PROJECT REPORT

TV News Channel Commercial Detection

GROUP – 4

SUMANTH SARA

AKHILA REDDY

SAI SIDDHARTHA

CHARAN DEVAPATLA

Introduction:

The project is about classifying a news channel video clip as commercial or non-commercial. We have taken the data from the UCI machine learning repository[1]. The data consists of 150 hours of video recordings. These recordings are not directly given, rather they have been processed and a specific set of features have been extracted. Some of the features include Motion distribution, Frame difference, Edge change ratio and Fundamental frequency. Basically, these features are extracted from the dataset using complicated algorithms. Thus, the final dataset shared in the website is just 61MB. All the 150 hours of data has been compressed into such a small file size by using complicated algorithms to extract only required meta-data from the videos.

This data is used to classify if a specific video clip contains a commercial or not. In this project, we go through the data, clean it and do some preprocessing steps to make it suitable for machine learning tasks.

1. Pre-processing of the data:

The data provided by the UCI machine learning repository is in compressed dense format. To understand the data, we need to extract it and expand the dense matrix. For this purpose, we are using a python library called libsvm2csv[2]. This library expands the dense matrix which is in libsvm[3] format into a sparse matrix into csv style[4] by filling zeros wherever needed. Thus, we get a proper 2d matrix saved into a csv file which can be used for machine learning tasks.

