K-Nearest Neighbors

CSGE603130 - Kecerdasan Artifisial dan Sains Data Dasar

Semester Genap 2021/2022

Pengajar: Siti Aminah

Slide ini mengambil dari slide KNN semester Gasal 2021/2022 dan Semester Gasal 2019/2020 yang disusun oleh Dina Chahyati, Siti Aminah, Dhimas Arief, dan Ari Wibisono dengan beberapa penambahan

UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia

FACULTY OF COMPUTER SCIENCE

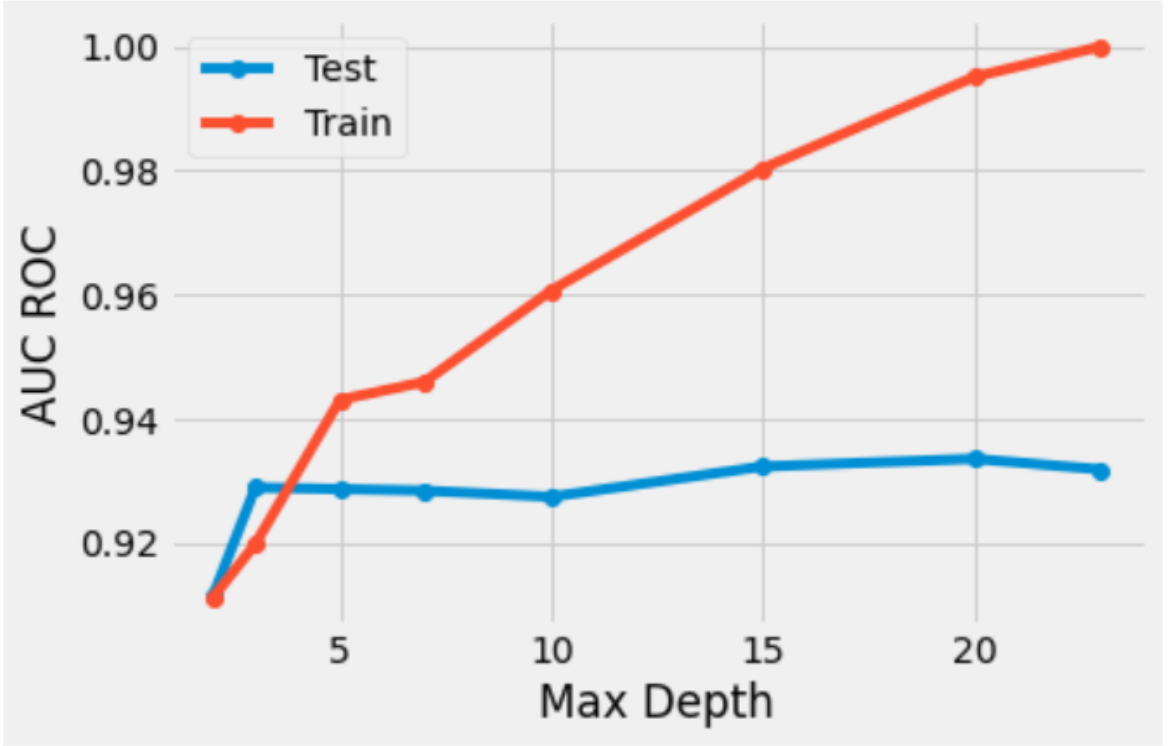
Review Topik Sebelumnya

1. Pernyataan berikut ini mana yang tidak benar terkait random forest yang terdiri dari M tree yang dibangun dari dataset berukuran N examples:
   a) Random forest tersebut akan memiliki variance yang lebih rendah dibandingkan sebuah decision tree
   b) Training set untuk sebuah tree pada random forest adalah sampel dengan ukuran N examples yang diambil secara acak dari dataset tanpa pengembalian
   c) Tidak semua atribut dipertimbangkan sebagai splitting point ketika membangun sebuah tree pada random forest
   d) Prediksi pada random forest tersebut dilakukan dengan mengambil mayoritas prediksi dari semua M tree untuk masalah klasifikasi
2. Jika training set terdiri dari M atribut, N baris, dan N jauh lebih kecil dari M, maka besar kemungkinan model prediksi dengan variance tinggi akan mengalami overfitting jika dilatih dengan training set tersebut. (T/F)
3. Decision tree yang dibangun menggunakan training set yang sudah distandarisasi dapat memberikan prediksi yang berbeda dibandingkan decision tree yang dibangun menggunakan training set yang sama tanpa standarisasi data. (T/F)
4. Apakah pruning itu? Apa tujuan pruning pada Decision Tree?

Review Topik Sebelumnya

Pre-Pruning

The pre-pruning technique refers to the early stopping of the growth of the decision tree. The pre-pruning technique involves tuning the hyperparameters of the decision tree model prior to the training pipeline. The hyperparameters of the decision tree including **`max_depth`**, **`min_samples_leaf`**, **`min_samples_split`** can be tuned to early stop the growth of the tree and prevent the model from overfitting.



