VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELAGAVI - 590018, KARNATAKA

A Mini Project Report
on

"INSTAGRAM FAKE ACCOUNT DETECTION"

SUBMITTED IN PARTIAL FULFILLMENT FOR THE REQUIREMENT OF THE VI SEMESTER MINI PROJECT (21ISMP67)

BACHELOR OF ENGINEERING
in
INFORMATION SCIENCE & ENGINEERING

AMULYA C	**1AM21IS009**
BHUVI T G	**1AM21IS018**
JYOTHI U	**1AM21IS045**
KEERTHANA M G	**1AM21IS051**

Under the Guidance of
Mrs. VINUTHA M
Assistant Professor **Dept. of ISE,**
AMCEC

AMC ENGINEERING COLLEGE

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

18th K.M. Bannerghatta Main Road, Bengaluru-560083

2023-24

AMC ENGINEERING COLLEGE

18th K.M. Bannerghatta Main Road Bengaluru – 560083

CERTIFICATE

Certified that the project work entitled **"INSTAGRAM FAKE ACCOUNT DETECTION"** has been successfully completed by **AMULYA C (1AM21IS009), BHUVI T G (1AM21IS018), JYOTHI U (1AM21IS045)** and **KEERTHANA M G (1AM21IS051)** all bona fide students of **AMC Engineering College, Bengaluru** in partial fulfilment of the requirements for the award of degree in **Bachelor of Engineering in Information Science and Engineering** of **Visvesvaraya Technological University, Belagavi** during the academic year **2023-2024**. The project report has been approved as it satisfies the academic requirements in respect of project work for the said degree.

Mrs. Vinutha M	Dr. Amutha R	Dr. K Kumar
Project Guide	Professor and HOD	Principal
Assistant Professor	Department of ISE	AMCEC
Department of ISE	AMCEC	

AMC ENGINEERING COLLEGE

18th K.M. Bannerghatta Main Road Bengaluru – 560083

DECLARATION

We, the students of VI semester of Information Science and Engineering, AMC Engineering College Bengaluru, hereby declare that the Mini project work entitled **"INSTAGRAM FAKE ACCOUNT DETECTION"** has been carried out by us at AMC Engineering College, Bengaluru and submitted in partial fulfilment of the course requirements of **Bachelor of Engineering** in **Information Science and Engineering** of **Visvesvaraya Technological University, Belagavi**, during the academic year **2023-2024**. We also declare that to the best of my knowledge and belief, the work reported here does not form part of any other dissertation on the basis of which a degree or an award was conferred on an earlier occasion on this by any other student.

Date:

Place: Bengaluru

NAME	USN	SIGNATURE
AMULYA C	**1AM21IS009**	
BHUVI T G	**1AM21IS018**	
JYOTHI U	**1AM21IS045**	
KEERTHANA M G	**1AM21IS051**	

ACKNOWLEDGMENT

It gives us immense pleasure to present before you our Mini project titled **“INSTAGRAM FAKE ACCOUNT DETECTION”**. The joy and satisfaction that accompany the successful completion of any task would be incomplete without the mention of those who made it possible. We are glad to express my gratitude towards my prestigious institution **AMC ENGINEERING COLLEGE** for providing me with utmost knowledge, encouragement and the maximum facilities in undertaking this Mini Project.

In this regard, we express our sincere gratitude to the Chairman **Dr. K Paramahamsa** and the Principal **Dr. K Kumar**, for providing us all the facilities in this college.

We are extremely grateful to our Professor and Head of the Department, Information Science and Engineering, **Dr. R. Amutha**, for having accepted to patronize us in the right direction with all her wisdom.

We place our heartfelt thanks to **Mrs. Vinutha M**, Assistant Professor, Department of Information Science and Engineering for having guided us for the project, and all the staff members of our department for helping us out at all times.

We thank **Ms. Poonam Siddarkar**, Mini Project Coordinator, Department of Information Science and Engineering. We thank our beloved friends for having supported us with all their strength and might. Last but not the least, we thank our parents for supporting and encouraging us throughout. We made an honest effort in this assignment.

AMULYA C
1AM21IS009

BHUVI T G
1AM21IS018

JYOTHI U
1AM21IS045

KEERTHANA M G
1AM21IS051

ABSTRACT

The project titled "**Instagram Fake Account Detection**" project aims to develop an innovative system that is designed to detect Instagram accounts as a fake account or a legit account. Social media plays a crucial role in our lives by creating connections, sharing information, and allowing people to find what they're looking for, whether it's an entertaining video or a new household product. Additionally, it provides a platform for individuals to share exciting news, pictures, and videos with family and friends, bridging distances.

These days spam accounts have become a major problem in in all the social media platforms. Many users are creating fake accounts to create an illusion of having many followers to their personal accounts. Fake accounts are being created to sell fake products and services. They are also being used to impersonate other account users from common people to celebrities in order to influence, criticize, hurt feelings and reputation.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

Detecting fake accounts on Instagram is crucial to ensure the safety and privacy of its users. With the platform's immense popularity, the presence of fake profiles and scammers has grown. These accounts can be used for malicious purposes, such as spreading misinformation, phishing, or identity theft. Machine learning models can help address this issue by classifying accounts as real or fake. This project aim is to build and train a deep neural network model to detect fake or spam Instagram accounts. There are few key input features which we considered to determine if the account is fake or not.

The Input Features are:

- Profile Picture - The user has profile picture or not.
- Nums/Length Username - The ratio of number of numerical chars in username to its length.
- Fullname Words - Full name in word tokens
- Name/Length of Full Name - The ratio of number of numerical characters in full name to its length.
- Name = = Username - Are username and full name literally the same?
- Description Length - Bio length in characters.
- External URL - Has external URL or not.
- Private - Private or not.
- Posts - Number of posts.
- Followers - Number of followers.
- Follows - Number of follows.

