BAB III. METODE PENELITIAN

3.1 Jenis Penelitian

Jenis penelitian ini merupakan penelitian kuantitatif yaitu metode penelitian yang menggunakan data penelitian berupa angka-angka dan analisis menggunakan statistik. Jenis data yang digunakan merupakan data skunder yang diperoleh dari Badan Pusat Statistik.

3.2 Tempat dan Waktu Penelitian

Tempat penelitian ini dilakukan di Provinsi Lampung. Waktu pelaksanaan penelitian ini dimulai dari bulan April 2023 hingga bulan Desember 2023.

Tabel 3. 1 Rencana Kegiatan Penelitian

No.	Kegiatan	Tahun 2023								
		Apr	Mei	Jun	Jul	Agt	Sept	Okt	Nov	Des
1.	Bimbingan Thesis									
2.	Permintaan Data									
3.	Pendahuluan									
4.	Landasan Teori									
5.	Metodologi									
6.	Seminar Proposal									
7.	Hasil Penelitian									

8.	Simpulan,									
9.	Sidang Thesis									

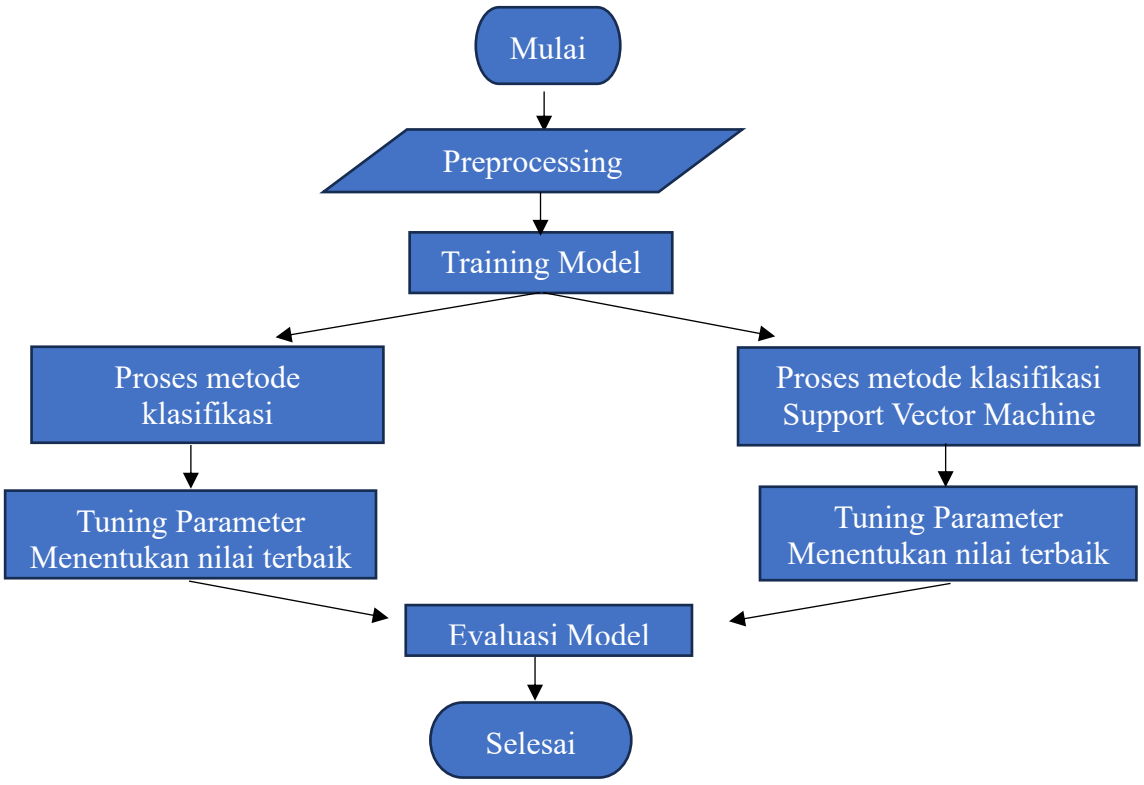
3.3 Alat dan Bahan

Penelitian ini menggunakan dataset Sakernas (Survei Angkatan Kerja Nasional) [26] yang dilaksanakan pada bulan Agustus 2022. File ini berekstensi .csv. sejumlah 22.999 *row data*. File ini dapat diunduh di silastik.bps.go.id. Pengolahan pada penelitian ini menggunakan bahasa pemrograman Python. Python adalah bahasa pemrograman tingkat tinggi yang sering digunakan dalam pengembangan perangkat lunak, pemrograman web, analisis data, dan kecerdasan buatan. Dikembangkan oleh Guido van Rossum pada tahun 1991, Python memiliki sintaks yang mudah dibaca dan ditulis, serta fokus pada keterbacaan kode yang baik. Python memiliki library yang banyak termasuk untuk pemodelan. Adapun library yang akan digunakan yaitu: Pandas, Numpy, Matplotlib, Scikit Learn. Tools editor yang digunakan adalah jupyter notebook. Spesifikasi laptop yang digunakan Acer Pentium core i7 dengan RAM tertanam 16 GB dan SSD 512 GB.

3.4 Proses Pemodelan

Pemodelan merujuk pada proses menciptakan representasi atau abstraksi dari suatu system, konsep, fenomena, atau objek dalam bentuk model. Model adalah suatu representasi yang disederhanakan, dipahami, diprediksi, atau dijelaskan.

Tahapan tahapan yang dilakukan pada proses pemodelan, yaitu Preporocessing, Training Model, Proses metode klasifikasi menggunakan algoritma KNN dan SVM yang dilanjutkan dengan tuning parameter dan evaluasi.



Gambar 3. 1 Proses Pemodelan Pengangguran

3.4.1 Pre Processing

Tahap yang pertama dalam proses pemodelan yaitu *Pre Processing*.. *Pre Processing* adalah langkah – langkah yang dilakukan pada data mentah sebelum data tersebut digunakan dalam analisis atau pemodelan. Tujuan dari *pre-processing* adalah untuk memastikan data siap dan sesuai untuk proses selanjutnya, serta meningkatkan kualitas data dan menghilangkan masalah atau hambatan yang mungkin muncul.

3.4.1.1 Seleksi Fitur

Dataset yang digunakan memiliki banyak sekali fitur, sehingga dibutuhkan melakukan pemilihan fitur untuk memilih subset fitur yang paling relevan dan

berpengaruh terhadap pemodelan. Hal ini dilakukan untuk membantu mengurangi dimensi data, menghindari *overfitting*, dan mempercepat waktu pemodelan.

Pada tahapan seleksi fitur perlu dilakukan analisis mendalam terhadap setiap fitur yang ada dalam dataset. Ini melibatkan pemahaman yang baik tentang setiap fitur, termasuk tipe datanya, rentang nilainya, dan hubungannya dengan variabel target. Proses yang dilakukan yaitu mengidentifikasi fitur yang tidak relevan dan menghapusnya. juga mengidentifikasi data duplikat atau tidak lengkap. Hal ini penting untuk memastikan bahwa data yang digunakan untuk pemodelan berkualitas dan tidak menciptakan bias yang tidak diinginkan.

3.4.1.2 *Eksploratory Data Analysis*

Identifikasi dan penanganan nilai yang hilang (*missing values*) yang ada dalam dataset. Ini dilakukan dengan menghapus baris atau kolom dengan missing values, mengisi missing values dengan nilai rata-rata atau median, atau menggunakan metode lain yang sesuai tergantung pada konteksnya. Pada tahap ini dataset dibagi dua menjadi input/fitur dan target/label. Input/fitur adalah variabel-variabel yang digunakan sebagai masukan atau informasi yang akan digunakan oleh model untuk membuat prediksi atau mengambil keputusan.

Fitur-fitur ini dapat berupa atribut atribut numerik atau kategorikal yang menggambarkan karakteristik dari objek atau entitas yang ingin diprediksi atau dianalisis, contohnya: usia, pendidikan, status, dan lain-lain. Sedangkan target atau label pada pemodelan adalah variabel yang ingin diprediksi atau dianalisis oleh model. Pada klasifikasi ini target yang dimaksud adalah pengangguran atau bukan pengangguran.

Beberapa kasus tertentu butuh data dalam bentuk numerik. Oleh karena itu perlu dilakukan transformasi pada data. Pada tahapan ini juga dilakukan transformasi data. Transformasi pada data yang dilakukan adalah normalisasi *(scaling data* ke rentang tertentu), standardisasi (mengubah data menjadi mean = 0 dan standard deviasi = 1) atau transformasi logaritmik.

3.4.2 *Training Model*

Training model adalah proses di mana sebuah model pembelajaran mesin atau model statistik dilatih menggunakan data pelatihan. Tujuan dari pelatihan model adalah untuk menghasilkan model yang dapat melakukan prediksi atau analisis yang akurat berdasarkan pola–pola di dalam data.

