Pemrosesan Bahasa Alami (IF5282)
Klasifikasi Keluhan

Oleh:
Robby Syaifullah / 13515013
Iftitakhul Zakiah / 13515114

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
2019

A. Dataset

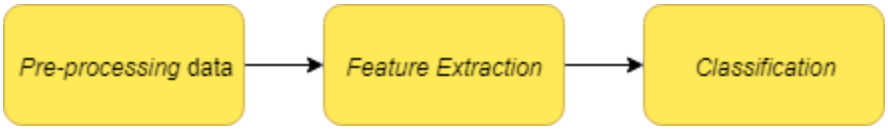
Dataset yang digunakan berasal dari Twitter yang dibagi menjadi tiga jenis kelas, yaitu kelas Keluhan, Respon, dan Bukan Keluhan/Respon. Dataset menggunakan Bahasa Indonesia dengan kolom 'Tweet' yang mencantumkan isi Tweet, serta nilai kolom Keluhan yang bernilai "Ya" jika Tweet tersebut masuk ke dalam kelas Keluhan dan bernilai "NaN" jika bukan masuk ke dalam kelas Keluhan. Begitu juga untuk kolom Respon dan Bukan Keluhan/Respon. Berikut ini merupakan contoh dataset yang digunakan.

	Tweet	Keluhan	Respon	Bukan Keluhan/Respon
0	@EL_Atheos @ridwankamil ya mungkin karena pere...	NaN	NaN	Ya
1	@ridwankamil @dbmpkotabdg kang teman saya tert...	Ya	NaN	NaN
2	Di tribun jabar biasanya suka di post agenda k...	NaN	NaN	Ya
3	@dbmpkotabdg RT @fajriattack: Lapor pak @ridwa...	Ya	NaN	NaN
4	@diskamtam bapak/ibu mau tanya, kalo pemelihar...	NaN	NaN	Ya

Gambar 1. Contoh dataset yang digunakan

Dataset yang terdiri dari tiga kelas ini memiliki jumlah yang tidak seimbang, atau *imbalanced dataset* dan juga terdapat banyak duplikasi. Dengan kondisi data tesebut maka dibutuhkan penanganan khusus, dalam eksperimen ini dilakukan penghapusan data yang duplikasi sebagai salah satu bagian dari eksperimen.

B. Arsitektur



Gambar 2. Arsitektur program untuk klasifikasi keluhan

1. *Pre-processing* data

   Proses ini bertujuan untuk membersihkan dan mengolah data sehingga data siap digunakan untuk proses selanjutnya. Dalam *pre-processing* data, terdapat beberapa proses, yaitu :

   a. Menghilangkan *term* yang memiliki pengaruh besar terhadap klasifikasi dokumen. *Term* ini dapat berupa *stopword* atau kata yang sering dipakai (contohnya : yang, kepada, dll). Selain itu, pada data Twitter juga terdapat *tag* dari nama-nama *username* dan juga *link* atau URL yang harus dihapuskan.

```
Raw Data : RT @se_bdg: #WartaBDG Lebih tegas,
```

```
@dishub_kotabdg Evaluasi Sistem Gembok Parkir
https://t.co/F9vGwDxZFm @IWGinting @ridwankamil
https://t.\x85
```

```
Output : RT se_bdg: #WartaBDG Lebih tegas, dishub_kotabdg
Evaluasi Sistem Gembok Parkir  IWGinting ridwankamil \x85'
```

b. Menghapus tanda baca yang ada pada data.

```
Output : RT sebdg WartaBDG Lebih tegas dishubkotabdg
Evaluasi Sistem Gembok Parkir  IWGinting ridwankamil \x85
```

c. Menghapus Stopword. Menggunakan *library* untuk menghapus kata yang sering digunakan di semua label.

```
Output : RT sebdg WartaBDG Lebih tegas dishubkotabdg
Evaluasi Sistem Gembok Parkir  IWGinting ridwankamil \x85
```