1.1 Problem Definition

It is difficult to determine the authenticity of an Instagram account with complete certainty. However, it is possible to build a model that can predict the likelihood that an Instagram account is fake, based on certain characteristics of the account and its activity. Many users are creating fake accounts to create an illusion of having many followers to their personal accounts. Fake accounts are being created to sell fake products and services. They are also being used to impersonate other account users from common people to celebrities in order to influence, criticize, hurt feelings and reputation.

1.2 Objectives of the Project

Some possible characteristics that could be used as input features for a fake Instagram account detection model include:

- The number of followers the account has
- The ratio of followers to following
- The age of the account
- The amount of activity on the account (e.g. number of posts, comments, likes)
- The type of content that is posted
- The use of hashtags
- The presence of a profile picture and biography
- The use of third-party apps to boost the account's activity

Using these and other relevant features, it is possible to train a machine learning model to predict the likelihood that an Instagram account is fake. However, it is important to note that building an accurate fake Instagram account detection model would likely require a large and diverse dataset of real and fake accounts, as well as careful feature engineering and model selection. It would also be important to continuously update the model as fake accounts evolve and change over time.

1.3 Scope of the Project

The project titled "Instagram Fake Account Detection" involves building and training a deep neural network model to identify fake or spam Instagram accounts. These days, spam accounts have become a significant issue across social media platforms. Users create fake accounts to inflate follower counts, sell counterfeit products, or impersonate others. This model is trained such that it considers the above given features and determines whether a particular account is fake or not. By resulting the output as either 0 or 1 meaning TRUSTED or FAKE respectively. Our intention is to make this software capable of thinking like a human, based on the data it is given and results in maximum probability of success.

Chapter 2

SYSTEM REQUIREMENTS

2.1 Hardware Requirements

- Processor : intel i5 / i7
- RAM : 8 GB
- Hard Disk : 512 GB

2.2 Software Requirements

- OS : Windows 10/ 11
- Python 3.8 or above.
- Jupyter notebook

Chapter 3

SYSTEM ANALYSIS

3.1 Proposed System

This model is trained such that it considers the above given features and determines whether a particular account is fake or not. By resulting the output as either 0 or 1 meaning TRUSTED or FAKE respectively. Our intention is to make this software capable of thinking like a human, based on the data it is given and results in maximum probability of success.

3.2 Methodology

3.2.1 ANN Training Process

ANN is rarely used for predictive modelling. The reason being that Artificial Neural Networks (ANN) usually tries to over-fit the relationship. ANN is generally used in cases where what has happened in past is repeated almost exactly in same way.

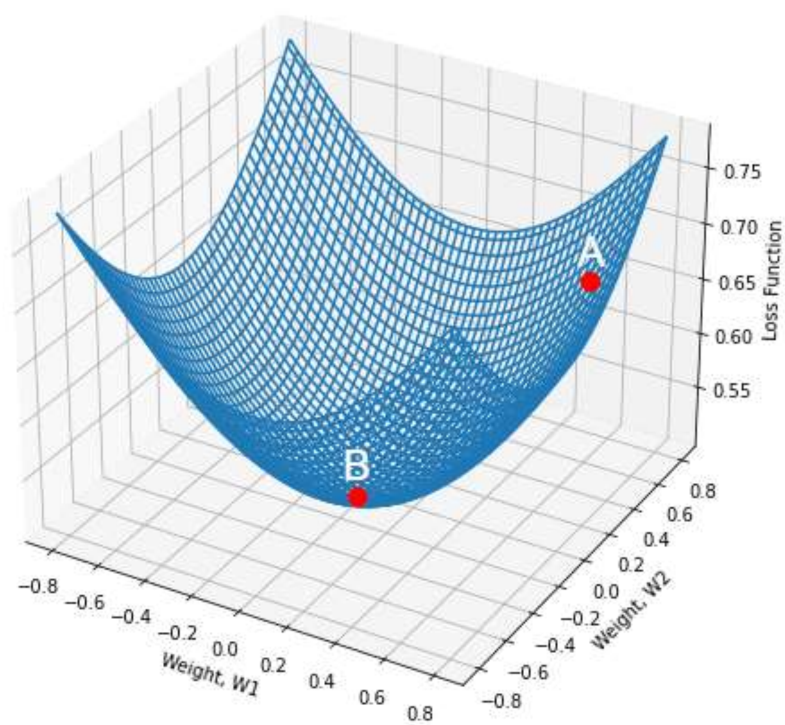

Fig 3.1: ANN Gradient Descent

- Gradient descent is an optimization algorithm used to obtain the optimized network weight and bias values.
- It works by iteratively trying to minimize the cost function.
- It works by calculating the gradient of the cost function and negative direction until the local or global minimum is achieved.
- If the positive of the gradient is taken, local or global maximum is achieved.

3.2.2 Back Propagation

Back propagation is used to train ANN's by calculating gradient needed to update network weights. It is commonly used as gradient descent optimization algorithm to adjust the weight of neurons by calculating the gradient of the loss function.

1. **PHASE 1:**
   - Propagation forward through the network to generate the output value(s)
   - Calculation of the cost (error term).
   - Propagation of output activations back through network using training pattern target in order to generate the deltas (differences between targeted and actual output values).
2. **PHASE 2:**
   - Weight update
   - Calculating weight gradient.
   - This ratio influences the speed and quality of learning also called as learning rate. The greater the ratio, the faster neuron train, but lower ratio, more accurate the training is.

3.2.3 Confusion Matrix

A confusion matrix is a matrix that summarizes the performance of a machine learning model on a set of test data. It is a means of displaying the number of accurate and inaccurate instances based on the model's predictions. It is often used to measure the performance of classification models, which aim to predict a categorical label for each input instance. The matrix displays the number of instances produced by the model on the test data.

