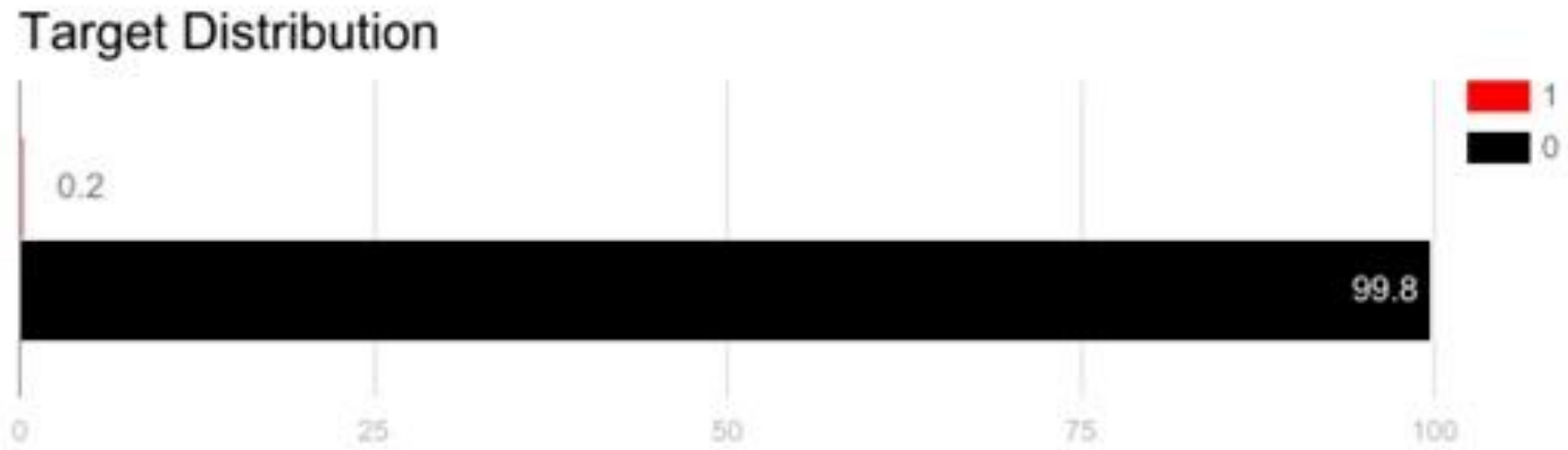
Data Mining: Data Imbalanced

Kuliah : 08-11-2023

Pengertian imbalanced data

- Imbalanced data atau data yang tak seimbang merupakan suatu kondisi dimana pada sebuah himpunan data terdapat satu kelas yang memiliki jumlah instance yang kecil bila dibandingkan dengan kelas lainnya.
- Imbalanced Dataset merupakan data yang biasanya diolah secara klasifikasi dengan salah satu kelas/label pada datanya mempunyai nilai yang sangat jauh berbeda jumlahnya dari kelas lainnya.
- Pada imbalanced dataset, biasanya kita memiliki data dengan kelas yang sedikit dan data dengan kelas yang banyak.

- Kelas yang tidak seimbang adalah masalah umum dalam klasifikasi pembelajaran mesin di mana terdapat rasio yang tidak proporsional di setiap kelas.
- Ketidakseimbangan kelas dapat ditemukan di berbagai bidang termasuk diagnosa medis, penyaringan spam, dan deteksi penipuan, klasifikasi emosi dll.
- Bagaimana contoh data yang tidak seimbang yaitu seperti deteksi penipuan di perbankan.
- Data ini merupakan kejadian langka dimana hanya 1% kemungkinan terjadinya penipuan diperbankan.
- Sebagian besar algoritma pembelajaran mesin tidak bekerja dengan baik dengan dataset tidak seimbang.
- Salah satu tiga teknik berikut dapat membantu, untuk melatih classifier untuk mendeteksi kelas abnormal/imbalanced.


Target Distribution
1
0
0.2
99.8
0
25
50
75
100

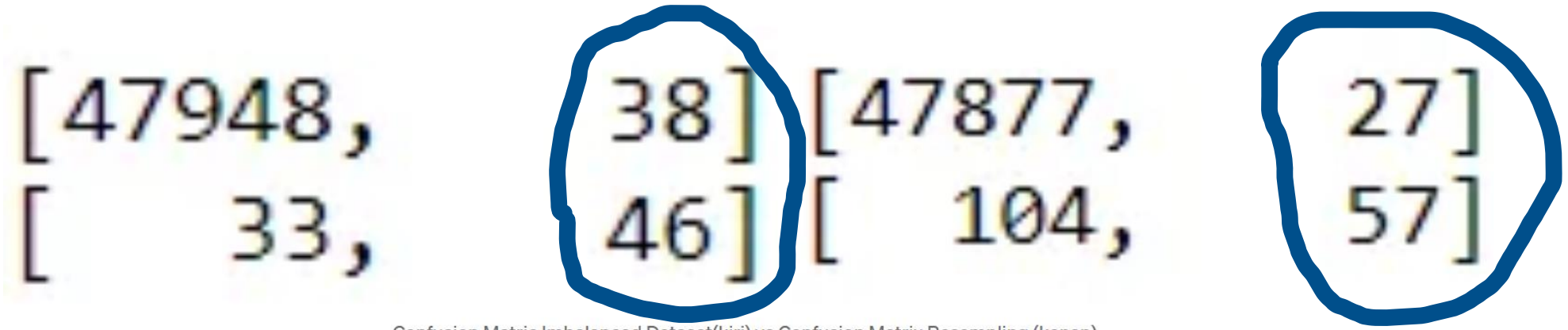
Degree of imbalance	Proportion of Minority Class
Mild	20-40% of the data set
Moderate	1-20% of the data set
Extreme	<1% of the data set

https://developers.google.com/machine-learning/data-prep/construct/sampling-splitting/imbalanced-data

