Klasifikasi Dengan Algoritma Random Forest Untuk Pengembagan Sistem Prediksi Kelulusan

Kelompok 1 Data Mining A

IF UIN SGD

Intro

Goals : dibuatnya model klasifikasi dari dataset yang teresdia dengan algoritma yang sesuai untuk dijadikan acuan dalam proses prediksi dari input yang diberikan user.

Preview cara kerja sistem

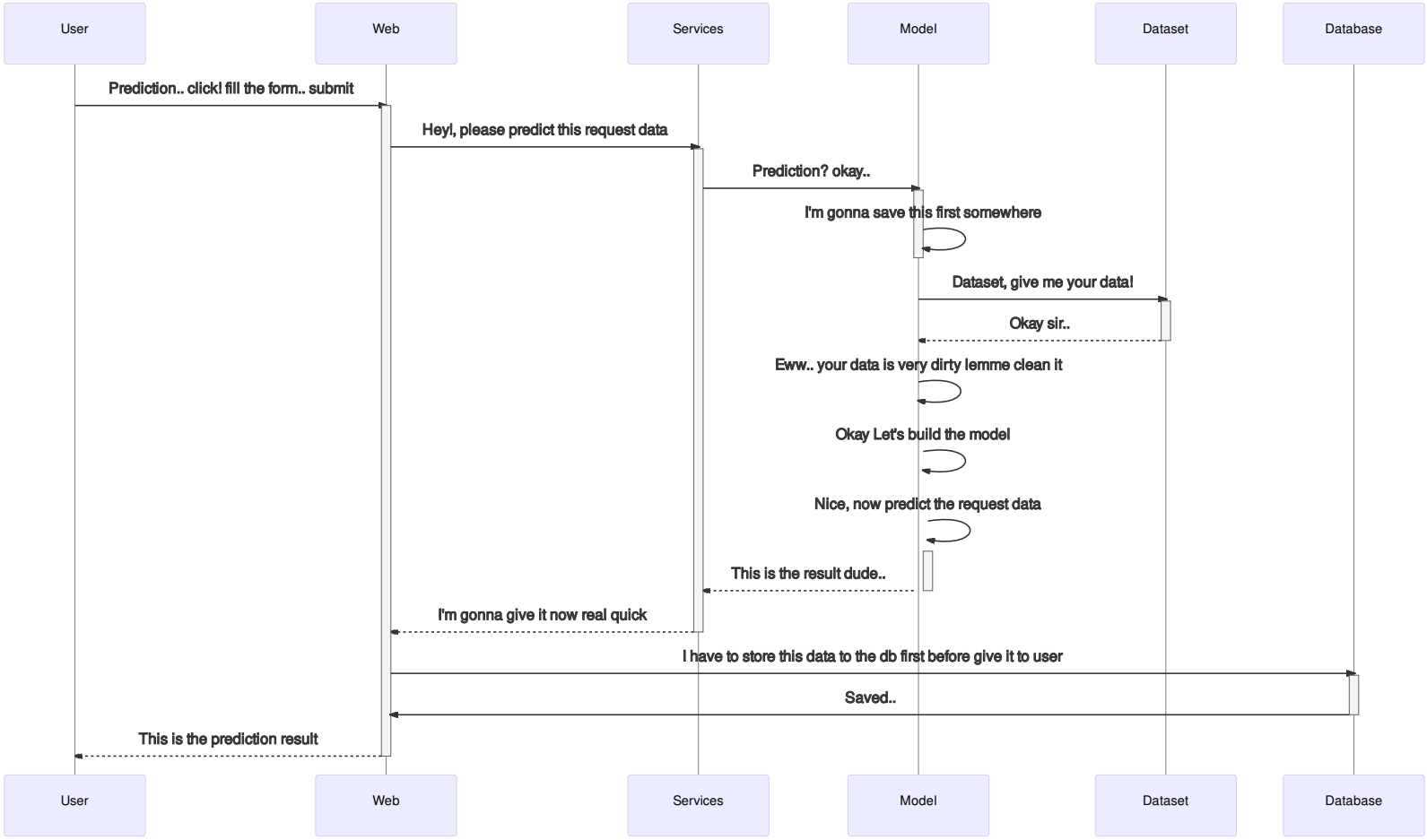

Figure 1: sequence

Proses Pengembangan

1. analisis data
2. penentuan algoritma
3. pembuatan model
4. pengujian model
5. pembuatan sistem

