MACHINE LEARNING

Nama : Hanif Ridal Warits

NPM : 41155050210060

Kelas : Informatika A2 - 2021

Tugas Pertemuan 2

1.0. Lakukan praktek dari https://youtu.be/lcjq7-2zMSA?si=f4jWJR6lY8y0BZKl dan buat screen shot hasil run dengan nama anda pada hasil run tersebut. Praktek tersebut yaitu:

1.1. Sample dataset

[46]:
```
print('Hanif Ridal Warits - 41155050210060')
```

```
Hanif Ridal Warits - 41155050210060
```

```
05 SIMPLE LINEAR REGRESSION
```

```
Simple Linear Regression memodelkan hubungan antara sebuah response variable dengan sebuah explanatory variable sebagai sebuah suatu garis lurus(linear)
```

[47]:
```
import pandas as pd

pizza = {'diameter': [6, 8, 10, 14, 18],
         'price' : [7, 9, 13, 17.5, 18]}
pizza_df = pd.DataFrame(pizza)
pizza_df
```

[47]:

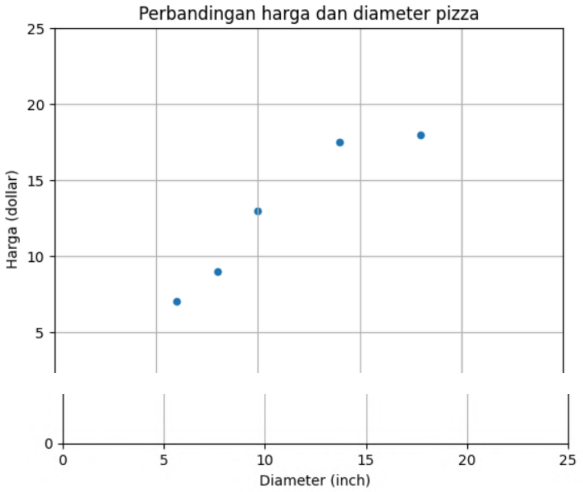
	diameter	price
0	6	7.0
1	8	9.0
2	10	13.0
3	14	17.5
4	18	18.0

Visualisasi Data

1.2. Visualisasi dataset

Visualisasi Data

[48]:
```
import matplotlib.pyplot as plt

pizza_df.plot(kind='scatter', x='diameter', y='price')

plt.title('Perbandingan harga dan diameter pizza')
plt.xlabel('Diameter (inch)')
plt.ylabel('Harga (dollar)')
plt.xlim(0, 25)
plt.ylim(0, 25)
plt.grid(True)
plt.show()
```



1.3. Transformasi dataset

```
PENYESUAIAN DATASET
```

[49]:
```
import numpy as np

X = np.array(pizza_df['diameter'])
y = np.array(pizza_df['price'])

print(f'X: {X}')
print(f'y: {y}')
```

```
X: [ 6  8 10 14 18]
y: [ 7.   9.  13.  17.5 18. ]
```

[50]:
```
X = X.reshape(-1,1)
X.shape
```

[50]:
```
(5, 1)
```

[51]:
```
X
```

[51]:
```
array([[ 6],
       [ 8],
       [10],
       [14],
       [18]])
```

1.4. Training Simple Linear Regression Model

```
TRAINING SIMPLE LINEAR REGRESSION
```

[52]:
```
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(X, y)
```

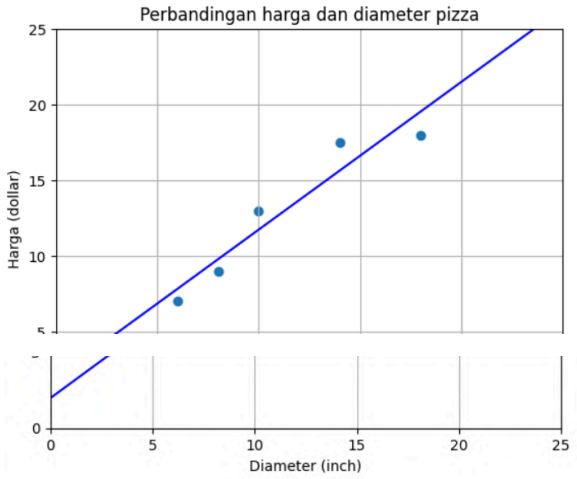
[52]:
```
LinearRegression()
```

1.5. Visualisasi Simple Linear Regression Model | Penjelasan persamaan garis linear

```
VISUALISASI SIMPLE LINEAR REGRESSION MODEL
```

