed	2
Tests failed	1
Evaluation date	Thu, Jan 28, 2021, 9:22 PM EST
Test data set	CreditEvaluation.csv
Number of test records	100
Number of explanations	2

Fairness
81% Age (44-67)
Red 9.50% below threshold
100 records evaluated

Fairness by feature & lowest scoring group

Sex	female	85.70%
Age	44-67	80.50%

Quality
0.82
Green no threshold violations
200 records evaluated

Quality metrics

Area under ROC	0.82
Area under PR	0.77
Accuracy	0.86
True positive rate (TPR)	0.70
False positive rate (FPR)	0.06
Recall	0.70
Precision	0.85
F1-Measure	0.77

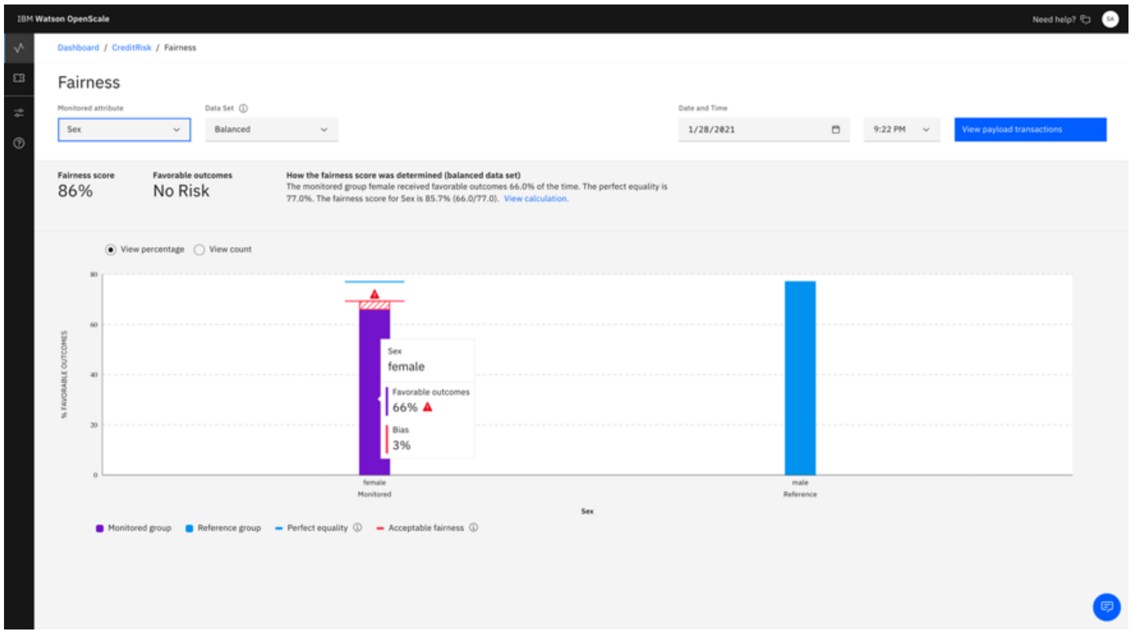
Drift
0%
Green no threshold violations
100 records evaluated

Drift metrics

Drop in data consistency	11.00%
Drop in accuracy	0.00%
Evaluated accuracy	86.00%
Base accuracy	83.80%

Notice that the Fairness level of 81% (your metrics might be slightly different) is less than the 90% threshold that you had set; hence, bias has been introduced to the observation.

