Dokumen Final Text Mining

Perbandingan Klasifikasi Teks Berita Menggunakan Algoritma Naive Bayes dan Support Vector Machine

Disusun oleh :

1. Kartini Nurfalah (1301154577)
2. Vatana Rianti Aldefi (1301154566)
3. Asih Wulandari A (1301154657)

Fakultas Informatika

Universitas Telkom

Bandung

2017/2018

1. **Daftar Bacaan**

a. @article{ariadi2016klasifikasi,

title={Klasifikasi Berita Indonesia Menggunakan Metode Naive Bayesian Classification dan Support Vector Machine dengan Confix Stripping Stemmer},

author={Ariadi, Dio and Fithriasari, Kartika},

journal={Jurnal Sains dan Seni ITS},

volume={4},

number={2},

year={2016}

}

Ringkasan : Jumlah aliran artikel berita yang diunggah di internet sangat banyak dan rentang waktu yang cepat. Jumlah yang banyak dan waktu yang cepat akan menyulitkan editor mengkategorikan secara manual. Terdapat metode agar berita dapat dikategorikan secara otomatis, yaitu klasifikasi. Data berita berbentuk teks, sehingga jauh lebih rumit dan perlu proses untuk mempersiapkan data. Salah satu prosesnya adalah confix-stripping stemmer sebagai cara untuk mendapatkan kata dasar dari berita Indonesia. Untuk metode klasifikasi yang digunakan adalah Naive Bayes Classifier (NBC) yang secara umum sering digunakan dalam data teks dan Support Vector Machine (SVM) yang diketahui bekerja sangat baik pada data dengan dimensi besar. Kedua metode tersebut akan dibandingkan untuk mengetahui hasil klasifikasi yang paling baik. Hasil penelitian menunjukkan bahwa SVM kernel Linier dan kernel RBF menghasilkan ketepatan klasifikasi yang sama dan bila dibandingkan dengan NBC maka SVM lebih baik.

b. @article{yusuf2013support,

title={Support Vector Machines yang didukung K-Means clustering dalam klasifikasi dokumen},

author={Yusuf, Ahmad and Priambadha, Tirta},

journal={JUTI: Jurnal Ilmiah Teknologi Informasi},

volume={11},

number={1},
pages={15--18},
year={2013}
}

Ringkasan : Dokumen dengan jumlah data yang besar dan bervariasi seringkali mempersulit proses klasifikasi. Hal ini dapat diperbaiki dengan mengatasi variasi data untuk menghasilkan akurasi yang lebih baik. Penelitian ini mengusulkan sebuah metode baru untuk kategorisasi dokumen teks berbahasa Inggris dengan terlebih dahulu melakukan pengelompokan menggunakan K-Means Clustering kemudian dokumen diklasifikasikan menggunakan multi-class Support Vector Machines (SVM). Dengan adanya pengelompokan tersebut, variasi data dalam membentuk model klasifikasi akan lebih seragam. Hasil uji coba terhadap judul artikel jurnal ilmiah menunjukkan bahwa metode yang diusulkan mampu meningkatkan akurasi dengan menghasilkan akurasi sebesar 88,1%, presisi sebesar 96,7% dan recall sebesar 94,4% dengan parameter jumlah kelompok sebesar 5.

c. @article{rasywir2016eksperimen,
title={Eksperimen pada Sistem Klasifikasi Berita Hoax Berbahasa Indonesia Berbasis Pembelajaran Mesin},
author={Rasywir, Errissya and Purwarianti, Ayu},
journal={Jurnal Cybermatika},
volume={3},
number={2},
year={2016}
}

Ringkasan : Klasifikasi berita hoax atau berita dengan informasi yang tidak benar merupakan salah satu aplikasi kategorisasi teks. Seperti aplikasi kategorisasi teks berbasis pembelajaran mesin pada umumnya, sistem ini terdiri atas praproses, ekstraksi fitur, seleksi fitur dan pengeksekusian model klasifikasi. Pada penelitian

ini, eksperimen dilakukan untuk memilih teknik terbaik pada setiap sub proses dengan menggunakan 220 artikel berbahasa Indonesia dalam 22 topik (89 artikel hoax dan 131 artikel bukan hoax). Untuk praproses, hasil eksperimen terbaik dicapai oleh praproses tanpa stemming dan dengan penghapusan stop word. Untuk ekstraksi fitur, fitur unigram memiliki akurasi terbaik dibandingkan dengan bigram dan unigram+bigram. Untuk seleksi fitur, teknik terbaik adalah penggunaan operasi union pada mutual information dan information gain. Sedangkan untuk algoritma klasifikasi, dengan berbagai kombinasi di atas, algoritma naïve bayes menunjukkan hasil akurasi yang terbaik dibandingkan dengan SVM dan C4.5 dengan nilai akurasi 91.36%.

d. @article{rachimawan2016ads,

title={Ads Filtering Mengunakan Jaringan Syaraf Tiruan Perceptron, Naive Bayes Classifier, dan Regresi logistik},

author={Rachimawan, Achmad Fachrudin and Ulama, Brodjol Sutijo Suprih},

journal={Jurnal Sains dan Seni ITS},

volume={5},

number={1},

pages={D83--D89},

year={2016}

}

Ringkasan : Email merupakan fasilitas yang mutlak diperlukan dalam berbagai bidang. Pentingnya email dan jumlahnya yang begitu banyak menyebabkan penyalahgunaan. Salah satu penyalahgunaan yang sering ditemui adalah email iklan yang dikirimkan oleh perusahaan penyedia konten internet saat pengguna mendaftar pada situs perusahaan tersebut. Terdapat metode agar email iklan dari perusahaan-perusahaan tersebut bisa secara otomatis dikenali yaitu klasifikasi. Data email berbentuk teks, sehingga jauh lebih rumit dan perlu proses untuk mempersiapkan data. Salah satu prosesnya adalah pembobotan ads atau adicity. Untuk metode klasifikasi yang digunakan adalah Naive Bayes Classifier (NBC)

yang secara umum sering digunakan dalam data teks dan Perceptron yang diketahui keduanya merupakan metode yang cukup sederhana untuk menyelesaikan permasalahan yang kompleks. Kedua metode tersebut akan dibandingkan untuk mengetahui hasil klasifikasi yang paling baik. Hasil penelitian menunjukkan bahwa NBC lebih unggul dibanding Perceptron, dan pada NBC False Positive Ratio lebih mudah untuk dikontrol.

e. @phdthesis{kurniawan2017implementasi,

title={Implementasi Text Mining pada Analisis Sentimen Pengguna Twitter Terhadap Media Mainstream Menggunakan Naive Bayes Classifier dan Support Vector Machine},

author={Kurniawan, Taufik},

year={2017},

school={Institut Teknologi Sepuluh Nopember}

}

Ringkasan : Keresahan masyarakat terhadap pemberitaan media mainstream sempat menjadi trending topic di media sosial Twitter akibat ketidakpuasan terhadap media yang dinilai tidak representatif dan independen dalam memuat berita. Bahkan pemerintah membahas secara khusus terkait bahaya berita palsu yang sering beredar. Beberapa media mainstream yang fokus sebagai media berita dan banyak mendapat tanggapan masyarakat di media sosial adalah TV One, Metro TV, dan Kompas TV. Sehingga perlu dilakukan penelitian guna mengetahui bagaimana sentimen publik terhadap ketiga media tersebut, apakah mayoritas publik menilai positif atau negatif. Tanggapan publik mengenai media mainstream didapat dari Application Programming Interface (API) pada Twitter karena media sosial tersebut memiliki pengguna yang sangat banyak di Indonesia bahkan hingga mencapai 19,5 juta pengguna dari total 300 juta pengguna global. Pada penelitian ini, praproses teks yang digunakan adalah case folding, tokenizing, stopwords, dan stemming. Untuk praproses stemming digunakan algoritma confix-stripping stemmer Sedangkan pada analisis klasifikasi data teks tersebut

digunakan metode Naïve Bayes Classifier dan Support Vector Machine. Klasifikasi menggunakan NBC pada data media TV One dan Kompas TV menghasilkan akurasi sebesar 95,6% dan 97,8%, sedangkan pada media Metro TV menghasilkan nilai G-mean dan AUC berturut-turut sebesar 81,3% and 82,36%. Klasifikasi menggunakan SVM pada data media TV One dan Kompas TV menghasilkan akurasi sebesar 97,9% dan 99,3%,, sedangkan pada media Metro TV menghasilkan nilai G-mean dan AUC berturut-turut sebesar 97,35% and 97,38%.

