Lab 5 - Classification - ML - Sentiment Analysis

```
import pandas as pd
```

```
seed = 42
```

```
df = pd.read_csv('/content/Dataset - Classification - Sentences_AllAgree.csv',
                 sep='.@',
                 engine='python')
```

```
df.head()
```

	text	label
0	Russia , although that is where the company is...	neutral
1	For the last quarter of 2010 , Componenta 's n...	positive
2	In the third quarter of 2010 , net sales incre...	positive
3	Operating profit rose to EUR 13.1 mn from EUR ...	positive
4	Operating profit totalled EUR 21.1 mn , up fro...	positive

```
df.describe()
```

	text	label
count	2264	2264
unique	2259	3
top	SSH Communications Security Corporation is hea...	neutral
freq	2	1391

```
X = df['text']
#.astype(str)
y = df['label']
```

```
y.value_counts()
```

label	count
neutral	1391
positive	570
negative	303

dtype: int64

```
y.value_counts(normalize=True)
```

label	proportion
neutral	0.614399
positive	0.251767
negative	0.133834

dtype: float64

```
#df['label'] == 'neutral' # compatori >, <, ==
df_neutral = df[df['label'] == 'neutral']
df_neutral_sampled = df_neutral.sample(303, random_state=seed)

df_postive_sampled = df[df['label'] == 'positive'].sample(303, random_state=seed)
df_negative_sampled = df[df['label'] == 'negative']
```

```
df_balanced = pd.concat([df_neutral_sampled, df_postive_sampled, df_negative_sampled])
df_balanced.shape
```

```
(909, 2)
```

```
df_postive_sampled.head()
```

	text	label
853	These measures are expected to produce annual ...	positive
75	The last quarter was the best quarter of 2009 ...	positive
136	In the fourth quarter of 2009 , Orion 's net p...	positive
465	Operating profit improved by 44.0 % to ER 4.7 ...	positive
887	Tiimari Latvian representative Ineta Zaharova ...	positive

```
df_balanced['label'].value_counts()
```

label	count
neutral	303
positive	303
negative	303

dtype: int64

```
from sklearn.model_selection import train_test_split

X = df_balanced['text']
y = df_balanced['label'] # negative, neutral, postive

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state=seed, stratify=y)
```

```
y_test.value_counts()
```

label	count
neutral	61
positive	61
negative	60

dtype: int64

```
# "Operating profit rose"
# ungirams: Operating, profit, rose
# bigrams: Operating profit, profit rose
# trigrams: Operating profit rose

from sklearn.feature_extraction.text import CountVectorizer
vec  = CountVectorizer(ngram_range=(1,3), stop_words='english') # Try: keeping upper case
#vec  = TfidfVectorizer(max_features=2000, ngram_range=(1,3), stop_words='english')
X_train_vec = vec.fit_transform(X_train)
X_test_vec = vec.transform(X_test)


#from sklearn.feature_extraction.text import TfidfVectorizer
#vec  = TfidfVectorizer(ngram_range=(1,3), stop_words='english'
                       #, lowercase = False
#                       ) # Try: keeping upper case
#vec  = TfidfVectorizer(max_features=2000, ngram_range=(1,3), stop_words='english')
#X_train_vec = vec.fit_transform(X_train)
#X_test_vec = vec.transform(X_test)
```

```
X_train
```

	text
2061	`` We 're delighted with the move '' says Morn...
2004	The floor area of the Yliopistonrinne project ...
751	Mika Stahlberg , VP F-Secure Labs , said , `` ...
1672	ADPnews - Dec 23 , 2009 - Norwegian financial ...
697	The deal was worth about EUR 1.2 mn
...	...
1269	ISMS does not disclose its financial results ,...
86	At the end of March 2007 , the group 's order ...
243	Profit per share was EUR 1.03 , up from EUR 0.87
118	Finnish lifting equipment maker Konecranes Oyj...
201	The Finnish government announced Wednesday tha...

727 rows × 1 columns

dtype: object

```
print(X_train_vec[0])
```

```
<Compressed Sparse Row sparse matrix of dtype 'int64'
        with 30 stored elements and shape (1, 15811)>
  Coords        Values
  (0, 4439)     1
  (0, 13116)    1
  (0, 9727)     1
  (0, 4097)     1
  (0, 10652)    1
  (0, 4760)     1
  (0, 13718)    1
  (0, 4156)     1
  (0, 13624)    1
  (0, 8715)     1
  (0, 6257)     1
  (0, 4440)     1
  (0, 13119)    1
  (0, 9728)     1
  (0, 4098)     1
  (0, 10653)    1
  (0, 4761)     1
  (0, 13719)    1
  (0, 4170)     1
  (0, 13629)    1
  (0, 8716)     1
  (0, 4441)     1
  (0, 13120)    1
  (0, 9729)     1
  (0, 4099)     1
  (0, 10654)    1
  (0, 4762)     1
  (0, 13720)    1
  (0, 4171)     1
  (0, 13630)    1
```

```
# LogisticRegression - Clasificare
# https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
from sklearn.linear_model import LogisticRegression

model = LogisticRegression()
model.fit(X_train_vec, y_train)
```

