Introduction:

When cooking, it is advisable to know the time commitments. Therefore this project can help people determine a relatively time commitment based on the ingredients and steps of a recipe. The initial dataset is taken from the internet. Features of the dataset include the name of the recipe, the number of steps involved, the number of ingredients, a list of ingredients, and a list of steps in the recipe.

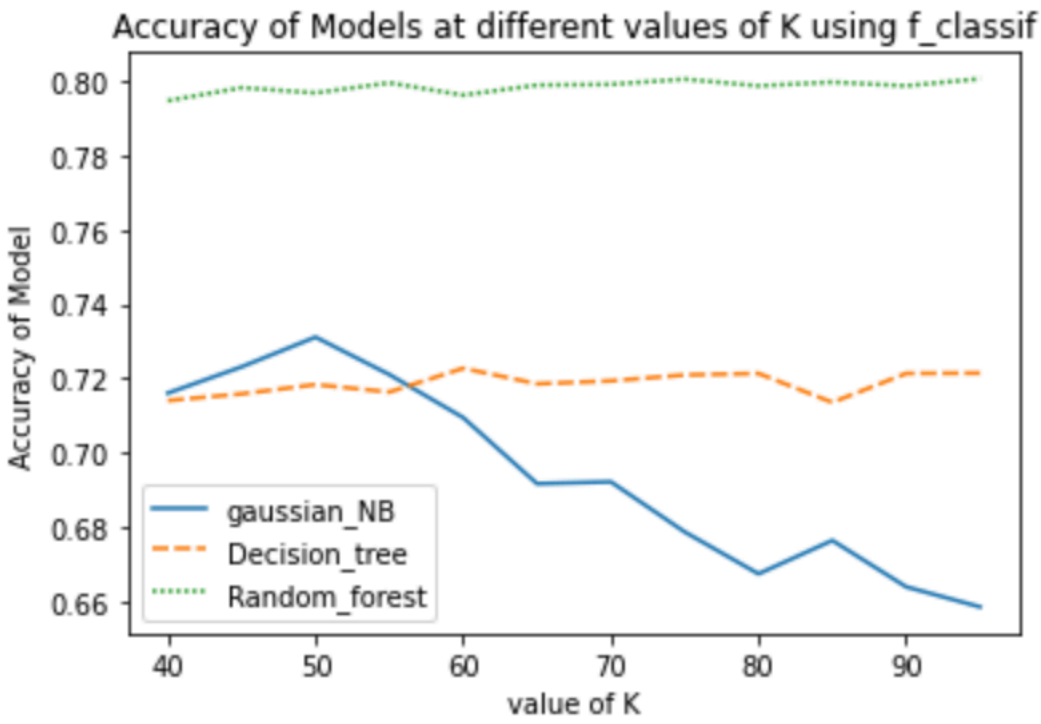
Program Outline:

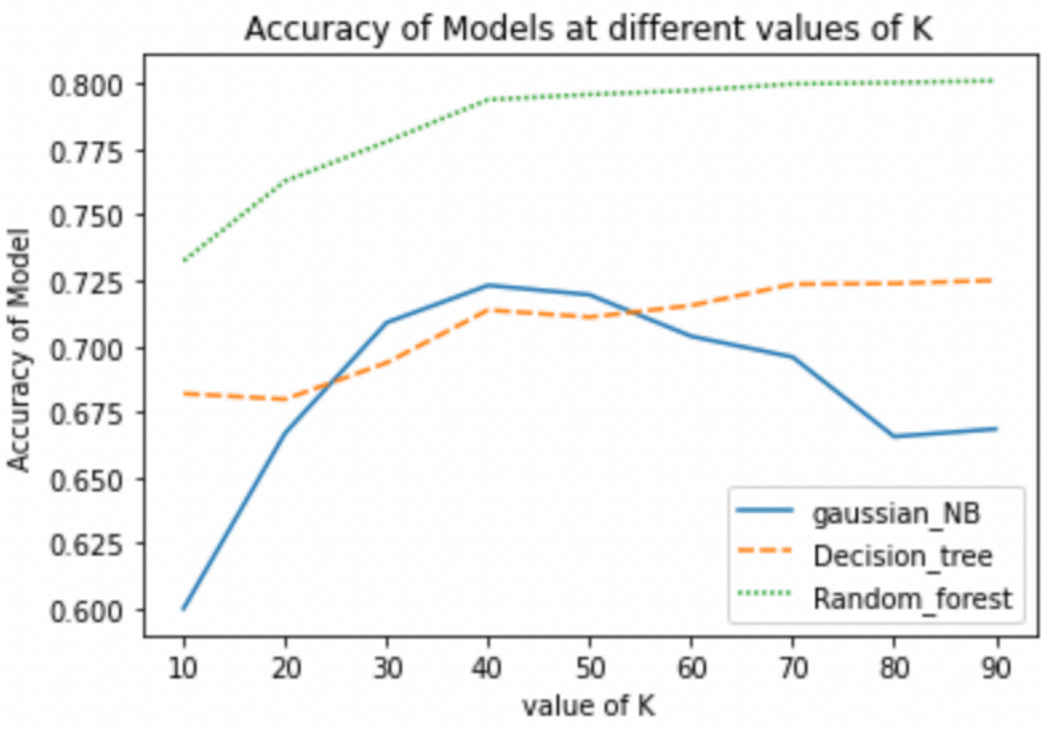
I created files for the count vectorisers and Doc2Vec features. Count vectorisers takes unique words in the document and gives a count of those word in each row which corresponds to a recipe. Doc2vectors take words in the document and represents them in the form of vectors. I chose to use Doc2Vec100 as it captures more relationships than Doc2Vec50.

