PROJECT REPORT ON

PREDICTING POPULARITY OF ONLINE ARTICLE

SUBMITTED BY:

Abhishek Kumar
Akash Kumar Singh
Aman Raj
Amit Kumar Singh

UNDER THE VALUABLE GUIDANCE OF:

Prof. Rumpa Hazra

CONTENTS

OBJECTIVE

To predict popularity of online article using a broad set of extracted features known before their publication. Then, optimizes a subset of the article's features to enhance popularity which is measured by considering the number of shares on social media.

We adopt the common binary (popular/unpopular) task and test four classification models namely Linear Regression, Logistic Regression, Neural Network and SVM.

INTRODUCTION

With the expansion of the Internet, there has been a growing interest in online news since it allows an easy and fast spread of information around the globe. Thus, predicting the popularity of online news is becoming a recent research trend. Popularity is often measured by considering the number of interactions in the Web and social networks (e.g., number of shares, likes and comments).

Predicting such popularity is valuable for authors, content providers, advertisers and even activists/politicians. There are two main popularity prediction approaches: those that use features only known after publication and those that do not use such features. The first approach is more common. Since the prediction task is easier, higher prediction accuracies are often achieved. The latter approach is more scarce and, while a lower prediction performance might be expected, the predictions are more useful, allowing to improve content prior to publication.

DATA DESCRIPTION

DATA ACQUISITION

The data was collected from the UCI Machine Learning repository. It contains the content of all the articles published during a two-year period on Mashable (mashable.com/), which is one of the largest news websites.

(https://archive.ics.uci.edu/ml/datasets/Online+News+Popularity)

DATASET DESCRIPTION

The prediction module uses a large list of inputs that includes features like: digital media content (e.g., images, video); earlier popularity of news referenced in the article; average number of shares of keywords prior to publication; and natural language features (e.g., title polarity, Latent Dirichlet Allocation topics). The attributes are broadly categorized into to six categories – words, links, digital media, time, keywords and NLP

DATA SET ATTRIBUTES

FEATURE	TYPE(#)
WORDS	
Number of words in the title	Number(1)
Number of words in the article	Number(1)
Average word length	Number(1)
Rate of non-stop words ratio (1)	Ratio(1)
Rate of unique words	Ratio(1)
Rate of unique non-stop words	Ratio(1)
LINKS	
Number of links	Number(1)
Number of Mashable article links	Number(1)
Minimum, average and maximum number of shares of Mashable links	Number(3)
DIGITAL MEDIA	
Number of images	Number(1)
Number of videos	Number(1)
TIME	
Day of the week	Bool(7)
Published on a weekend?	Bool(1)
KEYWORDS	
Number of keywords	Number(1)
Worst keyword (min./avg./max. shares)	Number(3)
Average keyword (min./avg./max. shares)	Number(3)
Best keyword (min./avg./max. shares)	Number(3)
Article category (Mashable data channel)	Bool(7)

NATURAL LANGUAGE PROCESSING	
Closeness to top 5 LDA topics	Ratio(5)
Title subjectivity	Ratio(1)
Article text subjectivity score and its absolute difference to 0.5	Ratio(2)
Title sentiment polarity	Ratio(1)
Rate of positive and negative words	Ratio(2)
Pos. words rate among non-neutral words ratio	Ratio(1)
Neg. words rate among non-neutral word	Ratio(1)
Polarity of positive words (min./avg./max.)	Ratio(3)
Polarity of negative words (min./avg./max.)	Ratio(3)
Article text polarity score and its absolute difference to 0.5	Ratio(2)

Target	TYPE(#)
Number of article Mashable shares	Number(1)

DATA MINING

1. Remove non predictive features– URL of article and time delta (which is Days between the article publication and the dataset acquisition.) were removed since these are non-predictive features that were not used in prediction module.

2. Removing Outliers using Z-Score: The approach was to remove the outlier points by eliminating any points that were greater than (μ+ kσ) and less than (μ-kσ).

For eg : The feature 'n_tokens_title'. The mean for this feature was found to be 10.21 and Standard Deviation was found to be 2.14.

