Emocify (Generación de Playlists Basadas en el Reconocimiento de Emociones)

Rodrigo Alejandro Ochoa Herrera, Luis Antonio Medellín Serna

CENTRO UNIVERSITARIO DE CIENCIAS EXACTAS E INGENIERÍAS (CUCEI, UDG)

rodrigo.ochoa5222@alumnos.udg.mx
luis.medellin@academicos.udg.mx

***Abstract* – Emocify is a web application that combines emotions with music through facial expression recognition and personalized music recommendations. It leverages technologies such as computer vision and classification algorithms to detect a set of emotions through facial expressions. Integration with Spotify service generates personalized playlists for each user based on the detected mood. This document describes the technical architecture and development methodology of the web application, emphasizing the emotion classification processes as well as song recommendations based on mood.**

***Palabras clave* – Proyecto modular, reconocimiento de expresiones faciales, clasificación de emociones, recomendación musical, maquinas de vectores de soporte.**

I. INTRODUCCIÓN

Los humanos expresamos emociones y sentimientos mediante distintos mecanismos de comunicación, uno de los más universales siendo las expresiones faciales. Estas son en su mayoria autodescriptivas para los seres humanos, una cara de disgusto o felicidad es ampliamente reconocida por distintos gestos clave en nuestro rostro. Estas expresiones son una forma de comunicar un estado de ánimo o una emoción transitoria. El reconocimiento de estas expresiones es una nueva forma de interacción entre el ser humano y la máquina gracias a las tecnologías derivadas de la visión por computadora. De igual manera, nuestro estado anímico se ve reflejado en otros aspectos de nuestra vida cotidiana tales como la manera en que nos desplazamos, el como respondemos a situaciones, hasta que tipo de música escuchamos.

El escuchar canciones que se acoplan a nuestro estado anímico nos facilita el proceso de liberar de mejor manera nuestras emociones[1], en especial cuando nos encontramos en estados considerados negativos tales como la tristeza. La capacidad de poder generar recomendaciones basadas en el estado anímico y personalizadas al gusto musical de cada persona sería un aumento positivo a la experiencia de escuchar música.

Tomando esto como referencia, el proyecto Emocify ha sido desarrollado con este proposito en general, hacer la experiencia de escuchar música más personalizada y adecuada al contexto emocional actual de cada usuario.

En este artículo se detalla la arquitectura técnica de la plataforma web Emocify, explorando los algoritmos detrás del análisis de expresiones faciales y la generación de listas de reproducción personalizadas. Cómo se logra la interación con el servicio de streaming Spotify, permitiendo transformar las emociones capturadas en experiencias auditivas hechas a la medida de cada usuario.

En resumen, el proyecto no solamente plantea una aplicación web que genera listas de reproducción, propone una solución innovadora que aprovecha la expresión humana para crear una experiencia de esucha profunda, personalizada e íntima para cada usuario.

A. Planteamiento del Problema

Los servicios de streaming como Spotify ofrecen una biblioteca musical muy extensa, que para algunos usuarios puede ser un tanto abrumadora cuando se busca una experiencia más personalizada a nuestro contexto emocional actual, es decir, podemos reproducir en aleatorio y vamos a escuchar canciones que tal vez no se adecuan a nuestro estado de ánimo. De igual manera, podemos seleccionar manualmente las canciones y crear un ambiente óptimo, pero esto conlleva tiempo y dedicación.

B. Justificación

Ante esta situación, el proyecto contempla una selección de música especial para cada usuario, generando recomendaciones de manera casi automática basadas en el estado de ánimo y en el historial de escucha. Las soluciones que actualmente existen generan estas recomendaciones mediante palabras o emojis que el usuario ingresa para describir su estado anímico, siendo Emocify una nueva propuesta al generar estas recomendaciones basandose en la expresion facial del usuario. Este reconocimiento de emociones mediante visión por computadora sin duda alguna introduce una innovación a las alternativas que actualmente se pueden utilizar.

II. TRABAJOS RELACIONADOS

Este proyecto esta fuertemente inspirado en aplicaciones web como Receiptify o las mismas funcionalidades que ofrecen los servicios de streaming como el Spotify Wrapped. Estas plataformas lo que permiten al usuario es ver en retrospectiva su historial de escucha a lo

largo del tiempo y mostrar distintos datos como sus canciones más escuchadas, los artistas descubiertos últimamente, el tiempo que reproducieron contenido, etc. Esta interacción con el usuario es muy personal desde mi punto de vista, ya que nos muestra datos sobre nuestros hábitos o gustos de manera muy detallada. El punto débil que tienen estas plataformas es que la informacion es unidireccional, es decir, únicamente se da un resumen al usuario de sus hábitos de escucha pero no se le da mucho a cambio.

El reconocimiento de expresiones faciales es un ámbito ampliamente explorado y conocido. El principal objetivo de este enfoque de estudio es automatizar el proceso de determinación de emociones en tiempo real analizando características clave del rostro como los ojos, boca, cejas, entre otros y relacionarlos con un set de emociones como el enojo, miedo, sorpresa, tristeza y felicidad. Existen distintos acercamientos al desarrollo de soluciones de este tipo de problemáticas, algunos utilizando algoritmos de clasificación de la vieja escuela por llamarlo de alguna manera, y otros utilizando soluciones más modernas como las redes neuronales y deep learning.

Se puede tener una amplia variedad de soluciones que aplicar para el desarrollo de este proyecto. La metodología basada en el uso de redes neuronales queda como una opción no viable ya que no se tiene el dominio necesario para desarrollar este proyecto utilizando ese tipo de tecnologías. Además, la mayoría de guías o proyectos desarrollados con este método utilizan como medio de entrenamiento imágenes de sujetos, haciendo del entrenamiento un proceso muy pesado computacionalmente que requiere de recursos y tiempo, mismos que el proyecto no contempla a tal magnitud.

La solución ideal es basar el desarrollo del proyecto en la extracción de características clave del rostro y entrenar el modelo con ese vector de datos, pudiendo utilizar una gran variedad de algoritmos de clasificación que no requieren tanto poder computacional para su entrenamiento, mismas que ya han sido puestas a prueba en distintos desarrollos y plataformas de reconocimiento de expresiones.

III. DESCRIPCIÓN DEL DESARROLLO DEL PROYECTO MODULAR

A. Hipótesis

La clasificación de emociones se puede realizar mediante tecnologías de visión por computadora y algoritmos de aprendizaje, sirviendo como punto de partida para la recomendación de música basada en la emoción detectada.

B. Metodología

Al tener un objetivo bien definido y ser un proyecto con un desarrollo relativamente sencillo se optó por una metodología en cascada. Teniendo en cuenta los pros y contras que el uso de esta metodología conlleva se definieron los tiempos necesarios para cada etapa del desarrollo, con objetivos especificos que cumplir encada una de ellas para evitar problemas a futuro. Se plantearon como objetivos a cumplir los siguientes puntos:

1) Identificar expresiones faciales mediante una cámara web
2) Entrenar un modelo de aprendizaje que clasifique expresiones en 4 emociones (neutral, felicidad, tristeza, enojo)
3) Permitir al usuario acceder con su cuenta de Spotify
4) Generar recomendaciones (playlist) de acuerdo a la emoción detectada y al historial de escucha del usuario
5) Desarrollar una plataforma web como interfaz de usuario, contemplando una arquitectura de servicios distribuidos (REST API, hosting)

El conjunto de tecnologías utilizadas para el desarrollo de este proyecto contemplan los stacks necesarios para el despliegue de una aplicación web así como para el entrenamiento de modelos de aprendizaje en un entorno Python. Para el entrenamiento se utilizaron las herramientas de desarrollo Anaconda, VSCode y JupyterNotebook junto con librerías como MediaPipe, OpenCV y SciKit Learn. El desarrollo de la aplicación web se realizó en el framework de JavaScript VueJS en su version 3 tomando como estandar de desarrollo el Composition API. Para el consumo de servicios se optó por utilizar Node + Express para el backend general de la aplicación y Flask como un servicio dedicado para el modelo de aprendizaje. Para la persistencia de datos de la aplicación se implemento una base de datos orientada a documentos utilizando el servicio de Google Firebase. Por último, se consume el servicio REST API de Spotify para la recolección de datos del usuario y el inicio de sesión en la aplicación.

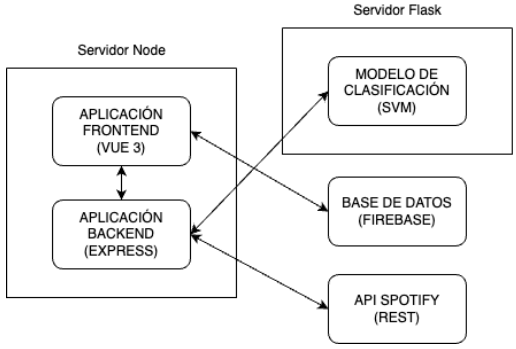

Fig 1. Distribución de servicios. Elaboración propia

C. Diseño y Fase del Prototipado del Proyecto Modular

El desarrollo del proyecto se dividió en tres etapas fundamentales para la optimización de tareas y la mantenibilidad del código fuente. Estas tres etapas contemplan desde la generación de datos para el entrenamiento del modelo hasta su despliegue en servicios web. A continuación se detallan a profundidad las metodologías, herramientas y problemas que se tuvieron en cada una de estas etapas.

1) *Extracción*. El objetivo principal de esta etapa es la obtención de puntos clave del rostro del usuario. Derivado de esto, se tiene que realizar el desarrollo de la aplicación web, ya que sin ella no es posible acceder al video de la camara web del dispositivo de manera interactuable con el usuario.

   El desarrollo de la aplicación como se mencionó anteriormente esta basado en el framework JavaScript VueJS tomando como estandar de desarrollo su versión 3 de Composition API y Vuex como gestor de estado. La interfaz de usuario se logró utilizando la librería de estilos de TailwindCSS.

Durante el desarrollo de la interfaz de usuario se tuvo que contemplar el cumplimiento de las politicas de uso de Spotify[2], ya que la plataforma Emocify hace uso de sus datos y el servicio requiere cumplir con sus guías de diseño. Estas normas incluyen el correcto uso de tipografías y logotipos de Spotify, así como mantener los datos e imágenes proporcionadas en su estado original.

