AUTOMATIC GENDER RECOGNITION THROUGH DIGITAL CONTENTS READING PATTERN USING MACHINE LEARNING

Rucha R. Mahajan[1], Aakash Ahuja[2], Umakant Mandawkar[3]

[1]PG Student, Department of Computer Science and Engineering, SOCSE, Sandip University, Nashik, India
[2]Managing Partner, ITMTB Technologies, Pune, India
[3]Assistant Professor, Department of Computer Science and Engineering, SOCSE, Sandip University, Nashik, India

---***---

Abstract: - In day-to-day life like many user use digital platform for various purposes that is difficult to understand that user is male or female. So the main aim of my research is to automatic gender recognition through digital content reading pattern using machine learning. The motive of research helps predict a gender by analyzing gender information which is helpful for website holder. The digital contents like url's, the userid of the gender, tags, description and long-description these contents are used to predict gender. These contents can be used for any type of url's like facebook, times of india, daily soaps, cricket and so on. Now Logistic Regression, Naïve Bayes, Support Vector Machine, Random Forest, ANN algorithms are used to recognize the gender. This research has purposeful use of the automatic gender prediction for the digital contents.

Keywords:* *Gender Recognition, Logistic Regression, SVM, Random Forest, ANN, Naïve Bayes.

I. Introduction

In this online life destinations like Twitter, Instagram, Facebook and so on, has an emerging trends for those users how can contact their near and dear ones by video call, sending pictures, doing chats and so on. Some of the sites generate a vast number of profiles for a specific user. For that user a unique ID is getting generated. But other than social media sites, there are more url's like cricket, daily soaps from which gender prediction can be made possible. Because every url contains a distinctive ID which help to predict the gender.When we open any url for specific search liking going through the news of a TV star or Film star, so these have some description and some tags which will be helpful in predicting gender.

For some url's there are various languages used. These various languages are interpreted in regular expressions, so that it can be easily understandable by every user. For solving this machine learning is used for text analysis. Here, by using NLP (Natural Language Processing) and the concept of NLP is used to generate tags which are used further for prediction. Finally, performing experiment on the url and userid dataset will give us approximate accuracy.

A. Social Media:

If some people do not have any application of social media, then we can use url to open the application. That url will generate a distinctive ID for that particular user. Many user login in day-to-day life, it is difficult to say that the user is male or female. So using this ID we can predict that the user is male or female.

B. News:

For particular news, there are many sites like Times of India, Navbharat Times, Economic Times and Cricbuzz and so on. This site contains the heading and the news. By using the description of that news we can predict that the user is male or female.

C. Daily Soaps:

For a particular daily soaps, there are more than one url's like pinksvilla, tellybuzz etc. These url's have long description, gossips etc. By using this long description we can easily say that the user is male or female.

II. Related Work

Gender prediction got more importance in previous few years and has built vast growth in investigation and growth of the field. We have discussed various techniques for predicting the gender.

Facial images based gender detection is described in [11]. Writer used color FERET, LFW, Adience dataset build with BHEP. In [12], writer used multiscale approximation based on DWT for detecting sex by BPNN. In [13], they have given us introduction about MLP and LPC by ANN. In [14], they discussed concept of patterns which is related to gender by SVM. In [15], we understand the methods of models like AAM, SAM and compare them. In [16], they used the concept of GA and PCA by FEI dataset. In [17], they have discussed about the concept ACF based on GD by K-means. In [18], here they used Gabor filter based on additive sum of non-linear functions by Logistic Regression. In [19], they used FERET database by LDA and PCA. In [20], here they used the concept of human gait identification by SVM.

By giving quick review about previous techniques and algorithms we can say that it is used for finite dataset. So we require a generalised structured method for sex determination by using digital contents.

In this paper, we are going to use the concept of digital contents and following are some steps: 1) Pre-processing: In pre-processing part we load json data file and 12 part files of the data-sets and drop the duplicates from the 12 part files of the data-sets. 2) Create Dataframe: Combining all the part files we create one data-frame and then merge it with json data file, then the final data-frame is created, so that we can convert the csv file from the data-frame. By using to_csv we can convert the data-frame into csv file.3) Generate Tags: For this we will use the concept of natural language processing. NLTK is the package that we use in NLP. In NLTK we use stemmer, lemmatizer, corpus, tokenization, Tfidftransformer and Tfidfvectorizer, Regular Expressions, part-of-speech. 4) Predict the gender: After generating tags from the old tags we will apply SVM, Random Forest, Logistic Regression, Naïve Bayes, and Artificial Neural Network which will produce accuracy approximately 86-87%.

