

# Machine Learning Evaluation & Supervised Learning

Classification



# Rezki Trianto

Data guy who spent the last 7 years to work around data analytics and machine learning. Working in a unicorn company with various domain expertise to work with, from an e-commerce, OTA, and ride-hailing company.



Rezki Trianto

[linkedin.com/in/rezkitrianto](https://www.linkedin.com/in/rezkitrianto)

## Education Background



**2010-2014**  
Bachelor Degree  
Computer Science



**2015-2017**  
Master Degree  
Computer Science

### Rezki Trianto

5+ years experience in Data  
Science & Analytics

# Hands On Required

**Hands - On : 3. Classification**

Klik disini untuk  
mengakses folder Hands  
On

# Topik Supervised Learning: Classification

- ☐ Review Klasifikasi
- ☐ Logistic Regression
- ☐ Hands On - Logistic Regression - Part 1
- ☐ Hands On - Logistic Regression - Part 2
- ☐ Hands On - Analyze Learning Curve
- ☐ k-Nearest Neighbor (kNN)
- ☐ Hands On - kNN
- ☐ Decision Tree - Intuition
- ☐ Decision Tree - Hyperparameters & Feature Importances
- ☐ Hands On - Decision Tree
- ☐ Hands On - Feature Importance & Summary

# Topik Supervised Learning: Classification



Review Klasifikasi



Logistic Regression



Hands On - Logistic Regression - Part 1



Hands On - Logistic Regression - Part 2



Hands On - Analyze Learning Curve



k-Nearest Neighbor (kNN)



Hands On - kNN



Decision Tree - Intuition



Decision Tree - Hyperparameters & Feature Importances



Hands On - Decision Tree



Hands On - Feature Importance & Summary

# Review Previous Topics

## Related to Classification

### 1. Metrics Evaluation:

- Accuracy -> ketika kepentingan label seimbang, dan jumlah target balance
- Precision -> target balance, dan fokus mengurangi False Positive
- Recall -> target balance, dan fokus mengurangi False Negative
- ROC-AUC -> target imbalance (namun tidak ekstrim), fokus pada label positive & negative
- F1 -> target imbalance (bisa ekstrim), fokus pada label positif

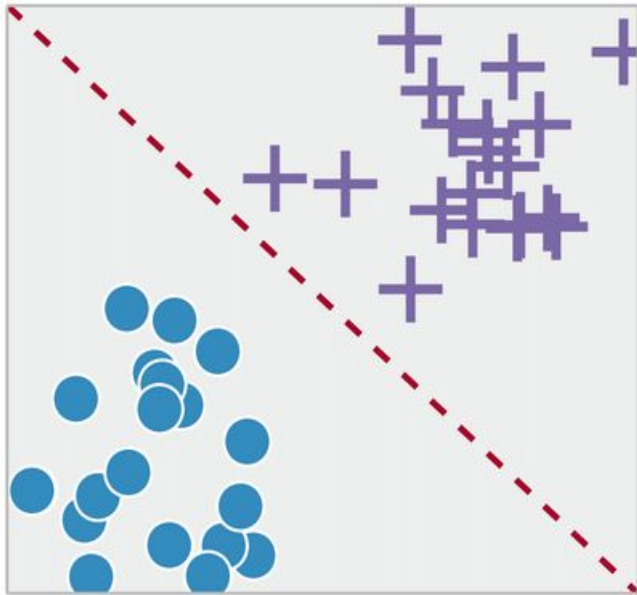
### 2. Regularization

- Salah satu teknik mencegah overfitting dengan menambahkan bias

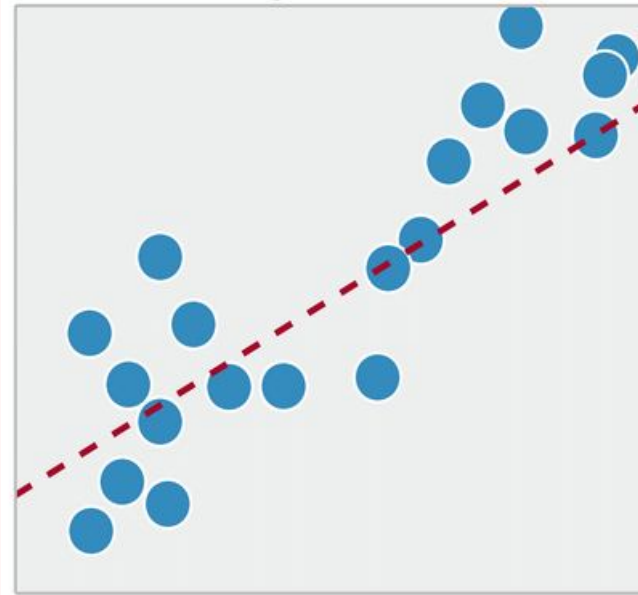


# Jenis Supervised Learning

Classification



Regression



# Target Label bertipe categorical

Target		Feature						
is_diabetes	glucose_concentration	blood_pressure	triceps_thickness	two_hour_insulin	bmi	pedigree_function	age	
yes	148	72	35	0	33.6	0.627	50	
no	85	66	29	0	26.6	0.351	31	
yes	183	64	0	0	23.3	0.672	32	
no	89	66	23	94	28.1	0.167	21	
yes	137	40	35	168	43.1	2.288	33	
no	116	74	0	0	25.6	0.201	30	
yes	78	50	32	88	31	0.248	26	
no	115	0	0	0	35.3	0.134	29	
yes	197	70	45	543	30.5	0.158	53	
yes	125	96	0	0	0	0.232	54	
no	110	92	0	0	37.6	0.191	30	
yes	168	74	0	0	38	0.537	34	
no	139	80	0	0	27.1	1.441	57	
yes	189	60	23	846	30.1	0.398	59	
yes	166	72	19	175	25.8	0.587	51	
yes	100	0	0	0	30	0.484	32	
yes	118	84	47	230	45.8	0.551	31	
yes	107	74	0	0	29.6	0.254	31	
no	103	30	38	83	43.3	0.183	33	



# Topik Supervised Learning: Classification



Review Klasifikasi



Logistic Regression



Hands On - Logistic Regression - Part 1



Hands On - Logistic Regression - Part 2



Hands On - Analyze Learning Curve



k-Nearest Neighbor (kNN)



Hands On - kNN



Decision Tree - Intuition



Decision Tree - Hyperparameters & Feature Importances



Hands On - Decision Tree



Hands On - Feature Importance & Summary

# Topik Supervised Learning: Classification

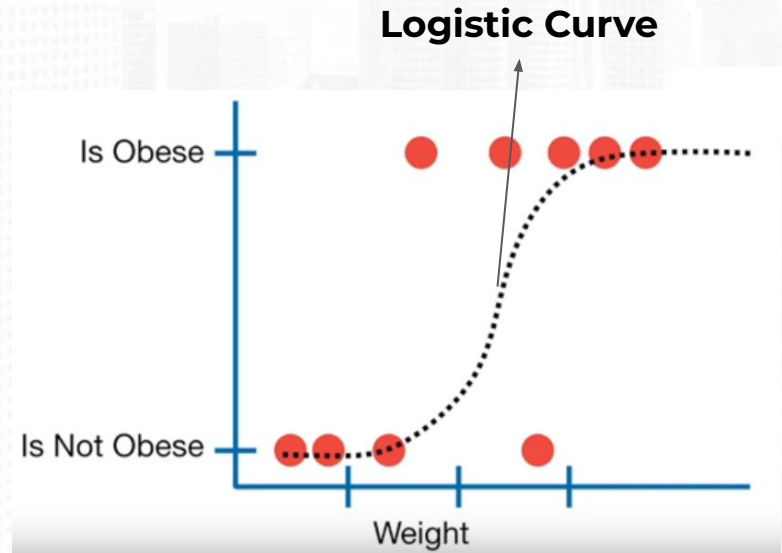
- |                                     |   |                          |   |
|-------------------------------------|---|--------------------------|---|
| <input checked="" type="checkbox"/> | Review Klasifikasi                      | <input type="checkbox"/> | Hands On - kNN  |
| <input type="checkbox"/>            | Logistic Regression                     | <input type="checkbox"/> | Decision Tree - Intuition                             |
| <input type="checkbox"/>            | Hands On - Logistic Regression - Part 1 | <input type="checkbox"/> | Decision Tree - Hyperparameters & Feature Importances |
| <input type="checkbox"/>            | Hands On - Logistic Regression - Part 2 | <input type="checkbox"/> | Hands On - Decision Tree                              |
| <input type="checkbox"/>            | Hands On - Analyze Learning Curve       | <input type="checkbox"/> | Hands On - Feature Importance & Summary               |
| <input type="checkbox"/>            | k-Nearest Neighbor (kNN)                |                          |   |

# Logistic Regression

aka. Logit / MaxEnt

# Logistic Regression

- Jika Linear regression melakukan prediksi pada data numerik, untuk data dengan tipe categorical, kita dapat menggunakan **logistic regression**.
- Tujuan dari logistic regression melakukan fitting logistic function yang berbentuk seperti huruf S.

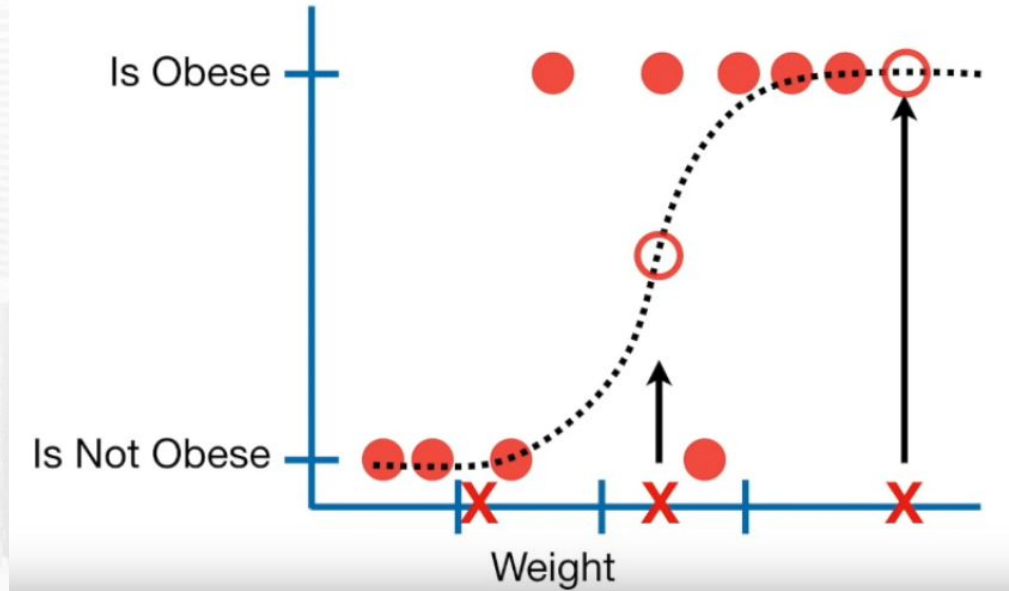


# Logistic Regression

Logistic regression mencari weight/parameter yang mempunyai error terkecil (mirip seperti linear regression).

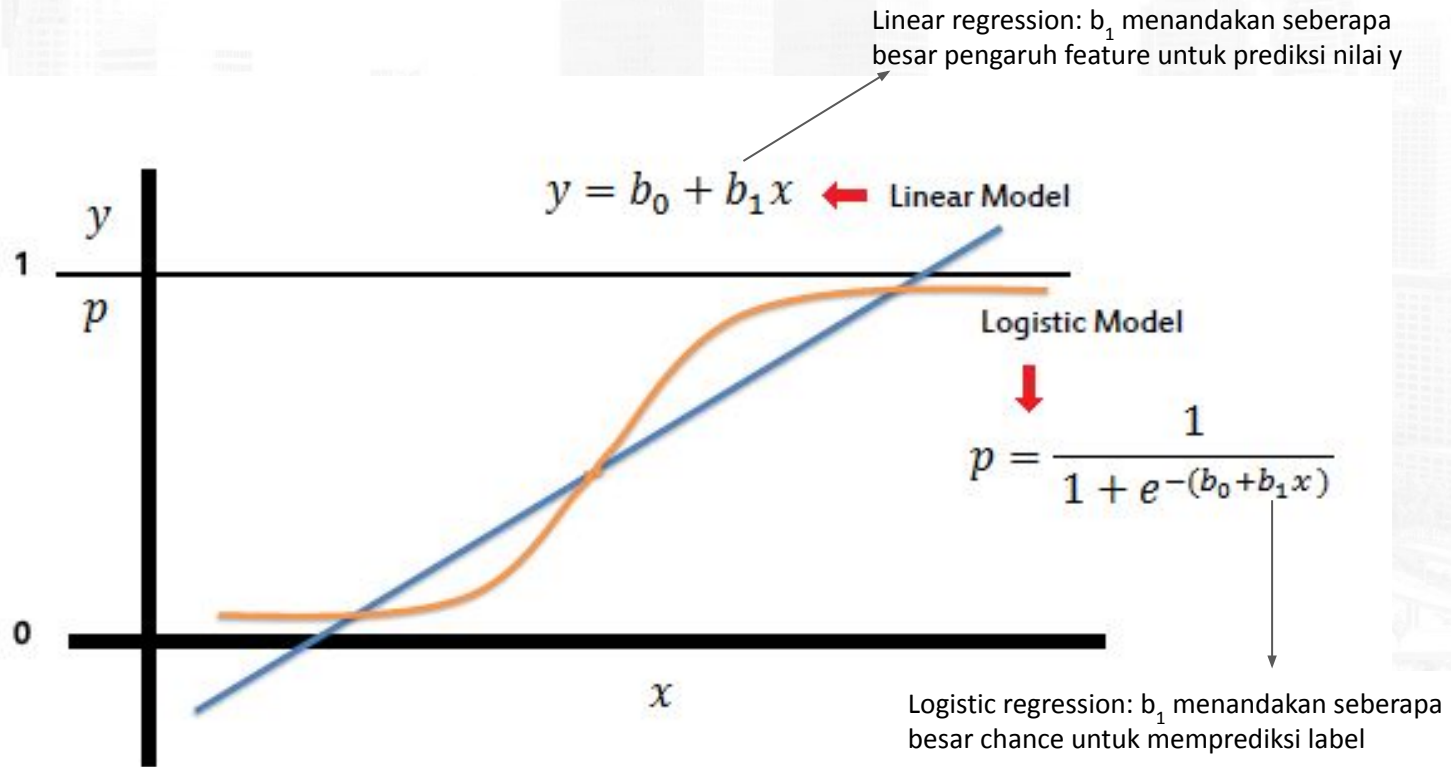


# Logistic Regression



Pada logistic regression, jika probabilitas  $> 0.5$ , maka kita dapat melakukan klasifikasi bahwa label = Obese

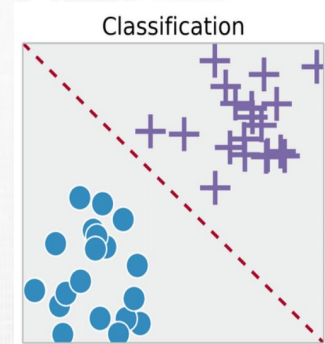
# Linear vs. Logistic Model



# Kapan menggunakan Logistic Regression?

## Asumsi Logistic Regression

1. Jika data dapat dipisahkan secara linear\*
2. Jika butuh model yang simple dan komputasi cukup cepat
3. Tidak terdapat extreme outlier yang sangat berpengaruh ke hasil prediksi\*
4. Tidak ada multikolinearitas atau repeated features\*
5. Digunakan untuk binary classification. untuk kasus multiclass bisa menggunakan one vs. rest\*



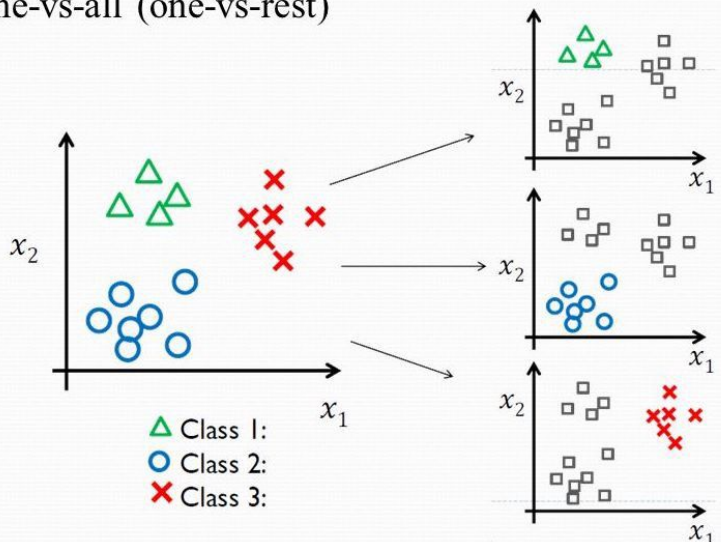
Linearly separable

\*) Asumsi logistic regression. [Referensi](#)

# One vs. Rest

1. Untuk prediksi kasus multiclass pada logistic regression menggunakan konsep one vs. rest

One-vs-all (one-vs-rest)



Pada gambar disamping, one vs. rest membagi menjadi 3 tahap,

1. menganggap class 1 sebagai label 1, dan sisanya adalah label lainnya
2. menganggap class 2 sebagai label 2, dan sisanya adalah label lainnya
3. menganggap class 3 sebagai label 3, dan sisanya adalah label lainnya

Masing-masing akan melakukan prediksi terhadap data test. one vs. rest akan mencari probabilitas yang paling besar diantara ketiga class

# Topik Supervised Learning: Classification

- |                                     |   |                          |   |
|-------------------------------------|---|--------------------------|---|
| <input checked="" type="checkbox"/> | Review Klasifikasi                      | <input type="checkbox"/> | Hands On - kNN  |
| <input checked="" type="checkbox"/> | Logistic Regression                     | <input type="checkbox"/> | Decision Tree - Intuition                             |
| <input type="checkbox"/>            | Hands On - Logistic Regression - Part 1 | <input type="checkbox"/> | Decision Tree - Hyperparameters & Feature Importances |
| <input type="checkbox"/>            | Hands On - Logistic Regression - Part 2 | <input type="checkbox"/> | Hands On - Decision Tree                              |
| <input type="checkbox"/>            | Hands On - Analyze Learning Curve       | <input type="checkbox"/> | Hands On - Feature Importance & Summary               |
| <input type="checkbox"/>            | k-Nearest Neighbor (kNN)                |                          |   |



# Topik Supervised Learning: Classification



Review Klasifikasi



Logistic Regression



Hands On - Logistic Regression - Part 1



Hands On - Logistic Regression - Part 2



Hands On - Analyze Learning Curve



k-Nearest Neighbor (kNN)



Hands On - kNN



Decision Tree - Intuition



Decision Tree - Hyperparameters & Feature Importances



Hands On - Decision Tree



Hands On - Feature Importance & Summary

# Implementasi di python



# Implementasi di python : Model Validation

```
from sklearn.model_selection import train_test_split  
Xtrain, Xtest, ytrain, ytest = train_test_split(x,y,test_size=1/3, random_state=42)
```

Split train & test set

# Implementasi di python : Fit Model

```
from sklearn.model_selection import train_test_split
Xtrain, Xtest, ytrain, ytest = train_test_split(x,y,test_size=1/3, random_state=42)
```

Split train & test set

```
from sklearn.linear_model import LogisticRegression# import logistic regression dari sklearn
logreg = LogisticRegression() # inisiasi object dengan nama logreg
logreg.fit(X_train, y_train) # fit model regression dari data train
```

Latih model pada  
train set

Dokumentasi sklearn Logistic Regression:

[https://scikit-learn.org/stable/modules/generated/sklearn.linear\\_model.LogisticRegression.html](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html)

# Implementasi di python : Prediksi & Evaluasi

```
from sklearn.model_selection import train_test_split
Xtrain, Xtest, ytrain, ytest = train_test_split(x,y,test_size=1/3, random_state=42)
```

Split train & test set

```
from sklearn.linear_model import LogisticRegression# import logistic regression dari sklearn
logreg = LogisticRegression() # inisiasi object dengan nama logreg
logreg.fit(Xtrain, ytrain) # fit model regression dari data train
```

Latih model pada  
train set

```
y_predicted = logreg.predict(Xtest) # prediksi data test

from sklearn.metrics import accuracy_score
print(accuracy_score(ytest, y_predicted)) # contoh evaluasi dengan akurasi
```

Prediksi dan  
Evaluasi Test Set

Dokumentasi sklearn Logistic Regression:

[https://scikit-learn.org/stable/modules/generated/sklearn.linear\\_model.LogisticRegression.html](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html)



# Hands On!

# Topik Supervised Learning: Classification

- |                                     |   |                          |   |
|-------------------------------------|---|--------------------------|---|
| <input checked="" type="checkbox"/> | Review Klasifikasi                      | <input type="checkbox"/> | Hands On - kNN  |
| <input checked="" type="checkbox"/> | Logistic Regression                     | <input type="checkbox"/> | Decision Tree - Intuition                             |
| <input checked="" type="checkbox"/> | Hands On - Logistic Regression - Part 1 | <input type="checkbox"/> | Decision Tree - Hyperparameters & Feature Importances |
| <input type="checkbox"/>            | Hands On - Logistic Regression - Part 2 | <input type="checkbox"/> | Hands On - Decision Tree                              |
| <input type="checkbox"/>            | Hands On - Analyze Learning Curve       | <input type="checkbox"/> | Hands On - Feature Importance & Summary               |
| <input type="checkbox"/>            | k-Nearest Neighbor (kNN)                |                          |   |

# Topik Supervised Learning: Classification

- ☒ Review Klasifikasi
- ☒ Logistic Regression
- ☒ Hands On - Logistic Regression - Part 1
- ☒ Hands On - Logistic Regression - Part 2
- ☐ Hands On - Analyze Learning Curve
- ☐ k-Nearest Neighbor (kNN)
- ☐ Hands On - kNN
- ☐ Decision Tree - Intuition
- ☐ Decision Tree - Hyperparameters & Feature Importances
- ☐ Hands On - Decision Tree
- ☐ Hands On - Feature Importance & Summary

# Hands On!

# Topik Supervised Learning: Classification

- |                                     |   |                          |   |
|-------------------------------------|---|--------------------------|---|
| <input checked="" type="checkbox"/> | Review Klasifikasi                      | <input type="checkbox"/> | Hands On - kNN  |
| <input checked="" type="checkbox"/> | Logistic Regression                     | <input type="checkbox"/> | Decision Tree - Intuition                             |
| <input checked="" type="checkbox"/> | Hands On - Logistic Regression - Part 1 | <input type="checkbox"/> | Decision Tree - Hyperparameters & Feature Importances |
| <input checked="" type="checkbox"/> | Hands On - Logistic Regression - Part 2 | <input type="checkbox"/> | Hands On - Decision Tree                              |
| <input type="checkbox"/>            | Hands On - Analyze Learning Curve       | <input type="checkbox"/> | Hands On - Feature Importance & Summary               |
| <input type="checkbox"/>            | k-Nearest Neighbor (kNN)                |                          |   |

