

DataSandBox - MinTIC



El futuro
es de todos

Unidad para la atención
y reparación integral
a las víctimas

Identificación de posibles casos de fraude en el RUV



	El futuro es de todos	Unidad para la atención y reparación integral a las víctimas		
			Versión	1.0

CONTENIDO

1	OBJETIVO	3
2	PROBLEMÁTICA	3
3	IMPLEMENTACIÓN MACHINE LEARNING CASOS TIPIFICADOS COMO POSIBLES DENUNCIAS ...	4
3.1	3.1 CONTEXTO DE LA SITUACIÓN	4
3.2	CREAR EL CONJUNTO DE DATOS	4
3.3	ALGORITMO SELECCIONADO	5
3.4	EXPERIMENTOS (IMPLEMENTACIÓN DEL MODELO).....	6
3.5	RENDIMIENTO DEL MODELO	8
3.6	PRUEBA DEL MODELO	8


	El futuro es de todos	Unidad para la atención y reparación integral a las víctimas		
			Versión	1.0

1 OBJETIVO

Ejercicio de clúster a partir de los fraudes que ya fueron identificados en el Registro Único de Víctimas (RUV) y que ya fueron denunciados ante la fiscalía por la Oficina Asesora Jurídica (OAJ) de la Unidad. Comportamientos que son identificados como fraude. Buscar en la base de datos (identificación de patrones en el RUV) Aprendizaje de máquina para que identifique dentro del RUV posibles casos de fraude a partir de las características previamente identificadas. Se parte de 3000 registros para que el modelo “Aprenda” y luego correrlo sobre los 9.000.000 de registros de víctimas.

2 PROBLEMÁTICA

Esta iniciativa nace en base a la necesidad de poder identificar posibles casos de fraudes en el RUV. La entidad pretende realizar un proyecto de Big Data y analítica que permita cumplir con el objetivo propuesto, la realización de esta iniciativa demanda una cantidad de recursos tecnológicos que permitan la ejecución.

	El futuro es de todos	Unidad para la atención y reparación integral a las víctimas		
			Versión	1.0

3 IMPLEMENTACIÓN MACHINE LEARNING CASOS TIPIFICADOS COMO POSIBLES DENUNCIAS

Para la implementación del Machine Learning se tomó como insumo la base de datos de la Oficina Asesora Jurídica de la unidad para las Víctimas (OAJ), se hizo el planteamiento de un escenario en particular, el cual se describirá a continuación:

3.1 3.1 CONTEXTO DE LA SITUACIÓN

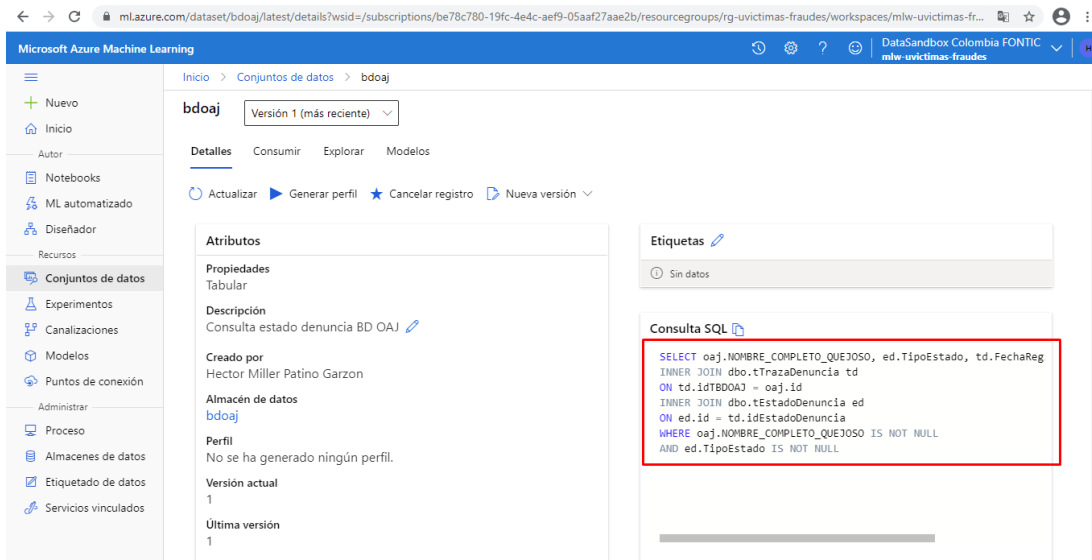
La oficina Asesora Jurídica (OAJ) de la Unidad de Víctimas, entre otras funciones tiene la recibir los casos que son tipificados como posibles fraudes, dentro de dicha gestión, esta recibir la queja, y a su vez hacer la gestión correspondiente ante la entidad competente, para este caso particular es la Fiscalía General de la Nación.

