

Yuru Cao. Testbot: A Chatbot-Based Interactive Interview Preparation Application. A Master's Project for the M.S. in I.S degree. April, 2020. 52 pages. Advisor: Arcot Rajasekar

Chatbots are programs that mimic human conversation using Artificial Intelligence. It has become particularly popular to serve as an ultimate virtual assistant, helping one to complete tasks with answering questions and solving problems. In this paper, I designed an efficient and accurate chatbot, Testbot, which is integrated with a web application to help students prepare interviews for data science. To meet this usability requirements, I deployed and customized an open-source chatbot framework Rasa and set a form action to keep track of the conversation. Testbot now supports basic functions such as responding to greetings and chitchat, and more advanced features that could recognize interview request, ask for user-generated keywords to retrieve questions from the database, score users' answers and send feedback. I demonstrated the application running on a local server with a user interface implemented.

Headings:

Web application

Chatbot

Interview preparation assistant

Rasa framework

Sentence similarity

TESTBOT: A CHATBOT-BASED INTERACTIVE
INTERVIEW PREPARATION APPLICATION

by
Yuru Cao

A Master's paper submitted to the faculty
of the School of Information and Library Science
of the University of North Carolina at Chapel Hill
in partial fulfillment of the requirements
for the degree of Master of Science in
Information Science.

Chapel Hill, North Carolina

April 2020

Approved by

Arcot Rajasekar

TABLE OF CONTENTS

1. INTRODUCTION	2
2. LITERATURE REVIEW.....	4
2.1 CHATBOT HISTORY	4
2.2 QUESTION ANSWERING SYSTEM	6
2.3 CHATBOT DESIGN TECHNIQUES	8
2.4 RESPONSE GENERATION SYSTEMS IN CHATBOT	10
2.5 OPEN SOURCE CHATBOT FRAMEWORK	11
2.6 EVALUATION METRICS FOR CHATBOT	12
3. SYSTEM DESIGN	13
3.1 OVERVIEW	13
3.2 SYSTEM INFRASTRUCTURE.....	14
4. SYSTEM DEVELOPMENT	16
4.1 CHATBOT ARCHITECTURE.....	16
4.2 CHATBOT DEPLOYMENT	20
4.3 WEB APPLICATION.....	26
5. CONCLUSION	29
6. REFERENCES.....	31
7. APPENDIX.....	33

1. INTRODUCTION

In definition, chatbots are programs that could mimic human interactions. It is a virtual assistant able to answer a number of user questions, providing the correct responses.

Interactive question answering systems, where a dialogue interface enables follow-up and clarification questions, are a recent field of research. With the aid of such technology, chatbots could keep track of the dialogue flow and provide response accordingly. We now see many major internet companies developed several dominant chatbots software that help with people's life and needs: Apple Siri, Microsoft Cortana, Facebook M. These bots have been widely applied to a variety of fields such as Healthcare, Marketing, Education, and supporting systems.

One of the major fields of chatbot applications are question answering systems, which serves as a virtual assistant answering users' question via chat. Question answering systems can be seen as information retrieval systems which aims at responding to natural language queries by returning answers rather than lists of documents. It normally includes natural language processing and artificial intelligence to process user's input and generate the best suitable answer. With this foundation technology, a chatbot targeting to meet the needs of students preparing for job interviews is created in this master project.

This chatbot-based interview preparation application, named Testbot, will have features as follows:

- A well-trained machine learning chat model to fully analyze users' behaviors
- Allow students to pick interview topics and sections to select interview questions according to their own needs
- Testbot will score their answer based on the coverage of the correct keywords and semantics of the sentence

A demo application of Testbot has been developed and attached. This master paper documented the process of designing the application, deploying the bot and customizing the models. Related methodologies implemented in this project are being introduced in details. The full project repository can be found on author's personal Github page.

2. LITERATURE REVIEW

This section is the general review of the chatbot development history and relevant technologies in this field.

2.1 Chatbot History

One of the early chatbots is ELIZA, which was designed around a pattern matching pattern. ELIZA is the primary chatbot created by Joseph Weizenbaum[1] utilizing a keyword coordinating strategy. To look for certain keywords, if a catchphrase was discovered, then the appropriate response was discovered. In the event that a catchphrase was not discovered, ELIZA would get more data from the client to keep the discussion going. The theory behind ELIZA is to break user queries into segments and extract those keywords to formulate new questions back to the users. In this way, ELIZA could dig deeper by asking questions, and thus ultimately getting closer to the core. The biggest problem with this ‘therapist’ is that it only goes with one on one question answering rule, thus having a high risk of being misled if users change the topic or provide nothing useful in a single query.

Below is an example of a ridiculous conversation with ELIZA.

ELIZA: Good Evening. Tell me your problems. Please terminate input with a period or a question mark.

PARRY: People get on my nerves sometimes.
ELIZA: I am not sure I understand you fully.
PARRY: You should pay more attention.
ELIZA: Suppose you should pay more attention.
PARRY: You're entitled to your own opinion.
ELIZA: What makes you think I am entitled to my own opinion?
PARRY: It bothers me just to be around people in general.

Figure 1: Example chat with ELIZA

Inspired by ELIZA, ALICE was introduced and became another popular technique of chatbot, provided by Richard Wallace in 1995[3]. ALICE used pattern matching techniques and was implemented based on a scripting language called AIML (Artificial Intelligence Mark-up Language). An AIML record is like an XML document that is created to store design information for chatbots, which contains units called topics or categories. It is currently among the most used languages for chatbots. ALICE will have a template for each category, inferring the robot answer for user input. There are more than 50,000 categories in the current public ALICE 'brain' added each year by manually.

Below is a sample chat with ALICE with a corpus-based approach.

User: Hello Alice.
Alice: Hi there!
User: How old are you?
Alice: I am eighteen in computer years.
User: Who is your father?
Alice: I was programmed by Dr. Richard S Wallace.

Figure 2: Example chat with ALICE

With modern technologies embedded into chatbots, more intelligent applications have been developed including: Amazon's Echo and Alexa, Apple's Siri, and Microsoft's Cortana. The architectures and retrieval processes of these bots take advantage of advances in machine learning to provide advanced "information retrieval" processes, in which responses are generated based on analysis of the results of web searches. Others have adopted "generative" models to respond; they use statistical machine translation (SMT) techniques to "translate" input phrases into output responses. Seq2Seq, an SMT algorithm that used recurrent neural networks (RNNs) to encode and decode inputs into responses is a current best practice [14].

2.2 Question Answering System

Question answering system (QA) is in the field of information retrieval (IR) and natural language processing (NLP). From the perspective of knowledge, the QA system can be divided into 'closed domain' and 'open domain'. Systems in closed domain [4] only focus on answering questions in the specific field, such as medicine or law, and can have better performance. An open-domain QA system [5] is able to answer all the questions, not limited to a specific field. Therefore, it is more difficult than a closed domain system. The traditional question answering system works as a 'smart' searching engine that would generate a tailed answer key for user queries. It could serve as an information retrieval assistant that 'talks' to you. The QA will process and analyze the information in user query, and extract the answer that is most similar to this question for the user. Frequently Asked Questions system is a common application for customer service, in

which selective questions and answers are predefined in the database. Users' query is compared with questions in existing FAQs to get a related answer to respond to the query.

The development of the question-answering system is a very complex process that includes a variety of natural language processing as well as data mining algorithms. In general case one can point out 12 key objectives in the field of the QA-systems building developed in 2002 by a group of researchers [6].

1. Question classification. It is considered as selection of question type or in certain field.
2. Stage of processing and analysis. It includes semantic analysis of the question phrases.
3. The ability of the system to see the context of questions allows it to make clarifying questions or alternatively, to find answers to the questions posed earlier.
4. Base of knowledge generation is one of the key stages in the QA-system development.
5. Stage of the answer selection and its evaluation.
6. Answer wording.
7. Real-time responding capability.
8. Support of multi-linguistics.
9. Interactivity to learn from users' feedback of the answer.
10. Ability for consideration.
11. Analyzing users' profile to provide customization.
12. Taking the advantages of certain systems, eliminating the weaknesses through the strengths of other systems.

2.3 Chatbot Design Techniques

Numerous researches have been conducted to apply different techniques on certain use case of chatbot systems. In most cases, their research work are focusing on improving a methodologies in chatbot development process, including: speech parsing, natural language processing, keyword identification, artificial intelligence, etc. In this section, the fundamental chatbot design technologies will be listed and summarized.

The design of a chatbot could be represented using diagram as follows:

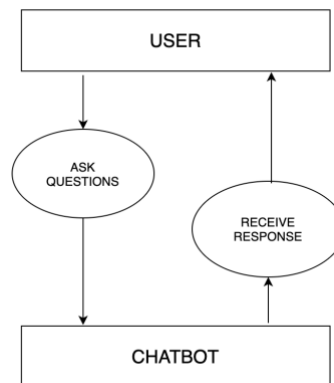


Figure 3: Use Case Diagram of Chatbot Design

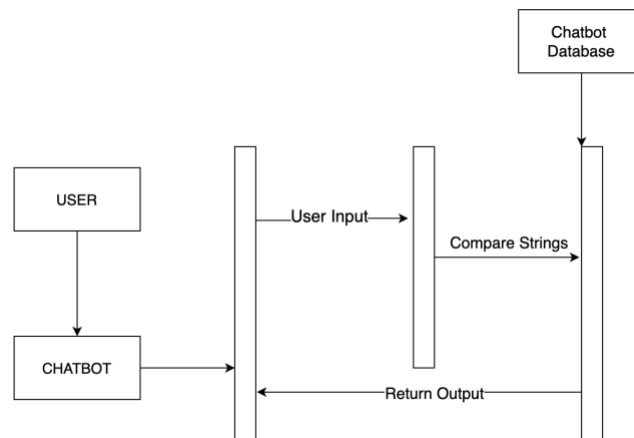


Figure 4: Sequence Diagram Representing Design of a Chatbot

To build a chatbot that performs the best practice giving suitable answers to keywords or phrases extracted from speech and keep conversation continues, a number of techniques needed to be implemented and approached:

A. Parsing: this technique includes analyzing the input text and manipulating it by using a number of NLP functions. For example, trees in Python NLTK.

