COBRA SMOTEBoost for Imbalanced Classification

MA691 (Advanced Statistical Algorithms)

COBRA-16

Dristiron Saikia (180101022)
Param Aryan Singh (180101055)
Parth Dhananjay Bakare (180101056)
Kaushal Chhalani (180123023)
Raunak Tiwari (180123038)
Mentored by Prof. Arabin Kumar Dey

Disclaimer

SMOTE

Imbalanced arrangement includes creating prescient models on order datasets that have an extreme class irregularity.

The test of working with imbalanced datasets is that most AI strategies will disregard, and thus have terrible showing on, the minority class, albeit ordinarily it is execution on the minority class that is generally significant.

One way to deal with addressing imbalanced datasets is to oversample the minority class. The easiest methodology includes copying models in the minority class, albeit these models don't add any new data to the model. All things considered, new models can be blended from the current models. This is a kind of information increase for the minority class and is alluded to as the Synthetic Minority Oversampling Technique, or SMOTE for short.

SMOTEBoost

SMOTEBoost is an oversampling technique dependent on the SMOTE algorithm. SMOTE uses k-nearest neighbors to make manufactured instances of the minority class. SMOTEBoost then, at that point, infuses the SMOTE method at each boosting iteration. The upside of this methodology is that while standard helping gives equivalent loads to all misclassified data, SMOTE gives more instances of the minority class at each boosting step.

COBRA Classifier

We use Cobra Classifier along with SMOTE Boost for the imbalance class problem. We use the Cobra Classifier reference https://github.com/bhargavvader/pycobra and then set our base model of SMOTE Boost. We run 6 instances of SMOTE Boost as our cobra base model. We implement the CobraClassifier class which has all of the required functions inside the class. We instantiate Cobra as ‘cobra’ and then call the predict function.

Results

We test our implementation of COBRA SMOTE Boost on the following dataset https://www.kaggle.com/jacintajacob/credit-card-fraud-detection/data .
Since the number dataset is very large, we use stratify split on the dataset and check the f1_score on the corresponding smaller dataset. Following are 3 instances.

For dataset of 14250 entries:

[85]:
```
x_train,new_x,y_train,new_y = train_test_split(df.drop('Class', axis=1),
                                               df['Class'],
                                               test_size=0.05,
                                               stratify= df['Class'],
                                               random_state=1)

print("stratified")
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(new_x,
                                               new_y,
                                               test_size=0.2,
                                               random_state=1)
```

```
stratified
```

[86]:
```
# print("split")
cc = ClassifierCobra(machine_list='cobra')

cc.fit(x_train, y_train)
print("Cobra SMOTE Boost f1 score: ", end="")
f1_score(y_test, cc.predict(x_test), average='weighted')
```

```
Cobra SMOTE Boost f1 score:
```

[86]:
```
1.0
```

For dataset of 42750 entries:

[61]:
```
x_train,new_x,y_train,new_y = train_test_split(df.drop('Class', axis=1),
                                               df['Class'],
                                               test_size=0.15,
                                               stratify= df['Class'],
                                               random_state=1)

print("stratified")
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(new_x,
                                               new_y,
                                               test_size=0.2,
                                               random_state=1)
print("split")
cc = ClassifierCobra(machine_list='cobra')

cc.fit(x_train, y_train)
print("Cobra SMOTE Boost f1 score: ", end="")
f1_score(y_test, cc.predict(x_test), average='weighted')
```

```
stratified
split
Cobra SMOTE Boost f1 score:
```

[61]:
```
0.9992978349912229
```

For dataset of 71250 entries:

```
[62]: x_train,new_x,y_train,new_y = train_test_split(df.drop('Class', axis=1),
                                                     df['Class'],
                                                     test_size=0.25,
                                                     stratify= df['Class'],
                                                     random_state=1)

      print("stratified")
      from sklearn.model_selection import train_test_split
      x_train,x_test,y_train,y_test = train_test_split(new_x,
                                                     new_y,
                                                     test_size=0.2,
                                                     random_state=1)
      print("split")
      cc = ClassifierCobra(machine_list='cobra')

      cc.fit(x_train, y_train)
      print("Cobra SMOTE Boost f1 score: ", end="")
      f1_score(y_test, cc.predict(x_test), average='weighted')

      stratified
      split
      Cobra SMOTE Boost f1 score:
[62]: 0.9990395686250492
```

Conclusion

From this work we conclude that SMOTE Boost works very well upon the imbalanced class problem. As it uses SMOTE along with Boosting accompanied by COBRA which improves the classification of minority class in case of Imbalanced class problem.

References

- What is imbalanced classification
- SMOTE for Imbalanced Classification with Python
- N. V. Chawla, K. W. Bowyer, L. O. Hall, and P. Kegelmeyer. "SMOTE: Synthetic Minority Over-Sampling Technique." Journal of Artificial Intelligence Research (JAIR), 2002.
- N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer. "SMOTEBoost: Improving Prediction of the Minority Class in Boosting." European Conference on Principles of Data Mining and Knowledge Discovery (PKDD), 2003.