[53]:
```
X_vis = np.array([0, 25]).reshape(-1, 1)
y_vis = model.predict(X_vis)
```

[54]:
```
plt.scatter(X, y)
plt.plot(X_vis, y_vis, '-b')

plt.title('Perbandingan harga dan diameter pizza')
plt.xlabel('Diameter (inch)')
plt.ylabel('Harga (dollar)')
plt.xlim(0, 25)
plt.ylim(0, 25)
plt.grid(True)
plt.show()
```



1.6. Kalkulasi nilai slope

```
MENCARI NILAI SLOPE
Nilai slope pada Linear regression bisa diperoleh dengan memanfaatkan formula : B = cov(x,y)/var(x)
```

[56]:
```
print(f'X : \n{X}\n')
print(f'X flatten : {X.flatten()}\n')
print(f'y : {y}')
```

```
X :
[[ 6]
 [ 8]
 [10]
 [14]
 [18]]

X flatten : [ 6  8 10 14 18]

y : [ 7.   9.  13.  17.5 18. ]
```

```
VARIANCE
```

[57]:
```
variance_x = np.var(X.flatten(), ddof=1)

print(f'variance : {variance_x}')
```

```
variance : 23.2
```

```
COVARIANCE
```

[58]:
```
np.cov(X.flatten(), y)
```

[58]:
```
array([[23.2 , 22.65],
       [22.65, 24.3 ]])
```

[59]:
```
covariance_xy = np.cov(X.flatten(), y)[0][1]

print(f'covariance : {covariance_xy}')
```

```
covariance : 22.650000000000002
```

```
SLOPE
```

[60]:
```
slope = covariance_xy / variance_x

print(f'slope : {slope}')
```

```
slope : 0.976293103448276
```

1.7. Kalkukasi nilai intercept

```
MENCCARI NILAI INTERCEPT
pada Linear Regression bisa diperoleh dengan memanfaatkan formula : alpha =
```

[61]:
```
intercept = np.mean(y) - slope * np.mean(X)

print(f'intercept: {intercept}')
```

```
intercept: 1.9655172413793096
```

1.8. Prediksi harga pizza dengan Simple Linear Regression Model

```
Prediksi Harga PIZZA
```

[62]:
```
diameter_pizza = np.array([12, 20, 23]).reshape(-1, 1)
diameter_pizza
```

[62]:
```
array([[12],
       [20],
       [23]])
```

[63]:
```
prediksi_harga = model.predict(diameter_pizza)
prediksi_harga
```

[63]:
```
array([13.68103448, 21.49137931, 24.42025862])
```

[64]:
```
for dmtr, hrg in zip(prediksi_harga, prediksi_harga):
    print(f'Diameter : {dmtr} prediksi harga : {hrg}')
```

```
Diameter : 13.681034482758621 prediksi harga : 13.681034482758621
Diameter : 21.491379310344826 prediksi harga : 21.491379310344826
Diameter : 24.42025862068965 prediksi harga : 24.42025862068965
```

1.9. Evaluasi model dengan Coefficient of Determination | R Squared

```
EVALUASI SLR MODEL
Training & Testing Dataset
```

[65]:
```
X_train = np.array([6, 8, 10, 14, 18]).reshape(-1, 1)
y_train = np.array([7, 9, 13, 17.5, 18])

X_test = np.array([8, 9, 11, 16, 12]).reshape(-1, 1)
y_test = np.array([11, 8.5, 15, 18, 11])
```

```
Training SLR Model
```

model = LinearRegression() model.fit(X_train(), y_train())

1.10. Kalkulasi nilai R Squared | Coefficient of Determination

```
Evaluasi LR Model dengan Cofficient of Determination atau R-squared(R^2)
```