After this comes processing the data as there is a lot of features which can make the machine learning models slow. I decided to keep the number of steps and ingredients as features because intuitively more ingredients or steps would likely mean a longer duration. For count vectorisers, I chose to use Select KBest from the sklearn library to select features. The reason for this was the thought that each individual word in the name, ingredients or steps would be relevant and related to the duration label. Therefore I decided it was better than PCA.

```
total variance for ingredients captured by 100 pca features:  [ 8.3 12.5 15.8 18.6 21.  23.1 25.1 26.9 28.6 30.2 31
.7 33.1 34.4 35.6
 36.7 37.8 38.8 39.8 40.8 41.7 42.6 43.5 44.3 45.1 45.9 46.6 47.3 47.9
 48.5 49.1 49.7 50.3 50.8 51.3 51.8 52.3 52.8 53.3 53.8 54.3 54.7 55.1
 55.5 55.9 56.3 56.7 57.1 57.5 57.9 58.3 58.7 59.1 59.5 59.9 60.2 60.5
 60.8 61.1 61.4 61.7 62.  62.3 62.6 62.9 63.2 63.5 63.8 64.1 64.4 64.7
 65.  65.3 65.6 65.9 66.1 66.3 66.5 66.7 66.9 67.1 67.3 67.5 67.7 67.9
 68.1 68.3 68.5 68.7 68.9 69.1 69.3 69.5 69.7 69.9 70.1 70.3 70.5 70.7
 70.9 71.1]
total variance for name captured by 100 pca features:  [ 2.5  4.   5.3  6.5  7.6  8.6  9.5 10.3 11.1 11.9 12.6 13.3
14.  14.7
 15.4 16.1 16.7 17.3 17.9 18.5 19.1 19.6 20.1 20.6 21.1 21.6 22.1 22.6
 23.1 23.6 24.  24.4 24.8 25.2 25.6 26.  26.4 26.8 27.2 27.6 28.  28.4
 28.8 29.1 29.4 29.7 30.  30.3 30.6 30.9 31.2 31.5 31.8 32.1 32.4 32.7
 33.  33.3 33.6 33.9 34.2 34.5 34.8 35.1 35.3 35.5 35.7 35.9 36.1 36.3
 36.5 36.7 36.9 37.1 37.3 37.5 37.7 37.9 38.1 38.3 38.5 38.7 38.9 39.1
 39.3 39.5 39.7 39.9 40.1 40.3 40.5 40.7 40.9 41.1 41.3 41.5 41.7 41.9
 42.1 42.3]
total variance for steps captured by 100 pca features:  [ 7.  11.5 13.7 15.5 17.2 18.8 20.2 21.5 22.8 23.9 25.  26.
26.9 27.8
 28.6 29.4 30.2 30.9 31.6 32.3 32.9 33.5 34.1 34.7 35.2 35.7 36.2 36.7
 37.2 37.7 38.2 38.7 39.1 39.5 39.9 40.3 40.7 41.1 41.5 41.9 42.3 42.7
 43.1 43.5 43.8 44.1 44.4 44.7 45.  45.3 45.6 45.9 46.2 46.5 46.8 47.1
 47.4 47.7 48.  48.3 48.6 48.9 49.2 49.5 49.8 50.1 50.4 50.7 51.  51.2
 51.4 51.6 51.8 52.  52.2 52.4 52.6 52.8 53.  53.2 53.4 53.6 53.8 54.
 54.2 54.4 54.6 54.8 55.  55.2 55.4 55.6 55.8 56.  56.2 56.4 56.6 56.8
 57.  57.2]
```

