# Applied Machine Learning

## HW 1 - Q4

Irfan Ali - CS16BTECH11019

**Problem Statement** : A Kaggle Challenge – “What’s Cooking”

**Language Used :** Python

**Library Used :** sklearn

**Solution :** The dataset given was a json object containing the ingredients for different dishes of different cuisines. First, the data had to be converted to a 2D array so that it could be passed to the ML models from the sklearn library for training. To convert the dataset to required form, I have first extracted all the different ingredients used in all the dishes. Then I have selected Top K mostly used ingredients as attributes of the model to prevent overfitting . Using these attributes, different models were trained to get the best prediction

I have done the challenge using 3 different Machine Learning models.

1. Multinomial Naive Bayes :
2. Bernoulli Naive Bayes
3. Decision Tree

Multinomial Naive Bayes :

```
    from sklearn.naive_bayes import MultinomialNB
```

We create a MultinomialNB classifier and train it on the training dataset using the fit function. Then we use the classifier to predict
Accuracy achieved : *74.688 %*
This accuracy was achieved by using the top 3500 ingredients.

Bernoulli Naive Bayes :

```
from sklearn.naive_bayes import BernoulliNB
```

We create a BernoulliNB classifier and train it on the training dataset using the fit function. Then we use the classifier to predict
Accuracy achieved : *73.541 %*
This accuracy was achieved by using the top 3500 ingredients

Decision Tree:

```
from sklearn import tree
```

We create a Decision Tree Classifier using :

```
tree.DecisionTreeClassifier(min_impurity_split = 0.2)
```

The “min_impurity_split” parameter specifies the threshold of the entropy to split the dataset. This is used to prevent overfitting the data.
After training the classifier, it is used to predict.
Accuracy achieved : *63.334 %*
This accuracy was achieved by using the top 3500 ingredients and a threshold of 0.2 for entropy

**Analysis :** Decision Tree is performing better than kNN for this dataset because there is no proper distance metric which can be used here for kNN. Decision Tree will overfit the data if all the ingredients are used as attributes. So we check for top K most used ingredients and use them only as attributes. Also decision trees decide the priority of the attributes on their own.

Naive Bayes performs better than Decision Tree because it uses probabilities. It does not overfit as much as decision trees do. There is no need of pruning or such stuff. So even if we have less data for training, naive bayes tends to work better