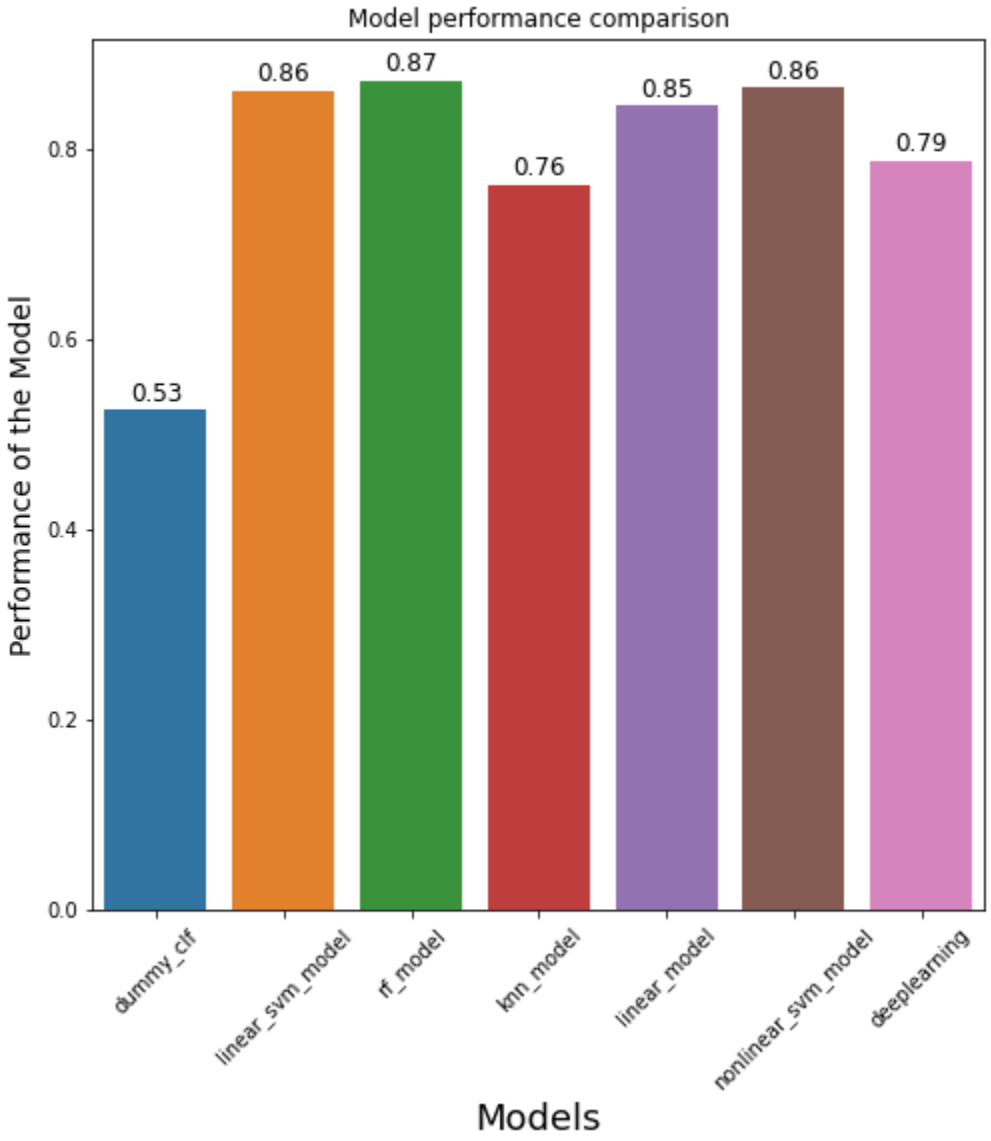
2. Results for all the single models

We have used a list of models for this binary classification task. The following are the models used

- Dummy Classifier[5]
- Linear model (Ridge classifier)[6]
- Linear SVM[7]
- Random Forest[8]
- KNN[9]
- Non-linear SVM[10]
- Deep Learning[11]

The following is the accuracy of each model plotted in a barplot. We have used the best parameters possible for each model by trial-and-error method. Initially default parameters were used and later Grid Search[12] technique for tuning the best parameters.

From the following figure [Figure 1], we can clearly see the performance of SVM is the highest compared to other models. Random forest is an ensemble[13] of multiple decision trees, so it's ambiguous if we can call it a single model or an ensemble model. So we choose SVM as our best single model.



[Figure 1]

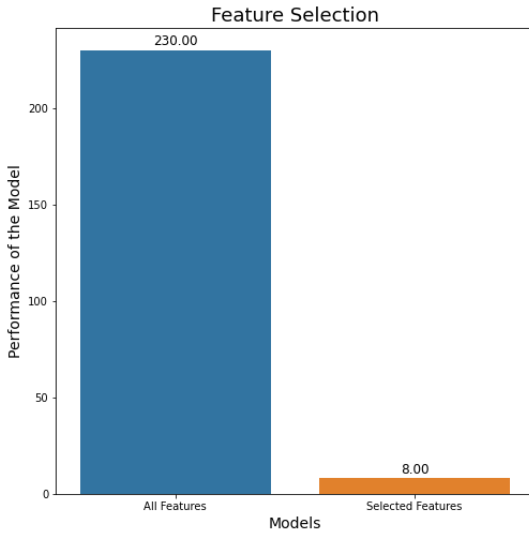
3. First Variable Selection:

We have a total of 230 features in our dataset. Using all the features is not required for classifying if a record is a commercial or non-commercial. So we can use some variable selection methods like KNN or Lasso[14] to select important features from the dataset.

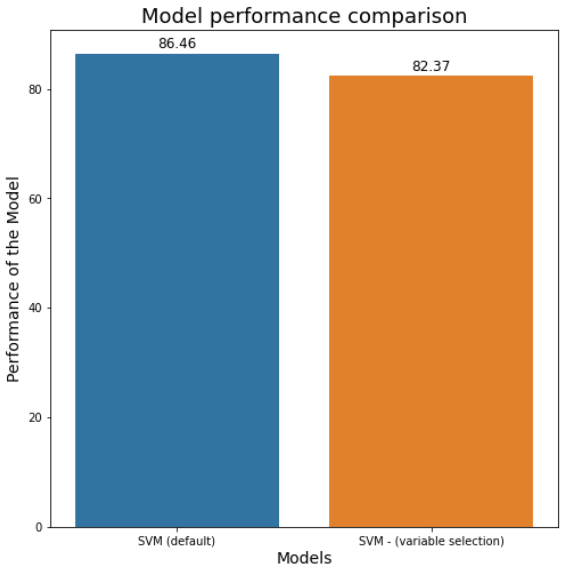
We have used the Lasso method to select the important features from the dataset. After applying lasso feature selection, we are left with only 8 features in the dataset.

That means, we have removed 96.5 % of the unwanted features from the data.

However, the accuracy of the model is still 82.37% which is reasonably good. By reducing 96.5% of input features, we still are getting 82.37% accuracy which is just 4.09% less than original.



