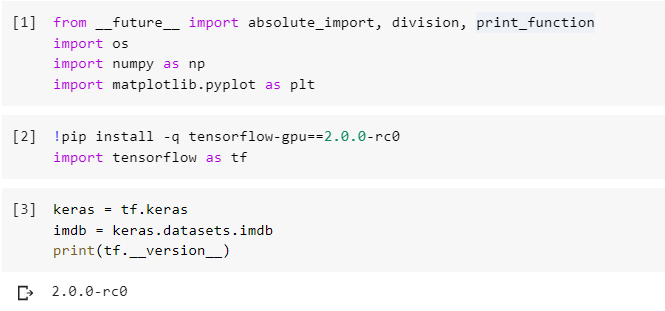
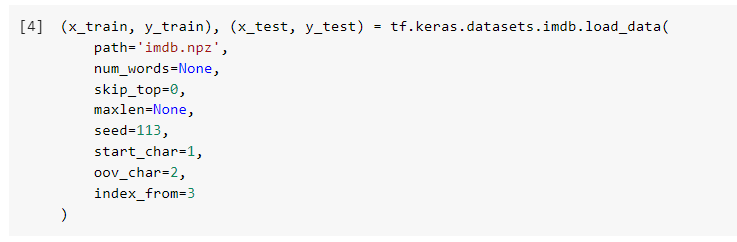
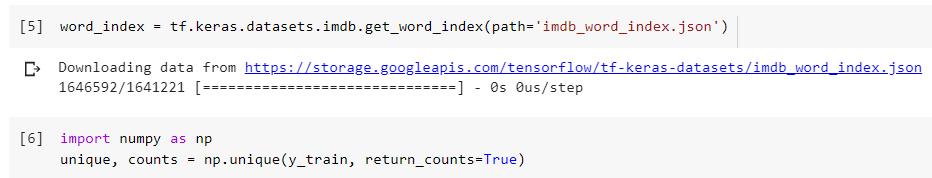
### Environment Setup



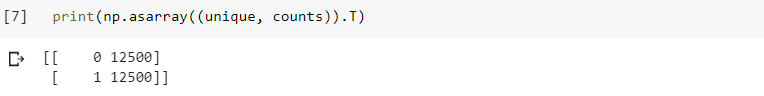
### Load IMDB movie review dataset



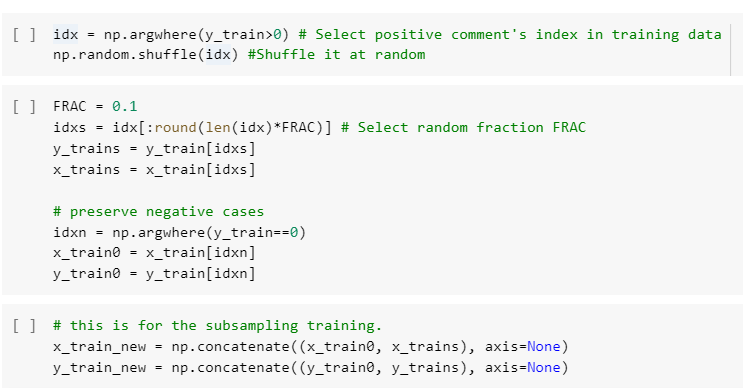
Load IMDB word index



### Check comment count

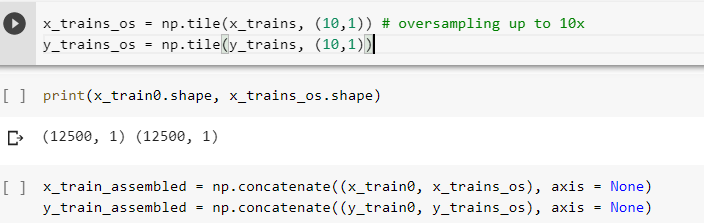


Lets randomly select a fraction of the positive reviews while keeping all negative reviews. We are going to use these subset of positive reviews as our positive base, and oversample these reviews in the base at random.

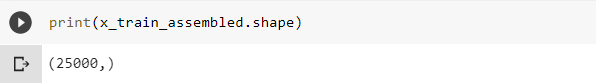


### Oversampling

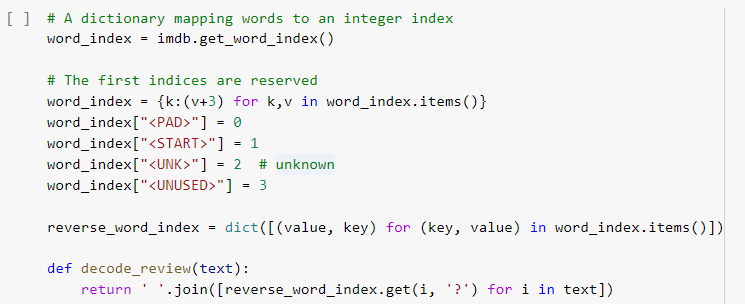
Lets oversample the positive comments up do ten times the original positive comment count. This ensures the counts of positive and negative reviews are same. This also constitutes our training dataset for this milestone.