[66]:
```
from sklearn.metrics import r2_score
y_pred = model.predict(X_test)

r_squared = r2_score(y_test, y_pred)

print (f'R-Squared : {r_squared}')

# r square salah satu teknik evaluasi
# semakin mendekati 0, model makin buruk apalagi -(negatif)
```

```
R-Squared : 0.6620052929422553
```

```
Mencari Nilai R-squared(R^2)
```

SSres

[67]:
```
ss_res = sum([(y_i - model.predict(x_i.reshape(-1, 1))[0])**2
              for x_i, y_i in zip(X_test, y_test)])
print (f'ss_res : {ss_res}')
```

```
ss_res : 19.1980993608799
```

```
SStot
```

[68]:
```
mean_y = np.mean(y_test)
ss_tot = sum([(y_i - mean_y)**2 for y_i in y_test])

print (f'ss_tot : {ss_tot}')
```

```
ss_tot : 56.8
```

```
R^2
```

[69]:
```
r_squared = 1 - (ss_res / ss_tot)

print (f'R-Squared : {r_squared}')
```

```
R-Squared : 0.6620052929422553
```

2.0. Lakukan praktek dari

https://youtu.be/nWJUJenAyB8?si=BQDzWwrMnr8jtzpV dan buat screen shot hasil run dengan nama anda pada hasil run tersebut. Praktek tersebut yaitu:

2.1. Persiapan sample dataset

[1]:
```
print('Hanif Ridal Warits - 41155050210060')
```

```
Hanif Ridal Warits - 41155050210060
```

```
SAMPLE DATASET
```

```
Training Dataset
```

[2]:
```
import pandas as pd

pizza = {'diameter' : [6, 8, 10, 14, 18],
         'n_topping': [2, 1, 0, 2, 0],
         'harga' : [7, 9, 13, 17.5, 18]}

train_pizza_df = pd.DataFrame(pizza)
train_pizza_df
```

[2]:

	diameter	n_topping	harga
0	6	2	7.0
1	8	1	9.0
2	10	0	13.0
3	14	2	17.5
4	18	0	18.0

```
Testing Dataset
```

[4]:
```
pizza = {'diameter' : [8, 9, 11, 16, 12],
         'n_topping': [2, 0, 2, 2, 0],
         'harga' : [11, 8.5, 15, 18, 11]}
#regression task

test_pizza_df = pd.DataFrame(pizza)
test_pizza_df
```

[4]:

	diameter	n_topping	harga
0	8	2	11.0
1	9	0	8.5
2	11	2	15.0
3	16	2	18.0
4	12	0	11.0

2.2. Preprocessing dataset

```
Preprocessing Dataset
```

[5]:
```
import numpy as np

X_train = np.array(train_pizza_df[['diameter', 'n_topping']])
y_train = np.array(train_pizza_df['harga'])

print(f'X_train:\n{X_train}\n')
print(f'y_train: {y_train}')
```

```
X_train:
[[ 6  2]
 [ 8  1]
 [10  0]
 [14  2]
 [18  0]]

y_train: [ 7.   9.  13.  17.5 18. ]
```

```
Prepocessing Dataset
```

[6]:
```
X_test = np.array(test_pizza_df[['diameter', 'n_topping']])
y_test = np.array(test_pizza_df['harga'])

print(f'X_test:\n{X_test}\n')
print(f'y_test: {y_test}')
```

```
X_test:
[[ 8  2]
 [ 9  0]
 [11  2]
 [16  2]
 [12  0]]

y_test: [11.   8.5 15.  18.  11. ]
```

2.3. Pengenalan Multiple Linear Regression | Apa itu Multiple Linear Regression?

```
MULTIPLE LINEAR REGRESSION
merupakan generalisasi dari SLR yang memungkinkan untuk menggunakan beberapa explanatory variable
```

[8]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score

model = LinearRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

print (f'r_sequared: {r2_score(y_test, y_pred)}')
```

```
r_sequared: 0.7701677731318468
```

2.4. Pengenalan Polynomial Regression | Apa itu Polynomial Regression?

