# ML Assignment Report

## Akash Bankar (23CS60R65)

Google Drive Project Link

### 1. Preprocessing Details

1. Used NLTK tweet tokenizer to tokenise emoji and urls
2. Used Emoji module to demojize emojis. i.e. Emoji to text
3. Removed Punctuations
4. Removed Stopwords
5. Used Porter stemmer to stemming
6. Used Tfidfvectorizer

### 2. Hyperparameter Tuning

### Support Vector Machine (SVM)

- **Hyperparameters Tuned**: We tuned the following hyperparameters for SVM:
    - Kernel: ('linear', 'rbf', 'poly', 'sigmoid')
    - C: [1, 10, 100, 1000, 0.1]
    - Gamma: ('scale', 'auto')
- **Tuning Technique**: GridSearchCV was employed for hyperparameter tuning.
- **Best Attributes Obtained** - (C=10, kernel='sigmoid', gamma='auto')

### Logistic Regression

- **Hyperparameters Tuned**: We explored the effect of different hyperparameters for Logistic Regression,
- **Tuning Technique**: Brute Force for each solver, penalty, C

- **Best Attributes Obtained** - (solver='saga', penalty='none')

## KNN

- **Hyperparameters Tuned**: We tuned the following hyperparameters for SVM:

  ```
  {'n_neighbors': [2,3,4,5,10,15,20,25,30],
  'weights' : ('uniform', 'distance'),
  'algorithm' : ('auto', 'ball_tree', 'kd_tree', 'brute')}
  ```

- **Tuning Technique**: GridSearchCV was employed for hyperparameter tuning.
- **Best Attributes Obtained** - (n_neighbors=4, weights='distance')

## MLPClassifier (Neural Network)

- **Hyperparameters Tuned**: Tried to learn a lot of parameters as follows -

hidden_layer_sizes

activation*{'identity', 'logistic', 'tanh', 'relu'}*

solver*{'lbfgs', 'sgd', 'adam'}*

learning_rate*{'constant', 'invscaling', 'adaptive'}*

But each iteration was taking a lot of time to learn, so couldn't tune. Used defaults.

Code Example (whole ipynb attached) -

```
from sklearn.model_selection import GridSearchCV
parameters = {'kernel':('linear', 'rbf', 'poly', 'sigmoid'),
svc = SVC()
clf = GridSearchCV(svc, parameters)
clf.fit(X_train_text, y_train)
print(clf.cv_results_)
```

## 3. Results

### KNN

- **Best Score**:

precision    recall  f1-score   support

```
    fake       0.88      0.92      0.90       501
    real       0.93      0.88      0.90       559

accuracy                           0.90      1060
```

macro avg       0.90      0.90      0.90      1060
weighted avg       0.90      0.90      0.90      1060

confusion matrix :  [[462  39]
[ 66 493]]
accuracy score : 0.9009433962264151
recall_average :  0.8819320214669052
precision_average :  0.9266917293233082
f1_score :  0.9037580201649863

### Support Vector Machine (SVM)

- **Best Score**:

precision    recall  f1-score   support

```
    fake       0.92      0.95      0.93       501
    real       0.95      0.93      0.94       559

accuracy                           0.94      1060
```

macro avg       0.94      0.94      0.94      1060
weighted avg       0.94      0.94      0.94      1060

confusion matrix :  [[475  26]
[ 41 518]]
accuracy score : 0.9367924528301886
recall_average :  0.9266547406082289

precision_average :  0.9522058823529411
f1_score :  0.9392565729827741

## Logistic Regression

- **Best Score**:

precision    recall  f1-score   support

```
    fake       0.92      0.95      0.93       501
    real       0.96      0.92      0.94       559

accuracy                           0.94      1060
```

macro avg       0.94      0.94      0.94      1060
weighted avg       0.94      0.94      0.94      1060

confusion matrix :  [[477  24]
[ 44 515]]
accuracy score : 0.9358490566037736
recall_average :  0.9212880143112702
precision_average :  0.9554730983302412
f1_score :  0.9380692167577414

## MLPClassifier (Neural Network)

- **Best Score**:

precision    recall  f1-score   support

```
    fake       0.92      0.96      0.94       501
    real       0.96      0.92      0.94       559

accuracy                           0.94      1060
```

macro avg       0.94      0.94      0.94      1060
weighted avg       0.94      0.94      0.94      1060

confusion matrix :  [[480  21]
[ 44 515]]

accuracy score : 0.9386792452830188
recall_average :  0.9212880143112702
precision_average :  0.960820895522388
f1_score :  0.9406392694063926

## KMeans

- **Best Score**:

precision    recall  f1-score   support

```
     fake       0.04      0.01      0.02       501
     real       0.45      0.73      0.56       559

 accuracy                           0.39      1060
```

macro avg       0.24      0.37      0.29      1060
weighted avg       0.26      0.39      0.30      1060

confusion matrix :  [[  6 495]
[152 407]]
accuracy score : 0.389622641509434
recall_average :  0.7280858676207513
precision_average :  0.45121951219512196
f1_score :  0.5571526351813826

## FastText

- **Best Score**:

Precision - 0.9837264150943397

Recall - 0.9837264150943397