



# Universidad **Ricardo Palma**

RECTORADO  
PROGRAMA DE ESPECIALIZACIÓN EN CIENCIA DE DATOS

*Formamos seres humanos para una cultura de paz*

## BIG DATA APLICADO


SESIÓN 09

**Expositores:**

**David Narváez**

**Eder Pineda**

**[bigdataplicado@gmail.com](mailto:bigdataplicado@gmail.com)**



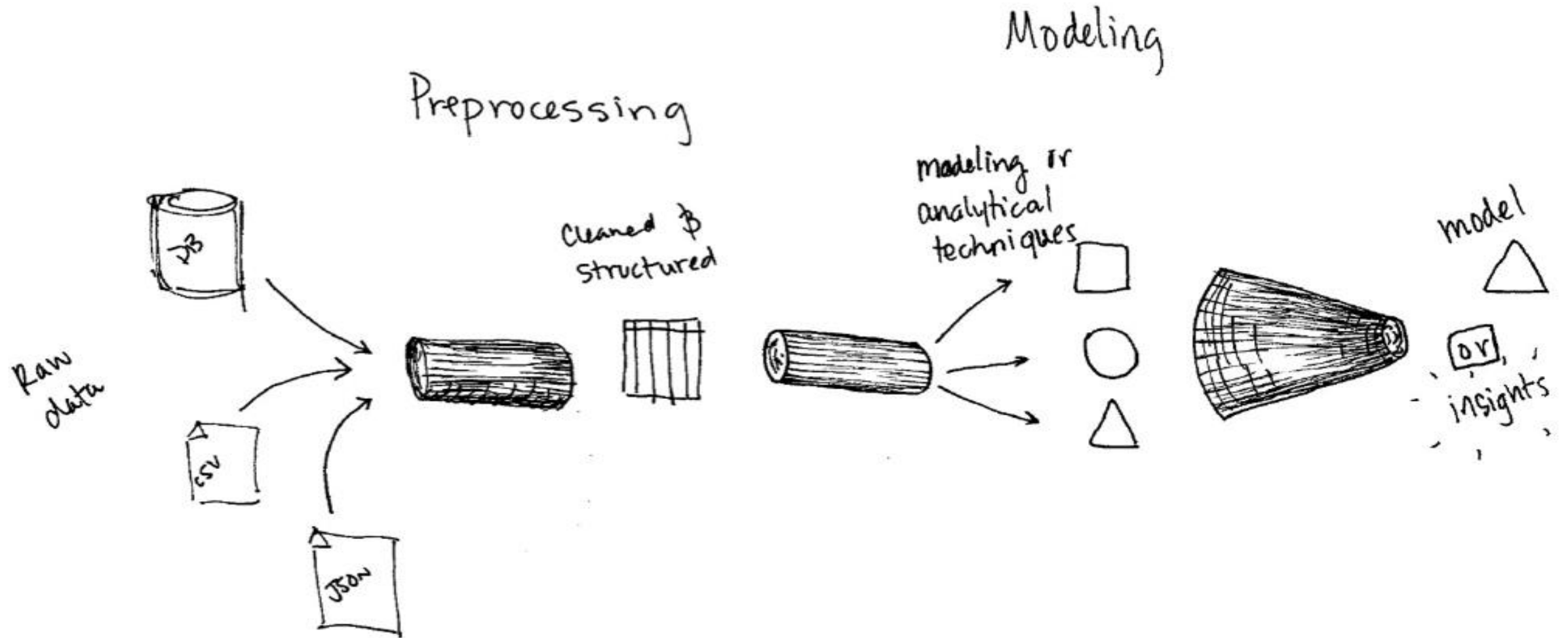
***Big data** is high-volume, high-velocity and/or high-variety information assets that demand cost-effective, innovative forms of information processing that enable enhanced insight, decision making, and process automation (**Gartner**).*





# **CASO DE APLICACIÓN DETECCIÓN DE MOROSIDAD**

# 1. Introducción a Machine Learning



### Caso Banco Chilean Credit

El conjunto de datos chilean credit contiene datos basados en seis meses de información de clientes, recolectados por un banco chileno, para desarrollar un modelo de score crediticio cuya finalidad es la de determinar la probabilidad de default (de que un cliente sea moroso). La variable respuesta es FlagGB, la cual representa el estado binario de default (0) y no default (1). El conjunto de datos tiene 7702 observaciones y 19 columnas..