2. *Feature extraction*

Dari data-data yang telah dilakukan *pre-processing,* maka akan dihitung nilai vektor dokumen-dokumen tesebut sebagai *input* dari *classifier.* Pada eksperimen ini, ekstraksi fitur menggunakan *library* Scikit Learn ialah sebagai berikut:

a. CountVectorizer

Adalah fitur ekstraksi standar dimana nilai sebuah fitur tergantung dari jumlah kemunculan fitur tersebut di dalam dokumen.

b. TfidfVectorizer

Mirip dengan CountVectorizer, namun dilakukan penekanan bobot untuk *term* dengan frekuensi kemunculan lebih tinggi. Caranya adalah mentransformasi dengan menormalisasi matriks frekuensi *term*. Dan TF-IDF cocok untuk komputasi perbandingan antara dua dokumen.

c. Threshold Variance

Tujuan dari teknik ini adalah menghapus fitur nilainya berada di bawah variansi data. Meskipun ini lebih cocok untuk unsupervised learning, dalam eksperimen tetap dimasukkan untuk melihat pengaruh dari teknik feature selection untuk *unsupervised* jika diterapkan pada *supervised*.

3. *Classification*

   Seperti pada ekstraksi fitur, pada bagian klasifikasi juga menggunakan *library* Scikit Learn. Berikut ini merupakan algoritma pembelajaran mesin yang digunakan pada eksperimen.

   a. KNeighborsClassifier

      Adalah *supervised classifier* yang membuat model dengan cara melihat sejumlah k tetangga terdekat dari setiap datanya.

   b. DecisionTreeClassifier

      Adalah model klasifikasi *supervised* dengan menggunakan bentuk seperti *tree*, klasifikasi ini menggunakan information gain sebagai tambahan penilaian pada *classifiernya.*

   c. SupportVectorMachines

      SVM adalah salah satu linear kernel yang sempat menjadi *state of the art* dalam klasifikasi teks dan memiliki keunggulan dalam waktu komputasi, ruang yang terpakai dan kinerja dari hasil kategorisasi.

C. Skenario Eksperimen

Eksperimen terbagi menjadi 4 tahap:

1. Tahap Data. Pada tahap ini, akan dibandingkan hasil kinerja dengan menggunakan data normal (7523) dan data yang unik(5076).
2. Tahap Preprocessing. Pada tahap precprocessing, akan dibandingkan hasil kinerja dari penggunaan stopwords dan *tag username* pada aplikasi twitter dan kombinasinya.
3. Tahap Ekstraksi Fitur. Ekstraksi fitur menggunakan tiga cara, yakni dengan memilih fitur dengan menggunakan Count Vectorizer, TF IDF atau Threshold Variance yang sebenarnya merupakan kelanjutan dari Count Vectorizer.
4. Tahap Pemilihan Model. Tahap ini menggunakan tiga jenis algoritma pembelajaran mesin dan dipilih model yang menghasilkan hasil klasifikasi paling optimum.

D. Hasil Eksperimen

1. Tahap Data

Tabel 1. Nilai Hasil Eksperimen Variasi Data

Variasi	*F1 Score*	*Accuracy Score*	*Precision Score*	*Recall Score*
Data NonDuplicate	0.874	0.850	0.870	0.878
Data Normal	0.904	0.888	0.903	0.905

Eksperimen dilakukan dengan menggunakan semua fitur yang ada pada sub bab sebelumnya, ekstraksi fitur menggunakan *Count Vectorizer* dan menggunakan algoritma pembelajaran Linear SVM One vs Rest.

2. Tahap *Preprocessing*

Tabel 2. Hasil Eksperimen Setiap Variasi *Preprocessing Dataset*

Variasi Fitur	*F1 Score*	*Accuracy Score*	*Precision Score*	*Recall Score*
removeURL+ Tag+Punct.+ Stopwords	0.904	0.888	0.903	0.905
removeURL+ Tag+Punct.	0.905	0.890	0.902	0.908
removeURL+ Punct.+Stop Words	**0.911**	**0.895**	**0.909**	**0.913**

Eksperimen dilakukan dengan menggunakan semua preprocessing, ekstraksi fitur menggunakan Count Vectorizer dan menggunakan algoritma pembelajaran Linear SVM One vs Rest.

