Hackathon

Which customers will leave "Kin Safety" product before 2 years?

A HISTORY WITH DATA TO IMPROVE CUSTOMER RETENTION

"Data are just summaries of thousands of stories – tell a few of those stories to help make the data meaningful." Dan Heath

M. Eng. David González,
Data Scientist Junior Candidate

Introduction

Hello dear reviewer, In this short presentation I will show you an abstract of the process to find a solution to predict which customers will leave “Kin Safety” product within 2 years.

For this purpose, I created 6 sections for each step of my proposed solution:

1) **Desired Population:** Explanation of the adequacy of the data , each filter to obtain only the desired customers to analyze.
2) **New Variables:** Explanation of how I got the age, number of products, account balance, and bureau score at the moment of application of each costumer.
3) **Exploring Data:** Explanation of how I decided the way to select a prediction model.
4) **Prediction model:** Explanation of the creation of different classifier models.
5) **Evaluation:** Explanation of comparison with a base model, cross-validation process, and some predictions for a confusion matrix.
6) **Conclusion:** Some thoughts about the process.

Kin ANALYTICS Hackathon

Desired Population

Objective | Process | Result

I provide two ways:
1) Online. Files uploaded to a folder on google drive.
2) Local files. The path must be changed on different computers.

Check and delete duplicate entries:
```
*clients.drop_duplicates(inplace = True)
*assert len(clients['CustomerId’].unique()) == len(clients)
```

Convert to datetime format and querying:
```
*clients['application_date'] = pd.to_datetime(clients['application_date’])
*clients = clients[(clients['application_date'] >= '2015') & (clients['application_date'] <= max(clients['application_date']) - pd.Timedelta('730d'))]
```

Check if Italy appears in the dataset:
```
*print(clients['Geography'].unique())
```

Check missing values:
```
*print(clients.apply(lambda x: np.sum(x.isna()), axis = 0))
```

Load Data

Only one contract per client.

- Contracts from 2015 onwards.
- Clients with at least two years of information within the company.

Operations in Italy.

Costumers missing more than 75% of their info.

clients dataset:
1545000 entries

Duplicated entries:
45000 duplicate entries removed
1500000 entries after filter

Querying data:
1490000 entries don’t have the requirements
10000 entries after filtering.

Italy and missing data:
Italy doesn’t appear in the new dataset; no filter needed.
Missing data only in one column. No client has more than 75% missing info; no filter needed
10000 entries, No filter was applied.

M. Eng. David González,
Data Scientist Junior Candidate

Kin ANALYTICS Hackathon

New Variables

Age

- It was calculated using the difference between the application date and birth date.

```
•clients['Age']=(clients['application_date']-clients['birth_date']).astype('timedelta64[Y]')
```

Number of products

- It was calculated with 'groupby' with the customer ID and 'agg' with the 'len' function. Then, the result was merged with the clients' dataset.

```
•products=clientProducts.groupby('CustomerId').agg({'Products':len})
•clients=pd.merge(clients,products,how='left',on='CustomerId')
```

Account balance

- It was calculated with 'groupby' with the customer ID and 'agg' with the 'np.sum' function. Then, the result was merged with the clients' dataset and converted the small numbers to 0.

```
•balance=transactions.groupby('CustomerId').agg({'Value':np.sum})
•clients=pd.merge(clients,balance,how='left',on='CustomerId')
```

Bureau Score

- It was calculated by searching the bureau's score dataset using the customer ID and month of the application. This was done by the 'get_score' function through apply function.

```
•clients['Score']=clients.apply(get_score,axis=1)
```

Statistics of new variables.

	Age	Products	Balance	Score
Mean	38.92	1.53	76485.88	650.52
Std	10.48	0.581	62397.40	96.65
Max	92	4	250898.09	850.00
Min	18	1	0.00	350.00

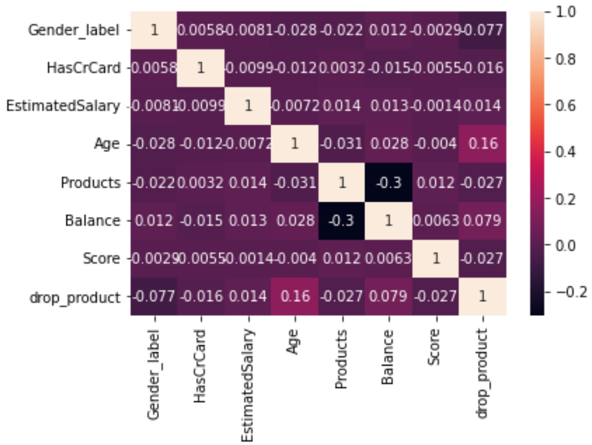
This table was made with:

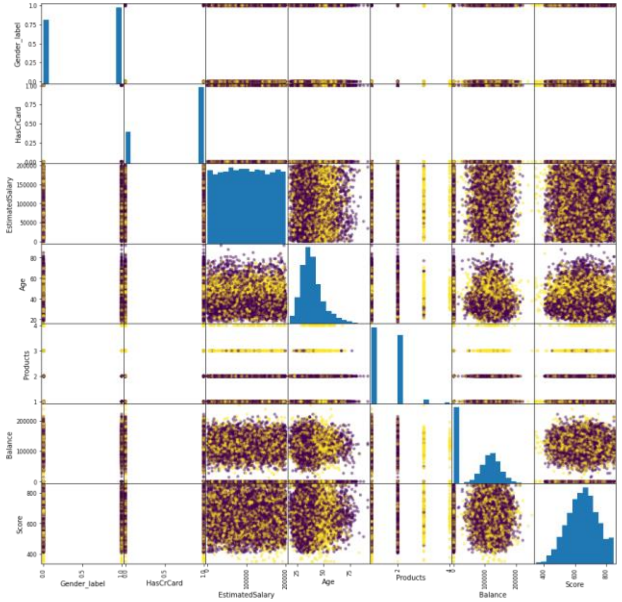
```
*clients[['Age', 'Products', 'Balance', 'Score']]
  .agg({'Age': [np.mean, np.std, np.max, np.min],
   'Products': [np.mean, np.std, np.max, np.min],
    'Balance': [np.mean, np.std, np.max, np.min],
     'Score': [np.mean, np.std, np.max, np.min]})
```

Exploring Data

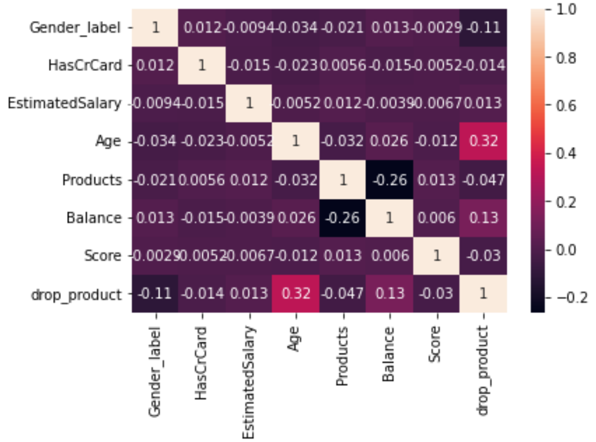
- First, I calculate the number of days of customer stay with the difference between the application date and the exit date, but a problem wasn't solved in the preparation data: The missing values of the exit date.
- By creating a new variable 'drop_product' with 1 if the client cancelled the product before 2 years and 0 otherwise, I had NaT values.
- I tried to fill this with 0 if the client is an active member but the result had too much noise.
- I decided to remove these values and the correlation matrix and scatter chart look better.

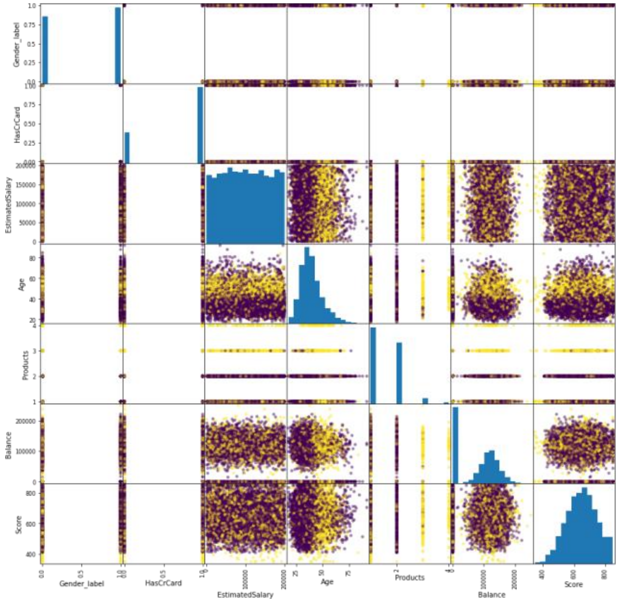
Without dropping missing values





Dropping missing values





Kin ANALYTICS Hackathon

Prediction model

Get train and test data

```
x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=0)
```

