# CM4

February 25, 2021

## 1 [CM4] Seeds Dataset (Naive Bayes)

### 1.1 Data Pre-processing

#### 1.1.1 Importing Libraries

[1]:
```
import pandas as pd
from sklearn.model_selection import KFold,GridSearchCV,train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import StandardScaler

import warnings
warnings.filterwarnings("ignore")
```

#### 1.1.2 Loading dataset

[2]:
```
seeds_data = pd.read_csv('seeds_dataset.txt', sep="\t", error_bad_lines=False,␣
 ↪warn_bad_lines=False)

seeds_data.
 ↪columns=['area','perimeter','compactness','length_kernel','width_kernel','asymmetry_coeff',
```

[3]:
```
X = seeds_data.iloc[:,0:7].values
y = seeds_data.iloc[:,7].values

# Without Standardization
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=0)

# With Standardization
scaler = StandardScaler()
X1 = scaler.fit_transform(X)
X_train1, X_test1, y_train1, y_test1 = train_test_split(X1, y, test_size=0.2,␣
 ↪random_state=0)
```

We have divided 20% dataset in testing and 80% for training and validation.

## 1.2 Naive Bayes Algorithm (Without Standardization)

### 1.2.1 Applying algorithm on training set

[4]:
```
kf = KFold(random_state=0,n_splits=10)
param_grid={'var_smoothing':[1e-10, 1e-9, 1e-5, 1e-3, 1e-1]}

classifier = GridSearchCV(GaussianNB(), param_grid=param_grid, cv=kf,␣
 ↪scoring="accuracy", n_jobs=-1)
classifier = classifier.fit(X_train, y_train)

print(classifier.best_params_)
results = classifier.cv_results_
print(results['mean_test_score'])
```

```
{'var_smoothing': 0.001}
[0.905  0.905  0.905  0.9175 0.905 ]
```

### 1.2.2 Applying algorithm on test set

[5]:
```
clf = classifier.best_estimator_
clf.fit(X_train, y_train)

predictions = clf.predict(X_test)
print(accuracy_score(y_test, predictions))
```

```
0.925
```

## 1.3 Naive Bayes Algorithm (With Standardization)

### 1.3.1 Applying algorithm on training set

[6]:
```
kf = KFold(random_state=0,n_splits=10)
param_grid={'var_smoothing':[1e-10, 1e-9, 1e-5, 1e-3, 1e-1]}

classifier = GridSearchCV(GaussianNB(), param_grid=param_grid, cv=kf,␣
 ↪scoring="accuracy", n_jobs=-1)
classifier = classifier.fit(X_train1, y_train1)

print(classifier.best_params_)
results = classifier.cv_results_
print(results['mean_test_score'])
```

```
{'var_smoothing': 1e-10}
[0.905   0.905   0.905   0.905   0.89875]
```

### 1.3.2 Applying algorithm on test set

[7]:
```
clf = classifier.best_estimator_
clf.fit(X_train1, y_train1)
predictions = clf.predict(X_test1)
print(accuracy_score(y_test1, predictions))
```

```
0.875
```

## 1.4 Observation

The highest accuracy achieved using Gaussian NB algorithm on training set is 91.75, which we got for var_smoothing value **1e-3**. Thus, for final testing we have selected best parameter given by GridSearchCV. In final test set, we got 92.5 accuracy.

The accuracy varies with varying of var_smoothing parameter in Gaussian NB. It can be observed that, as the value of var_smoothing parameter increases the mean_test_score of the algorithm increases as well. But, again for too high var_smoothing value(which is 1e-1) the accuraacy of algorithm starts decreasing.

**As small value of var_smoothing might miss the cases with higher variance, while too big var_smoothing values might take in consideration values with least correlations, so the best fit for algorithm is 1e-3(neither small nor big).**

With standardization, accuracy decreases.

## 1.5 References


https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html
https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html
https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html