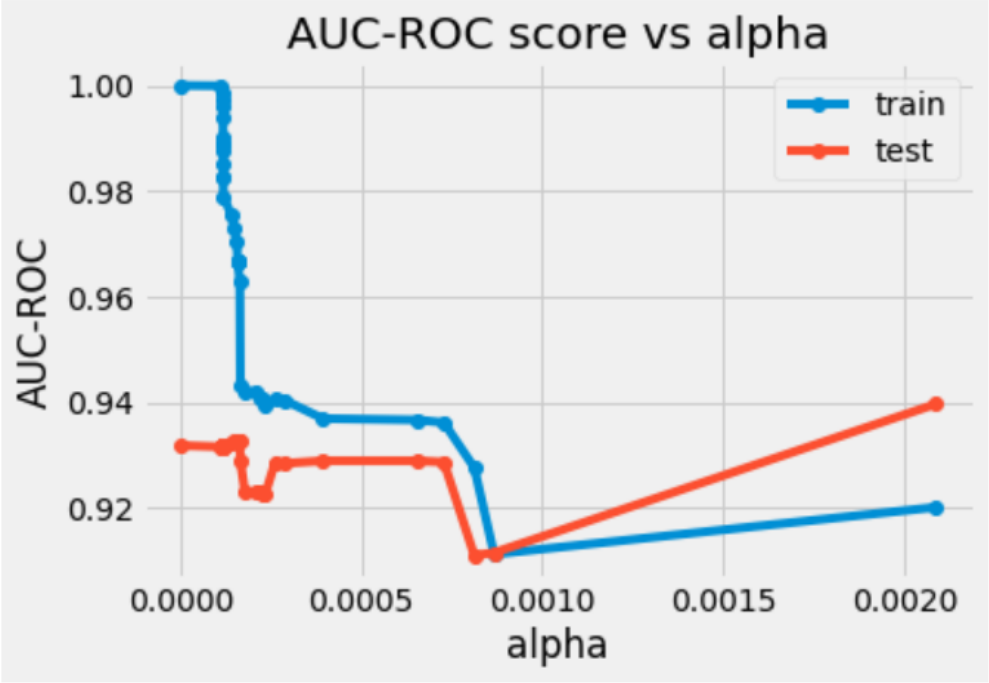
https://towardsdatascience.com/3-techniques-to-avoid-overfitting-of-decision-trees-1e7d3d985a09

Review Topik Sebelumnya

Post-Pruning

The Post-pruning technique allows the decision tree model to grow to its full depth, then removes the tree branches to prevent the model from overfitting. Cost complexity pruning (ccp) is one type of post-pruning technique. In case of cost complexity pruning, the **`ccp_alpha`** can be tuned to get the best fit model.

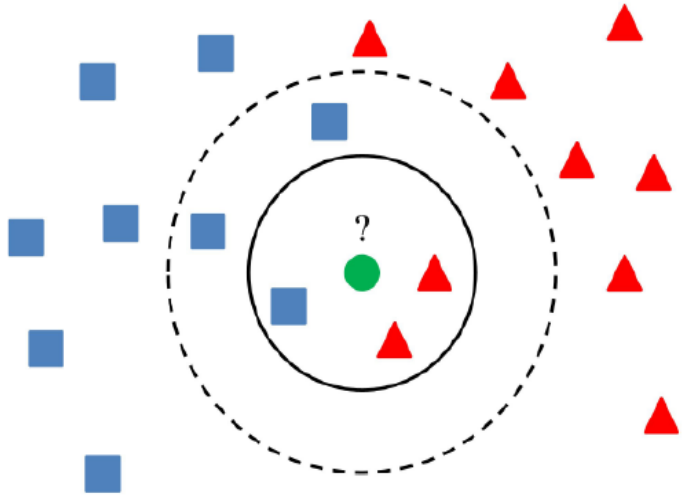
Scikit-learn package comes with the implementation to compute the **`ccp_alpha`** values of the decision tree using function cost_complexity_pruning_path(). With the increase in ccp_apha values, more nodes of the tree are pruned.



https://towardsdatascience.com/3-techniques-to-avoid-overfitting-of-decision-trees-1e7d3d985a09

Topik Hari ini: Model K-NN

Motivasi: Siapakah teman "karib" saya?



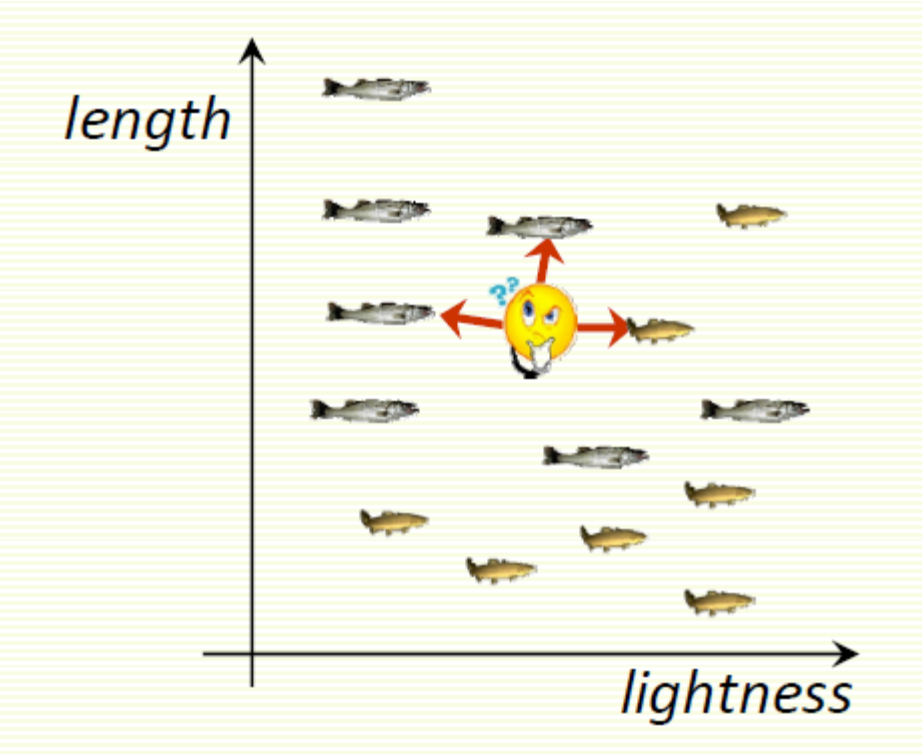
- Saya (lingkaran hijau) sedang berada di antara kumpulan segitiga merah dan persegi biru.
- Siapakah teman-teman "karib" saya?

