INDUSTRIAL PROJECT REPORT

Submitted in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Under the supervision of

Mr. Vedprakash Kaidri
(Lead Data Scientist, SetuServ)

By

Akshita Sood (141331)

To

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY

WAKNAGHAT, SOLAN – 173234

HIMACHAL PRADESH, INDIA

May-2018

CERTIFICATE

I hereby declare that the work presented in this report of my Industrial Project in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** submitted to the department of Computer Science & Engineering, Jaypee University of Information Technology, Waknaghat is an authentic record of my own work carried out over a period from 7 Feb, 2018 to 21 May, 2018 under the supervision of **Mr. Vedprakash Kaidri,** Lead Data Scientist at SetuServ.
The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Date: …………………….

Akshita Sood, 141331
This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Dr. Punit Gupta

Dated: 21 May, 2018

ACKNOWLEDGEMENT

It is my pleasure to be indebted to various people, who directly or indirectly contributed in the development of this work and who influenced my thinking, behaviour, and acts during the course of study. The internship opportunity I had with SetuServ was a great chance for learning and professional development. Therefore, I consider myself as a very lucky individual as I was provided with an opportunity to be a part of it. I am also grateful for having a chance to meet so many wonderful people and professionals who led me though this internship period.

I am using this opportunity to express my deepest gratitude and special thanks to the CEO and Founder of SetuServ, **Mr. Sanga Reddy Peerreddy** who in spite of being extraordinarily busy with his duties, took time out to hear, guide and keep me on the correct path and allowing me to carry out my project at their organization.

I owe our profound gratitude to **Mr. Vedaprakash Kadiri**, Lead Data Scientist for taking part in useful decision making & giving necessary advices and guidance and arranged all facilities to make working easier. I choose this moment to acknowledge his contribution gratefully.

I perceive this opportunity as a big milestone in my career development. I will strive to use gained skills and knowledge in the best possible way, and I will continue to work on their improvement, in order to attain desired career objectives. Hope to continue cooperation with all of you in the future.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 Introduction

1.1.1 Data Science

The term “Data Science” has emerged only recently to specifically designate a new profession that is expected to make sense of the vast stores of data. This data can be text, videos, audio. Data science today has expanded to every field. From predicting stock prices to generating images from captions, data science is ubiquitous. But making sense of data has a long history and has been discussed by scientists, statisticians, librarians, computer scientists and others for years.

Data science can be defined as a blend of skills in three major areas which is depicted in Figure 1.

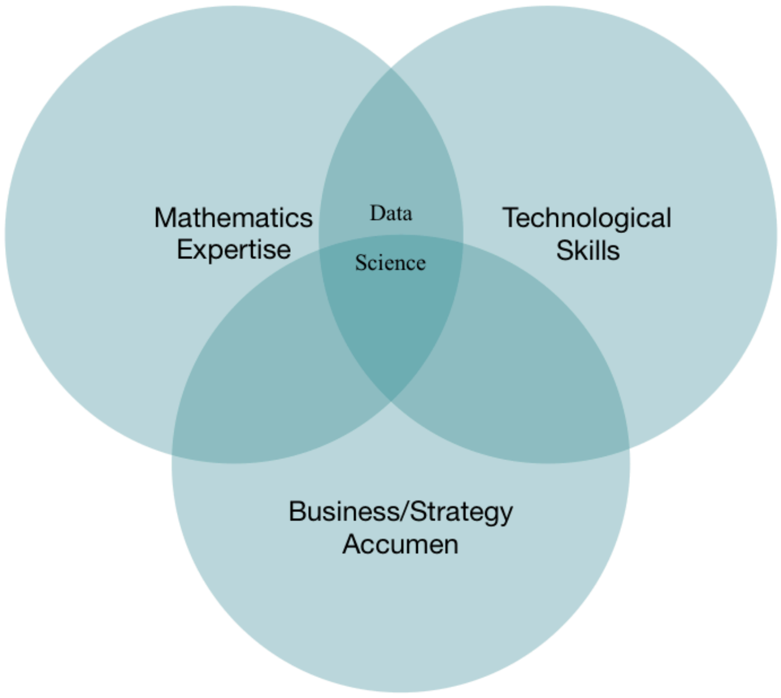


Figure 1: Data Science

1.1.2 Data Scientist

A common personality trait of data scientists is they are deep thinkers with intense intellectual curiosity. Data science is all about being inquisitive – asking new questions, making new discoveries, and learning new things. Data scientists are passionate about what they do, and reap. There is a glaring misconception out there that you need a sciences or math Ph.D to become a legitimate data scientist. E.g. a Ph.D statistician may still need to pick up a lot of programming skills and gain business experience, to complete the trifecta. The unyielding intellectual interest of facts scientists push them to be encouraged autodidacts, driven to self-learn the proper abilities, guided via their own willpower. In general, a Data Scientist takes on the following three main tasks –

- **Data Exploration** - This process mainly involves gathering, cleaning and analysing data. Because data scientists utilize technology so that it will wrangle huge facts sets and work with complicated algorithms, and it calls for tools far greater sophisticated than Excel. Data scientists need to be able to code - prototype brief solutions, in addition to integrate with complicated data systems. Core languages associated with data science include SQL, Python, R, and SAS. On the periphery are Java, Scala, Julia, and others.

- **Using machine learning** - Machine learning is a term closely associated with data science. It refers to a broad class of methods that revolve around data modelling to (1) algorithmically make predictions, and (2) algorithmically decipher patterns in data.

- **Data Product Engineering** - This involves communicating your findings via a product, which involves visualisation and presentations. Not only for presentations at high level meetings but also to explain things to the team, a data science expert would need good communication skills. The work that a team of data scientists do has to be aligned with the goals of the business to reap benefits. Hence it is crucial to convey the right message to the team and that's where someone with a good communication skill would be of immense value.

1.2 Overview of Technologies Used

1.2.1 Natural Language Processing

Natural language processing is a set of techniques that allows computers and people to interact. Consider the process of extracting information from some data generating process: A company wants to predict user traffic on its website so it can provide enough compute resources (server hardware) to service demand.

Natural Language Processing is a field that covers computer understanding and manipulation of human language, and it's ripe with possibilities for news gathering," Anthony Pesce said in Natural Language Processing in the kitchen. "You usually hear about it in the context of analyzing large pools of legislation or other document sets, attempting to discover patterns or root out corruption." NLP is a way for computers to analyze, understand, and derive meaning from human language in a smart and useful way. By utilizing NLP, developers can organize and structure knowledge to perform tasks such as automatic summarization, translation, named entity recognition, relationship extraction, sentiment analysis, speech recognition, and topic segmentation. "Apart from common word processor operations that treat text like a mere sequence of symbols, NLP considers the hierarchical structure of language: several words make a phrase, several phrases make a sentence and, ultimately, sentences convey ideas," John Rehling, an NLP expert at Meltwater Group, said in *How Natural Language Processing Helps Uncover Social Media Sentiment*. "By analyzing language for its meaning, NLP systems have long filled useful roles, such as correcting grammar, converting speech to text and automatically translating between languages."

NLP is used to analyse text, allowing machines to understand how human's speak. This human-computer interaction enables real-world applications like automatic text summarization, sentiment analysis,inamed entity recognition, parts-of-speech tagging, relationship extraction, stemming, and more. NLP is commonly used for text mining, machine translation, and automated question answering. NLP is characterized as a hard problem in computer science. Human language is rarely precise, or plainly spoken. To understand human language is to understand not only the words, but the concepts and how they're linked together to create meaning. Despite language being one of the easiest things for humans to learn, the ambiguity of language is what makes natural language processing a difficult problem for computers to master.”

NLP algorithms are typically based on machine learning algorithms. Instead of hand-coding large sets of rules, NLP can rely on machine learning to automatically learn these rules by analysing a set of examples (i.e. a large corpus, like a book, down to a collection of sentences), and making a statistical inference. In general, the more data analyzed, the more accurate the model will be.

- Summarize blocks of text using Summarizer to extract the most important and central ideas while ignoring irrelevant information.
- Create a chat bot using Pasey McParseface, a language parsing deep learning model made by Google that uses Point-of-Speech tagging.
- Automatically generate keyword tags from content using AutoTag, which leverages LDA, a technique that discovers topics contained within a body of text.
- Identify the type of entity extracted, such as it being a person, place, or organization using Named Entity Recognition.
- Use Sentiment Analysis to identify the sentiment of a string of text, from very negative to neutral to very positive.
- Reduce words to their root, or stem, using Porter Stemmer, or break up text into tokens using Tokenizer.

Open Source NLP Libraries provide the algorithmic building blocks of NLP in real world applications. Algorithm provides a free API endpoint for many of these algorithms, without ever having to setup or provision servers and infrastructure.

- Apache Open NLP: a machine learning toolkit that provides tokenizers, sentence segmentation, part-of-speech tagging, named entity extraction, chunking, parsing, co reference resolution, and more.
- Natural Language Toolkit (NLTK): a Python library that provides modules for processing text, classifying, tokenizing, stemming, tagging, parsing, and more.
- Standford NLP: a suite of NLP tools that provide part-of-speech tagging, the named entity recognizer, coreference resolution system, sentiment analysis, and more.

- MALLET: a Java package that provides Latent Dirichlet Allocation, document classification, clustering, topic modelling, information extraction, and more.

1.2.2 Machine Learning

Machine learning is a subfield of artificial intelligence (AI). The goal of machine learning generally is to understand the structure of data and fit that data into models that can be understood and utilized by people.

Despite the fact that system studying is a area inside computer science, it differs from conventional computational procedures. In traditional computing, algorithms are units of explicitly programmed commands used by computers to calculate or hassle clear up. gadget gaining knowledge of algorithms rather allow for computer systems to train on information inputs and use statistical analysis to be able to output values that fall within a selected variety. because of this, system gaining knowledge of helps computer systems in constructing models from pattern records for you to automate decision-making strategies based totally on statistics inputs.

Any technology user today has benefitted from Machine Learning. Facial recognition allows social media systems to help users tag and share photographs of friends. Optical character recognition (OCR) technology converts images of text into movable kind. recommendation engines, powered via machine learning, advise what films or television suggests to observe subsequent based totally on consumer possibilities. Self-riding cars that depend on gadget studying to navigate might also quickly be available to clients.

Machine learning is a continuously developing field. Because of this, there are some considerations to keep in mind as you work with machine learning methodologies, or analyze the impact of machine learning processes.

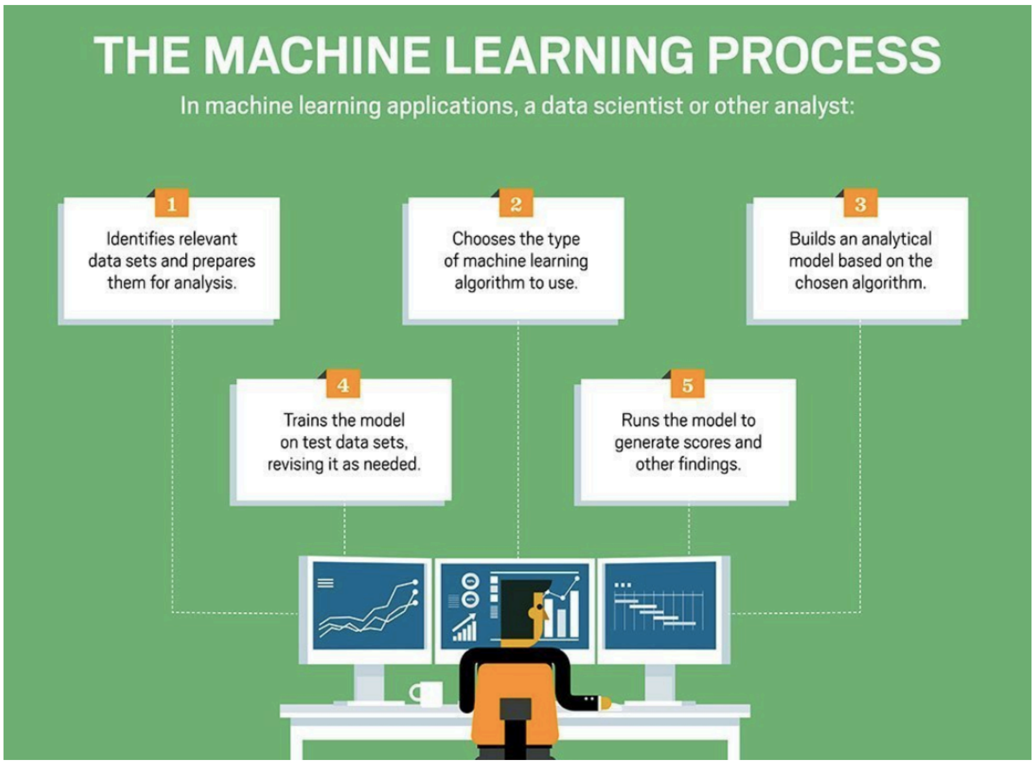

THE MACHINE LEARNING PROCESS
In machine learning applications, a data scientist or other analyst:
1
Identifies relevant data sets and prepares them for analysis.
2
Chooses the type of machine learning algorithm to use.
3
Builds an analytical model based on the chosen algorithm.
4
Trains the model on test data sets, revising it as needed.
5
Runs the model to generate scores and other findings.

Figure 2: Machine Learning Process

Machine Learning Methods

In machine learning, tasks are generally classified into broad categories. These categories are based on how learning is received or how feedback on the learning is given to the system developed.

Two of the mostiwidely adopted machine learning methods are **supervised learning** which trains algorithms based on example input and output data that is labeled by humans, and **unsupervised learning** which provides the algorithm with no labeled data in order to allow it to find structure within its input data. Let's explore these methods in more detail.

