

# **Fake News Predictor**

Summer Internship Report

submitted in partial fulfillment of the requirement for the degree of

Bachelor of Technology

in

Information Technology

By

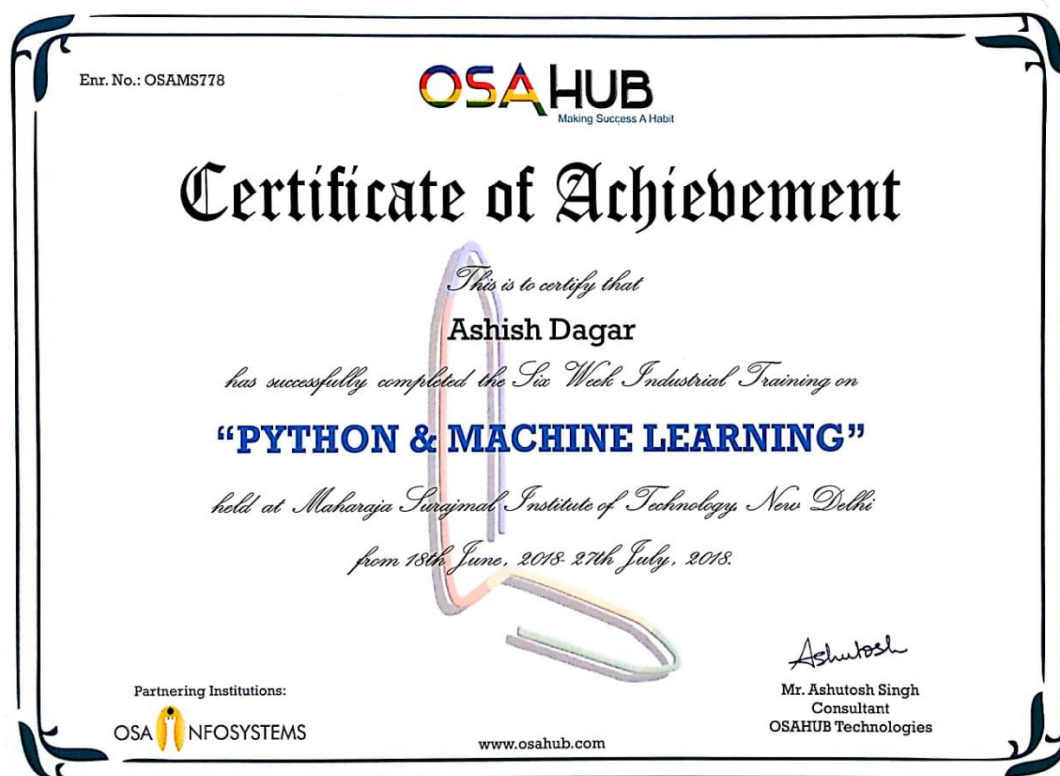
**Ashish Dagar – 00696303116**



Maharaja Surajmal Insitute of Technology  
(Affiliated to Guru Gobind Singh Indraprastha University)  
Janakpuri, New Delhi-58

October 2018

# Certificate



# Feedback Form

## MAHARAJA SURAJMAL INSTITUTE OF TECHNOLOGY

### Summer/Industrial Training Evaluation Form

F05 (MSIT-EXM-P-02)

(Year 2016. -- 2020.)

#### Details of the Student

Name: ASHISH DAGAR  
 Roll No.: 00696303116  
 Branch and Semester: IT, V  
 Mobile No.: 7011837738  
 E-mail ID: ashishdagar1980@gmail.com

#### Details of the Organisation

Name and address of organisation: OSAHUB,  
F-20, 21, A.R. Plaza, Beta-I, Greater Noida  
 Broader Area: IT 201301  
 Name of Instructor: ASHUTOSH SINGH  
 Designation and Contact No: ashutosh@osahub.com

#### Student Performance Record

	No. of days Scheduled for the training	Number of days actually attended	Curriculum Scheduled for the student	Curriculum actually covered by the student
Week 1	5	5	- Introduction - Python loops & data structures	Same
Week 2	5	5	- Pandas - Numpy - Bike share project - Introduction to ML	Same
Week 3	5	5	- Supervised algorithms - KNN, Linear regression - Optimization - SKlearn, Boston housing	Same
Week 4	5	5	- Decision tree - XGBoost - Titanic survivorship - Iris, Kmeans, clustering	Same
Week 5	5	5	- NMF - Fake news classifier - SVM, HOG, Dlib, face detection	Same
Week 6	5	5	- Neural Networks - Back Propagation - CNN, MNIST, CIFAR - Dog breed classifier	Same

(Signature of the student)

Any comments or suggestions for the student performance during the training program (to be filled by instructor):

consistent & dedicated. Takes & solves a problem.

Ashutosh  
 (Signature of the Instructor)  
 Along with Seal

Note : Every student has to fill and submit this Performa duly signed by his/her instructor to the faculty-in-charge by first week of September.

**MAHARAJA SURAJMAL INSTITUTE OF TECHNOLOGY**

**Summer/Industrial Training Evaluation Form**

**F05 (MSIT-EXM-P-02)**

**(Year 20..... -- 20.....)**

**Student Evaluation Record (to be filled by-Examiner)**

<b>S.No.</b>	<b>Evaluation Criteria</b>	<b>Performance</b>
1.	Familiarization with the organisation and its working environment (out of 10)	
2.	Level of technical content in the project (out of 30)	
3.	Organisation and formation of Project Report (out of 20)	
4.	Power Point presentation and Viva-Voce examination (out of 40)	

Overall evaluation of the student (Excellent / V. Good / Good/ Satisfactory / Unsatisfactory)

.....  
.....

**(Signature and Name of the Examiner)**

## **Candidate's Declaration**

I Ashish Dagar, Enrollment No. 00696303116, B.Tech (Semester – 5<sup>th</sup>) of the Maharaja Surajmal Institute of Technology, New Delhi hereby declare that the training report entitled “**Fake News Detector**” is an original work and data provided in the study is authentic to the best of my knowledge. No plagiarism is done while making this project and this project work has not performed the basis for the award of any Degree or diploma/ associateship/fellowship and similar project if any.

## **Acknowledgement**

A project work owes its success from commencement to completion to the people in love with project at various stages. Let me in this page express my gratitude to all those who helped in various stages. First I would like to express my sincere gratitude indebttness to **Mr. Manoj Malik** (HOD, Department of Information Technology) & **Mr. Satender Malik** (Proctor of IT Evening Shift) for allowing me to undergo the summer training of 6 weeks at OSAHUB, MSIT.

I am grateful to our guide **Mr. Ashutosh Singh** for the help provided in completion of the project, which was assigned to me. Whithout his friendly help and guidance it was difficult to develop this project.

Last but not least, I pay my sincere thanks and gratitude to all the staff members of **OSAHUB** for their support and for making our training valuable and fruitful.