III. Methodology

A. Methods:

1. NLP:

NLP is known as Natural Language Processing. It is particularly described because the automatic utilization of natural language like textual content, software and speech. It has been taken into consideration at some point of for more than 50 years and grows up inside the subject of morphological with the assist of PC[30]. In this method of NLP, the bundle referred to as NLTK is used.

1.1NLTK:

NLTK is known as Natural Language Toolkit. It is a plan used for structuring python codes that take duties with character speech statistics for claim in analytical NLP. It consists of text processing statistics centre like parsing, tokenization, stemming, category, character count, lemmatization, tagging. [4]

1.1.1 Stemming:

It is the operation of constructing structural alternative of a base or root term. Codes of stemming are regularly mentioned to as stemming layout or stemmers. A stemming layout abates the phrases "consultant", "consulting", "consultantative" and "consultants" to the root term "consult". [29]

1.1.2 Lemmatization:

It is the operation of accumulating collectively the numerous balance shapes of a term so that it will be explored as a one piece. It is near to stemming however it guides conditions to the terms. So it associates with phrases near relevant to the term. Text processing has both lemmatization as well as stemming. [28]

1.1.3 Corpus:

A corpus is a huge and organized set of device comprehensible narrative that has been created in a affordable expressive scenario. It is twin in nature called as corpora. This can be received in various techniques like narrative that turned into initially script of spoken language, optical person reputation and digital and so forth. [26]

1.1.4 Tokenization:

It is the movement of splitting up a chain of cable into slice such as terms, access, sentence, characters and any other issue called tokens. It can be singular terms, sentence or even full rap. The technique of tokenization a few symbols like punctuation marks are removed. It is performs a big role inside the system of lexical analysis. [27]

1.1.5 Tfidftransformer and Tfidfvectorizer:

With this Tfidftransformer you will correctly calculate the term total using the idea of count number vectorizer after which calculate the IDF i.e. Inverse Document Frequency values after which calculate the Tf-idf counts. With the Tfidfvectorizer, we must do 3 steps:-

- Firstly, it will compute the word count
- Then IFD values are calculated
- Lastly, a Tf-idf count uses the dataset.

If you require calculating tf-idf counts on reports inside your "training" dataset use Tfidfvectorizer. [23]

1.1.6 Regular Expressions:

It is a language for mainly narratively locate cable. It allows us to discover the take a look at or different cables or set of cables, that uses a particular syntax in a pattern. It is particularly used to locate narrative in UNIX as well as MS WORD in unique way. We have extraordinary locating mechanisms for features of RE. [25]

1.1.7 Part-of-Speech:

Labelling is a sort of category, which may be defined because the automated venture of detailing of the tokens. The descriptor is likewise known as tags, that's used to represent single of the part-of-speech, well-formed statistics and so forth. POS tagging is then explained as the operation of giving single of the part-of-speech to the given term. This tagging is known as POS Tagging. [24]

B. Algorithms:

1. Support Vector Machine:

It is a supervised mastering set of rules. It is used for each of the regression and classification problems. It has their personal way of imposing as differentiate with the other machine getting to know algorithms. It has potential to address many non-stop and categorical values. I actually have used linear kernel, this kernel is used for enforcing the SVM in python. Linear Kernel may be used as a dot product between two observations. The formula is given as- $k(x, x_i) = \text{sum}(x^*x_i)$ ------------- 1

From this above equation, we can say that the output of the 2 vectors say x & xi is the total of the product of every pair of the input values. [22]

2. Naïve Bayes:

Naïve Bayes is a supervised gaining knowledge of algorithm. It is simply used for classification problems. It produces higher results in a complex actual-international scenario. It also calls for little amount of dataset for training purpose. That is used to access the variables which are essential for category problems and that can be trained incrementally. Naïve Bayes is a conditional probability model, we are represented as a vector $x = (x_1, x2, ----,x_n)$. This vector represents n features i.e. (independent variables). This assigns the sample probabilities for every value of k for every particular outcomes or classes-

$$p\ (C_k|x_1, \text{-------}, x_n) \text{ ------------ } 2$$

The problem with the above formula is if we take n quantity of functions and that is massive, then biasing a model on the given probability table is impossible. So we have to rewrite the formula for the model to make it clean. So with the aid of using the Bayes theorem, the can say that-

$$p\ (C_k|x) = p\ (C_k)\ p(x|C_k)/p(x) \text{ ---------- } 3$$

So this means that the freedom premise, the conditional distribution over the class variable C is-[3]

$$p\ (C_k|x_1, \text{-------}, x_n) = 1/z\ p\ (C_k)\ \pi_{i=1}{}^{n}\ p(x_i|C_k). \text{ ---------------- } 4$$

I even have used multinomial naïve bayes for my work. So by using multinomial incident approach, characteristic vectors are represented with the frequencies with certain incidents which have been occurred by means of a multinomial $(p_1, \text{-------}, p_n)$, here p_i is the probability that incident i occurs for multinomials in the multiclass case. A vector function $x = (x_1, \text{-------}, x_n)$ is a histogram, which contains x_i counting the number of times incident i used to be visible in every instance. This is the incident method specifically used for report classification, with incidents representing the occurrence of a term in one report. For observing a histogram x is given by- [21]

$$p\ (x|C_k) = (\textstyle\sum_i x_i)!/\pi_i\ x_{i!}\ \pi\ p_{ki}{}^{xi} \text{ ------- } 5$$

3. Random Forest:

It is an ensemble set of rules. Random forest classifier generates a set of decision trees from randomly selected subset of training set. It then sums the votes from numerous decision trees to remedy the very last class of the check item. It is used mainly for category problems[6]. It is used for each of classification and regression issues. It is a supervised studying set of rules. This approach may be very bendy and easy to apply. A forest includes trees. It has various requests like photograph classification, characteristic choice and recommendation engines[5]. By Laymen's words, if training set is given as [x1, x2, x3, x4] with their particular variables as [l1, l2, l3, l4]. So random forest may generate three decision trees which takes input of subset for ex-

1 [x1, x2, x3]

2 [x1, x2, x4]

3 [x2, x3, x4]

So primarily based on this, sooner or later it's going to predict the votes on the basis of majority from every of the decision tree build. There is another way also i.e. we can apply weight method for considering the output from any decision tree. So those trees with massive mistake rate have given low weight values and vice versa. This likely increases the decision tree impact with low error rate. [6]

4. Logistic Regression:

It may be used for different category issues such as diabetes prediction, spam detection. It is quite simple and mostly used in machine learning algorithms for two class category. It is easy to develop and used as the measure for any binary classification problem. It explains and evaluates the relation between one binary dependent variable and independent variables. It is an analytical procedure for predicting binary classes. It calculates the probability of an incident situation. It is a selected sample of linear regression, where the desired variable is categorical. It uses a log of odds because of the dependent variable. It makes use of logit function i.e. - [7]

$$\text{logit (pi)} = \ln (\text{pi} / 1 - \text{pi})$$

$$= \beta_0+\beta_1 x_1, i\text{+-------+}\beta_k x_k, i \text{ ------- } 6$$

5. Artificial Neural Network:

It is a reality handling sample i.e. activated through the mind. It is built for a selected request like statistics classification or pattern recognition, along a learning technique. The training process has following 2 ways:

- **Forward Propagation:**

Now, consider the inputs, and then multiply it by the given weights-

$$y = w_i x_i = w_1 x_1 + w_2 x_2 + w_3 x_3\text{------- } 7$$

So skip the output by a sigmoid formula to compute the neuron's result. The output of the sigmoid function is between 0 and 1:

$$1 / 1 + e^{-y} \text{ ------- } 8$$

- **Back Propagation:**

Compute the mistake i.e. the difference between the real output and the predicted output. Depending on the mistake, we can alter the weight by multiplying the error with the input and again the sigmoid curve used by gradient:

> Weight += Error Input Output (1 - Output), where (1 – Output) is the derivative of sigmoid curve. [8]

6. Cross Validation:

In python and machine learning if we are not able to fit the model on the training data and is not ready to accept that the model will work properly and give accurate accuracy for the data. So for this we use the concept of cross validation. The easier side to work with cross validation is that we call cross_val_score, so that we get proper accuracy [9]. So here in cross_val_score there is a parameter called cv. CV is an integer argument that uses the KFold or StratifiesKFold methods by default. For prediction we use cross_val_predict, i.e. used to predict the output for the model [10]. Validation helps us to evaluate the standard of the model. Validation allows us to select the model that will perform well on the covered data. Validation allows you to avoid Overfitting and Underfitting. [9]

6.1 Underfitting:

It is referred not to catch the different patterns in the dataset. So this happens when training and test set is not working properly. [9]

6.2 Overfitting:

It is referred as following two ways: a) catch the sound b) catch the patterns which does not recognize the covered data. So it will work well in training set but badly in test set. [9]

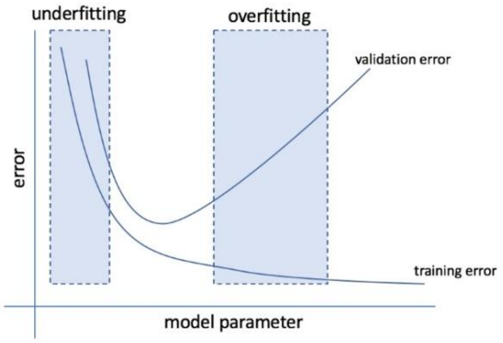


Fig 1: Shows the concept of Overfitting and under fitting [9].

IV. Experiment and Results

1. Comparative Result Analysis:

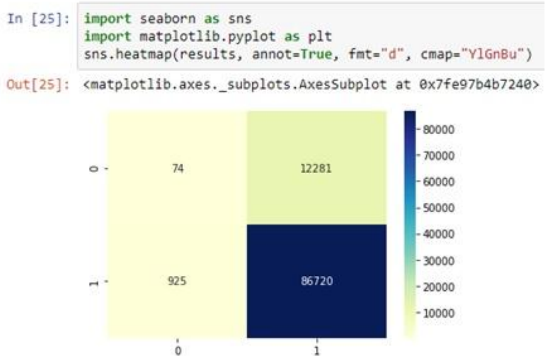
Algorithm	Train Accuracy	Test Accuracy
Random Forest Classifier	86.628	87.288
SVM	87.471	87.315
Logistic Regression	87.614	87.291
Logistic Regression (Multinomial)	87.547	87.304
Random Forest Classifier (n=100)	87.645	87.288
ANN	87.048	87.288
Naive Bayes (Multinomial)	86.998	87.098

Table 1: Compare the accuracy

So by comparing the accuracy based on the algorithms. So we can conclude that for Random Forest classifier test accuracy is better than train accuracy. In SVM train accuracy is better test accuracy. In Logistic Regression train accuracy is better than test accuracy. In Logistic Regression (Multinomial) train accuracy is better than test accuracy. In Random Forest Classifier (n=100) train accuracy is better than test accuracy. In ANN test accuracy is better than train accuracy. In Naive Bayes test accuracy is better than train accuracy. So we can say that Random Forest Classifier (n=100) got 87.645% accuracy than other algorithms.

2) Comparative Result Graph

- **Random Forest Classifier: Confusion Matrix**

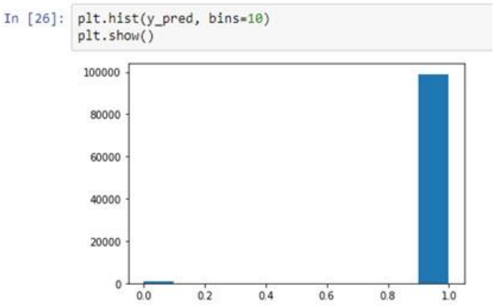


```
In [25]: import seaborn as sns
         import matplotlib.pyplot as plt
         sns.heatmap(results, annot=True, fmt="d", cmap="YlGnBu")

Out[25]: <matplotlib.axes._subplots.AxesSubplot at 0x7fe97b4b7240>
```

Fig 2: Confusion Matrix for Random Forest Classifier

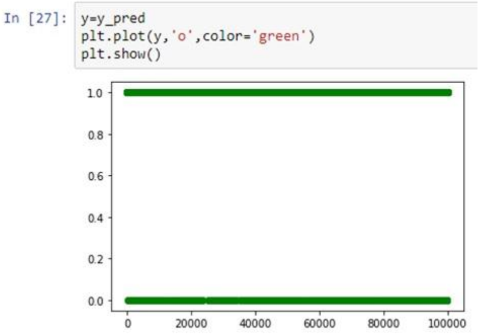
- **Random Forest Classifier: Histogram**



```
In [26]: plt.hist(y_pred, bins=10)
         plt.show()
```

Fig 3: Histogram for Random Forest Classifier

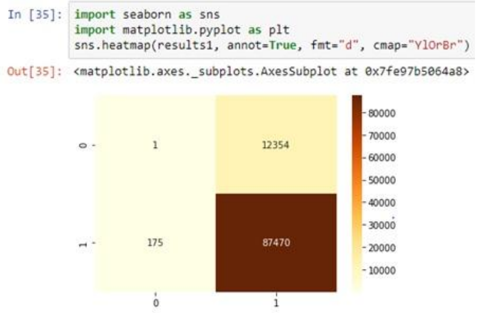
- **Random Forest Classifier: Scatter Plot**



```
In [27]: y=y_pred
         plt.plot(y,'o',color='green')
         plt.show()
```

Fig 4: Scatter Plot for Random Forest Classifier

- **SVM : Confusion Matrix**

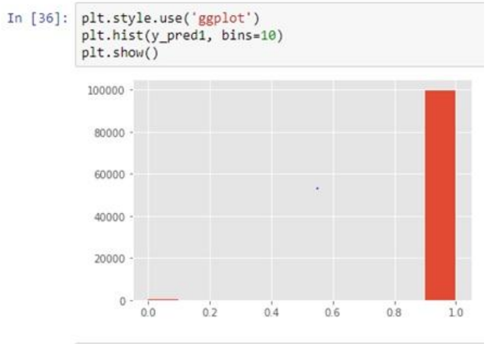


```
In [35]: import seaborn as sns
         import matplotlib.pyplot as plt
         sns.heatmap(results1, annot=True, fmt="d", cmap="YlOrBr")

Out[35]: <matplotlib.axes._subplots.AxesSubplot at 0x7fe97b5064a8>
```

