FINAL PROJECT

AUTOMATIC EMOTION RECOGNITION

SHRUTI KAR

GESU BAL

MOTIVATION BEHIND SELECTING THIS PROJECT:

Emotion expression is an essential part of human interaction. The same text can hold different meanings when expressed with different emotions. Thus, understanding the text alone is not enough for getting the meaning of an utterance. Embedding an effective emotion detection feature in speech recognition system seems a promising solution for decreasing the obstacles faced by the deaf when communicating with the outside world. There exist several applications that allow the deaf to make and receive phone calls normally, as the hearing-impaired individual can type a message and the person on the other side hears the words spoken, and as they speak, the words are received as text by the deaf individual. However, missing the emotion part still makes these systems not hundred percent reliable. Having an effective speech to text and text to speech system installed in their everyday life starting from a very young age will hopefully replace the human ear. Many speech databases for different languages including English, German, Chinese, Japanese, Russian, Italian, Swedish and Spanish exist for modeling emotion recognition. Since there is no reported reference of an available Arabic corpus, we decided to select this Arabic Natural Audio Dataset (ANAD) to recognize discrete emotions.

DATASET USED: ARABIC NATURAL AUDIO DATASET (FROM KAGGLE)

<u>Preparation of dataset:</u>

Eight videos of live calls between an anchor and a human outside the studio were downloaded from online Arabic talk shows. Each video was then divided into turns: callers and receivers. To label each video, 18 listeners were asked to listen to each video and select whether they perceive a happy, angry or surprised emotion. Silence, laughs and noisy chunks were removed. Every chunk was then automatically divided into 1 sec speech units forming our final corpus.

We have a total of **1520 instances,** which are to be classified into **three classes i.e. 'Surprised, Angry, Happy'**.

- **Total features: 950 Features**

25 ACOUSTIC FEATURES
• **Intensity** • **Zero Crossing Rates** • **MFCC 1-12 (Mel-frequency Cepstral Coefficients)** • **F0 (Fundamental Frequency) And F0 Envelope** • **Probability of Voicing** • **LSP Frequency 0-7.**

STATISTICAL FUNCTIONS
• **Maximum** • **Minimum** • **Range** • **Absolute Position of Maximum** • **Absolute Position of Minimum** • **Arithmetic of Mean** • **Linear Regression1, Linear Regression2, Linear RegressionA, Linear RegressionQ** • **Standard Deviation** • **Kurtosis** • **Skewness** • **Quartiles 1, 2, 3** • **Inter-quartile Ranges 1-2, 2-3, 1-3** • **delta coefficient for every LLD**

<u>Features explained:</u>

Intensity: Intensity (often called the acoustic intensity) also called as the loudness of the sound, i.e. the greater the intensity, the louder we perceive the sound to be or the lower the intensity, the quieter we perceive the sound to be.

Zero Crossing Rates: The zero crossing count is an indicator of the frequency at which the energy is concentrated in the signal spectrum.