KNN

- Simple
- Digunakan untuk **Klasifikasi** maupun **Regresi**
- Klasifikasi berdasarkan dengan perhitungan similaritas
- Lazy learning:
  - Tidak melakukan "learning" sampai diberikan testing data
  - Disebut juga "instant based learning"
    - Learning: storing all training instances
    - Classification: assigning target function to an instance

Contoh KNN

"tell me who your neighbors are, and I'll tell you who you are"

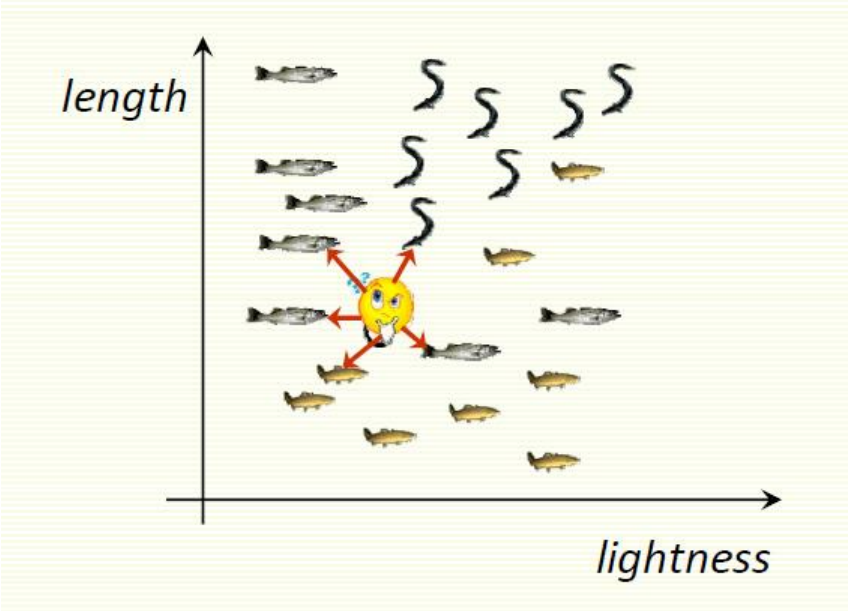


Contoh:

- k = 3
- 2 sea bass, 1 salmon
- Klasifikasikan sebagai sea bass

Contoh KNN

Mudah untuk mengimplementasikan klasifikasi multi-kelas



Contoh:

- k = 5
- 3 species: sea bass, salmon, eel
- 3 sea bass, 1 eel, 1 salmon
- Klasifikasikan sebagai sea bass

KNN

- Asumsikan dataset terdiri dari *N* pasangan *feature vectors* dan *labels* $(\mathbf{x}_i, y_i)$
- Prediksi label y untuk sampel baru $\mathbf{x}$ yang sering disebut sebagai *query example* atau *query*.
- Strategi dari *Nearest Neighbors* adalah menemukan sampel berlabel $\mathbf{x}_c$ yang paling dekat dengan $\mathbf{x}$, sehingga kelas tersebut merupakan hasil prediksi.
- Berdasarkan aturan, jika *feature vectors* $\mathbf{u}$ dan $\mathbf{v}$ cukup dekat, maka $p(y|\mathbf{u})$ serupa dengan $p(y|\mathbf{v})$.
- Artinya, jika sampel berlabel $\mathbf{x}_i$ dekat dengan $\mathbf{x}$, maka $p(y|\mathbf{x})$ serupa dengan $p(y|\mathbf{x}_i)$.

KNN

- Metode ini menuntut agar *queries* mirip dengan sampel-sampel pada dataset berlabel; sampel yang banyak muncul pada dataset berlabel akan muncul lebih sering dalam *queries*, begitu pula dengan sebaliknya.
- Bayangkan sebuah kondisi: *query* muncul pada lokasi di mana $p(y = 1|\mathbf{x}) \gg p(y \neq 1|\mathbf{x})$.
- Sampel pada dataset berlabel ($\mathbf{x}_c$) terdekat akan berada di sekitarnya sebab *queries* mirip dengan dataset berlabel.
- $\mathbf{x}_c$ akan berlabel 1 (sebab sampel yang berdekatan cenderung memiliki label yang sama), sehingga label prediksi adalah 1.
- Pada kondisi ini, *classifier* akan menghasilkan prediksi yang benar dengan probabilitas yang tinggi.

KNN

- Namun, bayangkan jika *query* muncul pada lokasi di mana $p(y = 1|\mathbf{x}) \approx p(y \neq 1|\mathbf{x})$.
- Sampel-sampel pada dataset berlabel yang berdekatan dengan *query* akan memiliki label yang bervariasi dikarenakan $p(y = 1|\mathbf{x}) \approx p(y \neq 1|\mathbf{x})$.
- Andaipun hasil prediksi label adalah 1, maka sedikit perubahan pada query akan menyebabkan kemungkinan prediksi label berubah menjadi -1 (begitu pula sebaliknya)
- Pada kondisi ini, *classifier* akan cenderung lebih sering menghasilkan kesalahan.
- Penggunaan sampel berlabel dalam jumlah sangat besar mungkin dapat membantu.
- Namun, hal tersebut hampir tidak mungkin untuk dilakukan.
- Lalu, bagaimana solusinya?

