Article
Niyaz Khafizov · Jul 27, 2018 4m read

Load a ML model into InterSystems IRIS

Hi all. Today we are going to upload a ML model into IRIS Manager and test it.

Note: I have done the following on Ubuntu 18.04, Apache Zeppelin 0.8.0, Python 3.6.5.

Introduction

These days many available different tools for Data Mining enable you to develop predictive models and analyze the data you have with unprecedented ease. InterSystems IRIS Data Platform provide a stable foundation for your big data and fast data applications, providing interoperability with modern DataMining tools.

In this series of articles we explore Data mining capabilities available with InterSystems IRIS. In the first article we configured our infrastructure and got ready to start. In the second article we built our first predictive model that predicts species of flowers using instruments from Apache Spark and Apache Zeppelin. In this article we will build a KMeans PMML model and test it in InterSystems IRIS.

Intersystems IRIS provides PMML execution capabilities. So, you can upload your model and test it against any data using SQL queries. It will show accuracy, precision, F-score and more.

Check requirements

First, download jpmml (look at the table and select suitable version) and move it to any directory. If you use Scala, it will be enough.

> PySpark2PMML must be paired with JPMML-SparkML based on the following compatibility matrix:
>
> | Apache Spark version | JPMML-SparkML development branch | JPMML-SparkML uber-JAR file |
> |---|---|---|
> | 2.0.X | `1.1.X` | 1.1.20 |
> | 2.1.X | `1.2.X` | 1.2.12 |
> | 2.2.X | `1.3.X` | 1.3.8 |
> | 2.3.X | `master` | 1.4.5 |

If you use Python, run the following in the terminal

```
pip3 install --user --upgrade git+https://github.com/jpmml/pyspark2pmml.git
```

After success message go to **Spark Dependencies** and add dependence to downloaded jpmml:

Dependencies

artifact	exclude
/home/guardian/Desktop/TMP/dev/java/lib/JDK18/intersystems-spark-1.0.0.jar	
/home/guardian/Desktop/TMP/dev/java/lib/JDK18/intersystems-jdbc-3.0.0.jar	
/home/guardian/Desktop/TMP/dev/java/lib/JDK18/jpmml-sparkml-executable-1.3.8.jar	

Create KMeans model

PMML builder uses pipelines, so I changed the code written in the previous article a bit. Run the following code in Zeppelin:

```
%pyspark
from pyspark.ml.linalg import Vectors
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.clustering import KMeans
from pyspark.ml import Pipeline
from pyspark.ml.feature import RFormula
from pyspark2pmml import PMMLBuilder

dataFrame=spark.read.format("com.intersystems.spark").\
option("url", "IRIS://localhost:51773/NEWSAMPLE").option("user", "dev").\
option("password", "123").\
option("dbtable", "DataMining.IrisDataset").load() # load iris dataset

(trainingData, testData) = dataFrame.randomSplit([0.7, 0.3]) # split the data into two sets
assembler = VectorAssembler(inputCols = ["PetalLength", "PetalWidth", "SepalLength", "SepalWidth"],
outputCol="features") # add a new column with features

kmeans = KMeans().setK(3).setSeed(2000) # clustering algorithm that we use

pipeline = Pipeline(stages=[assembler, kmeans]) # First, passed data will run against assembler and after
will run against kmeans.
modelKMeans = pipeline.fit(trainingData) # pass training data

pmmlBuilder = PMMLBuilder(sc, dataFrame, modelKMeans)
pmmlBuilder.buildFile("KMeans.pmml") # create pmml model
```

It will create a model, that predicts Species using PetalLength, PetalWidth, SepalLength, SepalWidth as features. It uses PMML format.

PMML is an XML-based predictive model interchange format that provides a way for analytic applications to describe and exchange predictive models produced by data mining and machine learning algorithms. It allows us to separate model building from model execution.

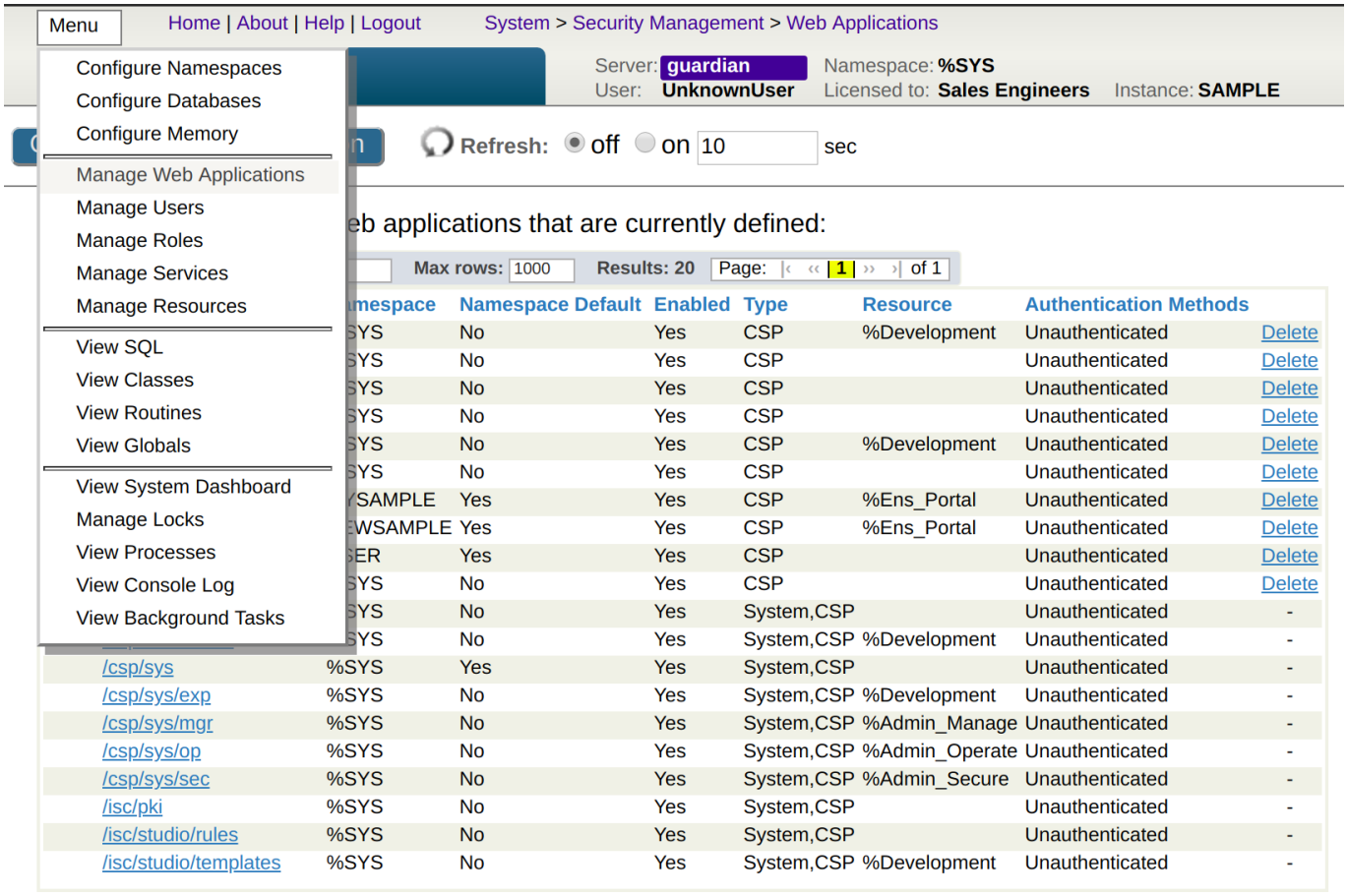
In the output, you will see a path to the PMML model.