### Caso Banco Chilean Credit

#### Variables:

- CustomerId: Identificador del cliente.
- TOB: Tiempo en los libros en meses desde que la primera cuenta estaba abierta.
- IncomeLevel: Nivel de ingresos de 0 (bajo) a 5 (alto).
- Bal: Saldo pendiente a la fecha.
- MaxDqBin: Rangos de Morosidad en un periodo dado bin. 0: No Dq., 1: 1-29 ... 6: 150-179.
- MtgBal: Saldo pendiente de la hipoteca en el Credit Bureau.
- NonBankTradesDq: Numero de operaciones no bancarias en un periodo dado (DQ).
- FlagGB: Indicador de Morosidad (1: Bueno, 0: Malo).
- FlagSample: Indicador de la muestra de entrenamiento y prueba (1: 75%, 0: 25%).

### Caso Banco Chilean Credit

Como dato adicional se indica que la entidad bancaria ofrece dentro de sus servicios:

- Prestamos: Hipotecarios / Otros.
- Operaciones no Bancarias (Pagos de servicios, compra de boletos, etc.).

### Caso Banco Chilean Credit

#### Fuentes de Información:

- Archivo chileancredit.csv, se encuentra en la ruta local del jupyter
  - Ruta: /home/jupyter/EderPineda/data/chileancredit.csv
- BD Clientes Mysql:
  - Host: dbbanco.cj83go6isi8e.us-east-1.rds.amazonaws.com
  - Port: 3306
  - Base de datos: Dbanco
  - Usuario/clav: root /admin123456
  - Tabla: base\_clientes



