MATICS: Jurnal Ilmu Komputer dan Teknologi Informasi ISSN : 1978-161X(p); 2477-2550(e)
(*Journal of Computer Science and Information Technology)* *Volume* 14, No. 2 (2022), pp 70-73
Received: March 19th 2022; **Accepted**: September 19th 2022; **Avalaible Online**: October 15th 2022

mVIF Package: A Tool for Detecting Multicollinearity without Dependent Variables

Angga Dwi Mulyanto

***Abstract*— This article discusses the Variance Inflation Factor (VIF), a tool used to test the assumption of non-multicollinearity in regression analysis. VIF measures the correlation between variables in a regression model and its impact on the accuracy of analysis results. The article highlights that VIF can also be used to determine the presence of multicollinearity among variables in various types of analyses, including Hierarchical Cluster Analysis. While there are several programs or packages available to calculate VIF, they usually require a dependent variable input. To address this issue, the author aims to create a new package using Python to calculate VIF without the need for a dependent variable input. The program calculates VIF using the sequential elimination method, which involves removing one variable at each iteration of the for loop. In use, the user needs to input data in the form of a matrix, and the program will return a list of VIFs and information about the presence of multicollinearity in the data. The program provides an alternative method for evaluating multivariate data and the presence of multicollinearity, making the testing process easier and faster for data analysts and researchers.**

***Index Terms*—Variance Inflation Factor (VIF), Multicollinearity, Python package for VIF calculation**

I. Introduction

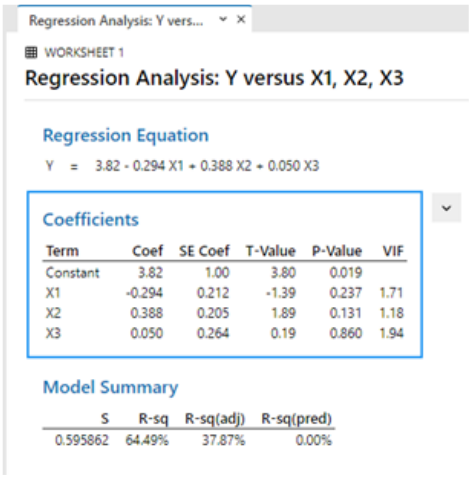
THE Variance Inflation Factor (VIF) is a tool used to test the assumption of non-multicollinearity in regression analysis [1]–[3]. VIF measures the extent of correlation between variables in a regression model and the impact of such correlation on the uncertainty or inaccuracy of the analysis results. While initially used for regression analysis, VIF can also be used to determine the presence of multicollinearity among variables in various types of analyses.

Manuscript received March 22, 2007. This work was supported in part by Program Studi Matematika, Universitas Islam Negeri Maulana Malik Ibrahim Malang.

Angga Dwi Mulyanto, Author is with Program Studi Matematika, Universitas Islam Negeri Maulana Malik Ibrahim Malang, Indonesia (corresponding author provide phone 08123467041; email angga.dwi.m@mat.uin-malang.ac.id)

One example of an analysis besides regression that utilizes VIF is Hierarchical Cluster Analysis [4]–[7]. This analysis is a data clustering method used to group objects based on similarities or differences in certain characteristics. In hierarchical cluster analysis, one assumption is the absence of multicollinearity among variables, so VIF is also used to evaluate this assumption.

There are many programs or packages in specific programming languages that can calculate VIF, such as Minitab [8] or the CAR package in R[9]. However, common multicollinearity tests like Minitab (Figure 1) and CAR initially require a dependent variable in regression assumption testing. Additionally, the Statmodels package in Python has a function to calculate VIF, although it does not require a dependent variable. However, Statmodels results may differ from those produced by Minitab, as seen in Figure 2.


Regression Analysis: Y vers...

WORKSHEET 1

Regression Analysis: Y versus X1, X2, X3

Regression Equation

Y = 3.82 - 0.294 X1 + 0.388 X2 + 0.050 X3

Coefficients

Term	Coef	SE Coef	T-Value	P-Value	VIF
Constant	3.82	1.00	3.80	0.019	
X1	-0.294	0.212	-1.39	0.237	1.71
X2	0.388	0.205	1.89	0.131	1.18
X3	0.050	0.264	0.19	0.860	1.94

Model Summary

S	R-sq	R-sq(adj)	R-sq(pred)
0.595862	64.49%	37.87%	0.00%

Fig. 1. VIF result using Minitab

```
[1]: import pandas as pd
data = pd.read_csv('data.txt', delimiter = "\t")
#print(data)
datanp = data.values
from statsmodels.stats.outliers_influence import variance_inflation_factor
a=variance_inflation_factor(datanp,0)
b=variance_inflation_factor(datanp,1)
c=variance_inflation_factor(datanp,2)
print(a,b,c)
```

```
12.404743490590356 3.3758700696055692 11.612786800721837
```

Fig. 2. VIF result using Package Statmodels in Python

As a solution to address these discrepancies, the author aims to create a new package using Python to calculate VIF without the need for a dependent variable input. The goal of this package is to produce the same output as Minitab, making the testing process for non-multicollinearity assumptions in data analysis easier and faster.