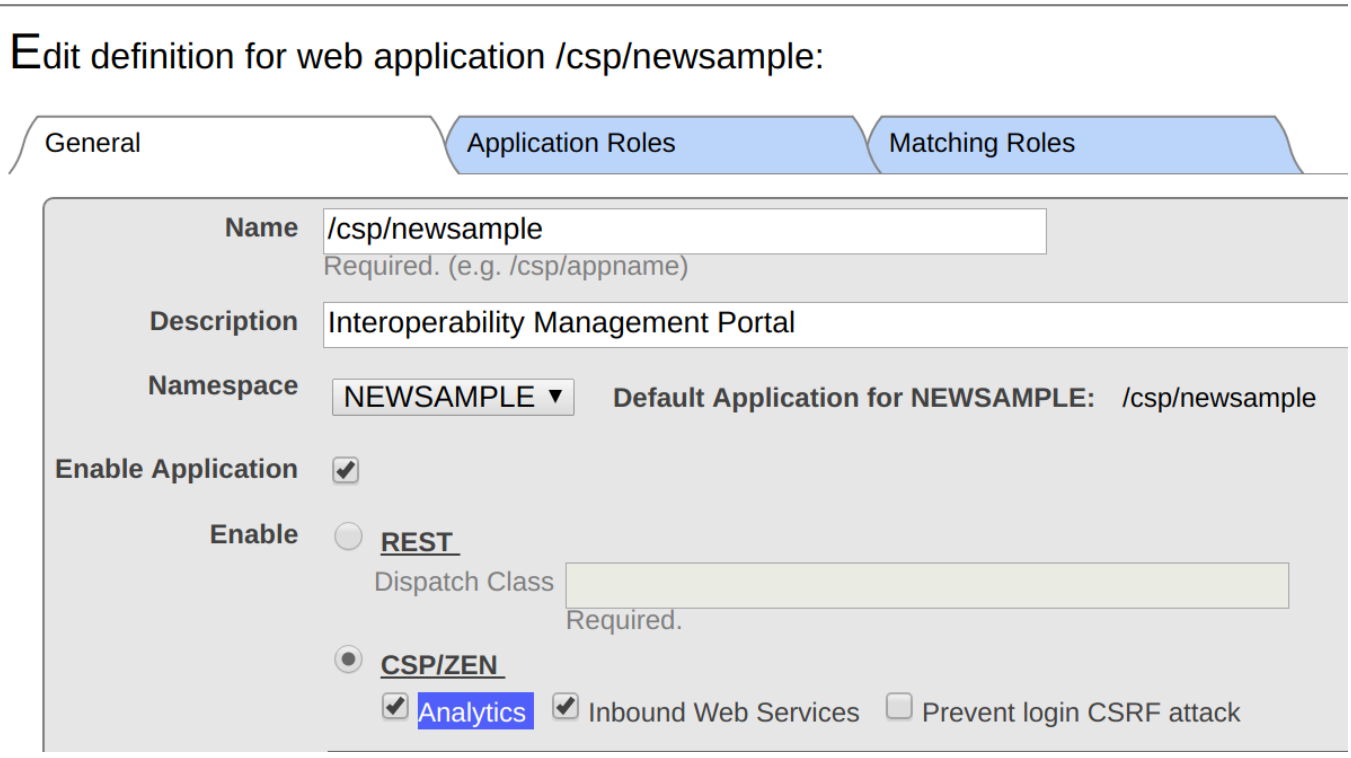
```
pmmlBuilder = PMMLBuilder(sc, dataFrame, modelKMeans)
pmmlBuilder.buildFile("KMeans.pmml") # create pmml model
```

```
'/home/guardian/Desktop/zeppelin-0.8.0-bin-all/bin/KMeans.pmml'
```