3.4.2.1 *Train Val Test Split*

Train-val-test split adalah proses membagi dataset menjadi tiga subset yang berbeda: data pelatihan (*train set*), data validasi (*validation set*), dan data pengujian (*test set*). Tujuan dari pembagian ini adalah untuk melatih, menyetel, dan menguji model secara objektif dengan menggunakan data yang tidak terlihat sebelumnya.

Berikut adalah penjelasan singkat tentang masing-masing subset:

a. Data Pelatihan (*Train Set*): Data pelatihan digunakan untuk melatih model. Ini adalah subset data yang digunakan oleh model untuk belajar pola-pola yang ada dalam data dan memperbarui parameter atau bobotnya. Data pelatihan harus mewakili keragaman dan variasi dalam dataset secara

menyeluruh untuk membantu model dalam mempelajari pola-pola yang umum.

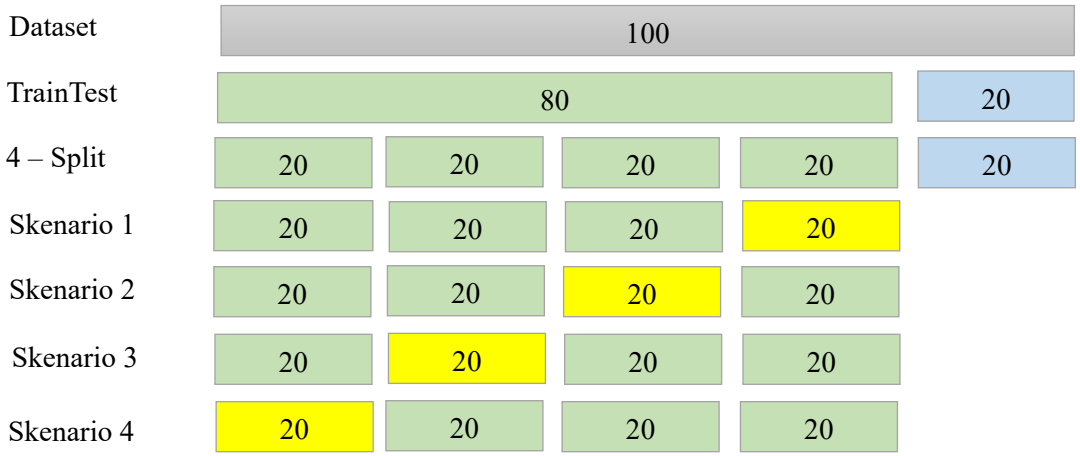
b. Data Validasi (*Validation Set*): Data validasi digunakan untuk menyetel model dan memilih hyperparameter yang optimal. Subset ini digunakan untuk mengukur performa model secara objektif selama pelatihan. Dengan memonitor performa model pada data validasi, kita dapat mengidentifikasi *overfitting* atau *underfitting* dan melakukan penyesuaian model yang diperlukan.
c. Data Pengujian (*Test Set*): Data pengujian digunakan untuk menguji kinerja akhir model yang sudah dilatih. Data ini seharusnya tidak digunakan selama proses pelatihan atau penyetelan model. Data pengujian adalah subset yang independen dan objektif yang mencerminkan data baru yang akan ditemui model di dunia nyata. Pengujian dilakukan untuk memperoleh perkiraan kinerja model pada data yang belum pernah dilihat sebelumnya.

Pemisahan yang tepat antara data pelatihan, data validasi, dan data pengujian penting.Rasio pembagian antara ketiga subset ini dapat bervariasi tergantung pada ukuran dataset dan kebutuhan spesifik dari masalah yang sedang diselesaikan. Umumnya, dataset dapat dibagi menjadi 70-80% untuk data pelatihan, 10-15% untuk data validasi, dan 10-15% untuk data pengujian.

3.4.2.2 *K-Fold Cross Validation*

K-Fold Cross Validation adalah sebuah metode yang digunakan untuk menguji dan mengevaluasi model statistik atau pembelajaran mesin.

Langkah – langkah yang dilakukan pada tahap K-Fold Cross Validation yaitu, Dataset awal dibagi menjadi K fold (misalnya K = 5), di mana setiap fold memiliki jumlah data yang serupa. Data K fold ini, Sebagian besar akan menjadi train set, satu fold akan menjadi validation set, dan satu fold akan menjadi test set. Setelah dataset dibagi, langkah selanjutnya adalah pelatihan dan validasi. Pada iterasi pertama, K-1 fold digunakan sebagai train set, 1 fold digunakan sebagai validation set, dan 1 fold sisanya digunakan sebagai test set. Model dilatih dengan menggunakan train set dan dievaluasi menggunakan validation set. Skenarionya dapat dibagi menjadi 4 skenario.



Gambar 3. 2 Skenario K-Fold Cross Validation Train Val Test Split

3.4.3 Proses metode klasifikasi K-*Nearest Neighbour* (KNN)

Pada tahapan ini kita menentukan nilai K dan metrik jarak. Nilai K merupakan jumlah tetangga terdekat yang akan digunakan dalam proses klasifikasi. Saat memilih nilai K, ada beberapa faktor yang perlu dipertimbangkan. Jika nilai K terlalu kecil, model dapat menjadi sangat sensitif terdapat noise atau outlier dalam data. Sebaliknya, jika nilai K terlalu besar, model dapat kehilangan kemampuan untuk menangkap pola yang lebih spesifik. Sebaiknya, nilai K dipilih secara bijaksana dengan mencoba beberapa nilai dan membandingkan kinerjanya melalui validasi silang atau teknik evaluasi lainnya.

Selanjutnya adalah mengenai metrik jarak, metrik jarak digunakan untuk mengukur jarak antara titik data dalam ruang fitur. Metrik jarak ini mempengaruhi bagaimana KNN menghitung dan memilih tetangga terdekat. Contoh metrik jarak yang umum digunakan adalah *Euclidean Distance (Pythagorean Distance), Manhattan Distance (Snake Distance / Cityblock Distance),* dan *Chebysev Distance (Chessboard Distance).* Penjelasan dari metrik jarak ini yaitu :

a. *Euclidean Distance (Pythagorean Distance)*, Jarak Euclidean juga dikenal sebagai *Pythagorean Distance*, adalah metrik jarak yang paling umum digunakan dalam KNN. Jarak Euclidean antara dua titik dalam ruang fitur dihitung sebagai jarak geometris langsung antara kedua titik tersebut. Dalam ruang dua dimensi, rumus jarak Euclidean dapat dituliskan sebagai berikut :

   d = $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

Di mana (x_1, y_1) dan (x_2, y_2) adalah koordinat duat titik yang akan diukur jaraknya. Secara umum, rumus ini dapat diperluas ke ruang fitur dengan dimensi yang lebih tinnggi.

b. *Manhattan Distance* (Jarak Manhattan), Jarak Manhattan, juga dikenal sebagai *Snake Distance* atau *Cityblock Distance*, mengukur jarak berdasarkan total perbedaan absolut antara koordinat titik dalam setiap dimensi. Dalam ruang dua dimensi, jarak Manhattan antara dua titik dapat dihitung dengan menjumlahkan perbedaan absolut dalam koordinat x dan y. Rumus jarak Manhattan dapat dituliskan sebagai :

   d = $|x_2 - x_1| + |y_2 - y_1|$

   Di ruang fitur dengan dimensi yang lebih tinggi, rumus ini diperluas dengan menjumlahkan perbedaan absolut dalam setiap dimensi

c. *Chebyshev Distance* (Jarak Chebysev), Jarak Chebyshev, juga dikenal sebagai *Chessboard Distance*, mengukur jarak berdasarkan perbedaan absolut terbesar antara koordinat titik dalam setiap dimensi. Dalam ruang dua dimensi, jarak Chebyshev antara dua titik dihitung dengan memilih perbedaan absolut terbesar antara koordinat x dan y. Rumus jarak Chebyshev dapat ditulis sebagai:

   d = max($|x_2 - x_1|$, $|y_2 - y_1|$)

Di ruang fitur dengan dimensi yang lebih tinggi, rumus ini diperluas dengan memilih perbedaan absolut terbesar dalam setiap dimensi. Perbedaan antara jarak Euclidean, Manhattan, dan Chebyshev terletak pada cara mengukur jarak antara dua titik dalam ruang fitur. Jarak Euclidean adalah jarak langsung antara titik, sementara

jarak Manhattan mengukur total perbedaan absolut dalam setiap dimensi, dan jarak Chebyshev memilih perbedaan absolut terbesar dalam setiap dimensi. Pilihan metrik jarak ini tergantung pada sifat data dan tujuan dari masalah klasifikasi yang dihadapi.

