Arabic Dialect Identification

1. Problem Description

Arabic language still considered by Natural Language Processing practitioners as a low resource language. However, the Arabic language is spoken in twenty-two countries by more than 250 million speakers.

Formal sources of Arabic texts are typically written in Modern Standard (or Written) Arabic (MSA), which is a form that is used in formal writing and taught in schools to Arabic speakers. However, informal communication among Arabic speakers is through informal local Diglossic dialects. A Diglossic language is one where the speakers of the same language have varying dialects. In Arabic, there are multiple dialects in different regions of the Arab world: Gulf, Levantine and North Africa. Users commonly communicate in social media using their local dialect rather than the formal MSA.

Dialect Identification (DID) is the task of automatically identifying the dialect of a particular segment of speech or text of any size. In this project we worked on Text.

Automatic DID is very important for several NLP tasks where prior knowledge about the dialect of an input text can be helpful, such as machine translation, sentiment analysis or author profiling.

2. Model Design

In this section we will discuss the used dataset and the preprocessing techniques used on it. And the traditional classifiers applied to solve our problem.

2.1. Dataset: Arabic Online Commentary

Our work is based on the AOC dataset. AOC is composed of 3M MSA and dialectal comments, of which 108, 173 comments are labeled via crowdsourcing (Egyptian, Gulf, Levantine). For our experiments, we randomly shuffle the dataset and split it into 80% training (Train), and 20% test (Test).

2.2. Preprocessing

We removed the Arabic stop words and Arabic propositions after tokenizing the texts based on white space, excluding all the non-Unicode characters. Then, we normalized the data by substituting Alf (أ آ إ) to Alf(ا), yaa(ي) to yaa(ى), hamza (ء) to (ئ), haa(ه) to taa marbota(ة), and kaf(ك) to (گ). Then, we applied stemming using Tashaphyre library, and lemmatization by Farasa API.

To improve the model Accuracy we removed any duplicated data from our training set.

2.3. Classifiers Experiments

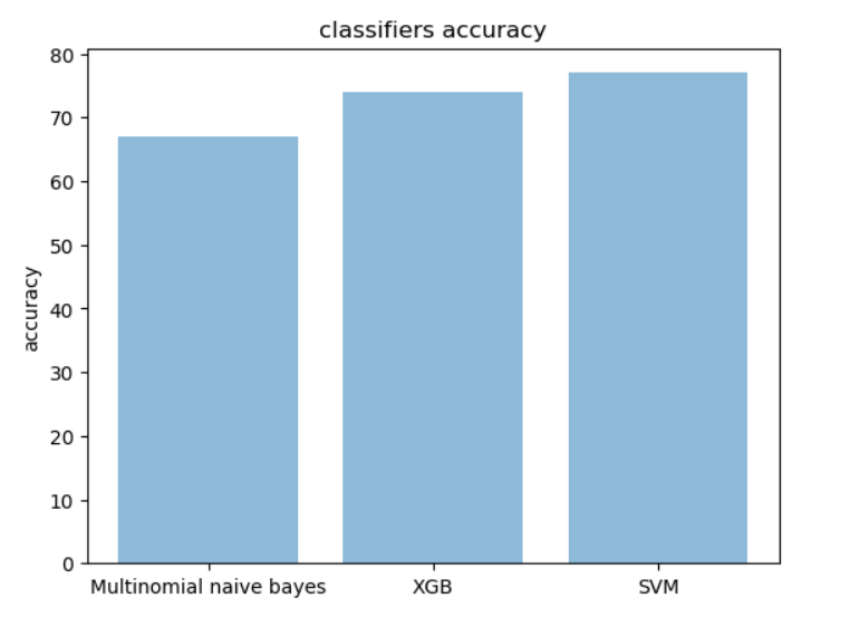
At first we build our bag of words using the countVectorizer and to make our model focus on the most distinct words for each language we built our TF_IDF from this bag of words. Then, we applied Multinomial Naïve bayes classifier, SVM classifier, and XGB classifier. From Sklearn library and Xgboost.