Teniendo en cuenta la accesibilidad de la plataforma, se siguio la ideología MobileFirst, en la cual se optimizan y desarrollan las vistas de la página para dispositivos móviles primeramente, para despues adecuarlas a pantallas más grandes. De igual manera, se incluye un modo obscuro para cambiar la paleta de diseño de la aplicación y adecuarse a su uso durante la noche o situaciones de poca iluminacion. Esto meramente para mejorar la experiencia de usuario y ofrecer una interfaz moderna y accesible.

El framework de desarrollo utilizado para el frontend es Vue 3, mismo que basa su lógica de funcionamiento en SFC (Single File Components) para la concentración de todo el código, template HTML y estilos necesarios en un solo archivo correspondiente a un componente de la interfaz. Aprovechando esta lógica, lo ideal fue implementar el patrón de diseño MVVM (Modelo-Vista-Vista-Modelo). Este patrón representa los datos y lógica empresarial dentro del modelo, la vista-modelo presenta estos datos de una manera que se pueda tener comunicación entre la vista y el modelo, es decir, la interfaz de usuario. Por último la vista presenta de manera estructurada todo el conjunto de datos en la lógica de negocio.

Este patrón de diseño facilita la legibilidad y mantenimiento del código, aprovechando las funcionalidades de SFC que ofrece Vue de mejor manera. Cada componente en la interfaz está pensado para poder ser reutilizado, mantenible y lo suficientemente independiente de la lógica de negocio para poder ser utilizado en otros proyectos.

La aplicación al ser un SPA (Single Page Application) cumple con características que se esperan de un desarrollo web moderno. Los tiempos de carga se reducen considerablemente al únicamente descargar y requerir los componentes necesarios. Los bundles minificados son mucho más ligeros, y se minimizan la cantidad de peticiones de red a los servidores. Al ser una aplicación CSR (Client Side Rendering) todos los procesos de ejecución de scripts y librerías son ejecutados en el dispositivo del usuario, esto hace que de igual manera se reduzca el estrés de computo de servidores y servicios en red. Las librerías estan altamente optimizadas para su ejecución en dispositivos móviles sin ningun problema de rendimiento.

El diseño de la aplicación contempla el desarrollo de cinco vistas principales: vista de inicio de sesión, vista de inicio, vista de FaceScan y vistas de Moods y Playlists. El inicio de sesión se logra mediante la inclusión de un botón que redirecciona al servicio de Spotify para que el usuario otorgue los permisos de acceso a su información. Estos permisos incluyen:

- Datos básicos de la cuenta de Spotify
  - Email
  - Tipo de suscripción (gratis o premium)
  - Nombre de usuario y foto de perfil
- Actividad en Spotify
  - Contenido y artistas principales
  - Cuentas seguidas
  - Playlists creadas y seguidas
  - Playlists colaborativas
- Realizar acciones en Spotify
  - Crear, editar y seguir playlists

Estos permisos son claramente detallados al usuario previo a su autorización o denegación, el uso de estos datos puede cancelarse en cualquier momento desde la configuración de cuenta en el servicio de Spotify. Las consideraciones de privacidad y ética en el manejo de estos datos en este proyecto se basan enteramente en la aceptación de los términos de acceso a la información de Spotify.

Una vez ingresado a la plataforma se nos da la bienvenida con una pantalla de inicio en donde se muestran nuestros hábitos de escucha (lista de artistas seguidos y canciones más reproducidas) en forma de tarjetas visuales con la ilustración o fotografía del artista, títulos y enlaces a los correspondientes perfiles de Spotify. Además, un historial de registros de los escaneos que hayamos hecho anteriormente ordenados por fecha de manera descendente.

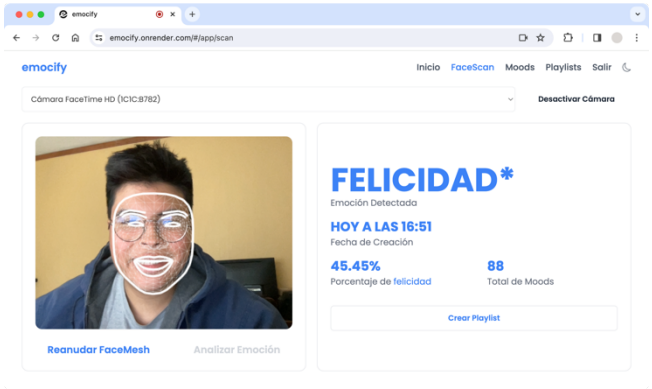

Fig 2. Vista FaceScan de Emocify. Elaboración propia

La vista donde se procesa la mayoría del trabajo de la aplicación es en la de FaceScan, en esta el usuario otorga los permisos de navegador para poder hacer uso de su cámara web para realizar el algoritmo de clasificación de expresiones faciales. Aquí el usuario verá un stream de video en tiempo real con una malla de puntos sobrepuesta en su cara y dos botones principales, uno para pausar la detección del rostro y otro para analizar y clasificar la expresión facial en una emoción. Una vez capturada la expresión deseada, el botón de analizar mandará la malla de puntos al servicio que hostea el modelo de clasificación y posteriormente mostrará la emoción inferida. Este proceso toma de 1 a 3 segundos dependiendo la velocidad de conexión a internet del usuario así como la capacidad de procesamiento del dispositivo. Cuando se tiene el resultado de la clasificación de puntos, se tiene la opción de generar una lista de reproducción basada en esa emoción detectada.