[Figure 2]



[Figure 3]

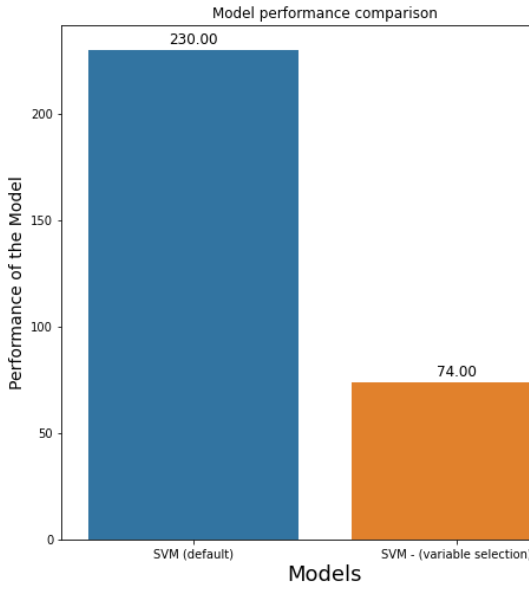
4. Bi-directional elimination:

Also called Stepwise selection[15] method is used as a wrapper method to the SVM model. This method selects important features based on the model used. After applying

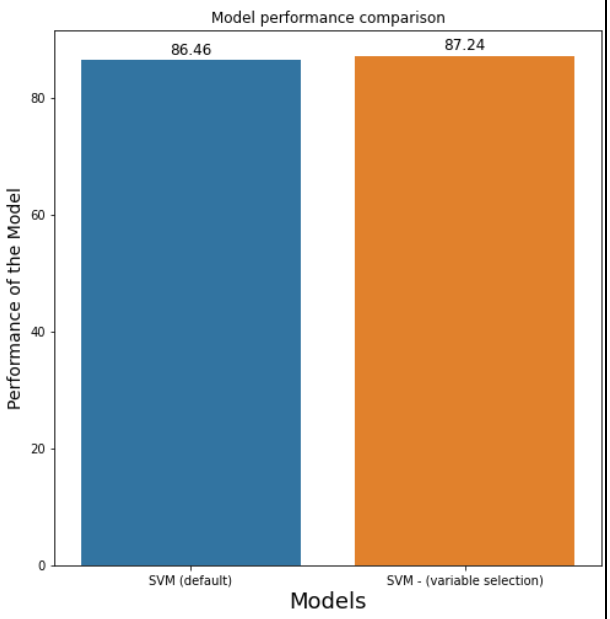
this method, we get 74 features which are useful for producing the results. After feature selection, the SVM model performs even better than the original…!

From figure [Figure 5] we can clearly see the improvement in the performance. By reducing the input features by 67.8%, we are able to get almost ~1% increase in accuracy.

Also, by feature selection, the input data is reduced which means model size is reduced and in result computation power is saved. So, with 67% reduction, we save almost 70% of the model size and computation power.



[Figure 4]

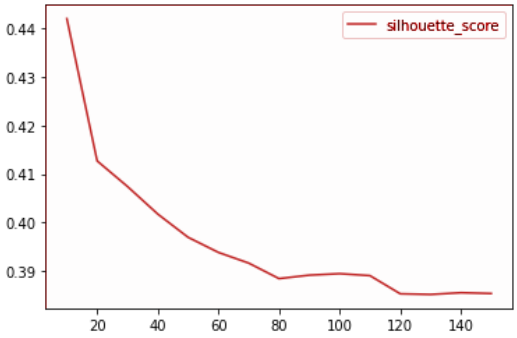


[Figure 5]

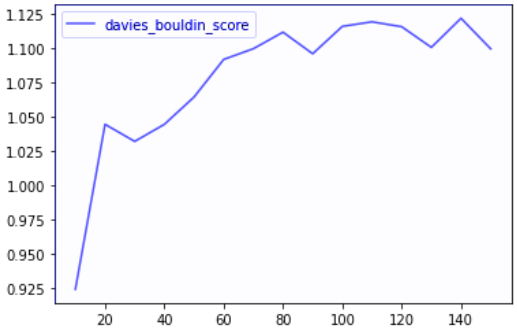
5. Clustering

For clustering we have used the K-means algorithm. For finding the best number of clusters, we ran the algorithm from k=2 to k=160. Also, we evaluated the resultant clusters using cluster evaluation techniques like Silhouette[16], Calinski Harabasz[17] and Davies-Bouldin score. Below figures [Figure 6] [Figure 7] [Figure 8] show the error rate for each k=#. However, from all the evaluation methods below, the best score is provided at 2 clusters. Silhouette has a score of 0.445 which is highest at k=2. Davies' bouldin score represents the average similarity between different clusters and its lowest at k=2. That means both the clusters are very distinct. Also, calinski score has the highest value at k=2. All the evaluation methods point to the same k value which is 2.

