



Implementasi Bahasa Pemrograman *Python* untuk *Path analysis*

^{1,*}Rommi Kaestria, ²Elok Faiqotul Himmah

¹Program Studi Sistem Informasi, STMIK Palangkaraya, Jl. G.Obos No.114, Palangka Raya, Kalimantan Tengah, Indonesia

²Program Studi Teknik Informatika, STMIK Palangkaraya, Jl. G.Obos No.114, Palangka Raya, Kalimantan Tengah, Indonesia

Abstrak — *Path analysis* merupakan bagian dari *Structural Equation Modelling* (SEM) yang sering digunakan untuk menyelesaikan masalah diberbagai bidang ilmu. Penyelesaian masalah dengan *path analysis* membutuhkan bantuan perangkat lunak ataupun bahasa pemrograman untuk mempermudah pengujian model dan analisis data. Penelitian ini bertujuan untuk mengimplementasikan bahasa pemrograman *Python* untuk menyelesaikan masalah *path analysis*. Studi kasus dilakukan berdasarkan penelitian yang sebelumnya telah dilakukan penulis yaitu mengenai *path analysis* untuk menentukan pengaruh mata kuliah prasyarat terhadap mata kuliah lanjutannya dengan menggunakan perangkat lunak statistika, R. Penulis menyelesaikan studi kasus tersebut dengan menggunakan bahasa pemrograman *Python* kemudian melakukan perbandingan terhadap hasil yang diperoleh dengan menggunakan *Python* dan R. Penelitian ini merupakan penelitian eksperimen yang meliputi tahap persiapan, pelaksanaan, analisis, dan pengambilan kesimpulan. Studi kasus dilakukan terhadap 94 sampel yaitu mahasiswa yang telah menempuh dan lulus mata kuliah prasyarat dan mata kuliah lanjutannya. Penulis menyelesaikan kasus tersebut dengan menggunakan *Python* kemudian menganalisis dan membandingkan hasil yang diperoleh ini dengan hasil yang diperoleh dengan menggunakan R pada penelitian sebelumnya. Hasil penelitian menunjukkan bahwa penggunaan bahasa pemrograman *Python* juga bisa menyelesaikan permasalahan pada *path analysis* ini dibuktikan dengan hasil yang diperoleh dengan *Python* tidak jauh berbeda dengan hasil yang diperoleh dengan R.

Kata Kunci: *path analysis*; *Python*.

Abstract — *Path analysis* is part of *structural equation modeling* (SEM), which is often used to solve problems in various fields of science. Solving problems with *path analysis* requires the help of software or programming languages to facilitate model testing and data analysis. This study aims to implement the *Python* programming language to solve *path analysis* problems. The case study was carried out based on research that had been carried out by previous authors, namely regarding *path analysis* to determine the effect of prerequisite courses on advanced courses using statistical software, R. The author completed the case study using the *Python* programming language and then compared the results obtained using *Python* and R in this study. This research is experimental and includes the stages of preparation, implementation, analysis, and drawing conclusions. Case studies were conducted on 94 samples, namely students who had taken and passed prerequisite and advanced courses. The author solves the case using *Python*, then analyzes and compares the results obtained with those obtained using R in previous studies. The results of the study show that the use of the *Python* programming language can also solve problems in *path analysis*. This is evidenced by the results obtained with *Python*, which are not much different from the results obtained with R.

Keywords: *path analysis*; *Python*.

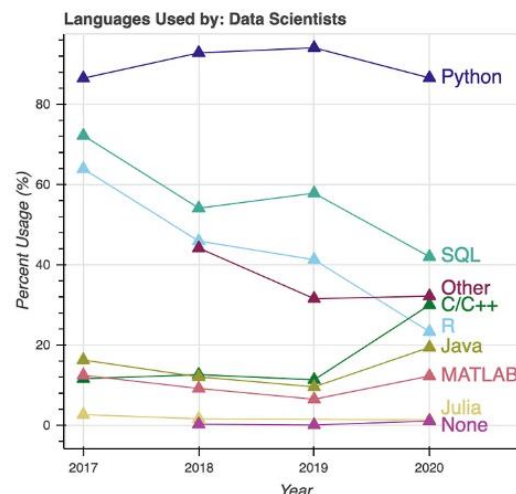
* Corresponding author :
Rommi Kaestria
Prodi Sistem Informasi, STMIK Palangkaraya, Indonesia
rokaforddev@gmail.com

1. PENDAHULUAN

Path analysis merupakan bagian dari *Structural Equation Modelling* (SEM) yang sering digunakan untuk memecahkan masalah diberbagai bidang ilmu seperti ekonomi, psikologi, pendidikan, dan

beberapa bidang ilmu lainnya. Penggunaan *path analysis* adalah untuk menggambarkan dan menguji model hubungan antarvariabel yang berbentuk sebab akibat [1]. Kompleksitas model yang terbentuk dalam penyelesaian kasus *path analysis* serta keterbatasan kemampuan manusia dalam mengolah dan menganalisis data menjadi alasan dibutuhkan bantuan perangkat lunak ataupun bahasa pemrograman yang mampu menyelesaikan permasalahan tersebut. Berbagai perangkat lunak statistika yang dapat digunakan untuk menyelesaikan permasalahan *path analysis* antara lain adalah R, AMOS, dan LISREL [2], [3]. Diantara ketiga perangkat lunak tersebut, R adalah satu-satunya yang bersifat *open source* dan tidak berbayar. R menyediakan berbagai macam paket untuk menyelesaikan beragam permasalahan statistika, salah satunya adalah paket ‘lavaan’ yang digunakan untuk menyelesaikan permasalahan *path analysis*.