f. @article{rofiqoh2548analisis,

title={Analisis Sentimen Tingkat Kepuasan Pengguna Penyedia Layanan Telekomunikasi Seluler Indonesia Pada Twitter Dengan Metode Support Vector Machine dan Lexicon Based Features},

author={Rofiqoh, Umi and Perdana, Rizal Setya and Fauzi, M Ali},

journal={Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN},

volume={2548},

pages={964X}

}

Ringkasan : Pendeteksian kekuatan sentimen adalah salah satu penelitian terbaru dari analisis sentimen. Yang berbeda adalah, sistem menggunakan perilaku berupa informasi bobot kekuatan sentimen dari setiap term yang terdeteksi untuk mendapatkan polaritas teks. Tugas Akhir ini membangun sebuah sistem yang mengadaptasi classifier SentiStrength. SentiStrength adalah algoritma sekaligus program opinion mining yang menggunakan pendekatan berbasis kamus/leksikon. Kamus/leksikon SentiStrength ini berisi terms berikut bobot kekuatan sentimennya. Dikarenakan kamus/leksikon default SentiStrength dibuat dalam bahasa Inggris (English), maka sesuai ijin dari pengembangnya, leksikon ini telah diterjemahkan ke dalam bahasa Indonesia untuk kebutuhan penelitian. Penelitian

ini menghasilkan nilai akurasi sebesar 57.33%. Hasil pengujian lainnya dapat dilihat lebih lanjut.

g. @inproceedings{mccallum1998comparison,
title={A comparison of event models for naive bayes text classification},
author={McCallum, Andrew and Nigam, Kamal and others},
booktitle={AAAI-98 workshop on learning for text categorization},
volume={752},
number={1},
pages={41--48},
year={1998},
organization={Citeseer}
}

h. @article{tong2001support,
title={Support vector machine active learning with applications to text classification},
author={Tong, Simon and Koller, Daphne},
journal={Journal of machine learning research},
volume={2},
number={Nov},
pages={45--66},
year={2001}
}

i. @article{chen2009feature,
title={Feature selection for text classification with Naive Bayes},
author={Chen, Jingnian and Huang, Houkuan and Tian, Shengfeng and Qu, Youli},
journal={Expert Systems with Applications},
volume={36},
number={3},

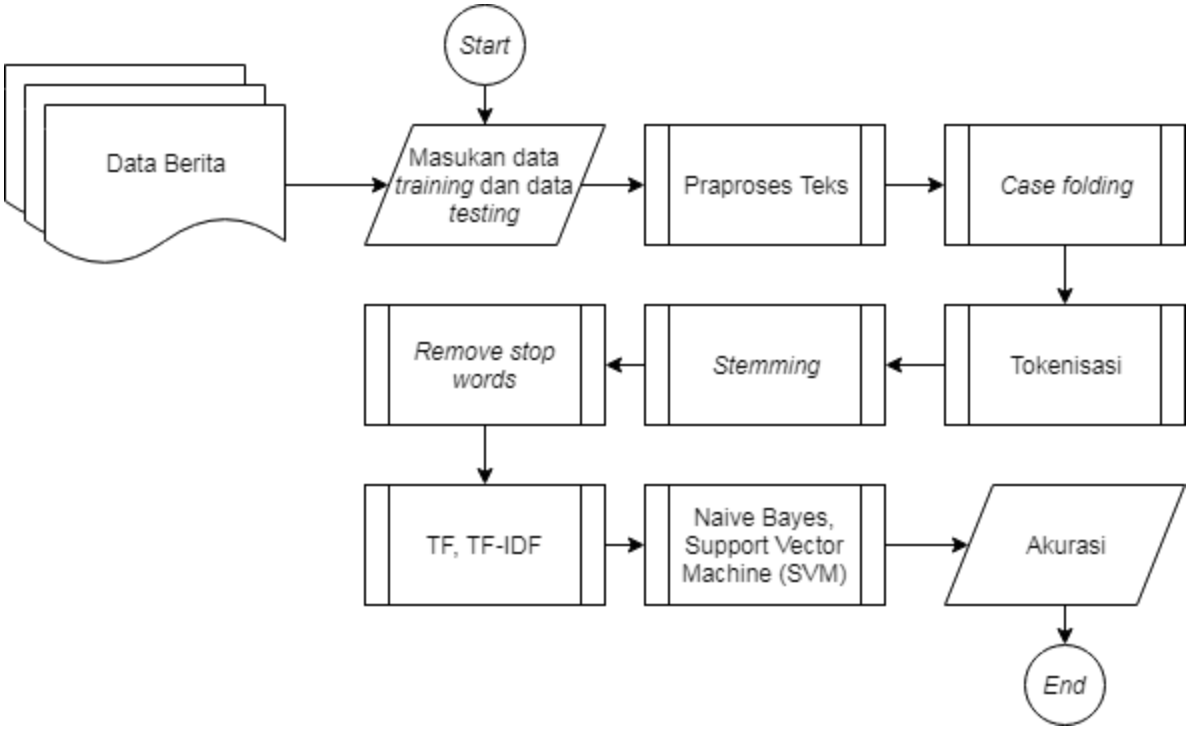
pages={5432--5435},
year={2009},
publisher={Elsevier}
}

j. @inproceedings{lewis1998naive,
title={Naive (Bayes) at forty: The independence assumption in information retrieval},
author={Lewis, David D},
booktitle={European conference on machine learning},
pages={4--15},
year={1998},
organization={Springer}
}

k. @article{tong2001support,
title={Support vector machine active learning with applications to text classification},
author={Tong, Simon and Koller, Daphne},
journal={Journal of machine learning research},
volume={2},
number={Nov},
pages={45--66},
year={2001}
}

2. Praproses Detil

Berikut gambaran praproses detil yang akan kami rancang :



Flowmap 2.1 Praproses Detil

a. Masukan Data

Pada tahap ini data yang kami masukan adalah data *training* (data latih) dan data *testing* (data uji). Dalam data *training* dan data *testing* berisi berita-berita dengan banyak jenis *tag* contoh: *tag newsitem*, *tag headline*, *tag text*, *tag* p, dan lainnya.

b. Praproses Teks

Setelah kita memuat data *training* dan data *testing*, kita ambil teks pada data tersebut yang hanya memiliki *tag headline* dan *tag text* yang didalamnya terdapat banyak *tag* p yang merupakan kumpulan *paragraph* yang selanjutnya akan dilakukan praproses.

- Data *Training*:

Index	Type	Size	Value
0	str	1	Inflation, GDP derivatives seen on the way.
1	str	1	Court says to rule on VW-GM lawsuit Oct 30.
2	str	1	FOCUS - Court says to rule on VW-GM lawsuit Oct 30.
3	str	1	German court says to rule in VW-GM lawsuit Oct 30.
4	str	1	Clinton signs annual intelligence bill.
5	str	1	Croatia says hires foreign auditor for state firms.
6	str	1	GM says to leave VW dispute to courts.
7	str	1	VW denies it hires ex-Clinton consultant.
8	str	1	VW hires Clinton consultant for GM probe - report.
9	str	1	GM, VW lawyers face off in Lopez spy case.
10	str	1	Yugo, London club to hold "technical" talks - paper.

Gambar 2.1 Pengambilan isi dari *tag headline* pada data *training*

Index	Type	Size	Value
0	list	12	['Companies may soon be able to trade derivatives of major economic in ...
1	list	9	['A German court said Wednesday it would rule at the end of the month ...
2	list	13	['A German court said on Wednesday it would rule at the end of the mon ...
3	list	7	['A German court said on Wednesday said it would rule by the end of th ...
4	list	10	['President Bill Clinton on Friday signed into law a spending authoris ...
5	list	7	['Croatia announced on Monday it had hired foreign auditor Arthur Ande ...
6	list	5	['General Motors Corp, responding to reports that Volkswagen AG hired ...
7	list	9	['German carmaker Volkswagen AG denied on Tuesday a newspaper report s ...
8	list	10	['German carmaker Volkswagen AG has hired as a consultant, former Whit ...
9	list	9	['Lawyers for General Motors Corp. and Volkswagen AG were to face off ...
10	list	7	['A delegation of the London club creditors is due to begin "technical ...

Gambar 2.2 Pengambilan isi dari *tag text* pada data *training*

- Data *Testing*

Index	Type	Size	Value
0	str	1	FOCUS-VW unveils new Passat, says sales well up.
1	str	1	US CREDIT OUTLOOK - Bearish technicals, long weekend.
2	str	1	Iran says five spy networks destroyed, 41 held.
3	str	1	At least 44 dead as vessel capsizes in India.
4	str	1	Closing stock market indices.
5	str	1	Closing stock market indices.
6	str	1	STOCKS -- Closing stock market indices, Sept 2.
7	str	1	VW chief wants to boost return on sales tenfold.
8	str	1	U.S. Congress moves to curb economic spying.
9	str	1	U.S. Congress moves to curb economic spying.
10	str	1	Chronology of North Korean infiltrations into South.

Gambar 2.3 Pengambilan isi dari *tag headline* pada data *testing*

Index	Type	Size	Value
0	list	22	['Germany's Volkswagen AG , unveiling a new Passat car aimed at boosti ...
1	list	20	['The U.S. 30-year Treasury bond will likely remain under pressure in ...
2	list	6	['Iranian security forces have broken up five espionage rings in north ...
3	list	3	['At least 44 people were feared drowned when their vessel capsized in ...
4	list	9	['Here is how major stock markets outside the United States ended Mond ...
5	list	9	['Here is how major stock markets outside the United States ended on M ...
6	list	9	['Here is how major stock markets outside the United States ended on M ...
7	list	22	['Volkswagen AG management board chairman Ferdinand Piech said in an i ...
8	list	6	['A bill intended to curb economic espionage by foreign countries and ...
9	list	6	['A bill intended to curb economic espionage by foreign countries and ...
10	list	11	['A North Korean infiltration into the enemy South on Wednesday is one ...

