Informe:

Trabajo practico

programacion exploratoria.

Universidad Nacional del Centro

Tema: Chatbot desarrollado en Rasa.

Alumnos: Viduzzi Franco, Garrido Andrés.

### Introducción.

Se pide implementar un agente inteligente con interfaz al usuario en formato de chat, pensado para un ambiente laboral en el que interactuara con los empleados. Dentro de estas interacciones se encuentran: saludar, asignación de tareas con características y requerimientos, pedir más tiempo para terminar una tarea, avisar su inasistencia indicando un motivo, informar su disconformidad con un compañero, pedir un cambio de horario, observar su perfil, observar su itinerario, dar aviso de una tarea completada y despedirse.

Para estas interacciones se implementó en lenguaje Python acciones lógicas, en JavaScript Object Notation los datos de los empleados y de las tareas, CSV para el análisis de datos y YAML para los datos de entrenamiento.

### Implementación del desarrollo.

Para el desarrollo del chatbot utilizamos los siguientes tipos de archivos:

* Python para realizar la parte lógica de las acciones. Principalmente nos enfocamos en una charla dinámica y amigable. Los pedidos se conceden fácilmente, de esa manera se trata de empatizar entre las partes.
* JSON para guardar la información de las tareas, las cuales contienen: nombre, Características, Requisito\_Lenguajes, Requisito\_Calidad, Semanas\_Restantes. Y por otra parte el archivo de Empleados que contiene: nombre, edad, años de experiencia, desempenio, lenguajes, faltas\_trabajo, pidió\_tiempo, molesto\_a, fue\_molestado\_por, cambio\_horario, Horario, Tarea.
* YAML para la implementación del entrenamiento al chatbot. Utilizamos las siguientes secciones en el dominio: slots, entities, intents, responses y actions.
* CSV para el análisis de datos con algoritmos de machine learning.

1. Características.

Las características principales del chatbot son:

* Cada empleado tiene una interacción propia, éstas se dan según las características de un empleado. Es decir, si un empleado pide tiempo extra varias veces, esto se almacena y la respuesta será según el historial entre el empleado y el chatbot.
* El sistema de archivos permite a otro software realizar modificaciones en el comportamiento de la charla. En caso de que un administrativo quisiera realizar cambios, por ejemplo, un despido o un nuevo empleado.
* Para que una conversación sea efectiva, el empleado debe identificarse y durante esa charla los datos que se ingresen se guardaran en el perfil de ese empleado. Por lo que el chatbot no permite una charla en simultaneo.

1. Aplicación de Machine Learning.

En este apartado decidimos utilizar un algoritmo de regresión lineal multivariable aprovechando la librería sklearn de Python para predecir un índice de probabilidad en la actividad de los empleados para que así el administrador pueda tenerlo en cuenta a la hora de contratar a un nuevo empleado, teniendo en cuenta su edad, género y desempeño.

Auto generamos una base de datos de 1000 empleados como para que el resultado sea un poco mas redundante y para la aplicación del algoritmo pasamos esos datos de un archivo json a un archivo csv, dadas las características de este tipo de archivo facilita mucho más el trabajo.

Formula aplicada:

Text, letter

Description automatically generated

Data sobre la cual se aplica el algoritmo:

A computer screen capture

Description automatically generated with medium confidence

En respuesta el administrador obtendrá una matriz de coeficientes y los índices de predicción (suponemos que el administrador posee conocimiento sobre el análisis de datos).

-Bibliografía utilizada para el apartado: -[Machine Learning - Tom M. Mitchell.](https://drive.google.com/file/d/1QlCjV4AEA8vRkjPVKQSqb0GeTUROyVPG/view) (enlace al libro)

1. Integración a Telegram.

Para la integración utilizamos el sistema propio de Telegram, el programa Ngrok para la conexión y además contratamos un host gratuito para que cualquier persona que tenga el contacto del bot pueda utilizarlo. Esto nos facilito mucho el desarrollo y además pudimos observar un mayor alcance a la utilidad del bot.

Para esto realizamos los siguientes pasos:

* En FreeNom conseguimos el nombre del dominio.

A picture containing timeline

Description automatically generated

* En InfinityFree anexamos el dominio a un DNS, esta página nos provee de servidores de poca capacidad, pero útiles para nuestro caso. Para el enlace utilizamos nombres personalizados dados por InfinityFree.

Graphical user interface, text, application, email

Description automatically generated

* Una vez realizada la conexión guardamos los archivos del bot en el servidor.

A picture containing graphical user interface

Description automatically generated

* Configuramos Ngrok para poder acceder a la url.

Graphical user interface, text, website

Description automatically generated

A screenshot of a computer

Description automatically generated

* Ese enlace lo colocamos en “config.yml” de Rasa.

Text

Description automatically generated

* Ahora utilizamos a “BotFather” para crear el bot en Telegram siguiendo los pasos de la [web](https://core.telegram.org/bots#privacy-mode).

Graphical user interface, text

Description automatically generated

* ¡Todo listo!

(Acá agregale captura/s de una charla)

1. Ejemplos.

A continuación, se muestra una interacción con un empleado. Utilizando la totalidad de las acciones.

Text

Description automatically generated

Como se puede observar, en empleado “Roberto” informo una disconformidad con otro empleado, cuando este se identifique el chatbot le notificara su comportamiento.

Text

Description automatically generated with medium confidence

A continuación, se muestran las negativas a los pedidos:

Tiempo prestado:

Text

Description automatically generated

Falta a la jornada laboral:

Graphical user interface, text

Description automatically generated

Cambio de turno:

Text

Description automatically generated

Doble asignación de tarea:

