Tugas 3 Data Mining dan Kecerdasan Bisnis

Nama:
Almas Fauzia Wibawa (17/409427/PA/17734)
Bryan Dwison (17/409430/PA/17737)
Fajar Abdul Malik (17/409432/PA/17739)
Faqih Ethana Prabandaru (17/409433/PA/17740)

Data

In [1]:

```
import pandas as pd

juice = pd.read_csv('juice.csv')
juice.head()
```

Out[1]:

	Id	Purchase	WeekofPurchase	StoreID	PriceCH	PriceMM	DiscCH	DiscMM	SpecialCH	Sp
0	1	CH	237	1	1.75	1.99	0.00	0.0	0	
1	2	CH	239	1	1.75	1.99	0.00	0.3	0	
2	3	CH	245	1	1.86	2.09	0.17	0.0	0	
3	4	MM	227	1	1.69	1.69	0.00	0.0	0	
4	5	CH	228	7	1.69	1.69	0.00	0.0	0	

In [2]:

```
# memisah x dengan y dan mengonversikan data biner yang masih berupa string ke 0 dan 1

x = juice.drop(['Id', 'Purchase'], axis = 1)
for i in range(len(juice)):
    if juice.Store7[i] == 'Yes':
        x.Store7[i] = 1
    else:
        x.Store7[i] = 0

y = []
for i in range(len(juice)):
    if juice.Purchase[i] == 'CH':
        y.append(0)
    else:
        y.append(1)
x.head()
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:8: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pand
as.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
ersus-a-copy)

/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:6: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pand
as.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
ersus-a-copy)
```

Out[2]:

	WeekofPurchase	StoreID	PriceCH	PriceMM	DiscCH	DiscMM	SpecialCH	SpecialMM	Loya
0	237	1	1.75	1.99	0.00	0.0	0	0	0.500
1	239	1	1.75	1.99	0.00	0.3	0	1	0.600
2	245	1	1.86	2.09	0.17	0.0	0	0	0.680
3	227	1	1.69	1.69	0.00	0.0	0	0	0.400
4	228	7	1.69	1.69	0.00	0.0	0	0	0.956

In [3]:

```
# split data

from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3)
```

1. Model SVM trained using default hyperparameter.

In [4]:

```
# SVM dengan default hyperparameter

from sklearn.model_selection import cross_val_score
from sklearn.svm import SVC

clf1 = SVC()
clf1.fit(x_train, y_train)
clf1_acc = clf1.score(x_test, y_test)
print('Accuracy for SVM with default hyperparameter = ', clf1_acc)
```

```
Accuracy for SVM with default hyperparameter =  0.5950155763239875
```

2. Model SVM trained and tuned using GridSearchCV.

In [5]:

```
# SVM dengan GridSearchCV

from sklearn.model_selection import GridSearchCV

space = dict()
space['C'] = [0.01, 0.1, 1]
space['kernel'] = ['linear', 'poly']
clf2 = GridSearchCV(clf1, space, scoring='accuracy', n_jobs=-1)
```

In [6]:

```
clf2.fit(x_train, y_train)
print('Best Score: %s' % clf2.best_score_)
print('Best Hyperparameters: %s' % clf2.best_params_)
```

```
Best Score: 0.8291275167785235
Best Hyperparameters: {'C': 1, 'kernel': 'linear'}
```

In [7]:

```
clf2_acc = clf2.score(x_test, y_test)
print('Accuracy for SVM with GridSearchCV = ', clf2_acc)
```

```
Accuracy for SVM with GridSearchCV =  0.8255451713395638
```

3. Model SVM trained and tuned using RandomSearchCV.

In [8]:

```
# SVM dengan RandomSearchCV

from sklearn.model_selection import RandomizedSearchCV
from scipy.stats import loguniform

space = dict()
space['C'] = loguniform(0.01, 1)
space['kernel'] = ['linear', 'poly']
clf3 = RandomizedSearchCV(clf1, space, n_iter=200, scoring='accuracy', n_jobs=-1, random_st
```

In [9]:

```
clf3.fit(x_train, y_train)
print('Best Score: %s' % clf3.best_score_)
print('Best Hyperparameters: %s' % clf3.best_params_)
```

```
Best Score: 0.8318031319910514
Best Hyperparameters: {'C': 0.6620591065093844, 'kernel': 'linear'}
```

In [10]:

```
clf3_acc = clf3.score(x_test, y_test)
print('Accuracy for SVM with RandomizedSearchCV = ', clf3_acc)
```

```
Accuracy for SVM with RandomizedSearchCV =  0.822429906542056
```

4. Model SVM trained and tuned using Bayesian Optimization (available in Scikit-optimize library).

In [11]:

```
# SVM dengan Bayesian Optimization

from skopt import BayesSearchCV
from skopt.space import Real, Categorical

space = dict()
space['C'] = Real(0.01, 1, prior='log-uniform')
space['kernel'] = Categorical(['linear', 'poly'])
clf4 = BayesSearchCV(clf1, space, n_iter=200, scoring='accuracy', random_state=1)
```

In [12]:

```
clf4.fit(x_train, y_train)
print('Best Score: %s' % clf4.best_score_)
print('Best Hyperparameters: %s' % clf4.best_params_)
```

```
/usr/local/lib/python3.6/dist-packages/skopt/optimizer/optimizer.py:449: Use
rWarning: The objective has been evaluated at this point before.
  warnings.warn("The objective has been evaluated "

Best Score: 0.8317757009345794
Best Hyperparameters: OrderedDict([('C', 0.45214395181090716), ('kernel', 'l
inear')])
```

In [13]:

```
clf4_acc = clf4.score(x_test, y_test)
print('Accuracy for SVM with Bayesian Optimization = ', clf4_acc)
```

```
Accuracy for SVM with Bayesian Optimization =  0.8255451713395638
```

Kesimpulan:

Akurasi dengan menggunakan default hyperparameter : **0.5950155763239875**
Akurasi dengan GridSearchCV : **0.8255451713395638**
Akurasi dengan RandomSearchCV : **0.822429906542056**
Akurasi dengan Bayesian Optimization : **0.8255451713395638**

Akurasi terbaik didapatkan dengan menggunakan tuning **GridSearchCV dan Bayesian Optimization** walaupun dengan selisih akurasi yang sedikit dengan tuning menggunakan RandomSearchCV.