- Mengapa perlu dilakukan Resampling? Mengapa tidak membiarkan mesin/komputer belajar dari data yang imbalance saja?
- Kalimat di atas merupakan segelintir pertanyaan yang paling sering disampaikan.
- Imbalanced dataset bukanlah permasalahan sepele yang dapat diabaikan.
- Dengan membiarkan mesin/komputer learning dari data tersebut dapat menyebabkan model yang terbentuk tidak bagus. Perhatikan Nilai akurasi model berikut :

```
Akurasi : 0.9985228336627484
Akurasi : 0.9972745240819724
```

- Nilai akurasi di atas menjelaskan akurasi menggunakan algoritma klasifikasi yang sama.
- Akurasi pertama diperoleh dari data yang imbalace sedangkan yang kedua diperoleh setelah data dilakukan resampling.
- Mungkin terlihat kontribusi resampling tidaklah terlalu signifikan



Confusion Matric Imbalanced Dataset(kiri) vs Confusion Matrix Resampling (kanan)

- Dari kedua confusion matrix di atas dapat terlihat model yang diperoleh dari learning menggunakan imbalanced dataset memiliki kecenderungan prediksi ke arah majority class dan tidak dapat mengenali dengan baik dari minority class karena minim jumlahnya ketika proses learning.
- Berbeda dengan data yang sudah melalui resampling, disini model yang terbentuk dapat mengenali dengan lebih baik untuk kedua kelas/label.
- Pada model dengan imbalanced dataset hanya dapat menebak minority class sebanyak 46 dari 84 atau sekitar 0.55%, sedangkan model dengan resampling mampu menebak 57 dari 84 atau sekitar 0.68%.
- Terlihat ada peningkatan sekitar 13% ketika menggunakan resampling.
- Bayangkan dari kasus data tersebut, untuk setiap 1% anda dapat menyelamatkan uang anda atau perusahaan anda sebesar 10, 20, atau 50 juta rupiah.

1-Gunakan metrik evaluasi yang tepat

- Menerapkan metrik evaluasi yang tidak tepat untuk model yang dihasilkan menggunakan data yang tidak seimbang bisa berbahaya.
- Bayangkan jika data pelatihan adalah yang diilustrasikan dalam grafik di atas.
- Jika akurasi digunakan untuk mengukur ketepatan suatu model, model yang mengklasifikasikan semua sampel pengujian menjadi “0” akan memiliki akurasi yang sangat baik (99,8%), tetapi jelas, model ini tidak akan memberikan informasi berharga.
- Dalam hal ini, metrik evaluasi alternatif lain dapat diterapkan seperti:
  - Presisi / Spesifisitas: berapa banyak instance terpilih yang relevan.
  - Recall / Sensitivitas: berapa banyak instance yang relevan dipilih.
  - F1 Score: rata-rata harmonis dari presisi dan recall.
  - MCC: koefisien korelasi antara klasifikasi biner yang diamati dan yang diprediksi.
  - AUC: hubungan antara tingkat true-positive dan false positive.

Predicted Class

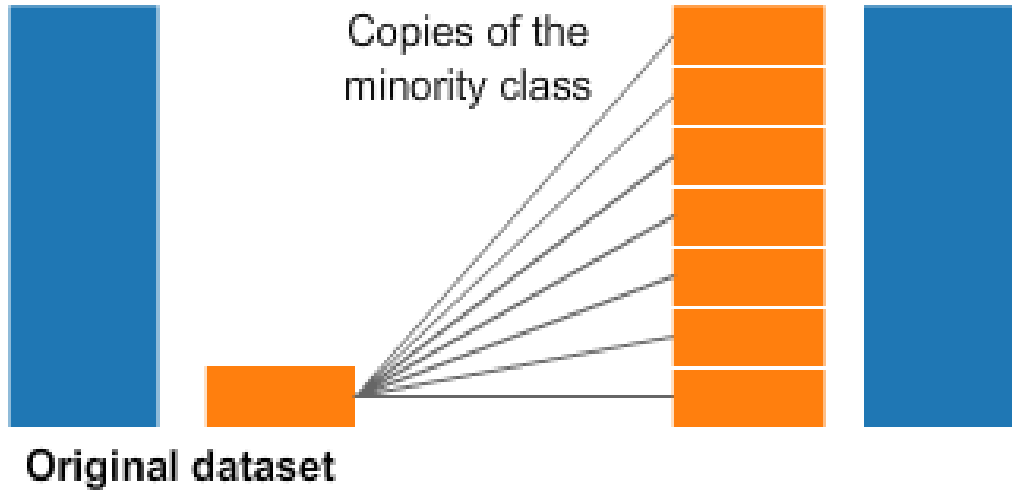
Actual Class	**Positive**	**Negative**	
Positive	True Positive (TP)	False Negative (FN) **Type II Error**	**Sensitivity** $\frac{TP}{(TP + FN)}$
Negative	False Positive (FP) **Type I Error**	True Negative (TN)	**Specificity** $\frac{TN}{(TN + FP)}$
	Precision $\frac{TP}{(TP + FP)}$	**Negative Predictive Value** $\frac{TN}{(TN + FN)}$	**Accuracy** $\frac{TP + TN}{(TP + TN + FP + FN)}$

2-Resample set pelatihan

- **Undersampling** menyeimbangkan dataset dengan mengurangi ukuran kelas yang berlimpah→Metode ini digunakan ketika jumlah data mencukupi.
- Dengan menjaga semua sampel di kelas langka dan secara acak memilih jumlah sampel yang sama di kelas berlimpah, dataset baru yang seimbang dapat diambil untuk pemodelan lebih lanjut.
- **Oversampling** digunakan ketika jumlah data tidak mencukupi.
- Mencoba menyeimbangkan dataset dengan meningkatkan ukuran sampel langka.
- Daripada membuang sampel berlimpah, sampel langka baru dihasilkan dengan menggunakan mis. SMOTE (Sintetis Minoritas Sampling Teknik).
- Tidak ada keunggulan absolut dari satu metode resampling atas yang lain.
- Penerapan kedua metode ini bergantung pada use case yang digunakan dan dataset itu sendiri.
- Kombinasi over-dan under-sampling sering berhasil juga.