▾ LogisticRegression

```
LogisticRegression()
```

```
y_pred = model.predict(X_test_vec)
#y_pred
#print(X_test.head(2))
```

```
from sklearn.metrics import accuracy_score
acc = accuracy_score(y_test, y_pred)
print(acc)
```

```
0.7472527472527473
```

```
from sklearn.metrics import classification_report
classification_report(y_test, y_pred)
```

```
'              precision    recall  f1-score   support\n\n    negative       0.71      0.85      0.77        60\n     neutr
al       0.76      0.87      0.81        61\n    positive       0.80      0.52      0.63        61\n\n    accuracy
0.75       182\n   macro avg       0.76      0.75      0.74       182\nweighted avg       0.76      0.75      0.74       18
2\n'
```

MultinomialNB, LinearSVC and Random Forest

```
# MultinomialNB
# https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html
from sklearn.naive_bayes import MultinomialNB
model = MultinomialNB()
model.fit(X_train_vec, y_train) # trainning
y_pred = model.predict(X_test_vec) # test
acc = accuracy_score(y_test, y_pred)
print(acc)
```

```
0.6923076923076923
```

```
# LinearSVC
# https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html
from sklearn.svm import LinearSVC
model = LinearSVC()
# model = LinearSVC(penalty='l1') # try to use different hyperparameters (penalty can be l1 or l2)
model.fit(X_train_vec, y_train) # trainning
y_pred = model.predict(X_test_vec) # test
acc = accuracy_score(y_test, y_pred)
print(acc)
#
```

```
0.7692307692307693
```

```
# Random Forest
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier(random_state=seed)
model.fit(X_train_vec, y_train) # trainning
y_pred = model.predict(X_test_vec) #test
acc = accuracy_score(y_test, y_pred)
print(acc)
```

```
0.7032967032967034
```

Grid Search for hyperparameter values

Note:

- parameter `C` below, does not have a big impact
- parameter `loss` below, can cause trainning errors

```
from sklearn.model_selection import GridSearchCV
from sklearn.svm import LinearSVC

# Define the parameter grid to search
param_grid = {
    'C': [0.1, 0.5, 1, 2, 3, 3.5, 3.9],  # Regularization parameter
    #'loss': ['hinge', 'squared_hinge'], # Loss function
    'penalty': ['l1', 'l2'], # Penalty norm
}

# Create a LinearSVC model
model = LinearSVC(random_state=seed)

# Create GridSearchCV object
grid_search = GridSearchCV(model, param_grid, cv=5, scoring='accuracy')

# Fit the grid search to the data
grid_search.fit(X_train_vec, y_train)

# Print the best parameters and the best score
print("Best parameters:", grid_search.best_params_)
print("Best cross-validation accuracy:", grid_search.best_score_)

# Evaluate the best model on the test set
best_model = grid_search.best_estimator_
y_pred = best_model.predict(X_test_vec)
acc = accuracy_score(y_test, y_pred)
print("Test set accuracy with best parameters:", acc)
```

```
/usr/local/lib/python3.12/dist-packages/sklearn/svm/_base.py:1249: ConvergenceWarning: Liblinear failed to converge, increas
  warnings.warn(
/usr/local/lib/python3.12/dist-packages/sklearn/svm/_base.py:1249: ConvergenceWarning: Liblinear failed to converge, increas
  warnings.warn(
Best parameters: {'C': 3, 'penalty': 'l1'}
Best cross-validation accuracy: 0.7455550307038261
Test set accuracy with best parameters: 0.7857142857142857
/usr/local/lib/python3.12/dist-packages/sklearn/svm/_base.py:1249: ConvergenceWarning: Liblinear failed to converge, increas
  warnings.warn(
```

```
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier

# Define the parameter grid to search
param_grid = {
    #'n_estimators': [100, 200, 300],  # Number of trees in the forest
    'n_estimators': [199, 200, 201],  # Number of trees in the forest
    'max_depth': [None, 10, 20, 30],  # Maximum depth of each tree
}

# Create a LinearSVC model
model = RandomForestClassifier(random_state=seed)

# Create GridSearchCV object
grid_search = GridSearchCV(model, param_grid, cv=5, scoring='accuracy')

# Fit the grid search to the data
grid_search.fit(X_train_vec, y_train)

# Print the best parameters and the best score
print("Best parameters:", grid_search.best_params_)
print("Best cross-validation accuracy:", grid_search.best_score_)

# Evaluate the best model on the test set
best_model = grid_search.best_estimator_
y_pred = best_model.predict(X_test_vec)
acc = accuracy_score(y_test, y_pred)
print("Test set accuracy with best parameters:", acc)
```

```
Best parameters: {'max_depth': None, 'n_estimators': 200}
Best cross-validation accuracy: 0.6863769485120453
Test set accuracy with best parameters: 0.7087912087912088
```