1. Select **Sex** as the Monitored attribute.



IBM Watson OpenScale

Dashboard / CreditRisk / Fairness

Fairness

Monitored attribute: Sex
Data Set: Balanced
Date and Time: 1/28/2021 9:22 PM

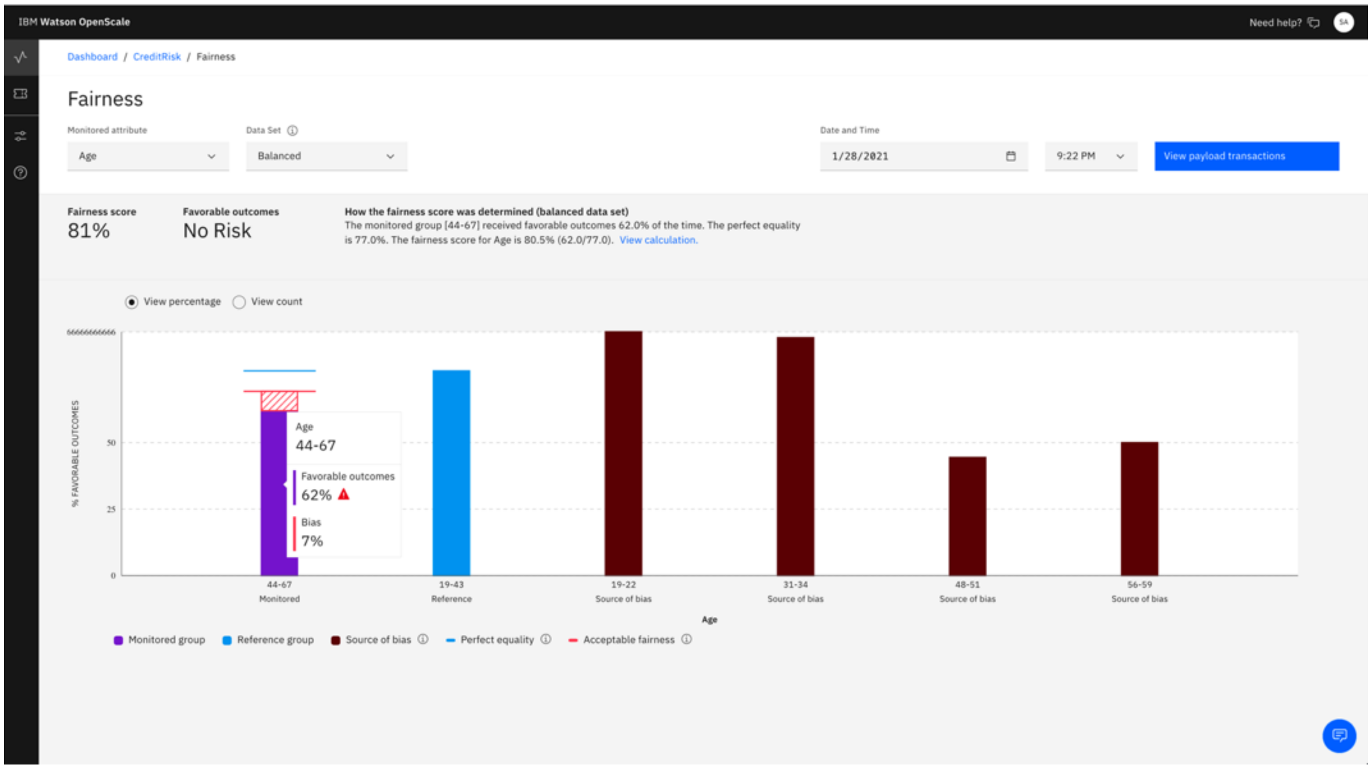
Fairness score
86%

Favorable outcomes
No Risk

How the fairness score was determined (balanced data set)
The monitored group female received favorable outcomes 66.0% of the time. The perfect equality is 77.0%. The fairness score for Sex is 85.7% (66.0/77.0). View calculation.

The red bar indicates that in-order to have met our threshold, we would need to increase that by 3% so we are below; our model missed, our model is more biased (3%) than what the threshold (90%) says it should be. So that 3% is the gap we need to address to make this prediction fairer.
Perhaps there are not enough records for females in the training data so they are under-representative of the sample so we may look into obtaining more data with a certain cohort. There may be some implicit bias built in the practice of giving out loans to certain cohort of the population.
You would also evaluate your training data for historical bias. The German Credit Report dataset is from the 1980's in Germany and so there may be some built in bias as to how many females were approved for the loan.

2. Change the **Monitored attribute** to **Age** from the drop-down list.


Fairness

Monitored attribute: Age

Data Set: Balanced

Date and Time: 1/28/2021 9:22 PM

Fairness score: 81%

Favorable outcomes: No Risk

How the fairness score was determined (balanced data set)
The monitored group [44-67] received favorable outcomes 62.0% of the time. The perfect equality is 77.0%. The fairness score for Age is 80.5% (62.0/77.0). View calculation.

The assumption here is that younger folks are being rejected more often that older folks. The reason is not necessarily age, but other contributing factors such as amount of funds in their checking/savings account or duration of employment, loan amount and so forth. These are indirect attributes that may have caused that 7% bias in the new evaluation based on a 90% threshold.
The black bars depict where the bias is most concentrated.

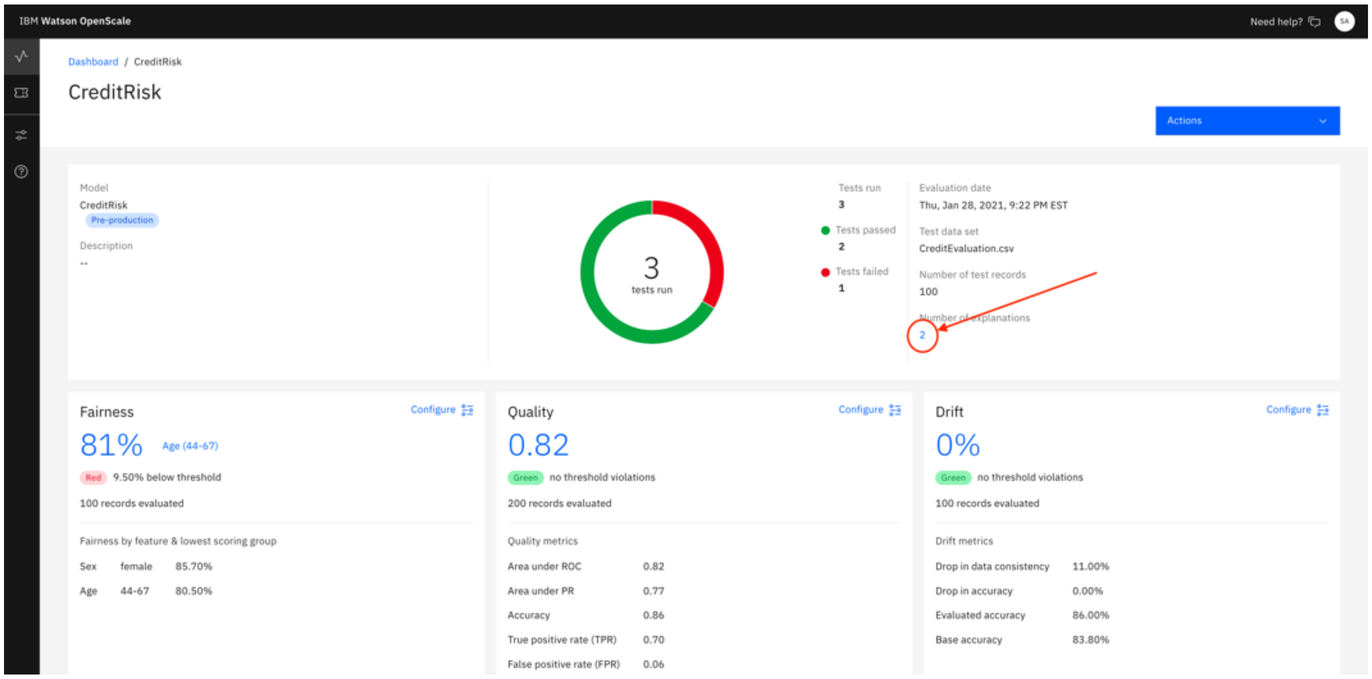
3. Hover your cursor over any of the black bars, the percent that shows up means that a certain percentage of folks in that age group are being approved for a loan.