B. Pattern matching: it is the technique that is used in most Chatbots and it is quite common in question answering systems depending on matching types, such as natural language enquiries, simple statements, or semantic meaning of enquiries [8].

C. AIML (Artificial Intelligence Mark-up Language): initially developed for ALICE

D. Chat Script: is the technique that helps when no matches occur in AIML. It concentrates on the best syntax to build a sensible default answer.

E. Relational Database: to store the dialogue history

F. Statistical method: for example, Markov Chain, is used in Chatbots to build responses that are more applicable probabilistically and, consequently, are more correct.

G. Ontologies: they are also named semantic networks and are a set of concepts that are interconnected relationally and hierarchically. The aim of using ontologies in a Chatbot is to compute the relation between these concepts, such as synonyms, hyponyms and other relations which are natural language concept names. The interconnection between these concepts can be represented in a graph enabling the computer to search by using particular rules for reasoning [9].

2.4 Response Generation systems in Chatbot

Response generation is arguably the most central component of the chatbot architecture.

As input, the response generator receives a structured representation of the spoken text.

This conveys information about who is speaking, the dialogue history, and the context.

As output, the RG generates a response to deliver to the user. There are three main ways bots can retrieve a response through response generation models [14].

2.4.1 Rule-Based Models

The principle idea for rule-based model is to contain a knowledge base with documents, which all content are being matched to certain rule. The bot could then use algorithm to pattern matching between input sentence and document in data corpus. ELIZA and ALICE are both early implementations of rule-based models. ELIZA was created with multiple “scripts” which indicate different responses to inputs. The ALICE architecture includes a knowledge base called Graphmaster, which can be thought of like a file system, with a root that contains files and directories.

2.4.2 Information Retrieval (IR) -Based Models

Information retrieval-based models were introduced as a generalization of the rule-based models. Programmers no long need to specify all the rules for pattern and template pairs. Instead, the principle behind IR-based models is, given an input sentence, to pattern match this against the set of pairs and select a response. The main challenge in IR algorithms is pattern matching algorithms. Both basic models such as keyword approach, and more complex models which include an ensemble of score function between document and query, are widely adopted.

2.4.3 Statistical Machine Translation Generative Models

IR-based models require a database of possible responses to choose from. Generative models, by contrast, build responses on the fly, mostly using machine learning techniques. Statistical Machine Translation (SMT) models are the most recent, effective models used to generate chatbot responses. Statistical Machine Translation is a field of machine translation which analyzes bilingual corpuses of text, and uses statistical analysis to derive exact transactions between individual words, phrases, and features of the texts. SMT generates cost savings relative to other automated approaches in that rule-based translation system require the manual creation of rules.

2.5 Open Source Chatbot Framework

According to Lebeuf [10], a chatbot project is composed of distribution and creation services. Distribution services are employed to manage the conversation data. It serves as the interface between chatbot and user. Creation services are used to develop the chatbot knowledge and behavior. The distribution service itself is composed of several components [11]: Natural Language Understanding (NLU), communication channels, user interface, and dialogue management.

Following the composition, a vast number of chatbot frameworks is available on open source software, such as Botkit, Rasa [12], Botpress [13], among others. Typically, open source chatbot frameworks are designed for developers and specialist. These frameworks have clearly defined user documentation to guide through each of the chatbot component. Rasa, focus mostly on artificial intelligence and building a framework that allow users to

improve AI performance. Instead of defining visual flows and intents within the platform, Rasa allow developers to create stores (training data scenarios) on which the bot is trained. Rasa will then feed the data into training model (NLU engine) and get better performance on responding to users, which makes the Rasa bot more flexible and capable.

2.6 Evaluation Metrics for Chatbot

The Loebner prize competition has been used to evaluate machine conversation chatbots. The Loebner Prize is a Turing test, which evaluates the ability of the machine to fool people that they are talking to human. In essence, judges are allowed a short chat (10 to 15 minutes) with each chatbot, and asked to rank them in terms of “naturalness”. There are a number of different perspectives on how to evaluate chatbot performance. From an information retrieval (IR) perspective, chatbots have specific functions: there are virtual assistants, question-answer and domain-specific bots. Evaluators should ask questions and make requests of the chatbot, evaluating effectiveness by measuring accuracy, precision, recall, and F-score relative to the correct chatbot response. From a user experience perspective, the goal of the bot is, arguably, to maximize user satisfaction. Evaluators should survey users (typically, measured through questionnaires on platforms such as Amazon Mechanical Turk), who will rank bots based on usability and satisfaction. From a linguistic perspective, bots should approximate speech, and be evaluated by linguistic experts on their ability to generate full, grammatical, and meaningful sentences. Finally, from an artificial intelligence perspective, the bot that appears most convincingly human (e.g. passes the Turing Test best) is the most effective. [14]

3. SYSTEM DESIGN

This section will illustrate the system design and system infrastructure of Testbot. It will cover the work flow, infrastructure and technology stack of the system, and dialogue scenario design.

3.1 Overview

The work flow of the system is shown in the figure 5. The chatbot will first start conversations with users about basic greetings or general questions. Chatbot has template matching method to recognize the chitchat and will send proper response to that. Once the interview request input is sent to the chatbot, the interview preparation system is triggered. Intent check will handle the request to confirm the request is valid. Then the bot will follow the pre-defined flows to ask for parameters, such as interview topic, or details from users. The user sends back the corresponding query which triggers the bot's validation and question retrieval from the system. The next submission from the users will be considered as users' interview answers and will be scored based on internal similarity calculation algorithm. Once this step is finished, the bot returns the score and correct answers. This workflow ends. The bot can still receive and respond to basic general questions and chats when the workflow ends.

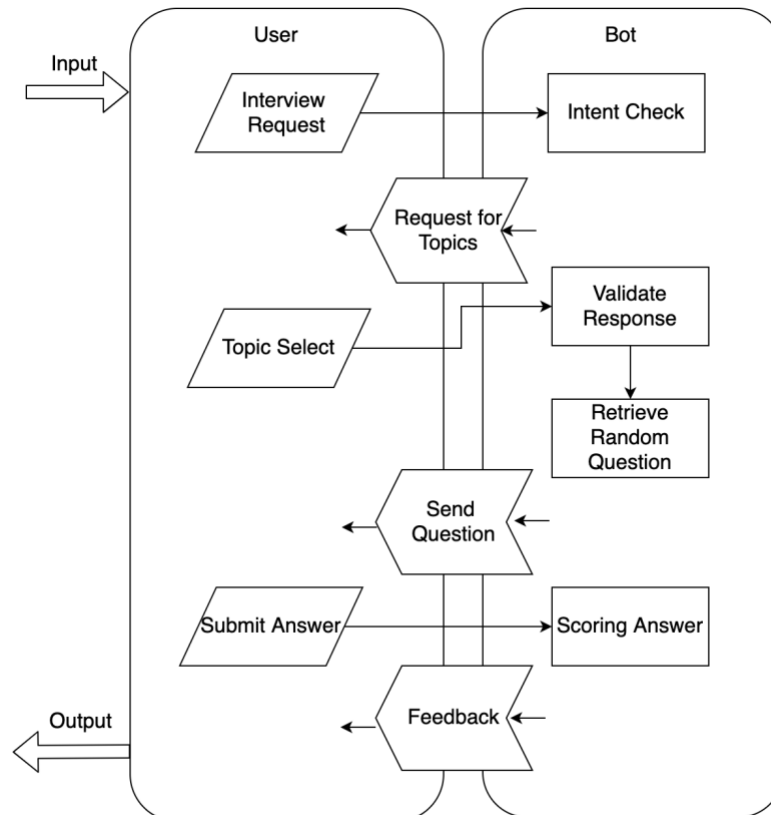


Figure 5: Workflow of System

3.2 System Infrastructure

This system will embed an open source chatbot framework ‘Rasa’ to implement the chat capabilities. Rasa architecture comes with two components – Rasa NLU and Rasa Core. Rasa NLU a library for natural language understanding (NLU) which does the classification of intent and extract the entity from the user input and helps bot to understand what the user is saying. a chatbot framework with machine learning-based dialogue management which takes the structured input from the NLU and predicts the next best action using a probabilistic model like LSTM neural network.