Gambar 2.4 Pengambilan isi dari *tag text* pada data *testing*

c. Case Folding

Setelah kita mendapatkan informasi berupa *headline* dan teks berita maka proses yang akan dilakukan, yaitu *case folding. Case folding* merupakan proses perubahan isi berita dari yang memiliki huruf besar dan huruf kecil menjadi huruf kecil saja (*lower case*). Proses ini perlu dilakukan karena kita membutuhkan suatu informasi yang pada saat diproses maka akan menghasilkan informasi yang selalu konsisten. Konsisten disini merupakan bentuk informasi atau tulisan yang semuanya dalam bentuk *lower case*. Proses ini membantu memudahkan pada proses klasifikasi nanti.

d. Tokenisasi

Tahap ini merupakan tahap pembagian informasi ke dalam kata per kata fungsinya untuk mempermudah pada tahap klasifikasi. Pada tahap selanjutnya tokenisasi digunakan untuk memisahkan tiap kata agar bisa dilakukan proses *stemming* dan proses penghapusan *stop words.*

e. Stemming

Setelah dilakukan tokenisasi, maka tahap selanjutnya, yaitu *stemming*. *Stemming* merupakan proses pengubahan bentuk kata menjadi kata dasar atau tahap pencarian *root* kata dari tiap kata. Dengan dilakukannya proses *stemming*, maka setiap kata berimbuhan akan berubah menjadi kata dasar, dengan demikian dapat lebih mengoptimalkan proses pada *text mining*.

f. Remove Stop Words

Pada tahap ini menghilangkan kata umum yang banyak muncul dan dianggap tidak memiliki makna**.** Kata yang diperoleh dari tahap *stemming* dicek dalam suatu daftar *stop word*, apabila sebuah kata masuk di dalam daftar *stop word* maka kata tersebut tidak akan diproses lebih lanjut. Sebaliknya apabila sebuah kata tidak termasuk di dalam daftar *stop word* maka kata tersebut akan masuk keproses berikutnya.

g. Perhitungan TF, TF-IDF

Pada tahap ini akan dilakukan perhitungan TF (*Term Frequency*) serta TF-IDF. TF merupakan kemunculan sebuah term dalam dokumen yang bersangkutan, sedangkan IDF merupakan sebuah perhitungan dari bagaimana term didistribusikan secara luas pada koleksi dokumen yang bersangkutan.

h. Klasifikasi

Setelah dilakukan praproses dan perhitungan TF, serta TF-IDF, maka langkah selanjutnya akan dilakukan klasifikasi menggunakan Naive Bayes dan Support Vector Machine. Naive Bayes merupakan sebuah metoda klasifikasi menggunakan metode probabilitas dan statistik yang dikemukakan oleh ilmuwan Inggris Thomas Bayes, sedangkan Support Vector Machine merupakan metode pengklasifikasian dengan supervisi yang mencari *hyperplane* terbaik yang

memisahkan data-data dengan kelas-kelas yang berbeda. Hasil Klasifikasi ini akan menunjukan Class(yes, no) tiap dokumen.

3. Proses Klasifikasi

Pada dokumen ini kami menggunakan klasifikasi dengan menggunakan Naive Bayes dan SVM (Support Vector Machine).

a. Pengujian praproses

Pengujian dilakukan dengan menggunakan dataset berupa data berita bahasa inggris dengan kelas biner, *yes* dan *no* untuk tahapan pelatihan dan pengujian. Tahap praproses yang dilakukan, yaitu *case folding*, tokenisasi, *stemming*, dan *remove stop words.*

```
def Cleaning(AllDocuments):
        CLEANDOCUMENTS = []
        for i, document in enumerate(AllDocuments):
            cleanContent = []
            for j, content in enumerate(document):
                if j == 0 :
                    cleanContent.append(content)
                    continue
                stopWords = set(stopwords.words('english'))
                cleanWords = (re.sub(r'[.,\/#!$%\^&\*;:{}=\-_+`~()\'\"{0-9}]', '
',content.lower()))
                words = word_tokenize(cleanWords)
                wordsFilteredStemmed = []
                for w in words:
                    if w not in stopWords:
                        wordsFilteredStemmed.append(stemming.stem(w))
                cleanContent.append(wordsFilteredStemmed)
            CLEANDOCUMENTS.append(cleanContent)
        return CLEANDOCUMENTS
```

Membuat fungsi yang berisi perulangan pada setiap data berita untuk menghilangkan tanda baca dan angka dengan menggunakan *library Regular Expressions*. Setelah menghilangkan tanda baca dan angka, lalu dilakukan *case*

folding dan selanjutnya dilakukan tokenisasi menggunakan word_tokenize. Tahap selajutnya yaitu melakukan penghapusan *stop words* menggunakan *library stop words* serta *stemming* menggunakan library PorterStemmer.

b. Pengujian Proses Klasifikasi

Setelah melewati praproses maka dataset akan diuji dengan algoritma Naive Bayes dan Support Vector Machine (SVM). Naive Bayes adalah sebuah metode klasifikasi menggunakan metode probabilitas dan statistik sedangkan Support Vector Machine merupakan metode pengklasifikasian dengan supervisi yang mencari *hyperplane* terbaik yang memisahkan data-data dengan kelas-kelas yang berbeda.

```
def CORPUS(Data):
        WORDS = []
        for i, document in enumerate(Data):
            for j, content in enumerate(document):
                if j == 0: continue
                for k, word in enumerate(content):
                    WORDS.append(word)
         return WORDS
```

Corpus merupakan fungsi untuk mendapatkan kata unik pada setiap berita. Kemudian kata unik tersebut menjadi sebuah atribut untuk melakukan perhitungan TF.

```
def TF(Data, CORPUS):
        RESULT = []
        c = Counter(CORPUS)
        for i, document in enumerate(Data):
            dataTF = []
            dataIDF = []
            for j, word in enumerate(CORPUS):
                countWord = document[1].count(word)
```

```
                tf = countWord / len(document[1])
                countWordAllDocs = c[word]
                idf = tf * np.log(len(Data) / countWordAllDocs)
                dataTF.append(tf)
                dataIDF.append(idf)
            RESULT.append([dataTF, dataIDF])
        return [item[0] for item in RESULT], [item[1] for item in RESULT]
```

Untuk melakukan pengklasifikasian SVM memerlukan perhitungan TF, maka dari itu hitung terlebih dahulu TFnya per data berita. Dengan melakukan perulangan pada tiap data berita kemudian dengan berdasarkan data corpus atau atribut, hitung berapa banyak muncul suatu kata dalam suatu berita dibagi dengan jumlah kata dalam satu berita. Lalu dilakukan perhitungan TF-IDF.

```
def NaiveBayes():
        content = [item[1] for item in
AllDocumentsTokenized][:TRAINCOUNT]
         concatenatedContent = []
        for i, cont in enumerate(content):
                concatenatedContent.append(' '.join(map(str, cont)))
        vectorizer = CountVectorizer(concatenatedContent)
        Generated = vectorizer.fit_transform(concatenatedContent).toarray()
        ids = [item[0] for item in AllDocumentsTokenized]
        clf = GaussianNB()
        clf.fit(Generated, LabelTrain)
        result = []
        if (isTesting == False):
                result = cross_val_score(clf, Generated, LabelTrain, cv=10)
                concatenated = np.column_stack((ids, clf.predict(Generated)))
        else:
                result = Accuracy(clf.predict(Generated), LabelTrain)
                concatenated = np.column_stack((ids, clf.predict(Generated)))
        return result, concatenated
AccuracyNaiveBayes, NaiveBayesTrain = NaiveBayes()
```

Proses klasifikasi naive bayes menggunakan library GaussianNB.

```
def SVM():
        ids = [item[0] for item in AllDocumentsTokenized]
        clf = svm.SVC(C=100, gamma=0.1)
        clf.fit(TfIdf, LabelTrain)
        result = []
        if (isTesting == False):
            result = cross_val_score(clf, TfIdf, LabelTrain, cv=10)
             concatenated = np.column_stack((ids, clf.predict(TfIdf)))
        else:
             result = Accuracy(clf.predict(TfIdf), LabelTrain)
             concatenated = np.column_stack((ids, clf.predict(TfIdf)))
        return result, concatenated
AccuracySVM, SVMTrain = SVM()
```

Kemudian proses pengkalsifikasian SVM berdasar pada TF-IDF yang sudah dihitung dan menggunkan library svm.