```
POLYNOMIAL REGRESSION
memodelkan hubungan antara independent variable x dan dependent variable y sebagai derajat polynomial dalam x.
y = a + b1x b2x + .. + bnxn
```

```
Prepocessing Dataset
```

[9]:
```
X_train = np.array(train_pizza_df['diameter']).reshape(-1, 1)
y_train = np.array(train_pizza_df['harga'])

print(f'X_train:\n{X_train}\n')
print(f'y_train: {y_train}')
```

```
X_train:
[[ 6]
 [ 8]
 [10]
 [14]
 [18]]

y_train: [ 7.   9.  13.  17.5 18. ]
```

2.5. Quadratic Polynomial Regression

```
POLYNOMIAL REGRESSION : QUADRATIC
y = a + b1x b2x^2

polynomial Features
```

[10]:
```
from sklearn.preprocessing import PolynomialFeatures

quadratic_feature = PolynomialFeatures(degree=2)
X_train_quadratic = quadratic_feature.fit_transform(X_train)

print (f'X_train_quadratic:\n{X_train_quadratic}\n')
```

```
X_train_quadratic:
[[  1.   6.  36.]
 [  1.   8.  64.]
 [  1.  10. 100.]
 [  1.  14. 196.]
 [  1.  18. 324.]]
```

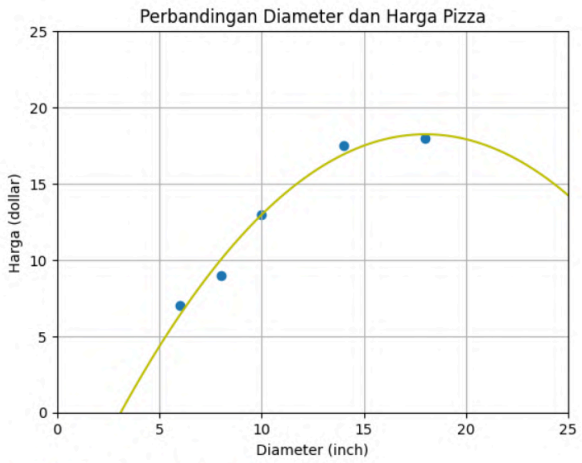
```
Training Model
```

[11]:
```
model = LinearRegression()
model.fit(X_train_quadratic, y_train)
```

[11]:
```
LinearRegression
LinearRegression()
```

```
Visualisasi Model
```

[13]:
```
import matplotlib.pyplot as plt

X_vis = np.linspace(0, 25, 100).reshape(-1, 1)
X_vis_quadratic = quadratic_feature.transform(X_vis)
y_vis_quadratic = model.predict(X_vis_quadratic)

plt.scatter(X_train, y_train)
plt.plot(X_vis, y_vis_quadratic, '-y')

plt.title('Perbandingan Diameter dan Harga Pizza')
plt.xlabel('Diameter (inch)')
plt.ylabel('Harga (dollar)')
plt.xlim(0, 25)
plt.ylim(0, 25)
plt.grid(True)
plt.show()

print('Hanif Ridal Warits - 41155050210060')
```



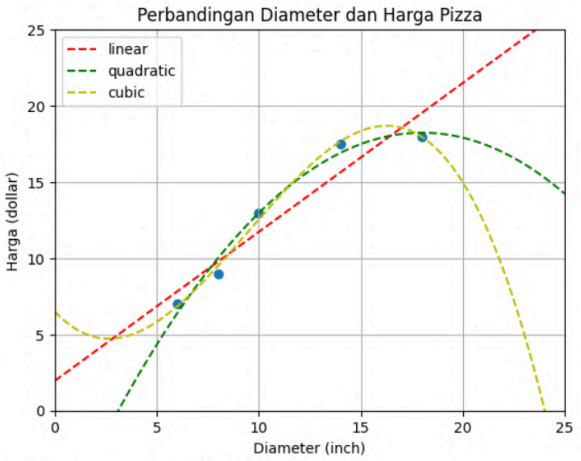

```
Hanif Ridal Warits - 41155050210060
```

2.6. Linear Regression vs Quadratic Polynomial Regression vs Cubic Polynomial Regression

POLYNOMIAL REGRESSION : QUADRATIC vs QUBIC

[14]:
```
# Training Set
plt.scatter(X_train, y_train)

# Linear
model = LinearRegression()
model.fit(X_train, y_train)
X_vis = np.linspace(0, 25, 100).reshape(-1, 1)
y_vis = model.predict(X_vis)
plt.plot(X_vis, y_vis, '--r', label='linear')