Supervised Learning

In supervised learning, the computer is provided with example inputs that are labeled with their desired outputs. The purpose of this method is for the algorithm to be able to "learn" by comparing its actual output with the "taught" outputs to find errors, and modify the model accordingly. Supervised learning therefore uses patterns to predict label values on additional unlabelled data.

For example, with supervised learning, an algorithm may be fed data with images of sharks labelled as fish and images of oceans labelled as water. By being trainedionithis data, the supervised learning algorithm should be able to later identify unlabelled shark images as fish and unlabelled ocean images as water.

A common use case of supervised learning is to use historical data to predict statistically likely future events. It may use historical stock market information to anticipate upcoming fluctuations, or be employed to filter out spam emails. In supervised learning, tagged photos of dogs can be used as input data to classify untagged photos of dogs.

Unsupervised Learning

In unsupervised learning, data is unlabelled, so the learning algorithm is left to find commonalities among its input data. As unlabelled data are more abundant than labeled data, machine learning methods that facilitate unsupervised learning are particularly valuable.

The goal of unsupervised learning may be as straightforward as discovering hidden patterns within a dataset, but it may also have a goal of feature learning, which allows the computational machine to automatically discover the representations that are needed to classify raw data.

Unsupervised learning is commonly used for transactional data. You may have a large dataset of customers and their purchases, but as a human you will likely not be able to make sense of what similar attributes can be drawn from customer profiles and their types of purchases. With this data fed into an unsupervised learning algorithm, it may be determined that women of a certain age range who buy unscented soaps are likely to be pregnant, and therefore a marketing campaign related to pregnancy and baby products can be targeted to this audience in order to increase their number of purchases.

Without being told a “correct” answer, unsupervised learning methods can look at complex data that is more expansive and seemingly unrelated in order to organize it in potentially meaningful ways. Unsupervised learning is often used for anomaly detection including for fraudulent credit card purchases, and recommender systems that recommend what products to buy next. In unsupervised learning, untagged photos of dogs can be used as input data for the algorithm to find likenesses and classify dog photos together.

Approaches

As a field, machine learning is closely related to computational statistics, so having a background knowledge in statistics is useful for understanding and leveraging machine learning algorithms.

For those who may not have studied statistics, it can be helpful to first define correlation and regression, as they are commonly used techniques for investigating the relationship among quantitative variables. **Correlation** is a measure of association between two variables that are not designated as either dependent or independent. **Regression** at a basic level is used to examine the relationship between one dependent and one independent variable. Because regression statistics can be used to anticipate the dependent variable when the independent variable is known, regression enables prediction capabilities.

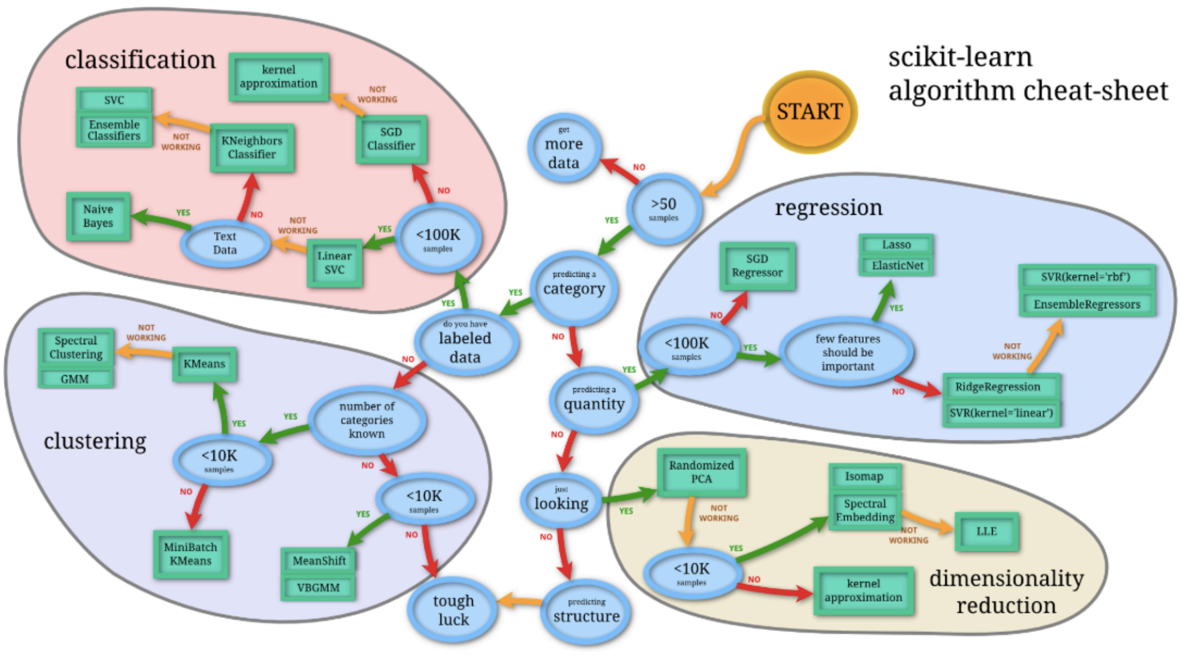


Figure 3: Approaches for Machine Learning as per the Data

1.1.3 Deep Learning (Neural Networks)

Deep Learning is a subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called artificial neural networks.

Deep learning (also known as deep structured learning or hierarchical learning) is part of a broader family of machine learning methods based on learning data representations, as opposed to task-specific algorithms. Learning can be supervised, semi-supervised or unsupervised.

Deep learning models are loosely related to information processing and communication patterns in a biological nervous system, such as neural coding that attempts to define a relationship between various stimuli and associated neuronal responses in the brain.

While traditional machine learning algorithms are linear, deep learning algorithms are stacked in a hierarchy of increasing complexity and abstraction. To understand deep learning, imagine a toddler whose first word is dog. The toddler learns what a dog is (and is not) by pointing to objects and saying the word dog. The parent says, "Yes, that is a dog," or, "No, that is not a dog." As the toddler continues to point to objects, he becomes more aware of the features that all dogs possess. What the toddler does, without knowing it, is clarify a complex abstraction (the concept of dog) by building a hierarchy in which each level of abstraction is created with knowledge that was gained from the preceding layer of the hierarchy."

1.3 Organization

What Do We Do ?

Analyse millions of customer comments (or other text data) at a speed no human can match and an accuracy level no computer algorithm can match. 80% of "big-data" is in text form – we help customer focused businesses harness the power of this data.

How Do We Do It ?

By combining machine learning algorithms with human intelligence to leverage the best of both. Our distinctive approach has won recognition from NIST (a US federal technology agency), TiE, Pan-IIT, NASSCOM, Google, and Yelp.

Does It Work ?

We have already generated significant sales from 40 clients, which include the world's leading e-commerce, CPG, Beauty companies & sports teams. We are also backed by a top tier VC company (Vedanta Capital) and angel investors with successful track record in building startups.

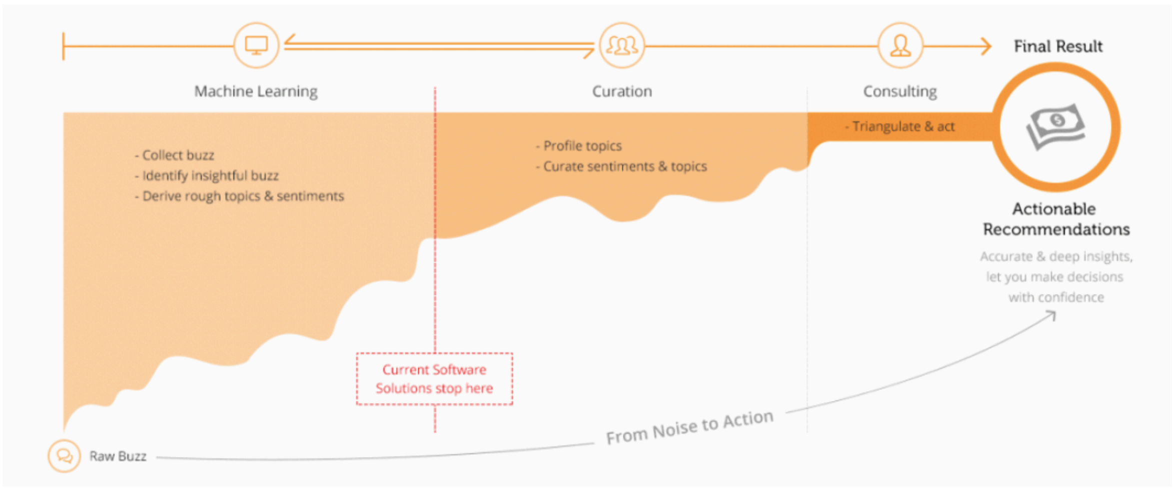
Can it scale?

Our software as a service (SAAS) product that will allow clients to access our solution directly from their computers, and customize it to their needs. With this product, we can serve hundreds of enterprise clients.

"SetuServ is a text analytics firm focused on building Voice of Customer (VoC) solutions using Natural Language Processing (NLP) and curation. These VoC solutions derive actionable insights that clients can use to "close the loop" with their customers and to make strategic & operational improvements. SetuServ has successfully applied this with 20+ clients on a variety of textual data sources such as Net Promoter Score (NPS)/survey "open" ends, social media comments, reviews and in-bound complaints. SetuServ's offering was demonstrated at the Text Retrieval Conference (TREC), and is proven to produce as accurate outputs as those of the professional curators of National Institute of Standards and Technology (NIST). SetuServ has given tech talks at Google and Yelp. SetuServ is the winner of the prestigious 2014 TiE50 Hot Startup Awards."

- **We build custom AI models to solve your specific business challenges**
  - **We collect data , wherever it resides and separate signal from the noise.**
    Our platform is agnostic to data source and will gather feedback from your customers, curate enormous volumes of unstructured text, and design specific taxonomies to reflect each individual business question. As your competitive ecosystem evolves, our solution will evolve in tandem.

    Our service-driven approach ensures incredibly accurate and high-performing machine learning models, resulting in granular and actionable insights. Whether you are measuring NPS, CSAT or star ratings, the SetuServ platform will provide sentiment and weight for every contributing trait.



Machine Learning
Curation
Consulting
Final Result
- Collect buzz
- Identify insightful buzz
- Derive rough topics & sentiments
- Profile topics
- Curate sentiments & topics
- Triangulate & act
Actionable Recommendations
Accurate & deep insights, let you make decisions with confidence
Current Software Solutions stop here
Raw Buzz
From Noise to Action

Figure 4: SetuServ's Skierarchy Platform
(Artificial Intelligence + Human Intelligence)

An out-of-the box product isn't enough. Vanity metrics won't give you insight. Our Skierarchy Platform architecture ensures that our analyst team is working with you to continually tune the machine and react to the fluid nature of your competitive ecosystem in real-time.

- **Customized AI : Machine Power + Human Accuracy**

  "Too often, businesses are using simple counting statistics or multiple choice survey feedback as proxies for complex concepts like engagement, affinity and loyalty. Such an approach maybe helpful to provide directional feedback, but is wholly-insufficient to support important business decisions, both tactical and strategic.

  Industry-specific AI solutions, while targeted, are typically modeled on generic data and therefore too broad to provide answers to the specific questions that your employees are asking on a daily basis.

  Our approach is designed to allow for significant up-front learning and collaboration between SetuServ consultants and client stakeholders. This allows us to build models that are uniquely-tailored to each client and for each specific functional purpose.

  80% of the answer isn't good enough. We'll get you the rest of the way there.

- **How does it work ?**

  - **Identify sources and collect data**

    The SetuServ platform is agnostic and can ingest data from any public or private source. We will work with you to identify valuable troves of customer feedback across social media platforms, ecommerce sites and discussion forums, and supplement with private resources such as NPS/CSAT surveys and customer support logs.

  - **Build custom taxonomies**

    Our consultants conduct a deep dive into the specifics of each respective business question to ensure that your unique taxonomies will provide actionable insight. We'll continually monitor the sources as consumer communication preferences change and curate the incoming text to match your individual needs.

- **Act, Adapt , Repeat**

  Armed with granular insights, SetuServ clients can take tactical action to address product, marketing, or customer service issues, while using higher-level trends and emerging themes to guide strategic brand direction, innovation programs or corporate strategy. As the competitive ecosystem evolves and the facts change, our analyst team will ensure that the SetuServ platform adapts and continually gets smarter.

Solutions : Our customers use the insights generated from the SetuServ platform to inform the tactical to the strategic. Whether you are testing the impact of marketing communications, tracking customer service metrics or providing deal support for an acquisition, SetuServ can help.

e.g. Change product packaging
Adjust customer support scripts
Inform innovation programs
Improve start ratings
Boost NPS and CSAT scores
Drive shareholder value

Consumer tastes change, competitors launch new products, and new alternatives emerge. Modern industries are in a state of near-constant change, and our platform is custom-built to reflect that.

Our customers include global CPG companies driving incremental innovation at the product level and/or disruptive innovation at the corporate level, retailers seeking to deconstruct online star ratings, consumer brands measuring the impact of digital media spend and hardware manufacturers quantifying customer success programs.

Across industry type, company size and stage, and functional department, organizations are turning to SetuServ to provide the “why” to their most pressing business issues.

Product Development

Tweak in-market products and inform your new product pipeline

Marketing Communications

Understand what messages are resonating with your audience, across platforms

Customer Service

Track emerging issues, quantify impact and capture improvements

Innovation

Identify white space in the market to inform large-scale innovation or small bets

Corporate Strategy

Insert the customer's voice into long-term planning and resource decisions at the highest level

Transaction Support

Provide robust and structured diligence for strategic transactions and scenario analyses.

1.4 Problem Statement

In terms of what text analytics does, there are a lot of misconceptions. To many business users, text analytics is a box where unstructured text goes in and structured information magically comes out. Through the use of a complex taxonomy and algorithms, text analytics is essentially about making human information (text, documents, language) understandable by computers and a way to discover value in text. Text Analytics is a process that can analyse any type of unstructured text, extracting high-quality and relevant information that can drive further analysis and strategic decision making.

Traditional analysis technologies are not able to effectively handle unstructured data because they merely manage lists organized in databases, and, equally important, without understanding the meaning of the terms, they cannot find the precise information contained within the text.

Text Mining is one of the most complex analysis in the industry of analytics. The reason for this is that, while doing text mining, we deal with unstructured data. We do not have clearly defined observation and variables (rows and columns). Hence, for doing any kind of analytics, we need to first convert this unstructured data into a structured dataset and then proceed with normal modelling framework. In terms of what text analytics does, there are a lot of misconceptions. To many business users, text analytics is a box where unstructured text goes in and structured information magically comes out. Through the use of a complex taxonomy and algorithms, text analytics is essentially about making human information (text, documents, language) understandable by computers and a way to discover value in text. Text Analytics is a process that can analyze any type of unstructured text, extracting high-quality and relevant information that can drive further analysis and strategic decision making.

The opportunity cost of any business to ignore unstructured data is paramount in today's fierce competitive world. According to an IDC survey, unstructured data takes a lion's share in digital space and approximately occupies 80% by volume compared to only 20 for structured data. While the unstructured data is available in abundance, the number of software products and solutions that can accurately analyze the text, present insights in an understandable manner along with the ability to integrate such insights readily into other extant models that use numerical only data are rare. A lot of the challenges in this space arise from the fact that natural language provides the flexibility to convey exactly the same meaning in umpteen different ways, or worse, exactly the same statement in a different context may convey completely different meaning.

Traditional analysis technologies are not able to effectively handle unstructured data because they merely manage lists organized in databases, and, equally important, without understanding the meaning of the terms, they cannot find the precise information contained within the text. Machine learning algorithms designed to analyze numerical data exactly know the structure of numbers and they are pre-programmed to process the data with precision. In case of natural language, it gets very problematic. Dialects, jargon, misspellings, short forms, acronyms, colloquialism, grammatical complexities, mixing one or more languages in the same text are just some of the fundamental problems unstructured data poses. It makes extremely difficult to precisely analyze unstructured data in the same way we process structured data.

1.5 Proposed Solution

- We are currently analysing millions of customer comments (or other text data) at a speed no human can match and an accuracy level no computer algorithm can match. 80% of “big-data” is in text form, we help customer focused businesses harness the power of this data. We do it combining machine learning algorithms with human intelligence to leverage the best of both. Our distinctive approach has won recognition from NIST (a US federal technology agency), TiE, Pan-IIT, NASSCOM, Google, and Yelp.
- Our software as a service (SAAS) product that will allow clients to access our solution directly from their computers, and customize it to their needs. With this product, we can serve hundreds of enterprise clients. We build custom AI models to solve your specific business challenges. We collect data , wherever it resides and separate signal from the noise.
- Our platform is agnostic to data source and will gather feedback from your customers, curate enormous volumes of unstructured text, and design specific taxonomies to reflect each individual business question. As your competitive ecosystem evolves, our solution will evolve in tandem.
- Our platform is agnostic to data source and will gather feedback from your customers, curate enormous volumes of unstructured text, and design specific taxonomies to reflect each individual business question. As the competitive ecosystem evolves, our solution will evolve in tandem.
- Our service-driven approach ensures incredibly accurate and high-performing machine learning models, resulting in granular and actionable insights. Whether you are measuring NPS, CSAT or star ratings, the SetuServ platform will provide sentiment and weight for every contributing trait.

CHAPTER 2
WORKFLOW

2.1 Our Workflow

2.1.1 Overview

Text classification is a widely studied subject in the information science sector. It has been relevant ever since the origin of digital text documents. Text classification generally involves a multi-step process. Text classification process can be summarised in a simple graph as shown below :

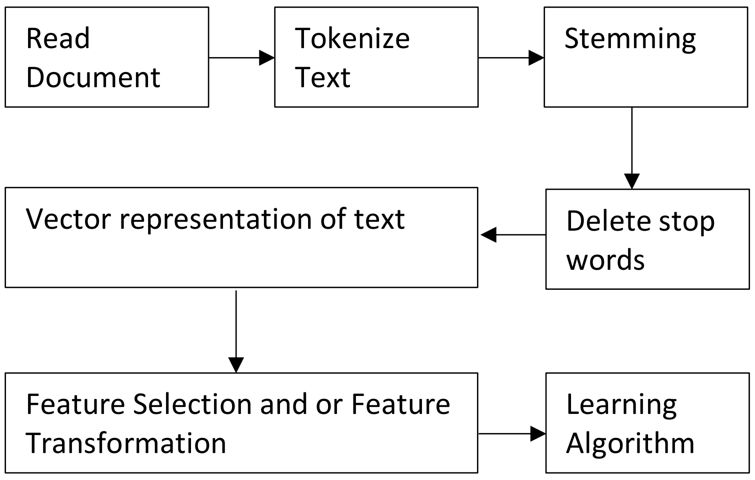

Read Document
Tokenize Text
Stemming
Delete stop words
Vector representation of text
Feature Selection and or Feature Transformation
Learning Algorithm

Figure 5: The text classification process

The text classification process can roughly be divided in two sections: pre-processing and classification. The pre-processing section roughly contains step one through six (see Figure 5). The main goal of the whole classification process is to assign a label to an event by evaluating its textual details. A text can be separated into a set of words, which we will use as features for classification. The performance of classification generally improves if the features are transformed and filtered, which reduces the size and sparsity of the feature set. The final step in Figure 5 is classification. In this step, a classifier is used to predict a label for unlabeled events. A model is built based on labeled training data and this model will predict the label of unlabeled, previously unseen test data. The difference in the size of these two sections might appear unusual as the actual classification seems to be an important part of the whole process. However, the performance of a classifier is mainly determined by the quality of the used feature set.

2.1.2 Data Wrangling and Cleaning

Data Wrangling is the very first step to any of the Data Analysis platforms. It is the most basic need of a data scientist. To perform an analysis using Machine Learning and Deep Learning techniques, you need to have a very good amount of data. Since, our company deals with fetching insights from customer reviews, the very first step is to scrape web for reviews.

Different sources are searched for data scraping , like product review sites , or social networking sites etc. Data scraped can vary from a few hundred to million reviews.

Web Scraping is a technique employed to extract large amounts of data from websites whereby the data is extracted and saved to a local file in your computer or to a database in table (spreadsheet) format.

Data displayed by most websites can only be viewed using a web browser. They do not offer the functionality to save a copy of this data for personal use. The only option then is to manually copy and paste the data - a very tedious job which can take many hours or sometimes days to complete. Web Scraping is the technique of automating this process, so that instead of manually copying the data from websites, the web scraping code will perform the same task within a fraction of the time.

- Libraries used for data scraping:
  - **Beautiful Soup**

    Beautiful Soup is a Python library for pulling data out of HTML and XML files. It works with your favourite parser to provide idiomatic ways of navigating, searching, and modifying the parse tree. It commonly saves programmers hours or days of work.
  - **Scrapy**

    It is not a library , but a complete framework with many functionalities for data crawling. Scrapy is a fast high-level web crawling and web scraping framework, used to crawl websites and extract structured data from their pages. It can be used for a wide range of purposes, from data mining to monitoring and automated testing.

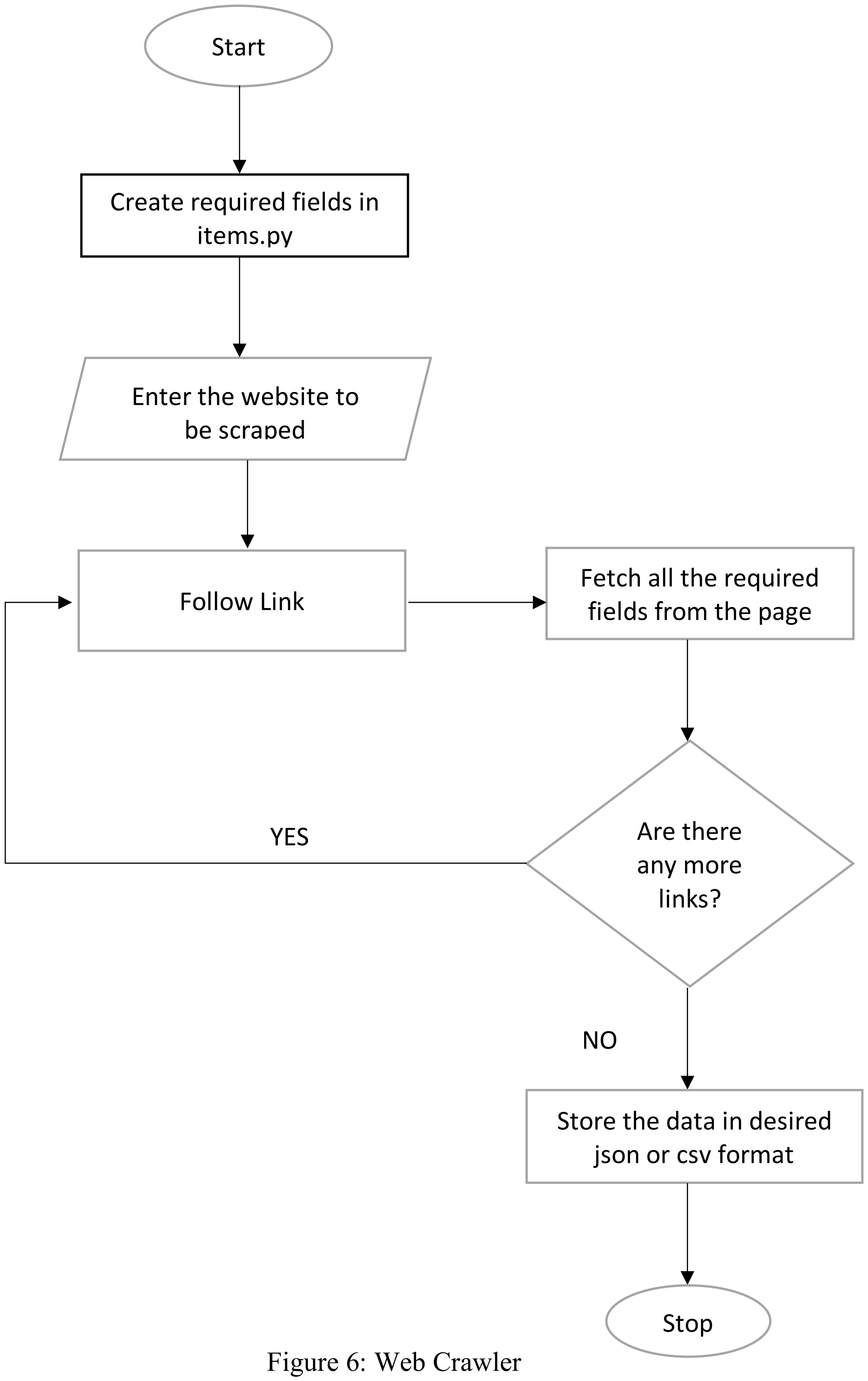

Start
Create required fields in items.py
Enter the website to be scraped
Follow Link
Fetch all the required fields from the page
Are there any more links?
YES
NO
Store the data in desired json or csv format
Stop

Figure 6: Web Crawler

2.1.3 Data Pre-processing

It is the next step that comes after Data Wrangling. Not all the data can be fed to the Machine Learning or Deep Learning models, as it will lead to improper training of the model, and it might start learning from the noise in the text. So, the noise from the text has to be removed before further processing.

Data pre-processing is a data mining technique that involves transforming raw data into an understandable format.

Real world data is often incomplete, inconsistent, and/or lacking in certain behaviours or trends, and is likely to contain many errors. Data pre-processing is a proven method of resolving such issues. Data pre-processing prepares raw data for further processing. Data pre-processing is used database-driven applications such as customer relationship management and rule-based applications (like neural networks).

Data goes through a series of steps during pre-processing:

- **Data Cleaning:** Data is cleaned through processes such as filling in missing values, smoothing the noisy data, or resolving the inconsistencies in the data.
- **Data Integration:** Data with different representations are put together and conflicts within the data are resolved.
- **Data Transformation:** Data is normalized, aggregated and generalized.
- **Data Reduction:** This step aims to present a reduced representation of the data in a data warehouse.
- **Data Discretization:** Involves the reduction of a number of values of a continuous attribute by dividing the range of attribute intervals.

Following techniques are used for data pre-processing :

- **Stopwords :** Not always does this happen that removing stopwords gives us the optimal results. Sometimes, the data with stopwords is more helpful.—doubt
- **Stemming and lemmatization :** The documents consist of different forms of verb as per the requirement, such as write, writes, and writing.

The goal of both stemming and lemmatization is reducing inflectional forms and sometimes the derivationally related forms of word like : democracy, democratic and democratization, and bringing the word down to a common base form.

However, stemming and lemmatization are not all the same.

Stemming is the crude heuristic process that chops off the ends of words, and often removes derivational affixes.

Lemmatization on the other hand makes use of vocabulary and morphological word analysis. This normally removes inflectional endings only and retuns the base/dictionary form of the word, known as lemma.

For eg: “saw”

Stemming : may return “s”

Lemmatization : will return “see” or “saw” depending on whether the word was used as a verb or noun.

