**Idea descriptions**

**Multinomial Naïve Bayes Classifier**

The idea for the Multinomial Naïve Bayes Classifier was used for the topic modelling question because the number of topics we had for this project were definite. Rather than generate topics based on the keywords from the inputs and what the program learns from the corpus, we designed our topic modeler to designate an input under one of the many topics it has been provided in the corpus during training.

Multinomial Naïve Bayes classifier worked because this classifier assigns inputs to classes based on the probabilities generated for the inputs using bag of words as features. Also, in multinomial Naïve Bayes classifiers, the classes are more than 2 as opposed to the Bernoulli Naïve Bayes classifier that does binary classification.

To use the concept of Multinomial Naive Bayes classifier in our work, we considered the different topics provided in the training corpus as the classes. Because the number was definite, and each topic appeared more than 5 times in the corpus, it was a good application.

Although we implemented a lot of normalization and tokenization in preprocessing both the training and the test data, we still used the usual bag of words feature to build this classifier. Calculation for the probabilities and classes were still done using the prior probability and likelihood probability values as is typical of any Naïve Bayes classifier, only this time, the probabilities were returned for a wide range of topics, each representing a class.

**Logistic Regression Classifier**

The logistic regression classifier was used for another implementation of the topic modeling problem. The Multinomial implementation of the logistic regression classifier was used. Again, each of the topics in the corpus was assigned as a class.

Naturally, a logistic regression classifier takes a set of weights and a set of inputs and then calculates an output for each one of them, then moves the set of weights and the bias to the opposite direction of the gradient (in an attempt to find the minimum). This is done during training and after the optimal weights for maximizing the observational likelihood and devset testing accuracy without overfitting are found, those weights are used in the formula to calculate the probabilities for an input. Each of these weights represent a feature of the corpus. We used the sklearn implementation of the logistic regression classifier that learnt the features and found the optimal set of weights for us.

After these have been derived from training, we then work our way through every input, calculating the likelihood of the input with relation to each of the classes present. In a sentiment classifier based off of logistic regression, the values generated will be the likelihood of each input in relation to its positivity or negativity. In this case, the values are matched against a decision variable that determine whether or not a value makes its input positive or negative.

However, in multinomial logistic regression classifiers, as was implemented for this problem, the classes are more than 2 (and in this case, a little over 130). In this case a single decision variable that matches against in a threshold form would not work. That is why, instead, for multinomial logistic regression classifiers, a maximum probability concept and a SoftMax function is used. Instead of calculating the probability and matching it against a decision variable, we calculate the probability of the input belonging to each of the classes identified, return the maximum of the probabilities and then return the class of that maximum probability as the most likely class or topic for the input or question. The presence of the SoftMax function keeps the values probabilistic (so that they are always only ever between 0 and 1, and the sum of the all the probabilities is 1).

**Jaccard Similarity**

The Jaccard similarity model is a model that is used to compute the similarity between 2 or more inputs or sets. It calculates using the numerical values of the items in the sets and calculates the similarities of the 2 values by dividing the number of intersections of the two sets by the number of unions of the 2 sets. This calculation was used for the question and answers modelling problem. To have our inputs, which were the questions, in a numerical form that the Jaccard function could work with, we used embeddings that converted the words in the questions to numbers so that each input became a vector or a set of values.

When a user passes in a question, the program for the Question and answering model first converts it into a vector of numerical values, compares it to all the vectorized questions in the corpus, all the while appending the similarity values into a list with corresponding indices, and then returns the maximum of all the similarities as the right answer. After attaining the similarity value, the program finds the index of the value and returns the answer at that same index to the user as the answer to their question. Basically, it finds the most similar question in the corpus to what the user inputted, then returns the answer of that question to the user.

Although similar to the cosine similarity method, this uses a set approach, using their intersections and unions (general overlaps) to calculate the similarity.

We tried using probabilistic classifiers for the question and answering modelling problem, but it was not an accurate algorithm to apply to the problem. This is because, almost all the answers – which would have been the classes in this problem – are distinct (very few of them appear more than once). Because the classifiers described above worked on probability values, most of these ‘classes’ would have the same probability. Also, if there was a common answer like ‘yes’, that appeared a lot in the corpus, then the prior probability will be the highest for that class and the calculations would always be biased towards that class. Hence, the most returned answer of any input will be ‘yes’. We realized that probabilistic classifiers are not effective for question and answering systems.