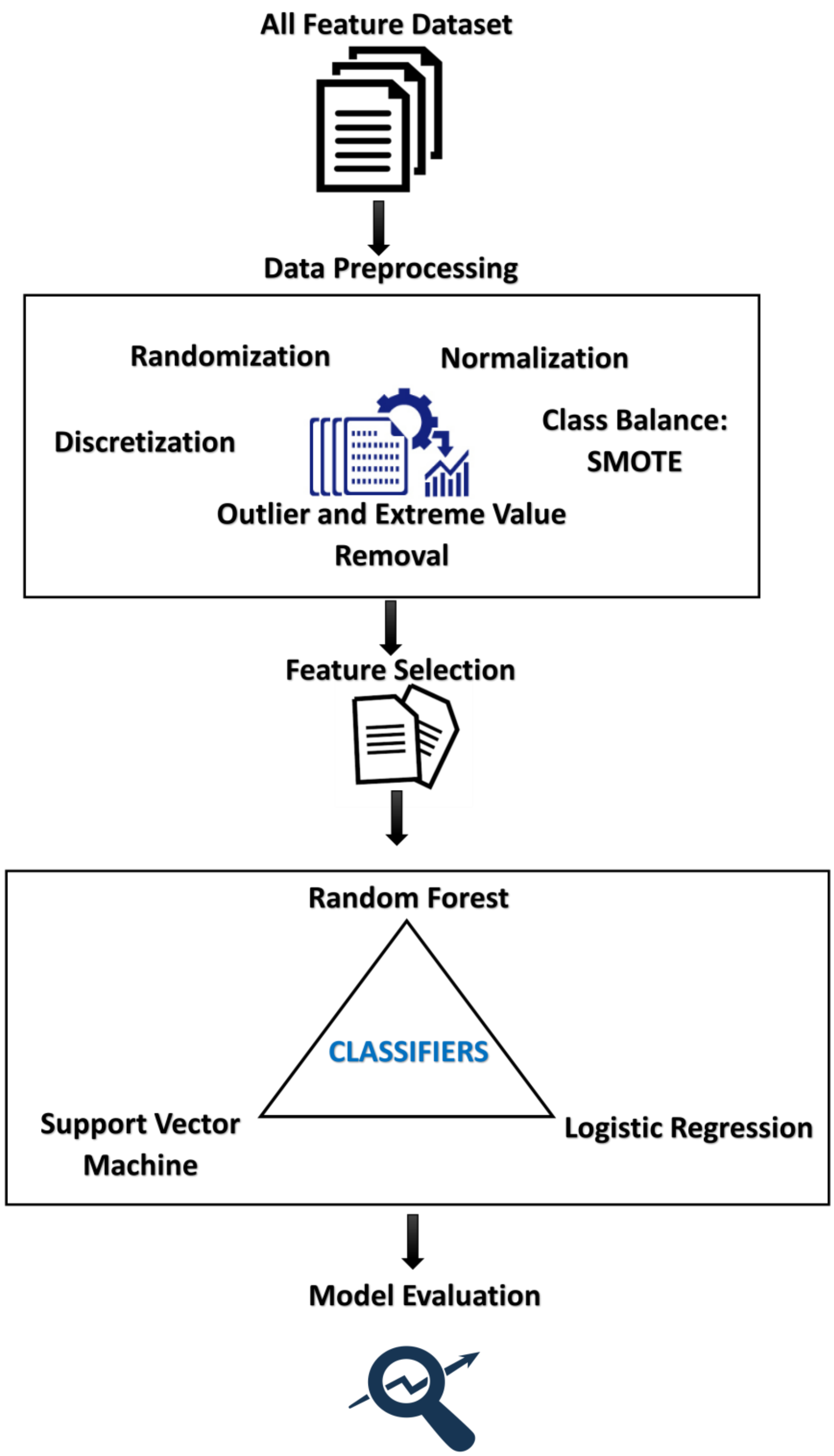
MFCC 1-12 (Mel Frequency Cepstral Coefficients): In audio, the mel-frequency cepstrum (MFC) is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency. Mel-frequency cepstral coefficients (MFCCs) are coefficients that collectively make up an MFC. It has 13 coefficients.

F0 (Fundamental Frequency): The fundamental frequency, often referred to simply as the fundamental, is defined as the lowest frequency of a periodic waveform.

Probability of voicing: A voicing probability determination method is provided for estimating a percentage of unvoiced and voiced energy

LSP Frequency: Line spectral pairs (LSP) or line spectral frequencies (LSF) are used to represent linear prediction coefficients (LPC) for transmission over a channel.

FRAMEWORK:


All Feature Dataset
Data Preprocessing
Randomization
Normalization
Discretization
Class Balance: SMOTE
Outlier and Extreme Value Removal
Feature Selection
Random Forest
CLASSIFIERS
Support Vector Machine
Logistic Regression
Model Evaluation

DATA PREPROCESSING:

- **Normalization:** We used min-max feature scaling.

Min max scaling: Replace feature x with (x - mean) / (max - min).

- **Class Balance:** Our dataset suffers from class imbalance problem which can be seen from figure below.

We observed very low F1-score for the class 'surprised' before balancing our dataset. This low F1-score was not even closely comparable with the other two classes' F1-score. So we used SMOTE i.e. Synthetic minority Oversampling technique to balance the minority class which is surprised in our case. SMOTE is an oversampling method. It works by creating synthetic samples from the minor class instead of creating copies. The algorithm selects two or more similar instances (using a distance measure) and perturbing an instance one attribute at a time by a random amount within the difference to the neighboring instances.

- **Outlier and Extreme Value removal:** Outliers are extreme values that fall a long way outside of the other observations. Our dataset had about 5% values which were outliers. We used a filter for detecting and removing outliers and extreme values based on interquartile ranges – **InterquartileRange.**
- **Discretization:** Discretization refers to the process of converting or partitioning continuous attributes, features or variables to discretized or nominal attributes/features. The filter we used for discretization – **Discretization.**

FEATURE SELECTION:

Initial number of total features: 950

Techniques we played with:

- **Wrapper methods** use a predictive model to score feature subsets. Each new subset is used to train a model, which is tested on a hold-out set. Counting the number of mistakes made on that hold-out set (the error rate of the model) gives the score for that subset. As wrapper methods train a new model for each subset, they are very computationally intensive, but usually provide the best performing feature set for that particular type of model.

- **Filter methods** use a proxy measure instead of the error rate to score a feature subset. This measure is chosen to be fast to compute, while still capturing the usefulness of the feature set.
- **Principal component analysis**: Principal component analysis is a quantitatively rigorous method for achieving this simplification. The method generates a new set of variables, called principal components. Each principal component is a linear combination of the original variables. All the principal components are orthogonal to each other, so there is no redundant information.

Although we played around with these techniques but we didn't get best results. The best results we got from

- **InfoGain Attribute Selection:** The InfoGain class is an implementation of a feature selection method by information gain. Information gain (InfoGain(t)) measures the number of bits of information obtained for prediction of a class (c) by knowing the presence or absence of a term (t) in a document. Concisely, the information gain is a measure of the reduction in entropy of the class variable after the value for the feature is observed.

So we ended up selecting 51 features from 950 features, which are 8 acoustic voice features.

'Emotion '	lspFreq_sma6_iqr13
pcm_fftMag_mfcc_sma1_iqr13	lspFreq_sma7_max
lspFreq_sma2_max	lspFreq_sma7_maxPos
lspFreq_sma2_minPos	lspFreq_sma7_minPos
lspFreq_sma2_linregc1	lspFreq_sma7_amean
lspFreq_sma2_kurtosis	lspFreq_sma7_linregc1
lspFreq_sma2_quartile1	lspFreq_sma7_linregc2
lspFreq_sma2_iqr13	lspFreq_sma7_linregerrA
lspFreq_sma3_max	lspFreq_sma7_linregerrQ
lspFreq_sma3_range	lspFreq_sma7_kurtosis
lspFreq_sma3_minPos	lspFreq_sma7_quartile1
lspFreq_sma3_amean	lspFreq_sma7_quartile2
lspFreq_sma3_linregc1	lspFreq_sma7_iqr23
lspFreq_sma3_kurtosis	lspFreq_sma_de2_iqr13
lspFreq_sma3_quartile1	lspFreq_sma_de3_minPos
lspFreq_sma3_quartile2	lspFreq_sma_de3_amean
lspFreq_sma4_max	lspFreq_sma_de3_linregc1
lspFreq_sma5_max	lspFreq_sma_de6_linregc1
lspFreq_sma5_kurtosis	lspFreq_sma_de6_linregc2
lspFreq_sma5_quartile1	lspFreq_sma_de6_linregerrA
lspFreq_sma5_quartile2	lspFreq_sma_de6_linregerrQ
lspFreq_sma5_iqr13	lspFreq_sma_de7_range
lspFreq_sma6_max	lspFreq_sma_de7_amean
lspFreq_sma6_min	lspFreq_sma_de7_linregc1
lspFreq_sma6_linregerrQ	lspFreq_sma6_kurtosis
lspFreq_sma6_stddev	lspFreq_sma6_quartile1

CLASSIFIERS:

- Random Forest
- Support Vector Machine
- Logistic Regression

Apart from the above three best performing classifiers on our dataset, we tried few other classifiers like Multilayer Perceptron, Naïve Bayes, Random tree, AdaBoost, Bagging and LogitBoost. But our dataset did not even give comparable results with these classifiers. Therefore, we compare the best three performing classifier's results in the following sections.

RANDOM FOREST:

Random forest is an ensemble technique which constructs multitude of decision trees and outputting the class that is the mode of the classes.

We played around with the following parameters:

- No. of folds in cross validation:

Number of folds	Root mean square error	F measure
2	0.2045	0.936
5	0.1855	0.949
10	0.1811	0.954
20	0.1811	0.952

So we chose number of folds=10 best for our model by looking at F measure and Root mean Square error.

- Bag Size percent:
  We tried various percentages of bag size but the model was best performing when the bag size is 100%
- Maximum depth of Tree:
  Increasing the depth of tree any more than 0 gave us poor results as limiting the tree size was not desirable in our case. Therefore we chose 0 as the maximum depth of tree which means that it is unlimited.
- Number of iterations:
  Increasing the number of iterations beyond 100 didn't make any significant improvement in our results, therefore we chose number of iterations as 100.
- Seed Value:
  We used random number seed value as 3.
- Number of Features:
  We take number of features as 0 which means it sets the number of features as int(log 2(#predictors)+1).

Results for Random Forest:

After Data Preprocessing step, we run our classifier on the dataset twice. Once before feature selection and again after feature selection. Therefore, we compare the results obtained from applying our classifier on all feature set with results on 51 feature set.