Therefore, Lower threshold = μ-kσ
Upper threshold = μ+kσ

Here k was chosen to be 3, for moderate outlier's removal.

FETURE SELECTION USING FISHER SCORES

The Fisher Score of the jth feature (for a binary classification problem) is given by:

$$F(j) = \frac{(\bar{x}_j^1 - \bar{x}_j^2)^2}{(s_j^1)^2 + (s_j^2)^2}$$

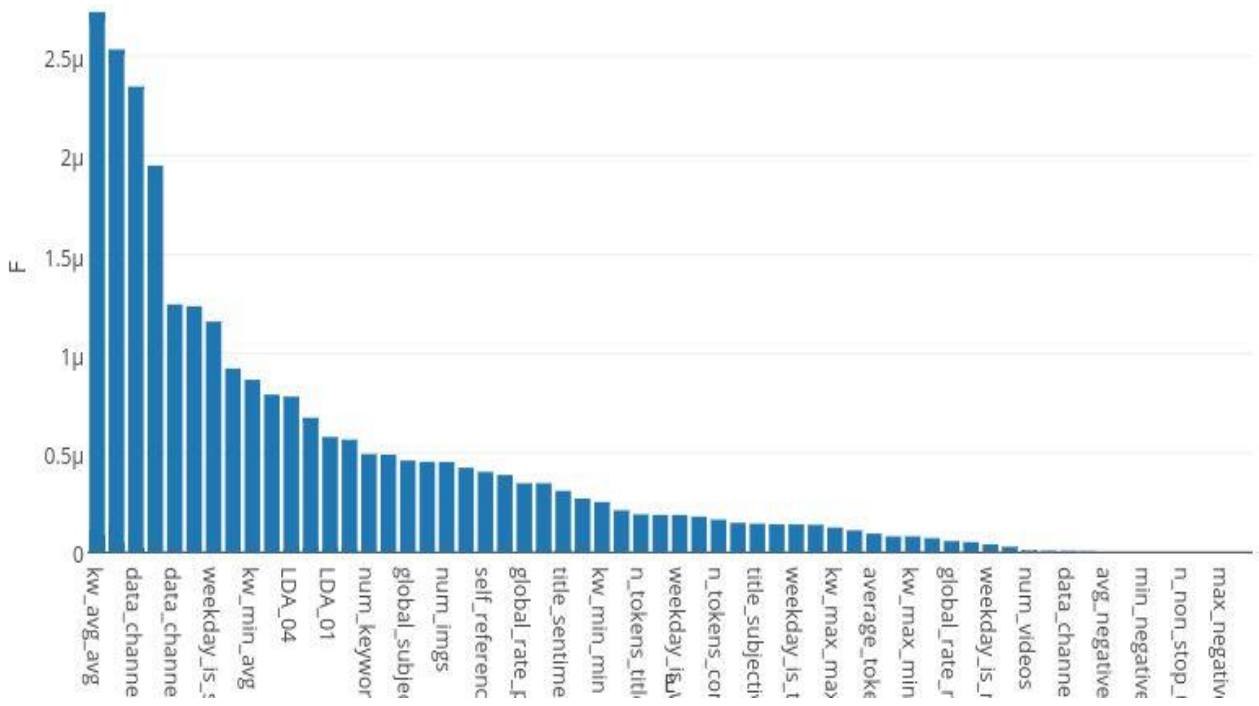
where

$$(s_j^k)^2 = \sum_{x \in X^k} (x_j - \bar{x}_j^k)^2$$

We aim to maximize the between class scatter – the numerator, and to maximize the within class scatter – the denominator.

Thus, higher the F score, the more likely it is that feature is more discriminative. We selected 20 features with the highest F scores.

FISHER SCORES



Selected features are following:

- kw_avg_wrd
- is_weekend
- LDA_04
- kw_max_avg
- Num_hrefs
- num_imgs
- num_keywords
- LDA_02
- data_channel_is_socmed
- data_channel_is_entertainment
- weekday_is_sunday
- global_subjectivity
- global_sentiment_polarity
- LDA_01
- data_channel_is_word
- weekday_is_saturday
- data_channel_is_tech
- LDA_00
- kw_min_avg
- rate_negative_words

DATA PREPARATION

1.Binary classification task- We assume a binary classification task, where an article is considered \popular" if the number of shares is higher than a fixed decision threshold (D1), else it is considered \unpopular.