Normalize data

```
sc=StandardScaler()
x_train=sc.fit_transform(x_train.values)
x_test=sc.transform(x_test.values)
```

Fit models

```
knn=KNeighborsClassifier(n_neighbors=10).fit(x_train,y_train)
dtc=DecisionTreeClassifier(max_depth=3).fit(x_train,y_train)
⋮
gbc=GradientBoostingClassifier(random_state=0,learning_rate=0.2).fit(x_train,y_train)
```

Results of the models

Model	Accuracy
K Neighbors	0.75835
Decision Tree	0.77394
Logistic Regression	0.70545
Support Vector, poly kernel	0.76726
Support Vector, rbf kernel	0.77116
Random Forest	0.78341
Gradient Boosting	0.79065

After this step, I selected the Random Forest Classifier and the Gradient Boosting Classifier to evaluate.

Kin ANALYTICS Hackathon

Evaluation

Compare with a baseline model

The models were compared with a Dummy Classifier from the sklearn package.

```
dc=DummyClassifier(strategy=
'most_frequent').fit(x_train,y_train)
```

Random Forest Classifier is **8.797%** better than the Dummy Classifier.

Gradient Boosting Classifier is **9.521%** better than the Dummy classifier

Cross-validation

Get the scores of cross-validation test.

Get the mean and standard deviation of the test score.

```
cross_val_score(rfc,np.concatenate((x_train,x_test)),np.concatenate((y_train,y_test)))
```

RFC mean is: **0.7938**, and std is: **0.01159**.

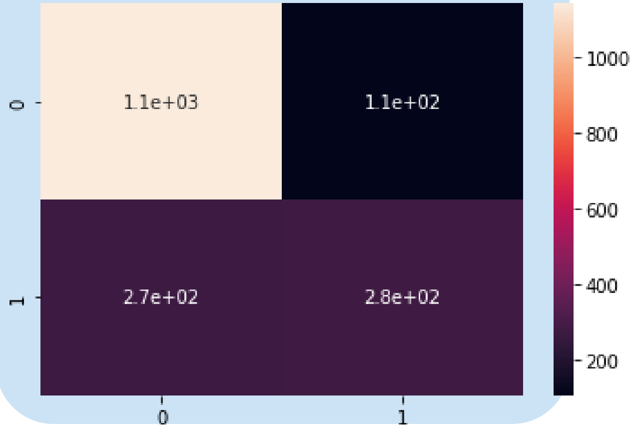
GBC mean is: **0.8013**, and std is: **0.0066**.

Confusion matrix

Get the predictions of the Gradient Boosting Classifier.

Create and graph the confusion matrix.

```
y_pred=gbc.predict(x_test)
cm=confusion_matrix(y_test,y_pred)heatmap=sn.heatmap(cm,annot=True)
```



Conclusion

In this section, I only share some ideas about the process and the result.

- I'm seeking the best binary classifier to do this task. The requirement is: Predict whether or not a customer leaves the "Kin safety" product before 2 years. It's a yes or no question, that is the reason to select a binary classifier.
- I decided not to take geographic location to make a prediction. The reason is that the leaked dataset has only 3 countries, and I am interested in making a model as general as possible.
- The result has a significant difference between the Dummy Classifier and the Gradient Boosting Classifier, I conclude the process was successful.
- The model has more false-positive rates than false-negative rates. In a business context, It isn't very bad. It's better to focus on clients that we believe will leave our product, although this may not happen.
- In order to do better prediction, it is interesting to explore AI techniques, especially a few-point learning.

Acknowledgment

Thanks for reading and for the opportunity to show what I can do. I hope we can work together, good luck with everything, and feel free to visit the GitHub site of this project : https://github.com/DataGonza/cancellationFinancialProducts

M. Eng. David González,
Data Scientist Junior Candidate