KNN

- Hal terpenting dari generalisasi adalah menemukan k nearest neighbors, kemudian pilih label dari kandidat tersebut.
- Sebuah (k, l) nearest neighbor classifier akan memilih k sampel berlabel yang paling dekat dengan query dan mengklasifikasikan query dengan kelas yang memiliki jumlah votes tertinggi, selama kelas memiliki lebih dari l votes.
- Jika tidak, query akan diklasifikasikan sebagai unknown.
- Pada praktiknya, biasanya jarang digunakan lebih dari 3 *nearest neighbors*.

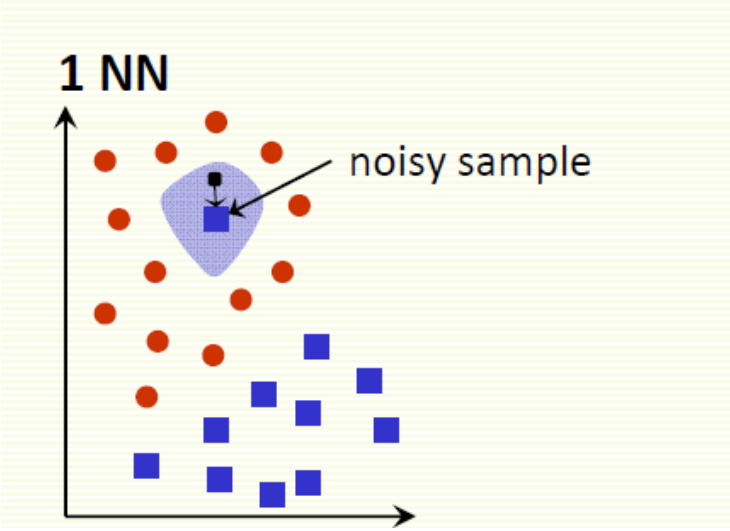
Algoritma KNN

- Memerlukan 3 hal:
  - Record (training) dataset.
  - Distance metric untuk mengukur jarak antar points.
  - Nilai *k*, jumlah tetangga terdekat.

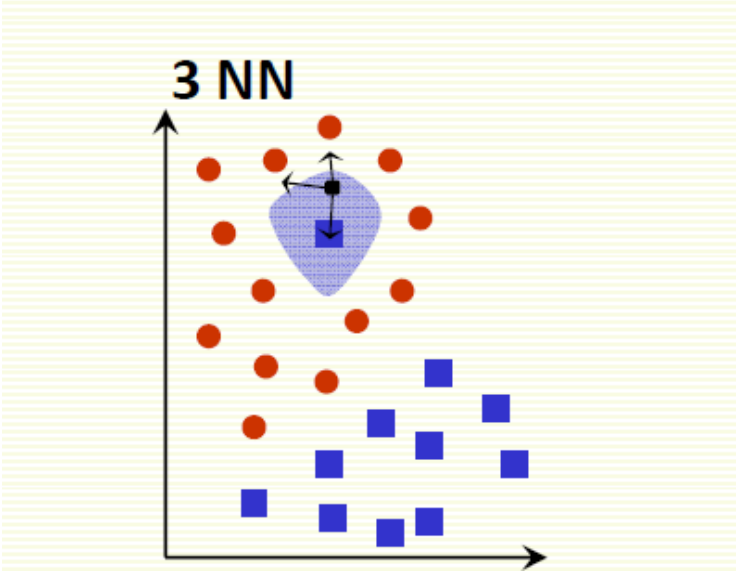
- Langkah untuk mengklasifikasikan unknown point:
  - Hitung jarak antara point tersebut ke training records.
  - Identifikasi *k* tetangga terdekat.
  - Gunakan label dari tetangga terdekat untuk menentukan label dari unknown point tersebut (misal dengan majority vote)

Bagaimana Memilih *k*?

- Dalam praktik, kadang digunakan *k*=1 untuk efisiensi, namun rentan terhadap 'noise'



