Lab 2 Logistic Regression

No.	program

1	`# Import tools`
	`import pandas as pd`
	`import numpy as np`
	`import seaborn as sns`
	`import matplotlib.pyplot as plt`
	`import math`

2	`# load data, pastikan membaca note`
	`data = pd.read_csv('titanic.csv')`
	`data.head(10)`

3	`# Analisa data`
	`# countplot`
	`sns.countplot(x="Survived", data=data)`
	`# 1 = selamat`

4	`sns.countplot(x="Survived", hue="Sex", data=data)` `# selamat banyak female`
5	`sns.countplot(x="Survived", hue="Pclass", data=data)` `# class 3 banyak tidak selamat`
6	`data['Age'].plot.hist()` `# banyak yang berumur 20 - 30`
7	`data['Fare'].plot.hist(bins=20, figsize=(10,5))` `# tarif (dolar)`

8	`# sibling/spouse`
	`sns.countplot(x='SibSp', data=data)`
	`# banyak yang tidak`

9	`# Data Wrangling/Cleaning Data`
	`# cek data kosong`
	`data.isnull()`

10	`data.isnull().sum()`
11	`# plot dengan heatmap` `sns.heatmap(data.isnull(), yticklabels=False, cmap='viridis')`
12	`# box plot` `sns.boxplot(x='Pclass',y='Age', data=data)`

13	`#imputation`
	`# cek data`
	`data.head()`

14	`# drop Cabin karena banyak yg kosong` `data.drop('Cabin', axis=1, inplace=True)`

15	`# fungsi untuk imputasi fitur Age`
	`def impute_age(cols):`
	`  Age=cols[0]`
	`  Pclass=cols[1]`
	`  if(pd.isnull(Age)):`
	`  if(Pclass==1):`
	`  return 37`
	`  elif(Pclass==2):`
	`  return 29`
	`  else:`
	`  return 24`
	`  else:`
	`  return Age`

16	`# jalankan fungsi` `data['Age'] = data[['Age','Pclass']].apply(impute_age, axis=1)`
17	`#cek lagi` `sns.heatmap(data.isnull(), yticklabels=False, cbar=False)`

18	`# value masih aneh, dikonversi ke kategorikal supaya bisa diproses`
	`# logistik regression`
	`data.head(2)`
	`# menggunakan strategi one hot encoding`

19	`# pemecahan fitur` `pd.get_dummies(data['Sex'])`

20	`# pemecahan fitur, ambil male saja`
	`jenkel = pd.get_dummies(data['Sex'], drop_first=True)`
	`jenkel.head()`

21	`# pecah embark`
	`emb = pd.get_dummies(data['Embarked'])`
	`emb.head()`

22	`# pecah embark, hilangkan C`

	emb = pd.get_dummies(data['Embarked'], drop_first=True) emb.head()

23	# Pcl perlu dipisah dan dihilangkan class 1
	Pcl = pd.get_dummies(data['Pclass'], drop_first=True)
	Pcl.head()

24	# gabungkan semuanya
	data = pd.concat([data, jenkel, emb, Pcl], axis=1)
	data.head()

25	# drop Pclass, sex, dan embarked,, selain hapus PassengerId, Name, ticket # karena tidak diolah data.drop(['Sex','Embarked','Pclass', 'PassengerId','Name','Ticket'], axis=1, inplace=True) data.head()

26	# Train Data
	# tentukan dependen dan independen var
	X= data.drop('Survived', axis=1)
	y= data['Survived']

27	# library untuk split data from sklearn.model_selection import train_test_split
28	# split data ke beberapa bagian X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1)
29	# buat model from sklearn.linear_model import LogisticRegression

30	# buat model
	logModel = LogisticRegression()

31	# latih model
	logModel.fit(X_train, y_train)

32	# gunakan untuk memprediksi
	prediksi = logModel.predict(X_test)

33	# confussion_matrix from sklearn.metrics import confusion_matrix

34	`# evaluasi model` `confusion_matrix(y_test, prediksi)`

35	`# accuracy score`
	`from sklearn.metrics import accuracy_score`
	`accuracy_score(y_test, prediksi)`