So, there is not much difference between a 44-year-old (monitored group) and 40-year-old (reference group), yet a big difference between a 27-year-old and a 50-year-old applicant when it comes to loan approval.

Understanding Explanations

Explanability feature helps you look at individual transactions and determine exactly which feature-values are influencing your prediction models.

1. Click the number **2** in the Number of explanations section



IBM Watson OpenScale

Need help?

Dashboard / CreditRisk

CreditRisk

Actions

Model
CreditRisk
Pre-production

Description
--

Tests run
3

● Tests passed
2

● Tests failed
1

Evaluation date
Thu, Jan 28, 2021, 9:22 PM EST

Test data set
CreditEvaluation.csv

Number of test records
100

Number of explanations
2

Fairness — Configure

81% Age (44-67)

Red 9.50% below threshold

100 records evaluated

Fairness by feature & lowest scoring group

Sex	female	85.70%
Age	44-67	80.50%

Quality — Configure

0.82

Green no threshold violations

200 records evaluated

Quality metrics

Area under ROC	0.82
Area under PR	0.77
Accuracy	0.86
True positive rate (TPR)	0.70
False positive rate (FPR)	0.06

Drift — Configure

0%

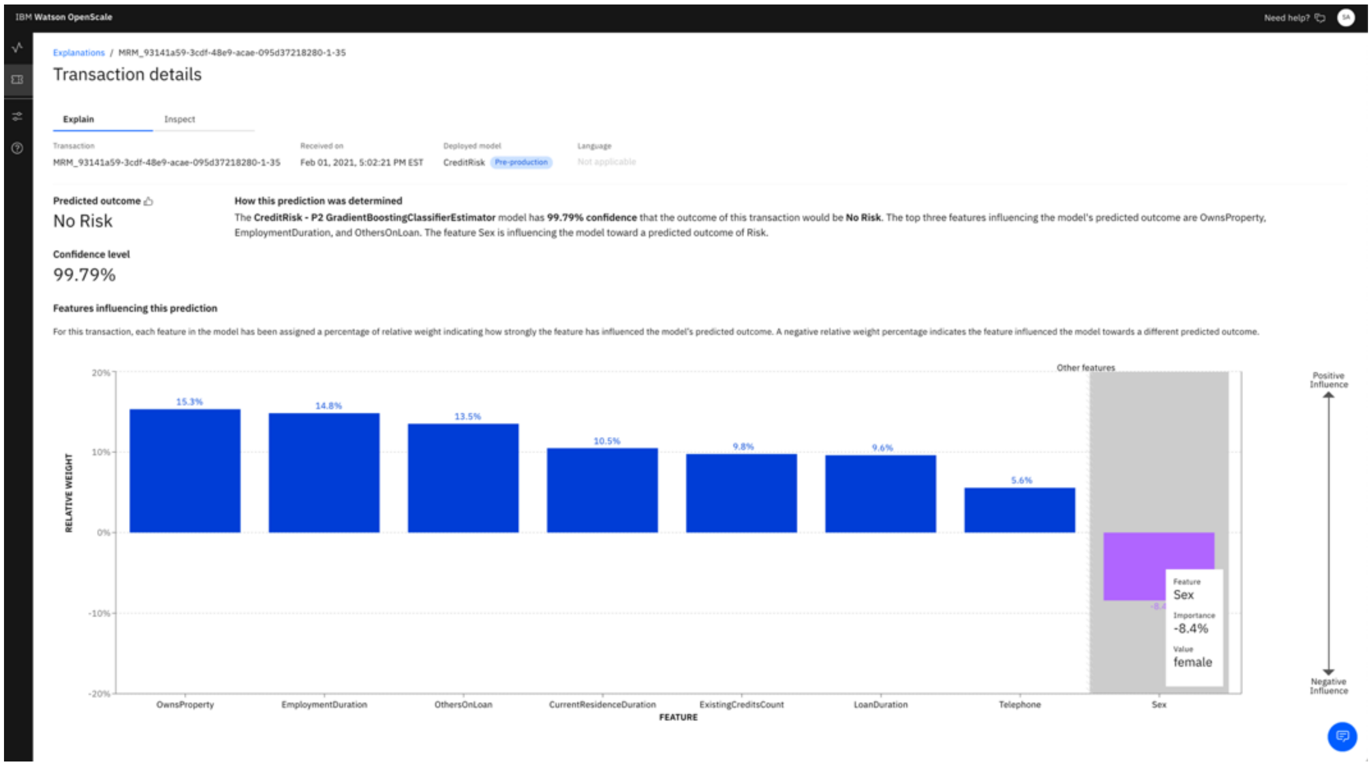
Green no threshold violations

100 records evaluated

Drift metrics

Drop in data consistency	11.00%
Drop in accuracy	0.00%
Evaluated accuracy	86.00%
Base accuracy	83.80%

2. Select the **No Risk** explanation.



IBM Watson OpenScale

Need help?