Bajo este contexto, se hizo la implementación de un Machine Learning con el objeto de evaluar el comportamiento de las denuncias, interpuestas ante la Fiscalía dependiendo la etapa del proceso. Los resultados de esta implementación se describen a continuación:

3.2 CREAR EL CONJUNTO DE DATOS

Para crear el conjunto de datos y hacer el entrenamiento del modelo, se hizo la construcción de una consulta a la base de datos implementada en el Azure DATASANDBOX de la OAJ, dicha consulta toma una muestra del nombre completo de la persona que interpone la queja, el estado de la denuncia, y la fecha de creación del registro:

 El futuro es de todos	Unidad para la atención y reparación integral a las víctimas		
	Versión		1.0



Microsoft Azure Machine Learning

Inicio > Conjuntos de datos > bdoaj

bdoaj Versión 1 (más reciente)

Detalles Consumir Explorar Modelos

Actualizar Generar perfil Cancelar registro Nueva versión

Atributos

Propiedades
Tabular

Descripción
Consulta estado denuncia BD OAJ

Creado por
Hector Miller Patino Garzon

Almacén de datos
bdoaj

Perfil
No se ha generado ningún perfil.

Versión actual
1

Última versión
1

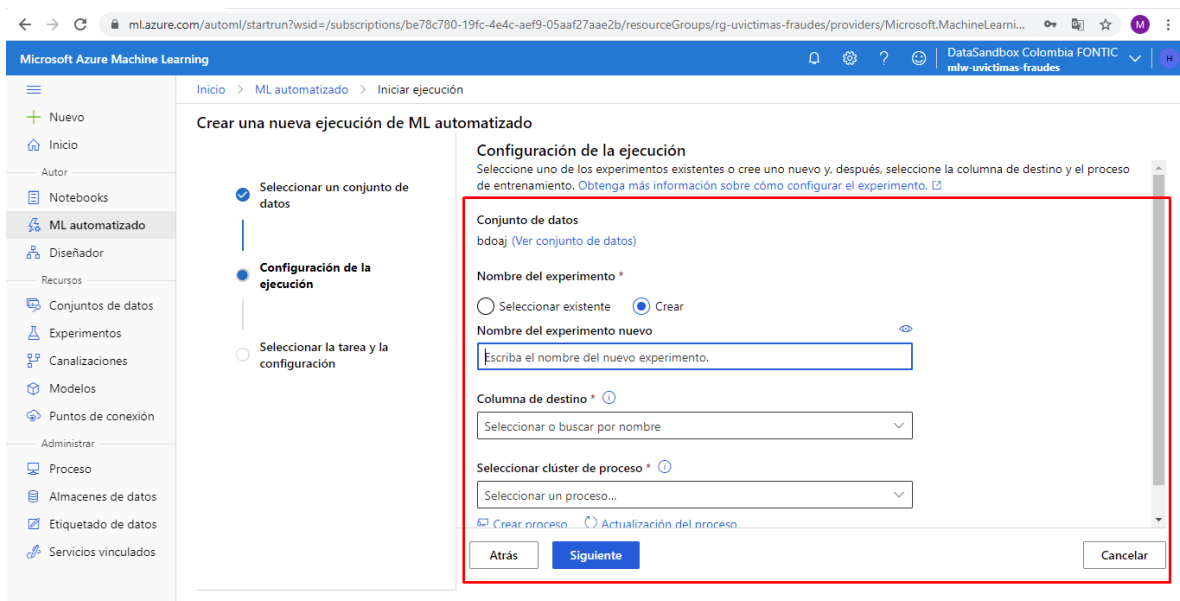
Etiquetas
Sin datos

Consultas SQL

```
SELECT oaj.NOMBRE_COMPLETO_QUEJOSO, ed.TipoEstado, td.FechaReg
INNER JOIN dbo.tTrazaDenuncia td
ON td.IDBDOAJ = oaj.id
INNER JOIN dbo.tEstadoDenuncia ed
ON ed.id = td.idEstadoDenuncia
WHERE oaj.NOMBRE_COMPLETO_QUEJOSO IS NOT NULL
AND ed.TipoEstado IS NOT NULL
```

3.3 ALGORITMO SELECCIONADO

Para hacer la selección del algoritmo, vamos a la opción ML automatizado y estando allí hago la creación del experimento, diligenciando los campos que se ven la imagen:



Microsoft Azure Machine Learning

Inicio > ML automatizado > Iniciar ejecución

Crear una nueva ejecución de ML automatizado

Selecciónar un conjunto de datos

Configuración de la ejecución

