UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Simulación de datos de sensores industriales

TRABAJO DE FIN DE MÁSTER

Máster en Big Data Analytics

Autor: Pedro Henrique Mano Figueiredo Fernandes
Tutor: Francisco Sánchez Cid

Curso 2015-2016

Abstract

Los entornos industriales hacen uso de sensores para obtener mediciones de varios factores en su cadena de producción. En las fases iniciales, la cantidad de datos generados es pequeña, por lo que no es significativa para poder aplicar técnicas de Big Data. El proyecto descrito en este documento busca simular nuevos datos a través de los pocos datos generados por sensores industriales. Las métricas de sensores no tienen, en general, buena calidad - la frecuencia de muestreo no es constante, hay ruido, existen períodos sin mediciones y redundancias en las muestras. Estos problemas se extienden a todo el contexto de proyectos IoT (*Internet of Things*), donde los sensores representan la principal fuente de datos y donde se suma la infraestructura de conexión como factor de complejidad. Así que la limpieza y normalización de los datos es fundamental para poder aplicar técnicas matemáticas y tener resultados precisos.

Palabras clave: Machine learning, Big Data, PCA

Estado del Arte

TODO

CAPÍTULO 1
Introducción

Este proyecto tiene como fuente de datos uno o varios sensores de un mecanizado industrial de injección de plástico. Los sensores efectuan mediciones de varios factores con una regularidad temporal, generando así muestras de datos en determinados intervalos de tiempo. Cuando se trata de sensores, es normal que los datos no tengan la calidad necesaria para aplicar técnicas matemáticas. Hay que tener en cuenta que la frecuencia de las mediciones no es necesariamente constante, la existencia de interrupciones, ruido y redundancias. Además, en el ambito de IoT (*Internet of Things*), se añaden otros factores, como la conectividad de los aparatos.

Las técnicas de Big Data, en particular *Machine Learning*, necesitan gran cantidad de datos para poder inducir modelos matemáticos que los expliquen. Cuanto más datos mejor, para un aprendizaje más robusto y fiable. Sin embargo, al principio de los proyectos, la cantidad de datos recolectados suele ser pequeña y insuficiente para poder extraer conocimiento significativo de los mismos. El objetivo de este proyecto es aprender y simular el comportamiento de uno o varios sensores a través de una cantidad reducida de datos recolectados en los mismos.

Una vez preparado el *dataset*, es necesario aprender un modelo matemático que lo describa. Las técnicas usadas para ese efecto son del ambito de *Machine Learning*. En particular, en este caso se tratan de técnicas de aprendizaje no supervisado (*unsupervised learning*), pues el aprendizaje no visa predecir el valor de otras variables (etiquetas).

A los datos generados, se tendrán que aplicar técnicas de validación, para así evaluar formalmente el método propuesto.

Con un volumen de datos muy grande, las herramientas convencionales no tienen capacidad de respuesta. El uso de herramientas Big Data surge como solución para ese problema.

1.1 Motivación

Los datos recolectados en sensores industriales tienen poco volumen al principio de la implantación. Con estos datos, es posible aprender un modelo para predecir el comportamiento futuro. Sin embargo, antes hay que tener en cuenta que la calidad de datos no siempre es la adecuada para aplicar técnicas matemáticas. Así que es muy importante conocer el *dataset*, limpiarlo y normalizarlo. Además hay que lidiar con frecuencias de mediciones inconstantes, con el ruido y repetición de mediciones. Para estos problemas, las técnicas de *Machine Learning* ofrecen buenas soluciones.

1.2 Objectivos

El objetivo de este proyecto es aprender y simular datos de sensores industriales. Los nuevos datos, de mucho mayor volumen, permitirán aplicar técnicas de Big Data. Como los datos iniciales son provenientes de sensores, la calidad no está asegurada, por lo que se tendrá que emplear técnicas de limpieza y normalización. El aprendizaje del *dataset* pasa por aplicación de *Machine Learning*, para inducir un modelo matemático y poder inferir datos futuros. Para comprobar la validez de la solución propuesta, se tiene que aplicar métodos estadísticos. Potencialmente, las herramientas convencionales no tengan capacidad para un volumen muy grande de datos- en ese caso se exige el uso de herramientas más poderosas (Big Data).

Los pasos descritos son muy comunes en análisis de datos y en particular en el *pipeline* de Big Data.