Selain R, terdapat bahasa pemrograman yang dapat digunakan untuk menyelesaikan kasus *path analysis* yaitu *Python*. Sejak kemunculannya di tahun 1991, *Python* merupakan salah satu bahasa pemrograman yang populer yang cocok untuk berbagai macam permasalahan. Perkembangan terkini telah memperluas jangkauan penerapan *Python* ke ekonometrik, statistik, dan analisis numerik umum [4]. Untuk analisis data dan interaktif, komputasi eksplorasi dan visualisasi data, *Python* sebanding dengan bahasa pemrograman lainnya seperti R, matlab, SAS, Stata, dan lain-lain [5]. Hasil survey terbaru menunjukkan bahwa *Python* merupakan salah satu diantara bahasa pemrograman teratas yang dapat digunakan untuk *data science* dan juga mengembangkan perangkat lunak manapun. *Python* memiliki sintaks yang sederhana, keterbacaan yang mudah dan portabilitas kode. Tingginya penggunaan *Python* untuk *data science* selama empat tahun terakhir hingga tahun 2020 disajikan pada Gambar 1.



Gambar 1. Tingkat Penggunaan Bahasa Pemrograman untuk *Data Science* [6]

Path analysis yang merupakan perluasan regresi juga merupakan salah satu dari penerapan *data science* sebab dalam metode ini digali informasi yang berharga dari data-data yang diperlukan. Namun demikian, literatur mengenai penggunaan *Python* untuk *path analysis* masih sangat minim. Beberapa penelitian terdahulu membahas mengenai analisis SEM dengan *Python* seperti [7] yang mengimplementasikan metode SEM dengan paket ‘lavaan’ di R dan paket ‘semopy’ di *Python* kemudian membandingkan hasil dari keduanya. Perbandingan meliputi fungsionalitas yang ditawarkan, keakuratan, kinerja dan kemudahan penggunaan paket melalui simulasi skenario data yang hilang, non normalitas dan variabel laten yang hilang. Hasil penelitian menunjukkan bahwa baik paket ‘lavaan’ di R maupun paket ‘semopy’ di *Python* memberikan hasil yang sebanding untuk model pengukuran dan model struktural namun pada kasus kumpulan data dengan data yang hilang, peneliti atau pengembang dapat selektif memilih metode dan fungsi yang tepat dari kedua paket tersebut. Sebelumnya, El-Sheikh et al mengkaji perbandingan antara 5 paket dalam *software* statistika yaitu AMOS, LISREL, dan tiga paket dalam *software* R yaitu sem, openMx dan lavaan berdasarkan metode estimasi yang digunakan untuk analisis SEM [2]. Hasil penelitian menunjukkan bahwa paket lavaan, AMOS, dan LISREL

memberikan estimasi parameter yang sama tanpa perbedaan, hanya ada sedikit perbedaan antara indeks kecocokan untuk paket yang disediakan sedangkan untuk analisis terhadap sem, openMx, dan lavaan diperoleh hasil bahwa paket lavaan merupakan paket yang mencakup paling banyak metode estimasi dalam statistika.

Selanjutnya mengenai penyelesaian kasus *path analysis* dengan R dibahas oleh [3] sedangkan pembahasan lebih detail mengenai pemanfaatan R untuk *path analysis* dilakukan oleh [8] pada penelitian keduanya. Kedua peneliti dalam tulisannya memaparkan pemanfaatan paket 'lavaan' di R untuk implementasi metode analisis jalur. Hasil penelitian menunjukkan bahwa software R yaitu paket 'lavaan' sangat membantu dalam analisis SEM khususnya *path analysis*. Dengan pemanfaatan *software* R ini berhasil dilakukan pengujian asumsi *path analysis*, yaitu uji normalitas, uji multikolinearitas dan uji heteroskedastisitas, pembuatan diagram jalur, penentuan koefisien jalur, dan pengujian model.

Penelitian-penelitian tersebut belum ada yang membahas secara khusus mengenai penyelesaian masalah *path analysis* dengan bahasa pemrograman *Python*. Oleh sebab itu, penulis melakukan penelitian ini. Meskipun SEM memiliki cakupan yang lebih luas daripada *path analysis* namun penelitian ini dapat mengisi kesenjangan (*gap*) dalam literatur mengenai *path analysis* dengan *Python*. Penelitian ini bertujuan untuk memberikan alternatif bahasa pemrograman dan wawasan bagi pengguna mengenai langkah-langkah untuk menyelesaikan permasalahan terkait SEM, khususnya *path analysis* dengan *Python*. Penulis memberikan contoh studi kasus mengenai *path analysis* yaitu menentukan pengaruh mata kuliah prasyarat terhadap hasil studi mata kuliah lanjutannya dan penyelesaiannya menggunakan *Python*.

2. METODOLOGI PENELITIAN

Penelitian ini merupakan penelitian eksperimental yaitu penulis melakukan uji coba pengimplementasian *Python* untuk menyelesaikan permasalahan *path analysis*. Studi kasus mengenai *path analysis* untuk menentukan pengaruh hasil studi mata kuliah prasyarat terhadap mata kuliah lanjutannya.

2.1. Tahapan Penelitian

Adapun tahapan yang dilakukan penulis dalam penelitian ini adalah (1) tahap persiapan yang meliputi pemilihan dan perumusan masalah, studi pendahuluan mengenai *path analysis* dengan bahasa pemrograman *Python*, penentuan variabel penelitian dan pengumpulan data untuk contoh studi kasus ;(2) tahap pelaksanaan yang meliputi implementasi *path analysis* dengan *Python*, analisis terhadap output yang dihasilkan *Python* dibandingkan dengan output yang dihasilkan oleh R pada penelitian sebelumnya, dan (3) pengambilan kesimpulan.