# Quadratic
quadratic_feature = PolynomialFeatures(degree=2)
X_train_quadratic = quadratic_feature.fit_transform(X_train)
model = LinearRegression()
model.fit(X_train_quadratic, y_train)
X_vis_quadratic = quadratic_feature.transform(X_vis)
y_vis = model.predict(X_vis_quadratic)
plt.plot(X_vis, y_vis, '--g', label='quadratic')

# Cubic
cubic_feature = PolynomialFeatures(degree=3)
X_train_cubic = cubic_feature.fit_transform(X_train)
model = LinearRegression()
model.fit(X_train_cubic, y_train)
X_vis_cubic = cubic_feature.transform(X_vis)
y_vis = model.predict(X_vis_cubic)
plt.plot(X_vis, y_vis, '--y', label='cubic')

plt.title('Perbandingan Diameter dan Harga Pizza')
plt.xlabel('Diameter (inch)')
plt.ylabel('Harga (dollar)')
plt.legend()
```

```
plt.xlim(0, 25)
plt.ylim(0, 25)
plt.grid(True)
plt.show()

print('Hanif Ridal Warits - 41155050210060')
```




```
Hanif Ridal Warits - 41155050210060
```

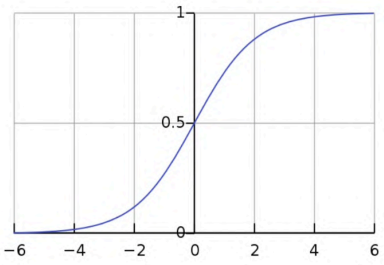
3.0. Lakukan praktek dari

https://youtu.be/oe7DW4rSH1o?si=H-PZJ9rs9-Kab-Ln dan buat screen shot hasil run dengan nama anda pada hasil run tersebut. Praktek tersebut yaitu:

3.1. Formula dasar pembentuk Logistic Regression | Fungsi Sigmoid

Logistic Regression

- $g(X) = sigmoid(\alpha + \beta X)$
- $sigmoid(x) = \frac{1}{1+exp(-x)}$



3.2. Persiapan dataset | SMS Spam Collection Dataset

[1]:
```
print('Hanif Ridal Warits - 41155050210060')
```
```
Hanif Ridal Warits - 41155050210060
```
```
LOGISTIK REGRESSION pada BINARY CLASSIFICATION TASK
```
```
Dataset : SMS Spam Collection Data Set
```

[5]:
```
import pandas as pd

df = pd.read_csv('./dataset/SMSSpamCollection',
                 sep='\t',
                 header=None,
                 names=['label', 'sms'])
df.head()
```

[5]:

	label	sms
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

[7]:
```
print('Hanif Ridal Warits - 41155050210060\n')
df['label'].value_counts()
```
```
Hanif Ridal Warits - 41155050210060
```

[7]:
```
label
ham     4825
spam     747
Name: count, dtype: int64
```

3.3. Pembagian training dan testing set

Training & Testing Dataset

[10]:
```
from sklearn.preprocessing import LabelBinarizer

X = df['sms'].values
y = df['label'].values

lb = LabelBinarizer()
y= lb.fit_transform(y).ravel() #ravel untuk mengembalikan array dari array 2 dimensi
lb.classes_
```

[10]:
```
array(['ham', 'spam'], dtype='<U4')
```

[11]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split (X,
                                                     y,
                                                     test_size = 0.25,
                                                     random_state=0)

print (X_train, '\n')
print (y_train)
```

```
['Its going good...no problem..but still need little experience to understand american customer voice...'
 'U have a secret admirer. REVEAL who thinks U R So special. Call 09065174042. To opt out Reply REVEAL STOP. 1.50 per msg recd. Cust care 07821230901'
 'Ok...' ...
 "For ur chance to win a £250 cash every wk TXT: ACTION to 80608. T's&C's www.movietrivia.tv custcare 08712405022, 1x150p/wk"
 'R U &SAM P IN EACHOTHER. IF WE MEET WE CAN GO 2 MY HOUSE'
 'Mm feeling sleepy. today itself i shall get that dear']

[0 1 0 ... 1 0 0]
```