All Features					
PRECISION	RECALL	F-1 SCORE	ROC AREA	PRC AREA	
0.973	0.938	0.955	0.996	0.985	SURPRISED
0.965	0.993	0.979	0.997	0.998	ANGRY
0.982	0.958	0.970	0.998	0.996	HAPPY
0.972	0.972	0.972	0.997	0.995	WEIGHTED AVERAGE

```
Correctly Classified Instances        1477               97.1711 %
Incorrectly Classified Instances        43                2.8289 %
Kappa statistic                          0.9541
Mean absolute error                      0.139
Root mean squared error                  0.194
Relative absolute error                 33.6484 %
Root relative squared error             42.6894 %
Total Number of Instances             1520
```

Figure: Results of Random Forest on All Feature set.

```
=== Confusion Matrix ===

   a   b   c   <-- classified as
 257  12   5 |   a = surprised
   1 736   4 |   b = angry
   6  15 484 |   c = happy
```

Figure: Confusion Matrix

Our model predicts:

- **Correctly classified Instances:** 1477 that is 97% of total number of instances, which is a good number.
- **Precision 0.972:** Higher precision is due to the fact the model predicted less False positives for the three classes.
- **Kappa statistic 0.9541**: Model did a very good job predicting the classes as the kappa statistic is very high. More the Kappa statistic is towards 1, better is the performance of the model.
- **Root Mean Square Error 0.194:** More the Root Mean Squared Error is towards 0, better is the performance of the model.

- ➢ The result of ‘surprised’ class is lower than any other classes’ result as it is our minority class.

51 features					
PRECISION	RECALL	F-1 SCORE	ROC AREA	PRC AREA	
0.933	0.909	0.921	0.994	0.975	SURPRISED
0.952	0.987	0.969	0.995	0.995	ANGRY
0.977	0.939	0.958	0.996	0.993	HAPPY
0.957	0.957	0.956	0.995	0.991	WEIGHTED AVERAGE

```
Correctly Classified Instances        1454               95.6579 %
Incorrectly Classified Instances        66                4.3421 %
Kappa statistic                          0.9295
Mean absolute error                      0.101
Root mean squared error                  0.1789
Relative absolute error                 24.448  %
Root relative squared error             39.3801 %
Total Number of Instances             1520
```

Figure: Results of Random Forest on 51 Feature set.

```
=== Confusion Matrix ===

   a   b   c   <-- classified as
 249  19   6 |   a = surprised
   5 731   5 |   b = angry
  13  18 474 |   c = happy
```

Figure: Confusion Matrix

Our model predicts:

- ➢ **Correctly classified Instances:** 1454 that is 96% of total number of instances, which is a good number, but is less than what we saw in all feature set.
- ➢ **Precision 0.956:** Higher precision is due to the fact the model predicted less False positives for the three classes. Again we see that the F1-score is less than what we saw in all feature set.
- ➢ **Kappa statistic 0.9295**: Model did a very good job predicting the classes as the kappa statistic is very high. More the Kappa statistic is towards 1, better is the performance of

the model. Again we see that the Kappa statistic is less than what we saw in all feature set.

- **Root Mean Square Error 0.1789:** More the Root Mean Squared Error is towards 0, better is the performance of the model. Here the value is less than what we saw in all feature set which means that it has lesser error in the prediction and so it gives better prediction.

We notice that, although our Correctly Classified Instances, Precision, Kappa statistic are giving lesser value in the 51 features set but the root mean square error is lower and so is better in the case of 51 feature set. Apart from this, there is a higher chance for our model in the case of all feature set to be overfitting as we have 950 features and only about 1500 instances. For the same reason we choose 51 feature set as the basis for comparison of different classifiers in the Evaluation section.

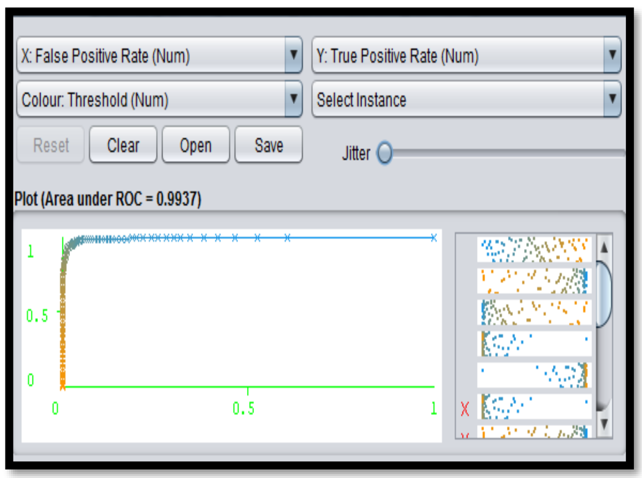


Figure: ROC of 'Surprised' class for Random Forest

The AUC value of 0.9937 for surprised class is a very good value. More the value of Area towards one, better the predictions.

