NLP Assignment 2

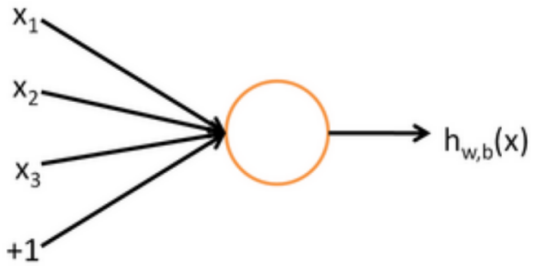
Name: Helena Julie Arpudaraj

PID: 4735186

1. Explain how ANN works and how multiple layers can be used for improving the accuracy.

Answer:

In an Artificial Neural network, each neuron is a binary logistic regression. In each neuron the weighted sum is taken and an activation function such as Sigmoid is applied on the weighted sum.

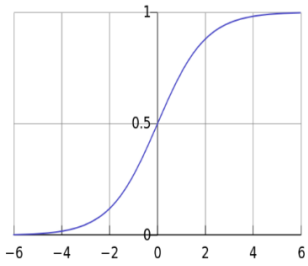


$H_{w,b}$ is the weighted sum it can be $\sum w_i x_i + b$, here b is the bias term, x are the inputs and w is the weight on the input. It can be represented as:

$$h_{w,b}(x) = f(w^{'}x + b)$$

Now an activation function such as sigmoid function is applied on the above function. The sigmoid function is represented as follows:

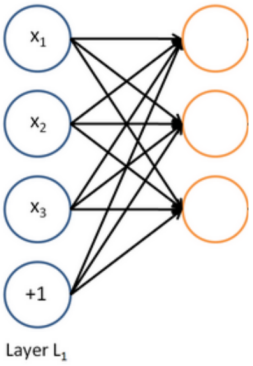
$$f(z) = \frac{1}{1+e^{-z}}$$



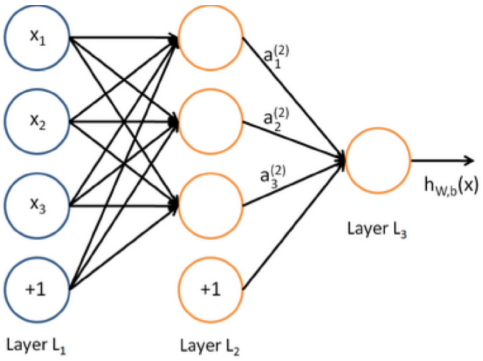
Here z = w' x + b

The activation function is used to determine the output of a neuron. It determines the value from 0 to 1.

Hence, a neural network can be treated as several logistic regressions running same time.



In each layer we get several outputs. This is fed to another logistic regression.



In the output layer, the output predicted is compared to the actual output. This is called loss function. The loss function is used to adjust the weights in the layers during back propagation. The goal is to achieve the least calculated loss.

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$

The most common loss function can be the Mean squared error

This can give us the below loss function. Our goal is to achieve the least error. The error is reduced using Stochastic Gradient Descent and the error is back propagted to readjust the weights to reduce the error.

Steps followed:

1. initaialize random weights close to 0

2. input first training data from left to right.

3. Each neuron is a binary logistic regression and activation function is applied on the weighted sum.

4. Continue above steps till the last output layer.

5. The predicted output in output layer is compared to actual output using loss function.

6. The error is back propagated right to left and weights are adjusted to minimize the error or loss function.

7. Repeat the steps for each training data in case of stochastic gradient descent.

Multiple layers:

Having multiple hidden layers will solve complex problems such as image recognition. Where each hidden layer will predict a feature like an edge in digits. This allows us to solve complex problems.

Each hiddenlayer there are several logistic regressions running same time. Each neuron is a logistic regression function.

2. Explain ANN using matrix and sum-based representation.

Answer:

In an Artificial Neural network, each neuron is a binary logistic regression. In each neuron the weighted sum is taken and an activation function such as Sigmoid is applied on the weighted sum.

$H_{w,b}$ is the weighted sum it can be $\sum w_i x_i + b$, here b is the bias term, x are the inputs and w is the weight on the input. It can be represented as:

$$h_{w,b}(x) = f(w^{'}x + b)$$

Now an activation function such as sigmoid function is applied on the above function. The sigmoid function is represented as follows:

$$f(z) = \frac{1}{1 + e^{-z}}$$

Here $z = w^{'} x + b$

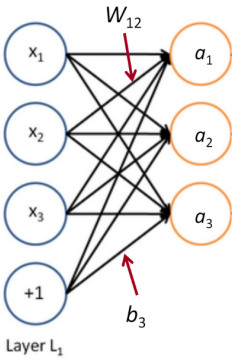
The activation function is used to determine the output of a neuron. It determines the value from 0 to 1.