Explanations / MRM_93141a59-3cdf-48e9-acae-095d37218280-1-35

Transaction details

Explain Inspect

Transaction	Received on	Deployed model	Language
MRM_93141a59-3cdf-48e9-acae-095d37218280-1-35	Feb 01, 2021, 5:02:21 PM EST	CreditRisk Pre-production	Not applicable

Predicted outcome
No Risk

Confidence level
99.79%

How this prediction was determined
The **CreditRisk - P2 GradientBoostingClassifierEstimator** model has **99.79% confidence** that the outcome of this transaction would be **No Risk**. The top three features influencing the model's predicted outcome are OwnsProperty, EmploymentDuration, and OthersOnLoan. The feature Sex is influencing the model toward a predicted outcome of Risk.

Features influencing this prediction

For this transaction, each feature in the model has been assigned a percentage of relative weight indicating how strongly the feature has influenced the model's predicted outcome. A negative relative weight percentage indicates the feature influenced the model towards a different predicted outcome.

Let's take a moment and understand the metrics revealed. Watson Openscale examines predictions made by a model that attempts to predict the risk of a credit application using features such as the amount and duration of the loan the credit history salary and employment status of the applicant and other information the model outputs a prediction of either risk or no risk as well as a prediction probability that represents how confident the model is in its decision to explain a transaction.

This is individual prediction level explanation as to why the model made the decision it did. The values in this table were generated by the industry standard open-source lime algorithm which performs thousands or tens of thousands of perturbations on the data slightly altering each feature value and sending the new data to the model. The impact on the resulting prediction and probability allows Watson OpenScale to assign weights to each value representing how important that value is to the model's final prediction

The percentages represent the weight each feature has on the prediction so adding up the percentages for features contributing to risk and features contributing to no risk will equal 100 percent.

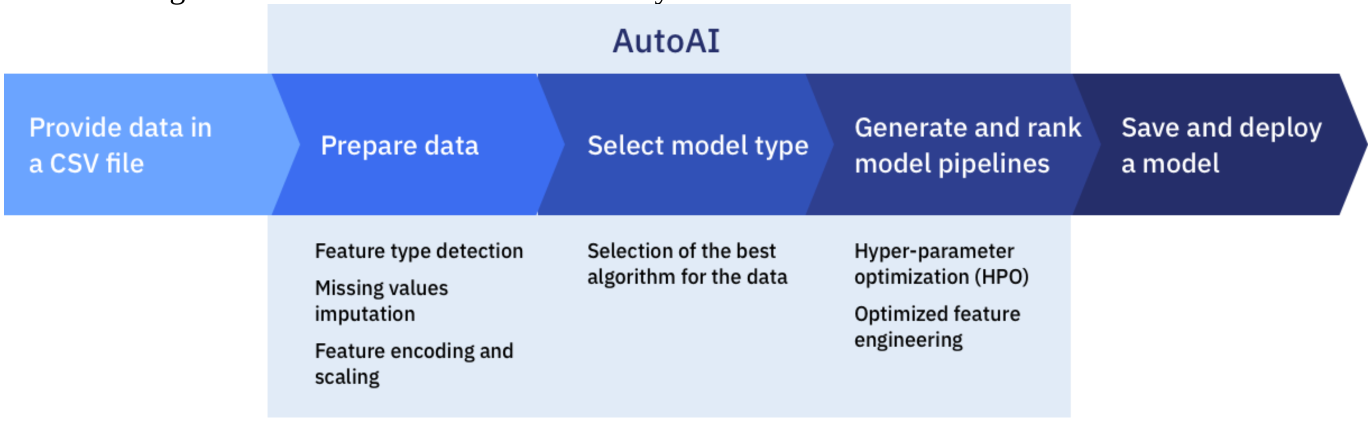
Not all feature values are shown here. Those whose influence on the prediction fall below a particular threshold are omitted which is why these numbers do not add up to 100% additionally, Watson OpenScale offers more insight into the prediction

In this example, if u have an 18-year-old who is apply for $5,000 loan over 48 months, then what if they were 20, and what if it was $10,000 loan? The model does 10000 of switches and comes back and says this were the biggest impact on the prediction:
For example, since there was no one on the loan, it is less risky. The purple box in the far right says this is the biggest factor that this loan application may be risky.
As a loan officer if the AI model predicted that a certain loan would be risky, the loan officer can use this information to support the decision to reject the loan, while being able to provide explanation to the applicant as to why the loan was rejected.

Section 2. Appendix

About AutoAI

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.


AutoAI
Provide data in a CSV file
Prepare data
Select model type
Generate and rank model pipelines
Save and deploy a model
Feature type detection
Missing values imputation
Feature encoding and scaling
Selection of the best algorithm for the data
Hyper-parameter optimization (HPO)
Optimized feature engineering

AutoAI automatically runs the following tasks to build and evaluate candidate model pipelines:

Below are additional explanations of the adjustments you can make to your experiment settings.

Data source, where you can adjust:

- whether to subsample data. If you have a large data set, you can choose to train with a representative sample of the data to speed up pipeline creation. You can specify whether subsampling should be done by a percentage of the training data or by a specified number of rows.
- the percentage of training data vs holdout data. Training data is used to train the model, and holdout data is withheld from training the model and used to measure the performance of the model.
- columns to include. You can choose to include columns with data that supports the prediction column and exclude irrelevant columns to speed up pipeline performance.

