

# Applied Data and Text Analytics

## Coursework Report

UP781439

22nd January 2021



**UNIVERSITY OF  
PORTSMOUTH**

School of Computing  
University of Portsmouth

# Contents

<b>Part 1</b>	<b>2</b>
Descriptive Analytics	2
Distribution	2
Outliers	3
Correlation/Relationship Between Attributes	4
Clustering	4
K-means Clustering	5
Hierarchical Clustering	6
Comparison	6
Classification	6
SVM	6
Bagging	6
Neural Network	7
Comparison	7
Association Rule Mining	7
Rules:	8
[1_Ethnic Group, 1_Population Base] => [1_Country of Birth]	8
[2_Student, 1_Ethnic Group,H_Residence Type] => [8_Age]	8
[-9_Hours Worked Per Week] => [2_Student, 1_Population Base]	8
[H_Resident Type] => [1_Population Base]	8
[1_Student, 1_Sex] => [6_Economic Activity]	8
Regression	9
Regression Tree Ensemble	9
Simple Linear Regression	9
Comparison	9
<b>Part 2</b>	<b>11</b>
Preprocessing Text Data	11
Distances Between Word Embeddings	12
Clustering	14
Classification	16
LSTM	16
SVM	18
Naive Bayes	18
Comparison	19
<b>Appendices</b>	<b>20</b>

# Part 1

## 1. Descriptive Analytics

In this section the findings of the analysis process will be discussed and interpreted.

### 1.1. Distribution

There are a number of attributes with only 2 categories (sex, student, residence type, country of birth) in the data set which makes looking at the distribution of those attributes much simpler. For these attributes I have used a pie chart to visualise the ratio between the two categories.

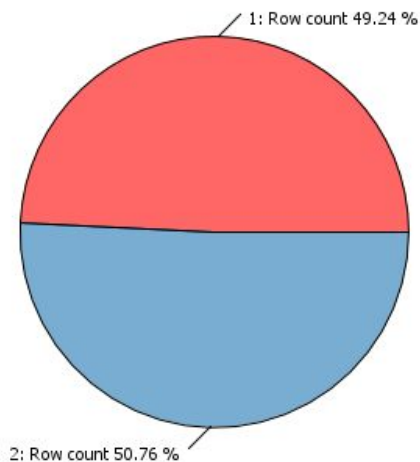


Fig. 1 Sex pie chart (1 = male, 2 = female)

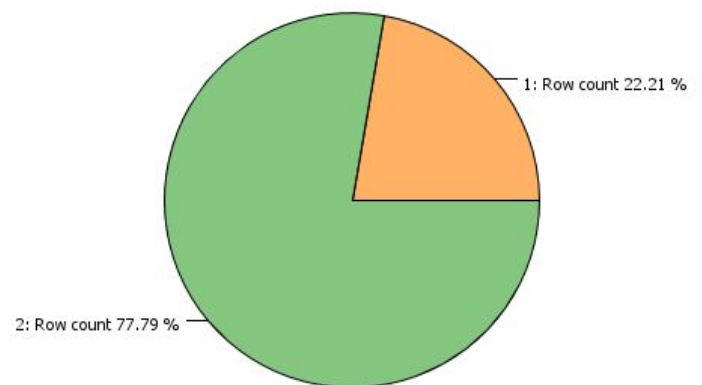


Fig. 2 Student pie chart (1 = student, 2 = non-student)

Figures 1 and 2 show the pie charts for sex and student status respectively. As you can see in Fig. 1, the data is split almost perfectly 50/50 in terms of gender (49.24% male, 50.76% female). This is good as it means we have an evenly distributed dataset in terms of gender which should result in a more generalisable classification model. While the student pie chart, Fig. 2, shows a large disparity between the categories, this is not unexpected as the majority of people are not students. When analysing the 'residence type' attribute it was found that 98.13% of people in the census live in non-communal residences, meaning 1.87% live in communal residence. Finally it was found that for people, excluding those for which a code was not required, in the census; 86.27% born in UK, 13.73% non-UK born.

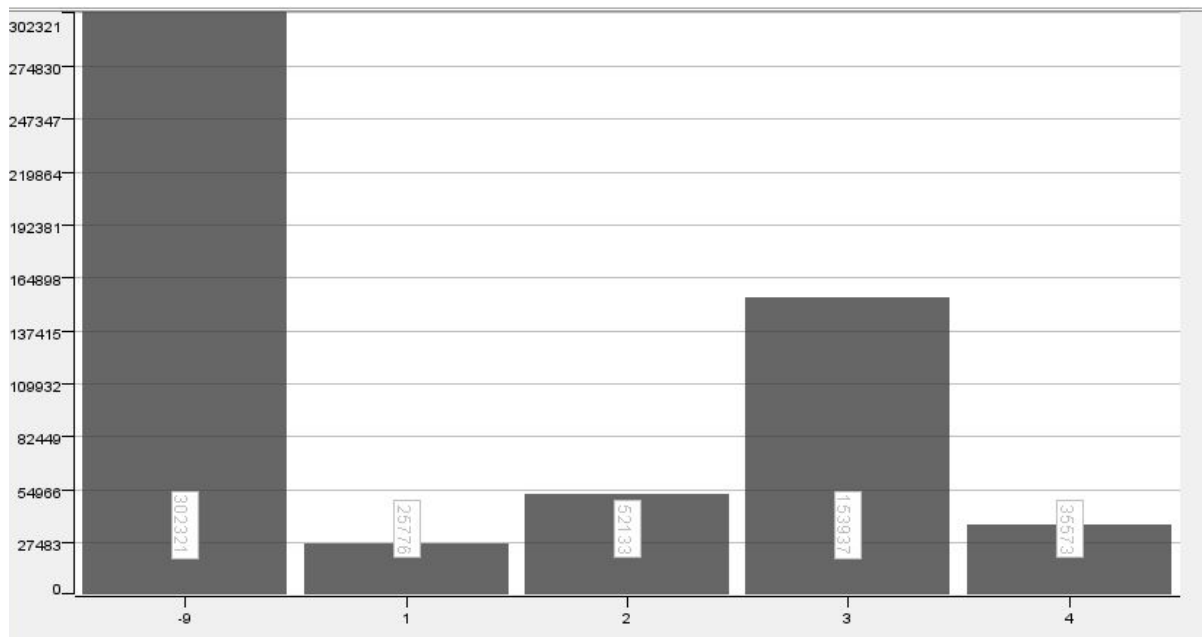


Fig. 3 Histogram showing distribution of 'Hours worked per week'

Fig. 3 shows that the majority, 53.06%, of people in the census were in the 'no code required' class for the hours worked attribute. The largest class of people who were eligible for classification was '3', those who worked Full-time for 31 to 48 hours. (See appendices A-L for remaining attribute distributions).

## 1.2. Outliers

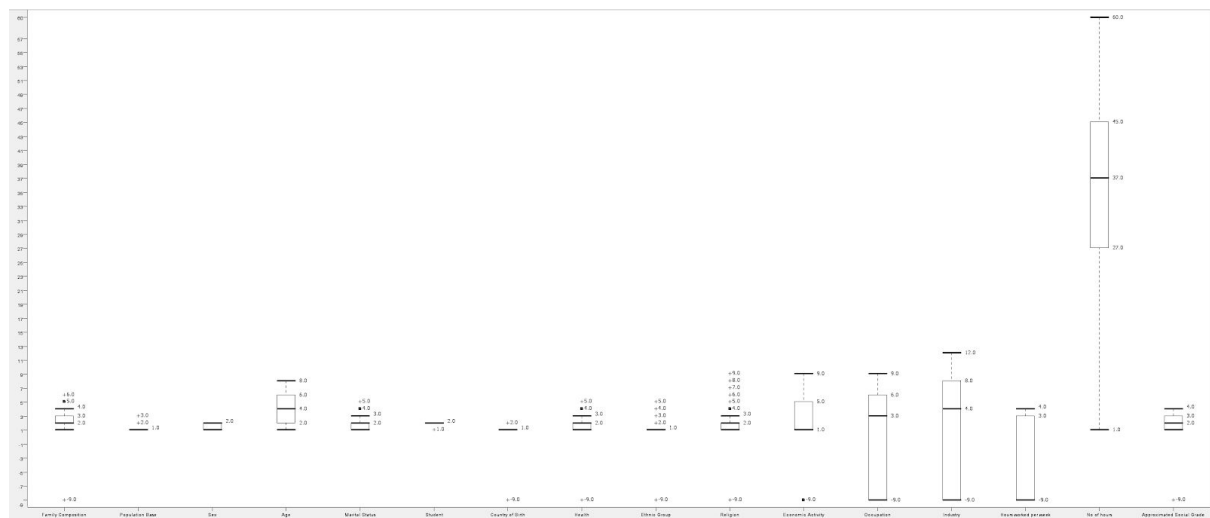


Fig. 4 Box plot analysis of all attributes

Fig. 4 shows that the attribute with the highest number of extreme outliers is the Religion attribute with 6 extremes and one mild outlier. Ethnic group attribute is second with 5 Extreme outliers. These large numbers of outliers suggest that the data is not distributed evenly and is heavily skewed with the majority of the data within 1 or 2 classes. Other attributes that contain outliers are Family Composition, Population Base, Marital Status, Health, Economic Activity and Economic Social Grade.

### 1.3. Correlation/Relationship Between Attributes

#### Cross Tabulation of Country of Birth by Ethnic Group

Frequency Row Percent	-9	1	2	3	4	5	Total
-9	6,804						6,804
	100%						
1		448,049	9,674	17,574	8,668	1,680	485,645
		92.2585%	1.992%	3.6187%	1.7848%	0.3459%	
2		35,428	2,535	25,137	10,118	4,073	77,291
		45.8372%	3.2798%	32.5225%	13.0908%	5.2697%	
Total	6,804	483,477	12,209	42,711	18,786	5,753	569,740

☒ Frequency  
☐ Expected  
☐ Deviation  
☐ Percent  
☒ Row Percent  
☐ Column Percent  
☐ Cell Chi-Square  
  
Max rows:   
Max columns:

#### Statistics for Table of Country of Birth by Ethnic Group

Statistic	DF	Value	Prob
Chi-Square	10	689,909.6546	0.0

Fig. 5 Crosstab and Chi-Square for country of birth by ethnic group

For the attributes in Figure 5, as well as every other, the chi-square value was so high that the probability value displayed was '0.0'. This made it difficult to understand whether or not the results were significant, however for some attributes the bivariate correlation can be seen by simply looking at the cross tabulation. For example, Figure 5 shows a clear relationship between being born in the UK and being a white person (92% of people born in the UK are white). Scatter plots on this data look like a grid of squares because the data is categorical, therefore scatter plots are not very useful in analysing this dataset.

