3rd Assignment: Sentiment MLP Classifier for Movie Reviews

CONTRIBUTORS: DIMITRA TSAKIRI (f3352123), PANTELEIMON SFAKIANAKIS(f3352121), DIMITRIOS GKAVERAS(f3352104), ALEXANDROS SKONDRAS (f3352119)

The link for our Google Colab notebook can be found: here

Exercise 9

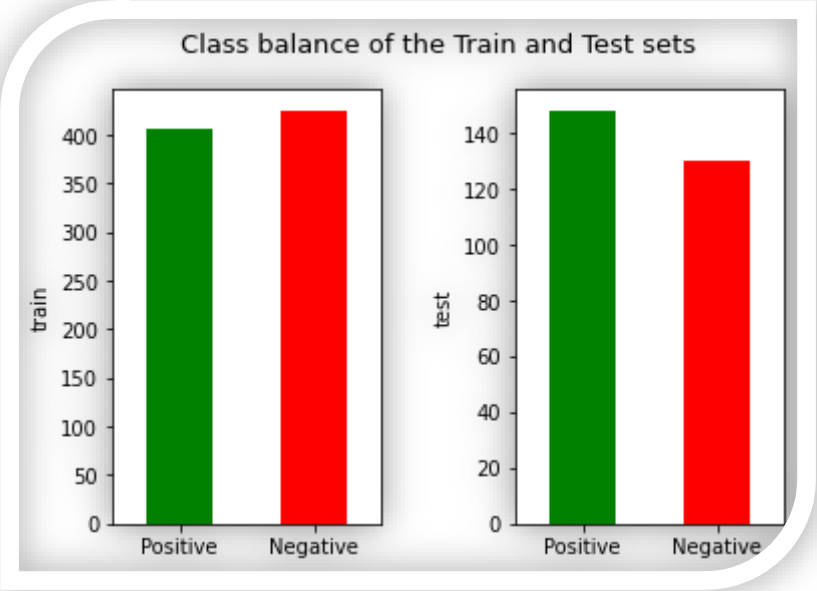
For the development of the sentiment classifier for this exercise, movie reviews were used from the Cornell Movie Review Data. More specifically, the dataset polarity dataset v1.1 was used, which contains 1386 movie reviews with average review length of 3560 characters, annotated into two mutually exclusive classes as either positive (1) or negative (0).

After downloading the files, the preprocessing of the reviews took place. Single characters and multiple spaces, numbers as well as non-word characters were all removed from the reviews. In addition, lemmatization was used to group the various inflected forms that a word might have as a single item. At this point, some statistics of the documents are shown in the following table.

Characters per Document		Characters per Negative Class			Characters per Positive Class		
Average	Vocabulary	Average	Min	Max	Average	Min	Max
3560	30797	3346	380	10109	3772	476	9919

The documents were randomly split into train, dev and test sets for the training of our classifiers, as shown below along with the class distribution:

Train	Test	Dev
831	278	277



Then, the features from the texts were extracted, using all the single words and bigrams and taking their TF-IDF scores. Stopwords were removed and the 5000 best features were selected according to their TF-IDF scores. Through the stated procedure, the most relevant words and bigrams (with the highest TF-IDF) score will be kept. Consequently, SVD was applied keeping eventually only 100 features, because the train set consists of only 831 reviews and keeping more features would potentially lead to overfitting. The dimensionality reduction that was achieved, is shown in the table of set shapes below:

Train		Test		Dev	
Before SVD	**After SVD**	**Before SVD**	**After SVD**	**Before SVD**	**After SVD**
(831, 5000)	(831, 100)	(278, 5000)	(278, 100)	(277, 5000)	(277, 100)

Baselines and MLP model

After the preprocessing phase, a baseline had to be set. For this task, the Dummy Classifier was used with 'most frequent' as strategy, which returns the most frequent class label as prediction for every test set item-review.

At this point, based on the best classifier of our previous analysis, an extra baseline was set, which is a Logistic Regressor Classifier. The parameters that we used were solver = "liblinear", C = 1, max_iter = 100, tol = 0.0001.

Since baselines were set, we implemented an MLP classifier using Keras/TensorFlow. Keras tuner-RandomSearch was used in order to hyper-parameter tune the architecture of the model. The MLP was of Sequential form with an input Dense layer of units 256 or 512 and "relu" activation. This was followed by a possible Dropout layer of given rates 0.2 or 0.35. Consequently, the option of 1 or 2 Dense layers with either 256 or 512 units and "relu" or "sigmoid" as activation, was examined followed by a possible Dropout with 0.5 rate after each aforementioned Dense layer. To close out the MLP a Dense layer of 1 unit and "sigmoid" activation is added. The model is compiled using binary cross-entropy loss and accuracy as metric. The objective of RandomSearch was set to minimize the validation loss and along the tuning, callbacks were applied. More particularly, EarlyStopping with patience=3 (number of max epochs tolerated without improving), min_delta=0.0001, monitoring the validation loss and restoring the best weights from the best performing epoch in each trial was implemented, along with ReduceLROnPlateau with patience=3 and cooldown=0, which reduces the learning rate when the appointed metric has stopped improving. The MLP was hyper-tuned with a maximum of 40 epochs per trial, so that it can train to a satisfactory level before it gets interrupted by EarlyStopping in every trial.

The result summary that defines the final architecture of the best model is showcased hereunder:

```
Showing 1 best trials
Objective(name='val_loss', direction='min')
Trial summary
Hyperparameters:
units: 512
activation: relu
dropout_1st: False
layers: 1
units0: 512
act_0: relu
dropout0: False
rate_1st: 0.2
rate0: 0.5
units1: 256
act_1: sigmoid
dropout1: True
rate1: 0.5
Score: 0.470020055770874
```

Note: If layers=1, then only parameters of index 0 are considered (matching the 1st layer). If layers=2, then index 1 parameters are also considered in the MLP.

In this instance of the MLP realization, we reach the model with the following architecture, which, of course, agrees with the hyper-parameters summary described above.

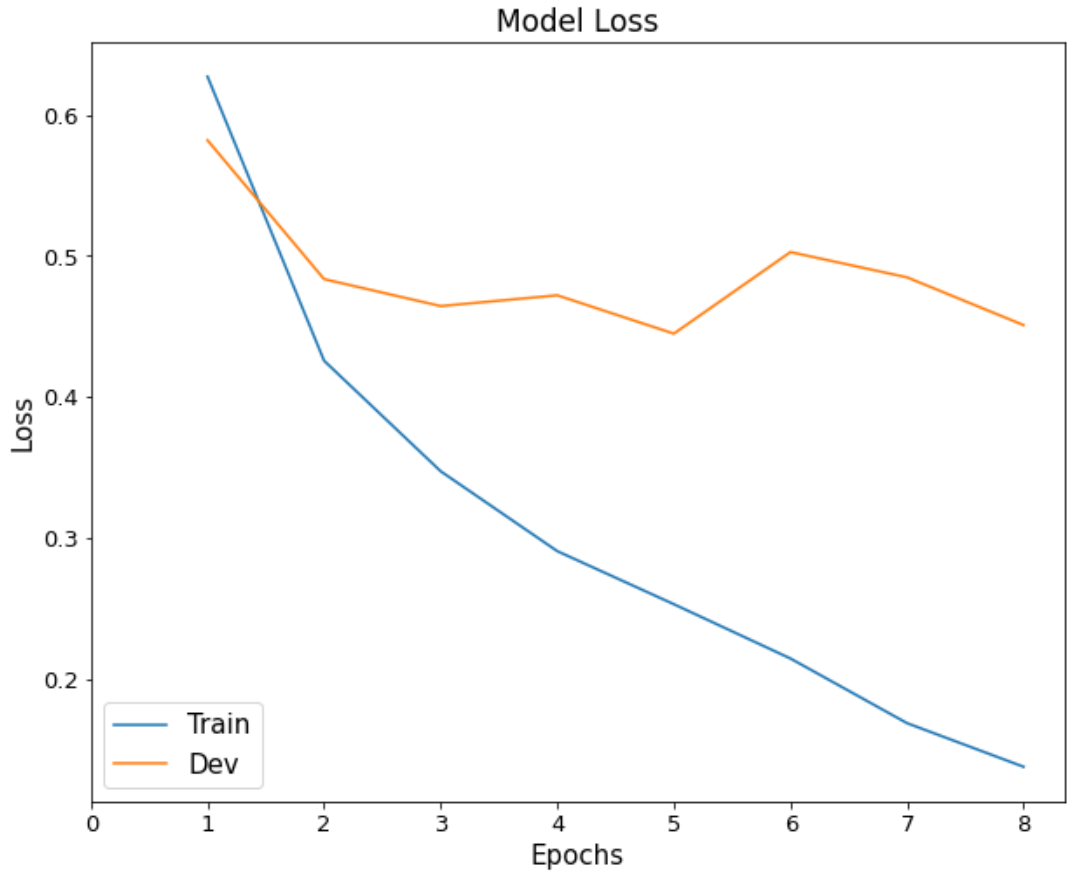
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 512)               51712