Undersampling

Oversampling



3-Gunakan K-fold Cros-Validation dengan cara yang benar

- Patut dicatat bahwa validasi silang harus diterapkan dengan benar saat menggunakan oversampling untuk mengatasi masalah ketidakseimbangan.
- Perlu diingat bahwa pengambilan oversampling mengambil sampel langka yang diamati dan menerapkan bootstrap untuk menghasilkan data acak baru berdasarkan fungsi distribusi.
- Jika Cross-Validation diterapkan setelah oversampling, pada dasarnya yang dilakukan adalah overfitting model dengan hasil bootstrap buatan tertentu.
- Itulah sebabnya Cross-Validation harus selalu dilakukan sebelum oversampling data, seperti halnya bagaimana pemilihan fitur harus dilaksanakan.
- Hanya dengan melakukan resampling data berulang kali, keacakan dapat dimasukkan ke dalam dataset untuk memastikan bahwa tidak akan ada masalah overfitting.

Metode resampling

Oversampling

SMOTE

ADASYN

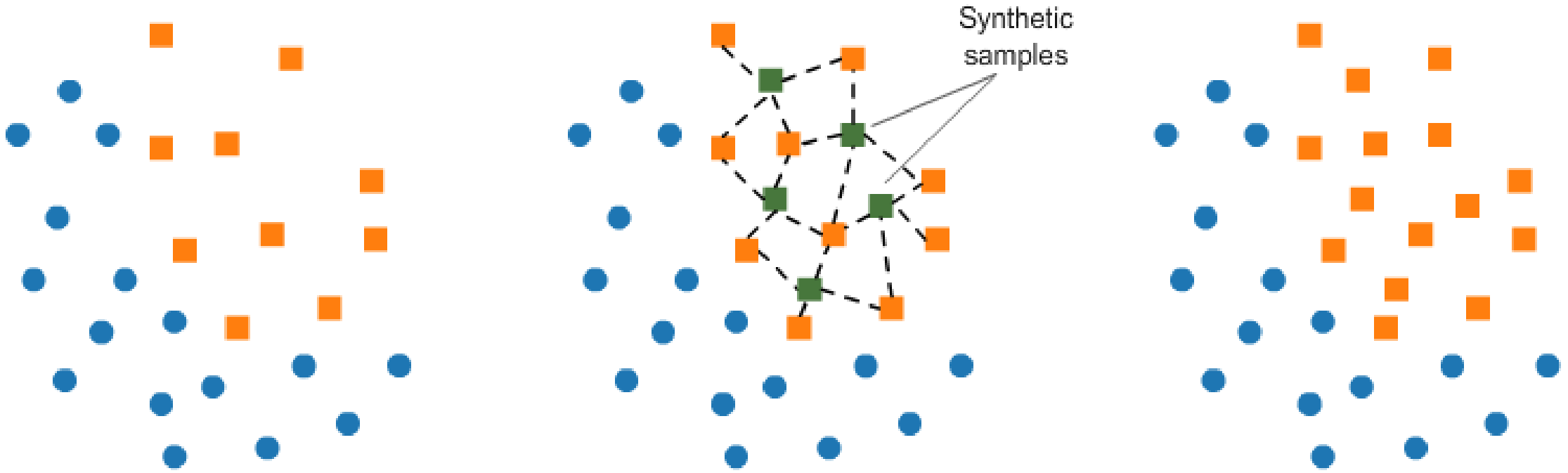
Undersampling

Random undersampling

Edited Nearest Neighbors (ENN)

Neighborhood Cleaning Rule (NCL)

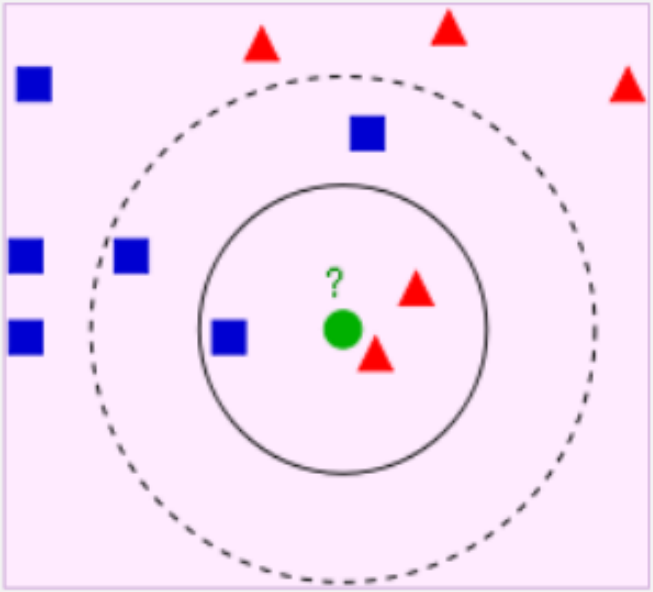
- SMOTE (Synthetic Minority Oversampling Technique), Menambah sampel kelas minoritas dengan cara mensintesis data baru berdasarkan metode k-nearest neighbour.



