Distinguishing Between Categorical and Numerical Data

Yair Shpitzer 313285942, Nerya Aberdam 311416457

March 22, 2022

Abstract

Distinguishing between numerical and categorical data is an important task for any data scientist during Exploratory Data Analysis (EDA). Unfortunately, when feature's values are integers, it requires an understanding of the data, which makes it hard to automate. Current solutions are not flexible enough, relying on a set of fixed heuristics. We suggest a solution based on a trained machine learning model, making it more adaptive to different datasets. However, Our experiments shows that we were unable to improve existing ways.

1 Problem Description

Exploratory Data Analysis (EDA) is the process of performing initial investigations on a data set in order to discover patterns, spot anomalies and check assumptions. This process consists of data cleaning, statistical tests and visualisations of the data. At the beginning of the EDA, it is important to differ between categorical and numerical data. This partition is crucial because we treat categorical and numerical data differently through all the data science pipeline, and we will give some examples for that:

- Using Kolmogorov–Smirnov (KS) test for testing goodness of fit can be used only for continuous probability distributions, hence it can not be used for categorical features (we use Chi-Squared instead).
- Checking for correlation between categorical features is done by Chi-Squared, while Pearson Correlation Coefficient is used for numeric features.
- Association rules help us discovering relations between features and they are relevant for categorical data only (unless the numeric feature is transferred to a categorical one, using binning).
- Most of machine learning algorithms do not get categorical data as input, which require some kind of encoding of the feature (e.g. one-hot encoding).

In some cases, dividing between categorical and numerical data is straightforward. For example, when the values of a feature are decimal, it is most certainly numeric. On the other hand, string values suggests that the feature is categorical. But this is often not the case, as features may have integers values also. Integer values may represent numeric values (e.g. age, height, amount, etc.) but can also define the feature's categories.

PassengerId	Survived	Pclass	...	Sex	Age	SibSp	...	Fare
1	0	3	...	male	22	1	...	7.25
2	1	1	...	female	38	1	...	71.2833
3	1	3	...	female	26	0	...	7.925
4	1	1	...	female	35	1	...	53.1
5	0	3	...	male	35	0	...	8.05

Table 1: First five entries of the Titanic dataset.

For example, Table 1 contains the first five entries of the famous Titanic dataset [1]. The 'Fare' feature contains decimal values, and it is indeed numeric. On the contrary, the values of the categorical feature 'Sex' are strings.

The features 'Survived', 'Pclass', 'Age' and 'SibSp' has integer values, making it hard to identify them as categorical or numerical. Understanding the data helps us to realize that 'Survived' and 'Pclass' are categorical, while 'Age' and 'SibSp' (which is the number of siblings/spouses aboard the Titanic the passenger had) are numeric.

As part of the effort of automating the EDA process, we try to find a solution that will be able to identify which of the 'integer features' are categorical and which are numeric. Current solutions are based on predefined rules (heuristics), which makes them less flexible to multiple kinds of datasets. Also, they rely mostly on the amount of unique values the feature has. We will discuss these solutions in more depth in Section 4. Our solution is based on a machine learning model, making it more adjustable to different datasets. Furthermore, our solution takes into account more characteristics of the feature (e.g. the min/max value of the feature). This was supposed to help the model to get a better understanding of the data, assisting it in the task of distinguishing between categorical and numerical data.

2 Solution Overview

2.1 Our Model

As we said in Sec. 1, We wanted out model to take into account more than just the absolute amount of unique values. The first property to consider is the size of the feature (i.e. the number of values the feature has, not only unique values). The motivation behind this is that an absolute amount of unique values can be considered large in some datasets but small in others. For example, if a certain feature has 100 values overall and 20 unique ones, it is more likely that this feature is numerical. On the other hand, if a feature has 100,000 values overall and 20 unique ones, the probability that the feature is categorical rises. However, there may still may be an absolute threshold that can help distinguishing

features, regardless of the size of the feature (taking a radical example, it is very unlikely that there is a feature with 100,000 categories, regardless the feature's size). For that reason we use both the absolute and the relative amount of unique values.

