**Question 1**

1. For pre-processor, choose 3 tokenizer shown as below:

"self.tokenize = nltk.tokenize.word\_tokenize"

training accuracy: 0.7075333333333333

validation accuracy: 0.6943333333333334

testing accuracy: 0.70555

"self.tokenize = nltk.tokenize.WhitespaceTokenizer().tokenize "

training accuracy: 0.7212

validation accuracy: 0.7112666666666667

testing accuracy: 0.70925

"self.tokenize = nltk.tokenize.SpaceTokenizer().tokenize"

training accuracy: 0.7211333333333333

validation accuracy: 0.7112666666666667

testing accuracy: 0.7092

Find that nltk.tokenize.WhitespaceTokenizer() shows better.

1. For parameters in CountVectorizer

stop\_words='english',

training accuracy: 0.7369333333333333

validation accuracy: 0.7283333333333334

testing accuracy: 0.72645

max\_features=None

training accuracy: 0.9343333333333333

validation accuracy: 0.8674666666666667

testing accuracy: 0.8696

Find that change the stop words and max\_features improve the model a lot.

1. For Logistic Regression model, comparing with 5 solver shown as below:

(training set: 0.3, validation set: 0.3, testing set: 0.4)

model = LogisticRegression(solver=' lbfgs')

training accuracy: 0.723

validation accuracy: 0.7086

testing accuracy: 0.7102

(training set: 0.3, validation set: 0.3, testing set: 0.4)

model = LogisticRegression(solver='newton-cg')

training accuracy: 0.7212

validation accuracy: 0.7113

testing accuracy: 0.7093

(training set: 0.3, validation set: 0.3, testing set: 0.4)

model = LogisticRegression(solver='liblinear')

training accuracy: 0.7212

validation accuracy: 0.7112

testing accuracy: 0.70905

(training set: 0.3, validation set: 0.3, testing set: 0.4)

model = LogisticRegression(solver='sag')

training accuracy: 0.7212

validation accuracy: 0.7112

testing accuracy: 0.7093

(training set: 0.3, validation set: 0.3, testing set: 0.4)

model = LogisticRegression(solver='saga')

training accuracy: 0.7213

validation accuracy: 0.7107

testing accuracy: 0.709

Find that solver='newton-cg' shows better in Logistic Regression

As is shown in the results of accuracy, the max\_features and stop\_words parameters effect the accuracy most. The max\_features = None means I don’t care about the frequency of the words and keep all the words as features. stop\_words='english' means to exclude the stop words by English way, which also increases the accuracy.

Meanwhile, WhitespaceTokenizer() method can improve the accuracy because it tokenize the word in a better way.

Choosing solver='newton-cg' and solver='sag' means the model fits the big data solver in logistic regression, and they performed well in this model.

Combining them together, the validation accuracy can improve to 0.8675.

**Question 2**

Choose max aggregation method, and the accuracy shows below:

training accuracy: 0.8202292323112488

validation accuracy: 0.7882462739944458

testing accuracy: 0.7928499579429626

**Question 3**

1. The aggregation method used by FastText.

Max aggregation method

training accuracy: 0.8202292323112488

validation accuracy: 0.7882462739944458

testing accuracy: 0.7928499579429626

Sum aggregation method

training accuracy: 0.5046419501304626

validation accuracy: 0.4978678226470947

testing accuracy: 0.4981499910354614

Mean aggregation method

training accuracy: 0.7543532252311707

validation accuracy: 0.7628153562545776

testing accuracy: 0.7638499736785889

Find that the max aggregation can get a better result.

1. The dimension of the embedding vectors

Dim = 32

training accuracy: 0.8165867328643799

validation accuracy: 0.6893435120582581

testing accuracy: 0.6880999803543091

Dim = 64

training accuracy: 0.8437944650650024

validation accuracy: 0.7887126803398132

testing accuracy: 0.7898499965667725

Dim = 128

training accuracy: 0.8733342289924622

validation accuracy: 0.8234274983406067

testing accuracy: 0.8240000009536743

Dim = 256

training accuracy: 0.8868381977081299

validation accuracy: 0.8363761305809021

testing accuracy: 0.8384000062942505

Dim = 512

training accuracy: 0.9040733575820923

validation accuracy: 0.836465060710907

testing accuracy: 0.8384999632835388

Find that more dimension can get better result.

