As per UGC guidelines an electronic bar code is provided to seure your paper

International Journal for Modern Trends in Science and Technology, 8(04):351-365, 2022

ISSN: 2455-3778 online
DOI: https://doi.org/10.46501/IJMTST0804060
Available online at: http://www.ijmtst.com/vol8issue04.html

A Systematic Review on Data Science Tools and Technologies

Karanam Poorna Siri[1] | Sanjay Venkat S[1] | Nivetha E[1] | Dr.Swagata Sarkar[2]

[1]Department of Artificial Intelligence and Data Science, Sri Sairam Engineering College, Chennai, India.,
[2]Head of the Department, Artificial Intelligence and Data Science, Sri Sairam Engineering College, Chennai, India.
Corresponding Author Email ID: sec20ad011@sairamtap.edu.in

To Cite this Article
Karanam Poorna Siri, Sanjay Venkat S, Nivetha E and Dr.Swagata Sarkar. A Systematic Review on Data Science Tools and Technologies. International Journal for Modern Trends in Science and Technology 2022, 8(04), pp. 351-365. https://doi.org/10.46501/IJMTST0804060

Article Info
Received: 17 March 2022; Accepted: 13 April 2022; Published: 18 April 2022.

ABSTRACT

Data Science is one of the advancing technologies in the modern world. Almost every human is familiar with the term 'data'. It is a huge blunder to speculate before analyzing data, unconsciously the facts are twisted to suit the speculations instead of theories to suit the facts. Data Science is the study of data, analyzing it, and deriving useful insights so as to predict with high accuracy. The data generated can be structured, semi-structured, unstructured, ordinal, numerical, categorical, or of any form. There are various steps involved before the data can be used for further predictions and modeling. However, data science involves various other fields like machine learning which is the most important element. It provides multiple algorithms to model the given data. There are various tools and technologies available for data manipulation that offers access to many operations on data. The paper gives a brief description of data pre-processing and the tools required for it.

KEYWORDS:Data Science, Data Preparation, Data, Database, Big data

1. INTRODUCTION

The digital world generates trillion MBs of data per second. This Big data is quite a lot for traditional computing systems to handle. Even before the prediction of a truly enormous amount of data generation over the next few years, Data Science showed up its existence. It lived through people's lives and developed into a whole new domain. It is an interdisciplinary study that uses machine learning algorithms, scientific methods, statistics, data analysis, and visualization to understand the data, extract insights, identify hidden patterns and make decisions or predictions pertaining to the future. Human intuition does not work on extensive data, for understanding probability. Data-driven, scientific approach to predictions and data analysis was introduced. Various terms were given to it including "Statistics", "Data Mining", "Predictive Analytics, "Data Analytics", "Knowledge Discovery in Data (KDD)" and "Data Science" among them. It involves many advanced tech concepts like (AI) Artificial Intelligence, (IoT) Internet of Things, Deep Learning to name a few. With the progress and technological developments of the modern world, data science's impact has increased drastically.

STRUCTURE OF PAPER

The construction of the paper is as follows: Section I provides a brief history of Data Science; next, Section II describes the Evolution of Data Science; Section III discusses the fundamentals of Big Data; next, Section IV

shows the methods of Data Preparation; later Section V provides source code for the Tools used in data Science; next, Section VI and VII gives an overview of data science, its application and future scope.

OBJECTIVES

This paper aims at giving a detailed description of the tools and technologies used by a Data Scientist. It is a beginner's guide to novice data scientists covering all details that a starter needs to know.

2. EVOLUTION OF DATA SCIENCE

Data Science may sound modern and emerging but it dates back to years ago. Horoscope is a way of predicting a person's future based on the position of stars and planets when he/she was born. Astronomy is the study of hidden patterns and relationships of planets in the universe, birth charts, synastry with others, the comprising of elements, and with that knowledge as a tool, the meaning is concluded. Astrologers examine the patterns and positions as they rise, culminate and set. Sherlock Holmes would have loved living in this century. We are drenched with data, so many of our problems including murder mysteries can be solved using massive data existing at personal and societal levels. The term "Data Science" was coined in the early 1960s. Astronomer Tobias Mayer, the first data scientist explained the motion of the moon. Big Data is so huge for existing computing systems to handle. With growing technology, data is being properly utilized.

3. BIG DATA

There are numerous image and pdf processing libraries that we can use to extract the raw text of our invoice from. We will discuss pdftotext, tesseract and tesseract4.
The modern world has created a huge amount of data that is stored in data warehouses. Data is classified as Big Data with the concept of 5 V's.

Velocity

It refers to the high-speed and continuous accumulation of data.

Volume

The amount of data collected should be enormously huge.

Variety

The structure in which data is received. The nature of data can either be structured, semi or unstructured.

Veracity

The inconsistencies and uncertainty in data lead to messy and less accurate modeling.

Value

The data collected should give valuable insights.