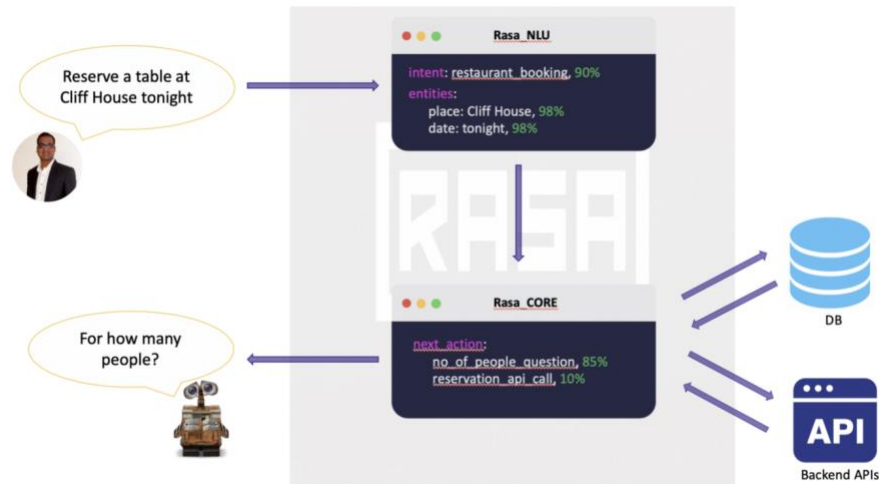


Figure 6: Rasa Architecture

To install and deploy this chatbot in our application, I used Python to get access to Rasa API. Spyder (Python 3.7) was used as the code editor. Github was used as the code version control tool. Detailed technology stack is shown in table 1 below.

Web Development	Flask (Python), HTML5, CSS, JavaScript
Chatbot Framework	Rasa Bot
Database	MySQL

Table 1: Technology Stack

Flask is a micro web framework written in Python. It does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. As a web framework, it provides functionality for building web applications, including managing HTTP requests and rendering templates in Python environment. I used Flask to connect to the database server and host a local server on the browser to receive request.

4. SYSTEM DEVELOPMENT

This section will describe the system development process in details. The architecture and deployment of Testbot will be introduced and demonstrated.

4.1 Chatbot Architecture

Rasa's architecture is modular by design. This allows easy integration with other systems. I implement Rasa's core and NLU models to my web application using HTTP's POST and GET request.

4.1.1 Tracker

Dialogue state is saved in a tracker object. There is one tracker object per conversation session, and this is the only stateful component in the system. A tracker stores slots, as well as a log of all the events that led to that state and have occurred within a conversation. The state of a conversation can be reconstructed by replaying all of the events. Rasa's tracker object has predefined methods including: `current_slot_values()`, `get_last_event()`, `current_state()`, `export_stories()`, and many others.

When a user message is received Rasa takes a set of steps as described in figure 1. Step 1 is performed by Rasa NLU, all subsequent steps are handled by Rasa Core.

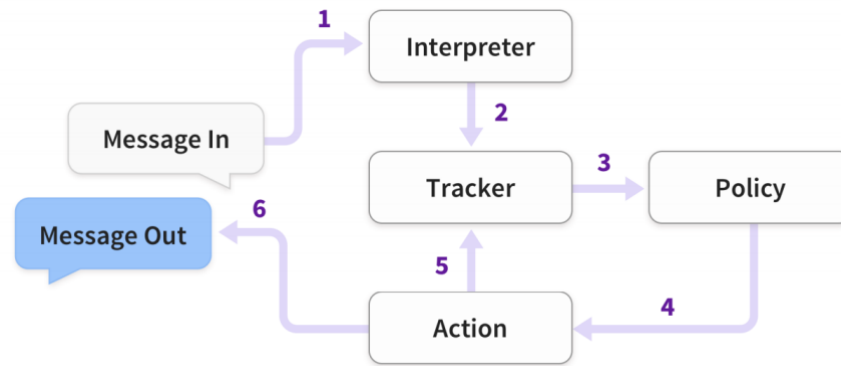


Figure 7: Rasa flow

In figure 7, a message is received and passed to an Interpreter (Rasa NLU) to extract the intent, entities, and any other structured information. The tracker maintains conversation state. It receives a notification that a new message has been received. The policy receives the current state of the tracker. The policy chooses which action to take next. The chosen action is logged by the tracker. The action is executed (this may include sending a message to the user). If the predicted action is not ‘listening’, go back to step 3.

4.1.2 Actions

I frame the problem of dialogue management as a classification problem. At each iteration, Rasa Core predicts which action to take from a predefined list. An action can be a simple utterance, i.e. sending a message to the user, or it can be an arbitrary function to execute. When an action is executed, it is passed a tracker instance, and so can make use of any relevant information collected over the history of the dialogue: slots, previous utterances, and the results of previous actions. Actions cannot directly mutate the tracker, but when executed may return a list of events. The tracker consumes these events to

update its state. There are a number of different event types, such as SlotSet, AllSlotsReset, Restarted, etc.

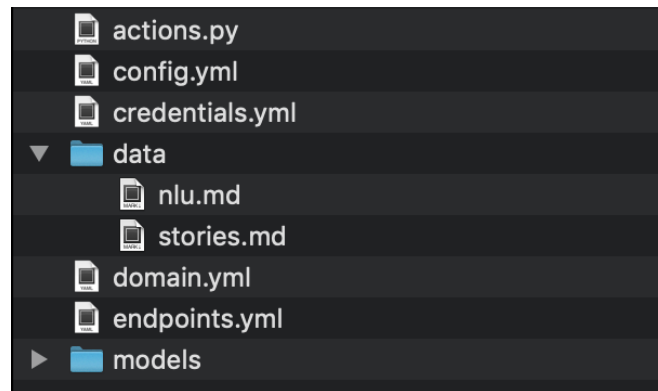


Figure 7: Rasa project structure example

```
actions.py - /Users/caoyuru/Desktop/MasterPaper/rasa/examples/restaurantbot/a...
from rasa_sdk import Action
from rasa_sdk.events import SlotSet

class RestaurantAPI:
    def search(self, info):
        return "papi's pizza place"

class ActionSearchRestaurants(Action):
    def name(self):
        return "action_search_restaurants"

    def run(self, dispatcher, tracker, domain):
        dispatcher.utter_message(text="looking for restaurants")
        restaurant_api = RestaurantAPI()
        restaurants = restaurant_api.search(tracker.get_slot("cuisine"))
        return [SlotSet("matches", restaurants)]

class ActionSuggest(Action):
    def name(self):
        return "action_suggest"

    def run(self, dispatcher, tracker, domain):
        dispatcher.utter_message(text="here's what I found:")
        dispatcher.utter_message(text=tracker.get_slot("matches"))
        dispatcher.utter_message(
            text="is it ok for you? hint: I'm not going to find anything else :)
        )
        return []
```

Figure 8: Rasa actions.py example snapshot

4.1.3 Natural Language Understanding

Rasa NLU is the natural language understanding module. It comprises loosely coupled modules combining a number of natural language processing and machine learning

libraries in a consistent API. Rasa aims to strike a balance between customizability and ease of use. To this end, there are pre-defined pipelines with sensible defaults which work well for most use cases.

For example, the recommended pipeline, `sklearn`, processes text with the following components. First, the text is tokenized and parts of speech (POS) annotated using the NLP library. Then the featurizer looks up a vector for each token and pools these to create a representation of the whole sentence. Then the scikit-learn classifier trains an estimator for the dataset, by default a multi-class support vector classifier trained with five-fold cross-validation. The crf component then trains a conditional random field to recognize the entities in the training data, using the tokens and POS tags as base features. Since each of these components implements the same API, it is easy to swap (say) the vectors for custom, domain-specific word embeddings, or to use a different machine learning library to train the classifier. There are further components for handling out-of-vocabulary words could be customized by users.

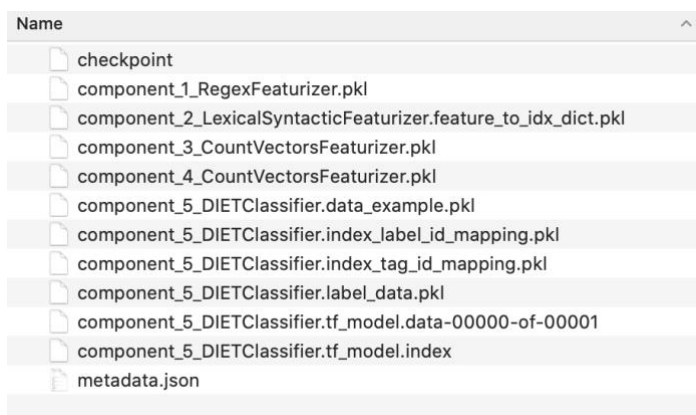


Figure 9: Rasa NLU model component snapshot

4.1.4 Policies

The job of a policy is to select the next action to execute given the tracker object. A policy is instantiated along with a featurizer, which creates a vector representation of the current dialogue state given the tracker.

The standard featurizer concatenates features describing:

- what the last action was
- the intent and entities in the most recent user message
- which slots are defined currently

The featurization of a slot may vary. In the simplest case, a slot is represented by a single binary vector element indicating whether it is filled. Slots which are categorical variables are encoded as a one-of-k binary vector, those which take on continuous values can specify thresholds which affect their featurization, or simply be passed to the featurizer as a float. There is a hyperparameter `max_history` which specifies the number of previous states to include in the featurization. By default, the states are stacked to form a two-dimensional array, which can be processed by a recurrent neural network or similar sequence model.

4.2 Chatbot Deployment

4.2.1 Training Data

Both Rasa NLU and Core work with human-readable training data formats. Rasa NLU requires a list of utterances annotated with intents and entities. These can be specified either as a JSON structure or in markdown format.