When assessing a classification model's performance, a confusion matrix is essential. It offers a thorough analysis of true positive, true negative, false positive, and false negative predictions, facilitating a more profound comprehension of a model’s recall, accuracy, precision, and overall effectiveness in class distinction. When there is an uneven class distribution in a dataset, this matrix is especially helpful in evaluating a model’s performance beyond basic accuracy metrics.

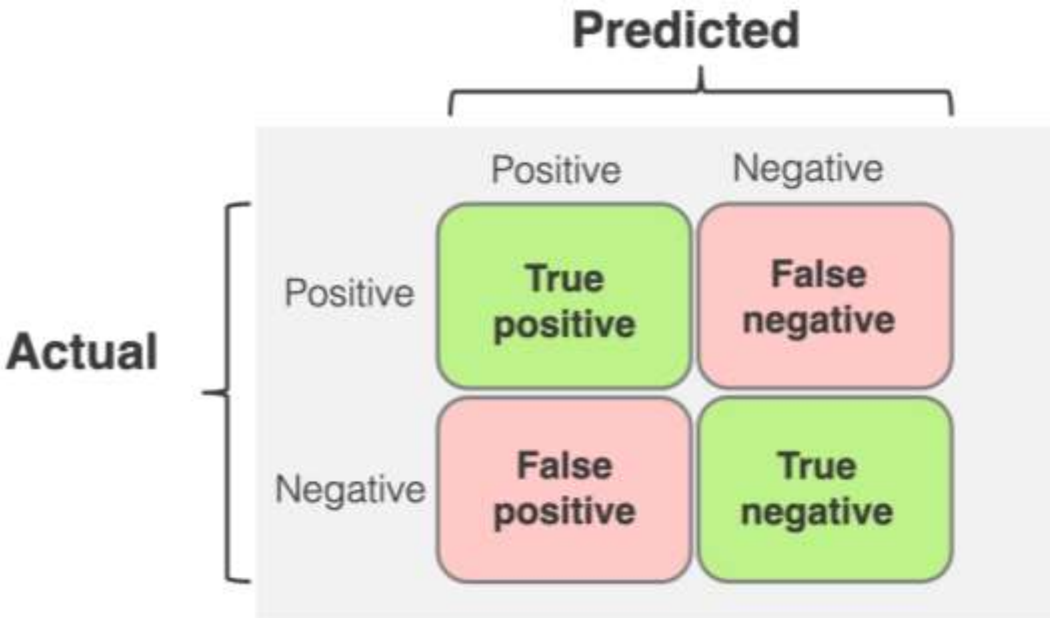

Fig 3.2: Confusion Matrix

- **True Positive (TP):** The model correctly predicted a positive outcome.
- **True Negative (TN):** The model correctly predicted a negative outcome.
- **False Positive (FP):** The model incorrectly predicted a positive outcome. Also known as a Type I error.
- **False Negative (FN):** The model incorrectly predicted a negative outcome. Also known as a Type II error.

Chapter 4

SYSTEM DESIGN

4.1 Data Flow Diagram

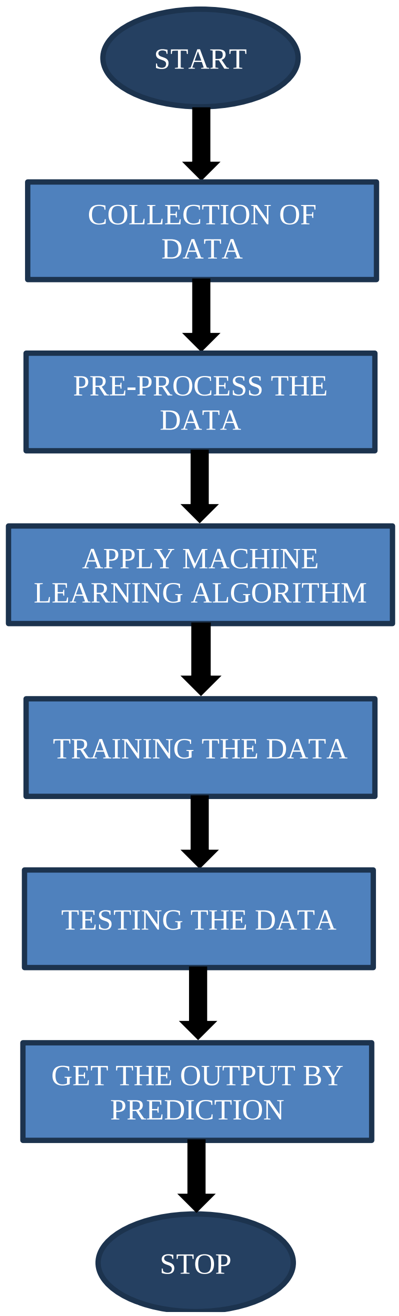

Fig 4.1: Flow Diagram of how the software uses the data to train the model and get the predicted output for the profiles to check if accounts are fake or real.