## 2. Clustering

The first task was to find an estimation of an optimal K value for the K-means algorithm. This was done by iterating over lots of values of K, 1 - 16, and evaluating the entropy score of each. When K = 14 the entropy was found to be lowest.

Row ID	I Size	D ▲ Entropy	D ▼ Quality	I currentIteration	I maxIterations	I K	S RowID	S knime.workspace	I Iteration
Overall#12	284870	1.213	... 0.478	12	14	14	Row12	C:\Users\samto\knime-workspace	12
Overall#13	284870	1.213	... 0.478	13	14	15	Row13	C:\Users\samto\knime-workspace	13
Overall#10	284870	1.234	... 0.468	10	14	12	Row10	C:\Users\samto\knime-workspace	10
Overall#11	284870	1.253	... 0.46	11	14	13	Row11	C:\Users\samto\knime-workspace	11
Overall#9	284870	1.4	... 0.397	9	14	11	Row9	C:\Users\samto\knime-workspace	9
Overall#8	284870	1.4	... 0.397	8	14	10	Row8	C:\Users\samto\knime-workspace	8
Overall#7	284870	1.518	... 0.346	7	14	9	Row7	C:\Users\samto\knime-workspace	7
Overall#6	284870	1.587	... 0.317	6	14	8	Row6	C:\Users\samto\knime-workspace	6
Overall#5	284870	1.593	... 0.314	5	14	7	Row5	C:\Users\samto\knime-workspace	5
Overall#4	284870	1.706	... 0.265	4	14	6	Row4	C:\Users\samto\knime-workspace	4
Overall#3	284870	1.76	... 0.242	3	14	5	Row3	C:\Users\samto\knime-workspace	3
Overall#2	284870	1.76	... 0.242	2	14	4	Row2	C:\Users\samto\knime-workspace	2
Overall#1	284870	1.774	... 0.236	1	14	3	Row1	C:\Users\samto\knime-workspace	1
Overall#0	284870	1.791	... 0.229	0	14	2	Row0	C:\Users\samto\knime-workspace	0

Fig. 6 Entropy scores for different K values

## 2.1. K-means Clustering

Fig. 7 shows the distribution of data within the clusters that were created with this algorithm.

Cluster
No. missings: 0
<b>Top 20:</b>
cluster_12 : 20506
cluster_1 : 9327
cluster_0 : 8270
cluster_2 : 8246
cluster_13 : 6276
cluster_3 : 5961
cluster_8 : 5841
cluster_4 : 4558
cluster_5 : 3319
cluster_6 : 3223
cluster_10 : 3135
cluster_9 : 3028
cluster_11 : 2726
cluster_7 : 1045
<b>Bottom 20:</b>

Fig. 7 Distribution of data within clusters

When looking at the scatter plot of age against clusters there is little useful information to be gathered because everyone in the dataset is split into 8 categories of age as opposed to having someone's exact age. However, the insight that can be gained from this process is that clusters 13, 3 and 2 contain predominantly younger people and the rest of the clusters have near enough even distribution across the age ranges. The distribution of other attribute categories within clusters can be seen and patterns do emerge, such as for the ethnic group attribute cluster 7 contains all persons in the '-9' category.

## **2.2. Hierarchical Clustering**

I used the distance matrix hierarchical clustering algorithm with the Manhattan distance function. Using the row filter to isolate the clusters I am able to take a much closer look at the clusters than I was with the k means clusters. For example, I can see histograms of the distribution of data within the clusters such as that for the second largest cluster, cluster 2, there are many more students than not, and this cluster contains the majority of students in the dataset.

## **2.3. Comparison**

Comparing the two I think they are algorithms that are best suited to different types of data. The hierarchical clustering algorithm offers more useful insight into the clusters being created when working with categorical data. While if we were working with more numerical, continuous data attributes then k-means clustering could produce better results. One downside of the hierarchical clustering algorithm is that it is a very computationally expensive algorithm that will take a very long time to execute. This potentially created an unfair comparison between the algorithms as I was only able to use 2500 rows in the hierarchical algorithm, out of the entire 199,409 rows that were used in the K-means algorithm.

## **3. Classification**

The algorithms used to classify the data were Support Vector Machine (SVM), Bagging with the Naive Bayes algorithm and a multilayer perceptron (MLP) neural network.

### **3.1. SVM**

When trained to make predictions on the region attribute, the SVM performed very poorly achieving an accuracy score of only 12.2%. This could suggest that the other attributes in the data set are not indicators of where someone will be located. However, the main reason for the poor accuracy is that this is a multiclass problem and the vanilla SVM algorithm does not perform well on these. Training the SVM to make predictions on the binary attribute, Residence Type, should produce much better results. As predicted, the SVM performed much better, scoring 99.9% accuracy.

### **3.2. Bagging**

Region was the first attribute used for prediction. This attribute has 10 classes and it is therefore expected that the naïve bayes algorithm will perform much better on this data than SVM due to it being a multiclass problem. This was confirmed by the results of the first test,

which achieved an accuracy score of 38.114%. While better these results are still poor, so the training/test split was altered to 78/22 and ran the algorithm again. This yielded much better results with an accuracy score of 79.288%. Finally testing on the binary, Residence attribute was conducted. This model predicted the residence type of people perfectly, achieving a score of 100% accurately for the binary classification problem.

### 3.3. Neural Network

For the final classification algorithm, we have again started with making predictions on the Region attribute. The hyper parameters used in the training of the MLP were 15 neurons per layer because there are 15 features of every input; 2 hidden layers; 100 iterations. The network scored an accuracy of 20.156% with these parameters and the 70/30 training/test split. I believe the poor performance is as a result of the MLP overfitting the data. This is a common problem with neural networks and Knime does not allow you to employ techniques such as dropout, weight regularisation or other techniques that reduce overfitting. When classifying the binary class attribute, residence, the MLP scored 99.901% accuracy. Which is near perfect and as expected when only classifying 2 classes.

### 3.4. Comparison

The Naïve Bayes bagging algorithm performed the best out of all three when classifying the region attribute, achieving the highest accuracy of 79.288%. This was 3.9 times higher than the second highest accuracy, 20.156% scored by the MLP. This highlights the advantage of bagging in that the voting system that is created gives you a more generalisable and accurate model. This has also shown that the ‘vanilla’ SVM algorithm does not handle multiclass problems very well. SVM could be made to handle a problem like this using a One against One or One against All.

## 4. Association Rule Mining

Fig. 8 displays some of the rules that were created by the association rule learner. The naming convention used in the rules is [*\*class\*\_\*attribute\**].

[Antecedent]	[S Consequent]	[I ItemSetSupport]	[D = RelativeItemSetSupport%	[D RuleConfidence%	[D AbsoluteBodySetSupport]	[D ▲ RelativeBodySetSupport%	[D RuleLift]	[D RuleLift%
[1_Ethnic Group,H_Residence Type,1_Population Base]	4_Occupation	48138	8.449	10.2	474,139	83.2	1.086	108.62
[1_Ethnic Group,H_Residence Type,1_Population Base]	E12000007_Region	46245	8.468	10.2	474,139	83.2	0.694	69.36
[1_Ethnic Group,H_Residence Type,1_Population Base]	2_Industry	49101	8.618	10.4	474,139	83.2	1.104	110.42
[1_Ethnic Group,H_Residence Type,1_Population Base]	E12000009_Region	49530	8.693	10.4	474,139	83.2	1.107	110.68
[1_Ethnic Group,H_Residence Type,1_Population Base]	9_Occupation	50324	8.833	10.6	474,139	83.2	1.034	103.4
[1_Ethnic Group,H_Residence Type,1_Population Base]	2_Age	51281	9.001	10.8	474,139	83.2	0.847	84.661
[1_Ethnic Group,H_Residence Type,1_Population Base]	5_Family Composition	51504	9.04	10.9	474,139	83.2	0.959	95.923
[1_Ethnic Group,H_Residence Type,1_Population Base]	E12000006_Region	52315	9.182	11	474,139	83.2	1.058	105.81
[1_Ethnic Group,H_Residence Type,1_Population Base]	2_Occupation	54999	9.653	11.6	474,139	83.2	1.031	103.08
[1_Ethnic Group,H_Residence Type,1_Population Base]	3_Age	59428	10.431	12.5	474,139	83.2	0.94	94.026
[1_Ethnic Group,H_Residence Type,1_Population Base]	4_Industry	59839	10.503	12.6	474,139	83.2	1.044	104.39
[1_Ethnic Group,H_Residence Type,1_Population Base]	6_Age	60716	10.657	12.8	474,139	83.2	1.111	111.11
[1_Ethnic Group,H_Residence Type,1_Population Base]	E12000002_Region	62545	10.978	13.2	474,139	83.2	1.052	105.21
[1_Ethnic Group,H_Residence Type,1_Population Base]	3_Health	64523	11.325	13.6	474,139	83.2	1.041	104.1
[1_Ethnic Group,H_Residence Type,1_Population Base]	4_Age	65986	11.582	13.9	474,139	83.2	1.008	100.83
[1_Ethnic Group,H_Residence Type,1_Population Base]	3_Family Composition	67394	11.829	14.2	474,139	83.2	1.115	111.48
[1_Ethnic Group,H_Residence Type,1_Population Base]	5_Age	68521	12.027	14.5	474,139	83.2	1.064	106.39
[1_Ethnic Group,H_Residence Type,1_Population Base]	1_Approximated Social Grade	72458	12.718	15.3	474,139	83.2	1.058	105.77
[1_Ethnic Group,H_Residence Type,1_Population Base]	3_Approximated Social Grade	73296	12.865	15.5	474,139	83.2	1.102	110.18

Fig. 8 Some of the Association Rules created



## Rules:

### **[1\_Ethnic Group, 1\_Population Base] => [1\_Country of Birth]**

This rule has a relative itemset support of 78.641% but a confidence of 92.8%. This difference in the 2 values shows that while the antecedent as a whole may not appear as frequently as others, the consequent is almost always true for that antecedent itemset. However, having a higher confidence can be misleading as it does not represent the popularity of the consequent, only the antecedent. The lift metric accounts for this and had a value of 1.089. Due to it being greater than 1 it can be said reliably that a person is likely to be born in the UK if they are white and a usual resident.

