In this module, you will learn about an enterprise-ready data science platform by IBM, called Watson Studio. You'll learn about some of the features and capabilities of what data scientists use in the industry. You’ll also learn about other IBM tools used to support data science projects, such as IBM Watson Knowledge Catalog, Data Refinery, and the SPSS Modeler.

**Objetivos de aprendizagem**

* Define the Watson Studio.
* Explain how Watson Studio can streamline data projects and make them easier to manage.
* List some of the primary assets included in Watson Studio.
* Create a project in Watson Studio and add an interactive Jupyter workbook to work with a real data set. Link GitHub to Watson Studio and explain when to use public vs. private repositories.
* Describe other tools essential to data science and their applications.
* Describe Data Refinery and how it benefits the data scientists who use it.
* Describe the IBM SSPS Modeler, its usage, and some of the capabilities it offers.
* Load and run a sample project in Watson Studio and examine the results.
* Use Watson Studio to run an autonumeric model, examine the results and get predictions for new use cases.
* Describe the use case for the IBM SPSS Modeler and list a few of its main features.
* Summarize the characteristics of the IBM SPSS Statistics software application and its use by data scientists.
* Explain how models are deployed into production environments.
* Summarize how Auto AI in Watson Studio benefits organizations who lack a fully staffed data science team.
* List the features included with IBM Watson and explain how it ensures fairness and explainability of model drift.

# **Other IBM Tools for Data Science**

In this video, we will look at

several other IBM tools that help

data scientists in their day to day work.

Watson Knowledge Catalog helps data scientists to

catalog and manage all their data resources.

Data refinery provides graphical tools

for analyzing and preparing data.

SPSS based products include

easy to use graphical interfaces for

wide varieties of

statistical and machine learning algorithms

and data transformations.

We will talk about approaches to model deployment,

including open standards and Watson Machine Learning.

Newer features of Watson Studio include

AutoAI that automatically computes

the best data pipeline and

Watson OpenScale which helps to ensure

fairness and explainability of the models.

# **Data Refinery**

Hi, I'm Sonali Surange Dev.

Reproduza o vídeo começando em ::11 e siga a transcrição0:11

Data scientists often end up spending a lot

of time doing mundane tasks like cleansing, shaping and preparing data.

Typically these tasks are roadblocks for starting the more enjoyable part of

analyzing the data sets or building and training machine learning models.

Reproduza o vídeo começando em ::28 e siga a transcrição0:28

This is because data sets typically are not in a format that can be readily used.

Reproduza o vídeo começando em ::34 e siga a transcrição0:34

They first need to be cleansed, refined before they are useable by a data scientist.

Reproduza o vídeo começando em ::40 e siga a transcrição0:40

IBM Data Refinery addresses this issue and simplifies the task of refining data and

its workflows.

It provides a self-service data preparation environment where you

can quickly analyze, cleanse and prepare data sets.

Data refinery is available with

Watson Studio on public cloud, private cloud and desktop.

In the rest of the

video we will walk through a scenario and see Data Refinery in action.

In this

scenario we will use Data Refinery to find the best deals

using data about discounts offered over time.

We will then automate the

analysis to run on a regular schedule.

Before the Data Scientist starts, she looks at the data

distribution and notices that the inSale column is missing data.

She visualizes the offer column and notices that it contains valuable

information about discounts.

Many fields contain the percent of information,

some contain references to previous price indicating a new reduced price

being available.

She decides to derive sale from offer.

She uses a conditional decrease operation to derive if the product is on

sale.

Next she uses a filter operation

to eliminate deals that are not on sale

She then wants to pick up the bargains.

She uses the replace substring operation

and provides a pattern that extracts the discounts from the offer.

After

converting the discount values to a decimal

she can visually see the discounts that were available.

She needs to find the

months that offered the best deals.

She visualizes the dateUpdated and notices

that the date field has a variety of formats, some with dashes some with

slashes and some with months as text.

She hopes that Data Refinery can normalize the

data and extract a month.

She uses the convert column operation to convert to

date and selects ymd.

Next she extracts month and creates a derived column

called discountMonth.

The data now represents all brands and products

providing sales and the month the offer was available.

The data scientist is only

interested in her preferred brands.

Over time she has built a list of preferred

brands and has imported the data in her project.

Data Refinery provides

relational transformations such as left, inner right, full, semi and anti-join.

To ensure

that the data only contains her preferred brand she uses a semi-join

operation which narrows the brands to match her preferences.