Prediction settings, where you can:

- change the model type. AutoAI selects a model type that best suits a sampling of the data, but you can override it. For example, if the sample data for the prediction column contains only two types of values, AutoAI will choose binary classification as the model type. If you know there are more than two values in the column, you can override the setting and choose multiclass classification instead. For binary classification models you can also edit the positive class.
- change the metric to be optimized for the experiment. **Note:** For a binary classification experiment, if you change the metric to *Precision*, *Average Precision*, *Recall*, or *F1*, a Positive Class is required. Confirm that the Positive Class is correct, or the experiment might generate inaccurate results.
- optionally specify which algorithms AutoAI should consider for pipeline creation. Only checked algorithms will be considered during the model selection phase of the experiment.

- change the number of algorithms to use to create pipelines. By default, AutoAI will choose the top two performing algorithms of the ones it considers and use those algorithms to generate 8 pipelines that you can view and compare, but you can change the number from 1 to 4. For example, if you select 3 algorithms, AutoAI will identify the top three performing algorithms and use them to generate a total of 12 pipelines that you can view, compare, and save as models. Note that more pipelines will increase the training time for the experiment and use more resources.

Runtime settings, where you can review experiment settings.

Analyze results

The AutoAI graphical tool in Watson Studio automatically analyzes your data and generates candidate model pipelines customized for your predictive modeling problem. These model pipelines are created iteratively as AutoAI analyzes your dataset and discovers data transformations, 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.
Required service: Watson Machine Learning service
Data format: Tabular: CSV files, with comma (,) delimiter
Data size: Less than 1 GB

2.1.1. AutoAI process

About Drift

Watson OpenScale detects both drift in accuracy and drift in data.

- Drop in accuracy

  Estimates the drop in accuracy of the model at runtime. Model accuracy drops if there is an increase in transactions that are similar to those that the model did not evaluate correctly in the training data. This type of drift is calculated for structured binary and multi-class classification models only.

- Drop in data consistency

  Estimates the drop in consistency of the data at runtime as compared to the characteristics of the data at training time.

A drop in either model accuracy or data consistency led to a negative impact on the business outcomes that are associated with the model and must be addressed by retraining the model.

2.1.2. Drift analysis

Depending on whether you perform drift detection as part of batch processing or not, the drift displays are different.

2.1.3. Drift visualization for non-batch processing data

The drift visualization includes both graphical and numeric statistical data. By clicking the chart, you can display specific transactions that contribute to drift. The top reasons for detected drift display and includes a natural-language description of the observation as well as a list of unexpected values.

Specifically, from the **Select a transaction set from the chart or list below** section, you can choose the following views:

- Transactions responsible for drop in accuracy
- Transactions responsible for drop in accuracy and data consistency
- Transactions responsible for drop in data consistency
- Drift transactions are available in the transaction details screen, where you can click **Explain** to understand how a specific transaction has made it into the drift category.

2.1.4. Limitations

The following limitations apply to the drift monitor:

- Drift is supported for structured data only.
- Although classification models support both data and accuracy drift, regression models support only data drift.
- Drift is not supported for Python functions.