3.4. Feature extraction dengan TF-IDF

```
Features Extraction dengan TF-IDF
#untuk extract dari text
```

[12]:
```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer(stop_words='english')

X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)

print(X_train_tfidf)
```

```
<Compressed Sparse Row sparse matrix of dtype 'float64'
        with 32656 stored elements and shape (4179, 7287)>
  Coords        Values
  (0, 2997)     0.23173982975834367
  (0, 3007)     0.21421364306658514
  (0, 5123)     0.308974289326673
  (0, 4453)     0.2297719954323795
  (0, 3926)     0.3126721340000456
  (0, 2554)     0.3825278811525034
  (0, 6739)     0.3546359942830148
  (0, 900)      0.4114867709157148
  (0, 2006)     0.2898082580285881
  (0, 6903)     0.3591386422223876
  (1, 5642)     0.24344998442301355
  (1, 799)      0.25048918791028574
  (1, 5441)     0.5009783758205715
  (1, 6472)     0.24039776602646504
  (1, 6013)     0.20089911182610476
  (1, 216)      0.28902673040368515
  (1, 4677)     0.24039776602646504
  (1, 5394)     0.16464655071448758
  (1, 6131)     0.16142609035094446
  (1, 532)      0.20186022353306565
  (1, 4358)     0.17341410292348694
  (1, 5301)     0.2711077935907125
  (1, 2003)     0.2711077935907125
  (1, 1548)     0.18167737976542422
  (1, 36)       0.28902673040368515
  :     :
  (4176, 6792)  0.1407604617250961
  (4176, 6693)  0.16491299289150899
  (4176, 6684)  0.22114159453800114
  (4176, 7083)  0.19523751585154273
  (4176, 1569)  0.18895085073406012
  (4176, 7195)  0.17892283441772988
  (4176, 779)   0.2811068572055718
  (4176, 1612)  0.21138425595332702
  (4176, 365)   0.2388005587702937
  (4176, 7114)  0.4512018097459442
  (4176, 637)   0.29968668460649284
  (4176, 4350)  0.29968668460649284
  (4176, 2004)  0.25589560236817055
  (4176, 107)   0.29968668460649284
  (4176, 343)   0.2811068572055718
  (4177, 3319)  0.43046342221720785
  (4177, 4177)  0.3636187667918345
  (4177, 5565)  0.5506066649743346
  (4177, 2362)  0.6158854885899457
  (4178, 2068)  0.3055766821331892
  (4178, 2641)  0.3993042639531407
  (4178, 6555)  0.2897850627168302
  (4178, 5720)  0.3963527249882828
  (4178, 4279)  0.4530624713751054
  (4178, 5883)  0.548491137555895
```

3.5. Binary Classification dengan Logistic Regression

```
Binary Classification dengan Logistic Regression
```

[13]:
```
from sklearn.linear_model import LogisticRegression

model = LogisticRegression()
model.fit(X_train_tfidf, y_train)
y_pred = model.predict(X_test_tfidf)

for pred, sms in zip(y_pred[:5], X_test[:5]):
    print (f'PRED: {pred} - SMS: {sms}\n')
```

```
PRED: 0 - SMS: Storming msg: Wen u lift d phne, u say "HELLO" Do u knw wt is d real meaning of HELLO?? . . . It's d name of a girl..! . . . Yes.. And u k
nw who is dat girl?? "Margaret Hello" She is d girlfrnd f Grahmbell who invnted telphone... . . . . Moral:One can 4get d name of a person, bt not his gir
lfrnd... G o o d n i g h t . . .@

PRED: 0 - SMS: <Forwarded from 448712404000>Please CALL 08712404000 immediately as there is an urgent message waiting for you.

PRED: 0 - SMS: And also I've sorta blown him off a couple times recently so id rather not text him out of the blue looking for weed

PRED: 0 - SMS: Sir Goodmorning, Once free call me.

PRED: 0 - SMS: All will come alive.better correct any good looking figure there itself..
```

3.6. Evaluation Metrics pada Binary Classification Task

Evaluation Metrics pada Binary Classification

- Confusion Matrix
- Accuracy
- Precission & Recall
- F1 Score
- ROC

3.7. Pengenalan Confusion Matrix

```
CONFUSION MATRIX
sering kali dikenal dengan error matrix
```

[14]:
```
from sklearn.metrics import confusion_matrix

matrix = confusion_matrix(y_test, y_pred)
matrix
```