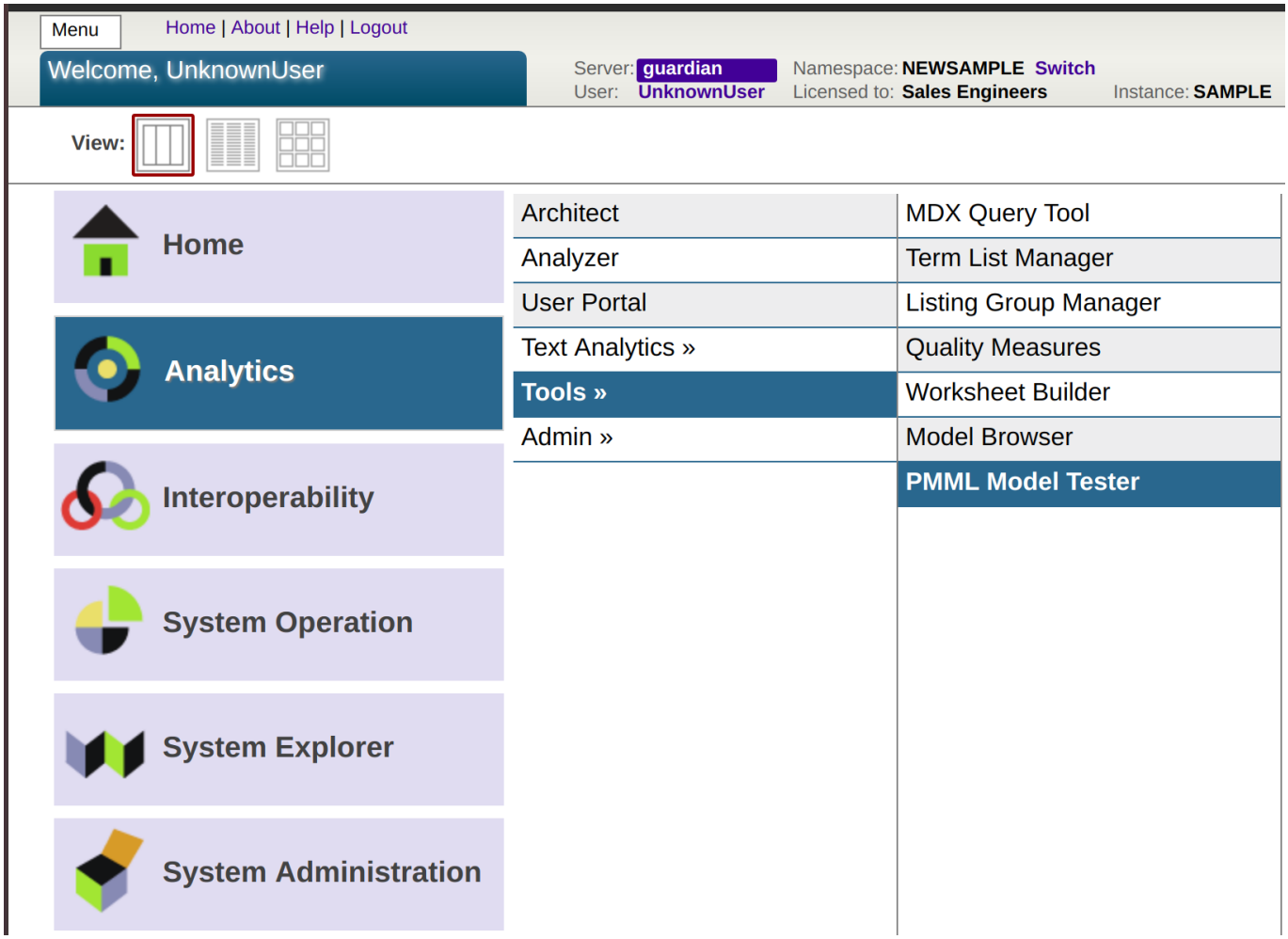
Upload and test the PMML model

Open **IRIS manager** -> **Menu** -> **Manage Web Applications** -> click on your **namespace** -> enable **Analytics** -> **Save**.