3.4.4 *Tuning* Parameter Algoritma KNN

Tuning parameter pada K-Nearest Neighbors (KNN) adalah proses memilih nilai optimal untuk parameter k dalam algoritma KNN. Parameter k mengacu pada jumlah tetangga terdekat yang akan digunakan untuk mengklasifikasikan atau memprediksi suatu data. Berikut adalah langkah-langkah umum dalam tuning parameter KNN:

a. Menentukan rentang nilai k yang akan dieksplorasi: Tentukan rentang nilai k yang akan dievaluasi untuk menemukan nilai k yang optimal. Rentang ini dapat ditentukan berdasarkan pengetahuan domain, eksperimen, atau dengan menggunakan metode pencarian grid (*grid search*).
b. Memisahkan dataset menjadi data pelatihan dan data validasi: Bagi dataset menjadi data pelatihan dan data validasi. Data pelatihan digunakan untuk melatih model KNN, sementara data validasi digunakan untuk mengevaluasi kinerja model dengan nilai k yang berbeda.
c. Menerapkan KNN dengan nilai k yang berbeda: Latih model KNN menggunakan data pelatihan dan variasi nilai k yang telah ditentukan. Untuk setiap nilai k, hitung jarak antara data yang ingin diprediksi dengan data pelatihan, dan ambil k tetangga terdekat.

d. Memilih nilai k yang optimal: Pilih nilai k yang memberikan kinerja terbaik berdasarkan metrik evaluasi yang telah ditentukan. Nilai k yang optimal dapat dipilih berdasarkan titik maksimum atau minimum metrik evaluasi, tergantung pada apakah metrik evaluasi harus maksimum atau minimum.

Selama proses tuning parameter, penting untuk memperhatikan *trade-off* antara kompleksitas model (nilai k yang tinggi) dan kemampuan model untuk menggeneralisasi (nilai k yang rendah). Nilai k yang terlalu rendah dapat menyebabkan model menjadi terlalu rumit dan rentan terhadap overfitting, sedangkan nilai k yang terlalu tinggi dapat menyebabkan model menjadi terlalu sederhana dan rentan terhadap underfitting. Oleh karena itu, pemilihan nilai k yang optimal adalah langkah penting dalam tuning parameter KNN.

3.4.5 Proses Metode Klasifikasi *Support Vector Machine* (SVM)

Pada pelatihan Model SVM, terdapat dua aspek utama yang perlu diperhatikan, yaitu pemilihan kernel dan penyetelan parameter.

a. Pemilihan Kernel: Kernel adalah fungsi matematika yang digunakan oleh SVM untuk memetakan data ke ruang fitur yang lebih tinggi. Pemilihan kernel yang tepat sangat penting karena mempengaruhi kemampuan SVM untuk menemukan hyperplane yang optimal untuk memisahkan kelas-kelas yang berbeda. Berikut adalah beberapa jenis kernel yang umum digunakan:
    - Linear Kernel digunakan ketika data dapat dipisahkan dengan garis lurus dalam ruang fitur yang sama. Ini adalah kernel paling sederhana dan biasanya efisien secara komputasi.

- Polynomial Kernel sebagai fungsi kernel dan memetakan data ke ruang fitur yang lebih tinggi. Kernel polinomial memungkinkan SVM untuk menangani data yang memiliki hubungan non-linier.
- RBF (Radial Basis Function) Kernel: Kernel ini sangat populer karena fleksibilitasnya dalam menangani masalah non-linier. Kernel RBF memetakan data ke ruang fitur tak terbatas dengan distribusi Gauss.
- Sigmoid Kernel digunakan untuk menggeneralisasi SVM sebagai model logistik non-linier. Kernel ini dapat memetakan data ke ruang fitur tak terbatas, mirip dengan kernel RBF.

Pemilihan kernel harus didasarkan pada karakteristik data dan masalah yang dihadapi. Terkadang, pemilihan kernel yang optimal dapat dilakukan melalui eksperimen atau dengan menggunakan teknik pencarian grid (grid search) untuk mengevaluasi kinerja SVM dengan berbagai kernel.

b. Penyetelan Parameter: Selain memilih kernel yang tepat, parameter SVM juga perlu disetel agar model memiliki kinerja yang baik. Dua parameter utama yang perlu diperhatikan adalah parameter C dan gamma:
   - Parameter C: Parameter C mengontrol tingkat penalti untuk kesalahan klasifikasi. Nilai C yang lebih tinggi mengakibatkan model yang lebih kompleks dan berpotensi overfitting pada data pelatihan. Sebaliknya, nilai C yang rendah memberikan toleransi kesalahan yang lebih tinggi dan model yang lebih sederhana.
   - Parameter Gamma: Parameter gamma mengontrol sejauh mana pengaruh dari titik data tunggal dalam pembentukan hyperplane. Nilai gamma yang

lebih tinggi menghasilkan hyperplane yang lebih adaptif dan kompleks, yang dapat menyebabkan *overfitting*. Nilai gamma yang rendah menghasilkan hyperplane yang lebih luas dan sederhana.

Penyetelan parameter dapat dilakukan dengan menggunakan metode pencarian grid (grid search) atau teknik optimasi lainnya untuk menemukan kombinasi nilai C dan gamma yang memberikan kinerja terbaik pada data validasi. Dalam praktiknya, pemilihan kernel dan penyetelan parameter pada SVM seringkali melibatkan eksperimen dan iterasi untuk menemukan kombinasi yang optimal. Tujuannya adalah untuk memastikan bahwa SVM dapat menghasilkan hyperplane yang sesuai dengan data dan memberikan kinerja klasifikasi yang tinggi pada data pengujian yang belum pernah dilihat sebelumnya.

3.4.6 *Tuning* Parameter Pada Algoritma *Support Vector Machine* (SVM)

Tuning parameter pada pelatihan model SVM melibatkan penyesuaian parameter C dan gamma untuk mencapai kinerja yang optimal. Dalam proses ini, tujuan utama adalah untuk menghindari *overfitting* dan *underfitting* dengan menyesuaikan kompleksitas model.

a. Parameter C: Parameter C mengontrol tingkat penalti yang diberikan kepada kesalahan klasifikasi pada data pelatihan. Nilai C yang lebih besar menghasilkan penalti yang lebih tinggi, yang berarti model SVM cenderung mencoba memisahkan setiap contoh dengan benar di data pelatihan. Ini dapat menyebabkan model menjadi lebih kompleks. Nilai C yang lebih kecil memberikan toleransi kesalahan yang lebih tinggi dan membuat model SVM

cenderung menggeneralisasi lebih baik pada data baru dengan mengabaikan beberapa kesalahan klasifikasi pada data pelatihan. Jika nilai C terlalu tinggi, model SVM mungkin overfit pada data pelatihan. Sebaliknya, jika nilai C terlalu rendah, model SVM mungkin underfit dan tidak dapat menangkap pola yang kompleks dalam data.

b. Parameter Gamma: Parameter gamma mengontrol sejauh mana pengaruh dari setiap titik data terdekat dalam membangun hyperplane. Nilai gamma yang lebih besar menghasilkan hyperplane yang lebih adaptif dan kompleks. Model SVM dengan nilai gamma yang tinggi dapat memberikan keputusan yang lebih tajam dan sensitif terhadap setiap titik data dalam data pelatihan. Nilai gamma yang lebih kecil menghasilkan hyperplane yang lebih luas dan lebih umum. Model SVM dengan nilai gamma yang rendah akan memberikan keputusan yang lebih halus dan mempertimbangkan lebih banyak titik data dalam membangun hyperplane. Jika nilai gamma terlalu tinggi, model SVM dapat overfit pada data pelatihan dan menjadi terlalu khusus pada data tersebut. Sebaliknya, jika nilai gamma terlalu rendah, model SVM dapat underfit dan tidak mampu menangkap pola yang kompleks dalam data.

Pada umumnya, *tuning* parameter pada SVM melibatkan eksperimen dan evaluasi kinerja model menggunakan teknik validasi silang (cross-validation) atau metode evaluasi lainnya. Beberapa metode yang umum digunakan untuk tuning parameter SVM adalah:

a. *Grid Search*: Menentukan kumpulan nilai yang mungkin untuk C dan gamma, dan mencoba semua kombinasi nilai tersebut dengan melatih dan mengevaluasi

model SVM pada setiap kombinasi. Metrik evaluasi yang relevan, seperti akurasi atau F1-*score*, digunakan untuk memilih kombinasi parameter yang memberikan kinerja terbaik.

b. *Random Search*: Pemilihan nilai parameter C dan gamma secara acak dari distribusi yang ditentukan sebelumnya. Model SVM dilatih dan dievaluasi dengan setiap kombinasi nilai parameter yang dipilih secara acak. Proses ini dapat diulang beberapa kali untuk mencapai nilai parameter yang optimal.

c. *Bayesian Optimization*: Menggunakan metode optimasi bayesian untuk menentukan kombinasi nilai parameter yang optimal berdasarkan informasi hasil evaluasi model sebelumnya. Metode ini memodelkan fungsi objektif yang memetakan nilai parameter ke kinerja model dan melakukan pencarian berbasis probabilitas untuk menemukan kombinasi

3.4.7 Evaluasi Model

Tahapan evaluasi model memiliki beberapa fungsi penting dalam proses pemodelan. Dengan menggunakan metrik evaluasi yang relevan, pemahaman yang objektif tentang sejauh mana model dapat melakukan prediksi atau menjelaskan data dengan akurat bisa didapatkan. Evaluasi model juga berfungsi untuk membandingkan diantara dua algoritma seperti KNN dan SVM. Berikut adalah penjelasan lebih rinci tentang beberapa metrik evaluasi yang sering digunakan pada model KNN/SVM:

a. Akurasi (*Accuracy*): Akurasi mengukur sejauh mana model KNN / SVM mampu mengklasifikasikan data dengan benar. Akurasi dihitung sebagai

jumlah prediksi yang benar (true positive dan true negative) dibagi dengan total jumlah data.

b. Presisi (*Precision*): Presisi mengukur sejauh mana prediksi positif yang dilakukan oleh model KNN / SVM adalah benar. Presisi dihitung sebagai jumlah true positive dibagi dengan jumlah true positive ditambah jumlah false positive.

c. *Recall* (Sensitivitas atau *True Positive Rate*): Recall mengukur sejauh mana model KNN / SVM dapat mengidentifikasi data positif secara keseluruhan. *Recall* dihitung sebagai jumlah *true positive* dibagi dengan jumlah *true positive* ditambah jumlah *false negative*.

d. F1-*Score*: F1-*Score* adalah ukuran gabungan yang mempertimbangkan presisi dan *recall* secara bersamaan. F1-*Score* dihitung sebagai dua kali perkalian antara presisi dan *recall*, dibagi dengan jumlah presisi dan *recall*.