Selecciónar la tarea y la configuración

Configuración de la ejecución
Seleccione uno de los experimentos existentes o cree uno nuevo y, después, seleccione la columna de destino y el proceso de entrenamiento. [Obtenga más información sobre cómo configurar el experimento.](#)

Conjunto de datos
bdoaj (Ver conjunto de datos)

Nombre del experimento *
☐ Seleccionar existente ☒ Crear

Nombre del experimento nuevo

Columna de destino *

Seleccionar clúster de proceso *

[Crear proceso](#) [Actualización del proceso](#)

Atrás **Siguiente** Cancelar

De acuerdo con el conjunto de datos creados, el mejor algoritmo para hacer la implementación del Machine Learning es el algoritmo VotingEnsemble, el cual consiste en que cada uno de los clasificadores utilizados, realiza una predicción independiente y al final se selecciona la que ha sido clasificada por la mayoría.

3.4 EXPERIMENTOS (IMPLEMENTACIÓN DEL MODELO)

A continuación, se describe el resumen del modelo y datos adicionales de su implementación:

Inicio > Experimentos > estadoDenuncia > Ejecución 1

Ejecución 1 Completado

Actualizar Cancelar

Detalles Límites de protección de datos Modelos Resultados y registros Ejecuciones secundarias Instantánea

Propiedades

Estado: Completado

Creada: Mar 10, 2021 3:42 PM

Iniciado: Mar 10, 2021 3:43 PM

Duración: 32 m 1.188 s

Destino de proceso: [cpu-cluster](#)

Id. de ejecución: AutoML_010e7904-eb45-4c27-a9f8-9fd1621940d0

Nombre del script: --

Creado por: Hector Miller Patino Garzon

Conjuntos de datos de entrada: Nombre de entrada: training_data; id: 403c52c1-e65a-45de-93be-ed5d6c152c21

Conjuntos de datos de salida: Ninguno

Argumentos: Ninguno

Mejor resumen del modelo

Nombre del algoritmo: [VotingEnsemble](#)

Precisión: 0.61941 [Ver todas las demás métricas](#)

Muestreo: 100.00 % [Ver](#)

Modelos registrados: [AutoML010e7904e46:1](#)

