Q1 Calculate **precision/recall/f-score** for each label, compare the results from the two clustering models, and write your analysis in a pdf file.

```
cosine
label    Disaster and Accident  News and Economy  Travel & Transportation
cluster
0                          161                 1                      79
1                           12                94                      99
2                           37               111                       6
Cluster 0: Topic Disaster and Accident
Cluster 1: Topic Travel & Transportation
Cluster 2: Topic News and Economy
                         precision    recall  f1-score   support

  Disaster and Accident       0.67      0.77      0.71       210
       News and Economy       0.72      0.54      0.62       206
Travel & Transportation       0.48      0.54      0.51       184

               accuracy                           0.62       600
              macro avg       0.62      0.61      0.61       600
           weighted avg       0.63      0.62      0.62       600
```

```
L2
label    Disaster and Accident  News and Economy  Travel & Transportation
cluster
0                           67                 0                      99
1                          137                71                      81
2                            6               135                       4
Cluster 0: Topic Travel & Transportation
Cluster 1: Topic Disaster and Accident
Cluster 2: Topic News and Economy
                         precision    recall  f1-score   support

  Disaster and Accident       0.47      0.65      0.55       210
       News and Economy       0.93      0.66      0.77       206
Travel & Transportation       0.60      0.54      0.57       184

               accuracy                           0.62       600
              macro avg       0.67      0.62      0.63       600
           weighted avg       0.67      0.62      0.63       600
```

From the result shown above, different distant methods show different sensitive on different cluster. That is caused by the nature of the distance. For example, in the L2 distance, the cluster of 'News and Economy' has a very high precision compared to the cosine distance's. In this case, the L2 distance can cluster the 'News and Economy' better. Probably because a L2 distance is very suitable to differ this cluster. Also we see L2 distance did a bad job on the precision of cluster 'Disaster and Accident', this may be because these the magnitude of these vectors does not matter. So in this case, the cosine distance is better.

In general, cosine and Euclidean Distance are two different methods to deal with the distance between to vectors in vector space. Euclidean Distance is widely used in many machine learning algorithm.However, Cosine similarity is generally used as a metric for measuring distance when the magnitude of the vectors does not matter. This happens for example when working with text data represented by word counts. We want to apply cosine similarity for other cases where some properties of the instances make so that the weights might be larger without meaning anything different.

Q2.3. Provide a pdf document which contains:
- performance comparison between Q1 and Q2.1
- describe how you tune the model parameters, e.g. alpha, max_iter etc. in Q2.1.
- discuss how effective the method in Q2.2 is to find similar documents, compared with the tfidf weight cosine similarity we used before.

```
label    Disaster and Accident  News and Economy  Travel & Transportation
cluster
0                           30                18                     138
1                           12               182                       8
2                          168                 6                      38
Cluster 0: Topic Travel & Transportation
Cluster 1: Topic News and Economy
Cluster 2: Topic Disaster and Accident
                        precision    recall  f1-score   support

  Disaster and Accident       0.79      0.80      0.80       210
      News and Economy        0.90      0.88      0.89       206
Travel & Transportation       0.74      0.75      0.75       184

              accuracy                           0.81       600
             macro avg        0.81      0.81      0.81       600
          weighted avg        0.81      0.81      0.81       600
```

(1) It is obvious that the LDA model has a better performance on clustering the documents into three clusters. The precision, recall and F1-score are all much higher than the K-means method.

(2) We use the perplexity to evaluate the LDA model, the lower, the better.

```
lda = LatentDirichletAllocation(n_components=num_topics, \
                                max_iter=25,verbose=1,
                                evaluate_every=1, n_jobs=1,
                                random_state=0).fit(X_train)
```

```
iteration: 1 of max_iter: 25, perplexity: 4250.5376
iteration: 2 of max_iter: 25, perplexity: 3801.0843
iteration: 3 of max_iter: 25, perplexity: 3609.7047
iteration: 4 of max_iter: 25, perplexity: 3530.5465
iteration: 5 of max_iter: 25, perplexity: 3494.8408
iteration: 6 of max_iter: 25, perplexity: 3473.0364
iteration: 7 of max_iter: 25, perplexity: 3454.9284
iteration: 8 of max_iter: 25, perplexity: 3439.2839
iteration: 9 of max_iter: 25, perplexity: 3426.7504
iteration: 10 of max_iter: 25, perplexity: 3416.5917
iteration: 11 of max_iter: 25, perplexity: 3409.4712
iteration: 12 of max_iter: 25, perplexity: 3403.4632
iteration: 13 of max_iter: 25, perplexity: 3396.3725
iteration: 14 of max_iter: 25, perplexity: 3387.7223
iteration: 15 of max_iter: 25, perplexity: 3382.7160
iteration: 16 of max_iter: 25, perplexity: 3380.7555
iteration: 17 of max_iter: 25, perplexity: 3379.6781
iteration: 18 of max_iter: 25, perplexity: 3378.8929
iteration: 19 of max_iter: 25, perplexity: 3378.2673
iteration: 20 of max_iter: 25, perplexity: 3377.7126
iteration: 21 of max_iter: 25, perplexity: 3377.2467
iteration: 22 of max_iter: 25, perplexity: 3376.8807
iteration: 23 of max_iter: 25, perplexity: 3376.5611
iteration: 24 of max_iter: 25, perplexity: 3376.2621
iteration: 25 of max_iter: 25, perplexity: 3375.9923
```

The result above shows that as the iteration going up, the perplexity is going down. So in this case, we know that the max_iter with 25 is better than 20. However, we can see the marginal decrease of the perplexity as the iteration going up. So this means that we can't increase the max_iter as much as we want.