La vistas de Moods y Playlists muestran detalles de los escaneos realizados y una visualización interactuable de las playlist generadas. Los detalles mostrados en cada escaneo

incluyen la emoción detectada, una visualización de la malla de puntos faciales, la fecha de creación y el porcentaje del total de escaneos al que pertenece esa emoción. La visualización de playlists es un iFrame embebido desde el servicio de Spotify, con el cual se puede ver las canciones incluidas en la lista de reproducción así como escucharlas directamente en la plataforma Emocify.

Para la persistencia de datos de usuario así como los registros generados en la plataforma, se integró el gestor de base de datos orientada a documentos de Google Firebase. Aprovechando que la naturaleza de este gestor es ofrecido como un SaaS (software as a service) y es autogestionado en la plataforma de Google Firebase, no se tuvo la necesidad de desarrollar un dashboard de administración para la plataforma. Únicamente se realizaron las configuraciones necesarias para la autorización de la aplicación Emocify en el servicio de Google Firebase para el uso de su base de datos No-SQL.

Dada la naturaleza del proyecto y las necesidades de mantener información persistente, no se requiere de una arquitectura compleja con muchas entidades o relaciones entre sí. Es por esto que la tecnología dicta el tipo de base de datos a utilizar y no de la manera contraria. Se pudo optar por un sistema de base de datos SQL o por uno No-SQL, por un gestor o por otro. Sin embargo, dadas las virtudes de la integración con Google Firebase y la facilidad de uso, se adecuó el diseño de la base de datos para acoplarse de la mejor manera con este gestor.

El servicio de Firebase ofrece una solución moderna y amigable con la arquitectura de red, ya que permite una conexión y actualización en tiempo real a la base de datos. Esto permite tener sincronización casi instantanea entre múltiples sesiones que pueda tener el usuario, así como aligerar la carga de peticiones web realizadas a los servidores.

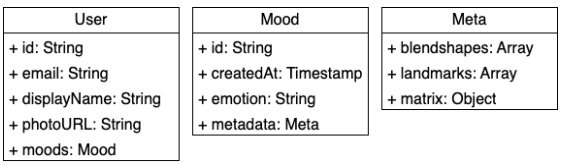

Fig 3. Adecuación de representación del modelado de base de datos. Elaboración propia

La arquitectura de la base de datos es muy ligera ya que únicamente se requiere la persistencia del usuario y los escaneos realizados. A continuación se muestra la representación de entidades de la base de datos, recalcando que al ser una base de datos orientada a documentos, las entidades Mood y Meta viven dentro del mismo documento User, pero para fines demostrativos se separaron en 3 entidades independientes.

El documento User se compone de cinco campos:

- Id: Corresponde al id de la cuenta de Spotify, esto garantiza que no haya cuentas duplicadas bajo una misma cuenta de streaming
- Email: Se obtiene de la información de perfil de la cuenta de Spotify y se almacena para facilitar la identificación de usuarios dentro de la base de datos
- DisplayName: Se usa como identificador amigable a la lectura, conocido como nombre de usuario
- PhotoURL: Se obtiene de la información de perfil de la cuenta de Spotify y se utiliza como avatar
- Moods: Esta propiedad es una colección de documentos que corresponden a la entidad de escaneos. En esta propiedad se almacenan los detalles de cada clasificación de emociones:
    - Id: Cada escaneo tiene un identificador único para ser incluido en la generación de las listas de reproducción y poder identificarlos facilmente en Spotify
    - CreatedAt: Es un timestamp para poder organizar y filtrar los escaneos por fecha de creación
    - Emotion: Es la emoción detectada en base a la clasificación de los puntos faciales
    - Metadata: Los metadatos del escaneo incluyen los vectores de puntos faciales, los blendshapes y la matriz de orientación tridimencional. Estos datos se utilizan para la visualización de la malla de puntos en los detalles de cada escaneo

Una vez desarrollada la interfaz de usuario se comenzó con la implementación del reconocimiento de expresiones faciales, o FaceScan como es llamado dentro de la aplicación. En este proceso se hizo uso de la librería MediaPipe en su modelo FaceLandmarker para web. Esta librería toma como entrada el stream de video del dispositivo y tiene como salida un objeto de vectores que contiene las coordenadas tridimensionales de los 478 puntos faciales clave, así como los 52 blendshapes utilizados para la clasificación de emociones en la etapa posterior.

Tanto la librería de MediaPipe para JavaScript como para Python son compatibles entre sí, esto es de suma importancia ya que el entrenamiento del modelo se realizó en un entorno Python pero la ejecución es en un entorno JavaScript y de esta manera se minimiza el trabajo necesario para hacer funcionar el modelo en distintos entornos de ejecución.

2) *Clasificación*. En esta etapa se realizó el entrenamiento del modelo de aprendizaje supervisado encargado de clasificar los vectores de la expresion facial en emociones.