### **[2\_Student, 1\_Ethnic Group, H\_Residence Type] => [8\_Age]**

The support for this is very low, 6.853% and the confidence is also very low at 10.1%, meaning that only 10% of the time does the antecedent imply the consequent. The lift value is 1.318 meaning that taking into account the popularity of the consequent there is a meaningful association. This shows that both the antecedent and the consequent are not very frequent in the data but where they do occur it is likely that the rule holds true.

### **[9\_Hours Worked Per Week] => [2\_Student, 1\_Population Base]**

The itemset support of this rule is 32.67% and the confidence is 42.1%. This means that the antecedent only occurred in 32.67% of all rows in the data. Out of that 32.67%, it was found that the consequent was true 42.1% of the time. These are relatively low numbers and do not indicate that this is a meaningful rule. Finally the lift for this rule is only 0.793 meaning that it is not very likely that the consequent is true for the antecedent while accounting for the popularity of the antecedent.

### **[H\_Resident Type] => [1\_Population Base]**

This rule has the highest support with 96.704% and confidence of 98.5%. The reason for these values being so high is down to mainly their high frequency within the rest of the dataset and this is confirmed by the lift value which is 1.001. This is not adequately larger than 1 and therefore it does not imply an association between not living as a communal resident and being a usual resident.

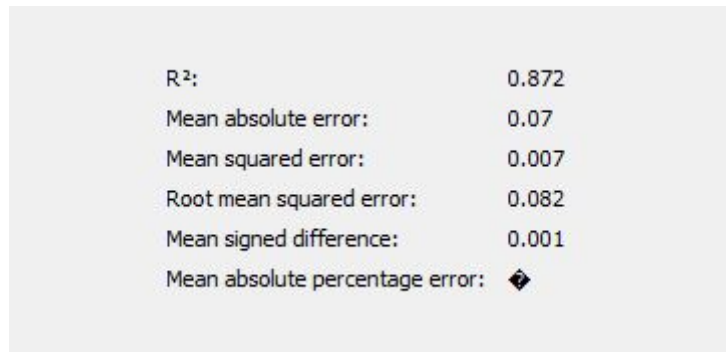
### **[1\_Student, 1\_Sex] => [6\_Economic Activity]**

The support for this rule is only 2.155%, but the confidence is much higher at 19.1% and finally the lift value is very high at 4.392. This suggests that while someone is unlikely to be a student and a male, when they meet those requirements they are extremely likely to be in the student class for economic activity. This makes sense because if you're a student then you fall into the '1' class for students and '6' class for economic activity.

## 5. Regression

### 5.1. Regression Tree Ensemble

The first algorithm I used to perform regression on the data was a Regression Tree Ensemble with 100 trees.



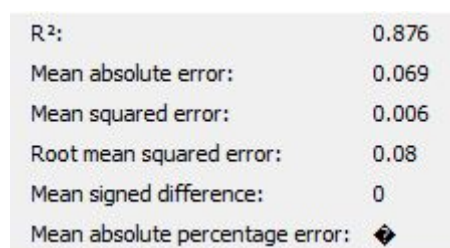
R <sup>2</sup> :	0.872
Mean absolute error:	0.07
Mean squared error:	0.007
Root mean squared error:	0.082
Mean signed difference:	0.001
Mean absolute percentage error:	◆

Fig. 9 Statistics of Regression Tree Ensemble

Fig. 9 displays the statistical results of predictions made on the 'No of Hours' attribute. The R squared value is really good with 87.2% meaning the regression line was able to account for that much variance in the data. The mean squared error is also extremely low meaning that where the variance in data was not accounted for by the regression line, the magnitude of that error is not significant.

### 5.2. Simple Linear Regression

R squared value is 87.6% which is a good indicator that the model has fit the data well. Furthermore, the mean squared error and root mean squared error are very low meaning that there are hardly any residuals in the predictions.



R <sup>2</sup> :	0.876
Mean absolute error:	0.069
Mean squared error:	0.006
Root mean squared error:	0.08
Mean signed difference:	0
Mean absolute percentage error:	◆

Fig. 10 Statistics of Simple Linear Regression

### 5.3. Comparison

Figures 11 and 12 show the line plot of the actual 'No of Hours' value against the value predicted by the model to show the variance. The Regression Tree Ensemble algorithm calculated the variance of each prediction (Displayed as the blue line in Figure 11) which as you can see is very low. The Simple Linear Regression did not provide this statistic to be plotted but comparisons can be made between their statistics from the numerical scorer node.

It is evident from Figures 9 and 10 that the Simple Linear Regression performed marginally better on this data. The R squared value is 0.4% higher showing that it was able to account for that much more variance. Furthermore, the root and mean squared error are both lower but only by a fraction of a percent. Because the performance and the statistics produced were so close I don't believe that one algorithm outperformed the other. In terms of computational complexity the simple linear regression was much more efficient because the ensemble method had to create 100 individual models. Therefore, if the performance is equal and the simple linear regression model is a more efficient algorithm then it has to be said that it is the better algorithm for the classification of this dataset.



Fig. 11 Regression Tree Ensemble Line Plot

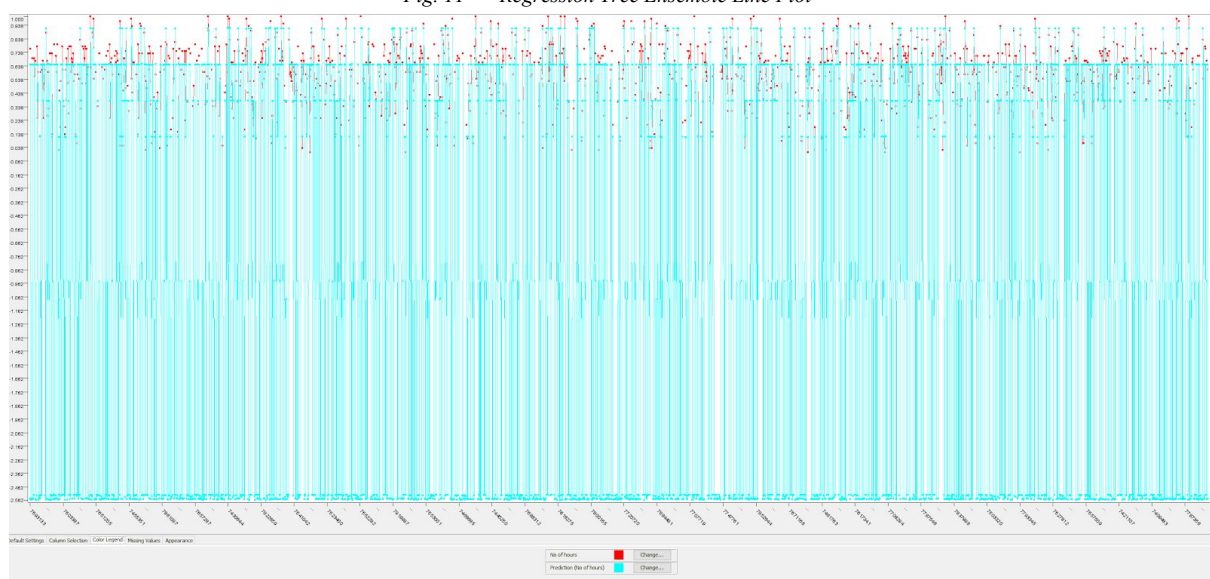


Fig. 12 Simple Linear Regression Line plo

## Part 2

For this section I chose to use the IMDb sentiment analysis dataset.

### 1. Preprocessing Text Data

The first step in preprocessing the data was to convert each row in the dataset into its own document. The second step is to remove the noise from each document, including numbers, stopwords, and words with less than 3 characters. Removing stop words is, however, not always beneficial to the task of sentiment analysis because what might be considered a stopword for one dataset might be a meaningful feature in another dataset.

After this each document was Part-of-Speech (POS) tagged, assigning every word a tag such as 'NN' if it is a noun or 'JJ' for adjectives. This was essential for the next step which was lemmatizing the data, the process of removing inflections of words and returning them to their root, or lemma. For example, 'ran' and 'running' both have the lemma 'run'. This method produces better results than stemming because lemmatization takes into account the POS tags. Lemmatization is also important for the next task word embedding, so that the contexts of all forms of a word are captured in one vector representation.

The word embedding algorithm used to turn the documents from words into vectors was the Doc2Vec algorithm. Doc2Vec is based on Word2Vec except that it trains a document vector that attempts to capture the semantics of a document and not just a word. The algorithm trains a Distributed Memory version of Paragraph Vector (PV-DM) model. PV-DM is similar to Continuous Bag of Words (CBOW) except that it adds a document feature to the word features already benign trained. The document feature is trained to remember the words that aren't in the current context, or as the topic of the paragraph. The other algorithm that can be used with Doc2Vec is Distributed Bag of Words (DBOW) but this was not chosen because the authors of the Doc2Vec paper do not recommend it (the paper can be found at <https://arxiv.org/pdf/1301.3781.pdf>).

S Text	[...] converted_document
slow-moving aimless movie distressed drift slow-moving aimless mo...	[-0.025469709187746048,0.037708647549152374,-0.115349300
attempt artiness black white clever camera angle movie disappoint...	[-0.015783997252583504,0.029760299250483513,-0.091109357
scene movie gerardo try song run head scene movie gerardo try s...	[-0.01377843413501978,0.020042523741722107,-0.0579031966
waste hour waste hour	[0.007035939954221249,0.03611591085791588,-0.07097425311
bit predictable bit predictable	[-0.0089357765391469,0.04072264954447746,-0.096244387328
love casting jimmy buffet science teacher love casting jimmy buffe...	[-1.680997374933213E-4,-5.242398474365473E-4,-0.001759122
movie lot florida look appeal movie lot florida look appeal	[-0.01563391089439392,0.03909681364893913,-0.10569445788
movie deliver movie deliver	[-0.025469709187746048,0.037708647549152374,-0.115349300
average acting main person low budget average acting main pers...	[0.002473607659339905,0.042981065809726715,-0.0837658196
review overdue consider tale sister single film review overdue con...	[-0.02349679544568062,0.02479437366127968,-0.08773373067
gem movie term screenplay cinematography act post-production e...	[-0.011739942245185375,0.024502700194716454,-0.070790708
structure film easily tightly construct history cinema structure film ...	[-0.02349679544568062,0.02479437366127968,-0.08773373067

Fig. 13 each document and its vector representation

Finally, the classes are re-appended to each document using the category-to-class node and then the file is written to memory for later use. Figure 13 shows the effect of removing noise and lemmatizing on the data as it no longer makes sense to a human reading

it and also shows you the corresponding word2vec vectors that were created and written to memory.