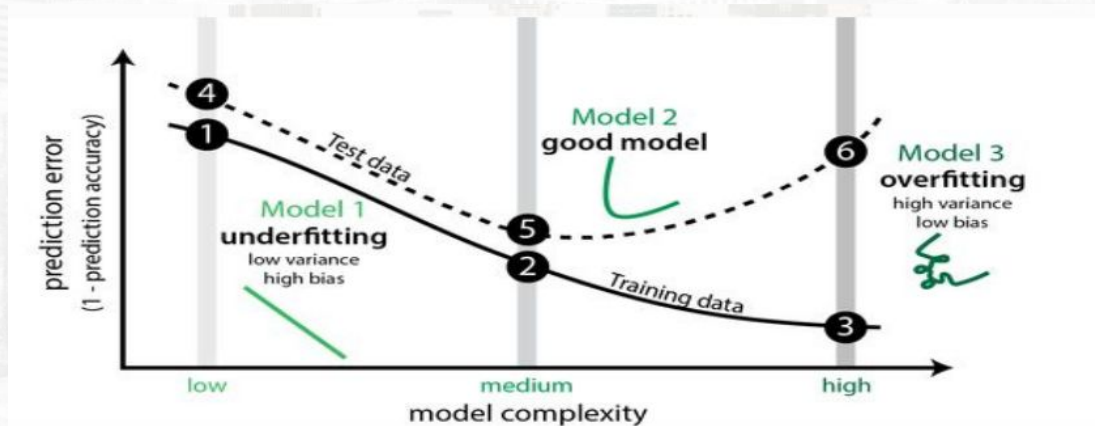


# Topik Supervised Learning: Classification

- ☒ Review Klasifikasi
- ☒ Logistic Regression
- ☒ Hands On - Logistic Regression - Part 1
- ☒ Hands On - Logistic Regression - Part 2
- ☒ Hands On - Analyze Learning Curve
- ☐ k-Nearest Neighbor (kNN)
- ☐ Hands On - kNN
- ☐ Decision Tree - Intuition
- ☐ Decision Tree - Hyperparameters & Feature Importances
- ☐ Hands On - Decision Tree
- ☐ Hands On - Feature Importance & Summary

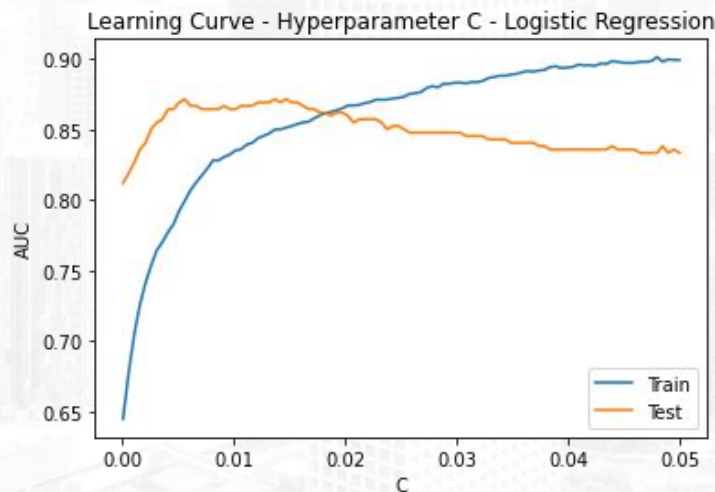
# Model Validation - Learning Curve

Evaluasi machine learning model dengan melihat learning curve, dan pahami bagaimana model machine learning berpotensi untuk mengalami underfit / overfit



# Model Validation - Learning Curve

Evaluasi machine learning model dengan melihat learning curve, dan pahami bagaimana model machine learning berpotensi untuk mengalami underfit / overfit



Interpretasi:

$C \geq 0$  and  $C < 0.02$ : Underfit

$C \geq 0.02$  and  $C \leq 0.03$ : Best Fit













$C \geq 0.04$ : Overfit

# Hands On!

# Topik Supervised Learning: Classification

- |                                     |   |                          |   |
|-------------------------------------|---|--------------------------|---|
| <input checked="" type="checkbox"/> | Review Klasifikasi                      | <input type="checkbox"/> | Hands On - kNN  |
| <input checked="" type="checkbox"/> | Logistic Regression                     | <input type="checkbox"/> | Decision Tree - Intuition                             |
| <input checked="" type="checkbox"/> | Hands On - Logistic Regression - Part 1 | <input type="checkbox"/> | Decision Tree - Hyperparameters & Feature Importances |
| <input checked="" type="checkbox"/> | Hands On - Logistic Regression - Part 2 | <input type="checkbox"/> | Hands On - Decision Tree                              |
| <input checked="" type="checkbox"/> | Hands On - Analyze Learning Curve       | <input type="checkbox"/> | Hands On - Feature Importance & Summary               |
| <input type="checkbox"/>            | k-Nearest Neighbor (kNN)                |                          |   |

# Topik Supervised Learning: Classification

- |   |  |   |   |
|---|--|---|---|
|  | Review Klasifikasi   |  | Hands On - kNN  |
|  | Logistic Regression  |  | Decision Tree - Intuition                             |
|  | Hands On - Logistic Regression - Part 1  |  | Decision Tree - Hyperparameters & Feature Importances |
|  | Hands On - Logistic Regression - Part 2  |  | Hands On - Decision Tree                              |
|  | Hands On - Analyze Learning Curve  |  | Hands On - Feature Importance & Summary               |
|   |  k-Nearest Neighbor (kNN) |   |   |



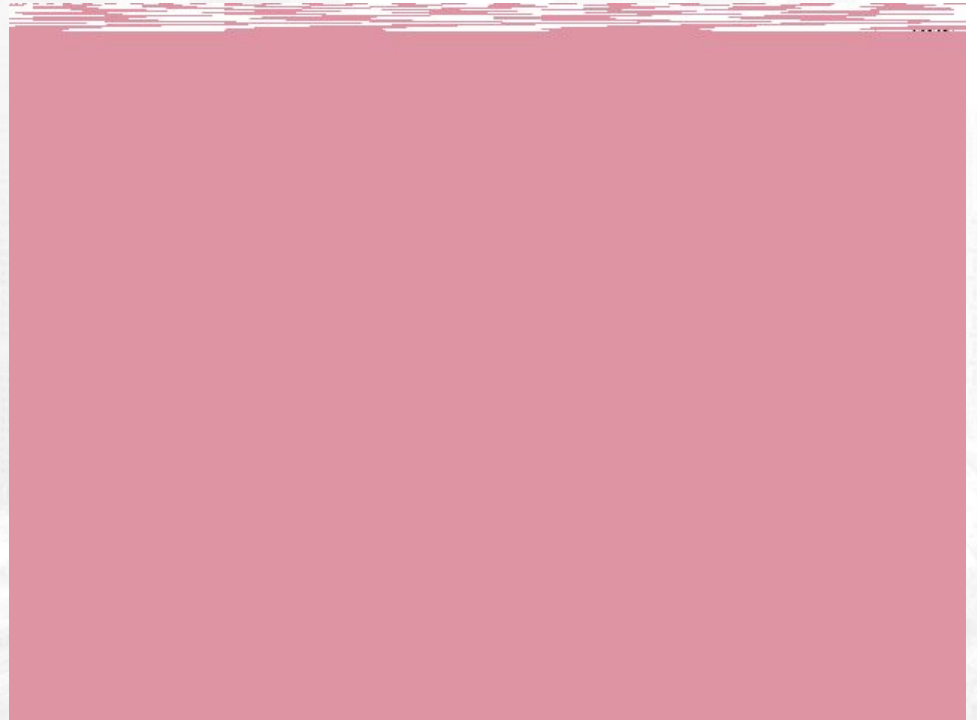
# k-Nearest Neighbor (kNN)

# K-Nearest Neighbor

Salah satu algoritma paling sederhana untuk melakukan klasifikasi

## **Step 1.**

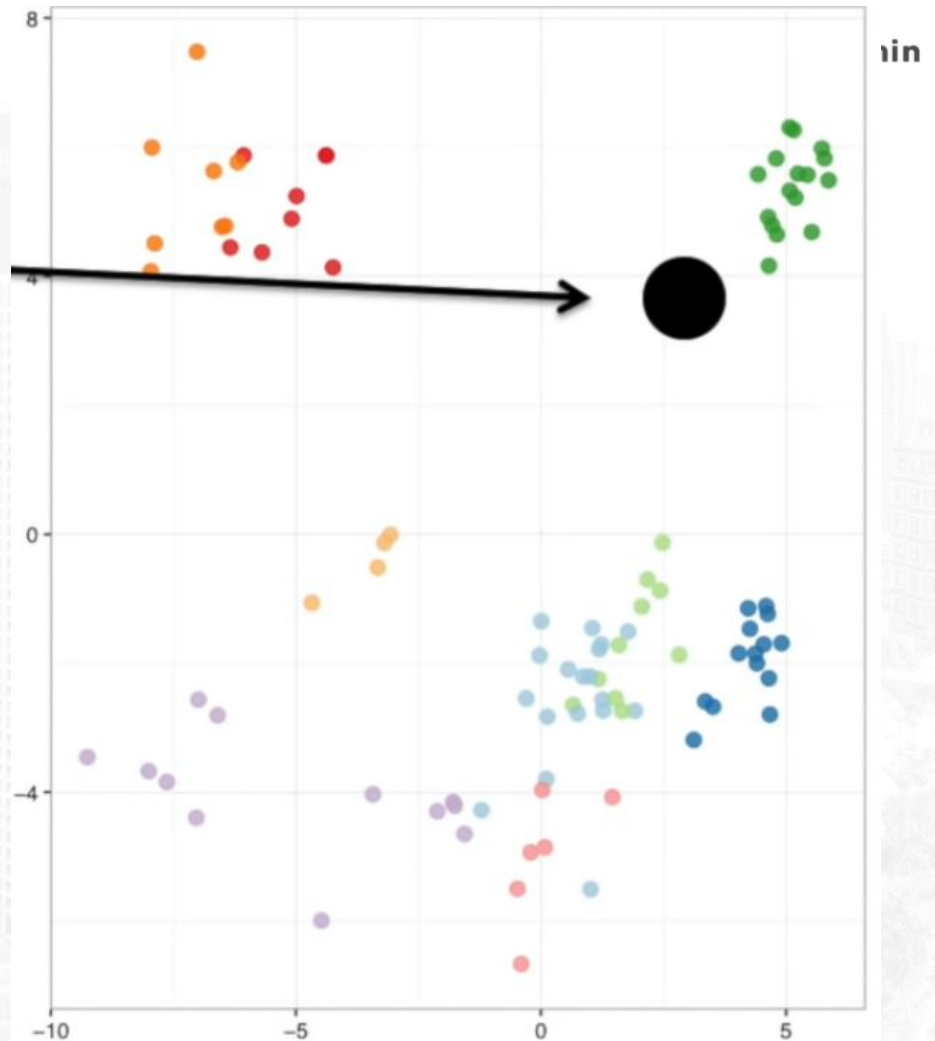
Dibutuhkan dataset dengan masing-masing labelnya



# K-Nearest Neighbor

## Step 2.

Kita mempunyai data test yang baru (warna hitam). Tujuan kita adalah ingin mengetahui label dari data test ini.