Hence, a neural network can be treated as several logistic regressions running same time.

In **sum-based representation** the output in each layer for each neuron is represented as:

$a1=f(W_{11}x_1 + W_{12}x_2 + W_{13}x_3 + b)$

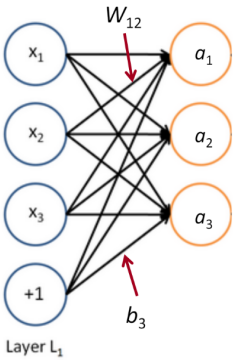
$a2=f(W_{21}x_1 + W_{22}x_2 + W_{23}x_3 + b)$ and so on



here a_1,a_2,... are the outputs from each neuron in each layer. [

In **matrix representation** z can be represented as $z=Wx + b$ and $a=f(z)$

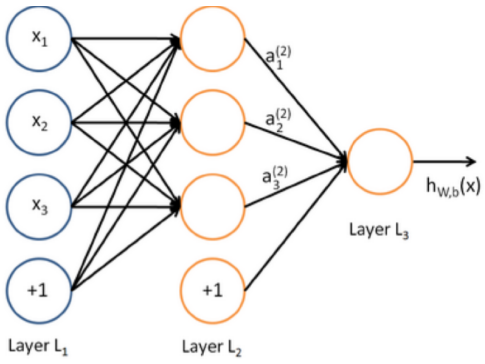
$f([z_1,z_2,z_3]) = [f(z_1),f(z_2),f(z_3)]$



here $a=f(z)$is the output from each neuron in each layer. $[f(z_1),f(z_2),f(z_3)]$ is the output from each layer represented in matrix form.

So each row represents a layer's output and each element in a row represents the output from a neuron in that layer.

In each layer we get several outputs. This is fed to another logistic regression.



In the output layer, the output predicted is compared to the actual output. This is called loss function. The loss function is used to adjust the weights in the layers during back propagation. The goal is to achieve the least calculated loss.

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$

The most common loss function can be the Mean squared error

This can give us the below loss function. Our goal is to achieve the least error. The error is reduced using Stochastic Gradient Descent and the error is back propagted to readjust the weights to reduce the error.

3. Explain what the problem is in named entities and how window classication can be used to solve the problem.

Answer:

Named Entity Recognition uses information extraction technique to identify names in the text. The name can be identified as a name of a person, location, organization, etc. the three categories of NER are entity names, temporal and numerical expressions. The names are categorized by using the context of the words around the center word.

Problems in NER are:

1. It is hard to workout boundaries of entity. Example Bank of America is a three word entity. Its difficult to set boundary of the entity words

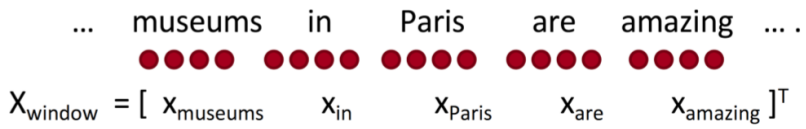
2. it is hard to know if something is an entity or not. Example Future college can mean its "Future college" is name of a school or a phrase.

3. It is also difficult to know the class of new entity and unknown entity.

3. Entity class is ambiguous. In a "named entity recognition" a word such as "Paris" can mean name of person or place. Like Paris in France or Paris Hilton. So it becomes a 4 way

classication of Person, Place, Organization or None. This kind of problems can be solved using the context in which it is used.

To solve the problem a window classication is used. It classies each word based on its context window. Window classication is classifying word in its context window. Meaning based on neighboring words.



Example: Window size 2 will have 2 context words left of the word and 2 context words at the right of the word. Here we classify Paris word.

Resulting vector $x_{window} = x \in R^{5d}$, a column vector.