For defining a popular class, we used a fixed value of D1 = 1400 shares, which resulted in a balanced popular or unpopular class distribution in the training set.

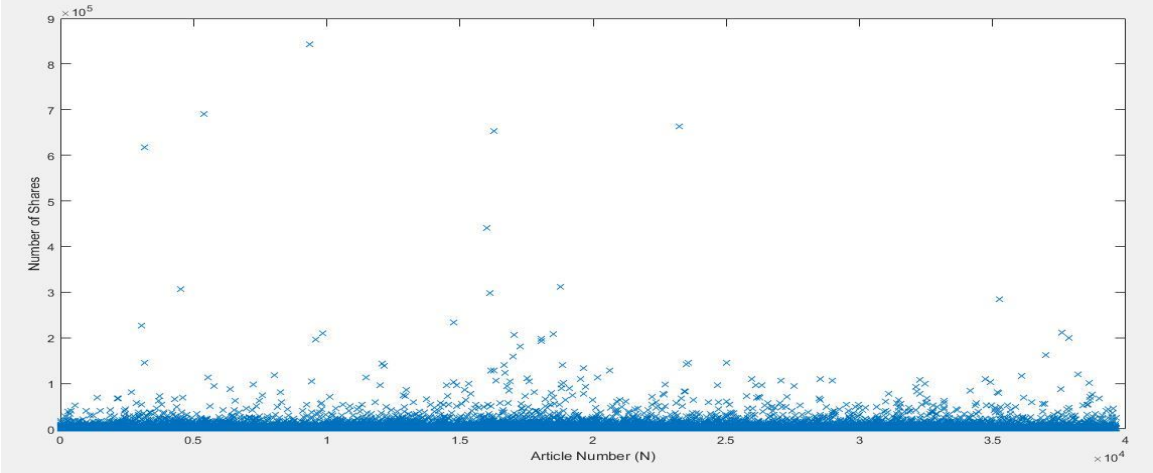


Figure 2 Number of Shares of each article

For chosen Threshold (D1) = 1400

No. Of articles with shares > 1400 = x1 = 18490

No of articles with shares < 1400 = x2 = 21153

Which gives P(x1) = 0.533587 and P(x2) = 0.466413

Data is not skewed

SUPERVISED LEARNING MODELS

We have applied following supervised learning models - Linear Regression Logistic Regression, Neural Networks and Support Vector Machine (SVM)