Algoritma k-nn

- KNN adalah sebuah metode klasifikasi terhadap sekumpulan data berdasarkan pembelajaran data yang sudah terklasifikasikan sebelumya.
- Termasuk dalam *supervised learning*, dimana hasil *query instance* yang baru diklasifikasikan berdasarkan mayoritas kedekatan jarak dari kategori yang ada dalam KNN.
- Diberikan titik query, akan ditemukan sejumlah k obyek atau (titik training) yang paling dekat dengan titik query.
- Klasifikasi menggunakan voting terbanyak diantara klasifikasi dari k obyek
- Algoritma k-nearest neighbor (KNN) menggunakan klasifikasi ketetanggaan sebagai nilai prediksi dari query instance yang baru.

- Jika k=3 maka lingkaran hijau akan masuk dalam kategori segitiga merah
- Jika k=5 maka lingkaran hijau akan masuk dalam kategori segiempat biru.



Ukuran Jarak

- Dekat atau jauhnya tetangga biasanya dihitung berdasarkan *Euclidean Distance.*

$$D(a,b) = \sqrt{\sum_{k=1}^{d}(a_k - b_k)^2},$$

- Dimana D(a,b) adalah jarak skalar dari dua buah vektor data a dan b yang berupa matrik berukuran d dimensi.

Beberapa macam jarak yang dapat digunakan

- Jarak Euclidean

$$d(x,y) = \|x - y\| = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

- Jarak Manhattan atau Cityblock

$$d(x,y) = \sum_{i=1}^{n}(|x_i - y_i|)$$

- Jarak Minkowski

$$d(x,y) = \|x - y\|_q = (\sum |x - y|^q)^{\frac{1}{q}}$$

- Jarak Mahalanobis

$$d(\vec{x},\vec{y}) = \sqrt{(\vec{x} - \vec{y})^T S^{-1}(\vec{x} - \vec{y})}$$

Algoritma

1. Menentukan parameter k (jumlah tetangga paling dekat).
2. Menghitung kuadrat jarak eucliden objek terhadap data training yang diberikan.
3. Mengurutkan hasil no 2 secara ascending
4. Mengumpulkan kategori Y (klasifikasi nearest neighbor berdasarkan nilai k)
5. Dengan menggunakan kategori nearest neighbor yang paling mayoritas maka dapat dipredisikan kategori objek .

X1 = Daya tahan asam (detik)	X2 = Kekuatan (Kg/m^2)	Y = Klasifikasi
8	4	Baik
4	5	Jelek
4	6	Jelek
7	7	Baik
5	6	Jelek
6	5	Baik

- Terdapat beberapa data yang berasal dari survey questioner tentang klasifikasi kualitas kertas tissue apakah baik atau jelek, dengan objek training menggunakan dua attribute yaitu daya tahan terhadap asam dan kekuatan. Dengan menggunakan K = 4.
- Akan diproduksi kembali kertas tisu dengan attribute X1=7 dan X2=4 tanpa harus mengeluarkan biaya untuk melakukan survey, maka dapat diklasifikasikan kertas tise tersebut termasuk yang baik atau jelek.

Proses perhitungan jarak

X1 = Daya tahan asam (detik)	X2 = Kekuatan (kg/m^2)	Square distance to query distance (7,4)
8	4	$(8-7)^2 + (4-4)^2 = 1$
4	5	$(4-7)^2 + (5-4)^2 = 10$
4	6	$(4-7)^2 + (6-4)^2 = 13$
7	7	$(7-7)^2 + (7-4)^2 = 9$
5	6	$(5-7)^2 + (6-4)^2 = 8$
6	5	$(6-7)^2 + (5-4)^2 = 2$

X1= Daya tahan asam (detik)	X2= Kekuatan (Kg/m^2)	Square distance to query distance (7,4)	Jarak terkecil	Apakah termasuk nearest neighbor (K)	Y= kategori nearest neighbor
8	4	$(8-7)^2 + (4-4)^2 = 1$	1	Ya	Baik
4	5	$(4-7)^2 + (5-4)^2 = 10$	5	Tidak	-
4	6	$(4-7)^2 + (6-4)^2 = 13$	6	Tidak	-
7	7	$(7-7)^2 + (7-4)^2 = 9$	4	Ya	Baik
5	6	$(5-7)^2 + (6-4)^2 = 8$	3	Ya	Jelek
6	5	$(6-7)^2 + (5-4)^2 = 2$	2	Ya	Baik

1. Ada 4 data yang paling dekat yaitu (8,4) , (6,5) , (5,6), dan (7,7). Kemudian hitung jumal kelas untuk ke empat data tersebut.
2. Sehingga diperoleh baik = 3 dan jelek = 1. Dengan voting maka diperoleh bahwa tissue dengan daya tahan 7 dan Kekuatan 4
3. Kesimpulan : termasuk kategori Baik.

X1 = Takaran saji (gr)	X2 = Jumlah Saji perkemasan	X3 = Energi Total	Y = Klasifikasi
40	5	60	Jelek
50	8	40	Bagus
50	7	30	Jelek
70	4	60	Bagus
80	4	80	Bagus
60	6	60	Bagus

- Tentukan *class* dari test data dengan nilai atribut (50,3,40)