A story is a representation of a conversation between a user and an AI assistant, converted into a specific format where user inputs are expressed as corresponding intents (and entities where necessary) while the responses of an assistant are expressed as corresponding action names.

```
## greet + location/price + cuisine + num people <!-- name of the story - just for debugging -->
* greet
  - action_ask_howcanhelp
* inform{"location": "rome", "price": "cheap"} <!-- user utterance, in format intent{entities} -->
  - action_on_it
  - action_ask_cuisine
* inform{"cuisine": "spanish"}
  - action_ask_numpeople <!-- action that the bot should execute -->
* inform{"people": "six"}
  - action_ack_dosearch
```

Figure 10: Rasa Stories training dialogue snapshot

In figure 10, an example story is given which intent to greet and ask for location, price, cuisine and number of people. All the entities ‘location’, ‘price’, ‘cuisine’ and ‘people’ have been defined in domain file. The actions are initiated in python file while responses format could be referenced from domain file.

The domain defines the universe in which your assistant operates. It specifies the intents, entities, slots, and actions your bot should know about. Optionally, it can also include responses for the things your bot can say. The domain file has yaml definition and has to follow yaml structures formatting.

```

intents:
- greet
- goodbye
- affirm
- deny
- mood_great
- mood_unhappy
- bot_challenge

responses:
utter_greet:
- text: "Hey! How are you?"

utter_cheer_up:
- text: "Here is something to cheer you up:"
  image: "https://i.imgur.com/nGF1K8f.jpg"

utter_did_that_help:
- text: "Did that help you?"

utter_happy:
- text: "Great, carry on!"

utter_goodbye:
- text: "Bye"

```

Figure 11: Rasa Domain training dialogue snapshot

The training data for NLU models could be either JSON or Markdown. In this project, I use markdown to format the sentence data. Common examples have three components: text, intent and entities. The first two are strings while the last one is an array.

- The text is the user message which is required.
- The intent is the intent that should be associated with the text.
- The entities are specific parts of the text which need to be identified.

```

## intent:check_balance
- what is my balance <!-- no entity -->
- how much do I have on my [savings](source_account) <!-- entity "source_account" has value "savings" -->
- how much do I have on my [savings account](source_account:savings) <!-- synonyms, method 1-->
- Could I pay in [yen](currency)? <!-- entity matched by lookup table -->

```

Figure 12: Rasa example NLU training data snapshot

In figure 12, the intent is defined to check balance. Following are four user messages to feed into training models, both for natural language processing, and for train examples for entities value. In order to gain a higher accuracy and effectivity, the user messages should be provided as many as possible in the file. The more human-like languages are learned by the core model, the better it performs when it processes on new sentences and phrases. The aim is to train the classification model so that every user message with the same intent could be recognized.

4.2.2 Form Action

In order to collect a few pieces of information from users to decide the next steps, I used one of the most common conversation patterns in this project – form action, which is also called slots filling in rasa. With form action, I could define several slots for users to fill, define methods extract slot data from user messages and validate the slots value. Three basic component methods are: name of the action, required_slots, and submit. Once the form action gets called for the first time, the form gets activated and it will ask the user for the next slot in required_slots which is not already set. In order to ask users for slot in a conversation flow, it will look for a response called utter_ask_{slot_name} in domain file and send that request to users. Once all the slots are filled, the submit() method is called, where I can use the information collected to run my methodology and send feedback response back to user. Figure 13 below demonstrates the form action to collect question topic from users and validate their selection.

```

import pandas as pd
from rasa_sdk.forms import FormAction
from typing import Dict, Text, Any, List, Union
from random import randint

from rasa_sdk import Tracker
from rasa_sdk.executor import CollectingDispatcher

class InterviewForm(FormAction):
    """Collects interview information and adds it to the spreadsheet"""
    def __init__(self):
        self.index = []
        self.sheet = []

    def name(self) -> Text:

        return "interview_form"

    @staticmethod
    def required_slots(tracker):
        return ["interview_topic", "section", "answer"]

    def slot_mappings(self) -> Dict[Text, Union[Dict, List[Dict]]]:
        return {
            "interview_topic": self.from_entity(entity="interview_topic", intent="inform"),
            "section":
                self.from_text(intent = "select"),
            "answer":
                self.from_text()
        }

    def validate_interview_topic(
        self,
        value: Text,
        dispatcher: CollectingDispatcher,
        tracker: Tracker,
        domain: Dict[Text, Any],
    ) -> Dict[Text, Any]:
        tracker.slots["interview_topic"] = value
        return {'interview_topic':value}

    def validate_section(

```

Figure 13: Interview form action screenshot

4.2.3 Interview Question Dataset

To retrieve interview questions for users, I need to have a database that support the system. The dataset is a list of interview questions collected online with four main topics: Machine Learning, Statistics, SQL and Programming. Each of these have breakdown section categories. I selectively pick up those questions that are more conceptual and have a specific key to the perfect answers. In that case users' interview could be better evaluated thus the interview preparation application will be more effective to them. Full dataset has been attached in the appendix. See below as an example:

- Question: What is a hash table?

- Answer: A hash table is a data structure that produces an associative array. A key is mapped to certain values through the use of a hash function. They are often used for tasks such as database indexing.

4.2.4 Scoring Methodology

In order to evaluate the users' response thus scoring their performance on this interview question, I tried on several algorithms on sentence similarity, which is to estimate the degree of similarity between two texts. The big idea is that to represent documents as vectors of features, and compare documents by measuring the distance between these features. The difference between all the algorithms is their ways to compute features that capture the semantics of documents and multiple algorithms to capture dependency structure of documents to focus on meanings of documents.

Considering in real-world interview cases where the interviewer's understanding of this question will be measured mainly on how many key concepts he could explain and illustrate, I focused on the word occurrence itself rather than semantics analysis. I set up the sentence similarity algorithm that simply takes the word sets of two sentences and calculates an index, which borrowed the idea from jaccard similarity. I took the intersection of A and B divided by B ($A \cap B / B$). It will get a degree between 0~1 which indicates the percentage of key words user has covered to this question compared to the correct answers.

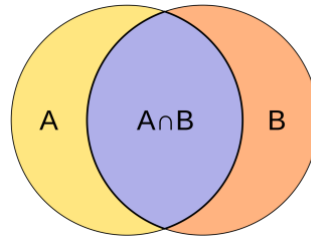


Figure 14: Relationship diagram

```
def jaccard_similarity(self, list1, list2):
    s1 = set(list1)
    s2 = set(list2)
    return len(s1.intersection(s2)) / len(s2)
```

Figure 15: Sentence similarity code screenshot

After normalizing the degree, I send both the score and correct answer back to the user as the submission of the form. The form action is complete.

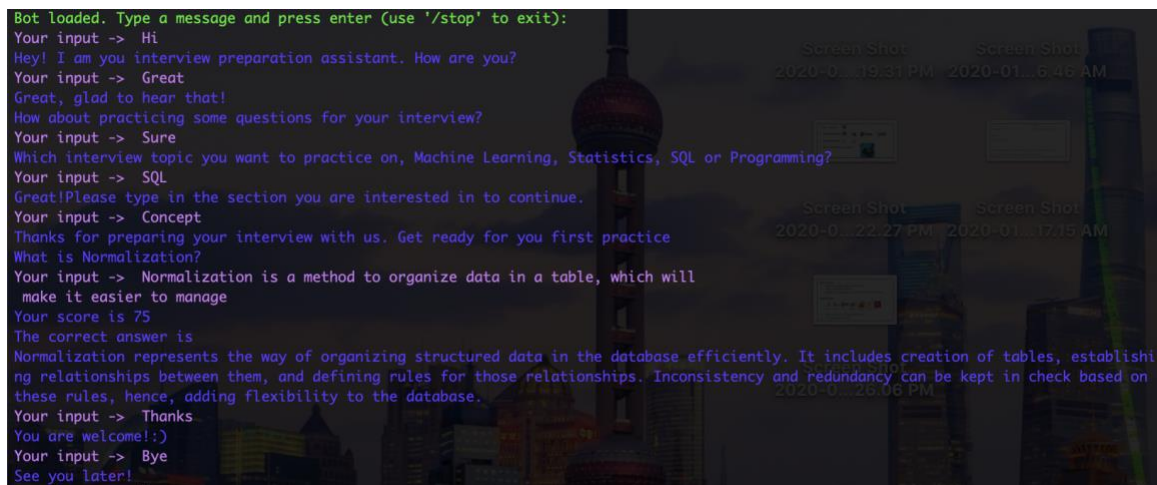


Figure 16: Interview Q&A dialogue screenshot

4.3 Web Application

In order to integrate the chatbot app with a website interface, I used flask framework to develop an introduction page. I set the chatbot in a toggle window and defined a click

function to trigger the event. Once the user clicks on the icon, the dialogue window will hover up initiating a chat session. The chat window is connecting to post and get request from another route defined in app. I used requests library in Python to send post request to chatbot API and receive response in that chat route.

Figure 17 and 18 demonstrates the completed web interface with chatbot responding to user requests.

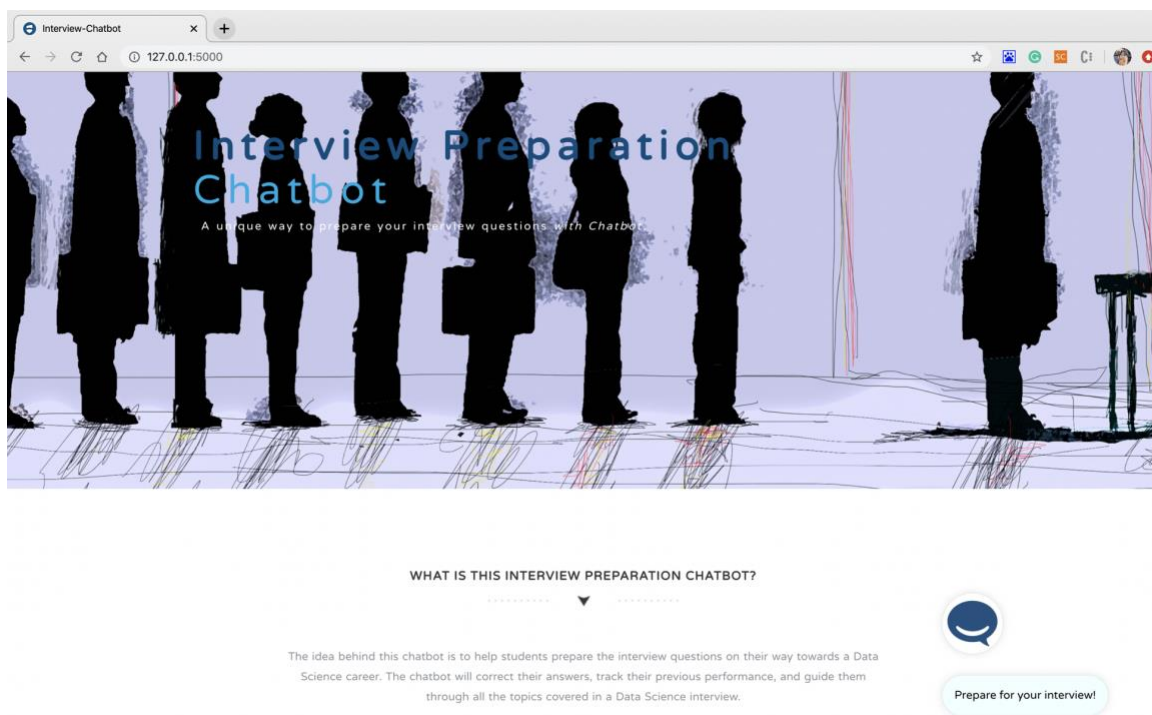


Figure 17: Web homepage screenshot

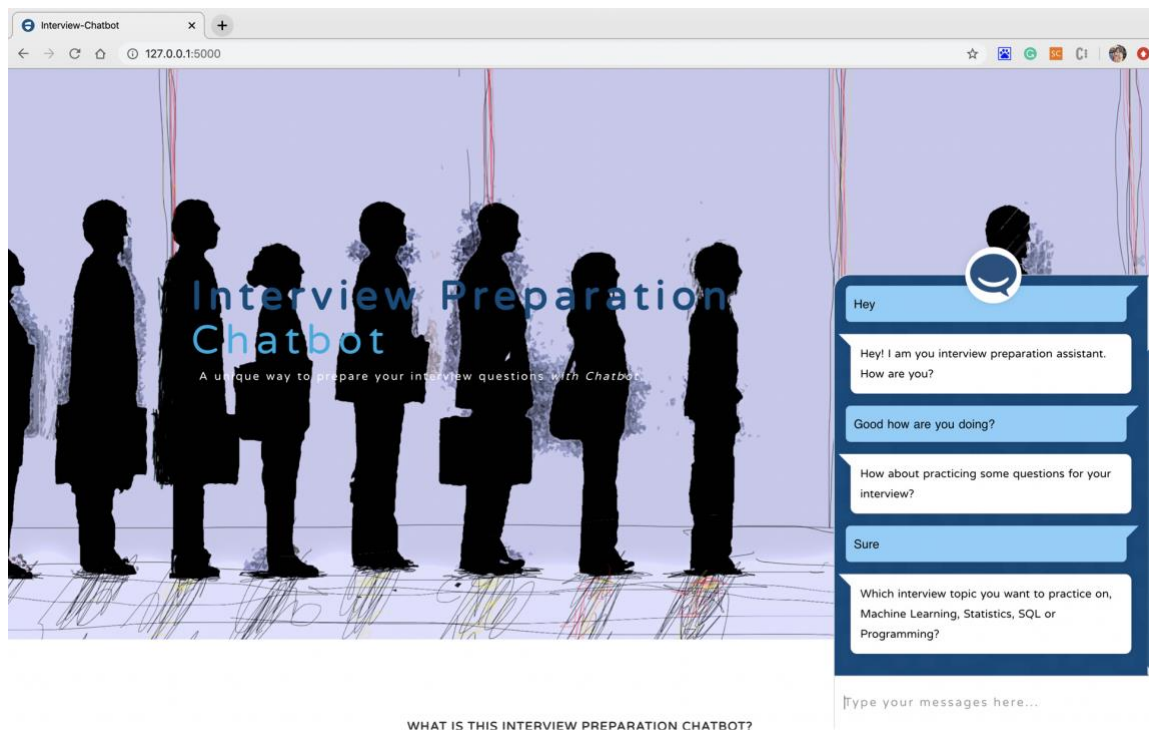


Figure 18: Web chat window screenshot

5. CONCLUSION

This master project achieved its proposed goal of designing and developing a web application prompting interactive interview conversation with a chatbot integration. It deployed a popular open source chatbot framework into a web interface and customized the bot into an interview assistant setting. Several advanced features such as tracking the dialogue flow, user input validation, scoring the users' answers based on sentence similarity are developed. More advanced intelligent function that could store interview progress in a session could be added in the future.

In the system design part, the interview preparation system workflow has been demonstrated with a diagram. Following is the system development part where this paper illustrated the deployment and customization of the chatbot integration in details. It introduced the architecture, core models, training data, and flow of the bot framework, as well as their roles that fit in this project scenario. A demo of the sample conversation has been provided in a screenshot.

The main limitation of this project is the usability of this chatbot. The core models of rasa framework have been set up with several natural language processing and classification models. But which model behaves the best in our scenario have not been addressed.

Much more work on the parameters tuning and model selection should be included to get a higher accuracy in processing users' languages.

Future work could be focused on two parts. First, this interview assistant application needs to store each users' selection and answers in a session. With every unique IP address corresponding to a user session, the chatbot could keep track of the users' past performance and avoid giving repeated interview question obtaining a better user experience. Second, a chatbot testing and evaluation should be conducted to better achieve this project goal. Possible solution for this could be setting up a pre-designed user input with expected dialogue flow, and evaluating on how well the bot would understand users' intent and give the right logical response. To reflect on how well the natural language processing models are trained, human-like language should be as tricky as possible. The flexibility of the bot needs to be tested upon how well it behaves with a change of topic or flow.

6. REFERENCES

1. Weizenbaum, J. (1966). ELIZA—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1), 36-45.
2. Garber, M. (2014). When PARRY met ELIZA: A ridiculous chat bot conversation from 1972. *The Atlantic*.
3. Wallace, R. (2005). Alice artificial intelligence foundation, from <http://www.alicebot.org>
4. Vargas-Vera, M., & Lytras, M. D. (2010). Aqua: A closed-domain question answering system. *Information systems management*, 27(3), 217-225.
5. Quarteroni, S., & Manandhar, S. (2009). Designing an interactive open-domain question answering system. *Natural Language Engineering*, 15(1), 73-95.
6. Burger, J., Cardie, C., Chaudhri, V., Gaizauskas, R., Harabagiu, S., Israel, D., ... & Moldovan, D. (2001, January). Issues, tasks and program structures to roadmap research in question & answering (Q&A). In *Document Understanding Conferences Roadmapping Documents* (pp. 1-35).
7. Schumaker, R. P., & Chen, H. (2007). Leveraging Question Answer technology to address terrorism inquiry. *Decision Support Systems*, 43(4), 1419-1430.

8. Meffert, K. (2006, March). Supporting design patterns with annotations. In 13th Annual IEEE International Symposium and Workshop on Engineering of Computer-Based Systems (ECBS'06) (pp. 8-pp). IEEE.
9. Bradeško, L., & Mladenić, D. (2012, October). A survey of chatbot systems through a loebner prize competition. In Proceedings of Slovenian Language Technologies Society Eighth Conference of Language Technologies (pp. 34-37).
10. Lebeuf, C., Storey, M. A., & Zagalsky, A. (2017). Software bots. *IEEE Software*, 35(1), 18-23.
11. Harms, J. G., Kucherbaev, P., Bozzon, A., & Houben, G. J. (2018). Approaches for dialog management in conversational agents. *IEEE Internet Computing*, 23(2), 13-22.
12. Bocklisch, T., Faulkner, J., Pawlowski, N., & Nichol, A. (2017). Rasa: Open source language understanding and dialogue management. arXiv preprint arXiv:1712.05181.
13. Botpress. 2019. Botpress - A Chatbot Maker & Development Framework <https://botpress.com/>
14. Cahn, J. (2017). CHATBOT: Architecture, design, & development. University of Pennsylvania School of Engineering and Applied Science Department of Computer and Information Science

7. APPENDIX

7.1 Sample Dialogue with Testbot

7.1.1 Dialogue 1

User: Hi.

Testbot: Hey! I am your interview preparation assistant. How are you?

User: I'm good

Testbot: Great, glad to hear that!

How about practicing some questions for your interview?

User: Sure

Testbot: Which interview topic you want to practice on, Machine Learning, Statistics, SQL or Programming?

