Prediksi Hujan di Denpasar

Praktikum ini menggunakan *dataset* Denpasar Weather Data dengan modifikasi. *Dataset* digunakan untuk melakukan prediksi penarikan kesimpulan kebenaran kondisi hujan pada kondisi tertentu. Hal itu diperoleh dengan meninjau `raining` (diekstrak dari `weather_main`) sebagai target. Fitur yang digunakan adalah sebagai berikut:

- `hour` (diekstrak dari `dt_iso`)
- `temp`
- `temp_min`
- `temp_max`
- `pressure`
- `humidity`
- `wind_speed`
- `wind_deg`

Tujuan praktikum:

1. Peserta memahami rangkaian proses analitik data menggunakan pendekatan pembelajaran mesin.
2. Peserta memahami bahwa proses pengembangan model pembelajaran mesin juga ditentukan dari kualitas data, penanganan data, dan penentuan algoritma serta hiperparameternya; tidak cukup hanya dengan memastikan implementasi algoritma berjalan tanpa kesalahan.
3. Peserta mampu menginterpretasikan hasil dari evaluasi model dalam proses analitik menggunakan pendekatan pembelajaran mesin.

Praktikum dilaksanakan secara berkelompok. Setiap kelompok terdiri atas 2 mahasiswa. Perhatikan bahwa terdapat berkas yang harus dikumpulkan sebelum waktu praktikum selesai (17 April 2023, pukul 10.59 WIB) dan berkas yang dikumpulkan setelah waktu praktikum selesai (17 April 2023, pukul 23.59 WIB).

Persiapan Data

```
import pandas as pd
import numpy as np
import seaborn as sb
import matplotlib.pyplot as plt
import missingno as msno
from sklearn.model_selection import train_test_split, GridSearchCV,
cross_val_score
from sklearn.preprocessing import LabelEncoder, MinMaxScaler
from sklearn.impute import KNNImputer
from sklearn.linear_model import LogisticRegression
from sklearn.svm import LinearSVC
```

```
from sklearn.ensemble import VotingClassifier, StackingClassifier
from sklearn.feature_selection import SelectFromModel
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report, precision_score, recall_score, f1_score
from sklearn.calibration import CalibratedClassifierCV

data = pd.read_csv("openweatherdata-denpasar-1990-2020v0.1-
simplified.csv")
data
```

```
        hour   temp  temp_min  temp_max  pressure  humidity
wind_speed  \
0          0  25.82     25.82     25.82    1010.0        86
1.36
1          1  26.20     26.20     26.20    1011.0        84
2.09
2          2  26.45     26.45     26.45    1011.0        84
2.44
3          3  26.80     26.80     26.80    1011.0        82
2.29
4          4  27.04     27.04     27.04    1010.0        82
1.71
...      ...    ...       ...       ...       ...       ...         ..
.
264919    19  27.00     27.00     27.00    1007.0        94
4.10
264920    20  27.00     27.00     27.00    1007.0        94
5.70
264921    21  28.00     28.00     28.00    1007.0        83
6.70
264922    22  28.00     28.00     28.00    1007.0        83
6.20
264923    23  28.00     28.00     28.00    1008.0        83
5.70

        wind_deg  raining
0            225     True
1            247     True
2            262     True
3            271     True
4            274    False
...          ...      ...
264919       300    False
264920       300    False
264921       290    False
264922       290    False
264923       300    False

[264924 rows x 9 columns]
```

```
X = data.drop(columns="raining")
y = data["raining"].copy()

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=123)
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train,
test_size=0.2, random_state=123)

df_train = pd.concat([X_train, y_train], axis=1)
df_val = pd.concat([X_val, y_val], axis=1)
df_test = pd.concat([X_test, y_test], axis=1)
```

Soal

Disediakan data yang sudah dibagi menjadi data latih (`df_train`), data validasi (`df_val`), dan data uji (`df_test`).

Bagian 1: (batas waktu: 17 April 2023, 10.59 WIB)

1. Buatlah *baseline* dengan menggunakan model *logistic regression*.
2. Lakukan analisis data terkait hal berikut:
   - *duplicate value*,
   - *missing value*,
   - *outlier*,
   - *balance of data*.
3. Jelaskan rencana penanganan yang ada pada poin 2.
4. Jelaskan teknik *encoding* yang digunakan terhadap data yang disediakan, disertai dengan alasan.
5. Buatlah desain eksperimen dengan menentukan hal berikut:
   - tujuan eksperimen,
   - variabel dependen dan independen,
   - strategi eksperimen,
   - skema validasi.