A real-world example of big data generation is social media. The list of digital data generated every minute is listed below

- 21 lakh Snaps
- 38 lakh search queries in Google
- 10 lakh people log on to Facebook
- 45 lakh videos are watched on Youtube
- 1880 lakh emails are sent

4. DATA PREPARATION

Depending on the nature of data, it is stored in various formats. The most commonly stored formats are Comma Separated Values (CSV), Tab Separated Values (TSV), eXtensible Markup Language (XML), Really Simple Syndication (RSS), JavaScript Object Notation (JSON), and many more.

Comma-Separated Values is the commonly used import and export database. A snippet of CSV file

Treat,before,after,diff
No treatment,15,18,3
Placebo,16,13,-3
Seroxat,19,12,-7
No treatment,17,15,-2
Effexor,13,10,-3
No treatment,16,16,0

Tab-Separated Values (TSV) are raw data that are imported and exported from spreadsheets. An advantage of TSV format is that the tab (delimiter) doesn't have to be avoided. It is less common format

Name<TAB>Age<TAB>Address
Divya<TAB>18<TAB>Franklin
Paul<TAB>25<TAB>Farm Way
Samatha<TAB>23<TAB>George St

eXtensible Markup Language (XML) is both human and machine readable. It is software and hardware independent. XML data can be shared by different applications which makes it much easier. An example of the XML page

<?xml version="1.0" encoding="UTF-8"?>
<flowerstore>
<flower fragrance="Odour ">
<title lang="en"> Bouquet </title>
<flower > Lavender </flower>
<colour>Lavender </colour>
<price>60</price>
</flower fragrance>
</flowerstore>

Unlike HTML, custom tags are used such as <book> and <price>.

Really Simple Syndication (RSS) is a template used to share data among services. The data is small, fast, and updated frequently. The RSS sample document
<?xml version="1.0" encoding="UTF-8"?>

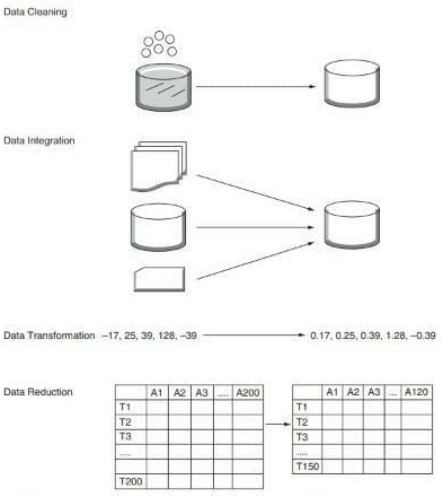
<rss version="2.0">
<page>
<title>Data Science
</title>
<link>http://datascience.org/
</link>
<description>"My data"
</description>
</page>
</rss>

JavaScript Object Notation (JSON) is a very simple data-interchange format. Machine needs to parse and generate. Machines can convert any JavaScript object into JSON. Sending and receiving data in JSON format

<!DOCTYPE html>
<html>
<body>

<script>
Vobj= {"name":"Peter" , "age" :25 , "state" : "New Jersey"};
Vobj=JSON.parse(obj_JSON)
Vobj=JSON.stringfy(obj);
</script>
</body>
</html>

After analyzing the business problem, relative questions are asked and the objective is defined for the problem. The right and required data are gathered, it is termed data acquisition. Data needs to be cleaned and transformed before analyzing. It involves handling many complex scenarios like misspelled attributes, missing values, duplicate values, etc:- The most important step is to define and refine the selection of feature variables that are used in the model. Various machine learning algorithms are applied to identify the best fit model as per the business requirement. The model is then trained and tested. Visualization and Communication of data follows after this. The model is then deployed and maintained.



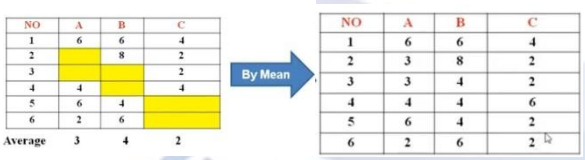
Data Cleaning

Any data-related domain requires data cleansing technique. An upstream is a source from which data is collected, it can be a database, web, or it is collected directly from a mobile app, folder, or another technology. Transformation of data is performed before transferring it downstream. Data preprocessing needs to be done to convert raw data into a human-understandable format. Data that is incomplete, inconsistent, noisy, and lacking in certain behavior or trends needs to be removed or updated to get better quality.

age	income	student	buys_computer
22	high	yes	**yes**
40_b	high	no	**yes**
-22	medium		**?**
21		#e	**no**
12/03/2001	low	yes	
22/06/2000	low	Y	**yes**
34	medium	no	no

Incomplete **Noisy** **Inconsistent**

Data Cleaning is the method to fill in and correct the missing, inconsistent, and irrelevant values, smooth out the noise and identify the outliers in the collected real-world data. Missing values can be handled by ignoring the tuple when the class label is missing or deleting a particular row if it contains 70-75% of missing values. Using global constants, mean (more specifically, attribute mean), or to fill in the missing values.