1. The optimizer, including learning rate, momentum or even algorithm

SGD lr=0.3

training accuracy: 0.8533226847648621

validation accuracy: 0.8041045069694519

testing accuracy: 0.8066999912261963

SGD lr=0.2

training accuracy: 0.8607854247093201

validation accuracy: 0.8085466027259827

testing accuracy: 0.8048999905586243

SGD lr=0.1

training accuracy: 0.8498356342315674

validation accuracy: 0.826537013053894

testing accuracy: 0.8240000009536743

find that when the learning rate is 0.1 the result shows better.

1. The number of epochs trained for

num\_epochs = 10 SGD lr=0.1

training accuracy: 0.9110029935836792

validation accuracy: 0.7701670527458191

testing accuracy: 0.7671999931335449

num\_epochs = 50 SGD lr=0.1

training accuracy: 0.7914450168609619

validation accuracy: 0.7868794202804565

testing accuracy: 0.7893869876861572

num\_epochs = 50 SGD lr=0.01

training accuracy: 0.850479781627655

validation accuracy: 0.7944207787513733

testing accuracy: 0.79339998960495

1. The batch size.

batch\_size = 32

training accuracy: 0.7914450168609619

validation accuracy: 0.7868794202804565

testing accuracy: 0.7893869876861572

batch\_size = 64

training accuracy: 0.8127880692481995

validation accuracy: 0.7822251915931702

testing accuracy: 0.7844448685646057

1. The preprocessor.

Without stop words

training accuracy: 0.8567209243774414

validation accuracy: 0.8009506464004517

testing accuracy: 0.8030999898910522

In conclusion, the max aggregation and mean aggregation shows better than sum aggregation thus, I choose max aggregation here.

More embedding dimensions can get more feathers for a word, which can increase the validation accuracy.

Smaller leaning rate can make gradient decent more specific in each step however the number of epochs should increase if the leaning rate is too small.

More epoch means more training iteration, if the learning rate small enough, more epoch can improve the accuracy of the model.

Exclude the stop words in tokens in preprocessor can increase the accuracy, which means that in this model, let words focus more on meaning rather than stop words can improve the accuracy.

Finally combining max aggregation method, Dim = 256, SGD lr=0.1, num\_epochs = 20, batch\_size = 32 and without stop words the accuracy becomes:

training accuracy: 0.9995335936546326

validation accuracy: 0.8469261527061462

testing accuracy: 0.8477999567985535

**Question 4**

For LogisticRegressor validation accuracy: 0.8674666666666667; For FastText model validation accuracy: 0.8469261527061462.

The accuracy of the two model performs really close. However, comparing the validation accuracy the LogisticRegressor only performed a little bit better. One reason about that may because the LogisticRegressor model contains softmax function which helps predicted data divide to two parts, which fits the classification problem better.

**Question 5**

The accuracy in train vector classifier

training accuracy: 0.1674668788909912

validation accuracy: 0.16674163937568665

testing accuracy: 0.16695594787597656

**Question 6**

Change num\_epochs, num\_hidden to the same as embedding

training accuracy: 0.1738324612379074

validation accuracy: 0.17324590682983398

testing accuracy: 0.17317675054073334

Original accuracy in train predicted classifier:

training accuracy: 0.5139703154563904

validation accuracy: 0.5653873682022095

testing accuracy: 0.5686500072479248

Change word2vec parameters:

1. num\_hidden = 256, epochs = 5, learning rate = 1.0

training accuracy: 0.5315609574317932

validation accuracy: 0.5826892256736755

testing accuracy: 0.5845000147819519

1. num\_hidden = 256, epochs = 10, learning rate = 1.0

training accuracy: 0.6211575865745544

validation accuracy: 0.502265453338623

testing accuracy: 0.5018999576568604

1. num\_hidden = 256, epochs = 5, learning rate = 2.0

training accuracy: 0.6126066446304321

validation accuracy: 0.5588353276252747

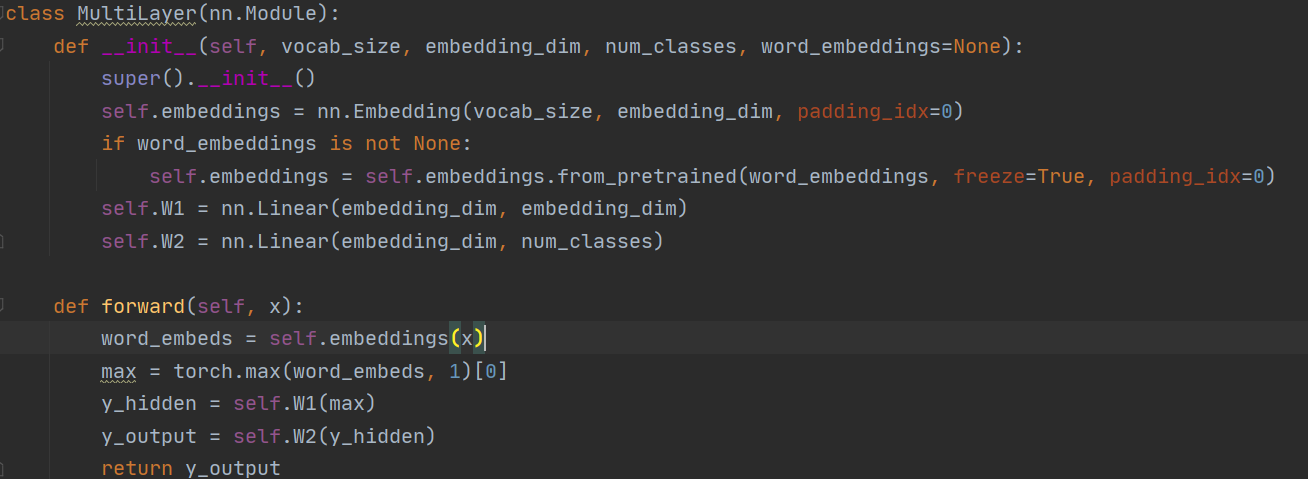
testing accuracy: 0.5561000108718872

find out that small leaning rate, more epochs and more embedding dimension can increase the accuracy.

According to the result, the FastText model performs better than the pretrained embeddings, because the pretrained model is trained by the movie reviews file, which is not such a big database. So the accuracy comparing to word embeddings in FastText is a little bit worth. So the accuracy performs worth in pretrained embeddings model.

**Question 7**

Change to multi-layer neural network:

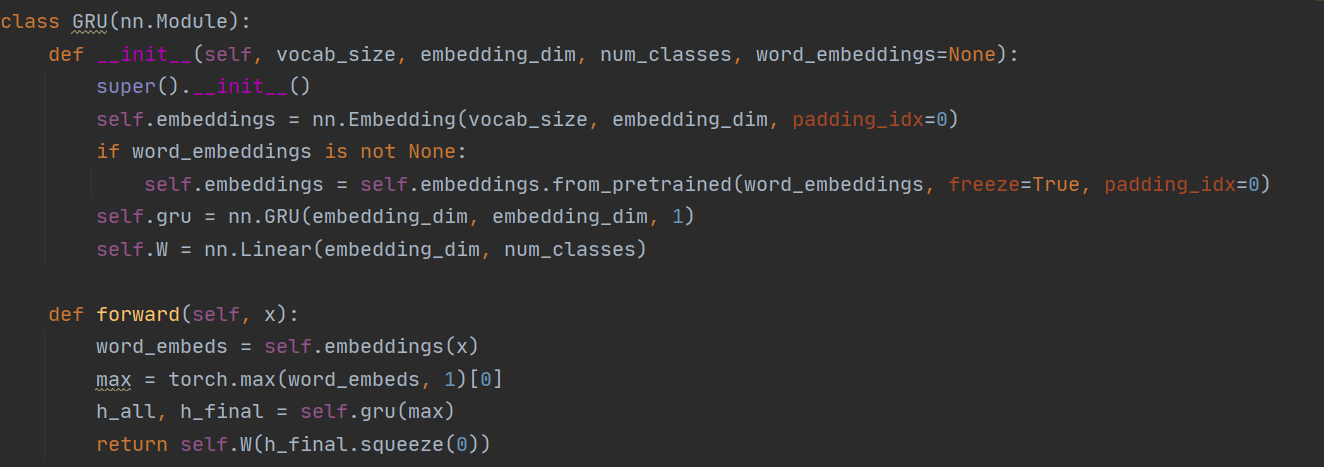


training accuracy: 0.806369960308075

validation accuracy: 0.7741871476173401

testing accuracy: 0.7726500034332275

Change to GRU neural network:



training accuracy: 0.8127880692481995

validation accuracy: 0.7822251915931702

testing accuracy: 0.7844448685646057

Change to LSTM neural network:

