Data Mining and Text Mining Project

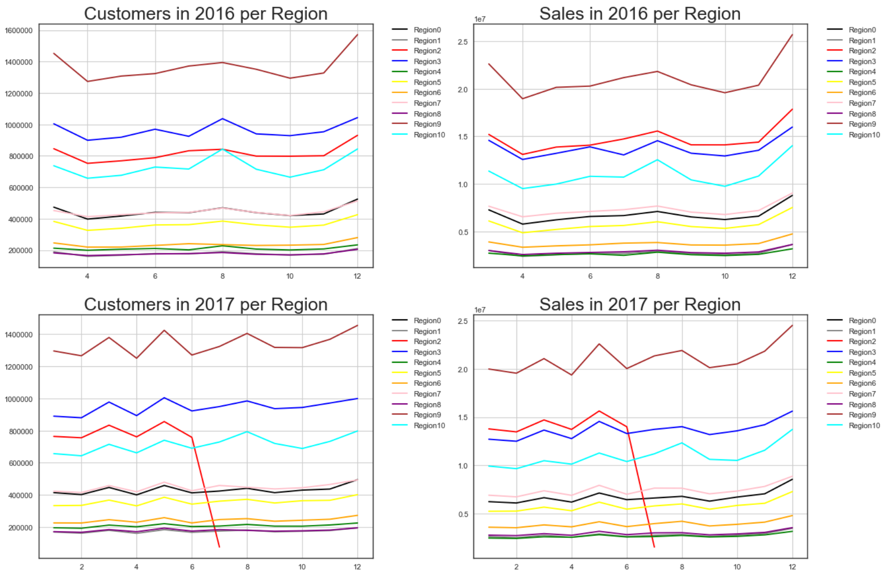
10/06/2018

Paticchio Alessandro Saverio, Sandrelli Federico, Santambrogio Davide, Scorsolini Philippe, Tricarico Andrea

Bip xTech - Sales Forecast

1. Data Exploration

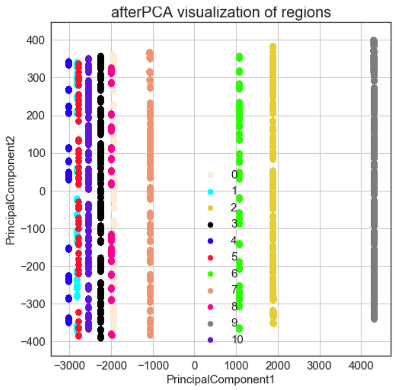
The first thing we did was to analyse the behavioural trends of every region. We plotted the **total customers** and **total sales** per month of all regions.
Here we attach our results:



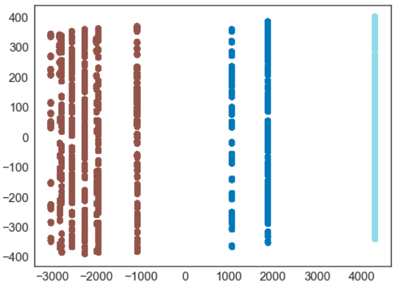
The very first thing we notice is a huge fall of Region 2 from July 2017: by exploring the dataset we discovered we were missing the tuples from July up to January 2018.

As the graph shows and as expected, **the number of customers and the number of sales in one month are strictly correlated.**
Furthermore, there is an increase of sales in March, May, August and in December. Region 2 asides, no significant differences of trends was found between 2016 and 2017 (please note that data from 2016 starts from March).

As a second step, we performed a PCA, and ran k-means clustering to pointing out the different regional trends:



We stopped k-means at k = 3, highlighting three clusters:



- **Region 0, Region 1, Region 3, Region 4, Region 5, Region 7, Region 8, Region 10**
- **Region 2, Region 6**
- **Region 9**

2. Preprocessing

IsOpen Drop

By exploring the rows with the attribute “IsOpen” set to 0, it turned out that the sales were also 0.
Therefore we dropped all the rows with the store closed, since they only caused noise in the model.
We will simply set NumberOfSales to 0 in the submission dataset.

