Files formats & Predicting Future Sales

Alexandre Alonso

July 9th 2018

Project 1: Predict shop's future sale

We want predict next month total sales of a given shop.

Steps to do:

1. Data preparation
2. Train-test splitting
3. Training linear model
4. Evaluating: implementing MSE & R2

```
library(sparklyr)
```

```
## Warning: package 'sparklyr' was built under R version 3.5.1
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
sc<-spark_connect(master="local")
```

```
## * Using Spark: 2.3.0
```

```
sales_sdf <- spark_read_csv(sc, "sales",
"../future_sales_data/sales_train.csv.gz")
```

```
sales_sdf %>%
  head
```

```
## # Source:   lazy query [?? x 6]
## # Database: spark_connection
##   date       date_block_num shop_id item_id item_price item_cnt_day
##   <chr>               <int>   <int>   <int>      <dbl>        <dbl>
## 1 02.01.2013              0      59   22154        999            1
## 2 03.01.2013              0      25    2552        899            1
```

```
## 3 05.01.2013              0      25    2552       899            -1
## 4 06.01.2013              0      25    2554      1709.            1
## 5 15.01.2013              0      25    2555      1099             1
## 6 10.01.2013              0      25    2564       349             1
```

```
sales_sdf %>%
  mutate(dt=to_timestamp(unix_timestamp(date, 'dd.MM.yyyy'))) %>%
  mutate(year=year(dt), month=month(dt)) %>%
  select(-dt) ->
  sales_sdf
```

```
monthly_sdf <-
sales_sdf %>%
  group_by(year, month,shop_id)%>%
  summarise(total_items=sum(item_cnt_day,na.rm=TRUE))
```

```
sdf_register(monthly_sdf, "sales_monthly")
```

```
## # Source:   table<sales_monthly> [?? x 4]
## # Database: spark_connection
##     year month shop_id total_items
##    <int> <int>   <int>       <dbl>
##  1  2013     1      24        1768
##  2  2013     1      23        1948
##  3  2013     1      29        2820
##  4  2013     1       1        2947
##  5  2013     1      14        1777
##  6  2013     1      47        2115
##  7  2013     1      43        1759
##  8  2013     1      52        1812
##  9  2013     2      42        3965
## 10  2013     2      43        2033
## # ... with more rows
```

```
library(DBI)
dbGetQuery(
  sc,
"SELECT *
,  LAG(total_items,3) OVER (PARTITION BY shop_id ORDER BY year, month) AS
prev_total_items_3

,  LAG(total_items,2) OVER (PARTITION BY shop_id ORDER BY year, month) AS
prev_total_items_2

,  LAG(total_items) OVER (PARTITION BY shop_id ORDER BY year, month) AS
prev_total_items

FROM sales_monthly") %>%

  mutate(lag1=ifelse(is.nan(prev_total_items),0,prev_total_items))%>%
```

```
  mutate(lag2=ifelse(is.nan(prev_total_items_2),0,prev_total_items_2))%>%
  mutate(lag3=ifelse(is.nan(prev_total_items_3),0,prev_total_items_3))%>%
  mutate(lags=(lag1+lag2+lag3)/3)->
  final_sdf

class(final_sdf)

## [1] "data.frame"

final_sdf<-sdf_copy_to(sc,final_sdf,"final_sdf", overwrite=TRUE)

train_sdf<- final_sdf %>%
  filter(!(year==2015 & month==10))

test_sdf<- final_sdf %>%
  filter(year==2015 & month==10)

test_sdf

## # Source:   lazy query [?? x 11]
## # Database: spark_connection
##     year month shop_id total_items prev_total_items_3
prev_total_items_2
##    <int> <int>   <int>       <dbl>              <dbl>
<dbl>
##  1  2015    10      12        4181               1554
1471
##  2  2015    10      14        1002                954
1061
##  3  2015    10      18        1211                987
1184
##  4  2015    10      25        6247               4676
4675
##  5  2015    10      37         833               1041
1248
##  6  2015    10      38        1110               1354
1781
##  7  2015    10      46        1320               1642
1670
##  8  2015    10      50         949               1126
1081
##  9  2015    10      52         847                828
932
## 10  2015    10      56        1263               1491
1604
## # ... with more rows, and 5 more variables: prev_total_items <dbl>,
## #   lag1 <dbl>, lag2 <dbl>, lag3 <dbl>, lags <dbl>

test_sdf %>%
  summarise(mean(total_items))
```

```
## Warning: Missing values are always removed in SQL.
## Use `AVG(x, na.rm = TRUE)` to silence this warning

## Warning: Missing values are always removed in SQL.
## Use `AVG(x, na.rm = TRUE)` to silence this warning

## # Source:   lazy query [?? x 1]
## # Database: spark_connection
##   `mean(total_items)`
##                 <dbl>
## 1               1615.

train_sdf%>%
  ml_linear_regression(total_items ~ lag1 + lag2 + lag1:lag2 + lag2:lag3
+ lag1:lag2:lag3 + year +  month + month:lag1) ->
  model

summary(model)

## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -4523.49  -345.56   -79.68   197.91  8616.69
