# *“FACULTAD DE INGENIERIA Y TECNOLOGIA”*

***Materia: Fundamentos de Consultoría***

***Trabajo hecho por:***
***Saulo Bosquez***
***Ulises Jiménez Pérez***

***Nombre del Profesor: Kevin Arody Aguilar Ruiz.***

***Documento del Proyecto***

***Alumno de 8vo Semestre.***

# Comprensión del negocio

Se requiere crear una aplicación donde las personas puedan registrarse y encontrar amigos con sus mismos intereses. Para esto al entrar a la aplicación las personas tendrán que llenar un formulario con preguntas específicas que nos ayudarán a clasificar a las personas dependiendo de sus gustos.

El algoritmo para encontrar personas con gustos similares es parte del requerimiento de esta aplicación. La ciencia de datos será aplicada para resolver esta problemática usando algoritmos de clustering. El clustering consiste en la agrupación automática de datos. Es un tipo de aprendizaje automático no-supervisado.

**Objetivo del proyecto:**
Crear un modelo de aprendizaje automático no-supervisado para agrupar los perfiles de las personas dependiendo de sus intereses, este modelo se montará en una aplicación móvil con el fin de buscar hacer match con amigos o citas en línea.

# Enfoque analitico

**(escribir las  tecnologías que se ocuparan)**

Entendiendo los requerimientos para este proyecto se ha tomado la decisión de usar ciertas tecnologías y herramientas que se pueden utilizar para aplicar la ciencia de datos. El lenguaje por defecto para desarrollar este requerimiento es Python por su facilidad de uso en el machine learning. Dentro de Python podemos encontrar diversas librerías que nos ayudan para desarrollar el proyecto, tales como: gspread, sklearn, pandas, matplotlib, seaborn y collections.

La API gspread nos ayuda a extraer la información que se genera de nuestro formulario de google y que se almacena en google sheets. Esta información se trae teniendo la key que nos proporciona google sheets y los datos se almacenan en un data frame gracias a la librería Pandas, que también nos ayuda a manipular los data frames.

Sklearn es una biblioteca para aprendizaje automático de software libre para el lenguaje de programación Python. Con ella podemos entrenar nuestro modelo y probarlo con los datos que se encuentran en nuestro conjunto de datos. También lo usamos para estandarizar, escalar los valores de nuestro conjunto de datos, análisis de componentes principales y para la agrupación en clústeres de nuestro datos.

Matplotlib y Seaborn nos ayudan a graficar las clusters que hemos creado con las librerías pasadas. Seaborn en específico nos ayuda a darle mejor vista a nuestras gráficas. Collections nos sirve para crear mejores contenedores para nuestros datos.

# Requisitos de datos

**(detalles de la encuesta que se realizó)**

Para alimentar nuestro modelo de aprendizaje automático necesitamos un conjunto de datos con las características necesarias para cumplir con el objetivo previsto. Se necesita agrupar los perfiles de personas y se han hecho preguntas que toman un papel especial a la hora de formar los agrupamientos con base en la respuesta a estas. Las preguntas se han creado con base en los gustos y tendencias sociales de la actualidad. Al principio se tuvo un banco de 66 preguntas que podrían ser clave dentro nuestro algoritmo, pero con el tiempo estas preguntas fueron descartadas hasta llegar a la cantidad de 16 preguntas claves para clasificar tu perfil y agruparse con otros más. A continuación veremos a detalle las preguntas que se tomaron en cuenta para el algoritmo:

| Pregunta | Respuestas | ¿Por qué? |
|---|---|---|
| ¿Qué tan religios@ eres? | 1-5 | Queremos clasificar que tan religiosas son las personas de nuestra aplicación, por lo tanto, les decimos que se clasifiquen del 1 al 5 en qué tan religiosos son. |
| ¿Crees que las relaciones a distancia pueden funcionar? | 1-5 | Las relaciones a distancia son muy comunes y a muchas personas no les gusta, lo mejor es segmentar a las personas que quieren algo de cerca. |
| ¿Hablas de tus problemas personales con tus amigos cercanos? | 1-5 | La comunicación es importante en las relaciones, debemos de segmentar a estas personas para que no hayan problemas de comunicación en el futuro. |
| ¿Eres propenso a tomar decisiones impulsivas? | 1-5 | |
| ¿Quieres casarte? | 1-5 | |
| ¿Crees en los horóscopos y signos del zodiaco? | 1-5 | |
| ¿Qué tan consciente eres de tu salud? | 1-5 | |
| ¿Te gusta hablar sobre tus | 1-5 | |

| | | |
|---|---|---|
| problemas? | | |
| ¿Tu ego te impide disculparte cuando te equivocas? | 1-5 | |
| ¿Cómo resuelves los problemas con tus amigos/familiares? | dialogando, violencia, gritando, pidiendo ayuda, internalizando tus problemas | |
| ¿Con qué tipo de personalidad te identificas? | Orientado a objetivos, Orientados a relaciones, Orientado a detalles, Orientado a tareas, combinación de dos o más tipos. | |
| ¿Crees en hacer ejercicio? | 1-5 | |
| Si no estoy de acuerdo contigo en algunos temas, ¿cómo te sientes? | Me siento mal, A veces me afecta, Neutral, Se me pasa después de un tiempo, Ni me topa. | |
| Prefieres ver películas o leer libros | Películas, Más ver películas que leer, Ambos, Más leer que ver películas, Leer. | |

# Recopilación de datos.

Para la recopilación de datos hemos dispuesto de un formulario de google para que las personas puedan responder desde sus dispositivos móviles y la información que llegue a nosotros se guarde en una hoja de cálculo de google. Para dar a conocer nuestro proyecto y recopilar los primeros datos se realizó una pequeña campaña de publicidad dentro de la plataforma de instagram invitando a las personas a sumarse dentro de nuestra campaña de recolección de datos para la aplicación de cupidum. Esto conllevaba contestar el formulario completo. Para entrar al formulario debían entrar con su correo electrónico institucional, esto para estar seguros que solo las personas de la universidad contestaran y así cerrarnos a respuestas solamente de nuestra población. Nuestra campaña de recolección duró una semana y se consiguieron 82 registros de hombres y mujeres.