Missing Values

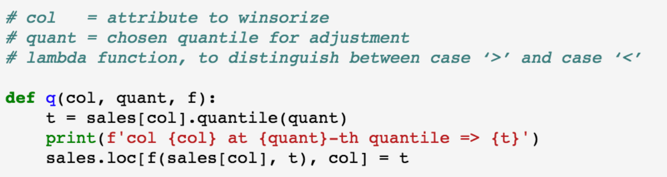
Listing all the attributes with missing values, this was the result:

• CloudCover	41181
• Events	124098
• Max_Gust_SpeedKm_h	409947
• Max_VisibilityKm	11338
• Mean_VisibilityKm	11338
• Min_VisibilitykM	11338

By asking to the domain expert, we were told that when “Events” is missing, it means no event has occurred, so we imputed those values as “Not Specified”.
We then imputed "CloudCover" with the mean, making a distinction for when it missed along with Events and when it missed on its own.
Finally, we noticed that the number of missing values of Max/Mean/Min_VisibilityKm was the same and when one of them missed, the others did as well: we imputed them with the respective mean.

Dealing with outliers

By plotting the trends of the numerical attributes, we applied **winsorization** in order to eliminate outliers.
We applied it to the vars with the furthest outliers using the following function:



```
# col   = attribute to winsorize
# quant = chosen quantile for adjustment
# lambda function, to distinguish between case ‘>’ and case ‘<’

def q(col, quant, f):
    t = sales[col].quantile(quant)
    print(f'col {col} at {quant}-th quantile => {t}')
    sales.loc[f(sales[col], t), col] = t
```

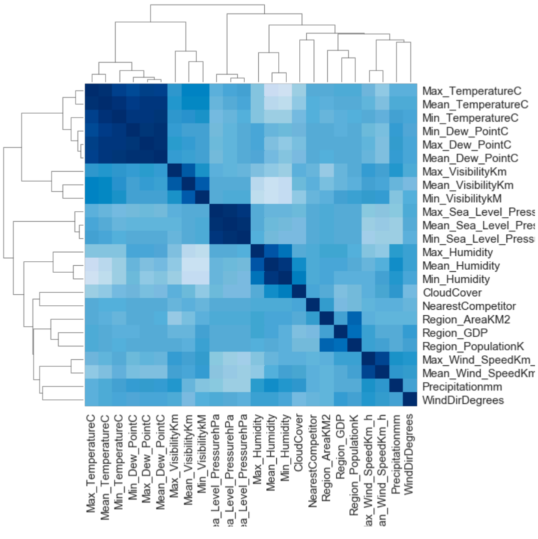
Normalization

We applied a **np.log1p** to all the numerical variables with a skewness greater than 0.75.
Obviously, we excluded NumberOfCustomers since it is one of our target variable (we will see later why).

Correlation Analysis

We plotted a **cluster-map** to analyze the feature’s correlations.
By analyzing the **dendrograms**, we saw that the following variables are quite uncorrelated from the target variable and very correlated to others. We decided to drop them:

- *Max_Dew_PointC*
- *Min_Dew_PointC*
- *Max_Sea_Level_PressurehPa*
- *Mean_Sea_Level_PressurehPa*
- *Max_Gust_SpeedKm_h*



3. Training the Model

Feature manipulation

From “**date**”, we extracted the day of the week (“**weekday_name”**) and “**month**” features, and considered them as categorical attributes. We decided not to considered “**Year**” as a feature since, in our first step in data exploration.,we realized there is no annual trend.

Then, in order to better fit the general trend of the data, we added the following features to the dataset:

- **AvgSalesForMonth**: the mean of the total sales for the StoreID of the considered tuple in that month.
- **VarSalesForMonth**: the variance of the total customers for the StoreID of the considered tuple in that month.
- **AvgCustomersForMonth**: the mean of the total customers for the StoreID of the considered tuple in that month.
- **VarCustomersForMonth**: the variance of the total customers for the StoreID of the considered tuple in that month.

How do we predict?

