Requirements met :

- ✓ Classification of dataset into 5 classes
- ✓ Computation of recall and precision - find in the same folder
- ✓ Confusion matrix - find in same folder
- ✓ Analysis of result - Next section
- ✓ Libraries used - requirements.txt
- ✓ Input read from file
- ✓ KFold cross validation to avoid overfitting
- ✓ Output is presented in two ways - screenshot and spyder variable explorer output
- ✓ Python is used for programming
- ✓ Method - Supervised learning with Random Forest

Analysis :

I have used Random Forest, an ensemble supervised learning algorithm. The choice of this algorithm is based on several reasons -

- ✓ Its an ensemble learning - Its a combination of 500 estimators.
- ✓ When results are compared against KMeans algorithm, supervised learning out performs KMeans. (KMeans can be found under Unsupervised.py).
- ✓ I would argue that frequency counting was not a feasible approach since dataset is really small and it would be nearly difficult to cover all the tags for a cluster.

To counter overfitting, I used following methods before finalising one -

- ✓ Test train split - Dataset is small. Therefore, it still overfits.
- ✓ KFold cross validation - I zeroed in on this because it arbitrarily changes test and train sets over many iterations.

Evaluation:

Confusion matrix (see in results folder) gives an idea of how well our prediction was. We had around 85% accuracy. However, accuracy is not the best measure at all times.

Precision projects false positives while recall projects false negatives. In our case, both had an impact. If suggestion is showing tickets from other clusters / classes (false positive), User might lost interest in the suggestions eventually because of non-relevancy of suggestions. On the other hand, if user does not see relevant tickets (false negative), he might miss out on an important ticket which would have been closely associated with ticket under consideration.

Class 0 had the least precision of 81% . This is essentially due to large portion of dataset corresponds to Class 0 and rest of the classes had very less entries compared with Class 0.

Recall is the probability that a relevant document is retrieved in a search (wikipedia). Class 4 and Class 5 had least recall of 66% and 33%. Since data corresponding to these classes is very small. Therefore probability that relevant ticket is retrieved is also small.

F1 score combines precision and recall which is generally a harmonic mean of the two.

Alternatives:

1. Many of the NLP problems used Naive Bayes algorithm. I did try with Naive bayes classifier as well. However, results were more or less the same as above.
2. KMeans clustering - Sicne it is an unsupervised learning, results did not meet expectations. Also, since KMeans generates random cluster labels every time, either I would need to label original clusters manually or by approximation (Assigning most frequent cluster label from prediction). It wasn’t nice either way!