intent-location-classify-single-file-final

June 19, 2022

[1]:

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import os
import spacy   # import the spacy nlp
import csv
from sklearn.preprocessing import LabelEncoder # for label encoding
from sklearn.svm import SVC
from sklearn.metrics import classification_report #for evaluation

# Change this path to path of task_dataset
base_path = '/home/manju/Desktop/assign/task_data/'

def readInputFile(csv_file):
    data = pd.read_csv(csv_file)
    transcript = data['transcription'].str.lower()  # convert to lower case
    location = data['location'].str.lower().str.replace(" ", "_")  # convert to␣
 ↪lower case then replace spaces by _
    return list(transcript.str.lower()), list(location) #Convert to list type &␣
 ↪return

## encode transcription to vec foramat using spacy package
def encode_sentences(sentences, embedding_dim, nlp):
    # Calculate number of sentences
    n_sentences = len(sentences)
    print('Number of sentences :-',n_sentences)
    X = np.zeros((n_sentences, embedding_dim))

    # Iterate over the sentences
    for idx, sentence in enumerate(sentences):
        # Pass each sentence to the nlp object to create a document
        doc = nlp(sentence)
        # Save the document's .vector attribute to the corresponding row in x
        X[idx, :] = doc.vector
    return X

### to convert string labels to integers
def label_encoding(labels):
    # Calculate the length of labels
    n_labels = len(labels)
    print('Number of labels :',n_labels)
    # instantiate labelencoder object
    le = LabelEncoder()
    y =le.fit_transform(labels)
    #print(y[:100])
    #print('Length of y : ',y.shape)
    return y

def svc_training(X,y):
    # Create a support vector classifier
    clf = SVC(C=1)

    # Fit the classifier using the training data
    clf.fit(X, y)
    return clf

def svc_validation(model,X,y):
    # Predict the labels of the test set
    y_pred = model.predict(X)

    # Count the number of correct predictions
    n_correct = 0
    for i in range(len(y)):
        if y_pred[i] == y[i]:
            n_correct += 1
    print("Predicted {0} correctly out of {1} training examples".
 ↪format(n_correct, len(y)))

def main(base_path):
    train_file = base_path + 'train_data.csv'
    valid_file = base_path + 'valid_data.csv'

    ################## DataSet preparation ###############
    sentences_valid,labels_valid = readInputFile(valid_file)
    sentences_train,labels_train = readInputFile(train_file)

    ### print unique elements in list ###
    print("Unique location labels in training data: ", set(labels_train),␣
 ↪"count ", len(set(labels_train)))
    print("Unique location labels in validataion data: ", set(labels_valid),␣
 ↪"count ", len(set(labels_valid)))

    print("Loading nlp spacy model :")

    # load nlp spacy model
    nlp = spacy.load('en_vectors_web_lg')

    # Calculate the dimensionality of nlp
    embedding_dim = nlp.vocab.vectors_length
    ###print(embedding_dim)

    print("Encoding train and validation sentences using spacy model")
    train_X = encode_sentences(sentences_train, embedding_dim, nlp)
    test_X = encode_sentences(sentences_valid, embedding_dim, nlp)

    print("Encoding labels to integers using skleran")
    train_y = label_encoding(labels_train)
    test_y = label_encoding(labels_valid)

    ###Intent classification with SVM | Training Step
    # X_train and y_train was given.
    print("Training SVM for Intent classification i.e predicting location using␣
 ↪transcription")
    model = svc_training(train_X,train_y)

    #Validation Step
    print("SVM Prediction Step: comparing predicted labels with correct labels")
    svc_validation(model,train_X,train_y)
    svc_validation(model,test_X,test_y)

    # Evaluation
    print("Evaluation")
    y_true, y_pred = test_y, model.predict(test_X)
    print(classification_report(y_true, y_pred))


### Invoking Main function
if __name__ == "__main__":
    main(base_path)
```

```
Unique location labels in training data:  {'washroom', 'kitchen', 'none',
'bedroom'} count  4
Unique location labels in validataion data:  {'washroom', 'kitchen', 'none',
'bedroom'} count  4
Loading nlp spacy model :
Encoding train and validation sentences using spacy model
Number of sentences :- 11566
Number of sentences :- 3118
Encoding labels to integers using skleran
Number of labels : 11566
Number of labels : 3118
Training SVM for Intent classification i.e predicting location using
```

```
transcription
SVM Prediction Step: comparing predicted labels with correct labels
Predicted 11566 correctly out of 11566 training examples
Predicted 3118 correctly out of 3118 training examples
Evaluation
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       327
           1       1.00      1.00      1.00       372
           2       1.00      1.00      1.00      1877
           3       1.00      1.00      1.00       542

    accuracy                           1.00      3118
   macro avg       1.00      1.00      1.00      3118
weighted avg       1.00      1.00      1.00      3118
```