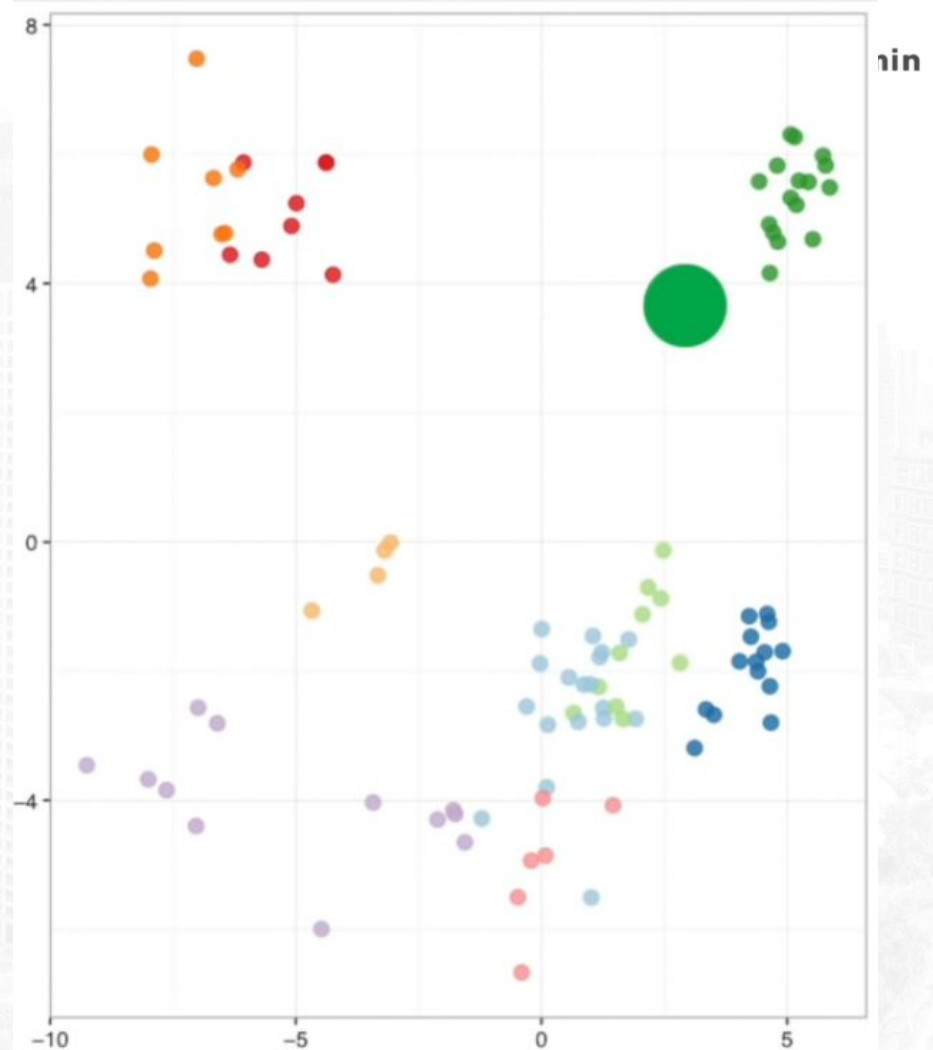


# K-Nearest Neighbor

## Step 3.

Lakukan klasifikasi terhadap data test baru dengan melihat data lainnya yang terdekat yang sudah memiliki label (nearest neighbors). Tentukan nilai K.

- Jika 'K' pada 'K-nearest neighbors' sama dengan 1, maka kita hanya menggunakan nearest neighbor untuk melabeli data test tersebut. **pada kasus ini, label data testnya adalah hijau.**
- Jika nilai  $K=7$ , kita dapat menggunakan 7 nearest neighbors, **pada kasus ini, label data testnya masih hijau.**



# K-Nearest Neighbor

Mari kita mencoba data test baru. Kali ini letaknya di tengah nih.

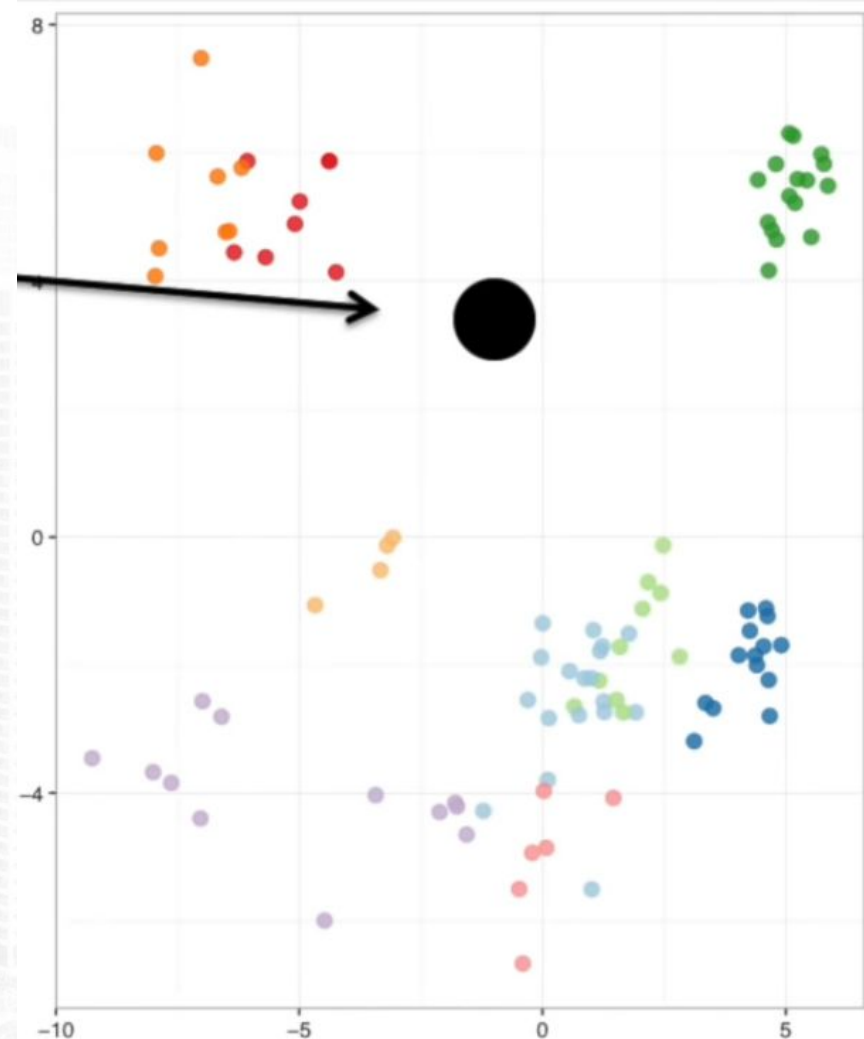
Jika nilai  $K=11$ , data yang sekarang ada diantara 2 (atau lebih) label. Jika hal ini terjadi, kita dapat menentukan label dengan cara **voting**.

Pada kasus ini:

7 nearest neighbors adalah **Merah**

3 nearest neighbors adalah **Orange**

1 nearest neighbors adalah **Hijau**



# K-Nearest Neighbor

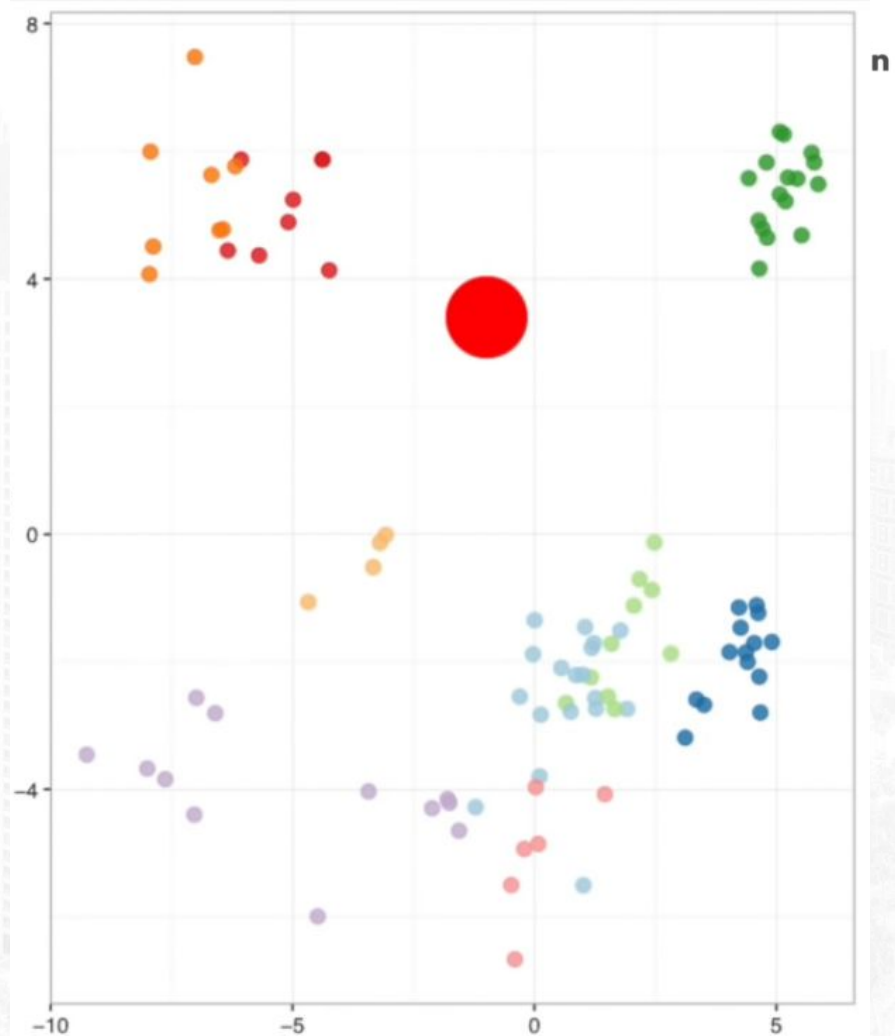
Mari kita mencoba data test baru. Kali ini letaknya di tengah nih.

Jika nilai  $K=11$ , data yang sekarang ada diantara 2 (atau lebih) label. Jika hal ini terjadi, kita dapat menentukan label dengan cara **voting**.

