Spark ML

Spark ML

01 Introducción
02 Transformación de variables
03 Selección de variables
04 Estandarización
05 Clasificación
06 Regresión
07 Clustering
08 Pipelines
09 Evaluación de modelos
10 Algoritmos no paralelizables

01

Introducción

ml vs mllib

Principales diferencias

Tanto pyspark.ml como pyspark.mllib son paquetes que incluyen algoritmos y funcionalidades de Machine Learning. Sus diferencias son las siguientes:

ml	mllib
- versión nueva - construido sobre DataFrames - permite construir Pipelines - menos funcionalidades	- versión antigua - construido sobre RDDs - no permite construir Pipelines - más funcionalidades

Pipeline

from pyspark.ml import Pipeline

- Una Pipeline o canalización es un flujo que combina varias transformaciones y estimadores.
- Define todas las fases y el orden de un proceso de aprendizaje automático.
- la salida de una fase es la entrada de la siguiente.
- Recibe como entrada un DataFrame y devuelve un modelo.

transformers vs estimators

Principales diferencias

Tanto un *transformer* como un *estimator* puede ser un *stage* de un Pipeline. Sus diferencias son las siguientes:

transformer	estimator
- abstracción que incluye transformadores de atributos y modelos aprendidos. - implementa el método transform() - el *input* de un *transformer* es un DataFrame y su *output* también es un DataFrame	- abstracción de un algoritmo que se entrena con datos. - implementa el método fit() - el *input* de un *estimator* es un DataFrame y su *output* es un modelo.

NOTEBOOK

Vista general de la resolución de un proyecto analítico con Machine Learning

Spark ML

Machine Learning End to End

02

Transformación de variables

VectorAssembler

from pyspark.ml.feature import VectorAssembler

VectorAssembler es un *transformer* que combina una lista dada de columnas en un único vector columna. Es necesario utilizar un VectorAssembler para crear un solo atributo de tipo vector y con él entrenar modelos de Machine Learning.

VectorAssembler permite los siguientes tipos de columna:

- cualquier tipo numérico
- booleano
- vector

StringIndexer

from pyspark.ml.feature import StringIndexer

StringIndexer codifica una columna de tipo *string* (categórica) en una columna de índices. Estos índices se encuentran en el intervalo [0, número categorías) y se ordenan por frecuencia de aparición. La categoría de mayor frecuencia toma el valor 0.

Advertencia:

Se puede dar el caso que una categoría sea desconocida para el modelo puesto que aparece en el dataset de test pero no en el de train. Es importante intentar prevenir estas situaciones.

OneHotEncoder

from pyspark.ml.feature import OneHotEncoder

OneHotEncoder mapea una columna de índices a una de un vector que toma valores binarios con un solo valor uno. Esta codificación permite incorporar el estudio de variables categóricas en un modelo. Sirve para la creación de variables *dummy*.

Nota:

- Si hay *n* categorías, por defecto OneHotEncoder tomará *n-1*: interpreta tomando las *n* serían linealmente dependientes. Debemos entender los datos porque no siempre es así conceptualmente.

- Para crear una variable *dummy* de una variable categórica, antes debemos hacer un StringIndexer de esa columna.

CountVectorizer

from pyspark.ml.feature import CountVectorizer

CountVectorizer ayuda a convertir una colección de documentos de texto a vectores con conteos de los *tokens.* Por otro lado, cuando no se conoce el diccionario, CountVectorizer extrae el vocabulario presente en los documentos.

NOTEBOOK

Ejemplos de las principales transformación características de Spark ML

Spark ML

Transformación Variables

03

Selección de variables

NOTEBOOK

Ejemplos de las principales técnicas de selección de variables

Spark ML

Selección Variables

04

Estandarización

NOTEBOOK

Ejemplos de las principales técnicas de estandarización de variables

Spark ML

Estandarización

05

Clasificación

NOTEBOOK

Ejemplos de los principales algoritmos de clasificación

Spark ML

Clasificación

06

Regresión

NOTEBOOK

Ejemplos de los principales algoritmos de regresión

Spark ML

Regresión

07

Clustering

NOTEBOOK

Ejemplos de los principales algoritmos de clustering

Spark ML

Clustering

08

Pipelines

NOTEBOOK

Ejemplos del funcionamiento de los Pipelines en un proyecto analítico

Spark ML

Pipelines

09

Evaluación de modelos

NOTEBOOK

Ejemplos de las principales métricas para evaluar modelos

Spark ML

Evaluación de Modelos

10

Algoritmos no paralelizables

Métodos no paralelizables

Debido al comportamiento de ciertos algoritmos, en algunas ocasiones es necesario volcar todos los datos en el mismo nodo con tal de analizarlos a la vez.

En tal caso, se habla de métodos no paralelizables. Aunque ciertos pasos se puedan paralelizar, al tener que juntarlos todos en un nodo en algún instante, provoca que Spark no sea el lenguaje idóneo para resolverlo.

Así pues, no todos los algoritmos se encuentran disponibles en Spark por este fenómeno.

Thanks