PROYECTO 3 (Spark GraphFrames)

Presentado por:
Juan Camilo Hernandez Saavedra
Julian Paredes Conde
Augustin Valencia

PROCESAMIENTO DE GRANDES VOLÚMENES DE DATOS

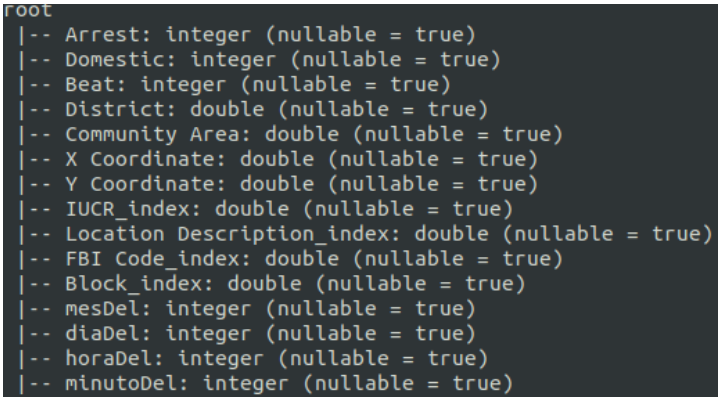
CALI, OCTUBRE DE 2021

- **Contexto:**

  Este conjunto de datos refleja los incidentes de delitos denunciados (con la excepción de los asesinatos) que ocurrieron en la ciudad de Chicago desde 2001 hasta el 2017. Los datos se extraen del sistema CLEAR (Análisis e informes de aplicación de la ley ciudadana) del Departamento de Policía de Chicago. Para proteger la privacidad de las víctimas de delitos, las direcciones se muestran solo a nivel de bloque y no se identifican ubicaciones específicas. El dataset se encuentra dividido en 4 archivos .csv donde cada archivo contiene información en un rango de 3 años, a excepción del último archivo que contiene información en un rango de 4 años (2012-2017). Para nuestro caso de estudio se trabajará con el último archivo pues consideramos que este cuenta con la cantidad de información que creemos justa para el estudio.

- **Estructura Inicial del Dataset:**

  Para esta entrega se va a utilizar el dataset ya limpio que se obtuvo de la entrega pasada. El dataset cuenta con 15 atributos de los cuales podemos encontrar:

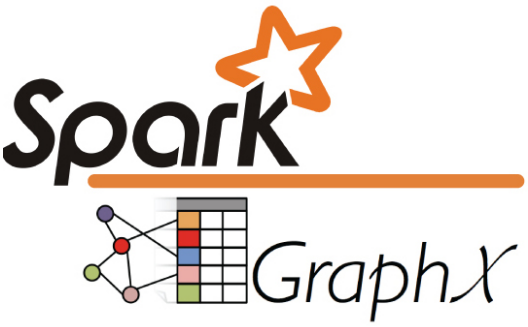


- **¿Qué se va a predecir?:**

  Continuando con la primera y segunda entrega, se busca predecir si dado un delito se va a realizar un arresto o no. Todo esto con base en la información relacionada con la demografía de los delitos cometidos. Esto es un problema de clasificación binaria, pues las salidas de los modelos estarán representados de manera binaria, donde 1 significa que se realizará un arresto y 0 significa que no se realizará un arresto.

A diferencia de la primera y segunda entrega, se desea agregar nuevos atributos al dataset. Estos atributos se obtendrán por medio de análisis de grafos utilizando las herramientas que provee Spark para este fin, e información que consideramos importante que se encuentra en otros datasets.

- **Tecnologías y Arquitectura:**



  - Para la creación del grafo en cuestión, se va a usar la herramienta Graph Frames, herramienta derivada de GraphX con implementación para los lenguajes de Python y Java que permite manipular y crear grafos por medio de data Frames.

  - Para poder visualizar el grafo, se va a utilizar el framework NetworkX, ya que la librería de Graph Frames no posee implementación para representar gráficamente los grafos.