Fig 5: Confusion Matrix for SVM

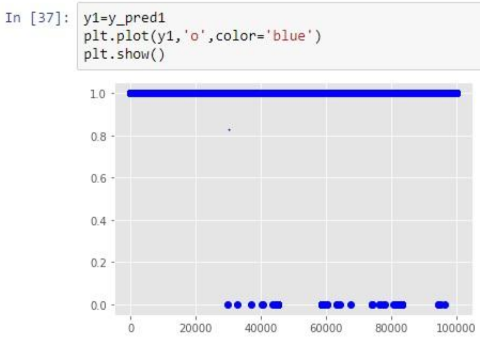
- **SVM : Histogram**



```
In [36]: plt.style.use('ggplot')
         plt.hist(y_pred1, bins=10)
         plt.show()
```

Fig 6: Histogram for SVM

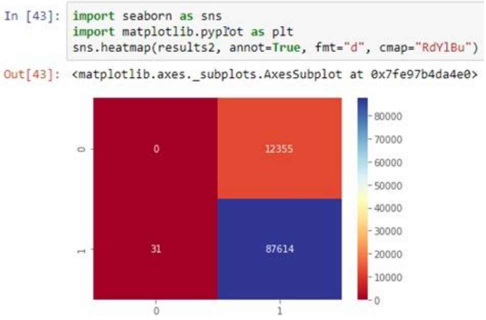
- **SVM : Scatter Plot**



```
In [37]: y1=y_pred1
         plt.plot(y1,'o',color='blue')
         plt.show()
```

Fig 7: Scatter Plot for SVM

- **Logistic Regression : Confusion Matrix**

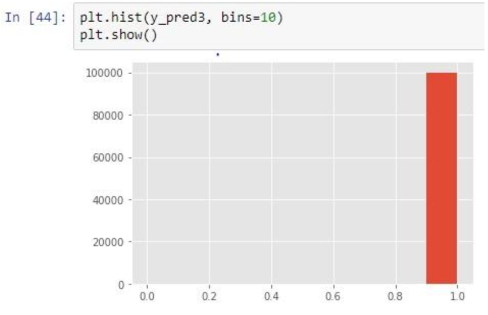


```
In [43]: import seaborn as sns
         import matplotlib.pyplot as plt
         sns.heatmap(results2, annot=True, fmt="d", cmap="RdYlBu")

Out[43]: <matplotlib.axes._subplots.AxesSubplot at 0x7fe97b4da4e0>
```

Fig 8: Confusion Matrix for Logistic Regression

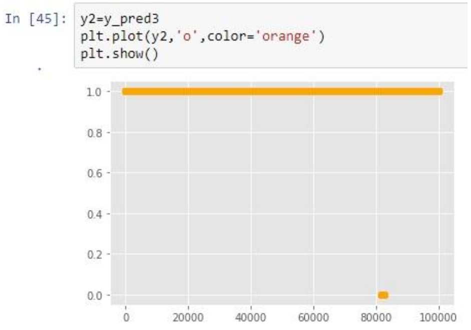
- **Logistic Regression : Histogram**



```
In [44]: plt.hist(y_pred3, bins=10)
         plt.show()
```

Fig 9: Histogram for Logistic Regression

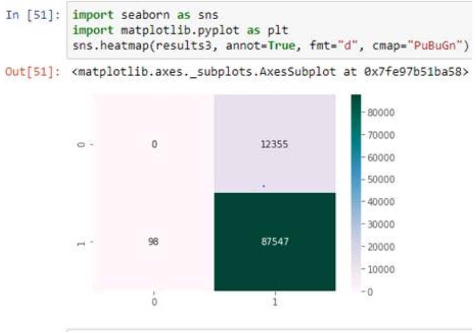
- **Logistic Regression : Scatter Plot**



```
In [45]: y2=y_pred3
         plt.plot(y2,'o',color='orange')
         plt.show()
```