- **Punctuation :**

  Removing punctuation is always a good idea. Bad quality of data can lead to bad results. Punctuations only matter when we are concerned with the sentiment of the data, in all other cases punctuation creates a trouble and makes it difficult for the model to learn. So, punctuation removal is one of the most basic steps in Data Cleaning. This can be done by directly using NLTK, or if you don’t want to remove all the punctuations, then a list of punctuations to be removed can be generated, and accordingly the punctuations can be removed.

- **Topic Modelling :**

  Another such technique used is **Topic Modelling** using **Non-negative matrix factorization. NMF** is a group of algorithms where a matrix M is factorised into two matrices W and H with the property that all three matrices have no negative elements. The aim of each algorithm is then to produce 2 smaller matrices; a document to topic matrix and a word to topic matrix that when multiplied together reproduce the bag of words matrix with the lowest error. Hence, top words for certain number of topics (determined by trial and error) are given to the team for creating taxonomies.

The team now randomly sample data for creating taxonomies, which after a feedback loop of 3 consecutive runs is finalized with the client. The client is also involved in the process because they might also want to capture certain specific features of product.

The tagging process now begins. The data that machine learning algorithm trains on can be upto few thousands. This is how the training data for ML to learn is obtained.

2.1.4 Converting Text to Vector Form

Making a vector representation of a text means transforming the feature set of tokens into a dictionary where numbers represent words. A classifier can only work with numbers and not with the textual representation of a word.

TF-IDF Vectoriser –

Word counts are a good starting point, but are very basic. One issue with simple counts is that some words like "*the*" will appear many times and their large counts will not be very meaningful in the encoded vectors.

An alternative is to calculate word frequencies, and by far the most popular method is called TF-IDF. This is an acronym than stands for "*Term Frequency – Inverse Document*" Frequency which are the components of the resulting scores assigned to each word.

- **Term Frequency**: Suppose we have a set of English text documents and wish to rank which document is most relevant to the query, "the brown cow". A simple way to start out is by eliminating documents that do not contain all three words "the", "brown", and "cow", but this still leaves many documents. To further distinguish them, we might count the number of times each term occurs in each document; the number of times a term occurs in a document is called its *term frequency*. The weight of a term that occurs in a document is simply proportional to the term frequency. The various methods of encoding term frequency are described below -
- **Inverse Document Frequency**: Because the term "the" is so common, term frequency will tend to incorrectly emphasize documents which happen to use the word "the" more frequently, without giving enough weight to the more meaningful terms "brown" and "cow". The term "the" is not a good keyword to distinguish

relevant and non-relevant documents and terms, unlike the less-common words "brown" and "cow". Hence an *inverse document frequency* factor is incorporated which diminishes the weight of terms that occur very frequently in the document set and increases the weight of terms that occur rarely. The specificity of a term can be quantified as an inverse function of the number of documents in which it occurs.

- **Term Frequency – Inverse Document Frequency** – tf-idf is calculated as

  $$\text{tfidf}(t,D) = \text{tf}(t,D) * \text{idf}(t,D)$$

  A high weight in tf–idf is reached by a high term frequency (in the given document) and a low document frequency of the term in the whole collection of documents; the weights hence tend to filter out common terms. Since the ratio inside the idf's log function is always greater than or equal to 1, the value of idf (and tf-idf) is greater than or equal to 0. As a term appears in more documents, the ratio inside decreases.

2.1.5 Sequence to Sequence Model

The Youtube data is a rich source of reviews, which can be further used for product analysis. The crawled YouTube data mostly is the data is without exclamation marks, periods, interrogation, hyphen, semicolon or any other punctuation mark. There comes into role, our sequence to sequence model.

Sequence to Sequence (Seq2Seq) model refers to training models in such a way, so as to convert sequences from one domain (eg. English sentences) to sequences in another domain (eg. Same sentences in French)

For eg: How are you doing ? → [Seq2Seq Model] → Comment allez vous ?

It is also used in Neural Machine Translation or for question-answer machine (i.e. generating a sequence of Natural Language answer on providing a sequence of Natural Language Question). In short, it is helpful every time you want to generate a sequence corresponding to another sequence.
It can be formed in various ways : character to character ,word to word.

Encoding Scheme tried :

1. One – hot Encoding
2. Word Vector Encoding

There are multiple ways of creating Seq2Seq model :

1. Using RNNs :

   The RNN can easily map sequences to sequences whenever the alignment between the inputs the outputs is known ahead of time. However, it is not clear how to apply an RNN to problems whose input and the output sequences have different lengths with complicated and non-monotonic relationships
2. Using LSTM(Long Short-Term Memory) :

   The Long Short-Term Memory (LSTM) is known to learn problems with long range temporal dependencies, so an LSTM may succeed in this setting(hence, we are preferring LSTM over RNN for punctuation prediction)

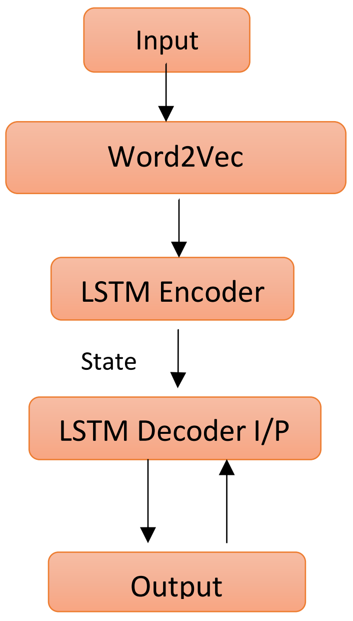


Figure 7: Seq2Seq using LSTM

Working : when we want to decode unknown input sequences, we:

- Encode the input sequence into state vectors

- Start with a target sequence of size (just the start-of-sequence character)
- Feed the state vectors and 1-char target sequence to the decoder to produce predictions for the next character
- Sample the next character using these predictions (we simply use argmax).
- Append the sampled character to the target sequence
- Repeat until we generate the end-of-sequence character or we hit the character limit.

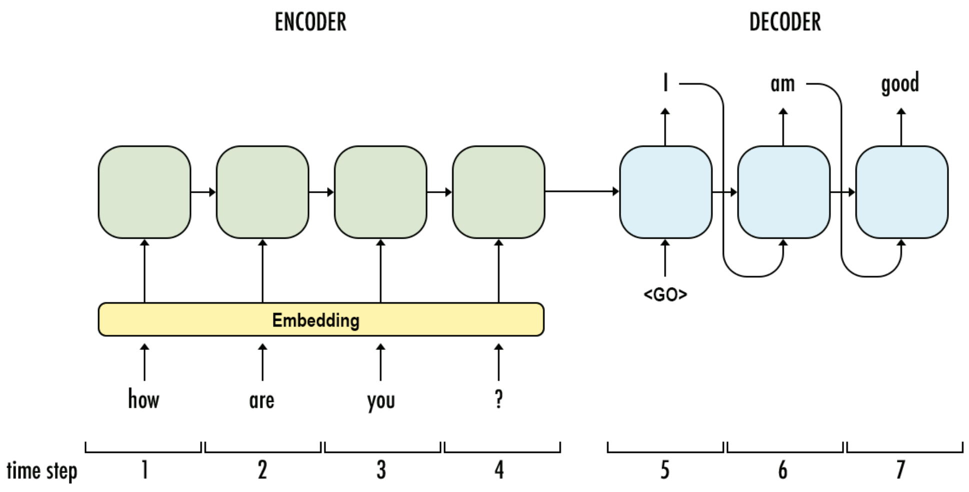


Figure 8: Input and Output of Seq2Seq model

Detailed Steps of Seq2Seq :

- Encode the input sequence into state vectors.
- Starting with the fixed input sequence (just a symbol/word for the start-of-sequence)
- Feed the state vectors and 1-word target sequence to the decoder to predict the next word in the sequence.
- Sample the next character using these predictions.
- Then, add the sampled word to the target sequence.
- This process is repeated until we generate the end-of-sequence word or we hit the character – limit.

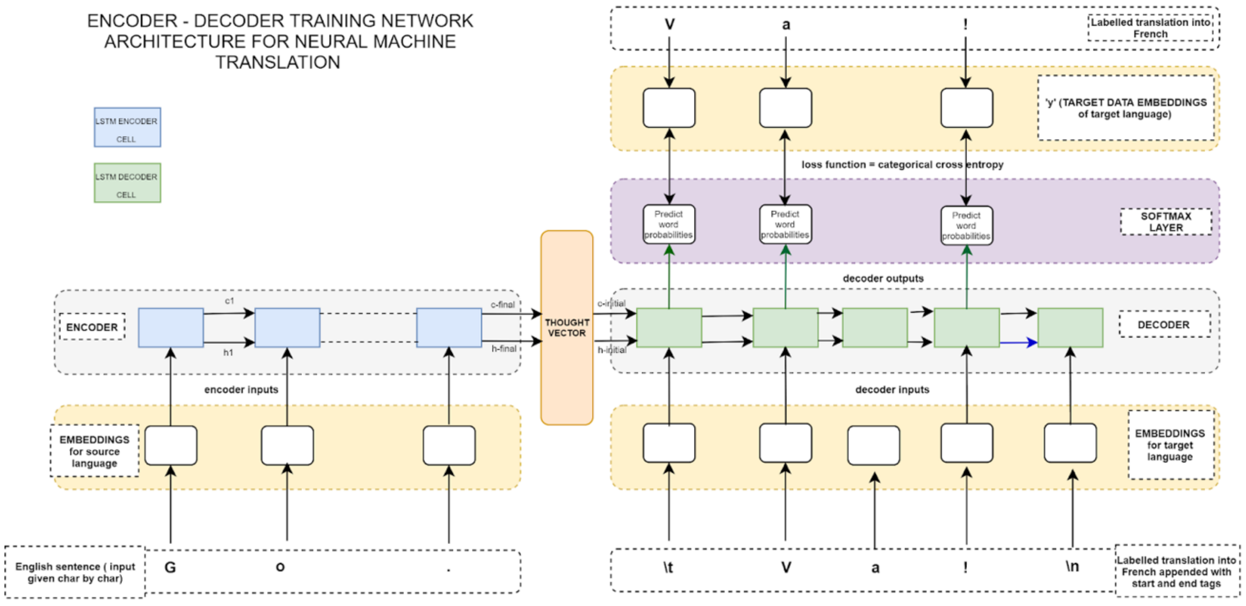

ENCODER - DECODER TRAINING NETWORK ARCHITECTURE FOR NEURAL MACHINE TRANSLATION
LSTM ENCODER CELL
LSTM DECODER CELL
V
a
!
Labelled translation into French
'y' (TARGET DATA EMBEDDINGS of target language)
loss function = categorical cross entropy
Predict word probabilities
SOFTMAX LAYER
decoder outputs
c1
h1
c-final
h-final
THOUGHT VECTOR
c-initial
h-initial
ENCODER
DECODER
encoder inputs
decoder inputs
EMBEDDINGS for source language
EMBEDDINGS for target language
English sentence (input given char by char)
G
o
.
\t
V
a
!
\n
Labelled translation into French appended with start and end tags

Figure 9: LSTM Encoder-Decoder Sequence

2.2 Analysing the Dataset

Analysing the dataset and fetching insights from the data is important for further processing.

2.2.1 Theme Classification: Keyword Pair Generation

The first thing that is to be done as soon as the data is available, is finding the subcategories in which our data is spread. Then, on the basis of these subheadings, the data is further divided into smaller chunks. This is done by creating dependencies for further theme creation.

So, the dependencies are generated using Spacy's "English" model. Spacy is also used for POS tagging in the sentences. The most commonly occurring keyword pairs are collected from the data.

The Flow:

To create the dependencies, first nlp is called on the text. Whenever nlp is called on the text, spaCy tokenises the text and produces a Doc object. This Doc object is then processed in several steps, also referred as **Processing Pipeline**. Pipeline used in default models contains a tagger, an entity recognizer and a parser. The pipeline components return the processed Doc, which then is passed to the next component.

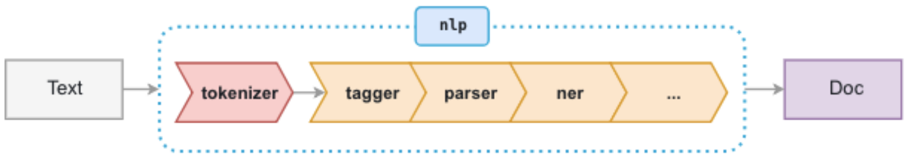


Figure 10: Processing Pipeline

Tokenizer : Segments text into tokens

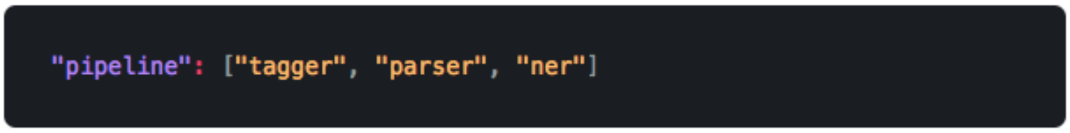
Tagger : Assigns POS(Part-of-speech) tags

Parser : Assigns dependency labels

Ner : Detects and labels named entities

The processing pipeline is dependent on the statistical model and its capabilities. Eg. ER(Entity Recognizer) can only be included in the component if model includes data to make predictions of entity labels. Therefore, the pipeline is specific to each model.

Specifying a pipeline :



```
"pipeline": ["tagger", "parser", "ner"]
```

Figure 11: Initializing Processing Pipeline

After successfully generating sentences, dependency parsing is applied over the sentences to extract the most common pairs. **Syntactic Parsing** or **Dependency Parsing** is the task of recognizing a sentence and assigning a syntactic structure to it. The most widely used syntactic structure is the parse tree which can be generated using some parsing algorithms. The following figure shows one such example.