Incorrect or irrelevant values occur due to erroneous data collection equipment, data recording problems, data transference problems, limitation in modern technology, inconsistent labeling of conventions, duplicate records to name a few. Smoothing techniques are used to remove such noisy data.

i) Binning – The data is sorted into 'buckets' or 'bins' by consulting the neighborhood or values around it.

ii) Regression – Best fit function is identified and used to predict the attribute value.

iii) Clustering – Outliers can be easily detected as the similar values are clustered together.

Data Integration

Data from multiple sources are combined into a coherent storage place for effective and efficient data analyses. Redundant data is commonly addressed and needs to be taken care of. Data value conflicts are detected and resolved so as to engage in schema integration.

Data Transformation

Lesser data directly implies lesser data analysis time. Hence redundant and unnecessary data can be removed from the data warehouse. Some of the strategies include

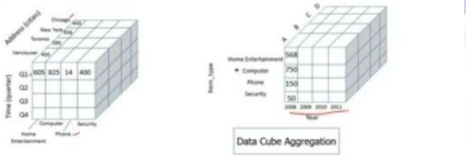
i) Data cube aggregation -

The size of data is reduced without losing information.

Year/Quarter	2014	2015	2016	2017
Quarter 1	200	210	320	230
Quarter 2	400	440	480	420
Quarter 3	480	480	540	460
Quarter 4	560	580	680	640

Year	Sales
2014	1640
2015	1710
2016	2020
2017	1750

The data cube aggregation is a multidimensional aggregation which eases multidimensional analysis.



ii) Attribute subset selection –

Feature selection is a method of removing irrelevant or redundant attributes, in other words abstracting only the required attributes.

iii) Dimensionality reduction –

The indigenous data is compressed and the dimension of data is reduced for an easy analysis process.

Principal Component Analysis (PCA) is the most popular and widely used method of dimensionality reduction.

iv) Numerosity reduction –

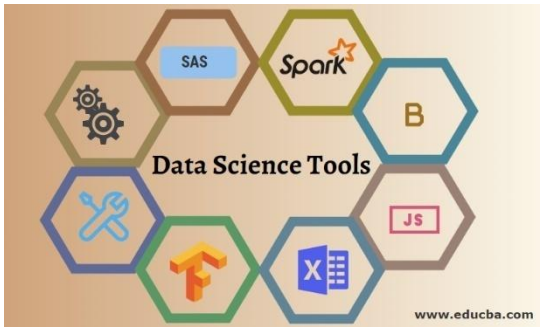
Reducing the data content or volume of data by choosing other possible substitutes that are smaller forms of data representation.

v) Data Discretization –

The data values are converted into a continuous range of intervals.

5. DATA SCIENCE TOOLS

Tools help to analyze, collect, retrieve, visualize and create powerful predictive models using the collected data. Most of the Data Science tools offer complex operations under one roof.



Data Collection Tools

The primary objective of data science is to identify the hidden patterns and relations in data. The primary pre-requisite for data modeling is the availability of required data. Precise and accurate data related to the problem statement must be collected. Data can be collected based around a core set of basic tools including interviews, experimenting and observing, surveys, questionnaires, case studies, group discussions, and many more.There exist technical tools that assist in the collection of related and required data.

1. **Semantria**

Semantria extracts data by analyzing the text and sentiments in it. It is a cloud-based technology. It is a very advanced NLP (neuro-linguistic programming) based application that can detect the sentiments or emotions present in the context based on the language used in it. Text analysis is offered via API and Excel plugin.

