Using Association Rules for Concept Drifting

Dorin Keshales ID. 313298424

Eran Hirsch ID. 302620745

Submitted as final project report for Tabular Data Science, BIU, 2022

Abstract

Concept drifting refers to a change in the underlying data distribution, usually over time. While most approaches today advocate to revise the model frequently, we present a tool that surfaces concept distribution changes already in the Exploratory Data Analysis phase, using if-then association rules. An expert can use it to better understand the data and build better models, or use it periodically to track changes in the data distribution. We show our tool efficacy by utilizing the concept drifts found in a logistic regression model, testing it on multiple datasets collected from kaggle.

1 Problem Description

Exploratory Data Analysis (EDA) is a crucial step in the *Data Science Pipeline*. In the EDA process, we are interested in understanding the impact that one feature has on the dataset's distribution, such as *Concept Drifting*, where understanding these data distribution shifts is crucial for building models and developing an understanding of the data. Concept Drift is when the statistical properties of the predicted variable change in unexpected ways, making the prediction less accurate. Today, in order to handle concept drifting, peri-

odic retraining is necessary, but most of the time the retraining costs are very expensive.

Therefore, in this project we aim to develop a new tool for dealing with the Concept Drifting problem, using *Association Rules*. The ambition is that with our tool, the user will be able to pre-test features that he suspects may be causing Concept Drifting. For this purpose, our tool will be based on the Association Rules framework. Association rule mining, at a basic level, involves the use of machine learning models to analyze data for common patterns or co-occurrences within a database. It identifies frequent if-then associations, which themselves are the association rules. Association rules are created by searching for frequent if-then patterns in the data and using the criteria of *support* and *confidence* to identify the most important relationships. Support is an indication of how often the items appear in the data. Confidence indicates the number of times the if-then statements have been found to be true. A third metric, called *lift*, can be used to compare confidence with expected confidence, or how many times an if-then statement is expected to be found true.

2 Solution overview

2.1 General Approach

As stated in section 1, our ambition is to allow a user who uses our tool to pre-test features he suspects may cause Concept Drifting. In our vision for the tool we want to develop, we want it to answer the following anecdotes:

- First, the user will be prompted to enter a concept (feature) from his business point of view, such as the "year" of the house sale (an example from the House Prices dataset, which will be briefly presented in 3.1).
- As a response, the user will receive a detailed list of association rules that change over different values of the concept.
- Through the response, the user will know whether he should handle these

features in a certain way or not.

Note that we want to make our tool as efficient as possible for the user. Therefore, we would like to give the user the option to control the output by selecting different thresholds (i.e. what value of difference in confidence is considered an interesting rule, and different parameters to the Apriori algorithm like minimum confidence), which will be provided as parameters to the method.

2.2 Design

Following our vision, our tool consists of the following steps:

1. Get from the user the relevant input to our tool - the dataset, target variable, the concept (feature) he wants to test for Concept Drifting and, optionally, the relevant thresholds.

2. Preprocess the input - Before passing the input to our tool, we must first preprocess it . Thus, we wrote a preprocessing code file (preprocessing.py), which converts numerical values into ordinal values (creates bins for each of the numeric columns), cleans the data (for example, by filling N/A values in the data with the relevant column's mean value from the training set as well as dropping columns that 70% of them are N/A values), converts categorical variables into dummy/indicator variables, etc.

3. Split the dataset using different cutoff values of the concept (e.g., quantiles) - In order to do so, we built a CutoffValuesFinder module (cutoff_values_finder.py) which classifies the concept, given in the input, as discrete or continuous, and based on that it decides for the ConceptDriftsFinder module which concept values to try.

4. Find concept drifts - After finding the relevant cutoff values for the requested concept, we then use the ConceptsDriftsFinder module (concept_drifts_finder.py) to automatically find concept drifts by running the Apriori algorithm once on each subset. The find_concept_drifts function

in the module receives a list of transactions and returns a list of ConceptDriftResult objects.

