JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 30, Number 11, 2023
© Mary Ann Liebert, Inc.
Pp. 1246–1249
DOI: 10.1089/cmb.2023.0191

A Python Package *itca* for Information-Theoretic Classification Accuracy: A Criterion That Guides Data-Driven Combination of Ambiguous Outcome Labels in Multiclass Classification

CHIHAO ZHANG,[1,2] SHIHUA ZHANG,[1,2] and JINGYI JESSICA LI[3]

ABSTRACT

The *itca* Python package offers an information-theoretic criterion to assist practitioners in combining ambiguous outcome labels by balancing the tradeoff between prediction accuracy and classification resolution. This article provides instructions for installing the *itca* Python package, demonstrates how to evaluate the criterion, and showcases its application in real-world scenarios for guiding the combination of ambiguous outcome labels.

Keywords: class label combination, information theory, noisy labels.

1. BACKGROUND

In real-world data sets, such as those in medical and biological applications, outcome labeling ambiguity and subjectivity are common challenges. Practitioners often resort to ad hoc methods to combine ambiguous labels for all data points to improve the accuracy of multiclass classification. However, there lacks a systematic and principled approach to guide the combination of ambiguous outcome labels. To fill this gap, we have developed the information-theoretic classification accuracy (ITCA) criterion (Zhang et al., 2022).

This article serves as a concise guide to using the *itca* Python package, which implements the ITCA criterion. For troubleshooting assistance or feedback, please visit the GitHub page (https://github.com/JSB-UCLA/ITCA), where additional documentation is available.

2. INSTALLATION

To install the *itca* package, users need Python version 3.6.8 or later. Running the following command in the terminal installs the package:

[1]Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, China.
[2]School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing, China.
[3]Department of Statistics and Data Science, University of California, Los Angeles, Los Angeles, California, USA.

```
pip install itca
```

After the installation, the package can be imported by running the following Python code:

```
import itca
```

3. CRITERION EVALUATION AND GUIDING AMBIGUOUS LABELS COMBINATION

ITCA evaluates a class combination by the performance of a specific classification algorithm on a data set whose labels have been combined according to the class combination. The data set consists of feature vectors and the corresponding outcome labels.

In the Python package, we have introduced a `bidict` class, which serves as a bidirectional dictionary representing a class combination. We denote the class combination as π_K, where K represents the number of classes after the combination has been applied. For example, if $K_0=4$ observed classes are combined into $K=3$ classes by merging the observed classes 2 and 3, then $\pi_3(0)=0$, $\pi_3(1)=1$, $\pi_3(2)=2$, and $\pi_3(3)=2$; π_3 can be represented using the `bidict` class as follows:

```
from itca import bidict
true_combination=bidict({0:0, 1:1, 2:2, 3:2})
```

In the following example code, we will simulate a data set, evaluate the ITCA criterion using the linear discriminant analysis (LDA) classification algorithm, and conduct a greedy search to determine the optimal class combination. In the `itca` package, the function for the ITCA criterion evaluation is `itca`, and the greedy search is encapsulated in a class called `GreedySearch`.

- We simulate $n=900$ observations from a three-component multivariate Gaussian mixture model, where the three Gaussian distributions have the same covariance matrix (i.e., the LDA model). Hence, the number of true classes is three. We randomly split the last true class into two classes by `bidict.reverse_map`. Consequently, the number of observed classes is $K_0=4$.

```
import numpy as np
from itca.simulator import SimLDA
np.random.seed(2023)
centriods=np.array([[1, 0, 0], [1, 0, -1], [0, 0.5, 0]])
A=np.random.randn(3, 3)
covariance=np.eye(3) + A @ A.T
sim_lda=SimLDA(centriods, covariance)
sim_data=sim_lda.gen(900)
y_observed=true_combination.reverse_map(sim_data.y)
```

- We split the data set into training and test data. With the observed classes (no combination), we first train the LDA classification algorithm on training data, and then evaluate ITCA of the trained classifier on test data.

```
from sklearn.model_selection import train_test_split
from sklearn.discriminant_analysis import
LinearDiscriminantAnalysis
from itca import itca
X_train, X_test, y_train, y_test=train_test_split(sim_data.X,
y_observed, test_size=0.2, random_state=2023)
no_combination=bidict({0:0, 1:1, 2:2, 3:3})
clf=LinearDiscriminantAnalysis()
clf.fit(X_train, y_train)
y_pred=clf.predict(X_test)
itca_score=itca(y_test, y_pred, no_combination)
print(''ITCA score: '', itca_score)
print(''Accuracy score: '', clf.score(X_test, y_test))
```