semantria

A sample code is given :-

```
var SemantriaSession = require("../").Session;
 var promise=required('promise');
var config = require('../test-config');
   var fs = require('fs');
    var path = require('path');

try { config = require('../test-config.override') } catch(e) {}

var new_config = Object.assign({}, config);
new_config.consumerKey = config.consumerKey || process.env.SEMANTRIA_KEY;
new_config.consumerSecret = config.consumerSecret || process.env.SEMANTRIA_SECRET;
var appConfigurationId = false,
  appConfigurationName = "DiscoveryModeTestApp Configuration",
  collectionId = false,
  SemantriaActiveSession = new SemantriaSession(new_config, "DiscoveryTest");

console.log("Semantria Discovery mode demo.");

//get or create test app configuration
SemantriaActiveSession.getConfigurations(true)
.then(func(configurations) {
  for(var i=0;i<configurations.length:i++) {
      if (configurations[i].name == appConfigurationName) {
       return promise.resolve([configurations[i]]);
    }
  }
  return SemantriaActiveSession.addConfigurations([{
    name: appConfigurationName,
    is_primary: false,
    auto_response: false,
    language: "English"
  }]);
})
.then(function(result){
  appConfigurationId = result[0].id;

  // Creates a sample collection which need to be processed on Semantria
  collectionId = '' + Math.floor(Math.random() * 10000000);

  // Queues collection for processing on Semantria service
  return SemantriaActiveSession.queueCollection({
    id: collectionId,
    documents: getTestDocuments()
  }, appConfigurationId);
})
.then(function() {
  console.log("Collection #" + collectionId + " queued successfully.");
```

```
  //Wait while Semantria process queued collection
  return new promise(function(resolve, reject) {
    var wait_fn = function () {
      console.log("Retrieving your processed results...");

  SemantriaActiveSession.getCollection(collectionId,
appConfigurationId)
      .then(function(processedCollection) {
        if
(processedCollection&&processedCollection.status ==
'PROCESSED') {
          return resolve(processedCollection)
        }
        setTimeout(wait_fn, 5000);
      });
    }
    setTimeout(wait_fn, 5000);
  });
})
.then(function(analyticData) {
  // Printing collection's facets
  console.log("\nCollection facets:");
  if (analyticData.facets) {
    for(var i=0, facet; facet=analyticData.facets[i]; i++) {
      console.log("  %s : %s", facet.label, facet.count);

      if (facet.attributes) {
        for(var j=0, attr; attr=facet.attributes[j]; j++) {
          console.log("    %s : %s", attr.label, attr.count);
        }
      }
    }
  } else {
    console.log("  No facets were extracted for this text");
  }
})
.catch(function(err) {
  console.log("DetailedModeTestApp failed\n");
  console.log(err);
  console.log(err.stack);
})
.then(function() {
  if (!appConfigurationId) return;
  return
SemantriaActiveSession.removeConfigurations([appCon
figurationId]);
});

function getTestDocuments() {
  var filename = path.resolve(__dirname,
'bellagio-data-100.utf8');
  var lines = fs.readFileSync(filename,
'utf8').toString().split("\n");
  // Filter out any empty or very short docs
  lines = lines.filter(function(item) { return item.length>
3; });
  return lines;
}
```

2. Trackur

Trackur utilizes similar text /semantic analysis and content management, which is yet another tool that collects data, particularly on forums and social network platforms. The purpose of Trackur is to track the feedback on brands and products on social media. It also works on sentiment analysis. It adds great value for marketing companies.

Data Storage Tools

The magnetic, optical, or mechanical way to preserve digital information for ongoing or future operations is referred to as data storage. The information is stored and analyzed for further modeling. They store massive amounts of data which are typically reposited in computer memories. It provides a platform to unite servers for easy data access. Various tools available for this purpose are listed below

1. Apache Hadoop

This framework deals with huge volumes of data and its computation. The data storage is distributed among various layered clusters of computers for easy data processing of big data.

2. Apache Cassandra

This tool is a free and open-source platform.SQL (Structured Query Language) and CSL (Cassandra structure language) communicate with the database providing swift availability of data stored on various servers.