Chapter 5

IMPLEMENTATION

```
#importing libraries
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.layers import Dense, Activation, Dropout
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.metrics import Accuracy
from sklearn import metrics
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import classification_report,accuracy_score,roc_curve,confusion_matrix
from jupyterthemes import jtplot
jtplot.style(theme = 'monokai', context = 'notebook', ticks = True, grid = False)
#Loading training and testing datasets
instagram_df_test = pd.read_csv('test.csv')
instagram_df_train = pd.read_csv('train.csv')
instagram_df_train
instagram_df_test
instagram_df_train.info()
instagram_df_train.describe()
instagram_df_train.isnull().sum()
instagram_df_train['profile pic'].value_counts()
instagram_df_train['fake'].value_counts()
instagram_df_train['external URL'].value_counts()
(instagram_df_train['description length'] > 50).sum()
```

```
instagram_df_test.info()
instagram_df_test.describe()
instagram_df_test.isnull().sum()
instagram_df_test['fake'].value_counts()
#number of fake and real accounts
sns.countplot(instagram_df_train['fake'])
#private column
sns.countplot(instagram_df_train['private'],palette = "PuBu")
#Visualizing the profile pic feature
sns.countplot(instagram_df_train['profile pic'],palette = "Pastel2")
#length of usernames(Histogram)
plt.figure(figsize = (20, 10))
sns.distplot(instagram_df_train['nums/length username'],kde=True)
#Correlation heatmap
plt.figure(figsize=(15,15))
cm = instagram_df_train.corr()
ax = plt.subplot()
sns.heatmap(cm, annot = True, ax = ax)
sns.countplot(instagram_df_test['fake'])
sns.countplot(instagram_df_test['private'],palette = "Set2")
sns.countplot(instagram_df_test['profile pic'])
#Preparing inputs
x_train = instagram_df_train.drop(columns = ['fake'])
x_test = instagram_df_test.drop(columns = ['fake'])
x_train
x_test
#Preparing the outputs
y_train = instagram_df_train['fake']
y_test = instagram_df_test['fake']
y_train
y_test
```

```
#Scaling the data before training
from sklearn.preprocessing import StandardScaler, MinMaxScaler
scaler_x = StandardScaler()
X_train = scaler_x.fit_transform(x_train)
X_test = scaler_x.transform(x_test)
Y_train = tf.keras.utils.to_categorical(y_train, num_classes=2)
Y_test = tf.keras.utils.to_categorical(y_test, num_classes=2)
X_train.shape,X_test.shape
Y_train.shape,Y_test.shape
#Percentage of Traininf data
Training_data_percentage = len(X_train)/(len(X_train) + len(X_test) ) * 100
Training_data_percentage
Testing_data_percentage = len(X_test)/(len(X_train) + len(X_test) ) * 100
Testing_data_percentage
#Main model
import tensorflow.keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
model = Sequential()
model.add(Dense(50,input_dim = 11, activation = "relu")) #Initial Layer
model.add(Dropout(0.3))
model.add(Dense(150, activation = "relu"))
model.add(Dropout(0.3))
model.add(Dense(25, activation = "relu"))
model.add(Dropout(0.3))
model.add(Dense(2, activation = "softmax")) #output layer
model.summary()
model.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['accuracy'])
epochs_hist = model.fit(X_train, Y_train, epochs = 20, verbose = 1, validation_split = 0.1)
print(epochs_hist.history.keys())
plt.plot(epochs_hist.history['loss'])
```

```
plt.plot(epochs_hist.history['val_loss'])
plt.title('Model Loss Progressioin During Training/Validation')
plt.xlabel('Epoch Number')
plt.ylabel('Training and Validation Losses')
plt.legend(['Training Loss','Valdiation Loss'])
predicted = model.predict(X_test)
predicted_value = []
test = []
for i in predicted:
    predicted_value.append(np.argmax(i))
for i in Y_test:
    test.append(np.argmax(i))
print(classification_report(test, predicted_value))
plt.figure(figsize=(10, 10))
con_matrix = confusion_matrix(test,predicted_value)
sns.heatmap(con_matrix, annot=True)
```

Chapter 6

RESULTS AND DISCUSIONS

The Instagram Fake Account Detection developed for this project has shown promising results by achieving 95 percent accuracy in detecting the fake accounts by training the model using datasets. We have checked whether the model has reached the ability to detect an account is fake or not by inputting different set of data values which consisted 120 account details. The model predicted true values for 106 accounts and predicted false values for 14 accounts out of 120 accounts.

Overall, the Instagram Fake Account Detection project represents a significant step forward in leveraging machine learning techniques to identify fake accounts and provide more privacy. While there are still challenges to overcome, the potential benefits in terms of efficiency, accuracy, and privacy makes this an area ripe for continued research and development.

Chapter 7

SCREENSHOTS

	profile pic	nums/length username	fullname words	nums/length fullname	name==username	description length	external URL	private	#posts	#followers	#follows	fake
0	1	0.27	0	0.00	0	53	0	0	32	1000	955	0
1	1	0.00	2	0.00	0	44	0	0	286	2740	533	0
2	1	0.10	2	0.00	0	0	0	1	13	159	98	0
3	1	0.00	1	0.00	0	82	0	0	679	414	651	0
4	1	0.00	2	0.00	0	0	0	1	6	151	126	0
...	...	...	...	...	...	...	...	...	...	...	...	...
571	1	0.55	1	0.44	0	0	0	0	33	166	596	1
572	1	0.38	1	0.33	0	21	0	0	44	66	75	1
573	1	0.57	2	0.00	0	0	0	0	4	96	339	1
574	1	0.57	1	0.00	0	11	0	0	0	57	73	1
575	1	0.27	1	0.00	0	0	0	0	2	150	487	1

576 rows × 12 columns

Fig 7.1: Loading Data from train.csv

	profile pic	nums/length username	fullname words	nums/length fullname	name==username	description length	external URL	private	#posts	#followers	#follows	fake
0	1	0.33	1	0.33	1	30	0	1	35	488	604	0
1	1	0.00	5	0.00	0	64	0	1	3	35	6	0
2	1	0.00	2	0.00	0	82	0	1	319	328	668	0
3	1	0.00	1	0.00	0	143	0	1	273	14890	7369	0
4	1	0.50	1	0.00	0	76	0	1	6	225	356	0
...	...	...	...	...	...	...	...	...	...	...	...	...
115	1	0.29	1	0.00	0	0	0	0	13	114	811	1
116	1	0.40	1	0.00	0	0	0	0	4	150	164	1
117	1	0.00	2	0.00	0	0	0	0	3	833	3572	1
118	0	0.17	1	0.00	0	0	0	0	1	219	1695	1
119	1	0.44	1	0.00	0	0	0	0	3	39	68	1