Fig 10: Scatter Plot for Logistic Regression

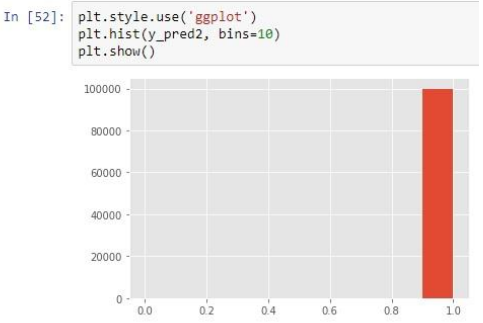
- **Logistic Regression (Multinomial) : Confusion Matrix**



```
In [51]: import seaborn as sns
         import matplotlib.pyplot as plt
         sns.heatmap(results3, annot=True, fmt="d", cmap="PuBuGn")
Out[51]: <matplotlib.axes._subplots.AxesSubplot at 0x7fe97b51ba58>
```

Fig 11: Confusion Matrix for Logistic Regression (Multinomial)

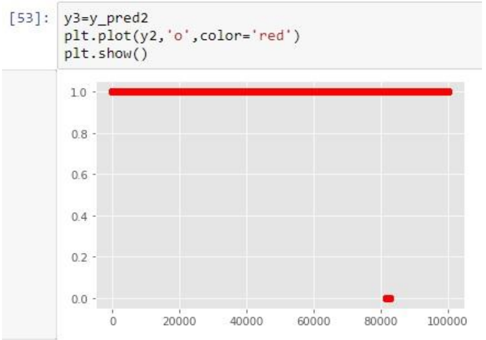
- **Logistic Regression (Multinomial) : Histogram**



```
In [52]: plt.style.use('ggplot')
         plt.hist(y_pred2, bins=10)
         plt.show()
```

Fig 12: Histogram for Logistic Regression (Multinomial)

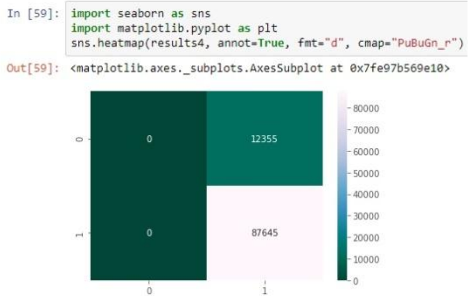
- **Logistic Regression (Multinomial) : Scatter Plot**



```
[53]: y3=y_pred2
      plt.plot(y2,'o',color='red')
      plt.show()
```

Fig 13: Scatter Plot for Logistic Regression (Multinomial)

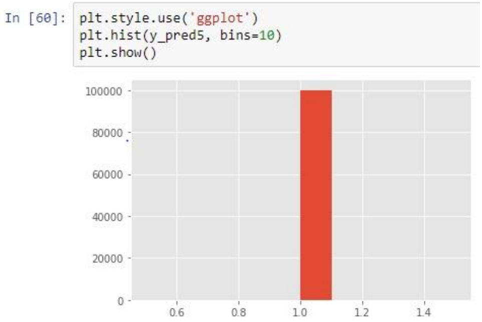
- **Random Forest Classifier (n=100) : Confusion Matrix**



```
In [59]: import seaborn as sns
         import matplotlib.pyplot as plt
         sns.heatmap(results4, annot=True, fmt="d", cmap="PuBuGn_r")
Out[59]: <matplotlib.axes._subplots.AxesSubplot at 0x7fe97b569e10>
```

Fig 14: Confusion Matrix for Random Forest Classifier (n=100)

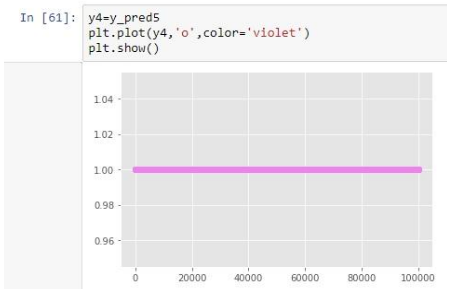
- **Random Forest Classifier (n=100) : Histogram**



```
In [60]: plt.style.use('ggplot')
         plt.hist(y_pred5, bins=10)
         plt.show()
```

Fig 15: Histogram for Random Forest Classifier (n=100)

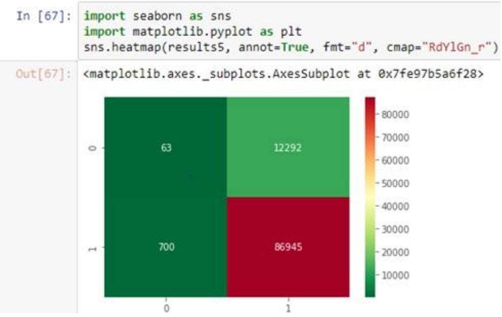
- **Random Forest Classifier (n=100) : Scatter Plot**



```
In [61]: y4=y_pred5
         plt.plot(y4,'o',color='violet')
         plt.show()
```

Fig 16: Scatter Plot for Random Forest Classifier (n=100)