## **List of figures**

<b>Fig 1.1 OSA Technologies.....</b>	<b>3</b>
<b>Fig 1.2 OSA Infosystems.....</b>	<b>3</b>
<b>Fig 1.3 OSA BOTS.....</b>	<b>3</b>
<b>Fig 2.1 Numpy Array.....</b>	<b>10</b>
<b>Fig 3.1 Machine Learning Process.....</b>	<b>18</b>
<b>Fig 3.2 Iterative Machine Learning Process.....</b>	<b>19</b>
<b>Fig 3.3 (a) Support Vector Machine.....</b>	<b>20</b>
<b>Fig 3.3 (b) Support Vector Machine.....</b>	<b>21</b>
<b>Fig 3.4 Logistic Regression .....</b>	<b>22</b>
<b>Fig 4.1 Output.....</b>	<b>23</b>
<b>Fig 4.2 Output.....</b>	<b>24</b>
<b>Fig 4.3 Output.....</b>	<b>25</b>
<b>Fig 4.4 Output.....</b>	<b>25</b>
<b>Fig 4.5 Level 0 Data Flow Diagram.....</b>	<b>26</b>
<b>Fig 4.6 Level 1 Data Flow Diagram.....</b>	<b>27</b>

## Table of Content

<b>Title Page.....</b>	<b>i</b>
<b>Certificate.....</b>	<b>ii</b>
<b>Feedback Form.....</b>	<b>iii</b>
<b>Candidate’s Declaration.....</b>	<b>v</b>
<b>Acknowledgment.....</b>	<b>vi</b>
<b>List of Figures.....</b>	<b>vii</b>
<b>Abstract of Project.....</b>	<b>1</b>
<b>Chapter 1: Company Profile.....</b>	<b>2</b>
<b>Chapter 2: Technology Used.....</b>	<b>4</b>
<b>2.1 Python.....</b>	<b>4</b>
<b>2.2 Jupyter Notebook .....</b>	<b>7</b>
<b>2.3 Numpy .....</b>	<b>10</b>
<b>2.4 Pandas .....</b>	<b>12</b>
<b>2.5 SciKit-learn.....</b>	<b>13</b>
<b>2.6 Software Discription.....</b>	<b>14</b>
<b>Chapter 3: Demonstration of Technology.....</b>	<b>16</b>
<b>3.1 About The Project.....</b>	<b>16</b>
<b>3.2 Steps Followed.....</b>	<b>16</b>
<b>3.3 Process Involved.....</b>	<b>18</b>
<b>3.4 Software Development Life Cycle.....</b>	<b>18</b>
<b>3.5 Concepts Involved.....</b>	<b>19</b>
<b>3.6 Models Used.....</b>	<b>20</b>



<b>Chapter 4: Screenshots of project and DFDs.....</b>	<b>23</b>
<b>4.1 Screenshots of project.....</b>	<b>23</b>
<b>4.2 Data Flow Diagrams.....</b>	<b>28</b>
<b>Chapter 5: Conclusion.....</b>	<b>28</b>
<b>References.....</b>	<b>29</b>
<b>Appendices.....</b>	<b>32</b>

## **Abstract of Project**

This project is based on Natural Language Processing on textual data. The project classifies news data into 'possibly fake' and 'authentic' by creating a sparse matrix and creating a 'bag-of-words' model

The **bag-of-words model** is a simplifying representation used in natural language processing and information retrieval (IR). Also known as the vector space model. In this model, a text (such as a sentence or a document) is represented as the bag (multiset) of its words, disregarding grammar and even word order but keeping multiplicity. The bag-of-words model has also been used for computer vision. The bag-of-words model is commonly used in methods of document classification where the (frequency of) occurrence of each word is used as a feature for training a classifier.

The classification is done using multiple commonly used classification algorithms in machine learning, out of which

The model predicts the authenticity of any news on the basis of mapping the input news text with the 'bag of words' created by training a machine learning model, making a matrix of more than 16000 columns, on a dataset having labels specified to each corresponding news as 'fake' or 'authentic'.

## Chapter 1 : Company Profile



OSAHUB Technologies is India's leading training provider. We focus on education solutions, covering all the latest technologies - be it in the field of **Computer Science, Electronics, Mechanical Engineering, Robotics, Animation, etc.** We are Ranked among Top Training Institutions / Education Centers and Graded among India's Most Trusted Service Brands. We provide Workshops, Trainings, Faculty Development Programs, Certifications Courses, Seminars, Career Counseling, etc. to engineering colleges and schools all across India. We have an extremely skilled network of subject matter experts and highly experienced trainers who have several years of experience in the industry. We are pioneers in the education industry and are proud of our excellent team.

Our vision is to act as a mediator between Technology and Students to improve their academic performance by providing non-syllabus inputs and Best Trainings on various Emerging Technologies. Our vision is to empower youth through high quality and dedicated education.

## Various Domains of OSAHUB:



Fig 1.1 OSA Technologies

## OSA TECHNOLOGIES

Bringing the latest technologies to all school and college students



Fig 1.2 OSA Infosystems

## OSA INFOSYSTEMS

A one stop solution for all the software requirements of the industry



Fig 1.3 OSA Bots

## OSA BOTS

Because we all know just how cool Robots can be

## OSA START-UP

Nurturing and supporting startups along every step on the way.

## Chapter 2: Technology Used

### 2.1 PYTHON



Python is one of those rare languages which can claim to be both **simple** and **powerful**. You will be pleasantly surprised to see how easy it is to concentrate on the solution to the problem rather than on the syntax (i.e. the structure of the program that you are writing) of the language.

The official Python introduction is

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

I will discuss these features in more detail in the next section.

By the way, Guido van Rossum (the creator of the Python language) named the language after the BBC show "Monty Python's Flying Circus". He doesn't particularly like snakes that kill animals for food by winding their long bodies around them and crushing them.

## **Features of Python**

### **Simple**

Python is a simple and minimalistic language. Reading a good Python program feels almost like reading English (but very strict English!). This pseudo-code nature of Python is one of its greatest strengths. It allows you to concentrate on the solution to the problem rather than the syntax i.e. the language itself.

### **Easy to Learn**

As you will see, Python is extremely easy to get started with. Python has an extraordinarily simple syntax as already mentioned.

### **Free and Open Source**

Python is an example of a FLOSS (Free/Libre and Open Source Software). In simple terms, you can freely distribute copies of this software, read the software's source code, make changes to it, use pieces of it in new free programs, and that you know you can do these things. FLOSS is based on the concept of a community which shares knowledge. This is one of the reasons why Python is so good - it has been created and improved by a community who just want to see a better Python.

### **High-level Language**

When you write programs in Python, you never need to bother about low-level details such as managing the memory used by your program.

### **Portable**

Due to its open-source nature, Python has been ported (i.e. changed to make it work on) to many many platforms. All your Python programs will work on any of these platforms without requiring any changes at all. However, you must be careful enough to avoid any system-dependent features.

You can use Python on Linux, Windows, Macintosh, Solaris, OS/2, Amiga, AROS, AS/400, BeOS, OS/390, z/OS, Palm OS, QNX, VMS, Psion, Acorn RISC OS, VxWorks, PlayStation, Sharp Zaurus, Windows CE and PocketPC !

## **Interpreted**

This requires a little explanation.

A program written in a compiled language like C or C++ is translated from the source language i.e. C/C++ into a language spoken by your computer (binary code i.e. 0s and 1s) using a compiler with various flags and options. When you run the program, the linker/loader software just stores the binary code in the computer's memory and starts executing from the first instruction in the program.

When you use an interpreted language like Python, there is no separate compilation and execution steps. You just *run* the program from the source code. Internally, Python converts the source code into an intermediate form called bytecodes and then translates this into the native language of your specific computer and then runs it. All this makes using Python so much easier. You just *run* your programs - you never have to worry about linking and loading with libraries, etc. They are also more portable this way because you can just copy your Python program into another system of any kind and it just works!

## **Object Oriented**

Python supports procedure-oriented programming as well as object-oriented programming. In *procedure-oriented* languages, the program is built around procedures or functions which are nothing but reusable pieces of programs. In *object-oriented* languages, the program is built around objects which combine data and functionality. Python has a very powerful but simple way of doing object-oriented programming, especially, when compared to languages like C++ or Java.

## **Extensible**

If you need a critical piece of code to run very fast, you can achieve this by writing that piece of code in C, and then combine that with your Python program.

## **Embeddable**

You can embed Python within your C/C++ program to give scripting capabilities for your program's users.

## **Extensive Libraries**

The Python Standard Library is huge indeed. It can help you do various things involving regular expressions, documentation generation, unit testing, threading, databases, web browsers, CGI, ftp, email, XML, XML-RPC, HTML, WAV files, cryptography, GUI(graphical user interfaces) using Tk, and also other system-dependent stuff. Remember, all this is always available wherever Python is installed. This is called the "batteries included" philosophy of Python.

Besides the standard library, there are various other high-quality libraries such as the [Python Imaging Library](#) which is an amazingly simple image manipulation library.

## **2.2 JUPYTER NOTEBOOK**



In this case, "notebook" or "notebook documents" denote documents that contain both code and rich text elements, such as figures, links, equations, ... Because of the mix of code and text elements, these documents are the ideal place to bring together an analysis description and its results as well as they can be executed perform the data analysis in real time.

These documents are produced by the Jupyter Notebook App.



For now, you should just know that "Jupyter" is a loose acronym meaning Julia, Python, and R. These programming languages were the first target languages of the Jupyter application, but nowadays, the notebook technology also supports many other languages.

As you just saw, the main components of the whole environment are, on the one hand, the notebooks themselves and the application. On the other hand, you also have a notebook kernel and a notebook dashboard.

Let's look at these components in more detail.

### **2.2.1 Jupyter Notebook App**

As a server-client application, the Jupyter Notebook App allows you to edit and run your notebooks via a web browser. The application can be executed on a PC without Internet access or it can be installed on a remote server, where you can access it through the Internet.

Its two main components are the kernels and a dashboard.

A kernel is a program that runs and introspects the user's code. The Jupyter Notebook App has a kernel for Python code, but there are also kernels available for other programming languages.

The dashboard of the application not only shows you the notebook documents that you have made and can reopen but can also be used to manage the kernels: you can which ones are running and shut them down if necessary.

### **2.2.1 History of IPython and Jupyter Notebooks**

To fully understand what the Jupyter Notebook is and what functionality it has to offer you need to know how it originated.

Let's back up briefly to the late 1980s. Guido Van Rossum begins to work on Python at the National Research Institute for Mathematics and Computer Science in the Netherlands..

Let's go to late 2001, twenty years later. Fernando Pérez starts developing IPython.

In 2005, both Robert Kern and Fernando Pérez attempted building a notebook system. Unfortunately, the prototype had never become fully usable.

Fast forward two years: the IPython team had kept on working, and in 2007, they formulated another attempt at implementing a notebook-type system. By October 2010, there was a prototype of a web notebook and in the summer of 2011, this prototype was incorporated and it was released with 0.12 on December 21, 2011. In subsequent years, the team got awards, such as the Advancement of Free Software for Fernando Pérez on 23 of March 2013 and the Jolt Productivity Award, and funding from the Alfred P. Sloan Foundations, among others.

Lastly, in 2014, Project Jupyter started as a spin-off project from IPython. IPython is now the name of the Python backend, which is also known as the kernel. Recently, the next generation of Jupyter Notebooks has been introduced to the community. It's called JupyterLab. Read more about it [here](#). After all this, you might wonder where this idea of notebooks originated or how it came about to the creators.

A brief research into the history of these notebooks learns that Fernando Pérez and Robert Kern were working on a notebook just at the same time as the Sage notebook was a work in progress. Since the layout of the Sage notebook was based on the layout of Google notebooks, you can also conclude that also Google used to have a notebook feature around that time.

For what concerns the idea of the notebook, it seems that Fernando Pérez, as well as William Stein, one of the creators of the Sage notebook, have confirmed that they were avid users of the Mathematica notebooks and Maple worksheets. The Mathematica notebooks were created as a front end or GUI in 1988 by Theodore Gray.

The concept of a notebook, which contains ordinary text and calculation and/or graphics, was definitely not new.

## 2.3 NUMPY



NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. This tutorial explains the basics of NumPy such as its architecture and environment. It also discusses the various array functions, types of indexing, etc. An introduction to Matplotlib is also provided. All this is explained with the help of examples for better understanding.

**Numeric**, the ancestor of NumPy, was developed by Jim Hugunin. Another package Numarray was also developed, having some additional functionalities. In 2005, Travis Oliphant created NumPy package by incorporating the features of Numarray into Numeric package. There are many contributors to this open source project.

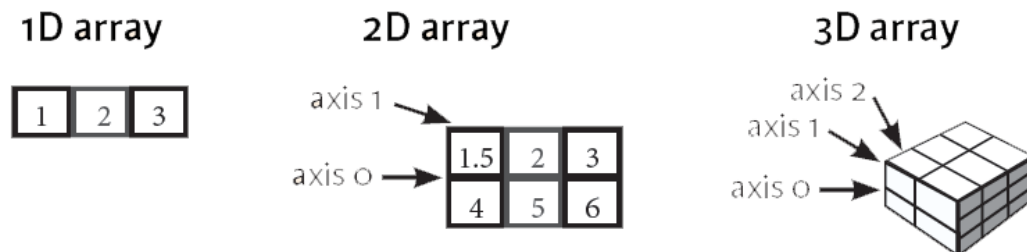


Fig. 2.1 Numpy Array

## 2.4 PANDAS



Pandas is a library that unifies the most common workflows that data analysts and data scientists previously relied on many different libraries for. Pandas has quickly become an important tool in a data professional's toolbelt and is the most popular library for working with tabular data in Python. Tabular data is any data that can be represented as rows and columns. The CSV files we've worked with in previous missions are all examples of tabular data.

To represent tabular data, pandas uses a custom data structure called a **dataframe**. A dataframe is a highly efficient, 2-dimensional data structure that provides a suite of methods and attributes to quickly explore, analyze, and visualize data. The dataframe is similar to the NumPy 2D array but adds support for many features that help you work with tabular data.

One of the biggest advantages that pandas has over NumPy is the ability to store mixed data types in rows and columns. Many tabular datasets contain a range of data types and pandas dataframes handle mixed data types effortlessly while NumPy doesn't. Pandas dataframes can also handle missing values gracefully using a custom object, `NaN`, to represent those values. A common complaint with NumPy is its lack of an object to represent missing values and people end up having to find and replace these values manually. In addition, pandas dataframes contain axis labels for both rows and columns and enable you to refer to elements in the dataframe more intuitively. Since many tabular datasets contain column titles, this means that dataframes preserve the metadata from the file around the data.

## 2.5 SciKit-Learn



Machine Learning with Scikit-Learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python.

It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

The library is built upon the SciPy (Scientific Python) that must be installed before you can use scikit-learn. This stack that includes:

- **NumPy**: Base n-dimensional array package
- **SciPy**: Fundamental library for scientific computing
- **Matplotlib**: Comprehensive 2D/3D plotting
- **IPython**: Enhanced interactive console
- **Sympy**: Symbolic mathematics
- **Pandas**: Data structures and analysis

Extensions or modules for SciPy are conventionally named SciKits. As such, the module provides learning algorithms and is named scikit-learn.

The vision for the library is a level of robustness and support required for use in production systems. This means a deep focus on concerns such as easy of use, code quality, collaboration, documentation and performance.

Although the interface is Python, c-libraries are leverage for performance such as numpy for arrays and matrix operations, LAPACK, LibSVM and the careful use of cython.

Take my free 2-week email course and discover data prep, algorithms and more (with code).

The library is focused on modeling data. It is not focused on loading, manipulating and summarizing data. For these features, refer to NumPy and Pandas.

Screenshot taken from a demo of the mean-shift clustering algorithm

Some popular groups of models provided by scikit-learn include:

- **Clustering:** for grouping unlabeled data such as KMeans.
- **Cross Validation:** for estimating the performance of supervised models on unseen data.
- **Datasets:** for test datasets and for generating datasets with specific properties for investigating model behavior.
- **Dimensionality Reduction:** for reducing the number of attributes in data for summarization, visualization and feature selection such as Principal component analysis.
- **Ensemble methods:** for combining the predictions of multiple supervised models.
- **Feature extraction:** for defining attributes in image and text data.
- **Feature selection:** for identifying meaningful attributes from which to create supervised models.
- **Parameter Tuning:** for getting the most out of supervised models.

**Supervised Models:** a vast array not limited to generalized linear models, discriminate analysis, naive bayes, lazy methods, neural networks, support vector machines and decision trees.

## 2.6 Software Description

**Anaconda** is a free and open source distribution of the Python and R programming languages for data science and machine learning related applications (large-scale data processing, predictive analytics, scientific computing), that aims to simplify package management and deployment. Package versions are managed by the package management system *conda*. The Anaconda distribution is used by over 6 million users, and it includes more than 250 popular data science packages suitable for Windows, Linux, and MacOS.

**Anaconda distribution** comes with more than 1,000 data packages as well as the Conda package and virtual environment manager, called **Anaconda Navigator**, so it eliminates the need to learn to install each library independently.



The open source data packages can be individually installed from the Anaconda repository <sup>[8]</sup> with the **conda install** command or using the **pip install** command that is installed with Anaconda. Pip packages provide many of the features of conda packages and in most cases they can work together.

You can also make your own custom packages using the **conda build** command, and you can share them with others by uploading them to Anaconda Cloud.

The default installation of Anaconda2 includes Python 2.7 and Anaconda3 includes Python 3.6. However, you can create new environments that include any version of Python packaged with conda <sup>[10]</sup>.

### **Anaconda Navigator**

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage conda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for Windows, macOS and Linux.

Navigator is automatically included with Anaconda version 4.0.0 or higher.

The following applications are available by default in Navigator :

- JupyterLab
- Jupyter Notebook
- QtConsole
- Spyder
- Glueviz
- Orange
- Rstudio
- Visual Studio Code

## **Conda**

Conda is an open source, cross-platform, language-agnostic package manager and environment management system that installs, runs, and updates packages and their dependencies. It was created for Python programs, but it can package and distribute software for any language (e.g., R), including multi-language projects.<sup>[14]</sup> The Conda package and environment manager is included in all versions of Anaconda, Miniconda, and Anaconda Repository.



## **Chapter 3: Demonstration of Technology**

### **3.1 About the project**

The Fake News Detector predicts the authenticity of any news on the basis of mapping the input news text with the 'bag of words' created by training a machine learning model, making a matrix of more than 16000 columns, on a dataset having labels specified to each corresponding news as 'fake' or 'authentic'.

The detector can be used to predict authenticity of any news preferably originating from the United States as the machine learning model is treated primarily on the U.S dataset.

### **3.2 Steps followed**

1. Understanding the problem
2. Requirement Analysis
3. Getting the required data
4. Data Exploration
5. Data Pre-Processing
6. Feature Engineering
7. Model Training
8. Predicting the inputs-

#### **1. Understand the problem:**

Before getting the data, we need to understand the problem we are trying to solve. If you know the domain, think of which factors could play an epic role in solving the problem. If you don't know the domain, read about it.

#### **2. Requirement Analysis:**

Making a comprehensive description of the intended purpose and environment for software under development. The SRS fully describes what the software will do and how it will be expected to perform

### **3. Get Data:**

Now, we download the data and look at it. Determine which features are available and which aren't, how many features we generated in hypothesis generation hit the mark, and which ones could be created. Answering these questions will set us on the right track.

### **4. Data Exploration:**

We can't determine everything by just looking at the data. We need to dig deeper. This step helps us understand the nature of variables (skewed, missing, zero variance feature) so that they can be treated properly. It involves creating charts, graphs (univariate and bivariate analysis), and cross-tables to understand the behaviour of features.

### **5. Data Pre-processing:**

Here, we impute missing values and clean string variables (remove space, irregular tabs, data time format) and anything that shouldn't be there. This step is usually followed along with the data exploration stage.

### **6. Feature Engineering:**

Now, we create and add new features to the data set. Most of the ideas for these features come during the hypothesis generation stage

### **7. Model Training:**

Using a suitable algorithm, we train the model on the given data set.

### **8. Model Evaluation:**

Once the model is trained, we evaluate the model's performance using a suitable error metric. Here, we also look for variable importance, i.e., which variables have proved to be significant in determining the target variable. And, accordingly we can shortlist the best variables and train the model again.

**9. Model Testing:** Finally, we test the model on the unseen data (test data) set.

#### **3.2.1 .Understand the problem**

The data set for this project has been taken from Kaggle's Fake News dataset. This project aims at predicting the authenticity of news primarily of origin from the USA. I believe this problem statement is quite self-explanatory and doesn't need more explanation. Hence, we move to the next step.