1.3 Estructura del documento

Este documento se estructura de la siguiente forma: un análisis sobre el *dataset* y respectivas transformaciones para adecuar los datos; el desarrollo del problema, con las soluciones propuestas y respectiva base teórica; conclusiones del trabajo realizado y posibles mejoras; y termina con las fuentes bibiográficas que han servido de base del estudio.

CAPÍTULO 2
Dataset

El dataset proporcionado es de reducida dimensión, tiene tan solo 5MB, por lo que la tarea de ingestión no es intensiva. Sin embargo, hay que tratar los datos en bruto antes de empezar a usarlos. El fichero de datos es de texto, así que hay que hacer determinadas conversiones para poder usar tipos más específicos, como sean fechas y números. El primer problema tiene que ver con los formatos de fecha, que no son correctos para el *locale* España, lo que exige una adaptación de los algoritmos de *parsing*, como se describe en el apartado *Transformación*.

2.1 Estructura

Un breve análisis del *dataset* en un editor de texto muestra que tiene un formato de campos separados por espacios y tabulaciones. La cantidad de espacios es variable:

```
Tiempoinicio␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣ APHu␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣ APVs␣...

06-oct-2015␣21:57:03␣ 44.6␣␣␣␣␣␣␣␣␣␣␣␣␣ 69.3␣...
06-oct-2015␣21:57:12␣ 45.1␣␣␣␣␣␣␣␣␣␣␣␣␣ 69.0␣...
06-oct-2015␣21:57:21␣ 44.8␣␣␣␣␣␣␣␣␣␣␣␣␣ 69.8␣...
...
```

La primera línea contiene un *header* (cabecera), con 15 nombres:

`Tiempoinicio, APHu, APVs, ACPv, ZSx, ZUs, H7x, H1x, H2x, H6x, H3x, H4x, H5x, ACPx, Svo`.

Se puede verificar también que hay una línea vacía después del *header*. El primer campo tiene un formato de fecha/hora y los demás campos tienen formato decimal.

2.2 Transformación

Para la ingestión y transformación de los datos se han usado librerías muy útiles y con muchas funcionalidades que facilitan bastante esas tareas: en los scripts Python se ha usado el paquete *Pandas* y en R la función *read.csv2* del paquete *utils*.

Las 14 variables decimales no ofrecen problemas en la ingestión del *dataset*. Para asegurar el formato decimal, es conveniente definir el separador decimal como punto (‘.’). Con eso es suficiente para un em parsing correcto.

Las fechas son más complejas de procesar. El formato de mes da indicios de estar escrito en castellano: `oct, dic, mar, abr, may, jun`. Sin embargo, en el *locale* de España, el *standard* de abreviación de mes es con punto ('.'), por ejemplo `oct.`. Así que el *parsing* de fechas tuvo que ser ajustado. La estratégia ha sido el uso de una expresión regular para añadir el punto ('.') necesario en las abreviaciones de meses, como se puede ver en los siguientes *snippets* Python y R:

Parsing de fechas en Python:

```
def parse_date(date_string):
    locale_date_string = re.sub("(.+-)(.+)(-.+)", "\\1\\2.\\3", date_string)
    return datetime.strptime(locale_date_string, "%d-%b-%Y %H:%M:%S")
```

Parsing de fechas en R:

```
data$Tiempoinicio <- sub("(\\d+-)(\\w+)(-\\d+\\s\\d+:\\d+:\\d+)", "\\1\\2.\\3",
    data$Tiempoinicio)
data$Tiempoinicio <- as.POSIXct(data$Tiempoinicio, format="%d-%b-%Y %H:%M:%S")
```

El resultado final es un *dataset* estructurado con 14 características y una marca temporal como índice.

CAPÍTULO 3
Desarrollo

TODO

3.1 PCA

TODO

3.2 Series Temporales

TODO

3.3 Evaluación de los datos simulados

TODO

CAPÍTULO 4
Conclusión

TODO

Bibliografía

[1] Jon Shlens. A TUTORIAL ON PRINCIPAL COMPONENT ANALYSIS. Derivation, Discussion and Singular Value Decomposition. Version 1, 25 March, 2003.

[2] Unknown Authors. The Truth about Principal Components and Factor Analysis. 28 September, 2009.

[3] Christopher M. Bishop. Pattern Recognition and Machine Learning. Springer, 2006.

[4] Principal component analysis (PCA). Consultar http://scikit-learn.org/stable/modules/decomposition.html#principal-component-analysis-pca.

APÉNDICE A
Configuración del sistema

TODO

A.1 Fase de inicialización

TODO