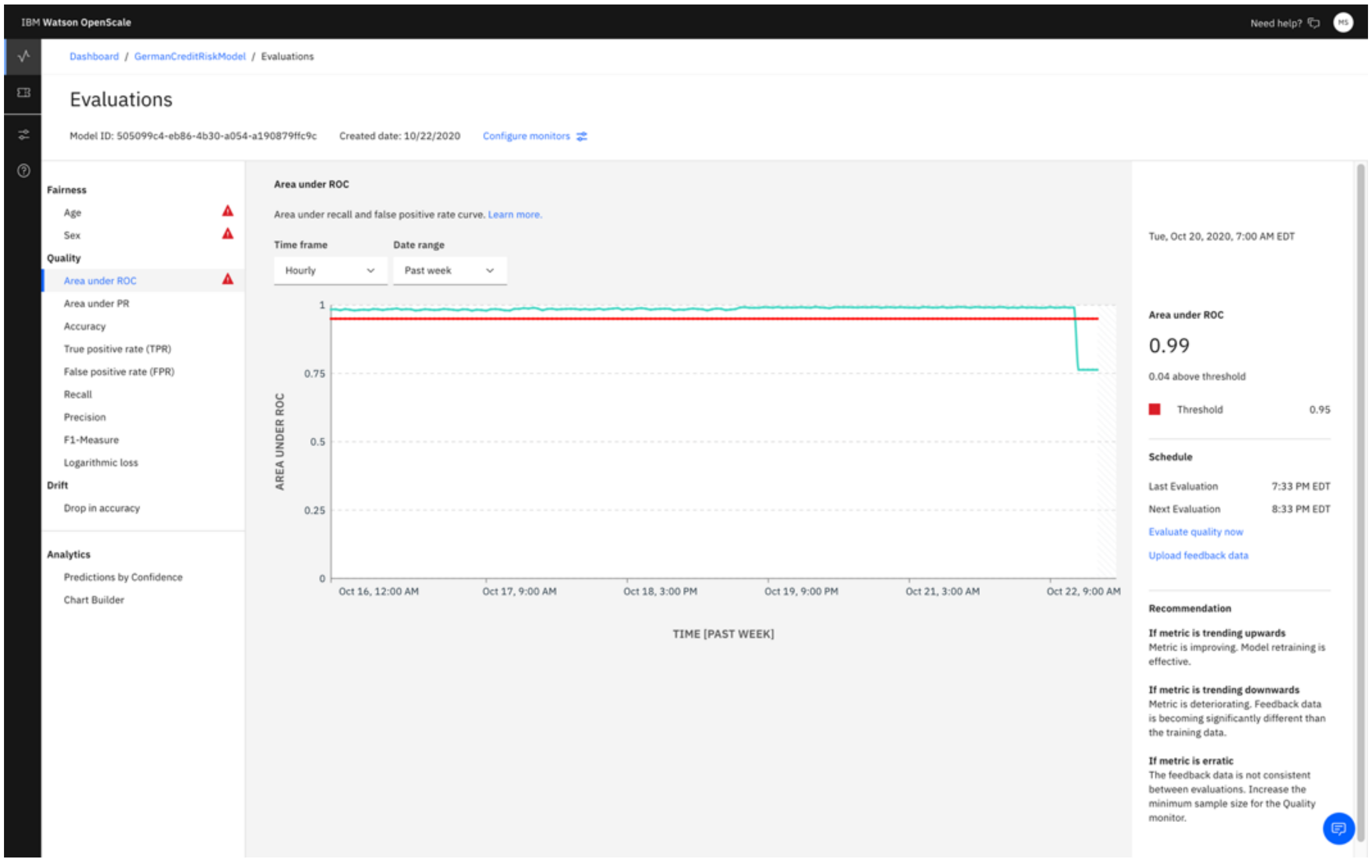
About Quality

Use quality monitoring to determine how well your model predicts outcomes. When quality monitoring is enabled, it generates a set of metrics every hour by default. You can generate these metrics on demand by clicking the **Check quality now** button or by using the Python client.

Quality metrics are calculated based on the following information:

- manually labeled feedback data,
- monitored deployment responses for these data.

For proper monitoring, feedback data must be logged to Watson OpenScale on a regular basis. The feedback data can be provided either by using “Add Feedback data” option or using Python client or REST API.
For machine learning engines other than Watson OpenScale, such as Microsoft Azure ML Studio, Microsoft Azure ML Service, or Amazon Sagemaker ML quality monitoring creates additional scoring requests on the monitored deployment.
You can review all metrics values over time on the Watson OpenScale dashboard:


IBM Watson OpenScale

Need help?

Dashboard / GermanCreditRiskModel / Evaluations

Evaluations

Model ID: 505099c4-eb86-4b30-a054-a190879ffc9c   Created date: 10/22/2020   Configure monitors

Area under ROC

Area under recall and false positive rate curve. Learn more.

Time frame: Hourly   Date range: Past week

Tue, Oct 20, 2020, 7:00 AM EDT

Area under ROC

0.99

0.04 above threshold

🟥 Threshold	0.95

Schedule

Last Evaluation	7:33 PM EDT
Next Evaluation	8:33 PM EDT

Evaluate quality now

Upload feedback data

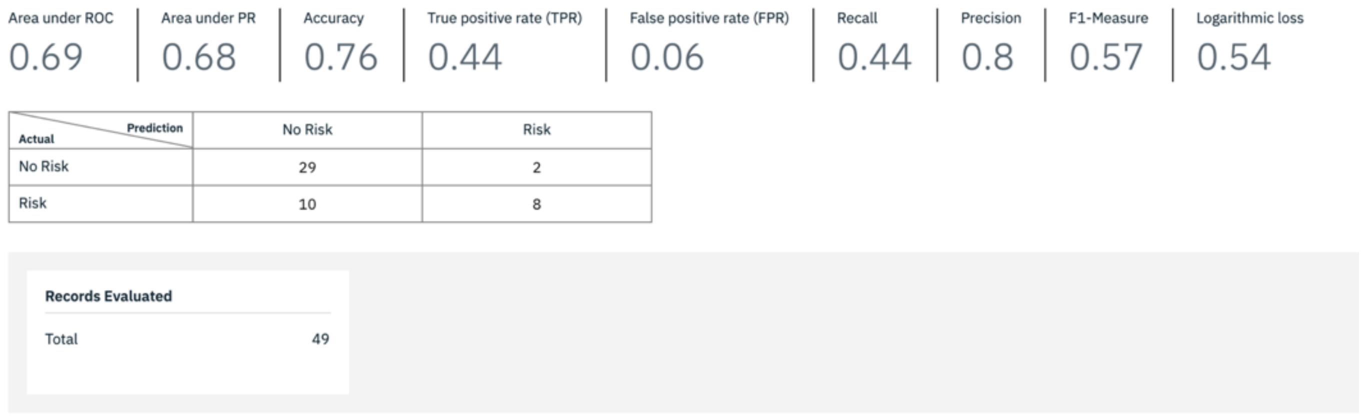
Recommendation

If metric is trending upwards
Metric is improving. Model retraining is effective.

If metric is trending downwards
Metric is deteriorating. Feedback data is becoming significantly different than the training data.

If metric is erratic
The feedback data is not consistent between evaluations. Increase the minimum sample size for the Quality monitor.

To review related details, such as confusion matrix for binary and multi-class classification, which are available for some metrics, click the chart.



Area under ROC	Area under PR	Accuracy	True positive rate (TPR)	False positive rate (FPR)	Recall	Precision	F1-Measure	Logarithmic loss
0.69	0.68	0.76	0.44	0.06	0.44	0.8	0.57	0.54

Actual \ Prediction	No Risk	Risk
No Risk	29	2
Risk	10	8

Records Evaluated

Total	49

2.1.5. Supported quality metrics

The following quality metrics are supported by Watson OpenScale:

2.1.6. Binary classification problems

For binary models, Watson OpenScale tracks when the quality of the model falls below an acceptable level. For binary classification models, it will check the **Area under ROC** score which measures the model’s ability to distinguish two classes. The higher the Area under ROC score, the better the model is at identifying class A as class A and class B as class B.

2.1.7. Regression classification problems

For regression models, Watson OpenScale tracks when the quality of the model falls below an acceptable level and checks the **R squared** score. R squared measures correlation between predicted values and actual values. The higher the R squared score, the better the model fits to the actual values.