3. Experimental Results

Classifier(unigram)	accuracy
Multinomial naïve bayes	**67.234125**
Support vector machine	**76.82**
XGB	**73.84**

As shown in the previous table. Among traditional models, the support vector machine achieves the best performance. Next, comes the XGB classifier.

This figure shows the accuracy between the three models we implemented.



in the below snapshots we show the results of each classifier we applied.

```
sentence = ['ياخي شلون تبي تحقق مونديال وانت بدونه ضايع']

#sentence = ["   يالبت والله علشان  يربحنا"]
#sentence = ["يا راجل فن ايه هو ده فن دى قلة ادب"]
#sentence = ["انا مش عارف انا ايه جابني هنا"]
# ياخي شلون تبي تحقق مونديال وانت بدونه ضايع  gulf مادري شفيني عليه
#ليه بتقول كده علينا طيب يعم انت مالك اصلا؟
X_testing_counts = count_vect.transform(sentence)
X_tfidf_testing = tfidf_transformer.transform(X_testing_counts)

#SVM_from_pickle.predict(X_tfidf_testing)
clf.predict(X_tfidf_testing)
```

```
array(['DIAL_GLF'], dtype='<U8')
```

The prediction from the multinomial naïve bayes classifier.

```
sentence = ["ليه بتقول كده علينا طيب يعم انت مالك اصلا؟"]

#sentence = ["   يالبت والله علشان  يربحنا"]
#sentence = ["يا راجل فن ايه هو ده فن دى قلة ادب"]
#sentence = ["انا مش عارف انا ايه جابني هنا"]
# ياخي شلون تبي تحقق مونديال وانت بدونه ضايع  gulf مادري شفيني عليه
#ليه بتقول كده علينا طيب يعم انت مالك اصلا؟
X_testing_counts = count_vect.transform(sentence)
X_tfidf_testing = tfidf_transformer.transform(X_testing_counts)

#SVM_from_pickle.predict(X_tfidf_testing)
svm_clf.predict(X_tfidf_testing)
```

```
array(['DIAL_EGY'], dtype='<U8')
```

The prediction from the SVM classifier.

4. Model Performance

As we discussed in the previous section, we achieved the best accuracy using the SVM classifier with approximate 77%. As shows in the figure below we can deduce the model performance from the classification report.

	Precision	recall	F1-score	support
DIAL_EGY	0.86	0.55	0.67	1899
DIAL_GLF	0.73	0.53	0.62	3078
DIAL_LEV	0.88	0.50	0.63	1790
MSA	0.76	0.96	0.84	8682
Accuracy			0.77	15449
Macro avg	0.80	0.63	0.69	15449
Weighted avg	0.78	0.77	0.75	15449

The accuracy of the classifier in classifying the data points DIAL_EGY is equal 86%, DIAL_GLF is equal 73%, DIAL_LEV is equal 88% and MSA equal 76%. The number of samples of the true response that lies in Egyptian dialect class are 1899 samples, Gulf dialect class are 3078 samples, Levantine dialect class are 1790 samples, and MSA class are 8682 samples.

The accuracy of the model using the multinomial naïve bayes classifier as we deduced before is approximately equals 67%. In the below table it shown the model performance using it as a classifier.

	Precision	recall	F1-score	support
DIAL_EGY	0.93	0.27	0.41	1899
DIAL_GLF	0.80	0.31	0.44	3078
DIAL_LEV	0.93	0.19	0.31	1790
MSA	0.64	0.99	0.78	8682
Accuracy			0.67	15449
Macro avg	0.83	0.44	0.49	15449
Weighted avg	0.74	0.67	0.61	15449

From comparing these two reports we can conclude that the best performance we get from the support vector machine classifier.

5. Model deployment

We saved the model using Pickle file. Then, using Flask API to deploy the model on Heroku app. You can find our life application link below.

https://arabic-dialect-identification2.herokuapp.com/

note: this deployed based on the multinomial Naïve bayes Classifier.