Estado de la implementación: [denunciafraude](#) Correcto

Resumen de ejecución

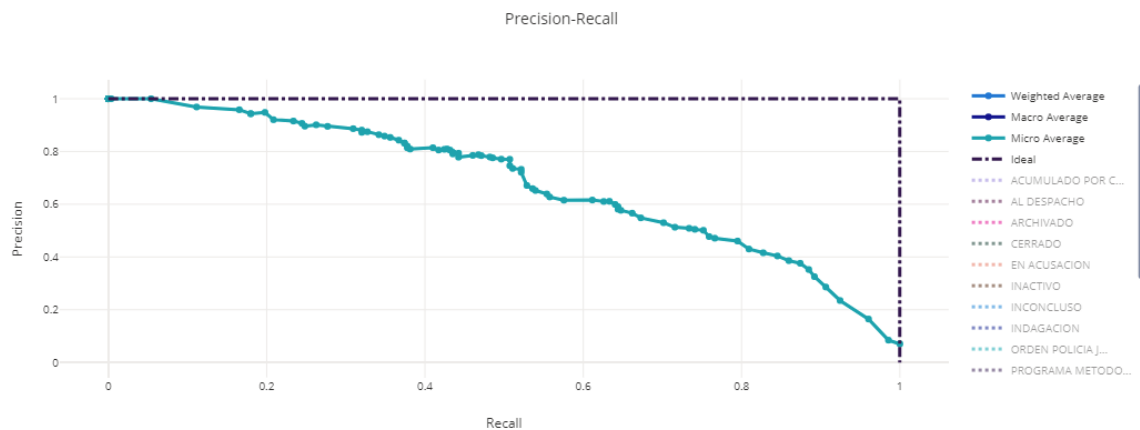
Tipo de tarea: Clasificación [Ver toda la configuración de la ejecución](#)

Métrica primaria: Precisión

Nombre del experimento: estadoDenuncia

Descripción [Ver](#)

Las graficas que se muestran a continuación, indican la interpretación estadística para la precisión del modelo:





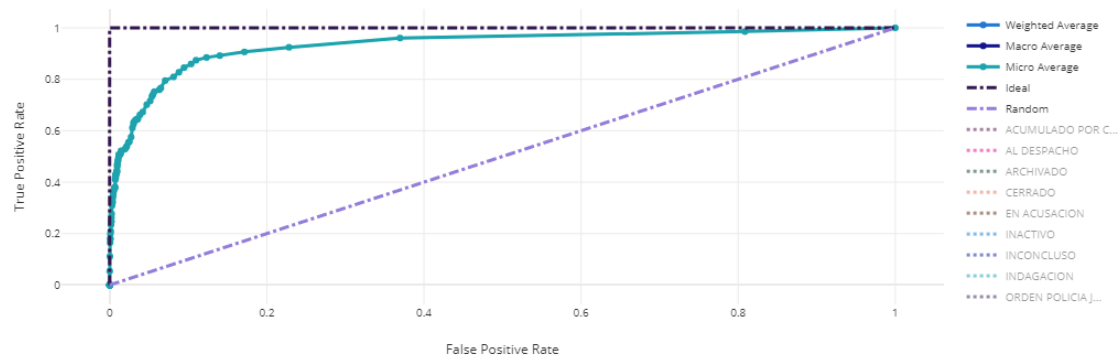
El futuro
es de todos

Unidad para la atención
y reparación integral
a las víctimas

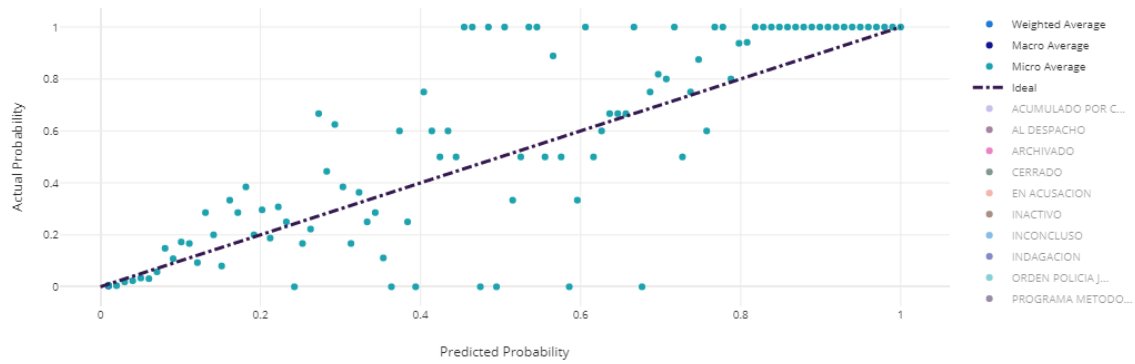
Versión

1.0

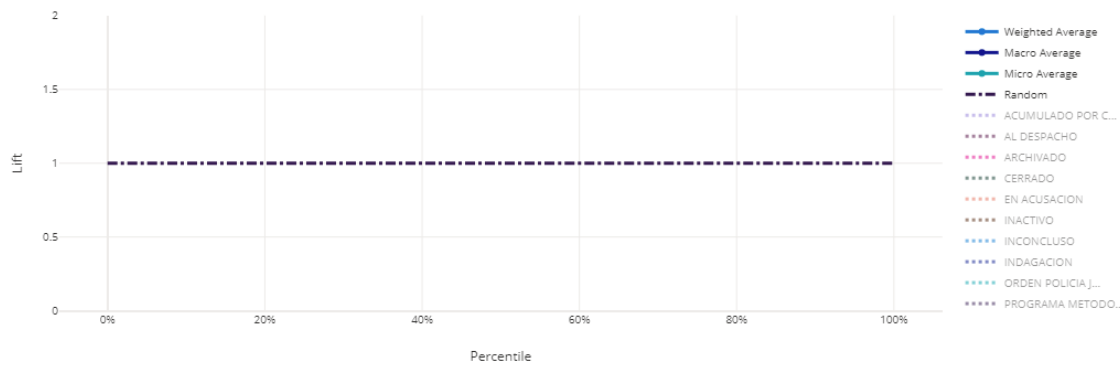
ROC

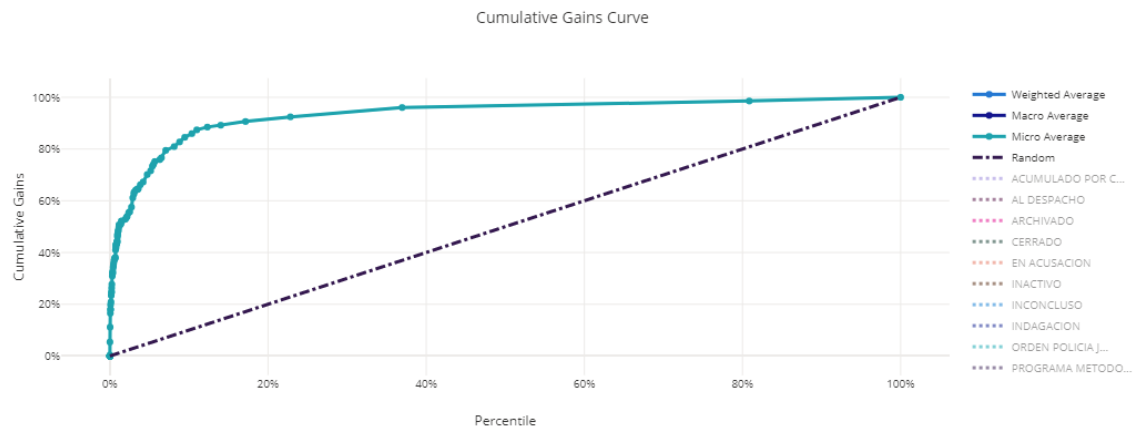