## 2. Distances Between Word Embeddings

The Doc2Vec algorithm is training a document vector, as well as individual word vectors, and trying to give it a similar value to other document and word vectors that it predicts have similar semantics. This offers several benefits in a semantic sense as theoretically words that have more similar semantics will have more similar vectors and will therefore be more useful to the classification algorithm that will be used later. Therefore, I will be exploring the euclidean distance between certain word vectors to see how effectively the doc2vec algorithm did its job. They will be analysed using both scatter plots and creating distance matrices.

S ▲ Object1	S Object2	D ▲ Dist...
strong	shot	1
black	bring	1
flaw	go	1
hour	obviously	1
glad	line	1
probably	episode	1
note	single	1
note	intelligent	1
themselves	fail	1
cover	serious	1.001

Fig. 14 The 10 most similar word embeddings

Figure 14 shows the top 10 most similar words based on the euclidean distance between their embeddings. You can see some intuitive associations have been made by the word2vec algorithm such as, 'strong' and 'shot' intuitively sound like they would appear together often and have a vector that is very close to each other to represent that. However for the other combinations in Figure 14, it is more confusing as to why they would be given such similar vectors. Fig 15 shows a descending list of euclidean distances between 'appreciate' and every word that appears in the context of 'appreciate'. The word whose vector representation is second furthest away from 'appreciate' is 'special'. You wouldn't expect these words to be placed very far apart however, this shows that the context/platform in which you gather your data affects how certain words are used, in this case our data is from IMDb reviews. This displays one of the main strengths of using a neural network to train word embeddings; it will pick up on patterns that even a human might struggle to see, such as 'appreciate' and 'special' being very dissimilar in this case.



S	Object1	S	Object2	D	▼ Dist...
	appreciate		suck	5.565	
	appreciate		special	5.068	
	appreciate		scamp	4.986	
	appreciate		music	4.448	
	appreciate		direction	4.258	
	appreciate		dialogue	4.222	
	appreciate		avoid	4.177	
	appreciate		totally	4.086	
	appreciate		produce	4.047	
	appreciate		effect	3.978	
	appreciate		half	3.876	
	appreciate		crap	3.791	

Fig 15 Euclidean distance between 'brilliant' and other words.

The final observation that was made was using the scatter plot. This was useful for visualising the proximity of more than just a pair of words and we could explore how groups of words are bunched together in a way. I reduced the dimensionality of the word vectors down to 2 for the purpose of plotting them on a graph, Figure 16 shows the results. The words 'flaw' and 'terrible' appear much closer together than 'actor' which is in closer proximity to 'imagination'. This is as you would expect, positive words placed nearer the positive ones and the same with negative words.

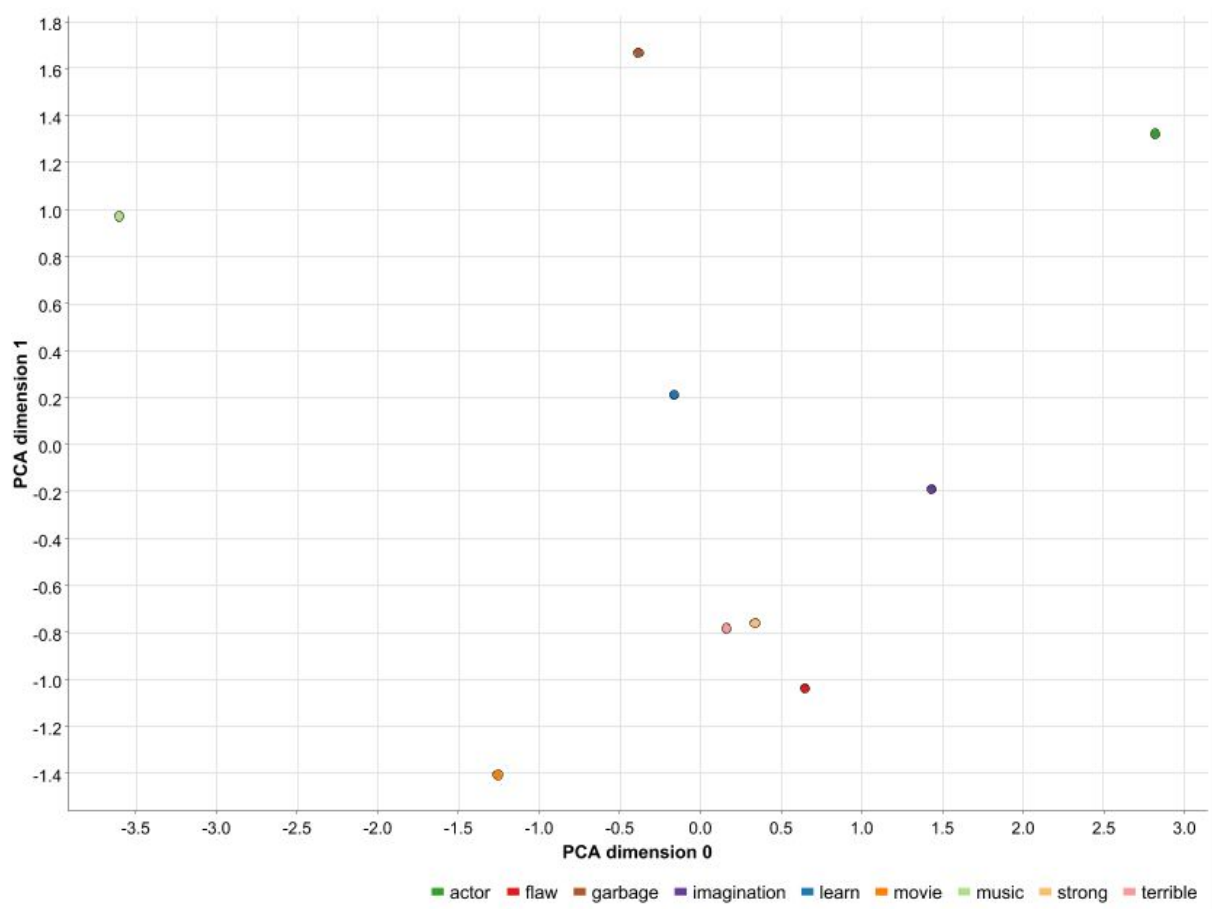


Fig. 16 Scatter plots showing several word vectors

However, there are a few anomalies exhibited within the dataset. The word ‘garbage’ is very far from any of the other words that you would expect to see in a negative review such as ‘flaw’. When looking for reasons why this might be I found that ‘garbage’ appears within the context of almost every word in the dictionary which might be confusing the algorithm because there is no specific subset of context that ‘special’ is a part of. This means the semantics that can be captured on this word might not be very specific due to such a broad contextual basis. It might also be the case that it is just more contextual related to words that are not being analysed in this graph plot.

### 3. Clustering

The clustering algorithm chosen to cluster the Doc2Vec vectors was Hierarchical Clustering with distance. Before running the hierarchical clustering algorithm I assigned each class, positive and negative, a different colour. This will allow me to inspect whether or not the hierarchical clusterer found a mapping between the clusters that are produced and the positive or negative classes.

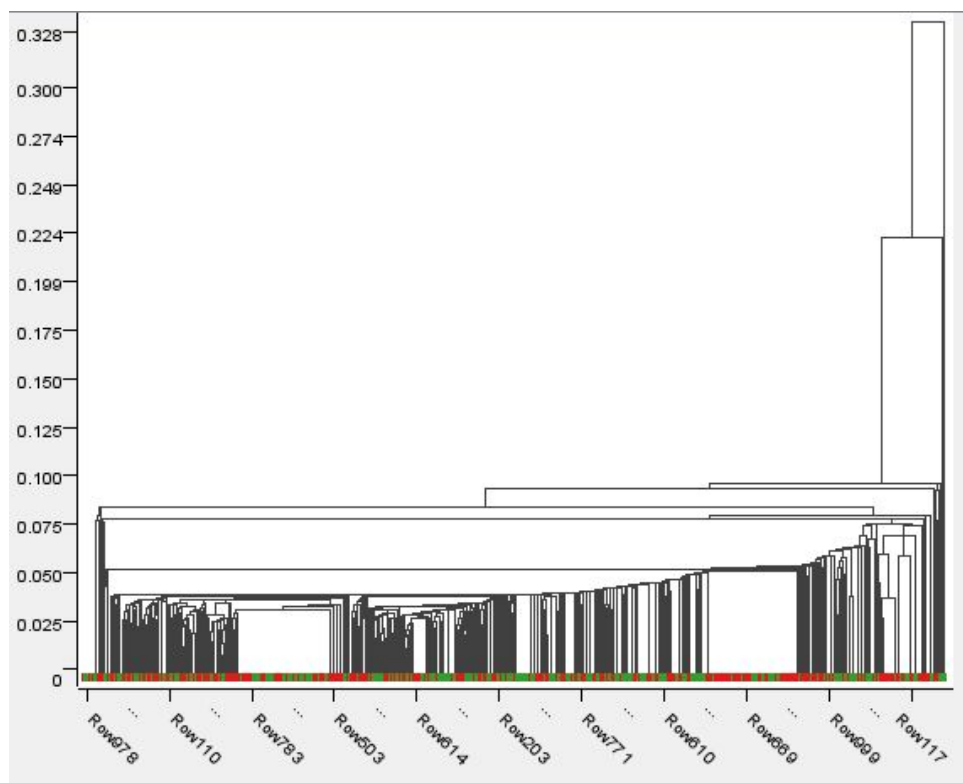


Fig 17. Hierarchical single linkage clustering dendrogram

The dendrogram that you get from the hierarchical cluster view node allows us to get a good insight into the clustering that has taken place, this can be seen in figure 17. The colours at the bottom of the dendrogram represent that class that the data point belongs to; green represents positive and red represents negative. Unfortunately, there are no large

concentrations of either red or green in the dendrogram. This indicates that the hierarchical clustering was unable to find a mapping between the positive and negative sentiment classes in the data and the clusters that were produced. However, the dendrogram displays a good shape which shows that many of the instances are assigned into clusters early and not added at the end where they would not have been in previous clusters first. Overall this was not a very good solution and did not produce any useful or significant clusters of data.