4. Hasil Pengujian

a. Hasil pengujian praproses

Pengujian dilakukan dengan menggunakan dataset berupa data berita bahasa inggris dengan kelas biner, yaitu *yes* dan *no* untuk tahapan pelatihan dan pengujian. Tahap praproses yang dilakukan, yaitu *case folding*, tokenisasi, *stemming*, dan *remove stop words*.

a) *Case Folding* dan Tokenisasi

- Data *Training*:

Index	Type	Size	Value
0	list	7	['inflation', 'gdp', 'derivatives', 'seen', 'on', 'the', 'way']
1	list	9	['court', 'says', 'to', 'rule', 'on', 'vw', 'gm', 'lawsuit', 'oct']
2	list	10	['focus', 'court', 'says', 'to', 'rule', 'on', 'vw', 'gm', 'lawsuit', ...
3	list	10	['german', 'court', 'says', 'to', 'rule', 'in', 'vw', 'gm', 'lawsuit', ...
4	list	5	['clinton', 'signs', 'annual', 'intelligence', 'bill']
5	list	8	['croatia', 'says', 'hires', 'foreign', 'auditor', 'for', 'state', 'fi ...
6	list	8	['gm', 'says', 'to', 'leave', 'vw', 'dispute', 'to', 'courts']
7	list	7	['vw', 'denies', 'it', 'hires', 'ex', 'clinton', 'consultant']
8	list	8	['vw', 'hires', 'clinton', 'consultant', 'for', 'gm', 'probe', 'report ...
9	list	9	['gm', 'vw', 'lawyers', 'face', 'off', 'in', 'lopez', 'spy', 'case']
10	list	8	['yugo', 'london', 'club', 'to', 'hold', 'technical', 'talks', 'paper' ...

Gambar 4.1 *Case folding* dan tokenisasi *headline* pada data *training*

Index	Type	Size	Value
0	list	12	[['companies', 'may', 'soon', 'be', 'able', ...], ['adrian', 'maconick ...
1	list	9	[['a', 'german', 'court', 'said', 'wednesday', ...], ['following', 'st ...
2	list	13	[['a', 'german', 'court', 'said', 'on', ...], ['following', 'statement ...
3	list	7	[['a', 'german', 'court', 'said', 'on', ...], ['volkwagen', 'is', 'see ...
4	list	10	[['president', 'bill', 'clinton', 'on', 'friday', ...], ['clinton', 'a ...
5	list	7	[['croatia', 'announced', 'on', 'monday', 'it', ...], ['head', 'of', ' ...
6	list	5	[['general', 'motors', 'corp', 'responding', 'to', ...], ['we', 'remai ...
7	list	9	[['german', 'carmaker', 'volkswagen', 'ag', 'denied', ...], ['the', 'i ...
8	list	10	[['german', 'carmaker', 'volkswagen', 'ag', 'has', ...], ['the', 'inte ...
9	list	9	[['lawyers', 'for', 'general', 'motors', 'corp', ...], ['in', 'a', 'he ...
10	list	7	[['a', 'delegation', 'of', 'the', 'london', ...], ['it', 'is', 'not', ...

Gambar 4.2 *Case folding* dan tokenisasi isi berita pada data *training*

- Data *Testing*:

Index	Type	Size	Value
0	list	9	['focus', 'vw', 'unveils', 'new', 'passat', 'says', 'sales', 'well', ' ...
1	list	7	['us', 'credit', 'outlook', 'bearish', 'technicals', 'long', 'weekend' ...
2	list	7	['iran', 'says', 'five', 'spy', 'networks', 'destroyed', 'held']
3	list	8	['at', 'least', 'dead', 'as', 'vessel', 'capsizes', 'in', 'india']
4	list	4	['closing', 'stock', 'market', 'indices']
5	list	4	['closing', 'stock', 'market', 'indices']
6	list	6	['stocks', 'closing', 'stock', 'market', 'indices', 'sept']
7	list	9	['vw', 'chief', 'wants', 'to', 'boost', 'return', 'on', 'sales', 'tenf ...
8	list	8	['u', 's', 'congress', 'moves', 'to', 'curb', 'economic', 'spying']
9	list	8	['u', 's', 'congress', 'moves', 'to', 'curb', 'economic', 'spying']
10	list	7	['chronology', 'of', 'north', 'korean', 'infiltrations', 'into', 'sout ...

Gambar 4.3 *Case folding* dan tokenisasi *headline* pada data *testing*

Index	Type	Size	Value
3	list	3	[['at', 'least', 'people', 'were', 'feared', ...], ['it', 'quoted', 'o ...
4	list	9	[['here', 'is', 'how', 'major', 'stock', ...], ['london', 'british', ' ...
5	list	9	[['here', 'is', 'how', 'major', 'stock', ...], ['london', 'uk', 'stock ...
6	list	9	[['here', 'is', 'how', 'major', 'stock', ...], ['london', 'uk', 'stock ...
7	list	22	[['volkswagen', 'ag', 'management', 'board', 'chairman', ...], ['piech ...
8	list	6	[['a', 'bill', 'intended', 'to', 'curb', ...], ['the', 'bill', 'would' ...
9	list	6	[['a', 'bill', 'intended', 'to', 'curb', ...], ['the', 'bill', 'would' ...
10	list	11	[['a', 'north', 'korean', 'infiltration', 'into', ...], ['wednesday', ...
11	list	6	[['argentine', 'bonds', 'were', 'slightly', 'higher', ...], ['a', 'tra ...
12	list	19	[['the', 'international', 'committee', 'of', 'the', ...], ['the', 'swi ...
13	list	5	[['the', 'house', 'of', 'representatives', 'tuesday', ...], ['sponsors ...

Gambar 4.4 *Case folding* dan tokenisasi isi berita pada data *testing*

b) *Stemming*

- Data *Training*:

Index	Type	Size	Value
0	list	7	['inflat', 'gdp', 'deriv', 'seen', 'on', 'the', 'way']
1	list	9	['court', 'say', 'to', 'rule', 'on', 'vw', 'gm', 'lawsuit', 'oct']
2	list	10	['focu', 'court', 'say', 'to', 'rule', 'on', 'vw', 'gm', 'lawsuit', 'o ...
3	list	10	['german', 'court', 'say', 'to', 'rule', 'in', 'vw', 'gm', 'lawsuit', ...
4	list	5	['clinton', 'sign', 'annual', 'intellig', 'bill']
5	list	8	['croatia', 'say', 'hire', 'foreign', 'auditor', 'for', 'state', 'firm ...
6	list	8	['gm', 'say', 'to', 'leav', 'vw', 'disput', 'to', 'court']
7	list	7	['vw', 'deni', 'it', 'hire', 'ex', 'clinton', 'consult']
8	list	8	['vw', 'hire', 'clinton', 'consult', 'for', 'gm', 'probe', 'report']
9	list	9	['gm', 'vw', 'lawyer', 'face', 'off', 'in', 'lopez', 'spi', 'case']
10	list	8	['yugo', 'london', 'club', 'to', 'hold', 'technic', 'talk', 'paper']

Gambar 4.5 *Stemming headline* pada data *training*

Index	Type	Size	Value
0	list	12	[['compani', 'may', 'soon', 'be', 'abl', ...], ['adrian', 'maconick', ...
1	list	9	[['a', 'german', 'court', 'said', 'wednesday', ...], ['follow', 'state ...
2	list	13	[['a', 'german', 'court', 'said', 'on', ...], ['follow', 'statement', ...
3	list	7	[['a', 'german', 'court', 'said', 'on', ...], ['volkwagen', 'is', 'see ...
4	list	10	[['presid', 'bill', 'clinton', 'on', 'friday', ...], ['clinton', 'also ...
5	list	7	[['croatia', 'announc', 'on', 'monday', 'it', ...], ['head', 'of', 'th ...
6	list	5	[['gener', 'motor', 'corp', 'respond', 'to', ...], ['we', 'remain', 'c ...
7	list	9	[['german', 'carmak', 'volkswagen', 'ag', 'deni', ...], ['the', 'inter ...
8	list	10	[['german', 'carmak', 'volkswagen', 'ag', 'ha', ...], ['the', 'intern' ...
9	list	9	[['lawyer', 'for', 'gener', 'motor', 'corp', ...], ['in', 'a', 'hear', ...
10	list	7	[['a', 'deleg', 'of', 'the', 'london', ...], ['it', 'is', 'not', 'a', ...

Gambar 4.6 *Stemming* isi berita pada data *training*

- Data testing:

Index	Type	Size	Value
0	list	9	['focu', 'vw', 'unveil', 'new', 'passat', 'say', 'sale', 'well', 'up']
1	list	7	['us', 'credit', 'outlook', 'bearish', 'technic', 'long', 'weekend']
2	list	7	['iran', 'say', 'five', 'spi', 'network', 'destroy', 'held']
3	list	8	['at', 'least', 'dead', 'as', 'vessel', 'capsiz', 'in', 'india']
4	list	4	['close', 'stock', 'market', 'indic']
5	list	4	['close', 'stock', 'market', 'indic']
6	list	6	['stock', 'close', 'stock', 'market', 'indic', 'sept']
7	list	9	['vw', 'chief', 'want', 'to', 'boost', 'return', 'on', 'sale', 'tenfol ...
8	list	8	['u', 's', 'congress', 'move', 'to', 'curb', 'econom', 'spi']
9	list	8	['u', 's', 'congress', 'move', 'to', 'curb', 'econom', 'spi']
10	list	7	['chronolog', 'of', 'north', 'korean', 'infiltr', 'into', 'south']

Gambar 4.7 Stemming *headline* pada data *testing*

Index	Type	Size	Value
0	list	22	[['germani', ''s', 'volkswagen', 'ag', 'unveil', ...], ['the', 'wolfsb ...
1	list	20	[['the', 'us', 'year', 'treasuri', 'bond', ...], ['the', 'treasuri', ' ...
2	list	6	[['iranian', 'secur', 'forc', 'have', 'broken', ...], ['jomhuri', 'esl ...
3	list	3	[['at', 'least', 'peopl', 'were', 'fear', ...], ['it', 'quot', 'offici ...
4	list	9	[['here', 'is', 'how', 'major', 'stock', ...], ['london', 'british', ' ...
5	list	9	[['here', 'is', 'how', 'major', 'stock', ...], ['london', 'uk', 'stock ...
6	list	9	[['here', 'is', 'how', 'major', 'stock', ...], ['london', 'uk', 'stock ...
7	list	22	[['volkswagen', 'ag', 'manag', 'board', 'chairman', ...], ['piech', 's ...
8	list	6	[['a', 'bill', 'intend', 'to', 'curb', ...], ['the', 'bill', 'would', ...
9	list	6	[['a', 'bill', 'intend', 'to', 'curb', ...], ['the', 'bill', 'would', ...
10	list	11	[['a', 'north', 'korean', 'infiltr', 'into', ...], ['wednesday', ''s', ...