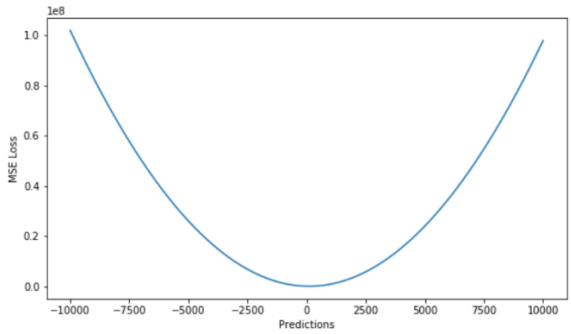
4. Explain how back propagation works and what it is used for.

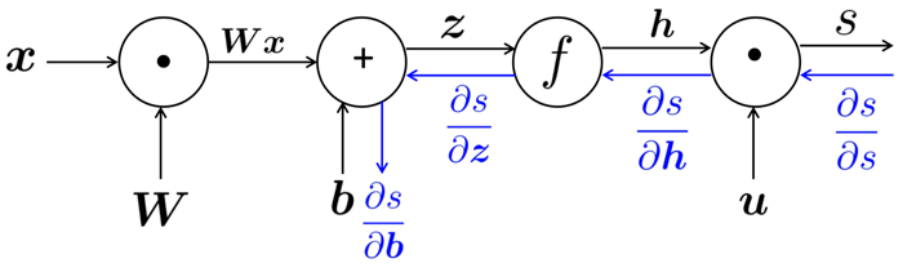
After forward propagation, in the output layer, the output predicted is compared to the actual output. This is called loss function or error. The loss function is used to adjust the weights in the layers during back propagation. The goal is to achieve the least calculated loss.

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$

The most common loss function can be the Mean squared error

Our goal is to achieve the least error. The error is reduced using Stochastic Gradient Descent and the error is back propagted to readjust the weights to reduce the error. We have to calculate the descent by having a negatiove slope(downstream gradient). Goal is the achieve the lease error i.e., to achive the lowest point in below gradient graph. For this purpose the gradient decent is passed in the back propagation as shown below:





In forward propagation we have below equations:

$$s = \boldsymbol{u}^T \boldsymbol{h}$$
$$\boldsymbol{h} = f(\boldsymbol{z})$$
$$\boldsymbol{z} = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}$$
$$\boldsymbol{x} \quad \text{(input)}$$

Equations for passing the gradients:

Downstream Gradient descent: ∂ s/∂ z = ∂ s/∂ h $*$ ∂ h/∂ z

The weights are updated as follows,

$$\mathrm{W} = \mathrm{W} - \alpha \ \partial \mathrm{s}/\partial \mathrm{W} \qquad \mathrm{b} = \mathrm{b} - \alpha \ \partial \mathrm{s}/\partial \mathrm{b}$$

Steps followed:

1. initaialize random weights close to 0

2. input first training data from left to right.

3. Each neuron is a binary logistic regression and activation function is applied on the weighted sum.

4. Continue above steps till the last output layer.

5. The predicted output in output layer is compared to actual output using loss function.

6. The error is back propagated right to left and weights are adjusted to minimize the error or loss function.

7. Repeat the steps for each training data in case of stochastic gradient descent.

Backpropagation is used to improve the accuracy of the prediction in machine learning and readjust the weights to reduce error. The backpropagation algorithm is also used in artificial intelligence, Natural language processing, machine learning and image processing.

5. Explain why we need structures to analyze language dependency.

Answer:

Language dependency is analyzed using structure of the words and phrases. Structure is very important to interpret and understand the language correctly.

For example, a sentence "Scientist count whales from space" has several meanings. It can mean scientist are in space or the whales are in space. There is no structure in this sentence which can result in ambiguity. Therefore, structure is very import to understand and interpret the language.

Each word in a sentence can have different parts of speech like nouns, pronouns, adjectives, preposition, conjunction, etc. These makes the structure and grammar for a language.

We need structures:

To analyze which words are connected to which words

To interpret the language correctly.

To understand complex sentences and ideas in which the complex words are formed to convey complex meanings

6. Explain how n-gram technique is used for: (a) language modelling, (b) feature representation of text

Answer:

a) language modelling:

Language modelling is assigning probabilities to sequence of words. N-gram is one of the language models. It assigns probabilities to sequence of n words. A 2- gram is sequence of 2 words. 3 gram is a sequence of 3 words and so on. We can estimate the probability of last n-gram word using the other words in the n-gram. We can also estimate probability of n-gram sequence.

