

SAKI SS 2021 Homework 1

Author: Tim Löhner

Program code: https://github.com/Mavengence/SAKI_Homework_1

Summary

Background

This project aims to quantify how accurately bank transfer data can classify different labels. Those labels are for example standard of living or income. Different data from the bank transfer could have an indication for which label this payment aims for.

Objective

Using a real dataset from the master course Software Applications with Artificial Intelligence, I try to predict based on bank transfer data, how accurately I can predict a given label. For this project, I analyse all the features given to us and undergo different preprocessing techniques, because we have categorical data, numbers and also text.

I try out different combinations with the data and use the naive bayes classifier for the classification task.

Furthermore, the dataset is split into a train, test and validation set, because of the slight data imbalance.

In the end, the trained model will be evaluated based on the accuracy and F1 score, because we have a multiclass classification task which is imbalanced. Therefore, only the accuracy score is not sufficient enough to evaluate the result.

Data Preprocessing

The data has the shape of 209 rows and 11 features. One row has mostly nan values for the most important features, therefore I drop this row. Other than that, there exist 40 rows where the feature Auftragskonto is missing, but since I am using four features in total, I still kept those rows.

Features

For this project, I tried to understand the features for combining them to the best possible outcome. It appeared, that the combination of Buchungstext, Kontonummer, Betrag and Bankleitzahl works the best.

Buchungstext has 19 unique values, where 13 of them uniquely are used for one specific label. This means, that only six of the nineteen features classify multiple labels. This allows for precise predictions.

Kontonummer has 53 unique values, where 53 of them uniquely are used for one specific label. Same as for the Buchungstext, this is a very useful feature to train on.

Bankleitzahl has 44 unique values, where 40 of them uniquely are used for one specific label. Same as for the Buchungstext and Kontonummer, this is a very useful feature to train on.

Plus Minus Betrag is an artificially created feature in which I am using the sign of the feature Betrag. At first, I used a binned version of the 94 different Betrag values, but this decreased the accuracy.

Model Training

The model is the gaussian naïve bayes classifier provided from sklearn. This classifier offers one parameter to optimize, the variable smoothing. Due to the dataset being very small (208 rows), I am using a repeated stratified K-Fold algorithm in combination with a GridSearch for the optimization of the variable smoothing and to be sure, that the accuracy I achieve is stable and not due to a lucky shot very high.

Evaluation

Train, Test and Validation Sets

For the evaluation, I created a trainset, validationset and a testset. Because the dataset is slightly unbalanced towards the leisure and standard of living label, I created the testset in such a way, that it includes a completely balanced set. It consists of five rows per label, so 30 rows in total. The validation set has the same balance as the train set.

Features

I tried out most of the different possible features. I used for example a NLP pipeline to make use of the Verwendungszweck, but it only decreased the accuracy. The Valueatdatum also did not offer any increase in accuracy, therefore I dropped it to keep the features as few as possible. Also Beguenstigter/Zahlungspflichtiger did not improve the accuracy at all. For those text features, a more in-depth NLP needs to be done. I therefore used only categorical data that can easily be one-hot-encoded.

Accuracy

Train Accuracy: 1.0
Validation Accuracy: 0.92
Test Accuracy: 0.5

This result shows, that the data is overfitted towards the imbalanced data, because the balanced testset is not able to perform good. Even though we have six labels, an accuracy of 50% shows somewhat of a “training”, it is not nearly as good as the 92% of the validation set.

F1-Score

It is very interesting to see, that the imbalanced features leisure and standard of living perform the worst on both precision and recall. This makes total sense, due to the fact that the data is overfitted towards the imbalance. It makes therefore no sense to further elaborate on the F1-Score, because the error already lies in the trainset.

Result

My project showed, that I did the mistake of disregarding the imbalance of the trainset. The validation accuracy performs with 92% rather high, but the real evaluation shows, that this model is not ready to be used in the real world. I avoided the rebalancing in the first place, because if I used a balanced trainset, I either need to oversampling or undersampling. Oversampling is creating artificial features, especially for our case this generates heavily wrong results. Because we already have so few data, I did not want to undersample either. I assume, that if proper NLP is used, the accuracy can be heavily increased by creating new features with the text.

Screenshot

3.1 Train, Test and Validation Accuracy

```
In [34]: print(f"Train Accuracy: {accuracy_score(gs_nb.predict(X_train), y_train)}")
print(f"Validation Accuracy: {np.round(accuracy_score(gs_nb.predict(X_val), y_val), 2)}")
print(f"Test Accuracy: {np.round(accuracy_score(gs_nb.predict(X_test), y_test), 2)}")

Train Accuracy: 1.0
Validation Accuracy: 0.92
Test Accuracy: 0.5
```

```
In [35]: plt.hist(gs_nb.predict(X_test))
plt.title("Test Prediction Balance")
plt.xticks([0, 1, 2, 3, 4, 5], classes)
plt.show()
```