Gambar 4.8 *Stemming* isi berita pada data *testing*

c) *Remove Stop words*

Index	Type	Size	Value
0	str	1	inflat
1	str	1	gdp
2	str	1	deriv
3	str	1	seen
4	str	1	way
5	str	1	compani
6	str	1	may
7	str	1	soon
8	str	1	abl
9	str	1	trade
10	str	1	deriv

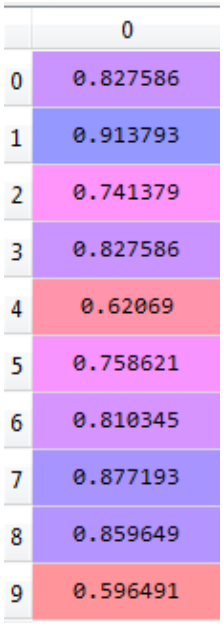
Gambar 4.9 Hasil penghapusan *stop words* pada data *testing*

b. Hasil pengujian proses klasifikasi

Setelah melewati praproses maka dataset akan diuji dengan algoritma Naive Bayes dan atau Support Vector Machine (SVM). Naive Bayes adalah sebuah metode klasifikasi menggunakan metode probabilitas dan statistik sedangkan Support Vector Machine merupakan metode pengklasifikasian dengan supervisi yang mencari *hyperplane* terbaik yang memisahkan data-data dengan kelas-kelas yang berbeda.

Hasil Akurasi:

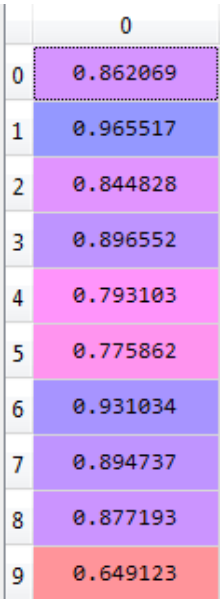
- Naive Bayes



	0
0	0.827586
1	0.913793
2	0.741379
3	0.827586
4	0.62069
5	0.758621
6	0.810345
7	0.877193
8	0.859649
9	0.596491

Gambar 4.9 Akurasi Naive Bayes

- SVM



	0
0	0.862069
1	0.965517
2	0.844828
3	0.896552
4	0.793103
5	0.775862
6	0.931034
7	0.894737
8	0.877193
9	0.649123

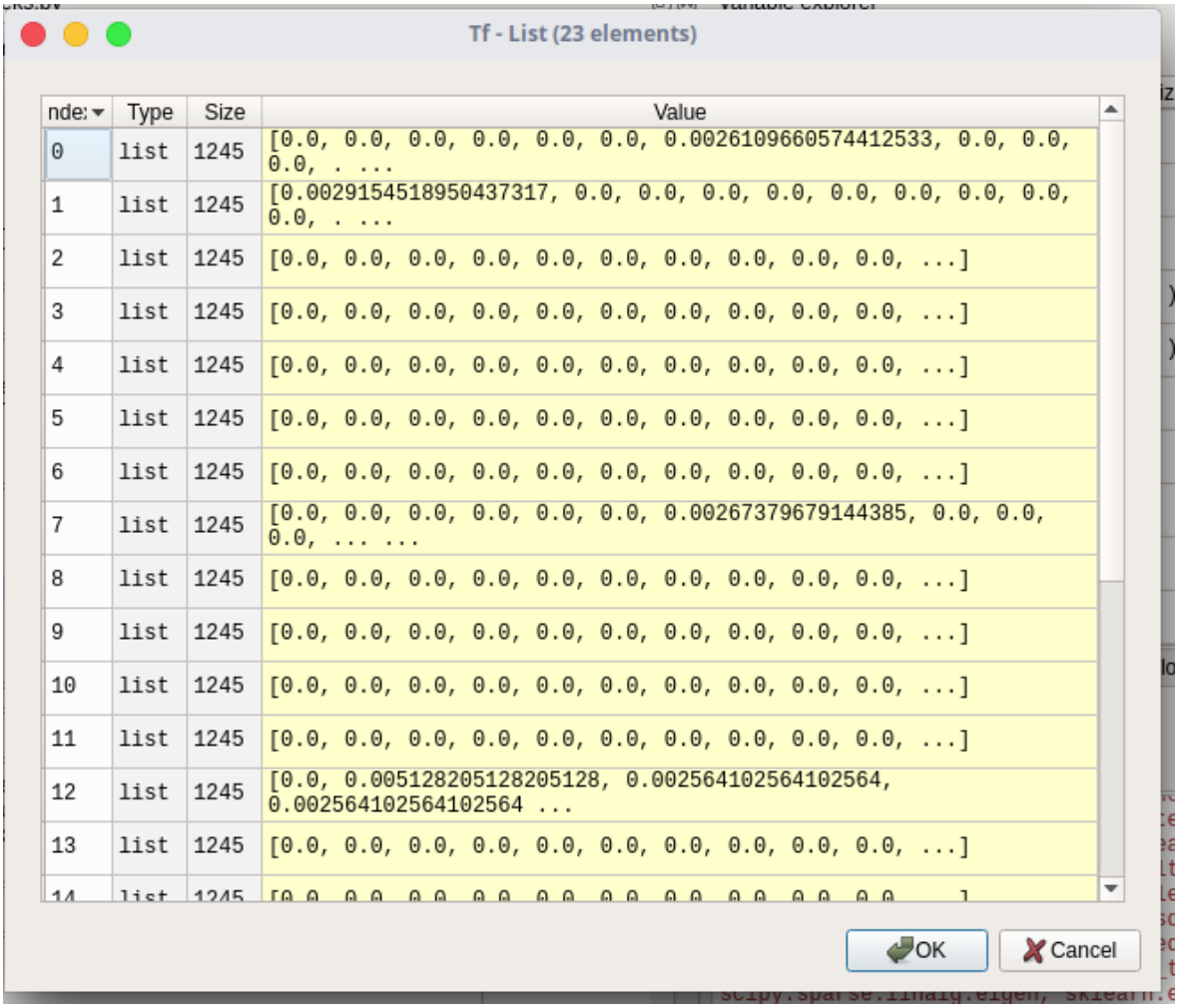
Gambar 4.10 Akurasi Support Vector Machine

5. Analisis dan Hasil Pengujian

a. Analisis hasil pengujian praproses

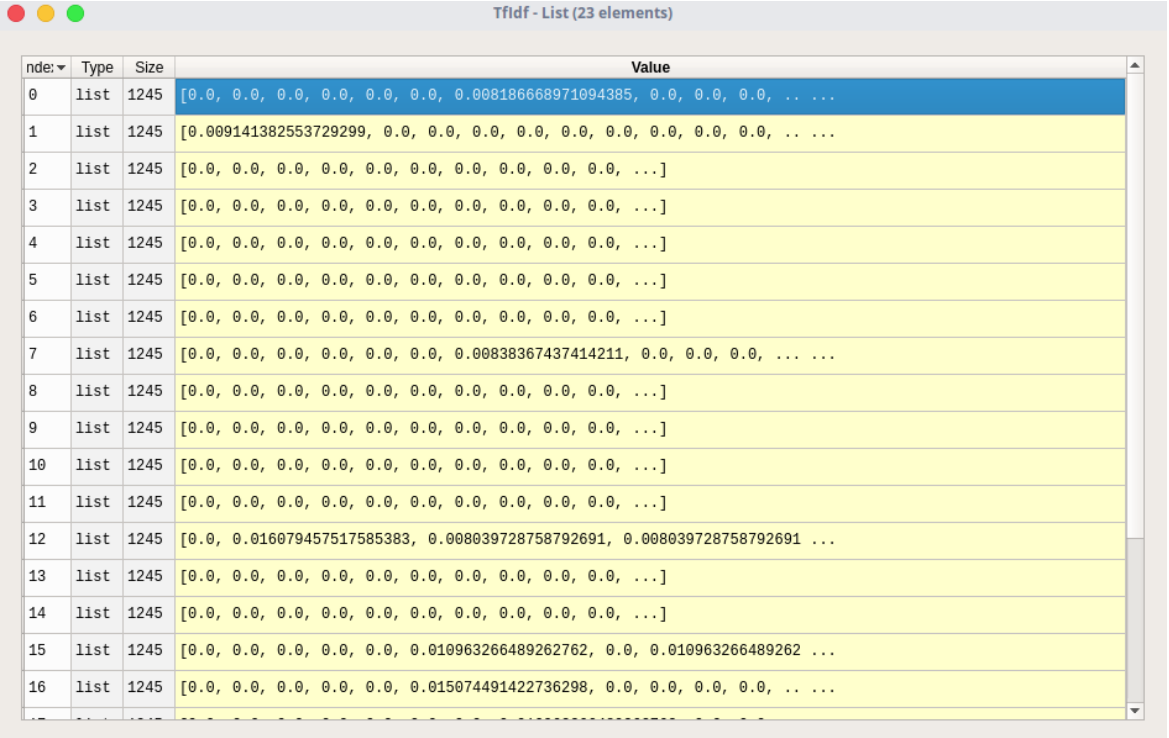
Hasil keluaran dari tahap pra-proses selanjutnya dilakukan pembobotan TF serta TF-IDF pada setiap term. Hasil tahap ekstraksi kalimat adalah bobot setiap kata dalam dokumen.