Pada kasus ini:

7 nearest neighbors adalah **Merah**  
3 nearest neighbors adalah **Orange**  
1 nearest neighbors adalah **Hijau**

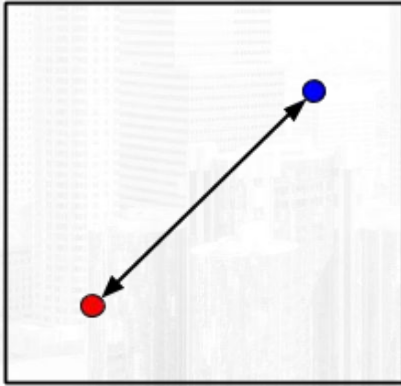
Dikarenakan **Merah** mendapatkan voting yang paling banyak, maka label akhirnya adalah **Merah**





# Distance: Perhitungan Jarak

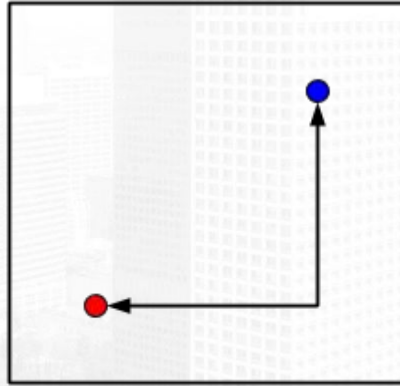
Euclidean (p=2)



Umum digunakan ketika menghitung jarak antara 2 numerical feature (float/integer)

$$\sqrt{\sum_{i=1}^N (fa_i - fb_i)^2}$$

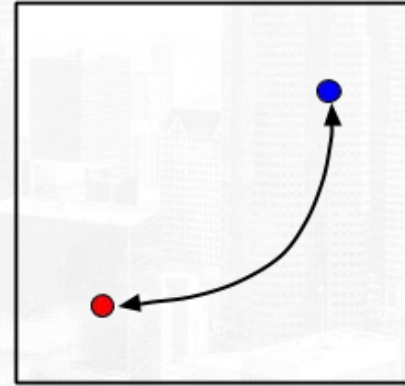
Manhattan (p=1)



digunakan ketika menghitung jarak dengan sifat grid (seperti chessboard atau city blocks)

$$\sum_{i=1}^N |fa_i - fb_i|$$

Minkowski



Generalisasi dari Euclidean & Manhattan. umum digunakan di ML algorithm karena fleksibilitasnya untuk mengukur jarak dari hyperparameter "p"

$$\left( \sum_{i=1}^n |(fa_i - fb_i)^p| \right)^{\frac{1}{p}}$$

# Beberapa tips dan summary

1. K adalah jumlah neighbor yang di cek
2. K sebaiknya angka ganjil, untuk menghindari voting seri
3. Melakukan data scaling penting sebelumnya
4. Voting dapat ditambahkan bobot (dengan masing-masing jarak contohnya) pada masing-masing neighbor. (gunakan hyperparameter weight)

# Bagaimana menemukan nilai K yang tepat?

Kita dapat menggunakan rule of thumb, dimana n adalah jumlah sampel

$$k = \sqrt{n}$$

# Kapan menggunakan K-Nearest Neighbor (kNN)

1. Salah satu algoritma paling sederhana, dapat digunakan menjadi baseline
2. kNN menghitung jarak antara masing-masing data point, cocok digunakan di data yang bersifat non-linear
3. kNN buruk digunakan dalam data yang berjumlah besar (komputasi yang boros)

## Contoh:

Jika jumlah data training: 1 juta

Jumlah Data test: 10ribu












Maka, rekomendasi nilai  $k = \sqrt{1 \text{ juta}} = 1000$

Dari setiap data testing, akan dihitung 1000 jarak

Total jarak yang dihitung untuk semua data test =  $1000 * 10,000 = 10 \text{ juta}$













Jumlah iterasi sangat besar

# Topik Supervised Learning: Classification

- |   |   |   |   |
|---|---|---|---|
|  | Review Klasifikasi                      |  | Hands On - kNN  |
|  | Logistic Regression                     |  | Decision Tree - Intuition                             |
|  | Hands On - Logistic Regression - Part 1 |  | Decision Tree - Hyperparameters & Feature Importances |
|  | Hands On - Logistic Regression - Part 2 |  | Hands On - Decision Tree                              |
|  | Hands On - Analyze Learning Curve       |  | Hands On - Feature Importance & Summary               |
|  | k-Nearest Neighbor (kNN)                |   |   |



# Topik Supervised Learning: Classification

-  Review Klasifikasi
  -  Logistic Regression
  -  Hands On - Logistic Regression - Part 1
  -  Hands On - Logistic Regression - Part 2
  -  Hands On - Analyze Learning Curve
  -  k-Nearest Neighbor (kNN)
-   Hands On - kNN
  -  Decision Tree - Intuition
  -  Decision Tree - Hyperparameters & Feature Importances
  -  Hands On - Decision Tree
  -  Hands On - Feature Importance & Summary



# Contoh Implementasi pada sklearn

```
from sklearn.neighbors import KNeighborsClassifier# import knn dari sklearn  
knn = KNeighborsClassifier() # inisiasi object dengan nama knn  
knn.fit(X_train, y_train) # fit model KNN dari data train
```

Latih model pada  
train set












*dan lakukan prediksi dan evaluasi dari data test...*

Dokumentasi sklearn kNN:













<https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html>

# Hands On!

# Topik Supervised Learning: Classification

- |   |   |   |   |
|---|---|---|---|
|  | Review Klasifikasi                      |  | Hands On - kNN  |
|  | Logistic Regression                     |  | Decision Tree - Intuition                             |
|  | Hands On - Logistic Regression - Part 1 |  | Decision Tree - Hyperparameters & Feature Importances |
|  | Hands On - Logistic Regression - Part 2 |  | Hands On - Decision Tree                              |
|  | Hands On - Analyze Learning Curve       |  | Hands On - Feature Importance & Summary               |
|  | k-Nearest Neighbor (kNN)                |   |   |

# Topik Supervised Learning: Classification

-  Review Klasifikasi
-  Logistic Regression
-  Hands On - Logistic Regression - Part 1
-  Hands On - Logistic Regression - Part 2
-  Hands On - Analyze Learning Curve
-  k-Nearest Neighbor (kNN)
-  Hands On - kNN
-   Decision Tree - Intuition
-  Decision Tree - Hyperparameters & Feature Importances
-  Hands On - Decision Tree
-  Hands On - Feature Importance & Summary

# Decision Tree

aka. CART

# Decision Tree



Membangun sebuah `tree` untuk melakukan klasifikasi (bisa berupa categorical atau numerik)

Note:

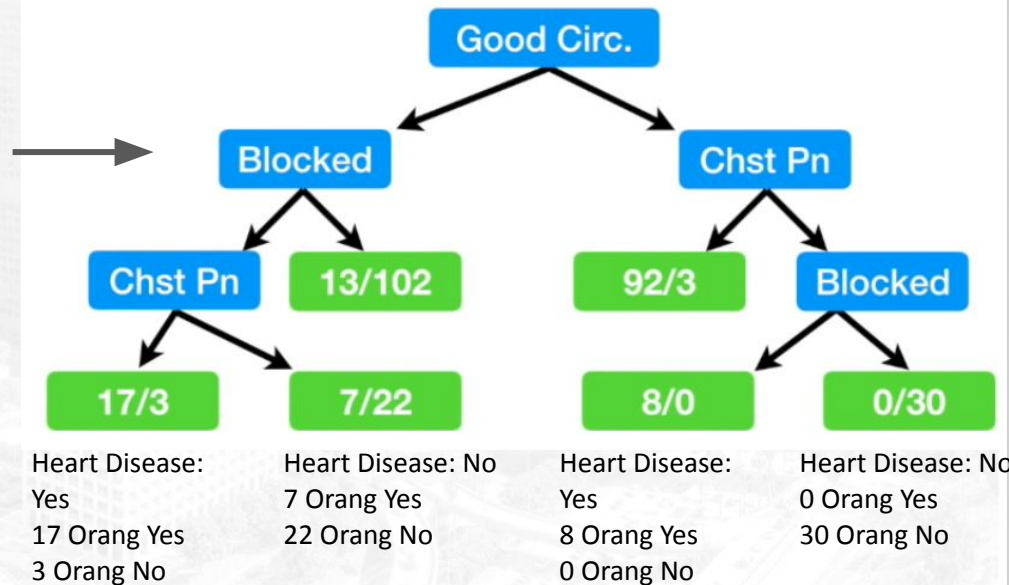
Pada tree selanjutnya setelah slide ini,

Link yang menuju sebelah **kiri bernilai TRUE**, sedangkan Link yang menuju sebelah **kanan bernilai FALSE**



# Contoh Decision Tree dari dataset yang ada

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...

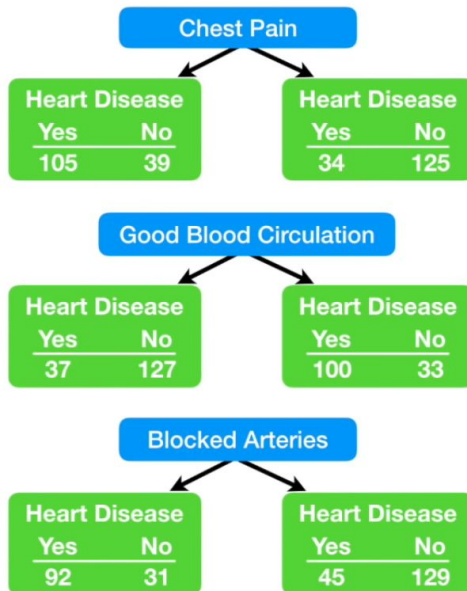


# Hitung Gini Impurity. Cek bagaimana masing-masing feature dapat melakukan prediksi terhadap label heart disease

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...

Lakukan pemetaan, jika chest pain bernilai **Yes**, berapa banyak yang mempunyai label heart disease dan tidak mempunyai heart disease.

Lakukan hal yang sama ketika chest pain bernilai **No**.

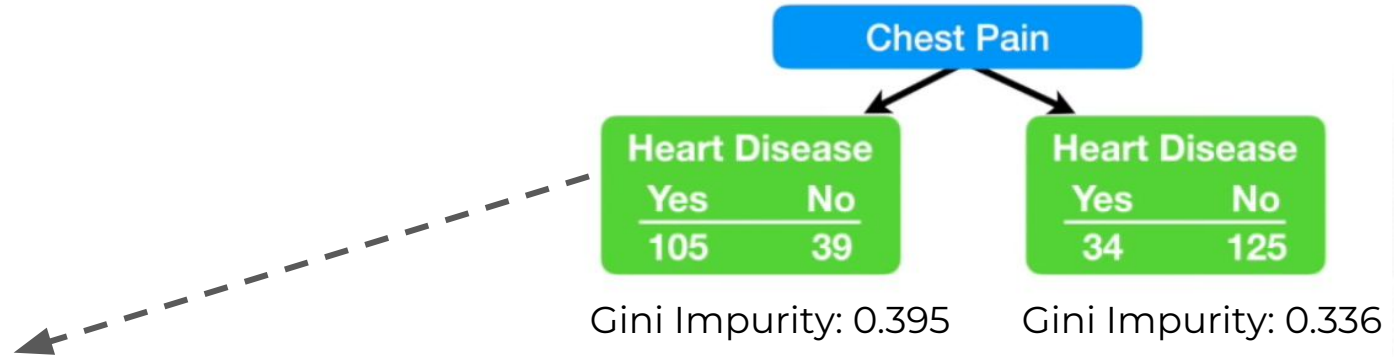


Setiap feature tidak dapat memberikan prediksi yang sempurna untuk melakukan prediksi label heart disease. Kasus ini dapat dikatakan bahwa variabel tersebut **impure**.

Untuk menentukan variabel mana yang bernilai lebih baik, kita dapat menghitung nilai **impurity**, cara yang paling sederhana adalah dengan menggunakan **Gini** algorithm.

Semakin kecil nilai impurity maka semakin baik. Pendekatan ini menjadi salah satu penentuan feature importance.