II. REASEARCH METHOD

In this study, the author used the Research and Development (R&D) approach. R&D research is a type of research aimed at producing a specific product. In this study, the product is a program to test multicollinearity using the VIF method. This program was built using the Python language.

III. RESULT AND DISCUSSION

To determine the presence of multicollinearity, one way is to measure it using VIF. It is considered that there is no multicollinearity if the VIF value is < 5. To obtain the VIF value, it is actually very easy if we have learned multiple regression calculations. For example, if we have 3 variables A, B, and C, and we want to obtain the VIF value of variable A, the steps are as follows:

1. Form a matrix X of size n x (k+1), where k is the number of remaining variables other than the variable whose VIF is being sought. In this example case, k=2, so the matrix size becomes n, where n is the number of observations. Fill the first column with all ones. Then fill the second column with data from variable B and the third column with data from variable C.
$$X = \begin{bmatrix} 1 & b_1 & c_1 \\ \vdots & \vdots & \vdots \\ 1 & b_n & c_n \end{bmatrix} \quad (1)$$
2. Form a Y matrix with size n x 1, where Y matrix contains data of the variable for which the VIF value will be calculated (in this example it is variable A).
$$Y = \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} \quad (2)$$
3. Create a matrix I which is an identity matrix with size n x n.
$$I = \begin{bmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1 \end{bmatrix} \quad (3)$$
4. Create a matrix J which is a matrix with all values equal to one and size n x n.
$$J = \begin{bmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{bmatrix} \quad (4)$$
5. Calculate the matrix H using the following formula:
$$H = X(X'X)^{-1}X' \quad (5)$$
6. Calculate the Sum of Squares Total (SST) value using the following formula:
$$SS_T = Y'\left(I - \frac{1}{n}J\right)Y \quad (6)$$
7. Calculate the Sum of Squares Regression (SSR) value using the following formula:
$$SS_R = Y'\left(H - \frac{1}{n}J\right)Y \quad (7)$$
8. Calculate the R^2 value using the following formula:
$$R^2 = \frac{SS_R}{SS_T} \quad (8)$$
9. Calculate the VIF value using the following formula:
$$VIF = \frac{1}{(1-R^2)} \quad (9)$$

Based on steps 1 to 9, a Python program code can be built which can be seen in Table 1.

Table 1. Source Code mVIF.py

```
import numpy as np
def VIF(data):
   VIF=[]
   k=len(data[1,:])
   n=len(data[:,1])
   j=0
   for i in range(0,k):
      satu=np.ones((n,1))
      x=np.delete(data,i,1)
      x=np.hstack((satu,x))
      y=data[:,i:i+1]
      mi=np.identity(n)
      mj=np.ones((n,n))

mh=(x.dot(np.linalg.inv((x.transpose()).dot(x)))).dot(x.transpose())
      sst=((y.transpose()).dot(mi-(1/n*mj))).dot(y)
      ssr=((y.transpose()).dot(mh-(1/n*mj))).dot(y)
      R2=ssr/sst
      vif=1/(1-R2)
      VIF.append(vif[0][0])
      if vif > 5:
         j=1
   print(VIF)
   if j == 1:
      print("Karena terdapat VIF yang lebih dari 5 maka dapat disimpulkan terjadi multikolinieritas.")
   else:
      print("Karena semua VIF kurang dari sama dengan 5 maka dapat disimpulkan tidak terjadi multikolinieritas.")
   print("VIF ini dihitung menggunakan code pyhton yang dibuat oleh Angga Dwi Mulyanto")
   return(VIF)
```

The following is the explanation of the Python code provided:

Table 2. Part 1

```
import numpy as np
```

This line imports the numpy library which is used for mathematical operations and arrays.

Table 3. Part 2

```
def VIF(data):
```

This initializes an empty list called "VIF".

Table 4. Part 3

```
k=len(data[1,:])
n=len(data[:,1])
```

These two lines calculate the number of columns and rows in the input matrix, respectively.

Table 5. Part 4

```
j=0
```

This initializes a variable called "j" with the value of 0.

Table 6. Part 5

```
for i in range(0,k):
```

This starts a loop that iterates through the columns of the input matrix.

Table 7. Part 6

```
x=np.delete(data,i,1)
x=np.hstack((satu,x))
y=data[:,i:i+1]
```

These three lines create three matrices:

- "x" is the input matrix with the i-th column removed and the "satu" matrix added as the first column.
- "y" is a column vector that contains only the i-th column of the input matrix.
- "satu" is the matrix of ones.

Table 8. Part 7

```
mi=np.identity(n)
mj=np.ones((n,n))
```

These two lines create two matrices:

- "mi" is the n-by-n identity matrix.
- "mj" is an n-by-n matrix filled with ones.

Table 9. Part 8

```
mh=(x.dot(np.linalg.inv((x.transpose()).dot(x)))).dot(x.transpose())
```

This calculates the matrix H (Hat matrix) using the input matrix "x".

Table 10. Part 9

```
sst=((y.transpose()).dot(mi-(1/n*mj))).dot(y)
ssr=((y.transpose()).dot(mh-(1/n*mj))).dot(y)
```

These two lines calculate the total sum of squares (SST) and regression sum of squares (SSR), respectively.

Table 11. Part 10

```
R2=ssr/sst
vif=1/(1-R2)
```

These two lines calculate the R-squared and VIF values, respectively.

Table 12. Part 11

```
VIF.append(vif[0][0])
```

This appends the calculated VIF value to the list "VIF".

Table 13. Part 12

```
if vif > 5:
    j=1
```

This checks if the calculated VIF value is greater than 5. If it is, the value of "j" is changed to 1.

Table 14. Part 13

```
print(VIF)
if j == 1:
    print("Karena terdapat VIF yang lebih dari 5 maka dapat disimpulkan terjadi multikolinieritas.")
else:
    print("Karena semua VIF kurang dari sama dengan 5 maka dapat disimpulkan tidak terjadi multikolinieritas.")
print("VIF ini dihitung menggunakan code pyhton yang dibuat oleh Angga Dwi Mulyanto")
 return(VIF)
```

This block of code prints out the list of VIF values, and checks whether there is any VIF value that is greater than 5. If there is, it prints out a message saying that there is multicollinearity. Otherwise, it prints out a message saying that there is no multicollinearity. Finally, it returns the list of VIF values.

In order to run the program, the following prerequisites are required:

1. Python 3 is installed on the computer (installation can be obtained at python.org)
2. Numpy package is installed on the computer (can be executed on cmd by pip install numpy)

Assuming we have a data.txt file with the following contents:

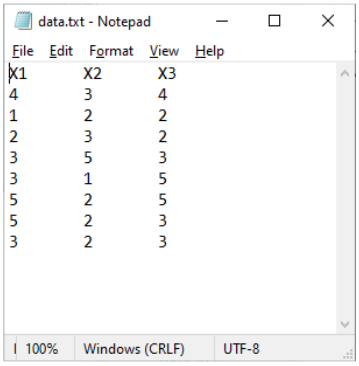


Fig. 3. Example of Data

Make sure data.txt and mVIF.py are in the same folder, then execute the command in table 1 which is saved as main.py and placed in the same folder as mVIF.py and data.txt. Running with code from Table 15.

Table 15. Running

```
import pandas as pd

import mVIF
data = pd.read_csv('data.txt', delimiter = "\t")
datanp = data.values

mVIF.VIF(datanp)
```

An example result of running the mVIF program can be seen in Figure 4.

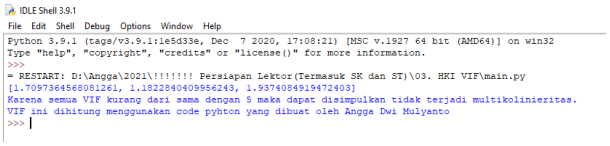


Fig. 4. Result mVIF

After we validate it with the results in Minitab, it produces the same values.

The program is an implementation of Variance Inflation Factor (VIF) calculation on multivariate data using the Python programming language. VIF is a method used to evaluate the correlation influence between each variable to multiple linear regression model.

The program begins by defining a function called VIF that takes the data argument as input. The VIF variable is initialized as an empty list, then the variables k and n are calculated as the number of columns and rows in the data. The j variable is initialized as 0 and will be changed to 1 if there is a VIF greater than 5, indicating the presence of multicollinearity in the data.

A for loop is used for each variable in the data. In each iteration of the for loop, the variable satu is initialized as a one matrix of size n x 1. The x variable is calculated by removing the i-th variable column from the data and adding a one column at the beginning, so x is of size n x k. The y variable is initialized as a matrix with one column containing the data from the i-th variable.

The identity matrix of size n x n (mi) and the ones matrix of size n x n (mj) are calculated. The H matrix is

calculated using the multiple linear regression formula, which results in a matrix of size n x n. The Sum Square Total (SST) and Sum Square Regression (SSR) values are calculated using the appropriate formulas. The R2 and VIF values are calculated using the appropriate formulas. The VIF value is added to the VIF list.

If there is a VIF greater than 5, the j variable is changed to 1. The program prints the VIF calculation results and provides information on whether there is multicollinearity in the data or not. The program returns a VIF list.

Note that the satu, x, and y variables function to form the X and Y matrices in VIF calculations. The mi and mj variables function to calculate the I and J matrices in SST and SSR calculations. The VIF calculation in this program is performed by removing one variable in each iteration of the for loop, then calculating the VIF for that variable. This method is called sequential elimination.

IV. CONCLUSION

Based on the previous program explanation, it can be concluded that the program is an implementation of the calculation of the Variance Inflation Factor (VIF) on multivariate data using the Python programming language. This program can be used to detect the presence of multicollinearity in data.

The program calculates VIF using the sequential elimination method, which involves removing one variable at each iteration of the for loop, then calculating the VIF for that variable and providing information about the presence of multicollinearity in the data.

In use, the user needs to input data in the form of a matrix. The program will return a list of VIFs and print the results of the VIF calculations along with information about the presence of multicollinearity in the data.

In conclusion, this program is very useful for data analysts and researchers who want to evaluate multivariate data and evaluate the presence of multicollinearity in data without the need for a dependent variable as in regression analysis.

ACKNOWLEDGMENT

I would like to state that this research was self-funded without any external support or sponsorship from any organization. I am grateful for the opportunity to conduct this research independently without any external influence. I also express my gratitude to our colleagues who provided valuable input and feedback during the course of this study.

REFERENCES

[1] R. K. H. Galvão and M. C. U. Araújo, “3.05 - Variable Selection,” in Comprehensive Chemometrics, S. D. Brown, R. Tauler, and B. Walczak, Eds., Oxford: Elsevier, 2009, pp. 233–283. doi: https://doi.org/10.1016/B978-044452701-1.00075-2.

[2] J. Ferré, “3.02 - Regression Diagnostics,” in Comprehensive Chemometrics, S. D. Brown, R. Tauler, and B. Walczak, Eds., Oxford: Elsevier, 2009, pp. 33–89. doi: https://doi.org/10.1016/B978-044452701-1.00076-4.

[3] R. N. Forthofer, E. S. Lee, and M. Hernandez, “13 - Linear Regression,” in Biostatistics (Second Edition), R. N. Forthofer, E. S. Lee, and M. Hernandez, Eds., Second Edition.San Diego: Academic Press, 2007, pp. 349–386. doi: https://doi.org/10.1016/B978-0-12-369492-8.50018-2.

[4] J. A. Bunge and D. H. Judson, “Data Mining,” in Encyclopedia of Social Measurement, K. Kempf-Leonard, Ed., New York: Elsevier, 2005, pp. 617–624. doi: https://doi.org/10.1016/B0-12-369398-5/00159-6.

[5] E. Segev, “4 - Users and uses of Google’s information,” in Google and the Digital Divide, E. Segev, Ed., in Chandos Information Professional Series. Chandos Publishing, 2010, pp. 75–110. doi: https://doi.org/10.1016/B978-1-84334-565-7.50004-6.

[6] L. R. Bergman and D. Magnusson, “Person-centered Research,” in International Encyclopedia of the Social & Behavioral Sciences, N. J. Smelser and P. B. Baltes, Eds., Oxford: Pergamon, 2001, pp. 11333–11339. doi: https://doi.org/10.1016/B0-08-043076-7/00764-6.

[7] T. Tullis and B. Albert, “Chapter 9 - Special Topics,” in Measuring the User Experience (Second Edition), T. Tullis and B. Albert, Eds., Second Edition.in Interactive Technologies. Boston: Morgan Kaufmann, 2013, pp. 209–236. doi: https://doi.org/10.1016/B978-0-12-415781-1.00009-1.

[8] Minitab, “Multicollinearity in regression,” https://support.minitab.com/en-us/minitab/21/help-and-how-to/statistical-modeling/regression/supporting-topics/model-assumptions/multicollinearity-in-regression/, 2023.

[9] J. Fox et al., “Package ‘car,’” 2023. [Online]. Available: https://r-forge.r-project.org/projects/car/,

Angga Dwi Mulyanto (M’89) was born in Malang on August 13, 1989. He completed his undergraduate studies in Statistics (S.Si.) at Brawijaya University and his graduate studies in Statistics (M.Si.) at the same institution. Currently, he works as a lecturer in the Mathematics study program at Maulana Malik Ibrahim State Islamic University Malang with a focus on computational statistics and applied statistics.