- **ANN : Confusion Matrix**

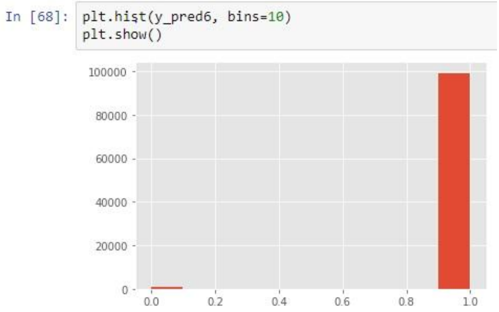


```
In [67]: import seaborn as sns
         import matplotlib.pyplot as plt
         sns.heatmap(results5, annot=True, fmt="d", cmap="RdYlGn_r")

Out[67]: <matplotlib.axes._subplots.AxesSubplot at 0x7fe97b5a6f28>
```

Fig 17: Confusion Matrix for ANN

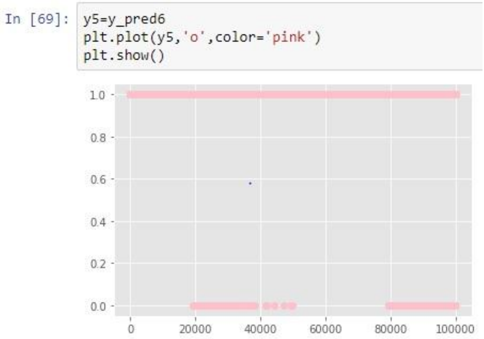
- **ANN : Histogram**



```
In [68]: plt.hist(y_pred6, bins=10)
         plt.show()
```

Fig 18: Histogram for ANN

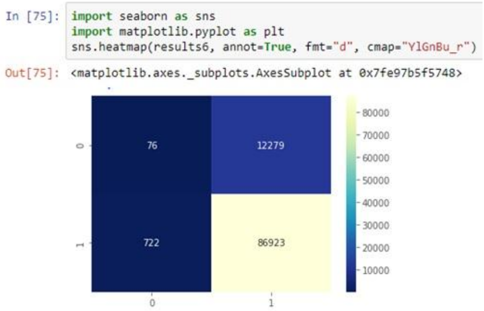
- **ANN : Scatter Plot**



```
In [69]: y5=y_pred6
         plt.plot(y5,'o',color='pink')
         plt.show()
```

Fig 19: Scatter Plot for ANN

- **Naive Bayes : Confusion Matrix**

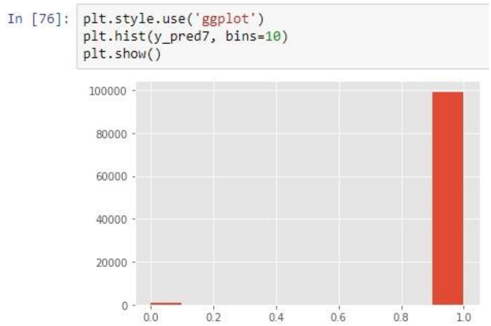


```
In [75]: import seaborn as sns
         import matplotlib.pyplot as plt
         sns.heatmap(results6, annot=True, fmt="d", cmap="YlGnBu_r")

Out[75]: <matplotlib.axes._subplots.AxesSubplot at 0x7fe97b5f5748>
```

Fig 20: Confusion Matrix for Naive Bayes

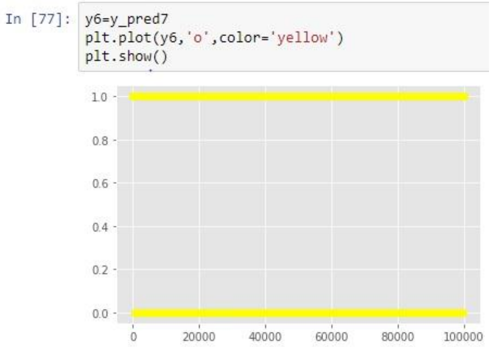
- **Naive Bayes : Histogram**



```
In [76]: plt.style.use('ggplot')
         plt.hist(y_pred7, bins=10)
         plt.show()
```

Fig 21: Histogram for Naive Bayes

- **Naive Bayes : Scatter Plot**



```
In [77]: y6=y_pred7
         plt.plot(y6,'o',color='yellow')
         plt.show()
```

Fig 22: Scatter Plot for Naive Bayes

V. Conclusion

We can conclude that a visitor's gender can be predicted to a good accuracy using url, content headings, and detailed content and known anonymized training data. This information can further be used by digital media houses t show targeted content to visitor's and increase revenue.