3. Tahap *Feature Extraction*

Tabel 3. Nilai Hasil Eksperimen Setiap Variasi Fitur *Dataset*

Variasi	*F1 Score*	*Accuracy Score*	*Precision Score*	*Recall Score*
TF-IDF Vectorizer	0.908	0.893	0.908	0.908
Count Vectorizer	0.904	0.888	0.903	0.905
Variance Threshold	**0.911**	**0.895**	**0.908**	**0.913**

Eksperimen dilakukan dengan menggunakan data normal, semua *preprocessing* pada sub bab sebelumnya, menggunakan algoritma pembelajaran Linear SVM One vs Rest.

4. Tahap Algoritma

Tabel 4. Nilai Hasil Eksperimen Setiap Variasi Algoritma Pembelajaran

Variasi	*F1 Score*	*Accuracy Score*	*Precision Score*	*Recall Score*

KNN	0.780	0.769	0.784	0.862
DTL	0.860	0.834	0.856	0.864
SVM	**0.904**	**0.888**	**0.903**	**0.905**

Eksperimen dilakukan dengan menggunakan data normal, semua *preprocessing* dan ekstraksi fitur menggunakan Count Vectorizer.

Tabel 5. Nilai Hasil Eksperimen Setiap Variasi Algoritma Pembelajaran

Variasi	*F1 Score*	*Accuracy Score*	*Precision Score*	*Recall Score*
KNN	0.780	0.769	0.784	0.862
DTL	0.862	0.837	0.860	0.864
SVM	**0.911**	**0.895**	**0.909**	**0.913**

Eksperimen dilakukan dengan menggunakan data normal, semua *preprocessing* dan ekstraksi fitur menggunakan Threshold Variance.

E. Analisis

Untuk data preprocessing, menghapuskan url tanpa menghapus username yang di tag dapat meningkatkan kinerja dari model, namun kami beranggapan secara heuristik hal ini tidak berpengaruh banyak dan juga ditunjukkan oleh hasil kinerja yang perbedaanya tidak terlalu signifikan.

Untuk ekstraksi fitur, countvectorizer memang memiliki kinerja lebih rendah dari TF IDF, namun ketika digabungkan, kinerja dari CountVectorizerldan Threshold Variance lebih optimal daripada TF IDF Untuk model yang digunakan, Linear SVM multilabel dengan teknik one vst rest memberikan hasil terbaik dari ketiga algoritma di eksperimen. Hal ini karena SVM termasuk dalam algoritma pembelajaran linear kernel dan data pada kategorisasi teks cenderung/umumnya *linearly separable*. Sehingga penggunaan Linear kernel classifier seperti SVM akan mendapatkan hasil yang optimal dengan waktu latih relatif cepat, dan hanya membutuhkan representasi teks yang baik untuk mendapatkan hasil yang lebih optimal.

Pada KNN, pelabelan *instance* hanya berdasarkan faktor jarak antar fitur, tanpa memperhitungkan bobotnya. Sehingga, nilai F1 dan akurasinya lebih rendah daripada algoritma pemodelan lainnya.

Pada DTL, ada perhitungan bobot dalam melakukan klasifikasi, yang disebut *information gain*. Namun DTL tidak lebih baik daripada *Logistic Regression* karena ketika *base-rate* rendah DTL akan mengalami masalah dalam melakukan *split*, atau bahkan tidak bisa melakukan *split*. Dikarenakan tidak adanya algoritma yang sempurna

untuk menyelesaikan masalah klasifikasi, maka akan selalu ada kesalahan dalam proses klasifikasi *tweet* aduan tersebut.