We decided to split the prediction in two steps:

1. Prediction of NumberOfCustomers
2. Prediction of NumberOfSales

First step

We decided to split the dataset in different groups, putting together similar regions. The groups we created were the ones pointed out by t-SNE:

1. Region 0, Region 1, Region 5, Region 8, Region 10 + Region 4 (its sales trend is very similar to the one of Region 8) and Region 7 (its sales trend is very similar to the one of Region 0)
2. Region 2
3. Region 3
4. Region 6, Region 9

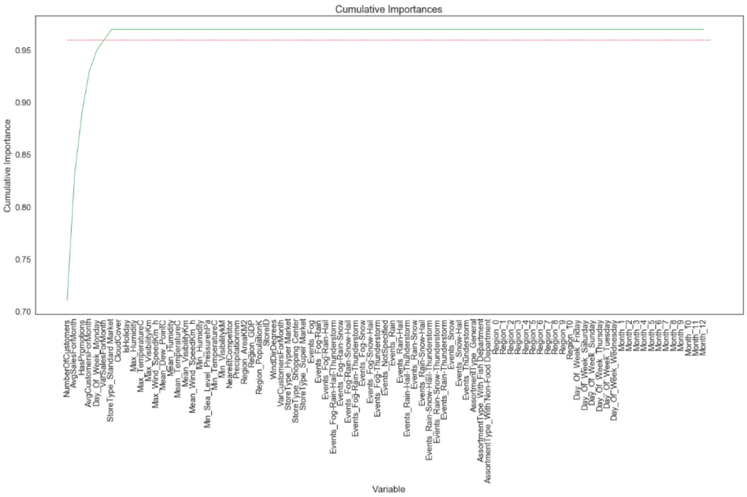
Therefore, we created 4 different models, each of them trained on the data coming from one of the groups. In this step, we trained the model using all the features selected by the stepwise selection.

Second step

Feature Selection
For the second step, we applied feature selection to achieve better runtime performance. Random Forests are quite complex and run slowly, speeding up the training process allowed for a faster evolution of the project. Furthermore, this brought us to remove unnecessary variables, improving the accuracy of our models and reducing overfitting by generalizing our model. Feature selection was tested on the first step as well but was then dropped because it worsened the error rate.

We attached the NumberOfCustomers, predicted in the first step, to our data and tried to predict the NumberOfSales, building a single model trained on the whole dataset.

The model only trains on the features which are mostly correlated with NumberOfSales (extracted during feature selection with a threshold of 0.96 on Cumulative Importance):



```
Variable: NumberOfCustomers    Importance: 0.71
Variable: AvgSalesForMonth     Importance: 0.12
Variable: HasPromotions        Importance: 0.06
Variable: AvgCustomersForMonth Importance: 0.04
Variable: Day_Of_Week_Monday   Importance: 0.02
Variable: VarSalesForMonth     Importance: 0.01
Variable: StoreType_Standard Market Importance: 0.01
Variable: CloudCover           Importance: 0.0
Variable: IsHoliday            Importance: 0.0
Variable: Max_Humidity         Importance: 0.0
```

Random Forests

We decided to use Random Forest predictors for both of our prediction steps.
Random forests are usually very accurate and robust, their main drawback being that they are slow. With our small dataset, though, this was not a problem. Furthermore Random Forests are particularly suited for fitting training data with features that are both categorical and numerical, such as what we find in our dataset.

How did we Evaluate our Model?

Error Function

We implemented the error function given in the problem definition. The output is the average delta, expressed as a percentage, between the total monthly sales per region predicted and the actual values.
The Error Function can be found in the source code.

Cross Validation

To evaluate the performance of our model we used an 10-fold Cross-Validation implemented with the error function previously described.

We divided the whole training dataset into subsets with a size of 2 months.

This approach seemed to best fit our case since the actual test set extends over a two months period (this is why alternative methods such as bootstrapping were discarded).

Here are the results of our latest run. The value on the right is the average of the regional results on the left (the format is: [last date of subset], [% average error over regions for the given subset]):

```
2016-05-01 0.04837209833416723
2016-07-01 0.04590234262487461
2016-09-01 0.031013976358494486
2016-11-01 0.04577325359924902
2017-01-01 0.03423919562800674
2017-03-01 0.07839412454892183
2017-05-01 0.06168749686615873
2017-07-01 0.06462148135290285
2017-09-01 0.0472151866208941
2017-11-01 0.04428474464258977
```

```
0.045721915593366955
```

The average computed error is below 4.6%.