# Comprensión de datos

Los datos recopilados muestran una alta en los registros de mujeres que de hombres.

# Preparación de datos

Para la preparación de de los datos se tomaron en cuenta los siguientes requisitos:

- Las entradas de datos solo fueran numéricas.
- El rango de cada valor debería de estar entre 1-4.

En nuestro código tenemos que guardar los datos en un dataframe.

# Modelado

**(se requiere una explicación más detallada)**

Para la creación del modelo utilizado en este programa, fue necesario cambiar el tipo de variable de cadena a números. Esto se logró recorriendo el marco de datos creado y cambiando el tipo de variable de cadena a códigos categóricos. Una vez que este ciclo y el cambio del tipo de datos se completaron, se realizó el método MinMaxScaler para normalizar todo el marco de datos. Posteriormente, se realizó la eliminación de las columnas dirección de correo electrónico y sello de tiempo, ya que estas columnas no se tienen en cuenta al entrenar el modelo o al predecir el clúster al que puede pertenecer un individuo. A partir de aquí, las columnas del marco de datos se renombraron de frases a palabras clave, esto se hizo únicamente para facilitar la manipulación de datos.

A continuación, se creó una lista que contenía los nombres de las columnas recién renombradas para tomar estos nombres y asignarlos a una variable. Dada la gran cantidad de dimensiones que proporciona el conjunto de datos y el hecho de que la visualización de datos es imprescindible, se utilizó el análisis de componentes principales. El análisis de componentes principales, o PCA, es un procedimiento estadístico que permite resumir el contenido de la información en grandes tablas de datos por medio de un conjunto más pequeño de "índices de resumen" que se pueden visualizar y analizar más fácilmente. El número óptimo de componentes determinado para trabajar con los datos y los resultados es dos, por lo tanto, esta es la cantidad de componentes que se utilizaron. Una vez que el modelo PCA se construye usando el número correcto de componentes, este modelo también se normaliza usando la función fit_transform.

A continuación, se crea un nuevo marco de datos para mostrar los datos manipulados recientemente que resultan del modelo PCA. Desde aquí, una columna que contiene el sexo de nuestros participantes se concatena al marco de datos existente. Para que las respuestas sexuales resultantes sean números y no cadenas, se realiza la función label_encoder; las etiquetas resultantes luego reemplazan la cadena 'sexo'. Una vez que los datos y el modelo están en este punto, se pueden dividir para usarlos para entrenar y probar la precisión del modelo. Con estos resultados se entrena el modelo de Machine Learning. El

modelo que produjo los mejores resultados y, por lo tanto, se utilizó en este proyecto fue MiniBatchKMeans. Después de realizar la prueba de Elbow para determinar el número óptimo de conglomerados, se determinó que el número óptimo de conglomerados a utilizar debería ser 3. Una vez creado este modelo y pasado el número de conglomerados, se procedió a ajustar el modelo con los puntos de datos de 'X_train' e 'y_train'. Con el fin de utilizar este modelo para futuras clasificaciones de conglomerados, el modelo se cargó en un pickle. Después de confirmar que tanto el pickle como el modelo funcionaban bien, el paso final fue predecir el número de grupos para cada uno de los puntos de datos encontrados dentro del conjunto de datos. Una vez que esto se pronosticó, la columna final denominada Cluster se agregó al marco de datos. Para finalizar, se creó un diagrama de dispersión a partir de la información del modelo, que luego se mostró al usuario.

# Evaluación

# Implementación

La implementación de este modelo fue probado por primera vez en una máquina local, después de varias pruebas el proyecto se movió de local a ser montado en un servidor en la web. Este servidor se encuentra montado en un servidor de nube Linux, el cual facilita la funcionalidad de Python y sus funciones. El servidor web se llama PythonAnywhere, el cual requiere la creación de una cuenta gratuita para poder usar sus servicios. Una vez creada la cuenta, se tuvo que crear una web app, en la cual se pueden subir archivos de python y por medio de una consola bash instalar las paqueterias que son requeridas para ejecutar la aplicación. El código fue subido a la parte correspondiente y su ruta es la ruta por default. Después de indicar la ruta, se creó una lista de variables las cuales toman información del JSON que recibe el API. Una vez teniendo la lista de variables en formato JSON, esta se le agrega al modelo, se le cambia el formato de varios campos, se normaliza, y se hace un predict de todos los datos para saber en que cluster queda los datos proporcionados de cada nuevo usuario. Al saber en que cluster queda el usuario, esto se le regresa a la aplicación en forma de un entero. Teniendo en cuenta esta información, la aplicación ya puede empezar a hacer sugerencias de relaciones potenciales.

La manera en la cual este proyecto ayudó en la aplicación CUPIDUM fue de la siguiente manera. Este proyecto ayudó a hacer las conexiones entre personas las cuales tienen perfiles similares y buscan lo mismo. Las conexiones se crean mediante los clusters en los cuales las personas son clasificadas. La clasificación depende de las respuestas a las preguntas, las cuales se hacen en la creación de perfil en la aplicación. Después estas respuestas se introducen en el algoritmo y este retorna el número de cluster al cual pertenece el individuo. De esta manera este proyecto aporta a la aplicación CUPIDUM.

# Retroalimentación