Bagian 2: (batas waktu: 17 April 2023, 23.59 WIB)

1. Implementasikan strategi eksperimen dan skema validasi yang telah ditentukan pada poin 5.
2. Berdasarkan hasil prediksi yang dihasilkan, buatlah kesimpulan analisis karakteristik kondisi hujan.

Jika terdapat perubahan jawaban pada poin 1—5 (contoh: perbedaan penanganan *outlier*), jelaskan pada laporan mengenai jawaban sebelum, jawaban sesudah, dan alasan pengubahan jawaban.

Pada sel ini, jelaskan pembagian tugas/kerja per anggota kelompok dalam eksperimen.

Deliverable

Deliverable yang akan dihasilkan adalah sebagai berikut:

1. berkas *notebook* dengan format nama `PraktikumIF3270_M1_NIM1_NIM2.ipynb` untuk Bagian 1;
2. berkas *notebook* dengan format nama `PraktikumIF3270_M2_NIM1_NIM2.ipynb` untuk Bagian 1 + Bagian 2; serta
3. berkas laporan dengan format nama `PraktikumIF3270_NIM1_NIM2.pdf` yang mencakup hal berikut:
   - hasil analisis data,
   - penanganan dari hasil analisis data,
   - justifikasi teknik-teknik yang dipilih,
   - perubahan yang dilakukan pada jawaban poin 1—5 jika ada,
   - desain eksperimen,
   - hasil eksperimen.
   - analisis dari hasil eksperimen,
   - kesimpulan,
   - pembagian tugas/kerja per anggota kelompok

Batas waktu pengumpulan:

- *Deliverable* poin 1: Senin, 17 April 2023, pukul 10.59 WIB
- *Deliverable* poin 2: Senin, 17 April 2023, pukul 23.59 WIB
- *Deliverable* poin 3: Senin, 17 April 2023, pukul 23.59 WIB

Bagian 1

```
def updateDataWithNewChange(df):
  features = list(df.columns)
  features.remove('raining')

  numericFeaturesDf = df.select_dtypes(include=['float64', 'int64'])
  numericFeatures = list(numericFeaturesDf.columns)

  categoricalFeaturesDf = df.select_dtypes(include=['bool'])
  categoricalFeaturesDf =
categoricalFeaturesDf.drop(columns=['raining'])
  categoricalFeatures = list(categoricalFeaturesDf.columns)

  return X, y, features, numericFeaturesDf, numericFeatures,
categoricalFeaturesDf, categoricalFeatures

X, y, features, numericFeaturesDf, numericFeatures,
```

```
categoricalFeaturesDf, categoricalFeatures =
updateDataWithNewChange(data)
```

1.1 Buatlah *baseline* dengan menggunakan model *logistic regression*.

```
logreg = LogisticRegression(solver='liblinear', random_state=0)
logreg.fit(X_train, y_train)
y_test_predict_log = logreg.predict(X_test)
y_train_predict_log = logreg.predict(X_train)
```

```
def displayEvaluationMetric(y_test, y_test_predict, name_of_set: str,
name_of_model: str):
  accuracyTestSet = accuracy_score(y_test, y_test_predict)
  precisionTestSet = precision_score(y_test, y_test_predict)
  recallTestSet = recall_score(y_test, y_test_predict)
  f1TestSet = f1_score(y_test, y_test_predict)
  print(f"=== Evaluation metric pada {name_of_set} set dengan
{name_of_model} ===")
  print(f"Skor akurasi pada test set: {round(accuracyTestSet, 4)}")
  print(f"Skor precision pada test set: {round(precisionTestSet, 4)}")
  print(f"Skor recall pada test set: {round(recallTestSet, 4)}")
  print(f"Skor f1 pada test set: {round(f1TestSet, 4)}\n")

displayEvaluationMetric(y_test, y_test_predict_log, "test", "logistic
regression")
```

```
=== Evaluation metric pada test set dengan logistic regression ===
Skor akurasi pada test set: 0.8728
Skor precision pada test set: 0.5819
Skor recall pada test set: 0.128
Skor f1 pada test set: 0.2099
```