SUPPORT VECTOR MACHINE:

Support Vector Machine is a supervised method which views each data point as a p dimensional vector and tries to classify it using a (p-1) dimensional hyperplane.

We played around with the following parameters:

- No. of folds in cross validation:

Number of folds	Root mean square error	F measure
2	0.3497	0.794
5	0.347	0.804

10	0.346	0.806
20	0.347	0.802

So we chose number of folds=10 best for our model by looking at F measure and Root mean Square error.

- Type of Kernel:
  We experimented with linear kernel and poly kernel and found that poly kernel performed much better with our dataset.
- Complexity Parameter:
  We kept the complexity parameter as 1. Changing it above that, it didn't make any significant change to the performance.
- Seed Value:
  We used random number seed value as 1.

Results for Support Vector Machine:

After Data Preprocessing step, we run our classifier on the dataset twice. Once before feature selection and again after feature selection. Therefore, we compare the results obtained from applying our classifier on all feature set with results on 51 feature set.

All Features					
PRECISION	RECALL	F-1 SCORE	ROC AREA	PRC AREA	
0.828	0.810	0.819	0.948	0.733	SURPRISED
0.972	0.973	0.972	0.973	0.962	ANGRY
0.959	0.962	0.960	0.975	0.941	HAPPY
0.953	0.953	0.953	0.972	0.932	WEIGHTED AVERAGE

```
Correctly Classified Instances        1476               97.1053 %
Incorrectly Classified Instances        44                2.8947 %
Kappa statistic                          0.9535
Mean absolute error                      0.2294
Root mean squared error                  0.2846
Relative absolute error                 55.5348 %
Root relative squared error             62.6291 %
Total Number of Instances             1520
```

Figure: Results of Support Vector Machine on All Feature set.

```
=== Confusion Matrix ===

   a   b   c   <-- classified as
 111  13  13 |   a = surprised
  12 721   8 |   b = angry
  11   8 486 |   c = happy
```

Figure: Confusion Matrix

Our model predicts:

- **Correctly classified Instances:** 1476 that is 97% of total number of instances, which is a good number.
- **Precision 0.953:** Higher precision is due to the fact the model predicted less false positives for the three classes.
- **Kappa statistic 0.9535**: Model did a very good job predicting the classes as the kappa statistic is very high. More the Kappa statistic is towards 1, better is the performance of the model.
- **Root Mean Square Error 0.2846:** More the Root Mean Squared Error is towards 0, better is the performance of the model.

51 features					
PRECISION	RECALL	F-1 SCORE	ROC AREA	PRC AREA	
0.742	0.766	0.754	0.894	0.656	SURPRISED
0.908	0.922	0.915	0.921	0.880	ANGRY
0.911	0.875	0.893	0.947	0.866	HAPPY
0.879	0.878	0.879	0.925	0.835	WEIGHTED AVERAGE

```
Correctly Classified Instances        1335               87.8289 %
Incorrectly Classified Instances       185               12.1711 %
Kappa statistic                          0.8035
Mean absolute error                      0.2544
Root mean squared error                  0.3239
Relative absolute error                 61.5873 %
Root relative squared error             71.2824 %
Total Number of Instances             1520
```

Figure: Results of Support Vector Machine on 51 Feature set.

```
=== Confusion Matrix ===

   a   b   c   <-- classified as
 210  39  25 |   a = surprised
  40 683  18 |   b = angry
  33  30 442 |   c = happy
```

Figure: Confusion Matrix

Our model predicts:

- **Correctly classified Instances:** 1335 that is 88% of total number of instances, which is less than what we saw in all feature set.
- **Precision 0.879:** Higher precision is due to the fact the model predicted less False positives for the three classes. Again we see that the F1-score is less than what we saw in all feature set.
- **Kappa statistic 0.8035**: Model did a very good job predicting the classes as the kappa statistic is very high. More the Kappa statistic is towards 1, better is the performance of the model. Again we see that the Kappa statistic is less than what we saw in all feature set.
- **Root Mean Square Error 0.3239:** More the Root Mean Squared Error is towards 0, better is the performance of the model. Here the value is higher than what we saw in all feature set which means that it has higher error in the prediction and so it does not give better prediction.

LOGISTIC REGRESSION:

Logistic Regression is a statistical method to classify the dataset.

We played around with the following parameters:

- No. of folds in cross validation:

Number of folds	Root mean square error	F measure
2	0.2691	0.870
5	0.2536	0.875
10	0.2386	0.885
15	0.242	0.883

So we chose number of folds=10 best for our model by looking at F measure and Root mean Square error.