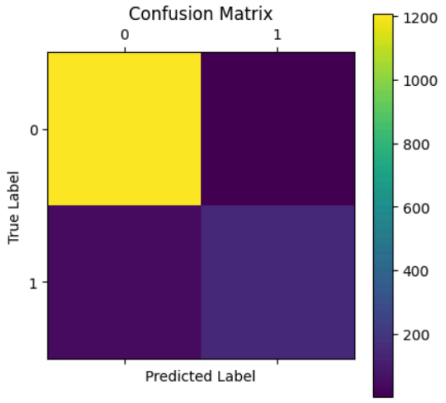
[14]:
```
array([[1207,    1],
       [  47,  138]])
```

[15]:
```
tn, fp, fn, tp = matrix.ravel()


print('Hanif Ridal Warits - 41155050210060\n')
print(f'TN: {tn}')
print(f'FP: {fp}')
print(f'FN: {fn}')
print(f'TP: {tp}')
```

```
Hanif Ridal Warits - 41155050210060

TN: 1207
FP: 1
FN: 47
TP: 138
```

[16]:
```
import matplotlib.pyplot as plt

plt.matshow(matrix)
plt.colorbar()

plt.title('Confusion Matrix')
plt.ylabel('True Label')
plt.xlabel('Predicted Label')
```

```
plt.show()

print('Hanif Ridal Warits - 41155050210060\n')
```




```
Hanif Ridal Warits - 41155050210060
```

3.8. Pengenalan Accuracy Score

```
ACCURACY
mengukur porsi dari prediksi yang tepat
```

```
[17]: from sklearn.metrics import accuracy_score

      accuracy_score(y_test, y_pred)
```

```
[17]: 0.9655419956927495
```

3.9. Pengenalan Precision dan Recall

```
PRECISSION & RECALL
```

```
[18]: from sklearn.metrics import precision_score

      precision_score(y_test, y_pred)
```

```
[18]: np.float64(0.9928057553956835)
```

```
[22]: from sklearn.metrics import recall_score

      recall_score(y_test, y_pred)
```

```
[22]: np.float64(0.745945945945946)
```

3.10. Pengenalan F1 Score | F1 Measure

```
F1-SCORE
atau F1-measure adalah harmonic mean dari precission dan recall
```

```
[23]: from sklearn.metrics import f1_score

      f1_score(y_test, y_pred)
```

```
[23]: np.float64(0.8518518518518519)
```

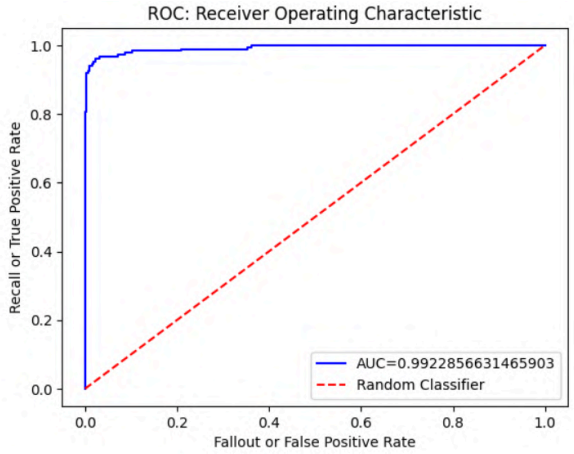
3.11. Pengenalan ROC | Receiver Operating Characteristic

[24]:
```
print('Hanif Ridal Warits - 41155050210060\n')
```

```
Hanif Ridal Warits - 41155050210060
```

ROC : RECEIVER OPERATING CHARACTERISTIC
menawarkan visualisasi terhadap performa dari classifier dengan membandingkan nilai recall(TPR) dengan nilai Fallout(FPR)