# Contoh perhitungan Gini Impurity

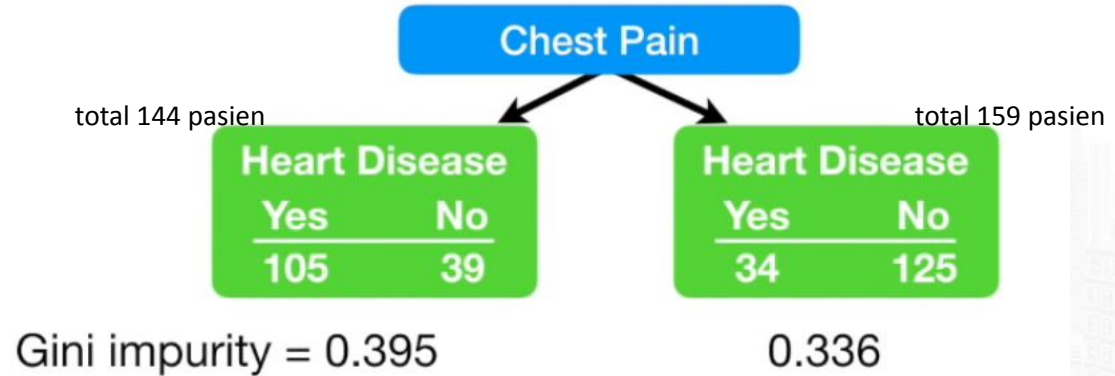


Pada leaf ini, Gini impurity =  $1 - (\text{probability of "yes"})^2 - (\text{probability of "no"})^2$

$$= 1 - \left(\frac{105}{105 + 39}\right)^2 - \left(\frac{39}{105 + 39}\right)^2$$

$$= 0.395$$

# Gini Impurity untuk Chest Pain



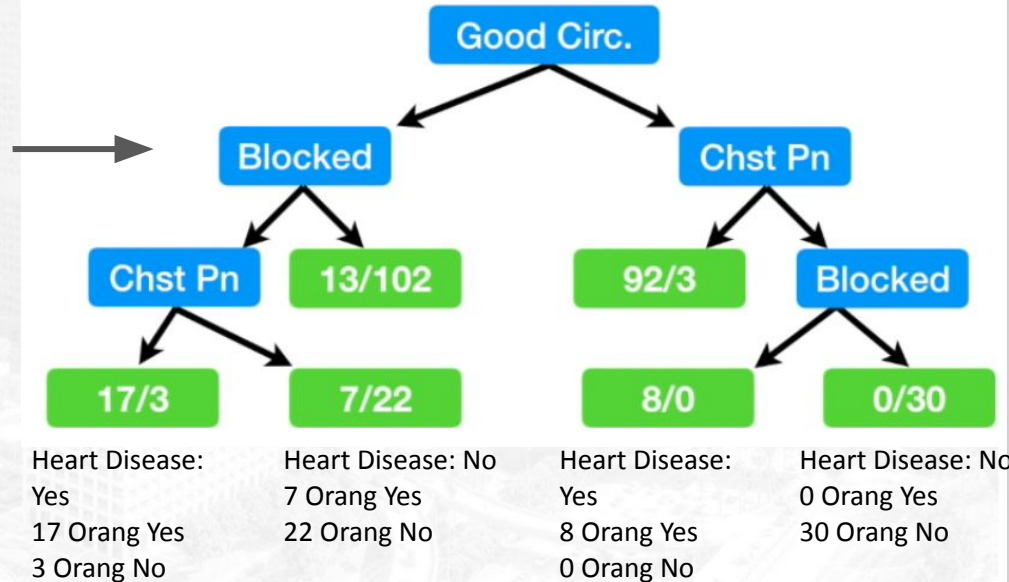
Gini Impurity (Chest Pain) = Weighted average dari Gini impurity di masing-masing node

$$= \left( \frac{144}{144 + 159} \right) 0.395 + \left( \frac{159}{144 + 159} \right) 0.336$$

$$= 0.364$$

Feature yang punya impurity paling kecil akan jadi root/node, lakukan terus hingga menjadi sebuah tree seperti dibawah ini

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...














# Detail Step By Step Decision Tree













Detail dari step by step decision tree dapat dilihat di bagian appendix dengan lebih detail.



# Topik Supervised Learning: Classification

- |   |   |   |   |
|---|---|---|---|
|  | Review Klasifikasi                      |  | Hands On - kNN  |
|  | Logistic Regression                     |  | Decision Tree - Intuition                             |
|  | Hands On - Logistic Regression - Part 1 |  | Decision Tree - Hyperparameters & Feature Importances |
|  | Hands On - Logistic Regression - Part 2 |  | Hands On - Decision Tree                              |
|  | Hands On - Analyze Learning Curve       |  | Hands On - Feature Importance & Summary               |
|  | k-Nearest Neighbor (kNN)                |   |   |

# Topik Supervised Learning: Classification

-  Review Klasifikasi
-  Logistic Regression
-  Hands On - Logistic Regression - Part 1
-  Hands On - Logistic Regression - Part 2
-  Hands On - Analyze Learning Curve
-  k-Nearest Neighbor (kNN)
-  Hands On - kNN
-  Decision Tree - Intuition
-   Decision Tree - Hyperparameters & Feature Importances
-  Hands On - Decision Tree
-  Hands On - Feature Importance & Summary



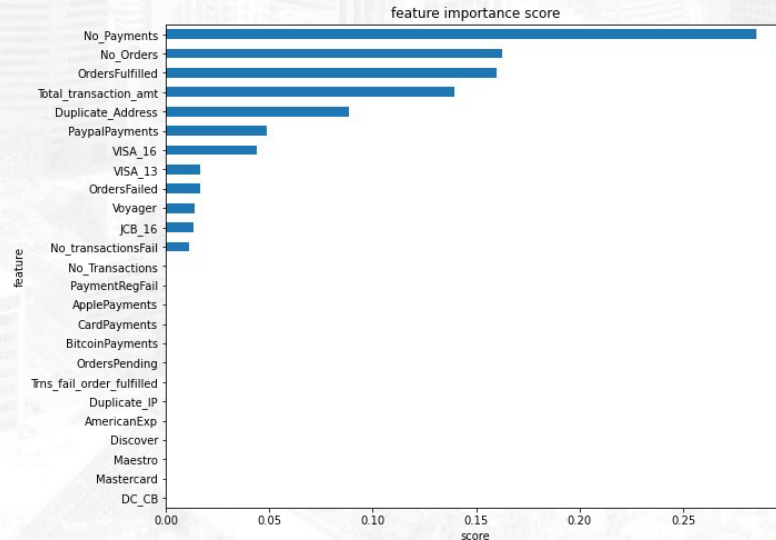
# Feature Importance

```
from sklearn.tree import DecisionTreeClassifier# import decision tree dari sklearn
dt = DecisionTreeClassifier() # inisiasi object dengan nama dt
dt.fit(X_train, y_train) # fit model decision tree dari data train
```

Latih model pada train set












```
dt.feature_importances_
```

Dapatkan hasil feature importance dari model















*Kita dapat menggunakan hasil dari feature importance sebagai business insight dan juga menggunakan feature ini sebagai tambahan eksperimen kita pada model (untuk mengurangi kompleksitas dan menghindari curse of dimensionality)*

# Topik Supervised Learning: Classification

- |   |   |   |   |
|---|---|---|---|
|  | Review Klasifikasi                      |  | Hands On - kNN  |
|  | Logistic Regression                     |  | Decision Tree - Intuition                             |
|  | Hands On - Logistic Regression - Part 1 |  | Decision Tree - Hyperparameters & Feature Importances |
|  | Hands On - Logistic Regression - Part 2 |  | Hands On - Decision Tree                              |
|  | Hands On - Analyze Learning Curve       |  | Hands On - Feature Importance & Summary               |
|  | k-Nearest Neighbor (kNN)                |   |   |



# Topik Supervised Learning: Classification

-  Review Klasifikasi
-  Logistic Regression
-  Hands On - Logistic Regression - Part 1
-  Hands On - Logistic Regression - Part 2
-  Hands On - Analyze Learning Curve
-  k-Nearest Neighbor (kNN)
-  Hands On - kNN
-  Decision Tree - Intuition
-  Decision Tree - Hyperparameters & Feature Importances
-   Hands On - Decision Tree
-  Hands On - Feature Importance & Summary



# Contoh Implementasi pada sklearn

```
from sklearn.tree import DecisionTreeClassifier# import decision tree dari sklearn
dt = DecisionTreeClassifier() # inisiasi object dengan nama dt
dt.fit(X_train, y_train) # fit model decision tree dari data train
```

Latih model pada  
train set












*dan lakukan prediksi dan evaluasi dari data test...*

Dokumentasi sklearn Decision Tree:













<https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html>

# Hands on

# Topik Supervised Learning: Classification

- |   |   |   |   |
|---|---|---|---|
|  | Review Klasifikasi                      |  | Hands On - kNN  |
|  | Logistic Regression                     |  | Decision Tree - Intuition                             |
|  | Hands On - Logistic Regression - Part 1 |  | Decision Tree - Hyperparameters & Feature Importances |
|  | Hands On - Logistic Regression - Part 2 |  | Hands On - Decision Tree                              |
|  | Hands On - Analyze Learning Curve       |  | Hands On - Feature Importance & Summary               |
|  | k-Nearest Neighbor (kNN)                |   |   |

# Topik Supervised Learning: Classification

- |   |   |   |   |
|---|---|---|---|
|  | Review Klasifikasi                      |    | Hands On - kNN  |
|  | Logistic Regression                     |    | Decision Tree - Intuition                             |
|  | Hands On - Logistic Regression - Part 1 |    | Decision Tree - Hyperparameters & Feature Importances |
|  | Hands On - Logistic Regression - Part 2 |    | Hands On - Decision Tree                              |
|  | Hands On - Analyze Learning Curve       |   | Hands On - Feature Importance & Summary               |
|  | k-Nearest Neighbor (kNN)                |   |   |

# Hands on

## In Summary: Kapan pakai masing-masing algoritma?

Tergantung kasus datanya :)

- Jika datanya bisa dipisahkan secara linear, gunakan logistic regression, sebaliknya jika non linear, decision tree merupakan pendekatan yang lebih baik
  - Namun, jika tidak yakin, kita bisa menggunakan logistic regression sebagai baseline, dan mencoba algoritma lainnya yang menghasilkan performa lebih baik
- Jika data bersifat lebih banyak categorical, decision tree cenderung lebih baik performanya
- Jika mau yang simple dan fast computation, gunakan logistic regression
- Jika datanya tidak balance, decision tree dan improvement-nya cenderung lebih baik



# Topik Supervised Learning: Classification

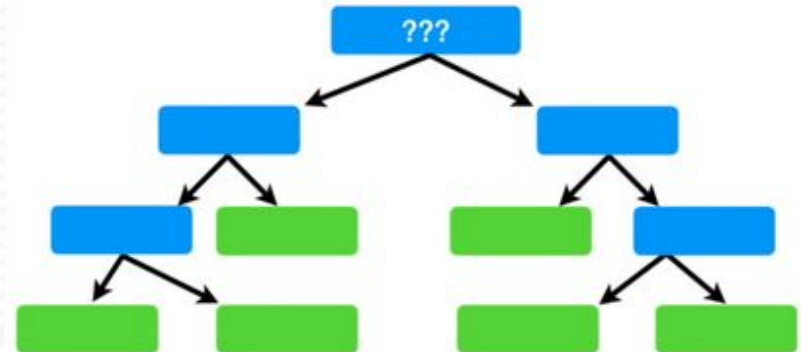
- ✓ Review Klasifikasi
- ✓ Logistic Regression
- ✓ Hands On - Logistic Regression - Part 1
- ✓ Hands On - Logistic Regression - Part 2
- ✓ Hands On - Analyze Learning Curve
- ✓ k-Nearest Neighbor (kNN)
- ✓ Hands On - kNN
- ✓ Decision Tree - Intuition
- ✓ Decision Tree - Hyperparameters & Feature Importances
- ✓ Hands On - Decision Tree
- ✓ Hands On - Feature Importance & Summary

# **Appendix: Step by Step Decision Tree**

# Bagaimana cara membentuk decision tree dari awal?

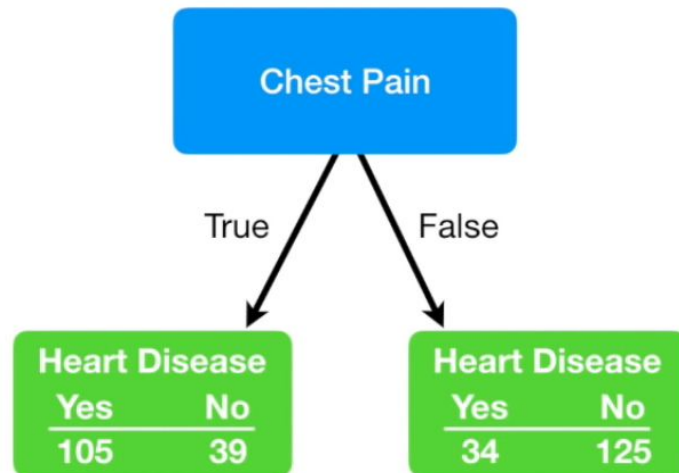
Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...