She then selects the keys for the join and the resulting fields.

The visual results now confirms that the brands match the preferences.

To find the best

possible deals she needs to perform some aggregations.

Several features determine a good deal.

She is interested in the best offer and

duration when the discounts are active.

Aggregating the sale data will help

understand the deals.

She groups the columns by brand and discountMonth and

calculates the maximum discount.

Finally she sorts the result in descending order

Data refinery is now displaying the best deals by brand preferences and the

duration which the offer is available.

The last step is to execute the analysis on the full dataset.

She starts the full

analysis, which she can monitor for the completion status.

It's time to automate the analysis which runs on a regular basis.

The data in the

database can grow over time.

She uses a personalized runtime to match the larger

data volumes and sets a schedule for automation.

The hourly schedule reads

from updated data from the database and writes to the target table.

Data Refinery has helped her uncover deals in

the raw data through a small set of

operations and transformations with the bulk of the work done for her.

Thank you

for watching

# **SPSS Modeler Flows in Watson Studio**

In this video, we will take a look at an easy-to-use, graphical way to build machine learning models

and pipelines.

SPSS Modeler Flows is a part of Watson Studio, which was inspired by another product, IBM

SPSS Modeler.

We'll discuss that product in a later unit.

Let’s have a look again at the overview of different tool categories.

Modeler flows include some data management capabilities, as well as tools for data preparation,

visualization, and model building.

All flows are created using a drag-and-drop editor and consist of “nodes” of various

types, with data “flowing” from one node to the next according to their connections.

A sample Modeler flow shown here includes two data source nodes shown in purple on the

left; type, aggregate, filter, merge, filler, and partition nodes in the middle; 2 model

building nodes shown in pentagons.

Once a flow is executed and the models are built, the upside-down pentagon “model nuggets”

are created.

They can be used to see information about the models and to get predictions for new

data.

And the three green square nodes on the right provide model evaluation information in the

form of tables and charts.

You can build your SPSS Modeler flows by dragging different types of nodes from the left, the

part of the screen called the “palette,” to the "canvas," the main part of the screen.

Each flow starts with one or more data sources located in the “Import” group, and can

include some or all other types of nodes.

Watson Studio provides some sample flows to help new users.

In the Drug Study example shown here, we are using a small artificial data set.

The target variable is a categorical field, “Drug,” that has five categories, and

there are several predictor variables.

This flow creates a new “derived” field by dividing the values of one of the predictors

by values of another one, and at the end builds a small neural network model and a decision

tree model.

When a user clicks the “Run” button on the top panel, denoted by a triangle, the

flow is executed and the models build.

This is reflected in the new pentagon nodes, called “model nuggets,” that display under

each model node.

If you click on the three dots in the upper right corner of one of those nodes and select

“View Model”, you will see various types of model information.

By connecting new data sources to the model nugget, you can get predictions on new data.

The first window in the model viewer shows model accuracy and related measures, such

as precision and recall.

This toy data example enabled us to get perfect accuracy, which is normally not the case with

real life data.

The Confusion Matrix view shows how model predictions on the training data matched the

observed target values.

Once again, in this toy example all cases were classified correctly.

We can also look at Model Information, which displays a table that tells us more about

the details of the model.

Feature Importance displays a diagram that indicates the relative predictive strength

of various model inputs.

Finally, the Network Diagram gives a visual representation of the neural network model

we built.

On the left is the input layer, with units corresponding to each continuous predictor

and each category of the categorical predictors, plus a bias unit that is usually present in

each layer of a neural network.

In the middle, we see a “hidden layer” with 7 units, or neurons, and a bias unit.

On the right is the output layer with 5 units corresponding to the five target categories.

Controls on the right and bottom of the diagram enable some interactive exploration of the

model.

The colors of the connections between units indicate the values of the weights on those

connections.

We can also look at the decision tree model built using the C5 algorithm.

A Model Information table and Feature Importance chart appear as before.

Additionally, a Top Decision Rules table is displayed.

Decision tree models are popular because they have a special structure that makes it easy

to explain predictions or extract decision rules.

The tree diagram is also displayed.

On the left side of the canvas, we see a part of the model palette that can be used in the

flows.

At the top are “Auto Classifier” and “Auto Numeric” nodes that can be used for categorical

and continuous targets, respectively.

Those nodes will build several kinds of models and pick the best one based on a certain criterion.