2.2. Variabel Penelitian

Penelitian ini memiliki dua variabel independen (eksogen), satu variabel *intervening* dan satu variabel dependen (endogen). Variabel eksogen dalam penelitian ini adalah hasil belajar mata kuliah Matematika Diskrit (X_1) dan hasil belajar mata kuliah Aljabar Linier dan Matriks (X_2), variabel *intervening*nya adalah hasil belajar mata kuliah Kecerdasan Buatan (Y_1) sedangkan variabel endogennya adalah hasil belajar mata kuliah Sistem Pakar (Y_2).

2.3. Teknik Pengumpulan Data

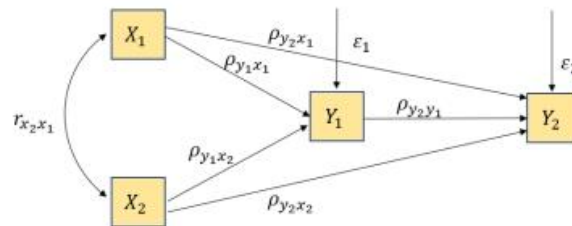
Pengumpulan data dilakukan dengan menggunakan metode studi pustaka dan dokumentasi. Studi pustaka dilakukan untuk mempelajari materi mengenai R dan *path analysis* sedangkan dokumentasi dilakukan untuk mendapatkan data nilai akhir mahasiswa dari Bagian Akademik dan Umum (BAU) STMIK Palangkaraya untuk contoh studi kasus.

2.4. Data yang Digunakan

Data yang digunakan untuk studi kasus dalam penelitian ini berupa data nilai akhir mahasiswa pada mata kuliah Matematika Diskrit (MD), Aljabar Linier dan Matriks (AL), Kecerdasan Buatan (KB), dan Sistem Pakar (SP).

2.5. Diagram Jalur dan Model Struktural

Diagram jalur dalam studi kasus ini ditunjukkan pada Gambar 2 sebagai berikut.



Gambar 2. Diagram Jalur Pengaruh Hasil Studi Mata Kuliah Prasyarat terhadap hasil Studi Mata Kuliah Lanjutan [8]

Berdasarkan diagram jalur seperti pada Gambar 2, maka diperoleh dua model substruktural yang akan digunakan dalam *path analysis* ini yaitu:

$$Y_1 = \rho_{y_1x_1}X_1 + \rho_{y_1x_2}X_2 + \varepsilon_1 \quad (1)$$

$$Y_2 = \rho_{y_2x_1}X_1 + \rho_{y_2x_2}X_2 + \rho_{y_2y_1}Y_1 + \varepsilon_2 \quad (2)$$

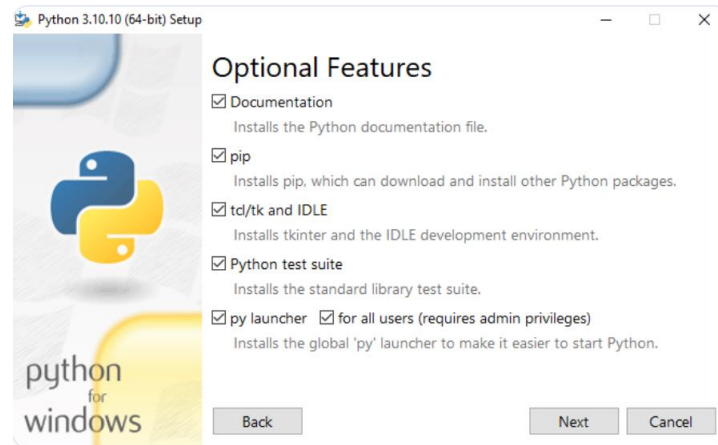
3. HASIL DAN PEMBAHASAN

Sebelum melakukan eksperimen pada bahasa pemrograman *Python* diperlukan komputer dengan spesifikasi yang disarankan untuk penggunaan sistem operasi minimal windows 10 dan juga koneksi internet, kemudian unduh versi stabilnya melalui situs resmi pada laman <https://www.Python.org/downloads/windows/>, versi *Python* pada saat penelitian ini dilakukan adalah versi 3.10. Setelah berhasil diunduh, lakukan penginstalan seperti biasa, pada saat proses penginstalan pastikan untuk klik *checkbox* untuk mengizinkan pengguna mengakses *Python* menggunakan *command line* seperti pada Gambar 3 berikut.



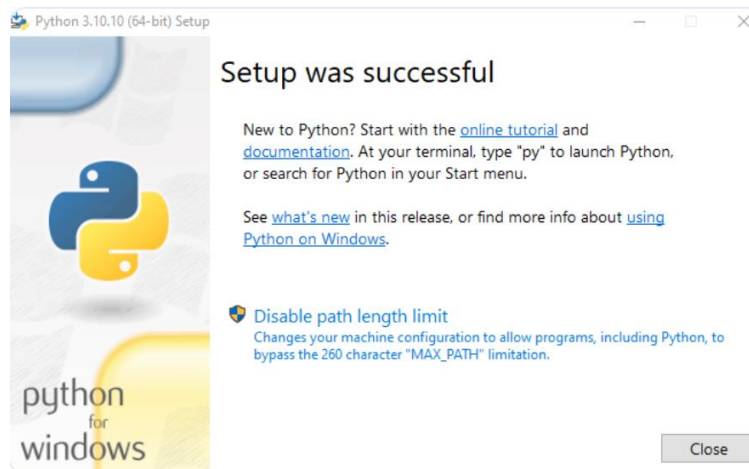
Gambar 3. Proses Instalasi *Python*

Jika baru saja memulai dengan *Python* dan ingin menginstalnya dengan fitur *default* seperti yang dijelaskan dalam dialog, lalu klik **Instal Sekarang** dan lanjutkan ke [Verifikasi Instalasi Python](#). Untuk menginstal fitur opsional dan lanjutan lainnya, klik **Sesuaikan penginstalan** dan lanjutkan. **Fitur Opsional** mencakup alat dan sumber daya umum untuk *Python* dan dapat diinstal semuanya, bahkan sekalipun tidak berencana untuk menggunakannya.



