Machine Learning capstone proposal

Deividi Pansera

Domain Background

Businesses in the consumer market and, in fact, in all enterprise sectors have to deal with churn. Very often, churn is excessive and influences policy decisions. The traditional solution that can be found is to predict high-propensity churners and address their needs via something like a concierge service, marketing campaigns, etc. It can vary from industry to industry.

Well, the common factor is that businesses need to minimize these special customer retention efforts. Therefore, given this perspective, a natural methodology would be to score a churn probability to every customer and to address the top N ones. The top customers might be the most profitable ones or not. It depends of a lot of variables.

The domain background of this project is churn prediction for a Brazilian newspaper media called Gazeta do Povo (GP). The main source of business income for GP is through payed subscription at its website. Therefore, the churn problem is crucial for its business.

Problem statement

The main source of business income for GP is through payed subscription, as it was said before. So, for this project, the purpose is to build a Machine Learning model to predict the churn based on a handful of algorithms that we will test (e.g. Random Forest, XGB, Gradient Boosting and others). We shall conduct experiments and with the results we will demonstrate the performance of the models by using statistical metrics like accuracy, F_β-score, precision, recall, etc. With the higher scoring of these metrics, we shall be able to judge the success of these models in predicting the churn.

Datasets and Inputs

The datasets to be used in the project were provided by GP and contains browsing histories of users. Information such as how the dataset was obtained, and the characteristics of the dataset should be included as well. Basically, there will be two datasets. Both of them containing data for the month of June. There will be one dataset consisting of browsing history of people who did not practice the churn and the other one consisting of browsing history of people who did practice the churn. The features contained in all datasets are listed and explained bellow:

1. *Subscription_id*: An "id" to identify an user at the database;
2. *type*: "T" for titular account and "D" for dependent account member;
3. *Recency*: A variable that varies from 0 to 30. It measures how recent a user have visited the website, where 0 means that he never had visited in the last 30 days and 30 that he came yesterday;
4. *last_access_date*: Last day of user's access at the website;
5. *lst_date*: a string containing all the days that the user have accessed the website;
6. *subscription_date*: the precise day that the user have subscribed;
7. *freq*: How many days the user have visited the website;
8. *qt_page_view*: quantity of page views a user has;
9. *qt_page_view_week*: quantity of page views during the weeks;
10. *qt_page_view_weekend*: quantity of page views during the weekends;
11. *qt_page_view_mornig*: quantity of page views during the mornings;
12. *qt_page_view_afternoon*: quantity of page views during the afternoons;
13. *qt_page_view_nigth*: quantity of page views during the nights;
14. *qt_art_read*: quantity of articles that were read by a user;
15. *qt_source_sm_cpc_facebook*: how many times a user came to the website via a paid campaign from facebook;

16. *qt_source_sm_cpc_others*: how many times a user came to the website via a paid campaign from others social medias;

17. *qt_source_sm_cpc_others*: how many times a user came to the website via a paid campaign from others social medias;

18. *qt_source_sm_facebook*: how many times a user came to the website via facebook (no paid campaign);

19. *qt_source_sm_facebook*: how many times a user came to the website via facebook (no paid campaign);

20. *qt_source_sm_other*: how many times a user came to the website via other social media (no paid campaign);

21. *qt_source_nedeal*: how many times a user came to the website via actions from a outsourced;

22. *qt_source_email*: how many times a user came to the website via other social media (no paid campaign);

23. *qt_source_others*: how many times a user came to the website via other social media (no paid campaign);

24. *qt_gazeta_capa*: how many times a user accessed the website frontpage;

25. *qt_gazeta_esportes*: how many times a user accessed the website section "esportes";

26. *qt_gazeta_politica*: how many times a user accessed the website section "politica";

27. *qt_gazeta_economia*: how many times a user accessed the website section "esportes";

28. *qt_gazeta_curitiba*: how many times a user accessed the website section "curitiba";

29. *qt_agronegocio*: how many times a user accessed the website section "agronegocio";

30. *qt_haus*: how many times a user accessed the website section "haus";

31. *qt_bom_gourmet*: how many times a user accessed the website section "bom_gourmet";