Later, we will talk about the AutoAI feature of Watson Studio; AutoAI takes this capability

to the next level by automatically finding not only the best model, but an entire data

pipeline, which includes various data transformations.

In this video, you've learned how Modeler Flows in Watson Studio can help analysts to

create powerful machine learning pipelines using a graphical interface without the need

to write any code.

This feature was based on IBM SPSS Modeler.

Next, after completing a lab to give you hands-on experience with this powerful technology,

we will take a look at two other IBM products that can be used for Data Science: IBM SPSS

Modeler and IBM SPSS Statistics.

# **IBM SPSS Modeler**

In this lesson we will discuss two products that are very helpful for data

scientists. Both came to IBM with the SPSS acquisition in 2009. First is IBM

SPSS Modeler. Let's review the different tool categories we discussed previously.

IBM SPSS Modeler includes data management capabilities and tools for

data preparation, visualization, model building and model deployment. The

product was created by Integral Solutions Limited in the United Kingdom

in 1994 and was originally called Clementine. It was acquired by a company

called SPSS in 1998 and SPSS was in turn acquired by IBM in 2009. SPSS Modeler is

a data mining and text analytics software application. It's used to build

predictive models and conduct other analytics tasks. It has a visual

interface that enables users to leverage statistical and data mining algorithms

without programming. One of its main goals from the beginning was to create

complex predictive modeling pipelines that are easily accessible. A sample

modeler stream shown here includes one round data source node, three triangular

graph nodes, one hexagonal node for computing, a new variable, and a square

node for an output table. Below the canvas, we can see the rich node palette

with separate tabs for data sources, record in field operations, graphs, models,

output and so on. Nodes and different tabs have different

shapes with Pentagon's used for modeling nodes. Let's examine the sample stream

that comes as an example with the product. It starts with a data set of

telecommunications records and the goal is to build a model to predict which

customers are about to leave the service otherwise known as churn. The data source

is shown by the round node on the left side, a hexagon type node typically follows

a data source node and it enables us to

specify roles, target predictor or none. And measurement levels such as

continuous nominal or flag for all variables. The term flag is used to

denote a variable with two categories one of which can be considered positive

and the other negative. In this example the measurement level for the churn

field is set to flag and the role is set to target. All others are set as

predictors and inputs. The original data set has many fields and some of them

are not relevant to the target variable, so we first need to decide which fields

are more useful as predictors. There is a feature selection modeling node that

helps to do this. After the stream with the feature selection node is executed a

yellow model nugget gets created below it in the flow diagram.Using that nugget

we can generate a filter node that filters out the variables that are not

good predictors for the target. The data audit node located below the filtering

node shows various properties of the data such as numbers of outliers in each

variable and the percentage of valid values. It can also help to create a

special node for missing value imputation that is replacing missing

values of a variable with some valid values that can be selected based on

domain knowledge. Here variable log toll has greater than 50% missing values and

we will specify a value the mean to replace them. A super node in modeler is

a special node that is not found in the palette but is created by the user with

special functions included in it. The data audit node enables us to create a

super node for imputing missing values. It is shaped as a star and shown on the

right of the screen. Finally we attach the logistic

regression model node to the stream and click run. Another model nugget appears

and by clicking it we can see various model information and other output. in the output window that opens when we click on the model nugget the summary

tab shows the target inputs and some model building settings. Based on certain

advanced output settings that were specified before the model was built we

can also see a classification table, accuracy, and some other generated

outputs for the model. Note that these results are based on training data only.

To assess how well the model generates two other real-world data you should

always use a partition node to hold out a subset of records for the purposes of

testing and validation. Then, in the model setup screen select the use partitioned

data check box. This will help detect and avoid model

overfitting. Overfitting is defined as having significantly higher accuracy on

the training data. Data used for training the model then on tests or unseen data.

The yellow model nugget added earlier can also be used to compute predictions,

also called scores on the original data or on a new data source. All we need to

do is to connect the data source in question to the nugget, make sure it has

the predictor variables used in the model, and create an output to a table or

other structure for storing the scores. We can also specify settings for scoring

inside the model nugget. Note that if the model was built on transformed predictor

data, the same data transformation steps would be applied to the new data before

it can be scored by the model. The analysis node is the final node in the

stream. It attaches to a model nugget and when executed it will compute some model

evaluation metrics, auch as a confusion matrix and accuracy. In this example

we've only looked at a logistic regression model. IBM SPSS Modeler offers

a rich modeling palette that includes many classification, regression

clustering, Association rules and other models. It also contains large selections

of data source types, data transformations, graphs,

and output notes. And we haven't even talked about text analytics, entity

resolution and many other features of the product that can be extremely

helpful to data scientists. We could create an entire course on IBM SPSS

Modeler alone. You've learned how IBM SPSS Modeler helps analysts to create

powerful machine learning pipelines using graphical interface. Next, we will

talk about the original SPSS product now called IBM SPSS Statistics.