BAB IV. HASIL DAN PEMBAHASAN

4.1 Pendahuluan

Bab ini memberikan gambaran hasil eksperimen dan analisis data terkait perbandingan algoritma *K-Nearest Neighbors* (KNN) dan *Support Vector Machine* (SVM) dalam prediksi pengangguran di Provinsi Lampung. Analisis yang mendalam akan dilakukan untuk mengevaluasi kinerja kedua algoritma dan memberikan pemahaman yang lebih baik tentang faktor-faktor yang mempengaruhi hasil prediksi.

4.2 Hasil

Setelah melakukan penelitian tentang perbandingan algoritma *K-Nearest Neighbors* (KNN) dan *Support Vector Machine* (SVM) dalam prediksi pengangguran di Provinsi Lampung yang menggunakan bahasa pemrograman Python. Dalam tahapannya terdapat proses *Train Val Test* untuk menghindari *Data Leakage* yaitu kebocoran data Test yang masuk ke dalam proses *Training* dan *Validation*. Pembagian pada tahapan *Data Splitting* sebanyak 80 : 20, yaitu 80% data *Training* dan 20% data *Testing*. Pada prosesnya juga melakukan beberapa *Feature Engineering* yang menghasilkan beberapa data turunan, membuat binning pada kolom umur, melakukan *Cross Validation*, dan juga melakukan *Grid Search*, dan lain-lain. Membungkus semua proses ke dalam beberapa Pipeline. Target dari penelitian ini yaitu bekerja berkode 1 dan pengangguran berkode 2.

4.2.1 *Preprocessing*

Penelitian ini menggunakan Data Sakernas (Survei Angkatan Kerja Nasional) Provinsi Lampung yang memiliki 29.999 baris data dan 220 kolom.

	TAHUN	URUTAN	WEIGHT	KODE_PROV	KODE_KAB	PSU	SSU	STRATA	KLAS	JUMLAHUMUR	...	R47H3	R47H4	R47H5	R48_1	R48_2
0	20228	248991	191	18	1	10573	145360	182	2	4	...	NaN	NaN	NaN	2.0	4.0
1	20228	248992	119	18	1	10573	145360	182	2	4	...	NaN	NaN	NaN	2.0	4.0
2	20228	248993	298	18	1	10573	145360	182	2	4	...	NaN	NaN	NaN	2.0	4.0
3	20228	248995	202	18	1	10573	97985	182	2	7	...	NaN	NaN	NaN	2.0	4.0
4	20228	248996	203	18	1	10573	97985	182	2	7	...	NaN	NaN	NaN	2.0	4.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
22994	20228	280558	90	18	72	5598	171768	181	1	4	...	NaN	NaN	NaN	1.0	4.0
22995	20228	280559	132	18	72	5598	171768	181	1	4	...	NaN	NaN	NaN	1.0	4.0
22996	20228	280560	150	18	72	5598	171768	181	1	4	...	NaN	NaN	NaN	1.0	3.0
22997	20228	280561	100	18	72	5598	240056	181	1	4	...	NaN	NaN	NaN	1.0	3.0
22998	20228	280562	133	18	72	5598	240056	181	1	4	...	NaN	NaN	NaN	1.0	3.0