Besides, I also use the Gridsearch method from the sklearn package:

In [22]:
```
from sklearn.model_selection import GridSearchCV
# Define Search Param
search_params = {'n_components': [2,3,4,5], 'learning_decay': [.5, .7, .9]}

# Init the Model
lda = LatentDirichletAllocation()

# Init Grid Search Class
model = GridSearchCV(lda, param_grid=search_params)

# Do the Grid Search
model.fit(X_train)
```

```
/Users/lilinsen/anaconda3/lib/python3.7/site-packages/sklearn/model_selection/_split.py:197
8: FutureWarning: The default value of cv will change from 3 to 5 in version 0.22. Specify
it explicitly to silence this warning.
  warnings.warn(CV_WARNING, FutureWarning)
```

Out[22]:
```
GridSearchCV(cv='warn', error_score='raise-deprecating',
             estimator=LatentDirichletAllocation(batch_size=128,
                                                 doc_topic_prior=None,
                                                 evaluate_every=-1,
                                                 learning_decay=0.7,
                                                 learning_method='batch',
                                                 learning_offset=10.0,
                                                 max_doc_update_iter=100,
                                                 max_iter=10,
                                                 mean_change_tol=0.001,
                                                 n_components=10, n_jobs=None,
                                                 perp_tol=0.1,
                                                 random_state=None,
                                                 topic_word_prior=None,
                                                 total_samples=1000000.0,
                                                 verbose=0),
             iid='warn', n_jobs=None,
             param_grid={'learning_decay': [0.5, 0.7, 0.9],
                         'n_components': [2, 3, 4, 5]},
             pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
             scoring=None, verbose=0)
```

]:
```
# Best Model
best_lda_model = model.best_estimator_

# Model Parameters
print("Best Model's Params: ", model.best_params_)

# Log Likelihood Score
print("Best Log Likelihood Score: ", model.best_score_)

# Perplexity
print("Model Perplexity: ", best_lda_model.perplexity(X_train))
```

```
Best Model's Params:  {'learning_decay': 0.7, 'n_components': 4}
Best Log Likelihood Score:  -2709709.830274194
Model Perplexity:  3326.668877618776
```

The result shows that in my search domain, the beat parameter is learning_decay with 0.7 and n_component with 4. When I use these parameters, the perplexity is lower than before. Actually I have tried a n_component with 25 and this have a perplexity which is 2000. This perplexity is much more lower than before. But when I compute the perplexity of the test set, the perplexity is over 10000. This means this may be an over-fitting situation.

(3) We print out the target document and the similar document:

```
Target document:
popular destinations from ny s jfk at a glance delays are expected at john f kennedy intern
ational when the airport s main runway is shutdown for four months of repairs starting marc
h here s a breakdown of the most popular destinations from jfk most traveled domestic desti
nations from jfk by number of passengers los angeles san francisco orlando las vegas san ju
an puerto rico most popular international destinations from jfk london paris frankfurt sant
o domingo dominican republic rome source port authority of new york and new jersey


Top 3 documents:
Doc_id = 337
web specials from alaska airlines alaska airlines is back with another round of web special
s travel rules and restrictions vary by city but most are good for travel from april throug
h may with a day advance purchase fares include anchorage to juneau round trip denver to ke
tchikan round trip fresno to portland round trip oakland to ketchikan round trip chicago to
portland round trip seattle to houston round trip sacramento to spokane round trip

Doc_id = 38
american eagle starts birmingham miami service fort worth texas ap american airlines region
al affiliate american eagle on tuesday launched nonstop service between birmingham ala and
miami the service between birmingham shuttlesworth international airport and miami internat
ional airport will be operated with seat embraer erj jets american and american eagle are u
nits of amr corp based in fort worth texas

Doc_id = 222
new airline flights to alaska may lower fares travel industry officials say more airline fl
ights to alaska this summer may boost competition and result in lower fares to some lower c
ities the anchorage daily news says continental united and us airways all plan to add daily
nonstop service between anchorage and portland chicago san francisco and philadelphia from
fairbanks delta plans a new flight to salt lake city and frontier adds a flight to denver t
rip to denver and salt lake city no flights are being added however to seattle a major tran
sfer hub for alaskans industry officials say the added competition should drop air fares as
the new flights are added in may and june
```

The target document talks about the popular of JFK airport, while the other 3 are not talking about JFK airport. However, they are all about the topics of airline or flight. This is different with the tfidf weight cosine similarity we used before.

Because LDA is a topic model. Each document is a distribution of topics. In this case, we divide the document into 3 topics and get the topic proportion array, which is a 600 * 3 matrix. So each row represent a document in the test set. We see each document as a vector. So the distance between different vectors is the different in topics. A close distance means that these two vector have similar distribution in topics. For example, these four document may all have a high percentage in airline topic so we can find them are similar.

For the tfidf weight cosine similarity, however, the tfidf actually describe the frequency of the terms appear. So the vector is different from the topic proportion array. The similar vectors in this case means their terms frequency may be similar. So in this case, we can find two documents are very similar, not just about topic but all is similar.