Analisis Data

attribut

nama, jenis_ kelamin, status mahasiswa, umur, status nikah, ip semester 1-8, ipk, status kelulusan.

jenis-jenis tipe data yang disajikan

polynomial, integer, float

attribut yang dibutuhkan

- training : jenis kelamin, status mahasiswa, umur, status nikah, ip semester 1-8, ipk, status kelulusan
- scoring : jenis kelamin, status mahasiswa, umur, status nikah, ip semester 1-8, ipk

label

status kelulusan

Penentuan Algoritma

untuk penentuan dilakukan dengan cara menguji beberapa algoritma klasifikasi dengan dataset yang digunakan sebagai parameter pengujian. kemudian dari nilai hasil pengujiannya kita pilih algoritma dengan nilai performa terbaik.

oleh karena itu sebelum kita menguji dan menentukan algoritma. harus mengolah datanya terlebih dahulu sebelum data tersebut dapat diuji.

Pembuatan Model

Persiapan Data

preview data sebelum diolah

pemuatan dan penyeleksian data

```
import pandas
uri = '../dataset/datakelulusanmahasiswa.csv'
# load dataset
df = pandas.read_csv(uri)
# hapus  tiap row yang isi column nya tidak bernilai
df.dropna(0, inplace=True)
# hapus column yang tidak termasuk perhitungan
df = df.drop(['NAMA'], axis=1)
df
```

Pembuatan Model

hasil

	JENIS KELAMIN	STATUS MAHASISWA	UMUR	STATUS NIKAH	IPS 1	IPS 2	IPS 3	IPS 4	IPS 5	IPS 6	IPS 7	IPS 8	IPK	STATUS KELULUSAN
0	LAKI - LAKI	MAHASISWA	29	MENIKAH	2.83	3.61	3.18	3.23	3.02	3.20	3.18	0.00	2.78125	TEPAT
1	LAKI - LAKI	MAHASISWA	23	BELUM MENIKAH	2.29	2.89	2.48	3.14	2.90	2.40	2.73	0.00	2.35375	TEPAT
2	LAKI - LAKI	MAHASISWA	28	BELUM MENIKAH	2.57	2.55	2.57	1.91	2.33	2.45	2.19	2.55	2.39000	TERLAMBAT
3	LAKI - LAKI	BEKERJA	28	BELUM MENIKAH	3.26	3.73	3.02	3.50	2.69	3.33	3.09	3.50	3.26500	TEPAT
4	LAKI - LAKI	MAHASISWA	23	BELUM MENIKAH	2.55	2.75	2.64	2.59	2.80	2.75	1.14	0.00	2.15250	TERLAMBAT
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
375	LAKI - LAKI	MAHASISWA	26	BELUM MENIKAH	2.76	2.86	2.57	2.91	2.84	3.08	3.35	0.00	2.54625	TEPAT
376	LAKI - LAKI	MAHASISWA	25	BELUM MENIKAH	2.57	3.02	3.13	3.25	3.29	3.58	1.91	4.00	3.09375	TEPAT
377	PEREMPUAN	MAHASISWA	26	BELUM MENIKAH	3.50	3.23	3.83	3.39	3.46	3.79	2.33	0.00	2.94125	TEPAT
378	PEREMPUAN	MAHASISWA	23	BELUM MENIKAH	2.93	2.89	2.70	2.77	2.75	3.00	2.93	2.50	2.80875	TEPAT
379	PEREMPUAN	MAHASISWA	23	BELUM MENIKAH	2.71	2.75	2.43	3.14	2.73	2.85	2.90	1.80	2.66375	TEPAT