22999 rows × 220 columns

Gambar 4. 1 Data Sakernas Provinsi Lampung

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 22999 entries, 0 to 22998
Columns: 220 entries, TAHUN to JENISKEGIA
dtypes: float64(175), int64(45)
memory usage: 38.6 MB
```

Berdasarkan beberapa penelitian terdahulu tidak semua atribut akan digunakan untuk penelitian ini melainkan hanya 21 atribut yang terpilih. Adapun atribut yang digunakan dalam penentuan klasifikasi pengangguran di Provinsi Lampung dapat dilihat pada tabel 4.1 :

Tabel 4. 1 Variabel yang Digunakan Sebagai Bahan Penelitian Berdasarkan Penelitian Terdahulu.

No	Variabel	Label
1	KLAS	Klasifikasi Perkotaan/Perdesaan
2	K4	Jenis Kelamin
3	K6	Umur
4	R4	Status Perkawinan
5	R6A	Pendidikan tertinggi yang ditamatkan
6	R6D	Pernah mengikuti pelatihan/kursus/training

7	R6E	Sertifikat dari pelatihan/kursus/training
8	R8A	Kesulitan melihat
9	R8B	Kesulitan mendengar
10	R8C	Kesulitan berjalan/naik tangga
11	R8D	Kesulitan memegang
12	R8E	Kesulitan berbicara
13	R8F	Kesulitan lain
14	R9A	Bekerja seminggu terakhir minimal 1 Jam
15	R9B	Melakukan kegiatan untuk memperoleh penghasilan / pendapatan uang dalam seminggu terakhir
16	R9C	Membantu kegiatan usaha atau pekerjaan keluarga/orang lain dalam seminggu terakhir
17	R10	Sementara tidak bekerja Seminggu terakhir dalam seminggu yang lalu, dan sebenarnya memiliki pekerjaan
18	R38A	Kapan memperoleh pekerjaan/memulai usaha setelah lulus dari pendidikan tertinggi yang ditamatkan
19	R38B	Pernah punya pekerjaan/usaha sebelumnya
20	R42A	Alasan utama berhenti bekerja di pekerjaan terakhir
21	R47B	Mendaftar program kartu prakerja

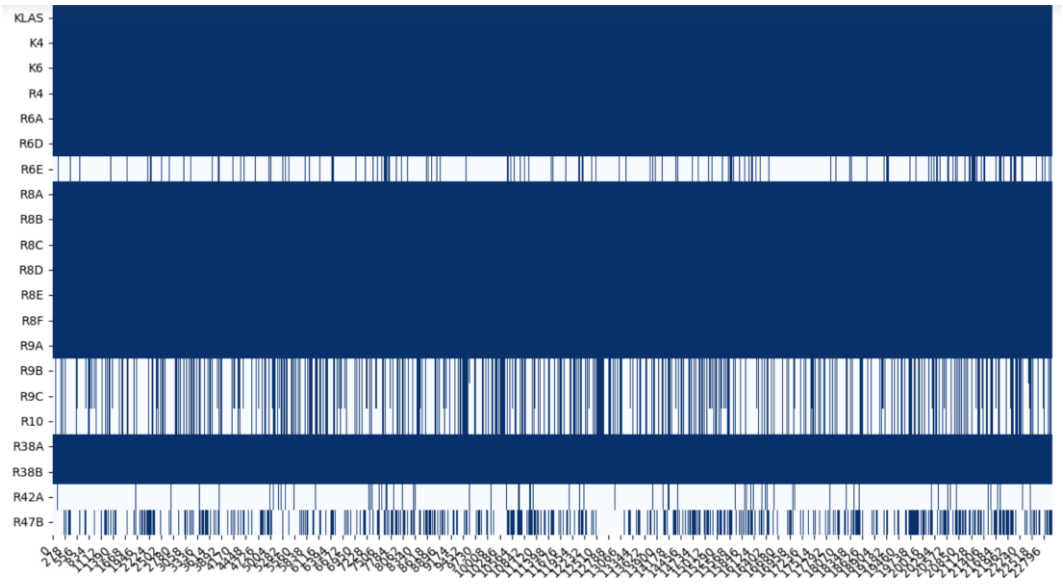
Terlihat masih ada beberapa kolom yang bernilai NaN. Sehingga perlu dilakukan perlakuan seperti imputasi nilai kedalam kolom tersebut pada setiap barisnya.

	KLAS	K4	K6	R4	R6A	R6D	R6E	R8A	R8B	R8C	...	R8E	R8F	R9A	R9B	R9C	R10	R38A	R38B	R42A	R47B
0	2	1	43	2	4	2	NaN	4	8	4	...	4	8	1	NaN	NaN	NaN	1	1	NaN	NaN
1	2	2	38	2	3	2	NaN	4	8	4	...	4	8	2	2.0	2.0	2.0	2	2	NaN	NaN
2	2	1	20	1	5	2	NaN	4	8	4	...	4	8	2	2.0	2.0	2.0	3	2	NaN	NaN
3	2	1	30	2	7	1	1.0	4	8	4	...	4	8	1	NaN	NaN	NaN	1	1	3.0	NaN
4	2	2	27	2	7	2	NaN	4	8	4	...	4	8	2	2.0	2.0	2.0	2	1	NaN	2.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
22994	1	2	46	2	3	2	NaN	4	8	4	...	4	8	2	2.0	2.0	2.0	1	1	NaN	NaN
22995	1	1	26	1	4	2	NaN	4	8	4	...	4	8	1	NaN	NaN	NaN	2	2	NaN	2.0
22996	1	2	18	2	4	2	NaN	4	8	4	...	4	8	2	2.0	2.0	2.0	3	2	NaN	2.0
22997	1	1	32	2	3	2	NaN	4	8	4	...	4	8	1	NaN	NaN	NaN	1	1	NaN	NaN
22998	1	2	26	2	3	2	NaN	4	8	4	...	4	8	2	2.0	2.0	2.0	1	1	NaN	NaN

22999 rows × 21 columns

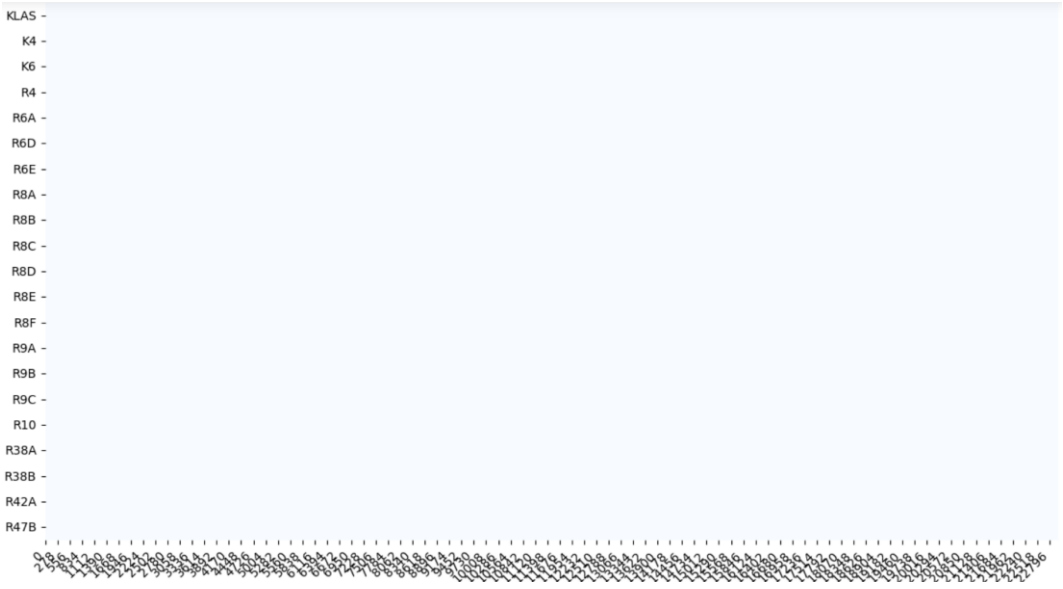
Gambar 4. 2 Data Sakernas masih ada bernilai NaN

Dari gambar berikut tampak pada R6E, R9B, R9C, R10, R42A, R47B, masih ada nilai-nilai yang kosong sehingga perlu di imputasi nilai.



Gambar 4. 3 *Missing Value*

Berdasarkan redaksi pertanyaan dari kuesioner Sakernas maka untuk rincian R6E, R9B, R9C, R10, R47B diisi dengan nilai 2, sedangkan untuk R42A diisi dengan nilai 10. Setelah dilakukan imputasi pada ke 6 rincian diatas maka sudah tidak ada lagi missing value dari data tersebut yang dibuktikan dengan gambar di bawah ini sehingga data bisa langsung dilakukan ke tahap selanjutnya.



Gambar 4. 4 *Missing Value* telah terisi

Setelah semua *Missing Value* sudah di imputasi maka tahapan selanjutnya adalah melakukan *Fiture Engineering* membuat data turunan yaitu dengan membuat fungsi disabilitas, dimana fungsi disabilitas ini menggabungkan beberapa variabel seperti kesulitan melihat (R8A), kesulitan mendengar (R8B), kesulitan berjalan/naik tangga (R8C), kesulitan memegang (R8D), kesulitan berbicara (R8E), dan kesulitan lain (R8F) menjadi satu variabel disabilitas yang diberi label 1 adalah penyandang disabilitas dan 2 bukan penyandang disabilitas. Agar mempermudah proses training nantinya kolom R8 sebagai kolom disabilitas diletakkan setelah kolom R6E. Seperti tampak pada gambar berikut ini.

	KLAS	K4	K6	R4	R6A	R6D	R6E	R8	R8A	R8B	...	R8E	R8F	R9A	R9B	R9C	R10	R38A	R38B	R42A	R47B
0	2	1	43	2	4	2	2.0	2	4	8	...	4	8	1	2.0	2.0	2.0	1	1	10.0	2.0
1	2	2	38	2	3	2	2.0	2	4	8	...	4	8	2	2.0	2.0	2.0	2	2	10.0	2.0
2	2	1	20	1	5	2	2.0	2	4	8	...	4	8	2	2.0	2.0	2.0	3	2	10.0	2.0
3	2	1	30	2	7	1	1.0	2	4	8	...	4	8	1	2.0	2.0	2.0	1	1	3.0	2.0
4	2	2	27	2	7	2	2.0	2	4	8	...	4	8	2	2.0	2.0	2.0	2	1	10.0	2.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
22994	1	2	46	2	3	2	2.0	2	4	8	...	4	8	2	2.0	2.0	2.0	1	1	10.0	2.0
22995	1	1	26	1	4	2	2.0	2	4	8	...	4	8	1	2.0	2.0	2.0	2	2	10.0	2.0
22996	1	2	18	2	4	2	2.0	2	4	8	...	4	8	2	2.0	2.0	2.0	3	2	10.0	2.0
22997	1	1	32	2	3	2	2.0	2	4	8	...	4	8	1	2.0	2.0	2.0	1	1	10.0	2.0
22998	1	2	26	2	3	2	2.0	2	4	8	...	4	8	2	2.0	2.0	2.0	1	1	10.0	2.0

22999 rows × 22 columns

Gambar 4. 5 Pemindahan letak kolom R8

Selain fungsi disabilitas *Fiture Engineering* lain yang dilakukan adalah binning kelompok umur. Berdasarkan klasifikasi usia menurut Kementerian Kesehatan sebagai berikut: 1) Masa Balita: 0–5 Tahun; 2) Masa Kanak-Kanak: 5–11 Tahun; 3) Masa Remaja Awal: 12–16 Tahun; 4) Masa Remaja Akhir: 17–25 Tahun; 5) Masa Dewasa Awal: 26–35 Tahun; 6) Masa Dewasa Akhir: 36–45 Tahun; 7) Masa Lansia Awal: 46–55 Tahun; 8) Masa Lansia Akhir: 56–65 Tahun; dan 9) Masa Manula: > 65 Tahun[27]. Hasil dari binning umur langsung menggantikan kolom sebelumnya. Seperti pada gambar berikut ini.

	KLAS	K4	K6	R4	R6A	R6D	R6E	R8	R8A	R8B	...	R8E	R8F	R9A	R9B	R9C	R10	R38A	R38B	R42A	R47B
0	2	1	6	2	4	2	2.0	2	4	8	...	4	8	1	2.0	2.0	2.0	1	1	10.0	2.0
1	2	2	6	2	3	2	2.0	2	4	8	...	4	8	2	2.0	2.0	2.0	2	2	10.0	2.0
2	2	1	4	1	5	2	2.0	2	4	8	...	4	8	2	2.0	2.0	2.0	3	2	10.0	2.0
3	2	1	5	2	7	1	1.0	2	4	8	...	4	8	1	2.0	2.0	2.0	1	1	3.0	2.0
4	2	2	5	2	7	2	2.0	2	4	8	...	4	8	2	2.0	2.0	2.0	2	1	10.0	2.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
22994	1	2	7	2	3	2	2.0	2	4	8	...	4	8	2	2.0	2.0	2.0	1	1	10.0	2.0
22995	1	1	5	1	4	2	2.0	2	4	8	...	4	8	1	2.0	2.0	2.0	2	2	10.0	2.0
22996	1	2	4	2	4	2	2.0	2	4	8	...	4	8	2	2.0	2.0	2.0	3	2	10.0	2.0
22997	1	1	5	2	3	2	2.0	2	4	8	...	4	8	1	2.0	2.0	2.0	1	1	10.0	2.0