An Example code for Apache Cassandra

```
<?xml version="2.0" encoding ="UTF--8" standalone="no"?>
<target variable="_rat_init" depends="resolver-init">
<typedef
Uri ="antlib;org.apache.rat;anttasks"
classpathref:="rat.classpath"/>
</target>

<target name ="_build_ratinclude"
depends="_Rat_init">
<exec executable="git" failifexecutionfails="false"
failonerror="false" resultproperty="git.success"
output="build/.versioned_files">
<arg line="ls-tree -r HEAD --name-only"/>
</exec>
<condition property="rat.skip" value="true">
<not>
<equals arg1="${git.success}" arg2="0"/>
</not>
</condition>
<copy file="build/.versioned_files"
tofile="build/.ratinclude">
<filterchain>
<linecontainsregexp>
<regexp
pattern=".*\.(java|py|sh|xml|spec|md|iml|bat|btm|cql|css|g|html|jflex|jks|mod|name|pom|textile|yml|yaml|MIDRES|HIGHRES|LOWRES)$"/>
</linecontainsregexp>
</filterchain>
</copy>
</target>
<target name="rat-check"
depends="_build_ratinclude" unless="${rat.skip}"
description="License checks on source" >
<rat:reportreportFile="${build.dir}/rat.txt">
<filesetdir="." includesfile="build/.ratinclude">
<!-- Config files with not much creativity -->
<exclude name="*/ide//"/>
<exclude
name="**/metrics-reporter-config-sample.yaml"/>
<exclude name="**/cassandra.yaml"/>
<exclude name="**/cassandra-murmur.yaml"/>
<exclude name="**/cassandra-seeds.yaml"/>
<exclude name="**/harry-generic.yaml"/>
<exclude NAME="**/doc/antora.yml"/>
<exclude name="**/test/conf/cassandra.yaml"/>
<exclude name="**/test/conf/cassandra-old.yaml"/>
<exclude
name="**/test/conf/cassandra_encryption.yaml"/>
<exclude name="**/test/conf/cdc.yaml"/>
<exclude
name="**/test/conf/commitlog_compression_LZ4.yaml"/>
<exclude
name="**/test/conf/commitlog_compression_Zstd.yaml"/>
<exclude
name="**/test/conf/system_keyspaces_directory.yaml"/>
<exclude
name="**/test/conf/sstableloader_with_encryption.yaml"/>
<exclude
name="**/test/conf/unit-test-conf/test-native-port.yaml"/>
<exclude
name="**/test/data/jmxdump/cassandra-3.0-jmx.yaml"/>
```

```
<exclude
name="**/test/data/jmxdump/cassandra-3.11-jmx.yaml"/>
<exclude
name="**/test/data/jmxdump/cassandra-4.0-jmx.yaml"/>
<exclude
name="**/test/resources/data/config/YamlConfigurationLoaderTest/shared_client_error_reporting_exclusions.yaml"/>
<exclude
name="**/tools/cqlstress-counter-example.yaml"/>
<exclude name="**/tools/cqlstress-example.yaml"/>
<exclude
name="**/tools/cqlstress-insanity-example.yaml"/>
<exclude
name="**/tools/cqlstress-lwt-example.yaml"/>
<!-- Documentation files -->
<exclude NAME="*/doc/modules//"/>
<exclude NAME="*/src/java/*/Paxos.md"/>
<!-- NOTICE files -->
<exclude NAME="**/NOTICE.md"/>
<!-- LICENSE files -->
<exclude NAME="**/LICENSE.md"/>
</fileset>
</rat:report>
<exec executable="grep"
outputproperty="rat.failed.files"
failifexecutionfails="false">
<arg line="-A5 'Unapproved licenses'
${build.dir}/rat.txt"/>
</exec>
<fail message="Some files have missing or incorrect
license information. Check RAT report in
${build.dir}/rat.txt for more details! \n
${rat.failed.files}">
<condition>
<and>
<not>
<resourcecontains resource="${build.dir}/rat.txt"
substring="0 Unknown Licenses"
casesensitive="false" />
</not>
</and>
</condition>
</fail>
</target>

<target name="_assert_rat_output">
<fail message="The rat report at build/rat.txt was not
generated. Please ensure that the rat-check task is
able to run successfully. For dev builds only, touch
build/rat.txt to skip this check">
<condition>
<not>
<available file="build/rat.txt" />
</not>
</condition>
</fail>
</target>

</project>
```

3. Mongo DB

Document-oriented database that is free to use Windows, Solaris, and Linux provide the convenience to work with Mongo DB. It is very easy to learn and is reliable. Similar platforms are CouchDB, Apache Ignite, and Oracle NOSQL Database etc:-

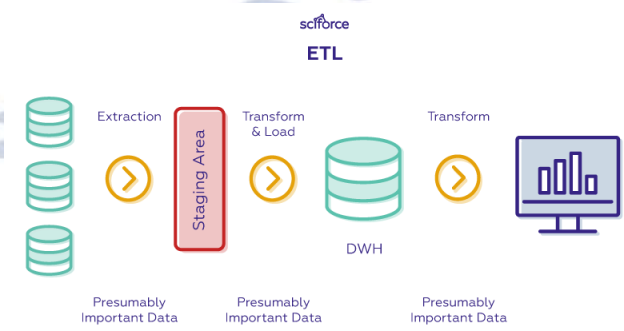
An Example code

```
    import logging
    import time

    import requests

    LOGGER = logging.getLogger(_name_)

    DEFAULT_API_SERVER =
    "https://api.github.com"


    class GithubApi(object):
      """Interface with interacting with the
    githubapi."""

      def _init_(self,
    api_server=DEFAULT_API_SERVER):
        """Create a githubapi object."""
    self.api_server = api_server

      @staticmethod
      def _make_request(url, params):
```

```
        """Make a request to github. Log the request,
param and request time."""
LOGGER.debug("making github request: %s,
params=%s", url, params)
        start = time.time()
        response = requests.get(url=url,
params=params)
LOGGER.debug("Request took %fs:",
round(time.time() - start, 2))
response.raise_for_status()

        return response

    @staticmethod
    def _parse_link(response):
        """Parse a github 'Link' header into an object
with paginated links."""
link_object = {}

        if not response.headers["Link"]:
            return link_object

  links
response.header["Link"].split(",")
   for link in links:
            link-parts = link_split(";")
link_type = link_parts[1].replace("rel=", "").strip("
\"")
link_address = link_parts[0].strip("<> ")
link_object[link_type] = link_address

        return link_object

    def get_commits(self, owner, project, params):
        """Get the list of commits from a specified
repository from github."""
url =
"{api_server}/repos/{owner}/{project}/commits".f
ormat(api_server=self.api_server,

owner=owner, project=project)

LOGGER.debug("get_commits project=%s/%s,
params: %s", owner, project, params)
        response = self._make_request(url, params)
        commits = response.json()

        # If there are more pages of responses, read
those as well.
        links = self._parse_link(response)
        while "next" in links:
            response =
self._make_request(links["next"], None)
            commits += response.json()

            links = self._parse_link(response)

LOGGER.debug("Commits from github
(count=%d): [%s - %s]", len(commits),
commits[-1]["sha"],
                commits[0]["sha"])

        return commits
```