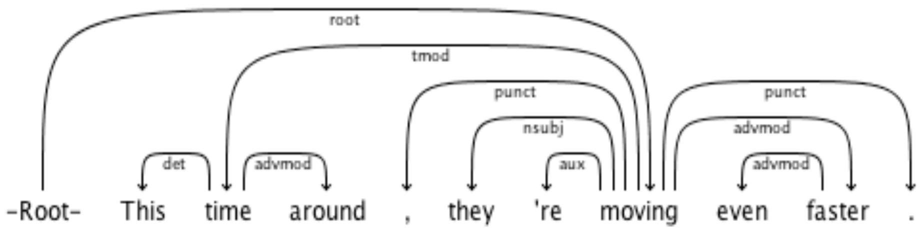


Figure 12: Dependency Pairs

The most common ancestor and child pairs are separated out. Our idea is that these pairs can help the team to get some common themes/classes (taxonomy).

2.2.2 Brand Name Detection : Word2Vec

Every firm in the market wants to know whether or not their customers like them, what all are the flaws that they need to work on and the most important : "**Who is their competition in the market ?**" These are the major things that every company wants to analyse in order to expand their business and keep going. Here comes into roll the part of Brand Name Detection.

So, fetching out all the brand names from the reviews data and then analysing reviews company-wise is an important thing that can help our customer know about their market struggle.

Word2Vec helps in finding all the words with similar vector, which in turn can help us coming up with the brand names users have mentioned in the reviews.

Further, the data will be segregated based on the Brands and analysed. This will help the company know, what the consumers feel about the similar products from other brands and how is their product superior or inferior from the other.

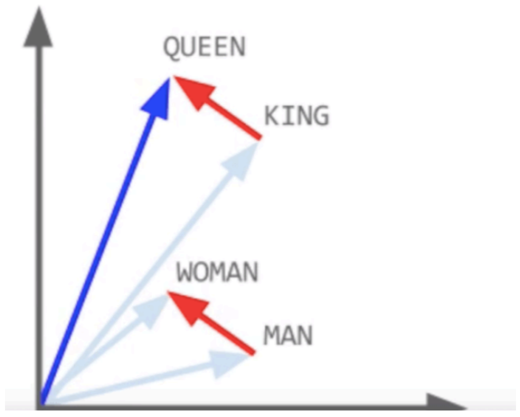


Figure 13: Vector Representation of Words

Workflow :

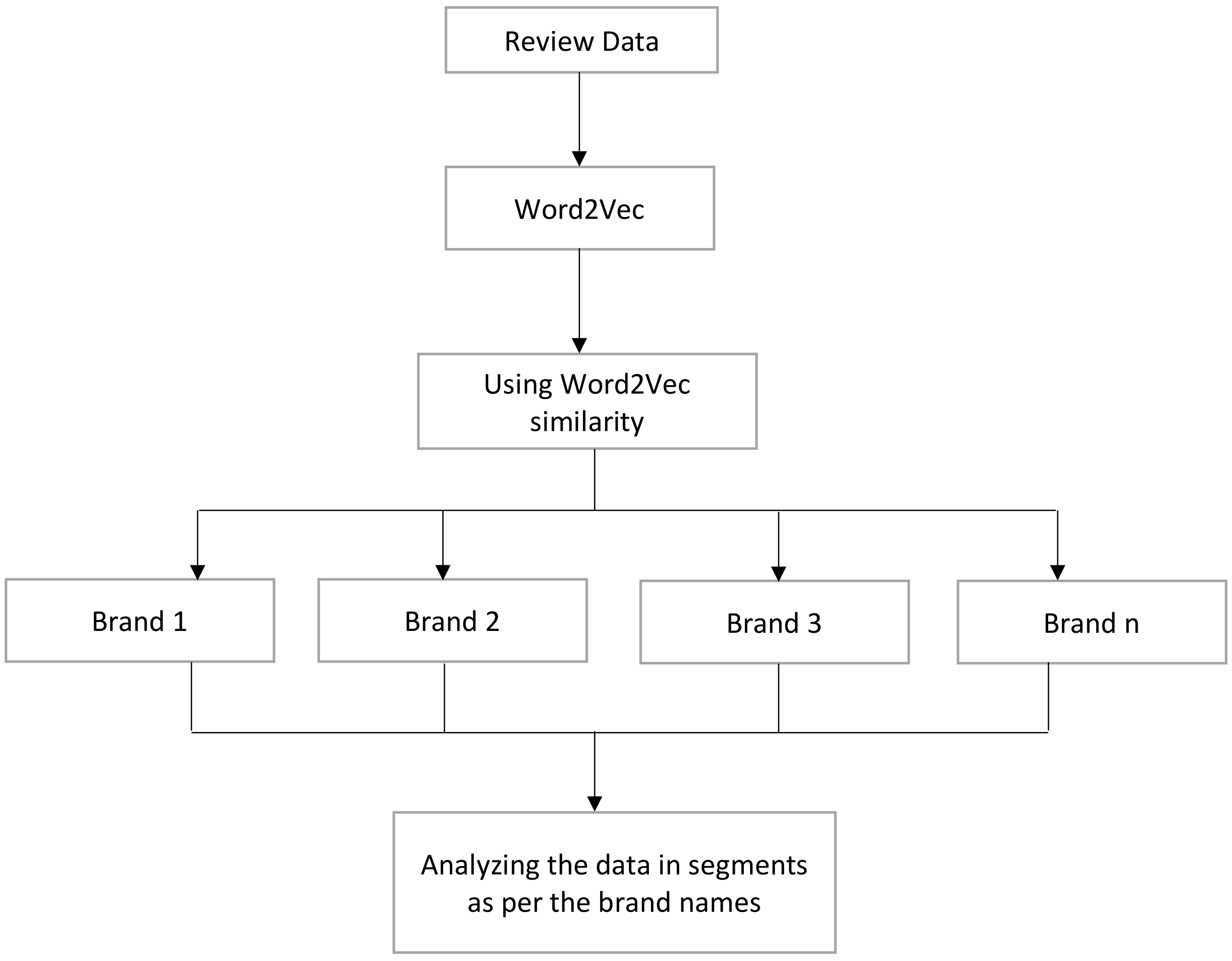

Review Data
Word2Vec
Using Word2Vec similarity
Brand 1
Brand 2
Brand 3
Brand n
Analyzing the data in segments as per the brand names

Figure 14: Using Word2Vec Model

2.3 Running Machine Learning Baseline

There are many machine learning algorithms to choose from. Hundreds in fact. We must know whether the predictions for a given algorithm are good or not. But how do we know?

The answer is to use a baseline prediction algorithm. A baseline prediction algorithm provides a set of predictions that you can evaluate as you would any predictions for your problem, such as classification accuracy. The scores from these algorithms provide the required point of comparison when evaluating all other machine learning algorithms on your problem. Once established, we can comment on how much better a given algorithm is as compared to the naive baseline algorithm, providing context on just how good a given method actually is. In our case we are using logistic regression as a baseline predictor method. Logistic regression

When starting on a new problem if the desired results are obtained using logistic regression model is kept as it is otherwise more advanced methods like deep learning techniques are used.

To find the best method suitable for the data, we apply different operations on the data and use the method with most optimal results. Here is the workflow of how different models are used for training the data :

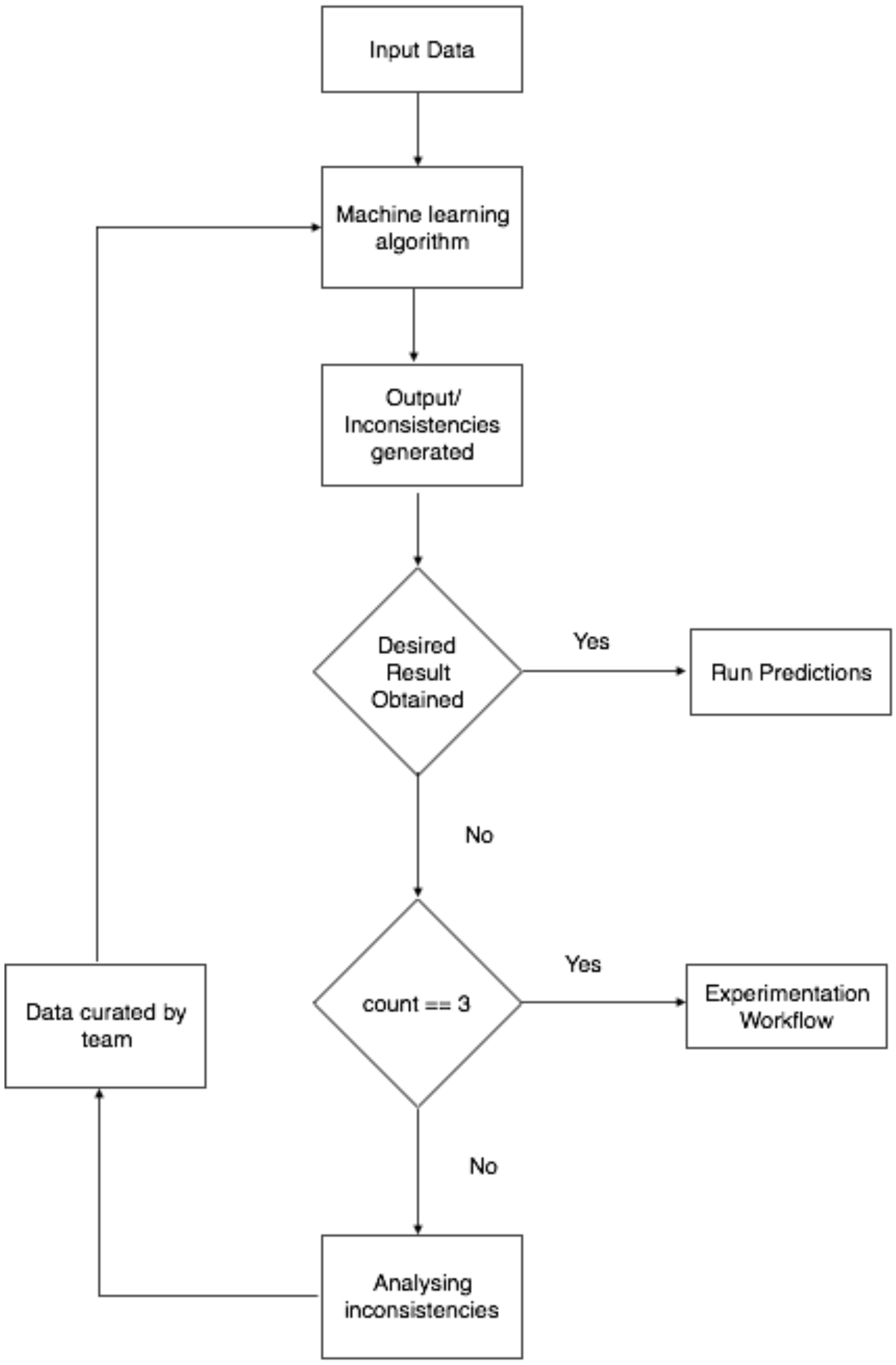


Fig 15: Machine Learning Workflow

2.3.1 Using Logistic Regression

Logistic Regression is the baseline model used for any dataset. For most of the datasets LR gives positive and satisfactory results. This is the most generic and basic model that can be used on a variety of datasets.

Logistic regression is a statistical method for analysing a dataset in which there are one or more independent variables that determine an outcome. The outcome is measured with a dichotomous variable (in which there are only two possible outcomes).

In logistic regression, the dependent variable is binary or dichotomous, i.e. it only contains data coded as 1 (TRUE, success, pregnant, etc.) or 0 (FALSE, failure, non-pregnant, etc.).

The goal of logistic regression is to find the best fitting (yet biologically reasonable) model to describe the relationship between the dichotomous characteristic of interest (dependent variable = response or outcome variable) and a set of independent (predictor or explanatory) variables. Logistic regression generates the coefficients (and its standard errors and significance levels) of a formula to predict a *logit transformation* of the probability of presence of the characteristic of interest:

$$logit(p) = b_0 + b_1X_1 + b_2X_2 + b_3X_3 + ... + b_kX_k$$

where p is the probability of presence of the characteristic of interest.

The logit transformation is defined as the logged odds:

$$odds = \frac{p}{1-p} = \frac{probability\ of\ presence\ of\ characteristic}{probability\ of\ absence\ of\ characteristic}$$

and

$$logit(p) = \ln\left(\frac{p}{1-p}\right)$$

Rather than choosing parameters that minimize the sum of squared errors (like in ordinary regression), estimation in logistic regression chooses parameters that maximize the likelihood of observing the sample values.

SciKit Learn support Logistic Regression :

class sklearn.linear_model.**LogisticRegression**(*penalty='l2'*, *dual=False*, *tol=0.0001*, *C=1.0*, *fit_intercept=True*, *intercept_scaling=1*, *class_weight=None*, *random_state=None*, *solver='liblinear'*, *max_iter=100*, *multi_class='ovr'*, *verbose=0*, *warm_start=False*, *n_jobs=1*)

2.3.2 Using Deep Learning Methods

Whenever the baseline fails in giving satisfactory results, we move towards advance Deep Learning Models. The larger the dataset, the better the deep learning models perform. There is an additional feature of early stopping in Deep Learning Models (whenever the models stops learning or starts overfitting, the training is interrupted).
Various Deep Learning Models are experimented with and the one that best fits is used.
Various Deep Learning Models have been explained in the further chapters.