Primeramente, se intentó utilizar datasets relacionados al campo del reconocimiento facial tales como Extended Cohn-Kanade Dataset (CK+)[3], Japanese Female Facial Expressions (JAFFE)[4], AffectNet[5], Real-World Affective Faces Database (RAF-DB)[6], Facial Expression Recognition Dataset (FER2013)[7], FACES[8], entre otros. Sin embargo, para la mayoría de estos datasets es necesaria una aprobación para su uso, mismas que fueron rechazadas o simplemente nunca se tuvo respuesta por parte de la organización.

Los datasets a los cuales si se tuvo acceso fueron CK+, FACES y FER2013, pero no fueron utilizados para el entrenamiento del

modelo de este proyecto. La razon por la cual no se utilizaron los datasets CK+ y FACES fue que se tenian pocas instancias para el entrenamiento y el dataset FER2013 no cumplia con los requerimientos de detección de la librería MediaPipe (imágenes con resolución mínima de 256x256). Se intentó un reescalado de las imágenes de este último dataset para cumplir con el requerimiento de la librería, pero las imágenes perdían mucha calidad y afectaban el rendimiento de la detección facial.

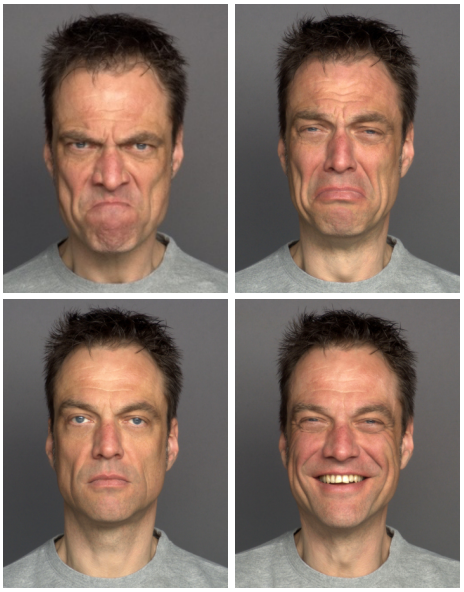

Fig 4. Representación de expresiones faciales. De izquierda-derecha-arriba-abajo: enojo, tristeza, neutral, felicidad.[8]

Ya que no se pudieron utilizar los datasets, se tuvo la necesidad de generar un dataset propio. Se desarrolló un pequeño programa en Python para capturar 4,000 imágenes (1,000 por cada emoción). Esta solución es muy limitada, ya que el número de instancias no es el adecuado para la problemática que se desea resolver y no se cuenta con una extensa variedad de rostros y expresiones. Teniendo el dataset generado y debidamente etiquetado, se optó por utilizar una máquina de vectores de soporte como modelo de aprendizaje supervisado. Inicialmente se contemplaron KNN y NaiveBayes ya que en distintos articulos[9, 10] similares a este fueron las opciones utilizadas en el desarrollo, pero el SVM mostró ser el más adecuado a la naturaleza de este proyecto.

La generación de este dataset consta principalmente de un script que captura imágenes de manera consecutiva y las almacena en directorios correspondientes a la emoción que se intenta capturar en esa imagen, es decir, el etiquetado de las instancias se basa en la organización de directorios. La captura de imágenes fue posible utilizando la librería OpenCV. Una vez que se tienen los directorios de imágenes, se aplicó el modelo de FaceLandmarker de la librería MediaPipe para cada una de las imágenes, obteniendo la malla de puntos y valores de blendshapes en forma de arreglos. Para el entrenamiento del modelo se genera un archivo CSV con los 52 valores de blendshapes y la etiqueta de clase para cada una de las instancias, estos dataframes se manejan con la librería Pandas.

Durante el desarrollo del entrenamiento, se tuvo una problemática que se resolvió posteriormente con una actualización de la librería MediaPipe. Inicialmente para la clasificación de emociones, se tomaba un vector de 1434 coordenadas (478 puntos de 3 coordenadas). Estas 1434 coordenadas corresponden a cada uno de los 478 puntos faciales detectados en el usuario, el principal problema de esto es que no todos los puntos tienen una gran relevacia para la clasificacion de emociones, la mayoria solamente son ruido. Además, los puntos estan directamente relacionados a la posición y orientación de la cara, por lo cual los modelos entrenados con estos vectores tenian un desempeño pobre.

En la actualización mencionada, se introdujo un nuevo vector con 52 caracteristicas más especificas llamadas blendshapes. Este vector contiene valores de 0 a 1 indicando el estado en que una parte del rostro se encuentra, por ejemplo, que tan abiertos estan los ojos, si estamos levantando las cejas, frunciendo el ceño, etc. Aprovechando esta nueva información, se reentrenó el modelo ahora utilizando estas 52 características únicamente, mejorando notablemente el desempeño. La principal ventaja que se tuvo al utilizar este nuevo vector de características sobre el vector de 1434 fue que los valores ahora se encontraban normalizados de 0 a 1, la posición de la cara ya no interfería con los valores y se reducía el tamaño del dataset de manera considerable.