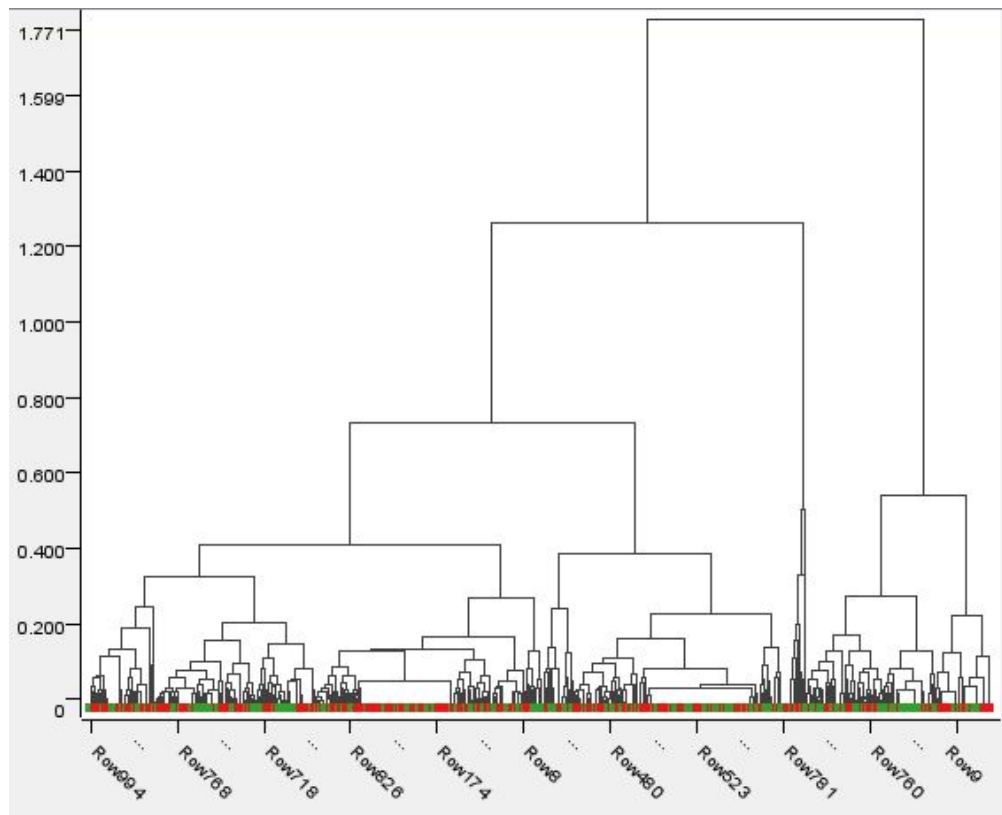


Fig 18. Hierarchical complete linkage clustering dendrogram

The first clustering attempt used only single linkage between pairs of observations, combining two clusters that contain the closest pair of elements not yet belonging to the same cluster as each other. A disadvantage of this is that it tends to create longer clusters where elements at opposite ends of the cluster may actually be closer to other clusters. Therefore, I ran the clustering again but using the complete-linkage, or furthest neighbour clustering, method of agglomerative hierarchical clustering. This places every data point in its own cluster and then combines clusters until they are all in one cluster. This differs from single linkage in that it combines clusters based on the distance between the furthest away members of each cluster. Figure 16 shows the results and they are more encouraging than the previous. There are longer bands of red and green, although it is still far from displaying a mapping between the classes and clusters, it is an improvement upon single linkage. Furthermore you can see a more even distribution of cluster merging happening and not all the merging is happening at the same distance.



## 4. Classification

The algorithms that have been used to classify this data are Long Short Term Memory Recurrent Neural Network (LSTM), SVM and Naive Bayes as these algorithms are all known to work well at classifying text data.

### 4.1. LSTM

LSTM is a type of recurrent neural network that can remember long term dependencies in sequential data really well, such as text data. This makes it particularly useful in sentiment analysis tasks.

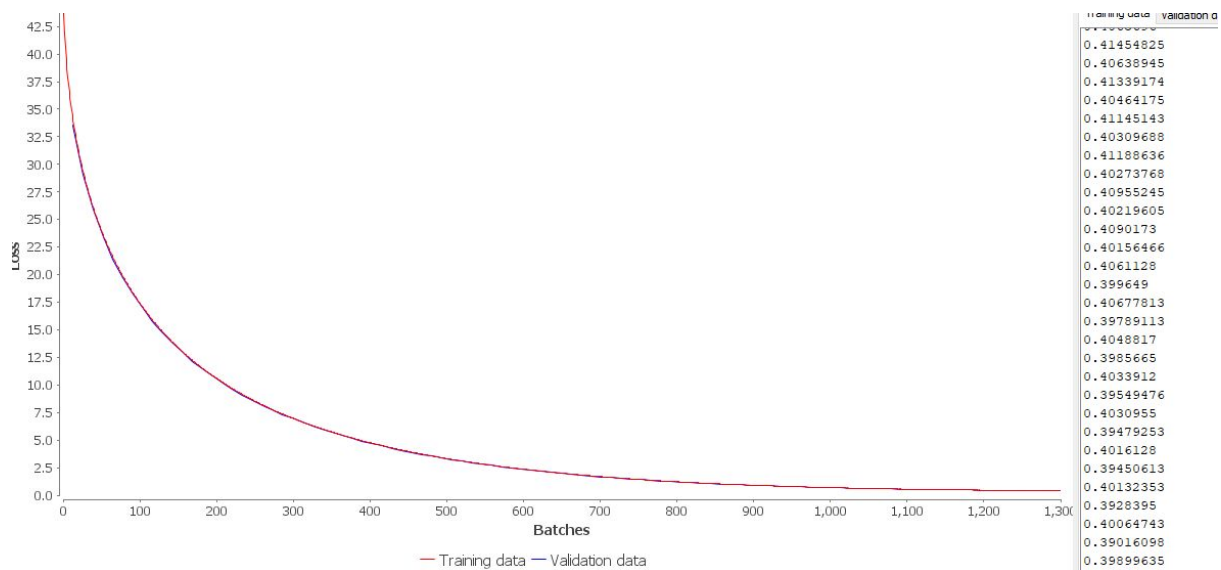


Fig. 19 LSTM Training and Validation Loss Curve

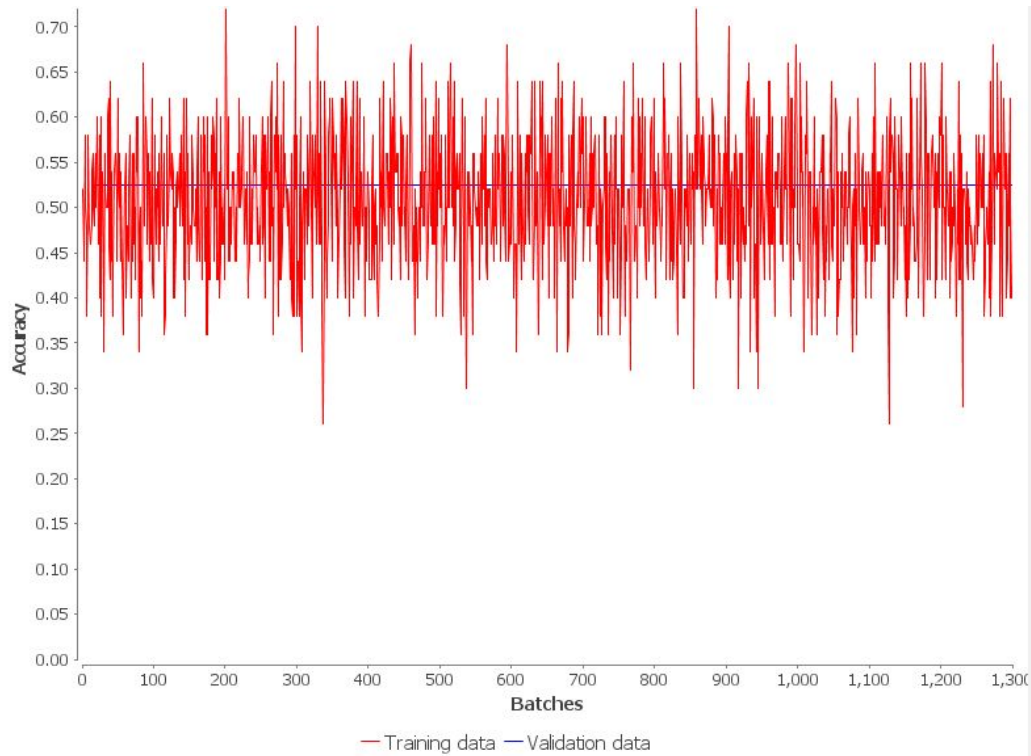


Fig 20. Training and Validation Accuracy

After 100 epochs the validation loss finished at 0.3899 and was still reducing so training could have gone on for longer. However, the accuracy of the validation results, Figure 20, was not improving, in fact it stayed at 52.5% throughout training. Figure 21 displaying the softmax probability output of the network reveals why the validation accuracy stayed at 52.5%. The network had learned to classify every input vector as the same class with exactly the same probabilities.

[...] doc_vector	[...] Aggreg...	[D] softmax_1_0:0_0	[D] softmax_1_0:0_1
[-0.056764692068099976,0.08226531744003296,-0.04899386316537857,...]	[1,0]	0.488	0.512
[0.14732114970684052,0.16304917633533478,-0.06701628863811493,...]	[0,1]	0.488	0.512
[-0.14219017326831818,-0.09412863105535507,0.5213900208473206,...]	[0,1]	0.488	0.512
[-0.02433893084526062,0.1599404662847519,0.4644819498062134,...]	[1,0]	0.488	0.512
[0.23253540694713593,0.08624503761529922,-0.19021791219711304,...]	[1,0]	0.488	0.512
[0.23253540694713593,0.08624503761529922,-0.19021791219711304,...]	[0,1]	0.488	0.512
[0.06382083892822266,0.0982852578163147,-0.14007091522216797,...]	[0,1]	0.488	0.512
[-0.07997270673513412,0.1776244044303894,0.22742731869220734,...]	[1,0]	0.488	0.512
[0.05071253329515457,0.11956261843442917,0.5967199206352234,...]	[0,1]	0.488	0.512
[0.12689979374408722,-0.0017983168363571167,-0.05991550907492637...]	[0,1]	0.488	0.512
[0.06949900835752487,-0.02039284259080887,0.06762731820344925,...]	[0,1]	0.488	0.512
[-0.15300485491752625,0.1825156807899475,0.1341111958026886,...]	[1,0]	0.488	0.512
[-0.12562505900859833,0.020455770194530487,0.6090948581695557,...]	[0,1]	0.488	0.512
[-0.05596841126680374,0.08741649240255356,0.043529342859983444,...]	[1,0]	0.488	0.512
[-0.0020913248881697655,0.1435491293668747,-0.0293094664812088,...]	[0,1]	0.488	0.512

Fig. 21 Softmax probability output

## 4.2. SVM

The SVM achieved an accuracy score of 63.298% which means that it outperformed the LSTM. However, it still did not achieve very good results and can not be said to have done much better than if you were to just randomly choose positive or negative.