The picture above shows the cumulative variation shown in PCA. The variance captured in 100 features in pca does not capture enough variance for the name and steps features. This makes sense as ingredients usually don't require relationships between the different ingredients but all the words in the name and steps are relevant in relation to each other. Therefore it seemed more reasonable to use Select KBest as it would produce less features.





To choose an evaluation function for the Select Kbest function from sklearn, I plotted the performance the functions and different K values against the model. The first graph shows the performance of models over different K values using the f_classif evaluation function. As you can see the performance of the model is relatively stable from 40 to 100 using the decision tree and random forests. For Gaussian Naive Bayes, there is a clear best value of K at 50. As for mutual information, in the second graph, it is similar for random forests and decision trees from K values of 40 to 10. The best value for gaussian Naive Bayes was 40. In the end I chose f_classif as there was not much difference but the accuracy was slightly higher for Random Forest and Decision Trees over the 40-100 K value interval.

After deciding on Select KBest for the sparse matrices, I looked to do feature selection on Doc2Vec features. As they were all vectors, I decided to evaluate PCA based on all of the vectors from the name, ingredients and steps. I decided to use Doc2Vec100 as it contains more information. To find the optimal number of features to use in the PCA, I decided to find the number of features until the variance stopped increasing. This came to around 175.

```
array([ 3.3,  5.9,  8.2, 10.4, 12.5, 14.5, 16.3, 18.1, 19.8, 21.4, 22.9,
       24.4, 25.9, 27.3, 28.6, 29.9, 31.2, 32.5, 33.7, 34.9, 36.1, 37.3,
       38.5, 39.6, 40.7, 41.8, 42.9, 44. , 45. , 46. , 47. , 48. , 49. ,
       50. , 51. , 51.9, 52.8, 53.7, 54.6, 55.5, 56.4, 57.3, 58.1, 58.9,
       59.7, 60.5, 61.3, 62.1, 62.9, 63.7, 64.4, 65.1, 65.8, 66.5, 67.2,
       67.9, 68.6, 69.3, 70. , 70.6, 71.2, 71.8, 72.4, 73. , 73.6, 74.2,
       74.7, 75.2, 75.7, 76.2, 76.7, 77.2, 77.7, 78.2, 78.7, 79.2, 79.6,
       80. , 80.4, 80.8, 81.2, 81.6, 82. , 82.4, 82.8, 83.2, 83.6, 84. ,
       84.4, 84.7, 85. , 85.3, 85.6, 85.9, 86.2, 86.5, 86.8, 87.1, 87.4,
       87.7, 88. , 88.3, 88.6, 88.8, 89. , 89.2, 89.4, 89.6, 89.8, 90. ,
       90.2, 90.4, 90.6, 90.8, 91. , 91.2, 91.4, 91.6, 91.8, 92. , 92.2,
       92.4, 92.6, 92.7, 92.8, 92.9, 93. , 93.1, 93.2, 93.3, 93.4, 93.5,
       93.6, 93.7, 93.8, 93.9, 94. , 94.1, 94.2, 94.3, 94.4, 94.5, 94.6,
       94.7, 94.8, 94.9, 95. , 95.1, 95.2, 95.3, 95.4, 95.5, 95.6, 95.7,
       95.8, 95.9, 96. , 96.1, 96.2, 96.3, 96.4, 96.5, 96.6, 96.7, 96.8,
       96.9, 97. , 97.1, 97.2, 97.3, 97.4, 97.5, 97.6, 97.7, 97.7])
```

Next I implemented the models and made predictions on the testing data. For this task, I have made three different models: Decision Tree Classifier, Gaussian Naive Bayes classifier, and Random Forest classifier.