373 rows × 14 columns

Figure 2: hasil pemuatan dan seleksi data

Pembuatan Model

transformasi data

tujuan : mengubah tipe data polynominal menjadi bentuk numerik

```
from sklearn.preprocessing import LabelEncoder
# list attribute yang akan diubah
attributes = [
    'JENIS KELAMIN',
    'STATUS MAHASISWA'
    'STATUS NIKAH',
    'STATUS KELULUSAN',
]
# ulang untuk setiap attribute pada list
# ubah menjadi numerik
for attr in attributes :
    df[attr] = LabelEncoder().fit_transform(df[attr])
```

Pembuatan Model

hasil

	JENIS KELAMIN	STATUS MAHASISWA	UMUR	STATUS NIKAH	IPS 1	IPS 2	IPS 3	IPS 4	IPS 5	IPS 6	IPS 7	IPS 8	IPK	STATUS KELULUSAN
0	0	1	29	1	2.83	3.61	3.18	3.23	3.02	3.20	3.18	0.00	2.78125	0
1	0	1	23	0	2.29	2.89	2.48	3.14	2.90	2.40	2.73	0.00	2.35375	0
2	0	1	28	0	2.57	2.55	2.57	1.91	2.33	2.45	2.19	2.55	2.39000	1
3	0	0	28	0	3.26	3.73	3.02	3.50	2.69	3.33	3.09	3.50	3.26500	0
4	0	1	23	0	2.55	2.75	2.64	2.59	2.80	2.75	1.14	0.00	2.15250	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
375	0	1	26	0	2.76	2.86	2.57	2.91	2.84	3.08	3.35	0.00	2.54625	0
376	0	1	25	0	2.57	3.02	3.13	3.25	3.29	3.58	1.91	4.00	3.09375	0
377	1	1	26	0	3.50	3.23	3.83	3.39	3.46	3.79	2.33	0.00	2.94125	0
378	1	1	23	0	2.93	2.89	2.70	2.77	2.75	3.00	2.93	2.50	2.80875	0
379	1	1	23	0	2.71	2.75	2.43	3.14	2.73	2.85	2.90	1.80	2.66375	0

373 rows × 14 columns

Figure 3: hasil transformasi data

Pembuatan Model

keterangan

	0	1
jenis kelamin	laki-laki	perempuan
status mahasiswa	bekerja	mahasiswa
status nikah	belum menikah	menikah
status kelulusan	tepat	terlambat

Table 1: keterangan hasil transformasi data

Penentuan Algoritma

memisahkan data train dan data test

```
X = df.drop('STATUS KELULUSAN', axis='columns')
Y = df['STATUS KELULUSAN']
X_train, X_test, Y_train, Y_test = train_test_split(X,Y)
```

next step

1. Import algoritma yang memungkinkan untuk digunakan
2. Menguji cross validation tiap algoritma
3. Pilih beberapa hasil terbaik lalu uji akurasinya
4. Pilih algoritma dengan nilai akurasi terbaik

Penentuan Algoritma

1. Mendefinisikan algoritma-algoritma yang akan diuji

```
from sklearn import tree
from sklearn.svm import SVC
from sklearn.naive_bayes import GaussianNB
from sklearn.discriminant_analysis import LDA
from sklearn.linear_model import SGDClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
```

Penentuan Algoritma

```
mdls = []

mdls.append(('SVM', SVM()))
mdls.append(('NVB', NVB()))
mdls.append(('DTR', DTR()))
mdls.append(('SGD', SGD()))
mdls.append(('KNN', KNN()))
mdls.append(('LDA', LDA()))
mdls.append(('LGR', LGR(max_iter=140)))
mdls.append(('RFR', RFR(n_estimators=50, max_features=13)))
```

Penentuan Algoritma

2. Menguji cross validation tiap algoritma

```
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score

results = []
names = []

for name, model in models:
 kfold = KFold()
 cv = cross_val_score(model, X_train, Y_train, cv=kfold)
 results.append(cv)
 names.append(name)
 msg = "%s: %f" % (name,cv.mean())
 print(msg)
```