2.1.8. Multiclass classification problems

For multi-classification models, Watson OpenScale tracks when the quality of the model falls below an acceptable level and checks the **Accuracy** score which is the percentage of predictions the model got right.

- Accuracy
- Weighted True Positive Rate (wTPR)
- Weighted False Positive Rate (wFPR)
- Weighted recall
- Weighted precision
- Weighted F1-Measure
- Logarithmic loss

2.1.9. Supported quality details

The following details for quality metrics are supported by Watson OpenScale:

2.1.10. Confusion matrix

Confusion matrix helps you to understand for which of your feedback data the monitored deployment response is correct and for which it is not.
For more information, see Confusion matrix.

About Fairness

Use IBM Watson OpenScale fairness monitoring to determine whether outcomes that are produced by your model are fair or not for monitored group. When fairness monitoring is enabled, it generates a set of metrics every hour by default. You can generate these metrics on demand by clicking the **Check fairness now** button or by using the Python client.

Watson OpenScale automatically identifies whether any known protected attributes are present in a model. When Watson OpenScale detects these attributes, it automatically recommends configuring bias monitors for each attribute present, to ensure that bias against these potentially sensitive attributes is tracked in production.
Currently, Watson OpenScale detects and recommends monitors for the following protected attributes:

- sex
- ethnicity
- marital status
- age
- zip code or postal code

In addition to detecting protected attributes, Watson OpenScale recommends which values within each attribute should be set as the monitored and the reference values. For example, Watson OpenScale recommends that within the `Sex` attribute, the bias monitor be configured such that `Female` and `Non-`

`Binary` are the monitored values, and `Male` is the reference value. If you want to change any of the recommendations, you can edit them via the bias configuration panel.
Recommended bias monitors help to speed up configuration and ensure that you are checking your AI models for fairness against sensitive attributes. As regulators begin to turn a sharper eye on algorithmic bias, it is becoming more critical that organizations have a clear understanding of how their models are performing, and whether they are producing unfair outcomes for certain groups.

2.1.11. Understanding Fairness

Watson OpenScale checks your deployed model for bias at runtime. To detect bias for a deployed model, you must define fairness attributes, such as `Age` or `Sex`, as detailed in the Configuring the Fairness monitor section.
It is mandatory to specify the output schema for a model or function in Machine Learning, for bias checking to be enabled in Watson OpenScale. The output schema can be specified using the `client.repository.ModelMetaNames.OUTPUT_DATA_SCHEMA` property in the metadata part of the `store_model` API. For more information, see the IBM Watson Machine Learning client documentation.

2.1.12. How it works

Before configuring the Fairness monitor, there a few key concepts that are critical to understand:

- Fairness attributes are the model attributes for which the model is likely to exhibit bias. As an example, for the fairness attribute **`Sex`**, the model could be biased against specific values, such as `Female` or `Non-binary`. Another example of a fairness attribute is **`Age`**, where the model could exhibit bias against people in an age group, such as `18 to 25`.
- Reference and monitored values: The values of fairness attributes are split into two distinct categories: Reference and Monitored. The Monitored values are those which are likely to be discriminated against. In the case of a fairness attribute like **`Sex`**, the Monitored values could be `Female` and `Non-binary`. For a numeric fairness attribute, such as **`Age`**, the Monitored values could be `[18-25]`. All other values for a given fairness attribute are then considered as Reference values, for example `Sex=Male` or `Age=[26,100]`.
- Favorable and unfavorable outcomes: The output of the model is categorized as either Favorable or Unfavorable. As an example, if the model is predicting whether a person should get a loan or not, then the Favorable outcome could be `Loan Granted` or `Loan Partially Granted`, whereas the Unfavorable outcome might be `Loan Denied`. Thus, the Favorable outcome is one that is deemed as a positive outcome, while the Unfavorable outcome is deemed as being negative.

The Watson OpenScale algorithm computes bias on an hourly basis, using the last `N` records present in the payload logging table; the value of `N` is specified when configuring Fairness. The algorithm perturbs these last `N` records to generate additional data.
The perturbation is done by changing the value of the fairness attribute from Reference to Monitored, or vice-versa. The perturbed data is then sent to the model to evaluate its behavior. The algorithm looks at the last `N` records in the payload table, and the behavior of the model on the perturbed data, to decide if the model is acting in a biased manner.
A model is deemed to be biased if, across this combined dataset, the percentage of Favorable outcomes for the Monitored class is less than the percentage of Favorable outcomes for the Reference class, by some threshold value. This threshold value is to be specified when configuring Fairness.

Fairness values can be more than 100%. This means that the Monitored group received more favorable outcomes than the Reference group. In addition, if no new scoring requests are sent, then the Fairness value will remain constant.

2.1.13. Balanced data and perfect equality

For balanced data sets the following concepts apply:

- To determine the perfect equality value, reference group transactions are synthesized by changing the monitored feature value of every monitored group transaction to all reference group values. These new synthesized transactions are added to the set of reference group transactions and evaluated by the model.

  If the monitored feature is `SEX` and the monitored group is `FEMALE`, all `FEMALE` transactions are duplicated as `MALE` transactions. Other features values remain unchanged. These new synthesized `MALE` transactions are added to the set of original `MALE` reference group transactions.

- From the new reference group, the percentage of favorable outcomes is determined. This percentage represents perfect fairness for the monitored group.
- The monitored group transactions are also synthesized by changing the reference feature value of every reference group transaction to the monitored group value. These new synthesized transactions are added to the set of monitored group transactions and evaluated by the model.