Langkah awal yang perlu dilakukan adalah menentukan variabel mana yang akan menjadi root?



# Step 1. Cek bagaimana variable **Chest Pain** saja dapat melakukan prediksi terhadap label heart disease

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...

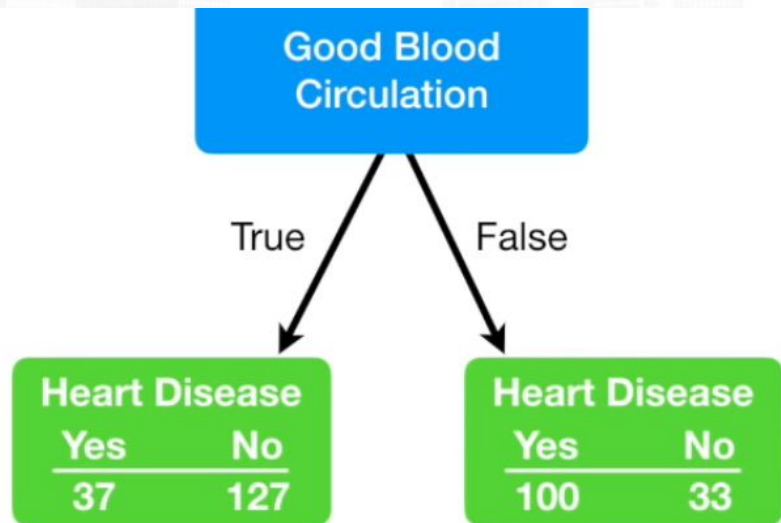


Lakukan pemetaan, jika chest pain bernilai **Yes**, berapa banyak yang mempunyai label heart disease dan tidak mempunyai heart disease.

Lakukan hal yang sama ketika chest pain bernilai **No**.

## Step 2. Bagaimana variable **Blood Circulation** saja melakukan prediksi terhadap label heart

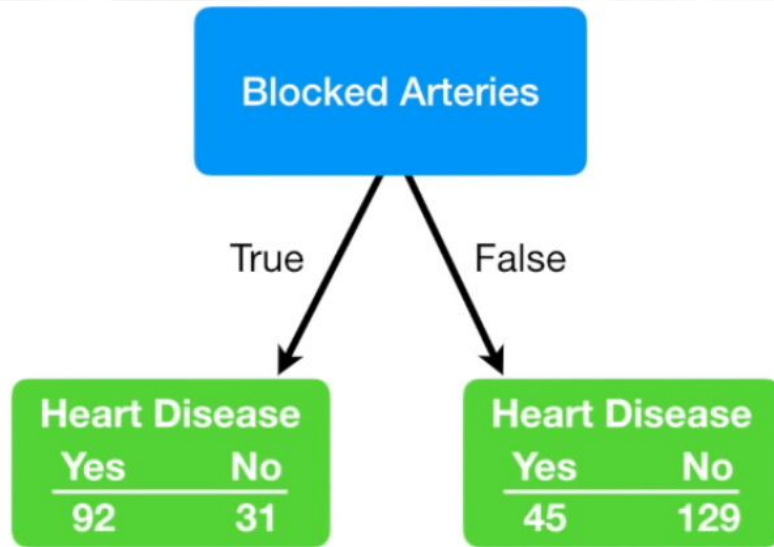
Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...



Lakukan pemetaan yang serupa seperti chest pain sebelumnya...

## Step 3. Bagaimana variable **Block Arteries** saja melakukan prediksi terhadap label heart disease

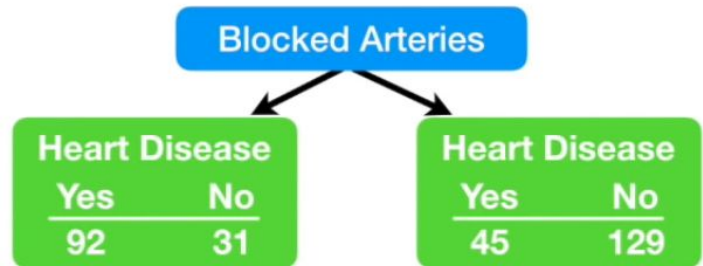
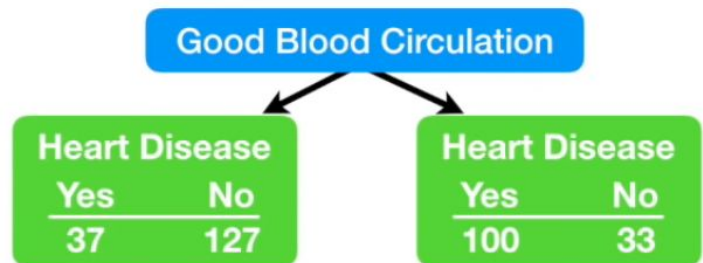
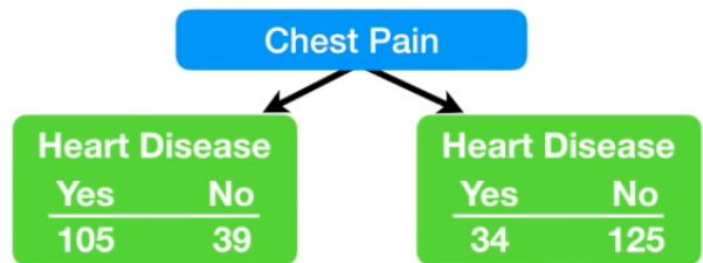
Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...



Lakukan pemetaan yang serupa seperti chest pain dan blood circulation sebelumnya...



# Mari kita summarize hasil sebelumnya



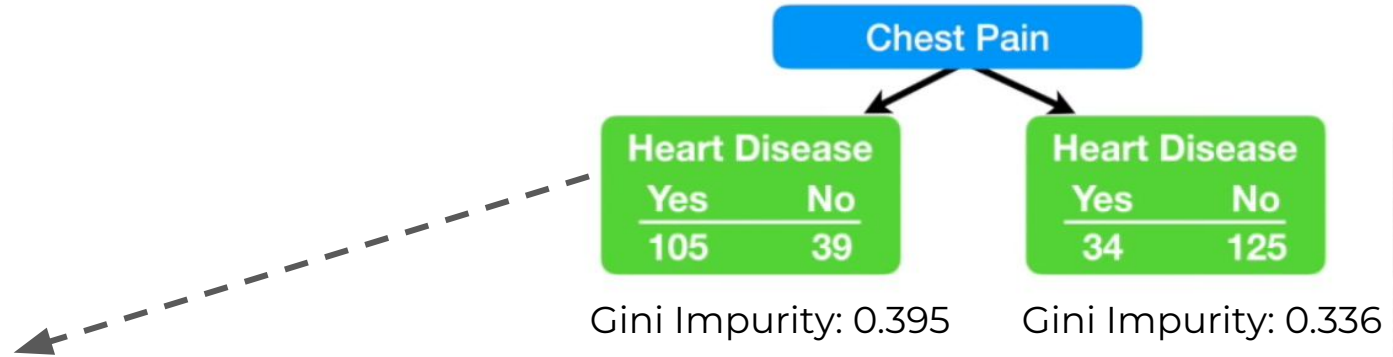
Setiap variable tidak dapat memberikan prediksi yang sempurna untuk melakukan prediksi label heart disease.

Karena tidak ada leaf yang menunjukkan bahwa dengan variabel tersebut dapat melakukan prediksi 100% YES heart disease atau 100% NO heart disease. Kasus ini dapat dikatakan bahwa variabel tersebut **impure**.

Untuk menentukan variabel mana yang bernilai lebih baik, kita dapat menghitung nilai **impurity**, cara yang paling sederhana adalah dengan menggunakan **Gini** algorithm.

Semakin kecil nilai impurity maka semakin baik. Hal ini digunakan dalam penentuan feature importance.

# Gini Impurity



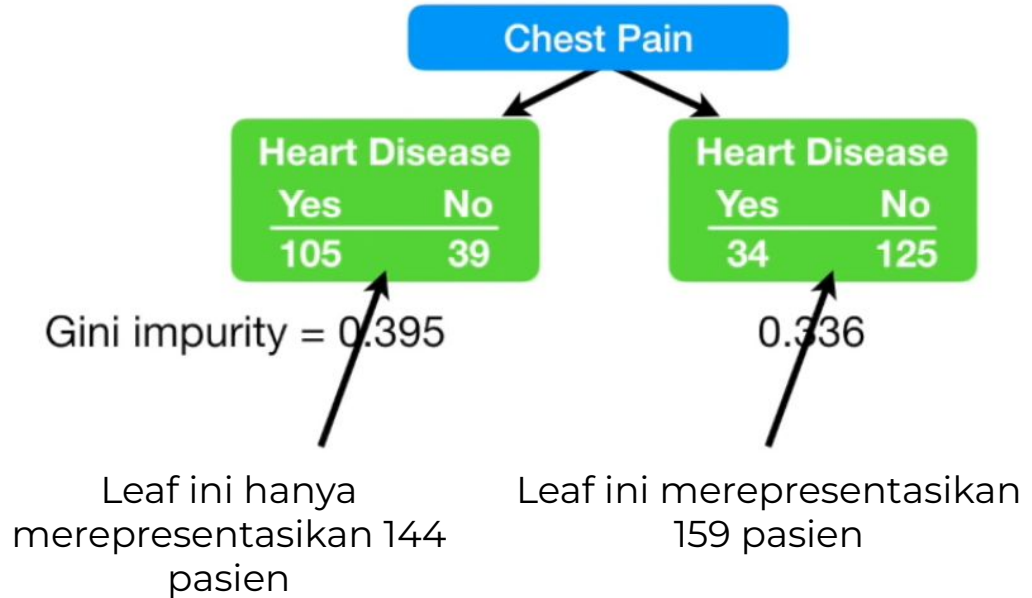
Pada leaf ini, Gini impurity =  $1 - (\text{probability of "yes"})^2 - (\text{probability of "no"})^2$

$$= 1 - \left( \frac{105}{105 + 39} \right)^2 - \left( \frac{39}{105 + 39} \right)^2$$

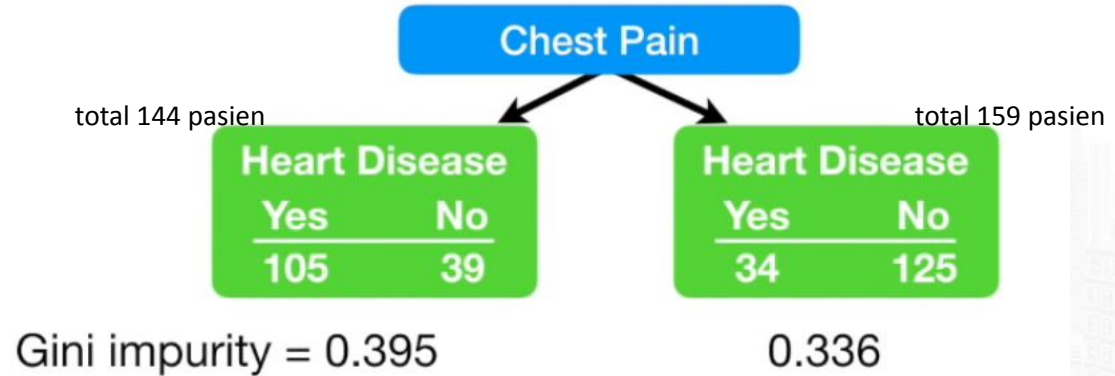
$$= 0.395$$

# Gini Impurity (Weighted Average)

Dikarenakan masing-masing leaf memiliki jumlah sampel yang berbeda, untuk menghitung Gini Impurity dari variabel Chest Pain, perlu dilakukan weighted average berdasarkan jumlah sampel yang sesuai.