LINEAR REGRESSION

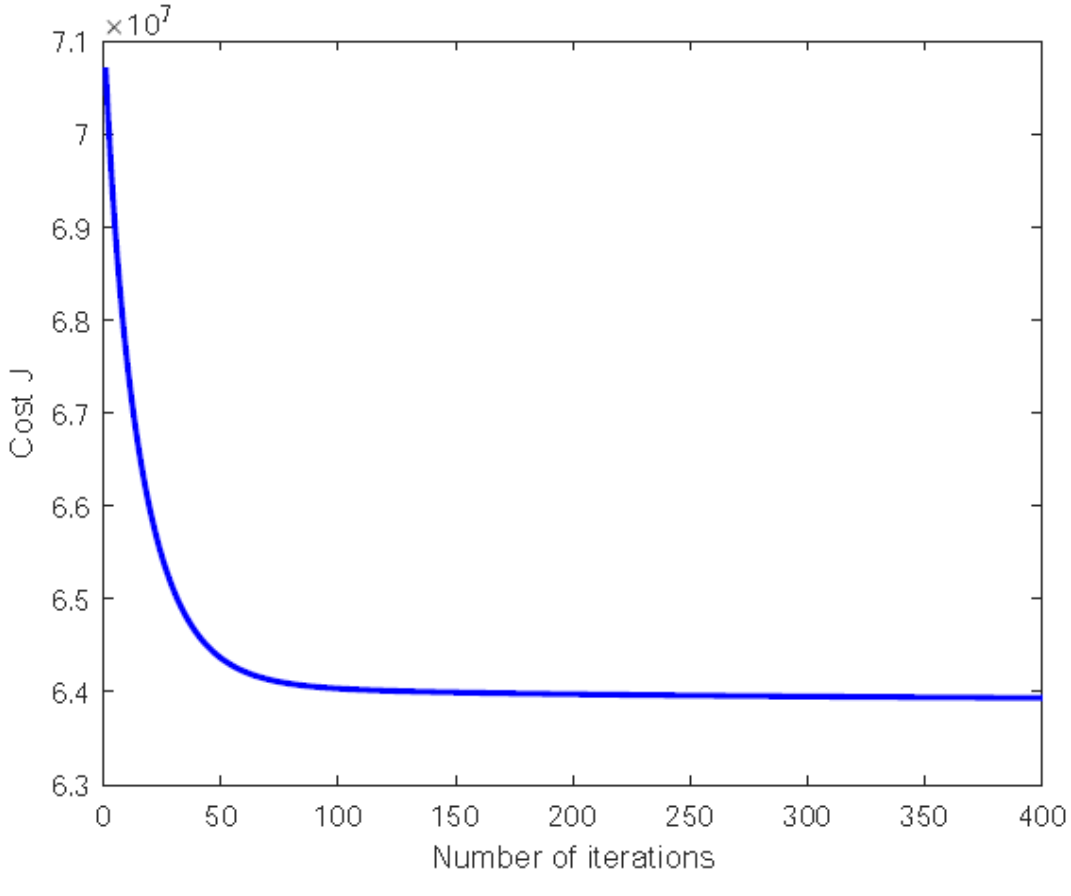
Steps involved:

The data set is trained using K-fold validation (with K=5). Our average accuracy will be calculated from mean of these five accuracies.

This is followed by feature normalization. It is used to standardize the range of independent variables or features of data.

We employ Gradient Descent method to find straight line with least cost. We have used the learning rate (alpha) as 0.03 and the number of iterations value 400. This will give us the appropriate thetas.

The average accuracy is 52.44% for this model. And the plot for number of iterations vs Cost function is



LOGISTIC REGRESSION

Steps involved:

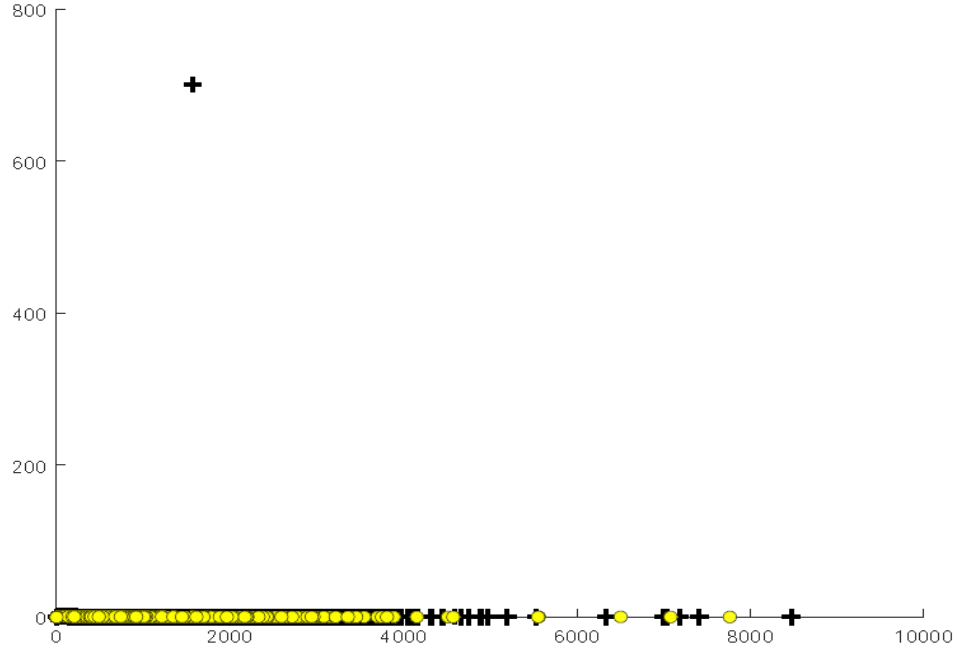
Initial steps are same as Linear regression till feature normalization.
The difference is in computation of cost function.