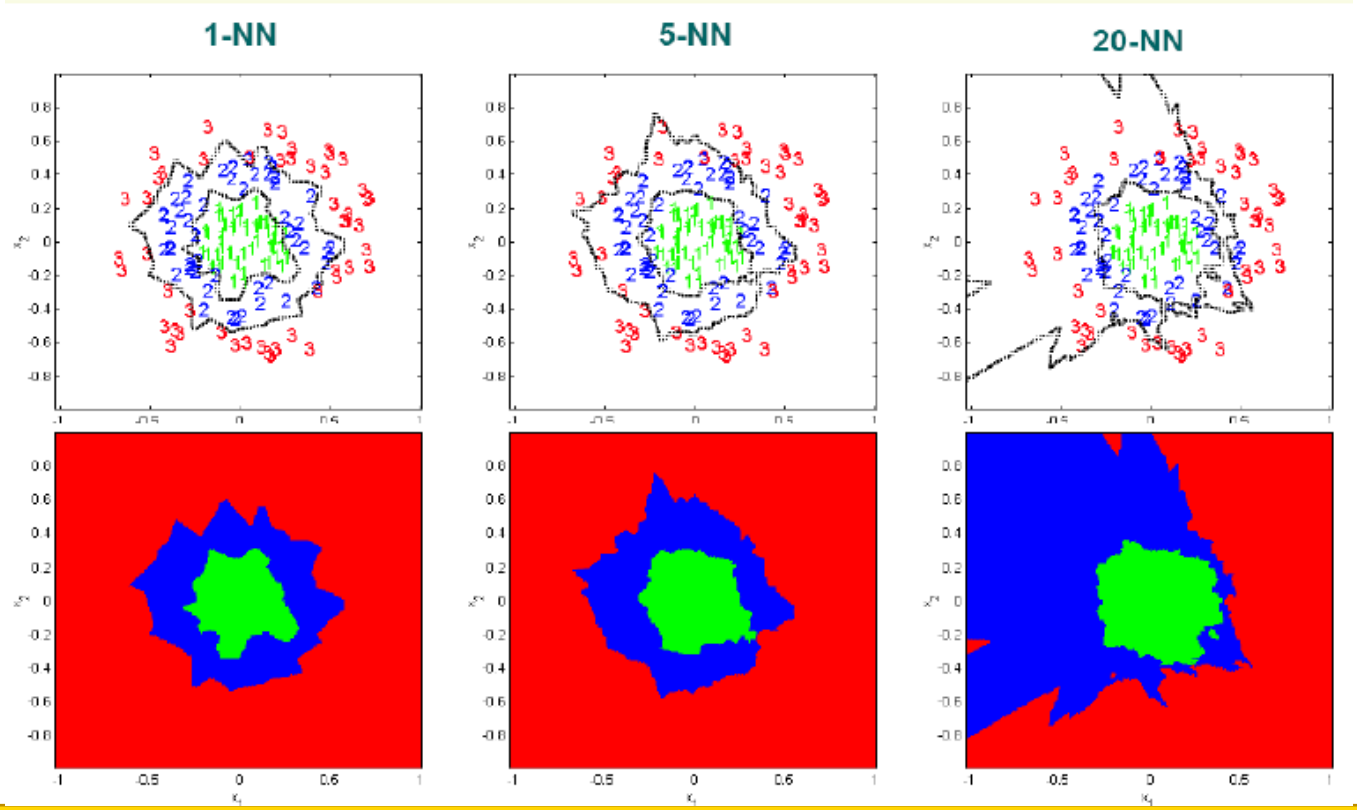
Semua sample pada area yang diarsir dikategorikan sebagai kelas "biru"



Semua sample pada area yang diarsir dikategorikan sebagai kelas "merah"

Bagaimana Memilih *k*?

- *k* yang lebih besar mungkin memperbaiki performance, namun *k* yang terlalu besar menghilangkan aspek lokalitas.
- Cross Validation dapat digunakan untuk memilih *k*



Bagaimana cara menghitung jarak?

- Metode k-Nearest neighbors menuntut kita untuk mencari sampel berlabel "terdekat" dengan *query*.
- Bagaimana kita dapat mendefinisikan "kedekatan" antara keduanya secara matematis?

Distance Metric

Mengukur Jarak

- Pada umumnya, kedekatan atau jarak antara *query* **x** dengan sampel berlabel $\mathbf{x}_i$ dapat diukur menggunakan **Minkowski distance**:

$$d(x, y) = \| x - y \|_m = \left[\sum_{i=1}^{N} (x_i - y_i)^m \right]^{1/m}$$

  - Bagaimana jika $m = 1$?
  - Bagaimana jika $m = 2$?

Mengukur Jarak

- Pada umumnya, kedekatan atau jarak antara *query* **x** dengan sampel berlabel $\mathbf{x}_i$ dapat diukur menggunakan **Minkowski distance**:

$$d(x, y) = \| x - y \|_m = \left[\sum_{i=1}^{N} (x_i - y_i)^m \right]^{1/m}$$

  - Bagaimana jika $m = 1$?
  - Bagaimana jika $m = 2$?

Distance Metric

- City-block distance (Manhattan distance)
  - Penjumlahan nilai absolute dari selisih titik *p* dan *q*

$$d(p,q) = \sum_i |p_i - q_i|$$

- Euclidian Distance

$$D(a,b) = \sqrt{\sum_k (a_k - b_k)^2} = \sqrt{a \cdot b}$$

Distance Metric

- Cosine Similarity
  - Mengukur sudut yang dibentuk dari 2 sampel (dengan origin).

$$d(p,q) = \frac{p \cdot q}{\|p\| \times \|q\|} = \frac{\sum_{i=1}^{n} p_i \times q_i}{\sqrt{\sum_{i=1}^{n} p_i^2} \times \sqrt{\sum_{i=1}^{n} q_i^2}}$$