Evaluation of Models:

Based on my results, the accuracy of my models on the test data are:

Decision Tree = 0.68666

Gaussian Naive = 0.73633

Random Forest = 0.78066

Behaviour of Models:

I did not choose other hyper parameters such as algorithm or weights as I did not think there was any tangible difference that would be optimal for our data.

There were no hyper parameters for the Gaussian Naive Bayes. Neither for Random Forests and Decision Trees other than setting a random state

Error Analysis:

```
Average confusion matrix for Random Forest classifier over 5 split_test_train
[[3230  748  109]
 [1175 4312  167]
 [   6    9  244]]

Accuracy: 0.78

Micro Precision: 0.78
Micro Recall: 0.78
Micro F1-score: 0.78

Macro Precision: 0.83
Macro Recall: 0.68
Macro F1-score: 0.73

Weighted Precision: 0.78
Weighted Recall: 0.78
Weighted F1-score: 0.78

Classification Report

                  precision    recall  f1-score   support

Duration Label 1       0.79      0.73      0.76      4411
Duration Label 2       0.76      0.85      0.80      5069
Duration label 3       0.94      0.47      0.63       520

        accuracy                           0.78     10000
       macro avg       0.83      0.68      0.73     10000
    weighted avg       0.78      0.78      0.78     10000
```

Above is the confusion matrix and results metrics for the Random Forest classifier. Predicted duration labels are the rows of the confusion matrix and the true labels are the columns of the matrix.

```
Average confusion matrix for Gaussian Naive Bayes classifier over 5 split_test_train
[[3805 1728  107]
 [ 460 3012   86]
 [ 146  329  327]]

Accuracy: 0.71

Micro Precision: 0.71
Micro Recall: 0.71
Micro F1-score: 0.71

Macro Precision: 0.64
Macro Recall: 0.70
Macro F1-score: 0.65

Weighted Precision: 0.75
Weighted Recall: 0.71
Weighted F1-score: 0.71

Classification Report

                  precision    recall  f1-score   support

Duration Label 1       0.67      0.86      0.76      4411
Duration Label 2       0.85      0.59      0.70      5069
Duration label 3       0.41      0.63      0.49       520

        accuracy                           0.71     10000
       macro avg       0.64      0.70      0.65     10000
    weighted avg       0.75      0.71      0.71     10000
```

Above is the confusion matrix and results metrics for the Gaussian Naive Bayes classifier.

```
Average confusion matrix for Decision Tree classifier over 5 split_test_train
[[3112 1300  109]
 [1185 3630  138]
 [ 114  139  273]]

Accuracy: 0.70

Micro Precision: 0.70
Micro Recall: 0.70
Micro F1-score: 0.70

Macro Precision: 0.65
Macro Recall: 0.65
Macro F1-score: 0.65

Weighted Precision: 0.70
Weighted Recall: 0.70
Weighted F1-score: 0.70

Classification Report

                  precision    recall  f1-score   support

Duration Label 1       0.69      0.71      0.70      4411
Duration Label 2       0.73      0.72      0.72      5069
Duration label 3       0.52      0.53      0.52       520

        accuracy                           0.70     10000
       macro avg       0.65      0.65      0.65     10000
    weighted avg       0.70      0.70      0.70     10000
```

Above is the confusion matrix and results metrics for the Decision Tree classifier

Other than the Random Forest classifier, the accuracy of other models were around 0.7. This could be due to the hyper parameters not being tuned enough or the features were not entirely suited for the model. In the case of the Decision Tree, it is more likely that hyper parameters could have been tuned more as Random Forests were able to achieve a much higher accuracy which means that the format of the data fed into the model was alright.

Potentially I could have looked into tuning hyper parameters such as the minimum samples in a leaf for the Decision Trees and Random Forests. However, over multiple train and validation iterations, the accuracy of the model as well as performance metrics were stable enough to think that the models were not overfit to the training data. The fact that the Decision Tree accuracy over 5 train and test splits was similar to the testing results suggests that the model was not overfit to the training data. The same goes for the Random Forest classifier.

Future Recommendations:

Extend this program so that new records can be entered and predicted. This means using Azure to link the machine learning to a website and make the model able to be deployed. Better fine-tuning to models and creating more useful features would allow for better performance. Finally, I could have used an ensemble method to create a better accuracy

Bibliography:

Generating Personalized Recipes from Historical User Preferences. Bodhisattwa Prasad Majumder, Shuyang Li, Jianmo Ni, Julian McAuley, in Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), 2019.