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left[-y^{(i)}\log(h_\theta(x^{(i)})) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2.$$

We regularize the cost function.
We add an additional parameter to our cost function (J) that increases as the value of your parameter weights (w) increase; keep in mind that the regularization we add a new hyperparameter, lambda, to control the regularization strength.
Here, we choose lambda as one.

The accuracy is nearly 65%.

After that we have plotted the decision boundary.



ARTIFICIAL NEURAL NETWORKS

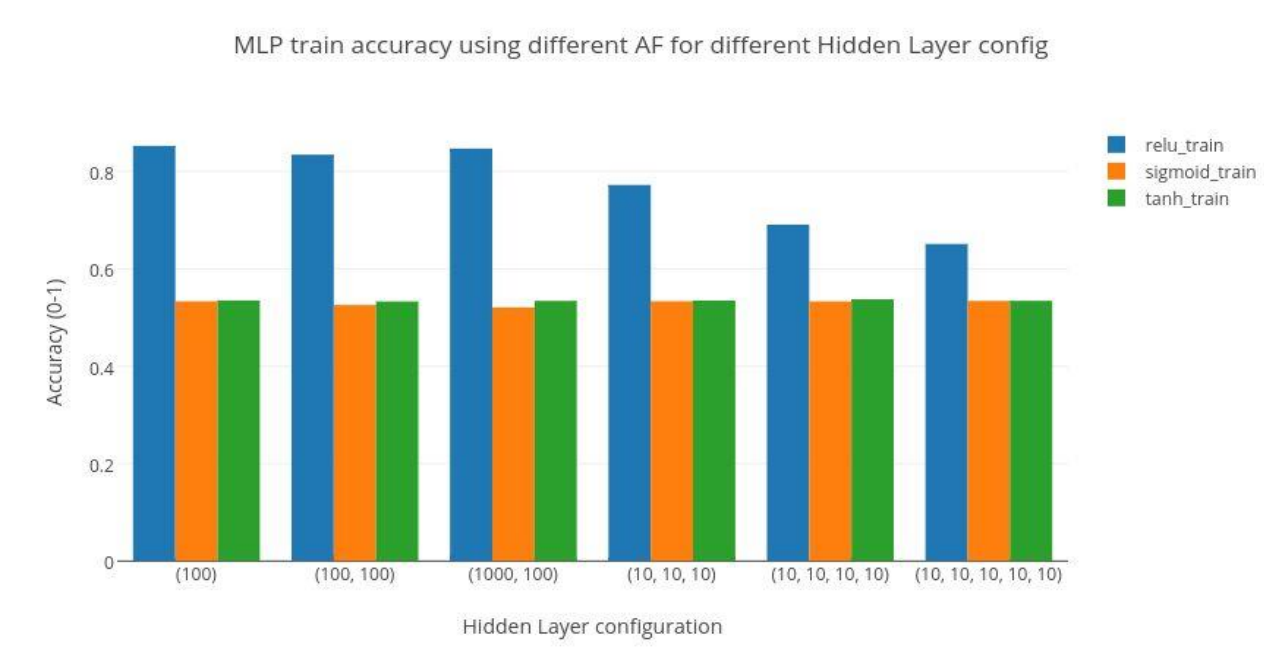
We've used scikit-learn for neural nets, also known as the Multi-Layer Perceptron Classifier – MLPClassifier, which uses a 'relu' (rectified linear unit function f(x) = max (0, x)) activation function by default, and we've tried sigmoid and tanh activation function

If we used 20 features which were taken at the start, with 10 epochs and cv of 70:30, 4 hidden layers of size 10 each we got an average accuracy of about 55-57%, with across all activation functions.