Other properties to consider are the standard deviation, the minimum and the maximum values of the feature. This helps us to understand the amount of dispersion of the feature. The reason it may be helpful for the model is that we believe that categorical features are usually less dispersed that numeric features. That is based on the assumption that when encoding categories to numeric values, people often choose small, successive numbers.

Our train dataset is taken from real datasets found online. We used the features that has only integer values and used a manual labeling determining for each feature if it is numeric or categorical.

Our model is built using the Keras framework. The architecture of our model is: one layer with 5 units and other layer with 1 unit with Sigmoid activation. We used in Adam optimizer and Binary Crossentropy for the loss.

2.2 Using NLP

2.2.1 Overview

When classifying a feature manually, something that comes in handy in many cases is the name of the feature. For example, it is straightforward that if the name has the prefix "num_of" then the feature is probably numeric. Also, some names are often used as numerical or categorical (e.g 'Age' and 'Sex', respectively). Therefore, We tried to use a Natural Language Processing (NLP) model that would determine if a feature is numerical or categorical by it's name. We then planned to take the level of confidence of the NLP model and using it in our solution. Unfortunately, the NLP model was unsatisfying and did not

yield sufficient results. For that reason, we did not include it in our solution.

2.2.2 NLP Model Explanation

In order to determine if a feature's name is numeric or categorical we used the pre-trained GloVe model [2], which is an unsupervised learning algorithm for obtaining vector representations for words. The train data we used is the labeled features dataset mentioned in Sec. 2.1. The learning process goes as follows:

Let x be the feature's name, D the train data and C the GloVe corpus (with size of 400,000 words). If $x \notin C$ then the model returns 'unkown query', meaning it cannot classify x. Otherwise, we calculate $\vec{d}$ for every $d \in D \cap C$ (when $\vec{d}$ is the embedding of d). Then we sum for each category (i.e. numeric or categorical) the dot product between $\vec{x}$ and $\vec{d}$ for every $d \in D \cap C$, and finally divide the sum in the amount of words in this category. The category of x is the one that is the max between these sums, meaning x is "closer" to this category.

2.2.3 Why the NLP Model Fails

Unfortunately, this solution does not work on a lot of real datasets. The reason is that many datasets contain features' names which are not a proper word, hence not part of the GloVe corpus. For example, the Titanic dataset [see 1] have features named: 'Pclass', 'SibSp' and 'Parch', none of them is a word in the English vocabulary. Accordingly, the model can not classify these features. This phenomenon is very common, making the model nearly insignificant (out of 1156 features tested, names of only 146 features belonged to the corpus - only 12.6%).

3 Experimental Evaluation

We tested our solution over four datasets taken from Kaggle [5]. For each dataset, we extracted only the features with integer values and manually la-

Type	Accuracy
threshold 5	0.72
threshold 10	0.75
threshold 15	0.77
threshold 20	0.74
unique values	0.74
our model	0.75

Figure 1: Accuracy values of every test type

beled them as numeric or categorical. Our metric for evaluating the solutions is accuracy. That means we check for each solution the percentage of features it was right about. We tested the results against two other methods:

1. **Simple Threshold.** Distinguishing between numeric and categorical features is done using a simple threshold (see Sec. 4 for more information). We tested over several thresholds - 5, 10, 15, 20. All the thresholds yielded similar results, ranging 0.72-0.77, threshold of 15 as the best of them.

2. **Unique Only.** As said in Sec. 2.1, we gave the model more properties than just the absolute amount of unique values. To check if these properties helped the model learn better, we trained the model (with the same configurations) by giving it only the amount of unique values.

It seems from the tests we performed that they all bring very similar results with a slight advantage to a threshold of 15 (see Fig. 1 and Fig. 2).

