Data Mining Assignment 1: Classification

Arbesa Jashanica

April 2023

1 Preprocessing

1.1 Filling in missing values

Values for the workclass, occupation and native-country column are sometimes missing for both the existing customers file and the potential customers file. Since these are categorical columns I used the SimpleImputer with strategy most_frequent to fill these values in.

1.2 Convert used features for yearly income prediction

For the predictions of the yearly income of potential clients I will use the features: age, education, education-num, occupation, race, sex, capital-gain, capital-loss, hours-per-week and native-country. To normalize and apply classifiers on our data we need to convert the categorical attributes to numerical, I used the Label Encoding approach for this.

1.3 Normalization

For the normalization of the data I used the MinMaxScaler(). I applied the normalization on the columns ['age', 'workclass', 'education', 'education-num', 'occupation', 'race', 'sex','capital-gain','capital-loss', 'hours-per-week', 'native-country']

1.4 Splitting data in train and test

I splitted the existing customers in test and train data. 30% of the data is used for test and the other 70% for train.

2 Prediction Income

For the prediction of the income I tried out several classifiers.

2.1 GaussianNB

For applying Gaussian Naïve Bayes', I used the build-in classifier GaussianNB() of scikit learn. This gave an accuracy of 0.8100112601085065, precision of 0.6917922948073701 and an Recall of 0.35695764909248057

2.2 KNeighborsClassifier

For KNN I used KNeighborsClassifier() in scikit. I checked which number of neighbors gave the best results using cross-validation. The best results gave 17 with an accuracy of 0.8318149247620023, precision of 0.6772319070258849 and recall of 0.554019014693172. the result of all different number of neighbors can be seen on the graph:

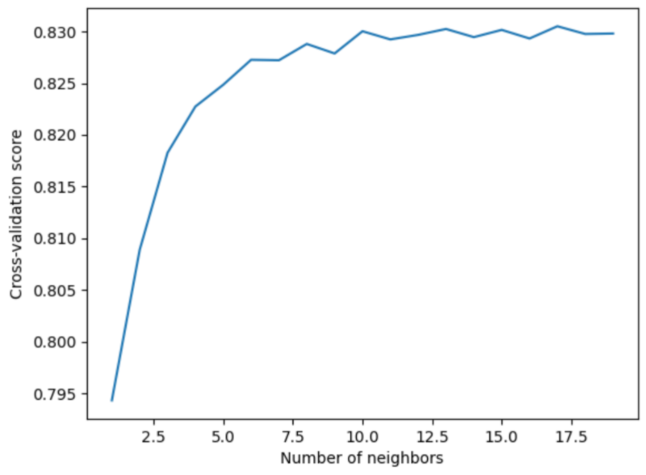

Figure 1: cross-validation score in function of number of neighbors

As can be seen on the graph, it is from 10 neighbors that the cross-validation score remains about the same.

2.3 DecisionTreeClassifier

For the decision tree classifier, I checked which depth gave the best results using Grid Search cross-validation. Depth 8 gave the best results with an accurancy of 0.8531067663015662, an precision of 0.7617629541393687 and recall of 0.5527225583405359.

2.4 BaggingClassifier

For the best results for bagging I checked which nulber of estimators gives the best results using Grid Search cross-validation. The best nulber of estimator is 100, which gives an result of 0.8460428292271741.

2.5 AdaBoostClassifier

For boosting I applied 2 classifiers, the AdaBoostClassifier with estimator the DecisionTreeClassifier with an max_depth of 8 and the GradientBoostingClassifier.
The AdaBoostClassifier with an number of estimators of 50 gives an accuracy of 0.8374449790152523, precision of 0.6668198529411765 and recall of 0.6270527225583405.
For the GradientBoostingClassifier classifier I checked which number of estimators gives the best results out of 50, 100, 150, 200 and 250 using Grid Search cross-validation. With an accuracy of 0.8699545166723919 gave 250 the best result.

2.6 RandomForestClassifier

For RandomForest I again checked which number of estimators the best results gave using Grid Search cross-validation. In this case it was 200, which gave an accuracy of 0.8496406675636223. But 100 as number of estimators wouldn't give much of a difference with an accuracy of 0.8497287337496161.

2.7 Conclusion

As we can see, GradientBoostingClassifier gives very good results. So we are using this classifier to predict whether or not a potential client has an high or low income. We predict 2910 as high income and 13371 as low income. To see which one the classifier predicted as low and which as high have a look in the file potential-customers-output.csv in the folder data in my public repository https://github.com/arbesajashanica/DataMining-Assignment1-Classification-.git . Rows indicated with an 0 as label are predicted as low and the ones with 1 are predicted as high.

3 To who are we sending the promotion?

To decide to who we are going to send an promotion, we will calculate the top 10% of the high-income potential customers who are nearest to the existing high-income customers. Because The potential clients that are the nearest are most similar to our existing customers so there is a big chance that they will accept our promotion. We do the same for low income but here we take the top 5%. For high-income this should be 291 rows in our case and for low-income this should be 669. To calculate the distance between the existing and potential clients, I used NearestNeighbors().

You can find the rowIDs of the potential customers to which we are going to send a promotion with an high income an low income in the files PotentialNewHighIncomeClients.txt and PotentialNewLowIncomeClients.txt in my folder. There are

4 Our revenue

To estimate our revenue we take our top 10% high income (291 clients) and 5% low income (669 clients) potential clients. The average profit of an high income client is 980 euro and a low income client cost us on average 310 euro. In total the high-income clients provide us Every promotion we send cost us 10 euro. So our calculation is :

Profit = (291 * 980) + (669 * (-310)) = 77790
TotalCost = (291 + 669)*10 = 9600
Expected gain = Profit - TotalCost = 68190

5 Code

You can find my code on the following public github repo: https://github.com/arbesajashanica/DataMining-Assignment1-Classification-.git