Para agilizar el proceso de selección y configuración del algoritmo a utilizar, se hizo uso de la herramienta Orange. Aquí se pueden configurar múltiples algoritmos y visualizar las afectaciones al desempeño de manera casi instantanea. Como se mencionó anteriormente los algoritmos que se tenían contemplados fueron SVM, NaiveBayes y KNN. Los tres algoritmos de clasificación presentan un desempeño similar con el dataset generado y bajo las mismas métricas de evaluación, pero el algoritmo SVM resulta ser más amigable para su uso en dispositivos de poca capacidad de procesamiento y se acopla mejor para la idea general de la plataforma del proyecto.

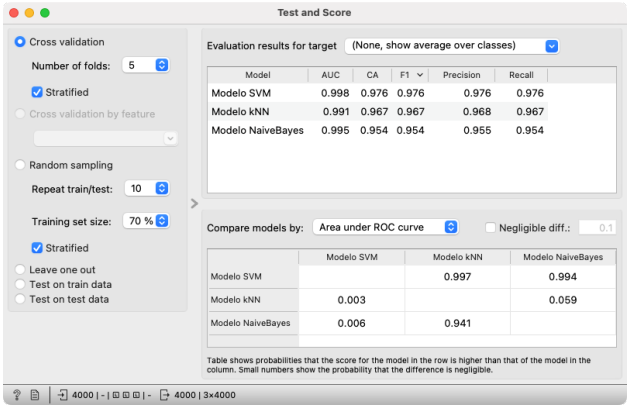

Fig 5. Comparación de modelos de clasificación. Elaboración propia

La configuración utilizada para el algoritmo SVM fue una clasificación multiclase ya que la naturaleza del SVM es de clasificación binaria. Esta clasificación multiclase se logra mediante el encadenamiento de clasificaciones binarias. El kernel es gaussiano (RBF) y los valores de Gamma y C fueron asignados a 0.01 y 0.10 respectivamente.

3) *Recomendación.* Esta etapa es la útlima del flujo de información en la aplicación, aquí se generan las recomendaciones musicales al usuario.

Para lograr la integración con el servicio de streaming Spotify fue necesario el registro de la aplicación para la obtención de un token de acceso para utilizarse en cada una de las peticiones solicitadas el servicio REST API.

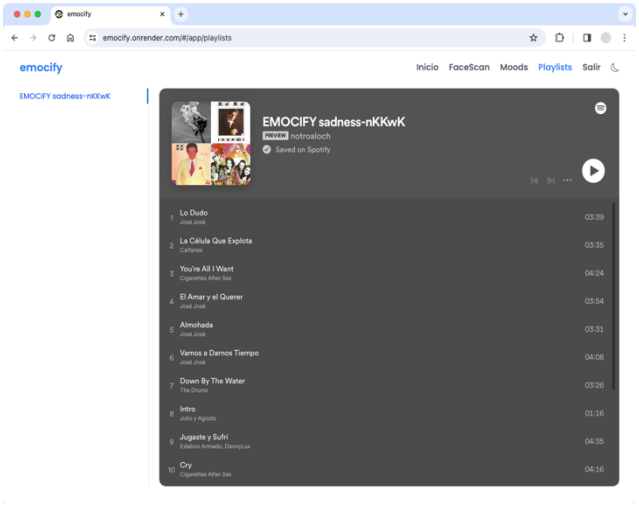

Fig 6. Vista de playlist generada para una emoción de tristeza. Elaboración propia

El diseño del backend se basa en una arquitectura REST API, en la cual se tienen endpoinds para realizar las acciones necesarias como el registro de usuarios, clasificación de expresiones y generación de playlist.

Estos endpoints son privados y accesibles únicamente dentro de la misma plataforma Emocify, es decir, no se ofrecen como servicios abiertos que se puedan utilizar para generar playlist o clasificar emociones.

El servicio de recomendación esta fuertemente ligado a la disponibilidad del servicio de clasificación, ya que para generar una lista de reproducción es necesario tener una emoción ya clasificada. Esto tiene como desventaja que la aplicación completa puede verse afectada con la simple caida de un servicio de la red. No se cuenta con servicios de soporte ante este tipo de fallos.

Otra de las medidas de tolerancia a posibles errores es el mecanismo de recomendación cuando no se tiene un suficiente historial de reproducción en la cuenta del usuario. Para la generación de recomendaciones se necesita por lo menos un artista o una cancion en el historial. Si este requisito no se cumple, se activará un mecanismo en el cual se recomienda al usuario playlist curadas por el servicio de Spotify, dejando completamente de lado la personalización basada en el usuario. Esto se hace principalmente para los casos en los que un usuario desea utilizar la plataforma pero no es suscriptor del servicio de streaming Spotify. La principal solución sería crear un perfil nuevo y ahora si podría ingresar a Emocify, sin embargo, los perfiles recien creados no cuentan con historial de reproducción. Si bien esto no ofrece una playlist personalizada generada por la plataforma, garantiza el correcto funcionamiento ante este tipo de exepciones.

El inicio de sesión en Emocify es mediante un tercero, es necesaria una cuenta de Spotify para inciar sesión, no importa si es de tipo premium o gratis. Esto se logra utilizando el medio de autenticación OAuth 2.0. Esto implico las adecuaciones necesarias en el backend para la protección de tokens y credenciales del usuario, así como la redirección a la plataforma de Spotify para la autenticación del usuario.