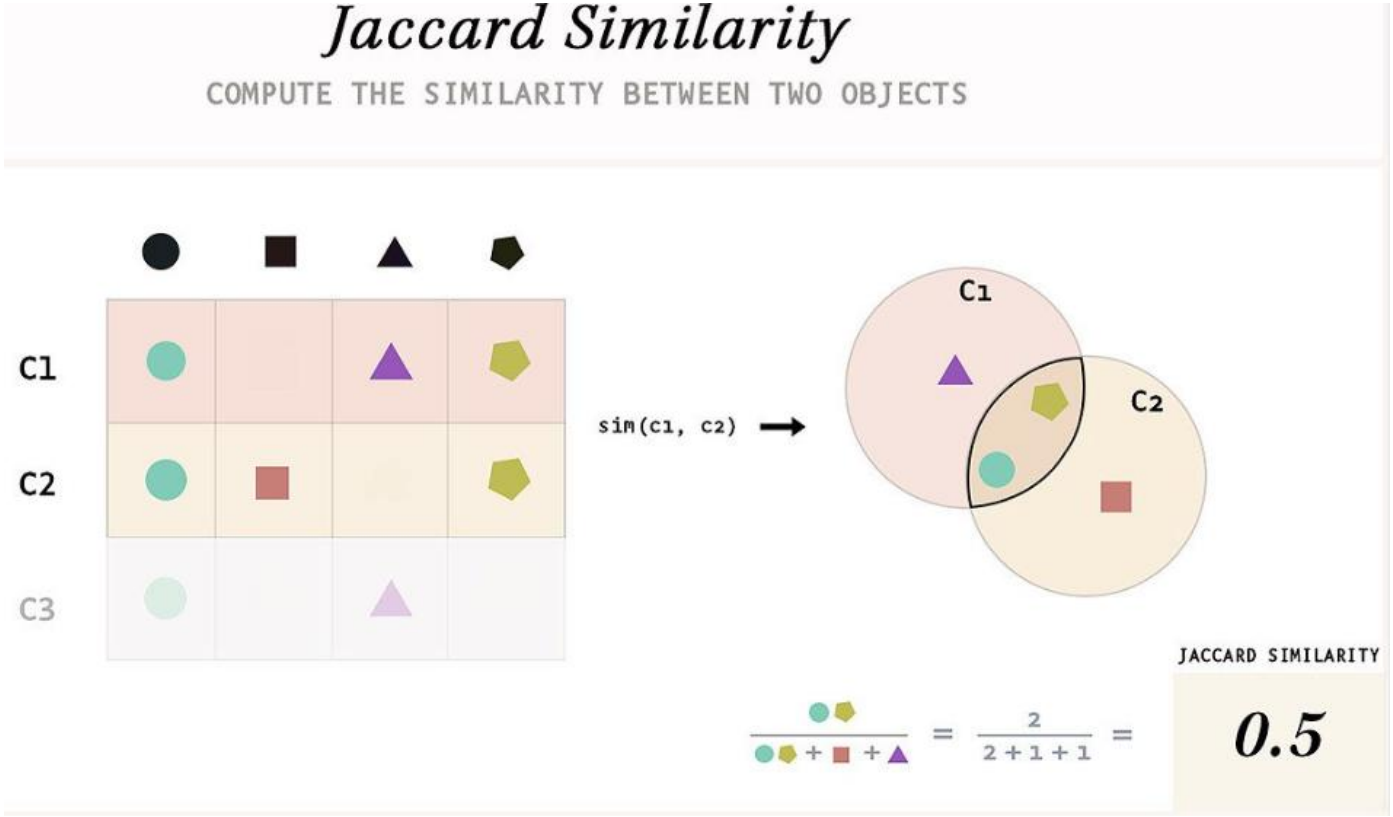
Distance Metric

- Jaccard distance
  - Menentukan persentase exact match antar sampel

$$\text{Jaccard coefficient}(p, q) = \frac{|p \cap q|}{|p \cup q|} = \frac{|p \cap q|}{|p| + |q| - |p \cap q|}$$

$$\text{Jaccard distance}(p, q) = 1 - \text{Jaccard coefficient}(p, q)$$

Jaccard Distance



https://www.learndatasci.com/glossary/jaccard-similarity/

Jaccard Similarity untuk 2 vektor biner

- Contoh: data pembelian oleh konsumen, 1 menandakan konsumen membeli dan 0 menandakan konsumen tidak membeli
  - a = jumlah atribut bernilai 1 untuk customer i dan j
  - b = jumlah atribut bernilai 0 untuk customer i dan bernilai 1 untuk customer j
  - c = jumlah atribut bernilai 1 untuk customer i dan bernilai 0 untuk customer j
  - d = jumlah atribut bernilai 0 untuk customer i dan j

- Perhitungan jaccard similarity: $J(i,j) = sim(i,j) = \frac{a}{a+b+c}$

	item1	item2	item2	item4	item5	item6	item7	item8	item9
C1	0	1	0	0	0	1	0	0	1
C2	0	0	1	0	0	0	0	0	1
C3	1	1	0	0	0	1	0	0	0

$$J(C1,C2) = \frac{a}{a+b+c} = \frac{1}{1+1+2} = 0.25$$

$$J(C1,C3) = \frac{a}{a+b+c} = \frac{2}{2+1+1} = 0.5$$

https://www.learndatasci.com/glossary/jaccard-similarity/

Jaccard Similarity untuk 2 himpunan

- Contoh diberikan 2 himpunan:
  - A = { 0, 2, 4, 6, 8, 10}
  - B = { 0, 2, 3, 5, 7, 8, 9, 10, 11}

- Jaccard similarity:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|\{0,2,8,10\}|}{|\{0,2,3,4,5,6,7,8,9,10\ ,11\}|} = \frac{4}{11}$$

- Jaccard distance:

$$d(A,B) = 1\ - J(A,B) = \frac{7}{11}$$

Contoh Klasifikasi dengan jarak Euclidian

- Contoh perhitungan jarak (Euclidian distance) untuk Klasifikasi
  - Fitur 1: nilai 1 atau 2
  - Fitur 2: nilai antara 100 sampai dengan 200.
  - Target: Kelas A atau Kelas B