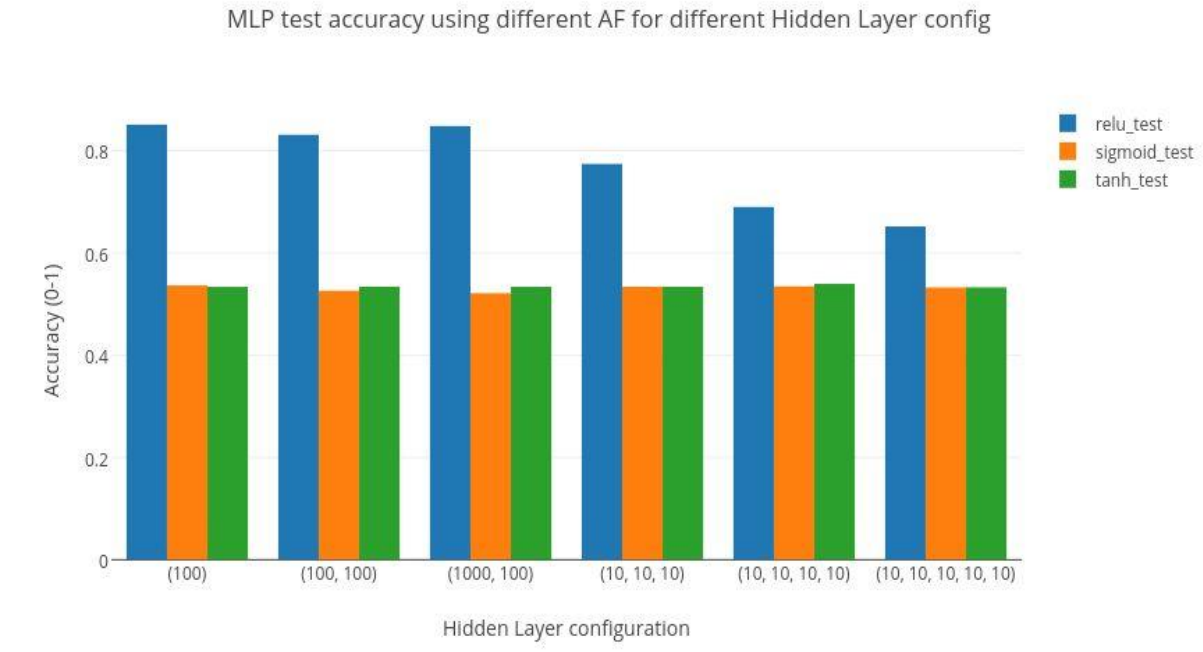
We decided to run it on **ALL** the features, and with 10 epochs and cross-validation of 70-30, and running a NN with 4 hidden layers and 10 layers size each. Using tanh and sigmoid, we achieved accuracy of ~56%, but surprisingly, for ReLU activation function, average accuracy came out to be 72%. On closer examination, we found out that either the accuracy came out to be ~53-56%, or it was ~79-89% (more than half the time), and very rarely in between. This was spread out across multiple configurations of hidden layer sizes and numbers (data on the next slides).

After varying the hidden layers and sizes, maximum accuracy in one epoch was once reported to be 92.6% at 1 hidden layer of 100 size, and it also had the maximum average accuracy. The model was re-run for 10 epochs again and the accuracies were reported to be similar.

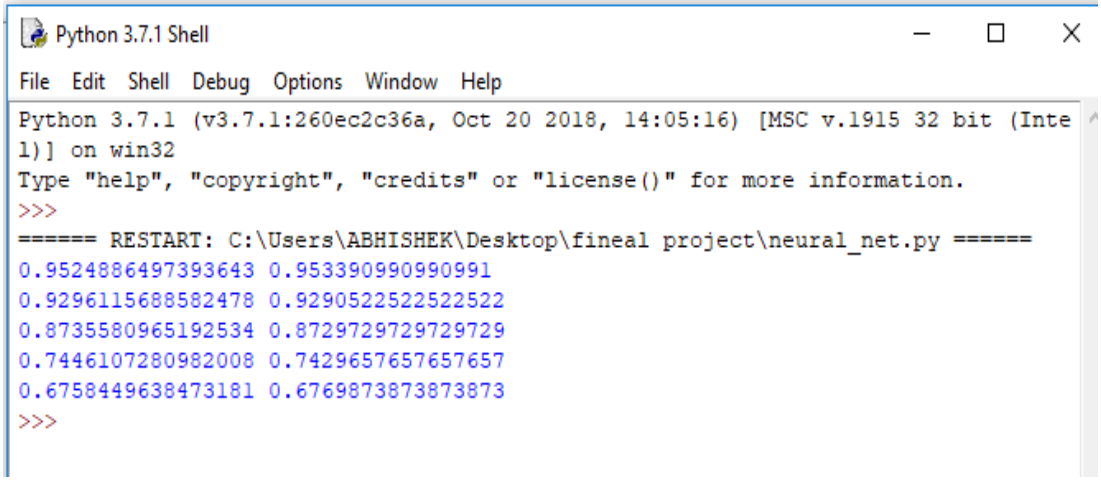
NEURAL NETWORKS ACROSS CONFIGURATIONS - TRAIN