Una vez vinculada la cuenta, se accede a los datos del usuario anteriormente descritos en III.C.1. y son almacenados en la base de datos para la persistencia de la cuenta.

Spotify cuenta con los endpoints necesarios para consultar los hábitos de escucha del usuario, podemos consultar sus canciones más reproducidas, los artistas que sigue, las playlist que tiene, etc. Esta información nos servirá para generar las recomendaciones basadas en su estado de ánimo.

Cuando la etapa de FaceScan es completada y se tiene una emoción clasificada del servicio que hostea el modelo de clasificación, se inicia el proceso de generar la lista de reproducción. En este proceso primeramente se compila una extensa biblioteca con las mejores canciones de los artistas escuchados/seguidos por el usuario. Teniendo este conjunto de canciones, se filtran en base a metadatos proporcionados por Spotify para tratar de adecuar las canciones al estado de ánimo.

Los valores con los que se decide si una canción pertenece a un estado de ánimo o no son los metadatos del análisis de audio de la cancion proporcionados: valence, energy y danceability. Estos metadatos son valores con rango 0 a 1 y describen como suena una cancion. Tomando la documentación del servicio de Spotify, podemos describir cada uno de los metadatos como características sonoras que se basan enteramente en el análisis sonoro, detallando cada uno de las características utilizadas en los parrafos siguientes.

Valence es una medida que describe la positividad musical, canciones con valores con tendencia a 1 suenan más positivas (felices, euforicas), mientras que las canciones con valores con tendencia a 0 suenan mas negativas (tristes, depresivas, enojadas). Este valor es uno de los metadatos clave utilizados en el algoritmo de clasificación de recomendaciones.

Energy representa la percepción de la intensidad y actividad de una canción, donde generalmente las canciones que se escuchan rápidas y ruidosas tienen valores de energy con tendencia a 1.

Danceability describe que tan bailable es una cancion basado en una combinación de elementos musicales como el tempo, la

estabilidad de ritmo y la fuerza del beat. Los valores inclinados a 0 son menos bailables a los inclinados a 1.

Utilizando estos valores como referencia, se desarrolló un algoritmo de clasificación basado en estos umbrales. La clasificación se basa enteramente a interpretación propia. Es importante mencionar que la clasificación realizada de esta manera no ofrece los mejores resultados, ya que únicamente se esta segmentando las canciones basadas en el análisis sonoro y no se contemplan otros factores externos como la letra y el género que pueden estar o no directamente relacionados con la emoción transmitida por dicha cancion.

TABLA I
VALORES DE CLASIFICACIÓN MUSICAL

Propiedad	**Emocion**			
	Enojo	**Tristeza**	**Neutral**	**Felicidad**
Valence	< 0.4	< 0.4	>= 0.4 <= 0.6	> 0.6
Energy	>= 0.5	<= 0.5		>= 0.5
Danceability		<= 0.5		>= 0.5

D. Modulo I. Gestión de la Tecnología de Información

Gestión del proyecto basada en la metodología en cascada con el uso de herramientas para el desarrollo de diagramas de Gantt, versionamiento de código en Git y diseño de mockups. Codificación de proyecto full-stack siguiendo estándares y buenas prácticas de desarrollo.

E. Modulo II. Sistemas Robustos, Paralelos y Distribuidos

Implementación de una arquitectura distribuida en servicios, base de datos orientada a documentos (NoSQL) utilizando el gestor Google Firebase. Hosting de servicios en distintas plataformas: aplicación web en servidor Node y modelo de clasificacion en servidor Flask. Despliegue de aplicación web en arquitectura cliente-servidor.

F. Módulo III. Computo Flexible (Softcomputing)

Integración de librerías y algoritmos de minería de datos para la clasificación de expresiones faciales en emociones. Generación de dataset propio y entrenamiento de modelo de aprendizaje supervisado utilizando el algoritmo de máquina de vectores de soporte (SVM) siguiendo la metodología CRISP-DM. Utilización de librerías de aprendizaje máquina como MediaPipe y SciKit Learn.

IV. RESULTADOS OBTENIDOS DEL PROYECTO

El desempeño del proyecto se basa en dos métricas de evaluación: la exactitud del modelo de clasificación de emociones y la calidad de recomendación de canciones. El primer modelo en la etapa de desarrollo mostró excelentes resultados con los datasets de entrenamiento, arrojando los siguentes valores de evaluación con un porcentaje de entrenamiento del 70% y un 30% de datos de prueba:

- AUC: 0.998
- CA, F1, PRECISION, RECALL: 0.976

Este desempeño no se vio reflejado en las pruebas realizadas en producción, ya que el modelo únicamente se entreno con los datos generados por un sujeto la exactitud del modelo bajó considerablemente al presentarse nuevos sujetos de prueba. La exactitud aproximada del modelo en producción es de 60%, siendo las emociones de enojo y neutral las instancias con mayores falsos positivos.