Document ...	0	1
0	63	36
1	33	56

Correct classified: 119	Wrong classified: 69
Accuracy: 63.298 %	Error: 36.702 %
Cohen's kappa ( $\kappa$ ) 0.265	

*Fig. 22 SVM Results in Confusion Matrix*

This is a binary classification problem as well so it is even more surprising that the vanilla SVM algorithm could only achieve 63% accuracy. Reasons why could be that the dimensionality of the Doc2Vec embeddings was too high, at 100, and the SVM algorithm got overwhelmed, in a sense, by all the data for each document. Although, I believe that given the poor results of both the LSTM and SVM that the main reason is down the Doc2Vec embeddings themselves.

## 4.3. Naive Bayes

Naive Bayes managed to classify the IMDb dataset with the highest accuracy of 75.532%. This can be considered as a good result and definitely better than if you were to randomly pick a class for each input vector.

Document ...	0	1
0	74	22
1	24	68

Correct classified: 142	Wrong classified: 46
Accuracy: 75.532 %	Error: 24.468 %
Cohen's kappa ( $\kappa$ ) 0.51	

Fig. 22 Naive Bayes Results in Confusion Matrix

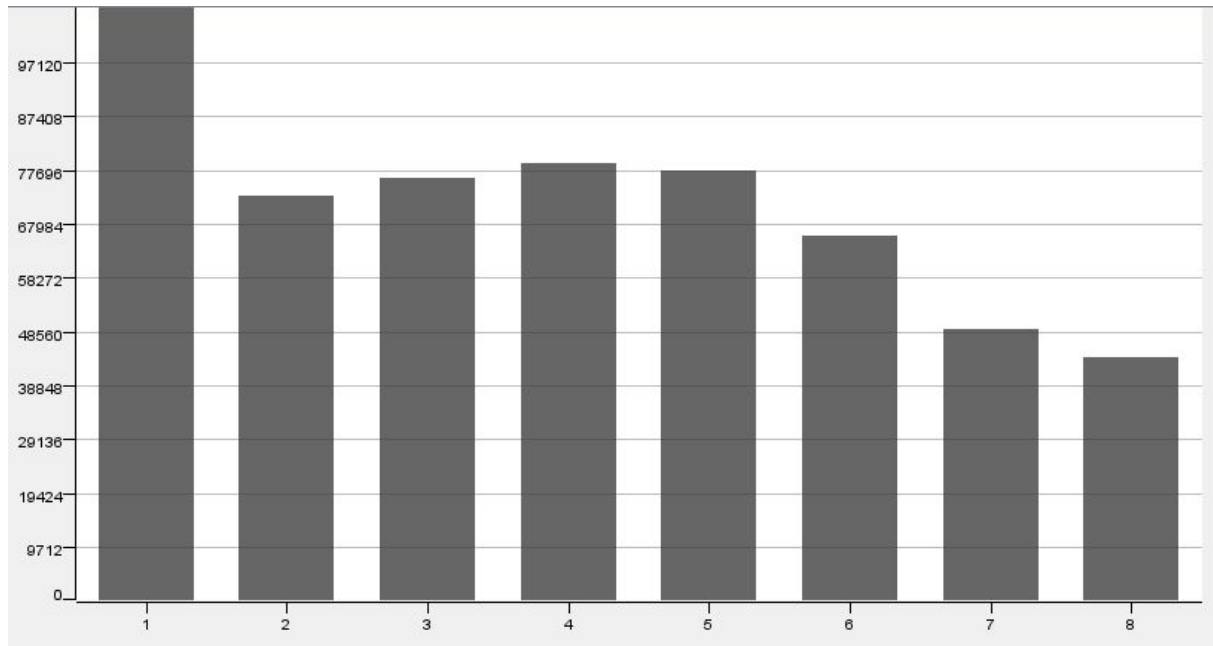
#### 4.4. Comparison

The LSTM is a very complicated classification algorithm that requires much more data to train it adequately and also needs a lot of revision to the hyper parameters as there is no analytical method to finding the optimal values, it's more or less trial and error. Given more time to produce a network that would be more specialised to this data and I believe it would be the top performing algorithm. SVM achieved poor results while still being better than the LSTM and better yet was the naive bayes classifier which actually achieved a high accuracy score for text data of 75%. This is a good value for classifying text data as it is a very ambiguous data type and even humans only catch around 80% of the intricacies of their language.

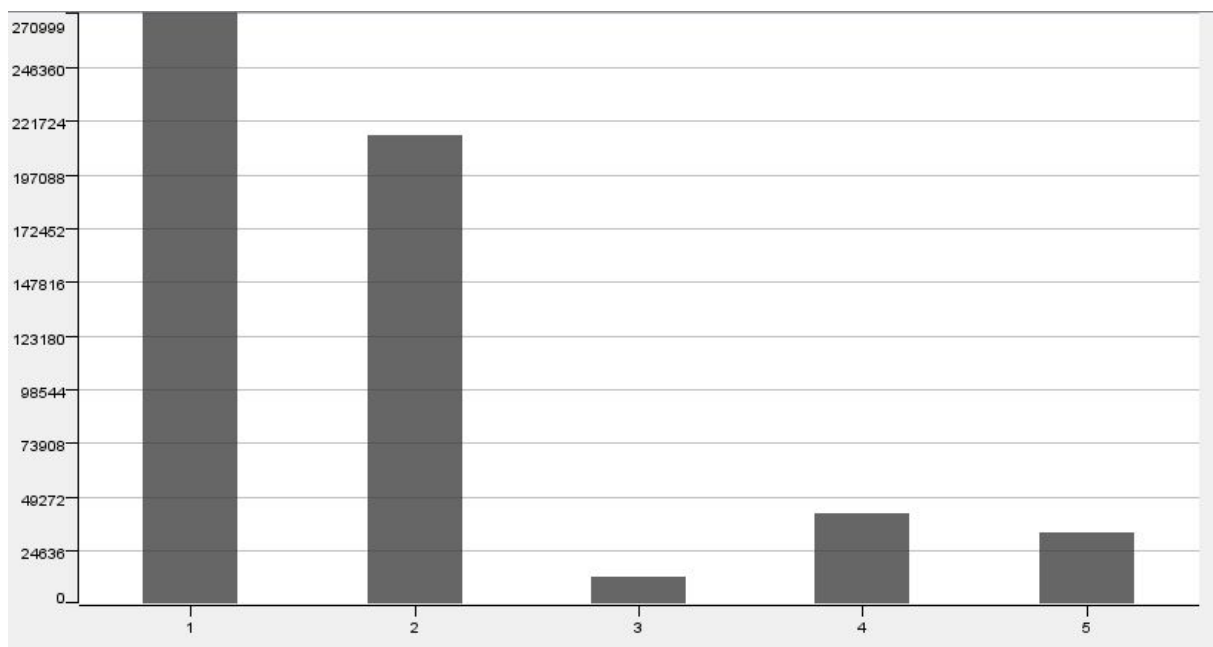
The main reason why I think the classification did not go very well is because the word embeddings used for not optimal. Again, I used a more sophisticated embedding technique, in Doc2Vec, which itself trains an algorithm to predict a good vector for each document. I think there was insufficient data in the dataset to adequately train this algorithm and also improvements could be made to the preprocessing stage. The algorithm that is being used is only as good as the data that you are giving to it. Therefore, if we had more time on this project, I would delve much deeper into the IMDB dataset to come up with a specific list of stopwords, and whether or not numbers are important to this dataset, or emojis. There are so many possibilities of improvements that could be made.