##
## Coefficients:
##    (Intercept)           lag1           lag2      lag1:lag2
lag2:lag3
##   5.439060e+05   3.425012e-01  -3.761112e-01   6.878502e-05
6.574351e-05
## lag1:lag2:lag3           year          month     month:lag1
##  -7.763451e-09  -2.693814e+02  -8.497595e+01   5.456323e-02
##
## R-Squared: 0.8285
## Root Mean Squared Error: 793.6

(significance <-
c(model$model["intercept"][[1]],model$model["coefficients"][[1]])/model$s
ummary["coefficient_standard_errors"][[1]])

## [1]  9.913822 11.827033 -8.549553  9.779269  9.872727 -9.490758 -
9.888963
## [8] -9.043832 18.127088

round(model$summary["p_values"][[1]],5)

## [1] 0 0 0 0 0 0 0 0 0

train_sdf %>%
  sdf_predict(model) %>%
  mutate(res=log(total_items+1) - log(prediction+1))%>%
  summarise(mean(res*res))   #implement mean(log(y+1)-log(y'+1))^2
```

```
## Warning: Missing values are always removed in SQL.
## Use `AVG(x, na.rm = TRUE)` to silence this warning

## Warning: Missing values are always removed in SQL.
## Use `AVG(x, na.rm = TRUE)` to silence this warning

## # Source:   lazy query [?? x 1]
## # Database: spark_connection
##   `mean(res * res)`
##               <dbl>
## 1             0.134

  print

## function (x, ...)
## UseMethod("print")
## <bytecode: 0x0000000008b0aec0>
## <environment: namespace:base>

test_sdf

## # Source:   lazy query [?? x 11]
## # Database: spark_connection
##     year month shop_id total_items prev_total_items_3
prev_total_items_2
##    <int> <int>   <int>       <dbl>              <dbl>
<dbl>
##  1  2015    10      12        4181               1554
1471
##  2  2015    10      14        1002                954
1061
##  3  2015    10      18        1211                987
1184
##  4  2015    10      25        6247               4676
4675
##  5  2015    10      37         833               1041
1248
##  6  2015    10      38        1110               1354
1781
##  7  2015    10      46        1320               1642
1670
##  8  2015    10      50         949               1126
1081
##  9  2015    10      52         847                828
932
## 10  2015    10      56        1263               1491
1604
## # ... with more rows, and 5 more variables: prev_total_items <dbl>,
## #   lag1 <dbl>, lag2 <dbl>, lag3 <dbl>, lags <dbl>
```

```
  sdf_predict(model) %>%
  mutate(res=log(total_items+1) - log(prediction+1))%>%
  summarise(mean(res*res))
```

```
## Warning in sdf_predict.ml_model(model): The signature
sdf_predict(model, dataset) is deprecated and will be removed in a future
version. Use sdf_predict(dataset, model) or ml_predict(model, dataset)
instead.

## Warning in sdf_predict.ml_model(model): Missing values are always
removed in SQL.
## Use `AVG(x, na.rm = TRUE)` to silence this warning

## Warning in sdf_predict.ml_model(model): Missing values are always
removed in SQL.
## Use `AVG(x, na.rm = TRUE)` to silence this warning

## # Source:   lazy query [?? x 1]
## # Database: spark_connection
##   `mean(res * res)`
##               <dbl>
## 1             0.134
```

```
  print
```

```
## function (x, ...)
## UseMethod("print")
## <bytecode: 0x0000000008b0aec0>
## <environment: namespace:base>
```

```
train_sdf %>%
  sdf_predict(model) %>%
  mutate(res=log(total_items+1) - log(prediction+1))%>%
  print->ploty
```

```
## # Source:   lazy query [?? x 13]
## # Database: spark_connection
##     year month shop_id total_items prev_total_items_3
prev_total_items_2
##    <int> <int>   <int>       <dbl>              <dbl>
<dbl>
##  1  2013     1      12         842                NaN
NaN
##  2  2013     2      12        1209                NaN
NaN
##  3  2013     3      12        1419                NaN
842
##  4  2013     4      12        1364                842
1209
##  5  2013     5      12         917               1209
1419
```

```
##  6  2013     6      12        1710                1419
1364
##  7  2013     7      12         723                1364
917
##  8  2013     8      12        1599                 917
1710
##  9  2013     9      12        2032                1710
723
## 10  2013    10      12        1890                 723
1599
## # ... with more rows, and 7 more variables: prev_total_items <dbl>,
## #   lag1 <dbl>, lag2 <dbl>, lag3 <dbl>, lags <dbl>, prediction <dbl>,
## #   res <dbl>
```

```
  library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.5.1
```

```
  ggplot(ploty, aes(prediction, res))+
  geom_point() +
  geom_smooth() +
  scale_size_area()
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

```
## Warning: Removed 4 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 4 rows containing missing values (geom_point).
```