Fig 7. Matriz de confusión del modelo de clasificación. Elaboración propia

La clasificación y recomendación de canciones se basa en una medición subjetiva, ya que cada usuario decide si una canción refleja o no su estado de ánimo actual. Además, el filtro de canciones se basa meramente en los metadatos del análisis de audio y no se contemplan otros factores como la letra de la canción o la verdadera intención que el artista busca transmitir con esa canción en específico. Cabe destacar que las recomendaciones para el estado de ánimo tristeza son las más aceptadas por los usuarios de la plataforma, ya que son más faciles de identificar y los metadatos utilizados para su clasificación tienen un enfoque hacia esta emoción.

Como métrica de evaluación general se tomó la experiencia del usuario al utilizar la aplicación, la cual fue muy positiva. El uso de la cámara web como principal atractivo y el desarrollo de una interfaz llamativa fueron elementos clave para lograr el interés y la retención del usuario. Además, el dinamismo de poder generar playlist hechas a la medida de cada usuario agrega una experiencia aún mas enriquecedora.

V. CONCLUSIONES Y TRABAJO A FUTURO

El proyecto cumplió con el propósito establecido al principio del desarrollo: hacer recomendaciones musicales personalizadas por medio de un algoritmo que detecta expresiones faciales y las clasifica en emociones mediante una plataforma web intuitiva.

A largo plazo se podría considerar la inclusión de más emociones como el miedo, disgusto o sorpresa para así poder generar recomendaciones con un espectro más amplio. También se podría vincular otros servicios de streaming a la plataforma como Apple Music, con la finalidad de incrementar usuarios y la accesibilidad.

El entrenamiento del modelo de clasificación con un conjunto de datos más grande sin duda es un aspecto importante a tener en cuenta para actualizaciones futuras, ya que esto garantiza un desempeño mayor al tener una variedad de rostros y expresiones.

El desarrollo de distintos modelos de clasificación podría ser una buena adición en futuras actualizaciones del sistema, a manera de comparacion entre distintos algoritmos de clasificación y sus desempeños.

RECONOCIMIENTOS

A mis amigos Juan Diego Gómez y Sergio Peña Durán, quienes fueron de gran apoyo durante el desarrollo del proyecto con su lluvia de ideas.

A mi profesor David Gomez Anaya, con quién en su clase de seminario de algoritmia descubrí un gran interés por la resolución de problemáticas utilizando visión por computadora.

A mi profesor Luis Antonio Medellín Serna, por quién fue la principal motivación para el desarrollo de este proyecto modular. El siempre estar dispuesto a ofrecer ayuda y mostrar interés por ideas sobre proyectos fue por lo que decidí fuera mi asesor y hacer el desarrollo basado en la idea del reconocimiento de expresiones faciales.

REFERENCIAS

[1] Sachs, M. E., Damasio, A., & Habibi, A. (2015). The pleasures of sad music: a systematic review. *Frontiers in human neuroscience*, *9*, 404. https://doi.org/10.3389/fnhum.2015.00404

[2] (2023) Spotify Design Guidelines. [Online]. Available: https://developer.spotify.com/documentation/design

[3] Lucey, P., Cohn, J. F., Kanade, T., Saragih, J., Ambadar, Z., & Matthews, I. (2010). The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified expression. In 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops. 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops). IEEE. https://doi.org/10.1109/cvprw.2010.5543262

[4] Michael J Lyons, Miyuki Kamachi, & Jiro Gyoba. (2020). Coding Facial Expressions with Gabor Wavelets (IVC Special Issue). https://doi.org/10.5281/zenodo.4029680

[5] Mollahosseini, A., Hasani, B., & Mahoor, M. H. (2017). AffectNet: A Database for Facial Expression, Valence, and Arousal Computing in the Wild. arXiv. https://doi.org/10.48550/ARXIV.1708.03985

[6] Li, S., Deng, W., & Du, J. (2017). Reliable Crowdsourcing and Deep Locality-Preserving Learning for Expression Recognition in the Wild. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE. https://doi.org/10.1109/cvpr.2017.277

[7] Dumitru, Ian Goodfellow, Will Cukierski, Yoshua Bengio. (2013). Challenges in Representation Learning: Facial Expression Recognition Challenge. Kaggle. https://kaggle.com/competitions/challenges-in-representation-learning-facial-expression-recognition-challenge

[8] Ebner, N. C., Riediger, M., & Lindenberger, U. (2010). FACES—A database of facial expressions in young, middle-aged, and older women and men: Development and validation. In Behavior Research Methods (Vol. 42, Issue 1, pp. 351–362). Springer Science and Business Media LLC. https://doi.org/10.3758/brm.42.1.351

[9] Dagher, I., Dahdah, E., & Al Shakik, M. (2019). Facial expression recognition using three-stage support vector machines. In Visual Computing for Industry, Biomedicine, and Art (Vol. 2, Issue 1). Springer Science and Business Media LLC. https://doi.org/10.1186/s42492-019-0034-5

[10] Murugappan, M., Mutawa, A. M., Sruthi, S., Hassouneh, A., Abdulsalam, A., S, J., & R, R. (2020). Facial Expression Classification using KNN and Decision Tree Classifiers. In 2020 4th International Conference on Computer, Communication and Signal Processing (ICCCSP). 2020 4th International Conference on Computer, Communication and Signal Processing (ICCCSP). IEEE. https://doi.org/10.1109/icccsp49186.2020.9315234