Assignment_2_B

May 1, 2025

1 ΕΠΕΞΕΡΓΑΣΙΑ ΦΥΣΙΚΗΣ ΓΛΩΣΣΑΣ - Ε 2

1.1 B. Traditional Text Classification

Μ : Ε Φ Γ

Σ : Ι Κ

Ε : 2025-04-14 | v.0.0.1

[51]:
```
# Ε datasets
%pip install datasets wordcloud --quiet
```

```
Note: you may need to restart the kernel to use updated packages.
```

[52]:
```
from datasets import load_dataset
import pandas as pd

# Φ AG News dataset
dataset = load_dataset("ag_news")

# Μ DataFrame
df_train = pd.DataFrame(dataset['train'])
df_test = pd.DataFrame(dataset['test'])
```

[53]:
```
# Ε
print('-----------------------------------------------------------------')
print(df_train.head())
print('-----------------------------------------------------------------')
print(df_train['label'].value_counts())  # 0=World, 1=Sports, 2=Business, 3=Sci/
 ↪Tech
print('-----------------------------------------------')
print(dataset['train'].features['label'].names)
print('-----------------------------------------------------------------')
print(dataset['train'].column_names)
print('-----------------------------------------------------------------')
print(f"Train size: {len(df_train)}")
print('-----------------------------------------------------------------')
print(f"Test size: {len(df_test)}")
```

```
------------------------------------------------------------------
                                                text  label
0  Wall St. Bears Claw Back Into the Black (Reute…      2
1  Carlyle Looks Toward Commercial Aerospace (Reu…      2
2  Oil and Economy Cloud Stocks' Outlook (Reuters…      2
3  Iraq Halts Oil Exports from Main Southern Pipe…      2
4  Oil prices soar to all-time record, posing new…      2
------------------------------------------------------------------
label
2    30000
3    30000
1    30000
0    30000
Name: count, dtype: int64
------------------------------------------
['World', 'Sports', 'Business', 'Sci/Tech']
------------------------------------------------------------------
['text', 'label']
------------------------------------------------------------------
Train size: 120000
------------------------------------------------------------------
Test size: 7600
```

[54]:
```python
df_train = pd.DataFrame(dataset['train'])
df_test = pd.DataFrame(dataset['test'])

# A   lowercase
X_train = df_train['text'].str.lower()
X_test = df_test['text'].str.lower()

y_train = df_train['label']
y_test = df_test['label']


# E

label_names = dataset['train'].features['label'].names

for i in range(3):
    print(f"Sample {i+1}")
    print("Text:", X_train.iloc[i])
    print("Label:", label_names[y_train.iloc[i]])
    print("-" * 50)
```

```
Sample 1
Text: wall st. bears claw back into the black (reuters) reuters - short-sellers,
wall street's dwindling\band of ultra-cynics, are seeing green again.
Label: Business
```

```
--------------------------------------------------
Sample 2
Text: carlyle looks toward commercial aerospace (reuters) reuters - private
investment firm carlyle group,\which has a reputation for making well-timed and
occasionally\controversial plays in the defense industry, has quietly placed\its
bets on another part of the market.
Label: Business
--------------------------------------------------
Sample 3
Text: oil and economy cloud stocks' outlook (reuters) reuters - soaring crude
prices plus worries\about the economy and the outlook for earnings are expected
to\hang over the stock market next week during the depth of the\summer doldrums.
Label: Business
--------------------------------------------------
```

1.1.1 E 1 – ϒ Π T

Σ scikit-learn.
Σ , (Multinomial Naive Bayes SVM kernel)
(TF-IDF word 1-grams TF-IDF character 3-grams).

[55]:
```
#O          benchmarking
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import LinearSVC
from sklearn.pipeline import make_pipeline
from sklearn.metrics import accuracy_score
from time import time

def evaluate_model(ngram_type='word', ngram_range=(1,1), model_type='nb'):
    analyzer = 'char' if ngram_type == 'char' else 'word'
    vectorizer = TfidfVectorizer(analyzer=analyzer, ngram_range=ngram_range)

    if model_type == 'nb':
        model = MultinomialNB()
    elif model_type == 'svm':
        model = LinearSVC(C=1)

    pipeline = make_pipeline(vectorizer, model)

    start = time()
    pipeline.fit(X_train, y_train)
    duration = time() - start

    predictions = pipeline.predict(X_test)
    acc = accuracy_score(y_test, predictions)

    dim = len(pipeline.named_steps['tfidfvectorizer'].get_feature_names_out())
```

```
    return acc, dim, round(duration, 2), predictions
```

[56]:
```
# Σ                4

results = []

# 1. NB - word unigrams
acc, dim, time_sec, _ = evaluate_model(ngram_type='word', ngram_range=(1,1),␣
 ↪model_type='nb')
results.append(('NB (word unigrams)', acc, dim, time_sec))

# 2. NB - char trigrams
acc, dim, time_sec, _ = evaluate_model(ngram_type='char', ngram_range=(3,3),␣
 ↪model_type='nb')
results.append(('NB (char trigrams)', acc, dim, time_sec))

# 3. SVM - word unigrams
acc, dim, time_sec, _ = evaluate_model(ngram_type='word', ngram_range=(1,1),␣
 ↪model_type='svm')
results.append(('SVM (word unigrams)', acc, dim, time_sec))

# 4. SVM - char trigrams
acc, dim, time_sec, _ = evaluate_model(ngram_type='char', ngram_range=(3,3),␣
 ↪model_type='svm')
results.append(('SVM (char trigrams)', acc, dim, time_sec))
```

1.1.2 E 2 – Σ A M

Σ , : (accuracy), (dimensionality) (training time).