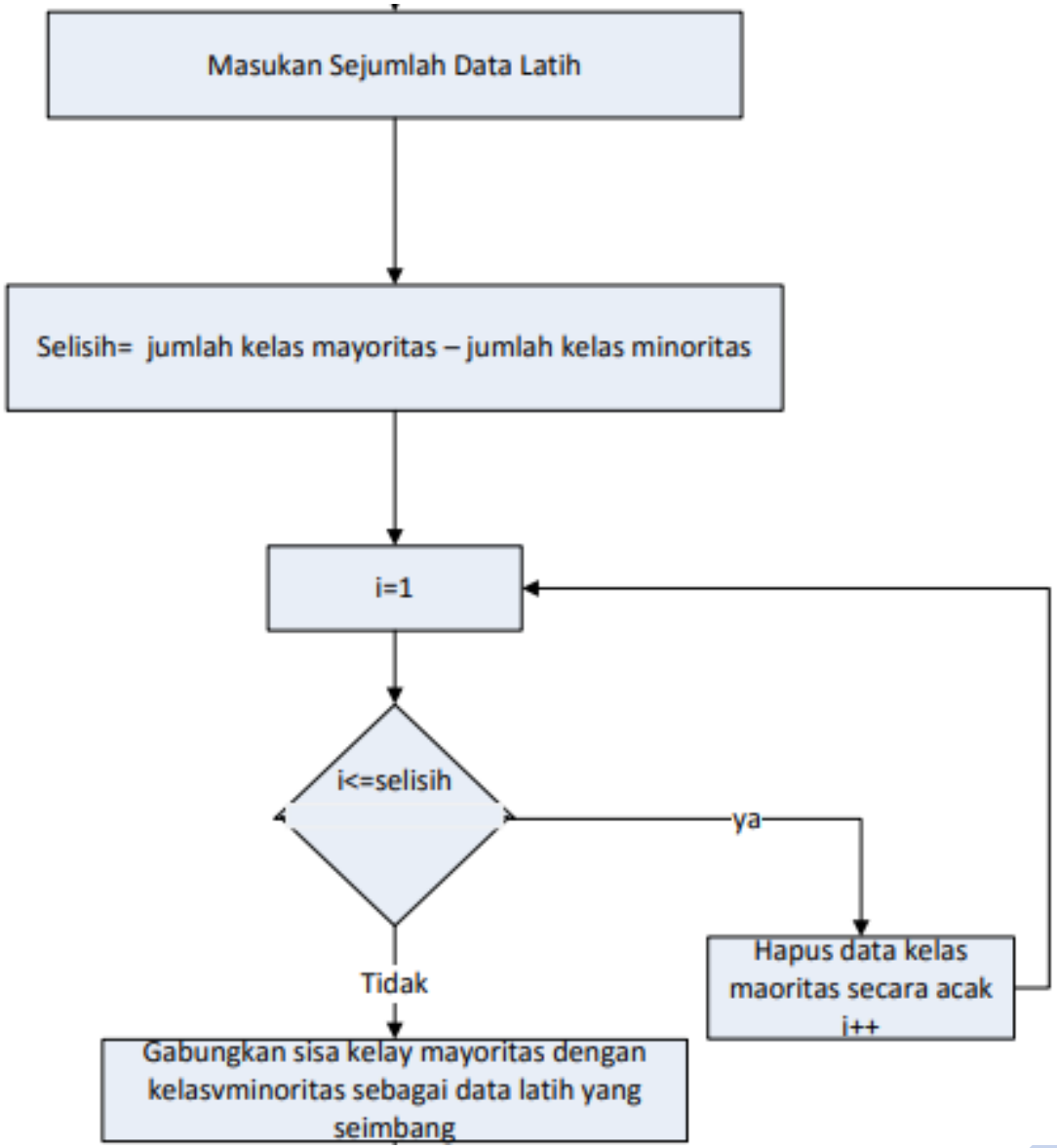
X1 = Takaran saji (gr)	X2 = Jumlah Saji perkemasan	X3 = Energi Total	Square distance to query distance (50,3,40)
40	5	60	$(40-50)^2 + (5-3)^2 + (60-40)^2 = 504$
50	8	40	$(50-50)^2 + (8-3)^2 + (40-40)^2 = 25$
50	7	30	$(50-50)^2 + (7-3)^2 + (30-40)^2 = 116$
70	4	60	$(70-50)^2 + (4-3)^2 + (60-40)^2 = 801$
80	4	80	$(80-50)^2 + (4-3)^2 + (80-40)^2 = 2501$
60	6	60	$(60-50)^2 + (6-3)^2 + (60-40)^2 = 509$

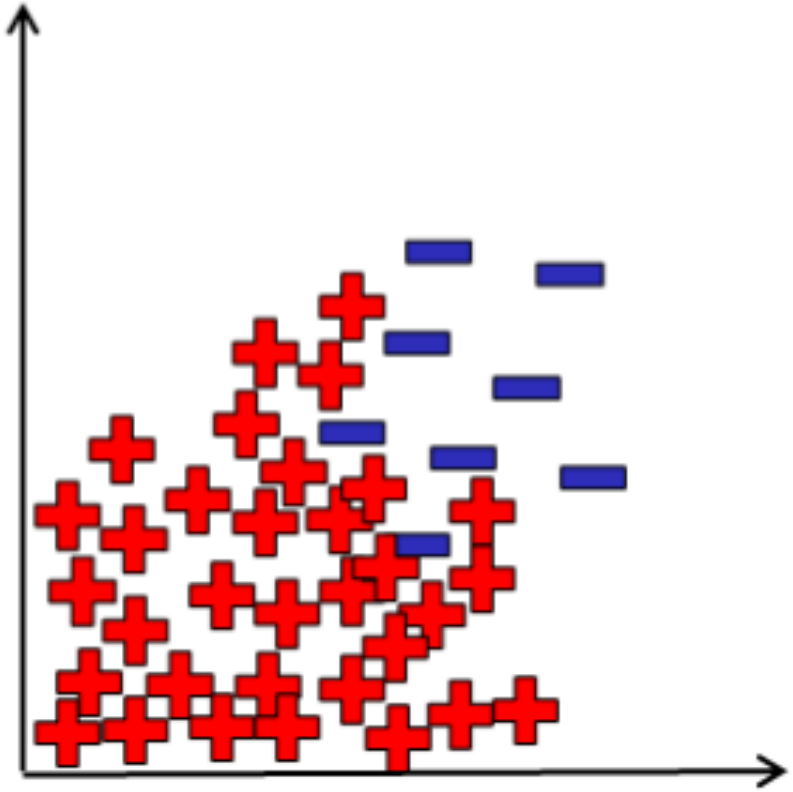
X1 = Takaran saji (gr)	X2 = Jumlah Saji perkemasan	X3 = Energi Total	Square distance to query distance (50,3,40)	Jarak Terkecil	Apakah termasuk nearest neighbor (K)	Y= kategori nearest neighbor
40	5	60	$(40-50)^2 + (5-3)^2 + (60-40)^2 = 504$	3	Ya	Jelek
50	8	40	$(50-50)^2 + (8-3)^2 + (40-40)^2 = 25$	1	Ya	Bagus
50	7	30	$(50-50)^2 + (7-3)^2 + (30-40)^2 = 116$	2	Ya	Jelek
70	4	60	$(70-50)^2 + (4-3)^2 + (60-40)^2 = 801$	5	Tidak	-
80	4	80	$(80-50)^2 + (4-3)^2 + (80-40)^2 = 2501$	6	Tidak	-
60	6	60	$(60-50)^2 + (6-3)^2 + (60-40)^2 = 509$	4	Ya	Bagus