2.4 Analysing Output/Inconsistencies

After all the steps of Data Scraping, Data Cleaning, running the machine learning models, the next thing that comes into role is Analysing. Every result has to be analysed before passing on to the client, and it must be taken care that the results are as promised to the client. If the results are satisfactory, then the output is deployed on the server. If not, then the inconsistencies are taken a look at and further rounds of inconsistencies are run and loop continues 3-4 times. Very rarely does this happen that the results are as per promised on the first round only, because the kind of data for every project varies, and so do the models and their parameters required.
If the model is still not performing upto the mark, then we switch to different Machine Learning/ Deep Learning Algorithms.

The Analyzation workflow has been described below in Figure 14.

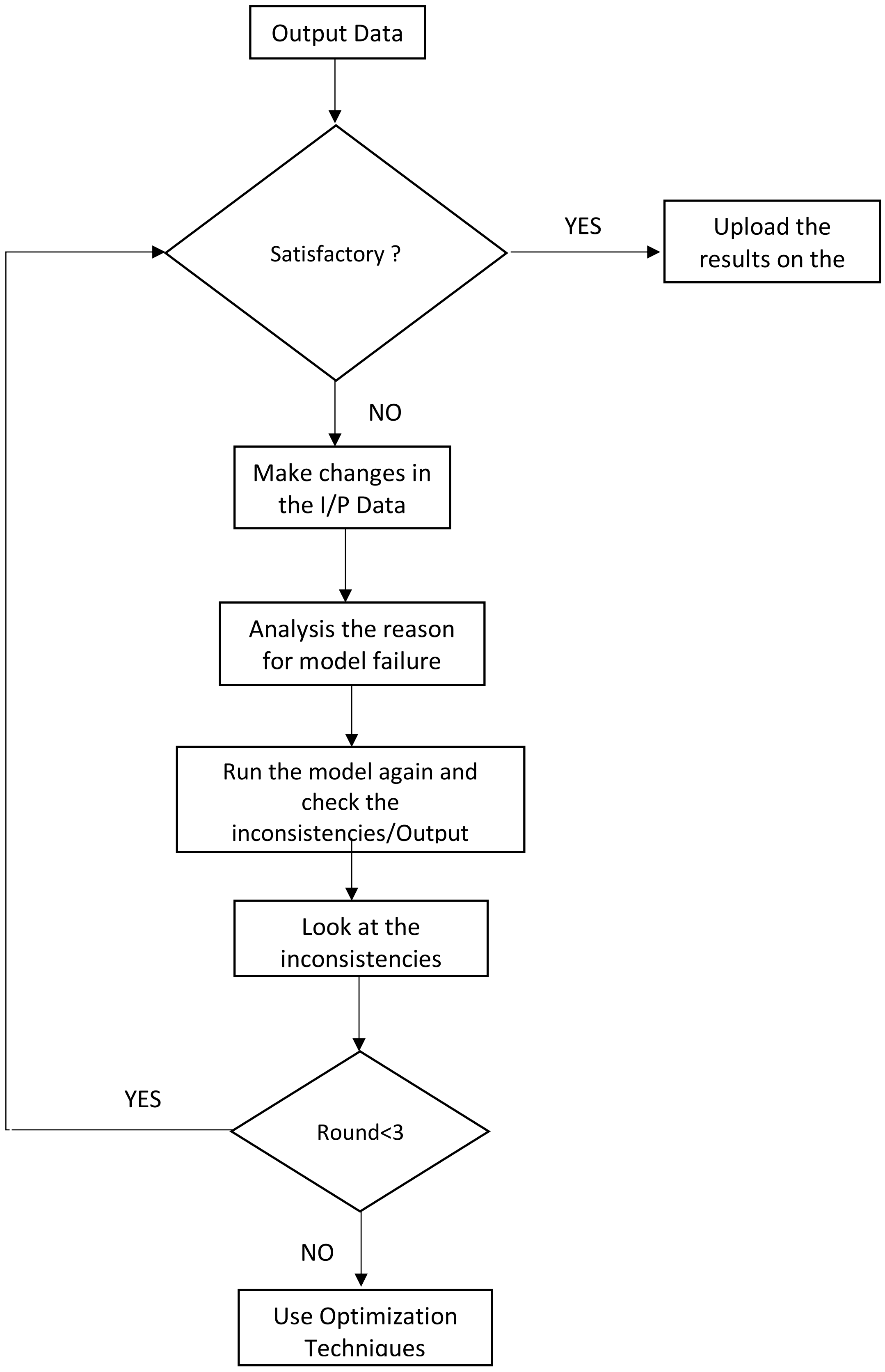

Output Data
Satisfactory ?
YES
Upload the results on the
NO
Make changes in the I/P Data
Analysis the reason for model failure
Run the model again and check the inconsistencies/Output
Look at the inconsistencies
YES
Round<3
NO
Use Optimization Techniques

Figure 16: Analyzation Workflow

2.5 Optimization Techniques

2.5.1 Generating/ Refining Inconsistencies in Loop

To improve the performance of the model, we need to remove the ambiguities from the data. This is done by analysing the output inconsistencies. All those inconsistencies are taken care of, and then the model is again put into the loop. The check on output is made for the themes that are not being learnt by the model, and the reason for the same is analysed (whether it is because of the quality of the input data, or maybe the model parameters have to adjusted, or maybe the tagging somewhere went wrong, or how the model is picking up the top features from the data). After analysing all these possibilities and making all possible changes the performance mostly improves. If still the model does not improve, then we switch to different Deep Learning Models, and try various other optimization techniques mentioned in the further section of the report.

2.5.2 N-Gram Generation

If unigrams are not performing good, we need to experiment with the word structures and try different word representations like n-grams and then again analyse the difference it brings in the results. Also, these n-gram structures can be used in Word2Vec and various other models.

***n*-gram** is a contiguous sequence of *n* items from a given sample of text or speech. The items can be phonemes, syllables, letters, words or base pairs according to the application. The *n*-grams typically are collected from a text or speech corpus. When the items are words, *n*-grams may also be called *shingles.*

Using Latin numerical prefixes, an *n*-gram of size 1 is referred to as a "unigram"; size 2 is a "bigram" (or, less commonly, a "digram"); size 3 is a "trigram". English cardinal numbers are sometimes used, e.g., "four-gram", "five-gram", and so on. In computational biology, a polymer or oligomer of a known size is called a *k*-mer instead of an *n*-gram, with specific names using Greek numerical prefixes such as "monomer", "dimer", "trimer", "tetramer", "pentamer", etc., or English cardinal numbers, "one-mer", "two-mer", "three-mer", etc.

- Method used : Gensim n-grams

Gensim is a robust open-source vector space modelling and topic modelling toolkit implemented in Python. It uses NumPy, SciPyand optionally Cython for performance. Gensim is specifically designed to handle large text collections, using data streaming and efficient incremental algorithms, which differentiates it from most other scientific software packages that only target batch and in-memory processing.

Gensim Phrasers automatically detect common phrases (multiword expressions) from stream of sentences .The Phrasers are collocations (frequently co-occurring tokens).

phrases = Phrases(sentence)

bigram = Phraser (phrases)

trigram = Phrases (bigram[sentence])

and so on….

Adding to vocabulary (dynamically updating the bigram model) :

bigram.add_vocab (new_sentences)

2.5.3 Custom Word2Vec

It sweeps through sentences in online fashion, handling each co-occurrence separately. Custom Word2Vec refers to feeding the Word2Vec embeddings of a similar dataset as an input, so as to improve the existing CNN model.

2.5.4 Google Word2Vec

If the custom Word2Vec is not performing good, then we go for Google Word2Vec. Google w2v includes vectors for vocab of around 3 million words and phrases trained on around 100 billion of words from Google News dataset. The vector length is 300 implying the number of features.

If we load this model, it prevents the resources and time invested in training custom Word2Vec. Since the vocab size is so large, it covers diverse cases in the data.

2.5.5 Glove

It is unsupervised learning algorithm to obtain vector representations of words. Training is done on aggregated global word – word co-occurrence stats from a corpus of data, the resulting representations show interesting linear substructues of word-vector space.

It precomputes word x word co-occurrence matrix in the memory and quickly factorizes it.

Glove is trained on non-zero entries of global word-word-co-ocuurence matrix. It tabulates the frequency with which the words are occurring together in the given corpus. So, one has to make single pass through the entire corpus in order to populate the matrix. If the corpora is large, this can be computationally expensive, though it is one-time cost only. Subsequent iterations are fast because it is a Sparse Matrix.

GloVe is log-bilinear model with weighted least-squares objective. The main assumption behind the model is that the ratios of word-word co-occurrence probabilities have potential to encode some form of semantic. There is a tradeoff between taking more memory (**GloVe**) **vs**. taking longer to train (**word2vec**).

2.5.6 Skip-gram Model

In plain English, whereas n-grams are *consecutive* word patterns, such as bigrams, trigrams, four-grams, etc., skip-grams are not entirely consecutive but contain gaps or spaces, thus skipping over words. Skip-gram models are *statistical* language models, which may be used for purposes such as prediction. Skip-gram models would be more *fuzzy* than regular n-gram models, but also more *data intensive*.

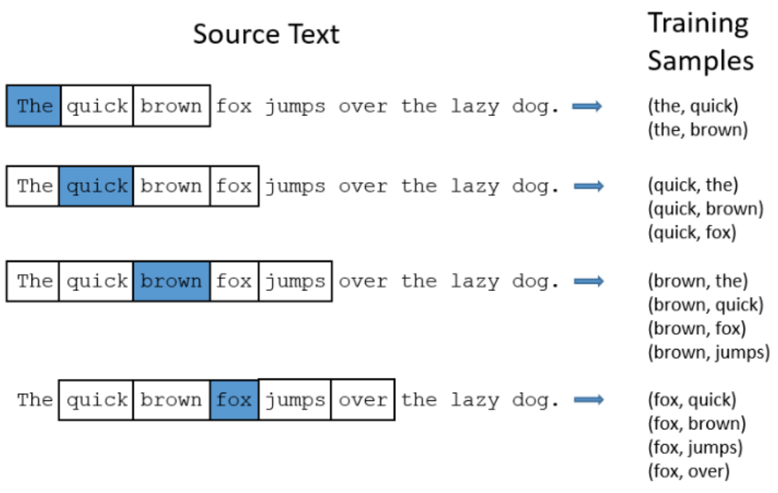


Figure 17: Skipgram

Skipgrams are usually beneficial when we are trying to fetch the top features from the dataset for any particular class. It helps fetch the non-consecutive top-features that have contributed the most in the training of the particular class, and hence the data is not biased.

2.5.7 CNN

Convolutional Neural Networks take advantage of the fact that the input consists of images and they constrain the architecture in a more sensible way. In particular, unlike a regular Neural Network, the layers of a ConvNet have neurons arranged in 3 dimensions: **width, height, depth**. (Note that the word *depth* here refers to the third dimension of an activation volume, not to the depth of a full Neural Network, which can refer to the total number of layers in a network.) For example, the input images in CIFAR-10 are an input volume of activations, and the volume has dimensions 32x32x3 (width, height, depth respectively). As we will soon see, the neurons in a layer will only be connected to a small region of the layer before it, instead of all of the neurons in a fully-connected manner. Moreover, the final output layer would for CIFAR-10 have dimensions 1x1x10, because by the end of the ConvNet architecture we will reduce the full image into a single vector of class scores, arranged along the depth dimension. Here is a visualization:

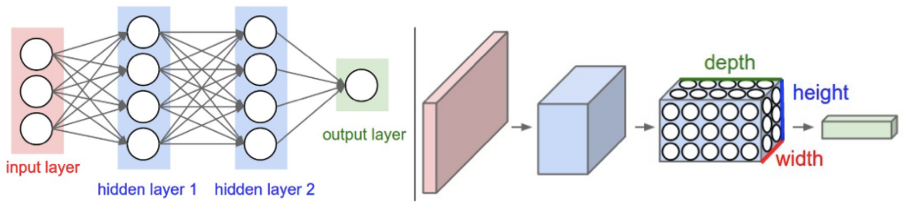


Figure 18: CNN Layers

Figure Description

Left: A regular 3-layer Neural Network. Right: A ConvNet arranges its neurons in three dimensions (width, height, depth), as visualized in one of the layers. Every layer of a ConvNet transforms the 3D input volume to a 3D output volume of neuron activations. In this example, the red input layer holds the image, so its width and height would be the dimensions of the image, and the depth would be 3 (Red, Green, Blue channels).

Layers used to Build ConvNets

As we described above, a simple ConvNet is a sequence of layers, and every layer of a ConvNet transforms one volume of activations to another through a differentiable function. We use three main types of layers to build ConvNet architectures: **Convolutional Layer**, **Pooling Layer**, and **Fully-Connected Layer** (exactly as seen in regular Neural Networks). We will stack these layers to form a full ConvNet **architecture**.

Example Architecture: Overview. We will go into more details below, but a simple ConvNet for CIFAR-10 classification could have the architecture [INPUT - CONV - RELU - POOL - FC]. In more detail:

- **INPUT** [32x32x3] will hold the raw pixel values of the image, in this case an image of width 32, height 32, and with three color channels R,G,B.
- **CONV** layer will compute the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and a small region they are connected to in the input volume. This may result in volume such as [32x32x12] if we decided to use 12 filters.
- **RELU** layer will apply an elementwise activation function, such as the max(0,x)max(0,x) thresholding at zero. This leaves the size of the volume unchanged ([32x32x12]).
- **POOL** layer will perform a downsampling operation along the spatial dimensions (width, height), resulting in volume such as [16x16x12].
- **FC** (i.e. fully-connected) layer will compute the class scores, resulting in volume of size [1x1x10], where each of the 10 numbers correspond to a class score, such as among the 10 categories of CIFAR-10. As with ordinary Neural Networks and as the name implies, each neuron in this layer will be connected to all the numbers in the previous volume.