120 rows × 12 columns

Fig 7.2: Loading Data from test.csv

	profile pic	nums/length username	fullname words	nums/length fullname	name==username	description length	external URL	private	#posts	#followers	#follows	fake
count	576.000000	576.000000	576.000000	576.000000	576.000000	576.000000	576.000000	576.000000	576.000000	5.760000e+02	576.000000	576.000000
mean	0.701389	0.163837	1.460069	0.036094	0.034722	22.623264	0.116319	0.381944	107.489583	8.530724e+04	508.381944	0.500000
std	0.458047	0.214096	1.052601	0.125121	0.183234	37.702987	0.320886	0.486285	402.034431	9.101485e+05	917.981239	0.500435
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000e+00	0.000000	0.000000
25%	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	3.900000e+01	57.500000	0.000000
50%	1.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	9.000000	1.505000e+02	229.500000	0.500000
75%	1.000000	0.310000	2.000000	0.000000	0.000000	34.000000	0.000000	1.000000	81.500000	7.160000e+02	589.500000	1.000000
max	1.000000	0.920000	12.000000	1.000000	1.000000	150.000000	1.000000	1.000000	7389.000000	1.533854e+07	7500.000000	1.000000

Fig 7.3: Fetching multiple data needed to count and train the model using train.csv

```
instagram_df_train['profile pic'].value_counts()
```

```
profile pic
1    404
0    172
Name: count, dtype: int64
```

```
instagram_df_train['fake'].value_counts()
```

```
fake
0    288
1    288
Name: count, dtype: int64
```

```
instagram_df_train['external URL'].value_counts()
```

```
external URL
0    509
1     67
Name: count, dtype: int64
```

```
(instagram_df_train['description length'] > 50).sum()
```

```
98
```

Fig 7.4: Collecting the data for different parameters such as profile pic, external URL

	profile pic	nums/length username	fullname words	nums/length fullname	name==username	description length	external URL	private	#posts	#followers	#follows	fake
count	120.000000	120.000000	120.000000	120.000000	120.000000	120.000000	120.000000	120.000000	120.000000	1.200000e+02	120.000000	120.000000
mean	0.758333	0.179917	1.550000	0.071333	0.041667	27.200000	0.100000	0.308333	82.866667	4.959472e+04	779.266667	0.500000
std	0.429888	0.241492	1.187116	0.209429	0.200664	42.588632	0.301258	0.463741	230.468136	3.816126e+05	1409.383558	0.502096
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000e+00	1.000000	0.000000
25%	1.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	6.725000e+01	119.250000	0.000000
50%	1.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	8.000000	2.165000e+02	354.500000	0.500000
75%	1.000000	0.330000	2.000000	0.000000	0.000000	45.250000	0.000000	1.000000	58.250000	5.932500e+02	668.250000	1.000000
max	1.000000	0.890000	9.000000	1.000000	1.000000	149.000000	1.000000	1.000000	1879.000000	4.021842e+06	7453.000000	1.000000

Fig 7.5: Fetching multiple data needed to count and train the model using test.csv

```
<Axes: ylabel='count'>
```

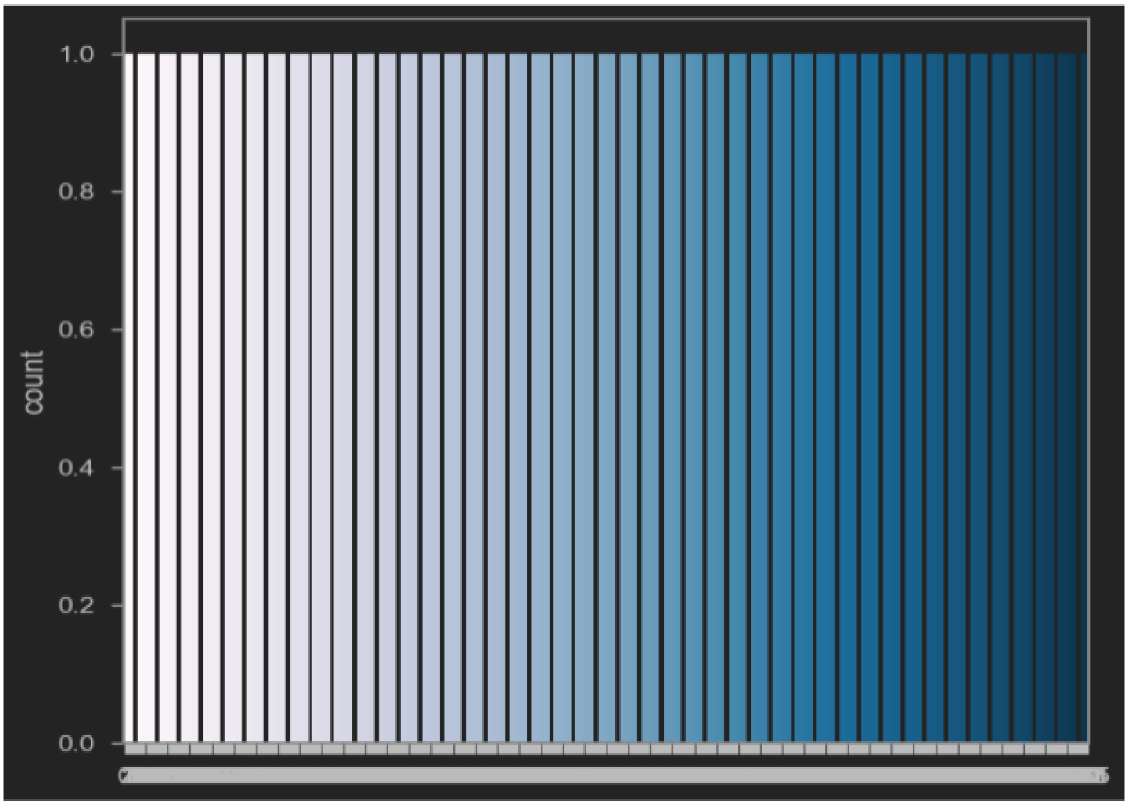

Fig 7.6: Histogram display for the account privacy as open or private

```
<Axes: xlabel='nums/length username', ylabel='Density'>
```

