MODUL 11: TEXT CLASSIFICATION (2)

11.1 Deskripsi Singkat

Klasifikasi teks (text classification) merupakan salah satu tugas penting dalam pembelajaran mesin (machine learning). Pada text classification, kita melakukan proses penetapan tag/kategori ke suatu dokumen agar secara otomatis dan cepat dapat menganalisis teks untuk digunakan pada aplikasi lanjutan seperti analisis sentimen, deteksi spam, pelabelan topik, deteksi hoax, dll. Data tidak terstruktur berupa teks pada email, media sosial, aplikasi chat, respon survei, dll tersedia secara melimpah saat ini. Teks dapat diolah menjadi sumber informasi yang kaya, namun karena sifatnya yang tidak terstruktur, ada tantangan tersendiri untuk melakukan ekstraksi pengetahuan dari teks tersebut.

11.2 Tujuan Praktikum

Setelah praktikum pada modul 11 ini diharapkan mahasiswa mempunyai kompetensi sebagai berikut:

1) Dapat mempraktekkan text classification dengan Rocchio Classification.
2) Dapat mempraktekkan text classification dengan k Nearest Neighbor.
3) Dapat mempraktekkan text classification dengan Support Vector Machine.

11.3 Material Praktikum

Tidak ada

11.4 Kegiatan Praktikum

Lanjutkan task klasifikasi teks pada modul 10 dengan menggunakan Rocchio Classification, k Nearest Neighbor (kNN), dan Support Vector Machine (SVM).

A. Pembangunan Model Klasifikasi Teks dengan Rocchio Classification

```
from sklearn.neighbors import NearestCentroid
rocchio_tfidf = NearestCentroid()
rocchio_tfidf.fit(X_train_vectors_tfidf, y_train)  #model

#Melakukan prediksi nilai y pada dataset testing
y_predict = rocchio_tfidf.predict(X_test_vectors_tfidf)
```

B. Pembangunan Model Klasifikasi Teks dengan kNN

```
from sklearn.neighbors import KNeighborsClassifier
n_neighbors=5
knn_tfidf = KNeighborsClassifier(n_neighbors, weights='distance')
knn_tfidf.fit(X_train_vectors_tfidf, y_train)  #model

#Melakukan prediksi nilai y pada dataset testing
```

```
y_predict = knn_tfidf.predict(X_test_vectors_tfidf)
```

C. Pembangunan Model Klasifikasi Teks dengan SVM

```
from sklearn import svm
svm_tfidf = svm.SVC(C=1.0, kernel='linear', degree=3, gamma='auto')
svm_tfidf.fit(X_train_vectors_tfidf, y_train)  #model

#Melakukan prediksi nilai y pada dataset testing
y_predict = svm_tfidf.predict(X_test_vectors_tfidf)
```

D. Mencari Parameter Terbaik dengan Grid Search

```
from sklearn.model_selection import GridSearchCV

# defining parameter range
param_grid = {'C': [0.1, 1, 10, 100, 1000],
              'gamma': [1, 0.1, 0.01, 0.001, 0.0001],
              'kernel': ['linear', 'rbf']}

grid = GridSearchCV(svm.SVC(), param_grid, refit = True, verbose = 3)

# fitting the model for grid search
grid.fit(X_train_vectors_tfidf, y_train)
```

Parameter terbaik hasil percobaan dengan Grid Search didapatkan dengan:

```
# print best parameter after tuning
print(grid.best_params_)

# print how our model looks after hyper-parameter tuning
print(grid.best_estimator_)
```

Selanjutnya, prediksi menggunakan parameter terbaik dapat dilakukan dengan kode berikut.

```
grid_predictions = grid.predict(X_test_vectors_tfidf)
```

E. Menggunakan k-Fold Cross Validation

Jika data yang dimiliki terdapat indikasi imbalanced, maka dapat menggunakan k-fold cross validation agar hasil evaluasi yang ditampilkan lebih mewakili keseluruhan data.
Karena dataset yang Anda gunakan sebelumnya telah dibagi menjadi data training dan testing, maka gabungkan terlebih dahulu data tersebut dengan melakukan concat, baru kemudian melakukan cross validation terhadap keseluruhan data.

```
X_join = pd.concat([X_train, X_test])
y_join = pd.concat([y_train, y_test])
```

```
X_join_vectors_tfidf = tfidf_vectorizer.transform(X_join)
```

Berikut kode yang digunakan untuk melakukan 5-fold cross validation, misalnya untuk model Naive Bayes.

```
from sklearn.model_selection import cross_val_score, cross_val_predict, cross_validate

scores = cross_validate(nb_tfidf, X_join_vectors_tfidf, y_join, cv=5, scoring=('accuracy', 'f1'), return_train_score=True)
predictions = cross_val_predict(nb_tfidf, X_join_vectors_tfidf, y_join, cv=5)
print(scores)
```

Bandingkan skor yang ditampilkan dengan hasil klasifikasi Naive Bayes pada modul 10.

11.5 Penugasan

1. Lakukan duplikasi dari latihan di modul ini dengan Google Colab kalian masing-masing.
2. Bandingkan hasil evaluasi klasifikasi teks dari Naive Bayes, Rocchio Classification, kNN, dan SVM. Lakukan interpretasi berdasarkan hasil perbandingan tersebut.