Data Extraction tools

The Cyber Server, internet, is nothing but a colossal pool of data that contains relevant information which is used to extract knowledge and useful insights to ensure growth for businesses in the fast-growing technological globe. These are web scraping tools, they extract data directly from websites. Leveraging cautiously collected data extraction tools lend a helping hand to companies to analyze and avail easily.



1. Mailparser

An example code

```
import argparse
import os
import runpy
import sys

import mailparser  from .exceptions
import MailParserOutlookError
from .utils import (
custom_log,
print_attachments,
print_mail_fingerprints,
safe_print,
write_attachments,
)

current =
os.path.realpath(os.path.dirname(_file_))

_version_ = runpy.run_path(
os.path.join(current, "version.py"))["_version_"]


def get_args():
  parser = argparse.ArgumentParser(
    description="Wrapper for email Python
Standard Library",
epilog="It takes as input a raw mail and
generates a parsed object.",
formatter_class=argparse.ArgumentDefaultsHel
pFormatter)

parsing_group =
parser.add_mutually_exclusive_group(required
=True)
parsing__group.add__argument(
    "_f",
    "__file",
    de="file",
    help="Raw email file")
parsing_group.add_argument(
    "-s",
    "--string",
dest="string",
    help="Raw email string")
parsing_group.add_argument(
    "-k",
    "--stdin",
dest="stdin",
    action="store_true",
    help="Enable parsing from stdin")

parser.add__argument(
    "_l",
    "__log_level",
    des="log__clevel",
    default ="WARNING",
    choice =["CRITICAL", "ERROR",
"WARNING", "INFO", "DEBUG", "NOTSET"],
    HELP="Set level of log")

parser.add__argument(
    "_j",
    "__json",
    des="json",
actio ="store__true",
    HELP="Show the JSON of parsed mail")

parser.add__argument(
    "_b",
    "--body",
    des="body",
actio ="store__true",
    HELP="Print the body of mail")

parser.add__argument(
    "_a",
    "__attachments",
    des="attachments",
actio ="store_true",
    HELP ="Print the attachments of mail")

parser.add__argument(
    "_r",
    "__headers",
    des="headers",
actio ="store_true",
    HELP="Print the headers of mail")

parser.add__argument(
    "_t",
```

```
        "__to",
        des="to",
actio ="store_true",
        HELP="Print the to of mail")

parser.add__argument(
        "_dt",
        "__delivered-to",
        des="delivered_to",
actio ="store_true",
    HELP="Print the delivered-to of mail")

parser.add__argument(
        "_m",
        "__from",
        des="from_",
actio ="store_true",
        HELP="Print the from of mail")

parser.add__argument(
        "_u",
        "__subject",
        des="subject",
actio ="store_true",
        HELP="Print the subject of mail")

parser.add__argument(
        "_c",
        "__receiveds",
        des="receiveds",
actio ="store_true",
        HELP="Print all receiveds of mail")

parser.add__argument(
        "_d",
        "__defects",
        des="defects",
actio ="store_true",
        HELP="Print the defects of mail")

parser.add__argument(
        "_o",
        "__outlook",
        des="outlook",
actio ="store_true",
        HELP="Analyze Outlook msg")

parser.add_argument(
        "-i",
        "--senderip",
dest="senderip",
metavar="Trust mail server string",
        help="Extract a reliable sender IP address heuristically")

parser.add_argument(
        "-p",
        "--mail-hash",
dest="mail_hash",
        action="store_true",
        help="Print mail fingerprints without headers")

parser.add_argument(
        "-z",
        "--attachments-hash",
dest="attachments_hash",
        action="store_true",
        help="Print attachments with fingerprints")

parser.add_argument(
        "-sa",
        "--store-attachments",
dest="store_attachments",
        action="store_true",
        help="Store attachments on disk")

parser.add_argument(
        "-ap",
        "--attachments-path",
dest="attachments_path",
        default="/tmp",
        help="Path where store attachments")

parser.add__argument(
        '_v',
        '__version',
actio ='version',
versio ='%(prog)s
 {}' .format (_versio_))

   return parsers
```

```
def main():

args= get_args().parse_args()
   log = custom_log(level=args.log_level)

   if args.file:
      if args.outlook:
log.debug("Analysis Outlook mail")
         parser                                    =
mailparser.parse_from_file_msg(args.file)
      else:
         parser                                    =
mailparser.parse_from_file(args.file)
elifargs.string:
      parser                                       =
mailparser.parse_from_string(args.string)
elifargs.stdin:
      if args.outlook:
         raise MailParserOutlookError(
            "You can't use stdin with msg Outlook")
      parser                                       =
mailparser.parse_from_file_obj(sys.stdin)

   if args.json:
safe_print(parser.mail_json)

   if args.body:
safe_print(parser.body)

   if args.headers:
safe_print(parser.headers_json)

   if args.to:
safe_print(parser.to_json)

   if args.delivered_to:
safe_print(parser.delivered_to_json)

   if args.from_:
safe_print(parser.from_json)

   if args.subject:
safe_print(parser.subject)

   if args.receiveds:
safe_print(parser.received_json)

   if args.defects:
log.debug("Printing defects")
      for i in parser.defects_categories:
safe_print(i)

   if args.senderip:
log.debug("Printing sender IP")
      r                                            =
parser.get_server_ipaddress(args.senderip)
      if r:
safe_print(r)
      else:
safe_print("Not Found")

   if args.attachments or args.attachments_hash:
log.debug("Printing attachments details")
print_attachments(parser.attachments,
args.attachments_hash)

   if args.mail_hash:
log.debug("Printing also mail fingerprints")
print_mail_fingerprints(parser.body.encode("utf
-8"))

   if args.store_attachments:
log.debug("Store attachments on disk")
write_attachments(parser.attachments,
args.attachments_path)


if _name_ == '_main_':
main()
```