T .

[57]:
```
import pandas as pd

# Δ            DataFrame   reset    index
df_results = pd.DataFrame(results, columns=["Model", "Accuracy",␣
 ↪"Dimensionality", "Time Cost (sec)"]).reset_index(drop=True)

# Δ      transposed
df_results_named = pd.DataFrame({
    "NB (word 1-grams)":    df_results.loc[0, ["Accuracy", "Dimensionality",␣
 ↪"Time Cost (sec)"]],
    "NB (char 3-grams)":    df_results.loc[1, ["Accuracy", "Dimensionality",␣
 ↪"Time Cost (sec)"]],
```

```
    "SVM (word 1-grams)":   df_results.loc[2, ["Accuracy", "Dimensionality",␣
 ↪"Time Cost (sec)"]],
    "SVM (char 3-grams)":   df_results.loc[3, ["Accuracy", "Dimensionality",␣
 ↪"Time Cost (sec)"]],
})


# Formatting: accuracy 2 dec, dim with comma, time 2 dec
df_results_named.loc["Accuracy"] = df_results_named.loc["Accuracy"].
 ↪astype(float).map("{:.2f}".format)
df_results_named.loc["Dimensionality"] = df_results_named.loc["Dimensionality"].
 ↪astype(int).map("{:,}".format)
df_results_named.loc["Time Cost (sec)"] = df_results_named.loc["Time Cost␣
 ↪(sec)"].astype(float).map("{:.2f}".format)


# Π
df_results_named
```

[57]:

	NB (word 1-grams)	NB (char 3-grams)	SVM (word 1-grams)	\
Accuracy	0.90	0.87	0.92	
Dimensionality	65,006	31,072	65,006	
Time Cost (sec)	1.22	5.16	6.90	

	SVM (char 3-grams)
Accuracy	0.91
Dimensionality	31,072
Time Cost (sec)	20.96

1.1.3 E 2 – Σ A

A :

- H **(accuracy)** , **SVM** (92%). H , , .
- O **Naive Bayes** , **10** SVM.
- H **(dimensionality)** n-gram:
  - T **character 3-grams** **50%** word 1-grams.
  - Δ NB SVM vectorizer.
- A ( ), **Naive Bayes** **word 1-grams** **“value for performance”**.
- E , , , **SVM**, .

1.1.4 E 3 – A Λ Π

Σ (Naive Bayes SVM word unigrams char trigrams) .

Γ : - Σ test set. - Φ
. - X , label
. - O **Word Cloud**,
.

H , .

[58]:
```
# Π
_, _, _, pred_nb_word = evaluate_model('word', (1,1), 'nb')
_, _, _, pred_nb_char = evaluate_model('char', (3,3), 'nb')
_, _, _, pred_svm_word = evaluate_model('word', (1,1), 'svm')
_, _, _, pred_svm_char = evaluate_model('char', (3,3), 'svm')
```

[59]:
```
import numpy as np

# M              vectorized
y_true = y_test.to_numpy()

# M
err_nb_word = pred_nb_word != y_true
err_nb_char = pred_nb_char != y_true
err_svm_word = pred_svm_word != y_true
err_svm_char = pred_svm_char != y_true

# K     :
all_wrong_mask = err_nb_word & err_nb_char & err_svm_word & err_svm_char
```

[62]:
```
# Φ
df_errors = df_test[all_wrong_mask].copy()

# Π            label (   )
df_errors['true_category'] = df_errors['label'].apply(lambda x: label_names[x])

# Π                        (       →     )
df_errors['nb_word'] = pred_nb_word[all_wrong_mask]
df_errors['nb_char'] = pred_nb_char[all_wrong_mask]
df_errors['svm_word'] = pred_svm_word[all_wrong_mask]
df_errors['svm_char'] = pred_svm_char[all_wrong_mask]

# M                 labels
df_errors['nb_word'] = df_errors['nb_word'].apply(lambda x: label_names[x])
df_errors['nb_char'] = df_errors['nb_char'].apply(lambda x: label_names[x])
df_errors['svm_word'] = df_errors['svm_word'].apply(lambda x: label_names[x])
df_errors['svm_char'] = df_errors['svm_char'].apply(lambda x: label_names[x])

df_errors[['text', 'true_category', 'nb_word', 'nb_char', 'svm_word',␣
 ↪'svm_char']] #.head()
```