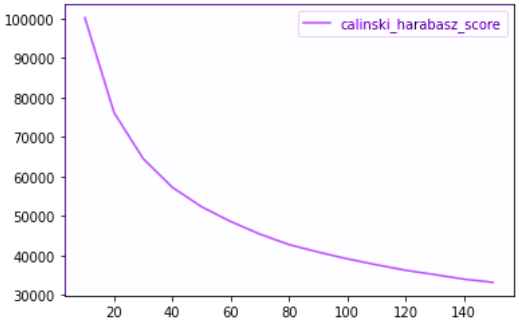
Also, our data is a binary classification data which has only 2 classes, that means we can use a clustering method with k=2 on this data to use it as a predictive model.



[Figure 6]



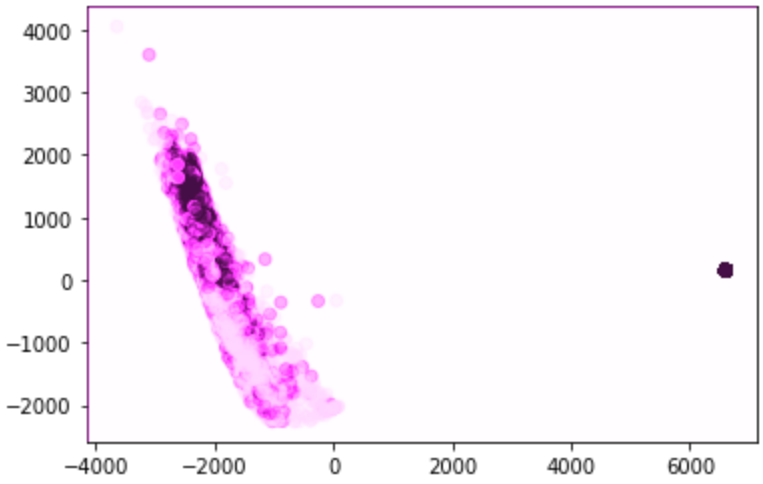
[Figure 7]



[Figure 8]

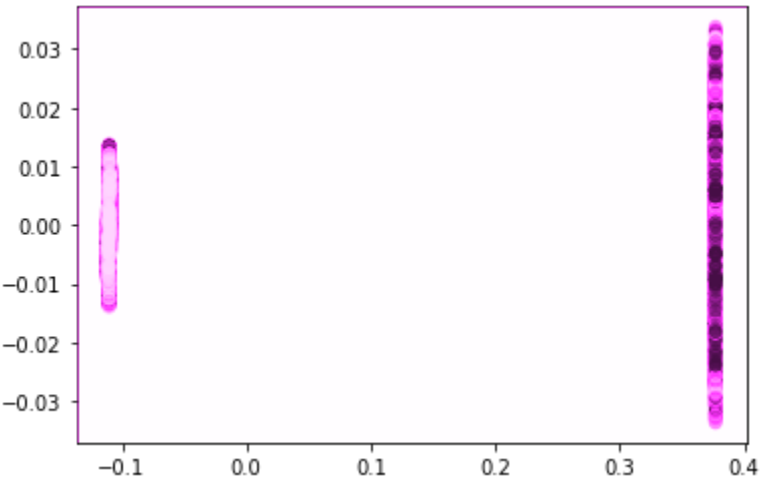
6. Visualization using Dimensionality Reduction:

For Visualization, we have used PCA (Principal Component Analysis) to reduce the dimensions and plot it on a 2d graph. The following figure shows the 2d representation of the data. The light color represents commercial and dark color represents non-commercial. The linear method is not able to perform well on reducing the dimensions. There is a high overlap of the data.



[Figure 9]

The following figure [Figure 10] is visualized using nonlinear dimensionality reduction technique. We have used Kernel PCA for this process. By using the sigmoid kernel, we get reasonably good results. Though there is some overlap, the separation is very high and clearly understandable. This shows that the kernel methods perform better on reducing dimensions for high dimensional data.