Gambar 4. Proses Opsi Penambahan Fitur pada *Python*

Selanjutnya klik instal dan proses penginstalan akan berjalan. Jika penginstalan telah berhasil dilakukan maka akan tampil halaman seperti pada Gambar 5.



Gambar 5. Tampilan Halaman Jika Instalasi *Python* Berhasil

Kemudian kita bisa gunakan *cmd / command prompt* untuk mengecek apakah versi installan berhasil terinstal.

```

C:\WINDOWS\system32\cmd. x + v
Microsoft Windows [Version 10.0.22621.1344]
(c) Microsoft Corporation. All rights reserved.

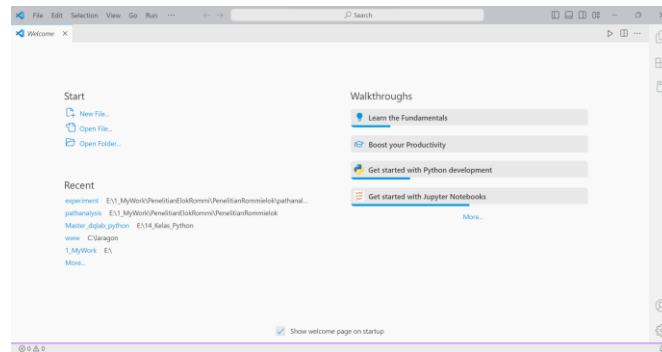
C:\Users\ASUS>python --version
Python 3.10.5

C:\Users\ASUS>

```

Gambar 6. Pengecekan versi *Python*

Untuk menjalankan baris code *Python* diperlukan teks editor. Salah satu teks editor yang bisa menjalankan dan mengeksekusi bahasa *Python* adalah VSCode. Visual Studio Code (disingkat VSCode) adalah perangkat lunak penyunting kode-sumber buatan Microsoft untuk Linux, macOS, dan Windows. Visual Studio Code menyediakan fitur seperti penyorotan sintaksis, penyelesaian kode, kutipan kode, merefaktorkan kode, pengawakutuan, dan Git. Hal ini sangat membantu penulis untuk mengoptimalkan proses pembuatan kode terutama untuk bahasa *Python* yang digunakan.

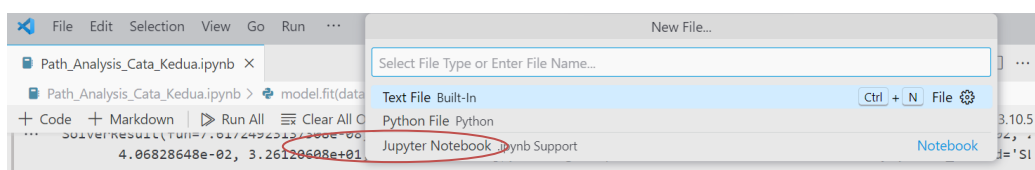


Gambar 7. Tampilan Awal VSCode

Langkah selanjutnya dilakukan penginstalan *extentions* pada VSCode yang mendukung pembuatan pemrograman *Python*, antara lain :

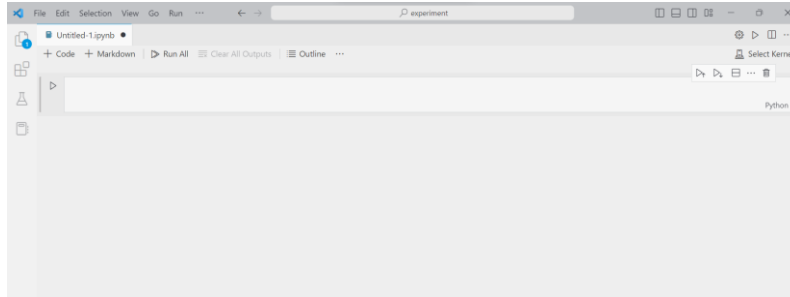
- Python* v2023.4.0 , ekstensi *Python* akan secara otomatis menginstal ekstensi [Pylance](#), Jupyter, dan [isort](#) untuk memberi Anda pengalaman terbaik saat bekerja dengan file *Python* dan notebook Jupyter.
- Jupyter Notebook adalah proyek *open-source* yang menggabungkan teks Markdown dengan kode *Python* (dapat dieksekusi secara langsung) dalam satu kanvas yang dinamakan *notebook*.
- Anaconda adalah *platform* bahasa pemrograman *Python* yang bersifat *open-source* yang bertujuan untuk menyederhanakan manajemen paket serta penyebaran.
- Pylance adalah *extension* yang berfungsi untuk mempermudah menulis code dengan *Python*. Fitur-fitur dari Pylance yaitu *code completion*, *type checking*, *paramater suggestion*, dll.

Semua *extention* tersebut diinstal melalui VSCode kemudian untuk mulai mengerjakan pilih *new file* kemudian akan muncul pilihan jupyter Notebook untuk melakukan manajemen *Python* agar mudah dibaca, seperti ditunjukkan pada Gambar 8.



Gambar 8. Cara membuat *file* baru pada *Python*

Setelah pilih jupyter Notebook diminta untuk membuat nama *file* dengan format *.ipynb* maka akan tampil halaman utama seperti Gambar 9.