4 Related work

A simple solution [3] to the problem of classifying features as numerical or categorical is using a constant threshold for the amount of unique values in a feature. Let t the threshold, and $u(f)$ the number of unique values in feature f. If $u(f) > t$ then f is numerical, otherwise it is categorical. The main advantage of this approach is that it is very easy to use and to implement. However, setting a good threshold is tricky, as sometimes there are numeric features with

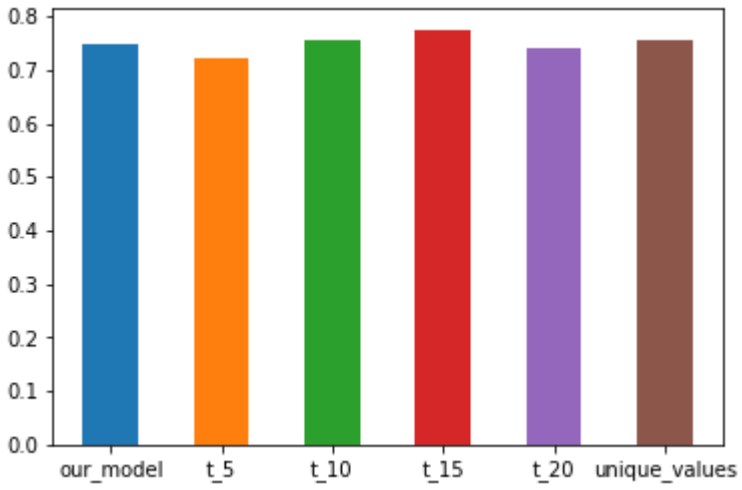

Figure 2: Accuracy of tests types

few unique values. On the other hand, there may be categorical features with relatively many categories.

Another solution is Bot, suggested by Majoor and Vanschoren [4]. Bot uses a set of heuristics in order to determine the type of the feature: (1) if the feature has less than 10 unique values it is considered categorical. (2) if the feature has more than 10 unique values, and less than a user-defined number k, then Bot calculate the average of absolute distances between integers. If this average is lower than the average of the integers itself, then the feature is categorical. Otherwise, the feature is numerical. Although this is a more complex solution than the previous one, it still suffers from the lack of flexibility.

We were inspired by Bot, which took into account also the average distance between integers. But unlike Bot (and the threshold solution), we suggested using a machine learning model, making our solution more robust and suitable to different datasets.

5 Conclusion

The task of identifying whether the feature is numeric or categorical is very difficult and no solution has yet been found. We wanted to offer a solution using two models:

1. **W**e did an NLP model in order to perform identification using the name of the feature.

2. **W**e have created a model that for each feature gets the amount of unique values, the percentage of unique values from the total amount of values in the feature, the standard deviation and the minimum and maximum values.

However, we have not been able to improve the existing ways of identifying the features. This is because using the NLP model to take advantage of feature names is very difficult to do, because in many cases feature names are not part of the English language. Although we were able to get percentages quite close to the existing ways in the model we presented in 2, but it could be that if we had more datasets with features that have integers or categorical, but here our information was limited.

References

[1] https://www.kaggle.com/c/titanic

[2] https://nlp.stanford.edu/projects/glove/

[3] Pieter Gijsbers. Arff-tools, 2017. https://github.com/openml/ARFF-tools/blob/master/csv-to-arff.py. 1, 7

[4] Angelo Majoor, J.V.: Auto-cleaning dirty data: The data encoding bot. Technical report, Technical University of Eindoven (2018), https://github.com/openml/ARFF-tools/blob/master/Data_Encoding_bot_Report.pdf [Accessed on 06/08/2020]

[5] 1. https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data

2. https://www.kaggle.com/sakshigoyal7/credit-card-customers

3. https://www.kaggle.com/borapajo/food-choices

4. https://www.kaggle.com/uciml/student-alcohol-consumption