# Appendices

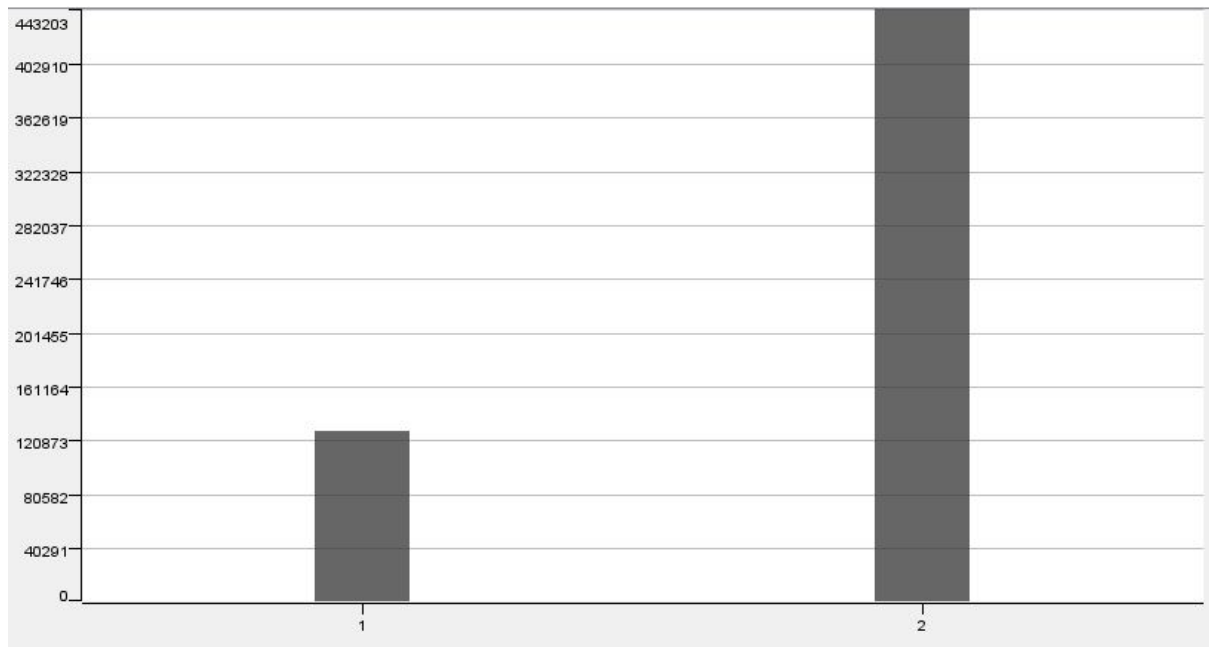
## A. Age Distribution



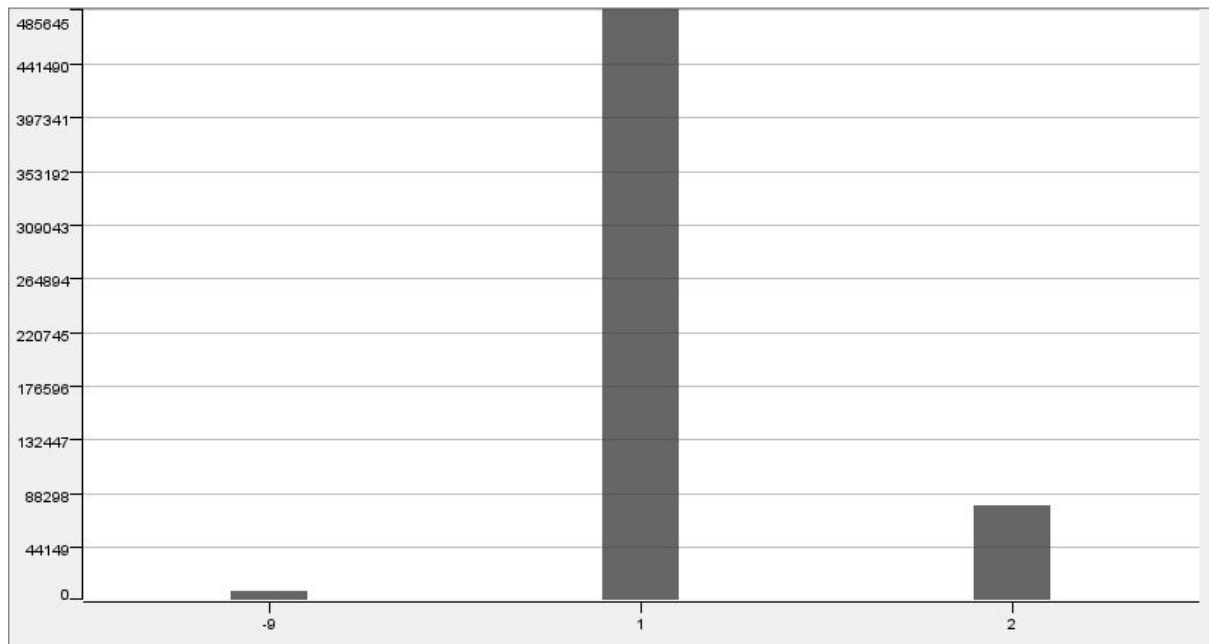
## B. Marital status Distribution



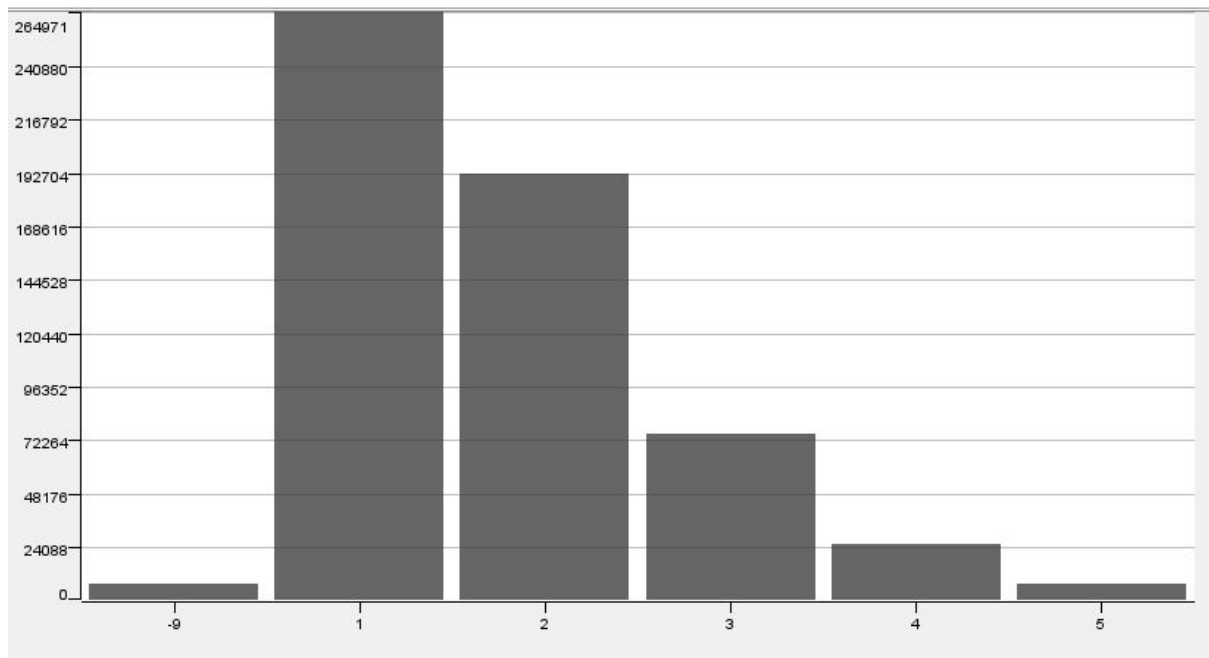
### C. Student Distribution



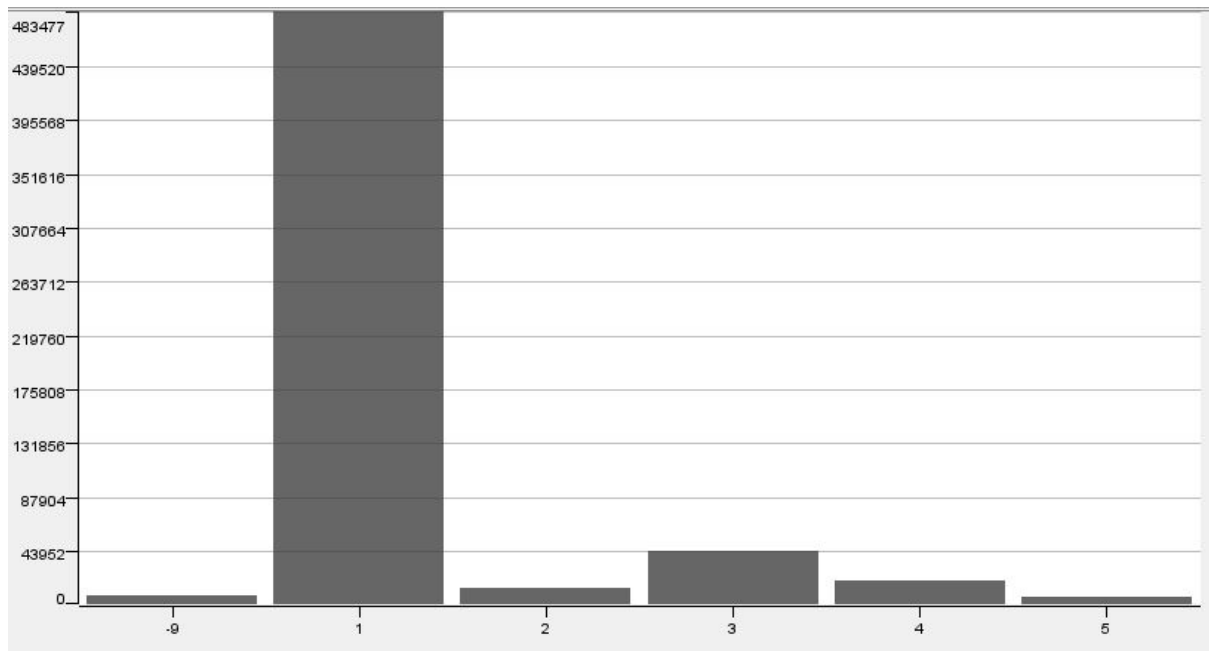
### D. Country of Birth Distribution



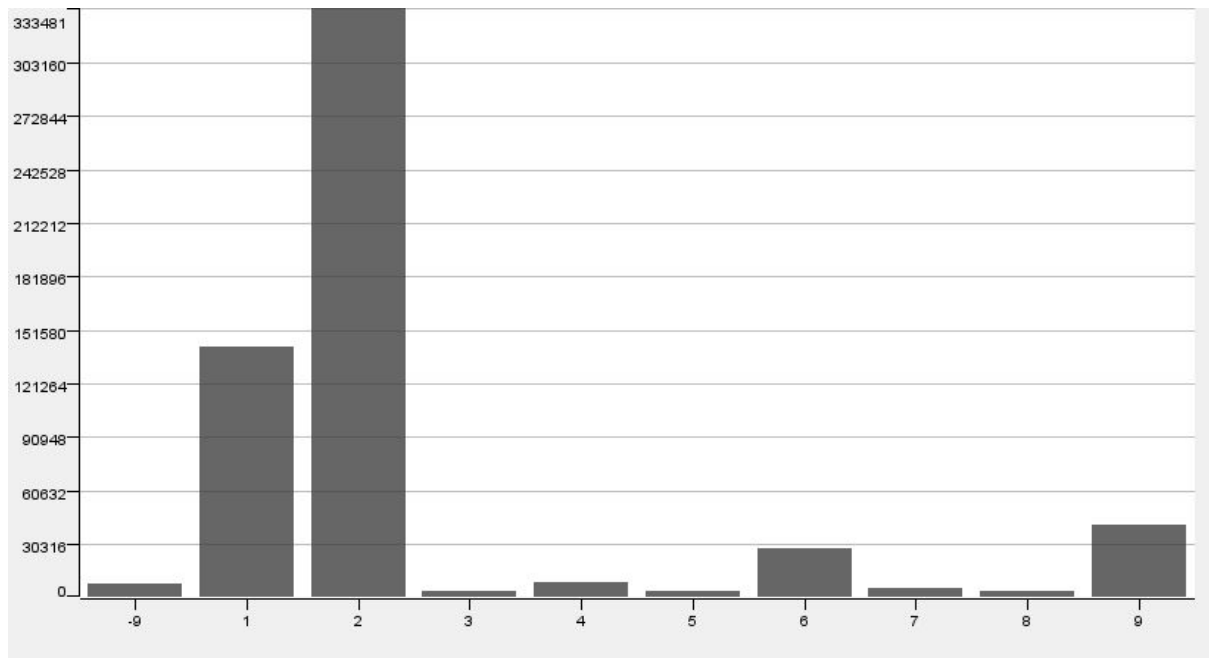
## E. Health Distribution



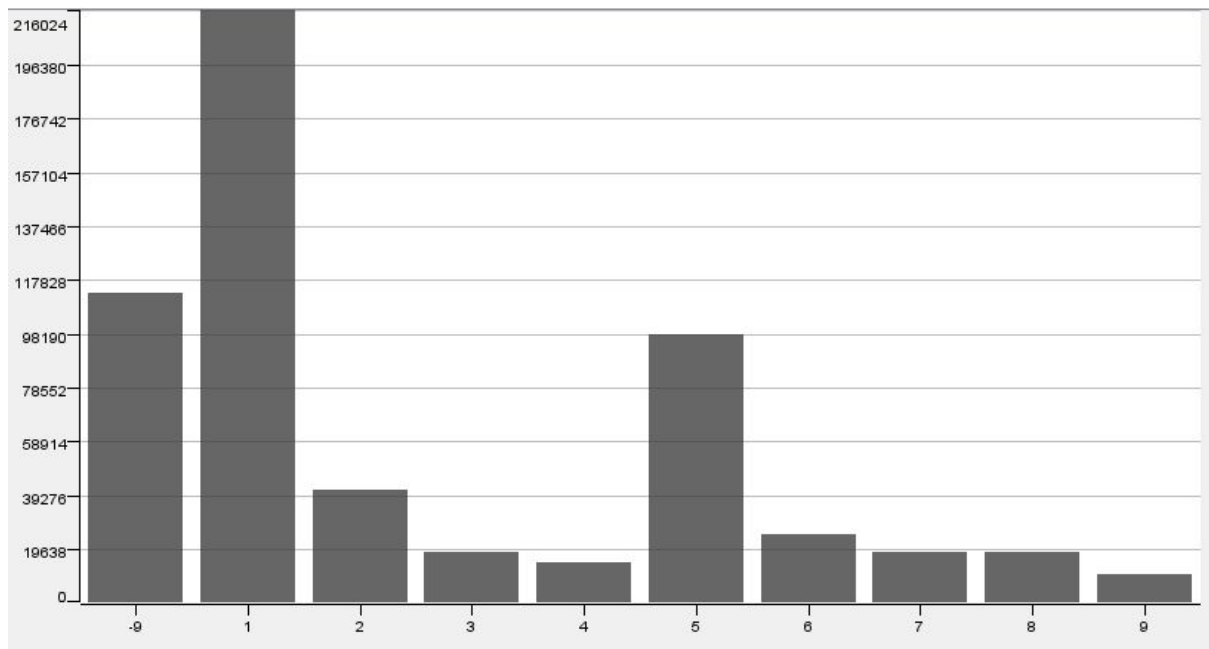