If the monitored feature is `SEX` and the monitored group is `FEMALE`, all `MALE` transactions are duplicated as `FEMALE` transactions. Other features values remain unchanged. These new synthesized `FEMALE` transactions are added to the set of original `FEMALE` monitored group transactions.

2.1.14. Do the math

The fairness metric used in Watson OpenScale is disparate impact, which is a measure of how the rate at which an unprivileged group receives a certain outcome or result compares with the rate at which a privileged group receives that same outcome or result.
The following mathematical formula is used for calculating disparate impact:

```
                      (num_positives(privileged=False) / num_instances(privileged=False))
Disparate impact =    ______________________________________________________________________

                      (num_positives(privileged=True) / num_instances(privileged=True))
```

where `num_positives` is the number of individuals in the group (either privileged=False, i.e. unprivileged, or privileged=True, i.e. privileged) who received a positive outcome, and num_instances is the total number of individuals in the group.
The resulting number will be a percentage, which is the percentage that the rate at which unprivileged group receives the positive outcome is of the rate at which the privileged group receives the positive outcome. For instance, if a credit risk model assigns the “no risk” prediction to 80% of unprivileged applicants and to 100% of privileged applicants, that model would have a disparate impact (presented as the fairness score in Watson OpenScale) of 80%.
In Watson OpenScale, the positive outcomes are designated as the favorable outcomes, and the negative outcomes are designated as the unfavorable outcomes. The privileged group is designated as the reference group, and the unprivileged group is designated as the monitored group.
The following mathematical formula is used for calculating perfect equality:

```
Perfect equality =   Percentage of favorable outcomes for all reference transactions,
                     including the synthesized transactions from the monitored group
```

For example, if the monitored feature is `SEX` and the monitored group is `FEMALE`, the following formula shows the equation for perfect equality:

```
Perfect equality for `SEX` = Percentage of favorable outcomes for `MALE` transactions, including the
synthesized transactions that were initially `FEMALE` but changed to `MALE`
```

2.1.15. Bias visualization

When potential bias is detected, Watson OpenScale performs several functions to confirm whether the bias is real. Watson OpenScale perturbs the data by flipping the monitored value to the reference value and then running this new record through the model. It then surfaces the resulting output as the debiased output. Watson OpenScale also trains a shadow debiased model that it then uses to detect when a model is going to make a biased prediction.

Two different datasets are used for computing fairness and accuracy. Fairness is computed by using the payload + perturbed data. Accuracy is computed by using the feedback data. To compute accuracy, Watson OpenScale needs manually labelled data, which is only present in feedback table.

The results of these determinations are available in the bias visualization, which includes the following views. (You only see the views if there is data to support

- **Balanced**: This balanced calculation includes the scoring request received for the selected hour plus additional records from previous hours if the minimum number of records required for evaluation was not met. Includes additional perturbed/synthesized records used to test the model's response when the value of the monitored feature changes.

  Take note of the following payload and perturbed details:

  - Monitored groups with fairness score
  - Reference groups with fairness score
  - Source of bias
- **Payload**: The actual scoring requests received by the model for the selected hour.

  Take note of the following payload details:

  - Payload data with stacked bar chart
  - Favorable and Unfavorable outcomes that correspond to the model labels
  - The Date and Time that transactions were loaded
- **Training**: The training data records used to train the model.

  Take note of the following training details:

  - Number of training data records. Training data is read one time, and distribution is stored in the `subscription/fairness_configuration` variable. While computing distribution, the number of training data records is also retrieved and stored in the same distribution.
  - When training data is changed, meaning if the `POST /data_distribution` command is run again, this value is updated in the

`fairness_configuration/training_data_distribution` variable. While sending the metric, this value is sent as well.

- **Debiased**: The output of the debiasing algorithm after processing the runtime and perturbed data. Selecting the **debiased** radio button shows you the changes in the debiased model, versus the model in production. The chart reflects the improved outcome status for groups.

  Take note of the following debiased details:

  - The result of debiasing the model
  - For monitored groups, such as `age` and `sex` the before and after scores

2.1.16. Example

Consider a data point where, for `Sex=Male` (Reference value), the model predicts a Favorable outcome, but when the record is perturbed by changing `Sex` to `Female` (Monitored value), while keeping all other feature values the same, the model predicts an Unfavorable outcome. A model overall is said to exhibit bias if there are sufficient data points (across the last `N` records in the payload table, plus the perturbed data) where the model was acting in a biased manner.

2.1.17. Supported models

Watson OpenScale supports bias detection only for those models and Python functions which expect some kind of structured data in its feature vector.
Fairness metrics are calculated based on the scoring payload data.
For proper monitoring purpose, every scoring request should be logged in Watson OpenScale as well. Payload data logging is automated for IBM Watson Machine Learning engines.
For other machine learning engines, the payload data can be provided either by using the Python client or the REST API.
For machine learning engines other than IBM Watson Machine Learning, fairness monitoring creates additional scoring requests on the monitored deployment.
You can review the following information:

- Metrics values over time
- Related details, such as favorable and unfavorable outcomes
- Detailed transactions
- Recommended debiased scoring endpoint

2.1.18. Supported fairness metrics

The following fairness metrics are supported by Watson OpenScale:

- Fairness for a group

The following protected attributes are supported by Watson OpenScale:

- sex
- ethnicity
- marital status
- age
- zip code

2.1.19. Supported fairness details

The following details for fairness metrics are supported by Watson OpenScale:

- The favorable percentages for each of groups
- Fairness averages for all the fairness groups
- Distribution of the data for each of the monitored groups
- Distribution of payload data