5. Compare the lift values that the same association rules receives over different subsets.

6. Release a detailed response to the user. See Figure 1 as an example.

	left_hand_side	right_hand_side	confidence_before	confidence_after	support_before	support_after	lift_before	lift_after	concept_cutoff	concept_column
0	{'BldgType': '1Fam'}	{'SalePrice': 1}	1.000000	0.185229	1.000000	0.155359	1.000000	0.941887	2.800000	OverallQual
1	{'FullBath': 1}	{'SalePrice': 1}	1.000000	0.374718	0.600000	0.163225	1.000000	1.905440	2.800000	OverallQual
2	{'GrLivArea': 1}	{'SalePrice': 1}	1.000000	0.530000	1.000000	0.104228	1.000000	2.695050	2.800000	OverallQual
3	{'YearBuilt': 1}	{'SalePrice': 1}	1.000000	0.509615	0.800000	0.104228	1.000000	2.591394	2.800000	OverallQual
4	{'BldgType': '1Fam', 'FullBath': 1}	{'SalePrice': 1}	1.000000	0.361809	0.600000	0.141593	1.000000	1.839799	2.800000	OverallQual
5	{'FullBath': 1, 'GrLivArea': 1}	{'SalePrice': 1}	1.000000	0.548387	0.600000	0.100295	1.000000	2.788548	2.800000	OverallQual
6	{'BldgType': '1Fam'}	{'SalePrice': 1}	0.675325	0.142125	0.541667	0.119870	0.953400	0.960644	4.600000	OverallQual
7	{'FullBath': 1}	{'SalePrice': 1}	0.739130	0.312997	0.531250	0.127430	1.043478	2.115588	4.600000	OverallQual
8	{'BldgType': '1Fam', 'FullBath': 1}	{'SalePrice': 1}	0.721311	0.302941	0.458333	0.111231	1.018322	2.047617	4.600000	OverallQual
9	{'BldgType': '1Fam'}	{'SalePrice': 5}	0.193470	0.967742	0.161943	0.882353	1.117829	0.997067	8.200000	OverallQual
10	{'FullBath': 2}	{'SalePrice': 5}	0.283810	1.000000	0.150810	0.764706	1.639788	1.030303	8.200000	OverallQual
11	{'GrLivArea': 5}	{'SalePrice': 5}	0.582418	0.956522	0.107287	0.647059	3.365079	0.985507	8.200000	OverallQual
12	{'OverallCond': 5}	{'SalePrice': 5}	0.243902	0.967742	0.131579	0.882353	1.409214	0.997067	8.200000	OverallQual
13	{'BldgType': '1Fam', 'FullBath': 2}	{'SalePrice': 5}	0.344660	1.000000	0.143725	0.676471	1.991370	1.030303	8.200000	OverallQual
14	{'BldgType': '1Fam', 'GrLivArea': 5}	{'SalePrice': 5}	0.642424	0.956522	0.107287	0.647059	3.711785	0.985507	8.200000	OverallQual
15	{'BldgType': '1Fam', 'OverallCond': 5}	{'SalePrice': 5}	0.290557	0.964286	0.121457	0.794118	1.678773	0.993506	8.200000	OverallQual
16	{'FullBath': 2, 'OverallCond': 5}	{'SalePrice': 5}	0.311170	1.000000	0.118421	0.735294	1.797872	1.030303	8.200000	OverallQual
17	{'BldgType': '1Fam', 'FullBath': 2, 'OverallCond': 5}	{'SalePrice': 5}	0.385417	1.000000	0.112348	0.647059	2.226852	1.030303	8.200000	OverallQual

Figure 1: A list of concepts drift results found for the concept column OverallQual from the House Prices dataset.

3 Experimental Evaluation

3.1 Datasets

The usability and effectiveness of our proposed tool have been tested and demonstrated on the following 4 datasets:

1. **House Prices** [1] - The dataset contains 81 explanatory variables describing (almost) every aspect of residential homes in Ames, Iowa, in order to predict the final price of each home. The target variable is 'SalePrice'.

2. **Rain in Australia** [2] - The dataset contains about 10 years of daily weather observations from many locations across Australia. 'RainTomor-

row' is the target variable to predict. It means – did it rain the next day, Yes or No? This column is Yes if the rain for that day was 1mm or more.

3. **Big Mart Sales** [3] - The dataset provides the product details and the outlet information of the products purchased with their sales value. Sales of a given product at a retail store can depend both on store attributes as well as product attributes. 'OutletSales' is the target variable.

4. **Latest Netflix data with 26+ joined attributes** - The dataset combines data sources from Netflix, IMDB, posters, trailers on YouTube and more. 'Hidden Gem Score' is the target variable to predict, which is calculated using low review count and high rating. Lower the review count and higher the user rating, higher the hidden gem score.

3.2 Evaluation Metric

As will be explained in the following subsection - 3.3, we use logistic regression models to evaluate the efficiency of our tool. Thus, in order to calculate the score of the model, we use the score function of the logistic regression, which uses the mean accuracy as an evaluation metric of the model.

3.3 Evaluation of Our Approach

As an evaluation for our approach, we aim to show that building models according to the response of our newly created tool improves the results over previously unseen data. First, we will take several datasets and build a baseline model. Then, we will attempt to achieve higher accuracy over the same dataset, by creating an automatic feature engineering process that uses our tool's response. Our assumption is that if our found concepts are significant, then we can improve the accuracy of simple one-vs-rest logistic regression models.

We call the automatic feature engineering process *concept engineering*, and it works in two steps: (1) It runs over all the features in the dataset, and finds concepts using the concept drifts finder, discussed in §2. (2) For each concept,

it will then calculate the lift scores before and after the concept cutoff. It will then split the dataset into two based on the concept cutoff. Finally, for the split of the dataset where the lift score is higher, it will increase the weights of the features from the *left hand side* of the rule by the lift scores difference calculated earlier. The idea is that we allow the logistic regression model to give these features lower weight, by using the concepts to take into account the cases where they should get higher weight. These data transformations are applied separately to each label, based on the *right hand side* of the association rule (e.g., rules that entail SalePrice: 1 will only be applied on the one-vs-rest classifier that classifies the label 1).

