



Generative AI for diabetologists: a concise tutorial on dataset analysis

Yoshiyasu Takefuji¹

Received: 17 November 2023 / Accepted: 18 December 2023
© The Author(s), under exclusive licence to Tehran University of Medical Sciences 2024

Abstract

Objectives This paper aims to provide a tutorial for diabetologists and endocrinologists on using generative AI to analyze datasets. It is designed to be accessible to those new to generative AI or without programming experience.

Methods The paper presents three examples using a real diabetes dataset. The examples demonstrate binary classification with the ‘Group’ variable, cross-validation analysis, and NT-proBNP regression.

Results The binary classification achieved a prediction accuracy of nearly 0.9. However, the NT-proBNP regression was not successful with this dataset. The calculated R-squared values indicate a poor fit between the predicted model and the raw data.

Conclusions The unsuccessful NT-proBNP regression may be due to insufficient training data or the need for additional determinants. The dataset may be too small or new metrics may be required to accurately predict NT-proBNP regression values. It is crucial for users to verify the generated codes to ensure that they can achieve their desired objectives.

Keywords Generative AI · Diabetologists and endocrinologists · Dataset analysis · Machine learning

Introduction

With the advent of generative AI, the effectiveness has been demonstrated in a variety of areas. ChatGPT is a generative language model tool by OpenAI that allows users to converse with a machine on various topics [1]. Eysenbach with ChatGPT demonstrated its potential for medical education, research, and practice, but also revealed its limitations [1]. JMIR Medical Education was launching a call for papers for a new e-collection and theme issue on ChatGPT [1].

Review by Sharma et al. explored the potential of ChatGPT as a substitute for diabetes educators [2]. ChatGPT is an AI technology that offers personalized and interactive education and support [2]. Their review discussed the challenges facing diabetes educators, the evidence supporting ChatGPT’s use in diabetes education, ethical considerations, and recommendations for further research and development [2].

Arora et al. discussed the potential of generative AI, including generative adversarial networks, to revolutionize clinical research and protect patient privacy by creating

✉ Yoshiyasu Takefuji
takefuji@keio.jp

¹ Faculty of Data Science, Musashino University, 3-3-3 Ariake Koto-Ku, Tokyo 135-8181, Japan

synthetic patient data [3]. Their paper focused on three key uses of synthetic data: clinical research, data privacy, and medical education, while also highlighting ethical and practical concerns.

Khan et al. discussed the potential of AI to improve healthcare, but also raises concerns about data security and privacy [4]. The lack of standard guidelines for the ethical use of AI in healthcare exacerbates these concerns. Their study highlighted the importance of maintaining the confidentiality of medical records and suggests solutions to address potential drawbacks.

This paper provides a concise tutorial for diabetologists and endocrinologists on how to analyze datasets using generative AI. The paper includes three examples using a publicly available diabetes dataset (https://dfzljdn9uc3pi.cloudfront.net/2022/14078/1/raw_data.csv) and offers a set of easy-to-follow instructions for those new to generative AI or without programming experience.

There are three generative AI systems via browser for public use free of charge. This paper uses Bing.com with ChatGPT-4: <https://bing.com/chat>

Methods and results

Python should be installed on your PC and run the generated code on the terminal. With generative AI, the dataset and query are crucial components in the analysis of data. When accessing Bing.com web site, choose a conversation style with 'More Precise'. In the current generative AI, the same query may not produce the same result due to pseudorandom numbers. When analyzing the dataset, column names are important. Note that the dataset is composed of 45 columns such as 'Group', 'id', 'Gender',..., 'NT-proBNP', and so on. The dataset (https://dfzljdn9uc3pi.cloudfront.net/2022/14078/1/raw_data.csv) is composed of 553 instances of patients in the Department of Cardiology at a tertiary hospital in Anhui Province. Of these patients, 201 had coronary angiography performed during hospitalization with 'Group' = 1 and 352 had non-coronary heart disease with 'Group' = 0.

The first query is to calculate binary prediction accuracy of 'Group' with determinants and to show feature-importances in descending order.

Query1: show a full code in Python to classify 'Group' with 0 or 1 using all determinants except 'id' in raw_data.csv. Calculate the prediction accuracy and feature importances in descending order.

Answer1: Generative AI generates the following Python code.

```
import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split

# Load the data
data = pd.read_csv('raw_data.csv')

# Separate the features and target variable
X = data.drop(['id', 'Group'], axis=1)
y = data['Group']

# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=0)

# Create a random forest classifier
clf = RandomForestClassifier(n_estimators=100)

# Train the classifier
clf.fit(X_train, y_train)

# Make predictions on the test set
y_pred = clf.predict(X_test)

# Calculate the accuracy of the predictions
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy:.2f}')

# Calculate feature importances in descending order
importances = clf.feature_importances_
indices = np.argsort(importances)[::-1]
for i in indices:
    print(f'{X.columns[i]}: {importances[i]:.2f}')
```

This paper demonstrates the execution of the generated Python code across various operating systems, including Linux, Windows, and MacOS. Save the generated source code text as q1.py. To execute the q1.py code, open the terminal on your operating system. The '\$' sign represents the terminal prompt. Download the raw_data.csv file using the wget command, which is specifically designed for downloading files. Please note that the 'wget' command might not be available by default on some operating systems like Windows and MacOS. You might need to install it separately or use alternative commands to download files.

```
$ wget
https://dfzljdn9uc3pi.cloudfront.net/2022/14078/1/raw_data.csv

$ python q1.py
```

The result will be displayed after executing the Python code, q1.py.

```
Accuracy: 0.92
NT-proBNP: 0.20
FT3: 0.11
AST: 0.07
...
```

The first example shows the binary classification with the dataset. The result shows that NT-proBNP is the highest feature importance. The prediction accuracy is 0.92. The code generation and executing the code was successful.

Query2: show a full code in Python to classify 'Group' with 0 or 1 with all determinants except 'id' in raw_data.

csv. With cross validation, Calculate the prediction accuracy and feature importances in descending order. Due to fivefold cross-validation, prediction accuracy decreases.

Answer2: Generated Python code is as follows.

```
import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split, cross_val_score

# Load the data
data = pd.read_csv('raw_data.csv')

# Separate the features and target variable
X = data.drop(['id', 'Group'], axis=1)
y = data['Group']

# Create a random forest classifier
clf = RandomForestClassifier(n_estimators=100)

# Calculate the cross-validated accuracy
accuracy = np.mean(cross_val_score(clf, X, y, cv=5))
print(f'Cross-validated accuracy: {accuracy:.2f}')

# Fit the classifier on the entire dataset
clf.fit(X, y)

# Calculate feature importances in descending order
importances = clf.feature_importances_
indices = np.argsort(importances)[::-1]
for i in indices:
    print(f'{X.columns[i]}: {importances[i]:.2f}')
```

Similarly, to execute the q12.py code, open the terminal on your operating system with the following command.

```
$ python q2.py
```

The following result will be displayed after executing the code, q2.py.

Cross-validated accuracy: 0.88

NT-proBNP: 0.22

FT3: 0.10

AST: 0.06

TSH: 0.05

The second example shows the cross-validation for binary classification. NT-proBNP is the highest feature importance. Cross-validation accuracy is lower than that in Query1.

Query3: show a full code in Python to predict the value of 'NT-proBNP' with all determinants except 'id' in raw_data.

csv using 438 trees of random forest. calculate feature importances of determinants in descending order. Calculate prediction accuracy and r-squared with 'Group'=1 and 'Group'=0 using the generated model respectively.

Answer3: Generated Python code is as follows.

```
import pandas as pd
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error, r2_score

# Load the data
data = pd.read_csv('raw_data.csv')

# Define the target variable and the features
y = data['NT-proBNP']
X = data.drop('NT-proBNP', axis=1)

# Split the data into training and validation sets
X_train, X_valid, y_train, y_valid = train_test_split(X, y,
random_state=0)

# Train a Random Forest Regressor model with 438 trees
model = RandomForestRegressor(n_estimators=438,
random_state=0)
model.fit(X_train, y_train)

# Calculate feature importances in descending order
importances = pd.Series(model.feature_importances_,
index=X.columns)
importances = importances.sort_values(ascending=False)
print('Feature importances:', importances)

# Make predictions and evaluate the model's prediction accuracy
and r-squared for 'Group'=1 and 'Group'=0
for group in [0, 1]:
    group_mask = (X_valid['Group'] == group)
    predictions = model.predict(X_valid[group_mask])
    mae = mean_absolute_error(y_valid[group_mask], predictions)
    r2 = r2_score(y_valid[group_mask], predictions)
    print(f'Mean absolute error for Group={group}:', mae)
    print(f'R-squared for Group={group}:', r2)
```

The following result will be shown after executing the Python command.

```
$ python q3.py
```

Feature importances:

```
CRP          0.172099
```

```
Blood calcium  0.143529
```

```
LVEF          0.125912
```

```
...
```

```
Mean absolute error for Group=0: 314.462
```

```
R-squared for Group=0: 0.410
```

```
Mean absolute error for Group=1: 826.546
```

```
R-squared for Group=1: 0.289
```

The final example illustrates the regression of NT-proBNP using the given dataset. R-squared values were computed to assess the fit between the predicted model and the raw data. The results suggest a poor fit between the two. For Group=0, the r-squared value is higher than that for Group=1. This implies that the given determinants are inadequate for accurately predicting NT-proBNP regression values. In other words, either the dataset is too small or additional determinants are required to improve the prediction of NT-proBNP regression values.

Discussions and conclusion

This paper presents three examples using a real diabetes dataset to demonstrate binary classification with the 'Group' variable, cross-validation analysis, and NT-proBNP regression for novice and non-programmers. The binary classification achieved a prediction accuracy of nearly 0.9. However, the NT-proBNP regression was not successful with this dataset, possibly due to insufficient training data or the need for additional determinants. In other words, the dataset may be too small or new metrics may be required to accurately predict NT-proBNP regression values. The calculated R-squared values indicate a poor fit between the predicted model and the raw data, suggesting that the regression was not successful. It is crucial for users to verify the generated codes to ensure that they can achieve their desired objectives.

Funding This research has no fund.

Data availability Not applicable.

Declarations

Competing interests The author has no conflict of interest.

References

1. Eysenbach G. The role of ChatGPT, generative language models, and artificial intelligence in medical education: a conversation with ChatGPT and a call for papers. *JMIR Med Educ.* 2023;9:e46885. <https://doi.org/10.2196/46885>.
2. Sharma S, Pajai S, Prasad R, Wanjari MB, Munjewar PK, Sharma R, Pathade A. A critical review of ChatGPT as a potential substitute for diabetes educators. *Cureus.* 2023;15(5):e38380. <https://doi.org/10.7759/cureus.38380>.
3. Arora A, Arora A. Generative adversarial networks and synthetic patient data: current challenges and future perspectives. *Future Healthc J.* 2022;9(2):190–3. <https://doi.org/10.7861/fhj.2022-0013>.
4. Khan B, Fatima H, Qureshi A, Kumar S, Hanan A, Hussain J, Abdullah S. Drawbacks of artificial intelligence and their potential solutions in the healthcare sector. *Biomed Mater Devices (New York, N.Y.).* 2023; 1–8. <https://doi.org/10.1007/s44174-023-00063-2>. (Advance online publication).

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.