22998	1	2	5	2	3	2	2.0	2	4	8	...	4	8	2	2.0	2.0	2.0	1	1	10.0	2.0

22999 rows × 22 columns

Gambar 4. 6 *Binning* Umur

Fiture Engineering selanjutnya yaitu menggabungkan kolom Bekerja seminggu terakhir minimal 1 jam (R9A), Melakukan kegiatan untuk memperoleh penghasilan/pendapatan uang dalam seminggu terakhir (R9B), Membantu kegiatan usaha atau pekerjaan keluarga / orang lain dalam seminggu terakhir (R9C), dan Sementara tidak bekerja seminggu terakhir dalam seminggu yang lalu, dan sebenarnya memiliki pekerjaan (R10). Keempat kolom digabungkan menjadi sebuah kolom yaitu kolom Target yang menentukan bekerja atau pengangguran. Kode 1 untuk bukan pengangguran dan kode 2 untuk pengangguran.

	KLAS	K4	K6	R4	R6A	R6D	R6E	R8	R8A	R8B	...	R8F	R9A	R9B	R9C	R10	R38A	R38B	R42A	R47B	TARGET
0	2	1	6	2	4	2	2.0	2	4	8	...	8	1	2.0	2.0	2.0	1	1	10.0	2.0	1
1	2	2	6	2	3	2	2.0	2	4	8	...	8	2	2.0	2.0	2.0	2	2	10.0	2.0	2
2	2	1	4	1	5	2	2.0	2	4	8	...	8	2	2.0	2.0	2.0	3	2	10.0	2.0	2
3	2	1	5	2	7	1	1.0	2	4	8	...	8	1	2.0	2.0	2.0	1	1	3.0	2.0	1
4	2	2	5	2	7	2	2.0	2	4	8	...	8	2	2.0	2.0	2.0	2	1	10.0	2.0	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
22994	1	2	7	2	3	2	2.0	2	4	8	...	8	2	2.0	2.0	2.0	1	1	10.0	2.0	2
22995	1	1	5	1	4	2	2.0	2	4	8	...	8	1	2.0	2.0	2.0	2	2	10.0	2.0	1
22996	1	2	4	2	4	2	2.0	2	4	8	...	8	2	2.0	2.0	2.0	3	2	10.0	2.0	2
22997	1	1	5	2	3	2	2.0	2	4	8	...	8	1	2.0	2.0	2.0	1	1	10.0	2.0	1
22998	1	2	5	2	3	2	2.0	2	4	8	...	8	2	2.0	2.0	2.0	1	1	10.0	2.0	2

22999 rows × 23 columns

Gambar 4. 7 Pembentukan Kolom Target

Setelah *Fiture Engineering* selesai dilakukan kolom pembentuk data turunan dihapus untuk mempermudah proses selanjutnya. Jumlah total keseluruhan kolom setelah Preprocessing adalah 13 kolom yaitu Klasifikasi Perkotaan/Perdesaan (KLAS), Jenis Kelamin (K4), Umur (K6), Status Perkawinan (R4), Pendidikan tertinggi yang ditamatkan (R6A), Pernah mengikuti pelatihan/kursus/training (R6D), Sertifikat dari pelatihan/kursus/training (R6E), Disabilitas (R8), Kapan memperoleh pekerjaan/memulai usaha setelah lulus dari

Pendidikan tertinggi yang ditamatkan (R38A), Pernah punya pekerjaan/usaha sebelumnya (R38B), Alasan utama berhenti bekerja di pekerjaan terakhir (R42A), Mendaftar program kartu prakerja (R47B).

	KLAS	K4	K6	R4	R6A	R6D	R6E	R8	R38A	R38B	R42A	R47B	TARGET
0	2	1	6	2	4	2	2.0	2	1	1	10.0	2.0	1
1	2	2	6	2	3	2	2.0	2	2	2	10.0	2.0	2
2	2	1	4	1	5	2	2.0	2	3	2	10.0	2.0	2
3	2	1	5	2	7	1	1.0	2	1	1	3.0	2.0	1
4	2	2	5	2	7	2	2.0	2	2	1	10.0	2.0	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...
22994	1	2	7	2	3	2	2.0	2	1	1	10.0	2.0	2
22995	1	1	5	1	4	2	2.0	2	2	2	10.0	2.0	1
22996	1	2	4	2	4	2	2.0	2	3	2	10.0	2.0	2
22997	1	1	5	2	3	2	2.0	2	1	1	10.0	2.0	1
22998	1	2	5	2	3	2	2.0	2	1	1	10.0	2.0	2

22999 rows × 13 columns

Gambar 4. 8 Menghilangkan kolom pembentuk data turunan

Tahapan *preprocessing* menghasilkan 12 variabel yang nantinya akan menjadi fitur sebelum dilakukan training model. Berikut adalah daftar variabel tersebut :

Tabel 4. 2 Ringkasan variabel yang digunakan dalam penelitian

No	Variabel	Label	Nilai
1	KLAS	Klasifikasi Perkotaan/Perdesaan	1 = Perkotaan 2 = Perdesaan
2	K4	Jenis Kelamin	1 = Laki laki 2 = Perempuan
3	K6	Umur	1 = Masa Balita: 0–5 Tahun 2 = Masa Kanak-Kanak: 6–11 Tahun 3 = Masa Remaja Awal: 12–16 Tahun 4 = Masa Remaja Akhir: 17–25 Tahun 5 = Masa Dewasa Awal: 26–35 Tahun

			6 = Masa Dewasa Akhir: 36–45 Tahun 7 = Masa Lansia Awal: 46–55 Tahun 8 = Masa Lansia Akhir: 56–65 Tahun 9 = Masa Manula: > 65 Tahun
4	R4	Status Perkawinan	1 = Belum kawin 2 = Kawin 3 = Cerai Hidup 4 = Cerai mati
5	R6A	Pendidikan tertinggi yang ditamatkan	1 = Tidak/belum tamat SD 2 = SD/MI/SDLB/Paket A 3 = SMP/MTs/SMPLB/Paket B 4 = SMA/MA/SMLB/Paket C 5 = SMK 6 = MAK 7 = Diploma I/II/III 8 = Diploma IV 9 = S1 10 = S2 11 = S2 Terapan 12 = S3
6	R6D	Pernah mengikuti pelatihan/kursus/training	1 = Ya 2 = Tidak
7	R6E	Sertifikat dari pelatihan/kursus/training	1 = Ya 2 = Tidak
8	R8	Disabilitas	1 = Ya 2 = Tidak
9	R38A	Kapan memperoleh pekerjaan/memulai usaha setelah lulus dari pendidikan tertinggi yang ditamatkan	1 = Bekerja setelah lulus pendidikan tertinggi 2 = Sudah bekerja sebelum lulus Pendidikan tertinggi 3 = Belum pernah bekerja / memulai usaha sejak lulus pendidikan tertinggi

10	R38B	Pernah punya pekerjaan/usaha sebelumnya	1 = Ya 2 = Tidak
11	R42A	Alasan utama berhenti bekerja di pekerjaan terakhir	1 = PHK 2 = Usaha terhenti / Bangkrut 3 = Pendapatan kurang memuaskan 4= Tidak cocok dengan lingkungan kerja 5= Habis masa kerja / kontrak 6= Mengurus rumah tangga 7= Takut terinfeksi Corona / COVID-19 8= Social/physical distancing, karantina mandiri. Perlakuan Pembatasan Kegiatan Masyarakat (PPKM) 9= Selain alasan diatas 10= Lainnya
12	R47B	Mendaftar program kartu prakerja	1 = Ya 2 = Tidak

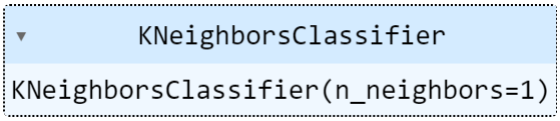
4.2.2 *Training Model*

Pelatihan model dalam konteks machine learning adalah proses di mana model pembelajaran mesin "mempelajari" pola atau hubungan dari data latihan. Data latihan adalah sekumpulan contoh input dan output yang digunakan untuk mengajar model. Tujuan pelatihan model adalah untuk membuat model yang dapat membuat prediksi yang akurat atau mengambil keputusan yang tepat tergantung pada jenis tugas pembelajaran mesin yang sedang dijalankan.

4.2.2.1. Proses Metode Klasifikasi *K-Nearest Neighbors* (KNN)

Tahapan awal proses *Training Model* pada metode klasifikasi KNN yaitu menentukan jumlah tetangga terdekatnya (n_neighbors). Jumlah tetangga terdekat

awal untuk penelitian ini adalah 1, menggunakan *KNeighborsClassifier*. Seperti pada gambar



Gambar 4. 9 KNeighborsClassifier dengan 1 tetangga terdekat

Dari ujicoba acak diatas didapat score 0.81.

4.2.2.2 *Tuning* Parameter Menentukan Nilai Terbaik

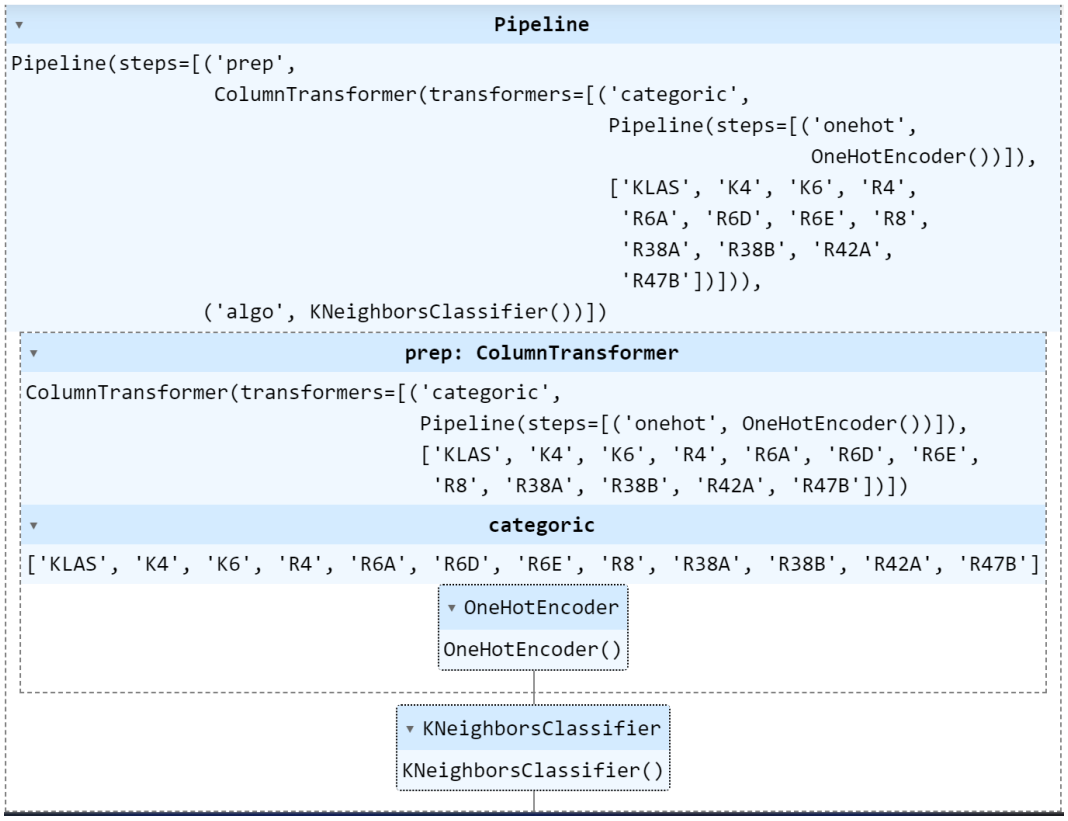
Untuk meningkatkan akurasi model, tahapan selanjutnya yaitu *Dataset Splitting*. Dataset dipisah menjadi data training dan data testing. Pada penelitian kali ini pembagiannya adalah sebanyak 80% training dan 20% testing. Jumlah data training setelah dataset splitting adalah 18.399 dan jumlah data testing setelah dataset splitting adalah 4600.