Gambar 9. Tampilan Awal Jupyter Notebook pada VSCode

Setelah itu penulis mulai melakukan eksperimen sebanyak dua kali dalam beberapa tahapan.

3.1. Eksperimen Pertama

Penulis memerlukan modul atau sering disebut *library* yang berkaitan dengan penyelesaian masalah yang akan dilakukan, modul tersebut harus diinstal melalui PIP. PIP adalah sebuah manajemen paket (*package management*) yang digunakan untuk instal, hapus, *upgrade* paket *Python* dan sebagainya. Paket (*Package*) adalah *file* yang berkaitan dengan modul dibutuhkan untuk program yang akan dibuat. PIP otomatis sudah ada didalam paket penginstalan *Python* versi 3.4 keatas.

Eksperimen pertama menggunakan modul Pandas, Statsmodels dan Matplotlib di *Python*. Adapun penjelasan masing-masing modul adalah sebagai berikut:

a. Pandas

Pandas adalah pustaka yang merupakan struktur data yang elastis, cepat, dan komunikatif dalam paket *Python* [9]. penulis bisa untuk melakukan penggunaan, modifikasi, dan distribusi perangkat lunak secara gratis. Pandas menyediakan struktur data dan analisis data yang biasa digunakan untuk membuat tabel, mengubah dimensi data, mengecek data, dan lain sebagainya. Struktur data dasar pada Pandas dinamakan *DataFrame*, yang digunakan untuk membaca sebuah file dengan banyak jenis format seperti file *.txt*, *.csv*, dan *.tsv*.

b. Statsmodels

Statsmodels berorientasi pada ilmu data, analisis data, dan statistik. Statsmodels dibangun di atas NumPy dan terintegrasi dengan Pandas untuk penanganan data. Statsmodels digunakan untuk mengeksplorasi data, memperkirakan model statistik dan melakukan uji statistik termasuk model path analysis yang akan dikerjakan pada penelitian ini.

c. Matplotlib

Penulis menggunakan matplotlib untuk library *Python* yang fokus pada visualisasi data seperti plot grafik [9].


Setelah itu penulis melakukan pemanggilan modul yang diperlukan sesuai dengan penjelasan diatas seperti pada Gambar 10.

```
import pandas as pd
import statsmodels.api as sm
from matplotlib import pyplot as plt
from statsmodels.formula.api import ols
```

✓ 0.0s Python

Gambar 10. Kode Import Library pada *Python*

Pada Gambar 10 tersebut penulis memanggil beberapa *library* dengan cara melakukan *import* dan melakukan inisialisasi dengan penamaan variabel seperti pada Gambar 10 tersebut. Kemudian pada aplikasi VSCode bila baris kode tersebut ingin dijalankan dapat dilakukan dengan menekan icon seperti pada Gambar 11.



```
import pandas as pd
import statsmodels.api as sm
from matplotlib import pyplot as plt
from statsmodels.formula.api import ols
```

[4] ✓ 0.0s

Gambar 11. Tombol untuk Menjalankan Baris Kode

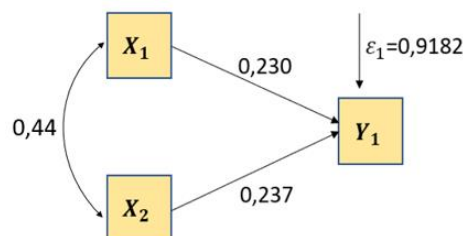
Data yang telah dipanggil kemudian dimunculkan kembali untuk membuktikan data sudah siap untuk diproses. Data nilai mahasiswa disajikan dalam Gambar 12.

	Matematika_Diskrit	Aljabar_Linier_dan_Matriks	Kecerdasan_Buatan	Sistem_Pakar
0	80.50	65.0	74.10	78.66
1	70.83	81.5	66.75	80.00
2	70.00	70.0	89.50	85.60
3	70.00	70.0	90.50	60.90
4	70.43	61.5	67.00	76.50
...
89	62.60	75.0	82.80	88.16
90	64.00	70.0	69.00	78.00
91	70.00	72.0	74.20	80.50
92	68.90	72.0	82.20	91.50
93	70.10	72.0	84.40	87.66

94 rows × 4 columns

Gambar 12. Tampilan Tabel Penyajian Data Nilai Mahasiswa

Langkah selanjutnya adalah mendeskripsikan model yang ingin disesuaikan karena ini harus mewakili arsitektur yang telah digambar dalam model data pada penelitian sebelumnya. Terdapat dua model yang bisa dipecah untuk diselesaikan masing masing seperti pada Gambar 13.



Gambar 13. Model 1 Regresi Linear Berganda

Selanjutnya digunakan simbol \sim untuk mendefinisikan model persamaan struktural sesuai diagram jalur pada Gambar 13 yaitu regresi dari satu variabel laten pada satu atau beberapa variabel laten lainnya [10]. Berdasarkan diagram jalur pada Gambar 13, hasil studi mata kuliah matematika diskrit dan aljabar linier dan matriks memberi pengaruh pada mata kuliah kecerdasan buatan sehingga berlaku persamaan regresi

linear berganda: Kecerdasan_Buatan~Matematika_Diskrit+Aljabar_Linier_dan_Matriks yang selanjutnya diberi nama Model 1.