VI. Acknowledgement

I Miss Rucha Rajiv Mahajan would like to express my gratitude to my external guide Mr. Aakash Ahuja for the useful comments, remarks and engagement through the learning process of this master thesis. Furthermore I would like to thank Aakash Ahuja for introducing me to the topic as well for the support and guide on the way. Also, I like to thank the internal guide Mr. Umakant Mandawkar for helping me and supporting me. I would like to thank my loved ones, who have supported me throughout entire process, both by keeping me harmonious and helping me putting pieces together. I will be grateful forever for your love.

References

1. https://www.tutorialspoint.com/natural_language_processing/natural_language_processing_introduction.htm

2. https://www.guru99.com/nlp-tutorial.html
3. https://www.tutorialspoint.com/big_data_analytics/naive_bayes_classifier.htm
4. https://www.techopedia.com/definition/30343/natural-language-toolkit-nltk
5. https://www.datacamp.com/community/tutorials/random-forests-classifier-python
6. https://medium.com/machine-learning-101/chapter-5-random-forest-classifier-56dc7425c3e1
7. https://www.datacamp.com/community/tutorials/understanding-logistic-regression-python
8. https://www.geeksforgeeks.org/implementing-ann-training-process-in-python/
9. https://medium.com/@george.drakos62/cross-validation-70289113a072
10. https://scikit-learn.org/stable/modules/cross_validation.html
11. Md. Nurul Ahad Tawhid, Emon Kumar Dey, "A Gender Recognition System from Facial Image", International Journal of Computer Applications (0975 – 8887) Volume 180 – No.23, February 2018.
12. Prabha, Jitendra Sheetlani, "Fingerprint-based Automatic Human Gender Identification", International Journal of Computer Applications (0975 - 8887) Volume 170 – No.7, July 2017.
13. Yusnita M. A., Hafiz A. M., Nor Fadzilah M., Aida Zulia Zulhanip, Mohaiyedin Idris, "Automatic Gender Recognition using Linear Prediction Coefficients and Artificial Neural Network on Speech Signal", 7th IEEE International Conference on Control System, Computing and Engineering (ICCSCE), 24–26 November 2017.
14. Ankita Jain, Vivek Kanhangad, "Investigating Gender Recognition in Smartphones using Accelerometer and Gyroscope Sensor Readings", International Conference on Computational Techniques in Information and Communication Technologies (ICCTICT), 2016.
15. Ali Maina Bukar, Hassan Ugail, David Connah, "Automatic age and gender classification using supervised appearance model", Journal of Electronic Imaging 25(6), 061605 (Nov/Dec 2016).
16. Yimin Zhou, Zhifei Li, "Real-time Gender Recognition based on Eigen-features selection from Facial Images", Institute of Electrical and Electronics Engineers (IEEE) 2016.
17. Mamta Kumari, Israj Ali, "An Efficient Algorithm for Gender Detection using Voice Samples", 2015 International Conference on Communication, Control and Intelligent Systems (CCIS) 978-1-4673-7541-2/15/$31.00©2015 IEEE.
18. Md. Hafizur, Md. Abul Bashar, Fida Hasan Md. Rafi, Tasmia Rahman, Abu Farzan Mitual, "An Automatic Face Detection and Gender Identification from Color Images using Logistic Regression", 978-1-4799-0400-6/13/$31.00©2013 IEEE.
19. Terishka Bissoon, Serestina Viriri, "Gender Classification using Face Recognition", 978-1-4799-3067-8/13/$31.00©2013 IEEE.
20. Xuelong Li, Stephen J. Maybank, Shuicheng Yan, Dacheng Tao, Dong Xu, "Gait Components and Their Application to Gender Recognition", IEEE Transactions On Systems, Man, And Cybernetics-Part C: Applications And Reviews, Vol 38. No. 2, March 2008, 1094-6977/$25.00©2008 IEEE.
21. https://en.wikipedia.org/wiki/Naive_Bayes_classifier#Multinomial_naive_Bayes
22. https://www.tutorialspoint.com/machine_learning_with_python/machine_learning_with_python_classification_algorithms_support_vector_machine.htm
23. https://kavita-ganesan.com/tfidftransformer-tfidfvectorizer-usage-differences/#Tfidftransformer-vs-Tfidfvectorizer
24. https://www.tutorialspoint.com/natural_language_processing/natural_language_processing_part_of_speech_tagging.htm
25. https://www.tutorialspoint.com/natural_language_processing/natural_language_processing_word_level_analysis.htm
26. https://www.tutorialspoint.com/natural_language_processing/natural_language_processing_linguistic_resources.htm
27. https://www.techopedia.com/definition/13698/tokenization
28. https://www.geeksforgeeks.org/python-lemmatization-with-nltk/
29. https://www.geeksforgeeks.org/python-stemming-with-nltk/
30. https://machinelearningmastery.com/natural-language-processing/