[25]:
```
from sklearn.metrics import roc_curve, auc

prob_estimates = model.predict_proba(X_test_tfidf)
fpr, tpr, threshold = roc_curve(y_test, prob_estimates[:, 1])
nilai_auc = auc(fpr, tpr)

plt.plot(fpr, tpr, 'b', label=f'AUC={nilai_auc}')
plt.plot([0, 1], [0, 1], 'r--', label='Random Classifier')

plt.title('ROC: Receiver Operating Characteristic')
plt.xlabel('Fallout or False Positive Rate')
plt.ylabel('Recall or True Positive Rate')
plt.legend()
plt.show()
```



[26]:
```
print('Hanif Ridal Warits - 41155050210060\n')
```

```
Hanif Ridal Warits - 41155050210060
```

NOTE :

Formula Dasar

Simple Linear Regression

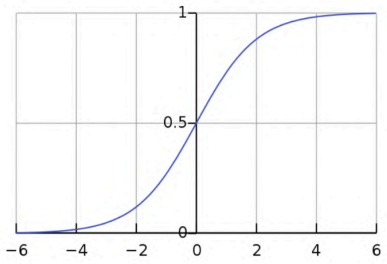
- $y = \alpha + \beta x$
- $g(x) = \alpha + \beta x$

Multiple Linear Regression

- $y = \alpha + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$
- $g(X) = \alpha + \beta X$

Logistic Regression

- $g(X) = sigmoid(\alpha + \beta X)$
- $sigmoid(x) = \frac{1}{1+exp(-x)}$



Evaluation Metrics pada Binary Classification

- Confusion Matrix
- Accuracy
- Precission & Recall
- F1 Score
- ROC

Terminologi Dasar

- True Positive (TP)
- True Negative (TN)
- False Positive (FP)
- False Negative (FN)

Confusion Matrix

Confusion matrix seringkali juga dikenal sebagai error matrix.

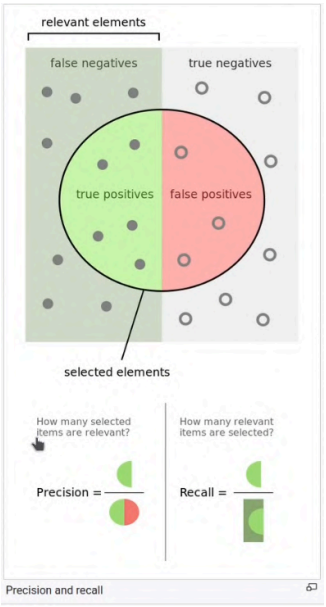
1. Referensi: https://en.wikipedia.org/wiki/Confusion_matrix

Accuracy

Accuracy mengukur porsi dari hasil prediksi yang tepat.

$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} = \frac{correct}{total}$

2. Referensi: https://en.wikipedia.org/wiki/Accuracy_and_precision



Precision and recall

Precission & Recall

Selain menggunakan accuracy, performa dari suatu classifier umumnya juga diukur berdasarkan nilai Precission dan Recall.

Referensi: https://en.wikipedia.org/wiki/Precision_and_recall

3.

Precission or Positive Predictive Value (PPV)

$$Precission = \frac{TP}{TP+FP}$$

Referensi: https://en.wikipedia.org/wiki/Positive_and_negative_predictive_values

Recall or True Positive Rate (TPR) or Sensitivity

$$Recall = \frac{TP}{TP+FN}$$

Referensi: https://en.wikipedia.org/wiki/Sensitivity_and_specificity

4.

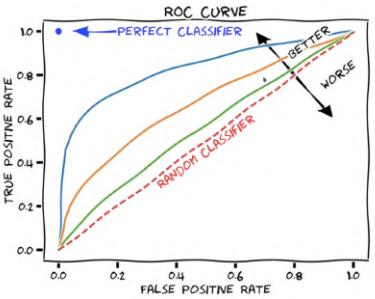
F1-Score

F1-score atau F1-measure adalah harmonic mean dari precission dan recall.

$$F1\ score = \frac{precission \times recall}{precission + recall}$$

Referensi: https://en.wikipedia.org/wiki/F-score

5.



ROC: Receiver Operating Characteristic

ROC menawarkan visualisasi terhadap performa dari classifier dengan membandingkan nilai Recall (TPR) dan nilai Fallout (FPR)

$$fallout = \frac{FP}{TN+FP}$$

Referensi: https://en.wikipedia.org/wiki/Receiver_operating_characteristic