 dense_1 (Dense)             (None, 512)               262656

 dense_2 (Dense)             (None, 1)                 513

=================================================================
Total params: 314,881
Trainable params: 314,881
Non-trainable params: 0
```

Learning Curves



In the learning curves of the training and development data sets, the loss was depicted as a function of epochs. As expected, the learning curves of both training and dev set tend to descend. The dev set's curve though descends more slightly than the train set's curve, reaching its lowest value point (approximately 0.445) at the 5th epoch. From that point on, the validation loss increases for 3 consecutive epochs, thus EarlyStopping interferes and interrupts the training of the best MLP model.

Evaluation of the MLP model

The following tables show the precision, recall, F1 and precision-recall AUC scores, per class and classifier, separately for the training, development, and test sets along with the macro-averaged precision, recall, F1 and precision-recall AUC scores. Precision can be interpreted as the proportion of the data points our model says were relevant and actually were relevant. Recall can define the ability to find all relevant instances in a dataset. F1 score is the harmonic mean of precision and recall. Lastly, precision recall AUC scores are the area below the precision-recall curves, using various thresholds. For this exercise all these metrics were calculated for both the positive and the negative classes. It is worth noting that the higher these values are simultaneously, the better a model is.

Train data set

Est.	Precision		Recall		F1		PR-AUC		Macro Average			
	N	P	N	P	N	P	N	P	Prec	Recall	F1	PR-AUC
Dum	0.51	0.00	1.00	0.00	0.68	0.00	0.75	0.74	0.26	0.50	0.34	0.75
LR	0.85	0.86	0.86	0.84	0.86	0.85	0.93	0.93	0.85	0.85	0.85	0.93
MLP	0.90	0.95	0.96	0.89	0.93	0.92	0.98	0.98	0.93	0.93	0.93	0.98

Dev data set

Est.	Precision		Recall		F1		PR-AUC		Macro Average			
	N	P	N	P	N	P	N	P	Prec	Recall	F1	PR-AUC
Dum	0.49	0.00	1.00	0.00	0.66	0.00	0.75	0.75	0.25	0.50	0.33	0.75
LR	0.76	0.80	0.81	0.75	0.78	0.77	0.86	0.84	0.78	0.78	0.78	0.85
MLP	0.78	0.80	0.80	0.78	0.79	0.79	0.88	0.86	0.79	0.79	0.79	0.87

Test data set

Est.	Precision		Recall		F1		PR-AUC		Macro Average			
	N	P	N	P	N	P	N	P	Prec	Recall	F1	PR-AUC
Dum	0.47	0.00	1.00	0.00	0.64	0.00	0.73	0.76	0.23	0.50	0.32	0.75
LR	0.74	0.84	0.84	0.74	0.78	0.78	0.83	0.89	0.79	0.79	0.78	0.86
MLP	0.77	0.85	0.85	0.78	0.81	0.82	0.86	0.90	0.81	0.81	0.81	0.88

Considering the previous tables , it can be concluded that, the MLP classifier accomplishes slightly better results compared to the Logistic Regression and Dummy Classifier baselines. It is worth mentioning that the results of the MLP are significantly better in the training set (e.g. Macro F1->0.93) and not as high as in the development and test sets (Macro F1-> 0.79 and 0.81, respectively). However, EarlyStopping does not allow the MLP classifier to overfit during training. This notable difference in the results between sets though, could be attributed to the fact that our dataset consists only of very few examples (1386). As a consequence, small variations in the partitioning into train-dev-test of the initial dataset could easily affect the final results of the MLP metrics. The final results could also be affected, of course, to a great degree in the case of small value changes to the hyper-parameters.

Manner of working on the assignment:

The presented assignment was processed with the equal participation of all team members/contributors. The team worked together in group Discord calls, during which the understanding and structure of the assignment were discussed and cleared upon. Subsequently, more Discord calls took place with one member each time sharing the screen, during which the code was created with the simultaneous attention/participation of all members. The code was each day at night worked upon or modified by each member individually, to eventually reach its final form.

The report was written in a similar manner.