[Figure 10]

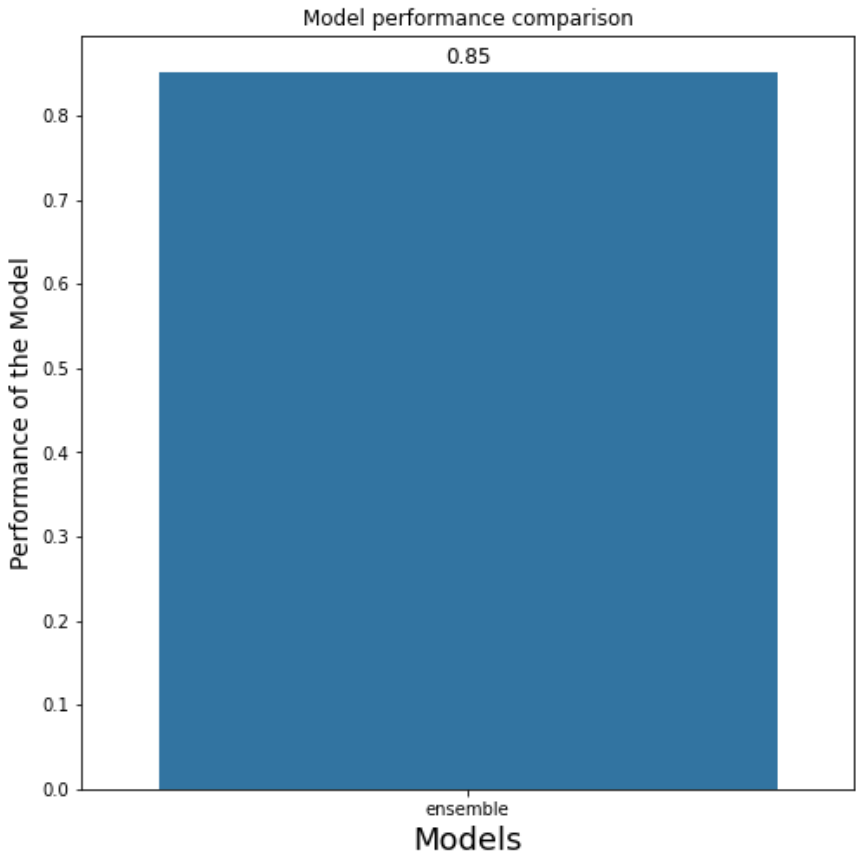
7. Ensemble Modeling:

We have used a total of 20 different models in the ensemble method. It comprises 8 Random Forest models, 8 K Nearest neighbor models and 4 Ridge classifiers. Each model selected performs reasonably well on the training and validation data.

Theoretically the performance of the models should increase, however the models used inside the ensemble are not always the best.

We have used a voting method to combine the results of all the models. All the 20 models return their output to the voting wrapper function. This function assigns weights to each model based on its trustability. If a model provides correct results most of the time during training, the weight is increased else vice versa.

Finally, the ensemble model gives an accuracy of 85.%.



[Figure 10]

8. General discussion:

We have implemented all types of models starting from simple machine learning to deep learning models. From the results we have observed that SVM performs consistently better. In visualization, nonlinear techniques are performing well compared to linear techniques. We can clearly see the separation of class 0 and 1 in the visualization graph.

For clustering, we have used k-means algorithm from which we get k=2 as the best number of clusters.

For variable selection we have used Lasso and bidirectional elimination and best results come from bidirectional elimination.

9. Conclusion

To conclude the report, we have used SVM as the best model. Before variable selection, its accuracy was around 87% and after the variable selection it has improved by ~1%. However, when we consider the performance,

10. References

[1]http://archive.ics.uci.edu/ml/datasets/tv+news+channel+commercial+detection+dataset

[2]http://archive.ics.uci.edu/ml/datasets/tv+news+channel+commercial+detection+dataset

[3]https://en.wikipedia.org/wiki/LIBSVM

[4]https://en.wikipedia.org/wiki/Comma-separated_values

[5]https://scikit-learn.org/stable/modules/generated/sklearn.dummy.DummyClassifier.html

[6]https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.RidgeClassifier.html

[7]https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html

[8]https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[9]https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

[10]https://scikit-learn.org/stable/auto_examples/svm/plot_svm_nonlinear.html

[11]https://scikit-learn.org/stable/modules/neural_networks_supervised.html

[12]https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

[13]https://scikit-learn.org/stable/modules/ensemble.html

[14]https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Lasso.html

[15]https://scikit-learn.org/stable/modules/feature_selection.html

[16]https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html

[17]https://scikit-learn.org/stable/modules/generated/sklearn.metrics.calinski_harabasz_score.html