In this way, ConvNets transform the original image layer by layer from the original pixel values to the final class scores. Note that some layers contain parameters and other don’t. In particular, the CONV/FC layers perform transformations that are a function of not only the activations in the input volume, but also of the parameters (the weights and biases of the neurons). On the other hand, the RELU/POOL layers will implement a fixed function. The parameters in the

CONV/FC layers will be trained with gradient descent so that the class scores that the ConvNet computes are consistent with the labels in the training set for each image.

In summary:

- A ConvNet architecture is in the simplest case a list of Layers that transform the image volume into an output volume (e.g. holding the class scores)
- There are a few distinct types of Layers (e.g. CONV/FC/RELU/POOL are by far the most popular)
- Each Layer accepts an input 3D volume and transforms it to an output 3D volume through a differentiable function
- Each Layer may or may not have parameters (e.g. CONV/FC do, RELU/POOL don't)
- Each Layer may or may not have additional hyperparameters (e.g. CONV/FC/POOL do, RELU doesn't)

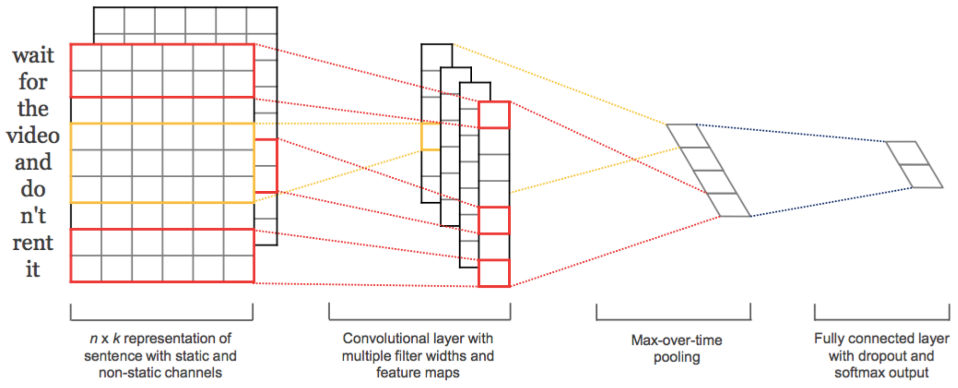

wait
for
the
video
and
do
n't
rent
it
n x *k* representation of sentence with static and non-static channels
Convolutional layer with multiple filter widths and feature maps
Max-over-time pooling
Fully connected layer with dropout and softmax output

Figure 19: Layer Representation in CNN

Model :

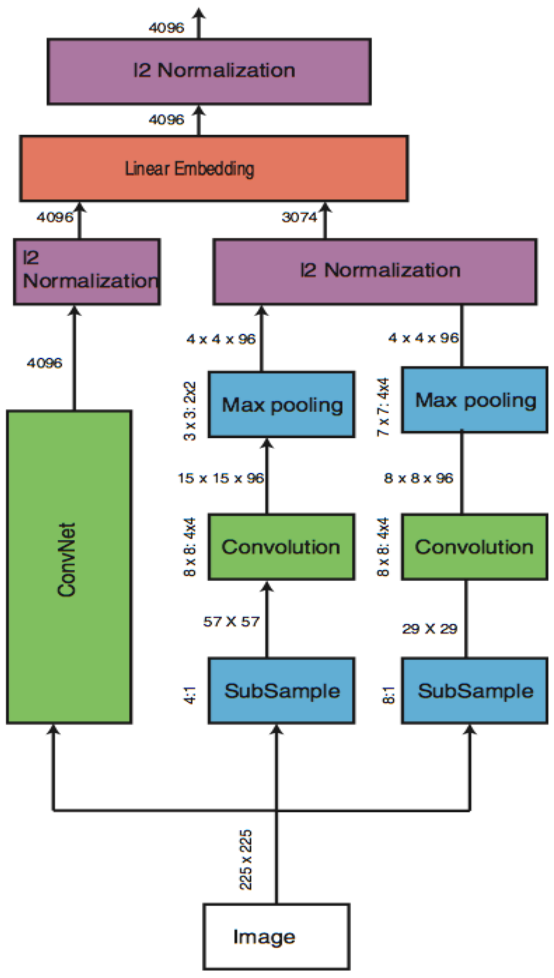


Figure 20: Describing the ConvNet Network

Input : We provide word2vec embeddings as the input layer to the model. The data is reshaped according to the input shape of the 2d-covnet. Eg: reshaping the I/P data from (x_train x 784) to (x_train x 28 x 28).

A convolution operation involves which is applied to a window of words to produce new feature. This function is applied to each possible word window in the sentence and produces a feature map.

2D Convolution : The main operation in CNN.

Then, we apply max-over-time pooling operation

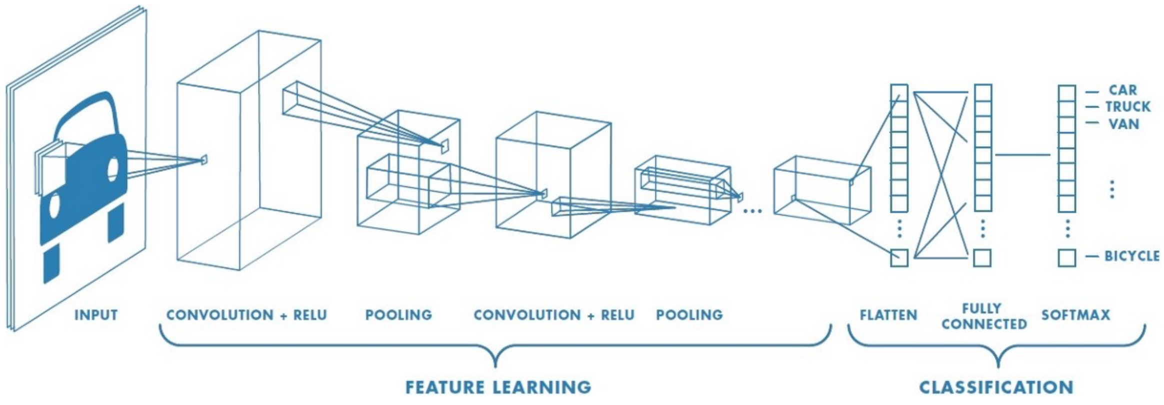


Figure 21: CNN

2.5.8 LSTM

Long short-term memory (**LSTM**) units (or blocks) are a building unit for layers of RNN. A RNN composed of LSTM units is often called an LSTM network. A common LSTM unit is composed of a **cell**, an **input gate**, an **output gate** and a **forget gate**. The cell is responsible for "remembering" values over arbitrary time intervals; hence the word "memory" in LSTM. Each of the three *gates* can be thought of as a "conventional" artificial neuron, as in a multi-layer (or feedforward) neural network: that is, they compute an activation (using an activation function) of a weighted sum. Intuitively, they can be thought as *regulators* of the flow of values that goes through the connections of the LSTM; hence the denotation "gate". There are connections between these gates and the cell.

The expression *long short-term* refers to the fact that LSTM is a model for the *short-term memory* which can last for a *long* period of time. An LSTM is well-suited to classify, process and predicttime series given time lags of unknown size and duration between important events. LSTMs were developed to deal with the exploding and vanishing gradient problem when training traditional RNNs. Relative insensitivity to gap length gives an advantage to LSTM over alternative RNNs, hidden Markov models and other sequence learning methods in numerous applications.

2.5.9 BiLSTM

Bidirectional LSTMs are an extension of traditional LSTMs that can improve model performance on sequence classification problems.

In problems where all timesteps of the input sequence are available, Bidirectional LSTMs train two instead of one LSTMs on the input sequence. The first on the input sequence as-is and the second on a reversed copy of the input sequence. This can provide additional context to the network and result in faster and even fuller learning on the problem.

2.5.10 CNN, LSTM- layers storing/dumping and making it reusable with other models

Storing the layers from different models to make them reuseable for other model on a similar dataset proves out to be more efficient rather than providing random weights in the initial layers. We carried out various tasks or experiments to check for layers reuseablity, which in turn woud help us increase the model performance. Some of the experiments carried have been mentioned below :

1. Get a custom pretrained word2vec embedding which can be usable for a similar dataset.

   Experiment1: with two similar datasets, Getting w2v from one, building model, training. Then training with another dataset by loading the embedding layer from prev training
2. Get a sentence vectorizer built with a CNN, LSTM, to be able to use for sentence similarity checks.

   Experiment 2: Get a vector using second last layer of CNN/LSTM-trained model

5CHAPTER 3
ANNOTATION TOOL

3.1 Prodigy

Prodigy is a powerful and efficient machine teaching annotation tool , powered by active learning.

Properties of Prodigy :

1. **Easy and Fast Training**

   Prodigy is an annotation tool designed especially for data scientists so that they can do annotations themselves , enabling rapid iteration at a newer level. It can help even if you are working on entity recognition , image classification , POS tagging or intent detection. Prodigy is designed to help you train and evaluate the models faster. It allows you to stream your own data or real-world data from live APIs, also updates the model in real-time and chains models together to build more complex systems.

2. **The missing piece in the Datascience Workflow**

   Prodigy brings together the state-of-the-art insights from machine learning and user experience. Because of it's **active learning** system ,you are only asked to annotate the examples that the model is not aware of and does not know the answer to. The prodigy web-application is powerful, immense and follows modern UX principles.
   Principle : Designed to help you focus on one decision at a time and keep you clicking and annotating.
   For eg : Tinder for data

3. **Helps try the new ideas quickly**

   Annotation is the phase where the project lingers. Hence, instead of trying out the idea , we start scheduling meetings, planning out the idea, writing specifications and wandering about the outcomes. So, with prodigy, we can have the and within a few hours, can get the first results. Once, the model is trained, it can be exported as a versioned Python package, paving a smooth path from prototype to production.

4. **Cloud Free**

Since ,AI is not a commodity ,hence cannot be bought from third-party provider. We need to build our own systems , own our tools and control our data. So, prodigy helps serve this purpose. This tool is self-contained, extensible , and ours forever. It will help even for a really complex pipeline – if it can be called from a python function, it can be used in prodigy

3.2 How Does it Help ?

Prodigy uses:

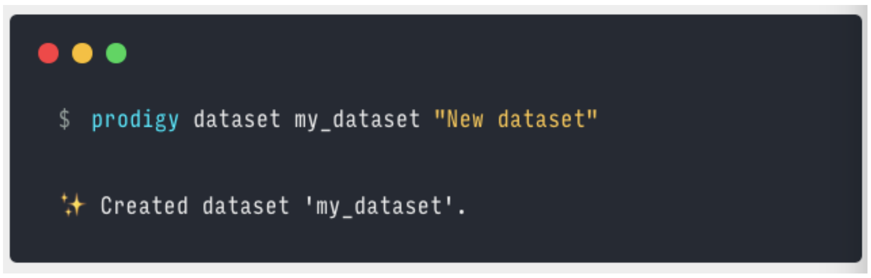
It provides the ease of using our own components for storage, classification, loading, example selection and annotation.

1. Helps us extend spaCy's NER (named entity recognizer)
2. Helps us increase the accuracy of spacy's models on our own text .
3. We can create , improve and evaluate the models for intent detection , sentiment analysis and text classification .
4. We can also annotate object detection and image segmentation .

Roadmap of Prodigy

1. **Get up and Running quickly :**

   Prodigy can be used straight out of the box – Python and a web browser is all that is required .The annotations hence can be stored in local file ,using SQLite. Other than SQLite , we can use built-in MySQL , PostgreSQL back-ends.



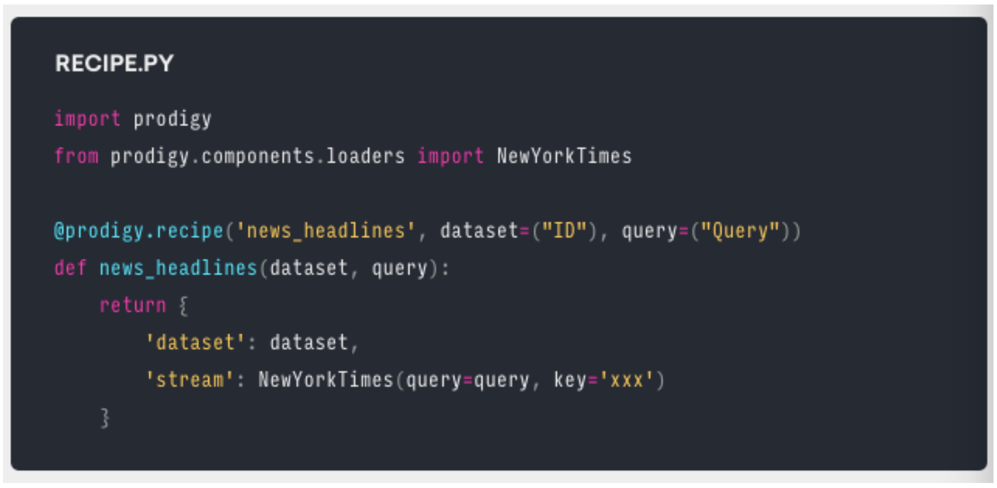
```
$ prodigy dataset my_dataset "New dataset"

✨ Created dataset 'my_dataset'.
```

Figure 22: Initializing a Dataset

2. **Built-in or Custom Recipe**

   Recipes help us control the annotation stream and processing logic , and define model updation in our own way .