```
((18399, 12), (4600, 12), (18399,), (4600,))
```

Setelah *dataset splitting*, dataset dimasukkan ke dalam pipeline untuk mempermudah proses machine learning. Setelah itu melakukan proses Column Transformer yang dimasukkan ke dalam variabel preprocessor dimana isinya adalah kolom:

['KLAS','K4','K6','R4','R6A','R6D','R6E','R8','R38A','R38B','R42A','R47B']

Tahapan selanjutnya adalah memasukkan proses sebelumnya ke dalam sebuah pipeline, dimana pipeline tersebut berisi tahapan sebelumnya yaitu preprocessor dan algoritma yang akan dipakai yaitu *KNeighborsClassifier.*



Gambar 4. 10 Dataset Splitting dan Column Transformer KNN

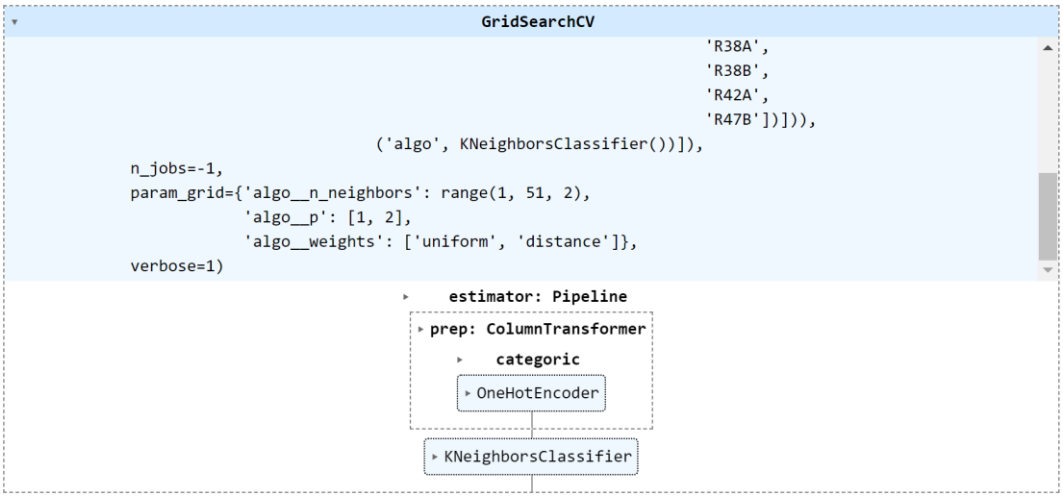
Dari ujicoba ini didapat hasil score sebesar 0.80.

Langkah selanjutnya untuk meningkatkan hasil score dari model adalah menggunakan *Grid Search.* Grid search adalah suatu teknik yang digunakan dalam peningkatan model machine learning untuk mencari kombinasi hyperparameter terbaik dari suatu model. Hyperparameter adalah parameter-parameter yang tidak dipelajari oleh model selama pelatihan, dan mereka perlu diatur sebelum proses pelatihan dimulai. Hyperparameter yang digunakan pada penelitian ini yaitu :

1. n_neighbors: Jumlah tetangga terdekat yang akan dipertimbangkan. (jumlah tetangga merupakan angka ganjil dari 1 sampai 49)
2. weights: Menentukan cara memberikan bobot pada tetangga (uniform atau distance-based).

3. p: Parameter untuk jenis jarak (1 untuk Manhattan distance, 2 untuk Euclidean distance).

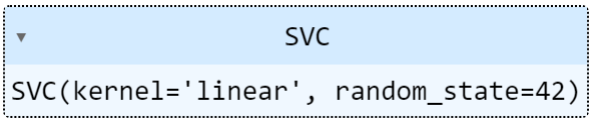
Selain parameter diatas, pada penelitian ini juga mengatur jumlah *cross validation* sebanyak 3, 4, 5, 6 fold.



Gambar 4. 11 GridSearch dan Cross Validation KNN

4.2.2.3 Proses Metode *Support Vector Machine* (SVM)

Tahapan awal proses *Training Model* pada metode klasifikasi SVM yaitu menentukan kernelnya dan menentukan nilai C nya. Nilai C yaitu parameter penalti yang mengontrol *trade-off* antara menciptakan margin yang sebesar mungkin dan mengizinkan beberapa titik pelatihan melanggar margin. Nilai C yang lebih tinggi memberikan penalti yang lebih besar terhadap pelanggaran margin. Dan besaran C untuk penelitian ini adalah 1. Kernel untuk penelitian ini menggunakan kernel linear. Kernel linear mengimplikasikan bahwa model SVM akan mencoba memisahkan kelas dengan sebuah hyperplane linear. Seperti pada gambar



Gambar 4. 12 Kernel Linear dan Nilai C = 1

Dari ujicoba diatas didapat score 0,79.

4.2.2.4 *Tuning* Parameter Menentukan nilai terbaik

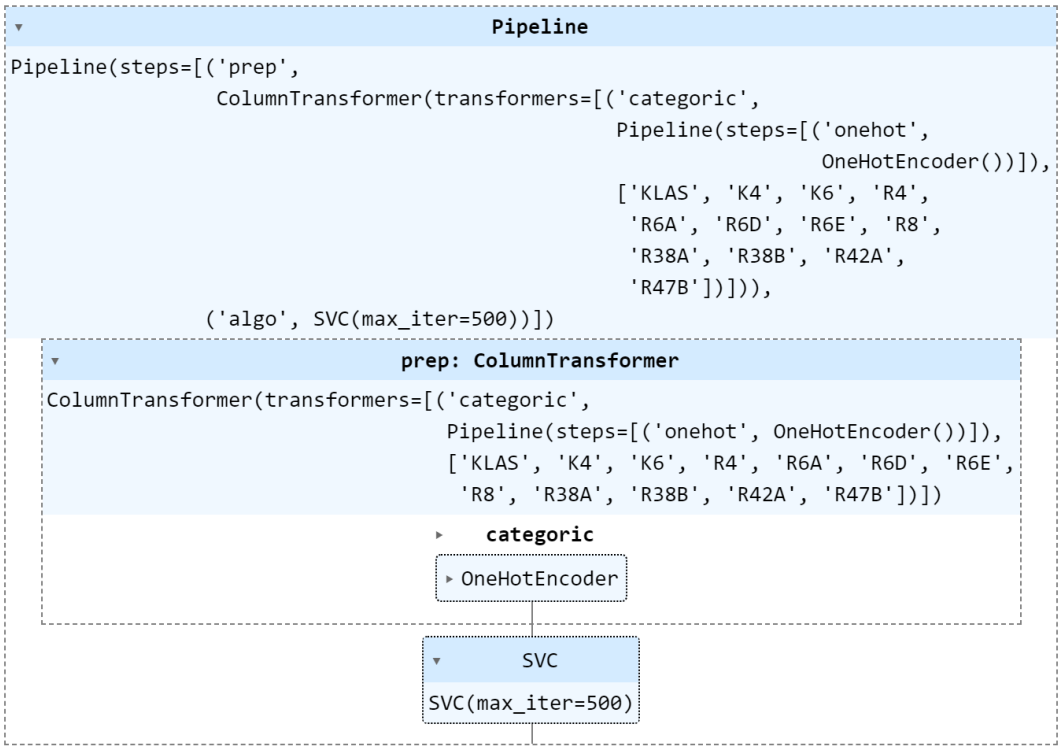
Untuk meningkatkan akurasi model, tahapan selanjutnya yaitu *Dataset Splitting*. Dataset dipisah menjadi data training dan data testing. Pada penelitian kali ini pembagiannya adalah sebanyak 80% training dan 20% testing. Jumlah data training setelah dataset splitting adalah 18.399 dan jumlah data testing setelah dataset splitting adalah 4.600.