- Maximum Number of Iterations:
  We chose maximum number of iterations to be a value until it converges. Therefore we chose maximum number of iterations as -1.
- Use Conjugate Gradient Descent:
  We didn't use conjugate Gradient Descent, instead used BFGS algorithm.

Results for Logistic Regression:

After Data Preprocessing step, we run our classifier on the dataset twice. Once before feature selection and again after feature selection. Therefore, we compare the results obtained from applying our classifier on all feature set with results on 51 feature set.

All Features					
PRECISION	RECALL	F-1 SCORE	ROC AREA	PRC AREA	
0.594	0.715	0.649	0.899	0.642	SURPRISED
0.888	0.856	0.871	0.928	0.922	ANGRY
0.870	0.820	0.844	0.936	0.906	HAPPY
0.829	0.818	0.822	0.926	0.866	WEIGHTED AVERAGE

```
Correctly Classified Instances        1244              81.8421 %
Incorrectly Classified Instances       276              18.1579 %
Kappa statistic                          0.7108
Mean absolute error                      0.1208
Root mean squared error                  0.3382
Relative absolute error                 29.2579 %
Root relative squared error             74.4326 %
Total Number of Instances             1520
```

Figure: Results of Logistic Regression on All Feature set.

```
=== Confusion Matrix ===

   a   b   c   <-- classified as
 196  45  33 |   a = surprised
  78 634  29 |   b = angry
  56  35 414 |   c = happy
```

Figure: Confusion Matrix

Our model predicts:

- **Correctly classified Instances:** 1244 that is 82% of total number of instances, which is a good number.
- **Precision 0.822:** Higher precision is due to the fact the model predicted less False positives for the three classes.
- **Kappa statistic 0.7108**: Model did an average job predicting the classes as the kappa statistic is average. More the Kappa statistic is towards 1, better is the performance of the model.
- **Root Mean Square Error 0.3382:** More the Root Mean Squared Error is towards 0, better is the performance of the model.

51 features					
PRECISION	RECALL	F-1 SCORE	ROC AREA	PRC AREA	
0.740	0.726	0.733	0.947	0.789	SURPRISED
0.911	0.916	0.914	0.970	0.960	ANGRY
0.925	0.927	0.926	0.980	0.973	HAPPY
0.885	0.886	0.885	0.969	0.933	WEIGHTED AVERAGE

```
Correctly Classified Instances        1346               88.5526 %
Incorrectly Classified Instances       174               11.4474 %
Kappa statistic                          0.8149
Mean absolute error                      0.0951
Root mean squared error                  0.2386
Relative absolute error                 23.0143 %
Root relative squared error             52.4977 %
Total Number of Instances             1520
```

Figure: Results of Logistic Regression on 51 Feature set.

```
=== Confusion Matrix ===

   a   b   c   <-- classified as
 199  49  26 |   a = surprised
  50 679  12 |   b = angry
  20  17 468 |   c = happy
```

Figure: Confusion Matrix

Our model predicts:

- **Correctly classified Instances:** 1346 that is 89% of total number of instances, which is more than what we saw in all feature set.
- **Precision 0.885:** Higher precision is due to the fact the model predicted less false positives for the three classes. Again we see that the F1-score is more than what we saw in all feature set.
- **Kappa statistic 0.8149**: Model did a very good job predicting the classes as the kappa statistic is very high. More the Kappa statistic is towards 1, better is the performance of the model. Again we see that the Kappa statistic is more than what we saw in all feature set.
- **Root Mean Square Error 0.2386:** More the Root Mean Squared Error is towards 0, better is the performance of the model. Here the value is lower than what we saw in all

features set which means that it has lower error in the prediction and so it gives better prediction.

We noticed one **Surprising result** in the case of logistic regression. In every other classifier, the performance was lower for the 51 feature set when compared to the results of all feature set. But, in logistic regression, we noticed it is reverse. The performance of the classifier on selected 51 feature is better than its performance on all feature set. This is because the classifier logistic regression gets affect by distracting features.

MODEL EVALUATION:

For model evaluation, we took the metrics Kappa Statistic, F-Measure and Root Mean Square Error.

- **Kappa Statistic**: The Kappa statistic (or value) is a metric that compares an Observed Accuracy with an Expected Accuracy (random chance).
- **F-Measure**: The F1 score (also F-score or F-measure) is a measure of a test's accuracy. It considers both the precision p and the recall r of the test to compute the score.
- **Root Mean Square Error**: Root-mean-square error (RMSE) is a measure which is difference between values predicted by a model or an estimator and the values observed.