Calibration Curve



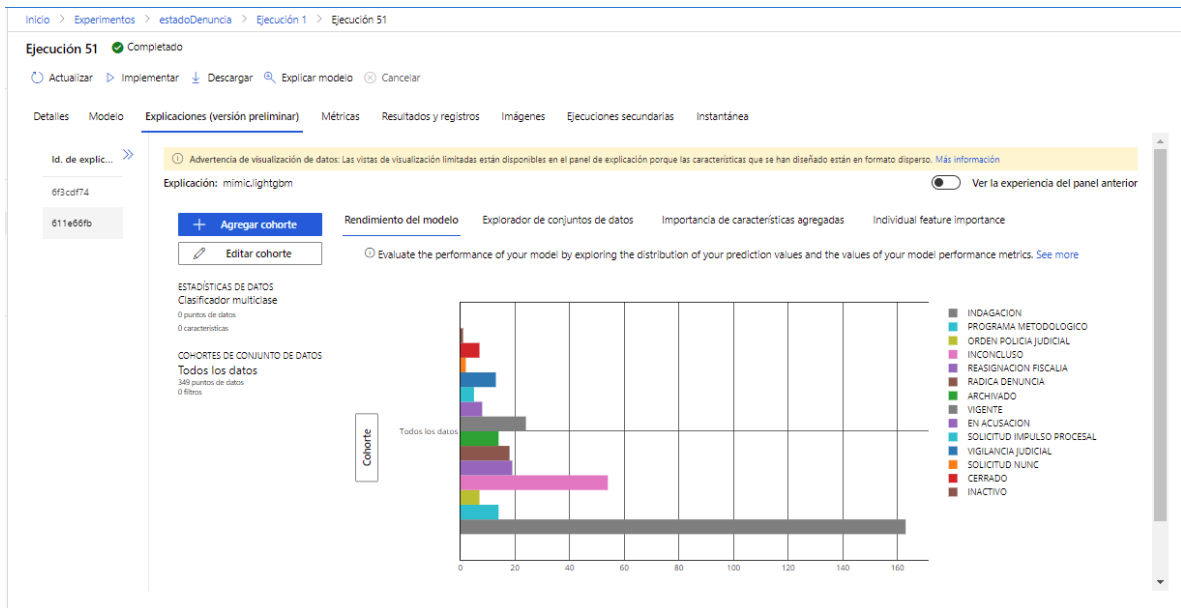
Lift Curve





3.5 RENDIMIENTO DEL MODELO

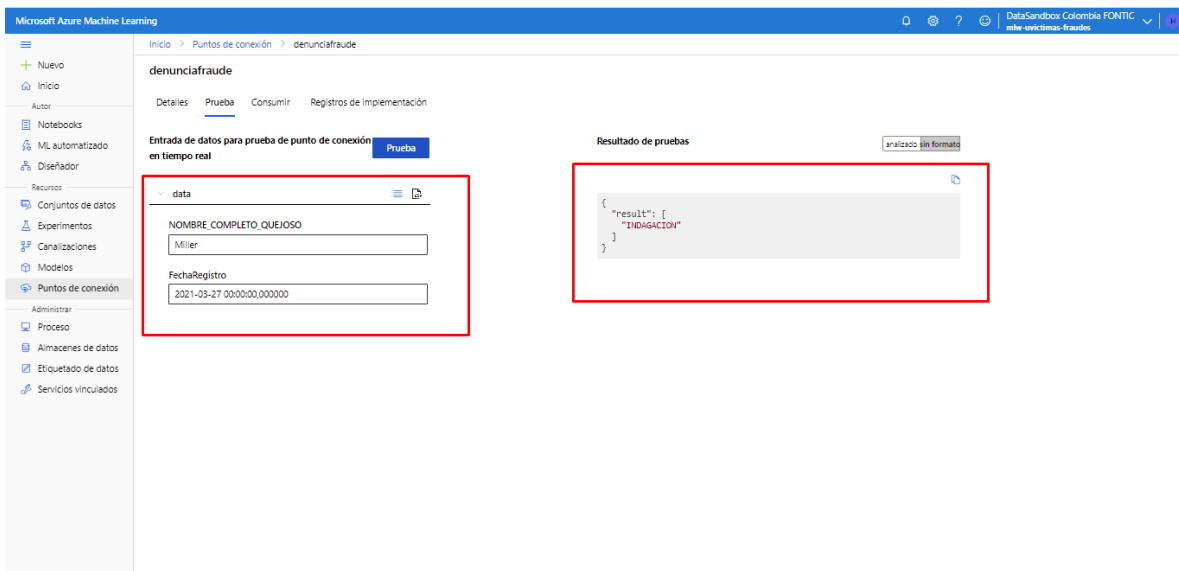
En la imagen que se muestra a continuación, se puede visualizar los datos con los cuales se realizó el entrenamiento del modelo:



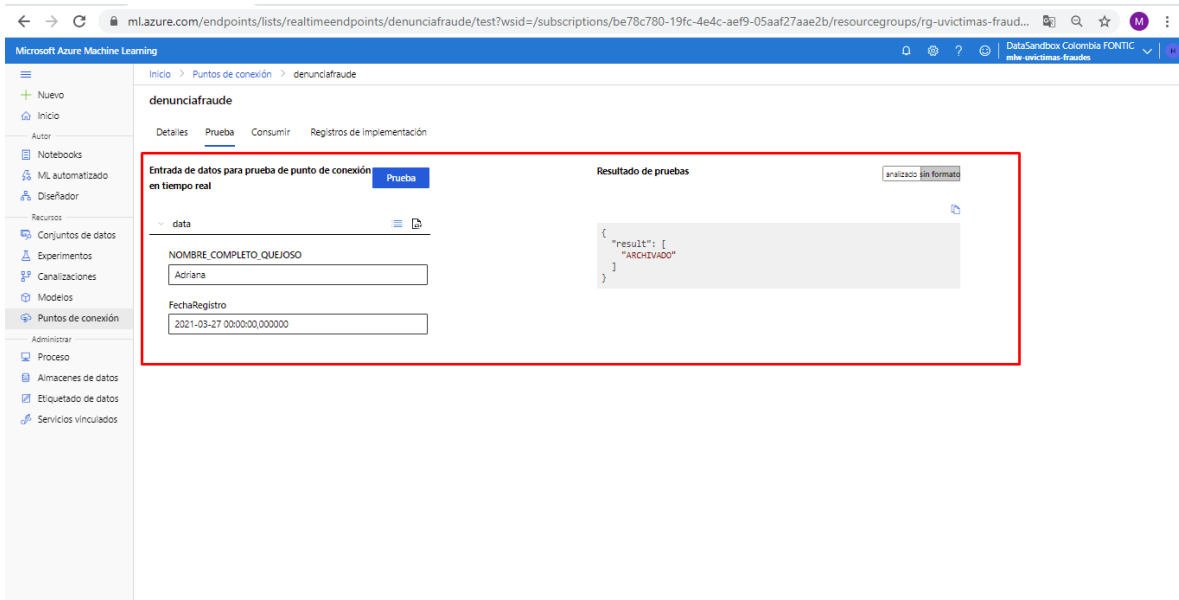
3.6 PRUEBA DEL MODELO

Con el entrenamiento del modelo, pasamos a ingresar datos de prueba. Con este ejercicio el algoritmo nos indica cual es la predicción de un caso reportado como fraude:

 El futuro es de todos	Unidad para la atención y reparación integral a las víctimas			
			Versión	1.0



En resumen, el algoritmo basado en cierta cantidad de casos ingresados puede determinar cuál será la etapa en la que se encuentre durante el proceso de investigación, de acuerdo con la fecha de la radicación de la denuncia. La imagen que se muestra a continuación arroja un resultado después de haber ingresado cierta cantidad de datos:



Para el consumo del modelo, se muestra cómo hacerlo por medio Notebooks haciendo uso de las librerías y código fuente de Python:



El futuro
es de todos

Unidad para la atención
y reparación integral
a las víctimas

Versión

1.0

Microsoft Azure Machine Learning

Inicio > Notebooks

Correcto: Se ha creado correctamente "estadoDenuncia3.ipynb" en "Users/hector.patino".

Notebooks

Archivos Muestras

ML-Instancia-Miller - Kernel de Jupyter interactivo

Python 3.6 - AzureML

```
6 # allowSelfSignedHttps(allowed):
7 # bypass the server certificate verification on client side
8 if allowed and not os.environ.get('PYTHONHTTPSVERIFY', '') and getattr(ssl, "_create_unverified_context", None):
9     ssl._create_default_https_context = ssl._create_unverified_context
10
11 allowSelfSignedHttps(True) # this line is needed if you use self-signed certificate in your scoring service.
12
13 data = {
14     "data":
15     [
16         {
17             "NOMBRE_COMPLETO_QUEJOSO": "Juan",
18             "FechaRegistro": "2021-03-27 00:00:00,000000",
19         },
20     ],
21 }
22
23 body = str.encode(json.dumps(data))
24 headers = {'Content-Type': 'application/json', 'Authorization': ('Bearer ' + api_key)}
25 req = urllib.request.Request(url, body, headers)
26 try:
27     response = urllib.request.urlopen(req)
28     result = response.read()
29     print(result)
30 except urllib.error.HTTPError as error:
31     print("The request failed with status code: " + str(error.code))
32
33 # Print the headers - they include the request ID and the timestamp, which are useful for debugging the failure
34 print(error.info())
35 print(json.loads(error.read().decode("utf-8", 'ignore')))
```

b""{"\\result\\": [{"INDAGACION\\":}]""