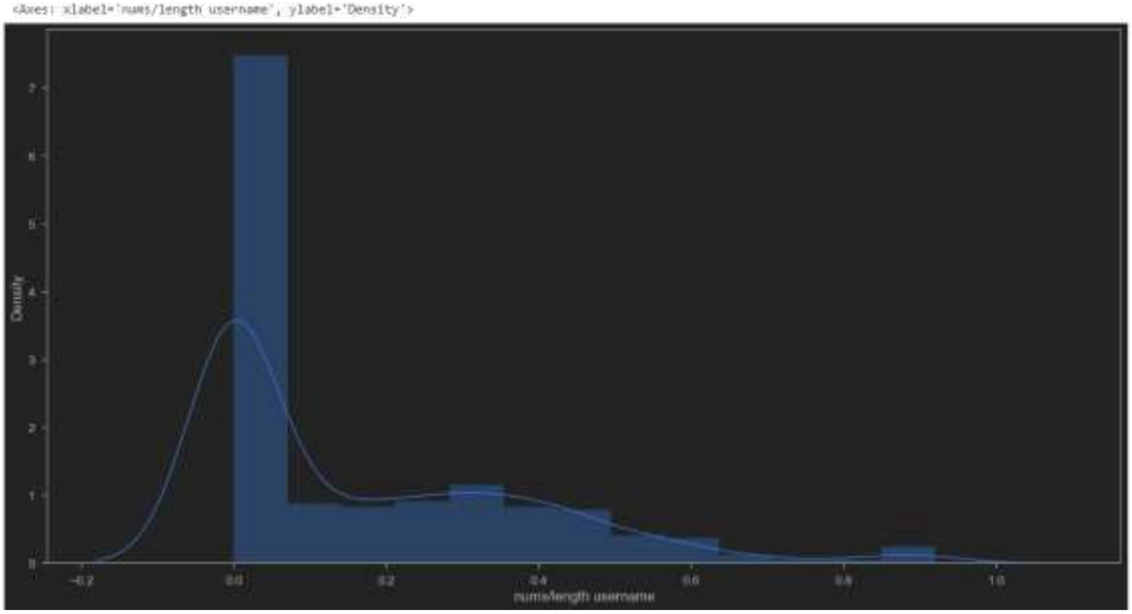

Fig 7.7: Graphical representation of density of the name of the user depending upon the length of the name of the user

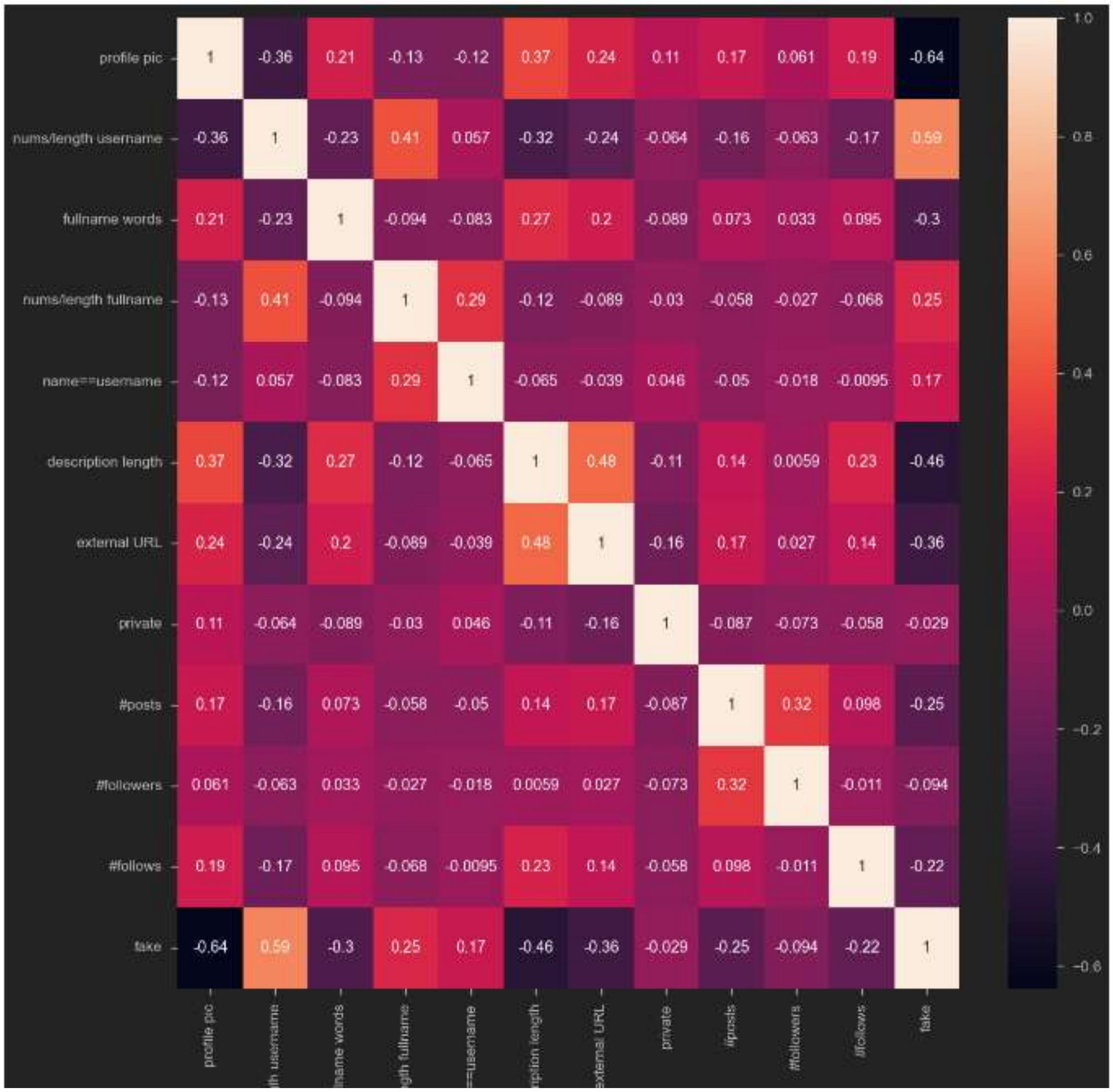

Fig 7.8: Heatmap for all the data to train the model to detect the real or fake account