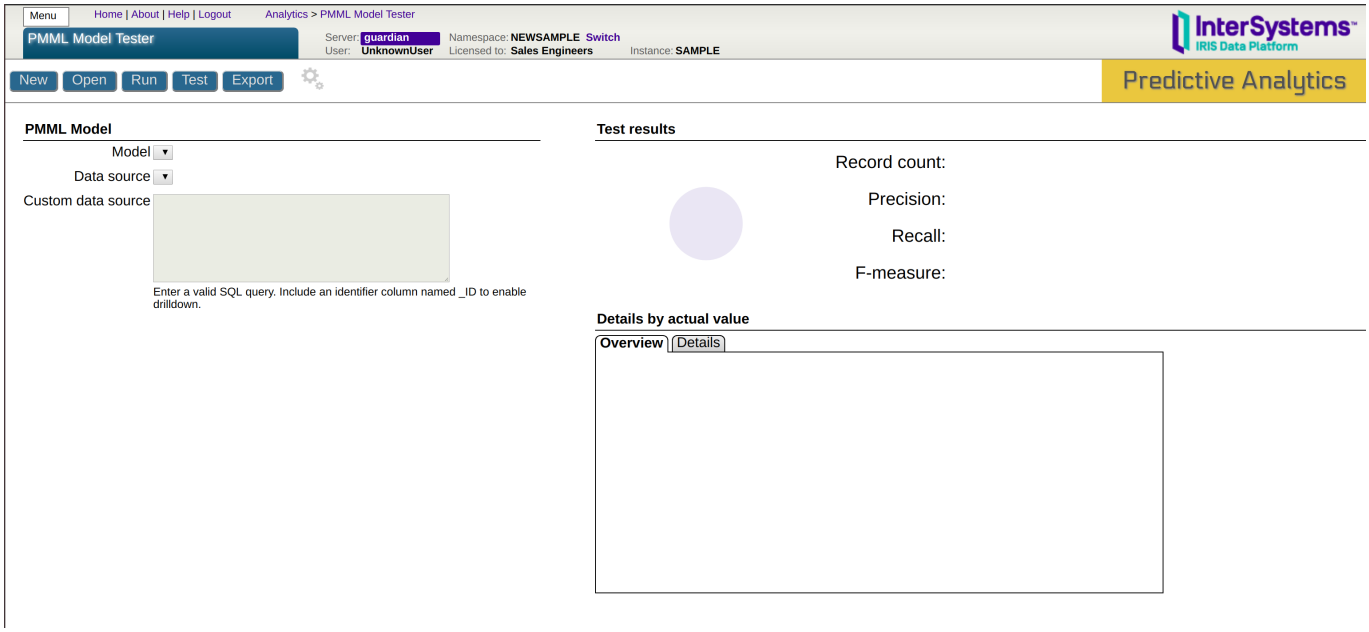




Now, go to **Analytics** -> **Tools** -> **PMML Model Tester**



You should see something like the image below:

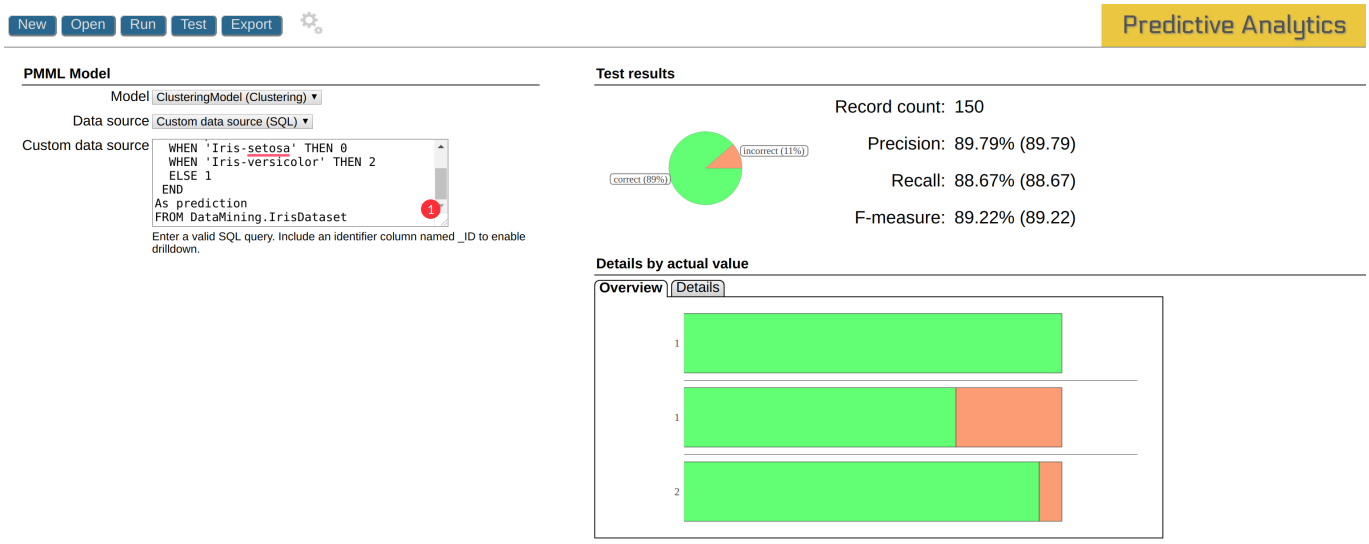


Click on **New** -> write a class name, upload PMML file (the path was in the output), and click on **Import** . Paste the following SQL querie in **Custom data source** :

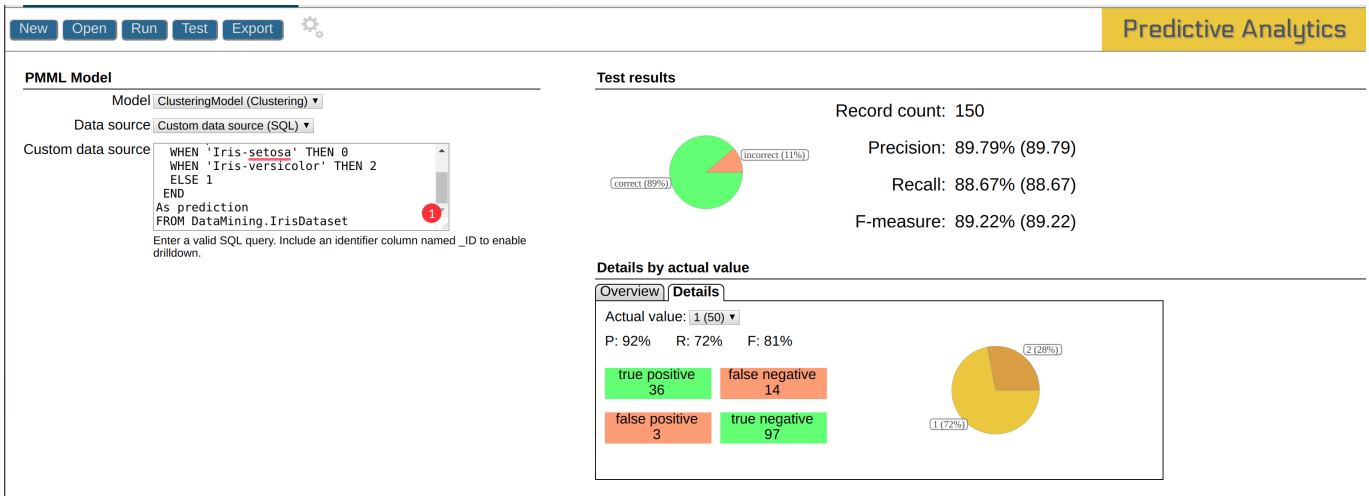
```
SELECT PetalLength, PetalWidth, SepalLength, SepalWidth, Species,
 CASE Species
  WHEN 'Iris-setosa' THEN 0
  WHEN 'Iris-versicolor' THEN 2
  ELSE 1
 END
As prediction
FROM DataMining.IrisDataset
```

We use CASE here because KMeans clustering returns clusters as numbers (0, 1, 2) and if we do not replace species to numbers it will count it incorrectly. Please comment if you know how can I replace cluster number with a species name.

My result is below:



There you can look at detailed analytics:



If you want to know better what is **true positive**, **false negative**, **etc**, read Precision and recall.

Conclusion

We have found out that PMML Model Tester is very useful tool to test your model against data. It provides detailed analytics, graphs, and SQL executor. So, you can test your model without any extended tool.

Links

Previous article

PySpark2PMML

JPMML

ML Pipelines

Apache Spark documentation

#Artificial Intelligence (AI) #Analytics #API #Beginner #Best Practices #Big Data #Machine Learning (ML) #Python #InterSystems IRIS