User: SQL

Testbot: Great! Please type in the section you are interested in to continue.

User: Concept

Testbot: Thanks for preparing your interview with us. Get ready for your first practice. What is normalization?

User: Normalization is a method to organize data in a table, which will make it easier to manage.

Testbot: Your score is 75. The correct answer is Normalization represents the way of organizing structured data in the database efficiently. It includes creation of tables, establishing relationships between them, and defining rules for those relationships. Inconsistency and redundancy can be kept in check based on these rules.

User: Thanks

Testbot: You are welcome! 😊

User: Bye

Testbot: See you later!

7.1.1 Dialogue 2

User: Good morning

Testbot: Hey! I am your interview preparation assistant. How are you?

User: I want to practice on my interview

Testbot: Which interview topic you want to practice on, Machine Learning, Statistics, SQL or Programming?

User: Machine Learning

Testbot: Great! Please type in the section you are interested in to continue.

User: Supervised Learning

Testbot: Thanks for preparing your interview with us. Get ready for your first practice. What is 'Overfitting' in Machine learning?

User: Overfitting happens when the model has been trained too much to fit the training data, it has a perfect accuracy on training set but low accuracy on testing set.

Testbot: Your score is 69. The correct answer is when a statistical model describes random error or noise instead of underlying relationship 'overfitting' occurs. When a model is excessively complex, overfitting is normally observed, because of having too many parameters with respect to the number of training data types. The model exhibits poor performance which has been overfitting.

User: okay thank you goodbye

Testbot: See you later!

7.1.1 Dialogue 3

User: hi again

Testbot: Hey! I am your interview preparation assistant. How are you?

User: Beautiful day, isn't it?

Testbot: How about practicing some questions for your interview? Will that help you?

User: um yes

Testbot: Which interview topic you want to practice on, Machine Learning, Statistics, SQL or Programming?

User: Programming

Testbot: Great! Please type in the section you are interested in to continue.

User: Data Structure

Testbot: Thanks for preparing your interview with us. Get ready for your first practice. Describe a hash table.

User: Hash table is a data structure that everything is stored based on certain encryption rules. Each value will be paired with a key, and the key is generated by a hash function. Once you retrieve the value, you match it with its key.

Testbot: Your score is 86. The correct answer is A hash table is a data structure that produces an associative array. A key is mapped to certain values through the use of a hash function. They are often used for tasks such as database indexing.

User: Amazing. Thanks

Testbot: You are welcome 😊

7.2 The Interview Question and Answer Dataset

interview_topic	section	question	answer
Machine Learning	Supervised Learning	What is 'Overfitting' in Machine learning?	In machine learning, when a statistical model describes random error or noise instead of underlying relationship 'overfitting' occurs. When a model is excessively complex, overfitting is normally observed, because of having too many parameters with respect to the number of training data types. The model exhibits poor performance which has been overfit.
Machine Learning	Supervised Learning	How can you avoid overfitting?	By using a lot of data overfitting can be avoided, overfitting happens relatively as you have a small dataset, and you try to learn from it. But if you have a small database and you are forced to come with a model based on that. In such situation, you can use a technique known as cross validation. In this method the dataset splits into two section, testing and training datasets,

			the testing dataset will only test the model while, in training dataset, the datapoints will come up with the model. In this technique, a model is usually given a dataset of a known data on which training (training data set) is run and a dataset of unknown data against which the model is tested. The idea of cross validation is to define a dataset to "test" the model in the training phase.
Machine Learning	Theory	What are the five popular algorithms of Machine Learning?	Decision Tree, Neural Network (back propagation), Nearest Neighbor, Support vector machine, Random Forest
Machine Learning	Theory	What are the different Algorithm techniques in Machine Learning?	Supervised Learning, Unsupervised Learning, Reinforcement Learning, Semi-supervised Learning
Machine Learning	Theory	What are the three stages to build the hypothesis or model in Machine Learning?	Model Building, Model Testing, Applying the model
Machine Learning	Theory	What is the difference between artificial learning and machine learning?	Designing and developing algorithms according to the behaviours based on empirical data are known as Machine Learning. While artificial intelligence in addition to machine learning, it also covers other aspects like knowledge representation, natural language processing, planning, robotics etc.
Machine Learning	Supervised Learning	What is classifier in machine learning?	A classifier in a Machine Learning is a system that inputs a vector of discrete or continuous feature values and outputs a single discrete value, the class.
Machine Learning	Theory	What is the difference between supervised learning and unsupervised learning?	Supervised learning requires training labeled data. For example, in order to do classification (a supervised learning task), you'll need to first label the data you'll use to train the model to classify data into your labeled groups. Unsupervised learning, in contrast, does not require labeling data explicitly.
Machine Learning	Unsupervised Learning	How is KNN different from k-means clustering?	K-Nearest Neighbors is a supervised classification algorithm, while k-means clustering is an unsupervised clustering algorithm. While the mechanisms may seem similar at first, what this really means is that in order for K-Nearest Neighbors to work, you need labeled data you want to classify an unlabeled point into (thus the nearest neighbor part). K-means clustering requires only

			<p>a set of unlabeled points and a threshold: the algorithm will take unlabeled points and gradually learn how to cluster them into groups by computing the mean of the distance between different points.</p> <p>The critical difference here is that KNN needs labeled points and is thus supervised learning, while k-means doesn't — and is thus unsupervised learning.</p>
Machine Learning	Evaluation	Explain how a ROC curve works.	<p>The ROC curve is a graphical representation of the contrast between true positive rates and the false positive rate at various thresholds. It's often used as a proxy for the trade-off between the sensitivity of the model (true positives) vs the fall-out or the probability it will trigger a false alarm (false positives).</p>
Machine Learning	Evaluation	Define precision and recall.	<p>Recall is also known as the true positive rate: the amount of positives your model claims compared to the actual number of positives there are throughout the data. Precision is also known as the positive predictive value, and it is a measure of the amount of accurate positives your model claims compared to the number of positives it actually claims</p>
Statistics	Theory	What is Bayes' Theorem? How is it useful in a machine learning context?	<p>Bayes' Theorem gives you the posterior probability of an event given what is known as prior knowledge.</p> <p>Mathematically, it's expressed as the true positive rate of a condition sample divided by the sum of the false positive rate of the population and the true positive rate of a condition.</p>
Statistics	Theory	Why is "Naive" Bayes naive?	<p>The Naive Bayes Algorithm is based on the Bayes Theorem. Bayes' theorem describes the probability of an event, based on prior knowledge of conditions that might be related to the event.</p> <p>The Algorithm is 'naive' because it makes assumptions that may or may not turn out to be correct.</p>
Machine Learning	Evaluation	What's the difference between Type I and Type II error?	<p>Type I error is a false positive, while Type II error is a false negative. Briefly stated, Type I error means claiming something has happened when it hasn't, while Type II error means that you claim nothing is happening when in fact something is.</p>

Machine Learning	Deep Learning	What is deep learning, and how does it contrast with other machine learning algorithms?	Deep learning is a subset of machine learning that is concerned with neural networks: how to use backpropagation and certain principles from neuroscience to more accurately model large sets of unlabelled or semi-structured data. In that sense, deep learning represents an unsupervised learning algorithm that learns representations of data through the use of neural nets.
Machine Learning	Theory	What's the difference between a generative and discriminative model?	A generative model will learn categories of data while a discriminative model will simply learn the distinction between different categories of data. Discriminative models will generally outperform generative models on classification tasks.
Machine Learning	Supervised Learning	How is a decision tree pruned?	Pruning is a technique in machine learning and search algorithms that reduces the size of decision trees by removing sections of the tree that provide little power to classify instances. So, when we remove sub-nodes of a decision node, this process is called pruning or opposite process of splitting. Pruning can happen bottom-up and top-down, with approaches such as reduced error pruning and cost complexity pruning. Reduced error pruning is perhaps the simplest version: replace each node. If it doesn't decrease predictive accuracy, keep it pruned. While simple, this heuristic actually comes pretty close to an approach that would optimize for maximum accuracy.
Machine Learning	Evaluation	What's the F1 score? How would you use it?	The F1 score is a measure of a model's performance. It is a weighted average of the precision and recall of a model, with results tending to 1 being the best, and those tending to 0 being the worst. You would use it in classification tests where true negatives don't matter much.
Machine Learning	Supervised Learning	When should you use classification over regression?	Classification produces discrete values and dataset to strict categories, while regression gives you continuous results that allow you to better distinguish differences between individual points. You would use classification over regression if you wanted your results to reflect the belongingness of data points in your dataset to certain explicit categories (ex: If you wanted to know

			whether a name was male or female rather than just how correlated they were with male and female names.)
Machine Learning	Supervised Learning	How do you ensure you're not overfitting with a model?	<p>1- Keep the model simpler: reduce variance by taking into account fewer variables and parameters, thereby removing some of the noise in the training data.</p> <p>2- Use cross-validation techniques such as k-folds cross-validation.</p> <p>3- Use regularization techniques such as LASSO that penalize certain model parameters if they're likely to cause overfitting.</p>
Machine Learning	Evaluation	What evaluation approaches would you work to gauge the effectiveness of a machine learning model?	<p>Removing collinear features.</p> <p>Performing PCA, ICA, or other forms of algorithmic dimensionality reduction.</p> <p>Combining features with feature engineering.</p>
Machine Learning	Theory	What are 3 ways of reducing dimensionality?	<p>Removing collinear features.</p> <p>Performing PCA, ICA, or other forms of algorithmic dimensionality reduction.</p> <p>Combining features with feature engineering.</p>
Machine Learning	Theory	If you split your data into train/test splits, is it still possible to overfit your model?	<p>Yes, it's definitely possible. One common beginner mistake is re-tuning a model or training new models with different parameters after seeing its performance on the test set.</p> <p>In this case, it's the model selection process that causes the overfitting. The test set should not be tainted until you're ready to make your final selection.</p>
Machine Learning	Supervised Learning	What are the advantages and disadvantages of decision trees?	<p>Advantages: Decision trees are easy to interpret, nonparametric (which means they are robust to outliers), and there are relatively few parameters to tune.</p> <p>Disadvantages: Decision trees are prone to be overfit. However, this can be addressed by ensemble methods like random forests or boosted trees.</p>
Machine Learning	Supervised Learning	What are the advantages and disadvantages of neural networks?	<p>Advantages: Neural networks (specifically deep NNs) have led to performance breakthroughs for unstructured datasets such as images, audio, and video. Their incredible flexibility allows them to learn patterns that no other ML algorithm can learn.</p> <p>Disadvantages: However, they require a large amount of training data to converge. It's also difficult to pick the right architecture, and the internal "hidden" layers are incomprehensible.</p>