   Implementing custom recipes as python functions makes it easy to integrate our own solutions, no matter how complex the logic is .

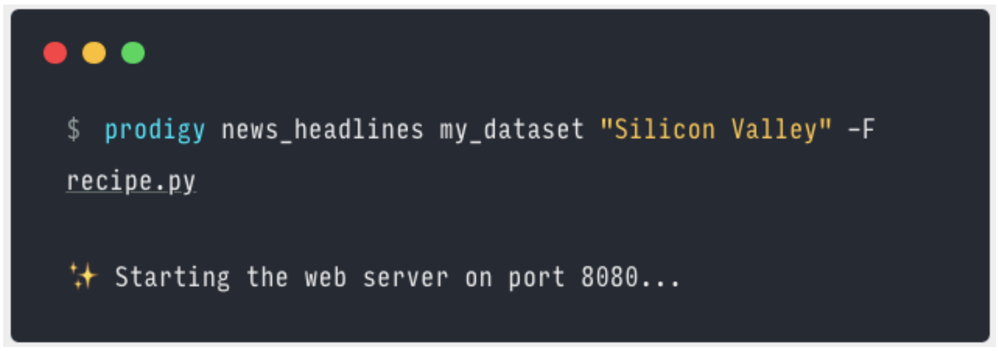


```
RECIPE.PY

import prodigy
from prodigy.components.loaders import NewYorkTimes

@prodigy.recipe('news_headlines', dataset=("ID"), query=("Query"))
def news_headlines(dataset, query):
    return {
        'dataset': dataset,
        'stream': NewYorkTimes(query=query, key='xxx')
    }
```

Figure 23: Describing Custom Recipes

3. **Running Recipes**

   Recipes can be run directly using the command line , or else they can be embedded in the python script and called within the python function. When the recipe is run, Prodigy starts the web server and hence, one can start annotating.

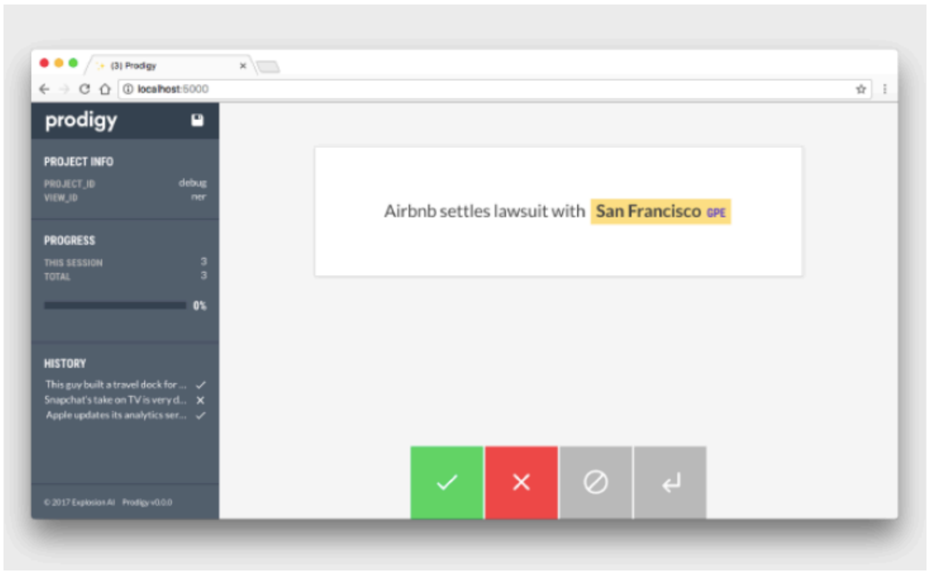


```
$ prodigy news_headlines my_dataset "Silicon Valley" -F recipe.py

✨ Starting the web server on port 8080...
```

Figure 24: Annotating on the port

4. **Modern Web Application**

   You can annotate – text , entities , classification and images straight from the browser -even on mobile devices . It's modern UI is really helpful in keeping the user focussed only on one task at a time and one binary decision at a time.
   As the user swipes through the examples, annotations are sent back to Prodigy via REST API. Prodigy involves real-time model updation, which leads to removing already known questions from the stream.



Airbnb settles lawsuit with San Francisco GPE

This guy built a travel dock for ... ✓
Snapchat's take on TV is very d... ✗
Apple updates its analytics ser... ✓

Figure 25: Web Application

Efficient Annotation Mechanism

Prodigy puts the model in loop , hence enabling active learning . The model uses what it already knows to figure out the next task , and is updated with the user provided answers . The configuration is really simple : just write the python function ,that returns the components in the form of a dictionary .
Prodigy's built-in recipes can be chained together to built complex systems .

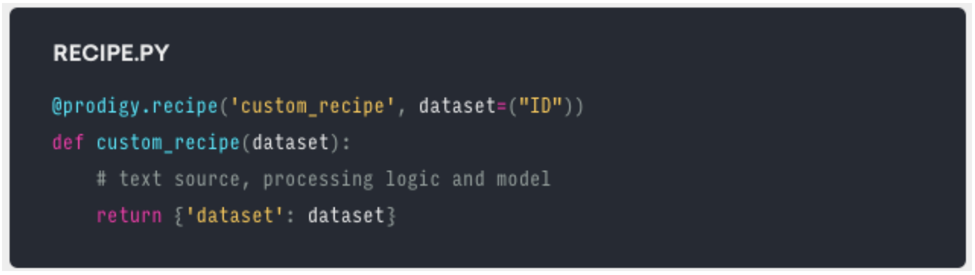

```
RECIPE.PY
@prodigy.recipe('custom_recipe', dataset=("ID"))
def custom_recipe(dataset):
    # text source, processing logic and model
    return {'dataset': dataset}
```

Figure 26: View of Custom Recipe

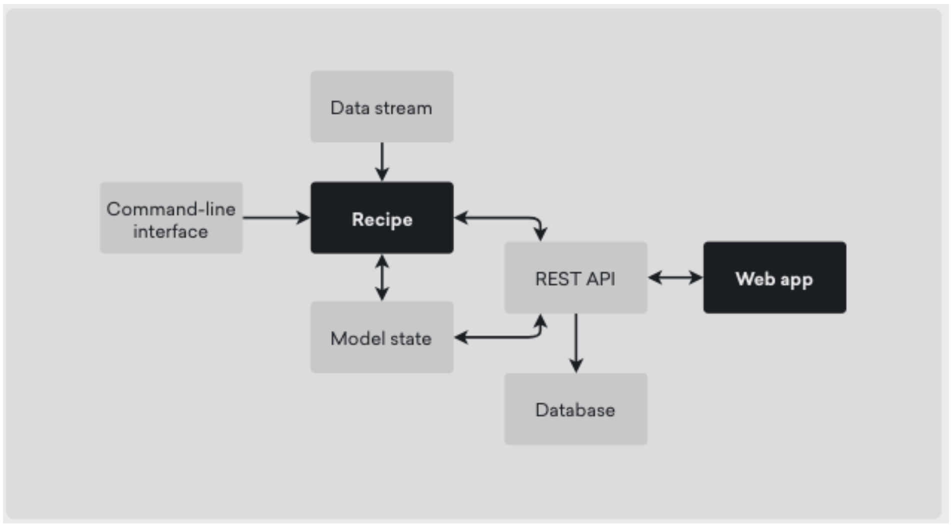


Figure 27: Flowchart of Prodigy Workflow

3.3 What We Did ?

We chose Prodigy over the traditional annotation method, because the main focus of Prodigy is time, which is an important factor in the life cycle of any project.

Since, prodigy focuses on one task at a time, so it is even more efficient because it keeps the annotator focused by restricting him from thinking about any other task at that point of time.

So, the main workflow of prodigy is mentioned in the following flowchart :

The input data is read from the excel sheet, and an input stream of the required data fields is created. This input field is streamed on the browser at different ports. These web addresses are shared to different users, as prodigy supports multiuser environment, using **ngrok**, which creates a public URL for exposing the local web server.

NGROK : creates instant secure URL to the localhost server via firewall or NAT.

The data is then displayed on different systems, and is annotated by the annotators. The annotated data is saved on the localhost and the output stream is converted into excel format.

This process is continued till all the required data is tagged and updated.

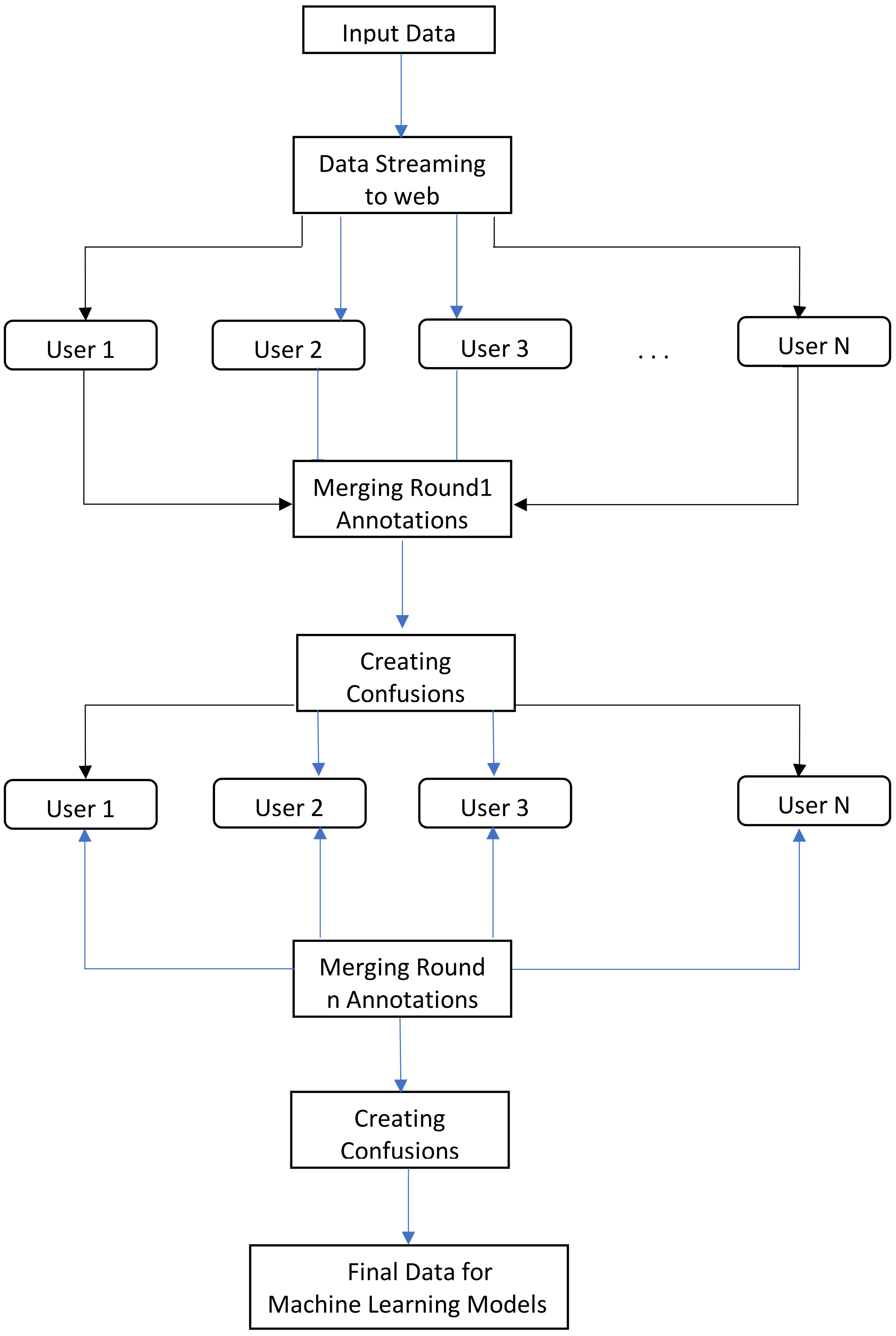


Figure 28: Prodigy Workflow

3.4 Testing

The tool went through various testing phases and the following changes were made

- Addition of exclude function so that the annotator does not have to re-annotate the data from the start is case of any software or hardware failure.
- **Inter-annotator** was created .Modules were added to stream same data to different users simultaneously
- There was a **Confusion Matrix** created across the users so as to check the confusions in the tagged data, for further improvement.
- There was **Flag** feature added, so that all the sentences that they are not sure of, or are having some error can be marked and taken a look at later on.
- **Instruction** link was added on the annotation page, so that if the annotators are unclear of any theme, then they can take a look at the example set and continue annotating.

CHAPTER 4
CONCLUSIONS AND FUTURE SCOPE

The first thing being Data Wrangling, the data is scraped from various sites. The main Workflow of our company includes manual Tagging of the data by the Data Analyst Team. And then this tagged data is forwarded to the Machine Learning team for running the models. If the data is not satisfactory, the models will not perform good. The main power lies in the hands of the Data Analyst team. And it is a fact that human beings are prone to error. So, the main focus of the company is automation. So, we are planning to automate all the small modules like running Inconsistencies/ Keyword Pair code/ Dependency Parsing etc. This will help the Machine Learning team to focus more on Research activities and finding better and optimised techniques, which are being used on daily basis and deploy them on the server so that they are available for direct use by the Analyst team and the Machine Learning team can focus more on Research area.

Our future experiments will include techniques like Experimenting Deep learning models with Keras wrapper for scikit-learn, optimizing the sentiment models, experimenting with more Deep Learning Methods and Machine Learning methods.

Our further focus will be on optimizing the work of Analyst team. We will further look for tools like Prodigy, so that tagging the data can be more efficient and faster. So, the main focus is on optimization techniques which would help us increase the speed and the quality of the results obtained.