[62]:
```
                                                   text true_category  \
24    Rivals Try to Turn Tables on Charles Schwab By…      Sci/Tech
56    India's Tata expands regional footprint via Na…         World
79    Live: Olympics day four Richard Faulds and Ste…         World
83    Intel to delay product aimed for high-definiti…      Business
106   Stocks Climb on Drop in Consumer Prices NEW YO…         World
…                                                   …             …
7413  Bush Pledges Strong-Dollar Policy President Bu…      Business
7470  Broadband charges set to tumble The cost of br…      Business
7492  FCC Mulls Airborne Mobile Phone Use Although t…      Business
7539  Mars water tops science honours The discovery …         World
7585  Pricey Drug Trials Turn Up Few New Blockbuster…         World

       nb_word   nb_char  svm_word  svm_char
24    Business  Business  Business  Business
56    Business  Business  Business  Business
79      Sports    Sports    Sports    Sports
83    Sci/Tech  Sci/Tech  Sci/Tech  Sci/Tech
106   Business  Business  Business  Business
…          …         …         …         …
7413     World     World     World     World
7470  Sci/Tech  Sci/Tech  Sci/Tech  Sci/Tech
7492  Sci/Tech  Sci/Tech  Sci/Tech  Sci/Tech
7539  Sci/Tech  Sci/Tech  Sci/Tech  Sci/Tech
7585  Business  Business  Business  Business

[341 rows x 6 columns]
```

[64]:
```python
from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt

# Σ    Λ
print(f"Σ                  : {len(df_errors)}")

# A
error_counts = df_errors['true_category'].value_counts()
print("\nK               :")
print(error_counts)

# Stopwords +
text_data = ' '.join(df_errors['text'])
custom_stopwords = set(STOPWORDS)
custom_stopwords.update(['said', 'say', 'will', 'quot', 'may', 'monday',␣
 ↪'tuesday', 'reuters', 'ap', 'new', 'york'])

text_data_clean = ' '.join([w for w in text_data.split() if len(w) > 2])
```

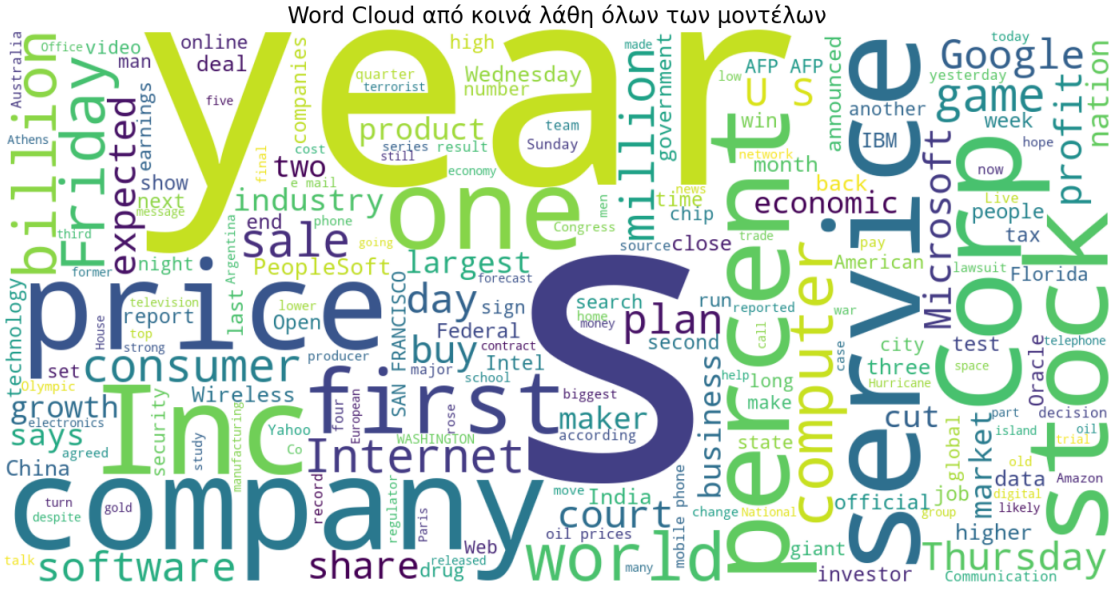
```
# Δ      word cloud
wc = WordCloud(
    width=1280, height=640,
    background_color='white',
    max_words=200,
    colormap='viridis',
    stopwords=custom_stopwords
).generate(text_data_clean)

# Π
plt.figure(figsize=(18, 9))
plt.imshow(wc, interpolation='bilinear')
plt.axis('off')
plt.title("Word Cloud                        ", fontsize=20)
plt.show()
```

```
Σ                    : 341

K                 :
true_category
Business     135
World        112
Sci/Tech      85
Sports         9
Name: count, dtype: int64
```

Word Cloud από κοινά λάθη όλων των μοντέλων



1.1.5 Σ – E 3

Σ test set . H
:

- **O** **“World”** **“Business”**, :
  –
  –
  – (. .)
- A , **“Sports”** ,
  (, , . .).
- T **Word Cloud** :
  `company`, `year`, `service`, `corp`, `price`, `percent`,
  , .
- Π , ( ) .
  A **ensemble** **(majority vote)** ,
  .