  - $\text{dataset} = \begin{bmatrix} \mathbf{1} & \mathbf{150} & \mathbf{A} \\ \mathbf{2} & \mathbf{110} & \mathbf{B} \end{bmatrix}$
  - $\text{classify} = [\mathbf{1} \quad \mathbf{100} \quad ?]$ kelas yang benar: $\mathbf{A}$
  - $D\left(\begin{bmatrix} 1 \\ 100 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 150 \end{bmatrix}\right) = \sqrt{(1-1)^2 + (100-150)^2} = 50$
  - $D\left(\begin{bmatrix} 1 \\ 100 \end{bmatrix} \cdot \begin{bmatrix} 2 \\ 110 \end{bmatrix}\right) = \sqrt{(1-2)^2 + (100-110)^2} = 10.5$
  - $[\mathbf{1} \quad \mathbf{100} \quad \mathbf{B}]$ misclassified

Mengukur Jarak

- Contoh perhitungan jarak (Euclidian distance) untuk Klasifikasi
  - Fitur 1: nilai 1 atau 2
  - Fitur 2: nilai antara 100 sampai dengan 200.
  - Target: Kelas A atau Kelas B,

  - $\text{dataset} = \begin{bmatrix} \mathbf{1} & \mathbf{150} & \mathbf{A} \\ \mathbf{2} & \mathbf{110} & \mathbf{B} \end{bmatrix}$
  - $\text{classify} = [\mathbf{1} \quad \mathbf{100} \quad ?]$ kelas yang benar: $\mathbf{A}$
  - $D\left(\begin{bmatrix} 1 \\ 100 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 150 \end{bmatrix}\right) = \sqrt{(1-1)^2 + (100-150)^2} = 50$
  - $D\left(\begin{bmatrix} 1 \\ 100 \end{bmatrix} \cdot \begin{bmatrix} 2 \\ 110 \end{bmatrix}\right) = \sqrt{(1-2)^2 + (100-110)^2} = 10.5$
  - $[\mathbf{1} \quad \mathbf{100} \quad \mathbf{B}]$ misclassified

Perlu normalisasi!

Metrik Jarak untuk Data Non Numerik

- Nilai dari features tidak selalu berupa angka
- Sebagai contoh:
  - Boolean values: Benar atau Salah, ada atau tidaknya suatu atribut
  - Kategori: Warna, Pendidikan, Jenis Kelamin
- Bagaimana caranya untuk melakukan komputasi jarak pada nilai tersebut?
  - Boolean values dapat dikonversi ke 0 atau 1
  - Non-binary characterizations
    - Menggunakan natural progression (tahapan perkembangan)
      - Pendidikan: SMP, SMA, Sarjana, Master, Doktoral → 1, 2, 3, 4, 5
    - Memberikan nilai sembarang tetapi perlu memperhatikan perhitungan jaraknya,
      - Warna: merah, kuning, biru → 1, 2, 3

KNN untuk Regresi

Regresi dengan KNN

- Proses regresi menggunakan KNN pada dasarnya mirip dengan penggunaan KNN untuk klasifikasi.
- Perlu penyesuaian pada penentuan *output* dari KNN untuk regresi, berupa variabel kontinu.
- Salah satu bentuk modifikasi yang dapat dilakukan adalah hasil prediksi pada proses regresi tidak dicari sebagai label dari *neighborhood* yang memilikijumlahvotes tertinggi, namun sebagai rerata dari nilai *y* (output) pada *neighborhoods.*

Langkah-Langkah Regresi

- Pilih sebuah bilangan bulat positif untuk nilai k.
- Untuk setiap *query*, tentukan k buah sampel berlabel terdekat dan mereka disebut sebagai *neighborhood*dari *query*.
- Hasil prediksi *output* $\hat{y}$ dari *query* adalah rerata dari *output neighborhood* y_i.

$$\hat{y} = \frac{1}{k}\sum_{i=1}^{k} y_i$$

Contoh

- Diberikan problem regresi, untuk memprediksi berat orang no 11

ID	Height	Age	Weight
1	5	45	77
2	5.11	26	47
3	5.6	30	55
4	5.9	32	59
5	4.8	40	72
6	5.8	36	60
7	5.3	19	40
6	5.8	28	60
9	5.5	23	45
10	5.6	32	58
11	5.5	39	?