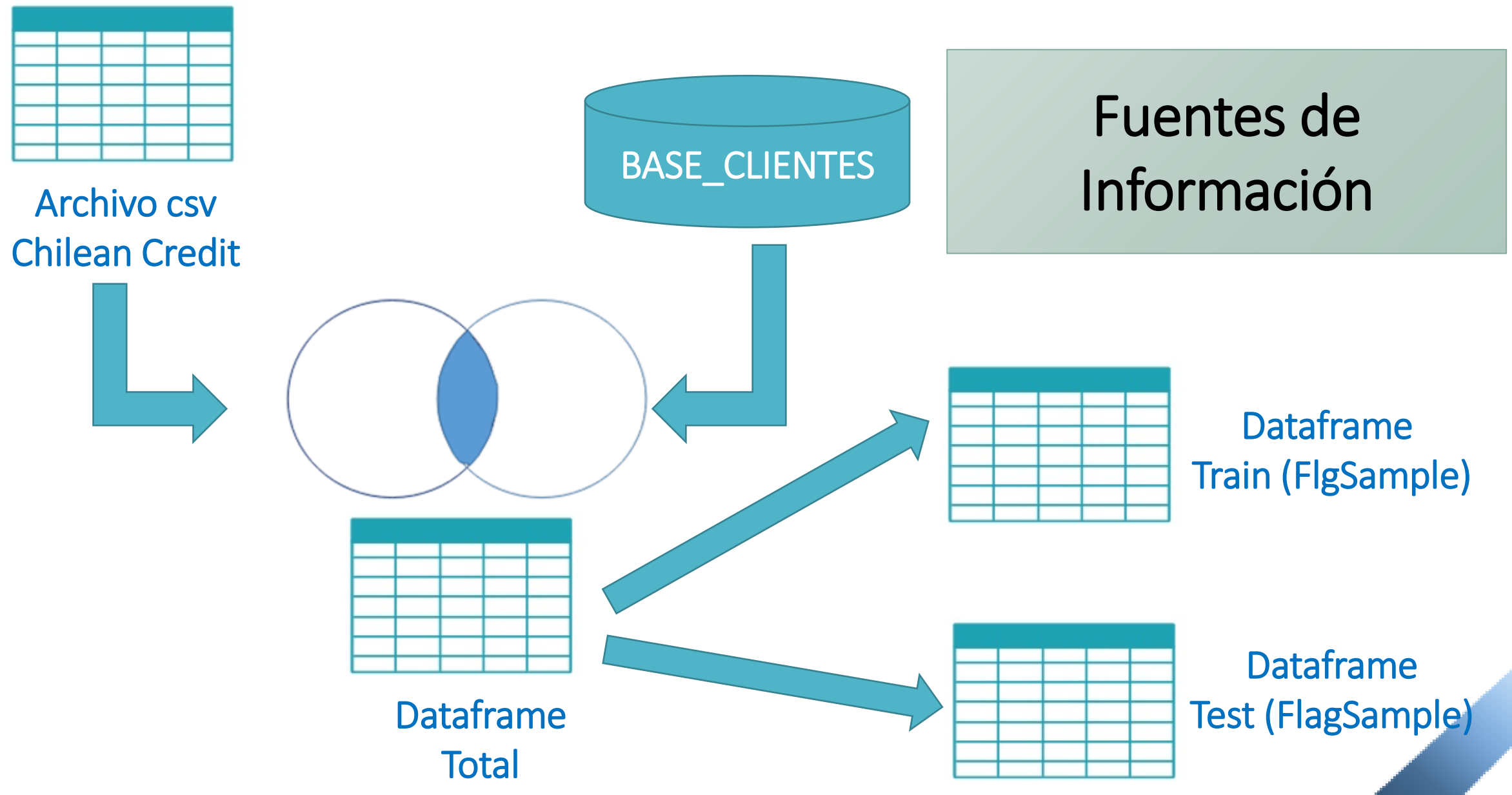


### Caso Banco Chilean Credit

#### Primera parte del taller:

1. Ingestar las fuentes de información al HDFS.
2. Crear tablas hive las cuales hagan referencia a las entidades ingestadas.
3. Crear un archivo en jupyter (Spark) el cual lean los objetos creados en el punto 2, luego proceder a unir los dos objetos por el numero\_documento y generen un nuevo dataframe con esta unión (con todas las columnas), revisar la nueva entidad (columnas, tipos de dato, etc.), y guardar el dataframe en el hdfs de procesamiento.
4. Utilice la columna FlagSample para dividir la muestra en entrenamiento y prueba.
5. Cree un modelo random forest con el nuevo dataframe (solo utilice las variables mencionadas en la diapositiva 6), pruebe el modelo y guarde el modelo.

## 7. MACHINE LEARNING



### Estructura HDFS Ingesta

A continuación, se muestra la estructura de los archivos que se deberá tener en cuenta para el almacenamiento de datos en BigData para cada proyecto que se desarrolle.

- /user/[su\_usuario]
  - /data-in
    - /fuente
      - /esquema
        - /tabla

Carpeta	Descripción
data-in	Representa el directorio en el cual se almacenará la información que está ingresando al data lake, organizada por subdirectorios
Fuente	Representa la fuente de donde proviene la información (nombre instancia de BD, nombre del archivo de entrada, nombre de la pagina de web scrapping).
Esquema	Representa el esquema (en caso que la fuente sea una base de datos) o contexto de la información.
Tabla	Nombre de la tabla en caso la fuente de información sea una tabla a ingestar

### Estructura Hive Ingesta

A continuación, se muestra la estructura de las tablas externas hive.

- /[Base Datos Usuario]
  - Tabla:  
[data\_in]\_[contexto]

Carpeta	Descripción
Base Datos Usuario	Representa la base de datos que registro el usuario para su trabajo (deberia ser el nombre del usuario)
Data_in	Representa que la tabla proviene de un archivo ingestado.
Contexto	Nombre del contexto de la información almacenada



### Estructura HDFS Procesamiento

A continuación, se muestra la estructura de los archivos que se deberá tener en cuenta para el almacenamiento de datos en BigData luego del procesamiento.

- /user/[su\_usuario]
  - /data-out
    - /esquema
      - /tabla

Carpeta	Descripción
data-out	Representa el directorio en el cual se almacenará la información producto de un procesamiento previo
Esquema	Representa el esquema (en caso que la fuente sea una base de datos) o contexto de la información.
Tabla	Representa el directorio en el cual se almacenará los datos de una tabla o archivo que son ingestado al ambiente Big Data.

### Estructura Hive Ingesta

A continuación, se muestra la estructura de las tablas externas hive.

- /[Base Datos Usuario]
  - Tabla:  
[data\_out]\_[contexto]

Carpeta	Descripción
Base Datos Usuario	Representa la base de datos que registro el usuario para su trabajo (deberia ser el nombre del usuario)
Data_out	Representa que la tabla proviene de un archivo de procesamiento.
Contexto	Nombre del contexto de la información almacenada

### Estructura FS

La siguiente estructura se deberá manejar para todos los archivos. Los nombres de los archivos (csv, txt, etc.) deberán ser cortos, descriptivos y que permitan determinar fácilmente su propósito, deben estar descritos como:

- /home/jupyter/[su\_usuario]
  - /app
    - /contexto
      - /proceso
        - /bin
        - /conf
        - /log

Carpeta	Descripción
app	Representa el directorio en el cual se almacenará el código de las aplicaciones o procesos ETL
Contexto	Representa el contexto al cual pertenece la información
Proceso	Representa el proceso (etl) en donde se generarán los datos
Bin	Directorio en el cual se almacenara el código compilado y/o los ejecutables
Conf	Directorio en el cual se almacenara los archivos de configuración propios de la aplicación
log	Representa el directorio en el cual se almacenará la información generada por la ejecución de los procesos ETL.