1.2 Lakukan analisis data terkait duplicate/missing value, outlier, dan balance data

```
# Duplicate value
print(data.duplicated().value_counts())
# Jumlah duplicate value = 7253
```

```
False    257671
True       7253
dtype: int64
```

```
# Missing value
numOfRows = data.shape[0]

listOfCountMissingVal = data.isna().sum().values
for i, col in enumerate(data):
  numOfMissingValue = listOfCountMissingVal[i]
  proportionOfMissingValue = round(numOfMissingValue / numOfRows *
```

```
100, 2)
  print(f"{col}: {numOfMissingValue} ({proportionOfMissingValue} %)")
```

```
hour: 0 (0.0 %)
temp: 0 (0.0 %)
temp_min: 0 (0.0 %)
temp_max: 0 (0.0 %)
pressure: 0 (0.0 %)
humidity: 0 (0.0 %)
wind_speed: 0 (0.0 %)
wind_deg: 0 (0.0 %)
raining: 0 (0.0 %)
```

```
# Outlier
numericFeatNoOutlier = []
for feature in numericFeatures:
  Q1 = X[feature].quantile(0.25)
  Q3 = X[feature].quantile(0.75)
  IQR = round(Q3 - Q1, 2)
  countOutlier = ((X[feature] < (Q1 - 1.5 * IQR)) | (X[feature] > (Q3
+ 1.5 * IQR))).sum()
  proportionOutlier = round(countOutlier / numOfRows * 100, 2)
  print(f"{feature}: {countOutlier} ({proportionOutlier} %)")
  if(countOutlier == 0):
    numericFeatNoOutlier.append(feature)
```

```
hour: 0 (0.0 %)
temp: 1458 (0.55 %)
temp_min: 1716 (0.65 %)
temp_max: 547 (0.21 %)
pressure: 1067 (0.4 %)
humidity: 231 (0.09 %)
wind_speed: 3439 (1.3 %)
wind_deg: 0 (0.0 %)
```

```
# Balance of data
df_true = data[data["raining"] == True]
df_false = data[data["raining"] == False]
print( "Class true = ", len(df_true), "; class false =",
len(df_false))
```

```
Class true =  34901 ; class false = 230023
```

1.3 Rencana penanganan yang ada pada poin 2.

Penanganan duplicate value

Duplicate value ditangani dengan menghapus row yang merupakan duplicate value.

Penangangan missing value

Tidak terdapat missing value sehingga penanganan tidak dilakukan

Penangangan outlier

Tidak dilakukan penghapusan data yang mengandung nilai outlier karena pada kasus ini, jumlah outlier proporsinya kecil dan kami melihat bahwa outlier pada dataset yang ada dapat memberikan insight pada data dan menunjukkan pola tertentu.

Penangangan imbalance dataset

Imbalance dataset ditangani dengan teknik oversampling. Terdapat dua kelas yaitu kelas True dan kelas False. Kelas True merupakan kelas minoritas yang akan diduplikasi sehingga memiliki jumlah yang mirip dengan kelas False. Pemilihan teknik oversampling dari pada undersampling adalah agar informasi pada kelas mayoritas tidak hilang.

1.4 Teknik *encoding* yang digunakan terhadap data yang disediakan, disertai dengan alasan

Encoding dilakukan pada kolom raining, yang merupakan target columns. Teknik encoding yang dilakukan adalah integer encoding, yakni dengan:

- Nilai False akan diencode menjadi nilai 0.
- Nilai True aka diencode menjadi nilai 1.

Hal ini mengikuti nilai boolean dimana False bernilai 0 dan True bernilai 1.

1.5 Desain eksperimen

Tujuan eksperimen

Eksperimen dilakukan untuk mengoptimalkan performa model dengan menggunakan data yang ada. Perfoma yang diinginkan dari eksperimen ini adalah seberapa benarnya prediksi model pada kelas True karena merupakan kelas minoritas dari imbalace dataset. Performa akan dicari adalah nilai f1 dan recall yang besar tanpa mengorbankan precision.

Variabel dependen dan independen

Pada kasus ini, variabel independennya adalah fitur-fitur yang ada, yakni hour, temp, temp_min, temp_max, pressure, humidity, wind_speed, dan wind_deg. Sedangkan variabel dependennya adalah kolom target, yakni raining

Strategi eksperimen

Eksperimen dilakukan *hyperparameter tuning* pada parameter logistic regression. Akan juga dilakukan pembuatan model lain dengan feature selection untuk memilih sebagian fitur yang paling penting dan akan membandingkan metrik evaluasinya dengan model menggunakan keseluruhan fitur

Skema validasi

Validasi dilakukan dengan menggunakan data validasi (df_val). Data validasi akan diprediksi kelasnya dengan model. Hasil prediksi tersebut akan dibandingkan dengan kelas

aslinya. Metrik yang digunakan adalah accuracy, precision, recall, dan f1. Model yang dipilih adalah model yang memiliki nilai recall dan f1 yang baik tanpa mengorbankan precision.

Bagian 2

Soon