- We perform greedy search with LDA and fivefold cross-validation to find the optimal class combination defined by ITCA. First, an instance of the `GreedySearch` class is created with the desired parameters. In this case, the `class_type` parameter is set to nominal to indicate that the outcome label is categorical. After the search is complete, the selected class combination is `gs.selected.mapping`.

```
from itca import GreedySearch
clf=LinearDiscriminantAnalysis()
gs=GreedySearch(class_type='nominal')
gs.search(sim_data.X, y_observed, clf, verbose=False, early_
stop=True)
print(gs.selected.mapping)
```

In this example, the true class combination is found (i.e., the observed classes 3 and 4 are combined).

3.1. Application to single-cell RNA-seq data

An important application of ITCA is to detect similar class types. Here, we demonstrate how to detect biologically similar cell types in a *Hydra* single-cell RNA-seq data set (Siebert et al., 2019) and build a cell type hierarchy by greedy search with ITCA as the criterion. For the purpose of illustration, we will use the first 10 classes.

```
from itca.datasets import load_hydra_data
X, y, labels=load_hydra_data(); X=X[y<10, :]; y=y[y<10]
```

In the code above, X contains cells' gene expression vectors, y contains cells' annotated types (with numerical labels), and `labels` contains the names of the cell types.

Next, we will construct a cell type hierarchy using a greedy search approach. This involves iteratively combining classes until all classes are combined into one. To ensure the search continues until a single class is left, we set the `early_stop` parameter to `False`.

```
gs=GreedySearch(class_type=''nominal'')
gs.search(X, y, clf, early_stop=False, verbose=True)
```

To visualize the constructed hierarchy, one can use the plot_ascii_tree function.

```
from itca.visualize import plot_ascii_tree
plot_ascii_tree(gs)
```

4. A NOTE ON CLASSIFICATION ALGORITHM CHOICE

In addition to its adaptability to all classification algorithms, ITCA offers comparability across different classification algorithms. This allows users to choose the most suitable classification algorithm for their specific tasks. When it comes to prediction tasks, it is recommended to use a strong classification algorithm that maximizes ITCA. In contrast, when the goal is to detect similar classes, a weak classification algorithm such as LDA can be employed.

4.1. Software availability

The itca Python package is under the MIT License and available at https://github.com/JSB-UCLA/ITCA

ACKNOWLEDGMENTS

We acknowledge the valuable feedback and contributions provided by the reviewers, which have improved the quality of this study.

AUTHORS' CONTRIBUTIONS

C.Z. contributed to methodology, implementation, data collection, experimental results, and writing original draft. S.Z. was involved in supervision and writing review and editing. J.J.L. was in charge of conceptualization, methodology, supervision, and writing review and editing.

AUTHOR DISCLOSURE STATEMENT

The authors declare they have no conflicting financial interests.

FUNDING INFORMATION

This study was supported by the National Science Foundation DMS 2113754, Johnson & Johnson WiSTEM2D Award, Alfred Sloan Foundation (Sloan Research Fellowship), and W.M. Keck Foundation (UCLA David Geffen School of Medicine W.M. Keck Foundation Junior Faculty Award) to J.J.L. This study was also supported by the National Key R&D Program of China [2019YFA0709501], the Strategic Priority Research Program of the Chinese Academy of Sciences (CAS) [XDPB17], the Key-Area Research and Development of Guangdong Province [2020B1111190001], the National Natural Science Foundation of China [61621003], the National Ten Thousand Talent Program for Young Top-notch Talents, and the CAS Frontier Science Research Key Project for Top Young Scientist [QYZDB-SSW-SYS008] to S.Z.

REFERENCES

Siebert S, Farrell JA, Cazet JF, et al. Stem cell differentiation trajectories in hydra resolved at single-cell resolution. Science 2019;365(6451):eaav9314.

Zhang C, Chen YE, Zhang S, et al. Information-theoretic classification accuracy: A criterion that guides data-driven combination of ambiguous outcome labels in multi-class classification. J Mach Learn Res 2022;23(341):1–65.

Address correspondence to:
Dr. Shihua Zhang
Academy of Mathematics and Systems Science
Chinese Academy of Sciences
Beijing 100190
China

E-mail: zsh@amss.ac.cn

Dr. Jingyi Jessica Li
Department of Statistics and Data Science
University of California, Los Angeles
Los Angeles, CA 90095-1554
USA

E-mail: jli@stat.ucla.edu