# Gini Impurity untuk Chest Pain



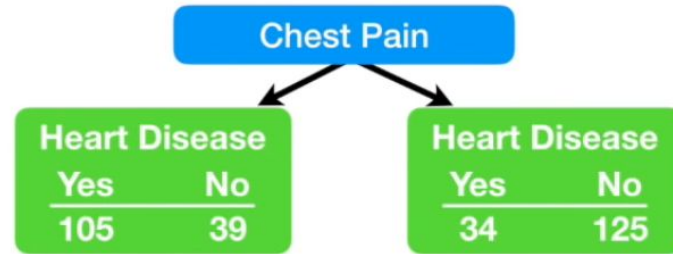
Gini Impurity (Chest Pain) = Weighted average dari Gini impurity di masing-masing node

$$= \left( \frac{144}{144 + 159} \right) 0.395 + \left( \frac{159}{144 + 159} \right) 0.336$$

$$= 0.364$$

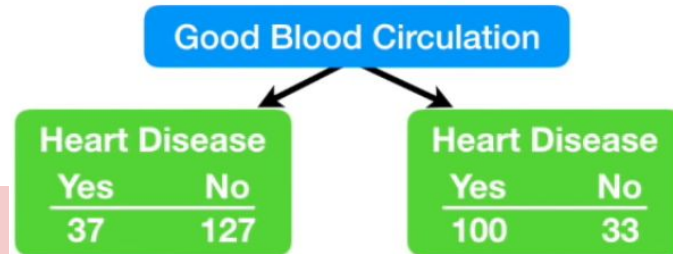
# Hitung Gini Impurity untuk variabel lainnya

Gini Impurity untuk Chest Pain =  
0.364

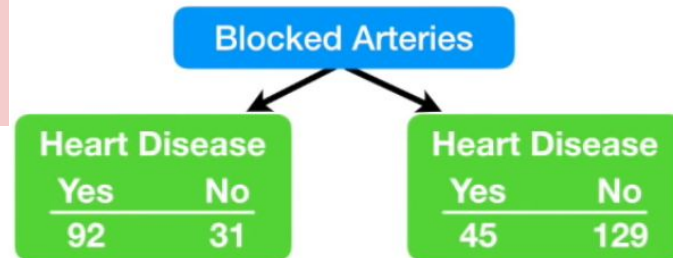


Gini Impurity untuk Good Blood  
Circulation = **0.360**

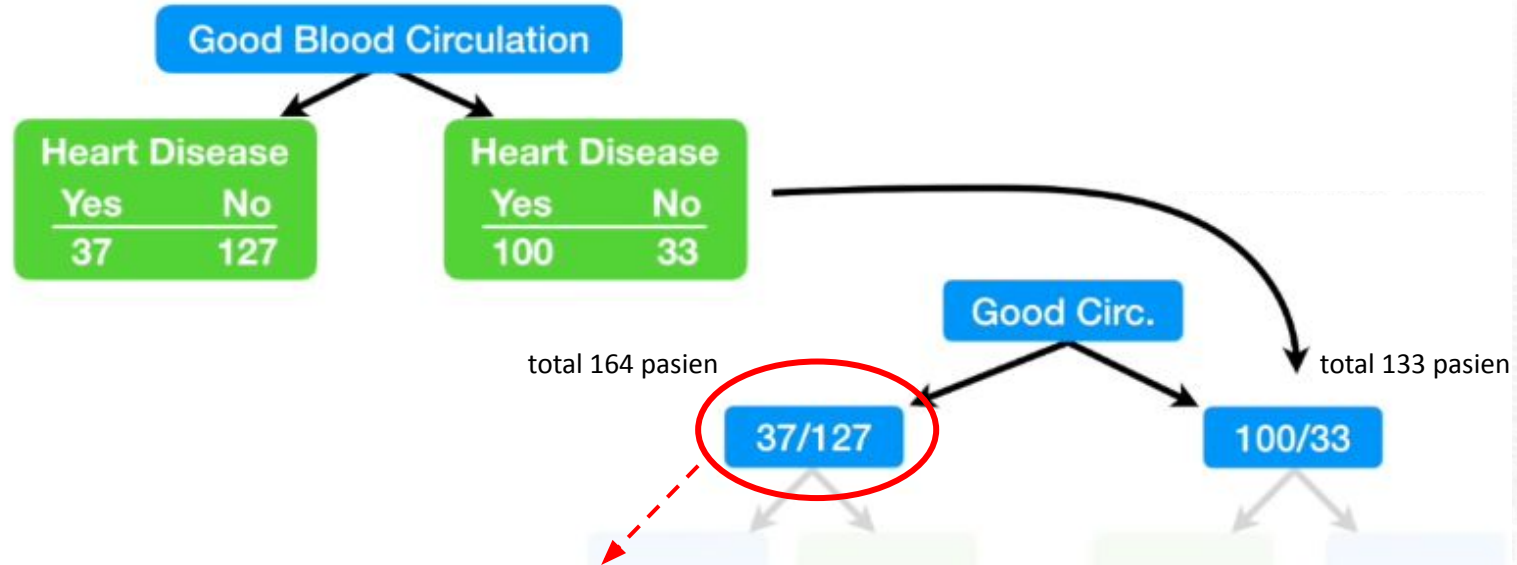
Variabel dengan nilai Gini Impurity yang paling rendah digunakan sebagai root pada tree



Gini Impurity untuk Blocked Arteries =  
0.381



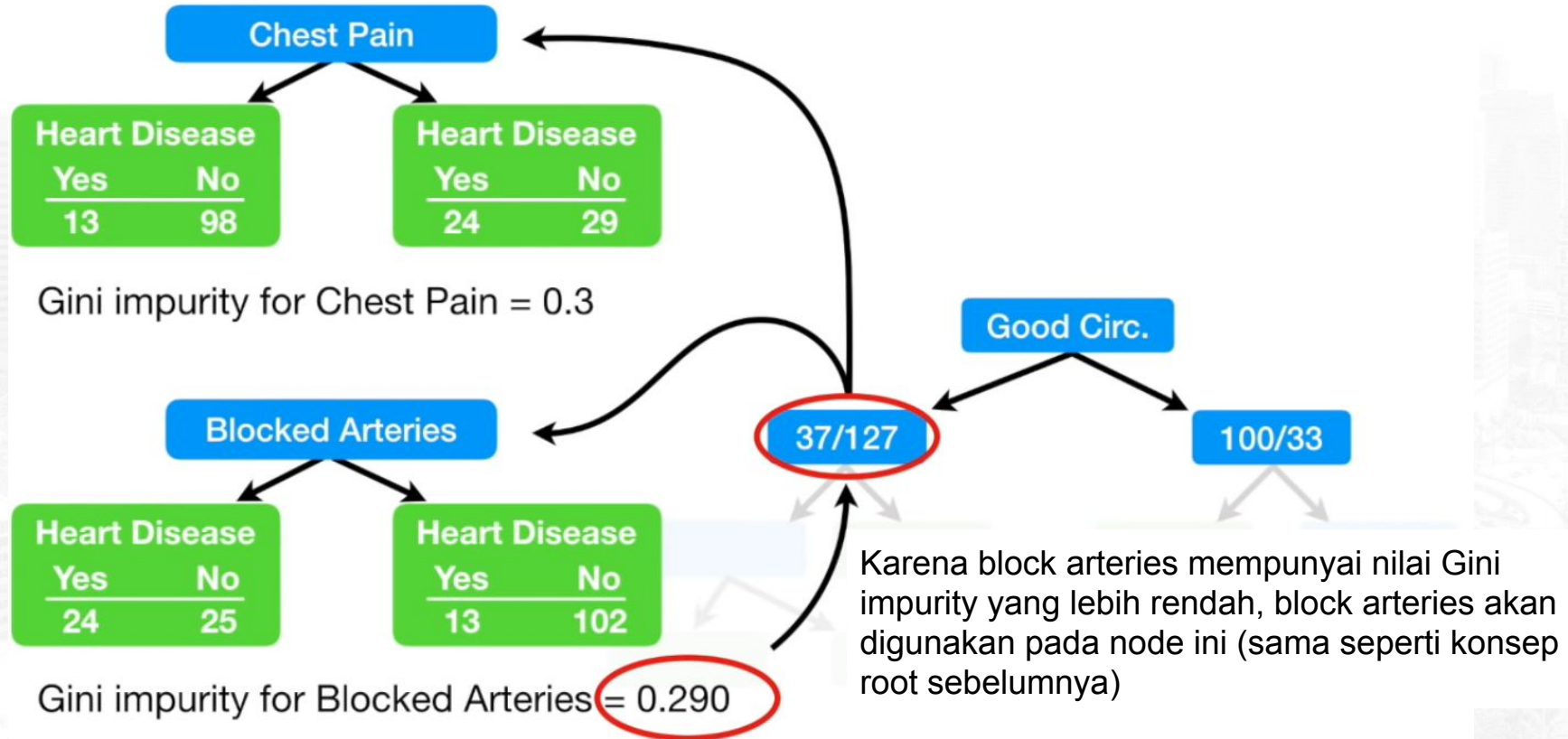
# Good Blood Circulation sebagai Root Tree



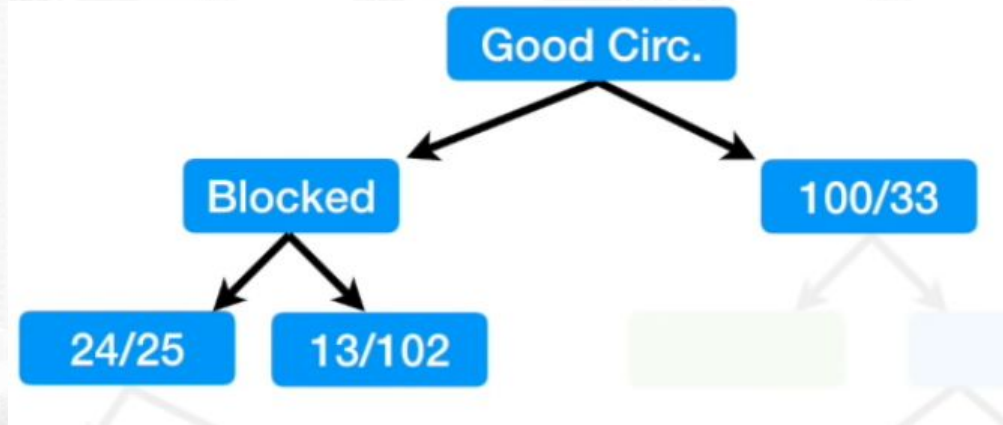
Setelah itu kita perlu mengetahui gini impurity dari chest pain / block arteries pada 164 pasien (good blood circulation) ini untuk prediksi heart disease. Lakukan juga dengan 133 pasien sisanya (not good blood circulation)



# Hitung Gini Impurity pada variabel lainnya



**Kemudian, kita akan memiliki tree yang berbentuk seperti ini**



**Lakukan dengan metode yang sama untuk semua variabel hingga kita mendapatkan semua leaf secara lengkap.**

Lakukan terus dengan iterasi yang sama hingga kita memiliki tree yang lengkap untuk melakukan klasifikasi