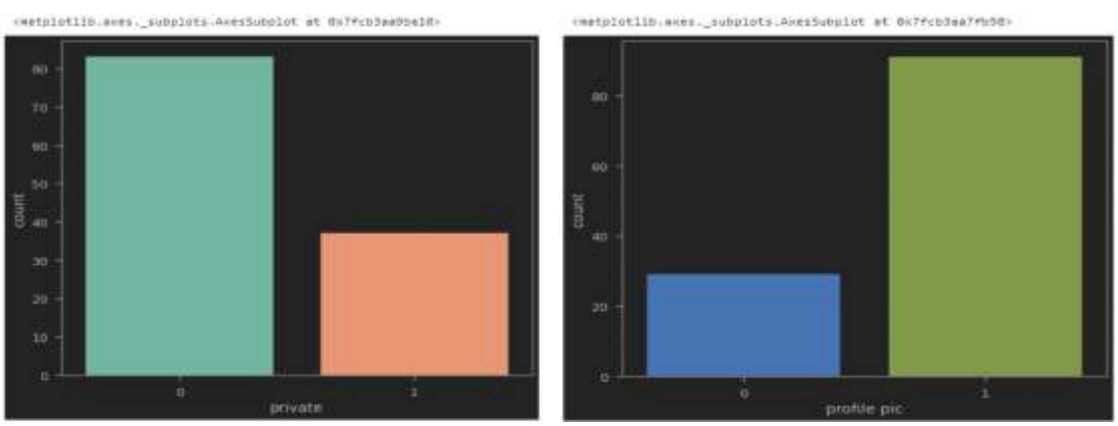

Fig 7.9 Graphical count for private accounts and accounts with profile pictures

```
X_train.shape,X_test.shape
```

```
((576, 11), (120, 11))
```

```
Y_train.shape,Y_test.shape
```

```
((576, 2), (120, 2))
```

```
#Percentage of Traininf data
Training_data_percentage = len(X_train)/(len(X_train) + len(X_test)*) * 100
Training_data_percentage
```

```
95.04950495049505
```

```
Testing_data_percentage = len(X_test)/(len(X_train) + len(X_test)*) * 100
Testing_data_percentage
```

```
6.756756756756757
```