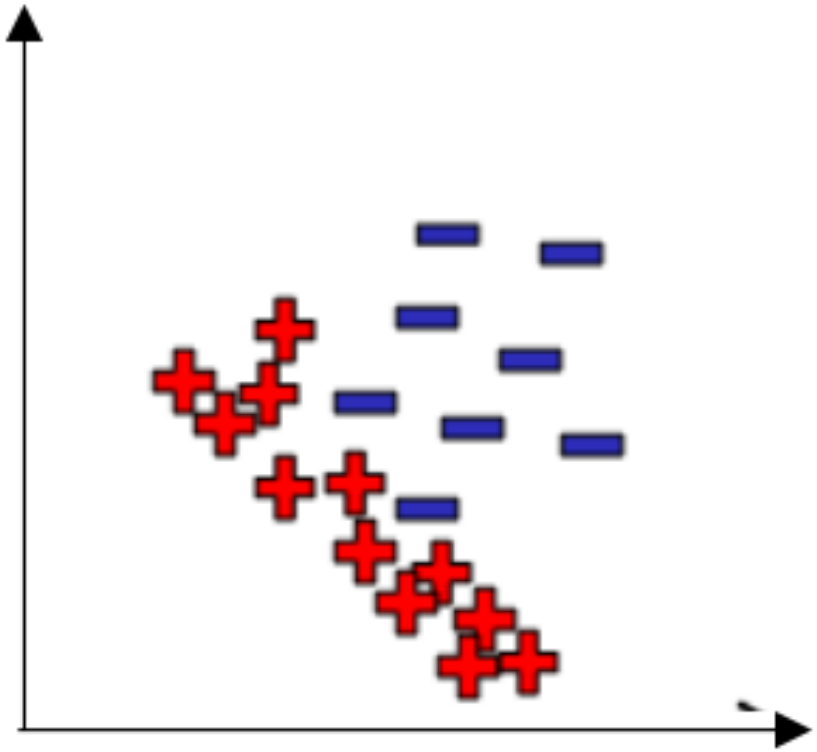
- Jika digunakan K = 4 maka voting akan seimbang Bagus = 2 dan Jelek = 2. Untuk menanggulangi hal tersebut maka nilai K dikurangi 1 untuk setiap ditemukan hasil voting yang seimbang.
- Dengan K = 3 maka voting kelas Bagus = 1 dan Jelek = 2. Maka *class* dari test data dengan nilai atribut (50,3,40) termasuk Kelas yang **Jelek.**

Random undersampling

- Random under sampling (RUS) menghitung selisih antara kelas mayoritas dan minoritas kemudian dilakukan perulangan selisih hasil perhitungan, selama perulangan data kelas mayoritas dihapus secara acak, sehingga jumlah kelas mayoritas sama dengan minoritas
- Langkah pertama pada Random Under Sampling adalah pemilihan dataset kemudian dihitung selisih antara kelas mayoritas dan minoritas, jika masih terdapat selisih antara jumlah kelas maka dataset kelas mayoritas akan dihapus secara acak sampai jumlah kelas mayoritas sama dengan kelas minoritas.


Masukan Sejumlah Data Latih
Selisih= jumlah kelas mayoritas – jumlah kelas minoritas
i=1
i<=selisih
ya
Hapus data kelas maoritas secara acak
i++
Tidak
Gabungkan sisa kelay mayoritas dengan kelasvminoritas sebagai data latih yang seimbang





Edited Nearest Neighbors (ENN)

Konsep ENN→data dari kelas mayoritas yang dianggap sebagai noise akan dikurangi sehingga dapat meningkatkan kualitas data yang dihasilkan.

Algoritma ini akan memeriksa setiap data kelas mayoritas dengan kelas mayoritas dan kelas minoritas terdekatnya.

Apabila kelas mayoritas lebih dekat dengan tetangga kelas minoritas maka data tersebut dianggap sebagai noise dan dihapus dari dataset.