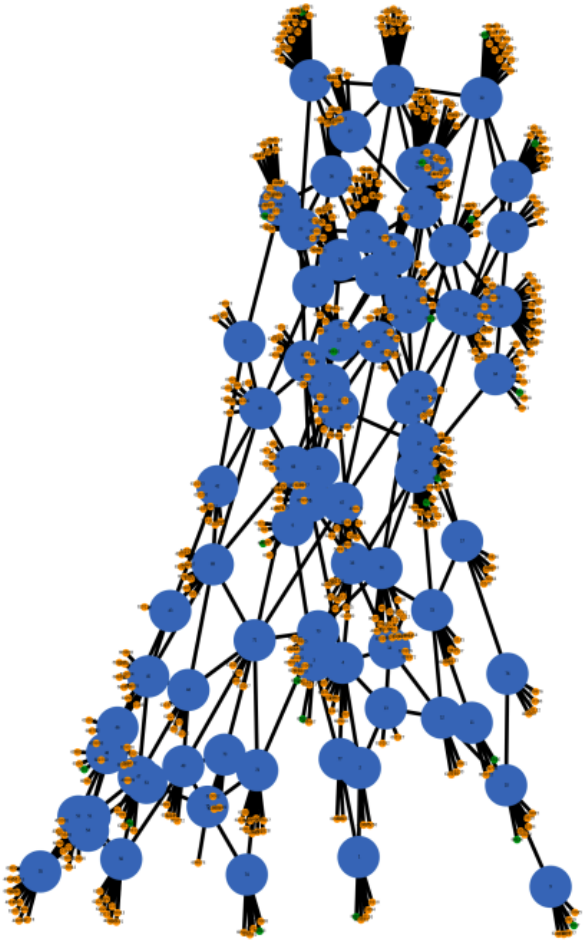
  - Para poblar el grafo, se va a utilizar información relevante proveniente de otros datasets. Estos datasets en cuestión son, las estaciones de policía de Chicago donde se almacena información acerca de las estaciones de policía de esta ciudad en cuestión, y las escuelas públicas de la ciudad de Chicago.

  - De estos dos datasets, solo se utilizará la información referente al área comunitaria a la cual estas edificaciones pertenecen. Como extra, a la información de las estaciones de policía, manualmente se le deberá asignar un código ID con el fin de ser identificadas dentro del grafo.

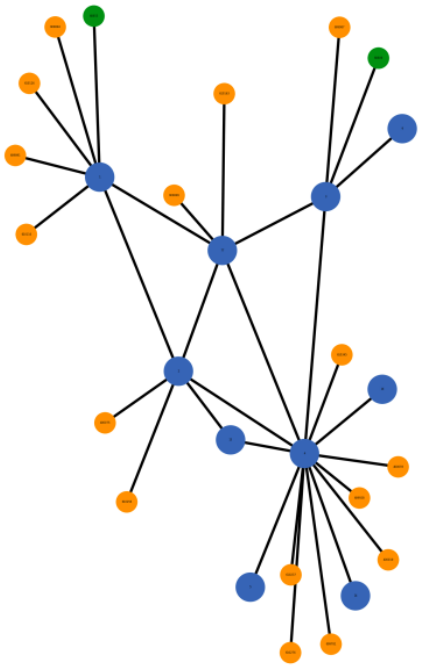
- **Estructura y Creación del Grafo:**

  Para la creación del grafo se tomó en cuenta la información de los dos datasets nombrados anteriormente. El grafo posee tres tipos de nodos, las áreas comunitarias de la ciudad de Chicago, las estaciones de policía de la ciudad de Chicago y las escuelas públicas de la ciudad de Chicago.

  una área comunitaria se conecta con otra área comunitaria si estas colindan, por otro lado un área comunitaria se conecta con una estación de policía o escuela pública si estas se encuentran dentro del área comunitaria en cuestión. de tal manera, el grafo obtenido en esta caso es el siguiente:



Grafo de manera general, los nodos azules hacen referencia a las áreas comunitarias, los nodos naranjas hacen referencia a las escuelas públicas y los nodos verdes hacen referencia a las estaciones de policía.



Acercamiento de una parte del grafo.

- **Análisis del Grafo:**

  Posteriormente a la creación del grafo mostrado anteriormente, se decidió analizar ciertos atributos de las áreas comunitarias con el fin de agregar ciertos nuevos atributos a la demografía del delito.

  - En primera parte, se decidió observar que áreas comunitarias poseen estación de policía, pues dentro de la ciudad de Chicago en términos policiales la ciudad se encuentra dividida en distritos, a su vez los distritos se encuentran divididos en varias áreas comunitarias. Cada distrito posee una única estación de policía, es decir no todas las áreas comunitarias poseen estaciones de policía.

```
def obtencionEstaciones(g, aComunitarias, crimenesDf):
    pru = g.find("(AreaComunitaria)-[]->(EstacionPolicia)")
    pru = pru.select(pru.AreaComunitaria.id.alias("areaC"), pru.EstacionPolicia.id.alias("Estacion")) \
    where((pru.AreaComunitaria.tipoNodo == "AreaComunitaria") & (pru.EstacionPolicia.tipoNodo == "EstacionPolicia"))
    tempEstaciones = aComunitarias.join(pru, aComunitarias["Community Area"] == pru["areaC"], "leftouter")
    tempEstaciones = tempEstaciones.select(tempEstaciones["Community Area"].alias("Area"), "Estacion")
    exprT = expr(
            """
            IF(Estacion IS NULL, 0, 1)
            """
            )
    tempEstaciones = tempEstaciones.withColumn("Estacion", exprT)
    crimenesDf = crimenesDf.join(tempEstaciones, crimenesDf["Community Area"] == tempEstaciones["Area"], "inner")
    crimenesDf = crimenesDf.drop("Area")
    return crimenesDf
```