Perhitungan TF:



Pada size menunjukan angka 1245, angka tersebut menunjukan banyaknya fitur (atribut) yang unik dalam semua dokumen.

Perhitungan TF-IDF:



nde:	Type	Size	Value
0	list	1245	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.008186668971094385, 0.0, 0.0, 0.0, ...
1	list	1245	[0.009141382553729299, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...
2	list	1245	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...]
3	list	1245	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...]
4	list	1245	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...]
5	list	1245	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...]
6	list	1245	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...]
7	list	1245	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.00838367437414211, 0.0, 0.0, 0.0, ...
8	list	1245	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...]
9	list	1245	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...]
10	list	1245	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...]
11	list	1245	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...]
12	list	1245	[0.0, 0.016079457517585383, 0.008039728758792691, 0.008039728758792691 ...
13	list	1245	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...]
14	list	1245	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...]
15	list	1245	[0.0, 0.0, 0.0, 0.0, 0.0, 0.010963266489262762, 0.0, 0.010963266489262 ...
16	list	1245	[0.0, 0.0, 0.0, 0.0, 0.0, 0.015074491422736298, 0.0, 0.0, 0.0, 0.0, ...

Pada perhitungan TF-IDF pun size menunjukan fitur yang sama yaitu sebanyak 1245.

b. Analisis hasil pengujian proses klasifikasi

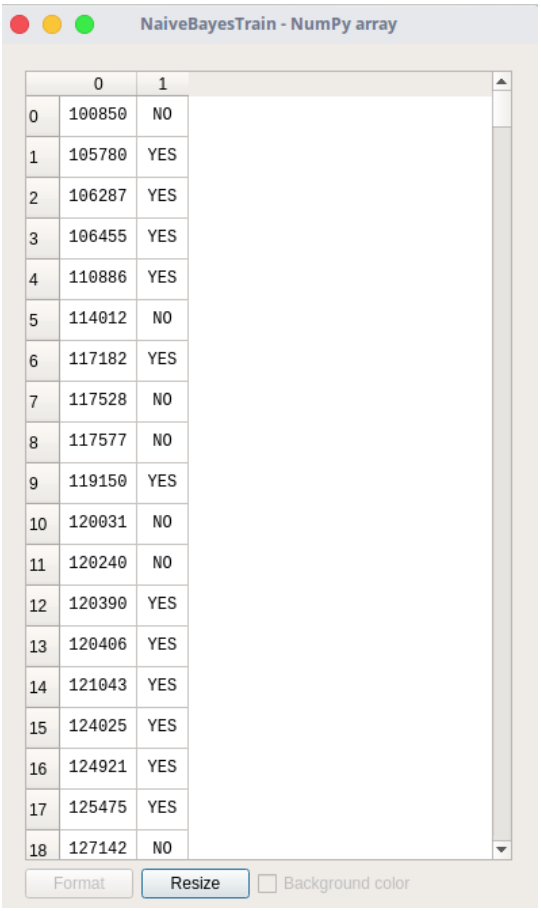
Membandingkan akurasi dari algoritma Naive Bayes dan Support Vector Machine (SVM). Pada Algoritma SVM terdapat parameter yang bisa mempengaruhi perubahan akurasi, yaitu nilai C dan gamma.

Kami sudah beberapa kali melakukan perubahan nilai C dan gamma. *Default* nilai C=1 dan gamma auto, kita bisa menentukan gamma sesuai hasil yang terbaik. Pada kasus ini kami menggunakan C = 100 dan gamma = 0.1 karena pada saat tersebut kami mendapatkan akurasi yang baik.

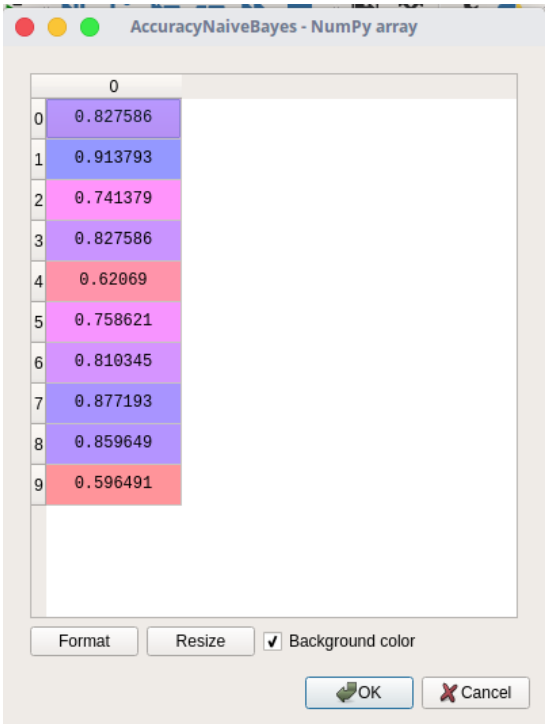
Pada saat pengujian pun waktu yang dibutuhkan untuk melakukan *running* tidak menentu karena fitur setiap dokumen bisa berbeda.

Akurasi Hasil dari Klasifikasi:

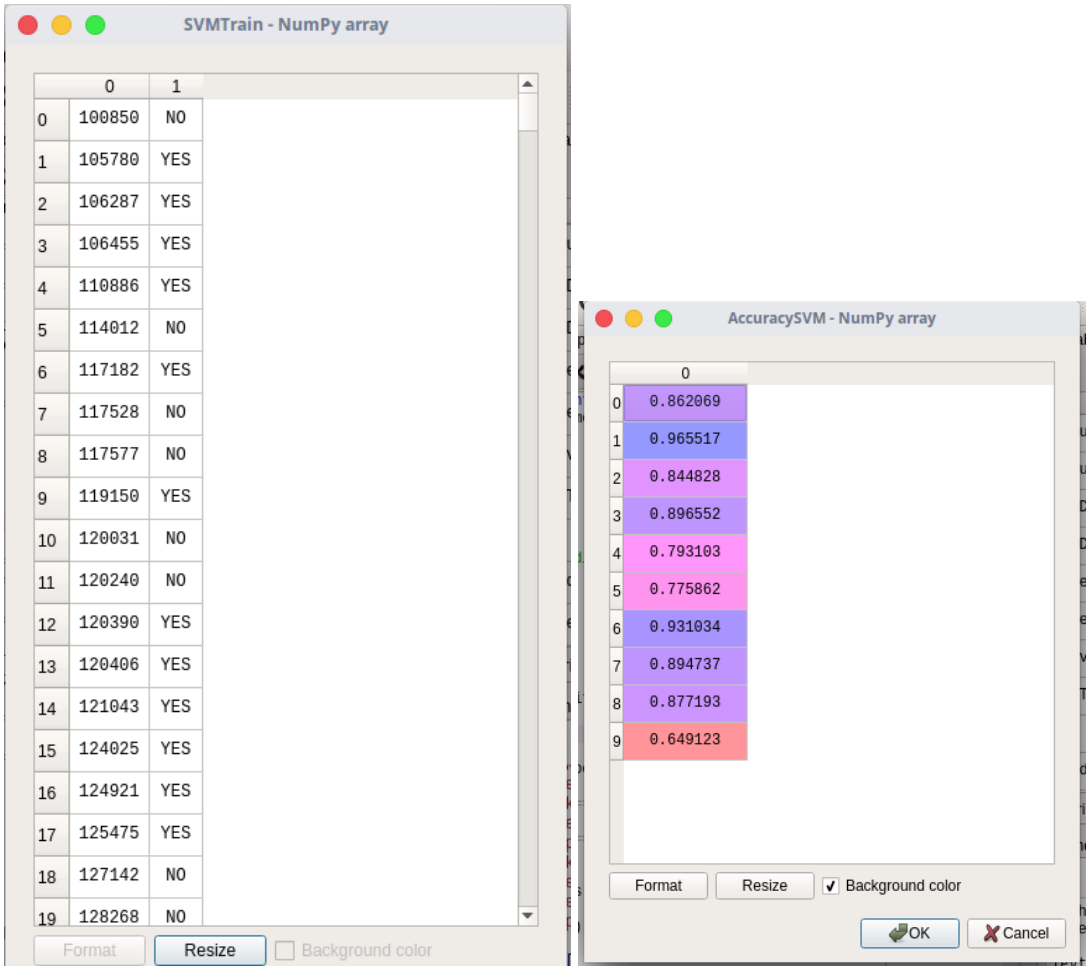
- Data Training



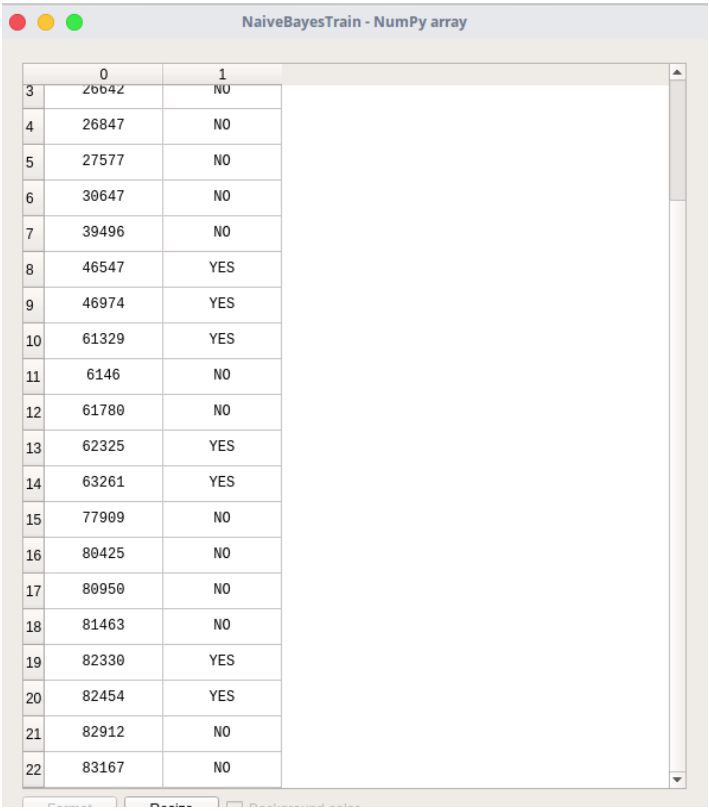
	0	1
0	100850	NO
1	105780	YES
2	106287	YES
3	106455	YES
4	110886	YES
5	114012	NO
6	117182	YES
7	117528	NO
8	117577	NO
9	119150	YES
10	120031	NO
11	120240	NO
12	120390	YES
13	120406	YES
14	121043	YES
15	124025	YES
16	124921	YES
17	125475	YES
18	127142	NO



	0
0	0.827586
1	0.913793
2	0.741379
3	0.827586
4	0.62069
5	0.758621
6	0.810345
7	0.877193
8	0.859649
9	0.596491

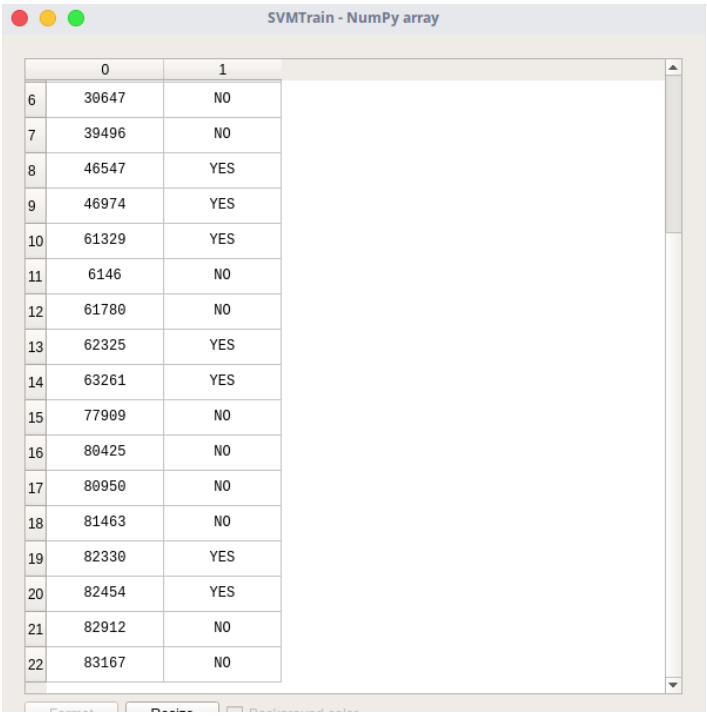

SVMTrain - NumPy array
0
1
0 100850 NO
1 105780 YES
2 106287 YES
3 106455 YES
4 110886 YES
5 114012 NO
6 117182 YES
7 117528 NO
8 117577 NO
9 119150 YES
10 120031 NO
11 120240 NO
12 120390 YES
13 120406 YES
14 121043 YES
15 124025 YES
16 124921 YES
17 125475 YES
18 127142 NO
19 128268 NO
Format
Resize
Background color
AccuracySVM - NumPy array
0
0 0.862069
1 0.965517
2 0.844828
3 0.896552
4 0.793103
5 0.775862
6 0.931034
7 0.894737
8 0.877193
9 0.649123
Format
Resize
Background color
OK
Cancel

- Data Testing



	0	1
3	26642	NO
4	26847	NO
5	27577	NO
6	30647	NO
7	39496	NO
8	46547	YES
9	46974	YES
10	61329	YES
11	6146	NO
12	61780	NO
13	62325	YES
14	63261	YES
15	77909	NO
16	80425	NO
17	80950	NO
18	81463	NO
19	82330	YES
20	82454	YES
21	82912	NO
22	83167	NO

AccuracyNaiveBayes	float	1	1.0



	0	1
6	30647	NO
7	39496	NO
8	46547	YES
9	46974	YES
10	61329	YES
11	6146	NO
12	61780	NO
13	62325	YES
14	63261	YES
15	77909	NO
16	80425	NO
17	80950	NO
18	81463	NO
19	82330	YES
20	82454	YES
21	82912	NO
22	83167	NO

AccuracySVM	float	1	1.0

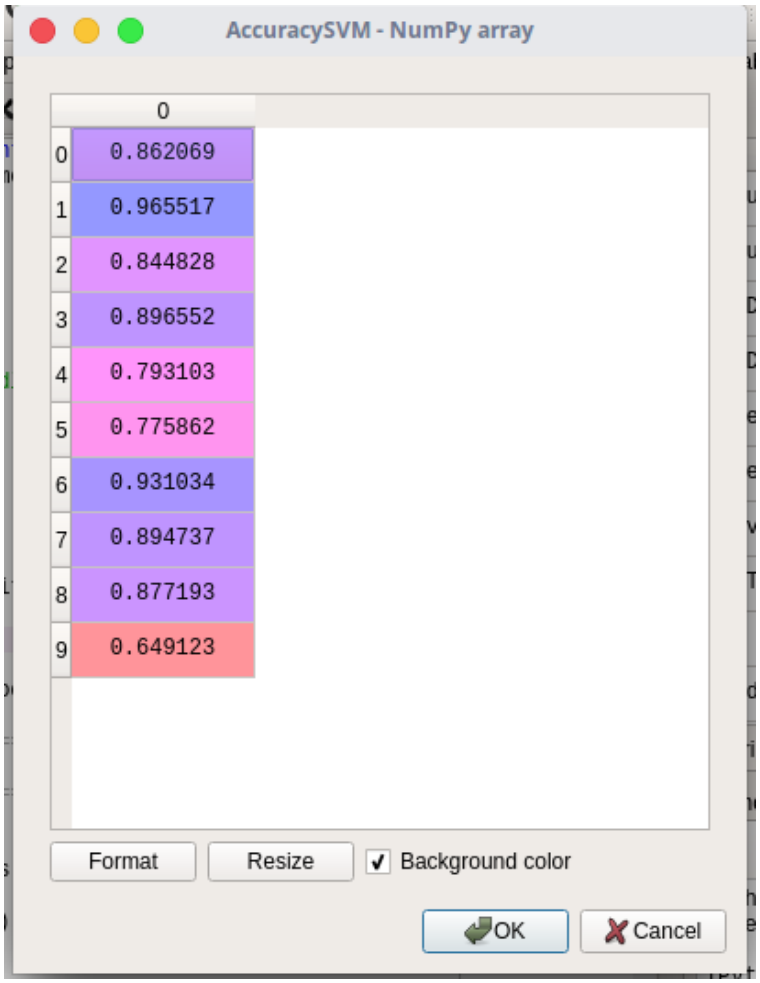
6. Kesimpulan

Dari hasil yang kami dapatkan bersdasarkan algoritma SVM dan Naive Bayes, Keduanya memiliki akurasi yang bagus, tetapi memang pada Dataset yang ada menunjukan bahwa Algoritma SVM memiliki akurasi lebih baik dari pada Naive Bayes walaupun selisih dari kedua algoritma ini tidak begitu signifikan.