P(w| h) – probability of word w based on previous words/history h

For a phrase such as "the weather is cloudy, and it will rain" . We want to predict the probability of last word "rain" based on the history "the weather is cloudy, and it will" . So the probability of P(w| h) would be the count of the sequence "the weather is cloudy and it will rain" appears in whole corpus divided by the count of "the weather is cloudy and it will" appearing in whole corpus.

P(rain| the weather is cloudy and it will) = C(the weather is cloudy and it will rain)/ C(the weather is cloudy and it will)

The probability of a sequence of words P(w1,w2,...,wn) in corpus is expressed as

$$P(X_1 ... X_n) = P(X_1)P(X_2|X_1)P(X_3|X_1^2) \dots P(X_n|X_1^{n-1})$$
$$= \prod_{k=1}^{n} P(X_k|X_1^{k-1})$$

Applying the chain rule to words, we get

$$P(w_1^n) = P(w_1)P(w_2|w_1)P(w_3|w_1^2) \dots P(w_n|w_1^{n-1})$$
$$= \prod_{k=1}^{n} P(w_k|w_1^{k-1})$$

b) feature representation of text

Feature representation of a text can be done using number of techniques such as bag-of-words, n-grams, parts-of-speech, etc.

Bag of words is a feature representation where each word is represented as a count of occurrences in corpus. To make it more useful we can remove stop words such as 'the' 'is' from the corpus. Words not used very often can be because the words are wrongly spelt. So we can also remove such rarely used words from corpus. Now we can represent each word as a frequency of the word appearing in the corpus.

n-grams: n-gram is a sequence of n words appearing in corpus. This kind of feature representation is useful to find the context of a word. A phrase "red apple" where red is followed by apple is more likely to appear than apple followed by red which conveys more meaning to apple using the n sequence of words. So, it is represented as a frequency of each n-words appearing in the corpus.

7. Using python, implement n-gram approach starting from n = 1 to n = 5 for feature representation. Let f1 ...fk be the vectors representation of the n-gram features, whereby f1 is bag of words, f2 is the frequency of pairs of words, as dened for the n-gram model, and so forth. Using the n-gram representations, build and test a binary classier on the three datasets attached (the datasets are sentiments; binary labels). Study the impact of n on the performance of the classier (more details are below). Details. Evaluate your classier on the three datasets, using the features extract with the n-gram model and 10-fold cross-validation. In reporting the results, calculate 7 metrics: false positive rate, false negative rate, true positive rate, true negative rate, accuracy, precision and recall, and report them in that order for each experiment. Report the results for each dataset, and for n = 1...5, combined and separated. That is, report the results for features representation of f1, f2, ..., f5 (5 experiments for each dataset; in total you'll have 15 experiments), and [f1f2], [f1f2f3], [f1f2f3f4], [f1f2f3f4f5] (4 experiments for each dataset; in total you'll have 12 experiments). Provide those 27 tables in your answer sheet.

Results:

Dataset 1: amazon_cells_labelled.txt

```
   ...:
   ...: ##--- END ----------------------------------------
Dataset 1
------------------------------------------------------------
f1
------------------------------------------------------------
False Positive Rate=  0.3168188677294516
False Negative Rate=  0.1413380707672173
True Positive Rate=  0.8586619292327828
True Negative Rate=  0.6831811322705483
Accuracy=  0.7719999999999999
Precision=  0.7320819112627986
Recall=  0.858
------------------------------------------------------------
f2
------------------------------------------------------------
False Positive Rate=  0.47565342200796545
False Negative Rate=  0.0997475378999729
True Positive Rate=  0.9002524621000273
True Negative Rate=  0.5243465779920345
Accuracy=  0.7089999999999999
Precision=  0.6521106259097527
Recall=  0.8959999999999999
------------------------------------------------------------
f3
------------------------------------------------------------
False Positive Rate=  0.5966792985317266
False Negative Rate=  0.2318128695351948
True Positive Rate=  0.7681871304648052
True Negative Rate=  0.4033207014682735
Accuracy=  0.562
Precision=  0.5454545454545455
Recall=  0.7440000000000001
```

```
------------------------------------------------------------
f4
------------------------------------------------------------
False Positive Rate=  0.7473269736891777
False Negative Rate=  0.20117563900101493
True Positive Rate=  0.7988243609989851
True Negative Rate=  0.2526730263108222
Accuracy=  0.49899999999999994
Precision=  0.4993531694695989
Recall=  0.772
------------------------------------------------------------
f5
------------------------------------------------------------
False Positive Rate=  0.5886652686894023
False Negative Rate=  0.396
True Positive Rate=  0.604
True Negative Rate=  0.41133473131059767
Accuracy=  0.4779999999999999
Precision=  0.4815436241610739
Recall=  0.574
------------------------------------------------------------
f1f2
------------------------------------------------------------
False Positive Rate=  0.32135329639664845
False Negative Rate=  0.1117479154571962
True Positive Rate=  0.8882520845428038
True Negative Rate=  0.6786467036033518
Accuracy=  0.7830000000000001
Precision=  0.7346600331674958
Recall=  0.8859999999999999
```

```
------------------------------------------------------------
f1f2f3
------------------------------------------------------------
False Positive Rate=  0.32408096791122337
False Negative Rate=  0.1117479154571962
True Positive Rate=  0.8882520845428038
True Negative Rate=  0.6759190320887767
Accuracy=  0.782
Precision=  0.7334437086092714
Recall=  0.8859999999999999
------------------------------------------------------------
f1f2f3f4
------------------------------------------------------------
False Positive Rate=  0.3280809679112234
False Negative Rate=  0.10792138484495131
True Positive Rate=  0.8920786151550487
True Negative Rate=  0.6719190320887767
Accuracy=  0.782
Precision=  0.7319078947368421
Recall=  0.89
------------------------------------------------------------
f1f2f3f4f5
------------------------------------------------------------
False Positive Rate=  0.32983535387613566
False Negative Rate=  0.10415643058738619
True Positive Rate=  0.8958435694126138
True Negative Rate=  0.6701646461238645
Accuracy=  0.7830000000000001
Precision=  0.0
Recall=  0.0

In [4]:
```

Dataset 2: imdb_labelled.txt

```
   ...: ##--- END ----------------------------------------------------
Dataset 2
------------------------------------------------------------
f1
------------------------------------------------------------
False Positive Rate=  0.34416753579852305
False Negative Rate=  0.3526683853007932
True Positive Rate=  0.6473316146992067
True Negative Rate=  0.6558324642014769
Accuracy=  0.6049999999999999
Precision=  0.6069246435845214
Recall=  0.596
------------------------------------------------------------
f2
------------------------------------------------------------
False Positive Rate=  0.4793751656487144
False Negative Rate=  0.42189426275469877
True Positive Rate=  0.5781057372453013
True Negative Rate=  0.5206248343512857
Accuracy=  0.44000000000000006
Precision=  0.4431818181818182
Recall=  0.46799999999999997
------------------------------------------------------------
f3
------------------------------------------------------------
False Positive Rate=  0.4890373056981933
False Negative Rate=  0.47483989281159095
True Positive Rate=  0.5251601071884091
True Negative Rate=  0.5109626943018066
Accuracy=  0.39
Precision=  0.391304347826087
Recall=  0.396
```

```
------------------------------------------------------------
f4
------------------------------------------------------------
False Positive Rate=  0.4902507232401157
False Negative Rate=  0.48881646655231564
True Positive Rate=  0.5111835334476844
True Negative Rate=  0.5097492767598844
Accuracy=  0.38
Precision=  0.38
Recall=  0.38
------------------------------------------------------------
f5
------------------------------------------------------------
False Positive Rate=  0.4966939890710382
False Negative Rate=  0.5
True Positive Rate=  0.5
True Negative Rate=  0.5033060109289618
Accuracy=  0.36999999999999994
Precision=  0.36947791164658633
Recall=  0.368
------------------------------------------------------------
f1f2
------------------------------------------------------------
False Positive Rate=  0.31462208381456447
False Negative Rate=  0.3731846131120712
True Positive Rate=  0.6268153868879288
True Negative Rate=  0.6853779161854355
Accuracy=  0.608
Precision=  0.6158798283261803
Recall=  0.574
```

```
------------------------------------------------------------
f1f2f3
------------------------------------------------------------
False Positive Rate=  0.3188455279308998
False Negative Rate=  0.3769903735616672
True Positive Rate=  0.6230096264383328
True Negative Rate=  0.6811544720691001
Accuracy=  0.603
Precision=  0.6102783725910064
Recall=  0.57
------------------------------------------------------------
f1f2f3f4
------------------------------------------------------------
False Positive Rate=  0.31130788618174365
False Negative Rate=  0.3801429887993076
True Positive Rate=  0.6198570112006925
True Negative Rate=  0.6886921138182562
Accuracy=  0.6050000000000001
Precision=  0.613882863340564
Recall=  0.5660000000000001
------------------------------------------------------------
f1f2f3f4f5
------------------------------------------------------------
False Positive Rate=  0.31204343217405794
False Negative Rate=  0.38333103363118803
True Positive Rate=  0.616668966368812
True Negative Rate=  0.6879565678259422
Accuracy=  0.602
Precision=  0.0
Recall=  0.0

In [3]:
```