- Como segundo atributo, se decidió analizar la cantidad de colegios públicos que se encuentran dentro de una área comunitaria, todo esto con el propósito de dar una idea acerca de que tan familiar puede llegar a ser un área comunitaria. Esto es solo un acercamiento, pues con los datasets correctos se puede hacer un mayor análisis de las áreas comunitarias y hacer un grafo mucho más completo.

```
def obtencionColegios(g, crimenesDf):
    pru = g.find("(AreaComunitaria)-[]->(Colegio)")
    pru = pru.select(pru.AreaComunitaria.id.alias("areaC"), pru.Colegio.id.alias("Colegio")) \
    where((pru.AreaComunitaria.tipoNodo == "AreaComunitaria") & (pru.Colegio.tipoNodo == "Colegio"))
    tempColegios = pru.groupBy("areaC").count()
    tempColegios = tempColegios.select(tempColegios["areaC"], tempColegios["count"].alias("cant Colegios"))
    crimenesDf = crimenesDf.join(tempColegios, crimenesDf["Community Area"] == tempColegios["areaC"], "inner")
    crimenesDf = crimenesDf.drop("areaC")
    return crimenesDf
```

Finalmente, todos estos atributos fueron agregados al dataset original con el fin de poder realizar las predicciones de machine learning correspondientes. la estructura final del dataset quedó de la siguiente manera:

```
root
 |-- Arrest: integer (nullable = true)
 |-- Domestic: integer (nullable = true)
 |-- Beat: integer (nullable = true)
 |-- District: double (nullable = true)
 |-- Community Area: double (nullable = true)
 |-- X Coordinate: double (nullable = true)
 |-- Y Coordinate: double (nullable = true)
 |-- IUCR_index: double (nullable = true)
 |-- Location Description_index: double (nullable = true)
 |-- FBI Code_index: double (nullable = true)
 |-- Block_index: double (nullable = true)
 |-- mesDel: integer (nullable = true)
 |-- diaDel: integer (nullable = true)
 |-- horaDel: integer (nullable = true)
 |-- minutoDel: integer (nullable = true)
 |-- Estacion: integer (nullable = false)
 |-- cant Colegios: long (nullable = false)
```

- **Implementación Algoritmos Machine Learning (MLlib):**

	Precision	Area bajo el ROC
Regresion Logistica	0,74	0,72
Arboles de Decision	0,78	0,54
Random Forest	0,83	0,84

Al implementar los modelos de machine learning sobre el dataset obtenido, luego de agregar los features, podemos observar que los modelos con la mejor precisión son árboles de decisión y random forest, teniendo en cuenta que este estimador de precisión fue calculado a través de la tabla de predicciones por medio de métodos de pyspark sql. Sin embargo, podemos observar que el algoritmo de árboles de decisión presenta una baja precisión en el estimador del área bajo el ROC. Probablemente eso se deba a la gran cantidad de rangos que existen entre los atributos del dataset, una posible solución para mejorar la precisión de este algoritmo puede ser normalizar ciertos atributos.

Por último podemos observar que los algoritmos que mejor precisión poseen son la regresión logística y random forest siendo este último ,el que mejor se acomoda al dataset. Probablemente esto se deba a que ciertos algoritmos basados en el método de ensamble tienden a adaptarse mejor a un dataset que posee grandes rangos entre sus atributos.

- **Cibergrafía:**
    - **https://spark.apache.org/docs/latest/api/python/**
    - **https://graphframes.github.io/graphframes/docs/_site/index.html**
    - **https://graphframes.github.io/graphframes/docs/_site/user-guide.html**
    - **http://www.thechicago77.com/chicago-neighborhoods/**
    - **https://towardsdatascience.com/customizing-networkx-graphs-f80b4e69bedf**
    - **https://www.kaggle.com/chicago/chicago-public-schools-data?select=chicago-public-schools-elementary-school-progress-report-2013-2014.csv**
    - **https://www.kaggle.com/chicago/chicago-police-stations/version/4**