Fig 7.10: Generated accuracy by the model for the correct and fake

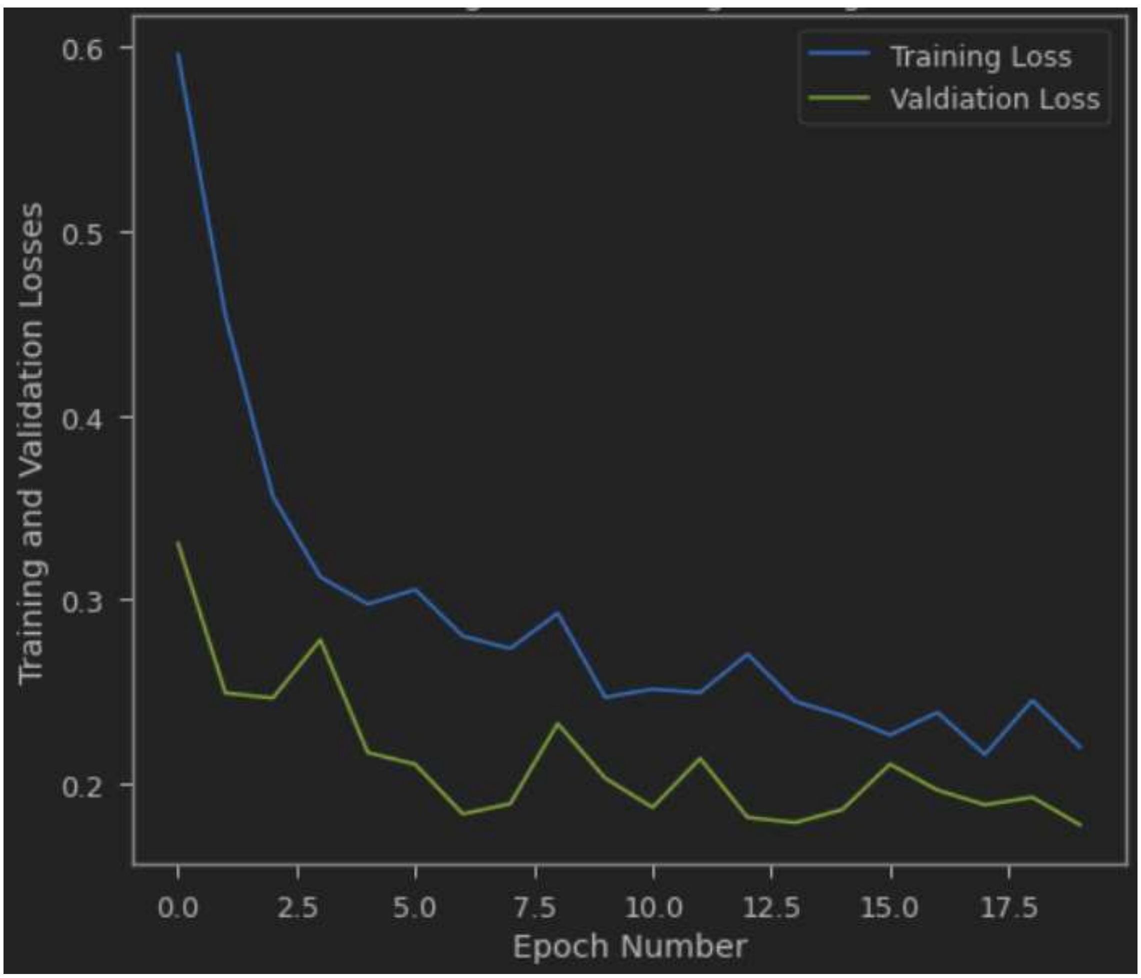

Fig 7.11: Model loss progression during the training of the model

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fcb33f2c510>
```

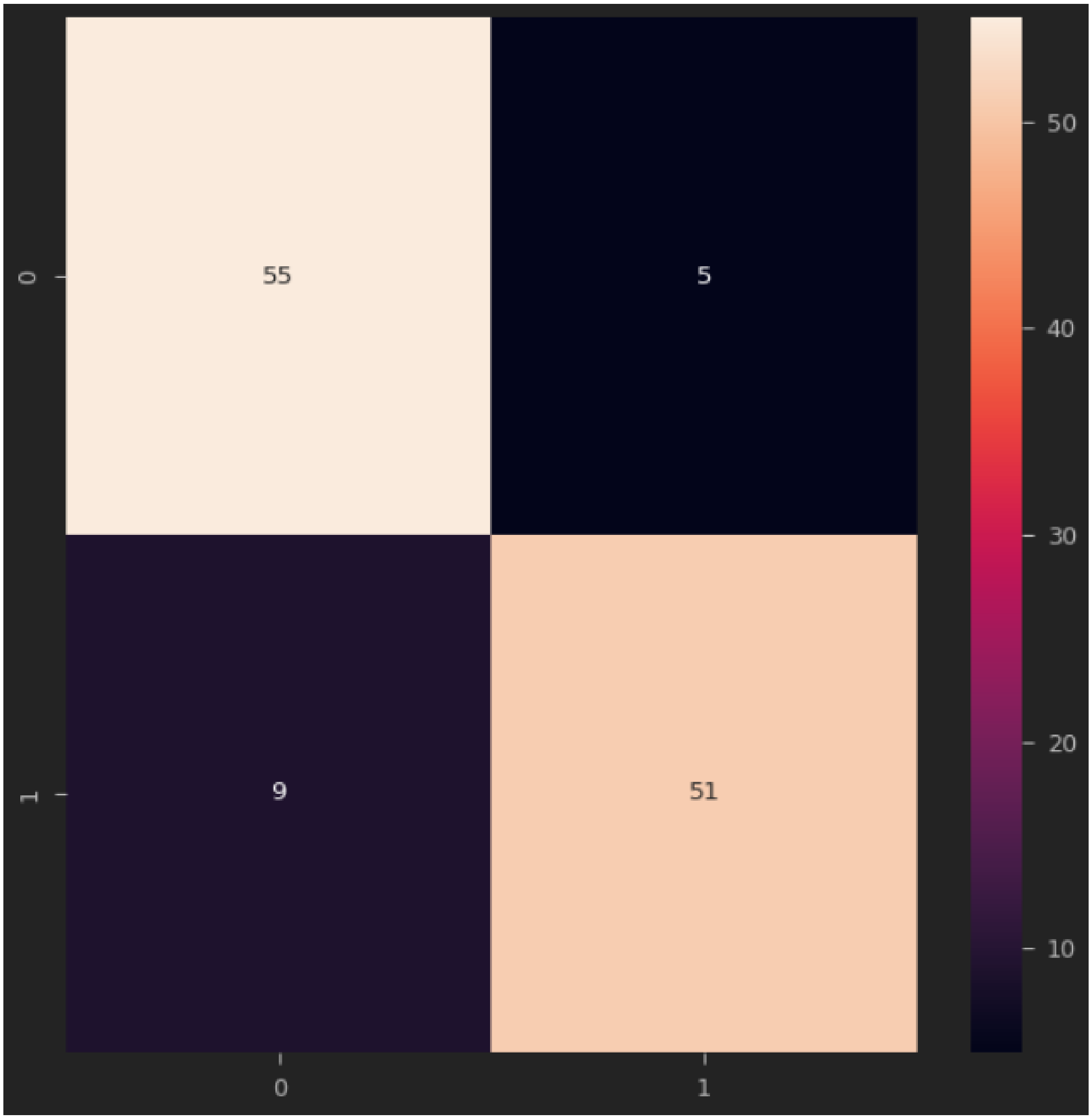

Fig 7.12: Confusion Matrix generated by the model

Chapter 8

CONCLUSION

We have analyzed the dataset and obtained a fairly accurate predictive model using Neural networks. The model is hence trained to detect fake accounts in Instagram based on the considered features. We achieved 95 percent accuracy in detecting the fake accounts by training the model using datasets (from train.csv). We have checked whether the model has reached the ability to detect an account is fake or not by inputting different set of data values (test.csv file) which consisted 120 account details. The model predicted true values for 106 accounts and predicted false values for 14 accounts out of 120 accounts.

REFERENCES

1. Pandas user guide: https://pandas.pydata.org/docs/user_guide/index.html
2. Matplotlib user guide: https://matplotlib.org/3.3.1/users/index.html
3. Seaborn user guide & tutorial: https://seaborn.pydata.org/tutorial.html
4. Tensorflow user guide : https://www.tensorflow.org/guide
5. Tenorflow Playground : https://playground.tensorflow.org
6. Neural Networks: https://towardsdatascience.com/the-mostly-complete-chart-of-neural-networks-explained-3fb6f2367464