Dataset 3: yelp_labelled.txt

```
   ...:
   ...: ##--- END ------------------------------------------------------------
Dataset 3
-------------------------------------------------------------
f1
-------------------------------------------------------------
False Positive Rate=  0.3322099200868506
False Negative Rate=  0.26202554915109777
True Positive Rate=  0.7379744508489023
True Negative Rate=  0.6677900799131494
Accuracy=  0.675
Precision=  0.6685934489402697
Recall=  0.6940000000000001
-------------------------------------------------------------
f2
-------------------------------------------------------------
False Positive Rate=  0.5006551665336721
False Negative Rate=  0.2098840775634308
True Positive Rate=  0.7901159224365693
True Negative Rate=  0.49934483346632774
Accuracy=  0.567
Precision=  0.5529225908372828
Recall=  0.7
-------------------------------------------------------------
f3
-------------------------------------------------------------
False Positive Rate=  0.7249948283759878
False Negative Rate=  0.20125324875556144
True Positive Rate=  0.7987467512444386
True Negative Rate=  0.27500517162401217
Accuracy=  0.43900000000000006
Precision=  0.4598155467720685
Recall=  0.698
```

```
------------------------------------------------------------
f4
------------------------------------------------------------
False Positive Rate=  0.7855912190655197
False Negative Rate=  0.19285714285714287
True Positive Rate=  0.8071428571428572
True Negative Rate=  0.21440878093448035
Accuracy=  0.4079999999999999
Precision=  0.442211055276382
Recall=  0.7040000000000001
------------------------------------------------------------
f5
------------------------------------------------------------
False Positive Rate=  0.7981132075471697
False Negative Rate=  0.19821428571428573
True Positive Rate=  0.8017857142857142
True Negative Rate=  0.2018867924528302
Accuracy=  0.398
Precision=  0.43624999999999997
Recall=  0.698
------------------------------------------------------------
f1f2
------------------------------------------------------------
False Positive Rate=  0.33203435178180923
False Negative Rate=  0.24264278517594245
True Positive Rate=  0.7573572148240575
True Negative Rate=  0.6679656482181907
Accuracy=  0.683
Precision=  0.6736242884250474
Recall=  0.71
```

```
------------------------------------------------------------
f1f2f3
------------------------------------------------------------
False Positive Rate=  0.3357380554855129
False Negative Rate=  0.2400686970300458
True Positive Rate=  0.7599313029699541
True Negative Rate=  0.6642619445144871
Accuracy=  0.6809999999999999
Precision=  0.6710775047258979
Recall=  0.71
------------------------------------------------------------
f1f2f3f4
------------------------------------------------------------
False Positive Rate=  0.34325343920779317
False Negative Rate=  0.23887536201150858
True Positive Rate=  0.7611246379884916
True Negative Rate=  0.6567465607922067
Accuracy=  0.6779999999999999
Precision=  0.6666666666666666
Recall=  0.7120000000000001
------------------------------------------------------------
f1f2f3f4f5
------------------------------------------------------------
False Positive Rate=  0.3518469056324756
False Negative Rate=  0.22596974008931067
True Positive Rate=  0.7740302599106894
True Negative Rate=  0.6481530943675244
Accuracy=  0.679
Precision=  0.0
Recall=  0.0

In [2]:
```

References:

1. https://www.udemy.com/deeplearning/learn/v4/overview

2. https://heartbeat.fritz.ai/5-regression-loss-functions-all-machine-learners-should-know-4fb140e9d4b0

3. https://towardsdatascience.com/common-loss-functions-in-machine-learning-46af0ffc4d23

4. https://github.com/khanhnamle1994/natural-language-processing/blob/master/Lecture-4-Word-Window-Classification-and-Neural-Networks/CS224n-Lecture4.pdf

5. https://web.stanford.edu/∼ jurafsky/slp3/3.pdf

6. http://uc-r.github.io/creating-text-features