# **SPSS Statistics**

IBM SPSS Statistics evolved from an original product that was released in 1968. That product

was called “Statistical Package for Social Sciences,” or “SPSS.”

IBM SPSS Statistics is a statistical and machine learning software application and is widely

used in academia, government agencies, and large enterprises. It’s used to build predictive

models, perform statistical analysis of data, and conduct other analytic tasks. It has a

visual interface, which enables users to leverage statistical and data mining algorithms without

programming, although the interface is very different from Modeler. As you can see, the

main section of the screen looks very much like a spreadsheet; it displays data and allows

manual editing. This particular small data set, called “Employee Data”, was created

some time ago and does not represent real people. It is shipped with the product for

use in demos and tutorials.

At the bottom of the screen, we can see two tabs: Data View and Variable View. In the

Variable View, we can see and edit the information about all variables, including names, labels,

data types, and measurement levels. We can also specify labels for values of categorical

variables, and missing values.

At the top of the data window is a menu. Under File, if you select “Import Data,” you

will see a list of a wide variety of data formats that you can import. The product uses

its own data file format with the extension “.sav” that saves all the information

about the variables we just saw in Variable view. The menu enables importing from and

exporting to many other formats.

Under “Data,” you’ll find an extensive menu of possible data operations. Note that

Data Validation can be performed using user-defined rules that specify the expected behavior of

variable values. For example, if the date and month are kept in separate columns, the

date cannot exceed “31,” but for February, the date can’t exceed “29.” A special

rule can therefore be created and applied during data validation. Additionally, you

can enable some checks, such as percentage of missing values in a record or in the field.

When you click the “Transform” menu item, you’ll find a variety of available data

transformations. Under “Compute Variable…” you can write

a formula for a new variable based on existing variables. You can use any of the many mathematical

and statistical functions available in the product.

You also have the option to use automatic data preparation, similar to Modeler.

In the “Analyze” menu, you will see many types of statistical and machine learning

analysis. Under “Regression,” there are a variety of regression-related models. There

are other kinds of regressions that appear separately on the Analyze menu, including

General Linear Model, Generalized Linear Models, Mixed Models, and Loglinear.

Now let’s build a decision-tree model on the data. For this exercise we’ll try to

predict the "Employment category" field based on other fields. In the “Analyze” menu,

select “Classify” and then “Tree”. <Click> In the Decision Tree window, we can

specify the dependent variable “Employment Category,” and use most other fields -- except

id and bdate -- as predictors, or independent variables. Usually the ID variable should

not be used as a predictor, because it will not help with new cases, and the birthdate

does not seem to be a useful predictor in this example either. We’ll select “Exhaustive

CHAID” as our Growing Method, although there are also three other options available. Data

scientists often try many different models to see which one works best for their data.

Here we are just looking at one example model in order to illustrate how the product works.

Click the “Validation” button to open the Decision Tree Validation window. Here,

we select “Split-sample validation” to make sure we test the model on new data. Click

“OK” in the Decision Tree window, to <Click> generate the output, including the tree diagram

shown here. <Click> A Classification table is also displayed that shows how well the

model works on training and test data. In this case, the accuracy is 91.2% on training

data and only 85.6% on test data, which means the model does not generalize to new data

very well. It’s possible that by using different models, we can get better results.

Let’s move to the next menu item. When you click “Graphs,” you’ll open a versatile

Chart Builder, in addition to several other options.

The Chart Builder enables us to choose a style from the gallery and to drag required fields

onto the canvas, select colors, and choose from other options.

Here’s an example after we drag the “Previous Experience,” “Current Salary,” and Gender

variables to the corresponding slots to define the axis and colors for the dots on the chart.

The plot in the canvas is not based on real data, this example simply gives you an idea

of what to expect.

Here is the real plot obtained from the data that we’ve been using. It shows different

colored dots for gender, and regression lines that show the relationship of the current

salary to previous experience for each gender.