Selanjutnya penulis melakukan penulisan kode untuk membuat modelnya dengan menggunakan Metode Kuadrat Terkecil atau *Ordinary Least Square* (OLS). OLS merupakan suatu metode penaksiran koefisien regresi atau hubungan antar variabel yang paling sederhana untuk melakukan analisis regresi berganda. Metode ini digunakan untuk meminimalisir jumlah kuadrat kesalahan dengan mengestimasi suatu garis regresi. Kemudian model diringkas untuk ditampilkan dan dibuatkan gambar dengan menggunakan `fig = plt.figure(figsize=(14, 8))`. Kode pemodelan ini seperti ditunjukkan pada Gambar 14.

```
# fit multi linear regression model
multi_model = ols('Kecerdasan_Buatan~Matematika_Diskrit+Aljabar_Linier_dan_Matriks', data=df).fit()

# display model summary
print(multi_model.summary())

# modify figure size
fig = plt.figure(figsize=(14, 8))

# creating regression plots
fig = sm.graphics.plot_regress_exog(multi_model, 'Matematika_Diskrit', fig=fig)
```

Gambar 14. Baris Kode Pemodelan Analisis Jalur Model Pertama

Selanjutnya diperoleh output berupa hasil pemodelan regresi seperti pada Gambar 15.

OLS Regression Results						
Dep. Variable:	Kecerdasan_Buatan	R-squared:	0.173			
Model:	OLS	Adj. R-squared:	0.155			
Method:	Least Squares	F-statistic:	9.518			
Date:	Mon, 06 Mar 2023	Prob (F-statistic):	0.000176			
Time:	13:56:18	Log-Likelihood:	-314.80			
No. Observations:	94	AIC:	635.6			
Df Residuals:	91	BIC:	643.2			
Df Model:	2					
Covariance Type:	nonrobust					
	coef	std err	t	P> t	[0.025	0.975]
Intercept	43.0492	8.863	4.857	0.000	25.444	60.655
Matematika_Diskrit	0.2207	0.101	2.179	0.032	0.020	0.422
Aljabar_Linier_dan_Matriks	0.3002	0.123	2.449	0.016	0.057	0.544
Omnibus:	2.926	Durbin-Watson:	2.040			
Prob(Omnibus):	0.232	Jarque-Bera (JB):	1.989			
Skew:	-0.153	Prob(JB):	0.370			
Kurtosis:	2.356	Cond. No.	1.27e+03			

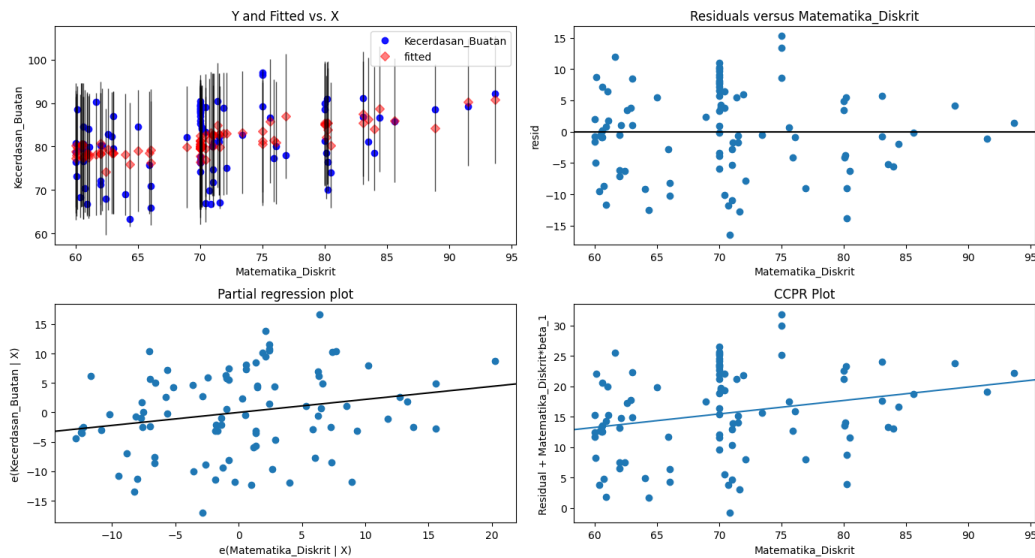
Gambar 15. Hasil Analisis Jalur menggunakan *Python* terhadap Model 1

Perhatikan nilai koefisien jalur (coef) pada output *Python* dan estimasi (estimate) pada output R, keduanya memiliki nilai yang hampir sama seperti yang ditandai pada Gambar 16.

Bahasa R		Python	
Regressions:	Estimate		coef
Kecerdasan_Buatan ~			
Matematk_Dskrt	0.221	Intercept	43.0492
Aljbr_Lnr_dn_M	0.274	Matematika_Diskrit	0.2207
Sistem_Pakar ~		Aljabar_Linier_dan_Matriks	0.3002
Matematk_Dskrt	0.015		
Aljbr_Lnr_dn_M	0.284		
Kecerdasan_Btn	0.048		

Gambar 16. Perbandingan Nilai Koefisien Jalur Model 1 Menggunakan R dan *Python*

Selanjutnya pada salah satu matakuliah yaitu matematika diskrit dapat dilihat bahwa titik-titik tersebut diplot secara acak dan titik-titik tersebut tidak didasarkan pada satu sisi sehingga tidak terjadi masalah heteroskedastisitas yaitu satu faktor yang menyebabkan model regresi linier sederhana tidak efisien dan akurat, juga mengakibatkan penggunaan metode kemungkinan maksimum dalam mengestimasi parameter (koefisien) regresi akan terganggu.