2. Octo Parse
3. Content Grabber
4. OutWitHub
5. Web Scraper
6. Spinn3r
7. ParseHub
8. Fminer
9. Table Capture
10. Tabula
11. Scrapy
12. Dexi.io

Data Cleaning / Refining Tools

A data scientist in the early stages realizes that data cleaning is the most important and efficient step in the data analysis process. It includes many complex operations like removing data that isn't relevant, ensuring data is consistent by mapping it to unified underlying structure, getting rid of outliers and resolving syntax errors.The tools such as MS Excel,Python and other data science tools are invaluable for data cleaning.Data cleaning tools are recapped in this section. It can be widely used to clean any kind of big data.

1. Data Cleaner
2. OpenRefine
3. Trifacta Wrangler
4. Drake
5. TIBCO Clarity
6. Winpure
7. Data Ladder
8. Data Cleaner
9. Cloudingo Reifier
10. IBM Infosphere Quality Stage

Data Analysis Tools

Data analysis tools are not only for the analysis of data but also for the performance of certain operations on the data. They inspect, study, and model the data to draw useful insights out of the data, which is conclusive and provides a helping hand for decision making to a certain problem or query.

1. **R**

It is most popularly used for statistical computing and graphical representation. Most commonly used among data miners and statisticians for data analysis.

2. **Python**

It is a powerful and high-level programming language with multiple uses.

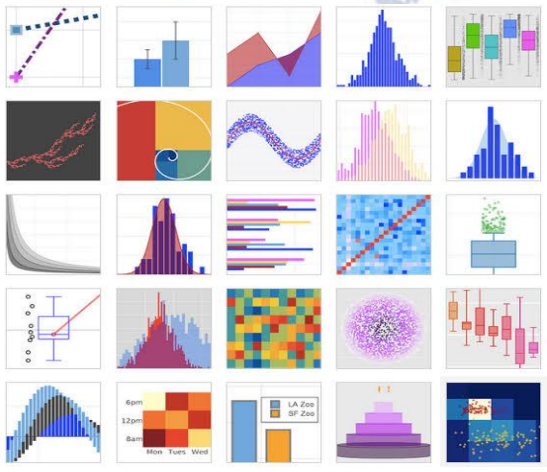
3. **Apache Spark**

Workflows are highly interactive that provide productive real-time data analysis.

Data Visualization Tools

It is a graphical or diagrammatic representation of the collected data. It is an interdisciplinary field that has particularly an efficient way of communicating huge amounts of data in graphical form. Various ways to represent data are

Pie charts, Bar charts, Stacked bar graphs, Stacked column charts, Scatter plot, Area charts, Line charts, Treemaps, Funnel charts, Histograms, Gantt charts, Heat maps, Box plot and Whisker plot, Waterfall charts, Bubble charts, Bullet charts, Dot distribution maps, Choropleth map, Violin plot, Matrix, Flow maps, Infographics, Maps, etc:- By using the simple and basic visual elements including charts, graphs, and maps, it provides an easy and accessible way to see and understand trends, hidden patterns in the data and pick out the outliers easily.



1. Circos
2. Data Wrapper
3. Google Charts
4. Google Fusion Tables
5. Infogr.am
6. Knoema
7. Mr. Data Converter
8. OpenHeatMap.com
9. Piktochart
10. Plot.ly
11. R Project for Statistical Computing
12. Tableau
13. Watson Analytics
14. Datawrapper
15. Zoho Analytics

6. APPLICATION

Data Science is revolutionizing the world. There was a time when this domain was still in its nascent stages and it was only used for research-based applications whereas now it exists in every part of our works. It has complete capability to provide solutions to various problems across multiple domains. The most interesting and popular implementation and use of data science are discussed.