Throughout IBM SPSS Statistics, you’ll see a “Paste” button. When you click the “Paste”

button, instead of executing the task right away the application will open another window,

called the Syntax editor. Here, you can see the code called “syntax” pasted for you.

SPSS syntax is a special programming language.

For example, here is the code for the decision tree we just built. Once we have the syntax,

we can execute it, manually edit it, store it for later use, or send it to other users

of IBM SPSS Statistics. Experienced SPSS users can write the code from scratch, while others

might prefer to have it generated by the graphical interface. Remember, the option to paste syntax

is available in throughout the program. If the syntax is generated by all the steps

in a data analytics process -- opening the data set, applying any data transformations,

building models -- and then saved as a syntax file with the extension “.sps”, it’s

similar to saving a stream in IBM SPSS Modeler. However, one important difference is that

it does not allow for an easy way of scoring new records with the model. We’ll talk about

different ways to deploy models in the next section.

You’ve learned how IBM SPSS Statistics helps data scientists to analyze their data using

many statistical and machine learning techniques. Using a graphical user interface, we can create

complicated analysis that can be saved in the form of syntax and reused later.

Next, we will talk about predictive model deployment, an important part of the overall

data science lifecycle.

# **Model Deployment with Watson Machine Learning**

# **Auto AI in Watson Studio**

In earlier sections we saw how IBM SPSS Modeler and Watson Studio Modeler flows allow you

to graphically create a stream or flow that includes data transformation steps and machine

learning models.

Such sequences of steps are called data pipelines or ML pipelines.

This section examines a feature of Watson Studio that helps to automate the creation

of machine learning pipelines.

This allows data scientists to produce results much faster and to focus on more creative

work.

There is currently a shortage of qualified data scientists.

Many operations that a data scientist typically performs are repetitive and time-consuming.

Therefore, automating some of that repetitive work will help free up both new and experienced

data scientists to do the important work that they are trained to do.

The AutoAI system was developed by IBM Research experts in collaboration with IBM Distinguished

Engineer and two-time Kaggle Grandmaster Jean-Francois Puget.

It provides a graphical interface to create and deploy machine learning models with real

time visualizations.

AutoAI automatically performs typical machine learning steps, such as:

Data preparation Model selection

Feature engineering Hyper-parameter optimization

Users can view the progress on the graphical interface.

This example shows the training of a model to predict whether or not a customer is likely

to buy a tent from an outdoor equipment store.

We start with structured data.

In this historical data, there are four feature, or “predictor,” columns:

GENDER: The customer’s gender AGE: The customer’s age

MARITAL\_STATUS: “Married”, “Single”, or “Unspecified”

and PROFESSION: The general category of the customer’s

profession, such “Hospitality” or “Sales”, or simply “Other.”

The model will learn to predict the value for the IS\_TENT column; that is, whether or

not the customer bought a tent.

After we choose IS\_TENT as the column to predict, AutoAI analyzes the data and determines that

the IS\_TENT column contains True/False information, making this data suitable for a binary classification

model.

The default metric for a binary classification is ROC/AUC.

After we click Run experiment, an infographic shows the process of building the pipelines

as the model trains.

Once the pipeline creation is complete, we can view and compare the ranked pipelines

in a leaderboard.

The pipelines for the sample binary classification model are quite uniform because of the underlying

sample data.

To see pipelines in action, re-run the experiment as a regression experiment to predict purchase

amount.

That experiment gives better variation in the resulting pipelines.

After clicking “Pipeline comparison,” we can see how the pipelines differ on various

measures of model quality.

The pipelines can be saved as Machine Learning assets in the Watson Studio project.

Then they can be deployed and tested.

Currently AutoAI is available only for classification and regression models; there is a plan to

add time series model support in the future.

In this unit, you have learned how AutoAI automates typical data science tasks and helps

get better performing data pipelines more quickly, while also simplifying pipeline deployment

into production in Watson Machine Learning.

In the next section, we will discuss Watson OpenScale, which helps to ensure that your

models are fair, explainable, and up to date.

# **IBM Watson OpenScale**

Observações

[**Discutir**](https://www.coursera.org/learn/open-source-tools-for-data-science/discussions/weeks/3)

This video examines the feature of Watson studio that helps to ensure fairness and explain-ability of machine learning pipelines, as well as monitored their performance after deployment.