### 3.3 Processes Involved

#### The Machine Learning Process



Fig.3.1 Machine Learning Process

### 3.4 Software Development Life Cycle (SDLC)

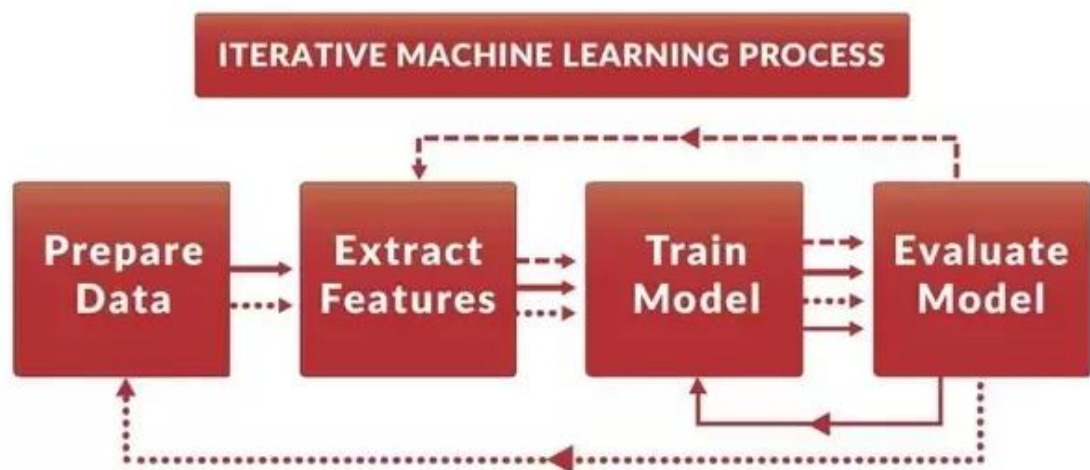


Fig 3.2 Iterative Machine Learning Process

## 3.5 Concepts Involved

### 3.5.1 CountVectorizer:

The CountVectorizer provides a simple way to both tokenize a collection of text documents and build a vocabulary of known words, but also to encode new documents using that vocabulary.

1. Create an instance of the *CountVectorizer* class.
2. Call the *fit()* function in order to learn a vocabulary from one or more documents.
3. Call the *transform()* function on one or more documents as needed to encode each as a vector.

An encoded vector is returned with a length of the entire vocabulary and an integer count for the number of times each word appeared in the document.

Because these vectors will contain a lot of zeros, we call them sparse. Python provides an efficient way of handling sparse vectors in the [scipy.sparse](#) package.

The vectors returned from a call to *transform()* will be sparse vectors, and you can transform them back to numpy arrays to look and better understand what is going on by calling the *toarray()* function.

Below is an example of using the CountVectorizer to tokenize, build a vocabulary, and then encode a document.

### 3.5.2 tfidfVectorizer

Tf idf is different from countvectorizer. Countvectorizer gives equal weightage to all the words, i.e. a word is converted to a column (in a dataframe for example) and for *each* document, it is equal to **1** if it is present in that doc else **0**. Apart from giving this information, tfidf says how important that word is to that document with respect to the corpus.

Tf idf has two parts :

#### **Term Frequency (Tf):**

How many times a particular word appears in a single doc. To understand it

better lets take a word : “the”

This word is quite common and would appear with high frequency in all our docs. But if we think about it, “the” does not give any extra info about my doc. So, we need to reduce the weightage of word “the”. For that we use idf.

#### **Inverse Document Frequency (idf):**

The problem of frequent and rare words is solved by adding this term.

It is calculated by taking the **log** of {number of docs in your corpus divided by the number of docs in which this term appears}.

So for “the”, the ratio will be close to 1. Therefore, the log will take this to 0.

### **3.6 Models Used**

#### **3.6.1 Support Vector Machine (SVM)**

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (*supervised learning*), the algorithm outputs an optimal hyperplane which categorizes new examples. In two dimensional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side.

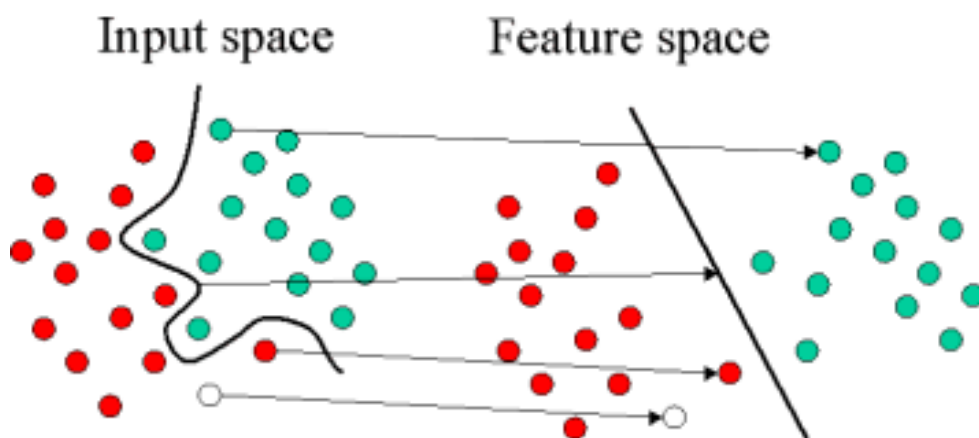


Fig 3.3 (a) Support Vector Machine

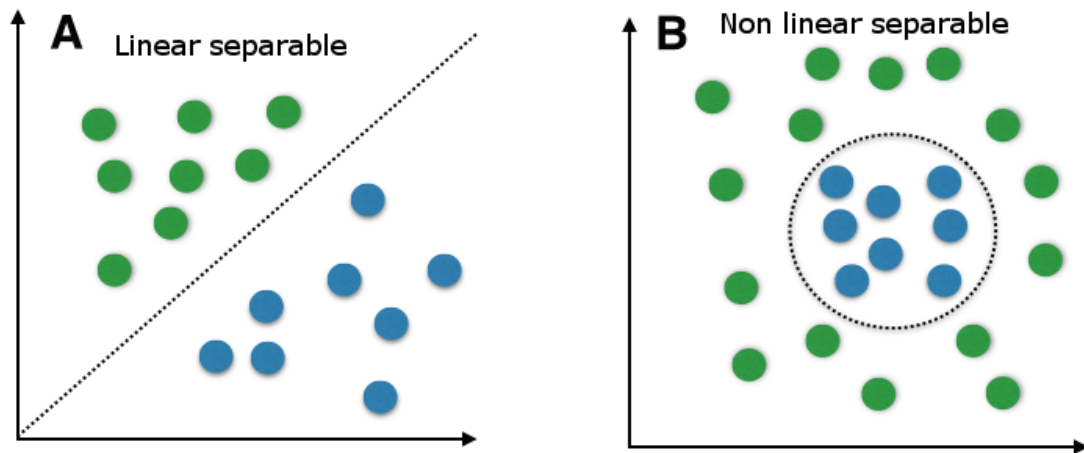


Fig 3.3 (b) Support Vector Machine

### 3.6.2 Logistic Regression

Logistic Regression is the appropriate regression analysis to conduct when the dependent variable is dichotomous (binary). Like all regression analyses, the logistic regression is a predictive analysis. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables

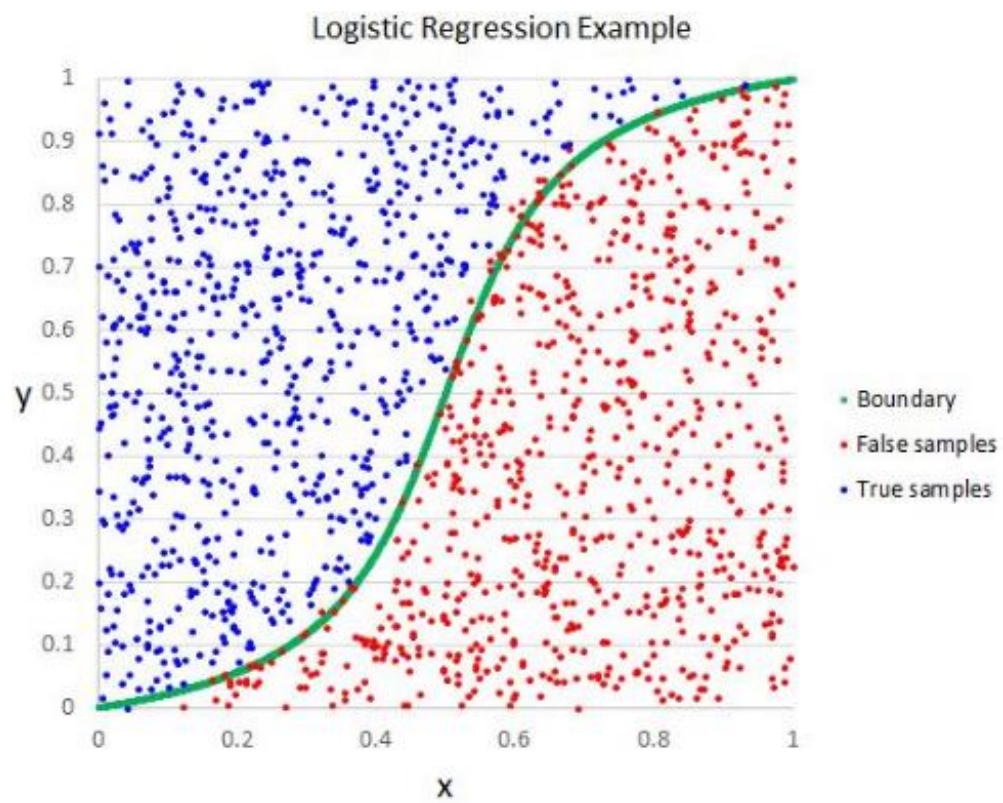


Fig 3.4 Logistic Regression

## Chapter 4: Screenshot of the project and DFD

### 4.1 Screenshots of the project

```
Enter the News

Russian warships ready to strike terrorists near Aleppo 08.11.2016 ! Source: Sou
rce: Mil.ru Attack aircraft of the Russian aircraft carrier Admiral Kuznetsov ge
t ready to strike terrorists' positions in the vicinity of Aleppo, sources at th
e Russian Defense Ministry said, RBC reports. "Insurgents' attempts to break int
o Aleppo from outside are meaningless," the source said. The main task of the ai
rcraft carrier aviation group is to strike missile and air blows on the terroris
ts , whose goal is to enter Aleppo. "After the attacks on terrorists' positions,
one will have to forget about the support for insurgents from the outside," the
source said. The Russian group in the Mediterranean Sea consists of the Admiral
Kuznetsov aircraft carrier , the heavy nuclear missile cruiser Pyotr Velikiy (P
eter the Great) and large anti-submarine ships Severomorsk and Vice-Admiral Kula
kov. Russia has increased intelligence activities in Syria to establish the area
s, where terrorists are concentrated, as well as the routes that they use to mov
e from one area to another. "The militants took advantage of the humanitarian pa
use and regrouped their forces to prepare for a new breakthrough into the easter
n part of Aleppo," the source added. According to the source, Russia will use ne
w weapons during the upcoming attacks on terrorists . It was said that the Russi
an warships in the Mediterranean Sea will launch "Caliber" cruise missiles, alth
ough it was not specified which ships would be responsible for the launches. Pra
vda.Ru Russian warships travel to Syria

Don't worry, news is a true one!!

Press a button to close
```

Fig 4.1 Output



ous relationship comedy that also takes its core sadness seriously. In this sense, it aims to be less like *Sex and the City* than like HBO's naturalistic *Girls* (whose producer Paul Simms serves as a showrunner). The difference, of course, is measured in years. *Divorce* feels in its bones, from the themes to the wintry setting to the soundtrack (Supertramp, Todd Rundgren, Climax Blues Band). It's not going to reinvent the breakup comedy or the HBO comedy. Its goal is more modest and : to tell one more story of two people trying to reinvent themselves.

Warning: News maybe a fake one!!

Press a button to close

(base) d:\delete>python fake\_news\_main.py

Enter the News

LOS ANGELES When Donald J. Trump claimed on Twitter that he was losing the popular vote because of major fraud by millions of voters, one of the states he pointed to was California, where the latest voting returns showed Hillary Clinton, the Democrat, crushing Mr. Trump. But Mr. Trump's baseless claim led to a furious reaction from California's top election official, Alex Padilla, the secretary of state, this weekend. Mr. Padilla asserted that there was no evidence for the claim by the and denounced Mr. Trump for what he said was unpresidential behavior. His unsubstantiated allegations of voter fraud in California and elsewhere are absurd, Mr. Padilla posted on Twitter. His reckless tweets are inappropriate and unbecoming of a . This state has historically been slow to count ballots, a reflection of both its vast size and inefficiencies in many county voting operations. Given the fact that this is an overwhelmingly Democratic state, that has meant that Mrs. Clinton's total vote count has grown steadily as ballots were tallied, adding to a national lead of close to two million votes. As of Saturday, Mrs. Clinton had 8.1 million votes in California, compared with 4.2 million for Mr. Trump, according to the secretary of state's office. It was not clear when the vote count might be concluded. The count, and outcome, has been no surprise to anyone in a state with a history of slow . Officials in both parties had predicted this would happen as early as election night. Given California's long Democratic history, it was never in play during this presidential election. But as the nation's most populated state, it tends to have a significant influence on final national voting margins, as is apparently the case this year. Mr. Trump signaled out three states in his post of Twitter on serious voter fraud: Virginia, New Hampshire and California. He offered no evidence to back up the claim. His remarks came as he denounced calls for a recount in three states that he won by relatively small margins: Wisconsin, Michigan and Pennsylvania. Jill Stein, who was the Green Party candidate for president, said that she would move for a recount. Aides to Mrs. Clinton said they would cooperate with the effort, even as they made it clear they thought it would not change the outcome. Mr. Padilla is the Latino elected to state office in California. Mr. Trump's poor showing here, many Democrats and Republicans said, came in no small part because of his attacks on what he described as the threat of illegal immigration particularly by Mexicans. About 40 percent of California's population is Latino. It appears that Mr. Trump is troubled by the fact that a growing majority of Americans did not vote for him, Mr. Padilla said.