NEURAL NETWORKS ACROSS CONFIGURATIONS – TEST



OUTPUT OF ANN MODEL



SUPPORT VECTOR MACHINE

We've used scikit-learn SVM linear and Radial basis Function kernel with cv randomization for classification. The result showed that mapping data to higher dimensional space using an RBF kernel was not useful.

Average accuracy obtained using linear kernel is 60.5% whereas accuracy obtained using RBF kernel is 52.45%

```
# 0 SVM with linear kernel, with cv randomization = 57.99999999999999 %
# 0 SVM with rbf kernel, with cv randomization = 55.666666666666664 %
# 1 SVM with linear kernel, with cv randomization = 56.333333333333336 %
# 1 SVM with rbf kernel, with cv randomization = 58.333333333333336 %
# 2 SVM with linear kernel, with cv randomization = 58.666666666666664 %
# 2 SVM with rbf kernel, with cv randomization = 54.0 %
# 3 SVM with linear kernel, with cv randomization = 57.666666666666664 %
# 3 SVM with rbf kernel, with cv randomization = 57.666666666666664 %
# 4 SVM with linear kernel, with cv randomization = 61.0 %
# 4 SVM with rbf kernel, with cv randomization = 56.99999999999999 %
# 5 SVM with linear kernel, with cv randomization = 57.666666666666664 %
# 5 SVM with rbf kernel, with cv randomization = 54.0 %
# 6 SVM with linear kernel, with cv randomization = 59.333333333333336 %
# 6 SVM with rbf kernel, with cv randomization = 57.666666666666664 %
# 7 SVM with linear kernel, with cv randomization = 55.666666666666664 %
# 7 SVM with rbf kernel, with cv randomization = 49.0 %
# 8 SVM with linear kernel, with cv randomization = 57.99999999999999 %
# 8 SVM with rbf kernel, with cv randomization = 55.00000000000001 %
# 9 SVM with linear kernel, with cv randomization = 59.66666666666667 %
# 9 SVM with rbf kernel, with cv randomization = 56.333333333333336 %
```

RESULTS

RESULTS OBTAINED

MODEL	ACURACCY
Linear Regression	0. 52441261
Logistic Regression	0.65328467
ANN	0.92820180
SVM	0.6054 (Linear) 0.5245 (RBF)

RESULTS OF REFERRED PAPERS

MODEL	ACURACCY
Linear Regression	0. 52441261
Logistic Regression	0.65328467
ANN	0.92820180
SVM	0.6054 (Linear) 0.5245 (RBF)

CONCLUSION

We've reached a maximum accuracy (avg.) of 92% with MLP classifier but with some variance as well, and all others vary from 53% to 66%. Most of these accuracies are similar to what has been reported in the paper.

REFERENCES

1.) Kelwin, Fernandes, K., Vinagre, P., & Cortez, P. (2015). A Proactive Intelligent Decision Support System for Predicting the Popularity of Online News.

2.) *Online News Popularity Data Set*. (2015). Retrieved from UCI REPOSITORY: https://archive.ics.uci.edu/ml/datasets/Online+News+Popularity

3.) Liu, Ziyi, "Statistical Models to Predict Popularity of News Articles on Social Networks" (2017). Arts & Sciences Electronic Theses and Dissertations. 1052. https://openscholarship.wustl.edu/art_sci_etds/1052

Thank you.