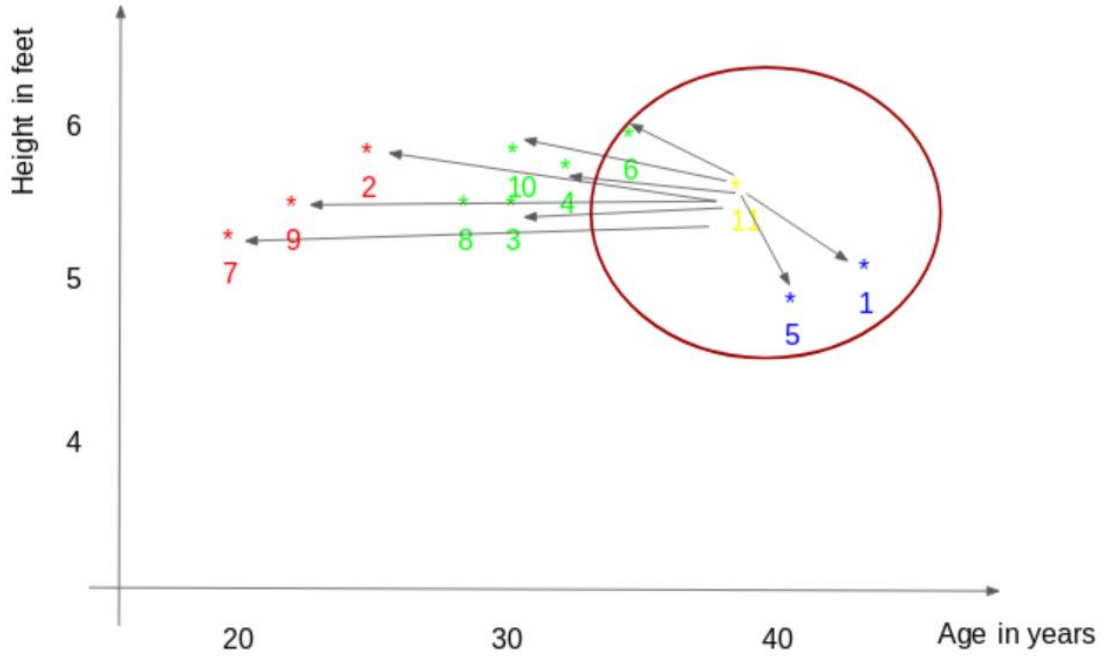
Contoh

- Langkah 1: Hitung jarak ID 11 dengan seluruh data points.
- Langkah 2: Sebanyak *k* jarak terdekat dipilih.

ID	Height	Age	Weight	Jarak Euclidian ke ID 11
1	5	45	77	6.02
2	5.11	26	47	13.01
3	5.6	30	55	9.00
4	5.9	32	59	7.01
5	4.8	40	72	1.22
6	5.8	36	60	3.01
7	5.3	19	40	20.00
6	5.8	28	60	11.00
9	5.5	23	45	16.00
10	5.6	32	58	7.00
11	5.5	39	?	

Jika k= 3, maka
yang terpilih adalah
ID 1, 5, 6

Contoh



- Langkah 3: Rerata berat badan dari 3 points tersebut merupakan hasil prediksi untuk ID 11. Jadi hasil prediksinya adalah (77 + 72 + 60)/3 = 69.66

Analisis KNN

Pertimbangan Praktis

- Memerlukan banyak labelled dataset agar bekerja dengan baik.
- Perlu memperhatikan pemilihan pengukuran jarak (distance)
  - Transformasi dari fitur non numerik
  - Normalisasi (scaling) fitur sehingga variance menjadi konsisten
  - Opsi lain: transformasi fitur agar matrix covariance menjadi identity (**whitening**).
- Perlunya menemukan nearest neighbors untuk query point.
  - Nearest neighbors dalam dimensi yang tinggi lebih sulit untuk ditemukan.
  - Biasanya diatasi dengan menggunakan **approximate nearest neighbors**, dengan probabilitas tinggi hampir mendekati query point
  - Memperoleh approximate nearest neighbors lebih mudah dibandingkan dengan nearest neighbors.
  - Metode ini mencakup proses tuning constant.

Pertimbangan Praktis

- Nearest neighbor classifier cukup mudah digunakan dengan cross-validation untuk mengukur estimasi error rate.
  - Pada dasarnya, split labelled data menjadi training set dan validation set.
  - Kemudian kelompokkan setiap validation set dengan kelas berdasarkan kedekatannya dengan training set.
  - Hitung fraksi dari setiap percobaan yang menghasilkan error, yaitu predicted label tidak sesuai dengan true label.
  - Lalu ulangi proses tersebut untuk setiap split, dan hitung average error yang diperoleh dari masing-masing split.

Kompleksitas Komputasi

- Algoritma KNN menyimpan seluruh sampel
- Jika kita memiliki **n** sampel dengan dimensi **d**
- **O(d)** untuk menghitung jarak (distance) untuk satu sampel.
- **O(nd)** untuk mencari satu tetangga terdekat (nearest neighbor).
- **O(knd)** untuk menemukan **k** sampel terdekat.
- Maka diperoleh total kompleksitas adalah **O(knd)**
- Algoritma ini memerlukan komputasi yang sangat mahal untuk sampel dalam jumlah yang besar.
- Namun, kita memerlukan sampel dalam jumlah yang besar agar kNN dapat bekerja dengan baik.

Pro's & Con's

- **Kelebihan**
  - Dapat diterapkan terhadap data dengan berbagai macam distribusi
  - Contoh, data tidak harus dapat dipisahkan dengan garis linear
  - Sangat mudah dan intuitif
  - Merupakan klasifikasi yang baik jika jumlah sampel yang tersedia cukup besar.

- **Kekurangan**
  - Menentukan ***k*** dapat menjadi sulit
  - Tahap testing memerlukan komputasi yang mahal
    - Tidak memiliki tahapan training, seluruh proses dilakukan pada tahap testing
    - Hal ini bertolak belakang dengan yang diinginkan. Biasanya tahap training memakan waktu yang lama, sedangkan kita menginginkan tahap testing yang cepat.
  - Memerlukan sampel dalam jumlah besar untuk memperoleh akurasi yang baik.

Referensi

- Forsyth, D.,"Probability and Statistics for Computer Science", Springer, 2018
- Charu C. Aggarwal, "Data Mining: The Textbook", Springer, 2015
- Tim Dosen Sains Data, "Data Science:Learning to Classify: Classifying with Nearest Neighbors", Fasilkom UI, 2020
- Tim Dosen KASDD, "K-Nearest Neighbors", Fasilkom UI, 2021
- https://www.analyticsvidhya.com/blog/2018/08/k-nearest-neighbor-introduction-regression-python/
- https://www.learndatasci.com/glossary/jaccard-similarity/

Wish You Success