51 FEATURES					
CLASSIFIER	F-SCORE	KAPPA STATISTIC	ROOT MEAN SQUARE ERROR	ROC	NUMBER OF CORRECTLY CLASSIFIED INSTANCES
RANDOM FOREST	0.956	0.9295	0.1789	0.995	1454
SVM	0.879	0.8035	0.3239	0.925	1335
LOGISTIC REGRESSION	0.885	0.814	0.2386	0.965	1346

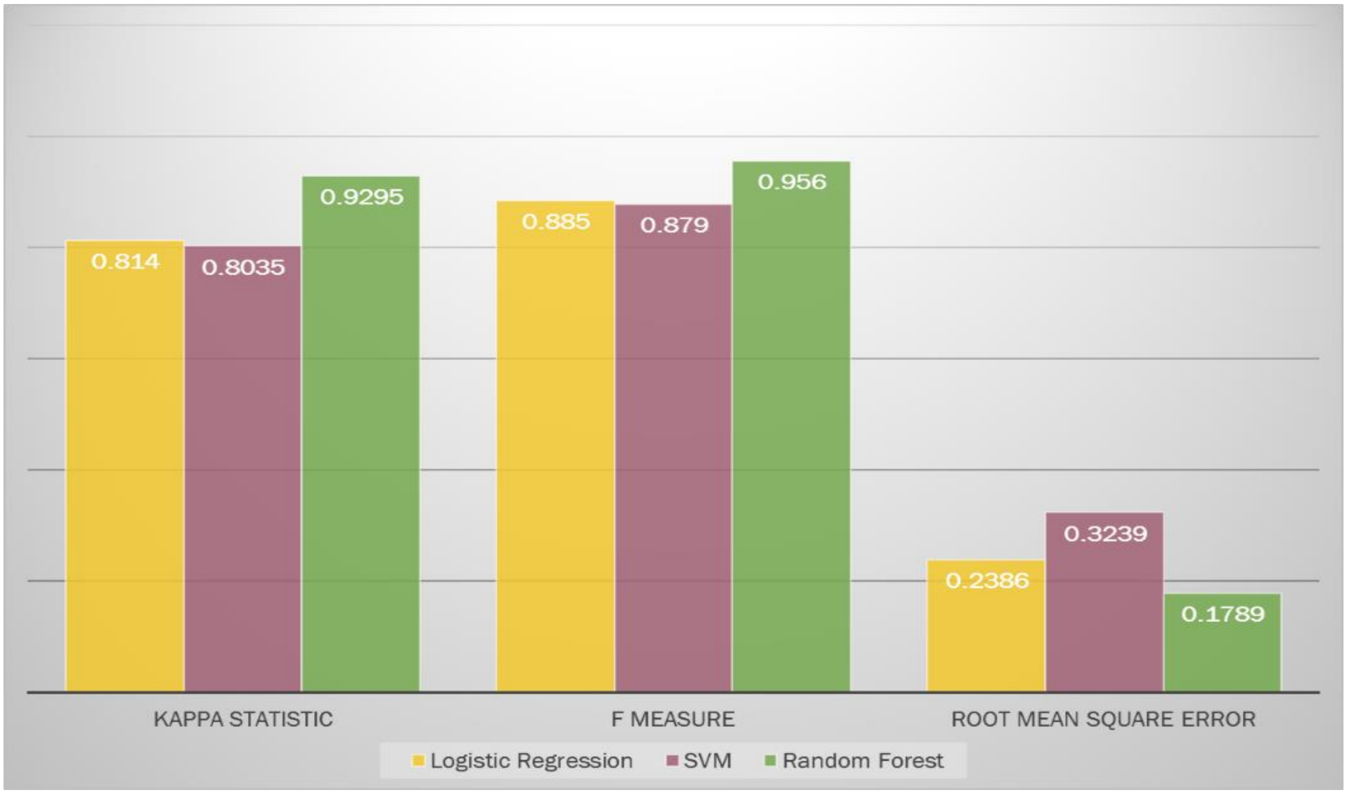
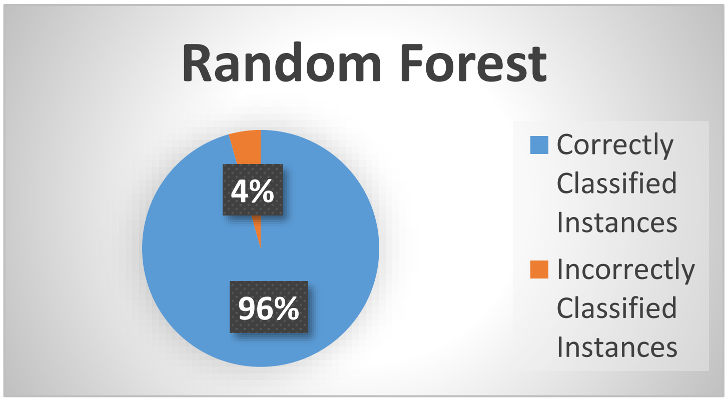