Gambar 17. Plot Regresi Sistem Pakar terhadap Matematika Diskrit

Eksperimen kedua dilakukan terhadap model yang kedua dengan melibatkan semua matakuliah yang terlibat dalam *path analysis* seperti pada Gambar 2. Persamaan regresi model kedua adalah $\text{Sistem_Pakar} \sim \text{Matematika_Diskrit} + \text{Aljabar_Linier_dan_Matriks} + \text{Kecerdasan_Buatan}$, selanjutnya disebut Model 2. Adapun kode pemodelan *path analysis* untuk Model 2 dapat dilihat pada Gambar 18.

```
# fit multi linear regression model
multi_model1 = ols('Sistem_Pakar~Matematika_Diskrit+Aljabar_Linier_dan_Matriks+Kecerdasan_Buatan', data=df).fit()

# display model summary
print(multi_model1.summary())

# modify figure size
fig = plt.figure(figsize=(14, 8))

# creating regression plots
fig = sm.graphics.plot_regress_exog(multi_model, 'Aljabar_Linier_dan_Matriks', fig=fig)
```

Gambar 18. Baris Kode Pemodelan *Path Analysis* Model Kedua

Output yang dihasilkan seperti tampak pada Gambar 19. Perhatikan kembali hasil koefisien jalur (coef) menggunakan *Python* hampir sama dengan hasil yang diperoleh saat menggunakan bahasa R (Estimate), berikut perbandingannya ditunjukkan pada Gambar 20. Dengan demikian dapat disimpulkan bahwa bahasa *Python* juga mampu membantu menyelesaikan permasalahan pada *path analysis*.

3.2. Eksperimen Kedua

Penulis mencoba untuk melakukan eksperimen kedua dengan hanya melibatkan dua *library* yaitu 'semopy' dan 'pandas'. *Semopy* sendiri merupakan singkatan dari *Structural Equation Models Optimization in Python* tujuannya untuk membantu optimasi model persamaan struktural termasuk pembahasan *path analysis* dengan *Python* [10]. Adapun mengenai *pandas* sudah dijelaskan sebelumnya. Berikut eksperimen yang dilakukan penulis dan sebelum menyetikkan modelnya terlebih dahulu kita panggil *library* tersebut dengan menggunakan perintah *import* seperti ditunjukkan pada Gambar 21.

OLS Regression Results

```

=====
Dep. Variable:      Sistem_Pakar      R-squared:          0.118
Model:             OLS                Adj. R-squared:    0.088
Method:           Least Squares       F-statistic:       4.002
Date:             Tue, 07 Mar 2023     Prob (F-statistic): 0.0101
Time:            22:22:13              Log-Likelihood:    -297.15
No. Observations: 94                  AIC:               602.3
Df Residuals:     90                  BIC:               612.5
Df Model:         3
Covariance Type:  nonrobust
=====

```

	coef	std err	t	P> t	[0.025	0.975]
Intercept	56.7224	8.289	6.843	0.000	40.255	73.190
Matematika_Diskrit	0.0133	0.087	0.153	0.879	-0.159	0.185
Aljabar_Linier_dan_Matriks	0.2908	0.105	2.757	0.007	0.081	0.500
Kecerdasan_Buatan	0.0406	0.087	0.464	0.644	-0.133	0.214

```

=====
Omnibus:          13.010      Durbin-Watson:      1.544
Prob(Omnibus):    0.001        Jarque-Bera (JB):   14.354
Skew:             -0.791        Prob(JB):           0.000764
Kurtosis:         4.077        Cond. No.           1.81e+03
=====

```

Gambar 19. Hasil Path Analysis Menggunakan *Python* terhadap Model 2

Regressions:

	Estimate	coef
Kecerdasan_Buatan ~		
Matematk_Dskrt	0.221	Intercept 56.7224
Aljbr_Lnr_dn_M	0.274	Matematika_Diskrit 0.0133
Sistem_Pakar ~		
Matematk_Dskrt	0.015	Aljabar_Linier_dan_Matriks 0.2908
Aljbr_Lnr_dn_M	0.284	Kecerdasan_Buatan 0.0406
Kecerdasan_Btn	0.048	

Gambar 20. Perbandingan Nilai Koefisien Jalur Model 2 Menggunakan R dan *Python*

```

import semopy as sem
import pandas as pd
✓ 0.0s Python

```

Gambar 21. Baris Kode Menggunakan *Library* 'semopy' dan 'pandas'

Selanjutnya dilakukan proses pemanggilan data nilai dan dimasukkan kedalam variabel data. Dengan menggunakan perintah `data = pd.read_csv('datanilaiakhir.csv', sep=';')` dan hasilnya apabila di panggil datanya akan sama seperti pada Gambar 12.

Perbedaan eksperimen kedua dengan eksperimen pertama adalah penulisan dan pemanggilan model dengan kedua model regresi sekaligus yaitu Model 1 dan Model 2 dan dieksekusi secara langsung seperti ditunjukkan pada Gambar 22.

```

analisis_jalur = """
# regressions
Kecerdasan_Buatan~Matematika_Diskrit+Aljabar_Linier_dan_Matriks
Sistem_Pakar~Matematika_Diskrit+Aljabar_Linier_dan_Matriks+Kecerdasan_Buatan
"""
✓ 0.0s Python

```

Gambar 22. Baris Kode Pendefinisian Model *Path Analysis* (Model 1 dan Model 2)

Langkah berikutnya, model tersebut dijadikan objek yaitu dengan melibatkan *library* 'semopy' yang telah diinisiasi menjadi 'sem' kemudian diberi nama objek Model sehingga Model merupakan objek yang berisikan informasi terkait dengan nilai koefisien jalur yang diharapkan. Langkah ini ditunjukkan pada Gambar 23.

```
model = sem.Model(analysis_jalur)
✓ 0.0s Python
```

Gambar 23. Baris Kode Pembuatan Objek Model

Selanjutnya dalam metode `fit()`, penulis menerapkan rumus yang diperlukan ke fitur data input yang ingin diubah dan menghitung hasilnya sebelum menyesuaikan hasilnya pada tampilan. Penulis harus menggunakan metode `.fit()` setelah model dan data diubah menjadi objek seperti pada Gambar 24.

```
model.fit(data)
✓ 0.1s Python
SolverResult(fun=7.61724923137308e-08, success=True, n_it=25, x=array([2.20739574e-01, 3.00228577e-01, 1.31593408e-02, ;
4.06828648e-02, 3.26120608e+01, 4.74733148e+01]), message='Optimization terminated successfully', name_method='SI
```