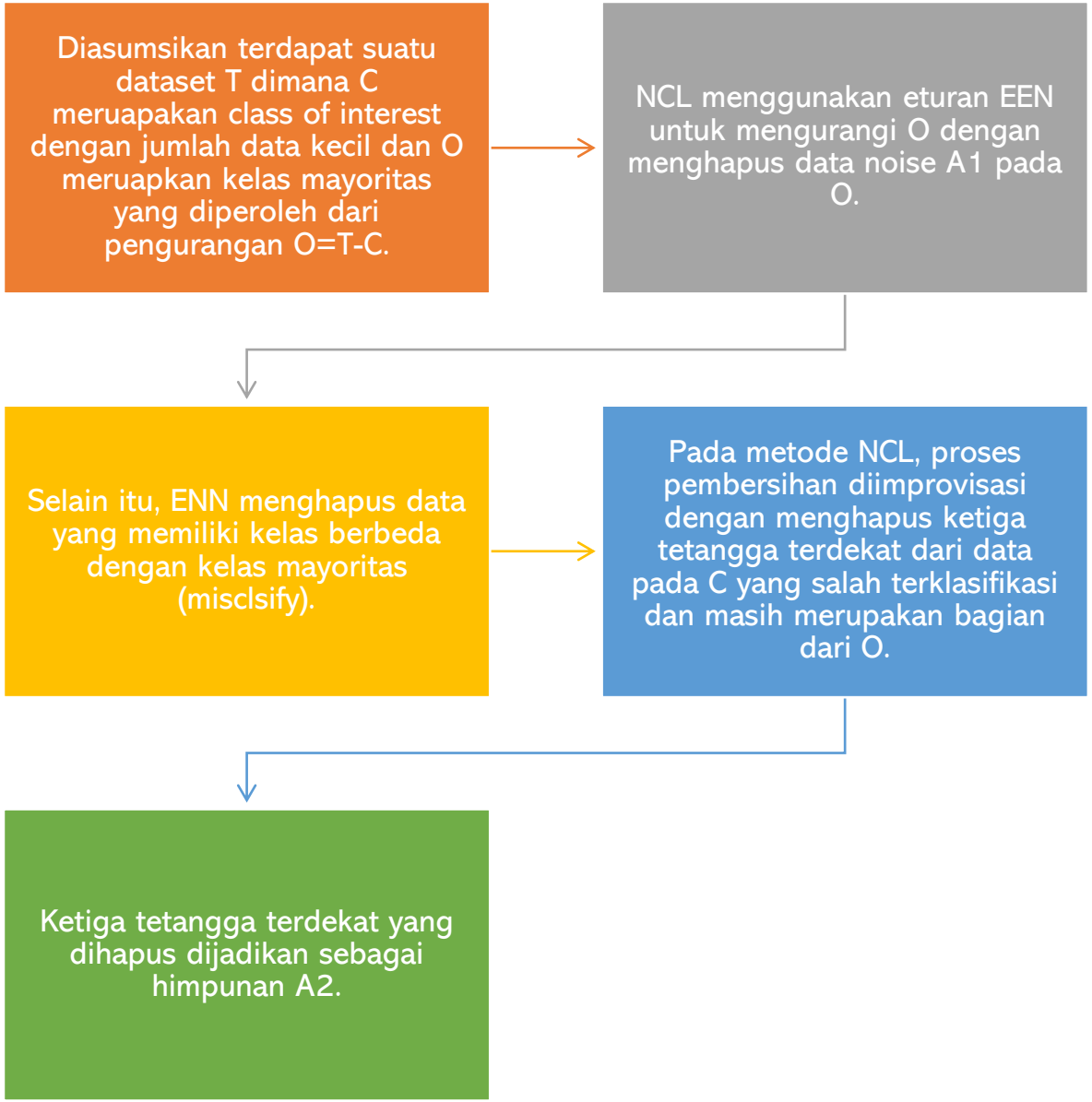
- Aturan ini menyatakan bahwa sampel tetangga terdekat dari setiap sampel mayoritas ditemukan berdasarkan jarak antara dua sampel dan identifikasi apakah sampel mayoritas merupkan noise sampel atau bukan dengan menilai konsistensi label tetangga terdekatnya
- Tetangga terdekat sebanyak k(k-nearest neighbor) dari sampel xi secara matematis dinyatakan:
- kNN(xi,k)=$\{yi \in X \mid dist(xj,xi) \leq dist(xi,,xi)\}$
- dimana xi, adalah sampel tetangga terdekat ke-k dari xi pada dataset X dan distadalah jarak antar sampel xi dan tetangga terdekatnya yang umumnya menggunakan rumus **euclidean distance**

Neighborhood Cleaning Rule (NCL)

- J. Laurikala menemukan salah satu metode undersampling untuk mengatasi distribusi kelas yang imbalanced dengan metode data berbasis cleaning
- Salah satu kelebihan pada NCL adalah NCL sangat mempertimbangkan kualitas dari data yang akan di hapus dengan tidak berfokus pada reduksi data saja melainkan berfokus pda pembersihan data.
- Proses cleaning data diperuntukkan tidak hanya untuk sample pada kelas mayoritas, namun juga minoritas.