Machine Learning	Supervised Learning	How can you choose a classifier based on training set size?	If training set is small, high bias / low variance models (e.g. Naive Bayes) tend to perform better because they are less likely to be overfit. If training set is large, low bias / high variance models (e.g. Logistic Regression) tend to perform better because they can reflect more complex relationships.
Programming	Data Structure	What are some differences between a linked list and an array?	An array is an ordered collection of objects. A linked list is a series of objects with pointers that direct how to process them sequentially. An array assumes that every element has the same size, unlike the linked list. A linked list can more easily grow organically: an array has to be pre-defined or re-defined for organic growth. Shuffling a linked list involves changing which points direct where — meanwhile, shuffling an array is more complex and takes more memory.
Programming	Data Structure	Describe a hash table.	A hash table is a data structure that produces an associative array. A key is mapped to certain values through the use of a hash function. They are often used for tasks such as database indexing.
Programming	Python	Name native data structures in Python	Dictionaries, Lists, Sets, Strings, Tuples
Programming	Python	Of Python data structures, which are mutable and which are immutable?	Lists, dictionaries, and sets are mutable. This means that you can change their content without changing their identity. Strings and tuples are immutable, as their contents can't be altered once they're created.
Programming	Python	Is There a Way to Get a List of All the Keys in a Dictionary? If So, How Would You Do It?	Mydict.keys()
Programming	Python	Can You Explain What a List or Dict Comprehension Is?	When you need to create a new list from other iterables, you have to use list comprehensions. As lists comprehensions return list results, they will be made up of brackets that contain the expressions that need to be executed for each element. Along with the loop, these can be iterated over each element. Example of the basic syntax: new_list = [expression for_loop_one_or_more conditions]
Programming	Python	When Would You Use a	A list is a common data type that is

		List vs. a Tuple vs. a Set in Python?	<p>highly flexible. It can store a sequence of objects that are mutable, so it's ideal for projects that demand the storage of objects that can be changed later.</p> <p>A tuple is similar to a list in Python, but the key difference between them is that tuples are immutable. They also use less space than lists and can only be used as a key in a dictionary. Tuples are a perfect choice when you want a list of constants.</p> <p>Sets are a collection of unique elements that are used in Python. Sets are a good option when you want to avoid duplicate elements in your list. This means that whenever you have two lists with common elements between them, you can leverage sets to eliminate them.</p>
Programming	Python	What Packages in the Standard Library, Useful for Data Science Work, Do You Know?	<p>NumPy NumPy (or Numerical Python) is one of the principle packages for data science applications. It's often used to process large multidimensional arrays, extensive collections of high-level mathematical functions, and matrices. Implementation methods also make it easy to conduct multiple operations with these objects. There have been many improvements made over the last year that have resolved several bugs and compatibility issues. NumPy is popular because it can be used as a highly efficient multi-dimensional container of generic data. It's also an excellent library as it makes data analysis simple by processing data faster while using a lot less code than lists.</p> <p>Pandas Pandas is a Python library that provides highly flexible and powerful tools and high-level data structures for analysis. Pandas is an excellent tool for data analytics because it can translate highly complex operations with data into just one or two commands. Pandas comes with a variety of built-in methods for combining, filtering, and grouping data. It also boasts time-series functionality that is closely followed by remarkable speed indicators.</p> <p>SciPy SciPy is another outstanding library for scientific computing. It's based on NumPy and was created to extend its capabilities. Like NumPy, SciPy's data</p>

			structure is also a multidimensional array that's implemented by NumPy. The SciPy package contains powerful tools that help solve tasks related to integral calculus, linear algebra, probability theory, and much more.
Machine Learning	Unsupervised Learning	Explain Latent Dirichlet Allocation (LDA).	<p>Latent Dirichlet Allocation (LDA) is a common method of topic modeling, or classifying documents by subject matter.</p> <p>LDA is a generative model that represents documents as a mixture of topics that each have their own probability distribution of possible words.</p> <p>The "Dirichlet" distribution is simply a distribution of distributions. In LDA, documents are distributions of topics that are distributions of words.</p>
Machine Learning	Unsupervised Learning	Explain Principle Component Analysis (PCA).	<p>PCA is a method for transforming features in a dataset by combining them into uncorrelated linear combinations. These new features, or principal components, sequentially maximize the variance represented (i.e. the first principal component has the most variance, the second principal component has the second most, and so on).</p> <p>As a result, PCA is useful for dimensionality reduction because you can set an arbitrary variance cutoff.</p>
Machine Learning	Ensemble Learning	Explain bagging	<p>Bagging, or Bootstrap Aggregating, is an ensemble method in which the dataset is first divided into multiple subsets through resampling. Then, each subset is used to train a model, and the final predictions are made through voting or averaging the component models.</p> <p>Bagging is performed in parallel.</p>
Statistics	Theory	What do you understand by the term Normal Distribution?	<p>Data is usually distributed in different ways with a bias to the left or to the right or it can all be jumbled up. However, there are chances that data is distributed around a central value without any bias to the left or right and reaches normal distribution in the form of a bell-shaped curve.</p>
Statistics	Theory	What is correlation and covariance in statistics?	<p>Correlation: Correlation is considered or described as the best technique for measuring and also for estimating the quantitative relationship between two variables. Correlation measures how</p>

			<p>strongly two variables are related.</p> <p>Covariance: In covariance two items vary together and it's a measure that indicates the extent to which two random variables change in cycle. It is a statistical term; it explains the systematic relation between a pair of random variables, wherein changes in one variable reciprocal by a corresponding change in another variable.</p>
Statistics	Theory	What is the difference between Point Estimates and Confidence Interval?	<p>Point Estimation gives us a particular value as an estimate of a population parameter. Method of Moments and Maximum Likelihood estimator methods are used to derive Point Estimators for population parameters.</p> <p>A confidence interval gives us a range of values which is likely to contain the population parameter. The confidence interval is generally preferred, as it tells us how likely this interval is to contain the population parameter. This likeliness or probability is called Confidence Level or Confidence coefficient and represented by $1 - \alpha$, where α is the level of significance.</p>
Statistics	Theory	What is the goal of A/B Testing?	<p>It is a hypothesis testing for a randomized experiment with two variables A and B.</p> <p>The goal of A/B Testing is to identify any changes to the web page to maximize or increase the outcome of interest. A/B testing is a fantastic method for figuring out the best online promotional and marketing strategies for your business. It can be used to test everything from website copy to sales emails to search ads</p>
Statistics	Theory	What is p-value?	<p>When you perform a hypothesis test in statistics, a p-value can help you determine the strength of your results. p-value is a number between 0 and 1. Based on the value it will denote the strength of the results. The claim which is on trial is called the Null Hypothesis. Low p-value (≤ 0.05) indicates strength against the null hypothesis which means we can reject the null Hypothesis. High p-value (≥ 0.05) indicates strength for the null hypothesis which means we can accept the null Hypothesis p-value of 0.05 indicates the Hypothesis could go either way.</p>