```
((18399, 12), (4600, 12), (18399,), (4600,))
```

Setelah *dataset splitting*, dataset dimasukkan ke dalam pipeline untuk mempermudah proses machine learning. Setelah itu melakukan proses Column Transformer yang dimasukkan ke dalam variabel *preprocessor* dimana isinya adalah kolom:

['KLAS','K4','K6','R4','R6A','R6D','R6E','R8','R38A','R38B','R42A','R47B']

Tahapan selanjutnya adalah memasukkan proses sebelumnya ke dalam sebuah pipeline, dimana pipeline tersebut berisi tahapan sebelumnya yaitu *preprocessor* dan algoritma yang akan dipakai yaitu *SupportVectorClassifier*.



Gambar 4. 13 Dataset Splitting dan Column Transformer SVM

Dari ujicoba ini didapat hasil score sebesar 0,54.

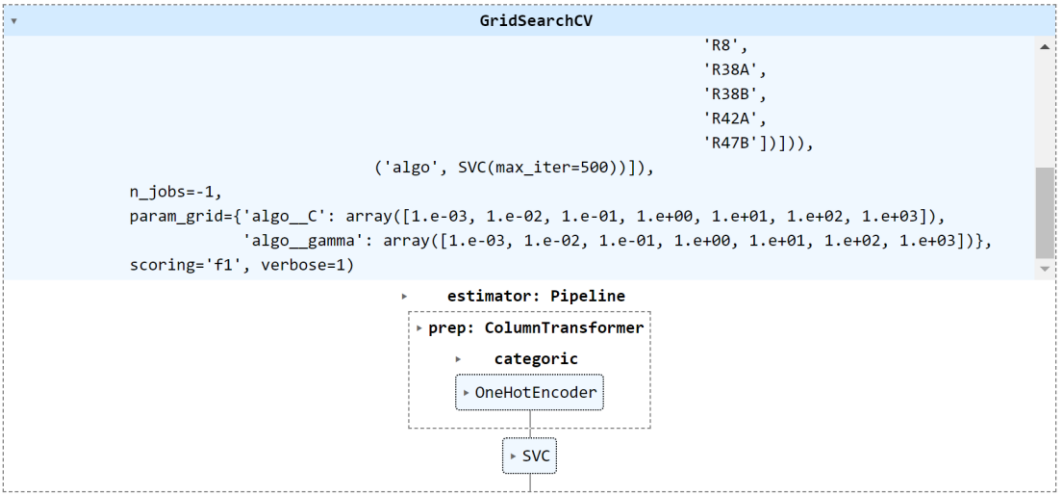
Langkah selanjutnya untuk meningkatkan hasil score dari model adalah menggunakan *Grid Search. Grid search* adalah teknik yang digunakan dalam peningkatan model *machine learning* untuk mencari kombinasi hyperparameter terbaik dari suatu model. Hyperparameter adalah parameter-parameter yang tidak dipelajari oleh model selama pelatihan, dan mereka perlu diatur sebelum proses pelatihan dimulai. Hyperparameter yang digunakan pada penelitian ini yaitu :

1. 'algo__C': Ini menunjukkan parameter C dari model SVM.

   Nilai-nilai yang dijelajahi adalah sejumlah nilai yang berada dalam rentang 10^{-3} sampai 10^{3}.

2. 'algo__gamma': Ini menunjukkan parameter gamma dari model SVM. Nilai-nilai yang dijelajahi adalah sejumlah nilai yang berada dalam rentang 10^{-3} sampai 10^{3}.

Selain parameter diatas, pada penelitian ini juga mengatur jumlah *cross validation* sebanyak 3, 4, 5, 6, 7 fold.



Gambar 4. 14 Grid Search dan Cross Validation SVM

4.2.4 Evaluasi Model

Berikut ini adalah hasil dari pengujian antara algoritma *K-Nearest Neighbors* (KNN) dan *Support Vector Machine* (SVM):

Tabel 4. 3 Hasil Pengujian antara KNN dan SVM

No	Algoritma	Pengujian Pertama	Dataset Splitting	GridSearchCV
1	KNN	81%	80%	**87%**
2	SVM	79%	54%	84%

Dari hasil pengujian pada penelitian ini dapat dilihat bahwa akurasi menurun saat dilakukan dataset splitting, lalu meningkat disaat ditambah perlakuan *GridSearch* dengan mengkombinasikan pula *Cross Validation*.

Algoritma *K-Nearest Neighbors* selalu lebih besar hasil akurasi nya dibandingkan dengan algoritma *Support Vector Machine* baik itu pada pengujian pertama, pengujuian dengan dilakukan dataset splitting, maupun setelah ditambahkan perlakuan *Grid Search* yang dikombinasikan dengan *Cross Validation* yaitu diangka 87% dimana *Support Vector Machine* berada di angka 84%, 3 angka dibawah algoritma *K-Nearest Neighbors.*

Pada Penelitian ini juga mengkombinasikan jumlah Cross Validation. Pada Algoritma KNN mengkombinasikan Cross Validation sebanyak 3, 4, 5, 6, sedangkan pada Algoritma SVM mengkombinasikan Cross Validation sebanyak 3, 4, 5, 6, 7.

Tabel 4. 4 Hasil Pengujian Antara KNN dan SVM dengan Kombinasi Jumlah CV

CV	**SVM**	**KNN**
3	0.8118673766338674	0.**87**41325189877114
4	0.**84**19987511926093	0.**87**43998525925529
5	0.8175338074583898	0.**87**42844104781478
6	0.**84**44999974544306	0.**87**46539638603877
7	0.8144522119799884	

4.3 *Eksploratory Data Analysis* (EDA)

Pada bab ini, penelitian akan menjelaskan proses eksplorasi data (EDA) yang dilakukan untuk memahami karakteristik dan pola data yang digunakan dalam penelitian. EDA merupakan tahapan kritis dalam siklus analisis data, yang bertujuan untuk merinci struktur, distribusi, dan relasi antar variabel dalam dataset. Dengan merinci informasi ini, penelitian dapat memperoleh pemahaman yang lebih mendalam tentang data, memastikan kualitas data yang diolah, dan menemukan wawasan yang mungkin berguna dalam pembangunan model prediksi pengangguran. Analisis ini mencakup visualisasi data, statistik deskriptif, dan eksplorasi hubungan antar variabel. Pemahaman mendalam terhadap data dapat membantu memvalidasi asumsi, mengidentifikasi outlier, dan mengeksplorasi pola yang mungkin relevan dalam konteks penelitian ini.

Penelitian akan membahas distribusi variabel kunci yang terkait dengan pengangguran di Provinsi Lampung. Selanjutnya, penjelasan mengenai tren dan pola-pola khusus akan disajikan untuk membantu membimbing pemilihan variabel yang signifikan untuk dimasukkan dalam model prediksi. Melalui eksplorasi ini, diharapkan penelitian dapat menggambarkan dengan jelas konteks data yang digunakan dan memberikan dasar yang kokoh untuk interpretasi hasil yang diperoleh dari model *K-Nearest Neighbors* (KNN) dan *Support Vector Machine* (SVM) yang akan dikembangkan pada bab-bab selanjutnya.

Langkah-langkah EDA yang dilakukan dalam penelitian ini akan diuraikan secara rinci, termasuk jenis visualisasi yang digunakan dan analisis statistik yang diterapkan. Proses ini menjadi landasan untuk pengambilan keputusan yang