32. *qt_viver_bem*: how many times a user accessed the website section "viver_bem"';
33. *qt_guia*: how many times a user accessed the website section "guia";
34. *qt_viver_bem*: how many times a user accessed the website section "viver_bem"';
35. *qt_access_others*: how many times a user accessed other pages of the website;
36. *qt_comments*: how many times a user have made some comment at the website comment section;
37. *qt_explicar*: how many times a user have read some article with the tag "explicar";
38. *qt_alegre*: how many times a user have read some article with the tag "alegre";
39. *qt_provocar*: how many times a user have read some article with the tag "provocar";
40. *qt_triste*: how many times a user have read some article with the tag "triste";
41. *qt_inspirar*: how many times a user have read some article with the tag "inspirar";
42. *qt_moderno*: how many times a user have read some article with the tag "moderno";
43. *qt_facilitar*: how many times a user have read some article with the tag "facilitar";
44. *qt_surpreendente*: how many times a user have read some article with the tag "surpreendente';
45. *qt_informar*: how many times a user have read some article with the tag "informar";
46. *qt_hardnews*: how many times a user have read some article with the tag "hardnews";
47. *qt_softnews*: how many times a user have read some article with the tag "softnews";

48. *qt_appGazeta*: how many times a user have used the app section "gazeta" to access the website;

49. *qt_appGuia*: how many times a user have used the app section "guia" to access the website;

50. *qt_mobile*: how many times a user have used a mobile to access the website;

51. *qt_other_devices*: how many times a user have used other devices rather than mobile to access the website;

52. *qt_login*: how many times a user have logged in;

53. *browser*: the user most common used browser to access the website;

54. *has_club*: if the user has club, which is another product of the newspaper media;

55. *has_print*: if the user has a printed version of the newspaper;

56. *Churn*: the output variable; if the user practiced the Churn or not.

Solution statement

We shall use some classification algorithms (Gradient Boosting Classifier, Ada Boost classifier, Random Forest Classifier etc.) to solve this problem. In order to do that, we first will solve the problem of imbalanced data because the number of people who have practiced the Churn is significantly smaller than the number of people who did not practice it. To solve the imbalanced problem, we shall introduce the RFV (Recency-Frequency-Volume) clustering method, which is an adaptation of the RFM (Recency-Frequency-Monetary) from e-commerce.

Benchmark models

There will be two benchmark models for this problem. The first one will be the Udacity first project "Finding Donors for CharityML" that it was submitted to Udacity, where we shall use techniques of analysis and so on. And the other one, it will be the RFV model mentioned before, where we shall compare the predictive power of our method with the clustering RFV model.

Evaluation metrics

GP is particularly interested in predicting who will practice the churn accurately. It would seem that using **accuracy** as a metric for evaluating a particular model's performace would be appropriate. Additionally, identifying someone that *did not* practice the churn as someone who does would be detrimental to GP, since they are looking to find individuals in order to take some action to prevent their churn. Therefore, a model's ability to precisely predict those that shall practice the churn is more important than the model's ability to recall those individuals. We will use $F - \beta$ score as a metric that considers both precision and recall:

$$F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot recall}{(\beta^2 \cdot precision) + recall}$$

In particular, when $\beta = 0.5$, more emphasis is placed on precision. This is called the $F_{0.5}$ score (or F-score for simplicity).

Looking at the distribution of classes (those who did not practice the churn, and those who make more), fortunately, it's clear most individuals do not practice than the ones who practice. This can greatly affect accuracy, since we could simply say "this person will not practice the churn" and generally be right, without ever looking at the data! Making such a statement is, of course, naive, since we have not considered any information to substantiate the claim. It is always important to consider the naive prediction for your data, to help establish a benchmark for whether a model is performing well. We recall that

1. **Accuracy**: measures how often the classifier makes the correct prediction. Its the ratio of the number of correct predictions to the total number of predictions (the number of test data points).

2. **Precision**: tells us what proportion of messages we classified as spam actually were spam. It is a ratio of true positives(words classified as spam, and which are actually spam) to all positives(all words classified as spam, irrespective of whether that was the correct classification), in other words it is the ratio of

$$\frac{\text{True Positives}}{(\text{True Positives} + \text{False Positives})}$$

3. **Recall(sensitivity)**: tells us what proportion of messages that actually were spam were classified by us as spam. It is a ratio of true

positives (words classified as spam, and which are actually spam) to all the words that were actually spam, in other words it is the ratio of

$$\frac{\text{True Positives}}{(\text{True Positives} + \text{False Negatives})}$$

Project Design

The workflow of solving this problem will be in the following order:

- Exploring the Data:
  - Loading libraries and Data;
  - Dimensions of the Data;
  - Statistical Summary;
- Data preprocessing and cleaning:
  - Preprocess feature columns;
  - Data cleaning;
  - Feature Scaling - Standardization, Normalizing data;
- Create the RFV model and create clusters;
- Evaluate Algorithms:
  - Build models;
  - Select best model;
  - Make predictions on the validation set;
  - Feature importance and feature selection;
- Final conclusions;