Penentuan Algoritma

Algoritma	CV
SVM	0.777646
NVB	0.910317
DTR	0.849603
SGD	0.802910
KNN	0.878042
LDA	0.860317
LGR	0.921032
RFR	0.924603

Table 2: cross validation score

Penentuan Algoritma

```
from sklearn.model_selection import train_test_split

model = modelClassifier()
model.fit(X_train, Y_train)
predictions = model.predict(X_test)
print(classification_report(Y_test, predictions))
```

Penentuan Algoritma

classification report

```
model = modelClassifier()
model.fit(X_train, Y_train)
predictions = model.predict(X_test)
print(classification_report(Y_test, predictions))
```

Penentuan Algoritma

Naive Bayes

0	0.95	0.80	0.87	51
1	0.80	0.95	0.87	43
accuracy			0.87	94
macro avg	0.88	0.88	0.87	94
weighted avg	0.89	0.87	0.87	94

Penentuan Algoritma

Logistic Regression

	precision	recall	f1-score	support
0	0.89	0.78	0.83	51
1	0.78	0.88	0.83	43
accuracy			0.83	94
macro avg	0.83	0.83	0.83	94
weighted avg	0.84	0.83	0.83	94

Penentuan Algoritma

Random Forest

	precision	recall	f1-score	support
0	0.96	0.86	0.91	51
1	0.85	0.95	0.90	43
accuracy			0.90	94
macro avg	0.91	0.91	0.90	94
weighted avg	0.91	0.90	0.90	94

Penentuan Algoritma

Algoritma yang dipilih

Random Forest

Sekilas Random Forest

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean/average prediction (regression) of the individual trees. Random decision forests correct for decision trees habit of overfitting to their training set. Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance.

Membuat Fungsi Prediksi

```
def prediksi(model, data):
    labels = ["TEPAT","TERLAMBAT"]
    array = numpy.asarray(data)
    prediction=model.predict(array)
    no_of_test_cases, cols = new_array.shape

    for i in range(no_of_test_cases):
        print("Status Mahasiswa dengan IPK grade = {},
        Diprediksi Lulus {}"
        .format(new_data[i][12],
        labels[int(prediction[i])]))
```

Tes Fungsi Prediksi

data

```
new_data = [
    [0,0,25,0,2.5,2.5,3,3,2.7,2.8,2.9,3,3],
    [1,1,25,1,2.5,2.5,3,3,2.7,2.8,2.9,3,3],
    [0,1,25,0,2.5,2.5,3,3,2.7,2.8,2.9,3,3],
    [0,0,25,1,2.5,2.5,3,3,2.7,2.8,2.9,3,3],
    [1,0,25,0,2.5,2.5,3,3,2.7,2.8,2.9,3,3],
    [1,1,25,0,2.5,2.5,3,3,2.7,2.8,2.9,3,3],
    [0,1,25,1,2.5,2.5,3,3,2.7,2.8,2.9,3,3],
    [1,1,25,1,4,4,4,4,4,4,4,4,4],
]

prediksi(model, new_data)
```

Tes Fungsi Prediksi

hasil

```
Mahasiswa Diprediksi Lulus TERLAMBAT
Mahasiswa Diprediksi Lulus TEPAT
Mahasiswa Diprediksi Lulus TEPAT
Mahasiswa Diprediksi Lulus TERLAMBAT
Mahasiswa Diprediksi Lulus TERLAMBAT
Mahasiswa Diprediksi Lulus TEPAT
Mahasiswa Diprediksi Lulus TEPAT
Mahasiswa Diprediksi Lulus TEPAT
```

Proses Pembuatan Sistem

Alur

1. Mengkonversi model yang dibuat kedalam fungsi yang dapat diakses oleh services
2. Membuat Services
3. Membuat Client

Tools yang digunakan

1. FastAPI -> Services
2. Laravel -> Client
3. MySQL -> Data Storage

Demo

Terimakasih