3.4 Results

Dataset	housing	sales	rain	netflix
Baseline	62.328	48.963	**82.237**	74.075
Rules (ours)	**64.383**	**49.080**	81.899	**74.325**

Table 1: Validation results.

Our results are presented in Table 1. The House Prices dataset had the highest improvement of 2%, while other datasets had only minor increase or decrease in performance. We attribute this to the relative small size of the House Prices dataset - 1460 examples, compared to Big Mart Sales - 8523 examples, Netflix - 13344 examples, and Rain in Australia - 35257 examples. The larger datasets have inherently smaller lift score differences, which make the automatic changes insignificant. Also, error analysis showed that the feature engineering process we took creates a tendency for models to predict classes with more rules. To conclude, it is possible that this evaluation metric is not a good indication of the quality of our tool. Nonetheless, we see value in the project as part of the EDA process, as indicated in the qualitative analysis in 3.5.

3.5 Qualitative Analysis

We aim to see if our tool has any added benefits to the EDA process, as a manually activated tool. We can see in Figure 1 several concepts found for the concept column *OverallQual*. For example, we see that *FullBath: 1* has increased confidence when *OverallQual* is lower than *2.8*, and *FullBath: 2* has increased confidence when *OverallQual* is greater than *8.2*. This makes sense that when the quality is very low or very high, other features, like the number of baths, are more consistent. This is an insight that can be helpful for a data scientist trying to understand the quirks of the dataset.

In another example, depicted in Figure 2, it initially seems that *BldgType_1Fam* is not a strong indication of *SalePrice* (blue dots). However, when segmenting the data using *OverallQual*, based on the found concept drift result, we see that it can be a very strong indication (orange dot).

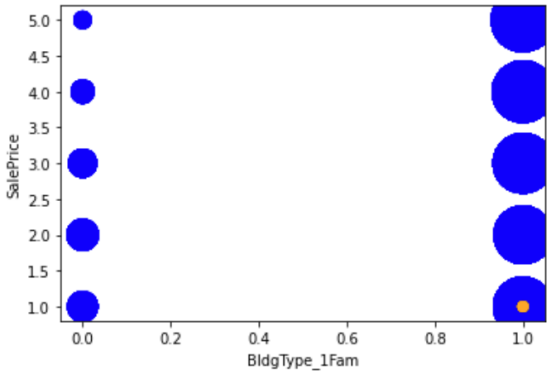

Figure 2: Blue dots represent the *BldgType_1Fam* / *SalePrice* scatterplot of the whole dataset, orange dots represent the same scatterplot after after segmenting the dataset based on the *OverallQual* found concept drift result.

4 Related Work

Concept drifting often refers to a gradual change in the data distribution over time, so the common solution is then to periodically revise the model [4]. To

avoid training a model from scratch every time, one approach is to create an ensemble of models, and on every revision simply deprecate old models from the ensemble while inserting new ones [5]. Other approaches advocate the importance of spatial data [6]. Rather than focusing on automatic retraining of the model, our approach focuses on observability, making the drifts visible in the EDA step, whether they already exist in the first model building iteration step or in a revision step. Our tool can then be used for the specific time-based drifts in a periodically manner to alert the model maintainer on new drifts.

While there are a lot of tools that find correlation between two features [7], our approach uses association rules that can have on the left hand side more complicated rules.

5 Conclusion

We created a new tool for finding concepts drifts, which can be useful in the EDA process to better understand the data. The automatic feature engineering approach is very simplistic and was only used to show the quality of the rules, we don't claim that this is a recommended approach, as we only tested it with simple LogisticRegression models. Collecting multiple baselines proved a useful and more reliable approach than testing with a single dataset. In future work, a real user with a business point of view can take this information for other more useful feature engineering approaches.

References

[1] Lisette. House Prices dataset. https://www.kaggle.com/datasets/lespin/house-prices-dataset.

[2] Joe Young. Rain in Australia dataset. https://www.kaggle.com/jsphyg/weather-dataset-rattle-package/version/2.

[3] Akashdeep Kuila. Big Mart Sales dataset. https://www.kaggle.com/akashdeepkuila/big-mart-sales.

[4] Jason Brownlee. A Gentle Introduction to Concept Drift in Machine Learning. https://machinelearningmastery.com/gentle-introduction-concept-drift-machine-learning/.

[5] Parneeta Sidhu and MPS Bhatia. A novel online ensemble approach to handle concept drifting data streams: diversified dynamic weighted majority. *International Journal of Machine Learning and Cybernetics*, 9(1):37–61, 2018.

[6] Anjin Liu. Concept drift adaptation for learning with streaming data. 2018.

[7] 8080labs. ppscore - a Python implementation of the Predictive Power Score (PPS).