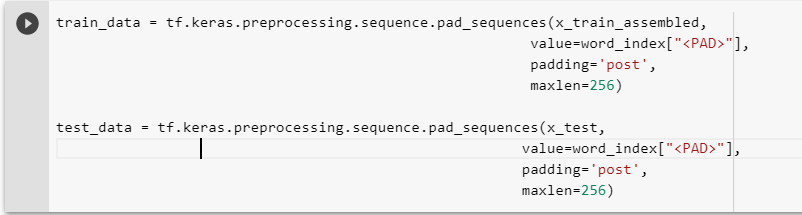
Now lets check tranining dataset size.



Lets also append IMDB word index with extra tokens for handling sentence length and demarcations.

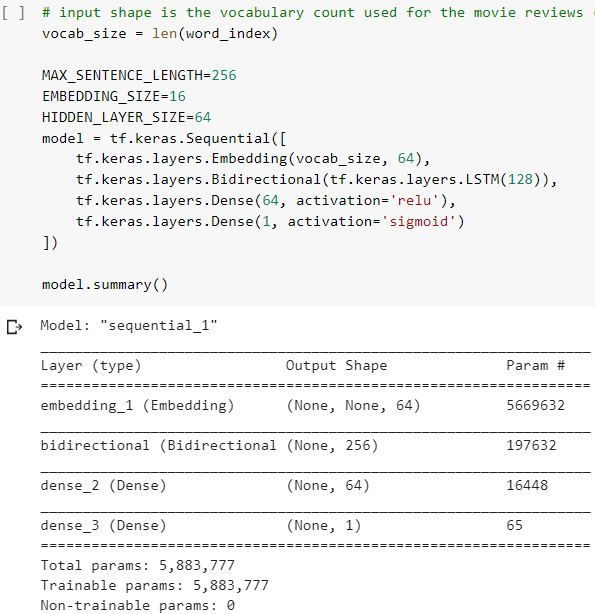


Lets pad each sentence to maximimum length of 256 words. We may take advantage of pad\_sequences function provided to speed simplify our task. We will pad sentences with <PAD> token up to 256 words.



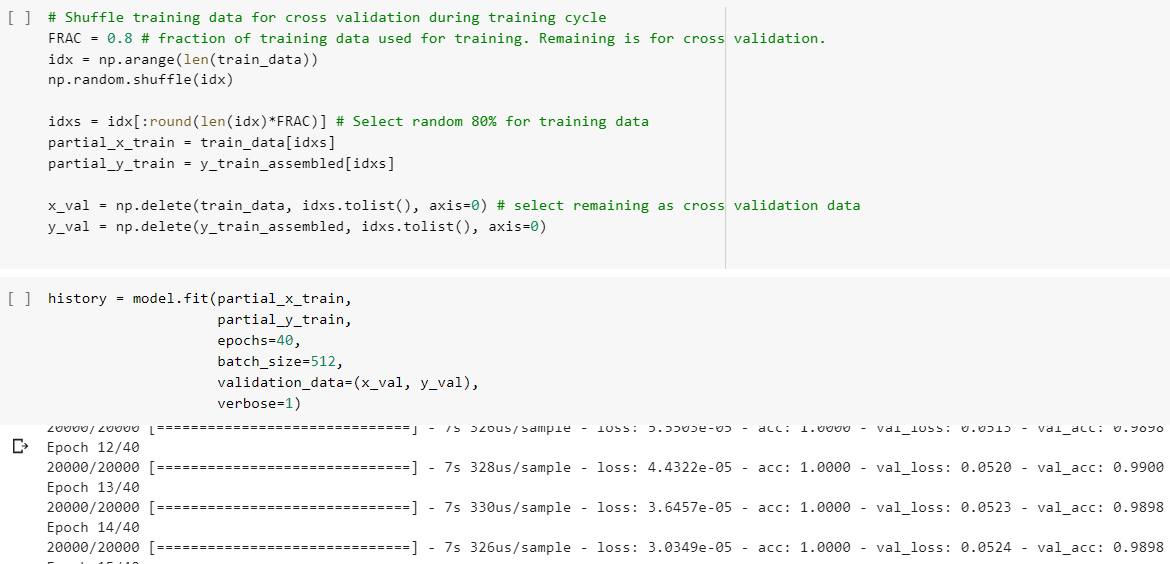
### Model Architecture

Lets build a simple text classification model. Start with embedding layer that convert a word into multi-dimensional vector representation. Then we feed that representation to a bidirectional Long-Short Terms Memory cell (LSTM) that uses 128 (a hyperparameter - arbitrarily chosen, feel free to experiment) dimensions to represent text sequence, follow by a dense layer to aggregate the LSTM output before making a classification.



### Cross Validation

### It is a good practice to set up a portion of training data for cross validation at the end of each training epoch. This helps us identify proper training epochs and prevent overfitting by memorizing training data.



### Scoring Test Data

### Lets take a look at training process to make sure the model is improving through training epochs.

### 

### 

### 

### 

### After we trained the model, now lets test it with holdout (test) data.

### 

### Prediction on test dataset

### Lets create a confusion matrix to see how the model perform with respect to each review type.

### 

### Lets take a closer look at model performance for each type of review.

### 

### The outcome of oversampling positive reviews is a trade-off and compromise of precision and recall between different types of reviews. The degradation in recall (performance with respect to false negative) for positive comment indicates that the model misclassifies 63% (1-0.37) positive comments as negative comments.