Gambar 24. Baris Kode untuk Menghitung Hasil pada *Python*

Setelah `model.fit(data)` dijalankan maka akan muncul angka yang menginformasikan tentang hasil dari model terhadap data, agar mudah untuk membaca datanya maka digunakan perintah yang baru yaitu `model.inspect()` seperti pada Gambar 25.

```
model.inspect()
✓ 0.0s Python
```

Gambar 25. Baris Kode untuk Menampilkan Hasil Perhitungan *Path Analysis*

Kemudian setelah dijalankan kodenya akan tampil hasilnya seperti pada Gambar 26.

	lval	op	rval	Estimate	Std. Err	z-value	p-value
0	Kecerdasan_Buatan	~	Matematika_Diskrit	0.220740	0.099665	2.214814	2.677284e-02
1	Kecerdasan_Buatan	~	Aljabar_Linier_dan_Matriks	0.300229	0.120629	2.488868	1.281506e-02
2	Sistem_Pakar	~	Matematika_Diskrit	0.013159	0.084733	0.155303	8.765822e-01
3	Sistem_Pakar	~	Aljabar_Linier_dan_Matriks	0.290763	0.103222	2.816865	4.849494e-03
4	Sistem_Pakar	~	Kecerdasan_Buatan	0.040683	0.085487	0.475895	6.341492e-01
5	Kecerdasan_Buatan	~~	Kecerdasan_Buatan	47.473315	6.924695	6.855655	7.098766e-12
6	Sistem_Pakar	~~	Sistem_Pakar	32.612061	4.756958	6.855655	7.098766e-12

Gambar 26. Tampilan Hasil (Nilai Koefisien Jalur) dari library 'semopy'

Jika diperhatikan nilai yang ditandai pada Gambar 26 yaitu koefisien jalur pada output *Python* hampir sama dengan hasil yang diperoleh menggunakan bahasa R. Nilai-nilai koefisien jalur tersebut menunjukkan besarnya pengaruh variabel bebas terhadap variabel tak bebas sesuai jalurnya masing-masing.

4. KESIMPULAN

Berdasarkan hasil kedua eksperimen yang telah dilakukan penulis diperoleh kesimpulan bahwa Python mampu menyelesaikan permasalahan *path analysis* sama halnya dengan R. Hasil yang diperoleh dari keduanya pun tidak jauh berbeda. Adapun penggunaan bahasa *Python* dan semua integrasi didalamnya baik itu teks editor, banyaknya ekstensi yang bisa digunakan maupun ketersediaan modul dan *library* nya lebih sederhana untuk diterapkan, begitu juga dalam penulisan baris kode jauh lebih sederhana dan hasilnya sesuai dengan yang diharapkan. Oleh sebab itu penulis merekomendasikan penggunaan Python untuk menyelesaikan analisis SEM khususnya *path analysis*.

UCAPAN TERIMA KASIH

Penulis menyampaikan terima kasih kepada Pengelola Unit Penelitian dan Pengabdian kepada Masyarakat (UP3M) STMIK Palangkaraya yang telah mendanai penelitian ini.

DAFTAR PUSTAKA

- [1] Y. Fan *et al.*, “Applications of structural equation modeling (SEM) in ecological studies: an updated review,” *Ecol. Process.*, vol. 5, no. 1, 2016, doi: 10.1186/s13717-016-0063-3.
- [2] M. R. Abonazel, A. A. El-Sheikh, and N. Gamil, “A Review of Software Packages for Structural Equation Modeling: A Comparative Study,” *Appl. Math. Phys.*, vol. 5, no. 3, pp. 85–94, 2017, doi: 10.12691/amp-5-3-2.
- [3] E. F. Himmah and R. Kaestria, “Pemanfaatan Software R untuk Path Analysis,” in *SENDIKA*, 2022, vol. 8, no. September 2016, pp. 83–96, [Online]. Available: <http://eproceedings.umpwr.ac.id/index.php/sendika/article/view/1834/1620>.
- [4] K. Sheppard, *Introduction to Python for Econometrics, Statistics and Data Analysis*. 2013.
- [5] Y. Hasija and R. Chakraborty, *Python for Data Analysis*. 2021.
- [6] D. Capellupo, “Data Science Trends Based on 4 Years of Kaggle Surveys,” 2021. <https://towardsdatascience.com/data-science-trends-based-on-4-years-of-kaggle-surveys-60878d68551f> (accessed Mar. 10, 2023).
- [7] S. K. Paul, A. Taneja, S. Riaz, and S. Das, “Cb-Sem Implementation in R and Python: a Review and Comparative Study,” *J. Appl. Struct. Equ. Model.*, vol. 6, no. 1, 2022, doi: 10.47263/JASEM.6(1)01.
- [8] E. F. Himmah and R. Kaestria, “Path Analysis to Determine the Effect of Learning Outcomes of Prerequisite Mathematics on Learning Outcomes of Expert Systems Courses,” *numerical*, vol. 6, no. 1, 2022, doi: <https://doi.org/10.25217/numerical.v6i1.1625>.
- [9] S. K. Sharma and M. Paliwal, “Overview of data mining with Python modules,” vol. 020070, no. February, 2023.
- [10] A. A. Igolkina and G. Meshcheryakov, “semopy: A Python Package for Structural Equation Modeling,” *Struct. Equ. Model.*, vol. 27, no. 6, pp. 952–963, 2020, doi: 10.1080/10705511.2019.1704289.