Statistics	Problem	How can you generate a random number between 1 – 7 with only a die?	To get our 7 equal outcomes we have to reduce this 36 to a number divisible by 7. We can thus consider only 35 outcomes and exclude the other one. A simple scenario can be to exclude the combination (6,6), i.e., to roll the die again if 6 appears twice
Statistics	Problem	A certain couple tells you that they have two children, at least one of which is a girl. What is the probability that they have two girls?	From the question, we can exclude the first case of BB. Thus from the remaining 3 possibilities of BG, GB & BB, we have to find the probability of the case with two girls. Thus, $P(\text{Having two girls given one girl}) = 1 / 3$
Statistics	Theory	What do you understand by statistical power of sensitivity and how do you calculate it?	Sensitivity is commonly used to validate the accuracy of a classifier (Logistic, SVM, Random Forest etc.). Sensitivity is nothing but “Predicted True events/ Total events”. True events here are the events which were true and model also predicted them as true. Calculation of seasonality is pretty straightforward. $\text{Seasonality} = (\text{True Positives}) / (\text{Positives in Actual Dependent Variable})$
Statistics	Theory	What is regularisation? Why is it useful?	Regularisation is the process of adding tuning parameter to a model to induce smoothness in order to prevent overfitting. This is most often done by adding a constant multiple to an existing weight vector. This constant is often the L1(Lasso) or L2(ridge). The model predictions should then minimize the loss function calculated on the regularized training set.
Statistics	Theory	What Are Confounding Variables?	In statistics, a confounder is a variable that influences both the dependent variable and independent variable.
Statistics	Theory	What Are the Types of Biases That Can Occur During Sampling?	Selection bias Under coverage bias Survivorship bias
Statistics	Theory	What is Survivorship Bias?	It is the logical error of focusing aspects that support surviving some process and casually overlooking those that did not work because of their lack of prominence. This can lead to wrong conclusions in numerous different means.
Statistics	Theory	What is selection Bias?	Selection bias occurs when the sample obtained is not representative of the population intended to be analysed.
Statistics	Theory	What is TF/IDF	TF-IDF is short for term frequency-

		vectorization?	inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. It is often used as a weighting factor in information retrieval and text mining. The TF-IDF value increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general.
Statistics	Theory	Why we generally use Softmax non-linearity function as last operation in-network?	It is because it takes in a vector of real numbers and returns a probability distribution. It should be clear that the output is a probability distribution: each element is non-negative and the sum over all components is 1.
Machine Learning	Evaluation	Can you explain the difference between a Validation Set and a Test Set?	A Validation set can be considered as a part of the training set as it is used for parameter selection and to avoid overfitting of the model being built. On the other hand, a Test Set is used for testing or evaluating the performance of a trained machine learning model. In simple terms, the differences can be summarized as; training set is to fit the parameters i.e. weights and test set is to assess the performance of the model i.e. evaluating the predictive power and generalization.
Machine Learning	Supervised Learning	Name algorithms of supervised learning	Support Vector Machines, Regression, Naive Bayes, Decision Trees, K-nearest Neighbor Algorithm and Neural Networks
Machine Learning	Unsupervised Learning	Name algorithms of unsupervised learning	Clustering, Anomaly Detection, Neural Networks and Latent Variable Models
Machine Learning	Supervised Learning	Explain SVM algorithm in detail.	SVM stands for support vector machine, it is a supervised machine learning algorithm which can be used for both Regression and Classification. If you have n features in your training data set, SVM tries to plot it in n-dimensional space with the value of each feature being the value of a particular coordinate. SVM uses hyperplanes to separate out different classes based on the provided kernel function.
Machine Learning	Supervised Learning	What is logistic regression?	Logistic Regression often referred to as the logit model is a technique to predict the binary outcome from a linear combination of predictor variables.

Machine Learning	Supervised Learning	What Are the Drawbacks of the Linear Model?	The assumption of linearity of the errors. It can't be used for count outcomes or binary outcomes There are overfitting problems that it can't solve
Machine Learning	Ensemble Learning	Describe boosting	Boosting is an iterative technique which adjusts the weight of an observation based on the last classification. If an observation was classified incorrectly, it tries to increase the weight of this observation and vice versa. Boosting in general decreases the bias error and builds strong predictive models. However, they may over fit on the training data.
Machine Learning	Supervised Learning	What is a Random Forest? How does it work?	Random forest is a versatile machine learning method capable of performing both regression and classification tasks. It is also used for dimensionality reduction, treats missing values, outlier values. It is a type of ensemble learning method, where a group of weak models combine to form a powerful model. In Random Forest, we grow multiple trees as opposed to a single tree. To classify a new object based on attributes, each tree gives a classification. The forest chooses the classification having the most votes(Overall the trees in the forest) and in case of regression, it takes the average of outputs by different trees.
Machine Learning	Deep Learning	Describe the structure of Artificial Neural Networks?	Artificial Neural Networks works on the same principle as a biological Neural Network. It consists of inputs which get processed with weighted sums and Bias, with the help of Activation Functions.
Machine Learning	Deep Learning	How Are Weights Initialized in a Network?	There are two methods here: we can either initialize the weights to zero or assign them randomly. Initializing all weights to 0: This makes your model similar to a linear model. All the neurons and every layer perform the same operation, giving the same output and making the deep net useless. Initializing all weights randomly: Here, the weights are assigned randomly by initializing them very close to 0. It gives better accuracy to the model since every neuron performs different computations. This is the most commonly used method.
Machine Learning	Deep Learning	What Is the Cost	Also referred to as "loss" or "error," cost

Learning	Learning	Function?	function is a measure to evaluate how good your model's performance is. It's used to compute the error of the output layer during backpropagation. We push that error backwards through the neural network and use that during the different training functions.
Machine Learning	Deep Learning	What Are Hyperparameters?	With neural networks, you're usually working with hyperparameters once the data is formatted correctly. A hyperparameter is a parameter whose value is set before the learning process begins. It determines how a network is trained and the structure of the network (such as the number of hidden units, the learning rate, epochs, etc.).
Machine Learning	Deep Learning	What Will Happen If the Learning Rate Is Set inaccurately (Too Low or Too High)?	When your learning rate is too low, training of the model will progress very slowly as we are making minimal updates to the weights. It will take many updates before reaching the minimum point. If the learning rate is set too high, this causes undesirable divergent behaviour to the loss function due to drastic updates in weights. It may fail to converge (model can give a good output) or even diverge (data is too chaotic for the network to train).
SQL	Concept	What is the difference between SQL and MySQL?	SQL is a standard language for retrieving and manipulating structured databases. On the contrary, MySQL is a relational database management system, like SQL Server, Oracle or IBM DB2, that is used to manage SQL databases.
SQL	Concept	What are Constraints in SQL?	Constraints are used to specify the rules concerning data in the table. It can be applied for single or multiple fields in an SQL table during creation of table or after creating using the ALTER TABLE command. The constraints are: NOT NULL - Restricts NULL value from being inserted into a column. CHECK - Verifies that all values in a field satisfy a condition. DEFAULT - Automatically assigns a default value if no value has been specified for the field. UNIQUE - Ensures unique values to be inserted into the field. INDEX - Indexes a field providing faster retrieval of records. PRIMARY KEY - Uniquely identifies

			each record in a table. FOREIGN KEY - Ensures referential integrity for a record in another table.
SQL	Concept	What are some common clauses used with SELECT query in SQL?	WHERE clause in SQL is used to filter records that are necessary, based on specific conditions. ORDER BY clause in SQL is used to sort the records based on some field(s) in ascending (ASC) or descending order (DESC). GROUP BY clause in SQL is used to group records with identical data and can be used in conjunction with some aggregation functions to produce summarized results from the database. HAVING clause in SQL is used to filter records in combination with the GROUP BY clause. It is different from WHERE, since WHERE clause cannot filter aggregated records.
SQL	Concept	What are UNION, MINUS and INTERSECT commands?	The UNION operator combines and returns the result-set retrieved by two or more SELECT statements. The MINUS operator in SQL is used to remove duplicates from the result-set obtained by the second SELECT query from the result-set obtained by the first SELECT query and then return the filtered results from the first. The INTERSECT clause in SQL combines the result-set fetched by the two SELECT statements where records from one match the other and then returns this intersection of result-sets.
SQL	Concept	What is Cursor? How to use a Cursor?	A database cursor is a control structure that allows for traversal of records in a database. Cursors, in addition, facilitates processing after traversal, such as retrieval, addition and deletion of database records. They can be viewed as a pointer to one row in a set of rows. DECLARE a cursor after any variable declaration. The cursor declaration must always be associated with a SELECT Statement. Open cursor to initialize the result set. The OPEN statement must be called before fetching rows from the result set. FETCH statement to retrieve and move to the next row in the result set. Call the CLOSE statement to deactivate the cursor. Finally use the DEALLOCATE statement to delete the cursor definition

			and release the associated resources.
SQL	Concept	List the different types of relationships in SQL.	<p>One-to-One - This can be defined as the relationship between two tables where each record in one table is associated with the maximum of one record in the other table.</p> <p>One-to-Many & Many-to-One - This is the most commonly used relationship where a record in a table is associated with multiple records in the other table.</p> <p>Many-to-Many - This is used in cases when multiple instances on both sides are needed for defining a relationship.</p> <p>Self Referencing Relationships - This is used when a table needs to define a relationship with itself.</p>
SQL	Concept	What is Normalization?	Normalization represents the way of organizing structured data in the database efficiently. It includes creation of tables, establishing relationships between them, and defining rules for those relationships. Inconsistency and redundancy can be kept in check based on these rules, hence, adding flexibility to the database.
SQL	Concept	What are the TRUNCATE, DELETE and DROP statements?	DELETE statement is used to delete rows from a table. TRUNCATE command is used to delete all the rows from the table and free the space containing the table. DROP command is used to remove an object from the database. If you drop a table, all the rows in the table is deleted and the table structure is removed from the database.
SQL	Concept	What is Collation? What are the different types of Collation Sensitivity?	<p>Collation refers to a set of rules that determine how data is sorted and compared. Rules defining the correct character sequence are used to sort the character data. It incorporates options for specifying case-sensitivity, accent marks, kana character types and character width. Below are the different types of collation sensitivity:</p> <p>Case sensitivity: A and a are treated differently.</p> <p>Accent sensitivity: a and á are treated differently.</p> <p>Kana sensitivity: Japanese kana characters Hiragana and Katakana are treated differently.</p> <p>Width sensitivity: Same character represented in single-byte (half-width) and double-byte (full-width) are treated differently.</p>

			Collation can be set at different levels in SQL Server. Below are the three levels: SQL server level, database level, and column level.
--	--	--	---