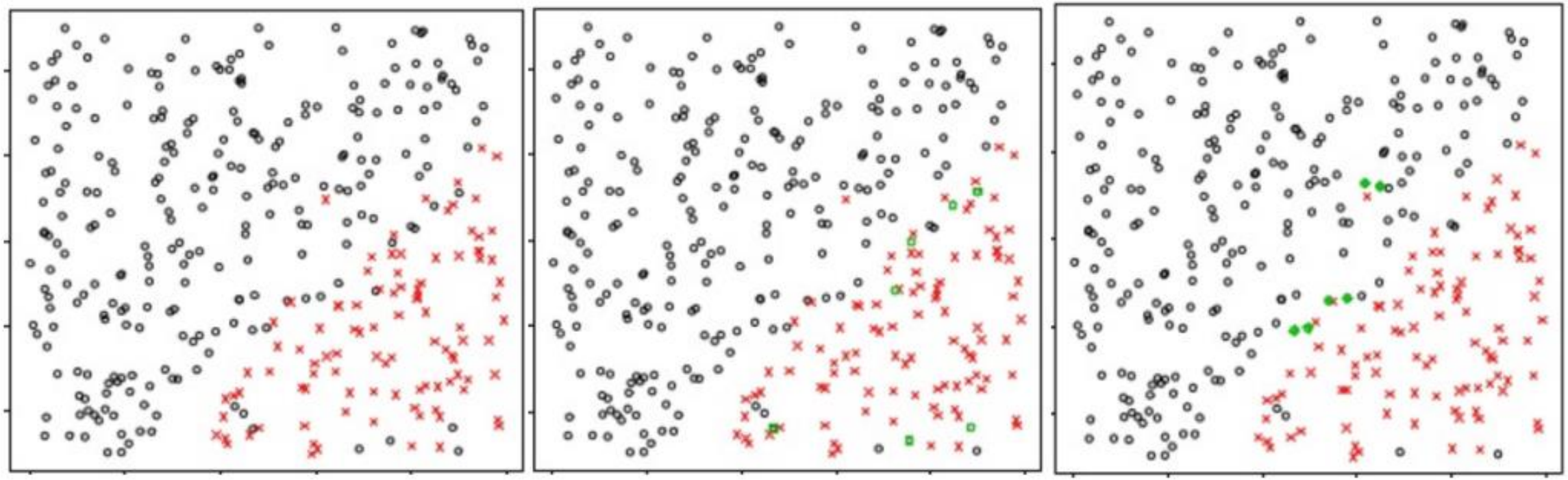
- NCL bekerja berdasarkan pada konsep one-side selection (OSS) yang merupakan salah satu teknik untuk mereduksi data berbasis instance untuk mereduksi kelas.
- Tujuan utamanya untuk mengurangi data yang tidak relevan, secara hati-hati.
- Proses cleaning pada NCL diterapkan pada sample mayoritas dan minoritas secara terpisah.
- NCL mengadopsi metode Edited Nearest Neighbor (ENN) untuk membersihkan data pada kelas mayoritas.
- Contoh : E1 termasuk sebagai kelas mayoritas dan hasil klasifikasi ternyata berlawanan dengan kelas original pada E1, maka E1 akan dihapus, sebaliknya apabila E1 merupakan kelas minoritas dan ketiga anggotanya terklasifikasi berlawanan (mayoritas), maka tetangga terdekat akan dihapus.

Algoritma NCL



Algoritma NCL

1.	Split data T menjadi *interest class* C dan data lain O
2.	Identifikasi data *noise* A_1 pada O dengan menggunakan *Edited Nearest Neighbor* (ENN)
3.	Pada setiap kelas C_i pada O
	Jika $(x \in Ci)$pada 3 tetangga terdekat dari misklasifikasi $y \in C)$ dan $(\|Ci\| > 0.5\|C\|)$ maka $A_2 = \{x\} \cup A_2$
4.	Data reduksi $S = T - (A_2 \cup A_2)$



Original data

Dataset setelah ENN

Dataset setelah NCL

Rangkuman

Metode	Jenis *Resampling*	Cara Kerja
ROS	*Oversampling*	ROS bekerja dengan cara memilih *instance* dalam kelas minoritas secara acak kemudian melakukan duplikasi berulang kali sampai jumlah *instance* pada kelas minoritas dan mayoritas menjadi seimbang. Pendekatan ini memiliki kelemahan karena menghasilkan banyak duplikasi *instance* yang akhirnya dapat menimbulkan masalah *overfitting*.
SMOTE	*Oversampling*	Pendekatan lain yang cenderung lebih cerdas daripada ROS adalah SMOTE yang bekerja dengan cara mengambil secara acak tetangga terdekat sebanyak *k* dari setiap *instance* dalam kelas minoritas kemudian membuat *instance* baru (sintetis) antara *instance* tersebut dengan tetangga terdekat *k* yang dipilih secara acak. Dengan pendekatan SMOTE maka dapat dipastikan tidak terjadi masalah duplikasi data sehingga lebih kebal terhadap masalah *overfitting*.
ADASYN	*Oversampling*	ADASYN bekerja dengan cara menggunakan bobot distribusi untuk *instance* pada kelas minoritas berdasarkan pada tingkat kesulitan pembelajaran data oleh model, di mana *instance* baru (sintetis) dihasilkan dari kelas minoritas yang susah untuk belajar dibandingkan dengan data minoritas yang lebih mudah untuk belajar.
RUS	*Undersampling*	RUS bekerja dengan cara memilih secara acak *instance-instance* pada kelas mayoritas kemudian menghapusnya. Proses ini dilakukan berulang kali sampai jumlah *instance* dalam kelas minoritas sama dengan kelas mayoritas. Pendekatan ini memiliki kelemahan karena membuat banyak *instance* penting terhapus sehingga dapat mempengaruhi kinerja *classifier*.
SMOTETomek	*Over-Under Sampling*	SMOTETomek adalah kombinasi dari SMOTE dan Tomek Link yang termasuk dalam kategori *over-under sampling* dimana teknik *oversampling* menggunakan SMOTE dan teknik *undersampling* adalah Tomek Link.
SMOTEENN	*Over-Under Sampling*	SMOTEENN adalah kombinasi antara SMOTE dan ENN dimana yang berperan sebagai *oversampling* adalah SMOTE sedangkan *undersampling* adalah ENN.

Pendalaman Materi

- Jelaskan cara kerja dengan menggunakan contoh kasus sederhana, untuk algoritma berikut
  - SMOTE
  - ADASYN
  - ENN
  - NCL
- Cara kerja dijelaskan per-tahapan
- Kerjakan secara kelompok, dengan anggota maksimal 5 mhs
- Hasilnya di upload di spada dengan format docx