## F. Ethnic Group Distribution



## G. Religion Distribution

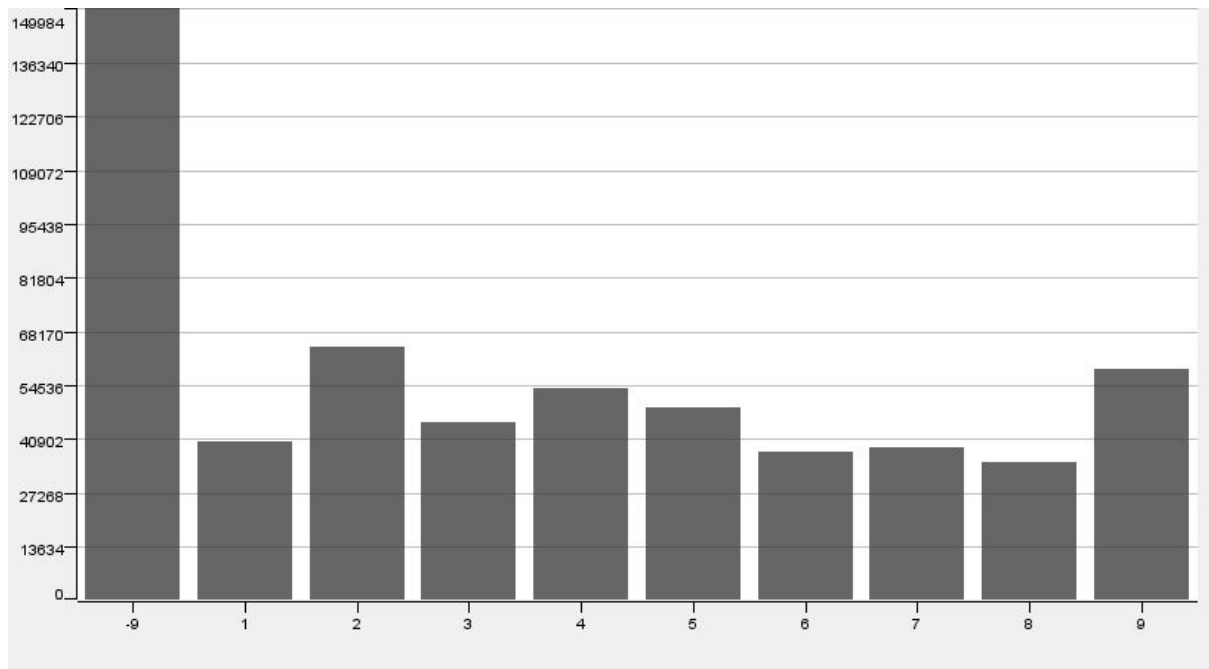


## H. Economic Activity Distribution

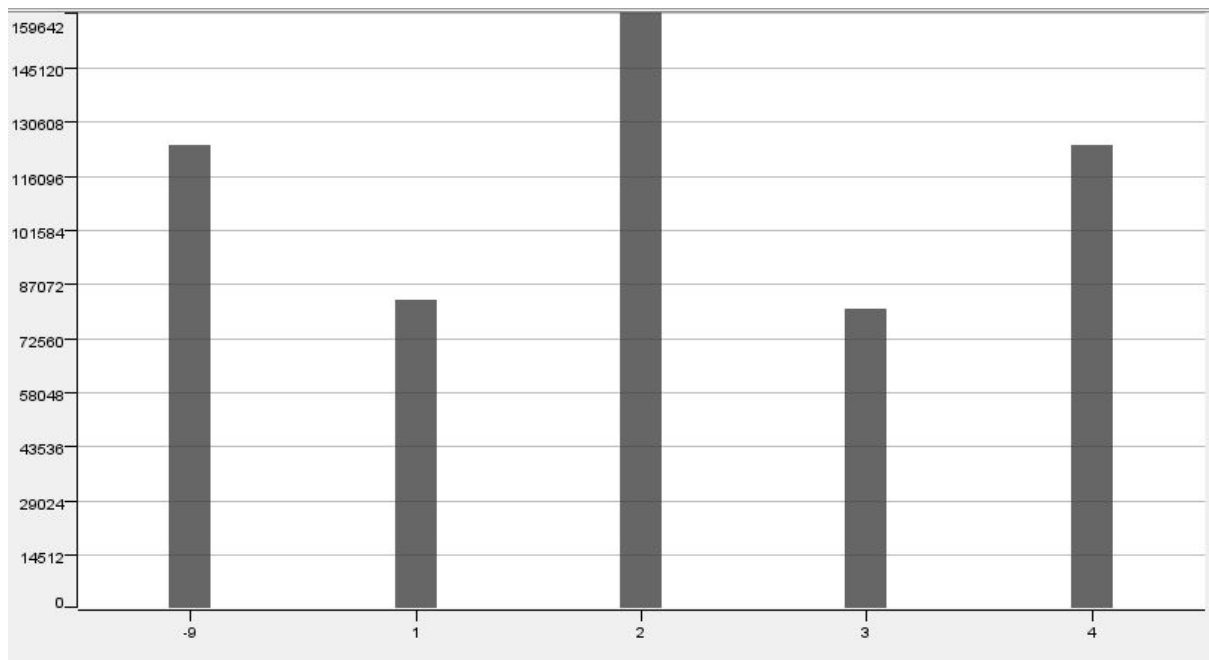




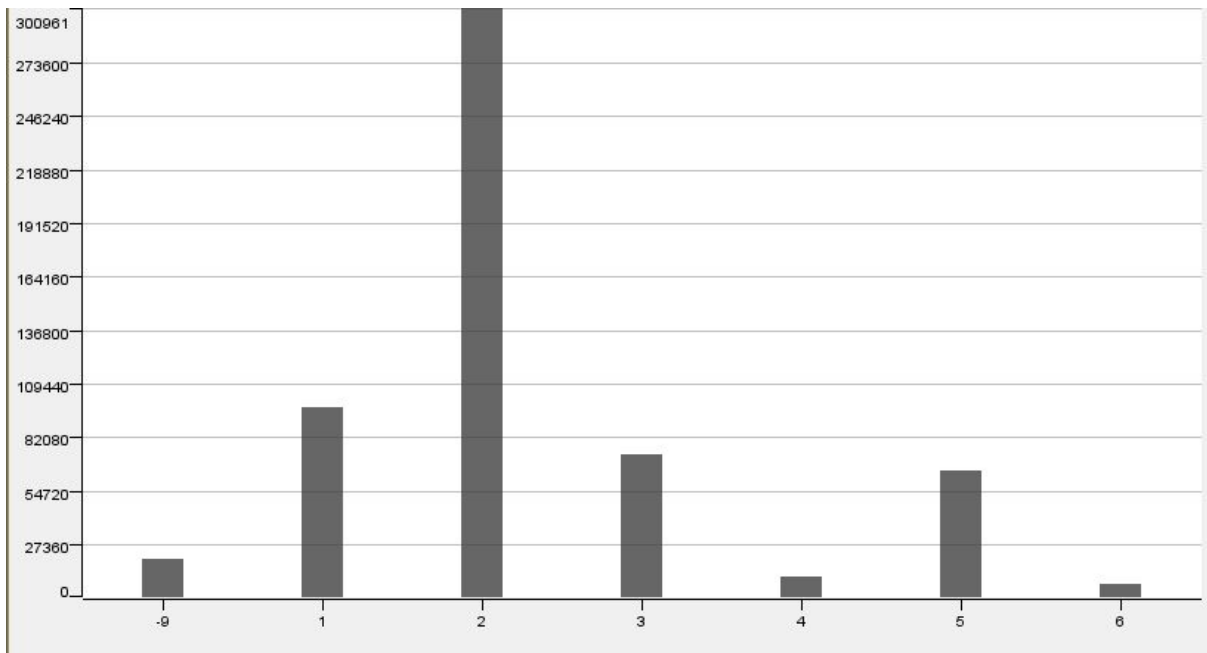
## I. Occupation Distribution



## J. Approximated Social Grade Distribution



## K. Family Composition Distribution



## L. Population Base Distribution