Warning: News maybe a fake one!!

Fig 4.2 Output

```

Enter the News

If at first you don't succeed, try a different sport. Tim Tebow, who was a Heisman
quarterback at the University of Florida but was unable to hold an N. F.
L. job, is pursuing a career in Major League Baseball. He will hold a workout for
M. L. B. teams this month, his agents told ESPN and other news outlets. This
may sound like a publicity stunt, but nothing could be further from the truth.
said Brodie Van Wagenen, of CAA Baseball, part of the sports agency CAA Sports,
in the statement. I have seen Tim's workouts, and people inside and outside the
industry scouts, executives, players and fans will be impressed by his talent.
It's been over a decade since Tebow, 28, has played baseball full time, which means
a comeback would be no easy task. But the former major league catcher Chad Moeller,
who said in the statement that he had been training Tebow in Arizona, said he was
beyond impressed with Tebow's athleticism and swing. I see bat speed and power and
real baseball talent. Moeller said. I truly believe Tim has the skill set and potential
to achieve his goal of playing in the major leagues and based on what I have seen
over the past two months, it could happen relatively quickly. Or, take it from Gary
Sheffield, the former outfielder. News of Tebow's attempted comeback in baseball
was greeted with skepticism on Twitter. As a junior at Nease High in Ponte Vedra,
Fla. Tebow drew the attention of major league scouts, batting .494 with four home
runs as a left fielder. But he ditched the bat and glove in favor of pigskin,
leading Florida to two national championships, in 2007 and 2009. Two former
scouts for the Los Angeles Angels told WEEI, a Boston radio station, that Tebow
had been under consideration as a high school junior. I'd love to draft him, but
he never sent back his information card, said one of the scouts, Tom Kotchman,
referring to a questionnaire the team had sent him. He had a strong arm and had a
lot of power, said the other scout, Stephen Hargett. If he would have been there
his senior year he definitely would have had a good chance to be drafted. It
was just easy for him, Hargett added. You thought, If this guy dedicated everything
to baseball like he did to football how good could he be? Tebow's high school
baseball coach, Greg Mullins, told The Sporting News in 2013 that he believed
Tebow could have made the major leagues. He was the leader of the team with his
passion, his fire and his energy, Mullins said. He loved to play baseball, too.
He just had a bigger fire for football. Tebow wouldn't be the first athlete to
switch from the N. F. L. to M. L. B. Bo Jackson had one season as a Kansas City
Royal, and Deion Sanders played several years for the Atlanta Braves with mixed
success. Though Michael Jordan tried to cross over to baseball from basketball
as a minor league team. As a football player, Tebow was unable to match his
college success in the pros. The Denver Broncos drafted him in the first round
of the 2010 N. F. L. Draft, and he quickly developed a reputation for clutch
performances, including a memorable pass against the Pittsburgh Steelers in the
2011 Wild Card round. But his stats and his passing form weren't pretty, and he
spent just two years in Denver before moving to the Jets in 2012, where he spent
his last season on an N. F. L. roster. He was cut during preseason from the New
England Patriots in 2013 and from the Philadelphia Eagles in 2015.

Warning: News maybe a fake one!!

Press a button to close_

```

Fig 4.3 Output

```

(base) d:\delete>python fake_news_main.py

Enter the News

President Obama and President-Elect Donald Trump Meet at White House: Share:

Don't worry, news is a true one!!

Press a button to close

```

Fig 4.4 Output

## 4.2 Data Flow Diagram (DFD)

- Level 0 DFD

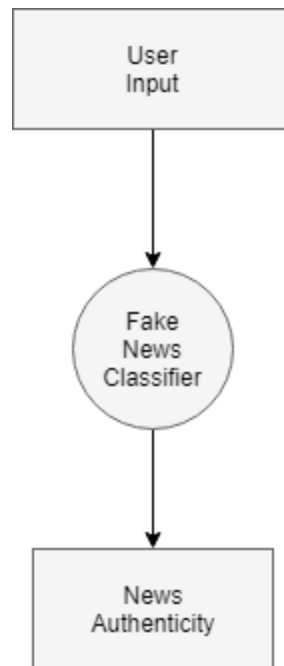
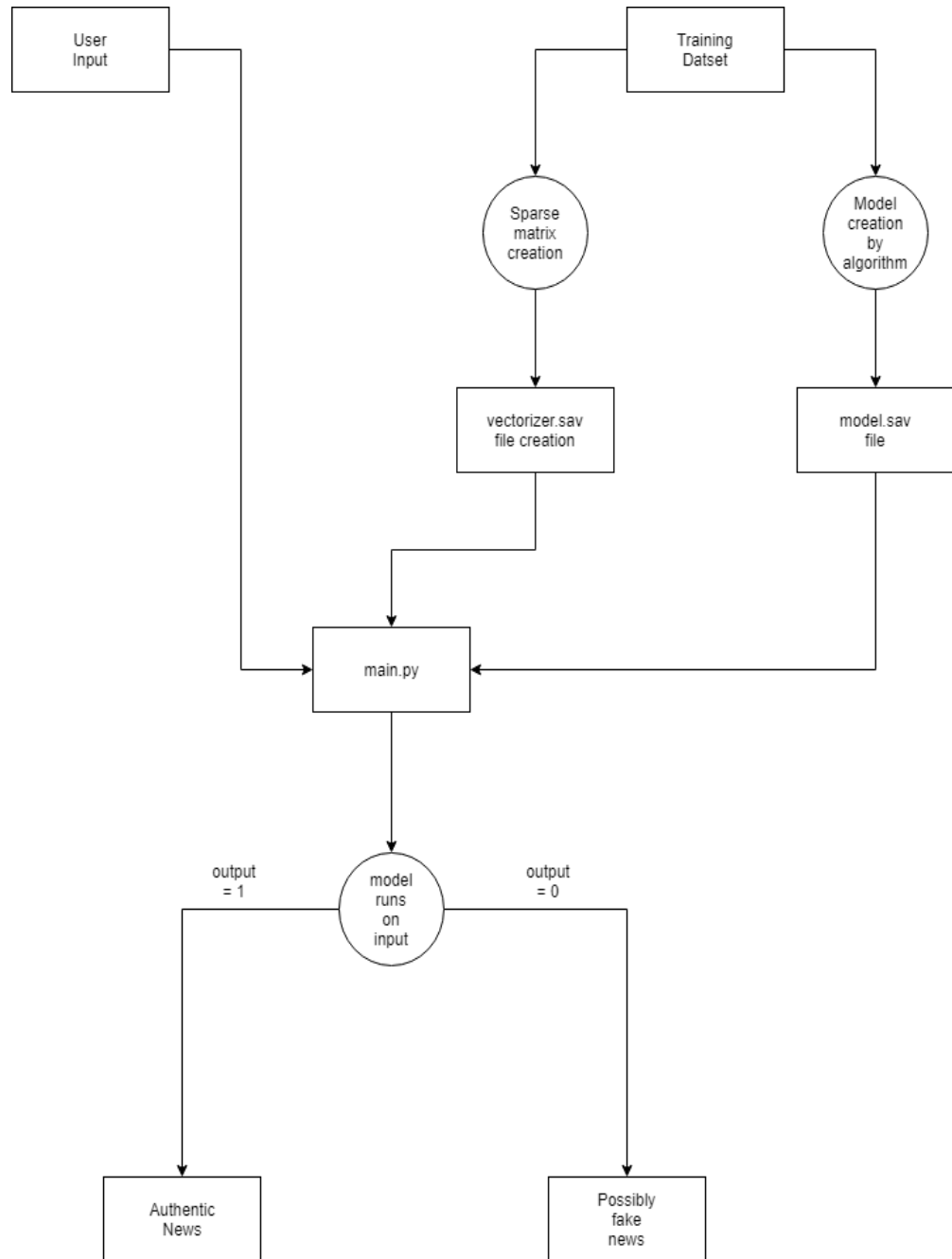