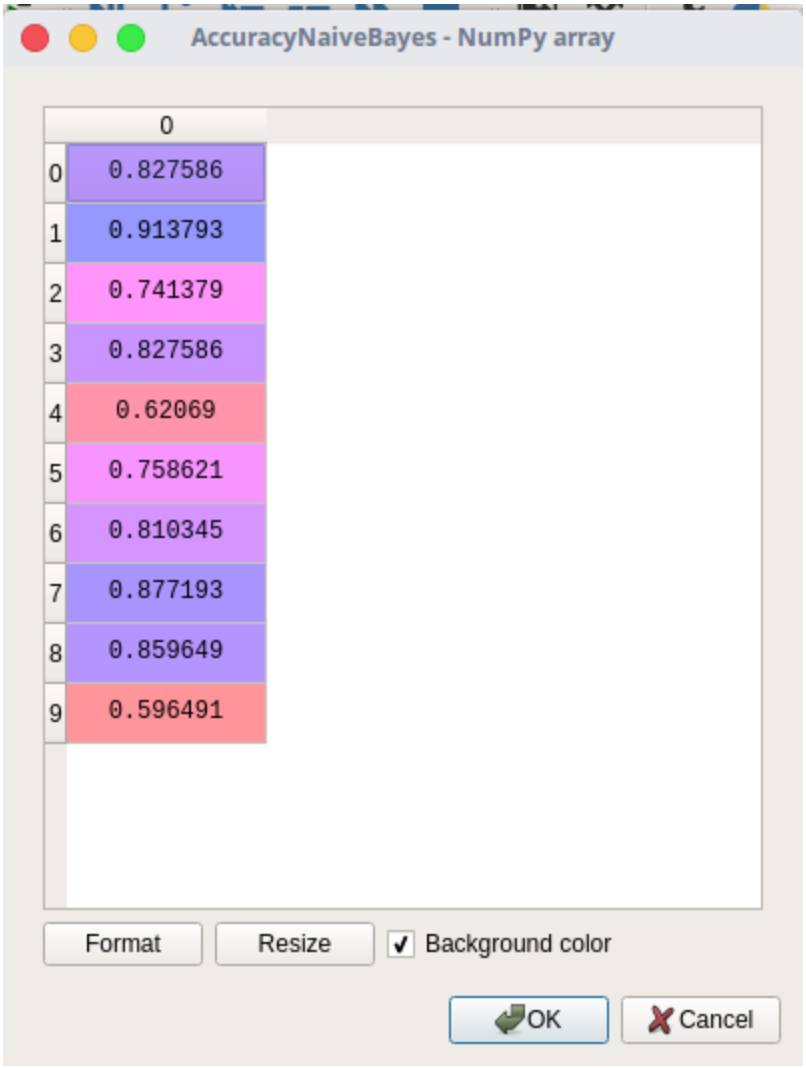
Berikut akurasi yang di hasilkan masing-masing algoritma terhadap data training dan data testing.

Data Training:

- SVM



- Naive Bayes



Data Tesing

- SVM

AccuracySVM	float	1	1.0

- Naive Bayes

AccuracyNaiveBayes	float	1	1.0

7. Lampiran Dokumentasi Program

DEPENDENCIES

```
#Dependencies
import xml.etree.ElementTree as ET
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import cross_val_score
from sklearn.naive_bayes import GaussianNB
from sklearn import svm
import os
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem.porter import PorterStemmer
stemming = PorterStemmer()
import re
import numpy as np
from collections import Counter
```

Load Data

```
PATHTRAINING = '/home/helmisatria/@FALAH/KlasifikasiBerita/Training set'
PATHTESTING = '/home/helmisatria/@FALAH/KlasifikasiBerita/Testing set'
PATHLABEL = '/home/helmisatria/@FALAH/KlasifikasiBerita/Label kelas unt traning dan testing set/Training set.txt'
PATHLABELTEST = '/home/helmisatria/@FALAH/KlasifikasiBerita/Label kelas unt traning dan testing set/Testing set.txt'
```

Load DATASET

```
LabelTrain   = np.genfromtxt(PATHLABEL, delimiter=' ', dtype=str)[:, 1][:TRAINCOUNT]
LabelTest    = np.genfromtxt(PATHLABELTEST, delimiter=' ', dtype=str)[:, 1][:TRAINCOUNT]
AllDocuments = getData(PATHTRAINING)
isTesting=False
# ================================================================================
#
# Testing?
#
#LabelTrain = LabelTest
#AllDocuments = getData(PATHTESTING)
#isTesting=True
#
```

Function GetPathData

```
def getData(path):
    allDocuments = []
    for filename in os.listdir(path):
        itemIds = ''
        isiBerita = ''
        if not filename.endswith('.xml'): continue
        fullname = os.path.join(path, filename)
        tree = ET.parse(fullname)
        root = tree.getroot()

        childRoot = []
        for child in root:
            childRoot.append(child.tag)

        indexHeadline = childRoot.index('headline')
        indexText = childRoot.index('text')

        itemIds = root.attrib.get('itemid')
        isiBerita = root[indexHeadline].text

        for p in (root[indexText]):
            isiBerita += p.text
        allDocuments.append([itemIds, isiBerita])

#         allDocuments.append(document)

    return sorted(allDocuments, key=lambda doc: doc[0])[:TRAINCOUNT]
```

Function Cleaning Document

```
def Cleaning(AllDocuments):
    CLEANDOCUMENTS = []
    for i, document in enumerate(AllDocuments):
        cleanContent = []
        for j, content in enumerate(document):
            if j == 0 :
                cleanContent.append(content)
                continue
            stopWords = set(stopwords.words('english'))
            cleanWords = (re.sub(r'[.,\/#!$%\^&\*;:{}=\-_+`~()\'\"{0-9}]', ' ',content.lower()))
            words = word_tokenize(cleanWords)
            wordsFilteredStemmed = []

            for w in words:
                if w not in stopWords:
                    wordsFilteredStemmed.append(stemming.stem(w))

            cleanContent.append(wordsFilteredStemmed)

        CLEANDOCUMENTS.append(cleanContent)

    return CLEANDOCUMENTS
```

Function GetCorpus

```
def CORPUS(Data):
    WORDS = []
    for i, document in enumerate(Data):
        for j, content in enumerate(document):
            if j == 0: continue
            for k, word in enumerate(content):
                WORDS.append(word)
    return WORDS
```

Function get TF and IDF

```
def TF(Data, CORPUS):
    RESULT = []
    c = Counter(CORPUS)
    for i, document in enumerate(Data):
        dataTF = []
        dataIDF = []
        for j, word in enumerate(CORPUS):
            countWord = document[1].count(word)
            tf = countWord / len(document[1])

            countWordAllDocs = c[word]
            idf = tf * np.log(len(Data) / countWordAllDocs)

            dataTF.append(tf)
            dataIDF.append(idf)
        RESULT.append([dataTF, dataIDF])
    return [item[0] for item in RESULT], [item[1] for item in RESULT]
```

Function NaiveBayes Clasifier

```
def NaiveBayes():
    content = [item[1] for item in AllDocumentsTokenized][:TRAINCOUNT]
    concatenatedContent = []
    for i, cont in enumerate(content):
        concatenatedContent.append(' '.join(map(str, cont)))
    vectorizer = CountVectorizer(concatenatedContent)
    Generated = vectorizer.fit_transform(concatenatedContent).toarray()

    ids = [item[0] for item in AllDocumentsTokenized]

    clf = GaussianNB()
    clf.fit(Generated, LabelTrain)

    result = []
    if (isTesting == False):
        result = cross_val_score(clf, Generated, LabelTrain, cv=10)

        concatenated = np.column_stack((ids, clf.predict(Generated)))
    else:
        result = Accuracy(clf.predict(Generated), LabelTrain)
        concatenated = np.column_stack((ids, clf.predict(Generated)))

    return result, concatenated
```

Function SVM Clasifier

```
def SVM():
    ids = [item[0] for item in AllDocumentsTokenized]
    clf = svm.SVC(C=100, gamma=0.1)
    clf.fit(TfIdf, LabelTrain)
    result = []
    if (isTesting == False):
        result = cross_val_score(clf, TfIdf, LabelTrain, cv=10)

        concatenated = np.column_stack((ids, clf.predict(TfIdf)))
    else:
        result = Accuracy(clf.predict(TfIdf), LabelTrain)
        concatenated = np.column_stack((ids, clf.predict(TfIdf)))

    return result, concatenated
```

Function getAccurcy

```
def Accuracy(Data, Validator):
    count = 0
    for i, x in enumerate(Data):
        if (x==Validator[i]): count+= 1
    return count/len(Data)
```

Running Function

Running Function

```
#If Data=Training
LabelTrain   = np.genfromtxt(PATHLABEL, delimiter=' ', dtype=str)[:, 1][:TRAINCOUNT]
LabelTest    = np.genfromtxt(PATHLABELTEST, delimiter=' ', dtype=str)[:, 1][:TRAINCOUNT]
AllDocuments = getData(PATHTRAINING)
isTesting=False

#If Data=Testing
LabelTrain = LabelTest
AllDocuments = getData(PATHTESTING)
isTesting=True

#Preprocessing (Lower,Tokenize, Remove Stop Words, Stemming
AllDocumentsTokenized = Cleaning(AllDocuments)
Corpus                = set(CORPUS(AllDocumentsTokenized))
Tf  , TfIdf           = TF(AllDocumentsTokenized, Corpus)

#Running Klasifier and accuracies
#1. SVM
AccuracySVM, SVMTrain = SVM()
#2. Naive Bayes
AccuracyNaiveBayes, NaiveBayesTrain = NaiveBayes()
```