### Caso Banco Chilean Credit

#### Segunda parte del taller:

1. Del dataframe resultante del modelo del primer taller (con la data de prueba), crear un nuevo dataframe con las columnas que contenga solo las siguientes columnas NUMERODOCUMENTO, FLAGDB (LABEL), P1(PROBABILIDAD1), guardar el resultado en HDFS.
2. Convertir el dataframe anterior en un pandas dataframe.
3. Importar la función (/home/jupyter/funciones/pyspark\_ml\_util.py)
4. Utilice la función importada y con el método density\_chart muestre el gráfico de densidad
5. Utilice la función importada y con el método indicadores\_modelos muestre los principales indicadores del modelo
6. Utilice la función importada y con el método reporte\_ganancia\_lift muestre los principales indicadores del modelo
7. Analizar y resumir los resultados obtenidos

### Caso Banco Chilean Credit

Comandos para guardar y abrir modelos HDFS (random forest):

Importar:

```
from pyspark.ml.classification import RandomForestClassificationModel
```

Guardar:

```
rfModel.write().overwrite().save("/user/usr_big_data/modelado/undersampling/rf2")
```

Abrir:

```
rfModel =  
RandomForestClassificationModel.load("/app/cu_televentas/mercado_personas/ml_personas  
_forecast/")
```



### Caso Banco Chilean Credit

Comandos para dividir columnas vectorizadas:

```
split1_udf = func.udf(lambda value: value[0].item(), FloatType())
```

```
split2_udf = func.udf(lambda value: value[1].item(), FloatType())
```

```
df_1_total_output = predictions_xgb_proceso.select("MESFILE", "TELEFONO", "CODIGOCONTRATOBSCS", "RUCCOM  
PANIA", "prediction",  
split1_udf('probability').alias('p0'),  
split2_udf('probability').alias('p1'))
```

```
df_1_total_output_exporta=df_1_total_output.selectExpr("*", "(case when prediction == 1  
then p1 else p0 end) as probability")
```

### Caso Banco Chilean Credit

#### GridSearch Tunning:

El siguiente método **no** debe de ser ejecutado en el laboratorio.

```
assembler_train = VectorAssembler(inputCols=["UNIDS_POST_SIN_2T", "UNIDS_PRE"],
outputCol="features")
scaler_test = StandardScaler(inputCol="features", outputCol="scaled_features", withStd=True,
withMean=True)
rfr = RandomForestRegressor(labelCol="TARGET_1", featuresCol="scaled_features")

stages = [assembler_train, scaler_test, rfr]
pipe = Pipeline(stages=stages)
```

### Caso Banco Chilean Credit

```
estimatorParam = ParamGridBuilder()\n    .addGrid(rfr.maxDepth, [4, 6, 8])\n    .addGrid(rfr.maxBins, [5, 10, 20, 40])\n    .addGrid(rfr.impurity, ["variance"])\n    .build()
```

```
evaluator      =      RegressionEvaluator(labelCol="TARGET_1",      predictionCol="prediction",\nmetricName="r2")
```

```
crossval = CrossValidator(estimator=pipe,\n    estimatorParamMaps=estimatorParam,\n    evaluator=evaluator,\n    numFolds=4)
```

### Caso Banco Chilean Credit

```
model      =      pd.DataFrame(cvmodel.bestModel.stages[-1].featureImportances.toArray(),
columns=["values"])
features_col = pd.Series(features)
model["features"] = features_col
model.to_csv('model_03_09_2018_1.csv')
```

```
bestPipeline = cvmodel.bestModel
bestLRModel = bestPipeline.stages[2]
bestParams = bestLRModel.extractParamMap()
```

```
print bestParams
print bestLRModel
print bestPipeline
```

A close-up photograph of a right hand holding a silver ballpoint pen, writing the words "Thank you" in a cursive script on a white surface. The pen is positioned at the end of the word "you".

Thank you