Internet Search – Many web browsers such as Google, Yahoo, Bing, etc:- rely on data science algorithms to provide the best result for the searched query in milliseconds.

Digital Advertisement – Also termed Targeted Advertising is an important application of data science. The complete digital marketing system consisting of banner displays on various websites and even the digital billboards at public places are determined using data science related algorithms.

It is targeted based on the user's behavior.

Website Recommendation – Similar product suggestions on online markets, movie recommendation based on our interests and previous search improves user experience.

Image Recognition – Facebook identifies similar faces and recommends mutual people, Whatsapp is used through the web by scanning the QR code, and similarly, Google photos use image recognition to identify similar pictures.

Speech Recognition – Siri, Alexa, Cortana, Google Assistant, etc uses speech recognition feature to get our works done.

Airline Route Planning –Airline service providers analyze the passenger travel details to find out the travel demand for specific cities. To get an insight about the amount of fuel required so that air-fuel consumption can be reduced. It also deals with the controlling of the air-traffic and safe landing of flights. With the help of Data Science identify areas of improvement like flight prediction, consumer experience, and decision making.

Fraud and Risk Detection – To reduce and recover losses. Customer profiling divides and conquers data, checks the various expenditures to analyze the likelihood of risk in default.

Medical Sciences – Medical Image Analysis, Genetics and Genomics, Creation of drugs and calculating success rate as well as a virtual assistant for patients.

Gaming – Data Science is closely related to Virtual Reality (VR) which considers the computing knowledge and algorithms to deliver the best viewing experience. (AR – Augmented Reality)

7. CONCLUSION

There is still a lot to come and improve such that these tech giants can become successful in the quest to develop futuristic designs. The massive growth of digital data provides a glimpse of how the future will be. Data Scientist has become the most popular and in-demand job. The data in the database must be protected from destructive forces and unauthorized users i.e from cyberattacks and data breaches. Data security and data safety are the major concerns. It must be protected against loss by ensuring regular back-ups and safe storage. If we look a little bit ahead, jobs in the data science field are expected to skyrocket. It has tremendous opportunities for advancement in the future.

Conflict of interest statement

Authors declare that they do not have any conflict of interest.

REFERENCES

[1] Big Data with Cloud Computing: Discussions and Challenges

[2] Social Set Analysis: a Theoretical Approach to huge information Analytics

[3] A Small-Review on Machine Learning in huge information Analytics and Prospects

[4] Big Data Science on COVID-19 information Survey on Lie cluster Machine Learning

[5] Multi-Attention Fusion Modeling for Sentiment Analysis of instructional big data

[6] Big data, Knowledge: huge information for personalized trending

[7] Applying Big data primarily based on Deep Learning System to Impingement Detection

[8] Huge empirical Improved information Acquisition and Storage System for planning Industrial information Platform

[9] A Survey of information Partitioning and Sampling strategies to Support huge information Analysis

[10] Associate in Nursing energy-efficient information assortment theme mistreatment denoising autoencoder in wireless detector networks

[11] Distributed information ways to Support Large-Scale information Analysis Across Geo-Distributed information Centers

[12] Revealing the User Behavior Pattern mistreatment HNCORS RTK Location Big data.

[13] A technique of period information Fusion for Localized huge information Analytics

[14] Huge empirical Improved information Acquisition and Storage System for planning Industrial information Platform

[15] Program Reform in huge information Education at Applied Technical faculties and Universities in China

[16] A review on machine learning in Big Data analytics: applications and challenges.

[17] Biface feedback dynamic particle filter with huge information for the particle degeneracy drawback

[18] A scientific Review of massive information Analytics for Oil and Gas trade four.0

[19] Big Data Platform for instructional Analytics

[20] Distinctive Similarities of massive information Projects–A Use Case Driven Approach

[21] A Web model for Remote Sensing processing and Production System.

[22] Comparative Analysis of Energy-Efficient programming Algorithms for giant information Applications

[23] A knowledge base Discovery and data processing method Model for Metabolomics

[24] Mapping the massive information Landscape: Technologies, Platforms, and Paradigms for period Analytics of information Streams

[25] Grady, N. W. (2016). KDD meets Big Data. In Big Data (Big Data), IEEE International Conference on. IEEE.

[26] Shearer, C. (2000). The CRISP-DM model: The new blueprint for data mining," Data Warehousing Journal5(4)

[27] Analysis and solution of data quality in the data warehouse of Chinese material medical.

[28] An Assessment on Classification in Python Using Data Science.

[29] Open-sourcing education for Data Engineering and Data Science.

[30] Big data analytics: Analytics Ops for data science.

[31] Embedding Data Science into Computer Science Education