Fig 4.5 Level 0 Data Flow Diagram

- Level 1 DFD



Text

Fig 4.6 Level 1 Data Flow Diagram

## **Chapter: 5 Conclusion**

The Fake News Detector predicts the authenticity of any news on the basis of mapping the input news text with the 'bag of words' created by training a machine learning model, making a matrix of more than 16000 columns, on a dataset having labels specified to each corresponding news as 'fake' or 'authentic'.

The detector can be used to predict authenticity of any news preferably originating from the United States as the machine learning model is trained primarily on the U.S dataset.

**The model is reliable with an accuracy of 96%.**

## REFERENCES

1. [https://github.com/avs20/MSIT\\_ML\\_CLASS](https://github.com/avs20/MSIT_ML_CLASS)
2. [https://github.com/IISourcecell/Learn\\_Machine\\_Learning\\_in\\_3\\_Months](https://github.com/IISourcecell/Learn_Machine_Learning_in_3_Months) by Siraj Raval
3. <http://www.andrewng.org/courses/> by Andrew Angie
4. [Introduction to python programming by Udacity](#)
5. [Python courses on datacamp.com](#)
6. [Harvard's CS50 Course](#)
7. [Introduction to Statistical Learning](#) by James, Hastie et. Al
8. <https://www.kaggle.com/c/fake-news/data>
9. <https://www.kaggle.com/plarmuseau/minimalistic-logistic-ngram-tfidf-lb-0-975>
10. <https://www.kaggle.com/plarmuseau/minimalistic-nb-ngram>
11. <https://www.kaggle.com/plarmuseau/fork-of-minimalistic>
12. [Kaggle Tutorial EDA & Machine Learning \(Datacamp-Blog Post\)](#)
13. [Exploratory Data Analysis in Advanced ML Specialisation \(Coursera—Online Course\)](#)
14. [Machine Learning with Kaggle: Feature Engineering \(Datacamp—Blog Post\)](#)
15. [Feature Engineering for Continuous Numeric Data \(Towards Data Science—Blog Post\)](#)
16. [Feature Engineering for Categorical Data \(Towards Data Science—Blog Post\)](#)
17. [Feature Engineering for Text Data—Traditional Methods \(Towards Data Science—Blog Post\)](#)
18. [Feature Engineering for Text Data—Deep Learning Methods \(Towards Data Science—Blog Post\)](#)

19. [Prepare Text Data for Machine Learning \(Machine Learning Mastery—Blog Post\)](#)
20. [Feature Selection \(Machine Learning Mastery—Blog Post\)](#)
21. [An example kernel of Text Data Processing for Kaggle Mercari Competition \(Kaggle—Kernel\)](#)
22. [An example kernel of Feature Engineering for House Prices: Advanced Regression Techniques Competition \(Kaggle—Kernel\)](#)
23. [Advanced Feature Engineering Part 1 and Part 2 in Advanced ML Specialisation \(Coursera—Online Course\)](#)
24. [Complete Machine Learning Project Walkthrough Part 2 \(Towards Data Science—Blog Post\)](#)
25. [Exploratory Study on ML algorithms \(Kaggle—Kernel\)](#)
26. [Comparing ML algorithms \(Machine Learning Mastery—Blog Post\)](#)
27. [Comparing various ML models \(Kaggle—Kernel\)](#)
28. [How to tune ML algorithm parameters \(Machine Learning Mastery—Blog Post\)](#)
29. [Choosing the right metric for ML Models Part 1 and Part 2](#)
30. [Metrics To Evaluate Machine Learning Algorithms \(Machine Learning Mastery—Blog Post\)](#)
31. [Cheat sheet for AI—Neural Networks—Machine Learning—Deep Learning—Big Data \(Becominghuman.ai\)](#)
32. [Pandas cheat sheet \(Datacamp\)](#)
33. [Python for Data Science \(Datacamp\)](#)
34. [Data Exploration in Python \(Analytics Vidhya\)](#)

35. [Data Visualisation in Python \(Analytics Vidhya\)](#)
36. [Scikit Learn cheat sheet \(Analytics Vidhya\)](#)



## 7. Appendices

**Artificial neural networks (ANN)** or **connectionist systems** are computing systems vaguely inspired by the biological neural networks that constitute animal brains. Such systems "learn" to perform tasks by considering examples, generally without being programmed with any task-specific rules. For example, in image recognition, they might learn to identify images that contain cats by analyzing example images that have been manually labeled as "cat" or "no cat" and using the results to identify cats in other images. They do this without any prior knowledge about cats, e.g., that they have fur, tails, whiskers and cat-like faces. Instead, they automatically generate identifying characteristics from the learning material that they process.

An ANN is based on a collection of connected units or nodes called artificial neurons which loosely model the neurons in a biological brain. Each connection can transmit a signal from one artificial neuron to another. An artificial neuron that receives a signal can process it and then signal additional artificial neurons connected to it.

In common ANN implementations, the signal at a connection between artificial neurons are a real number, and the output of each artificial neuron is computed by some non-linear function of the sum of its inputs. The connections between artificial neurons are called 'edges'. Artificial neurons and edges typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Artificial neurons may have a threshold such that the signal is only sent if the aggregate signal crosses that threshold. Typically, artificial neurons are aggregated into layers. Different layers may perform different kinds of transformations on their inputs. Signals travel from the first layer (the input layer), to the last layer (the output layer), possibly after traversing the layers multiple times.

The original goal of the ANN approach was to solve problems in the same way that a human brain would. However, over time, attention moved to performing specific tasks, leading to deviations from biology. Artificial neural networks have been used on a variety of tasks, including computer vision, speech recognition, machine translation, social network filtering, playing board and video games and medical diagnosis.