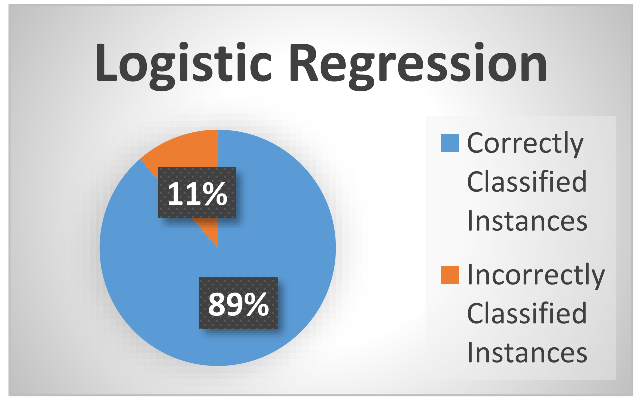
Figure: Comparison of classifiers on metrics.

We observed that:

- As Higher the Kappa Statistic, better is the performance of the classifier. We notice that the performance of Random Forest is the best whereas the performance of SVM is the worst with this metric.
- As Higher the F-measure, better is the performance of the classifier. We notice again that the performance of Random Forest is the best whereas the performance of SVM is the worst with this metric.
- As Lower the Root Mean Square Error, better is the performance of the classifier. We notice that the performance of Random Forest is again the best whereas the performance of SVM is the worst with this metric.
- Although there is not much of a difference in the values between SVM and Logistic Regression, but in comparison logistic did better than SVM.
- Random Forest performed the best as it implicitly performs feature selection. The number of features we selected, 51, is still a higher number and there is a probable chance for the model to overfit. Random Forest eliminates this problem by elimination overfitting problem.

- SMOTE in the class balancing process did reduce the class imbalance effect but did not eliminate it completely. This is the reason SVM did not perform well because SVM's prediction is biased towards the majority class.





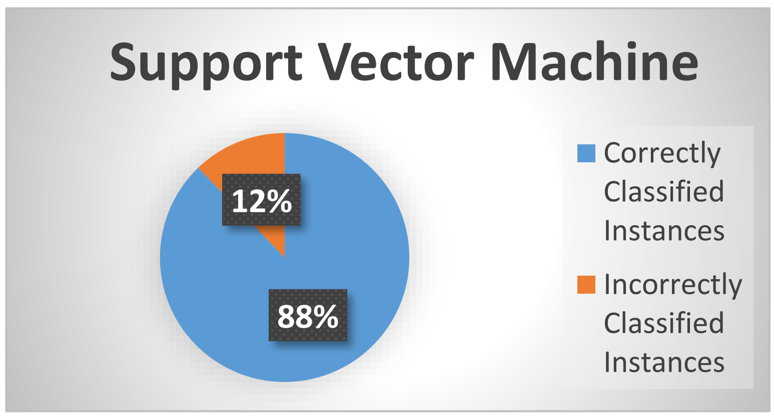


Figure: Comparison between classifiers based on correctly classified instances

We observed that:

- We can clearly observe that Random Forest is performing the best by classifying 96% of instances correctly.
- Among SVM and Logistic Regression, Logistic Regression did a better job by classifying 89% of instances correctly where as compared to 89% of instances in SVM.

LIMITATIONS:

- Lack of thorough knowledge about feature engineering. Better feature selection could have been performed.
- Lack of understanding of nature of features: Lack of understanding about if the features are independent or dependable.
- Lack of application of other algorithms to boost the number of instances. More number of instances could have let us to apply more complex classifiers for better results.

CONCLUSION AND FUTURE WORK:

Though the results of classifiers on all features are slightly better, we consider the results from selected features to be the optimal one as the model over-fits with all features. Random Forest outperforms every other classifier based on the fact that it performs implicit feature selection by itself.

SVM in our case performed the least because of class imbalance problem. There are algorithms to boost the performance of an algorithm which can be tested for SVM. Also, twinSVM could also be tried which solves the problem faced by SVM in case of class imbalance problem.

APPENDIX:

WORK CONTRIBUTION

WORK	CONTRIBUTOR
Dataset selection	Shruti (50%), Gesu (50%)
Data preprocessing	Shruti (100%)
Application of Classifiers	Gesu (100%)
Model evaluation	Shruti (50%), Gesu (50%)
Presentation	Shruti (40%), Gesu (60%)
Report	Shruti (60%), Gesu (40%)