Reproduza o vídeo começando em ::20 e siga a transcrição0:20

IBM Watson Openscale is a product that includes several important features. It can test the model

and its predictions for fairness and apply ways to overcome bias. It can also help to

provide explanations for model predictions that are often hard to get but are necessary

for compliance in some application areas. It monitors the model performance and can detect

its deterioration or model drift over time. It can alert the users when drift is detected and

explain which predictors are causing it. We can specify criteria under which the model

gets automatically retrained on fresh data; it also helps to measure how the model helps

the business. The attributes to monitor for bias are automatically recommended based on

prior experience. They can be edited as needed. Openscale then keeps track of model predictions

for the specified groups and checks the bias in the predictions. Users need to know that

their AI models are fair but the date of their models were trained on and include

unwanted biases\a which may unintentionally be included in the resulting models. IBM Watson

Openscale can detect bias when a model is in production and not just when it's being built.

In this demo of Watson Openscale we'll monitor a credit risk model which has been trained to

determine whether or not someone is eligible for a loan, based on a variety of different features,

such as their credit history age and their number of dependents. After launching Openscale we can

see a few highlighted metrics for the monitored model, such as its quality and a fairness score.

Reproduza o vídeo começando em :2:5 e siga a transcrição2:05

What Openscale does is measure a model's fairness by calculating the difference between the rates at

which different groups, for example, women versus men, received the same outcome. A fairness value

below 100% means that the monitored group receives an unfavorable outcome more often

than the reference group. In this case, we see that women are receiving the no-risk outcome,

or getting approved for loans, at a lower rate than men. Openscale

enables the inspection of each model's training data and this reveals that there was more training

data for men than women. This can give some insight as to why the model exhibits bias against

women who apply for loans. Data scientists can use this information to approve the model. Now,

detecting bias is one thing-- Openscale can also mitigate it by creating a D bias model that runs

alongside the monitored one. In this case the D bias model is 12% more fair than the production

model. The D bias model has been trained to detect when your production model will make

a bias prediction so that you can isolate the specific transactions that result in the bias.

For each of these transactions Watson Openscale will flip the monitored value in a record to the

reference value, in this case from female to male, and leave all other data points in that

record the same. If this changes the prediction from risk to no-risk then the D biased model

will surface the no-risk outcome as the D biased result. This is just one of the ways that Watson

open scale helps you ensure that your models are fair explainable and compliant wherever your model

was built or is running. Insurance underwriters can use machine learning and Openscale to more

consistently and accurately assess claims risk, ensure fair outcomes for customers, and explain

AI recommendations for regulatory and business intelligence purposes. Why does an AI model arrive

at a given recommendation or prediction? Users and customers want an explanation and with most

models providing this information is not an easy task. IBM Watson Openscale explains predictions

in business friendly language. This credit application, for instance, was predicted to

be a risk. Openscale determines the features which contributed positively or negatively

to that prediction and spells them out. The explanation is presented visually, as well as in

a sentence-based text summary in order to ensure maximum clarity. Using proprietary IBM research

technology, Openscale also generates a contrast of explanations. Here we see the minimum changes to

this input record which would produce a different output, changing the prediction from risk to

no-risk. The explanations provided by Watson Openscale can help organizations comply with

regulations such as the Fair Credit Reporting Act and GDPR which give customers the right to ask for

reasons why their applications were denied. Before an AI model is put into production it must prove

it can make accurate predictions on test data, a subset of its training data; however, over time,

production data can begin to look different than training data, causing the model to start making

less accurate predictions. This is called drift. IBM Watson Openscale monitors a model's accuracy

on production data and compares it to accuracy on its training data. When a difference in accuracy

exceeds a chosen threshold Openscale generates an alert. Watson Openscale reveals which transactions

caused drift and identifies the top transaction features responsible. For instance, 25% of a

transactions causing drift in this loan approval model were problematic because of these features,

which contained data crucially different from the training data. The transactions causing drift can

be sent for manual labeling and use to retrain the model so that its predictive accuracy does not

drop at run time. Watson Openscale not only helps identify drift but also highlights its root cause

and provides transactions which can be turned into training data useful at fixing drift. It gives you

the insight you need to ensure that your models will consistently deliver the results you want

over time. For instance, the retrain version of the model, but based on the recommendations made

by Watson Openscale, started making accurate recommendations alleviating the drift. This is

just one of the ways that Watson Openscale helps you ensure your models are fair explainable and

compliant wherever your model was built or is running. In this video you have learned how

Openscale ensures fairness and explain ability of models and monitors for model drift in production.

This completes the model on IBM products for data scientists. Good luck on the quizzes!